МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт з дисципліни
«Обчислювальна техніка та програмування за фахом»
для студентів спеціальності 141
«Електроенергетика, електротехніка та електромеханіка»
(освітня програма «Електричні машини і апарати»)
усіх форм навчання

2023

Методичні вказівки до лабораторних робіт з дисципліни «Обчислювальна техніка та програмування за фахом» для студентів спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» (освітня програма «Електричні машини і апарати») усіх форм навчання. / Укл.: Т.П. Солодовнікова, С. О. Лапкіна. – Запоріжжя : НУ «Запорізька політехніка», 2023. – 47 с.

Укладачі: Т.П. Солодовнікова, старш. викл.
С.О. Лапкіна, асист.

Рецензент Т. Є. Дівчук, доцент

Відповідальний за випуск С.О. Лапкіна, асист.

Затверджено
на засіданні кафедри
«Електричні машини»
Протокол №1
від 14.08.2023 р.

Рекомендовано до видання
НМК Електротехнічного
факультету
Протокол № 1
від 21.09.2023 р.

ЗМІСТ

ВСТУП

Однією з основних областей застосування персональних комп'ютерів є складні обчислювальні задачі, що виникають при моделюванні технічних пристроїв і процесів, можна розбити на ряд елементарних: обчислення інтегралів, рішення рівнянь, рішення диференціальних рівнянь і т.д. Для таких задач уже розроблені методи рішення, створені математичні системи, доступні для вивчення.

Мета виконання лабораторних робіт – навчити користуватися найпростішими методами обчислень із використанням сучасних інформаційних технологій.

У результаті вивчення навчальної дисципліни студент повинен отримати загальні компетентності: здатність до абстрактного мислення, аналізу і синтезу; здатність застосовувати знання у практичних ситуаціях; здатність до пошуку, оброблення та аналізу інформації з різних джерел; здатність виявляти, ставити та вирішувати проблеми; здатність працювати автономно;

Фаховими компетентностями є здатність вирішувати практичні задачі із застосуванням систем автоматизованого проектування і розрахунків (САПР); здатність розробляти проекти електроенергетичного, електротехнічного та електромеханічного устаткування із дотриманням вимог законодавства, стандартів і технічного завдання; усвідомлення необхідності підвищення ефективності електроенергетичного, електротехнічного та електромеханічного устаткування;

Очікувані програмні результати навчання: студент повинен вміти самостійно вчитися, опановувати нові знання і вдосконалювати навички роботи з сучасним обладнанням, вимірювальною технікою та прикладним програмним забезпеченням; обирати і застосовувати придатні методи для аналізу і синтезу електромеханічних та електроенергетичних систем із заданими показниками.

Кожна лабораторна робота містить стислий опис методів обчислень, приклади з необхідними коментарями, порядок виконання лабораторної роботи і варіанти індивідуальних завдань для групи студентів із 30 чоловік, контрольні запитання.

1 ЛАБОРАТОРНА РОБОТА. РІШЕННЯ РІВНЯНЬ

Мета: вивчити способи рішення рівнянь і систем рівнянь засобами SMath Studio.

1.1 Загальні відомості

Як відомо, багато рівнянь і системи рівнянь не мають аналітичних рішень. У першу чергу це відноситься до більшості трансцендентних рівнянь. Доведено також, що не можна побудувати формулу, по якій можна було б вирішити довільне алгебраїчне рівняння ступеня вище четвертого. Проте такі рівняння можуть вирішуватися чисельними методами з заданою точністю (не більш значення заданого системною змінною **TOL**).

1.1.1 Чисельне рішення нелінійного рівняння

Для найпростіших рівнянь виду $f(x) = 0$ рішення в SMat Studio знаходиться за допомогою функції *root* (Рисунок 1.1).

root(f(x1, x2, …), x1, a, b)

Повертає значення ***x1***, що належить відрізку **[*a*, *b*]**, при якому вираження або ***f*(*x*)** обертається в 0. Обидва аргументи цієї функції повинні бути скалярами. Функція повертає скаляр.

Аргументи: ***f*(*x1*, *x2*, …)** - функція, визначена або в робочому документі, або вираження. Вираження повинно повертати скалярні значення; ***x1*** - ім'я змінної, що використовується у вираженні. Цієї змінної перед використанням функції *root* необхідно привласнити числове значення. SMat Studio використовує його як початкове наближення при пошуку кореня; ***a*, *b*** - необов'язкові, якщо використовуються, то повинні бути речовинними числами, причому ***a* < *b*.**

Наближені значення коренів (*початкові наближення*) можуть бути:

- відомі з фізичного змісту задачі;
- відомі з рішення аналогічної задачі при інших вихідних даних;
- знайдені графічним способом.

Найбільше поширений *графічний спосіб* визначення початкових наближень. Приймаючи в увагу, що дійсні корені рівняння $f(x) = 0$ - це точки перетинання графіка функції $f(x)$ із віссю абсцис, достатньо побудувати графік функції $f(x)$ і відзначити точки перетинання $f(x)$ із віссю Ox, або відзначити на осі Ox відрізки, що містять по одному кореню. Побудова графіків часто вдасться сильно спростити, замінивши рівняння $f(x) = 0$ *подібним* йому рівнянням:

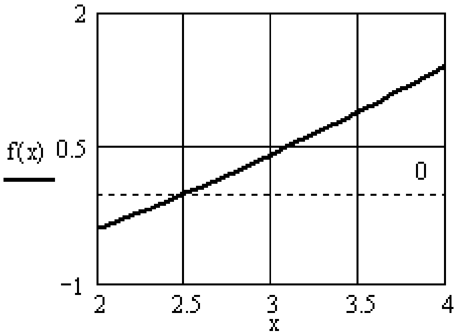
$$f_1(x)=f_2(x),$$

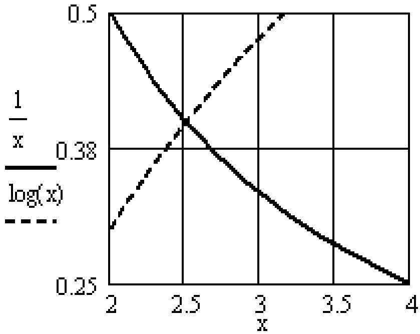
де функції $f_1(x)$ і $f_2(x)$ - більш прості, чим функція $f(x)$. Тоді, побудувавши графіки функцій $y = f_1(x)$ і $y = f_2(x)$, шукані корені одержимо як абсциси точок перетинання цих графіків.

Рішення рівняння $x \cdot \lg(x) = 1$ за допомогою функції *root*

$f(x) := x \cdot \log(x) - 1 \qquad x := 2, 2.1 .. 4 \qquad TOL := 1 \cdot 10^{-4}$

Графічне рішення рівняння $f(x)$, за графіком виявлене початкове наближення $x = 2.5$





1 спосіб

$x := 2.5$ - початкове наближення

$\mathrm{root}(f(x), x) = 2.5062$

2 спосіб

$x := 2.5$ - початкове наближення

$\mathrm{root}(x \cdot \log(x) - 1, x) = 2.5062$

3 спосіб

$\mathrm{root}(x \cdot \log(x) - 1, x, 2, 4) = 2.5062$

У способах 1 і 2 початкове наближення показує функції *root*, де шукати корінь. У способі 3 - параметри визначають область, де шукати корінь.
У способі 1 перший аргумент - це функція $f(x)$, визначена в документі. У способах 2 і 3 - це вираження.

Рисунок 1.1 – Рішення рівнянь засобами SMat Studio

Якщо після багатьох ітерацій SMat Studio не знаходить підхожого наближення, то з'явиться повідомлення Can't converge to a solution. (відсутня збіжність).

1.1.2 Знаходження коренів поліному

Для знаходження коренів вираження, що має вид

$$v_n x^n + ... + v_2 x^2 + v_1 x + v_0,$$

краще використовувати функцію *polyroots*, ніж *root*. На відміну від функції *root*, функція *polyroots* не потребує початкового наближення і повертає відразу всі корені, як речовинні, так і комплексні.

Polyroots(*v*)

Повертають корені поліному ступеня *n*. Коефіцієнти поліному знаходяться у векторі ***v*** довжини *n* + 1. Повертає вектор довжини *n*, що складає з коренів поліному. ***Аргументи:* v -** вектор, що містить коефіцієнти поліному

Вектор ***v*** зручно створювати використовую команду **Simbolics ⇒ Polynomial Coefficients**. Рисунок 1.2 ілюструє визначення коренів поліному засобами SMat Studio.

1.1.3 Рішення систем рівнянь

SMat Studio дає можливість вирішувати також і системи рівнянь. Максимальне число рівнянь і змінних дорівнює 50. Результатом рішення системи буде чисельне значення шуканого кореня.

Для рішення системи рівнянь необхідно виконати наступне:

- задати початкове наближення для всіх невідомих, що входять у систему рівнянь. SMat Studio вирішує систему за допомогою ітераційних методів;

- набрати ключове слово *Given*. Воно вказує SMat Studio, що далі слідує система рівнянь;

- введіть рівняння і нерівності в будь-якому порядку. Використовуйте **[Ctrl]=** для печатки символу **=**. Між лівими і правими частинами нерівностей може стояти будь який із символів <, >, ≥ і ≥;

- введіть будь-яке вираження, що включає функцію *Find*, наприклад: *a*:= *Find*(*x, y*).

***Find*(*z*1, *z*2, . . .)**

Повертає точне рішення системи рівнянь. Число аргументів повинно бути рівно числу невідомих.

Ключове слово *Given*, рівняння і нерівності, що слідують за ним, і будь яке вираження, що містить функцію *Find***,** називають ***блоком рішення рівнянь***.

Знаходження коренів поліному $\quad 0.75x^3 - 8x + 5$

$$v := \begin{pmatrix} 5 \\ -8 \\ 0 \\ .75 \end{pmatrix}$$

$$r := \mathrm{polyroots}(v)$$

$$r = \begin{pmatrix} -3.542 \\ 0.651 \\ 2.892 \end{pmatrix}$$

Для створення вектора v:

1 Встановіть курсор на змінну *x* у виразі $0.75x^3$-8x+5

2 Виберіть команду **Simbolics ⇒ Polynomial Coefficients**.

3 Виберіть команду **Edit⇒Cut**.

4 Наберіть v:= та виберіть команду **Edit ⇒Paste**.

Побудова графіка функції

$$f(x) := 0.75x^3 - 8x + 5 \qquad x := -4, -3.9 .. 4 \qquad j := 0 .. 2$$

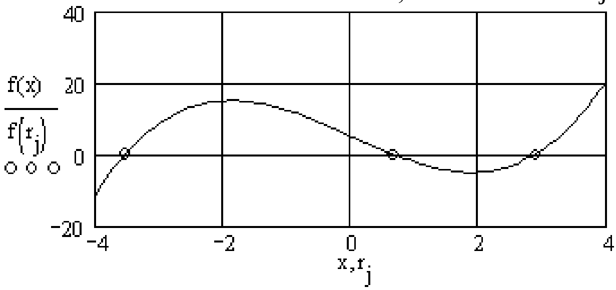


Рисунок 1.2 – Визначення коренів поліному

Наступні вираження неприпустимі усередині блока рішення:

- обмеження зі знаком ≠;
- дискретний аргумент або вираження, що містять дискретний аргумент у будь-якій формі;
- нерівності виду $a < b < c$.

Блоки рішення рівнянь не можуть бути вкладені друг у друга, кожний блок може мати тільки одне ключове слово *Given* і ім'я функції *Find*.

Функція, що завершує блок рішення рівнянь, може бути використана аналогічно будь-якій з інших функцій.

1 Можна вивести знайдене рішення:

$$Find(var1, var2, \ldots) =.$$

2 Визначити змінну за допомогою функції *Find*:

$$a := Find(x) \text{ – скаляр,}$$

$$var := Find(var1, var2,\ldots) \text{ – вектор.}$$

Це зручно зробити, якщо потрібно використовувати рішення системи рівнянь в іншому місці робочого документа.

3 Визначити іншу функцію за допомогою *Find*

$$f(a, b, c, \ldots) := Find(x, y, z, \ldots).$$

Ця конструкція зручна для багатократного рішення системи рівнянь для різноманітних значень деяких параметрів *a, b, c,…,* безпосередньо вхідних у систему рівнянь.

Рисунок 1.3 ілюструє рішення системи рівнянь у SMat Studio.

x1 := 0   x2 := 0   x3 := 0   - початкові наближення

Given

100 x1 + 6 x2 − 2 x3 = 100   - використовуйте **[Ctrl]=** для печатки символу **=**

6 x1 + 200 x2 − 10 x3 = 600

x1 + 2 x2 + 100 x3 = 500

$$\mathrm{Find}(x1, x2, x3) = \begin{pmatrix} 0.905 \\ 3.219 \\ 4.927 \end{pmatrix}$$

Рисунок 1.3 – Рішення системи рівнянь за допомогою функції *Find*

1.1.4 Рішення матричних рівнянь

Матричним рівнянням називається рівняння, коефіцієнти і невідомі якого - прямокутні матриці відповідної розмірності

Розглянемо систему *n* лінійних алгебраїчних рівнянь щодо *n* невідомих $x_1, x_2, \ldots, x_n$:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n. \end{cases} \tag{1.1}$$

Відповідно до правила множення матриць розглянута система лінійних рівнянь може бути записана в матричному виді

$$Ax = b \tag{1.2}$$

де:

$$A=\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix},\ x=\begin{bmatrix} x_1 \\ x_2 \\ ... \\ x_n \end{bmatrix},\ b=\begin{bmatrix} b_1 \\ b_2 \\ ... \\ b_n \end{bmatrix} \qquad (1.3)$$

Матриця A, стовпчиками якої є коефіцієнти при відповідних невідомих, а рядками - коефіцієнти при невідомих у відповідному рівнянні, називається *матрицею системи*; матриця-стовпчик b, елементами якої є праві частини рівнянь системи, називається *матрицею правої частини*. Матриця-стовпчик x, елементи якої - шукані невідомі, називається *рішенням системи*.

Якщо матриця A - не особлива, тобто $\det A \neq 0$, то система (1.1), або еквівалентне їй матричне рівняння (1.2), має єдине рішення.

Справді, за умови $\det A \neq 0$, існує обернена матриця A^{-1}. Примножуючи обидві частини рівняння (1.2) на матрицю A^{-1} одержимо:

$$A^{-1}Ax = A^{-1}b,$$
$$x = A^{-1}b. \qquad (1.4)$$

Формула (1.4) дає рішення рівняння (1.2) і воно єдине. Системи лінійних рівнянь зручно вирішувати за допомогою функції *lsolve*.

lsolve(A, b)

Повертається вектор рішення x такий, що $Ax = b$.

Аргументи: ***A*** - квадратна, не сингулярна матриця; ***b*** - вектор, що має стільки ж рядів, скільки рядів у ***матриці*** А.

На рис. 1.4 показане рішення системи трьох лінійних рівнянь щодо трьох невідомих.

1.1.5 Наближені рішення

Функція *Minerr* дуже схожа на функцію *Find* (використовує той же алгоритм). Якщо в результаті пошуку не може бути отримане подальше уточнення поточного наближення до рішення, *Minerr* повертає це наближення. Функція *Find* у цьому випадку повертає повідомлення про помилку. Правила використання функції *Minerr* такі ж, як і функції *Find*.

Minerr(z1, z2, . . .)

Повертає наближене рішення системи рівнянь. Число аргументів повинно бути рівно числу невідомих.

Якщо *Minerr* використовується в блоці рішення рівнянь, необхідно завжди включати додаткову перевірку достовірності результатів.

Матриця системи; Матриця правої частини

$$A := \begin{pmatrix} 3 & -1 & 0 \\ -2 & 1 & 1 \\ 2 & -1 & 4 \end{pmatrix} \qquad b := \begin{pmatrix} 5 \\ 0 \\ 15 \end{pmatrix}$$

Обчислення визначника $|A| = 5$

Визначник відмінний від нуля, система має єдине рішення.

Рішення системи *Ax*=*b*

$$x := A^{-1} \cdot b \qquad x = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$$

Примітка: Зразки матриці і вектора відповідають лінійній системі

$$\begin{aligned} 3 \cdot x_1 - x_2 &= 5 \\ -2 \cdot x_1 + x_2 + x_3 &= 0 \\ 2 \cdot x_1 - x_2 + 4 \cdot x_3 &= 15 \end{aligned}$$

Рішення системи за допомогою функції *lsolve*:

$$x := \mathrm{lsolve}(A, b) \qquad x = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$$

Перевірка правильності рішення

$$A \cdot x - b = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Рисунок 1.4 – Рішення матричних рівнянь

1.2 Порядок виконання лабораторної роботи

1.2.1 Побудувати графік функції *f*(*x*) (таблиця 1.1) і приблизно визначити один із коренів рівняння. Вирішити рівняння *f*(*x*)=0 із точністю $\varepsilon = 10^{-5}$ за допомогою убудованої функції SMat Studio *root*.

Таблиця 1.1 – Варіанти до завдання 1.2.1

№	*f(x)*	*x*∈[*a*;*b*]
1	$e^{x-1}-x^3-x$	[0;1]
2	x-1/(3+sin(3.6x))	[0;1]
3	$arcos(x)-x^3$	[0;1)
4	$2x^2$-arcsin(x)	(0;1)
5	$3x-14+e^x-e^{-x}$	[1;3]
6	$2x^2$+1.2-cos(x)-2	[0;1]
7	cos(2/x)-2sin(1/x)+1/x	[1;2]
8	$0.1x^2-x \cdot ln(x)$	[1;2]
9	$0.25x^3+x-2$	[0;2]
10	$arcos((1-x^2)/(1+x^2))-x$	[2;3]
11	$3x-4 \cdot ln(x)-5$	[2;4]
12	$e^x-e^{-x}-2$	[0;1]
13	1-x-tg(x)	[0;1]
14	1-x+sin(x)-*ln*(1+x)	[0;2]
15	x^5–x-1	[1;2]
16	3sin(2x)-cos(x)	[1;2]
17	x^4-x^2-2	[0;2]
18	$cos(x^2)-7x$	[0;1]
19	3sin(x)+x-1	[0;2]
20	$10+66x-3x^3$	[2;5]
21	$3xe^{-x}-1$	[0;1]
22	2sin(6x)	(0;1)
23	$(e^x-1.5)/(x^2+1)$	[0;1]
24	$e^x/(e^x+1)-x$	[0;2]
25	cos(3x/2)+sin(2x)	[0;2]
26	*ln*(x)-2x+4	[2;4]
27	$0.3x^4$-cos(3x)	[0;2]
28	2x-3cos(4x)	[0;1]
29	x^3-cos(5x)	[0;1]
30	cos(2x)-x	[0;2]

1.2.2 Для поліному $g(x)$ (таблиця 1.2) виконати наступні дії: за допомогою команди **Simbolics ⇒ Polynomial Coefficients** створити вектор V, що містить коефіцієнти поліному; вирішити рівняння $g(x)=0$ за допомогою *функції* polyroots; вирішити рівняння символьно, використовуючи команду **Simbolics ⇒Variable ⇒Solve**.

Таблиця 1.2 – Варіанти до завдання 1.2.2

№	$\boldsymbol{g(x)}$	**№**	$\boldsymbol{g(x)}$
1	$x^4 - 2x^3 + x^2 - 12x + 20$	**16**	$x^4 + 2x^3 + 13x^2 + x + 20$
2	$x^4 + 6x^3 + x^2 - 4x - 60$	**17**	$x^4 + x^3 + 6x^2 + 17x + 25$
3	$x^4 - 14x^2 - 40x - 75$	**18**	$x^4 - 4x^3 + 8x^2 - 2x + 50$
4	$x^4 - x^3 + x^2 - 11x + 10$	**19**	$x^4 - 6x^3 + 5x^2 - 2x + 70$
5	$x^4 - x^3 - 29x^2 - 71x - 140$	**20**	$x^4 + 6x^3 + 20x^2 - 54x + 40$
6	$x^4 + 7x^3 + 9x^2 + 13x - 30$	**21**	$x^4 - 4x^2 - 10x - 5$
7	$x^4 + 3x^3 - 23x^2 - 55x - 150$	**22**	$x^4 - 8x^3 + 30x^2 - 10x + 70$
8	$x^4 - 6x^3 + 4x^2 + 10x + 75$	**23**	$x^4 - 2x^3 - 59x^2 - 7x - 25$
9	$x^4 + x^3 - 17x^2 - 45x - 100$	**24**	$x^4 + 14x^3 + 2x^2 + 3x - 70$
10	$x^4 - 5x^3 + x^2 - 15x + 50$	**25**	$x^4 + 9x^3 - 33x^2 - 5x – 45$
11	$x^4 - 4x^3 - 2x^2 - 20x + 25$	**26**	$x^4 - 7x^3 + x^2 + 18x + 175$
12	$x^4 + 5x^3 + 7x^2 + 7x - 20$	**27**	$x^4 + x^3 - 17x^2 - 45x - 100$
13	$x^4 - 7x^3 + 7x^2 - 5x + 100$	**28**	$x^4 - 15x^3 + 44x^2 - 35x + 40$
14	$x^4 + 10x^3 + 36x^2 + 70x + 75$	**29**	$x^4 - 3x^3 - 12x^2 - x + 125$
15	$x^4 + 9x^3 + 31x^2 + 59x + 60$	**30**	$x^4 + 8x^3 + 27x^2 + 13x - 120$

1.2.3 Вирішити систему лінійних рівнянь (таблиця 1.3): використовуючи функцію *Find*; матричним засобом і використовуючи функцію *lsolve*.

Таблиця 1.3 – Варіанти до завдання 1.2.3

№	Система лінійних рівнянь	**№**	Система лінійних рівнянь
1	$4x_1 + 20x_2 + x_3 - 24 = 0$ $16x_1 + 2x_2 - 2x_4 + 13 = 0$ $-4x_1 + 4x_3 + 32x_4 = 0$ $2x_1 + 10x_3 - 7 = 0$	**3**	$4x_1 - 5x_2 + 40x_3 - 19 = 0$ $10x_1 - 4x_2 + 50x_4 = 0$ $32x_1 + 4x_3 - 4x_4 - 34 = 0$ $32x_2 - 9x_4 + 49 = 0$
2	$3x_1 + 12x_2 - x_3 - 18 = 0$ $-5x_1 + 2x_2 + 32x_4 + 15 = 0$ $2x_1 + 16x_3 - 3x_4 = 0$ $12x_1 + 3x_2 - 21 = 0$	**4**	$4x_1 + 2x_2 + 32x_3 + 19 = 0$ $2x_1 + 30x_2 - 4x_4 - 39 = 0$ $36x_1 + 4x_3 - 5x_4 - 40 = 0$ $11x_3 + 40x_4 - 31 = 0$

Продовження таблиці 1.3

№	Система лінійних рівнянь	№	Система лінійних рівнянь
5	$2x_1 + 16x_2 - 3x_3 - 9 = 0$ $-8x_1 + x_2 + x_3 + x_4 - 98 = 0$ $25x_1 - 2x_3 - 7x_4 - 5 = 0$ $-3x_2 + 20x_3 + 7 = 0$	**13**	$9x_1 + 40x_2 + 2x_3 - 81 = 0$ $12x_1 - 4x_2 + 96x_4 - 119 = 0$ $-4x_1 + 64x_3 + 8x_4 + 15 = 0$ $36x_1 + 9x_4 - 7 = 0$
6	$5x_1 - 2x_2 + x_3 + x_4 - 27 = 0$ $4x_1 + 25x_2 - 3x_4 - 34 = 0$ $20x_1 + 2x_3 - 7x_4 + 28 = 0$ $-9x_3 + 40x_4 - 5 = 0$	**14**	$7x_1 - 5x_2 + 64x_3 - 18 = 0$ $9x_1 + 50x_2 - 4x_4 = 0$ $9x_2 - 7x_3 + 80x_4 - 128 = 0$ $40x_1 + 11x_2 + 19 = 0$
7	$-7x_1 + 2x_2 + 40x_3 - 21 = 0$ $9x_1 - 5x_2 + 50x_4 + 14 = 0$ $25x_1 + 4x_3 - x_4 - 13 = 0$ $32x_2 + 9x_4 - 21 = 0$	**15**	$11x_1 + 64x_2 - 2x_3 + 34 = 0$ $50x_1 + 3x_2 - 12x_4 = 0$ $13x_2 - 9x_3 + 100x_4 - 131 = 0$ $17x_1 + 80x_3 - 85 = 0$
8	$8x_1 + 40x_2 - 3x_3 - 28 = 0$ $-7x_1 + 5x_2 + 50x_4 = 0$ $8x_1 + 64x_3 - 11x_4 - 18 = 0$ $32x_1 + 5x_4 - 12 = 0$	**16**	$15x_1 + 80x_2 - 4x_3 - 93 = 0$ $64x_1 + 7x_2 - 5x_4 - 131 = 0$ $11x_2 - 8x_3 + 128x_4 + 34 = 0$ $37x_2 + 100x_3 - 125 = 0$
9	$-9x_1 + 4x_2 + 64x_3 - 24 = 0$ $10x_1 + 50x_2 - 4x_4 + 5 = 0$ $-14x_2 + 7x_3 + 80x_4 - 14 = 0$ $40x_1 + 9x_2 - 29 = 0$	**17**	$17x_1 + 100x_2 - 9x_3 = 0$ $80x_1 - 7x_2 - 5x_4 + 79 = 0$ $21x_2 + 128x_3 - 4x_4 - 139 = 0$ $19x_3 + 256x_4 + 54 = 0$
10	$-8x_1 + 64x_2 + 5x_3 - 37 = 0$ $50x_1 - 13x_2 + 2x_4 - 38 = 0$ $17x_2 - 9x_3 + 100x_4 = 0$ $-11x_1 + 80x_3 - 115 = 0$	**18**	$4x_1 - x_2 + 20x_3 - 38 = 0$ $18x_1 + 3x_2 - 2x_4 + 14 = 0$ $10x_2 + x_3 - x_4 - 15 = 0$ $4x_2 + 20x_4 - 29 = 0$
11	$-13x_1 + 80x_2 + 2x_3 - 64 = 0$ $64x_1 + 9x_2 - 5x_4 - 29 = 0$ $12x_2 - 9x_3 + 128x_4 = 0$ $27x_2 + 100x_3 - 231 = 0$	**19**	$3x_1 + 20x_2 - 2x_3 - 41 = 0$ $5x_1 - 4x_2 + 20x_4 + 19 = 0$ $5x_2 + 32x_3 - 3x_4 - 34 = 0$ $12x_1 + 3x_4 - 29 = 0$
12	$-13x_1 + 100x_2 + 9x_3 + 128 = 0$ $80x_1 + 10x_2 - 5x_4 - 34 = 0$ $-14x_2 + 128x_3 + 7x_4 - 95 = 0$ $31x_3 + 256x_4 + 69 = 0$	**20**	$4x_1 + 25x_2 - x_3 - 17 = 0$ $6x_1 + 5x_2 + 40x_4 = 0$ $25x_1 + 3x_3 + 4x_4 + 34 = 0$ $-5x_2 + 30x_3 - 9 = 0$

Продовження таблиці 1.3

№	Система лінійних рівнянь	№	Система лінійних рівнянь
21	$x_1 - 2x_2 + 16x_3 - 31 = 0$ $10x_1 - x_2 + x_4 = 0$ $12x_2 + x_3 - x_4 + 28 = 0$ $2x_2 + 16x_4 - 29 = 0$	**26**	$9x_1 - 2x_2 + 36x_3 - 19 = 0$ $4x_1 + 25x_2 - 3x_4 + 18 = 0$ $40x_1 + 5x_3 - 4x_4 - 44 = 0$ $11x_3 + 40x_4 - 21 = 0$
22	$2x_1 + 20x_2 - 3x_3 - 39 = 0$ $4x_1 - 2x_2 + 24x_4 = 0$ $2x_2 + 16x_3 - x_4 + 25 = 0$ $12x_1 + 3x_4 - 18 = 0$	**27**	$9x_1 - 2x_2 + 40x_3 - 78 = 0$ $11x_1 - 3x_2 + 50x_4 + 114 = 0$ $30x_1 - 4x_3 + 5x_4 + 21 = 0$ $32x_2 + 8x_4 - 40 = 0$
23	$2x_1 + 16x_2 - x_3 - 32 = 0$ $3x_1 - 8x_2 + 60x_4 + 64 = 0$ $4x_1 + 24x_3 - 3x_4 = 0$ $12x_1 + 3x_2 - 45 = 0$	**28**	$2x_1 + 40x_2 + 5x_3 - 42 = 0$ $4x_1 - 9x_2 + 72x_4 - 88 = 0$ $4x_1 + 64x_3 + 8x_4 - 119 = 0$ $36x_1 + 9x_4 - 54 = 0$
24	$5x_1 - 2x_2 + 40x_3 - 39 = 0$ $4x_1 + 32x_2 - 6x_4 = 0$ $7x_1 + 3x_3 + 32x_4 - 21 = 0$ $20x_1 + 4x_3 + 19 = 0$	**29**	$8x_1 - 3x_2 + 64x_3 - 131 = 0$ $-7x_1 + 50x_2 + 5x_4 + 84 = 0$ $12x_2 - 9x_3 + 80x_4 - 52 = 0$ $40x_1 + 9x_2 - 78 = 0$
25	$5x_1 + 30x_2 - 3x_3 - 17 = 0$ $-8x_1 + 5x_2 + 40x_4 - 31 = 0$ $24x_1 + 3x_3 - 4x_4 - 39 = 0$ $7x_2 + 25x_3 - 8 = 0$	**30**	$7x_1 + 64x_2 - 2x_3 - 111 = 0$ $50x_1 + 5x_2 - 8x_4 - 98 = 0$ $18x_2 + 5x_3 + 112x_4 -219 = 0$ $15x_1 + 80x_3 + 31 = 0$

1.2.4 Перетворити нелінійні рівняння системи з таблиці 1.2.4 до виду $f_1(x) = y$ і $f_2(y) = x$. Побудувати їхні графіки і визначити початкове наближення рішення. Вирішити систему нелінійних рівнянь за допомогою функції *Minerr*.

Таблиця 1.4 – Варіанти до завдання 1.2.4

№	**Система нелінійних рівнянь**	№	**Система нелінійних рівнянь**
1	$\sin(x_1+x_2)-x_2-1.2=0$ $2x_1+\cos(x_2)-2=0$	**4**	$\sin(0.5x_1+x_2)-1.2x_1-1=0$ $(x_1)^2+(x_2)^2-1=0$
2	$\cos(x_1-1)+x_2-0.5=0$ $\sin(x_1)+2x_2-2=0$	**5**	$\tan(x_1x_2+0.3)-(x_1)^2=0$ $0.9(x_1)^2+2(x_2)^2-1=0$
3	$\sin(x_1)+2x_2-2=0$ $\cos(x_1)+x_2-1.5=0$	**6**	$\sin(x_1+x_2)-1.3x_1-1=0$ $(x_1)^2+0.2(x_2)^2-1=0$

Продовження таблиці 1.4

№	Система нелінійних рівнянь	№	Система нелінійних рівнянь
7	$\cos(x_1)+x_2-1.5=0$ $2x_1-\sin(x_2-0.5)-1=0$	**19**	$\tan(x_1x_2)-(x_1)^2=0$ $0.8(x_1)^2+2(x_2)^2-1=0$
8	$\sin(x_1+1.5)-x_2+2.9=0$ $\cos(x_2-2)+ x_1=0$	**20**	$\sin(x_1+ x_2)-1.5x_1-0.1=0$ $3(x_1)^2+(x_2)^2-1=0$
9	$\cos(x_1+0.5)+x_2-0.8=0$ $\sin(x_2)-2x_1-1.6=0$	**21**	$\tan(x_1x_2+0.2)-(x_1)^2=0$ $0.7(x_1)^2+2(x_2)^2-1=0$
10	$\sin(x_1-1)+x_2-0.1=0$ $x_1-\sin(x_2+1)-0.8=0$	**22**	$\sin(x_1+ x_2)-1.2x_1-0.1=0$ $(x_1)^2+(x_2)^2-1=0$
11	$\cos(x_1+x_2)+2x_2=0$ $x_1+\sin(x_2)-0.6=0$	**23**	$\tan(x_1x_2+0.2)-(x_1)^2=0$ $0.6(x_1)^2+2(x_2)^2-1=0$
12	$\cos(x_1+0.5)-x_2-2=0$ $\sin(x_2)+2x_1-1=0$	**24**	$\sin(x_1+ x_2)-x_1+0.1=0$ $x_2-\cos(3x_1)+0.1=0$
13	$\sin(x_1+x_2)-x_2-1.5=0$ $x_1+\cos(x_2-0.5)-0.5=0$	**25**	$\cos(x_2-2)+x_1=0$ $\sin(x_1+0.5)-x_2+2.9=0$
14	$\sin(x_2+1)+x_1-1.2=0$ $2(x_1)^2+x_2-2=0$	**26**	$\sin(x_1)+2x_2-2=0$ $\cos(x_2-1)+x_1-0.7=0$
15	$\cos(x_2-1)+x_1-0.5=0$ $x_2-\cos(x_1)-3=0$	**27**	$\sin(x_2)+x_1+0.4=0$ $2x_2-\cos(x_1+1)=0$
16	$\sin(x_1+x_2)-1.6x_1-1=0$ $(x_1)^2+(x_2)^2-1=0$	**28**	$\sin(x_1)-2x_2-1=0$ $\sin(x_2-1)+x_1-1.3=0$
17	$\tan(x_1x_2+0.4)-(x_1)^2=0$ $0.6(x_1)^2+2(x_2)^2-1=0$	**29**	$\sin(x_1+0.5)-x_2-1=0$ $\cos(x_2-2)+x_1=0$
18	$\cos(x_1+0.5)+x_2-1=0$ $\sin(x_2)-2x_1-2=0$	**30**	$\sin(x_1+2)-x_2-1.5=0$ $\cos(x_2-2)+x_1-0.5=0$

1.3 Контрольні запитання

1 Назвіть способи знаходження початкового наближення.

2 Які функції для рішення одного рівняння в SMat Studio ви знаєте? У чому їх відмінність?

3 Які аргументи функції root не обов'язкові?

4 У яких випадках SMat Studio не може знайти корінь рівняння?

5 Яка системна змінна відповідає за точність обчислень?

6 Як змінити точність, із яким функція root шукає корінь?

7 Як системна змінна TOL впливає на рішення рівняння за допомогою функції root?

8 Назвіть функції для рішення систем рівнянь у SMat Studio і особливості їхній застосування.

9 Опишіть структуру блока рішення рівнянь.

10 Який знак рівності використовується в блоці рішення? Якою комбінацією клавіш вводиться в документ?

11 Які вираження не припустимі усередині блока рішення рівняння?

12 Опишіть способи використання функції Find.

13 У яких випадках SMat Studio не може знайти рішення системи рівнянь?

14 Дайте порівняльну характеристику функціям Find і Minerr.

15 Які рівняння називаються матричними?

16 Як вирішувати матричні рівняння? Назвіть способи рішення матричних рівнянь.

1.4 Зміст звіту

1 Стислий конспект по теоретичному матеріалу.

2 Розрахунки в системі SMat Studio.

3 Висновки.

2 ЛАБОРАТОРНА РОБОТА. УБУДОВАНІ ФУНКЦІЇ. ЛОГІЧНІ ОПЕРАТОРИ

Мета: засвоїти призначення та особливості використання функцій, що служать для керування обчисленнями, і логічних операторів у SMat Studio.

2.1 Загальні відомості

2.1.1 Убудовані функції

SMat Studio має багатий набір убудованих функцій. Більшість із них просто повертає значення, проте є дві функції, що служать для керування обчисленнями.

Функція if(умова,оператор 1,оператор 2).

Якщо **умова** істинна, виконується **оператор1**, інакше **оператор2**.

умова:	$x = y$	Ctrl=	$x \neq y$	Ctrl 3
	$x > y$	>	$x \geq y$	Ctrl 0
	$x < y$	<	$x \leq y$	Ctrl 9

Результат логічної операції дорівнює 0, якщо умова не виконана, і 1, якщо умова істинна. Цією властивістю можна користуватися для створення більш складних логічних конструкцій, наприклад логічне множення:

$(x > -1)\cdot(x < 1)$ дорівнює 1, якщо виконуються обидві умови і 0 у противному випадку.

$(x < -1) + (x > 1)$ діє подібно логічному додаванню.

Використовуємо, наприклад, функцію ***if()*** для коректного визначення коренів квадратного рівняння:

$$a\cdot x^2 + b\cdot x + c = 0 \qquad D(a,b,c) := b^2 - 4\cdot a\cdot c$$

$$x(a,b,c) := if\left[D(a,b,c) < 0, \text{"дійсних коренів немає"}, \begin{pmatrix} \dfrac{-b+\sqrt{D(a,b,c)}}{2\cdot a} \\ \dfrac{-b-\sqrt{D(a,b,c)}}{2\cdot a} \end{pmatrix}\right]$$

$$x(1,2,-1) = \begin{pmatrix} 0.414 \\ -2.414 \end{pmatrix} \quad x(2,-1,-1) = \begin{pmatrix} 1 \\ -0.5 \end{pmatrix}$$

Проте: $x(1,2,2)$ = "дійсних коренів немає"

$x(-1,4,-5)$ = "дійсних коренів немає"

Використовуючи властивості логічних операцій можна вирішити таку задачу: нехай дано 1000 випадкових чисел на інтервалі від 0 до 1. Потрібно визначити скільки чисел буде більше 0.5.

$$i := 1 .. 1000 \qquad A_i := rnd(1) \qquad n := \sum_i (A_i > 0.5) \qquad n = 502$$

Функція ***rnd(n)*** повертає випадкове число з діапазону [0*;* *n*].

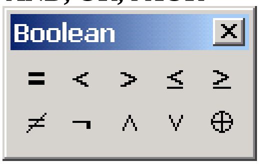
Дійсно, вираження в скобках дорівнює 1, якщо умова істинна, і 0 у противному випадку. Ми одержимо необхідне число.

Примітка. У SMat Studio припустимі двосторонні нерівності типу a< x < b .

Так, кількість чисел, що потрапили в діапазон [0.2; 0.5], буде

$$\sum_i (0.2 \le A_i \le 0.5) = 290$$

2.1.2 Використання логічних операторів NOT, AND, OR, і XOR



Знаки логічних операторів уводяться з палітри **Boolean**

Логічне заперечення (NOT – Ctrl+Shift+1):

$$P := \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad \overrightarrow{(\neg P)} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Логічне множення (AND – Ctrl+Shift+7):

$$P := \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \qquad Q := \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \overrightarrow{(P \wedge Q)} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Логічне додавання (OR – Ctrl+Shift+6):

$$P = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \qquad Q = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \overrightarrow{(P \vee Q)} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Логічний виняток OR (XOR – Ctrl+Shift+5):

$$P = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \qquad Q = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \overrightarrow{(P \oplus Q)} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

2.1.3 Робота з масивами даних

Масиви даних подані в SMat Studio у виді матриць. Створити матрицю можна використовуючи інтервальну змінну. Наприклад, необхідно створити масив із 100 випадкових цілих чисел із діапазону від -10 до 10.

$i := 0..99$ Діапазон із 100 значень.

$A_i := floor(rnd(20) - 10)$ Створення матриці з одного стовпчика і 100 рядків.

Обчислимо кількість позитивних, негативних і нульових значень.

$$\sum_i (A_i > 0) = 52 \qquad \sum_i (A_i < 0) = 45 \qquad \sum_i (A_i = 0) = 3$$

Обчислимо суму позитивних і негативних елементів масиву.

$$\sum_i (A_i > 0) \cdot A_i = 288 \qquad \sum_i (A_i < 0) \cdot A_i = -233$$

Створимо ще один масив із 100 значень заповнений 0 і 1 випадковою образом. $B_i := floor(rnd(2))$

Обчислимо тепер суму всіх позитивних чисел масиву *A* для який відповідні значення масиву *B* рівні 1.

$$\sum_i \left[(A_i > 0) \; and \; (B_i = 1) \right] \cdot A_i = 167$$

Робота з двомірною матрицею принципово нічим не відрізняється, тільки з'являється другий індекс - номер стовпчика, що потребує введення другої інтервальної змінної.

Створимо двомірний масив розміром 5∗5, у першому стовпчику розташуємо випадкове число від 0 до 1, у другому від 1 до 2 і т.д.

$$i := 0..4 \qquad j := 0..4 \qquad D_{i,j} := rnd(1) + j$$

Обчислимо тепер середнє значення в кожному стовпчику:

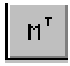
$$E_j := \frac{1}{5}\cdot\sum_i D_{i,j} \qquad D = \begin{pmatrix} 0.597 & 1.873 & 2.236 & 3.234 & 4.655 \\ 0.76 & 1.511 & 2.42 & 3.984 & 4.022 \\ 0.093 & 1.317 & 2.832 & 3.494 & 4.92 \\ 0.113 & 1.418 & 2.996 & 3.03 & 4.118 \\ 0.34 & 1.02 & 2.781 & 3.776 & 4.614 \end{pmatrix} \quad E = \begin{pmatrix} 0.381 \\ 1.428 \\ 2.653 \\ 3.504 \\ 4.466 \end{pmatrix}$$

Примітка. Є убудована функція, що обчисляє середнє значення ***mean()***. У якості аргументу використовується матриця

$$mean\left(D^{\langle 0\rangle}\right) = 0.381$$

Підрахуємо кількість значень у кожному стовпчику, що не перевершують середнє значення.

$$N_j := \sum_i \left(D_{i,j} \le E_j\right)$$

Виводимо **транспоновану матрицю** для економії екранного простору. 

$$N^T = (\,3 \quad 3 \quad 2 \quad 3 \quad 2\,)$$

Приведемо для прикладу досить корисну убудовану функцію під ім'ям *Delta(x,y)* **Дельта Кронекера** : *δ(m,n):=if(m=n,1,0)*.

Функція повертає 1, якщо аргументи збігаються, інакше 0.

δ(1,1)=1 δ(1,0)=1

За допомогою цієї функції нескладно створити одиничну матрицю, наприклад:

$n := 4 \quad i := 0..n$

$j := 0..n$

$a_{i,j} := \delta(i, j)$

$$a = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

або на побічній діагоналі: $b_{i,j} := \delta(n - i, j)$

$$b = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

2.2 Порядок виконання лабораторної роботи

2.2.1 Дана матриця розміром 3*3 із довільним набором значень. Створити матрицю також розміром 3*3, де встановити 1, якщо відповідне значення першої матриці позитивно, 0 - у противному випадку.

2.2.2 Дано набір із 500 випадкових чисел із діапазону від 0 до 8. Знайти, скільки чисел, що потраплять в інтервал від 3.2 до 4.7. Порівняти з тим, що повинно утворитися з пропорційності інтервалів.

2.2.3 Створити масив 50 цілих випадкових чисел у діапазоні від -2 до +4. ***Обчислити:***

- Кількість позитивних і негативних чисел.
- Суму всіх позитивних і негативних чисел.

2.2.4 Створити двомірний масив розміром 10*10 із цілих випадкових чисел у діапазоні від -10 до +10. ***Обчислити:***

- кількість позитивних, негативних і нульових значень у кожному рядку;
- суму всіх позитивних і негативних чисел у кожному рядку;
- суму і множення елементів головної і побічної діагоналі.

2.2.5 Використовуючи матрицю створену в попередній задачі:

- створити 2 матриці, що є дзеркальним відображенням щодо головної і побічної діагоналі;
- створити матрицю, установив відповідні елементи в 1, якщо елементи вихідної матриці є парними числами, і 0 у противному випадку.

2.2.6 Побудувати графік фікції (табл. 2.1), з використанням логічних операторів і функцій для заелементної обробки масивів.

Таблиця 2.1 - Варіанти до завдання 2.2.6

№	Функція	№	Функція
1	$y=\begin{cases}\dfrac{1+x^2}{\sqrt{1+x^4}}, x\le 0\\ 2x+\dfrac{\sin^2 x}{3+x}, x>0\end{cases}$	2	$y=\begin{cases}3\sin x-\cos^2 x, x\le 0\\ \dfrac{3\sqrt{1+x^2}}{\ln(x+5)}, x>0\end{cases}$

Продовження таблиці 2.1

№	Функція	№	Функція
3	$y=\begin{cases}\dfrac{3+\sin^2(2x)}{1+\cos^2 x}, x\le 0\\ 2x+\dfrac{\sin^2 x}{3+x}, x>0\end{cases}$	10	$y=\begin{cases}\dfrac{3x^2}{1+x^2}, x\le 0\\ \sqrt{1+\dfrac{2x}{e^{0,5x}+x^2}}, x>0\end{cases}$
4	$y=\begin{cases}\dfrac{3+\sin^2 x}{1+x^2}, x\le 0\\ 2x^2\cos^2 x, x>0\end{cases}$	11	$y=\begin{cases}\sqrt{1+2x^2-\sin^2 x}, x\le 0\\ \dfrac{2+x}{\sqrt[3]{2+e^{-0,1x}}}, x>0\end{cases}$
5	$y=\begin{cases}\sqrt{1+x^2}, x\le 0\\ \dfrac{1+x}{\sqrt[3]{1+e^{-0,2x}}+1}, x>0\end{cases}$	12	$y=\begin{cases}\sqrt{1+\lvert x\rvert}, x\le 0\\ \dfrac{1+3x}{\sqrt[3]{1+x}+2}, x>0\end{cases}$
6	$y=\begin{cases}\dfrac{\sqrt{1+\lvert x\rvert}}{2+\lvert x\rvert}, x\le 0\\ \dfrac{1+x}{2+\cos^3 x}, x>0\end{cases}$	13	$y=\begin{cases}\dfrac{1+\lvert x\rvert}{\sqrt[3]{1+x+x^2}}, x\le -1\\ \dfrac{1+\cos^4 x}{3+x}, x>-1\end{cases}$
7	$y=\begin{cases}\sqrt[3]{1+x^2}, x\le 0\\ \sin^2 x+\dfrac{1+x}{1+e^x}, x>0\end{cases}$	14	$y=\begin{cases}\dfrac{1+x}{\sqrt[3]{1+x^2}}, x\le 0\\ -x+2e^{-2x}, x>0\end{cases}$
8	$y=\begin{cases}2\ln(1+x^2), x\le -1\\ (1+\cos^2 x)^{3/5}, x>-1\end{cases}$	15	$y=\begin{cases}3x+\sqrt{1+x^2}, x\le 0\\ 2\cos x e^{-2x}, x>0\end{cases}$
9	$y=\begin{cases}\sqrt{1+\dfrac{x^2}{1+x^2}}, x\le 0\\ 2\lvert\cos x\rvert, x>0\end{cases}$	16	$y=\begin{cases}\lvert x\rvert^{1/3}, x\le 0\\ -2x+\dfrac{x}{3+x}, x>0\end{cases}$

Продовження таблиці 2.1

17	$y=\begin{cases}\dfrac{1+x}{1+x^2}, x\le 0\\ \sqrt{1+\dfrac{\cos x}{3+x}}, x>0\end{cases}$	24	$y=\begin{cases}\dfrac{1+x+x^2}{1+x^2}, x\le 0\\ \sqrt{1+\dfrac{2\sin x}{1+x^2}}, x>0\end{cases}$
18	$y=\begin{cases}1+\dfrac{3+x}{1+x^2}, x\le 0\\ \sqrt{1+(1-\sin x)^2}, x>0\end{cases}$	25	$y=\begin{cases}\dfrac{1+2x}{1+x^2}, x\le 0\\ \sin^2 x\sqrt{1+x}, x>0\end{cases}$
19	$y=\begin{cases}\dfrac{\lvert x\rvert}{1+x^2}e^{-2x}, x\le 0\\ \sqrt{1+x^2}, x>0\end{cases}$	26	$y=\begin{cases}\dfrac{1+x}{1+\sqrt{\lvert x\rvert e^{-x}}}, x\le 0\\ \cos(3x), x>0\end{cases}$
20	$y=\begin{cases}\dfrac{1+x^2}{1+\sqrt{\lvert\sin x\rvert}}, x\le 0\\ e^{-x}\cos(3x), x>0\end{cases}$	27	$y=\begin{cases}\dfrac{1+\cos x}{1+e^{2x}}, x\le 0\\ 1+\sqrt{1-(x-1)^2}, x>0\end{cases}$
21	$y=\begin{cases}\dfrac{e^{-2x}}{1+\lvert x\rvert}-1, x\le 0\\ e^{-3x}\sin(2x), x>0\end{cases}$	28	$y=\begin{cases}\dfrac{2+\sin x}{1+\sqrt{1+x+x^2}}, x\le 0\\ 1-\sqrt{1-(x-1)^2}, x>0\end{cases}$
22	$y=\begin{cases}\sin xe^{-2x}, x\le 0\\ \dfrac{x^{2/3}}{1+x^2}, x>0\end{cases}$	29	$y=\begin{cases}\sqrt[4]{1+e^{3x}}, x\le 0\\ \dfrac{\cos(5x)}{1+x^2}, x>0\end{cases}$
23	$y=\begin{cases}\dfrac{2+\sin^2 x}{1+x^2}, x\le 0\\ \dfrac{4\cos(3x)}{1+e^{3x}}, x>0\end{cases}$	30	$y=\begin{cases}\dfrac{1+\cos x}{1+e^{2x}}, x\le 0\\ 1+\sin(2x), x>0\end{cases}$

2.3 Контрольні запитання

1 Які види функцій у SMat Studio Вам відомі?
2 Як вставити убудовану функцію в документ SMat Studio?
3 Призначення і використання функції if.
4 Які логічні оператори у SMat Studio Вам відомі?
5 Як вставити логічні оператори в документ SMat Studio?
6 Як створити складну логічну конструкцію?
7 Опишіть способи створення масивів у SMat Studio.
8 Призначення і використання функції mean.
9 Призначення і використання палітри Boolean.
10 Призначення і використання функції rnd.

2.4 Зміст звіту

1 Стислий конспект по теоретичному матеріалу.
2 Розрахунки в системі SMat Studio.
3 Висновки.

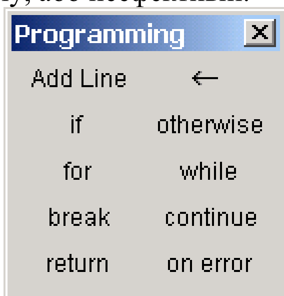
3 ЛАБОРАТОРНА РОБОТА. ПРОГРАМУВАННЯ

Мета: Вивчити призначення операторів мови програмування SMat Studio та засвоїти методи їх використання.

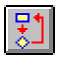
3.1 Загальні відомості

3.1.1 Оператори мови програмування SMat Studio

Для підвищення гнучкості у системі SMat Studio передбачена можливість написання невеликих програм для рішення тих проблем, що не можуть бути реалізовані стандартними засобами. Звичайно прибігати до програмування доводиться в тих випадках, коли стандартні засоби або не можуть вирішити задачу, або неефективні.



Для написання програм використовується програмна палітра, що визивається кнопкою панелі керування. Як видно, усього є 10 операторів, із котрих і будується програма.



Оператори повинні вводитися **тільки з палітри**, писати їх "вручну" не рекомендується.

Опис операторів програмування

Add Line - додати програмний ряд.

← - оператор локального присвоювання.

У програмі не можна використовувати оператор присвоювання ":=", замість нього використовується оператор локального присвоювання, відмінність якого полягає в тому, що локальна змінна визначена тільки усередині свого блока і при виході з програми втрачає своє значення.

Приклад:

$$\left| \begin{array}{l} s \leftarrow 0 \\ i \leftarrow 1 \end{array} \right.$$

if - умовний оператор

Створює конструкцію виду: $\blacksquare \; if \; \blacksquare$
де перший операнд виконується, якщо справедлива умова, котра є другим операндом, наприклад: $x \leftarrow -1 \;\; if \;\; x < 0$

Аналог традиційної конструкції **Якщо ... То ... Інакше ...**

$$\left| \begin{array}{l} x \leftarrow -1 \;\; if \;\; x < 0 \\ x \leftarrow 1 \;\; otherwise \end{array} \right.$$

З оператором **if** можливі більш складні конструкції при використанні ще одного оператора, що реалізує альтернативу.

З оператором **if** можливі більш складні конструкції при використанні ще одного оператора, що реалізує альтернативу. `otherwise`

`for` - оператор циклу

Забезпечує повторювані обчислення якщо відомо кількість кроків. $for \; \blacksquare \in \blacksquare$

$\blacksquare$

Перший операнд - змінна циклу, це інтервальна змінна і її значення визначені в другому операнді. Третій операнд - тіло циклу, що може складатися з блока операторів і виконується поки не вичерпаються всі значення змінної циклу.

Приклад:

$$\left| \begin{array}{l} s \leftarrow 0 \\ for \;\; i \in 0..100 \\ \quad s \leftarrow s + i^2 \end{array} \right.$$

`break` - оператор, введений для підвищення гнучкості програмування і дозволяє закінчити цикл достроково, не вичерпавши всього списку.

Розглянемо наступну задачу: нам необхідно знайти перше входження 0 у числовому масиві і повернути його індекс.

$$t(M) := \left| \begin{array}{l} for \;\; k \in 0..last(M) \\ \quad break \;\; if \;\; M_k = 0 \\ k \end{array} \right.$$

Ми наводимо працюючу програму, де уведена функція *last(M)* яка повертає останній індекс масиву. Значенням програми, що повертається, є останній виконуваний оператор - *k*.

continue - оператор що дозволяє перервати виконання поточної ітерації і перейти до наступної.

Розглянемо наприклад задачу знаходження максимального і мінімального елемента масиву.

$$
minmax(M) := \left|
\begin{array}{l}
min \leftarrow M_0 \\
max \leftarrow M_0 \\
for\ \ k \in 1..last(M) \\
\quad \left|
\begin{array}{l}
if\ \ M_k < min \\
\quad \left|
\begin{array}{l}
min \leftarrow M_k \\
continue
\end{array}
\right. \\
max \leftarrow M_k \quad if\ \ M_k > max
\end{array}
\right. \\
\begin{pmatrix} min \\ max \end{pmatrix}
\end{array}
\right.
$$

$$
B := \begin{pmatrix} 1 \\ -5 \\ 4 \\ 8 \end{pmatrix} \qquad\qquad minmax(B) = \begin{pmatrix} -5 \\ 8 \end{pmatrix}
$$

Дійсно, якщо $M_k < min$, те нема рації перевіряти $M_k > max$, а краще перейти до наступного кроку циклу.

while - оператор циклу з передумовою.

Використовується в тих випадках, коли заздалегідь невідомо кількість кроків, необхідних для рішення задачі. Умова перевіряється перед початком кожного кроку циклу.

$$
\begin{array}{l}
while\ \blacksquare \\
\quad \blacksquare
\end{array}
$$

Реалізуємо наприклад алгоритм обчислення квадратного кореня a використовуючи ітераційну формулу

$$x_n = \frac{x_{n-1}}{2} + \frac{a}{2x_{n-1}}.$$

$$sqrt(a) := \left| \begin{array}{l} x \leftarrow \dfrac{a}{2} \\ \varepsilon \leftarrow 1 \\ while\ \ \varepsilon > TOL \\ \quad \left| \begin{array}{l} z \leftarrow \dfrac{x}{2} + \dfrac{a}{2\cdot x} \\ \varepsilon \leftarrow |x - z| \\ x \leftarrow z \end{array} \right. \\ x \end{array} \right.$$

$$sqrt(2) = 1.414 \qquad\qquad sqrt(10000) = 100$$

return - оператор служить для припинення роботи програми і повернення результату: $return\ \blacksquare$

Припустимо нам необхідно обчислити позицію першого входження числа в масив.

$$num(x, M) := \begin{array}{l} for\ \ i \in 0..last(M) \\ \quad return\ \ i\ \ if\ \ x = M_i \end{array}$$

$$num(4, B) = 2$$

on error - оператор служить для обробки помилкових ситуацій

$$\blacksquare\ \ on\ error\ \blacksquare$$

Наприклад, нам необхідно описати функцію $f(x) := \dfrac{1}{x}$, щоб не було особливості в 0.

$$f(x) := 0\ \ on\ error\ \frac{1}{x} \qquad f(0) = 0 \qquad f(0.01) = 100$$

Примітка. Оператор **on error** може використовуватися в арифметичних виразеннях.

Приклад: Робота з масивами даних(матрицями).

Розглянемо програму упорядкування чисел по убуванню в одномірному масиві. Нехай даний масив чисел:

Ми скористаємося вкладеними циклами й у якості тіла циклу по і використовуємо ще один цикл по j.

$$msort(W) := \left| \begin{array}{l} k \leftarrow last(W) \\ for\ \ i \in 0..k-1 \\ \quad for\ \ j \in k-1, k-2..i \\ \quad\quad if\ \ W_j < W_{j+1} \\ \quad\quad\quad \left| \begin{array}{l} q \leftarrow W_j \\ W_j \leftarrow W_{j+1} \\ W_{j+1} \leftarrow q \end{array} \right. \\ W \end{array} \right.$$

$$Z := \begin{pmatrix} 1 \\ 6 \\ -2 \\ 0 \\ -1 \end{pmatrix} \qquad msort(Z) = \begin{pmatrix} 6 \\ 1 \\ 0 \\ -1 \\ -2 \end{pmatrix}$$

Тут реалізований найпростіший алгоритм сортування, коли більше число як би випливає наверх при кожному кроку циклу по *i,* у той час як у циклі по *j* на кожному кроку відбувається порівняння пари чисел і заміна, якщо більше число знаходиться нижче, причому ця заміна здійснюється знизу. Відзначимо, що в системі є стандартна функція сортування *sort().*

Примітка. Другий цикл ми організували з негативним кроком від кінцевого значення до початкового.

3.1.2 Чисельні методи

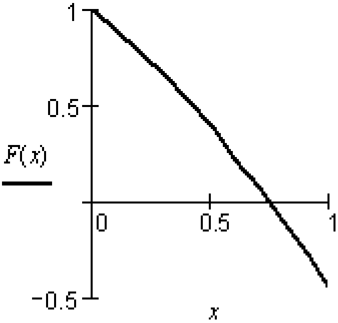
Система програмування в SMat Studio організована досить добре, але на жаль абсолютно відсутня можливість налагодження, що призводить до певних труднощів у пошуках помилки. До речі, більшість помилок пов'язані з виходом індексу за межі масиву.

У якості такої задачі розглянемо найпростіший метод чисельного рішення рівнянь - "метод половинного поділу". Ідею методу розглянемо на прикладі рівняння, аналітичне рішення якого в принципі неможливо.

Перетворимо рівняння $x = cos(x)$ переносячи усе за знак рівності. Розглянемо отриману функцію $F(x) := cos(x) - x$. Очевидно, рішення задачі еквівалентно рішенню рівняння $F(x) = 0$

Побудуємо графік

$x := 0, 0.1 .. 1$



Вирішувати задачу будемо таким способом:

1 задамо відрізок [*a;b*] на якому знаходиться рішення, наприклад [0;1];

2 якщо виконана умова $|a - b| < \varepsilon$ закінчити роботу і повернути результат, інакше;

3 поділимо відрізок навпіл $c := \frac{a+b}{2}$;

4 якщо функція *F(x)* перемінила знак на відрізку [*a;c*], значить там знаходиться рішення і ми визначимо нове значення змінної *b:=c*; інакше *a:=c;*

5 повернемося на пункт 2.

$$bisection(f, a, b) := \left| \begin{array}{l} while\ \ |a - b| > TOL \\ \quad \left| \begin{array}{l} c \leftarrow \dfrac{a+b}{2} \\ b \leftarrow c \quad if\ \ (f(a) \cdot f(c)) < 0 \\ a \leftarrow c \quad otherwise \end{array} \right. \\ \dfrac{a+b}{2} \end{array} \right.$$

Ми скористалися системною змінною *TOL* для завдання точності обчислень і використовували оператор циклу з передумовою *while.*

Для нашої задачі: $bisection(F, 0, 1) = 0.739$

Програма має недоліки:

- якщо рішення на відрізку немає, то результат буде недостовірний. Так, при невдалому завданні інтервалу, одержимо

$$bisection(F, 0, 0.5) = 0.5 \ ;$$

- якщо рішень на відрізку буде більше одного, результат також не очевидний.

Більш коректний алгоритм "метод Ньютона" реалізований в убудованій функції *root().*

3.1.3 Обчислення безкінечних сум

Розглянемо формулу:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - ... + \frac{(-1)^n x^{(2n+1)}}{(2n+1)!}....$$

Для обчислення функції з заданою точністю ε необхідно обірвати підсумовування коли черговий доданок стане менше ε.

$$sin(x) := \left| \begin{array}{l} s \leftarrow x \\ d \leftarrow x \\ n \leftarrow 1 \\ \text{while } |d| > TOL \\ \quad \left| \begin{array}{l} d \leftarrow -d \cdot \dfrac{x^2}{2 \cdot n \cdot (2 \cdot n + 1)} \\ s \leftarrow s + d \\ n \leftarrow n + 1 \end{array} \right. \\ s \end{array} \right.$$

Тут ми скористаємося тим, що кожний наступний член суми виражається через попередній

$$s_n = -s_{n-1} \cdot \frac{x^2}{2n(2n+1)}$$

Для перевірки обчислимо:

$$sin(0) = 0 \qquad sin\left(\frac{\pi}{6}\right) = 0.5 \qquad sin\left(\frac{\pi}{2}\right) = 1$$

3.2 Порядок виконання лабораторної роботи

3.2.1 Виконати всі приклади, приведені в методичних указівках для даної лабораторної роботи.

3.2.2 Складіть програму для знаходження кореня рівняння *f(x)=0* методом Ньютона. **Вказівка:** скористайтеся ітераційною формулою обчислення кореня:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Застосовуючи метод Ньютона, варто керуватися таким правилом: у якості вихідної точки x_0 вибирається той кінець інтервалу [*a, b*], якому відповідає ордината того ж знака, що і знак $f''(x)$.

Варіанти завдань взяти з таблиці 1.1.

3.2.3 Обчислити з заданою точністю 10^{-5} значення функції з таблиці 3.1. Перш ніж писати програму, скласти для даного ряду рекурентне співвідношення, що дозволяє обчислити наступний член ряду через попередній.

Таблиця 3.1 – Варіанти до завдання 3.2.3

№	Функція
1	$S = x + \frac{x^5}{5} + \frac{x^9}{9} + \frac{x^{13}}{13} + \ldots$
2	$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5} - \frac{x^7}{7} + \ldots$
3	$S = 1 - \frac{3}{2!}x^2 + \ldots + (-1)^n \frac{2n^2+1}{(2n)!}x^{2n}$
4	$arctgx = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \ldots,\ 0.1 \le x \le 0.5$
5	$Six = x - \frac{x^3}{3 \cdot 3!} + \frac{x^5}{5 \cdot 5!} - \frac{x^7}{7 \cdot 7!} + \ldots$
6	$S = -(1+x)^2 + \frac{(1+x)^4}{2} + \ldots + (-1)^n \frac{(1+x)^{2n}}{n}$
7	$S = \sqrt{\frac{2x}{\pi}}\left(\frac{x}{3} - \frac{x^3}{7 \cdot 3!} + \frac{x^5}{11 \cdot 5!} - \frac{x^7}{15 \cdot 7!} + \ldots\right)$
8	$arcctg(x) = \frac{1}{x} - \frac{1}{3x^3} + \frac{1}{5x^5} - \ldots + (-1)^n \frac{1}{(2n+1)x^{2n+1}} \ldots.$ $\lvert x \rvert > 1$
9	$\pi \approx 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots + (-1)^n \frac{1}{2n-1} + \ldots\right)$

Продовження таблиці 3.1

№	Функція
10	$berx = 1 - \frac{x^4}{2^2 \cdot 4^2} + \frac{x^8}{2^2 \cdot 4^2 \cdot 6^2 \cdot 8^2} - \ldots$
11	$\varphi(x) = \frac{2}{\sqrt{\pi}}\left[x - \frac{x^3}{1! \cdot 3} + \frac{x^5}{2! \cdot 5} - \ldots + (-1)^n \frac{x^{2n+1}}{n! \cdot (2n+1)}\right]$
12	$u(x) = 1 + \cos\left(\frac{\pi}{4}\right)\frac{x}{1} + \ldots + \frac{\cos\left(n\frac{\pi}{4}\right)}{n!}x^n \ldots$
13	$S = \frac{x^3}{3} - \frac{x^5}{15} + \ldots + (-1)^{n+1}\frac{x^{2n+1}}{4n^2 - 1}$
14	$S = 1 + \frac{\cos x}{1!} + \frac{\cos^2 x}{2!} + \ldots + \frac{\cos^n x}{n!}$
15	$ci(x) = C - \ln x - \frac{x^2}{2 \cdot 2!} + \frac{x^4}{4 \cdot 4!} - \frac{x^6}{6 \cdot 6!} + \ldots,\ C = 0.57722$
16	$li(x) = C + \ln(-\ln x) + \frac{\ln^2 x}{2 \cdot 2!} + \frac{\ln^3 x}{3 \cdot 3!} + \ldots,\ 0 < x < 1,\ C = 0.57722$
17	$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots + (-1)^n \frac{x^{2n}}{(2n)!}$
18	$C(x) = \sqrt{\frac{2x}{\pi}}\left(1 - \frac{x^2}{5 \cdot 2!} + \frac{x^4}{9 \cdot 4!} - \frac{x^6}{13 \cdot 6!} + \ldots\right)$
19	$ber(x) = 1 - \frac{\left(\frac{x}{2}\right)^2}{(2!)^2} + \frac{\left(\frac{x}{2}\right)^8}{(4!)^2} - \frac{\left(\frac{x}{2}\right)^{12}}{(6!)^2} + \ldots$
20	$bei(x) = \frac{\left(\frac{x}{2}\right)^6}{(1!)^2} - \frac{\left(\frac{x}{2}\right)^8}{(3!)^2} + \frac{\left(\frac{x}{2}\right)^{10}}{(5!)^2} - \ldots$
21	$S = 1 + \frac{\ln 3}{1!}x + \frac{\ln^2 3}{2!}x^2 + \ldots + \frac{\ln^n 3}{n!}x^n$

Продовження таблиці 3.1

№	Функція
22	$S=\frac{x\cos\frac{\pi}{3}}{1}+\frac{\left(x\cos\frac{\pi}{3}\right)^2}{2}+\ldots+\frac{\left(x\cos\frac{\pi}{3}\right)^n}{n}$
23	$S=1+\frac{\cos\frac{\pi}{4}}{1!}x+\frac{\left(\cos\frac{\pi}{4}\right)^2}{2!}x^2+\ldots+\frac{\left(\cos\frac{\pi}{4}\right)^n}{n!}x^n$
24	$Ei(x)=C+\ln x+\frac{x}{1\cdot 1!}+\frac{x^2}{2\cdot 2!}+\ldots,\quad C=0.57722$
25	$beix=\frac{x^2}{2^2}-\frac{x^6}{2^2\cdot 4^2\cdot 6^2}+\frac{x^{10}}{2^2\cdot 4^2\cdot 6^2\cdot 8^2\cdot 10^2}-\ldots$
26	$e^{\frac{-x^2}{2}}=1-\frac{x^2}{2}+\frac{x^4}{2^2\cdot 2!}-\frac{x^6}{2^3\cdot 3!}+\ldots$
27	$\ln\frac{1+x}{1-x}=2\left(x+\frac{x^3}{3}+\frac{x^5}{5}+\ldots\right),\quad \lvert x\rvert<1$
28	$\frac{1}{\sqrt{1-x^2}}=1+\frac{1}{2}x^2+\frac{1\cdot 3}{2\cdot 4}x^4+\frac{1\cdot 3\cdot 5}{2\cdot 4\cdot 6}x^6+\ldots$
29	$e^{x^2}=1+x^2+\frac{x^4}{2!}+\frac{x^6}{3!}+\ldots$
30	$\ln(1+x)=x-\frac{x^2}{2}+\frac{x^3}{3}-\frac{x^4}{4}+\ldots,\ \lvert x\rvert<1$

3.3 Контрольні запитання

1 Призначення і використання функції операторів програмування в SMat Studio . Навести приклади.

2 Призначення і використання функції *last*.

3 Призначення і використання функції *sort*.

4 У чому складається ітераційний процес?

5 Як у SMat Studio організувати ітераційний процес?

6 У чому сутність методу половинного поділу?

7 Яка убудована функція знаходить корені рівняння за алгоритмом методу Ньютона?

3.4 Зміст звіту

1 Стислий конспект по теоретичному матеріалу.
2 Розрахунки в системі SMat Studio.
3 Висновки.

4 ЛАБОРАТОРНА РОБОТА. МЕТОДИ АПРОКСИМАЦІЇ Й ІНТЕРПОЛЯЦІЇ ПРИ ОБРОБЦІ ЕКСПЕРИМЕНТАЛЬНИХ ДАНИХ

Мета: засвоїти методи інтерполяції і апроксимації, правила їх застосування в системі SMat Studio.

4.1 Загальні відомості

Експериментальні дані, отримані у лабораторних або промислових умовах, являються основою для проведення подальших досліджень. В результаті проведення експерименту дослідник одержує деяку таблицю значень - табл. 4.1.

Таблиця 4.1 - Таблиця експериментальних значень.

X	x_0	x_1	x_2	. . .	x_N
Y	y_0	y_1	y_2	. . .	y_N

При обробці експериментальних даних можуть виникнути два типа задач:

1 Для функції, що задана як таблиця, потрібно обчислити значення даної функції для проміжного значення аргументу. Цей тип задач розв'язується методом інтерполяції.

2 Для функції, що задана як таблиця або графічно, підібрати аналітичну формулу, яка зображує з якоюсь точністю дані значення функції. Такі формули називаються емпіричними. Задачі даного типу вирішуються методом апроксимації.

4.1.1 Інтерполяція

Розрізняють два види інтерполяції:

- *глобальна* - з'єднання всіх точок $f(x)$ єдиним інтерполяційним поліномом (параболічна інтерполяція, інтерполяційні формули Лагранжа і Ньютона та ін.);

- *локальна* - з'єднання точок відрізками прямої (по двох точках), відрізками параболи (по трьох точках).

Інтерполяція при великому числі вузлів призводить до необхідності працювати з багаточленами високого ступеня (наприклад, 50-го або навіть 100-го). Тому на практиці часто використовують *локальну інтерполяцію*.

Локальна інтерполяція

При *локальній* інтерполяції між різноманітними вузлами вибираються різноманітні багаточлени невисокого ступеня. У середовищі SMat Studio є для цього інструментарій: засоби *лінійної інтерполяції* (функція *linterp*) і *інтерполяції сплайном* (функція *interp*) - лінійним (*lspline*), параболічним (*pspline*) і кубічним (*cspline*).

linterp(*vx, vy, x*)	Використовує вектори даних *vx* і *vy*, щоб повернути лінійно інтерпольоване значення *y*, що відповідає третьому аргументу *x*.
lspline(*vx, vy*) pspline(*vx, vy*) cspline(*vx, vy*)	Всі ці функції повертають вектор коефіцієнтів других похідних, що ми будемо називати *vs*. Вектор *vs*, використовується у функції *interp*:
interp(*vs, vx, vy, x*)	Повертає інтерпольоване значення *y*, що відповідає аргументу *x*.

Найпростішим і часто використовуваним видом локальної інтерполяції є *лінійна інтерполяція.* Вона складається в тому, що задані точки з'єднуються прямолінійними відрізками, і функція $f(x)$ наближається до ламаної з вершинами в даних точках. Рівняння кожного відрізка ламаної лінії в загальному випадку різні. Оскільки є n інтервалів $(x_i\ ,\ x_{i+1})$, то для кожного з них у якості рівняння інтерполяційного полінома використовується рівняння прямої, що проходить через дві точки. У випадку *квадратичної інтерполяції* в якості інтерполяційної функції на відрізку (x_{i-1},x_{i+1}) приймається квадратний тричлен. Інтерполяція для будь-якої точки $x \in [x_0,\ x_n]$ проводиться по трьох найближчих точках.

Кубічна сплайн -інтерполяція

У останні роки інтенсивно розвивається новий розділ сучасної обчислювальної математика - теорія *сплайнів.* Сплайни дозволяють ефективно вирішувати задачу опрацювання експериментальних залежностей між параметрами, що мають достатньо складну структуру. Рисунок 4.1 показує приклади локальної інтерполяції.

Передбачення

Якщо необхідно оцінити значення функції в точках не приналежному відрізку $[x_0,x_n]$, використовуйте функцію *predict*

predict(v, m, n)

Повертає n пророкованих значень, заснованих на m послідовних значеннях вектору даних v.

Ця функція використовує лінійний алгоритм пророкування, що є корисним, коли функція є гладкою і осцілюючою, хоча не обов'язково періодичною. Лінійне пророкування можна розглядати як різновид екстраполяції, але не можна плутати з лінійною або поліноміальною екстраполяцією. Приклад екстраполяції показано на Рисунку 4.2.

$f(x) := x^2 \cdot e^{-x^2}$ - Вихідна функція

$i := 0 .. 5$

$vx_i := \frac{i}{2} \qquad vy_i := f(vx_i)$

- Завдання вихідної функції $f(x)$ таблицею на відрізку [0, 2.5] у 6 вузлових точках.

$z := 0, 0.1 .. 2.5$

Приклад 1: Лінійна інтерполяція

$l(z) := linterp(vx, vy, z)$

Приклад 2: Інтерполяція кубічним сплайном

$vs := cspline(vx, vy)$

$ls(z) := interp(vs, vx, vy, z)$

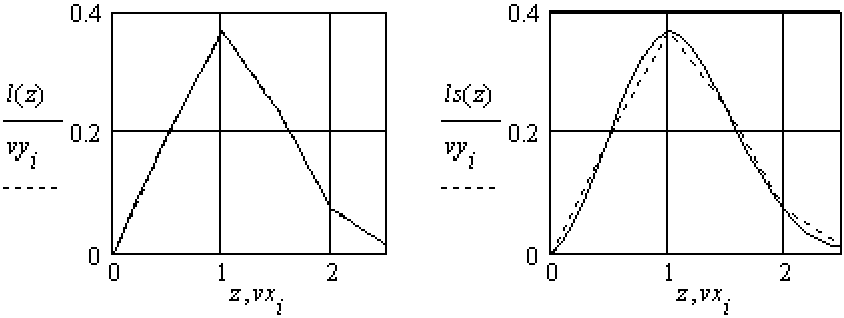


Рисунок 4.1 - Локальна інтерполяція

4.1.2 Апроксимація комбінацією довільних функцій

Лінійна або поліноміальна регресія не у усіх випадках підходять для опису залежності даних. Буває, що потрібно шукати цю залежність у виді лінійних комбінацій довільних функцій, жодна з який не є поліном. Якщо передбачається, що дані могли б бути змодельовані у виді лінійної комбінації довільних функцій

$$f(x) = a_0 f_0(x) + a_1 f_1(x) + \ldots + a_n f_n(x),$$

варто використовувати *linfit*, щоб обчислити a_i. Це так називана *лінійна регресія загального виду.* Приклад показано на Рисунку 6.3.

linfit(vx, vy, F)

Повертає вектор коефіцієнтів лінійної регресії загального виду, щоб створити лінійну комбінацію функцій із *F*, що дає найкращу апроксимацію даних із *векторів* vx *і vy.* *F* - функція-вектор, що складається з функцій, що потрібно об'єднати у виді лінійної регресії.

$a := 0 \quad b := 2$ $\qquad f(x) := exp\left(-\frac{x}{2}\right)\cdot sin(5\cdot x)$ - Вихідна функція

$i := 0 .. (b-a)\cdot 10$

$x_i := a + \frac{i}{10} \qquad y_i := f(x_i)$ - Визначення вихідної функції *f*(*x*) у виді вектори даних на відрізку [0, 2.5]

$p := predict(y, 7, 20)$ - Завбачення значень функції *f*(*x*) у наступних 20 точках по останнім 7 значеннях функції

$t := 0, 1 .. 20 - 1 \quad xp_t := \frac{t + b\cdot 10}{10}$ - Графічна перевірка екстраполяції

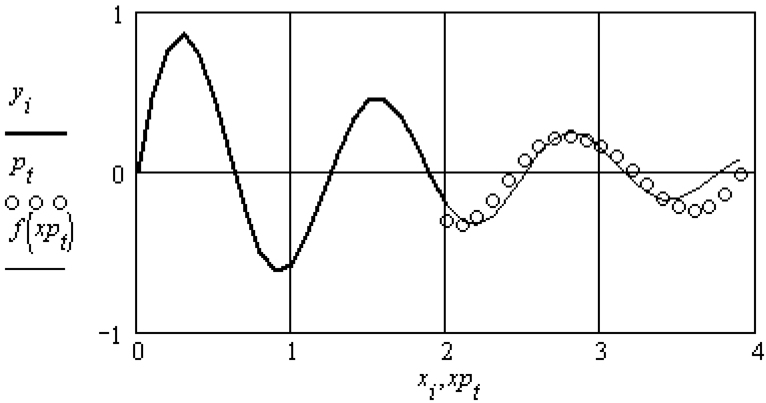


Рисунок 4.2 – Екстраполяція функцій

4.1.3 Згладжування

Згладжування припускає використання набору значень *y* (і можливо *x*) і повернення нового набору значень *y*, що є більш гладким, чим вихідний набір. На відміну від регресії й інтерполяції, згладжування призводить до нового набору значень *y*, а не до функції, яка може оцінювати значення між заданими точками даних.

ksmooth(vx, vy, b)

Повертає *n-мірний* вектор, створений згладжуванням за допомогою Гауссова ядра даних із *n-мірного* вектора *vy*. Параметр *b* управляє вікном згладжування і повинний бути в декілька разів більше розміру інтервалу між точками *x*.

medsmooth(vy, m)

Повертає *n-мірний* вектор, створений згладжуванням *n-мірного* вектора *vy* за допомогою ковзної медіани. *m* - ширина вікна, по якому відбувається згладжування, причому *m* повинно бути непарним числом і $m < n$.

supsmooth(vx, vy)

Повертає *n-мірний* вектор, створений локальним використанням симетричної лінійної процедури згладжування МНК.

$i := 0..5$ $\qquad vx_i := \dfrac{i}{5} + 0.05$ - Вихідні дані

$vy_i :=$

1.9
1.605
1.34
1.22
1.13
1.05

Приклад: Лінійна регресія загального виду

$$F(z) := \begin{pmatrix} \dfrac{1}{z+0.1} \\ z^2 \\ e^z \end{pmatrix}$$ - функція-вектор

$a := linfit(vx, vy, F)$

$l(z) := F(z) \cdot a$

$z := 0, 0.01 .. 1.1$

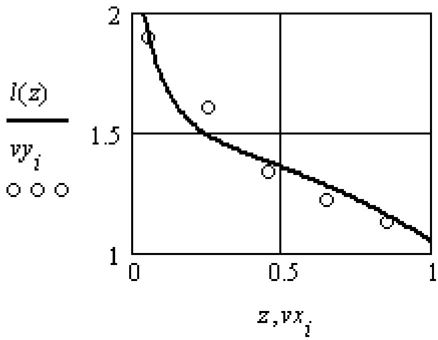


Рисунок 4.3 – Апроксимація комбінацією довільних функцій

4.2 Порядок виконання лабораторної роботи

4.2.1 Обчислити значення заданої функції $y_i = f(x_i)$ у вузлах інтерполяції $x_i = a + h\,i$, де $h = (b - a)/5$, $i = 0, 1, ..., 5$, на відрізку $[a, b]$. Варіанти завдань взяти з табл. 4.2.

Таблиця 4.2 – Варіанти індивідуальних завдань до п. 4.2.1

№	$f(x)$	$[a, b]$	№	$f(x)$	$[a, b]$
1	$\sin x^2$	[0;2]	16	$2x+\cos(x^2)$	[0;2]
2	$\cos x^2$	[0;2]	17	$\sin(x^2)-x$	[0;2]
3	$e^{\sin x}$	[0;5]	18	$0.1x^4-\cos(3x)$	[0;2]
4	$1/(1 + x^2)$	[0;3]	19	$ln(x) +\cos(x)$	[2;4]
5	$e^{-(x + \sin 2x)}$	[0;3]	20	$e^x/(e^x+1-x)$	[0;5]
6	$1/(1 + e^{-2x})$	[0;3]	21	$2x-3\cos(4x)$	[0;1]
7	$\sin(x + e^{\sin x})$	[0;3]	22	$x^3-\cos(5x)$	[0;1]
8	$e^{-(x +2/x)}$	[1;3]	23	$\cos(3x)-x$	[0;2]
9	$x\cdot\cos(x+\ln(1+x))$	[1;5]	24	$e^x/(x^2+1)$	[0;5]
10	$10\cdot\ln 2x/(1+x)$	[1;5]	25	$2\sin(6x)$	[0;1]
11	$\sin(x^2)\cdot e^{-(x/2)}$	[0;3]	26	$3xe^{-x}-1$	[0;2]
12	$\cos(x+\cos^3 x)$	[0;3]	27	$10+66x-3x^3$	[0;4]
13	$\cos(x+e^{\cos x})$	[2;6]	28	$3\sin(x)+x-1$	[1;5]
14	$\cos(2x+x^2)$	[1;2]	29	$\cos(x^2)-0.5x$	[0;2]
15	$e^{\cos x} \cos x^2$	[0;2]	30	x^4-x^2-2	[0;2]

4.2.2 Провести *лінійну інтерполяцію* заданої функції за допомогою убудованої інтерполяційної функції *linterp*.

Побудувати графік функції *linterp* і відзначити на ньому вузлові точки (x_i, y_i).

4.2.3 Провести *сплайн-інтерполяцію* за допомогою функцій *lspline, pspline, cspline* и *interp*. Побудувати графік функції *interp* і відзначити на ньому вузлові точки (x_i, y_i).

4.2.4 Обчислити значення заданої функції $y_i = f(x_i)$ у точках $x_i = a + i/10$, де, $i = 0, 1, ..., 10(b - a)$, на відрізку $[a, b]$.

З використанням функції *predict* виконати *пророкування* (*екстраполяцію*) отриманого вектора даних y_i у наступних 10 точках по останнім 7 значенням функції. Відобразити графічно наявні дані, пророковані дані і істинний вид функції $f(x)$.

4.2.5 Створіть таблицю експериментальних даних: $x_i = a + hi$, $i=0,1,...,10$, $h=(b - a)/10$ на відрізку $[a, b]$; y_i з табл. 4.3.

Таблиця 4.3 – Варіанти індивідуальних завдань до п. 4.2.5

№	y_i	[*a*, *b*]
1	2.86; 2.21; 2.96; 3.27; 3.58; 3.76; 3.93; 3.67; 3.90; 3.64; 4.09	[0, 1]
2	1.14; 1.02; 1.64; 1.64; 1.96; 2.17; 2.64; 3.25; 3.47; 3.89;3.36;	[-1, 1]
3	4.70; 4.64; 4.57; 4.45; 4.40; 4.34; 4.27; 4.37; 4.42; 4.50; 4.62	[2, 4]
4	0.43; 0.99; 2.07; 2.54; 1.67; 1.29; 1.24; 0.66; 0.43; 0.35; 0.70	[2, 4]
5	1.55; 1.97; 1.29; 0.94; 0.88; 0.09; 0.02; 0.84; 0.81; 0.09; 0.15	[1, 4]
6	3.24; 1.72; 1.95; 2.77; 2.47; 0.97; 1.75; 1.55; 0.12; 0.70; 1.19	[0, 4]
7	2.56; 1.92; 2.85; 2.94; 2.39; 2.16; 2.51; 2.10; 1.77; 2.28; 1.70	[-1, 2]
8	1.77; 0.92; 2.21; 1.50; 3.21; 3.46; 3.70; 4.02; 4.36; 4.82; 4.03	[-1, 3]
9	1.53; 0.45; 1.68; 0.12; 0.68; 2.36; 2.58; 2.53; 3.45; 2.70; 2.82	[4, 8]
10	2.50; 3.90; 3.54; 4.63; 3.87; 5.25; 4.83; 3.24; 3.08; 3.00; 4.70	[0, 5]
11	2.95; 3.38; 2.71; 2.37; 2.29; 2.75; 2.76; 2.74; 2.57; 2.40; 2.99	[1, 5]
12	-0.23;-0.03;-0.98;-0.97;-0.43;-0.91;-0.27;-0.19;0.88;1.06;0.72	[2, 4]
13	2.36;0.03;-0.38;-1.33; 0.25;-1.36; 0.95; 3.16; 4.03; 4.92; 4.20	[0, 2]
14	3.82; 4.07; 3.53; 4.83; 5.53; 5.04; 5.09; 5.87; 5.53; 4.72; 4.73	[3, 4]
15	2.35; 2.16; 2.39; 2.39; 2.18; 2.09; 2.44; 2.56; 3.35; 3.22; 2.65	[-3, 4]

4.2.6 Апроксимувати експериментальні дані з таблиць значень x_i і y_i лінійною комбінацією функцій:

$$f(x) = a_1 f_1(x) + a_2 f_2(x) + a_3 f_3(x).$$

Коефіцієнти вектора *a* знайти за допомогою функції *linfit*. Відобразити графічно сукупність точок векторів x_i і y_i та результати проведеної *лінійної регресії загального виду.*

Таблиця 4.4 – Варіанти індивідуальних завдань до п. 4.2.6

№ варіанта	$f_1(x)$	$f_2(x)$	$f_3(x)$
1	e^x	$1/(1+2\cos^2 x)^{1/2}$	sin x
2	$1/(1 + x^2)$	e^x	sin (3x)
3	$1/(1 + x^2)$	$e^{\sin x}$	x
4	arctg x	ln (ln x)	sin x
5	$e^{-\sqrt{x}}$	1 / x	e^{-x}
6	(1+x)/(2 + x)	cos (x/10)	cos x
7	$1/(1+e^{x^2})$	$(1+x^2)^{1/2}$	cos x
8	cos (x/2)	2 - cos x	sin (x/2)
9	$1/(1 + e^x)$	$\text{arctg}\,(x^{1/2})$	sin (3x)
10	ln (x + 5)	$(1+x)^{1/2}$	sin x

Продовження таблиці 4.4

11	1 / x	$(1+x)^{1/2}$	1 / x^2
12	cos x	$1/(1 + x + x^2)$	1/(1 + x)
13	e^{x}	cos 4x	$- e^{x/2}$
14	$(1+e^{-x})^{1/2}$	$e^{x/3}$	$\sin^2 (3x)$
15	$1/(1 + x + x^2)$	cos(x/10)	cos(x/10)

4.2.7 Виконати згладжування експериментальної функції, заданою таблицею значень x_i і y_i за допомогою убудованих функцій SMat Studio: *medsmooth***,** *ksmooth* і *supsmooth***.** Результати згладжування відобразити графічно.

4.3 Контрольні запитання

1 Що таке апроксимація функцій?
2 Для чого потрібна інтерполяція функцій?
3 Охарактеризуйте види інтерполяції.
4 Які види глобальної інтерполяції вам відомі?
5 Що таке емпірична формула і як її підібрати?
6 Що таке екстраполяція?
7 Як можна підвищити точність інтерполяції?
8 Які методи локальної інтерполяції вам відомі? Який із них найменш точний?
9 Який методу локальної інтерполяції проводиться по трьох точках?
10 Яка функція SMat Studio реалізує лінійну інтерполяцію?
11 Які функції кубічної сплайн-інтерполяції вам відомі, охарактеризуйте послідовність із використання?
12 Яка функція SMat Studio використовується для реалізації екстраполяції, опишіть її аргументи?
13 Коли застосовується апроксимація комбінацією довільних функцій? За допомогою яких операторів?
14 Що таке згладжування?
15 Які функції SMat Studio реалізують згладжування?

4.4 Зміст звіту

1 Стислий конспект по теоретичному матеріалу.
2 Розрахунки в системі SMat Studio.
3 Висновки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

Базова

1 Загребельний, С. Л. Програмування на мові С++ у середовищі Visual Studio 2010 : навчальний посібник для студентів спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» С. Л. Загребельний, С. В. Малигіна, М. В. Брус, С. С. Гурковська 2019. – Краматорськ : ДДМА, ISBN 978-966-379-886 146 с.

2 Лозинський, А. Розв'язування задач електромеханіки в середовищах пакетів SMat Studio і MATLAB: Навчальний посібник / Лозинський А., Мороз В., Паранчук Я. – Львів: Видавництво Національного університету "Львівська політехніка", 2000. – 166 с.

3 Інформатика. Комп'ютерна техніка. Комп'ютерні технології: Підручник для студентів вищих навчальних закладів/ За редакцією О.І.Пушкаря. Вид. 2-ге, перероб., доп. - К.: Видавничий центр "Академія", 2002.-704 с.

4 Аверкін С. Короткий посібник з програми SMath Studio та основним її можливостям. URL: http://smath.info/7file =738777.

5 Можливості SMath Studio. URL: http://wiki.pocketz.ru/wiki/SMath_Studio.

Допоміжна

6 Солодовнікова, Т.П. Методичні вказівки до виконання практичних робіт з дисципліни «Обчислювальна техніка та програмування за фахом» для студентів спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» (освітня програма «Електричні машини і апарати») усіх форм навчання. / Укл.: Т.П. Солодовнікова, О.О. Шлянін, Г.В. Дьомічева. – Запоріжжя : НУЗП, 2023. – 59 с.

7 Солодовнікова, Т.П. Методичні вказівки до самостійних робіт з дисципліни «Обчислювальна техніка та програмування за фахом» для студентів спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» (освітня програма «Електричні машини і апарати») усіх форм навчання. / Укл. : Т.П. Солодовнікова, О.О. Шлянін, Г.В. Дьомічева. – Запоріжжя : НУЗП, 2023. – 38 с.

8 Лозинський А., Мороз В., Паранчук Я. Розв'язування задач електромеханіки в середовищах пакетів MathCAD і MATLAB: Навчальний посібник. – Львів: Видавництво Національного університету "Львівська політехніка", 2000. – 166 с.

9 Steinhaus S. Comparison of Mathematical Programs for Data Analysis (Edition 5.03) [Електронний ресурс] – Munchen/Germany. – 64 p. – Режим доступу : http://www.scientificweb.de/ncrunch/.

Інформаційні ресурси

10 Державна бібліотека України для юнацтва [Електронний ресурс] : [сайт] / Державний заклад «Державна бібліотека України для юнацтва». – Електрон. дані. – К., © 2002-2013. – Режим доступу: http://www.4uth.gov.ua , вільний. – Заголовок з екрана. – Мови: укр., рос., англ. – Останнє поновлення: 19.02.2019.

11 Система підтримки дистанційного навчання ЧДТУ. [Електронний ресурс] – Режим доступу: http://ias.cdtu.edu.ua/.

12 Перетворення чисел у різні системи числення [Електронний ресурс] – Режим доступу: http://office.microsoft.com/uk-ua/excel-help/HA010070511.aspx#BMconverts_a_decimal_number_to_binary.

Додаток А
ВБУДОВАНІ ФУНКЦІЇ

Тригонометричні функції

sin(*z***)**	– синус	**csc(***z***)**	– косеканс
cos(*z***)**	– косинус	**sec(***z***)**	– секанс
tan(*z***)**	– тангенс	**cot(***z***)**	– котангенс

Гіперболічні функції

sinh(*z***)** – гіперболічний синус
tanh(*z***)** – гіперболічний тангенс
csch(*z***)** – гіперболічний косеканс
cosh(*z***)** – гіперболічний косинус
sech(*z***)** – гіперболічний секанс
coth(*z***)** – гіперболічний котангенс

Обернені тригонометричні функції

asin(*z***)** – обернений тригонометричний синус
acos(*z***)** – обернений тригонометричний косинус
atan(*z***)** – обернений тригонометричний тангенс

Показові і логарифмічні функції

exp(*z***)** – показова функція (або e^z)
ln(*z***)** – натуральний логарифм (по основі *e*)
log(*z***)** – десятковий логарифм (по основі 10)

Функції роботи з частиною числа

Re(*z***)** - виділення дійсної частини *z*
Im(*z***)** - виділення мнимої частини *z*
arg(*z***)** - обчислення аргументу (фази)
floor(*x***)** - найбільше ціле, менше або рівне *x*
ceil(*x***)** - найменше ціле, більше або рівне *x*
mod(*x,y***)** - залишок від ділення *x/y* із знаком *x*
angle(*x,y***)** - позитивний кут із віссю *x* для точки з координатами (*x,y*)