实验一 信号的时域描述与运算

庞晓宇 2024100192

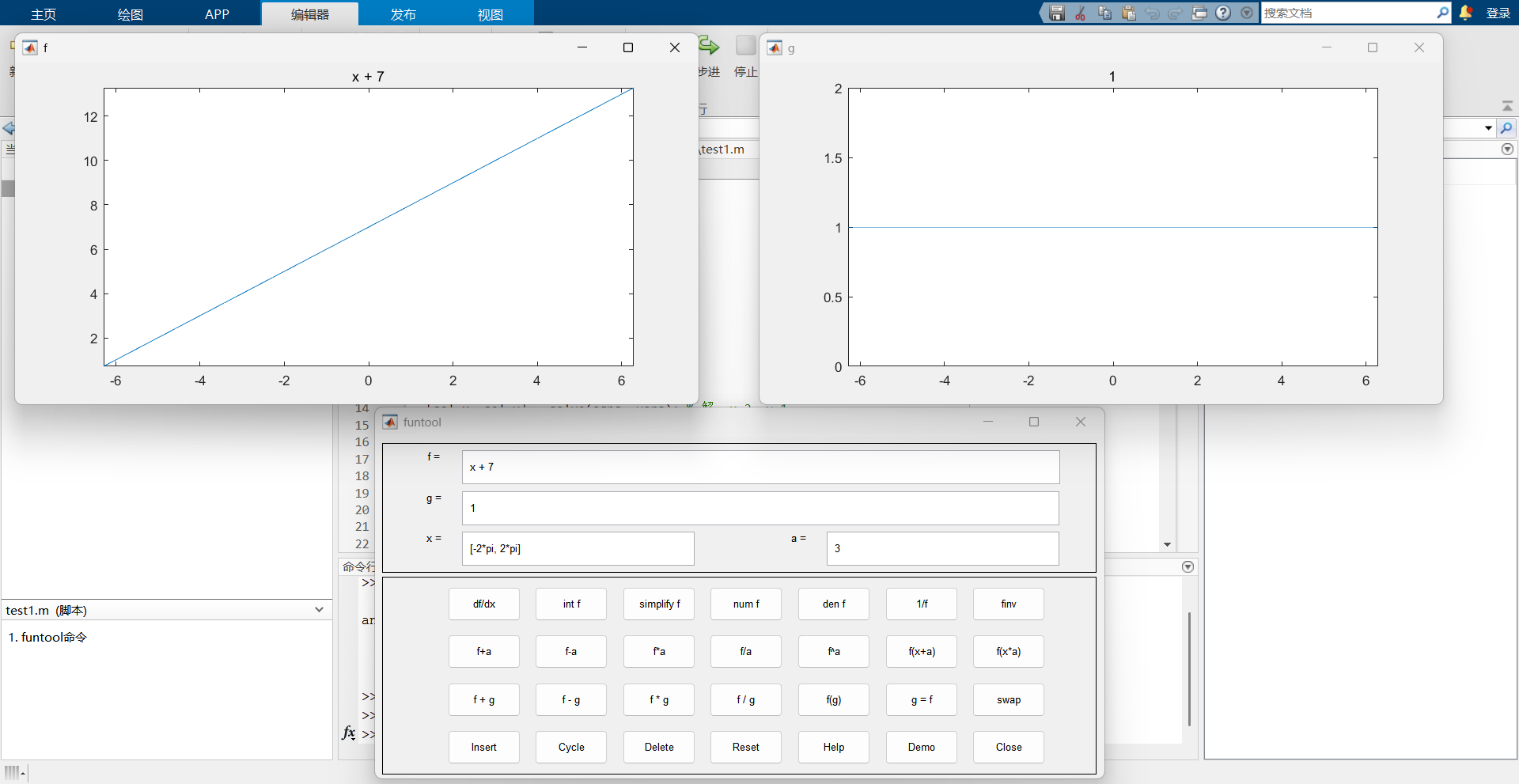
1. 实验内容
2. 学习使用MATLAB的funtool命令进行函数的时域运算（加、减、乘、除、平移等）。
3. 利用MATLAB符号函数工具箱进行符号运算，包括解方程、解微分方程、符号积分与级数展开。
4. 实现数值计算功能，如数值积分（trapz函数）和数值精度分析（eps函数）。
5. 建立符号表达式，包括符号常量、变量、矩阵及符号级数求和。
6. 实验目的
7. 掌握MATLAB在信号时域分析与运算中的基本操作，为后续信号与系统分析奠定基础。
8. 理解符号运算与数值计算的区别及适用场景，熟悉MATLAB工具包的核心功能。
9. 通过生成典型信号（如阶跃信号、门函数）及运算，加深对信号时域特性的理解。
10. 实验原理

* ****Funtool命令****：可以直接在MATLAB命令行窗口中输入“funtool”打开图形化工具箱，用于快速验证函数运算效果。
* ****符号运算****：使用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。
* ****数值计算****：利用MATLAB中的trapz函数进行近似数值积分，eps函数表示数值精度。

1. 实验内容

% 1. funtool命令

Funtool



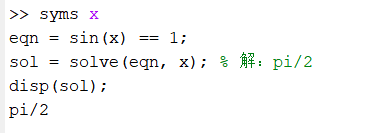
% 2. syms函数 & solve函数

syms x

eqn = sin(x) == 1;

sol = solve(eqn, x); % 解：pi/2

disp(sol);



% 方程组求解

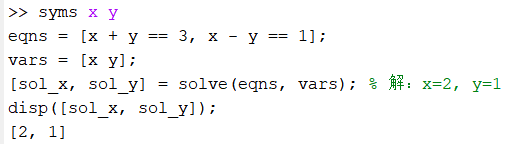
syms x y

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]);



% 3. dsolve函数

% 解微分方程

syms y(x)

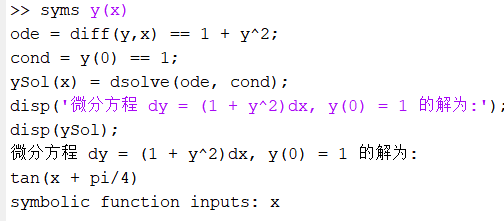
ode = diff(y,x) == 1 + y^2;

cond = y(0) == 1;

ySol(x) = dsolve(ode, cond);

disp('微分方程 dy = (1 + y^2)dx, y(0) = 1 的解为:');

disp(ySol);



% 4. 求 sin(t) 在 [0, 2π] 的近似导数

t = linspace(0, 2\*pi, 1000);

x = sin(t);

dxdt = diff(x)./diff(t);  % 近似导数

t\_diff = t(1:end-1);      % 对应的 t 值（少一个点）

% 绘制结果

figure;

subplot(2,1,1);

plot(t, x);

title('原函数 x = sin(t)');

xlabel('t');

ylabel('x');

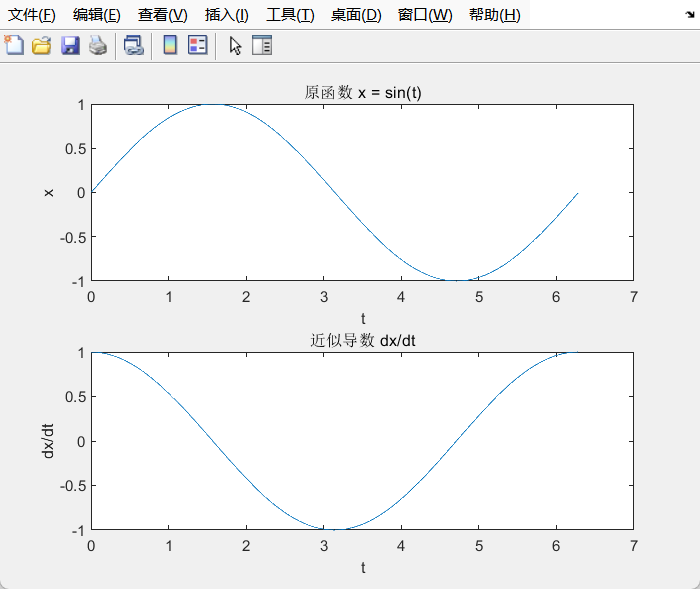
subplot(2,1,2);

plot(t\_diff, dxdt);

title('近似导数 dx/dt');

xlabel('t');

ylabel('dx/dt');



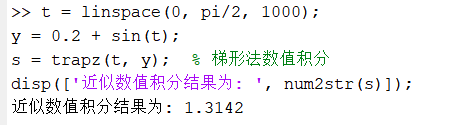
% 5. 求 ∫(0.2 + sin(t))dt 从 0 到 π/2

t = linspace(0, pi/2, 1000);

y = 0.2 + sin(t);

s = trapz(t, y);  % 梯形法数值积分

disp(['近似数值积分结果为: ', num2str(s)]);



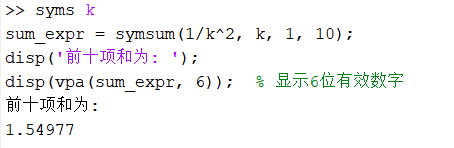
% 6. 求 1 + 1/2^2 + 1/3^2 + ... + 1/k^2 的前十项和

syms k

sum\_expr = symsum(1/k^2, k, 1, 10);

disp('前十项和为: ');

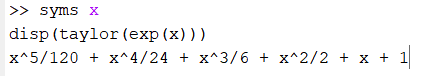
disp(vpa(sum\_expr, 6));  % 显示6位有效数字



% 7. 求 exp(x) 的泰勒展开

syms x

taylor(exp(x));



1. 分析总结
2. 符号运算与数值计算对比：符号运算能精确求解微分方程，但依赖解析表达式；数值计算适用于无解析解的场景，但存在截断误差。
3. Funtool的实用性：提供直观的图形化操作，适合快速验证函数运算效果，但复杂运算仍需编程实现。
4. 泰勒展开的局部逼近特性：泰勒级数在展开点附近逼近效果最佳，随阶数增加，逼近范围扩大，但计算复杂度提高。