

第5章

控制系统时域分析的MATLAB实现

控制系统时域分析是指对被控对象输入典型信号,分析系统随时间变化的过程和特征。通过分析系统在给定输入信号作用下的输出响应,研究控制系统的稳定性和动态性能指标。时域分析的特点是直观、准确。本章借助 MATLAB 完成对控制系统建模、绘制不同输入信号下的响应曲线、分析改变系统参数对输出性能的影响并进行稳定性判断。

5.1 模型建立与化简

框图化简是将复杂系统转换为典型环节传递函数,实现系统建模的一种等效变换方法。

5.1.1 串联结构

串联结构如图 5.1 所示。

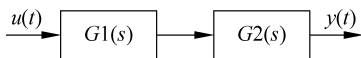


图 5.1 串联结构

语法格式:

$G = G1 * G2$ 或 $G = \text{series}(G1, G2)$

若已知 $G1(s)$ 和 $G2(s)$ 的分子和分母,也可直接写成 $[\text{num}, \text{den}] = \text{series}(\text{num1}, \text{den1}, \text{num2}, \text{den2})$ 。

【例 5-1】 建立图 5.2 所示串联结构模型的传递函数。

程序命令:

```
G1 = tf([2,5,1],[1,2,3]); G2 = zpk(-2,-10,5);  
G = G1 * G2
```

或

```
G = series(G1,G2)
```

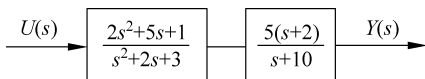


图 5.2 串联结构模型

5.1.2 并联结构

并联结构如图 5.3 所示。

语法格式：

```
G = G1 + G2
G = parallel(G1,G2)
```

若已知 $G1(s)$ 和 $G2(s)$ 的分子和分母,也可直接写成 $[\text{num}, \text{den}] = \text{parallel}(\text{num1}, \text{den1}, \text{num2}, \text{den2})$ 。

【例 5-2】 建立图 5.4 所示并联结构模型的传递函数。

程序命令：

```
G1 = tf([2,5,1],[1,2,3]);
G2 = zpk(-2,-10,5);
G = G1 + G2
```

或

```
G = parallel(G1,G2)
```

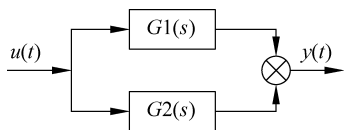


图 5.3 并联结构

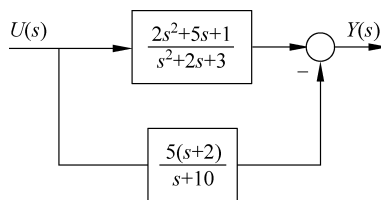


图 5.4 并联结构模型

5.1.3 反馈结构

反馈结构如图 5.5 所示。其中,“+”为正反馈,“-”为负反馈。

语法格式：

```
G = feedback(G1,G2,Sign)
```

其中,Sign 为表示反馈的符号,Sign=1 表示正反馈,Sign=-1 表示负反馈,默认为负反馈。也可以直接写成[num,den]=feedback(num1,den1,num2,den2,sign)。

【例 5-3】 建立图 5.6 所示反馈结构模型的传递函数。

程序命令:

```
G1 = tf([2,5,1],[1,2,3]);
G2 = zpk(-2,-10,5);
G = feedback(G1,G2,-1)
```

或

```
G = feedback(G1,G2)
```

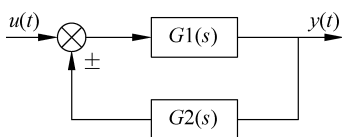


图 5.5 反馈结构

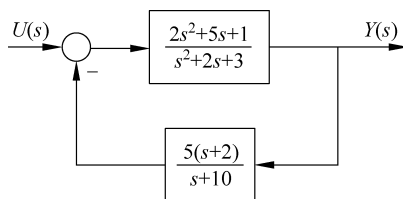


图 5.6 反馈结构模型

5.1.4 复杂结构

复杂结构是多种形式的组合,包括串联结构、并联结构和反馈结构,如图 5.7 所示。

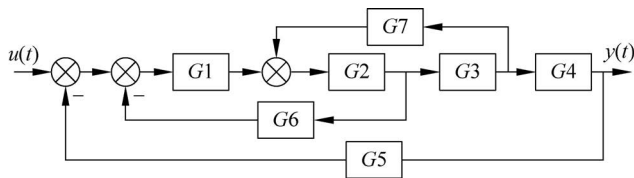


图 5.7 复杂结构

建立复杂结构模型一般需要下面的步骤:

(1) 按照梅逊公式画出信号流图,并按各个模块的通路顺序编号,主通路从左到右顺序排列,如图 5.8 所示。

(2) 使用 append 命令实现各模块的连接。

$$G = \text{append}(G1, G2, G3, \dots)$$

(3) 指定连接关系: 写出各通路的输入/输出关系矩阵 $\mathbf{Q}$, 它的第一列是模块通路编号, 从第二列开始为进入该节点模块的通路编号。

(4) 指定输入/输出编号。

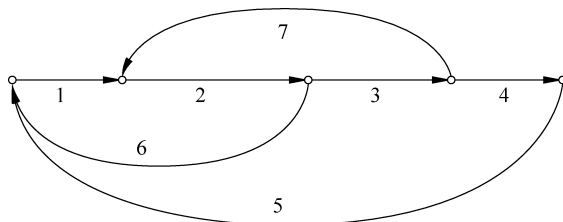


图 5.8 信号流图

输入：INPUTS 为系统整体输入信号的通路编号。

输出：OUTPUTS 为系统整体输出信号的通路编号。

(5) 使用 connect 命令构造整个系统的模型。

$\text{Sys} = \text{connect}(G, Q, \text{INPUTS}, \text{OUTPUTS})$

说明：各模块传递函数也可以通过 blkbuild 命令建立无连接的数学模型，则第(2)步修改如下：将各通路的信息存放在变量中，通路数存放在 nblocks 中，各通路传递函数的分子和分母分别存放在不同的变量中；用 blkbuild 命令求取系统的状态方程模型。

【例 5-4】 建立图 5.9 所示的复杂结构传递函数。

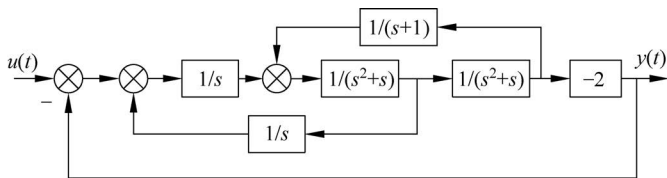


图 5.9 复杂结构

(1) 根据系统结构框图绘制的信号流图如图 5.10 所示。

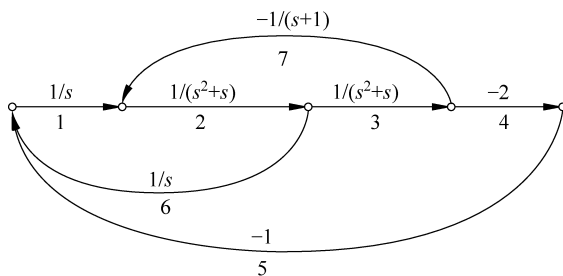


图 5.10 模块的信号流图

(2) 使用 append 命令实现各模块无连接的系统矩阵。

$G1 = \text{tf}(1, [1 \ 0]); G2 = \text{tf}(1, [1 \ 1 \ 0]); G3 = \text{tf}(1, [1 \ 1 \ 0]);$

$G4 = \text{tf}(-2, 1); G5 = \text{tf}(-1, 1); G6 = \text{tf}(1, [1 \ 0]);$

$G7 = \text{tf}(-1, [1 \ 1]);$

```
Sys = append(G1,G2,G3,G4,G5,G6,G7)
```

(3) 指定连接关系。

```
Q = [1 6 5;           % 通路 1 的输入信号为通路 6 和通路 5
      2 1 7;           % 通路 2 的输入信号为通路 1 和通路 7
      3 2 0;           % 通路 3 的输入信号为通路 2
      4 3 0;           % 通路 4 的输入信号为通路 3
      5 4 0;           % 通路 5 的输入信号为通路 4
      6 2 0;           % 通路 6 的输入信号为通路 2
      7 3 0];          % 通路 7 的输入信号为通路 3
```

(4) 指定输入/输出。

```
INPUTS = 1;           % 系统总输入由通路 1 输入
OUTPUTS = 4;          % 系统总输出由通路 4 输出
```

(5) 使用 connect 命令建立整个系统的模型。

```
G = connect(Sys,Q, INPUTS, OUTPUTS)
```

结果:

$$G = - \frac{-2s^2 - 2s - 1.11e-01}{s^7 + 3s^6 + 3s^5 + s^4 - s^3 - 3s^2 - 3s - 2.815e-016}$$

5.1.5 多输入多输出系统

在多输入多输出系统中,需要增加输入变量和输出变量的编号。

语法格式:

```
sys = series(G1,G2,outputA,inputB)           % 级联
sys = parallel(G1,G2,InputA,InputB,OutputA,OutputB) % 并联
```

5.2 控制系统的瞬态响应分析

瞬态响应分析是指分析系统在某一输入信号作用下,输出量从初始状态到稳定状态的响应过程。可利用阶跃信号、斜坡信号、脉冲信号等典型输入信号数学表达简单、便于分析的特点,分析系统输出的特征。

5.2.1 单位脉冲响应

当输入信号为单位脉冲函数 $\delta(t)$ 时,系统输出为单位脉冲响应,可使用 impulse() 函数

计算和显示连续系统的响应曲线。

语法格式：

```
[y, x, t] = impulse(num, den, t)
```

或

```
impulse(G)
```

式中, t 为仿真时间; y 为时间 t 的输出响应; x 为时间 t 的状态响应。

5.2.2 单位阶跃响应

当输入为单位阶跃信号时,系统的输出为单位阶跃响应,使用 `step()` 函数来计算和显示连续系统的响应曲线。

语法格式：

```
[y, x, t] = step(G, t)
```

或

```
step(G)
```

式中, t 为设置的仿真时间,默认时,系统将自动设置时间。

5.2.3 零输入响应

当无输入信号时,使用 `initial()` 函数来计算和显示连续系统的响应曲线。

语法格式：

```
initial(G, x0)           % 绘制系统的零输入响应曲线  
initial(G1, G2, ..., x0) % 绘制系统多个系统的零输入响应曲线  
[y, t, x] = initial(G, x0)
```

式中, G 为系统模型,必须是状态空间模型; x_0 是初始条件; y 为输出响应; t 为时间向量(可省略); x 为状态变量响应(可省略)。

5.2.4 任意函数作用下系统的响应

语法格式：

```
[y, x] = lsim(G, u, t)
```

式中, y 为系统输出响应; x 为系统状态响应; u 为系统输入信号; t 为仿真时间。

若输入为斜波信号,可设置 t =初值:步长:终值,再令 $u=t$ 即可。

【例 5-5】 某系统闭环传递函数为 $G(s) = \frac{s+1}{s^3+2s^2+3s+1}$ 。

要求:

- (1) 画出单位脉冲响应(Impulse Response)曲线;
- (2) 画出单位阶跃响应(Step Response)曲线;
- (3) 画出初始条件 Initial Conditions 为 $[1 \ 2 \ 1]$ 时的零输入响应;
- (4) 画出单位斜波响应曲线,即线性仿真结果(Linear Simulation Results)。

程序命令:

```
num = [1,1]; den = [1,2,3,1]; G = tf(num,den);
subplot(2,2,1); impulse(G);
subplot(2,2,2); step(G);
subplot(2,2,3); G2 = ss(G);
X0 = [1; 2; 1]; initial(G2,X0);
t = 0: 0.1: 10; subplot(2,2,4); u = t; lsim(G,u,t);
```

4 种典型输入信号下的输出结果如图 5.11 所示。

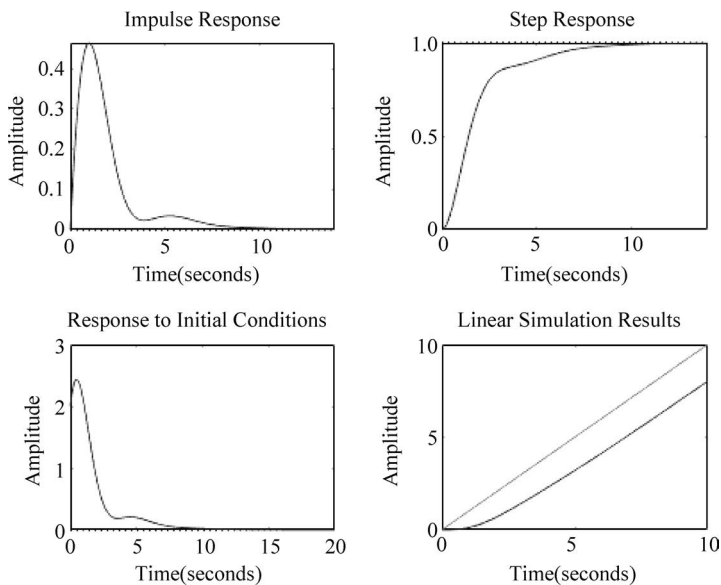


图 5.11 典型输入响应曲线

注: Amplitude 表示振幅。

【例 5-6】 闭环系统如图 5.12 (a)所示,系统输入信号为图 5.12(b)所示的三角波,求系统的输出响应。

程序命令:

```
Gg = tf([10,20],[1,3,5 0]);
```

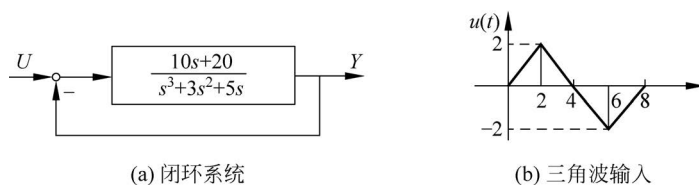


图 5.12 反馈系统及输入信号

```
G = feedback(Gg, 1);
v1 = [0: 0.1: 2]; v2 = [1.9: -0.1: -2]; v3 = [-1.9: 0.1: 0];
t = [0: 0.1: 8]; u = [v1, v2, v3]; [y, x] = lsim(G, u, t);
plot(t, y, t, u); xlabel('Time (seconds)');
ylabel('theta (rad)'); grid on;
```

结果如图 5.13 所示。

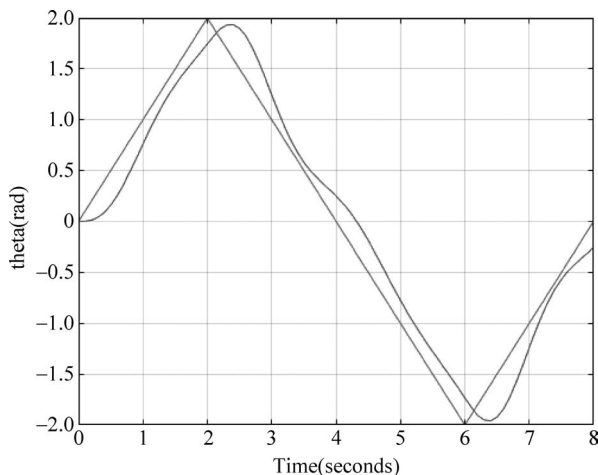


图 5.13 斜坡响应曲线

5.3 二阶系统阶跃响应分析

二阶系统是控制理论中最典型的系统,先辈们对二阶系统时域指标进行了深入细致的研究,总结出了二阶系统超调量、稳态时间、稳态误差、上升时间、峰值时间等多个计算公式。由此可分析二阶闭环系统的阻尼比 ζ 和自由振动频率 ω_n 参数变化对系统输出的影响。

5.3.1 二阶系统时域动态性能指标

对于二阶系统,闭环传递函数的标准形式为 $G(s) = \frac{Y(s)}{U(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$ 。 $\zeta = 0$ 时

称为无阻尼状态, $0 < \zeta < 1$ 时为欠阻尼状态, $\zeta = 1$ 时为临界阻尼状态, $\zeta > 1$ 时为过阻尼状态。当输入阶跃信号时, 表征系统输出的动态性能指标包括超调量 M_p 、稳态时间 t_s 、稳态误差 e_{ss} 、上升时间 t_r 和峰值时间 t_p , 各参数的含义如图 5.14 所示。

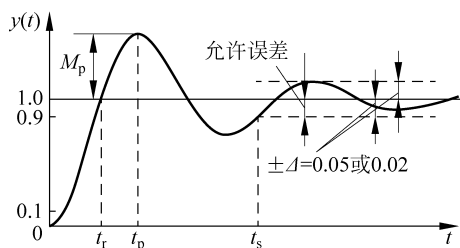


图 5.14 二阶系统响应曲线

其中:

(1) 超调量 M_p 它反映系统的平稳性, 指系统输出曲线第一个波的峰值与给定值的最大偏差 $y(t_p)$ 与终值之差的百分比, 即 $M_p = \frac{y(t_p) - y(\infty)}{y(\infty)} \times 100\%$ 。

(2) 稳态时间 t_s (调节时间) 反映系统的整体快速性, 指输出曲线达到并保持在允许误差范围内所需的最短时间。

(3) 稳态误差 e_{ss} 反映控制系统精度, 指输出曲线结束时稳态值与给定值之差, 用百分数表示。工程上常取 $\pm 5\%$ 或 $\pm 2\%$ 的误差范围。

(4) 上升时间 t_r 反映系统输出的速度快慢, 指响应曲线从 0 时刻开始首次到达稳态值的时间。对于无超调系统, 定义为从到达稳态的 10% 上升到 90% 所需的时间。

(5) 峰值时间 t_p 反映系统的初始快速性, 指阶跃响应输出曲线达到第一峰值所需要的时间。

5.3.2 使用函数获取时域动态指标

实验中使用 MATLAB 的 `step()` 函数获取时域动态指标, 可单击阶跃响应曲线幅值及稳态值的点读取值, 或使用程序自动获取参数。

(1) 计算超调量:

```
y = step(sys)           % 求阶跃响应曲线值
[Y,k] = max(y)          % 求 y 的峰值及峰值时间
C = dcgain(sys)         % 求取系统的终值
Mp = 100 * (Y - C) / C  % 计算超调量
```

(2) 计算稳态时间:

```
[y,t] = step(sys); C = dcgain(sys); i = length(t);
while (y(i) > 0.98 * C) & (y(i) < 1.02 * C)
```

```
i = i - 1;
end
ts = t(i) % 获得稳态时间
```

(3) 计算上升时间:

```
[y,t] = step(sys); C = dcgain(sys); n = 1;
while y(n) <= C; n = n + 1; end;
tr = t(n) % 获得上升时间
```

(4) 计算峰值时间:

```
y = step(sys); [Y,k] = max(y) % 求 y 的峰值
tp = t(k) % 获得峰值时间
```

(5) 计算稳态误差:

```
t = [0:0.001:15]; y = step(sys,t);
ess = 1 - y; Ep = ess(length(ess)) % 获得稳态误差
```

【例 5-7】 根据闭环系统传递函数 $G(s) = \frac{Y(s)}{U(s)} = \frac{100}{s^2 + 3s + 100}$, 绘制阶跃响应曲线, 在曲线上单击获取动态特性参数并查找稳态误差为 2% 时的稳态时间。

程序命令:

```
num = [100]; den = [1, 3, 100];
G = tf(num, den)
step(G)
```

根据图 5.15 所示的结果, 单击图上的峰值点和稳态时间点, 从图上观测出超调量为 62%, 峰值时间为 0.311s; 上升时间为 0.173s; 在 2% 稳态误差下, 稳态时间是 2.58s。

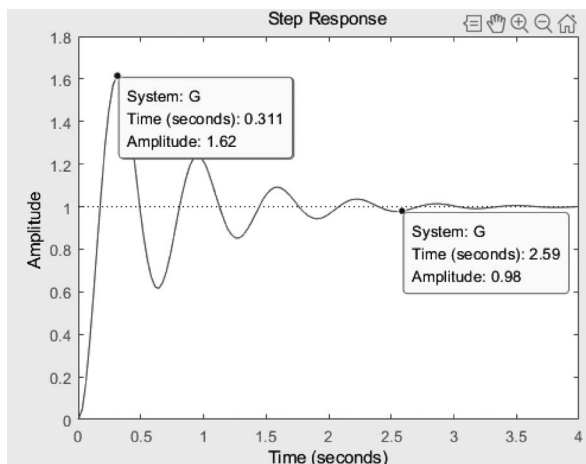


图 5.15 欠阻尼阶跃响应曲线

【例 5-8】 根据标准二阶系统传递函数,在自由振动频率 $\omega_n=1$ 的情况下,改变阻尼比分别为 $\zeta=0$ (无阻尼), $\zeta=0.5$ (欠阻尼), $\zeta=1$ (临界阻尼)和 $\zeta=2$ (过阻尼),绘制阶跃响应曲线,并对结果进行总结。

程序命令:

```
num = 1; den1 = [1, 0, 1]; den2 = [1, 0.5, 1];
den3 = [1, 2, 1]; den4 = [1, 4, 1];
t = 0:0.1:10; % 横坐标的线性空间
G1 = tf(num, den1); G2 = tf(num, den2);
G3 = tf(num, den3); G4 = tf(num, den4);
step(G1, t); hold on; % 保持曲线
text(3, 1.8, '\zeta = 0') % 标注曲线
step(G2, t); hold on; text(3, 1.4, '\zeta = 0.5')
step(G3, t); hold on; text(3, 0.8, '\zeta = 1')
step(G4, t); hold on; text(3, 0.4, '\zeta = 2')
```

结果如图 5.16 所示。

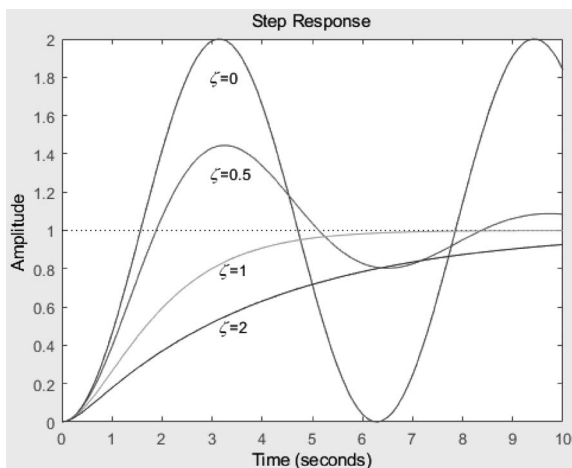


图 5.16 改变阻尼比的单位阶跃响应曲线

结论: 阻尼比越大,超调量越小,达到稳定时间越长,且临界阻尼时超调量为 0。

【例 5-9】 根据标准二阶系统传递函数,在阻尼比 $\zeta=0.5$ 的情况下,改变自由振动频率 $\omega_n=1, \omega_n=2, \omega_n=3$,绘制阶跃响应曲线。

程序命令:

```
t = [0:0.1:10]; num1 = 1; den1 = [1, 1, 1];
G1 = tf(num1, den1);
step(G1, t); hold on; text(0.2, 1.1, '\omega_n = 1');
num2 = 4; den2 = [1, 2, 4]; G2 = tf(num2, den2);
step(G2, t); hold on; text(1.8, 1.1, '\omega_n = 2');
num3 = 9; den3 = [1, 3, 9]; G3 = tf(num3, den3);
```

```
step(G3,t);hold on;text(3.5,1.1,' $\omega_n = 3$ ');
```

结果如图 5.17 所示。

结论： ω_n 相同时， ζ 越大，响应越快； ζ 相同时， ω_n 越大，响应越快。

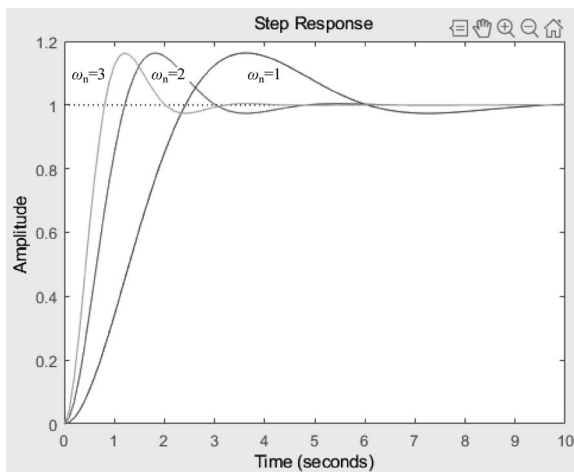


图 5.17 改变频率的单位阶跃响应曲线

5.4 稳定性分析

控制系统得到实际应用的首要条件是系统稳定。本节借助 MATLAB 绘图函数研究系统稳定与不稳定的现象,使用闭环特征根、零极点图和劳斯判据三种方法判别系统的稳定性,并分析改变系统增益对输出性能的影响。

5.4.1 使用闭环特征多项式的根判别稳定性

线性系统稳定的充分必要条件是闭环系统特征方程的所有根具有负实部,使用 MATLAB 求根函数即可判别。

语法格式:

```
roots(den) % 由特征多项式 den, 确定系统的根极点
```

【例 5-10】 已知闭环传递函数 $G(s) = \frac{11}{s^4 + 5s^3 + 7s^2 + 9s + 11}$, 使用 roots() 函数判定系统稳定性。

程序命令:

```
den = [1 5 7 9 11]; % 输入闭环传递函数特征多项式
```

```

p = roots(den);           % 求特征多项式极点
p1 = real(p)              % 求极点的实部
if p1 < 0
    disp(['稳定'])
else
    disp(['不稳定'])
end

```

结果:

```

p1 =    -3.465
      -1.6681
       0.06653
       0.06653
      不稳定

```

结论: 由于极点存在正实部, 所以系统不稳定。

5.4.2 使用零极点图判别稳定性

语法格式:

```

p = pole(G)               % 计算传递函数 G 的极点, 当系统有重极点时, 计算结果不一定准确
z = tzero(G)              % 得出连续和离散系统的零点
[z, gain] = tzero(G)      % 获得零点和零极点增益
pzmap(G)                  % 绘制传递函数 G 的零极点图

```

或

```

pzmap(num, den)           % num, den 表示传递函数的分子、分母

```

该命令计算极点和零点, 并在复平面上画出。极点用“x”表示, 零点用“o”表示。

若极点全部落在左半平面, 则系统稳定, 否则系统不稳定, 因为这是系统稳定的充分必要条件。

【例 5-11】 根据例 5-10 的传递函数, 使用零极点图判定系统的稳定性。

程序命令:

```

num = 11;
den = [1 5 7 9 11];
pzmap(num, den)

```

结果如图 5.18 所示。

结论: 从图 5.18 可以看出, 右半平面上有 2 个极点, 因此该系统是不稳定的。

5.4.3 使用劳斯判据判别稳定性

根据系统的闭环特征方程, 列出劳斯阵列进行判别, 若闭环特征方程为

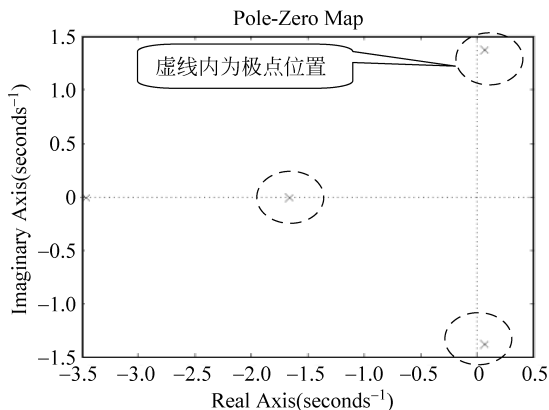


图 5.18 零极点图

$$a_0 S^n + a_1 S^{n-1} + a_2 S^{n-2} + \cdots + a_{n-1} S + a_n = 0$$

将各项系数按下面的格式排成劳斯阵列。

$$\left\{ \begin{array}{l} S^n \quad a_0 \quad a_2 \quad a_4 \quad a_6 \quad \cdots \\ S^{n-1} \quad a_1 \quad a_3 \quad a_5 \quad a_7 \quad \cdots \\ S^{n-2} \quad b_1 \quad b_2 \quad b_3 \quad b_4 \quad \cdots \\ S^{n-3} \quad c_1 \quad c_2 \quad c_3 \quad \cdots \\ \cdots \quad \cdots \\ S^2 \quad d_1 \quad d_2 \quad d_3 \\ S^1 \quad e_1 \quad e_2 \\ S^0 \quad f_1 \end{array} \right. \quad (5-1)$$

计算第一列的数据见式(5-2)。

$$\left\{ \begin{array}{l} b_1 = \frac{a_1 a_2 - a_0 a_2}{a_1}, b_2 = \frac{a_1 a_2 - a_0 a_2}{a_1}, b_3 = \frac{a_1 a_2 - a_0 a_2}{a_1}, \cdots \\ c_1 = \frac{b_1 a_2 - a_1 b_2}{b_1}, c_2 = \frac{b_1 a_2 - a_0 b_2}{b_1}, c_3 = \frac{b_1 a_2 - a_1 b_2}{b_1}, \cdots \\ \vdots \\ f_1 = \frac{e_1 d_2 - d_1 e_2}{e_2} \end{array} \right. \quad (5-2)$$

若劳斯阵列的第一列值 $a_1, b_1, c_1, \cdots, f_1$ 都大于零,则系统是稳定的;若出现至少一个小于零的值,则系统是不稳定的;若第一列有等于零的值,说明系统处于临界稳定状态。

【例 5-12】 已知系统的闭环特征方程为 $s^5 + 2s^4 + s^3 + 3s^2 + 4s + 5 = 0$,使用劳斯判据判断系统的稳定性。

程序命令:

```
clc; p = [1, 2, 3, 4, 5]; p1 = p;
n = length(p);           % 计算闭环特征方程系数的个数 n
if mod(n, 2) == 0        % n 为偶数时
    n1 = n/2;            % 劳斯阵列的列数为 n/2
else
    n1 = (n + 1)/2;      % n 为奇数时, 劳斯阵列的列数为 (n + 1)/2
    p1 = [p1, 0];        % 劳斯阵列左移一位, 后面填写 0
end
routh = reshape(p1, 2, n1); % 列出劳斯阵列前 2 行
RouthTable = zeros(n, n1); % 初始化劳斯阵列行和列为零矩阵
RouthTable(1:2, :) = routh; % 将前 2 行系数放入劳斯阵列
for i = 3:n              % 从第 3 行开始计算劳斯阵列数值
    ai = RouthTable(i - 2, 1)/RouthTable(i - 1, 1);
    for j = 1:n1 - 1      % 按照式(5-2)计算劳斯阵列所有值
        RouthTable(i, j) = RouthTable(i - 2, j + 1) - ai * RouthTable(i - 1, j + 1)
    end
end
end
p2 = RouthTable(:, 1)    % 输出劳斯阵列的第一列数值
if p2 > 0                % 取劳斯阵列的第一列进行判定
    disp(['所要判定系统是稳定的!'])
else
    disp(['所要判定系统是不稳定的!'])
end
end
```

结果:

```
RouthTable =
     1     3     5
     2     4     0
     1     5     0
    -6     0     0
     5     0     0

p2 = 1
     2
     1
    -6
     5

所要判定系统是不稳定的!
```

5.4.4 延迟环节稳定性判别

MATLAB 对纯时间延迟环节 e^{-Ts} 用有理函数来近似, 系统提供 `pade()` 函数拟合为多项式或状态空间传递函数, 一般用于二阶系统拟合计算, 传递函数形式见式(5-3)。

$$G(s) = \frac{s^2 - as + b}{s^2 + as + b} \quad (5-3)$$

语法格式:

```
[num,den] = pade(T,n)      % 也可用[A,B,C,D] = pade(T,n)
printsys(num,den,'s')      % 输出传递函数
```

其中, T 为延迟时间, n 为拟合的阶数。

【例 5-13】 已知下列系统的开环传递函数 $G(s) = \frac{K}{s+1}e^{Ts}$ 是一阶惯性带延迟环节, 其中 $T=0.1$, 当 $K=5, 15, 25, 35$ 时, 要求:

- (1) 利用闭环特征多项式的根判定系统稳定性;
- (2) 利用零极点图判定系统稳定性;
- (3) 使用劳斯判据判定系统稳定性。

步骤:

(1) 用 MATLAB 实现二阶拟合表达式, 将 $T=0.1s, n=2$ 代入函数输出拟合结果。

```
[num,den] = pade(0.1,2);
printsys(num,den,'s')
```

结果:

```
num/den =      s^2 - 60 s + 1200
              -----
              s^2 + 60 s + 1200
```

(2) 根据拟合结果形成闭环二阶系统进行分析, 得到如图 5.19 所示的等价系统框图。

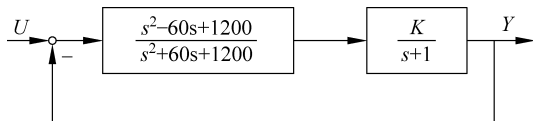


图 5.19 等价系统框图

(3) 利用闭环特征多项式的根编程实现稳定性判定。

程序命令:

```
for K = [5, 15, 25, 35]; g1 = tf([1 -60 1200], [1 60 1200]);
g2 = tf(K, [1 1]);
G = g1 * g2; sys = feedback(G, 1); p = sys.den{1};
p1 = roots(p);          % 求特征多项式的极点
p2 = real(p1);          % 求极点的实部
if p2 < 0
    disp(['K = ', num2str(K), '时所判定系统是稳定的!']);
else
```



```

disp(['K= ',num2str(K),'时所判定系统是不稳定的!']);
end
end

```

结果:

```

K= 5 时所判定系统是稳定的!
K= 15 时所判定系统是稳定的!
K= 25 时所判定系统是不稳定的!
K= 35 时所判定系统是不稳定的!

```

(4) 利用零极点图判定系统稳定性。

程序命令:

```

a = 0;
for K = [5,15,25,35]
a = a + 1; g1 = tf([1 -60 1200],[1 60 1200]); g2 = tf(K,[1 1]);
G = g1 * g2; sys = feedback(G,1);
figure(a); pzmap(sys);
if real(p1) < 0
    disp(['K= ',num2str(K),'时所判定系统是稳定的!']);
else
    disp(['K= ',num2str(K),'时所判定系统是不稳定的!']);
end
end
end

```

绘制的 $K=5, K=15, K=25, K=35$ 时的零极点图如图 5.20(a)、图 5.20(b)、图 5.20(c) 和图 5.20(d) 所示。

结论: 从图 5.20(a) 和图 5.20(b) 可以看出, 当 $K=5, 15$ 时, 极点都落到了左半平面, 系统是稳定的。图 5.20(c) 和图 5.20(d) 中, 当 $K=25, 35$ 时, 分别有 2 个极点落到了右半平面, 系统是不稳定的。

结果:

```

p1 =    -49.5612 + 0.0000i
        -8.2194 + 8.8157i
        -8.2194 - 8.8157i
z1 =    30.0000 + 17.3205i
        30.0000 - 17.3205i
K= 5 时所判定系统是稳定的!
p1 =    -74.6236 + 0.0000i
        -0.6882 + 16.0255i
        -0.6882 - 16.0255i
z1 =    30.0000 + 17.3205i
        30.0000 - 17.3205i
K= 15 时所判定系统是稳定的!
p1 =    -92.2661 + 0.0000i
        3.1331 + 18.1200i

```

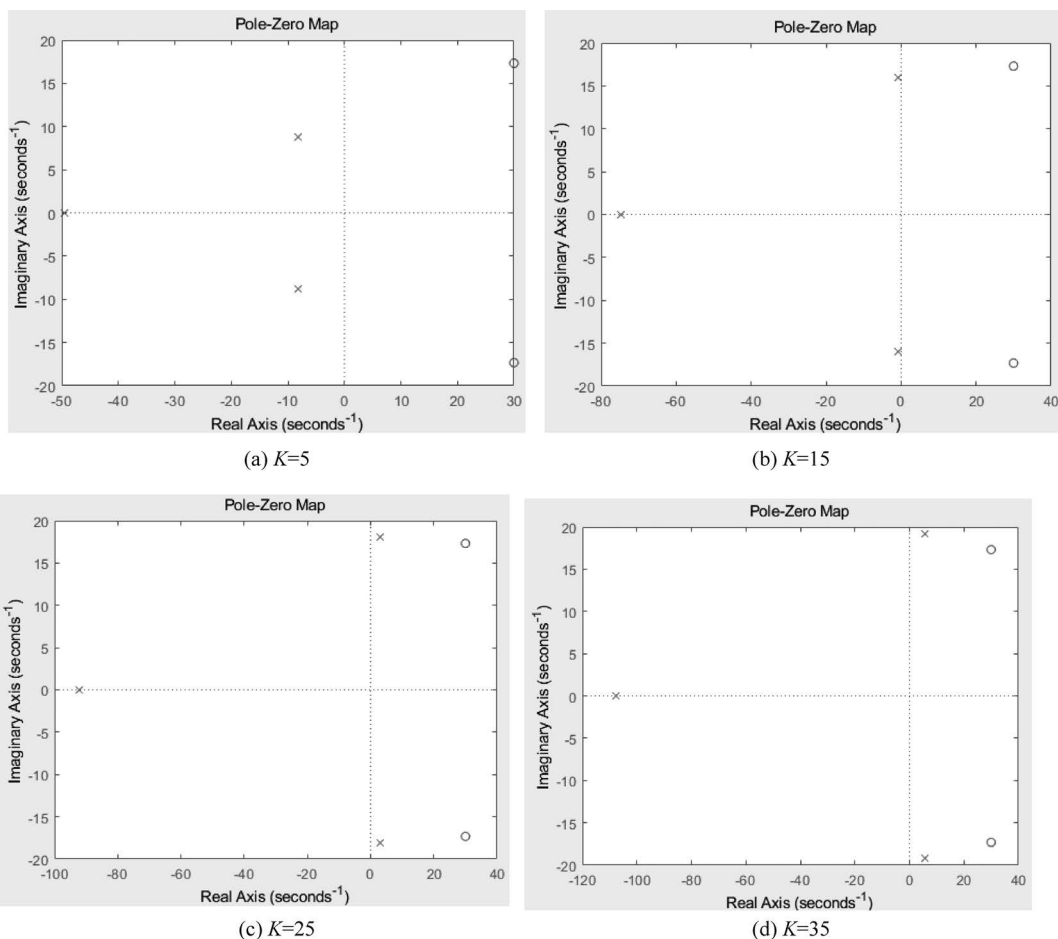


图 5.20 不同 K 值的零极点图

```

3.1331 - 18.1200i
z1 = 30.0000 + 17.3205i
30.0000 - 17.3205i
K = 25 时所判定系统是不稳定的!
p1 = 1.0e+02 *
-1.0755 + 0.0000i
0.0577 + 0.1919i
0.0577 - 0.1919i
z1 = 30.0000 + 17.3205i
30.0000 - 17.3205i
K = 35 时所判定系统是不稳定的!

```

(5) 利用劳斯判据判定系统的稳定性。

程序命令：

```

for K = [5, 15, 25, 35]
    g1 = tf([1 - 60 1200], [1 60 1200]); g2 = tf(K, [1 1]);
    G = g1 * g2; sys = feedback(G, 1);
    p = sys.den{1} % 取闭环的分母系数
    p1 = p; n = length(p);
    if mod(n, 2) == 0
        n1 = n/2;
    else
        n1 = (n + 1)/2; p1 = [p1, 0];
    end
    routh = reshape(p1, 2, n1);
    RouthTable = zeros(n, n1);
    RouthTable(1:2, :) = routh;
    for i = 3:n
        ai = RouthTable(i - 2, 1)/RouthTable(i - 1, 1);
        for j = 1:n1 - 1
            RouthTable(i, j) = RouthTable(i - 2, j + 1) - ai * RouthTable(i - 1, j + 1)
        end
    end
    p2 = RouthTable(:, 1); % 取劳斯阵列第一列
    if p2 > 0
        disp(['K = ', num2str(K), '时所判定系统是稳定的!'])
    else
        disp(['K = ', num2str(K), '时所判定系统是不稳定的!'])
    end
end

```

结果:

K = 5 时所判定系统是稳定的!

K = 15 时所判定系统是稳定的!

K = 25 时所判定系统是不稳定的!

K = 35 时所判定系统是不稳定的!

结论: 以上分别通过三种方法判断不同 K 值系统的稳定性,其结果是一致的。