МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
„КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Числові методи

Методичні вказівки до виконання лабораторних робіт з
дисципліни „Числові методи”
для студентів спеціальності „Автоматизація та комп'ютерно-інтегровані технології”

Рекомендовано Вченою радою інженерно-хімічного факультету

Київ
НТУУ «КПІ»
2016

Числові методи: Метод. вказівки до виконання лабораторних робіт з дисципліни „Числові методи” для студентів спеціальності „Автоматизація та комп'ютерно-інтегровані технології ” / Уклад.: О.В. Ситніков”, 2016. – 96с.

Гриф надано Вченою радою ІХФ

(Протокол № від 2016р.)

Навчальне видання

ЧИСЛОВІ МЕТОДИ
Методичні вказівки до виконання лабораторних робіт з дисципліни „Числові методи” для студентів спеціальності „Автоматизація та комп'ютерно-інтегровані технології”

Укладачі: Ситніков Олексій Володимирович

Відповідальний редактор А.І.Жученко, д-р техн.наук, проф.

Рецензент : О.Л. Сокольський, к.т.н., доц.

Авторська редакція

Зміст

Вступ

В даному методичному посібнику розглянуто основи роботи з числовими методами, розуміння їх суті та призначення. Як засоб реалізації алгоритмів використовується алгоритмічна мова програмування *Pascal* та математичний пакет *MathCad*.

Перші 3 роботи присвячені роботі беспосереднь з мовою програмування *Pascal* та розглянуто основні принципи ООП (оператори, підпрограми). Приведено приклад розрахунок значень функції в циклі, заповнення масиву, робота із заповненим масивом.

Наступні 2 роботи призначені для ознайомлення студентів з середовищем *Delphi* та візуалізація результатів роботи у вигляді побудови графіків функції, приведено приклади створення системного меню.

Подальші задачі будуть вирішуватись за допомогою математичного пакету *MathCad*, максимально наближений до звичайної мови описання математичних задач.

Необхідність застосування пакетів з числовим поданням інформації обумовлена неможливістю для ряду інженерних задач отримати аналітичний розв’язок.

В другій частині 4 роботи призначені для вивчення можливостей *MathCad*, що застосовуються для розв’язання типових інженерних і математичних задач студентами спеціальності “Автоматизація та комп’ютерно-інтегровані технології”, а подальші роботи безпосередньо для реалізацій задач числових методів.

Посібник може бути використаний для самостійної роботи, але слід звернути уваги на те, що при виникненні запитань під час самостійної роботи не буде можливості їх задавати. Всі програми робочі і студенти мають можливість самостійно вносити зміни до програм, вдосконалювати їх.

Лабораторна робота №1

Введення в *Pascal*. Оператори циклу

Мета роботи : Дослідити середовище програмування *Pascal*, операції, які виконуються з операторами циклу. Навчитись працювати з операторами циклу.

Теоретичні відомості.

Мова *Pascal* пристосована для використання на сучасних персональних комп'ютерах типа *IBM PC*. Загальний вигляд вікна приведений на рис. 1.

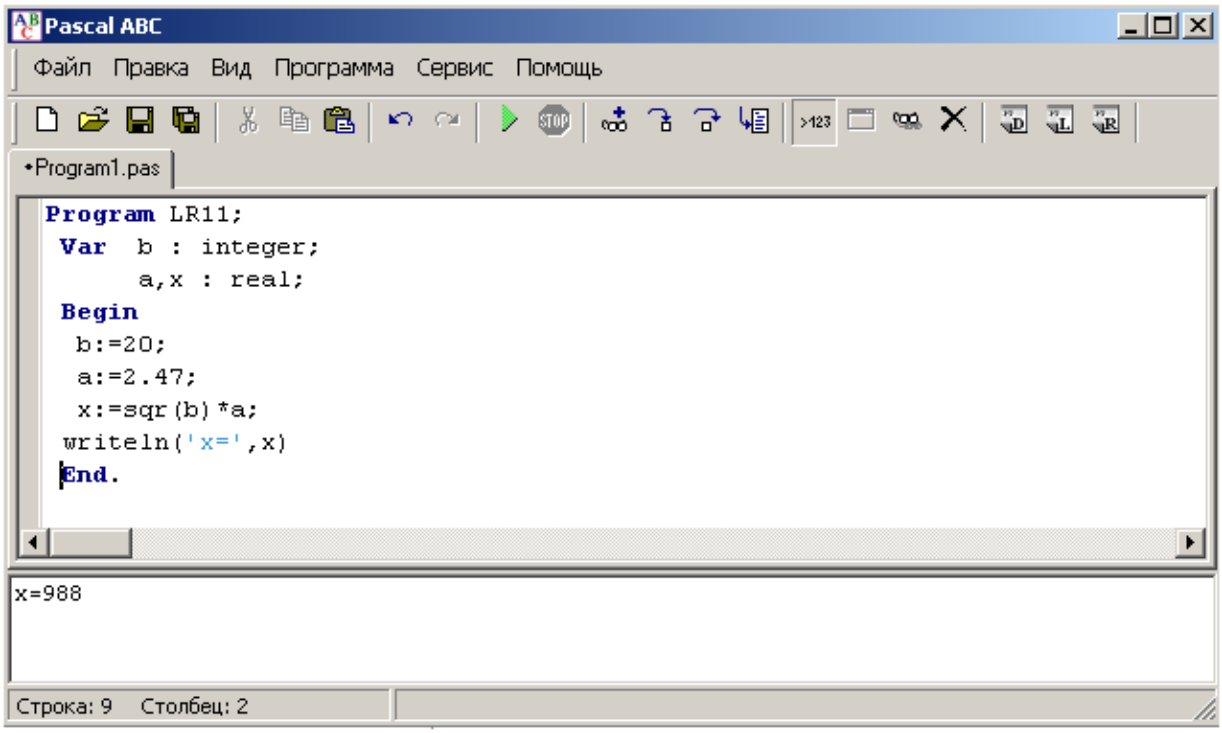


Рис. 1. Загальний вигляд вікна *Pascal ABC*

Меню *Pascal ABC* подібне до стандартного меню *Windows* рис. 2

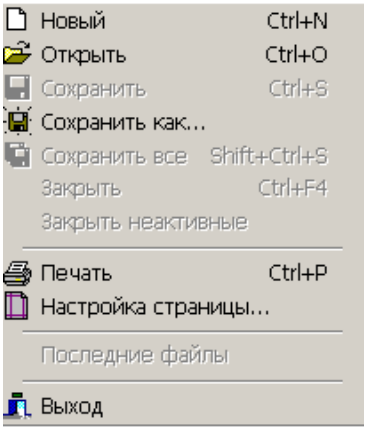


Рис. 2. Складові пункта меню «файл» *Pascal ABC*

- *New*- команда створення нового файла.

- *Save*- команда збереження файла.

-*Open*- команда відкриття файла

-*Exit*- вихід з *Pascal ABC*

-*F*9- виконання програми. 

Програма на *Pascal* складається із заголовка, описової та операторної частини. Заголовок програми починається словом *program*.

В описовій частині відбувається підключення модулів (*uses*), констант (*const*), змінних (*var*), міток (*label*), об'явлення типів (*type*), підпрограм (*procedure*, *function*).

Операторна частина починається словом *begin*, а закінчується *end*.

Переходимо безпосередньо до операторів та деяких стандартних процедур.

Надання значення зміній " := "

a:=5; - "зміній *a* надати значення 5"

Ввід-вивід результату:

Write() може виводити текстову інформацію (в лапках '…') та числову значення змінних : write(A) або write('A=',A).

Writeln(…) – перевід курсору на новий рядок після виконання процесу виводу.

read() – зчитати в значення змінної в дужках те, що введено з клавіатури.

Оператор умовного переходу:

if < логічний вираз > then < оператор >.

Спочатку перевіряється < логічний вираз > (умова). Якщо логічний вираз - *true*, то виконується < оператор >, якщо логічний вираз *false*, то виконується наступний оператор.

Повний оператор.

if < логічний вираз > then < оператор 1 > else < оператор 2 >.

Спочатку перевіряється < логічний вираз > (умова). Якщо логічний вираз - *true*, то виконується < оператор 1 >, якщо логічний вираз *false*, то виконується < оператор 2 >.

В *Pascal* передбачено 3 цикли: арифметичний, з після умовою, з перед умовою.

Арифметичний цикл

for < параметр(змінна) циклу >: = < початкове значення > to < кінцеве значення > do < оператор >;

у випадку коли < початкове значення > менше < кінцеве значення >

for < параметр(змінна) циклу >: = < початкове значение > downto < кінцеве значення > do < оператор >;

у випадку коли < початкове значення > більше < кінцеве значення >

< параметр(змінна) циклу >, < початкове значення > , < кінцеве значення > - тільки цілого типу

Цикл з перед умовою

while < умова зупинки циклу> do < оператор >.

Цикл з після умовою

```
repeat
   < тіло циклу >
until   < умова виходу >.
```

Порядок виконання роботи.

1. Завантажити середовище *Pascal* (з віконки на робочому столі *PascalABC*).
2. Створити новий файл (меню *File* → команда *New*).
3. Зберегти файл під назвою *LR*11(меню *File* → команда *Save*)
4. Набрати текст програми :

```
Program LR11;
    Var  b : integer;
         a,x : real;
    Begin
       b:=20;
       a:=2.47
       x:=sqr(b)*a;
       writeln('x=',x)
     End.
```

5. Відкомпілювати програму (*F9*).

6. Зберегти файл з новою назвою *LR*12 (меню *File* → команда *Save As*).

```
Program LR12;
        Var a : integer;
             y : real;
        Begin
            writeln(‘введите a=’);
            read(a);
            if  a>0 then y:=sqrt(a);
            writeln(‘y=’,y)
         End.
```

7. Відкомпілювати програму.

8. y=_____.

9. Написати програму по розрахунку наступного виразу : $y=tg(a^2+\sqrt{b})-e^{|cos(c)|}$, при умові, що *a*, *b*, *c* константи і дорівнюють відповідно 1.2, 3.7, 4.7. Вивести результат на екран.

y=_____.

10. Написати програму по розрахунку $y=log_2(x)$, якщо $x>0{,}5$ та $y=sin(x)$ в інших випадках, *x* вводити з клавіатури.

Вивести результат на екран. Занести результат.

X_____. Y_____.

11. Зберегти файл з новою назвою *LR*13

12. Набрати текст програми :

```
Program LR13;
Var a, b : integer;
        x : real;
    Begin
    b:=20;
    for a:=1 to 10 do
     begin
```

```
        x:=sqr(b)+sin(sqrt(a));
        writeln('x=',x)
    End.
```

13. Відкомпілювати програму

14 Занести значення *x* до протоколу

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

15. Внести зміни до програми використовуючи цикл з перед умовою.

Початкове значення *a*=1, кінцеве значення *a*=10, крок зміни *a*=0.5

16. Занести результати роботи програми до протоколу

Значення *x :*

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

17. Внести зміни до програми використовуючи цикл з після умовою.

Початкове значення *a*=1, кінцеве значення *a*=10, крок зміни *a*=0.5

18. Занести результати роботи програми до протоколу

Значення *x :*

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, значення *x* та *y* в п. 8, 9, 10, 14, 16, 18; програми з п.9, 10, 15, 17

Контрольні запитання

1. В чому відмінність *Write*() від *Writeln*() ?
2. Пояснити принцип роботи оператора *if*?
3. Які є стандарті функції *Pascal*?
4. Як зміниться вивід змінних на екран при заміні *writeln*('*x*=',*x*) на *writeln*(*x*)
5. Які є стандарті цикли *Pasacl* ?
6. Пояснити принцип роботи арифметичного циклу?
7. Пояснити принцип роботи циклу з після умовою?
8. Пояснити принцип роботи циклу з перед умовою?

Лабораторна робота №2

Робота з масивами.

Мета роботи : Дослідити операції, які виконуються з масивами.

Теоретичні відомості.

Масиви (матриці) можуть бути одно- або багатовимірні, тобто бути розмірністю 1 рядок на декілька стовпців (1 стовпець на декілька рядків) або декілька стовпців на декілька рядків.

В переліку змінних записуються :

одномірні масиви – *A*:*array*[-1..31] *of real*;

багатомірні масиви *B*:*array*[1..10, 1.. 15] *of real*,

де *A*,*B* – змінна типу масив, *array* – ключове слово, яке означає масив, [1..10] – вказує на кількість елементів масиву, 1 – номер першої комірки. 10 – номер кінцевої комірки, *real* – тип змінних, що записуються у комірки масиву.

В загальному випадку масив являє собою таблицю для зберігання значень змінних, що логічно віднесені до якоїсь окремої групи і потрібен для швидкого доступу до цих даних.

Масив заповнюється в циклі наступним чином

```
For s:=1 to 5 do
    A[s]:=sqr(s);
```

де s – номер комірки масиву (з 1 до 5). В данному випадку комірки масиву заповнюються квадратами номерів. Для заповнення комірок використовуються цикл *For* внаслідок того. Що він працює тільки з цілими числами та за замовченням крок 1. Аналогічним чином відбувається вивід елементів масиву.

В масив можна заносити значення кількості елементів. Прийнято для зручності, що кількість елементів масиву заноситься у комірку з індексом -1.

В нульову комірку заноситься перший елемент послідовності. В розглядуваних задачах в масив будуть заноситися коефіцієнти полінома, тобто в нульову комірку заноситься коефіцієнт, що стоїть перед змінною в степені 0, в першу – в степені 1 і так далі до коефіцієнта, що стоїть перед змінною в степені *n*.

Порядок виконання роботи.

1. Завантажити середовище *Pascal*.
2. Набрати текст програми (вивід елементів масиву)

```
Program MAS1;
    Var A:  array [1..5] of integer;
            s:  integer;
    begin
            A[1]:=10;
            A[2]:=20;
            A[3]:=25;
            A[4]:=100;
            A[5]:=-55;
       For s:=1 to 5 do
            Writeln('A[',s,']=',A[s])
    end.
```

3. Відкомпілювати програму. Переглянути результат.
4. Набрати текст програми по вводу елементів масиву з клавіатури та виводу елементів масиву на екран (*uses Crt* – модуль текстового режиму):

```
  Program MAS2;
    Uses Crt;
    Var s,n:integer;
          A:  array [-1..30] of real;
    begin
      Write('Введите количество элементов массива (от 15 до 30) – n=')
      Read(n);
       A[-1]:=n;
       For s:=0 to n do
         begin
           Write('Элемент   ',s,'=');
```

```
        Read(A[s])
      end;
    for s:=0 to n do
        Writeln('A[',s,']=',A[s]:4:2)
  end.
```

5. Відкомпілювати програму. Переглянути та перенести результат в протокол. n=___.

s																											
A[s]																											

6. Набрати текст програми по знаходженню кількості негативних елементів в масиві:

```
Program MAS3;
  Uses Crt;
  Var s, neg: integer;
        A:  array [1..15] of real;
  begin
    For s:=1 to 15 do
      begin
        Write('Элемент  ',s,'=');
        Read(A[s])
      end;
    neg:=0;
    for s:=1 to 15 do
        if A[s]<0  then  inc(neg);
    Writeln('Кількість негативних елементів', neg);
end.
```

7. Відкомпілювати програму. Перенести результат в протокол.

Кількість негативних елементів=____.

8. Написати програму по розрахунку кількості пазетивних елементів в масиву. Вивести результат на екран. Перенести результат в протокол.

Кількість пазетивних елементів=____.

9. Написати програму по розрахунку суми елементів масиву. Вивести результат на екран. Перенести результат в протокол.

Сума елементів масиву=____.

10. Виконати сортування масиву п. 4 від більшого до меншого елемента (в комірку з індексом 0 переноситься значення комірки з максимальним значення) занести дані до протоколу

s																										
A[s]																										

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, тексти виконуваних програм п.5, 7, 8, 9, 10 та їх результати роботи.

Контрольні запитання

1. В чому особливість роботи з масивом *A* коли *A*: *array*[-1..30] *of char*;

2. Як зміниться вивід елементів масиву на екран при заміні *writeln*(‘*A*[‘,*s*,’]=’,*A*[*s*]:4:2) на *writeln*(‘*A*[*s*]=’,*A*[*s*]:4:2) ?

3. При заміні *A*: *array*[-1..30] *of real* на *A*: *array*[3..30] *of real* які зміни відбудуться в програмі п.4 ?

4. Як зміниться процес вводу елементів масиву при заміні від *Write*(‘Элемент ‘,*s*,’=’) на *Write*(‘Элемент *s*=’)

Лабораторна робота №3

Підпрограми

Мета роботи : Дослідити особливості роботи з підпрограмою процедурою та підпрограмою функцією.

Теоретичні відомості.

Важливим принципом сучасного програмування – виступає принцип модульності. Це принцип структурованої програми, шляхом розбиття її на ряд самостійних фрагментів, зв'язаних з основною програмою лише декількома параметрами.

Function < ідентификатор > (< список параметрів >): < тип функції >.

function (функція) – зарезервованне слово. < Ідентифікатор > – ідентификатор функції. Значение цього ідентифікатора повертає підпрограма-функція. Тип ідентифікатора є <типом функції>.

<Список параметрів> – довільний набір ідентификаторів-параметрів, що передаються в функцію з вказівкою на їх тип. Однотипові параметри можуть передаватися групами. Групи отділені один від одного крапкою з комою. Список параметрів може бути відсутній (разом з дужками).

Procedure < ідентификатор > (<список параметрів >);

procedure (процедура) – зарезервованне слово.

< ідентификатор > – назва підпрограми.

< Список параметрів > – довільний набір ідентификаторів-параметрів, що передаються в процедуру з вказівкою на їх тип. Однотипові параметри можуть передаватися групами. Групи отділені один від одного крапкою з комою. Список параметрів може бути відсутній (разом з дужками).

Якщо стоїть необхідність в поверненні з процедури групи параметрів, необхідно встановити слово *var* перед відповідною групою.

Порядок виконання роботи.

1. Завантажити середовище *Pascal*.
2. Набрати текст програми по роботі підпрограми функції

   Program LR31;

```
Var a, b : integer;
        x, y : real;
function  Fx(x: real): real;
 begin
        Fx:=sin(x)+sqrt(x)
 end;
    Begin
    b:=20;  a:=5;
    writeln('x='); read(x);
    y:=Fx(x);
    writeln('y=',y)
End.
```

3. Занести результат роботи до протоколу

Y=_____.

4. Набрати текст програми по роботі підпрограми процедури

```
Program LR32;
Var a, b : integer;
        x, y : real;
procedure  Fx(x: real; var y:real);
 begin
        y:=sin(x)+sqrt(x)
 end;
    Begin
    b:=20;  a:=5;
    writeln('x='); read(x);
    Fx(x,y)
    writeln('y=',y)
End.
```

5. Занести результат роботи до протоколу

Y=_____.

6. Підпрограма може бути використана в циклі. Приклад обрахунку значення функції в циклі при умові зміни аргумента *x* від 1 до 5 з кроком 0,5.

```
Program LR33;
Var a, b : integer;
        x, y : real;
function  Fx(x: real): real;
 begin
        Fx:=sin(x)+sqrt(x)
 end;
Begin
    b:=20;  a:=5; x:=1;
    while x<=5 do
        begin
                y:=Fx(x);
                writeln('y=',y);
                x:=x+1;
        end;
End.
```

7. Занести результат роботи до протоколу

Y:

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

8. Написати програму по обрахунку значення *y* :

Функція: $y=\sqrt{e^{x+4}}$,

Підпрограма: процедура,

x змінюється: в циклі з кроком 1 від 5 до 15,

В кожному кроці циклу виводи значення *x* та *y*.

9. Занести значення *x* та *y* до протоколу

X:_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

Y:_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

11.Підпрограми також можуть бути використані для заповнення масиву.

```
Program LR35;
Var    s: integer;  x : real;
          A: array[1..10] of real;
procedure  Fx(x:real; var A: array[1..10] of real);
 begin
          A[s]:= sqrt(s)/x;
 end;
    Begin
    x:=0.5;
    for s:=1 to 10 do
        Fx(x,A);
    for s:=1 to 10 do
        writeln ('A[',s,']=',A[s]:4:2);
End.
```

11. Занести значення *A*[]до протоколу

s										
A[s]										

12.Заповни в циклі масив з використанням підпрограми процедури, квадратами номерів комірок з 10 по 19, вивести заповнений масив на екран

13.Занести значення *A*[]до протоколу

s										
A[s]										

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, тексти виконуваних програм п.2, 4, 6, 8, 10, 12 та їх результати роботи.

Контрольні запитання

1. Навести приклад заголовку підпрограми-функції ?
2. Навести приклад заголовку підпрограми-процедури ?
3. Де в наступному рядку помилка Function Fx (x:real; var y:real) :integer; ?
4. Де в наступному рядку помилка Procedure ABC(x:integer, y:real): real; ?

Лабораторна робота №4

Створення найпростіших програм та їх виконання. Компонент «Меню», вибору із списків

Мета роботи : Навчитись створювати найпростіші програм засобами *Delphi*, програмне меню, використовувати елементи списків.

Теоретичні відомості

Delphi являє собою середовище розробки, за основу синтаксисису взята мова *Pascal*.

На рис. 1 приведено назви основних складових робочого вікна *Delphi*.

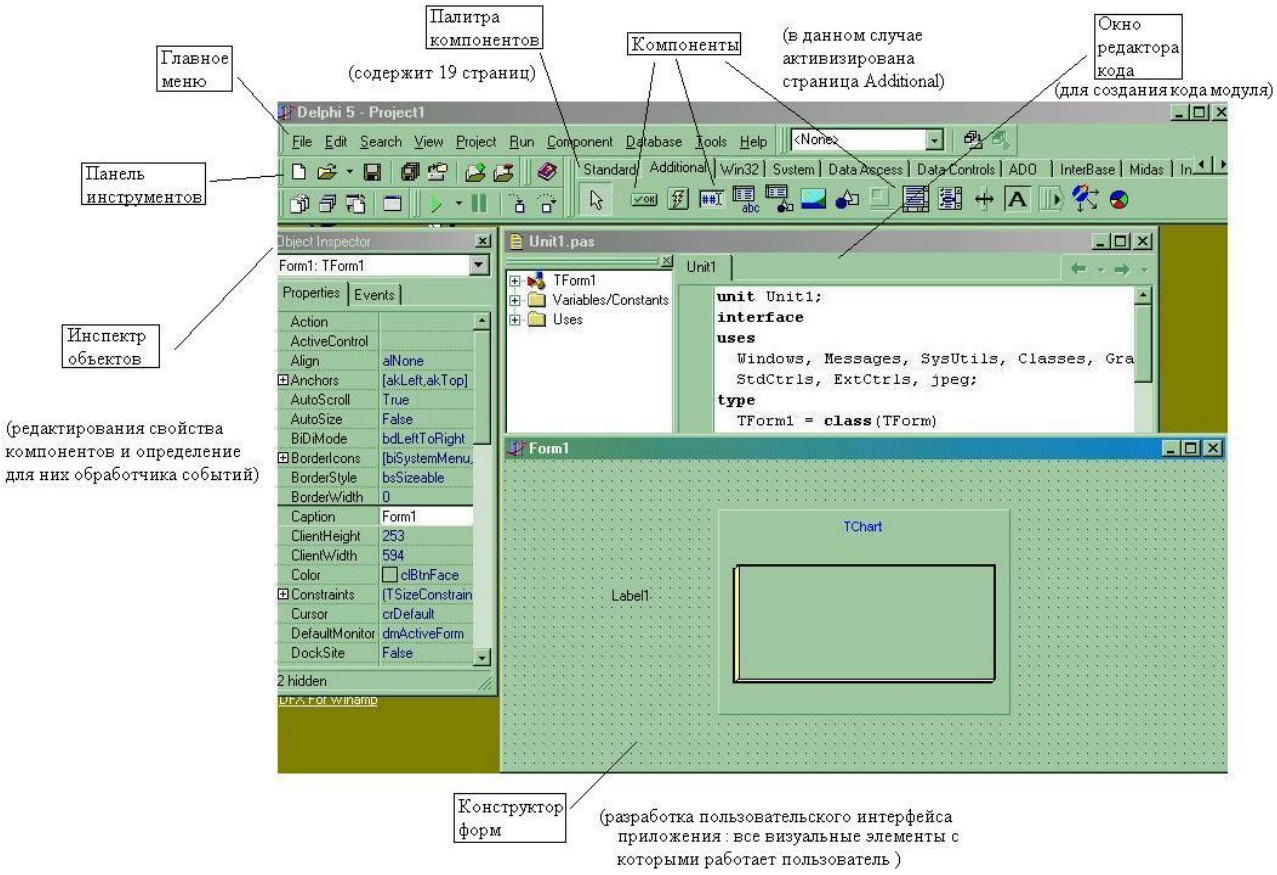


Рис.1 основні складові робочого вікна *Delphi*.

В палітрі компонентів знаходяться об’єкти (компоненти) клікаючи по компоненту, а потім по полю форми об’єкт переноситься в поле форми для використання. Подвійне натискання по компоненту лівою кнопкою миші призводить до переходу у програмний код даного компоненту. Якщо клікнути

по об’єкту один раз – він стає активний, що дозволяє вностит зміни на його властивості в інспекторі об’єктів.

При переході до програмного коду в даній роботі будуть використані наступні процедури :

- IntToStr (…) - перетворення значення в дужках цілочислового типу до рядкового типу.
- StrToInt (…) - перетворення значення в дужках рядкового типу до цілочислового типу.
- FloatToStr (…) - перетворення значення в дужках дійсного типу до рядкового типу
- StrToFloat (…) - перетворення значення в дужках рядкового типу до дійсного типу

Об’єкти *Edit* працює виключно з числовим типом, на відміну йому *Label* працює з рядковими змінними.

Створення меню *MainMenu*. - створення меню програми.

За допомогою цього об'єкту можна створити меню у програмі. Для встановлення назви пунктів та підпунктів меню досить двічі клацнути на об'єкт лівою клавішею мишки. На екран буде виведено меню (рис. 2)

Рис.2. Вигляд етапу створення меню

Далі слід вказати назву головного пункту меню. Наприклад *File*. Дана дія відбувається в інспекторі об’єктів. Далі, рухаючись униз можна додавати підпункти меню (слід вказувати їх назви), і вправо – пункти меню. Якщо

клацнути у будь-який з підпунктів меню двічі лівою клавішею мишки, то автоматично відбудеться перехід, у якій треба буде вказати послідовність операторів, що виконуватимуться при виборі відповідного пункту меню при роботі програми. Об'єкт Меню розташовується у програмі автоматично у лівому верхньому куті вікна програми.

Текстове вікно *ListBox*. Використовується для виведення більше за один рядок тексту у вікно з можливістю перегляду (прокрутки) тексту вверх-вниз. Видима ширина рядку залежить від ширини вікна *ListBox*. Частину властивостей рядок редагування має таку, яка є і у кнопки та статичного тексту.

Процедури та функції *ListBox*:

- ListBox1.Items.Append. - для зміни тексту у *ListBox* протягом роботи програми, параметр s є рядком. При виконанні цієї процедури, знизу до рядків дописується ще один з s.
- ListBox1.Items.Clear. - процедура очищає вікно *ListBox* від всього тексту.
- ListBox1.Items.Delete. - знищує рядок з номером i, де i – ціла змінна(див. типи змінних).
- ListBox1.Items.Insert. - вставляє текст s у рядок з номером i. Змінна s – рядкова, i – ціла. При цьому всі рядки після i зсуваються донизу. Наприклад у нас є *ListBox* з текстом з чотирьох рядків

  ‘1 перший’

  ‘2 другий’

  ‘3 третій’

  ‘4 четвертий’

  Тоді після запуску процедури ListBox1.Items.Delete(2); стан ListBox буде

  ‘1 перший’

  ‘2 другий’

  ‘4 четвертий’

Після запуску процедури ListBox1.Items.Insert(1,'новий рядок');

стан *ListBox* буде

'1 перший'

'новий рядок'

'2 другий'

'4 четвертий'

Порядок виконання роботи

1. Відкрити середовище *Delphi*.
2. На палітрі компонентів оберіть закладку *Standard*.
3. У вибраній закладці *Standard*, натисніть на піктограмі *Button*, а потім на формі *Form1*. У результаті цих дій на форму буде додана кнопка, розміри і розміщення якої ви можете змінити. Розмістіть її ближче до нижньої частини форми.
4. Натисніть на *Button1* ще раз і у вікні Інспектора Об'єктів змініть властивість *Caption* кнопки на значення "Додати".
5. Тепер, так само, розмістіть на формі 4 мітки *Label* і 2 вікна редагування *Edit*. Розмістіть ці об'єкти по вертикалі у такому порядку: *Label1*, *Edit1;* *Label2*, *Edit2;* *Label3*, *Label4;* *Button1*.
6. Використовуючи Інспектора Об'єктів змініть надписи на мітках. Змініть властивість *Caption* (надпис) мітки *Label1* на а, а мітки *Label2* – на в; мітки *Label3* – на а+в=, мітки *Label4* – зробіть пустими. Змініть властивість *Caption* для форми.
7. Привласніть пустий рядочок в якості значення властивості *Text* об'єктам *Edit1* і *Edit2*.
8. Натисніть двічі на зображення кнопки *Button1* для того, щоб додати в кнопку програмний код обробника подій. *Delphi* створить код по замовчуванню для події кнопки *OnClick* і розмістить його в редактор між *begin* і *end*.

9. Впишіть між цими словами код:

Label4. Caption:=IntToStr (StrToInt(Edit1.Text)+StrToInt(Edit2.Text));

9. Виконайте команду *File*/*Save All* (зберегти всі) – збережіть програму у своїй власній папці на диску *D*.

10. Натисніть комбінацію клавіш *Ctrl + F9* для компіляції і запуску програми.

11. Скласти програму для розв'язку квадратного рівняння

12. Розмістіть на формі 4 мітки *Label*, 3 вікна редагування *Edit* та кнопку *Button1*.

Розмістіть ці об'єкти по вертикалі у такому порядку: *Label1*, *Edit1;*
Label2, *Edit2;*
Label3, *Edit3*
Label4;
Button1.

13. Використовуючи Інспектора Об'єктів змініть надписи на мітках. Змініть властивість *Caption* мітки *Label1* на *a*, а мітки *Label2* – на *b*; мітки *Label3* – на *c*, мітки *Label4* – зробіть пустою. Змініть властивість *Caption* для форми *TForm1*– „Розв'язування квадратного рівняння."

14. Привласніть пустий рядочок в якості значення властивості *Text* об'єктам *Edit1*, *Edit2* та *Edit3*.

15. Натисніть на ПКМ *Button1* і у вікні Інспектора Об'єктів змініть властивість *Caption* кнопки на значення "Порахувати".

16. Натисніть двічі ЛКМ на зображення кнопки *Button1* для того та додати в кнопку програмний код обробника подій

```
var a,b,c,x1,x2:real;
begin
   a:=StrToFloat(Edit1.Text);
   b:=StrToFloat(Edit2.Text);
   c:=StrToFloat(Edit3.Text);
```

```
d:=b*b-4.0*a*c;
 if d<0.0 then Label4. Caption:=’Коренів немає’
            else
              begin
                    x1:=(-b+sqrt(d))/2.0*a;
                     x2:=(-b-sqrt(d))/2.0*a;
                    Label4. Caption:=Format(‘x1=%8.4f  x2=%10.4f’, [x1,x2]);
               end;
  end;
```

17. Виконайте команду *File*/*Save All* (зберегти всі) – збережіть програму у своїй власній папці на диску *D*.

18. Натисніть комбінацію клавіш *F*9 для компіляції і запуску програми.

19. Створити новий проект

20. Виберіть на панелі компонентів *Standard* компонент *TmainMenu* і розмістіть його на формі в будь-якому місці форми . У властивості *Caption* (Заголовок) введіть назву першого пункту “Файл” та натисніть Enter.

21. Натисніть один раз правою на об’єкті *TMainMenu1* (Файл) і один раз на кнопці (меню - внизу), яке створилося .

22. Система переключиться до заголовку *Caption* нового пункту. Введіть нову назву (&Додати) + *Enter*.

23. Знову натисніть на кнопці „Додати” – у формі, і один раз на кнопці (меню - внизу), яке створилося .

24. У властивості *Caption* введіть символ „-” (дефіз), щоб вставити лінію, яка розділяє надписи.

25. Для вставки нових пунктів служить клавіша - *Insert*, для видалення – *Delete*. Останнім додаємо пункт “Вихід”.

26. Створити обробник події - двічі натисніть на пункті Вихід у редакторі меню і запишіть: *Close*;

27. Створити вкладений обробник події - двічі натисніть на пункті Додати у редакторі меню і запишіть : *Button1Click*(*Sender*);

28. У властивості *Caption* нової Форми введіть її назву *tmm*.

29. Виконайте першу частину лабораторної роботу №12.

30. Виконайте команду *File*/*Save All* (зберегти всі) – збережіть програму у своїй власній папці на диску D.

31. Натисніть *F*9, продивитись роботу програми, закрити вікно програми.

32. Виберіть компонент *Image* вкладниці *Additional* і розмістить його на форму.

33. У вікні *Object Inspector* виберіть властивість *Picture* і натисніть на кнопці з трьома крапками та натисніть *Load*, виберіть довільний малюнок з папок диску, та натисніть ОК. (якщо розмір ілюстрації більший за розмір компонента, то властивості *Strech* потрібно присвоїти значення *True*.)

34. Відкрити новий проект

Размістіть на формі компоненти у відповідності :

ListBox *Label1*

Label2

Label3

35. В *Object Inspector* для *Label1* у властивості *Caption* написати “Гороскоп с 22 ноября по 23 декабря”.

36. В *Label2*, у властивості *Caption* написати “Благоприятные Неблагоприятные”.

37. В *Label3* у властивості *Caption* видалити слово “*Label3*”.

38. Виділити *ListBox1*, в *Object Inspector* у властивості *Items*, праворуч від нього натиснути кнопку з трьома крапками. У вікні, що з`явилося, ввести назви знаків зодіаку, кожний з нового рядочку, натиснути ОК

39. В *Object Inspector’e* знайти подію *OnKeyPress*, справа від неї у полі зробити подвійне натскання лівою кнопкою миші та ввести програмний код

```
if key=#13 then
case Listbox1.ItemIndex  of
0: Label3.Caption:='20, 24-13 ';
1: Label3.Caption:='26-4                    14,15,22      ';
```

```
2: Label3.Caption:='12-19                                    ';
3: Label3.Caption:='24,30,2,6,12,16          26,5,18       ';
4: Label3.Caption:='14,15,16,19-22           27,2,3        ';
5: Label3.Caption:='31,6,7,10                24-26         ';
6: Label3.Caption:='29-31,6,12,20            26            ';
7: Label3.Caption:='24,29,31,2,6,12,20       26,28,5,18    ';
8: Label3.Caption:='24-30                    5,12          ';
9: Label3.Caption:='23-22                                  ';
10: Label3.Caption:='23-22                                  ';
11: Label3.Caption:='20                       1,2           ';
end;
```

40. Виконайте команду *File*/*Save All* (зберегти всі) – збережіть програму у своїй власній папці на диску *D*.

41. Натисніть клавішу *F*9 і продивитись роботу програми.

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з текстами програм.

Контрольні запитання

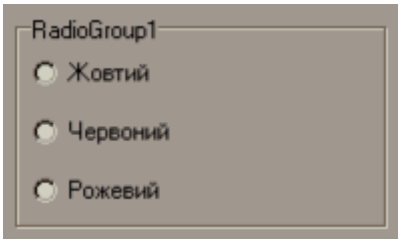
1. Що таке палітра компонентів?
2. Яка відмінність у StrToFloat та StrToInt ?
3. Яка відмінність у StrToInt та IntToStr ?
4. Що робить об’єкт *Edit*?
5. Що робить об’єкт *Button*?
6. Що робить об’єкт *Label*?
7. Що означає запис *Label*1.*Caption*:= ‘123’ ?
8. Що таке інспектор об’єктів?
9. Що робить об’єкт *TMainMenu*?
10. Як розділити пункти меню лінією?
11. Що робить об’єкт *ListBox*?
12. Що робить об’єкт *Image*?

Лабораторна робота №5

Таймер, група перемикачів. Побудова графіків функці

Теоретичні відомості

RadioGroup дозволяє вибрати одне значення з переліку. Для встановлення значень слід вказати початкові значення. Для цього треба зробити це або через параметр *Radiogroup*1.*Items* (як це робилось у *ListBox*) або програмно. У першому випадку на екран буде виведено вікно, у якому можна ввести рядкі. Наприклад вводяться три кольра жовтий, червоний, рожевий:



Головною подією для *RadioGroup* є натискання на неї (яке взагалі відбувається разом з вибором значення). Частину властивостей *RadioGroup* має таку, яка є і у *ListBox*. З важливих нових властивостей рядок редагування має - порядковий номер вибраного значення у переліку. За замовчанням він має значення –1. Нумерація рядків здійснюється від 0. Тобто при виборі першого рядку значення *RadioGroup*1.*ItemIndex* буде 0. Знищення, додавання та вставка рядку здійснюється так само, як і у *ListBox*, тільки назвою об'єкту є не *ListBox*, а *RadioGroup*. Тобто використовуються процедури *RadioGroup*1.*Items*.*Append*, *RadioGroup*1.*Items*.*Clear*, *RadioGroup*1.*Items*.*Delete*, *RadioGroup*1.*Items*.*Insert* відповідно.

Для об'єкту *Timer* стає необхідність наявності в коді процедури циклічної зміни параметру. Слід зазначити, щозначення швидкодії роботи таймеру знаходиться у інспекторі об'єктів.

Об'єкт *SpinEdit* містить в собі елемент прокрутки, може приймати значення тільки цілочислового типу. Зручний інструмент коли необхідно задати границю та спостерігати зміну результату.

Для представлення даних з деякого набору даних у вигляді графіків різних видів призначений компонент *TChart*. За його допомогою можна

одночасно показувати графіки для декількох полів даних. Графіки будуються на основі всіх наявних в наборі даних значень полів.

Налаштування параметрів компоненту здійснюється спеціальним редактором, який можна відкрити подвійним клацанням на перенесеному на форму компоненті.

Основою будь-якого графіка в компоненті *TChart* є так звана серія, властивості якої представлені класом Tchartseries. Для того, щоб побудувати графік значень деякого поля набору даних, необхідно виконати наступні дії, більшість з яких виконується в спеціалізованому редакторі компоненту:

Створити нову серію і визначити її тип.

Задати для серії набір даних.

Пов'язати з осями координат потрібні поля набору даних і, залежно від типу серії, задати додаткові параметри.

Відкрити набір даних.

Редактор має дві головні сторінки - *Chart* і *Series*. Сторінка *Chart* містить багатосторінковий блокнот і призначена для настройки параметрів самого графіка. Сторінка *Series* також містить багатосторінковий блокнот і використовується для настройки серій значень даних.

Для створення нової серії необхідно в редакторові перейти на головну сторінку *Chart*, а на ній відкрити сторінку *Series*. На цій сторінці потрібно клацнути на кнопці *Add,* а потім в діалозі, що з'явився, вибрати тип серії. Після цього в списку на сторінці Series з'являється рядок нової серії. Тут можна перевизначити тип, колір і видимість серії, клацнувши на відповідній зоні рядка. Решта всіх сторінок блокнота на головній сторінці *Chart* призначена для настройки параметрів графіка. Можна перейти на головну сторінку *Series* і на ній із списку назв серій вибрати необхідну.

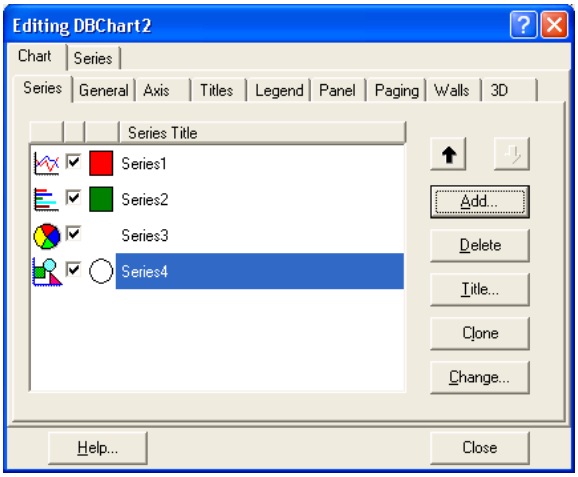


Рис. 1. Спеціалізований редактор компоненту TChart.

Список *X* дозволяє вибрати поле набору даних, значення якого послідовно відкладатимуться по осі абсцис. Список *Y* дозволяє вибрати поле набору даних, значення якого будуть відкладені по осі ординат. Відповідність між значеннями полів по двох осях визначається приналежністю до одного запису набору даних. Вибір поля в списку *Labels* прив'язує його значення у вигляді міток до осі абсцис.

Тепер залишилося тільки відкрити набір даних і компонент *TChart* побудує графік. Аналогічним чином на цей же компонент можна помістити і інші графіки. Кожна серія відповідатиме одній кривій на графіку.

Редактор діаграм викликається:

- кнопкою з три крапкою поряд з назвою властивості в інспекторові об'єктів; подвійним клацанням на компоненті *Chart* при проектуванні форми;
- вибором команди *Edit Chart* в контекстному меню компоненту *Chart* при проектуванні форми.

Для задання значень, що відображаються, треба використовувати методи серій Series:

clear - очищає серію від занесених раніше даних;

add - дозволяє додати в діаграму нову точку:

Add(Const Avalue:double; Const Alabel:string; Acolor:tcolor)

Параметр *Avalue* відповідає значенню, що додається, параметр *Alabel* - назва, яка буде відображатися на діаграмі і в легенді, параметр *Acolor* - колір. Параметр *Alabel* необов'язковий, його можна задавати порожнім.

addxy - дозволяє додати нову крапку в графік функції:

ADDXY(Const Axvalue, Ayvalue: Double; Const Alabel: String; Acolor: Tcolor).

Параметри *Axvalue* і *Ayvalue* відповідають аргументу і функції, параметри *Alabel* і *Acolor* - ті ж, що і в методі *Add*.

Порядок виконання роботи

1. Створіть нову форму, у властивості *Caption* введіть назву проекту „Компонент Перемикач". Збережіть проект, натиснувши *File*/*Save All* (зберегти всі).
2. На палітрі компонентів панелі *Delphi* оберіть закладку *Standard*.
3. У вибраній закладці *Standard* оберіть піктограму *TRadioButton* і покладіть її на форму.
4. Так як ми створюємо групу Перемикачів, то на форму треба покласти 2 компонента *TRadioButton*. Програма зрозуміє, що виділеним повинен бути тільки один із них. Властивість *Alignment* визначає положення підпису праворуч або ліворуч від перемикача, а властивість *Checked* - стан об'єкта (*true* , якщо Перемикач включено). Для відслідкування стану конкретного Перемикача потрібно оброблювати подію *OnClick*.
5. Покладіть на форму *Label1*.

6. Натисніть двічі на зображення кнопки *TRadioButton1* для того, щоб додати в кнопку програмний код обробника подій *OnClick*.

Між словами begin і end впишіть програмний код:

```
if RadioButton1.Checked
then Label1.Caption := 'Вімкнено перший'
end;
```

7. Натисніть двічі на зображення кнопки *TRadioButton2* для того, щоб додати в кнопку програмний код обробника подій *OnClick*.

Між словами begin і end впишіть програмний код:

```
if RadioButton2.Checked
then Label1.Caption := ‘Вімкнено другий’
end;
```

8. Натисніть комбінацію клавіш *Ctrl + F9* для компіляції і запуску програми.

9. Збережіть проект, натиснувши File/Save All.

10. Створіть нову форму, у властивості *Caption* введіть назву проекту „Тест”.

12. На закладці *Standard* оберіть піктограму *TRadioGroup* і покладіть її на форму.

13. У властивості *Caption TRadioGroup* введіть текст „Оберіть вірну відповідь”.

14. Покладіть на форму також компоненти *Label1* і *Label2*, *BitBtn*.

15. У властивості *Caption Label1* введіть текст „Як переключитись із дизайнера форм у редактор коду?”.

16. У властивості *Caption Label2* видаліть текст.

17. У властивості *King* компонента *BitBtn* оберіть вкладку *bkCancel*.

18. У властивості *Items* компонента *TRadioGroup* введіть назви Перемикачів (F1, F2, F3, F12, F11, F5), по одному на кожному рядочку, натискаючи Enter. Натисніть OK.

19. Створіть обробник події. Двічі лівою натисніть на компоненті *TRadioGroup*.

19. Між словами *begin і end* треба записати код. Оскільки правильна відповідь F12 (номер у списку 3, рахуємо 0, 1, 2, 3, 4, 5), то маємо:

```
if RadioGroup1.ItemIndex = 3 then
Label1.Caption := ‘Вірно’ else
Label1.Caption := ‘Невірно’
end;
```

21. Збережіть проект, натиснувши *File*/*Save All*

22. Натисніть комбінацію клавіш *Ctrl + F9* для компіляції і запуску програми.

23. Створіть нову форму, у властивості *Caption* введіть назву проекту „Таймер”.

24. На палітрі компонентів панелі *Delphi* оберіть закладку *Samples*

25. Розмістіть на формі компонент *SpinEdit* та *Label,* з палитри компонентів *Standard*.

26.Використовуючи Інспектора Об’єктів зробити пустим надпис на мітці (властивість *Caption*),

27.Розмістіть на формі компонент *Timer* з палітри компонентів *System*, двічі натиснути по об’єкту.

26.Ввести наступний код

```
x := spinedit1.value;
Q := x*x;
Label1.Caption := 'Q= '+ floattostrf(Q, ffFixed, 3, 1)+";
```

27.В описі змінних програми задати x:integer; та Q:real;

28. Натисніть комбінацію клавіш *Ctrl + F9* для компіляції і запуску програми.

29. Вийти з режиму виконання.

30.Змінити програмний код :

```
Q := Q+x;
```

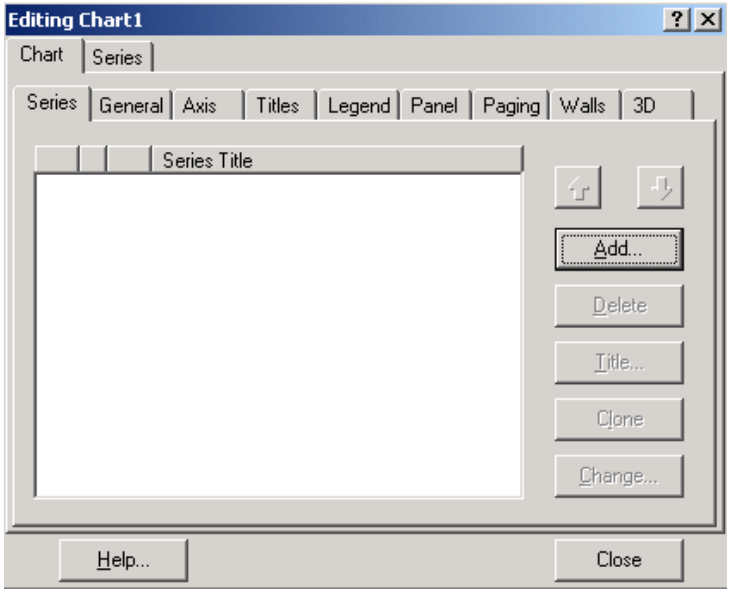
31. Збережіть проект, натиснувши File/Save All.

32. Натисніть комбінацію клавіш *Ctrl + F9* для компіляції і запуску програми.

33. Відкрийте форму *Form1*.

34. Розмістіть на ній компонент *Chart* з палітри компонентів *Additional*, двічі натиснути по об’єкту.

35. З’явиться наступне вікно



необхідно натиснути *Add*, вибрати тип побудови графіка та натиснути *Ок* і закрити вікно настройки графіка.

36. Розмістіть на формі об’єкт *Button*, двічі натиснути по ньому і ввести наступний код

```
for i:=0 to 100 do
begin
  Series1.AddXY(0.02*Pi*i,sin(0.02*Pi*i),'',clRed);
end;
```

вказати локальні змінні

```
var i:integer;
```

37. У властивості Caption прописати : «ПОБУДОВА».

38. Вибрати знову об’єкт *Button*, розмістити на формі, двічі натиснути по ньому і ввести

наступний код

```
Series1.clear;
```

39.У властивості Caption прописати : «СТЕРТИ»

40.Запустити програму на виконання

41. Двічі натиснути по компоненту *Chart*

42. Натиснути *Add*, вибрати ще один графік та натиснути *Ок* і закрити вікно настройки графіка.

43. Розмістіть на формі об’єкт *Button*, двічі натиснути по ньому і ввести наступний код

```
for i:=0 to 100 do
begin
  Series2.AddXY(0.02*Pi*i,cos (0.02*Pi*i),",clRed);
end;
```

44. Вибрати знову об’єкт *Button* 2, двічі натиснути по ньому і ввести та додати наступний код

```
Series2.clear;
```

45. Збережіть проект, натиснувши File/Save All.
46. Натисніть клавішу *F9* для компіляції і запуску програми.

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з текстами програм.

Контрольні запитання

1. Основна відмінність *TRadioButton* від *CheckBox*?
2. Призначення об’єкту *Time*?
3. Основна відмінність *SpinEdit* від *Edit*?
4. Що означає даний запис floattostrf(Q, ffFixed, 3, 1) ?
5. Які пункти присутні у вікні побудови графіка, їх призначення?
6. Якій принцип побудови графіка функцій засобами *Delphi* виходячи із наступного кода Series1.AddXY(k,cos(k),",clRed);
7. Як внести зміни в побудовану графічну залежність?

Лабораторна робота №6

Побудова виразів обчислення з *MahtCad*. Застосування програмування для розв'язання поставлених задач

Мета роботи – дослідити можливості редактора формул, навчитися використовувати можливості програмування *MathCad* для розв'язання поставлених задач

Теоретичні відомості

Створення математичних виразів виконується за підтримки формульного редактора. До складу математичного виразу входять оператори та операнди. Прості операнди можуть бути числом, ідентифікатором змінної, матриці, функції. Складні оператори, в свою чергу, є математичними виразами. Оператор може складатися з символа або декількох символів, і вказує *MathCad*, яку дію виконати над операндами (операндом), що належать до цього оператора.

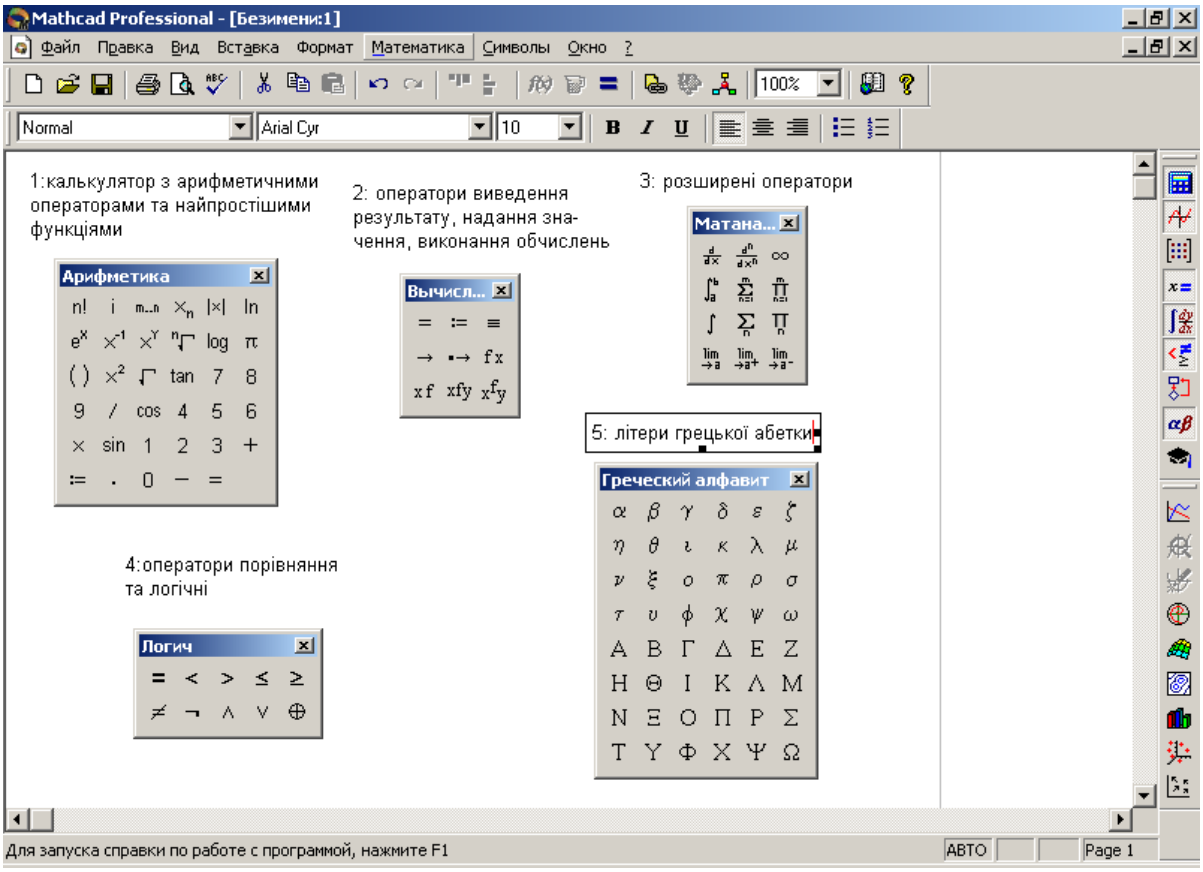


Рис..1. Палітри з панелі математичних символів:
1 – палітра обчислень; 2 – палітра операцій надання значення та виведення; 3 – палітра математичних операцій; 4 – палітра булевих операцій; 5 – палітра грецьких символів

Оператори є арифметичні, розширені, порівняння, логічні, надання значення та виведення в документі *MathCad* результату обчислень.

Звертатися до них зручно, відкривши кнопкою з палітри математичних знаків відповідне вікно. На рис.1 панель палітр математичних знаків розташовано поряд з лінійкою вертикальної прокрутки. Кнопки цієї палітри відкривають вікна з відповідними номерами, наведеними на рисунку.

Найчастіше використовуваний оператор надання значення в документі *MathCad* має вигляд := та вводиться кнопкою з таким же зображенням. Можна використати кнопку або клавішу зі знаком рівності "=", якщо змінна в документі вже набула значення, після введення знаку рівності праворуч від нього буде виведено це значення.

Якщо змінній в певному місці документу *MathCad* надано значення оператором **:=**, то воно буде відомо *MathCad* правіше та нижче цього місця. Якщо ж змінна набула значення за допомогою оператора глобального надання значення ≡ в будь-якому місці документу, то вона має це значення також у будь-якому місці документу.

MathCad розрізняє три типи змінних: скаляр, матриця (або вектор) та рядок. Скаляр (змінна або константа) може бути заданий такими типами даних:

- цілим числом ;
- дійсним числом з мантисою і порядком (наприклад, 2.564×10^{-3});
- двійковим числом, що закінчується літерою *b* (від слова binary);
- вісімковим числом, що закінчується літерою *o* (від слова octal);
- шістнадцятковим числом, що закінчується літерою *h* (від слова hexadecimal); якщо це число починається з літери, то перед нею треба поставити нуль;
- числом з розмірністю;
- комплексним числом.

Комплексне число подається сумою його дійсної та уявної частини, поряд з якою праворуч вказується позначення уявної одиниці *i* або *j*.

Встановити бажане позначення уявної одиниці можна, звернувшись до команди ***Format*** (Формат)=> ***Result*** (Результат). У вікні, що відкривається, вкладка ***Result Format*** (Настройка показа) дає таку можливість командою ***Imaginary Value*** (Мнимые значения) (рис.2).

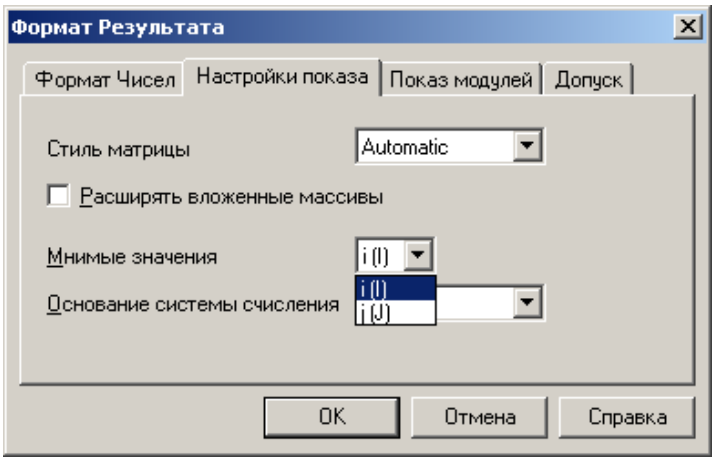


Рис.2. Можливість зміни подання уявної одиниці

Якщо уявна частина комплексного числа дорівнює одиниці, її треба обов'язково вказати перед позначенням уявної одиниці. При цьому в документі *MathCad* таке комплексне число (наприклад, 0,3+1*i*) відобразиться як 0.3+*i*.

Тип "рядок" задається послідовністю символів, обмежених лапками. Для дій над змінними цього типу призначені такі функції:

- ***concat(S1,S2)*** – об'єднує рядкові змінні *S1* та *S2*;
- ***num2str(N)*** – перетворює скалярну змінну *N* (число) у змінну типу "рядок";
- ***str2num(S)*** – перетворює значення рядкової змінної *S* у число;
- ***str2vek(S)*** – повертає вектор, елементами якого є *ASCII*-коди символів, з яких складається змінна *S*;
- ***strlen(S)*** – визначає кількість символів, що входять до складу рядкової змінної *S* та ін.

Формульний редактор синхронно з введенням математичного виразу перевіряє його коректність та наявність значень змінних, ідентифікатори яких

входять до складу виразу, обчислює значення виразу та виводить його значення після введення оператора обчислення результату (знаку рівності).

Якщо функції редактора гальмують формування документа, можна вимкнути їх, знявши прапорець з функції ***Automatic Calculation*** команди ***Math*** (Математика) головного меню.

Якщо в документі вказано ідентифікатор, після якого в круглих дужках записані один або декілька (через кому) ідентифікаторів або виразів, то цей запис *MathCad* сприймає як ідентифікатор функції з одним або декількома аргументами, наприклад:

$$F1(x,a):=a\cdot\sin(a\cdot x) \quad \text{або} \quad F1(\pi,-0.8)=$$

Крім звичайної змінної у *MathCad* існує дискретний аргумент (панель матрици – m..n).

На відміну від звичайної змінної, що в будь-який момент зберігає лише одне значення, дискретний аргумент (ранжована змінна) зберігає сукупність значень та задається так:

$$N:= Nstart, Nnext..Nend$$

або

$$N:= Nstart..Nend.$$

Тут *Nstart* і *Nend* – початкове і кінцеве значення дискретного аргумента, *Nnext* – наступне після *Nstart* значення. Отже, різниця (*Nnext—Nstart*) — це крок, з яким змінюється значення дискретного аргумента в межах між Nstart і Nend; він може бути цілим або дійсним, додатнім або від'ємним.

Якщо використано скорочений формат дискретного аргумента (без заданого значення *Nnext*), то кроком є 1 або -1 залежно від того, чи *Nstart<Nend* або *Nstart>Nend*. Наприклад:

$$X:= 2.5,3.0..10,$$

$$S1:= -2.2,-2.19..-1.$$

Дискретний аргумент зручно застосовувати як аргумент функції для побудови її графіка або визначення ряду її значень, наприклад так, як показано на рис.3.

$a := 2.5 \qquad x := -2, -1.6 .. 2$

$F(x) := a \cdot \sin(x)$

x =	F(x) =
-2	-2.273
-1.6	-2.499
-1.2	-2.33
-0.8	-1.793
-0.4	-0.974
0	$-1.11 \cdot 10^{-15}$
0.4	0.974
0.8	1.793
1.2	2.33
1.6	2.499
2	2.273

Рис.3. Приклади застосування дискретного аргумента

Як видно з рис.3, якщо задані дискретний аргумент та функція, аргументом якої є цей дискретний аргумент, можна отримати у вигляді таблиці їх значення, поставивши після їх ідентифікаторів знаки рівняння.

Іноді виникає задача обрахунку суми (добутку) послідовності чисел або значень функції, тобто, наприклад, на проміжку аргумента x від 1 до 5 обрахувати суму $x^1+x^2+x^3+x^4+x^5$. Для цього треба з палітри *Матанализ* вибрати кнопку виду $\sum_{n=1}^{m}$ (див. рис.1) і задати початкове (у нижньому полі) та кінцеве (у верхньому полі) значення та формулу для обчислення суми у полі після символу підсумовування. Аналогічно задається добуток.

Для вставлення програмного коду в документи в *Mathcad* є спеціальна панель інструментів ***Programming*** (Програмування), яку можна викликати на екран натисненням кнопки *Programming Toolbar* на панелі ***Math*** (Математика), як показано на рис.4. Більшість кнопок цієї панелі має назву операторів програмування.

Викладемо послідовно основні складові елементи мови програмування *Mathcad* і розглянемо приклади її використання.

Програма в *Mathcad* має блокову структуру з підпорядкованими блоками,

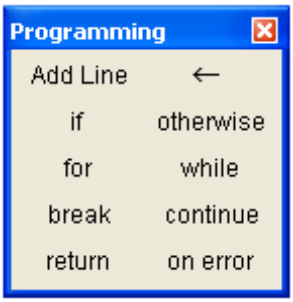


Рис.4 . Панель інструментів *Programming*

що виділені вертикальними лініями. Щоб створити програмний модуль, наприклад, наведений на рис. 6, треба виконати наступні дії.

1) Введіть частину виразу, що знаходитиметься зліва від оператора надання значення і сам оператор надання значення. У нашому прикладі це функція *f(x)*.

2) При необхідності викличте на екран панель інструментів ***Programming*** (**Програмування**) (див. рис. 4).

3) Натисніть на цій панелі кнопку ***Add Line*** (Добавить линию).

4) Якщо наперед відомо, скільки рядків коду міститиме програма, можна створити потрібну їх кількість повторним натисненням кнопки ***Add Line*** (Добавить линию) відповідну кількість разів (на рис. 5 показаний результат триразового натиснення).

5)У полях введення, що з'явилися, введіть бажаний програмний код, використовуючи програмні оператори (див рис. 6, 7).

Після того, як програмний модуль повністю визначений і жодне поле введення не залишиться порожнім, функція *f(x)* може використовуватися звичним способом.

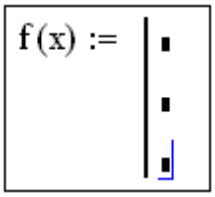


Рис.5. Початок створення програмного модуля

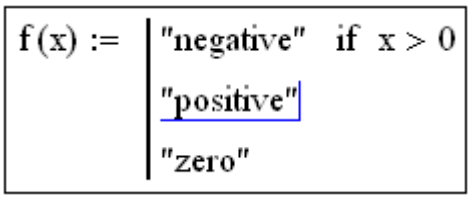


Рис.6. Вставлення програмного оператора

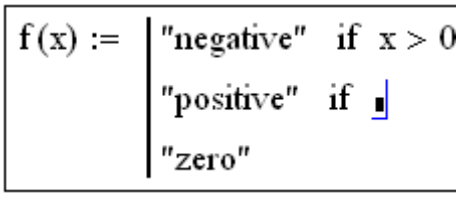


Рис.7. Встлення умови в програму

Вставити рядок програмного коду у вже створену програму можна у будь-який момент за допомогою тієї ж кнопки *Add Line* (*Додати лінію*). Для цього слід заздалегідь помістити на потрібне місце всередині програмного модуля курсор введення. Наприклад, розташування курсора введення на рядку, показаному на рис.8, приведе до появи нової лінії з полем введення перед цим рядком. Якщо пересунути курсор введення з початку рядка (як на рис.8) в його кінець, то нова лінія з'явиться після рядка.

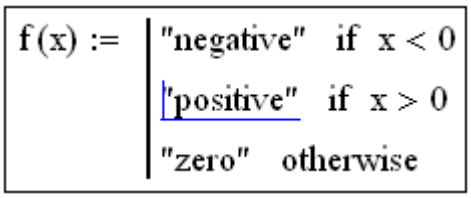


Рис.8. Вставка нового рядка в існуючу програму

Якщо виділити рядок не цілком, а лише деяку його частину (рис.9), то це вплине на місце розташування в програмі нового рядка (результат натиснення кнопки *Add Line* показаний на рис.10).

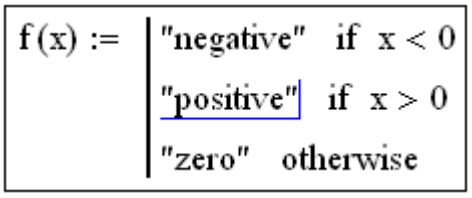


Рис.9. Положення курсора введення впливає на розташування нового рядка

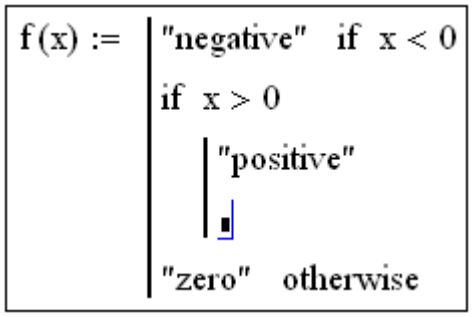


Рис.10. Результат вставлення нової лінії в програму (з положення рис.9)

Результати утворення нових рядків наведені на рис.11.

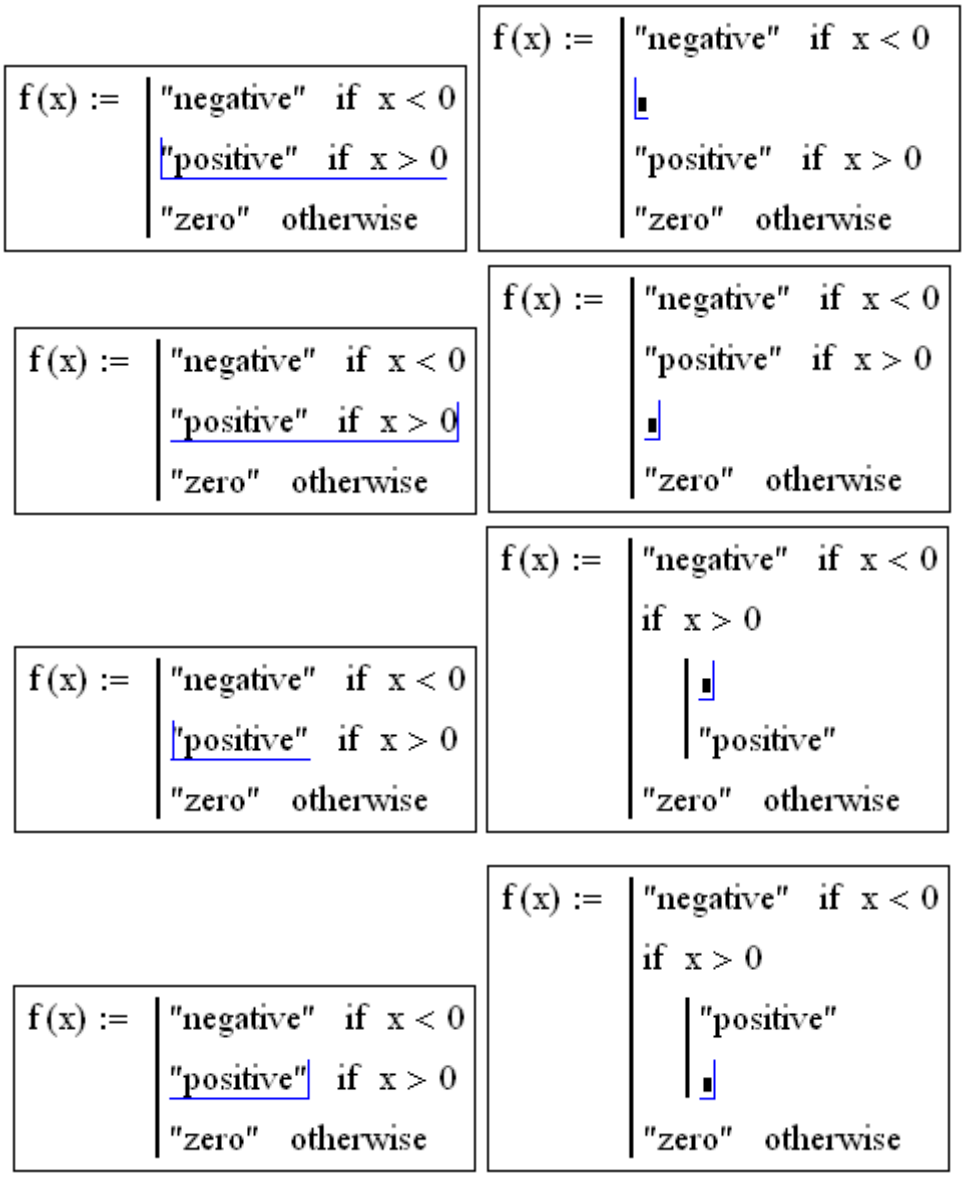


Рис.11. Результат вставлення нового рядкав програму з різних положень курсора (праві фрагменти відтворюють результат операції)

Нова вертикальна межа з двома лініями виділяє фрагмент програми, який

належить до умови $x > 0$, що знаходиться в його заголовку. У режимі виконання програми, а це відбувається при будь-якій спробі обчислити $f(x)$, виконується послідовно кожен рядок коду.

Надання значення в межах програм проводиться за допомогою оператора ***Local Definition*** (Локальное присвоение значения), який вставляється натисненням кнопки із зображенням стрілки ← на панелі ***Programming***(Програмированние) (див. рис. 12).

$$f(x) := \left| \begin{array}{l} z \leftarrow 4 \\ z + x \end{array} \right.$$

$$f(1) = 5$$

Рис. 12. Локальне надання значення в програмі

Дія умовного оператора *if* складається з двох частин. Спочатку перевіряється логічний вираз (умова) праворуч від нього. Якщо він є істиною, то виконується вираз зліва від оператора *if*. Якщо він помилковий – нічого не відбувається, а виконання програми продовжується переходом до її наступного рядка.

У мові програмування Mathcad є два оператори циклу: *for і while*. Перший з них дає можливість організувати цикл за деякою змінною, що приймає ряд значень із заданого діапазону зі сталим кроком. Другий створює цикл, вихід з якого здійснюється за заданою логічною умовою. Щоб вставити в програмний модуль оператор циклу, потрібно виконати дії показані на рис. 13 та 14.

$$f(x) := \left| \begin{array}{l} z \leftarrow 0 \\ \text{for } \blacksquare \in \blacksquare \\ \quad \blacksquare \end{array} \right.$$

Рис.13. Вставка оператора циклу

```
x := | z ← 0                x := | z ← 0
     | for i ∈ 0.. 5             | while z < 10
     |   z ← z + i              |   z ← z + 1
x = 15                     x = 10
```

Рис.14. Робота оператор циклу *for* з ранжованою змінною та *while*

Порядок виконання роботи

1. Надайте значення таким змінним : m=4; n=2; β=0.25.
2. Обчисліть числове значення виразу, номер якого збігається з номером бригади (при формуванні заданого виразу зверніть увагу на розбіжність місця розташування показника степеня у рукописній та *Mathcad*–формі)

1. $\frac{\sin^{m+1}(\beta)}{(n-1)\cos^{n-1}(\beta)} - \frac{m-n+2}{n-1}$;

2. $\left(m-2n-\frac{\beta}{10{,}5}\right)^2 - \sin^2(\beta)$;

3. $\frac{1}{(n-1)\cos^{n-1}(\beta)} - \frac{1}{(n-3)\cos^{n-3}(\beta-0.1)}$;

4. $\frac{\beta}{n}\left[\sin(\beta)+1^{5/3} - m^{-1/4}\right]$;

5. $\frac{1}{\sqrt{\beta}}\left(m^2-\frac{1}{n}\right)+\sin(\beta)-\cos^3(e/\beta)$;

6. $\frac{x^3}{3} - \frac{n\beta}{6}\sqrt{x^2-n^2} - \frac{n^3}{6}(\beta+\sqrt{x^{2n}-n^2})$;

7. $\frac{1}{(m-1)\sin^{m-1}(\beta)} + \frac{2(x-4)}{\sin^{m-2}(\beta^2)\cos(\beta)}$;

8. $\frac{\sqrt{e^m+1}-\cos^2(\beta)}{\sin(\beta)+1/n}$;

9. $\frac{1}{(n-1)\sin^{m-1}(\beta)\cos^{n-1}(\beta/n)} + \frac{m+n-2}{n^3-1}$;

10. $\dfrac{3\sqrt{(4+n^2)^3} - (3m+4)\dfrac{3}{\beta}(n+4)^{1/2}2n}{(n+2n)^3}$;

3. Зробіть змінну β ранжованою (крок та межі брати, як рис.3 для *x*).

4. Задайте функцію п.2 з аргументом β, надайте їй вираз, заданий у п.3, та обчисліть її значення .

5. Надайте трьом різним змінним значення цілого, комплексного числа, десяткового дробу з порядком та перетворіть їх на рядкові змінні.

6. Об'єднайте всі отримані рядкові змінні в одну та визначте кількість символів, з яких вона складається.

8. Обрахуйте суму результатів виразу з п.2 за умови, що змінюється тільки одна змінна(на власний розсуд), а інші є константами.

9. Обрахуйте добуток результатів виразу з п.2 за умови що змінюється тільки одна змінна(на власний розсуд), а інші є константами.

10. Створіть програмний блок, результатом виконання якого значення функції з п.2

11. Розрахуйте значення функції з використанням програмування (номер формули відповідає номеру бригади)за умови, що:

1) $y=x$-sin(x), частина графіка з III-го квадранту дзеркально відобразиться у II-ий квадрант;

2) $y=x+1/x$ (врахуйте наявність розриву функції), частина графіка з III-го квадранту дзеркально відобразиться у II-ий квадрант;

3) $y=x\cdot\sin(x)$, від'ємна частина графіка дзеркально відобразиться відносно осі абсцис;

4) $y=x$-sin(x), від'ємна частина графіка дзеркально відобразиться відносно осі абсцис;

5) $y=x+1/x$ (врахуйте наявність розриву функції), частина графіка з III-го квадранту відобразиться у II-ий, а з I-го у IV-ий;

6) $y=x\cdot\mathrm{tg}(x)$ (врахуйте наявність розриву функції), частина графіка з III-го квадранту відобразиться у II-ий, а з I-го у IV-ий;

7) $y=x\cdot e^{-x}$ від'ємна частина графіка дзеркально відобразиться відносно осі абсцис;

8) $y=e^{x}/x$ (врахуйте наявність розриву функції), частина графіка з I-го квадранту відобразиться у II-ий, а з III-го у IV-ий;

9) $y=e^{-x}\cdot\cos(x)$, від'ємна частина графіка дзеркально відобразиться відносно осі абсцисс;

10) $y=e^{x}\cdot\sin(x)$, частина графіка з I-го квадранту відобразиться у IV-ий.

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з результатами виконання завдань.

Контрольні запитання

1. Який вигляд матиме змінна, значенням якої є двійкове число?
2. Як при формуванні математичного виразу вийти з режиму введення показника степеня, підкореневого виразу?
3. Як задати дискретний аргумент, значення якого матимуть від`ємний крок?
4. Що відбуватиметься з документом Mathcad з вимкненим режимом автоматичних обчислень?
5. Як, задавши функцію, отримати таблицю її значень, якщо вона має :

- один аргумент;
- два аргумента?

6. Поясніть призначення кнопки *Add Line*.
7. Які цикли є у мові програмування Mathcad, чим вони відрізняються від вам відомих?
8. Який формат має умовний оператор у програмному блоці Mathcad ? Наведіть приклад.
9. Що означає локальне надання значення?

Лабораторна робота №7

Символьні обчислення.

Поліноми

Операції з матрицями і векторами.

Мета роботи – навчитися за допомогою *Mathcad* виконувати символьні операції над математичними виразами та дослідити особливості їх застосування, навчитися виконувати дії над поліномами, задавати та виконувати операції над матрицями та векторами

Теоретичні відомості

Розв'язання великої кількості математичних та інженерних задач потребує виконання різного виду аналітичних перетворень, які у ряді випадків значно спрощують подальші викладки.

Виконання символьних (аналітичних) операцій реалізується через команду ***Symbolics*** (Символы) головного меню в командному режимі або за допомогою спеціального оператора символьного виведення результату у вигляді подовженої горизонтальної стрілки ($\rightarrow$), яку можна через панель символьних операцій (кнопка із зображенням вказаної стрілки).

При застосуванні палітри символьних операцій результат перетворення виводиться праворуч від стрілки (приклад 3), а результат аналітичних перетворень, що виконуються у командному режимі, може розташовуватися праворуч, нижче (приклади 1, 2 на рис. 1) та замість початкового виразу. Змінити місце розташування результату можна командою ***Symbolics*** (Символы) => ***Evaluate Style*** (Стиль вычислений).

Існує суттєве обмеження при виконанні аналітичних перетворень в командному режимі: у виразах, що підлягають перетворенню, не повинні міститися функції користувача. При застосуванні оператора символьного виведення результату (через палітру символьних перетворень) такого обмеження не існує.

Команда ***Symbolics*** головного меню містить список команд перетворень, які можна поділити на дві групи за правилами їх застосування.

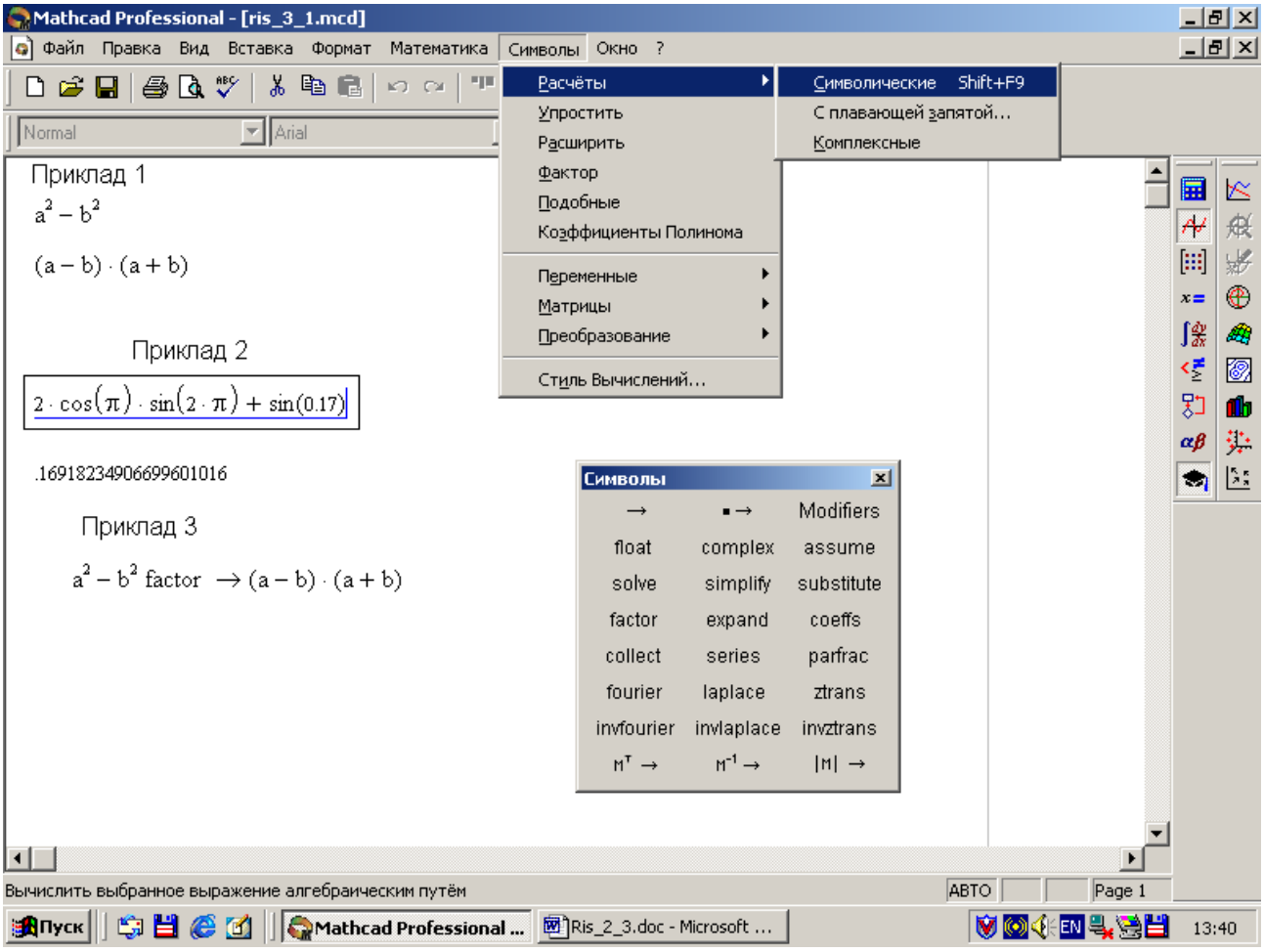


Рис.1. Приклади виконання символьних перетворень різними способами

До першої групи належать такі команди:

- ***Evaluate*** (Расчеты) – перетворення виразу з вибором виду перетворення (приклад 2 на рис.1);
- ***Simplify*** (Упростить) – спрощення виразу зі зведенням подібних, зведенням до спільного знаменника, із застосуванням основних тригонометричних тотожностей і т.і.(рис.2, приклад 1);
- ***Expand*** (Расширить) – розкладання за степенями (приклад 2 на рис.2);
- ***Factor*** (Фактор) – розкладання числа або виразу на множники; протилежна Expand команда (приклад 1 на рис.3.1);
- ***Collect*** (Подобные) – розкладання виразу за підвиразом (приклади 3,4 на рис.2);
- ***Polynomial Coefficients*** (Коэффициенты полинома) – визначення коефіцієнтів поліноміального виразу (приклади 5,6 на рис.2).

Ці команди можна виконати, попередньо виділивши курсором вираз або його частину, що підлягає перетворенню. Винятком є команда пошуку подібних та визначення коефіцієнтів полінома. Перед їх виконанням треба виділити не весь вираз, а ту його частину, що є аргументом.

У прикладах 3 та 4 (рис.2) при зведенні подібних отримані різні результати, оскільки виділеним підвиразом у прикладі 3 був *x,* а у прикладі 4 – *d*.

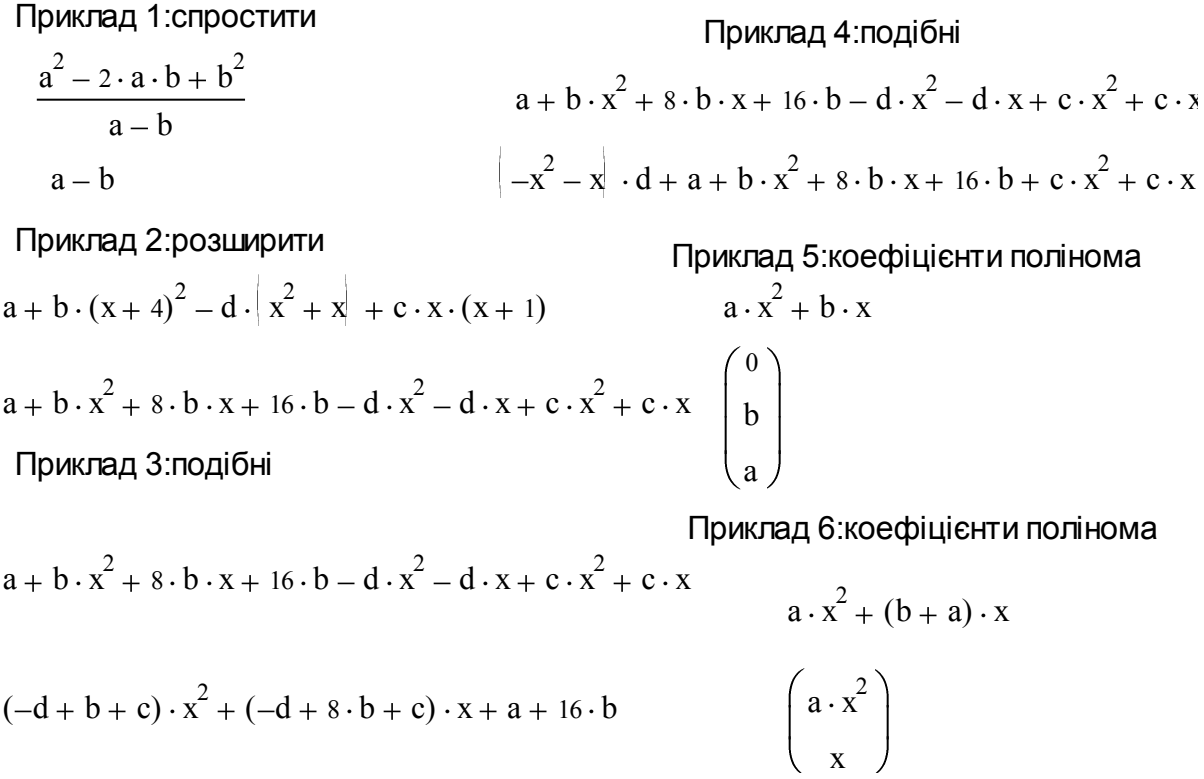


Рис.2. Приклади виконання операцій спрощення, зведення подібних та визначення коефіцієнтів полінома

У прикладах 5, 6 (рис.2) були виділені відповідно аргументи *x* та *a+b*. Як видно з рисунка, результатом команди визначення коефіцієнтів полінома є вектор-стовпець коефіцієнтів, першим елементом якого є вільний коефіцієнт, останнім – старший.

До другої групи команд належать команди, що вимагають виділення змінної у виразі. Це команди ***Variable*** (Переменные), ***Matrix*** (Матрицы) та ***Transform*** (Преобразование). Кожна з них має своє меню.

Меню команди Variable містить такі пункти:

- ***Solve*** (Вычислить) – пошук розв'язку рівняння чи нерівності (приклади 1,2 на рис.3);
- ***Substitude*** (Замена) – виконання заміни змінної у виразі заданим підвиразом (приклад 3 на рис.3);
- ***Differentiate*** (Дифференциал) – диференціювання виразу за виділеною змінною;
- ***Integrate*** (Интегралы) – інтегрування виразу за виділеною змінною;
- ***Expand to Series*** (Разложить на составляющие) – визначення членів розкладання виразу в ряд Тейлора відносно виділеної змінної;
- ***Convert to Partial Fraction*** (Преобразование в Частичные Доли) – розкладання дробово-раціонального виразу на елементарні дроби (приклад 4 на рис.3).

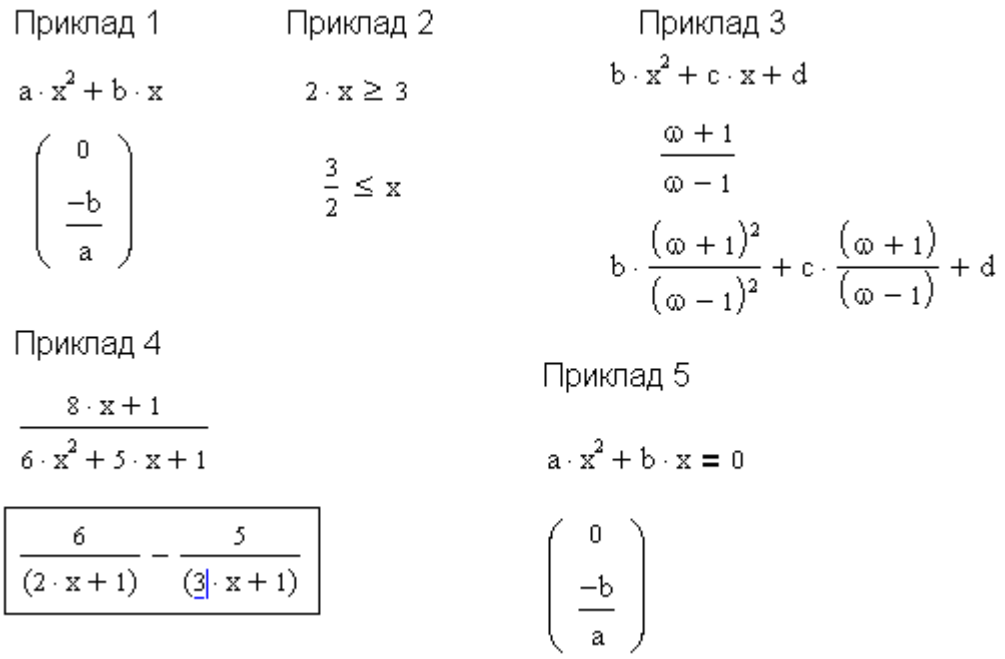


Рис.3. Приклади розв'язання рівнянь і нерівностей, виконання заміни змінної, розкладання на елементарні дроби і визначення коефіцієнтів полінома

Результатом виконання команди ***Solve*** є вектор-стовпець з коренями виразу з виділеною змінною (приклад 1 на рис.3). Для розвя'язання можна задати рівняння у загальноприйнятому вигляді (приклад 5 на рис.3), вказавши знак рівняння з палітри логічних операторів, а не у вигляді виразу.

При підвищенні порядку рівняння вирази з його коренями у символьному вигляді можуть мати громіздкий вигляд. У такому випадку результат у текстовому вигляді розташовується в буфері обміну, звідки користувач може вивести його в робоче вікно для аналізу.

Приклад 3 на рис.3 ілюструє виконання команди заміни ***Substitude***, яка вимагає задання виразу зі змінною, що замінюватиметься, виразу для заміни (у прикладі $(\omega+1)/(\omega-1)$) з його подальшим вирізанням чи копіюванням до буферу обміну та виділення у першому виразі змінної (у прикладі x), що зазнає змін.

Меню команди ***Matrix*** містить такі пункти:

- ***Transpose*** (Транспонировать) – транспонування виділеної матриці або її виділеної частини;
- ***Invert*** (Инвертирование) – обернення квадратної матриці або її виділеної частини;
- ***Determinant*** (Определитель) – визначення детермінанту матриці.

Меню команди ***Transform*** містить команди для прямого та зворотного перетворень Фур'є, Лапласа та z-перетворення.

MathCad надає можливість визначення похідних заданого порядку та інтегралу від функції.

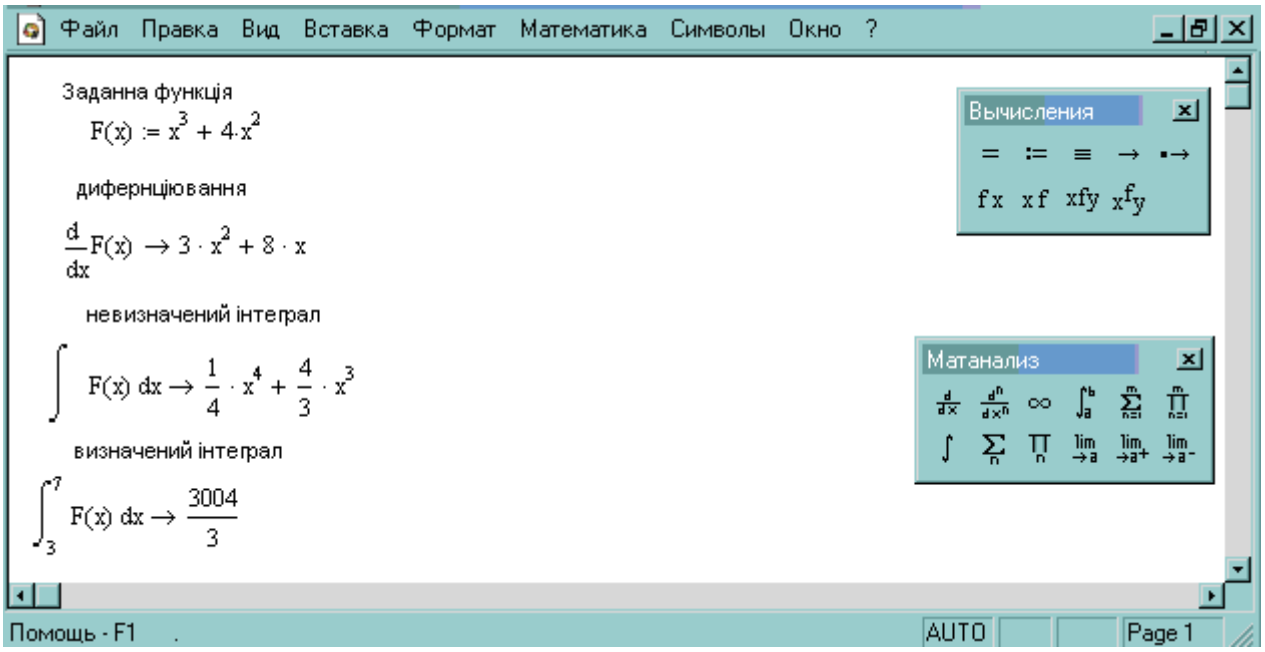


Рис. 4. Обчислення похідної, визначеного і невизначеного інтегралу функції

Похідна береться за змінною, яка записується у нижньому полі, а у верхнє заноситься функція з аргументом, за яким відбувається диференціювання $\frac{dF(x)}{dx}$. Даний об'єкт знаходиться на панелі *Математіз*. Після виразу вводиться оператор обчислення символьного виразу $\longrightarrow$ і натискається *Enter*. Інтегрування здійснюється також з панелі *Математіз*. Інтеграл може бути визначеним та невизначеним (рис. 4) .

Поліноми – дуже зручний інструмент в математиці. З одного боку – що може бути простіше від полінома? З іншого – це досить гнучкий інструмент. Нарощуючи степінь полінома, можна розширювати його можливості чи не до нескінченності. Поліноми можна використовувати для інтерполяції, екстраполяції, згладжування. Характеристичні поліноми, що відповідають диференціальним рівнянням та їх системам, несуть масу цінної інформації про досліджувані системи. Поліном має вид $a_n x^n+a_{n-1}x^{n-1}+\ldots+a_1x^1+a_0$, де a_i – коефіцієнти, x – аргумент полінома, i=0, 1, 2,…, n.

Розглянемо найпростіші операції над поліномами, такі як додавання та віднімання. Задамо два поліноми *A(x)* та *B(x)* третього степеня з рівними коефіцієнтами. Далі поліном *C(x)* буде служити поліномом-результатом, наприклад, *C(x):=A(x)+B(x)*. В наступному рядку напишемо *C(x)* і виберемо команду ***Evaluate Symbolically*** (Символический знак равенства) з панелі ***Evaluation*** (рис. 5). Віднімання та множення поліномів виконується аналогічно.

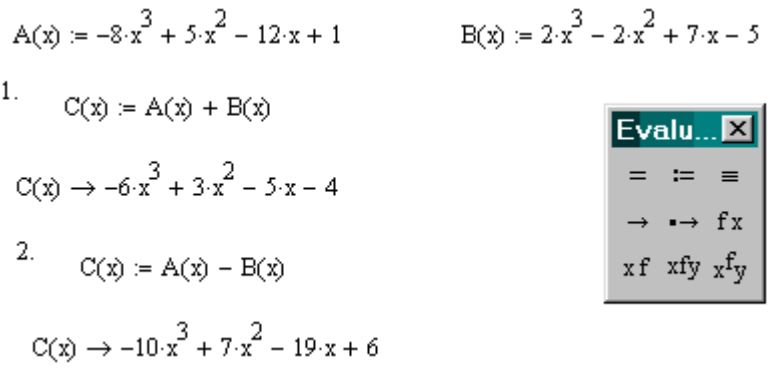


Рис. 5. Додавання та віднімання поліномів

При множенні поліномів необхідно ввести поліноми *A(x)* та *B(x),* далі поліному-добутку *C(x),* задавши вираз *C(x):=A(x)*B(x).* Для виведення

результату на екран використовується команда ***Expand*** (Символическая оценка) з панелі ***Evaluation*** (рис 6).

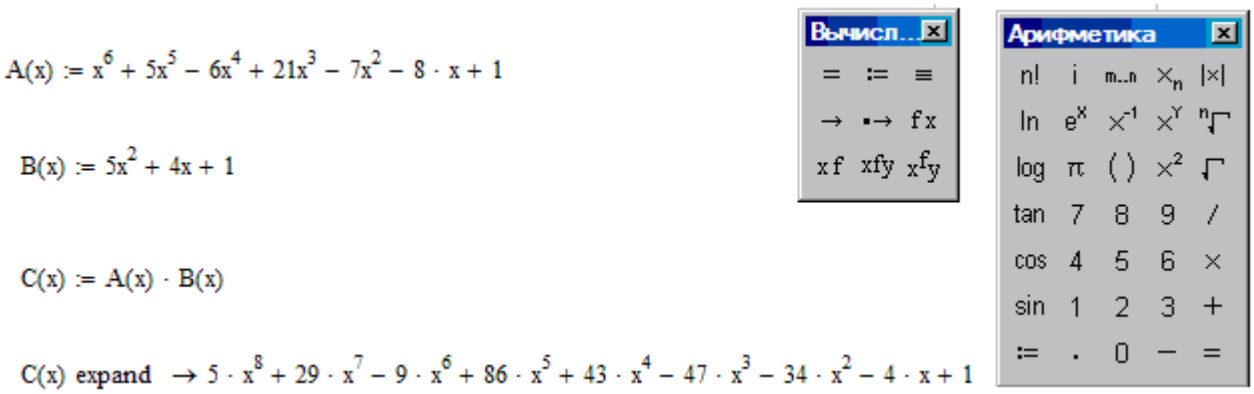


Рис. 6. Множення поліномів

Матриці у вигляді двовимірних масивів та вектори у вигляді одновимірних широко застосовуються при розв'язанні технічних задач.

Матриця чи вектор, як і ранжована змінна, є сукупністю елементів. Елементи ранжованої змінної розташовуються у порядку зростання або зменшення. До того ж різниця між двома сусідніми елементами ранжованої змінної є однаковою у межах заданого її діапазону. Значення елементів матриці чи вектора можуть утворюватись довільно.

Звернення до ранжованої змінної означає послідовне звернення до кожного з її елементів. На відміну від ранжованої змінної до будь-якого елемента матриці або вектора можна звернутися окремо, вказавши номери рядка і стовпця, на перетині яких він знаходиться, наприклад, $M_{z,s}:=2zx+s$, $B_s:=a+2s^n$, де M, B – ідентифікатори матриці та вектора відповідно; i – номер рядка, s – номер стовпця елемента матриці M, та номер елемента вектора-стовпця B.

Якщо в *MathCad* відбувається звернення до елементів матриці або вектора, які до цього моменту не набули значень, вони вважаються рівними нулю.

Нумерація елементів у межах рядка та стовпця починається з нуля. Початковий номер елемента зберігається у вбудованій константі *ORIGIN*. Його можна змінити, надавши константі *ORIGIN* бажане значення, використавши

оператори локального або глобального надання значення (:=, ≡), а також визначити, застосувавши оператор обчислення результату =.

Вектором *MathCad* вважає лише вектор-стовпець, а тому його елементи мають лише один номер – рядка, наприклад, a_s, B_k. Вектор-рядок сприймається *MathCad* як матриця. Тому його елементи, як і елементи матриці мають два номери. Отже, елементом матриці є індексована змінна, індексом якої є номери рядка і стовпця, розділені комою.

Для того, щоб увійти в режим введення індексу, треба натиснути клавішу із зображенням квадратної дужки, що відкривається. Щоб вийти з режиму введення індексу, треба потрібну кількість разів натиснути клавішу пропуску.

Палітра операцій з матрицями відкривається кнопкою із зображенням порожнього шаблону матриці на панелі математичних знаків (рис.7).

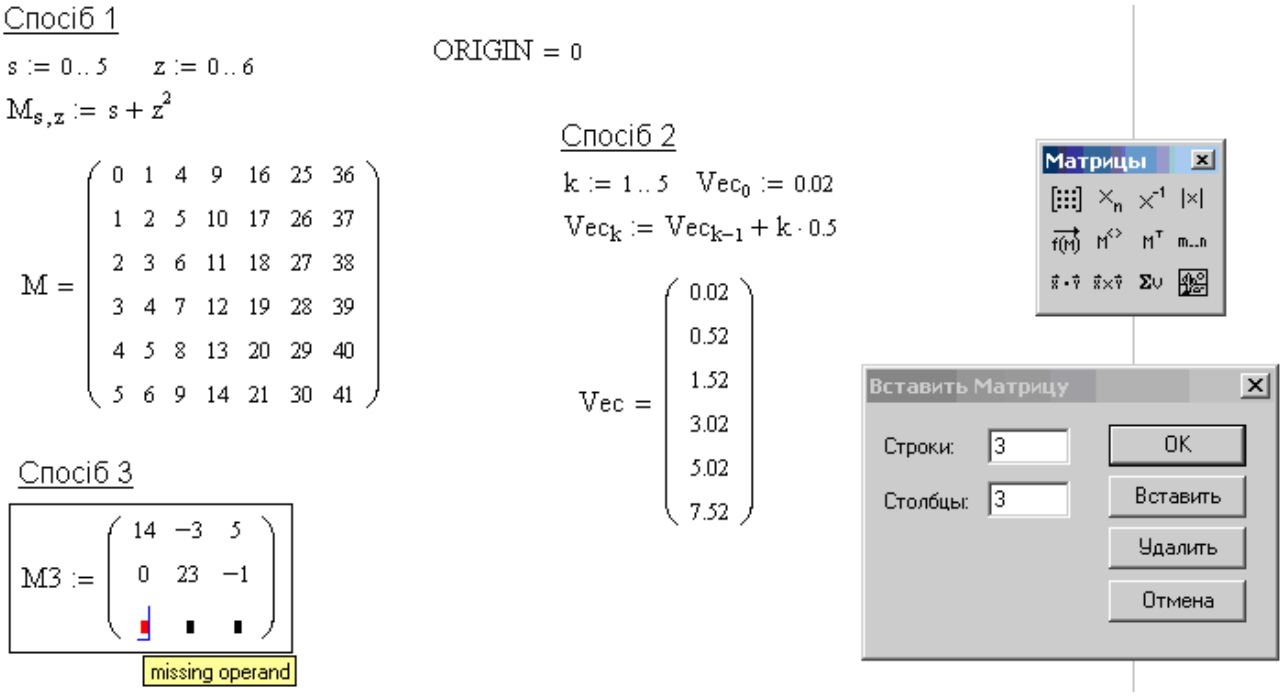


Рис.7. Способи введення матриць

Задавати матрицю або вектор можна різними способами. Якщо значення елемента матриці залежить від номерів його рядка і стовпця, зручно використовувати спосіб 1 (див. рис.7); якщо існує залежність між сусідніми елементами – спосіб 2 (рис.7). Спосіб 3 передбачає заповнення шаблону

матриці вручну. Сформувати шаблон можна в командному режимі командами ***Insert => Matrix*** (Вставка => Матрица) або, звернувшись до палітри математичних операцій.

У вікні, що відкриється (рис.7), треба задати кількість рядків і стовпців шаблону.

Можна надавати окремим елементам значення вручну (вектор *V* на рис.8), решту елементів *MathCad* заповнить нулями.

До матриці можна звернутися в цілому, вказавши її ідентифікатор. До елемента матриці можна звернутись, вказавши ідентифікатор індексованої змінної, тобто ідентифікатор матриці з нижнім числовим індексом, який міститиме номери рядка і стовпця елемента матриці, розділені комою. До стовпця матриці можна звернутися так: вказати ідентифікатор матриці, натиснути клавіші „*Ctrl*” та „^ ” та в полі верхнього індексу вказати номер стовпця.

$s := 0..40 \qquad z := 0..4$

$M2_{s,z} := \sin(s) + z$

$V_0 := 5.5 \qquad V_4 := 12$

$V_8 := 6$

Стиль "Матриця"

$$V = \begin{pmatrix} 5.5 \\ 0 \\ 0 \\ 0 \\ 12 \\ 0 \\ 0 \\ 0 \\ 6 \end{pmatrix}$$

Стиль "Таблиця"

M2 =

	0	1	2	3	4
0	0	1	2	3	4
1	0.84	1.84	2.84	3.84	4.84
2	0.91	1.91	2.91	3.91	4.91
3	0.14	1.14	2.14	3.14	4.14
4	-0.76	0.24	1.24	2.24	3.24
5	-0.96	0.04	1.04	2.04	3.04
6	-0.28	0.72	1.72	2.72	3.72
7	0.66	1.66	2.66	3.66	4.66
8	0.99	1.99	2.99	3.99	4.99
9	0.41	1.41	2.41	3.41	4.41
10	-0.54	0.46	1.46	2.46	3.46
11	-1	$9.79 \cdot 10^{-6}$	1	2	3
12	-0.54	0.46	1.46	2.46	3.46
13	0.42	1.42	2.42	3.42	4.42
14	0.99	1.99	2.99	3.99	4.99
15	0.65	1.65	2.65	3.65	4.65

Рис.8. Способи виведення матриці

Формат виведення матриці можна змінювати командами ***Format*** => ***Result*** (Формат => Результат) (рис.8). Вкладка „*Display Options*” (Настройка

показа) вікна, що відкриється, пропонує різні стилі виведення матриці: ***Matrix*** – матричний, ***Table*** – табличний (рис.8).

Крім операцій з матрицями та векторами, представленими на панелі математичних операцій, *MathCad* підтримує ряд векторних та матричних функцій та функцій, що визначають характеристики матриць.

Серед матричних функцій є:

augment (*M1*, *M2*) - утворює матрицю, лівою частиною якої є матриця *M1*, правою - *M2*; кількість рядків *M1* і *M2* має бути однаковою (об'єднання по горизонталі);

diag (*V*) – утворює діагональну матрицю, елементами головної діагоналі якої є елементи вектора *V*;

identity (*n*) – утворює одиничну квадратну матрицю розміру *n*;

stack(*M1*,*M2*) – утворює матрицю, верхньою частиною якої є матриця *M1*, нижньою *M2*; кількість стовпців *M1* і *M2* має бути однаковою (об'єднання по вертикалі);

submatrix (*M*,*nr*,*mr*,*nl*,*ml*) – утворює з матриці *M* підматрицю з елементів, що знаходяться з *nr* по *mr* рядок та з *nl* по *ml* стовпець (рис.9).

$$M := \begin{pmatrix} 2 & 5 & 6 \\ 23 & 5 & 8 \\ 11 & 0 & 5 \end{pmatrix} \qquad V := \begin{pmatrix} -2 \\ 1 \\ 9 \end{pmatrix}$$

$$M2 := \begin{pmatrix} 2 & 3+2i & 5 \\ 3i & 6 & 8 \end{pmatrix}$$

$$\mathrm{Re}(M2) = \begin{pmatrix} 2 & 3 & 5 \\ 0 & 6 & 8 \end{pmatrix}$$

$$M1 := \mathrm{augment}(M, V)$$

$$\mathrm{Im}(M2) = \begin{pmatrix} 0 & 2 & 0 \\ 3 & 0 & 0 \end{pmatrix}$$

$$M1 = \begin{pmatrix} 2 & 5 & 6 & -2 \\ 23 & 5 & 8 & 1 \\ 11 & 0 & 5 & 9 \end{pmatrix}$$

$$|M| = -415$$

$$\mathrm{identity}(4) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathrm{length}(V) = 3$$

Рис.9. Приклади застосування матричних функцій

Серед функцій, що визначають характеристики матриць, є:

cols (*M*) **-** повертає кількість стовпців матриці *M*;

rows (*M*) **–** повертає кількість рядків матриці *M*;

rank (*M*) **-** повертає ранг матриці *M*.

Послідовність виконання роботи

1. Обчисліть в символьному вигляді вираз:

1) $2\cos(\pi)\sin(2\pi)+\sin(0.17)$;

2) $2\sin(a)\sin(b)$;

3) $\frac{\sin(\pi)}{\cos(\pi/4)}+\sin^2(0.2)+\cos(\pi/2)$;

4) $\sqrt{1-\cos^2(a)}$;

5) $\sin(\pi)+\cos(\pi/4)\sin(\pi/4)\cos(\pi/2)$;

6) $\sqrt{1-\sin^2(a)}$;

7) $5\cos(3\pi/2)+\frac{\sin(2\pi)}{e^{0.6\pi}}+\cos^2(0.2)$;

8) $2\sin\left(\frac{a+b}{2}\right)\sin\left(\frac{b-a}{2}\right)$;

9) $e^{-0.25\pi}\sin\pi+\cos 1.2$;

10) $\cos(a)\cos(b)+\sin(a)\sin(b)$.

2. Спростіть вираз:

1) $\frac{\sin^2(x)+\cos^2(x)}{2e^{-x}}$;

2) $\frac{\sqrt{\sin(x)}}{\sin^2(x)+2\sin(x)+1}$;

3) $\frac{25\cdot(x+1)+30\cdot(x-1)-0.15}{(x+2)^2}$;

4) $\frac{a^2-x^2}{a+x}-a+x$;

5) $\frac{\sin^2(x)+\cos^2(x)}{1-x^2}-\frac{x}{x-x^3}$;

6) $\left(1+\frac{2}{3x-1}\right)\left(1-\frac{9x-9x^2}{3x+1}\right)+1$;

7) $\frac{9x^2-4}{2(x+1)+x\cos^2(x)+\sin^2(x)}$;

8) $\frac{(x+1)(x-1)-x^2}{\sin(\frac{x}{2})}$;

9) $\frac{x^2+2x+1}{(x+1)^2}-\frac{\sin(\pi)}{x}$;

10) $\frac{2e^{-x}}{\sin^2(x)+\cos^2(x)}$.

3. Розкладіть за степенями аргумента *x* вираз:

$(-a^3+2a^2x-x^3)(4a^2+8ax)+(a^3x^2+2a^2x^3-4ax^4)-(a^5+4a^3x^4-3a^2x^3-4ax^4)$.

4. Визначте коефіцієнти полінома, отриманого у п.3.

5. Задайте поліном 5-го порядку та знайдіть його корені (коефіцієнти a_5=номер бригади х 5, a_4=номер бригади х 4 і т.д.)

6. Обрахуйте диференціал функції з п.2 .

7. Обрахуйте визначений та невизначаний інтеграли функції з п.2 (для визначеного інтегралу - у межах від 3 до 8).

8. Задайте два полінома 5-го степеня з числовими коефіцієнтами (коефіцієнти a_5=номер бригади х 5, a_4=номер бригади х 4 і т.д.)

9. Знайдіть третій поліном, що буде їх сумою, різницею, добутком.

10. Сформуйте вектор-стовпець, елементами якого є квадрати номерів цих елементів (кількість елементів вектора n=5).

11. Задайте матрицю розміру 3х3, кожний елемент якої є сумою номерів його рядка і стовпця. Обчисліть визначник цієї матриці.

12. Визначте, який номер має перший елемент головної діагоналі матриці.

13. Транспонуйте матрицю з пункта 11.

14. Сформуйте діагональну матрицю із сформованого в п.10 вектора.

15. Змініть нумерацію елементів матриць і векторів документу.

16. Задайте матрицю розміру 3×3 в символьному вигляді, та обчисліть її визначник.

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з результатами виконання завдань.

Контрольні запитання

1. Яким може бути результат символьних перетворень?
2. Чи може бути результатом символьних перетворень текстовий вираз?
3. Розкладіть заданий викладачем дробово-раціональний вираз на елементарні дроби.
4. Які з наведених виразів є коректними для визначення коефіцієнтів полінома:

$$A(p) := p^2 + (p + 4)^3 + 3;$$

$$P^2 + (p + 4)^3 + 3;$$

$$P^2 + (p + 4)^3 + 3 = 0;$$

$$2p^6 + 3p^3 \ ?$$

5. У якому вигляді треба задати рівняння, щоб отримати його розв'язок? Наведіть приклади.
6. Що називають поліномом?
7. Опишіть алгоритм додавання поліномів, якщо один з них 3-го степеня, а другий 6-го?
8. Чи можна помножити поліноми, коли в одного ненульові коефіцієнти тільки при парних степенях аргумента, а в іншого – при непарних?
9. Як видалити рядок (рядки) з матриці або її шаблону?
10. Як додати в матриці або її шаблоні рядки (стовпці)?
11. Як змінити нумерацію елементів матриці?
12. Як об'єднати матриці по горизонталі, вертикалі?
13. Як обчислити ранг матриці вручну та за допомогою Mathcad?
14. Як у символьному вигляді знайти обернену матрицю?
15. Як змінити стиль виведення матриці?

Лабораторна робота №8

Двовимірна та тривимірна графіка

Мета роботи – навчитися будувати графіки функцій однієї та декількох змінних, будувати поверхні функцій, дослідити можливості їх форматування.

Теоретичні відомості

Графічний процесор *MathCad* забезпечує можливість користування дво- та тривимірного графікою за допомогою відповідних шаблонів. Відкрити список шаблонів можна командою ***Insert*** => ***Graph*** (Вставка =>График) або кнопкою із зображенням двовимірного графіка на панелі палітр математичних знаків (рис. 1).

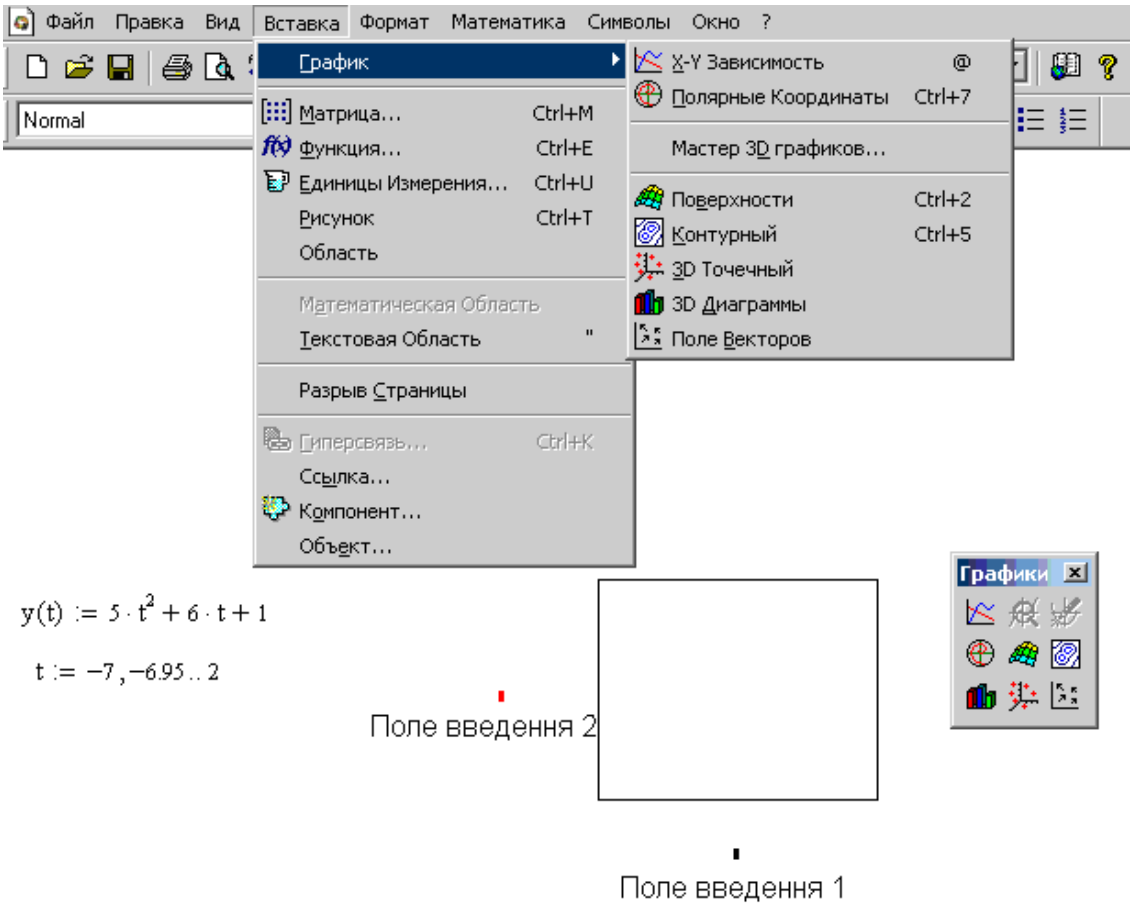


Рис.1. Побудова графіка функції однієї змінної у декартових координатах

Шаблон для побудови графіка в декартових координатах містить поле для побудови графіка та два поля введення. У полі введення 1 (рис.1) треба вказати аргумент функції, графік якої будуватиметься, у полі введення 2— ідентифікатор функції зі своїм аргументом.

Якщо шаблон повинен містити декілька графіків, ідентифікатори усіх функцій зі своїми аргументами вказують списком у полі введення 2 через кому. Якщо ці функції мають різні аргументи, то їх список вказують у полі введення 1 (рис.2). Поля введення 1 та 2 можуть містити ідентифікатори змінної чи функції, вираз (наприклад, nT) або індексовану змінну (наприклад, ya_n на рис.2.).

$T := 0.4 \qquad m := 100 \qquad n := 0..m$

$p(t) := 166.67 \cdot \exp(-.2 \cdot t) - .4 \cdot \exp(-1. \cdot t)$

$Y(t) := 8. + 42.67 \cdot \exp(-.5 \cdot t) - 111.1 \cdot \exp(-.4 \cdot t) + 227.56 \cdot \exp(-.25 \cdot t) - p(t)$

$Y1(t) := Y(t) + (rnd(1.4) - 0.7)$

$ya_n := 10.172 - 11.818 \cdot \exp\left[-\left(5.032 \cdot 10^{-2}\right) \cdot n \cdot T\right]$

$t := 0..40$

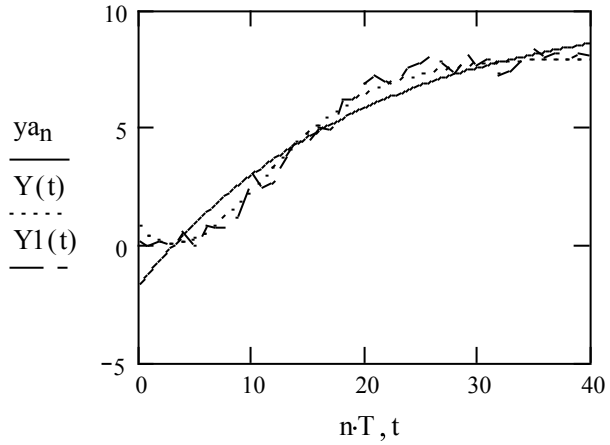


Рис.2. Способи побудови серій графіків

Якщо користувач не вказав діапазон і крок зміни аргумента, за мовчазною угодою він вважається заданим дискретним аргументом виду : -10..10.

Графік утворюється сукупністю точок, кількість яких дорівнює кількості заданих значень аргументу, що з'єднуються між собою лініями різного типу (рис.2). Зменшення кроку зміни аргумента робить графік плавнішим. До того ж, зменшення кроку аргумета може виявити на графіку особливості функції.

Якщо *MathCad* працює в автоматичному режимі обчислень, обидва поля введення коректно заповнені, то в шаблоні побудується графік після виведення курсора миші за межі графічної області та натискання на її ліву кнопку.

Для побудови годографів у декартових координатах застосовують функції, задані параметрично, а саме рівняннями виду

$$x = \varphi(\omega); \quad y = \psi(\omega),$$

де $\varphi(\omega)$, $\psi(\omega)$ є неперервними при $\omega \in (\alpha, \beta)$. Ці рівняння, що описують залежність декартових координат (*x,y*) точки площини від значення параметра $\omega \in (\alpha, \beta)$, визначають на площині криву, задану в параметричній формі (задану параметрично).

Приклади зображення таких кривих наведені на рис.3.

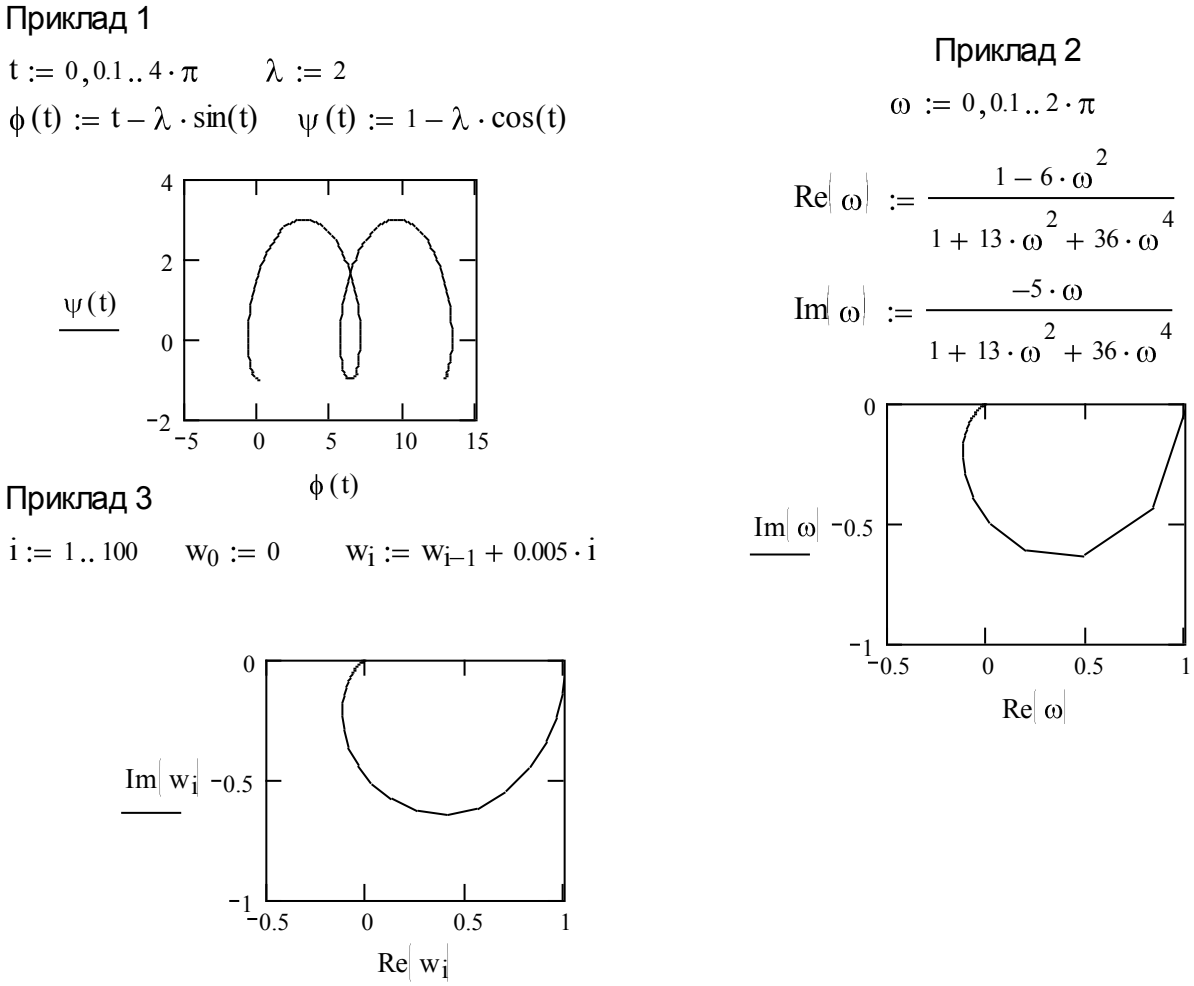


Рис.3. Приклади зображення кривих, заданих в параметричній формі

Якщо зображення годографа будується у вигляді ламаної лінії (рис.3, приклад 2), можна зменшити крок зміни параметра (у прикладі - ω), заданого дискретним аргументом, або задати параметр як індексовану змінну (рис.4.3, приклад 3), що змінюється з кроком, який, наприклад, зростає. Так зручно

задавати параметр при побудові годографа, якщо цей параметр (ω у прикладі 3 на рис.3) змінюється нерівномірно.

При побудові годографів для подальшого їх аналізу важливо, щоб масштаби вздовж обох осей були однаковими.

MathCad пропонує широкі можливості форматування графіків. До команд форматування можна звернутися, виділивши попередньо графічну область і натиснувши праву клавішу миші або через команди ***Format*** =>***Graph*** (Формат => График). Розгорнуте після цього вікно форматування міститиме вкладки форматування осей, ліній графіка, відображення написів поблизу осей та вкладку для роботи з параметрами, що встановлюються автоматично (рис.4).

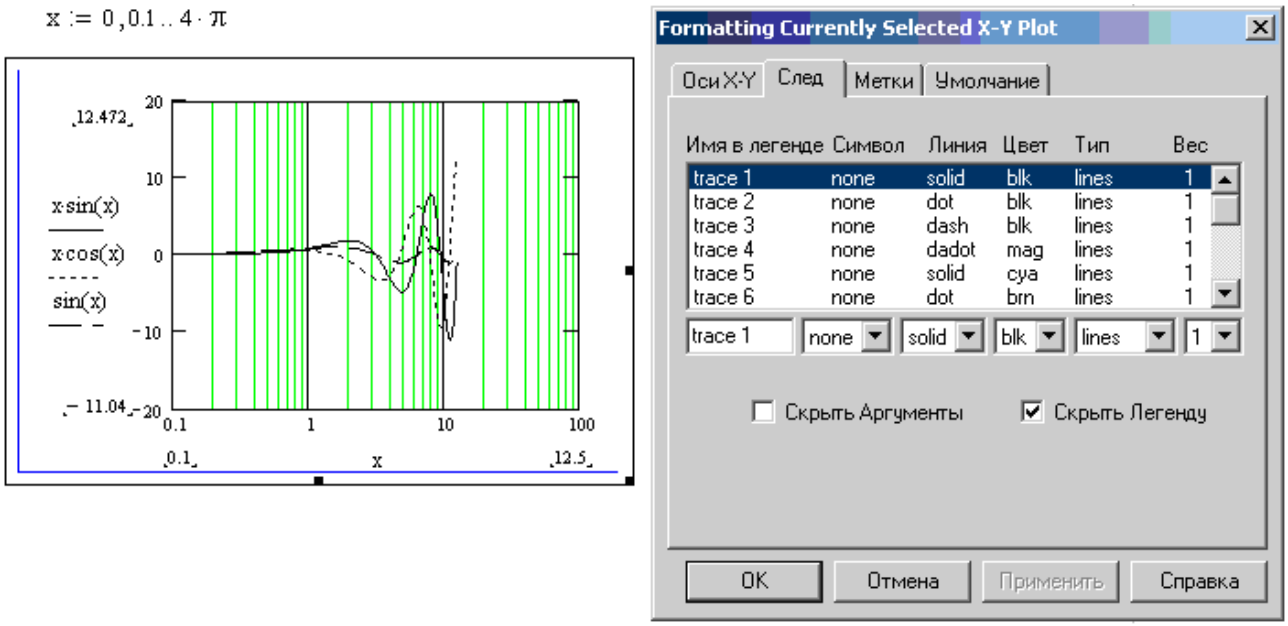


Рис.4. Ілюстрація можливостей форматування декартового графіка

MathCad пропонує два способи побудови поверхні функції двох змінних *F(x, y)*.

Перший спосіб (рис.5) передбачає попереднє формування матриці, елементами якої є значення функції *F*.

$$F(x,y) := \sin(x \cdot y) \cdot \exp\left[-\left(x^2 + y^2\right)\right]$$

$$N := 25 \quad s := 0..N \quad z := 0..N$$

$$x_s := -1 + 0.1 \cdot s \qquad y_z := -1 + 0.1 \cdot z$$

$$M_{s,z} := F\left(x_s, y_z\right)$$

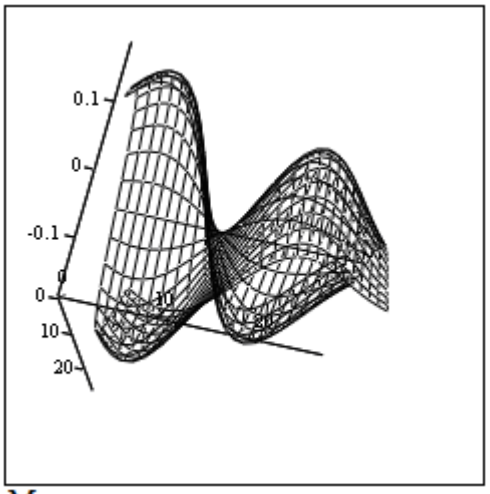


Рис.5. Ілюстрація першого способу побудови поверхні функції двох змінних

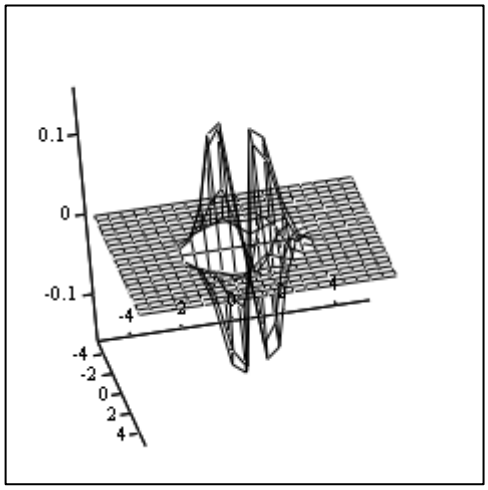
Перший спосіб реалізує алгоритм, застосований на рис.5, а саме:

- задається функція двох змінних (*F(x, y)*);
- задається кількість значень кожної змінної (*N*);
- задаються обидві змінні як ранжовані (*s*, *z*);
- формується набір значень аргументів функцій, заданих індексованими змінними (x_s, y_z);
- формуються елементи матриці $M_{s,z}$, які є значеннями функції;
- з палітри графіки обирається шаблон тривимірного графіка;
- у полі введення шаблону вводиться ідентифікатор *M* сформованої матриці.

Другий спосіб потребує для побудови задання лише самої функції, став можливим, починаючи з версії *MathCad* 2000, реалізує такий алгоритм (див. рис.6):

- задається функція двох змінних (*F(x, y)*);
- обирається шаблон тривимірного графіка;
- у полі введення шаблону вводиться ідентифікатор змінної (*F*).

$$F(x,y) := \sin(x \cdot y) \cdot \exp\left(-\left|x^2 + y^2\right|\right)$$



F

Рис.6. Ілюстрація другого способу побудови поверхні функції двох змінних

Результатом застосування обох алгоритмів буде графік поверхні (рельєфу) функції двох змінних у вигляді прозорого дротяного каркасу.

При необхідності зображення поверхні можна повертати навколо осей, розташувавши курсор миші в межах графічної області, натиснувши її ліву кнопку і, не відпускаючи, рухаючи її в бажаному напрямку.

Якщо під час руху миші буде натиснута клавіша Ctrl, зображення масштабуватиметься.

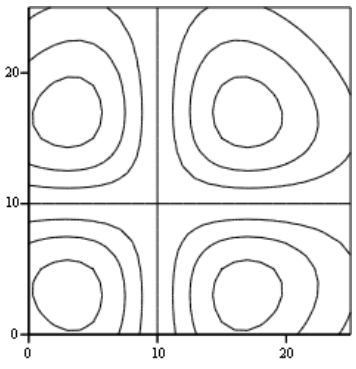
Палітра графіки пропонує можливість подання поверхні функції двох змінних у вигляді ліній однакового рівня (ізоліній). Її, як і поверхню, можна будувати одним із вищенаведених способів, а в полі ведення шаблону вказати відповідний ідентифікатор (рис.7).

$$F(x,y) := \sin(x \cdot y) \cdot \exp\left[-\left(x^2 + y^2\right)\right]$$

$$N := 25 \quad s := 0..N \quad z := 0..N$$

$$x_s := -1 + 0.1 \cdot s \qquad y_z := -1 + 0.1 \cdot z$$

$$M_{s,z} := F\left(x_s, y_z\right)$$



M

Рис.7. Поверхня функції двох змінних, побудована ізолініями

Широкі можливості форматування поверхні, побудованої першим способом, стають доступними після того, як, розташувавши курсор в межах

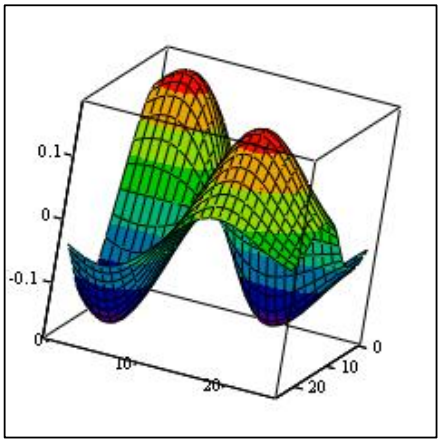
графічної області, два рази натиснути на ліву кнопку миші (рис.8). Застосування кольорів, підсвічування робить рельєф інформативнішим (порівняйте з рис.6).

$$F(x,y) := \sin(x \cdot y) \cdot \exp\left[-\left(x^2 + y^2\right)\right]$$

$$N := 25 \quad s := 0..N \quad z := 0..N$$

$$x_s := -1 + 0.1 \cdot s \qquad y_z := -1 + 0.1 \cdot z$$

$$M_{s,z} := F\left(x_s, y_z\right)$$



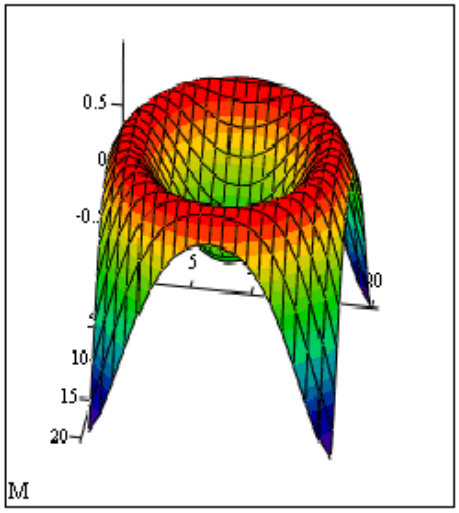
M

$$N := 20$$

$$i := 0..N \qquad j := 0..N$$

$$x_i := -1.5 + 0.15 \cdot i \qquad y_j := -1.5 + 0.15 \cdot j$$

$$f(x,y) := \sin\left(x^2 + y^2\right) \qquad M_{i,j} := f\left(x_i, y_j\right)$$



$$G(a,b) := \begin{bmatrix} (5 + 2 \cdot \cos(a)) \cdot \cos(b) \\ (5 + 2 \cdot \cos(a)) \cdot \sin(b) \\ 2 \cdot \sin(a) \end{bmatrix}$$

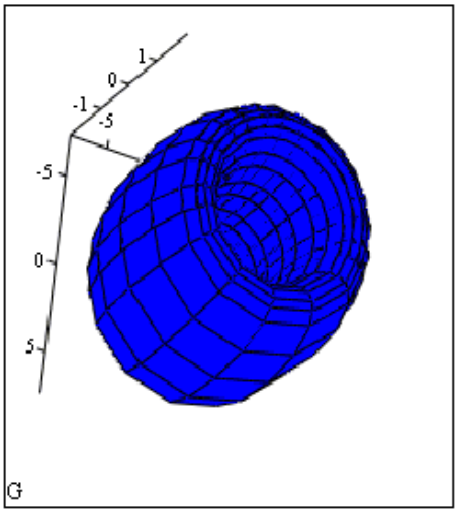


Рис.8. Ілюстрація можливостей форматування тривимірних зображень

Також засобами *MathCad* існує можливість будувати фігури стереометрії, приклади побудови наведені на рис. 9.

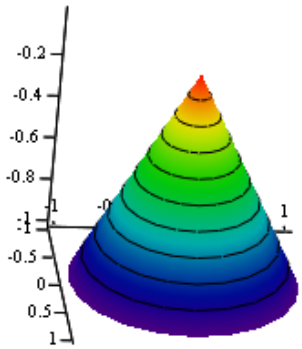
$s := 0..20 \qquad n := 0..20$

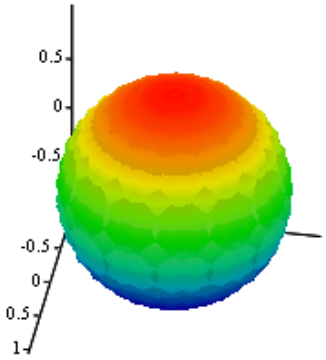
$$X_{s,n} := \frac{s \cdot \cos\left(\frac{\pi \cdot n}{10}\right)}{20}$$

$$Y_{s,n} := \frac{s \cdot \sin\left(\frac{\pi \cdot n}{10}\right)}{20}$$

$$Z_{s,n} := \frac{-s}{20}$$

$$F(x,y) := \begin{pmatrix} \cos(x) \cdot \cos(y) \\ \cos(x) \cdot \sin(y) \\ \sin(x) \end{pmatrix}$$





(X,Y,Z) F

Рис.9. Ілюстрація можливостей побудови фігур стереометрії

Послідовність виконання роботи

1. Задайте поліноми другого та п’ятого степеня з коренями відомими наперед.
2. Побудуйте в одних координатних осях графіки цих поліномів.
3. Оформіть графік написами і коментарями, створеними як частина графічної області і як текстовий блок.
4. Перегляньте у збільшеному масштабі та виконайте трасування графіка.
5. Введіть параметрично задану функцію та побудуйте її годограф:

1	Завиток Паскаля	$x = a \cdot \cos^2(\omega) + I \cdot \cos(\omega);$ $y = a \cdot \sin(\omega) \cdot \cos(\omega) + I \cdot \sin(\omega);$ $I = 10; \quad a = 50; \quad \omega = 0..2\pi$
2	Епіциклоїду	$x = (A+a) \cdot \cos(\omega) - a \cdot \cos\left(\frac{A+a}{a}\omega\right);$ $y = (A+a) \cdot \sin(\omega) - a \cdot \sin\left(\frac{A+a}{a}\omega\right);$ $A = 50; \quad a = 10; \quad \omega = 0..2\pi$

3	Гіпоциклоїду	$x = (A+a)\cdot\cos(\omega) - \lambda\cdot a\cdot\cos\left(\frac{A+a}{a}\omega\right);$ $y = (A+a)\cdot\sin(\omega) - \lambda\cdot a\cdot\sin\left(\frac{A+a}{a}\omega\right);$ $A = 50;\quad a = 10;\quad \lambda = 15;\quad \omega = 0..2\pi$
4	Кардіоїду	$x = a\cdot\cos(\varphi)\ 1+\cos(\varphi)\ ;$ $y = a\cdot\sin(\varphi)\ 1+\cos(\varphi)\ ;$ $a = 10;\quad \varphi = 0..2\pi$
5	Криву Штейнера	$x = 2\cdot a\cdot\cos\left(\frac{\omega}{3}\right) + a\cdot\cos\left(\frac{2\omega}{3}\right);$ $y = 2\cdot a\cdot\sin\left(\frac{\omega}{3}\right) - a\cdot\sin\left(\frac{2\omega}{3}\right);$ $a = 10;\quad \omega = 0..2\pi$
6	Декартів лист	$x = 3a\omega/(1+\omega^3);\ \ y = 3a\omega^2/(1+\omega^3);$ $\omega = -1..100;\ \ a = 2$
7	Циссоїду	$x = \frac{a}{\omega^2+1};\ y = \frac{a}{\omega\ \ \omega^2+1};$ $a = 5;\ \ \omega = -100..100;$
8	Строфоїду	$x = a(\omega^2-1)/(\omega^2+1);$ $y = a\omega(\omega^2-1)/(\omega^2+1);$ $a = 0.5;\quad \omega = -100..100$
9	Звичайну циклоїду	$x = a(\omega - \sin(\omega);\ \ y = a(1-\cos(\omega));$ $a = 10;\quad \omega = 0..30$
10	Астроїда	$x = a\cos^3(t);\ \ y = a\sin^3(\omega);$ $a = 10;\quad t = -20..20$

6. Побудуйте в одній системі координат серію годографів п.5 різними співвідношеннями коефіцієнту a.
7. Побудувати приклади поверхонь приведених на рис. 5, 6, 8, 9

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з результатами виконання завдань.

Контрольні запитання

1. Як звузити на побудованому графіку діапазон аргументу?
2. Як встановити однакові масштаби на обох осях декартового графіка?
3. Як зробити видимими (невидимими) коментарі на осях?
4. Як утворити графік сукупністю значень функції у вузлах у вигляді символів обраної форми?
5. Як нанести сітку на графік?
6. Які існують особливості форматування поверхонь?
7. Як, використовуючи команди форматування, перебудувати поверхню функції двох змінних у її зображення лініями однакового рівня?
8. Як за графічним зображенням функції двох змінних визначити її нулі?
9. Як з матриці, яку *Mathcad* використовує для побудови поверхні, отримати сукупність значень функції, що відображувала б вплив на неї лише однієї змінної?

Лабораторна робота №9

Розв'язання систем рівнянь та диференціальних рівнянь

Мета роботи – дослідити способи розв'язання систем лінійних та не лінійних алгебраїчних рівнянь, поняття та роботу з диференціальними рівняннями.

Теоретичні відомості

Розглянемо спочатку можливості, які пропонує *MathCad* для пошуку розв'язку системи лінійних алгебричних рівнянь.

Нехай задана система n алгебричних рівнянь виду

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

і треба знайти її розв'язок. Для цього треба сформувати матрицю коефіцієнтів системи

$$A = \begin{bmatrix} a_{11} & a_{12} & ... & a_{1n} \\ a_{21} & a_{22} & ... & a_{2n} \\ a_{n1} & a_{n2} & ... & a_{nn} \end{bmatrix}$$

і вектор-стовпець $b = b_1 b_2 ... b_n^T$ її правих частин. Вектор розв'язків x шукають як розв'язок матричного рівняння $Ax=b$ так (приклад 1 на рис.1):

$$A := \begin{pmatrix} 2 & 1 & -8 \\ -5 & 4 & 1 \\ -1 & -1 & 3 \end{pmatrix} \quad b := \begin{pmatrix} 3 \\ -14 \\ -1 \end{pmatrix}$$

Приклад 1

Спосіб 1

$$x := A^{-1} \cdot b$$

$$x = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}$$

Спосіб 2

$$A^{-1} \cdot b = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}$$

Приклад 2

$$\text{lsolve}(A, b) = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}$$

Рис.1. Приклади розв'язання систем лінійних алгебричних рівнянь

Для пошуку розв'язку системи лінійних рівнянь призначена також вбудована функція ***lsolve***(*A*,*b*), що повертає вектор розв'язків (приклад 2 на рис.1). Якщо розв'язок системи використовуватиметься в подальших розрахунках, його треба запам'ятати, використавши ідентифікатор вектора.

Для символьного розв'язання систем лінійних рівнянь можна застосовувати метод, що використовує обернену матрицю коефіцієнтів системи (аналогічно способу 2 на рис.1), але замість оператора обчислення результату треба використати оператор $\rightarrow$ з палітри символьних перетворень.

$$M := \begin{pmatrix} a & b \\ -b & c \end{pmatrix} \quad V := \begin{pmatrix} -d \\ c \end{pmatrix}$$

$$M^{-1} \cdot \begin{pmatrix} -d \\ c \end{pmatrix} \rightarrow \begin{bmatrix} \frac{-c}{(a \cdot c + b^2)} \cdot d - \frac{b}{(a \cdot c + b^2)} \cdot c \\ \frac{-b}{(a \cdot c + b^2)} \cdot d + \frac{a}{(a \cdot c + b^2)} \cdot c \end{bmatrix}$$

Рис.2. Символьне розв'язання систем лінійних рівнянь

Для пошуку точного розв'язку системи нелінійних рівнянь в *MathCad* існує вбудована функція ***Find*** (*x1,x2,…,xn*), результатом виконання якої є *n* невідомих системи (для системи порядку *n*) – її розв'язок.

Застосування цієї функції вимагає створення в документі спеціального обчислювального блоку, що відкривається службовим словом *Given* та має таку структуру:

початкові умови

Given

рівняння

обмежувальні умови

вираз з функцією *Find*.

Блок такої структури дозволяє шукати розв'язок одного нелінійного рівняння або системи.

Початкові умови задають початкові значення шуканих змінних. В рівняннях системи, що задаються, замість звичайного знаку рівності використовується "жирний" знак рівності з палітри булевих операторів.

Обмежувальні умови зазвичай задаються рівностями або нерівностями, яким повинен задовольняти розв'язок.

Система рівнянь, задана на рис.3, має два набори розв'язку, оскільки пряма, описана другим рівнянням системи, перетинає параболу, що описана першим рівнянням, у двох точках. Кожний набір розв'язків шукається окремо зі своїми початковими і обмежувальними умовами.

$x := 1 \quad y := 0$

Given

$3 \cdot x \cdot y + 2 \cdot x^3 = 0 \qquad 5 \cdot x - y = 11$

$x > 0$

$\begin{pmatrix} x0 \\ y0 \end{pmatrix} := \text{Find}(x,y) \qquad \begin{pmatrix} x0 \\ y0 \end{pmatrix} = \begin{pmatrix} 1.778 \\ -2.108 \end{pmatrix}$

Given

$3 \cdot x \cdot y + 2 \cdot x^3 = 0 \qquad 5 \cdot x - y = 11$

$x < 0$

$\begin{pmatrix} x1 \\ y1 \end{pmatrix} := \text{Find}(x,y) \qquad \begin{pmatrix} x1 \\ y1 \end{pmatrix} = \begin{pmatrix} 0 \\ -11 \end{pmatrix}$

Рис.3. Приклади розв'язання системи нелінійних рівнянь

Для символьного розв'язання систем нелінійних рівнянь використовується обчислювальний блок зі словом *Given*, але без початкових і обмежувальних умов, та зі знаком $\rightarrow$ замість = після ідентифікатора функції пошуку розв'язку *Find*.

Приклад символьного розв'язання рівняння наведений на рис.4.

Given

$x \cdot y = a \qquad b \cdot x + c \cdot y = b$

$$\text{Find}(x,y) \rightarrow \begin{bmatrix} \dfrac{\left[\frac{1}{2} \cdot b - \frac{1}{2} \cdot \left(b^2 - 4 \cdot c \cdot a \cdot b\right)^{\left(\frac{1}{2}\right)}\right]}{b} & \dfrac{\left[\frac{1}{2} \cdot b + \frac{1}{2} \cdot \left(b^2 - 4 \cdot c \cdot a \cdot b\right)^{\left(\frac{1}{2}\right)}\right]}{b} \\ \frac{1}{(2 \cdot c)} \cdot \left[b + \left(b^2 - 4 \cdot c \cdot a \cdot b\right)^{\left(\frac{1}{2}\right)}\right] & \frac{1}{(2 \cdot c)} \cdot \left[b - \left(b^2 - 4 \cdot c \cdot a \cdot b\right)^{\left(\frac{1}{2}\right)}\right] \end{bmatrix}$$

Рис.4. Символьне розв'язання системи нелінійних рівнянь

Можливості *MathCad* забезпечують лише числове розв'язання звичайних диференційних рівнянь та їх систем.

Усі функції *MathCad* призначені для розв'язання задачі Коші. Функція

rkfixed (y, t0, tn, n, D)

повертає матрицю розв'язків, яку формує із застосуванням методу Рунге-Кутта.

У векторі *y* мають бути задані початкові умови, у змінних *t0* та *tn* – початкове та кінцеве значення незалежної змінної, у змінній *n* – кількість кроків на інтервалі (*t0, tn*), у символьному векторі-функції *D* – праві частини рівнянь системи у формі Коші.

Нехай потрібно знайти розв'язок диференційного рівняння першого порядку виду $5y'(t)+y(t)=2x(t)$

з початковою умовою $y(0)=0$ за умови, що $x(t)=1(t)$.

Подамо рівняння у формі Коші, підставивши замість загального позначення $x(t)$

його конкретне значення: $y'(t)=\frac{1}{5}(2\cdot 1(t)-y(t))$.

Для заданого рівняння вектор початкових умов матиме лише один елемент і його можна задати напряму (див.рис.5). Функція *D* набуде значення правої частини рівняння у формі Коші.

Матриця розв'язків (*RY* у прикладі на рис.5), яку повертає функція *rkfixed*, має таку структуру: перший стовпець містить *n* значень незалежної змінної, починаючи з *t0*, другий стовпець – відповідні значення розв'язку *y*; для диференційного рівняння *к*-го порядку наступні стовпці містять значення y', $y'',...,y^{(k)}$.

При розв'язанні рівняння першого порядку будуть заповнені перші три стовпці матриці-результату.

Побудувати графік розв'язку можна, обравши будь-який варіант, наведений на рис.5.

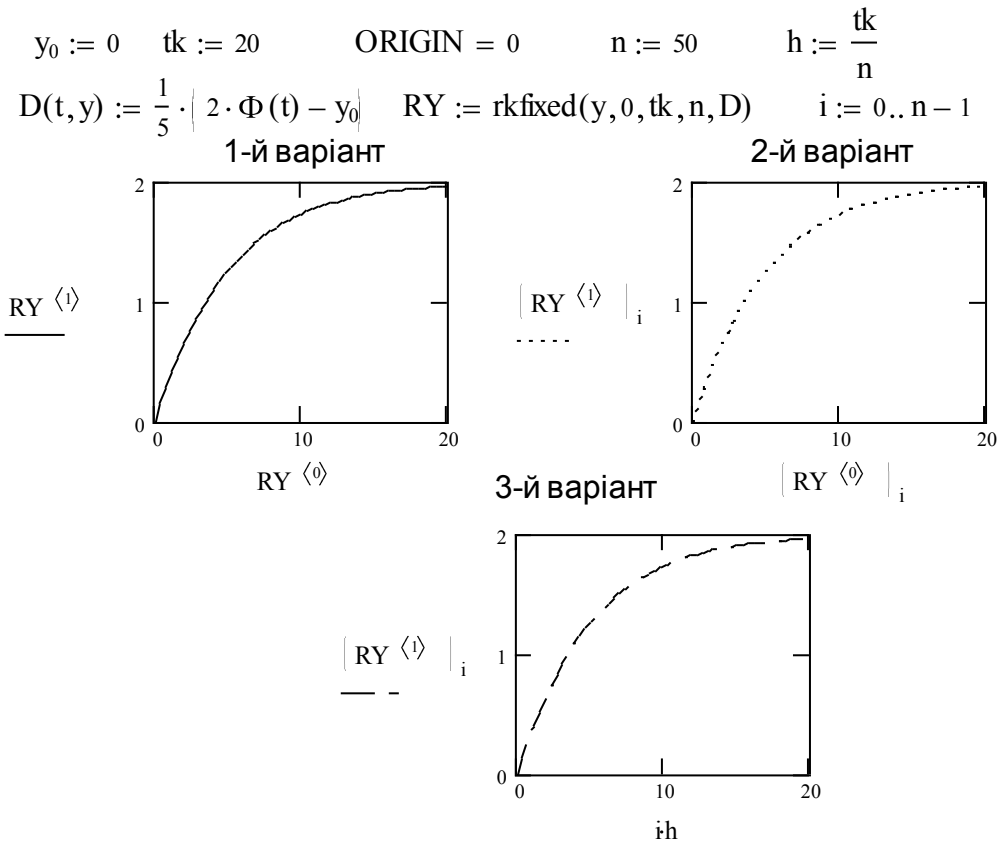


Рис.5. Варіанти побудови графіків розв'язку диференційного рівняння

У прикладі $\Phi(t)=1(t)$ – функція Хевісайда.

Нехай треба розв'язати диференційне рівняння третього порядку виду

$$ay'''(t)+by''(t)+cy'(t)+dy(t)=r$$

з початковими умовами $y(0)=1;\quad y'(0)=0;\quad y''(0)=0.$

Введемо заміну: $y(t)=x_0;\quad y'(t)=x_1;\quad y''(t)=x_2.$

Тоді початкові умови набудуть виду $x_0=1;\ x_1=0;\ x_2=0,$

а початкове рівняння замінить система рівнянь у формі Коші:

$$\begin{cases}\dfrac{dx_0}{dt}=x_1,\\ \dfrac{dx_1}{dt}=x_2,\\ \dfrac{dx_2}{dt}=\dfrac{1}{a_3}(-a_0-a_1x_1-a_2x_2-c).\end{cases}$$

Приклад розв'язання цього рівняння наведений на рис.6.

$$x := \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \qquad n := 50 \quad a := 1 \quad b := 6 \quad c := 11 \quad d := 6 \quad r := 2$$

$$D(t,x) := \begin{bmatrix} x_1 \\ x_2 \\ \frac{1}{a} \cdot \left| -d \cdot x_0 - c \cdot x_1 - b \cdot x_2 + r \right| \end{bmatrix} \qquad Z := rkfixed(x, 0, 15, n, D)$$

$$i := 0 .. n-1$$

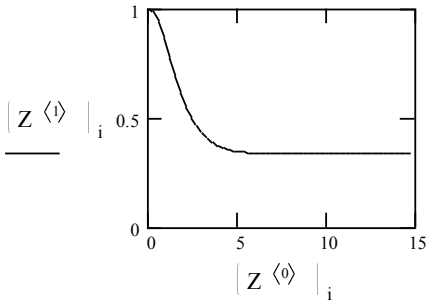


Рис.6. Приклад розв'язання диференційного рівняння третього порядку

Для розв'язання диференційних рівнянь, записаних у *MathCad* у звичному вигляді, призначена функція *odesolve(t,tn,n),* що використовується в обчислювальному блоці *Given.* Тут *t* – незалежна змінна рівняння, *tn* – кінцеве значення незалежної змінної, *n* – кількість кроків пошуку розв'язку на інтервалі *(0, tn)*. Третій параметр функції *odesolve* може бути відсутнім.

Приклад розв'язання вже розглянутого рівняння третього порядку за допомогою функції *odesolve* наведений на рис.7.

Given

$$\frac{d^3}{dt^3}y(t) + 6 \cdot \frac{d^2}{dt^2}y(t) + 11 \cdot \frac{d}{dt}y(t) + 6 \cdot y(t) = 2 \cdot \Phi(t)$$

$$y(0) = 1 \qquad y'(0) = 0 \qquad y''(0) = 0$$

$$y := odesolve(t, 15, 100) \qquad t := 0, 0.15 .. 15$$

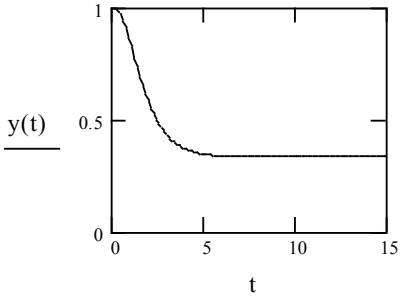


Рис.7. Приклад застосування блоку ***Given*** для розв'язання диференційного рівняння

Послідовність виконання роботи

1. За наперед відомим розв'язком сформуйте систему чотирьох лінійних рівнянь.
2. Розв'яжіть систему рівнянь різними способами в числовому вигляді
3. Перевірте вірність отриманного розв'язку графічно.
4. Розв'яжіть систему рівнянь у символьному вигляді.
5. Розв'яжіть заданну систему рівнянь, застосувавши функцію Find :

1) $\begin{cases} x+y=5, \\ x-2y=14; \end{cases}$ 2) $\begin{cases} -2x+y=7, \\ x-3y=-1; \end{cases}$ 3) $\begin{cases} -2x-2y=-10, \\ 3x-6y=42; \end{cases}$

4) $\begin{cases} -4x+2y=14, \\ x-3y=-1; \end{cases}$ 5) $\begin{cases} 4x+5y=17, \\ x-2y=1; \end{cases}$ 6) $\begin{cases} -2x-5y=4, \\ 3x+y=7; \end{cases}$

7) $\begin{cases} -8x+y=12, \\ 9x-5y=2; \end{cases}$ 8) $\begin{cases} x-7y=7, \\ 5x-y=1; \end{cases}$ 9) $\begin{cases} 8x+10y=-34, \\ -x+2y=1; \end{cases}$

10) $\begin{cases} 2x-14y=14; \\ -5x+y=-1. \end{cases}$

6. Перевірте графічно отриманий у п.5 розв'язок.
7. Знайдіть коефіцієнти функції $F(x)=a_2\cdot x^2+a_1\cdot x+a_0$, якщо відомі такі її значення : F(3)=5, F(-1)=0, F(-2)=3.5.
8. Перевірте графічно отриманий розв'язок.
9. За наперед відомим розв'язком сформуйте систему трьох нелінійних рівнянь з двома невідомими.
10. Розв'яжіть систему рівнянь в числовому вигляді за допомогою блоку *Given*.
11. Перевірте вірність отриманного розв'язку системи рівнянь з пп.9 графічно.
12. Задайте диференційне рівняння першого порядку $Ty'(t)+y(t)=k$ з нульовими початковими умовами та довільними числовими значеннями коефіцієнтів T та κ.
13. Знайдіть розв'язок рівняння і побудуйте графік розв'язку.

14. Задайте диференційне рівняння $y''(t) + w^2 y(t) = 0$ з початковими умовами $y(0)=1, y'(0)=0;$ w – номер бригади.

15. Розв'яжіть диференційне рівняння 6y``(t)+5y`(t)+y(t)=2·1(t) з нульовими початковими умовами різними способами.

16. Побудуйте графік функції Хевісайда.

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з результатами виконання завдань.

Контрольні запитання

1. Які способи розв'язання систем рівнянь передбачають задання матриць їх коефіцієнтів?
2. В якому вигляді може подавати Mathcad символьний розв'язок системи рівнянь?
3. Чи завжди система рівнянь має розв'язок?
4. Чи може система рівнянь мати декілька розв'язків?
5. Яке рівняння називають нелінійним? Наведіть приклади.
6. Наведіть структуру обчислювального блоку *Given***.** Поясніть призначення його складових.
7. Чи можливо отримати графічний розв'язок системи нелінійних рівнянь? У чому його суть?
8. Чи може система нелінійних рівнянь не мати жодного розв'язку? Наведіть приклади.
9. Що таке задача Коші, який вигляд має диференційне рівняння *n* – го порядку, записане у формі Коші?
10. Які функції Mathcad призначені для розв'язання диференційних рівнянь?

Лабораторна робота № 10

Інтерполяційний поліном

Мета роботи: Навчитись застосовувати інтерполяційний поліном, Дослідити апроксимацію полінома методом найменших квадратів. Розглянути схему Хорнера.

Теоретичні відомості

Розглядається випадок, коли деяка функціональна залежність $y = f(x)$ задається рядом точок на площині $X0Y$ (нумерація точок загальному випадку може бути довільною), тобто маємо множину пар значень Xs, Ys для $0 \le S \le m$.

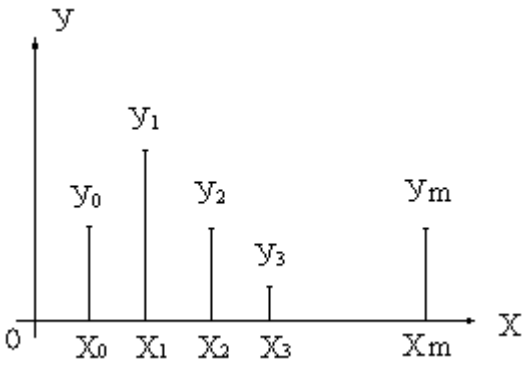


Рис. 1. Вузли інтерполяції

Задача **інтерполяції** буде полягати в тому, щоб знайти якийсь спосіб знаходження значень функції у проміжках між вузлами (вузли – це набір значень Xs, $0 \le S \le$ m). Мається на увазі, що "справжня" функція $f(x)$ невідома за замовчанням, приймається, що апроксимуюча функція має бути неперервною

Значення апроксимуючої функції у вузлах інтерполяції, має співпадати з відповідним значенням Ys, $0 \le S \le m$. Одним з найпопулярніших інструментів для розв'язання задач наближення (апроксимації) функції є поліном виду:

$$A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n.$$

Степінь полінома n та значення його коєфіціентів a_0, a_1, a_2,…, a_n підбирається (визначається) дослідником так ,щоб задовольнити необхідні умови. В розглянутому випадку треба забезпечити проходження графіка

полінома через задані точки (*Xs*,*Ys*) , $0 \le S \le m$ на площині *X*0*Y* (значення полінома у вузлах інтерполяції мають співпадати з відповідним значення *Ys* , $0 \le S \le m$) тобто:

$$\begin{cases} a_0 + a_1x_0 + a_2x_0^2 + \ldots + a_nx_0^n = y_0, \\ a_0 + a_1x_1 + a_2x_1^2 + \ldots + a_nx_1^n = y_1, \\ a_0 + a_1x_2 + a_2x_2^2 + \ldots + a_nx_2^n = y_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_0 + a_1x_m + a_2x_m^2 + \ldots + a_nx_m^n = y_m, \end{cases}$$

Виписана система умов може розглядатись як система рівнянь для визначення невідомих коефіцієнтів полінома a_s, $0 \le S \le m$. Щоб ця система мала єдиний розв'язок необхідно, щоб кількість рівнянь дорівнювала кількості невідомих в них, тобто щоб *m*+1=*n*+1 або ,що рівноцінно *n* = *m*.

Отже , степінь інтерполяційного полінома має (принаймні,формально) дорівнювати m. При n = m система рівнянь (лінійних-відносно невідомих коефіцієнтів-алгебраїчних) матиме розширену матрицю коефіцієнтів

	s=1	2	3	…	n+1	n+2
z=1	1	X0	$X0^2$	…	$X0^n$	y0
2	1	X1	$X1^2$	…	$X1^n$	y1
3	1	X2	$X2^2$	…	$X2^n$	y2
…						
n+1	1	Xn	Xn^2	…	Xn^n	yn
	a0	a1	a2	…	an	праві частини

Під матрицею А показано значення невідомих, коефіцієнтами при яких є елементи матриці у відповідному стовпчику.

Задача по знаходженню інтерполяційного полінома може бути реалізована на *Pascal*:

```
procedure intpol(x,y:coefr;var a:coef);
 var z,s,n:integer;
 d:matr;
   begin
    n:=round(x[-1]);
```

```
  for z:=1 to n+1 do
    begin
      d[z,1]:=1; d[z,n+2]:=y[z-1];
      for s:=2 to n+1 do
      d[z,s]:=d[z,s-1]*x[z-1]
    end;
  systur (n+1,D,A);
  if A[-1]>0 then
    begin
     a[-1]:=n;
     for s:=0 to n do
     a[s]:=a[s+1]
    end;
end;
```

Використовуючи засоби *MathCad* процес інтерполяції виглядає як показано на рис.2. В математичному пакеті є своя вбудована функція для процесу інтерполяції: *linterp*(*X*,*Y*,*t*), де X – вектор-значення аргумента, Y – вектор-значення, t – значення аргумента при яких обраховується функція.

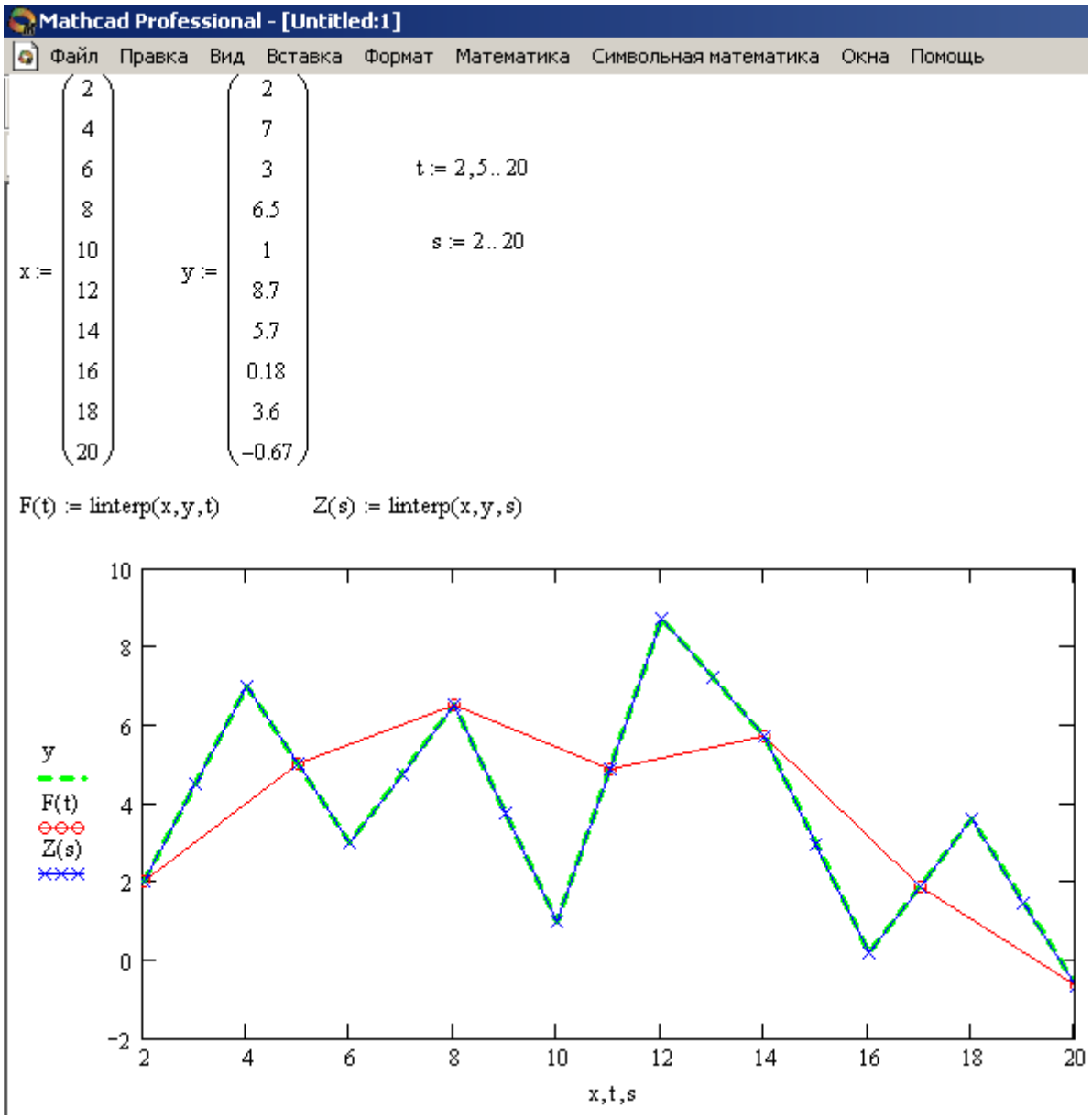


Рис. 2. Інтерполяція функції

Метод найменших квадратів

Точки $(x_i,\ y_i)$, $0 \le i \le m$, що задають досліджувану функцію $y=f(x)$, можуть бути відомі з похибками, або ж апроксимуючу функцію бажано було б мати по можливості більш простою ніж інтерполяційна. В такому випадку замість умови проходження графіка апроксимуючої функції через усі задані точки ставиться більш “м’яка” умова, а саме, щоб показник якості апроксимації

$$E = \sum_{i=0}^{m} \left(a_0 + a_1 x^1 + a_2 x_i^{\ 2} + \ldots + a_n x_i^{\ n} - y_i \right)^2 \to \min.$$

При такій постановці задачі n і m можуть задаватись довільно. Чим більше n тим, можна сподіватись, буде кращою якість апроксимації (менше значення E). Чим менше n – тим більш згладженою буде апроксимуюча функція (при n=0 – це буде константа, при n=1 – пряма лінія, при n=2 – квадратна парабола і т.д.). Умови мінімізації E

$$\partial E/\partial a_r = 0, \qquad 0 \le r \le n$$

або в розгорнутому вигляді

$$2\sum_{i=0}^{m} (a_0 + a_1 x_i^1 + a_2 x_i^2 + ... + a_n x_i^n - y_i) x_i^r = 0, \quad 0 \le r \le n.$$

Якщо в останній формулі виконати скорочення на 2, то розширена матриця коефіцієнтів отримана в системі рівнянь набуває вигляду (z=r+1)

	s=1	2	3	…	n+1	n+2
z=1	$\sum_{i=0}^{m} x_i^0$	$\sum_{i=0}^{m} x_i^1$	$\sum_{i=0}^{m} x_i^2$	…	$\sum_{i=0}^{m} x_i^n$	$\sum_{i=0}^{m} y_i x_i^0$
2	$\sum_{i=0}^{m} x_i^1$	$\sum_{i=0}^{m} x_i^2$	$\sum_{i=0}^{m} x_i^3$	…	$\sum_{i=0}^{m} x_i^{n+1}$	$\sum_{i=0}^{m} y_i x_i^1$
3	$\sum_{i=0}^{m} x_i^2$	$\sum_{i=0}^{m} x_i^3$	$\sum_{i=0}^{m} x_i^n$	…	$\sum_{i=0}^{m} x_i^{n+2}$	$\sum_{i=0}^{m} y_i x_i^2$
…	…					…
n+1	$\sum_{i=0}^{m} x_i^n$	$\sum_{i=0}^{m} x_i^{n+1}$	$\sum_{i=0}^{m} x_i^{n+2}$	…	$\sum_{i=0}^{m} x_i^{2n}$	$\sum_{i=0}^{m} y_i x_i^n$

D

a_0	a_1	a_2	…	a_n	праві частини

Аналіз структури матриці D показує, що при її формуванні, доцільно спочатку формувати масив B, C такої структури

	-1	0	1	2	3	…	m	
B		X_0^R	X_1^R	X_2^R	X_3^R	…	X_m^R	

де R послідовно приймає значення 0, 1, 2, … , 2n

-1	0	1	2	3	…	2n

C

	$\sum_{i=0}^{m} x_i^0$	$\sum_{i=0}^{m} x_i^1$	$\sum_{i=0}^{m} x_i^2$	$\sum_{i=0}^{m} x_i^3$	…	$\sum_{i=0}^{m} x_i^{2n}$	

Формування масиву згладжуючого полінома можна формувати за допомогою процедури

```
    procedure NaimKV(X,Y:Coefr; n:integer; var E:real; var A:Coef);
 var m,z,s: integer;
    B,C: Coefr;
    D: Matr;
    Sum: real;
 begin
  m:=round(X[-1]);
  for s:= 0 to m do
   B[s]:=1; if X[s]=0 then B[s]:=0
   for z:=0 to 2*n do
    begin
     C[z]:=0; Sum:=0;
     for s:=0 to m do
      begin
       C[z]:=C[z]+B[s];
       if z<= n then
       Sum:= Sum+B[s]*Y[s];
       B[s]:=B[s]*X[s]
      end;
     if z<= n then D[z+1,n+2]:=Sum
    end;
   for z:=1 to n+1 do
    for s:=1 to n+1 do
     D[z,s]:=C[z+s-2];
   Systur(n+1,D,A);
   A[-1]:=n;E:=0;
```

```
  for s:=0 to n do
  A[s]:=A[s+1];
  for s:=0 to L do
  E:=E+sqr(horreal(A,X[s])-Y[s])
end;
```

Тут *X,Y*– масиви значень *x* та y для заданих точок згладжуваної функції. *n* – заданий степінь апроксимуючого (згладжуючого) полінома. *E* – значення показника якості апроксимації, що відповідає заданому *n*, *A* – масив з інформацією про згладжуючий поліном.

Використовуючи засоби *MathCad* метод найменших квадраті представлено на рис.3 та рис 4. В математичному пакеті присутній ряд функцій необхідний для розв'язку задачі методу найменших квадратів. Спочатку необхідно мати значення вузлів апроксимації у вигляді *X* – вектор-значення аргумента, *Y* – вектор-значення, *D*(*t*), *D*2(*t*)– апроксимуючі функції, що відповідає степіню апроксимуючого полінома.

Використовуються функції лінійної регресії *intercept* та *slope*, яким передається масив вузлів апроксимації по віссі абцис та ординат.

Функція *intercept* повертає скалярне значення просунення по віссі координат лінії регресії, а функція *slope* – повертає скалярне значення нахилу лінії регресії. Фактично формується рівнянн прямої *y=kx+b*.

Для апроксимації поліномом вищого порядку використовується функція *regress*, що повертає вектор для функції *interp*, необхідно передати масиви вузлів апроксимації та порядок апроксимуючого полінома.

Interp(*v*,*x*,*y*,*t*) – повертає інтерпольоване значення, *v= regress*(*x*,*y*,*n*), *x*,*y* – масиви вузлів апроксимації.

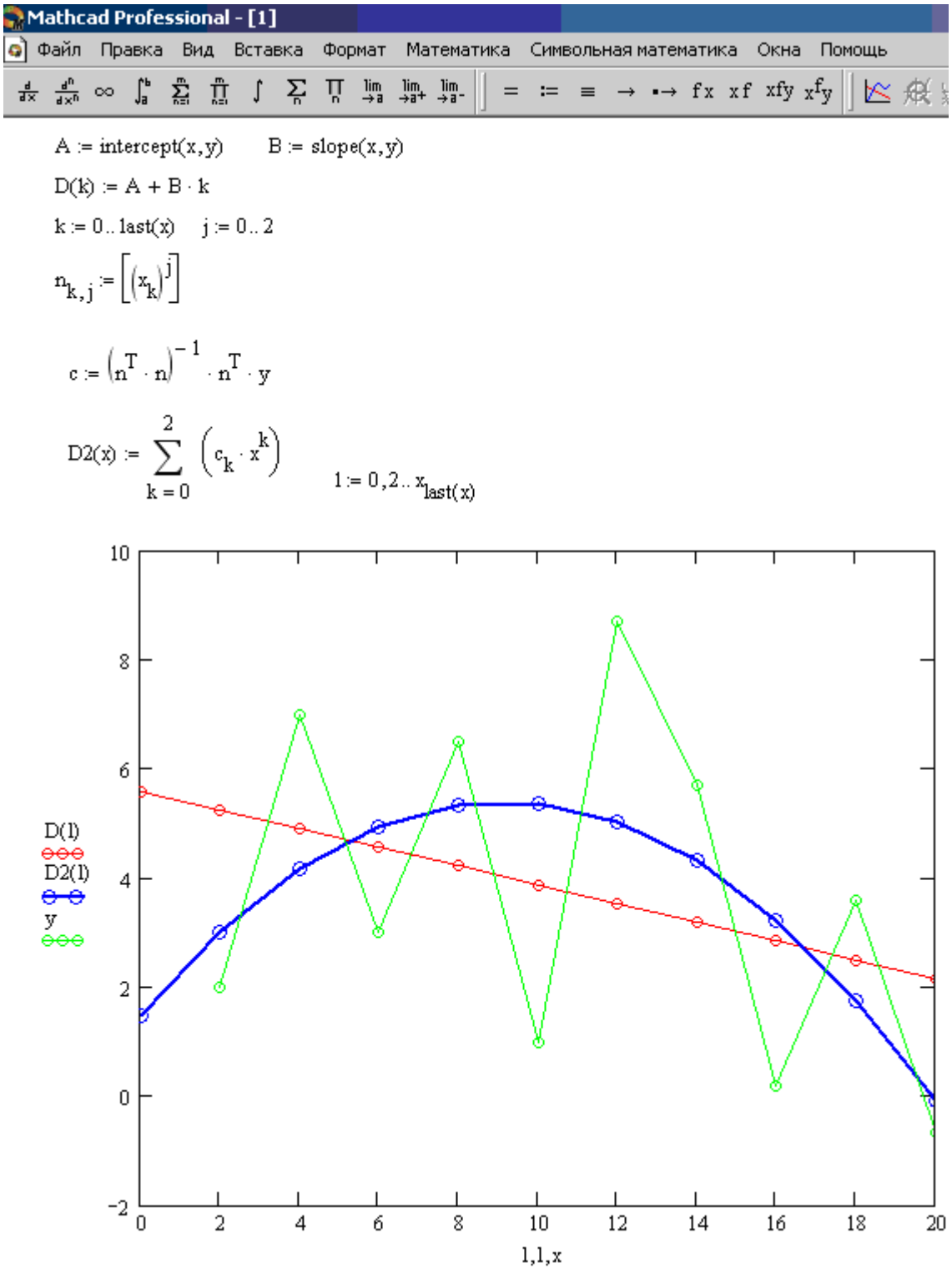

Mathcad Professional - [1]
Файл Правка Вид Вставка Формат Математика Символьная математика Окна Помощь
A := intercept(x,y) B := slope(x,y)
D(k) := A + B · k
k := 0.. last(x) j := 0.. 2
$n_{k,j} := \left[\left(x_k\right)^j\right]$
$c := \left(n^T \cdot n\right)^{-1} \cdot n^T \cdot y$
$D2(x) := \sum_{k=0}^{2} \left(c_k \cdot x^k\right)$
$l := 0, 2.. x_{last(x)}$
D(l)
D2(l)
y
10
8
6
4
2
0
-2
0
2
4
6
8
10
12
14
16
18
20
l,l,x

Рис. 3. МНК

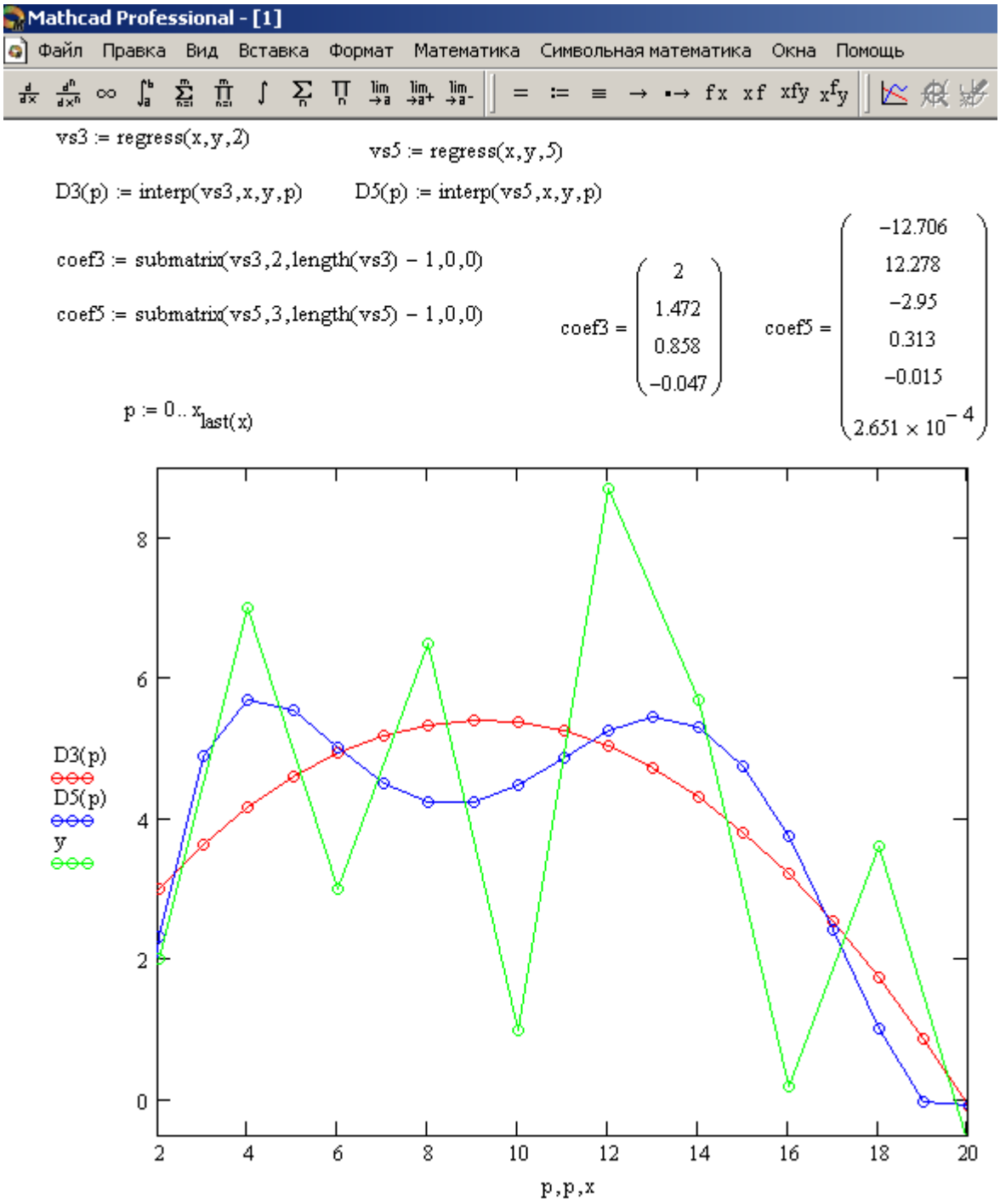


Рис. 4. МНК

Схема Хорнера призначена для знаходження значення полінома для дійсних або комплексних значень аргумента.

Запишемо поліном A(x) у такому вигляді

$$A(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 .$$

Якщо значення аргумента дійсне, цей же поліном можна представити

$$A(x) = (\ldots((a_n x + a_{n-1})x + a_{n-2})x + \ldots + a_1)x + a_0 .$$

Якщо значення аргумента комплексне (x=R+jI, де R,I – дійсна та уявна частина), то схема обрахунку виглядає настіпним чином :

$$Re^H + jIm^H = (Re^c + jIm^c)(R + jI) + a_s,$$

Відображає один крок в циклі, що реалізує схему Хорнера при комплексному x. $Re+jIm$ – поточне значення полінома, a_s – черговий коєфіцієнт полінома, $Re^c + jIm^c$ – „старе" (отримане на попередньому кроці) значення полінома, $Re^H + jIm^H$ – „нове" значення полінома. Комплексну рівність можна представити двома дійсними рівностями.

$$Re^H = Re^c \cdot R - Im^c \cdot I + a_s,$$
$$Im^H = Re^c \cdot I + Im^c \cdot R.$$

Порядок виконання роботи.

1. Задати (довільні) абсциси та ординати точок функції, що інтерполюють, як елементи векторів-стовпців (10 рядків).
   Будуть використовуватися для завдань роботи 10 та 11.
2. В одних координатних осях побудувати графіки інтерполяційних поліномів для 5, 10 та 20 точок інтерполяції. Вивести значення функцій інтерполяції на екран.
3. Отримати для набору значень x та y п.1 коефіцієнти апроксимуючих за МНК для степіня апроксимуючого полінома 1, 2, 3.
4. Побудувати одних координатних осях графіки апроксимуючих функцій за методом найменших квадратів
5. Визначте значення поліному 5-го степеня (коефіцієнти a_5=номер бригади х 5, a_4=номер бригади х 4 і т.д.) за схемою Хорнера за умови, що аргумент дійсний і відповідає номеру бригади .
6. Визначте значення полінома 5-го степіня (коефіцієнти a_5=номер бригади х 5, a_4=номер бригади х 4 і т.д.) за схемою Хорнера за умови, що аргумент комплексний – дійсна частина відповідає номеру бригади, уявна – номер бригади /2.

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з результатами виконання завдань.

Контрольні запитання

1. В чому суть інтерполяційного полінома?
2. Принцип роботи функції *linterp* ?
3. В чому суть метода найменших квадратів?
4. Принцип роботи функцій *intercept* та *slop* ?
5. Принцип роботи функцій *regress* та *interp*?
6. Механізм заповнення масиву *B*, *C* та *D*.
7. Призначення схеми Хорнера?
8. Особливість схеми Хорнера при комплексному значені аргумента.

Лабораторна робота № 11

Інтерполяційні кубічні сплакни та В-сплайни

Мета роботи: Дослідити апроксимацію полінома інтерполяційними кубічними сплайнами та характер поведінки В-сплайна та його похідних

Теоретичні відомості

Нехай маємо упорядкований набір вузлів апроксимації та відповідні їм значення функції , що підлягає апроксимації

$X_0<X_1<X_2<...<X_{m-1}<X_m$ та $Y_0, Y_1, Y_2, ... , Y_{m-1}, Y_m$.

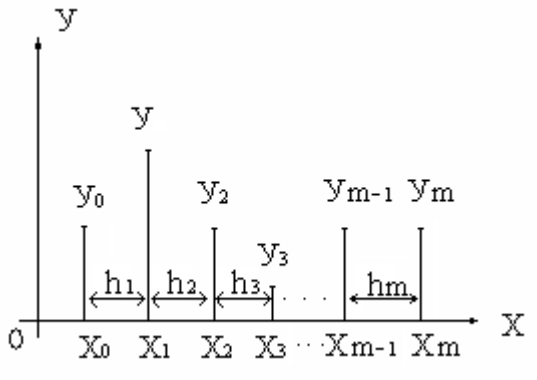


Рис. 1. Інтервали кубічних сплайнів

Розрбиваємо інтервал ($X_0 \le X \le X_m$) на підінтервали $X_{s-1} \le X \le X_s$, $1 \le S \le m$. Ширина S-го підінтервалу $h = X_s - X_{s-1}$. На s-му інтервалі апроксимуючу функцію $\Psi_s(x)$ шукаємо у вигляді кубічного полінома у формі

$$\Psi_s(x) = a_{0,s} + a_{1,s}(X - X_{s-1}) + a_{2,s}(X - X_{s-1})^2 + a_{3,s}(X - X_{s-1})^3 , \; 1 \le S \le m \quad (1).$$

Перша та друга похідна за X від (1) мають вигляд :

$$\Psi_s'(X) = a_{1,s} + 2a_{2,s}(X - X_{s-1}) + 3a_{3,s}(X - X_{s-1})^2 , \; 1 \le S \le m \quad (2).$$

$$\Psi_s''(X) = 2a_{2,s} + 6a_{3,s}(X - X_{s-1}) , \; 1 \le S \le m \quad (3).$$

Структура (1) буде виконувати функції інтерполяційного сплайна , якщо її коєфіцієнти підібрати таким чином ,щоб :

а) значення функції (1) у вузлах апроксимації співпадали б з відповідними значеннями функції Y_s, $0 \le S \le m$, що апроксимується ;

б) на стиках інтервалів мають бути неперервними значення першої та другої похідних структури (1).

Отже , умова а) – умова неперервності кубічного сплайна полягає в тому ,що значення (1) на початку (точка X_{s-1}) і в кінці (точка X_s) s-го інтервалу мають бути рівними Y_{s-1} та Y_s відповідно :

$$\Psi_s(X_{s-1}) = Y_{s-1}\ ,\ 1 \le S \le m \qquad (4)$$

$$\Psi_s(X_s) = Y_s\ ,\ 1 \le S \le m \qquad (5)$$

Підставляючи X_{s-1} та X_s замість X в (1) отримуємо з (4) та (5)

$$a_{0,s} = Y_{s-1}\ ,\ 1 \le S \le m \qquad (6)$$

$$a_{0,s} + a_{1,s} h_s + a_{2,s} h_s^2 + a_{3,s} h_s^3 = Y_s,\ 1 \le S \le m,\ h_s = X_s - X_{s-1} \qquad (7)$$

Умова неперервності 1-ої та 2-ої похідних у точці X_s :

$$a_{1,s} + 2a_{2,s} h_s + 3a_{3,s} h_s^3 = a_{1,s+1}\ ,\ 1 \le S \le m\text{-}1 \qquad (8)$$

$$a_{2,s} + 3a_{3,s} h_s^3 = a_{2,s+1}\ ,\ 1 \le S \le m\text{-}1 \qquad (9)$$

Система (6) є уже фактично розв'язаною відносно $a_{0,s}$. Рівняння (7),(8),(9) утворюють систему відносно решти невідомих коєфіцієнтів структури (1) , яка може бути остаточно розв'язана за умов розширення її за рахунок двох додакових рівнянь , а саме рівнянь граничних умов на лівому (точка $X = X_0$) та правому (точка $X = X_m$) кінцях сплайна .

Сам термін "сплайн" означає "гнучка лінійка", оскільки в механіці показується , що форма гнучкої балки (лінійки) , описується саме рівнянням типу (1) ,за умови , що вузлами апроксимації є шарнірні опори. Існує ряд варіантів задання граничних умов , найбільш часто використовуваними з яких є граничні умови 1-го та 2-го роду.

Гранична умова 1-го роду задає значення першої похідної , а 2-го роду – значеня другої похідної на відповідному кінці сплайна.

$$Ngl = 1,\quad \Psi_1'(X_0) = Y_0' \qquad (10)$$

$$Ngl = 2,\quad \Psi_1''(X_0) = Y_0'' \qquad (11)$$

$$Ngr = 1,\quad \Psi_m'(X_m) = Y_m' \qquad (12)$$

$$Ngr = 2,\quad \Psi_m''(X_m) = Y_m'' \qquad (13)$$

Система (7)-(9) шляхом виключення $a_{1,s}$ та $a_{3,s}$ зводиться до системи відносно $a_{2,s}$ виду :

$$h_s a_{2,s} + e_s a_{2,s+1} + h_{s+1} a_{2,s+2} = r_s\ , \qquad (14)$$

де $e_s = 2(h_s\text{-}h_{s\text{-}1})$, $1 \le S \le m\text{-}1$,

$$r_s = 3((Y_s\text{+}1\text{-}Y_s)/h_{s+1}\text{-}(Y_s\text{-}\ Y_{s\text{-}1})/h_s).$$

Якщо систему (14) доповнити двома рівняннями граничних умов , то вона може бути розв'язана методом прогонки , після чого можуть бути визначені коєфіцінтами $a_{1,s}$ та $a_{3,s}$. Шукані коєфіцієнти $a_{0,s}$, $a_{1,s}$, $a_{2,s}$, $a_{3,s}$ зручно розміщуюти у 4-х масивах, де $1 \le S \le m$.

Ngl, *Ngr* – номер граничної умови на лівому (*Ngl*) та правому (*Ngr*) кінцях сплайна .

Кубічний В-сплайн

Нехай вузли апроксимації утворюють арифметичну прогресію

$$x_S = x_0 + sh, \quad 1 \le s \le m, \qquad (15)$$

де $h = const$. У відповідність кожному x_S поставлено значення y_S функції, що підлягає апроксимації.

Кубічний В-сплайн (базовий сплайн) для інтервалу $x_{S\text{-}2} \le x \le x_{S+2}$ визначається так:

$$B_S(x) = \begin{cases} \frac{1}{6}(u_s + 2)^3, x \in [x_{s-2}, x_{s-1}) \\ \frac{2}{3} - \frac{1}{2}(u_s^3 + 2u_s^2), x \in [x_{s-1}, x_s) \\ \frac{2}{3} + \frac{1}{2}(u_s^3 - 2u_s^2), x \in [x_s, x_{s+1}) \\ \frac{1}{6}(2 - u_s)^3, x \in [x_{s+1}, x_{s+2}) \\ 0 \text{ для усіх інших x} , \end{cases} \qquad (16)$$

де $u_s = \dfrac{x - x_s}{h}$.

Перша, друга та третя похідні від В-сплайна визначаються відповідно

$$\mathrm{B}_s'(x) = \begin{cases} \frac{1}{2\mathrm{h}}(u_s + 2)^2, x \in [x_{s-2}, x_{s-1}], \\ -\frac{1}{2h}(3u_s^2 + 4u_s), x \in [x_{s-1}, x_s], \\ \frac{1}{2h}(3u_s^2 - 4u_s), x \in [x_s, x_{s+1}], \\ -\frac{1}{2\mathrm{h}}(2 - u_s)^2, x \in [x_{s+1}, x_{s+2}], \\ 0 \text{ для усіх інших x}. \end{cases}, \quad (17)$$

$$\mathrm{B}_s''(x) = \begin{cases} \frac{1}{\mathrm{h}^2}(u_s + 2), x \in [x_{s-2}, x_{s-1}], \\ -\frac{1}{h^2}(3u_s + 2), x \in [x_{s-1}, x_s], \\ \frac{1}{h^2}(3u_s - 2), x \in [x_s, x_{s+1}], \\ \frac{1}{\mathrm{h}^2}(2 - u_s), x \in [x_{s+1}, x_{s+2}], \\ 0 \text{ для усіх інших x}, \end{cases}, \quad (18)$$

$$\mathrm{B}_s'''(x) = \begin{cases} \frac{1}{\mathrm{h}^3}, x \in [x_{s-2}, x_{s-1}], \\ -\frac{1}{3h^3}, x \in [x_{s-1}, x_s], \\ \frac{1}{3h^3}, x \in [x_s, x_{s+1}], \\ -\frac{1}{\mathrm{h}^3}, x \in [x_{s+1}, x_{s+2}], \\ 0 \text{ для усіх інших x}. \end{cases}, \quad (19)$$

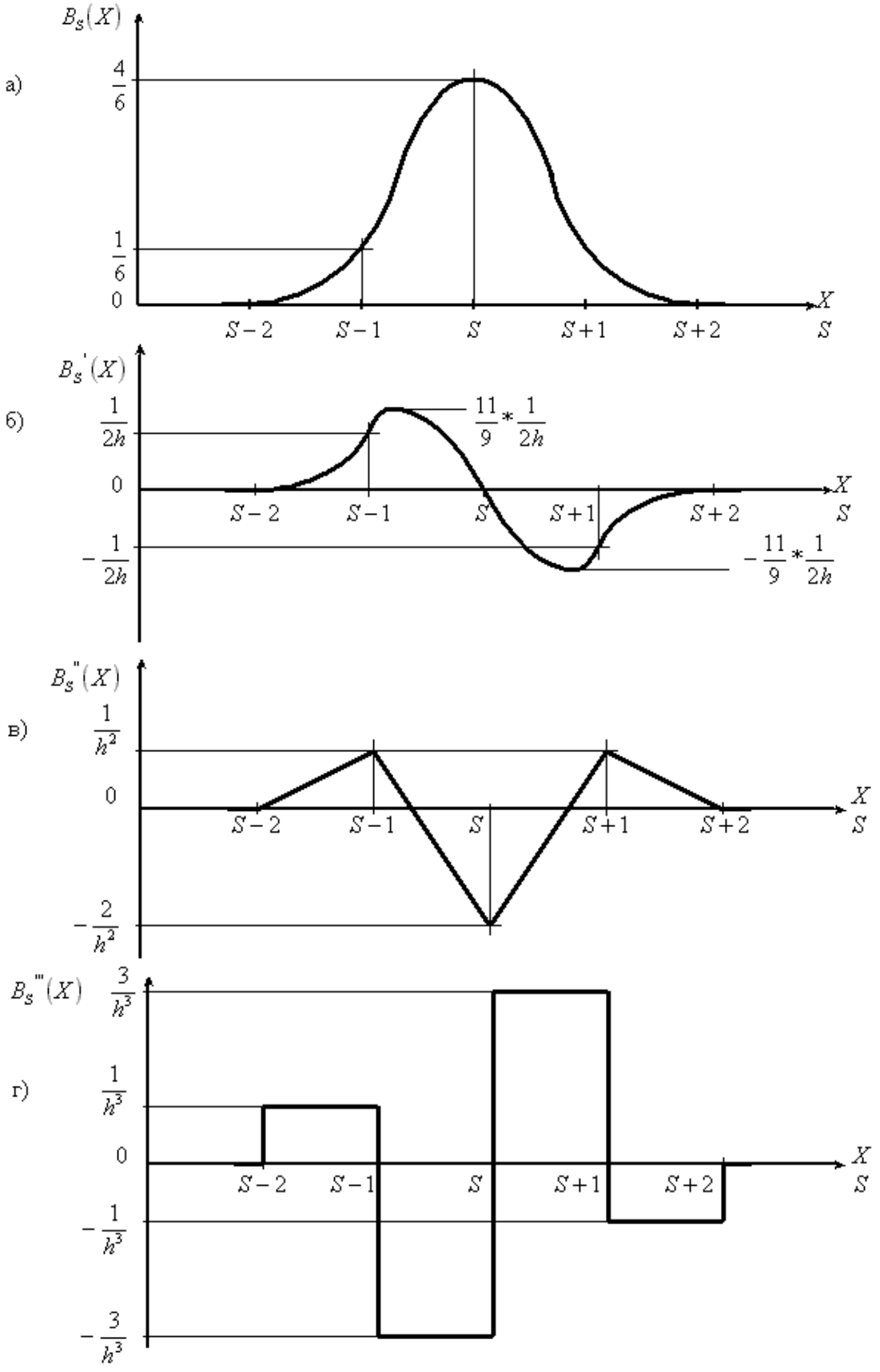


Рис.2. Графіки B-сплайна та його похідних

Ідея використання B-сплайнів для апроксимації, зокрема для інтерполяції полягає в тому, що до кожного із вузлів (нумерація від 1 до *m*+1, вузли – рівновіддалені, тобто утворюють арифметичну прогресію)

«прив'язується» (з коефіцієнтом b_s) В-сплайн. Сума усіх прив'язаних таким чином В-сплайнів і утворюють арифметичну функцію.

Якщо врахувати, що дія окремого В-сплайна розраховується на 2 кроки вліво та вправо відносно точки його «прив'язкиується», апроксимуючу функцію можна представити так

$$\Psi_s(x) = \sum_{s=-1}^{m+1} b_s B_s(x), \tag{20}$$

А якщо врахувати, що В-сплайн в точку на крок лівіше та крок правіше від центрального вузла (точки «прив'язки») дорівнює 1/6, а в центральній точці він дорівнює 2/3=4/6, то формула (20) набуває вигляду

$$b_{s-1}B_{s-1}\ x_s\ + b_s B_s\ x_s\ + b_{s+1}B_{s+1}\ x_s\ = y_s, 0 \le s \le m. \tag{21}$$

де s-номер інтервалу, в якій попадає значення x. Якщо система (21) доповнити двома рівняннями граничних умов (лівому s=0 та правомуs=m кінцях інтервалу апроксимації), то отримана система лінійних алгебраїчних рівнянь може бути розв'язана відносно «коефіцієнтів прив'язки» b_s, $-1 \le s \le m+1$.

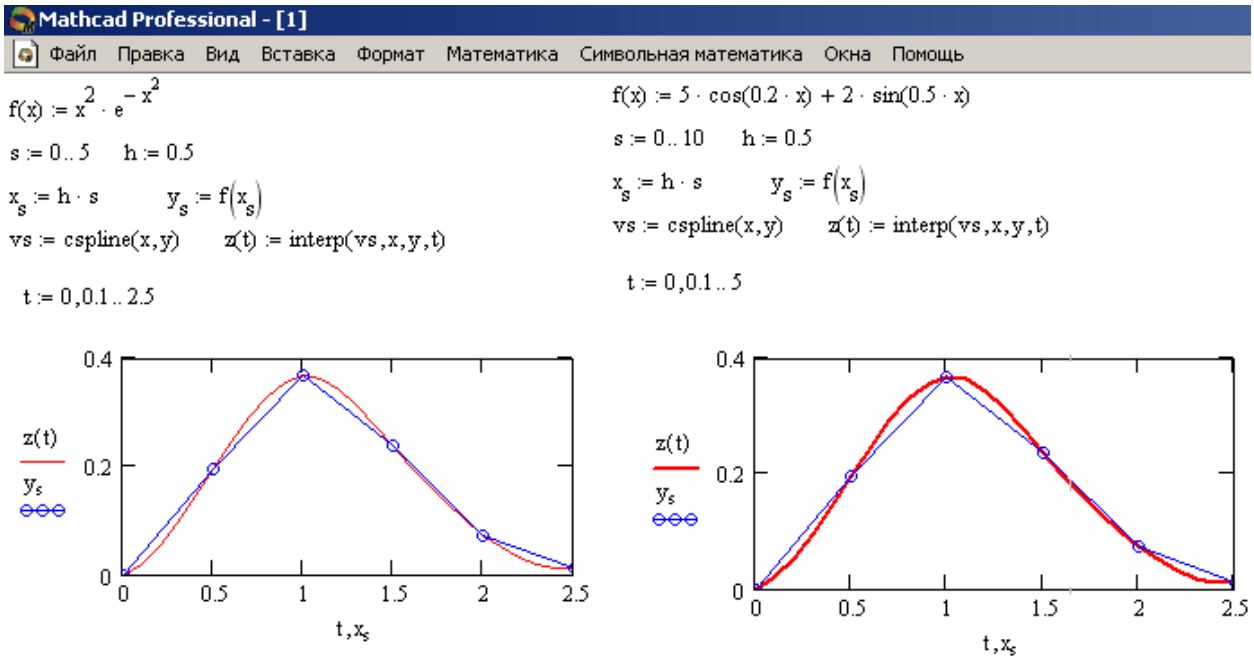


Рис. 3. Інтерполяція кубічним сплайном

Задачу інтерполяції у *MathCad* (визначення значень апроксимуючої функції у проміжках між вузлами) вирішують функції *cspline* та *bspline*.

cspline – повертає вектор-значень коефіцієнтів кубічного сплайна

bspline – повертає вектор-значень коефіцієнтів В-сплайна

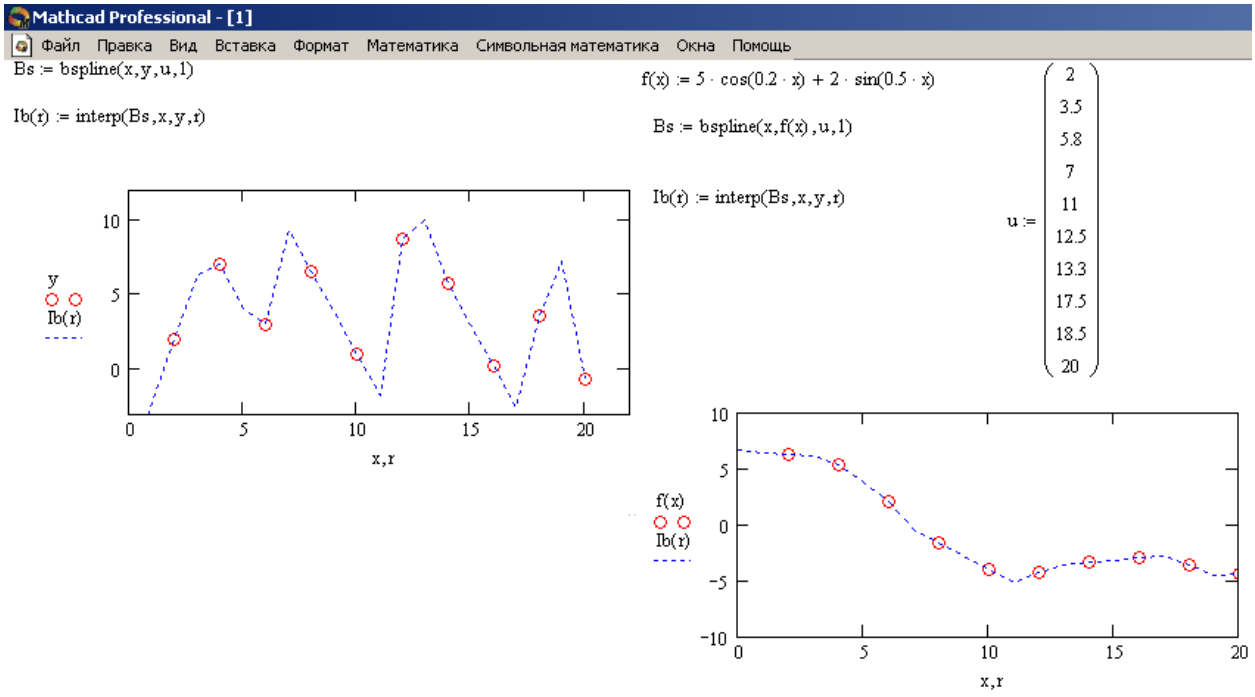


Рис. 4. Інтерполяція В-сплайном

u – вектор значень аргумента в яких відбувається зшивка В-сплайнів, розмірність повинна бути на 1, 2 чи 3 меньша вектора *x*. Перший елемент повинен бути менший або дорівнювати першому елементу вектора *x*. Останій – більший або дорівнювати останьому елементу вектора *x*.

Порядок виконання роботи.

1. Виконати інтерполяцію функції $y = B1\ sin\ (w1*x)+ B2\ cos\ (w2*x)$ за допомогою кубічного сплайна

Коєфіцієнти *B*1 – номер бригади, *B*2 – номер бригади /2, a1 – номер бригади /10, a2 – номер бригади /20, кількість кроків (інтервалів) *h* – номер бригади /10.

2. Виконати інтерполяцію функції $y = B1\ sin\ (w1*x)+ B2\ cos\ (w2*x)$ за допомогою В-сплайна

Коєфіцієнти $B1$ – номер бригади, $B2$ – номер бригади /2, a1 – номер бригади /10, a2 – номер бригади /20, кількість кроків (інтервалів) h – номер бригади /10.

3. Побудувать графіки B-сплайна при різних похідних (рис. 2), S – номер бригади, h – номер бригади /10.

Звіт повинен вміщувати назву роботи, її мету, перелік завдань та результати їхнього виконання. Роздруковані документи з текстами програм.

Контрольні запитання

1. Що таке кубічний сплайн ?
2. В чому суть інтерполяції кубічними сплайнами ?
3. Що таке гранична умова?
4. Основна різниця B-спалйнів і інтерполяційних кубічних сплайнів ?
5. Як працює функція *cspline* та *bspline*?
6. Принцип апроксимації B-спалйнами

Список рекомендованої літератури.

1. Аверіна Т.В., Кубрак Н.А. Динаміка елементів систем : Навч. посібник – К.: ІЗМН, 1998 – 224 с.

2. Калиткин Н.Н. Численные методы – М.: Наука, 1978 – 512 с.

3. Кошляков Н.С., Глинер Э.Б., Смирнов М.М. Основные дифференциальные уравнения математической физики – М.: Физмат, 1962 – 767 с.

4. Гурский Д.А. Вычисления в Mathcad.- Минск: Новое Знание,2003.-814с.