**实验一 信号的时域描述与运算**

实验说明

1、实验类型：验证性实验

实验目的

1、学习funtool命令

2、学习Matlab的信号处理工具箱

3、学会数值运算

4、学会符号运算

实验原理与方法

Funtool命令可以直接在matlab命令行窗口直接输入funtool来实现打开funtool工具箱。通过图像界面可以实现对指定函数的图形的绘制。

在数学应用在，常常需要做求导数、极限、微分的运算，matlab称这些运算为符号运算，通过调用符号函数工具箱来实现。matlab的符号函数工具箱包括微积分运算，解方程，线性代数，化简和代换等运算。

Syms函数可以实现对变量的定义，生成指定的符号对象。

Sovle函数可以实现对指定方程f关于变量x的解

Dsolve函数可以实现解微分方程S，初始条件为s1,s2...,自变量为x。调用格式：dsolve('S','s1','s2',...,'x')。

1. 若不指定自变量为x，则默认自变量为t
2. Dy表示微分，在D后面带有数字，则表示多阶导数，如D2y
3. 如果不指定初值，则最后的结果会包含常数项，如C1,C2

数值计算【numericalcomputation】有效使用数字计算机求数学问题近似解的方法与过程。与符号计算相比，数值计算在科研和工程中应用更加广泛。

Trapz函数可以实现近似求取积分，调用格式：S=trapz(x,y)

eps函数：表示的是一个数可以分辨的最小精度。eps（N）其中N表示数字N的最小精度

符号计算是对未赋值的符号对象（常数、变量、表达式）进行运算和处理。

符号常量是不含变量的符号表达式，用sym命令来创建符号常量。

(1) 创建符号变量和表达式

sym('表达式')

(2) 创建多个符号变量

syms('a','b','c','x')或syms a b c x

符号级数

（1）symsum函数（符号表达式的级数求和）

语法：symsum(s,x,a,b)

说明：x为自变量，s为符号表达式，[a,b]为参数x的取值范围。

（2）taylor函数

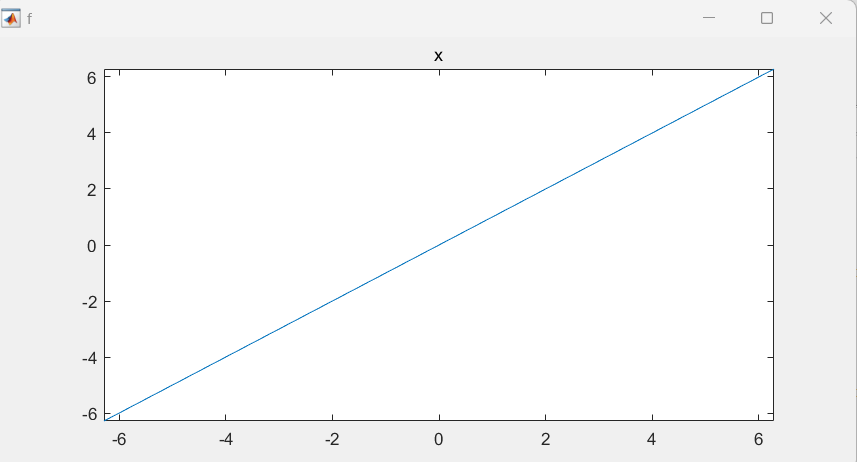
语法：taylor(F,x,n)

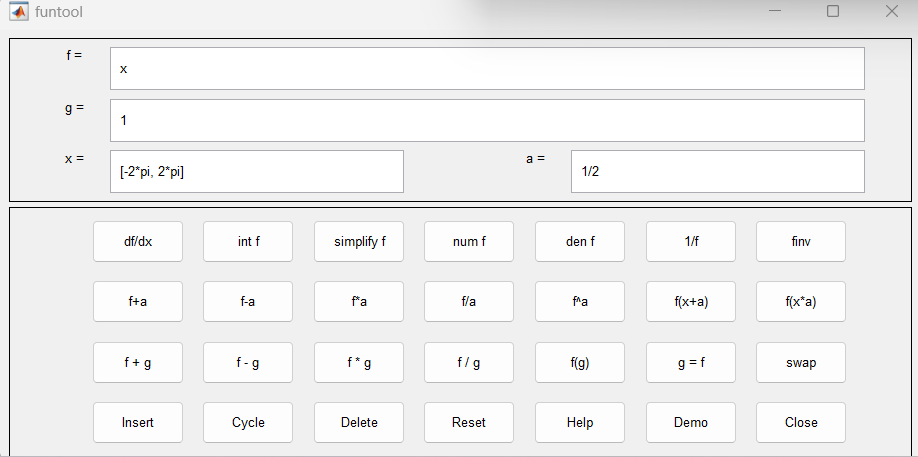
说明：x为自变量，F为符号表达式，对F进行泰勒级数展开至n项，n省略则默认展开前五项。

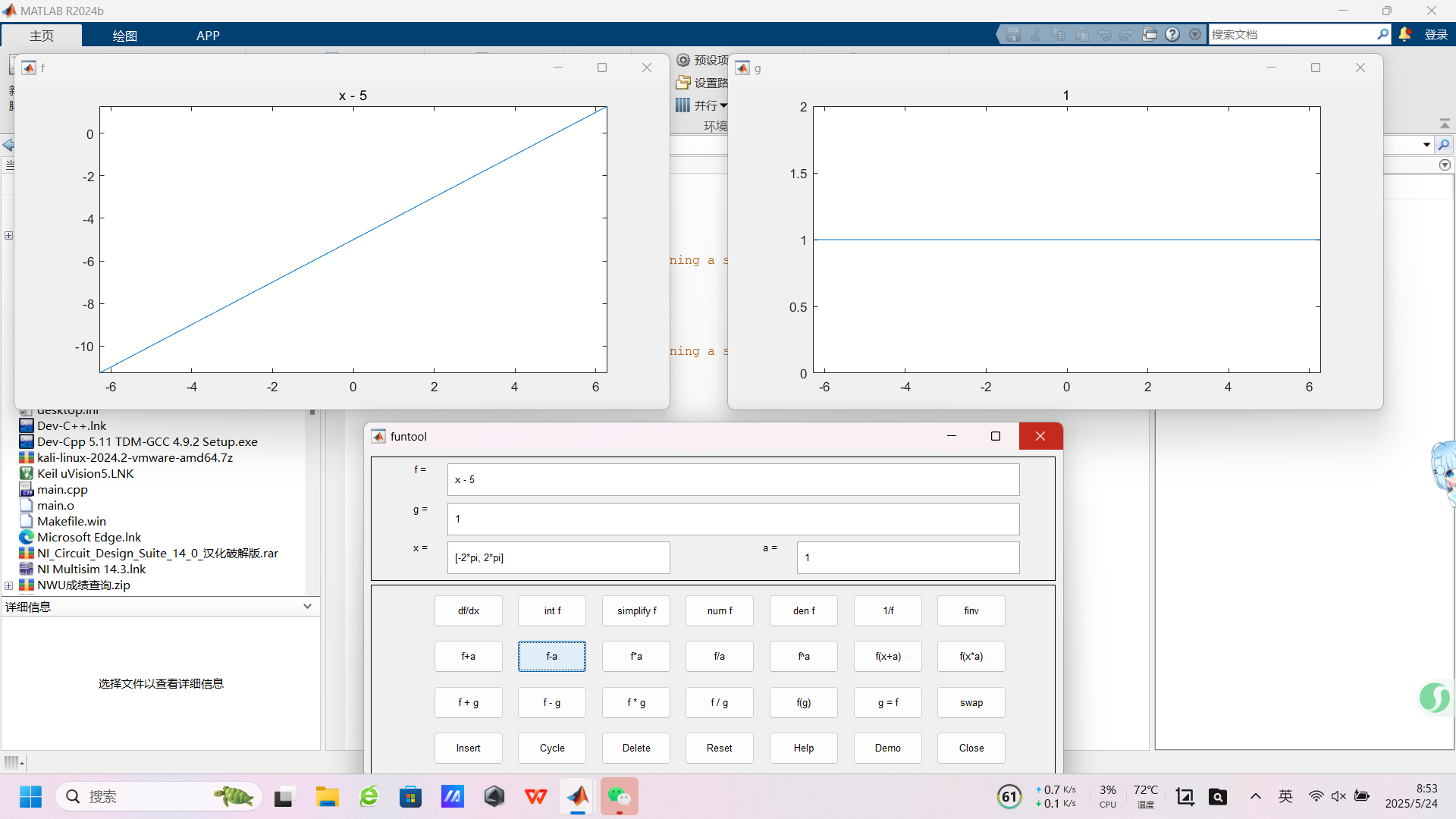
实验内容及步骤

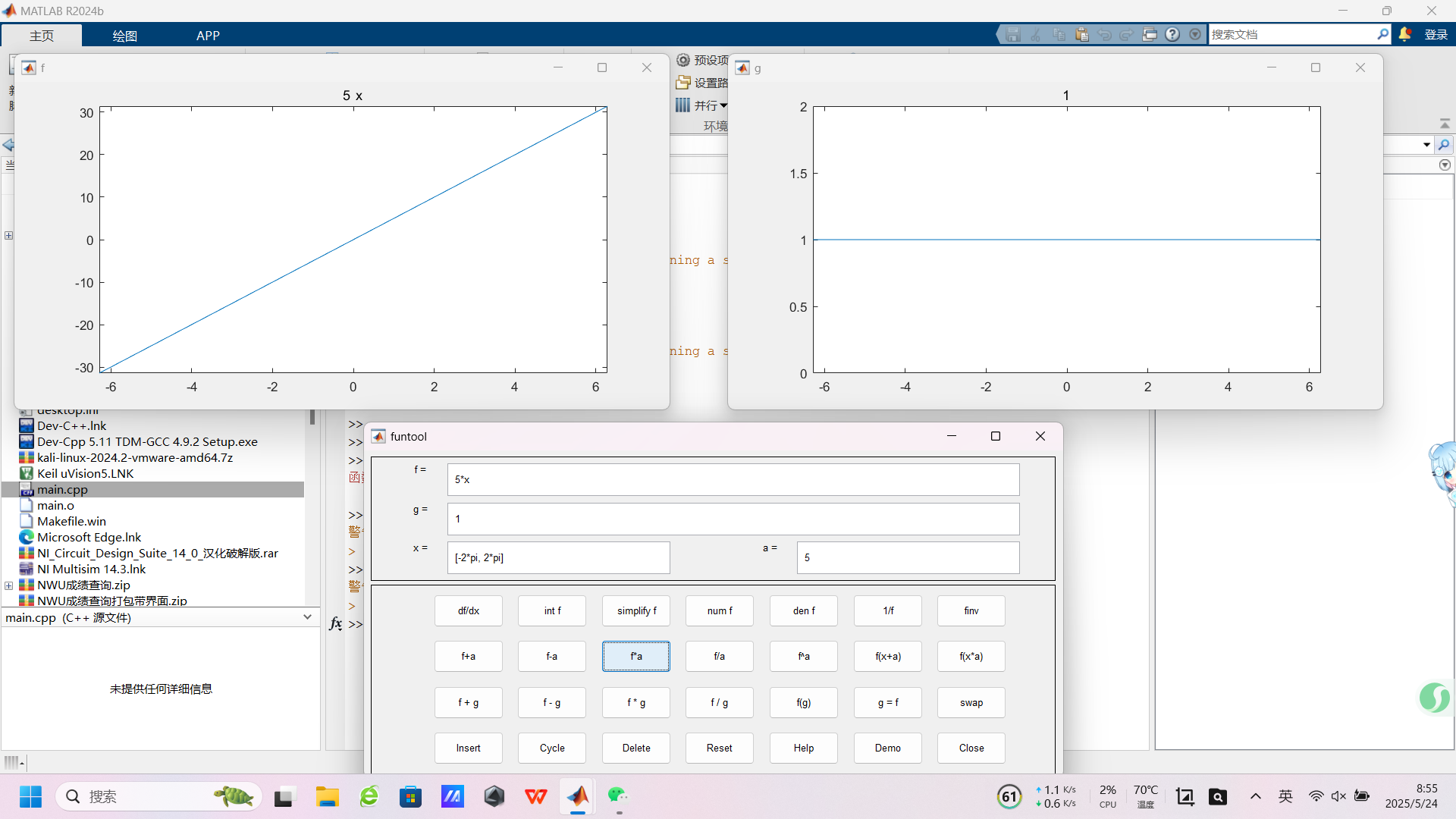
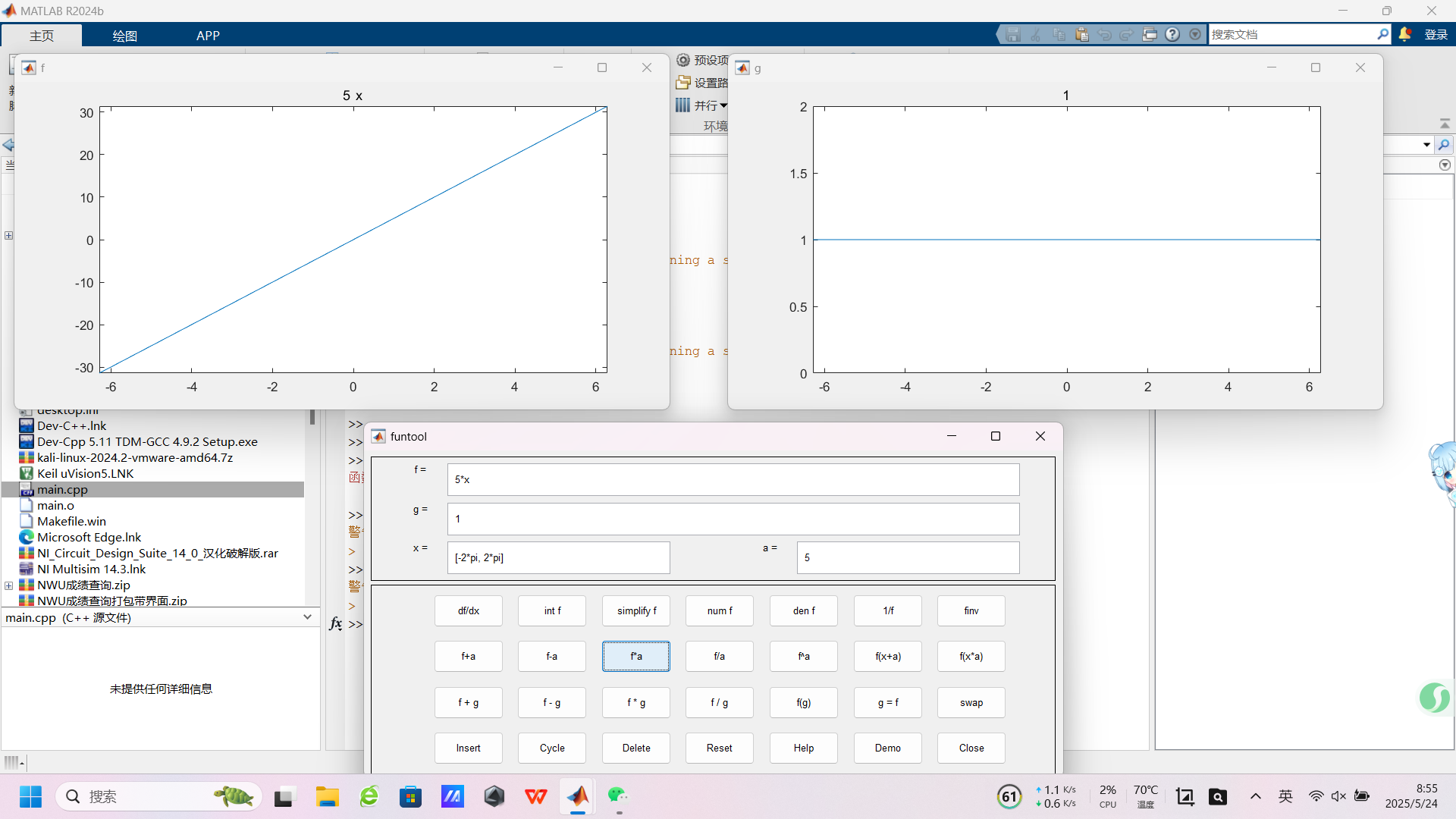
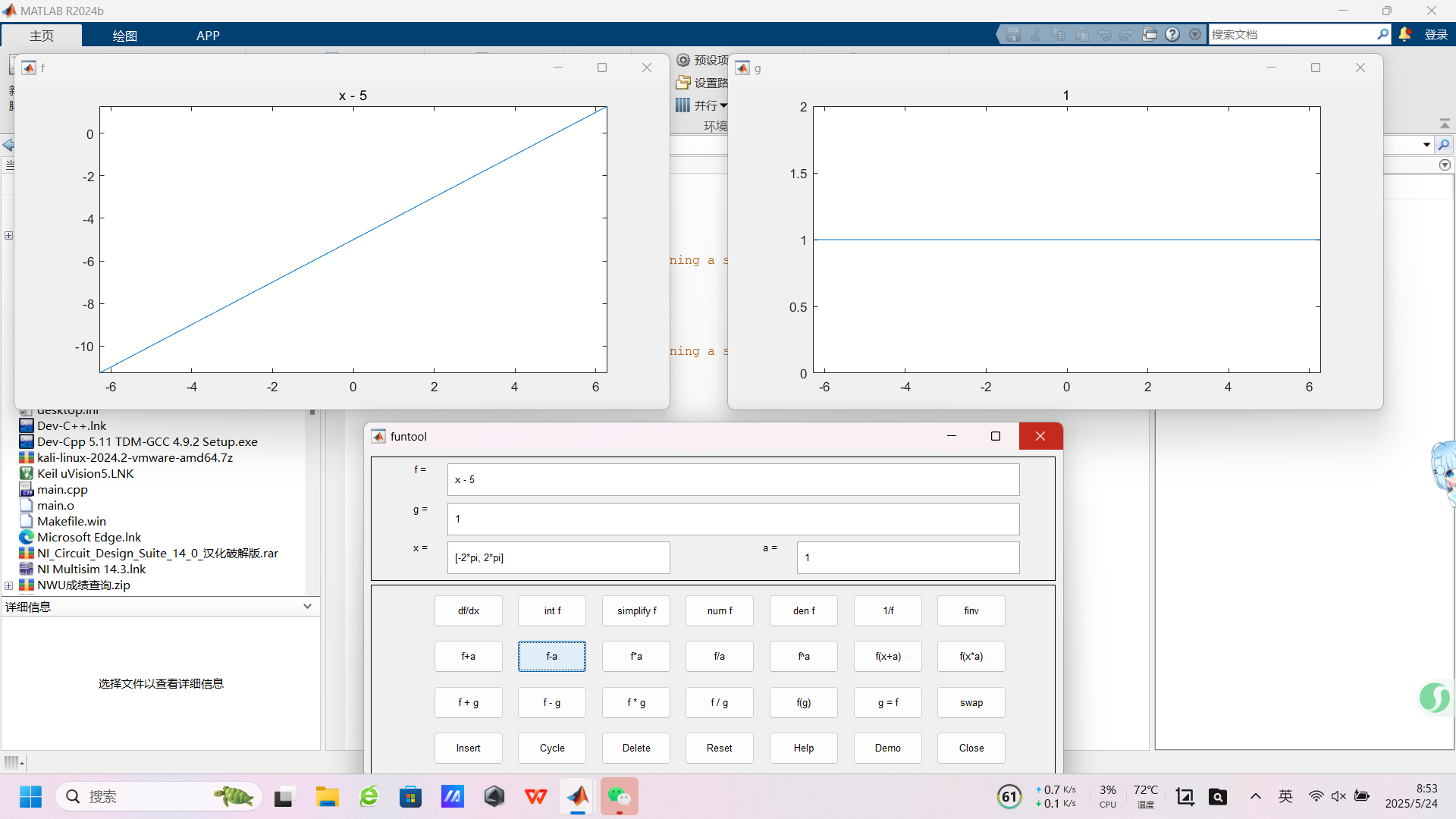
**1、**利用funtool工具以图形化方法生成信号。

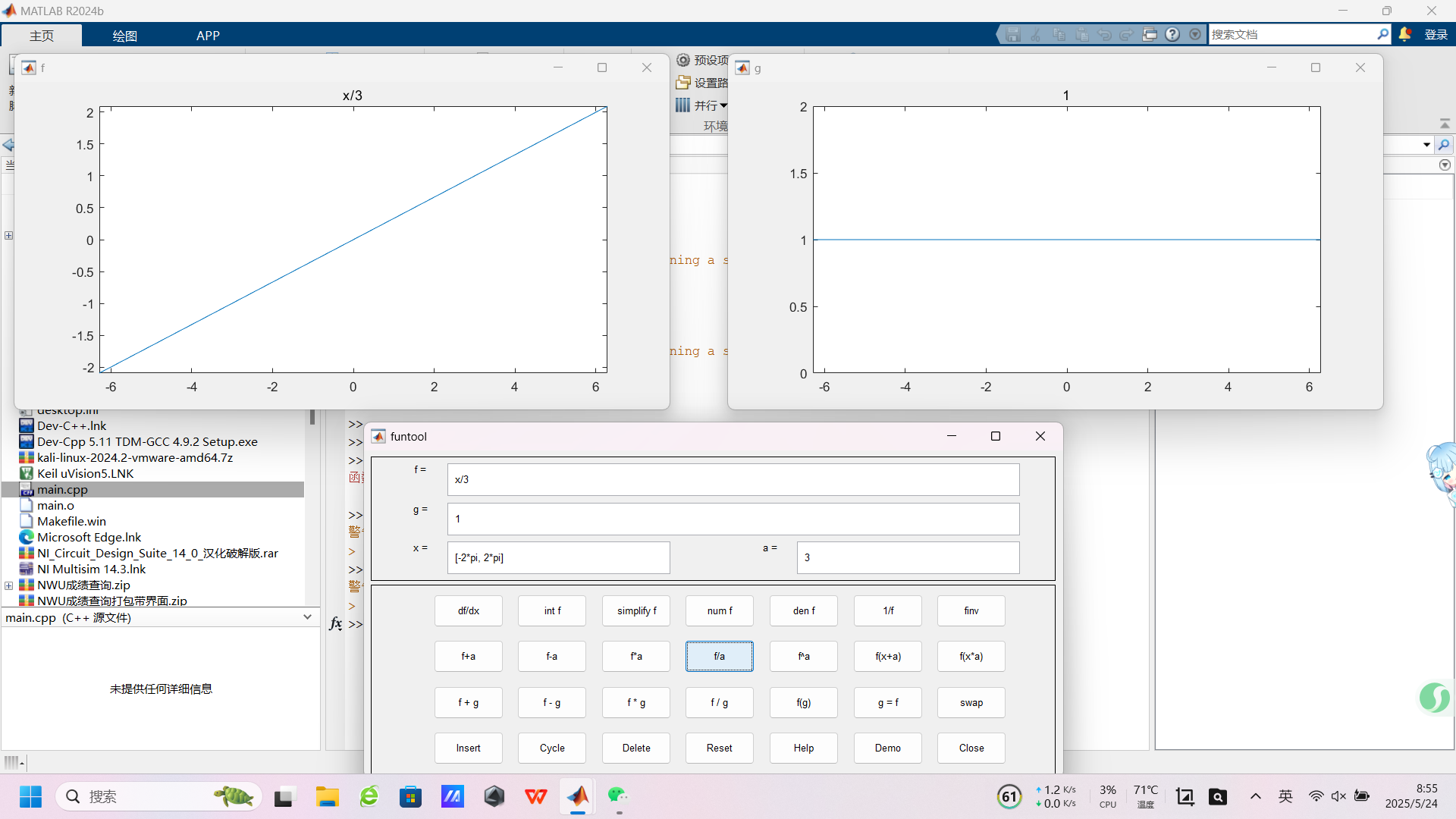
在命令窗口中，输入“funtool”，点击enter键，打开函数图像计算器，改变a的值，生成f=x,f=x-5,f=5x,f=x/3的图形。

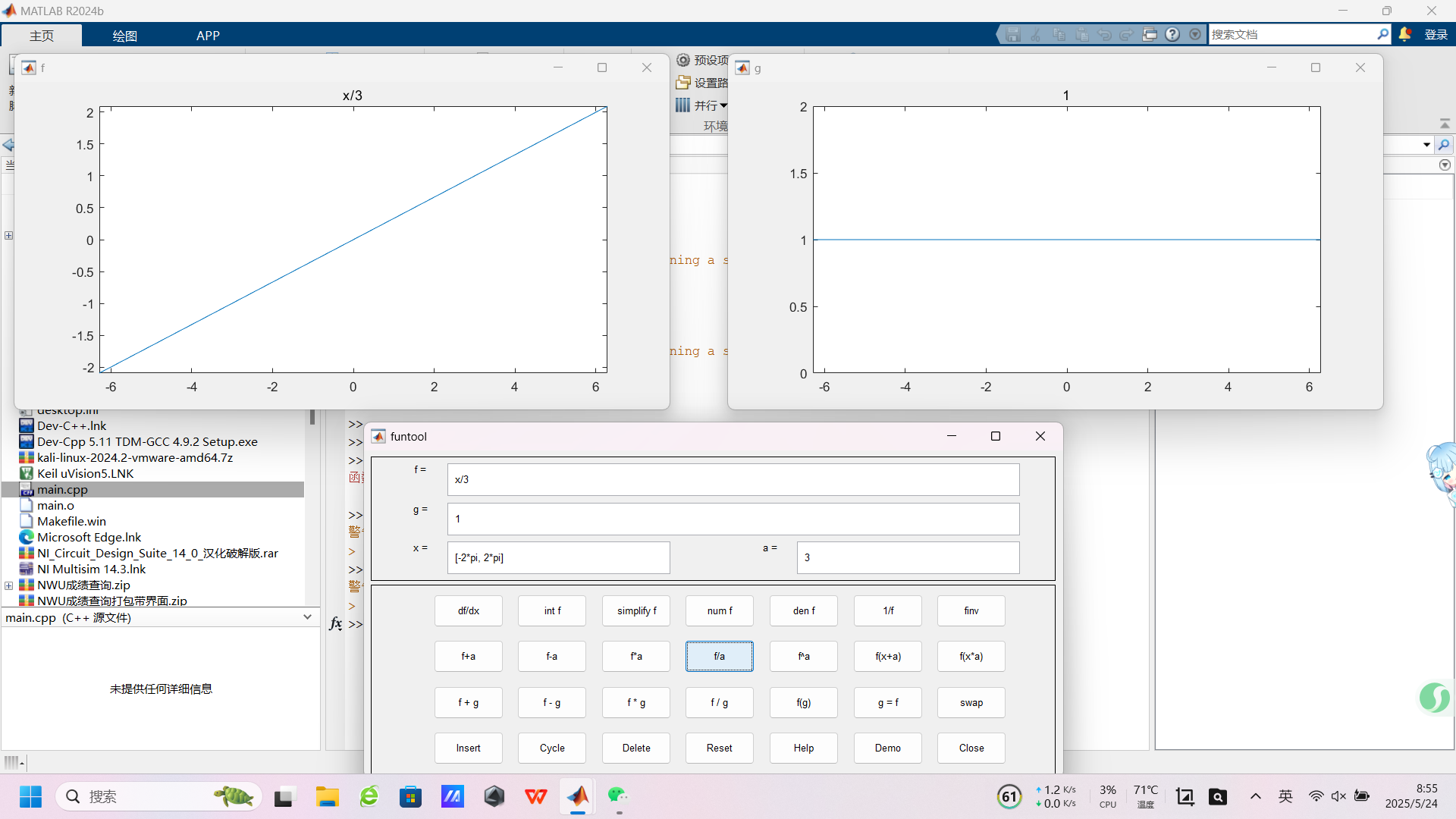
****

****

****

****

****

****

**2、**利用Matlab符号函数工具包以公式符号形式生成信号，符号对象的创建，微积分基本运算。

编程解方程：

（1）syms函数：

调用格式：syms var1 var2 var3...

功能：用于生成多个符号对象

注意：变量之间使用空格；

**以下代码通过matlab的实时编辑器生成代码结果的展示，其中灰色区域为代码，无色区域为输出结果。相比截图等展示方式，更易于插入word与展示。**

syms x

eqn = sin(x) == 1;

sol = solve(eqn, x); % 解：pi/2

disp(sol);

Pi/2

% 方程组求解

syms x y

eqns = [x + y == 3, x - y == 1];

vars = [x y];

[sol\_x, sol\_y] = solve(eqns, vars); % 解：x=2, y=1

disp([sol\_x, sol\_y]);

[2,1]

(2) 解微分方程：

dsolve函数：

调用格式：dsolve('S','s1','s2',...,'x')

功能：解微分方程组S，初始条件为s1,s2...,自变量为x.

注意：① 若不指定自变量为x，则默认自变量为t

② Dy表示微分，在D后面带有数字，则表示多阶导数，如D2y

③ 如果不指定初值，则最后的结果会包含常数项c1 c2。

clear

syms x y;

s=dsolve('Dy=(1+y^2)\*Dx','x','y(0)=1')

警告: Support of character vectors and strings will be removed in a future release. Use sym objects to define differential equations instead.

s = *包含以下字段的 struct:*

x: [1×1 sym]  
 y: [1×1 sym]

s=[s.x,s.y]

s = 

**3、**利用数值计算方法实现基本运算，如数值求和与近似数值积分，近似数值极限及导数；

编程解决如下问题：

1. 求近似导数：已知求该函数在区间中的近似导函数。

（2）求近似数值积分：求积分。

clear

syms t

x=sin(t)

x = 

dx=diff(x,t)

dx = 

t=0:2\*pi/200:2\*pi

t = 1×201

0 0.0314 0.0628 0.0942 0.1257 0.1571 0.1885 0.2199 0.2513 0.2827 0.3142 0.3456 0.3770 0.4084 0.4398 0.4712 0.5027 0.5341 0.5655 0.5969 0.6283 0.6597 0.6912 0.7226 0.7540 0.7854 0.8168 0.8482 0.8796 0.9111 0.9425 0.9739 1.0053 1.0367 1.0681 1.0996 1.1310 1.1624 1.1938 1.2252 1.2566 1.2881 1.3195 1.3509 1.3823 1.4137 1.4451 1.4765 1.5080 1.5394

y=0.2+sin(t)

y = 1×201

0.2000 0.2314 0.2628 0.2941 0.3253 0.3564 0.3874 0.4181 0.4487 0.4790 0.5090 0.5387 0.5681 0.5971 0.6258 0.6540 0.6818 0.7090 0.7358 0.7621 0.7878 0.8129 0.8374 0.8613 0.8845 0.9071 0.9290 0.9501 0.9705 0.9902 1.0090 1.0271 1.0443 1.0607 1.0763 1.0910 1.1048 1.1178 1.1298 1.1409 1.1511 1.1603 1.1686 1.1759 1.1823 1.1877 1.1921 1.1956 1.1980 1.1995

s=trapz(t,y)

s = 1.2566

**4、**符号常量是不含变量的符号表达式，用sym命令来创建

符号常量。

语法：sym('常量')

% 1. 清空命令行窗口和工作区

clc; clear;

% 2. 创建符号常量

s\_const = sym('cos(3)'); % 将字符串转换为符号常量

disp('符号常量:');

disp(s\_const); % 显示: cos(3)

% 3. 计算符号常量的数值近似

num\_val = double(s\_const);

fprintf('数值近似值: %.4f\n', num\_val); % 输出类似: -0.9900（实际结果以运行时为准）

**5、**利用符号函数方法实现基本运算。符号变量是不含变量的符号表达式，用sym命令来创建符号变量；

用matlab符号功能实现下列运算

1. 创建符号变量和表达式sym('表达式')

**创建符号常量**

clc;clear;

const\_pi = sym('pi'); % 创建符号常量pidisp('符号常量的值为：');disp(const\_pi);

num\_approx = double(const\_pi); % 求符号常量的数值近似

fprintf('数值近似值为：%.4f\n',num\_approx);

1. 创建多个符号变量syms('a','b','c','x')或syms a b c x

**创建符号变量和表达式**

clc;clear;

expr = sym('x^2 + 3\*x + 1'); % 创建符号表达式

disp('符号表达式为：');

disp(expr);

**创建多个符号变量**

clc;clear;

syms m n p q; % 创建多个符号变量

matrix\_sym = [m,n;p,q]; % 用符号变量构建符号矩阵

disp('构建的符号矩阵为：');

disp(matrix\_sym);

**6、**符号级数

（1）symsum函数（符号表达式的级数求和）

语法：symsum(s,x,a,b)

说明：x为自变量，s为符号表达式，[a,b]为参数x的取值范围。

% 3. 符号级数求和

clc; clear;

syms k n;

S = symsum(1/k^2, k, 1, inf); % 求和: Σ(1/k^2) from k=1 to ∞

disp('无穷级数求和:');

disp(S); % 输出: pi^2/6

无穷级数求和:

pi^2/6

% 有限项求和

S\_finite = symsum(k^2, k, 1, n); % Σ(k^2) from k=1 to n

disp('有限项求和:');

disp(S\_finite); % 输出: (n\*(2\*n + 1)\*(n + 1))/6

有限项求和:

(n\*(2\*n + 1)\*(n + 1))/6

（2）taylor函数

语法：taylor(F,x,n)

说明：x为自变量，F为符号表达式,对F进行泰勒级数展开至n项，n省略则默认展开前五项。

% 4. 泰勒级数展开

clc; clear;

syms x;

f = exp(x); % 原函数

% 默认展开前5项

taylor\_default = taylor(f, x);

disp('默认泰勒展开(5项):');

disp(taylor\_default); % 输出: x^4/24 + x^3/6 + x^2/2 + x + 1

默认泰勒展开(5项):

x^5/120 + x^4/24 + x^3/6 + x^2/2 + x + 1

% 指定展开到x^3项

taylor\_custom = taylor(f, x, 'Order', 4); % Order = 项数+1

disp('自定义项数展开(到x^3):');

disp(taylor\_custom); % 输出: x^3/6 + x^2/2 + x + 1

自定义项数展开(到x^3):

x^3/6 + x^2/2 + x + 1

% 在x=1处展开

taylor\_point = taylor(f, x, 1, 'Order', 3);

disp('在x=1处展开:');

disp(taylor\_point); % 输出: exp(1) + exp(1)\*(x - 1) + (exp(1)\*(x - 1)^2)/2

在x=1处展开:

exp(1) + exp(1)\*(x - 1) + (exp(1)\*(x - 1)^2)/2