第 5 章

线性控制系统的计算机辅助分析

若建立起了系统的数学模型，就可以对系统的性质进行分析。对线性系统来说，最重要的性质是其稳定性，在控制理论发展初期，相关的理论成果都是有关系统稳定性的，人们受传统数学理论影响甚至误导，认为高阶系统对应的高阶代数方程不能求出所有特征根，故需通过间接方法判定系统的稳定性，于是出现了各种间接判定方法，如连续系统的Routh表、Hurwitz矩阵法，以及离散系统的Jury判据等。其实有了MATLAB这样的计算机语言，求解系统特征根是轻而易举的。本书中将介绍基于直接求解方法的控制系统稳定性判定方法。此外，状态方程模型的可控性和可观测性都是比较重要的指标，5.1节将对这些性质及相关内容介绍基于MATLAB语言及其控制系统工具箱的定性分析方法，并对系统的状态方程标准型实现及变换方法加以介绍，还将介绍鲁棒控制等领域经常使用的范数测度指标。5.2节介绍线性系统的时域解析分析方法，首先介绍基于传递函数部分分式展开的解析解分析方法，再介绍基于状态方程系统的自治化方法及解析解法。还将引入系统阶跃响应指标的定义与应用。5.3节将介绍连续、离散系统时域响应的数值解法，包括二阶系统的数值解法与物理解释，各种常见输入，如阶跃输入、脉冲输入及任意给定输入下的系统时域响应分析的数值解法，并介绍用MATLAB语言及控制系统工具箱对线性系统进行时域分析的直接方法。5.4节将介绍连续与离散线性系统的根轨迹分析方法，并介绍利用交互方法对其关键的临界增益的求取方法与稳定性分析方法等。5.5节将介绍系统的频域分析方法，对单变量系统来说将介绍用MATLAB语言如何绘制系统的Bode图、Nyquist图及Nichols图等，介绍稳定性分析的间接方法，并进行幅值、相位裕度的分析。对多变量系统来说，可以用5.6节介绍的方法进行Nyquist阵列的分析方法与对角占优分析方法，介绍MATLAB的多变量频域设计工具箱的入门内容，并介绍多变量系统的奇异值分析。

通过本章的介绍,读者将能对已知的线性系统模型进行比较全面的分析,为后面介绍的系统设计打下较好的基础。

5.1 线性系统性质分析

在系统特性研究中,系统的稳定性是最重要的指标,如果系统稳定,则可以进一步分析系统的其他性能;如果系统不稳定,则该系统根本不能直接应用。首先需要引入控制器来使得系统稳定。这种使得系统稳定的方法又称为系统的镇定。本节首先介绍线性系统稳定性的直接判定方法;其次介绍系统的可控性和可观测性等系统性质的分析,并介绍其他的各种标准型实现;最后还将介绍系统的范数测度等指标。

5.1.1 线性系统稳定性的直接判定

前面已经介绍了,连续线性系统的数学描述包括系统的传递函数描述和状态方程描述。通过适当地选择状态变量,则可以容易地得出系统的状态方程模型,在 MATLAB 语言的控制系统工具箱中,直接调用 `ss()` 函数则能立即得出系统的状态方程实现,所以这里统一采用状态方程描述线性系统的模型。

考虑连续线性系统的状态方程模型。

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t) \end{cases} \tag{5-1-1}$$

在某给定信号 $\boldsymbol{u}(t)$ 的激励下,其状态变量的解析解可以表示成

$$\boldsymbol{x}(t) = \mathrm{e}^{\boldsymbol{A}(t-t_0)}\boldsymbol{x}(t_0) + \int_{t_0}^{t} \mathrm{e}^{\boldsymbol{A}(t-\tau)}\boldsymbol{B}\boldsymbol{u}(\tau)\mathrm{d}\tau \tag{5-1-2}$$

可见,如果输入信号 $\boldsymbol{u}(t)$ 为有界信号,若想使得系统的状态变量 $\boldsymbol{x}(t)$ 有界,则要求系统的状态转移矩阵 $\mathrm{e}^{\boldsymbol{A}t}$ 有界,亦即 $\boldsymbol{A}$ 矩阵的所有特征根的实部均为负数。故而可以得出结论:连续线性系统稳定的前提条件是系统状态方程中 $\boldsymbol{A}$ 矩阵的特征根均有负实部。由控制理论可知,系统 $\boldsymbol{A}$ 的特征根和系统的极点是完全一致的,所以若能获得系统的极点,则可以立即判定给定线性系统的稳定性。

在控制理论发展初期,由于没有直接可用的计算机软件能求取高阶多项式的根,所以无法由求根的方法直接判定系统的稳定性,故出现了各种各样的间接方法,例如,在控制理论中著名的 Routh 判据、Hurwitz 判据和 Lyapunov 判据等。对线性系统来说,既然现在有了类似 MATLAB 这样的语言,直接获得系统特征根是轻而易举的事,所以判定连续线性系统稳定性就没有必要再使用间接方法了。

事实上,采用 Routh 判据判定稳定性可能被认为是“知其然不知其所以然”,因为很多人不知道为什么 Routh 表第一列不变号和没有不稳定极点之间的必然联

系；而采用直接判定法的好处在于“知其然知其所以然”，因为，由式（5-1-2）可知，如果有位于 s 的右半平面的极点，$e^{\boldsymbol{A}t}$ 将发散。所以，稳定系统的极点必须不能位于 s 的右半平面。

在MATLAB控制系统工具箱中，求取一个线性定常系统特征根只需用 `p=eig(G)` 函数即可，其中，$\boldsymbol{p}$ 返回系统的全部特征根。不论系统的模型 G 是传递函数、状态方程还是零极点模型，且不论系统是连续的或离散的，都可以用这样简单的命令求解系统的全部特征根，这就使得系统的稳定性判定变得十分容易。另外，由 `pzmap(G)` 函数能用图形的方式绘制出系统所有特征根在 s-复平面上的位置，所以判定连续系统是否稳定只需看一下系统所有极点在 s-复平面上是否均位于虚轴左侧即可。

如果在MATLAB工作空间内已经定义了系统的数学模型 G，则 `pole(G)` 和 `zero(G)` 函数还可以分别求出系统的极点和零点。

再考虑离散状态方程模型

$$\begin{cases} \boldsymbol{x}[(k+1)T] = \boldsymbol{F}\boldsymbol{x}(kT) + \boldsymbol{G}\boldsymbol{u}(kT) \\ \boldsymbol{y}(kT) = \boldsymbol{C}\boldsymbol{x}(kT) + \boldsymbol{D}\boldsymbol{u}(kT) \end{cases} \tag{5-1-3}$$

其状态变量的解析解为

$$\boldsymbol{x}(kT) = \boldsymbol{F}^k\boldsymbol{x}(0) + \sum_{i=0}^{k-1} \boldsymbol{F}^{k-i-1}\boldsymbol{G}\boldsymbol{u}(iT) \tag{5-1-4}$$

可见，若使得系统的状态变量 $\boldsymbol{x}(kT)$ 有界，则要求系统的指数矩阵 $\boldsymbol{F}^k$ 有界，亦即 $\boldsymbol{F}$ 矩阵的所有特征根的模均小于1。故而可以得出结论：离散系统稳定的前提条件是系统状态方程中 $\boldsymbol{F}$ 矩阵所有的特征根的模均小于1，或系统所有的特征根均位于单位圆内，这就是离散系统稳定性的判定条件。

在MATLAB这样的工具出现之前，由于很难求出该矩阵的特征根，所以出现了判定离散系统稳定的Jury判据，其构造比连续系统判定的Routh表更复杂。同样，有了MATLAB这样强有力的计算工具，可以用直接方法求出系统的特征根，观察其位置是否位于单位圆内就可用直接判定离散系统的稳定性，同样还能用 `pzmap(G)` 命令在复平面上绘制系统所有的零极点位置，用图示的方法也可以立即判定离散系统的稳定性，故而没有必要再用复杂的间接方法去判定稳定性了。

更简单地，控制系统工具箱还提供了 `key=isstable(G)` 函数来直接判定系统的稳定性，如果 `key` 为1则稳定，否则不稳定，其中 G 可以为单变量、多变量、连续与离散的线性系统模型，但不能处理带有内部延迟的状态方程模型。

例5-1 假设有开环高阶系统的传递函数

$$G(s) = \frac{10s^4 + 50s^3 + 100s^2 + 100s + 40}{s^7 + 21s^6 + 184s^5 + 870s^4 + 2384s^3 + 3664s^2 + 2496s}$$

试分析单位负反馈闭环系统的稳定性。

解 可以通过下面的MATLAB语句输入系统的传递函数模型并得出单位负反馈构成的闭环系统模型,然后使用三种方法分析系统的稳定性。

```
>> num=[10,50,100,100,40]; den=[1,21,184,870,2384,3664,2496,0];
   G=tf(num,den); GG=feedback(G,1); %输入开环传递函数并得出闭环模型
   eig(GG), pzmap(GG), isstable(GG) %三种不同判定方法
```

闭环系统的极点为$-6.922, -3.65 \pm \mathrm{j}2.302, -2.0633 \pm \mathrm{j}1.7923, -2.635, -0.0158$,因为该系统全部极点都在$s$-左半平面,故此闭环系统是稳定的,`isstable()`函数返回的结果也是1。图5-1中显示的极点位置分布也证实了上面的结论。此外,由于其中一个实极点离虚轴较近,可以认为是主导极点,所以可以断定该系统的性能接近于一阶系统。这样的结论是Routh判据这类间接方法不可能得到的,由此可见直接方法的优势。

其实,采用零极点变换语句`zpk(GG)`可以得出如下的零极点模型。

$$G(s)=\frac{10(s+2)(s+1)(s^2+2s+2)}{(s+6.922)(s+2.635)(s+0.01577)(s^2+4.127s+7.47)(s^2+7.3s+18.62)}$$

例5-2 假设离散受控对象传递函数与控制器模型如下

$$H(z)=\frac{6z^2-0.6z-0.12}{z^4-z^3+0.25z^2+0.25z-0.125},\quad G_{\mathrm{c}}(z)=0.3\frac{z-0.6}{z+0.8}$$

且已知采样周期为$T=0.1\,\mathrm{s}$。试分析单位负反馈下闭环系统的稳定性。

解 闭环系统的特征根及其模可以由下面的MATLAB语句求出。

```
>> num=[6 -0.6 -0.12]; den=[1 -1 0.25 0.25 -0.125];
   H=tf(num,den,'Ts',0.1);    %输入系统的传递函数模型
   z=tf('z','Ts',0.1); Gc=0.3*(z-0.6)/(z+0.8);  %控制器模型
   G=feedback(H*Gc,1);        %闭环系统的模型
   v=abs(eig(G)), pzmap(G), isstable(G) %三种不同判定方法
```

这些闭环特征根的模分别为$\boldsymbol{v}=[1.1644, 1.1644, 0.5536, 0.3232, 0.3232]$。可以看出,由于前两个特征根的模均大于1,`isstable()`返回的结果为0,所以可以判定该闭环系统是不稳定的。闭环系统的零极点还可以由`pzmap(G)`语句绘制出来,如图5-2所示。从图中可以看出,系统含有单位圆外的极点,所以系统是不稳定的。

利用系统零极点变换的语句`zpk(G)`也能容易地得出系统的零极点模型。

$$G(z)=\frac{1.8(z-0.6)(z-0.2)(z+0.1)}{(z-0.5536)(z^2-0.03727z+0.1045)(z^2+0.3908z+1.356)}$$

如果不采用直接方法,而采用像Routh和Jury判据这样的间接判据,则除了系统稳定与否这一判定结论之外,不能得到任何其他的信息。但若采用了直接判定的方法,除了能获得稳定性的信息外,还可以立即看出零极点分布,从而对系统的性能有一个更好的了解。比如对连续系统来说,如果存在距离虚轴特别近的复极点,则可能会使得系统有很强的振荡,对离散系统来说,如果复极点距单位圆较近,也可能得出较强的振荡,这样的定性判定用间接判据是不可能得出的。从这个方面可以看出直接方法和间接方法相比存在的优越性。由于传统观念的影响,很多控制理

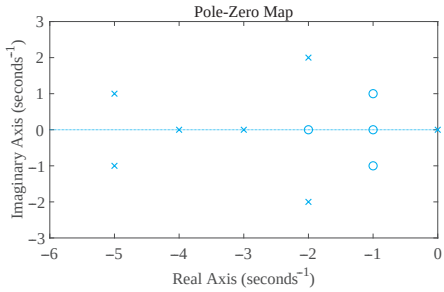


图 5-1　连续系统零极点位置

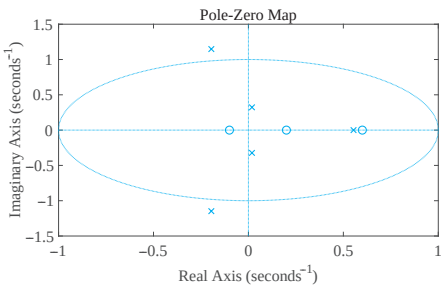


图 5-2　离散闭环系统零极点位置

论教科书至今仍认为直接求取高阶系统特征根的方法是件困难的事[1]，其实，从科学计算现有的发展水平看，直接求取高阶系统特征根是轻而易举的事，其求解过程远比建立 Routh 表或 Jury 表容易得多，况且 Routh 表、Jury 表本身也是工具，同样是借助工具，当然应该使用更直观、有效的工具进行稳定性分析，而没有必要再使用落后的底层工具去分析系统的稳定性了。

5.1.2　内部延迟系统的稳定性分析

前面通过例子介绍过，带有内部延迟系统对应的数学模型是延迟微分方程，其稳定性分析并不容易。这里首先看一个例子，然后试图给出一般的延迟闭环系统稳定性的判定方法。

例 5-3　考虑例 4-18 给出的带有内部延迟的复杂模型，如果这个模型是受控对象，而控制器为 $G_c = 0.3 + 0.15/s$，试判定单位负反馈下闭环系统的稳定性。

解　由于系统模型不是传统意义下的传递函数，所以使用 Routh 表这样的工具是不能分析系统稳定性的，当然可以尝试前面介绍的直接方法，不过延迟系统是不能由传统方法得出零极点位置的，所以 eig() 等函数不能判定系统的稳定性。现在尝试 isstable() 函数，看看能得出什么结果。

```
>> s=tf('s'); G=(1+3*exp(-s)/(s+1))/(s+1);
   Gc=0.3+0.15/s; isstable(feedback(G*Gc,1))
```

遗憾的是，该函数得出如下的错误信息“isstable() 函数不能分析带有内部延迟系统的稳定性，可以用 step() 或 impulse() 函数分析稳定性”。

除了该函数建议的仿真方法外，这里将介绍两种直接分析方法，一种是通过 Padé 近似的稳定性近似判定方法，另一种是特征方程的数值求解方法。具体的判定方法将通过一个演示例子给出。

例 5-4　重新考虑例 5-3 未解决的问题，试判定闭环系统的稳定性。

解　如果对延迟项采用二阶 Padé 近似，则可以给出下面的语句，得出的 key 值为 1，说明闭环系统是稳定的。如果采用其他不同的阶次，也可以得出一致的结果。

```
>> s=tf('s'); G=(1+3*exp(-s)/(s+1))/(s+1);
   Gc=0.3+0.15/s; key=isstable(feedback(pade(G*Gc,2),1))
```

3.2.3 节介绍过 more_sols() 函数,该函数可以求取非线性方程全部的根,可以尝试这个函数得出变换传递函数全部的奇点(即特征方程全部的根),由根的分布判定闭环系统的稳定性。

在分析稳定性之前,首先用符号运算提取特征方程。

```
>> syms s; G=(1+3*exp(-s)/(s+1))/(s+1);
   Gc=0.3+0.15/s; G1=feedbacksym(G*Gc,1), [n,d]=numden(G1)
```

得出的分子与分母表达式为

$$n = 3(2s+1)(\mathrm{e}^{s}(1+s)+3),\ d = 18s+9+(3+29s+46s^{2}+20s^{3})\mathrm{e}^{s}$$

如果将分子与分母同时乘以 e^{-s},则可以用下面的语句描述分母多项式方程,并试图得出方程的全部特征根。

```
>> f=@(s)(18*s+9)*exp(-s)+(3+29*s+46*s^2+20*s^3);
   more_sols(f,zeros(1,1,0),50+10000i)  % 大范围求复数根
   xx=X(:); plot(real(xx),imag(xx),'x') % 绘制闭环特征根的分布
```

由本书提供的求解程序可以求解含有超越函数的闭环特征方程。经过反复试算,得出方程全部特征根共 113 个,总耗时 213 s。系统闭环特征根的分布如图 5-3 所示。可见,所有的特征根都位于 s 的左半平面,所以闭环系统是稳定的。

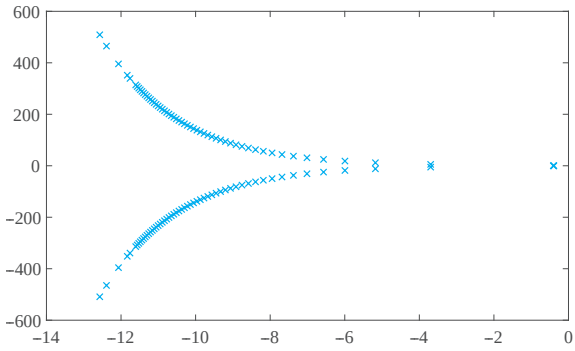


图 5-3 闭环系统的特征根分布

5.1.3 线性反馈系统的内部稳定性分析

在反馈控制系统的分析中,为了得到更好的控制效果,仅分析系统的输入输出稳定性是不够的,因为这样的稳定性分析只能保证由稳定输入激励下的输出信号的有界性,但不能保证系统的内部信号都是有界的。若系统的内部信号变成无界的,即使原系统稳定,也将破坏原系统的物理结构。

考虑图 5-4 中所示的反馈系统结构,可见这个结构是典型反馈控制系统结构的扩展,在系统中还带有扰动信号。在这个系统结构下,扰动信号 d 经常称作外部扰动信号,而 n 常称为量测噪声。

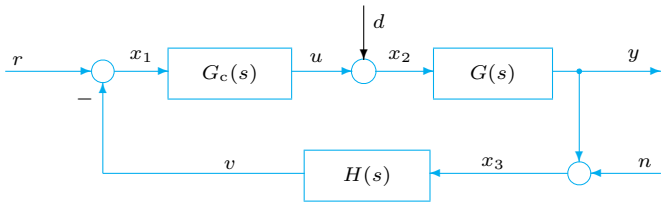


图5-4 带有扰动的线性反馈控制系统

如果图5-4中所示的系统从输入信号 (r,d,n) 到内部输出信号 (x_1,x_2,x_3) 的所有9个闭环传递函数都是稳定的，则称该系统是内部稳定的。

可以证明，这9个传递函数可以表示成

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \frac{1}{1+G(s)G_c(s)H(s)} \begin{bmatrix} 1 & -G(s)H(s) & -H(s) \\ G_c(s) & 1 & -G_c(s)H(s) \\ G(s)G_c(s) & G(s) & 1 \end{bmatrix} \begin{bmatrix} r \\ d \\ n \end{bmatrix} \quad (5\text{-}1\text{-}5)$$

逐一去判定每个子传递函数的稳定性无疑是很烦琐的，所以可以根据内部稳定性定理，用简单方法直接判定。该定理为：闭环系统内部稳定的充要条件为

（1）传递函数 $1+H(s)G(s)G_c(s)$ 没有 $\mathscr{R}[s]\geqslant 0$ 的零点。

（2）乘积 $H(s)G(s)G_c(s)$ 中没有满足 $\mathscr{R}[s]\geqslant 0$ 的零极点对消。

基于上述条件，可以对这个定理进行简化。仔细观察定理中的条件，不难看出，其中第一个条件等效于闭环系统的稳定性，所以只需判定第二个条件，该条件判定起来也不困难。其实，内部稳定性的定义及判定定理可以直接拓展到多变量系统及离散系统。这样，可以编写出判定反馈系统内部稳定性的函数如下：

```
function key=intstable(G,Gc,H)
GG=minreal(feedback(G*Gc,H)); Go=H*G*Gc; Go1=minreal(Go);
p=eig(GG); z0=eig(Go); z1=eig(Go1);
zz=setdiff(z0,z1); %找开环对消极点
if (G.Ts>0)          %离散系统判定
   key=any(abs(p)>1); if key==0, key=2*any(abs(zz)>1); end
else, key=any(real(p)>0);
if key==0, key=2*any(real(zz)>0); end, end
```

若闭环系统不稳定，则返回key的值为1，若稳定系统内部不稳定，则返回的key值为2，否则返回key的值为0。此函数同样适用于多变量系统和离散系统。

5.1.4 线性系统的线性相似变换

前面已经介绍过，由于状态变量可以有不同的选择，故系统的状态方程实现将不同，这里将研究这些状态方程之间的关系。

假设存在一个非奇异矩阵 $\boldsymbol{T}$，且定义了一个新的状态变量向量 $\boldsymbol{z}$ 使得 $\boldsymbol{z}=\boldsymbol{T}^{-1}\boldsymbol{x}$，

这样关于新状态变量 $\boldsymbol{z}$ 的状态方程模型可以写成

$$\begin{cases} \dot{\boldsymbol{z}}(t) = \boldsymbol{A}_{\mathrm{t}}\boldsymbol{z}(t) + \boldsymbol{B}_{\mathrm{t}}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \boldsymbol{C}_{\mathrm{t}}\boldsymbol{z}(t) + \boldsymbol{D}_{\mathrm{t}}\boldsymbol{u}(t), \end{cases} \quad \text{且 } \boldsymbol{z}(0) = \boldsymbol{T}^{-1}\boldsymbol{x}(0) \tag{5-1-6}$$

式中，$\boldsymbol{A}_{\mathrm{t}} = \boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T}, \boldsymbol{B}_{\mathrm{t}} = \boldsymbol{T}^{-1}\boldsymbol{B}, \boldsymbol{C}_{\mathrm{t}} = \boldsymbol{C}\boldsymbol{T}, \boldsymbol{D}_{\mathrm{t}} = \boldsymbol{D}$。在矩阵 $\boldsymbol{T}$ 下的状态变换称为相似性变换，而 $\boldsymbol{T}$ 又称为变换矩阵。

控制系统工具箱中提供了 ss2ss() 来完成状态方程模型的相似性变换，该函数的调用格式为 G_1=ss2ss(G,$\boldsymbol{T}$)，其中，G 为原始的状态方程模型，$\boldsymbol{T}$ 为变换矩阵，在 $\boldsymbol{T}$ 下的变换结果由 G_1 变元返回。注意，在本函数调用中输入和输出的变元都是状态方程对象，而不是其他对象。

例 5-5　在实际应用中，变换矩阵 $\boldsymbol{T}$ 可以任意选择，只要它为非奇异矩阵即可。假设已知系统的状态方程模型为

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -24 & -50 & -35 & -10 \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t) \\ y(t) = [24 \quad 24 \quad 7 \quad 1]\boldsymbol{x}(t) \end{cases}$$

若选择一个反对角矩阵，使得反对角线上的元素均为 1，而其余元素都为 0，试得出该系统相似变换后的结果。

解　在这一变换矩阵下新的状态方程模型可以由下面的 MATLAB 语句得出

```
>> A=[0 1 0 0; 0 0 1 0; 0 0 0 1; -24 -50 -35 -10];
   G1=ss(A,[0;0;0;1],[24 24 7 1],0); % 系统状态方程模型
   T=fliplr(eye(4)); G2=ss2ss(G1,T)  % 系统的线性相似变换结果
```

得出的变换后系统模型如下，其作用是对状态变量作逆序排列。

$$\begin{cases} \dot{\boldsymbol{z}}(t) = \begin{bmatrix} -10 & -35 & -50 & -24 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{z}(t) + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 1 & 7 & 24 & 24 \end{bmatrix} \boldsymbol{z}(t) \end{cases}$$

事实上，这样得出的状态方程模型和很多教科书 [2] 中定义的可控标准型一致。

5.1.5　线性系统的可控性分析

线性系统的可控性和可观测性是基于状态方程的控制理论的基础，可控性和可观测性的概念是美国学者 Rudolf Emil Kálmán（1930–2016，经常写作 Kalman）于 1960 年提出的 [3]，这些性质为系统的状态反馈设计、观测器的设计等提供了依据。假设系统由状态方程 $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D})$ 给出，对任意的初始时刻 t_0，如果状态空间中任一状态 $x_i(t)$ 可以从初始状态 $x_i(t_0)$ 处，由有界的输入信号 $\boldsymbol{u}(t)$ 的驱动下，在

有限时间t_n内能够到达任意预先指定的状态$x_i(t_n)$，则称此状态是可控的。若系统中所有状态都是可控的，则称该系统为完全可控的系统。

通俗点说，系统的可控性就是指系统内部的状态是不是可以由外部输入信号控制的性质，对线性时不变系统来说，如果系统某个状态可控，则可以由外部信号任意控制。

1. 线性系统的可控性判定

可以构造一个可控性判定矩阵

$$\boldsymbol{T}_c = \left[\boldsymbol{B}, \boldsymbol{AB}, \boldsymbol{A}^2\boldsymbol{B}, \cdots, \boldsymbol{A}^{n-1}\boldsymbol{B}\right] \tag{5-1-7}$$

若矩阵$\boldsymbol{T}_c$是满秩矩阵，则系统称为完全可控的。如果该矩阵不是满秩矩阵，则它的秩为系统的可控状态的个数。在MATLAB下求一个矩阵的秩是再容易不过的事，如果已知矩阵为$\boldsymbol{T}$，则用MATLAB提供的可靠算法用`rank(T)`即可以求出矩阵的秩。再将得出的秩和系统状态变量的个数相比较，就可以判定系统的可控性。

构造系统的可控性判定矩阵用MATLAB也很容易，用`Tc=ctrb(A,B)`函数就可以立即建立起可控性判定矩阵$\boldsymbol{T}_c$。用最底层的MATLAB命令也可以直接建立可控性判定矩阵。这里给出的判定方法既适用于连续系统，也适用于离散系统。下面将通过例子演示系统可控性判定矩阵建立和系统可控性判定的问题求解。

例5-6 给定离散系统状态方程模型如下，试判定其可控性。

$$\boldsymbol{x}[(k+1)T] = \begin{bmatrix} -2.2 & -0.7 & 1.5 & -1 \\ 0.2 & -6.3 & 6 & -1.5 \\ 0.6 & -0.9 & -2 & -0.5 \\ 1.4 & -0.1 & -1 & -3.5 \end{bmatrix} \boldsymbol{x}(kT) + \begin{bmatrix} 6 & 9 \\ 4 & 6 \\ 4 & 4 \\ 8 & 4 \end{bmatrix} \boldsymbol{u}(kT)$$

解 可以通过下面的MATLAB语句将系统的$\boldsymbol{A}$和$\boldsymbol{B}$矩阵输入到MATLAB的工作空间，这样就可以用下面的语句直接判定系统的可控性。

```
>> A=[-2.2,-0.7,1.5,-1; 0.2,-6.3,6,-1.5; ...
      0.6,-0.9,-2,-0.5; 1.4,-0.1,-1,-3.5];
   B=[6,9; 4,6; 4,4; 8,4]; Tc=ctrb(A,B) %建立可控性判定矩阵
   rank(Tc) %判定系统的可控性,因为可得秩为3,所以系统不可控
```

生成如下的可控性判定矩阵，可以根据其秩来判定系统的可控性。

$$\boldsymbol{T}_c = \begin{bmatrix} 6 & 9 & -18 & -22 & 54 & 52 & -162 & -118 \\ 4 & 6 & -12 & -18 & 36 & 58 & -108 & -202 \\ 4 & 4 & -12 & -10 & 36 & 26 & -108 & -74 \\ 8 & 4 & -24 & -6 & 72 & 2 & -216 & 34 \end{bmatrix}$$

系统完全可控的另外判定方式是，系统的可控Gram矩阵为非奇异矩阵。系统的可控Gram矩阵由下式定义

$$\boldsymbol{L}_c = \int_0^\infty \mathrm{e}^{-\boldsymbol{A}t}\boldsymbol{B}\boldsymbol{B}^{\mathrm{T}}\mathrm{e}^{-\boldsymbol{A}^{\mathrm{T}}t}\mathrm{d}t \tag{5-1-8}$$

当然,看起来求解系统的可控 Gram 矩阵也并非简单的事,可以证明,系统的可控 Gram 矩阵为对称矩阵,是下面的 Lyapunov 方程的解。

$$\boldsymbol{A}\boldsymbol{L}_{\mathrm{c}}+\boldsymbol{L}_{\mathrm{c}}\boldsymbol{A}^{\mathrm{T}}=-\boldsymbol{B}\boldsymbol{B}^{\mathrm{T}} \tag{5-1-9}$$

在 MATLAB 环境中用 $\boldsymbol{L}_{\mathrm{c}}$=lyap($\boldsymbol{A}$,$\boldsymbol{B}$*$\boldsymbol{B}$') 命令就能直接求出 Lyapunov 方程的解,如果调用该函数不能求出方程的解,则该系统不完全可控。控制系统的可控 Gram 矩阵还可以由 G_{c}=gram(G,'c') 直接求出来。离散系统的 Gram 矩阵是离散 Lyapunov 方程的解,但在函数的调用格式上与连续系统完全一致。

例 5-7 已知离散系统模型如下,采样周期为 $T=0.1\,\mathrm{s}$,试计算系统的可控 Gram 矩阵。

$$H(z)=\frac{6z^2-0.6z-0.12}{z^4-z^3+0.25z^2+0.25z-0.125}$$

解 可以用下面的语句将其输入到 MATLAB 工作空间,然后通过相应的函数调用直接求出系统的可控性 Gram 矩阵。

```
>> num=[6 -0.6 -0.12]; den=[1 -1 0.25 0.25 -0.125];
   H=tf(num,den,'Ts',0.1)   %输入并显示系统的传递函数模型
   Lc=gram(ss(H),'c')       %先获得状态方程模型,再求可控 Gram 矩阵
```

通过运算可以得出可控 Gram 矩阵为

$$\boldsymbol{L}_{\mathrm{c}}=\begin{bmatrix}10.765 & 15.754 & 7.3518 & 0\\ 15.754 & 43.061 & 31.508 & 3.6759\\ 7.3518 & 31.508 & 43.061 & 7.8769\\ 0 & 3.6759 & 7.8769 & 2.6913\end{bmatrix}$$

2. 可控性阶梯分解

对于不完全可控的系统,还可以对之进行可控性阶梯分解,即构造一个状态变换矩阵 $\boldsymbol{T}$,就可以将系统的状态方程 $(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D})$ 变换成如下形式

$$\boldsymbol{A}_{\mathrm{c}}=\begin{bmatrix}\widehat{\boldsymbol{A}}_{\bar{\mathrm{c}}} & \boldsymbol{0}\\ \widehat{\boldsymbol{A}}_{21} & \widehat{\boldsymbol{A}}_{\mathrm{c}}\end{bmatrix},\quad \boldsymbol{B}_{\mathrm{c}}=\begin{bmatrix}\boldsymbol{0}\\ \widehat{\boldsymbol{B}}_{\mathrm{c}}\end{bmatrix},\quad \boldsymbol{C}_{\mathrm{c}}=[\widehat{\boldsymbol{C}}_{\bar{\mathrm{c}}},\widehat{\boldsymbol{C}}_{\mathrm{c}}] \tag{5-1-10}$$

该形式称为系统的可控阶梯分解形式,这样就可以将原系统的不可控子空间 $(\widehat{\boldsymbol{A}}_{\bar{\mathrm{c}}},\boldsymbol{0},\widehat{\boldsymbol{C}}_{\bar{\mathrm{c}}})$ 和可控子空间 $(\widehat{\boldsymbol{A}}_{\mathrm{c}},\widehat{\boldsymbol{B}}_{\mathrm{c}},\widehat{\boldsymbol{C}}_{\mathrm{c}})$ 直接分离出来。构造这样的变换矩阵不是简单的事,但可以借用 MATLAB 中的现成函数 ctrbf() 对状态方程模型进行这样的阶梯分解 [$\boldsymbol{A}_{\mathrm{c}}$,$\boldsymbol{B}_{\mathrm{c}}$,$\boldsymbol{C}_{\mathrm{c}}$,$\boldsymbol{T}_{\mathrm{c}}$]=ctrbf($\boldsymbol{A}$,$\boldsymbol{B}$,$\boldsymbol{C}$),该函数就可以自动生成相似变换矩阵 $\boldsymbol{T}_{\mathrm{c}}$,将原系统模型直接变换成可控性阶梯分解模型。如果原来系统的状态方程模型是完全可控的,则此分解不必进行。

例 5-8 考虑例 5-6 中给出的不完全可控的系统模型,试得出其可控阶梯形式。

解 可以通过下面的语句对之进行分解,得出可控性阶梯分解形式。

```
>> A=[-2.2,-0.7,1.5,-1; 0.2,-6.3,6,-1.5;...
       0.6,-0.9,-2,-0.5; 1.4,-0.1,-1,-3.5];
   B=[6,9; 4,6; 4,4; 8,4]; C=[1 2 3 4]; [Ac,Bc,Cc,Tc]=ctrbf(A,B,C);
```

得出的可控阶梯标准型如下,这时,左上角的子空间是不可控的。

$$\hat{\boldsymbol{x}}[(k+1)T]=\left[\begin{array}{c|ccc}-4 & 0 & 0 & 0\\ \hline -4.638 & -3.823 & -0.5145 & -0.127\\ -3.637 & 0.1827 & -3.492 & -0.1215\\ -4.114 & -1.888 & 1.275 & -2.685\end{array}\right]\hat{\boldsymbol{x}}(kT)+\left[\begin{array}{cc}0 & 0\\ \hline 0 & 0\\ 2.754 & -2.575\\ -11.15 & -11.93\end{array}\right]\boldsymbol{u}(kT)$$

5.1.6 线性系统的可观测性分析

假设系统由状态方程$(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D})$给出，对任意的初始时刻$t_0$，如果状态空间中任一状态$x_i(t)$在任意有限时刻$t_\mathrm{n}$的状态$x_i(t_\mathrm{n})$可以由输出信号在这一时间区间内$t\in[t_0,t_\mathrm{n}]$的值精确地确定出来，则称此状态是可观测的。如果系统中所有的状态都是可观测的，则称该系统为完全可观测的系统。

类似于系统的可控性，系统的可观测性就是指系统内部的状态是不是可以由系统的输入、输出信号重建起来的性质。对线性时不变系统来说，如果系统某个状态可观测，则可以由输入、输出信号重建出来。

从定义判定系统的可观测性是很烦琐的，还可以构造起可观测性判定矩阵。

$$\boldsymbol{T}_\mathrm{o}=\left[\begin{array}{c}\boldsymbol{C}\\ \boldsymbol{C}\boldsymbol{A}\\ \boldsymbol{C}\boldsymbol{A}^2\\ \vdots\\ \boldsymbol{C}\boldsymbol{A}^{n-1}\end{array}\right]\tag{5-1-11}$$

该矩阵的秩为系统的可观测状态数。如果该矩阵满秩，则系统是完全可观测的，即系统的所有状态都可以由输入、输出信号重建。

由控制理论可知，系统的可观测性问题和系统的可控性问题是对偶关系，若想研究系统$(\boldsymbol{A},\boldsymbol{C})$的可观测性问题，可以将其转换成研究$(\boldsymbol{A}^\mathrm{T},\boldsymbol{C}^\mathrm{T})$系统的可控性问题，故前面所述的可控性分析的全部方法均可以扩展到系统的可观测性研究中。

当然，可观测性分析也有自己的相应函数，如对应于可控性的函数`ctrb()`和`ctrbf()`，有`obsv()`和`obsvf()`，对应`gram(`G`,'c')`有`gram(`G`,'o')`等，也可以利用这些函数直接进行可观测性分析与变换，系统可观测性Gram矩阵定义为

$$\boldsymbol{L}_\mathrm{o}=\int_0^\infty \mathrm{e}^{-\boldsymbol{A}^\mathrm{T}t}\boldsymbol{C}^\mathrm{T}\boldsymbol{C}\mathrm{e}^{-\boldsymbol{A}t}\mathrm{d}t\tag{5-1-12}$$

该矩阵满足Lyapunov方程。

$$\boldsymbol{A}^\mathrm{T}\boldsymbol{L}_\mathrm{o}+\boldsymbol{L}_\mathrm{o}\boldsymbol{A}=-\boldsymbol{C}^\mathrm{T}\boldsymbol{C}\tag{5-1-13}$$

5.1.7 Kalman规范分解

从上面的叙述可以看出，通过可控性阶梯分解则可以将可控子空间和不可控子空间分离出来，同样进行可观测性阶梯分解则可以将可观测子空间和不可观测子空间分离出来，这样就可能组合出4个子空间。如果先对系统进行可控性阶梯分

解,再对结果进行可观测性阶梯分解,则可以得出下面的规范形式。

$$\begin{cases} \dot{\boldsymbol{z}}(t)=\begin{bmatrix} \widehat{\boldsymbol{A}}_{\bar{\mathrm{c}},\bar{\mathrm{o}}} & \widehat{\boldsymbol{A}}_{1,2} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \widehat{\boldsymbol{A}}_{\bar{\mathrm{c}},\mathrm{o}} & \boldsymbol{0} & \boldsymbol{0} \\ \widehat{\boldsymbol{A}}_{3,1} & \widehat{\boldsymbol{A}}_{3,2} & \widehat{\boldsymbol{A}}_{\mathrm{c},\bar{\mathrm{o}}} & \widehat{\boldsymbol{A}}_{3,4} \\ \boldsymbol{0} & \widehat{\boldsymbol{A}}_{4,2} & \boldsymbol{0} & \widehat{\boldsymbol{A}}_{\mathrm{c},\mathrm{o}} \end{bmatrix}\boldsymbol{z}(t)+\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \widehat{\boldsymbol{B}}_{\mathrm{c},\bar{\mathrm{o}}} \\ \widehat{\boldsymbol{B}}_{\mathrm{c},\mathrm{o}} \end{bmatrix}\boldsymbol{u}(t) \\ \boldsymbol{y}(t)=\begin{bmatrix} \boldsymbol{0} & \widehat{\boldsymbol{C}}_{\bar{\mathrm{c}},\mathrm{o}} & \boldsymbol{0} & \widehat{\boldsymbol{C}}_{\mathrm{c},\mathrm{o}} \end{bmatrix}\boldsymbol{z}(t) \end{cases} \tag{5-1-14}$$

其中,子空间 $(\widehat{\boldsymbol{A}}_{\bar{\mathrm{c}},\bar{\mathrm{o}}},\boldsymbol{0},\boldsymbol{0})$ 为既不可控,又不可观测的子空间,$(\widehat{\boldsymbol{A}}_{\bar{\mathrm{c}},\mathrm{o}},\boldsymbol{0},\widehat{\boldsymbol{C}}_{\bar{\mathrm{c}},\mathrm{o}})$ 为不可控但可观测的子空间,$(\widehat{\boldsymbol{A}}_{\mathrm{c},\bar{\mathrm{o}}},\widehat{\boldsymbol{B}}_{\mathrm{c},\bar{\mathrm{o}}},\boldsymbol{0})$ 和 $(\widehat{\boldsymbol{A}}_{\mathrm{c},\mathrm{o}},\widehat{\boldsymbol{B}}_{\mathrm{c},\mathrm{o}},\widehat{\boldsymbol{C}}_{\mathrm{c},\mathrm{o}})$ 分别为可控但不可观测的子空间和既不可控又可观测的子空间。这样的分解又称为 Kalman 分解。在实际系统分析中,更关心的是既可控又可观测的子空间,该子空间事实上就是前面提及的最小实现模型。

5.1.8 系统状态方程标准型的MATLAB求解

单变量系统常用的状态空间实现有可控标准型、可观测标准型和 Jordan 标准型实现,多变量系统又经常需要变换成 Luenberger 标准型。

1. 单变量系统的标准型

单变量系统常用的标准型是可控标准型、可观测标准型和 Jordan 标准型实现,若系统的传递函数模型由下式给出

$$G(s)=\frac{b_0s^n+\hat{b}_1s^{n-1}+\hat{b}_2s^{n-2}+\cdots+\hat{b}_{n-1}s+\hat{b}_n}{s^n+a_1s^{n-1}+a_2s^{n-2}+\cdots+a_{n-1}s+a_n} \tag{5-1-15}$$

则可以将其改写成

$$G(s)=b_0+\frac{b_1s^{n-1}+b_2s^{n-2}+\cdots+b_{n-1}s+b_n}{s^n+a_1s^{n-1}+a_2s^{n-2}+\cdots+a_{n-1}s+a_n} \tag{5-1-16}$$

式中 $b_i=\hat{b}_i-b_0a_i$。系统可控标准型的一般形式为

$$\begin{cases} \dot{\boldsymbol{x}}=\boldsymbol{A}_{\mathrm{c}}\boldsymbol{x}+\boldsymbol{B}_{\mathrm{c}}u \\ y=\boldsymbol{C}_{\mathrm{c}}\boldsymbol{x}+D_{\mathrm{c}}u \end{cases}\Longrightarrow\begin{cases} \dot{\boldsymbol{x}}=\begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & \cdots & -a_1 \end{bmatrix}\boldsymbol{x}+\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}u \\ y=[b_n \quad b_{n-1} \quad \cdots \quad b_1]\boldsymbol{x}+b_0u \end{cases} \tag{5-1-17}$$

可观测标准型的一般形式为

$$\begin{cases} \dot{\boldsymbol{x}}=\boldsymbol{A}_{\mathrm{o}}\boldsymbol{x}+\boldsymbol{B}_{\mathrm{o}}u \\ y=\boldsymbol{C}_{\mathrm{o}}\boldsymbol{x}+D_{\mathrm{o}}u \end{cases}\Longrightarrow\begin{cases} \dot{\boldsymbol{x}}=\begin{bmatrix} 0 & 0 & \cdots & 0 & -a_n \\ 1 & 0 & \cdots & 0 & -a_{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -a_1 \end{bmatrix}\boldsymbol{x}+\begin{bmatrix} b_n \\ b_{n-1} \\ \vdots \\ b_1 \end{bmatrix}u \\ y=[0 \quad 0 \quad \cdots \quad 0 \quad 1]\boldsymbol{x}+b_0u \end{cases} \tag{5-1-18}$$

可见，可控标准型和可观测标准型互为对偶形式，即

$$\boldsymbol{A}_{\mathrm{c}}=\boldsymbol{A}_{\mathrm{o}}^{\mathrm{T}},\quad \boldsymbol{B}_{\mathrm{c}}=\boldsymbol{C}_{\mathrm{o}}^{\mathrm{T}},\quad \boldsymbol{C}_{\mathrm{c}}=\boldsymbol{B}_{\mathrm{o}}^{\mathrm{T}},\quad D_{\mathrm{c}}=D_{\mathrm{o}} \tag{5-1-19}$$

模型G的对偶状态方程模型在MATLAB下可以用G'命令直接得出。可控标准型和可观测标准型可以由下面给出的`sscanform()`函数直接获得。

```
function Gs=sscanform(G,type)
switch type
   case 'ctrl', G=tf(G); Gs=[];
      G.num{1}=G.num{1}/G.den{1}(1); %传递函数首一化
      G.den{1}=G.den{1}/G.den{1}(1); b0=G.num{1}(1);
      G1=G; G1.ioDelay=0; G1=G1-b0;
      num=G1.num{1}; den=G1.den{1}; n=length(den)-1;
      A=[zeros(n-1,1) eye(n-1); -den(end:-1:2)];
      B=[zeros(n-1,1);1]; C=num(end:-1:2); D=b0;
      Gs=ss(A,B,C,D,'Ts',G.Ts,'ioDelay',G.ioDelay);
   case 'obsv', Gs=sscanform(G,'ctrl').';
   otherwise
      error('Only options ''ctrl'' and ''obsv'' are allowed.')
end
```

Jordan标准型则是根据系统矩阵Jordan变换构成的一种标准型形式。假设系统矩阵$\boldsymbol{A}$的特征根为$\lambda_1,\lambda_2,\cdots,\lambda_m$，第$i$个特征根$\lambda_i$对应的特征向量为$\boldsymbol{v}_i$，则

$$\boldsymbol{A}\boldsymbol{v}_i=\lambda_i\boldsymbol{v}_i,\quad i=1,2,\cdots,m \tag{5-1-20}$$

矩阵$\boldsymbol{A}$对应的模态矩阵$\boldsymbol{\Lambda}$定义为

$$\boldsymbol{\Lambda}=\boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T}=\begin{bmatrix}\boldsymbol{J}_1 & & & \\ & \boldsymbol{J}_2 & & \\ & & \ddots & \\ & & & \boldsymbol{J}_k\end{bmatrix} \tag{5-1-21}$$

其中$\boldsymbol{J}_i$称为Jordan矩阵。`canon()`函数可以直接获得Jordan标准型。

2. 多变量系统的Luenberger标准型

多变量系统一种重要的可控标准型实现是Luenberger标准型，其具体实现方法是，构造可控性判定矩阵，并按照下面的顺序构成一个矩阵$\boldsymbol{S}$[4]：

$$\boldsymbol{S}=\left[\boldsymbol{b}_1,\boldsymbol{A}\boldsymbol{b}_1,\cdots,\boldsymbol{A}^{\sigma_1-1}\boldsymbol{b}_1,\boldsymbol{b}_2,\cdots,\boldsymbol{A}^{\sigma_2-1}\boldsymbol{b}_2,\cdots,\boldsymbol{A}^{\sigma_p-1}\boldsymbol{b}_p\right] \tag{5-1-22}$$

其中，σ_i是能保证前面各列线性无关的最大指数值，亦即最大可控性指数，取该矩阵的前n列就可以构成一个$n\times n$的方阵$\boldsymbol{L}$。如果这样构成的满秩矩阵不足n列，亦即多变量系统不是完全可控，则可以在后面补足能够使得$\boldsymbol{L}$为满秩方阵的列，可以通过添补随机数的方式构造该矩阵。该矩阵求逆，则可以按照如下的方式提取出相

关各行

$$\boldsymbol{L}^{-1}=\begin{bmatrix}\boldsymbol{l}_1^{\mathrm{T}}\\ \vdots\\ \boldsymbol{l}_{\sigma_1}^{\mathrm{T}}\\ \vdots\\ \boldsymbol{l}_{\sigma_1+\sigma_2}^{\mathrm{T}}\\ \vdots\end{bmatrix}\begin{matrix}\\ \\ \leftarrow \text{提取此行}\\ \\ \leftarrow \text{提取此行}\\ \\ \end{matrix} \tag{5-1-23}$$

这样,依照下面的方法可以构造出变换矩阵的逆矩阵 $\boldsymbol{T}^{-1}$。

$$\boldsymbol{T}^{-1}=\begin{bmatrix}\boldsymbol{l}_{\sigma_1}^{\mathrm{T}}\\ \vdots\\ \boldsymbol{l}_{\sigma_1}^{\mathrm{T}}\boldsymbol{A}^{\sigma_1-1}\\ \vdots\\ \boldsymbol{l}_{\sigma_1+\sigma_2}^{\mathrm{T}}\boldsymbol{A}^{\sigma_2-1}\\ \vdots\end{bmatrix} \tag{5-1-24}$$

通过变换矩阵 $\boldsymbol{T}$ 对原系统进行相似变换,即可以得出 Luenberger 标准型。前面介绍的方法很适合用 MATLAB 语言直接实现,根据算法,可以编写出如下的函数来生成变换矩阵 $\boldsymbol{T}$。

```
function T=luenberger(A,B)
n=size(A,1); p=size(B,2); S=[]; sigmas=[]; k=1;
for i=1:p
   for j=0:n-1, S=[S,A^j*B(:,i)];
      if rank(S)==k, k=k+1;
      else, sigmas(i)=j-1; S=S(:,1:end-1); break; end
   end
   if k>n, break; end
end
k=k-1; %如果不是完全可控,则用随机数补足满秩矩阵
if k<n, while rank(S)=n, S(:,k+1:n)=rand(n,n-k); end, end
L=inv(S); iT=[];
for i=1:p, for j=0:sigmas(i)
   iT=[iT; L(i+sum(sigmas(1:i)),:)*A^j];
end, end
if k<n, iT(k+1:n,:)=L(k+1:end,:); end  %不可控时补足满秩矩阵
T=inv(iT);                             %构造变换矩阵
```

这样,状态方程的各种标准型可以由下面的函数直接获得[5]。

```
Gs=sscanform(G,'ctrl')      %求取可控标准型
Gs=sscanform(G,'obsv')      %求取可观测标准型
[Gs,T]=canon(G,'modal')     %求取 Jordan 标准型的函数,T 为变换矩阵
```

```
T=luenberger(A,B)        % 多变量系统Luenberger标准型的转换矩阵
```

例5-9 试求取传递函数 $G(s)=\dfrac{6s^4+2s^2+8s+10}{2s^4+6s^2+4s+8}$ 的可观测标准型实现。

解 该问题可以由sscanform()函数直接求解。

```
>> num=[6 0 2 8 10]; den=[2 0 6 4 8]; %分子多项式和分母多项式
   G=tf(num,den); Gs=sscanform(G,'obsv')
```

可以得出可观测标准型为

$$\begin{cases}\dot{\boldsymbol{z}}(t)=\begin{bmatrix}0&0&0&-4\\1&0&0&-2\\0&1&0&-3\\0&0&1&0\end{bmatrix}\boldsymbol{z}(t)+\begin{bmatrix}-7\\-2\\-8\\0\end{bmatrix}u(t)\\ y(t)=\begin{bmatrix}0&0&0&1\end{bmatrix}\boldsymbol{z}(t)+3u(t)\end{cases}$$

可见，由给定的系统传递函数模型可以直接得出系统的可观测性标准型。求取系统的可控标准型和Jordan标准型也同样容易，调用相应的函数即可。

例5-10 试得出下面给出的状态方程模型的Luenberger标准型。

$$\dot{\boldsymbol{x}}(t)=\begin{bmatrix}15&6&-12&9\\4&14&8&-4\\2&4&10&-2\\9&6&-12&15\end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix}3&3\\2&2\\-2&-2\\3&9\end{bmatrix}\boldsymbol{u}(t)$$

解 用luenberger()函数可以构造所需变换矩阵，获得系统的Luenberger标准型。

```
>> A=[15,6,-12,9; 4,14,8,-4; 2,4,10,-2; 9,6,-12,15];
   B=[3,3; 2,2; -2,-2; 3,9]; T=luenberger(A,B) %获得Luenberger阵
   A1=inv(T)*A*T, B1=inv(T)*B  %对系统进行变换,即可得出此标准型
```

其数学表示形式为

$$\boldsymbol{T}=\begin{bmatrix}18&3&61.2&3\\-48&2&-79.2&2\\48&-2&43.2&-2\\18&3&-46.8&9\end{bmatrix},\ \dot{\boldsymbol{z}}(t)=\left[\begin{array}{cc:cc}0&1&0&0\\-144&30&-57.6&9.6\\\hdashline0&0&0&1\\0&0&-108&24\end{array}\right]\boldsymbol{z}(t)+\left[\begin{array}{cc}0&0\\1&0\\\hdashline0&0\\0&1\end{array}\right]\boldsymbol{u}(t)$$

5.1.9 系统的范数测度及求解

正如矩阵的范数是矩阵的测度一样，线性系统模型也有自己的范数定义，例如线性连续系统的 $\mathcal{H}_2$ 范数与无穷范数的定义分别为

$$\|\boldsymbol{G}(s)\|_2=\sqrt{\frac{1}{2\pi\mathrm{j}}\int_{-\mathrm{j}\infty}^{\mathrm{j}\infty}\sum_{i=1}^{p}\sigma_i^2\left[\boldsymbol{G}(\mathrm{j}\omega)\right]\mathrm{d}\omega},\ \|\boldsymbol{G}(s)\|_\infty=\sup_{\omega}\bar{\sigma}|\boldsymbol{G}(\mathrm{j}\omega)| \tag{5-1-25}$$

从式(5-1-25)中可以看出，$\mathcal{H}_\infty$ 范数实际上是频域响应幅值的峰值。对线性离散系统来说，系统的 $\mathcal{H}_2$ 范数与无穷范数的定义分别为

$$\|\boldsymbol{G}(z)\|_2=\sqrt{\int_{-\pi}^{\pi}\sum_{i=1}^{p}\sigma_i^2\left[\boldsymbol{G}\left(\mathrm{e}^{\mathrm{j}\omega}\right)\right]\mathrm{d}\omega},\ \|\boldsymbol{G}(z)\|_\infty=\sup_{\omega}\bar{\sigma}\left[\boldsymbol{G}\left(\mathrm{e}^{\mathrm{j}\omega}\right)\right] \tag{5-1-26}$$

其中 $\sigma_i(\cdot)$ 为矩阵的第 i 奇异值，而 $\bar{\sigma}(\cdot)$ 为矩阵奇异值的上限。

若系统模型已经由变量 $\boldsymbol{G}$ 给出，则系统的范数 $||\boldsymbol{G}(s)||_2$ 和 $||\boldsymbol{G}(s)||_\infty$ 可以分别调用 MATLAB 函数 `norm(G)` 和 `norm(G,inf)` 直接求出。离散系统的范数也可以同样求出。系统的范数概念可以用于系统的鲁棒控制器设计，可以将其作为指标进行控制。

例 5-11　试求例 5-6 中给出多变量离散系统的 $\mathcal{H}_2$ 范数和 $\mathcal{H}_\infty$ 范数。

解　这些范数可以用下面命令直接求出。

```
>> A=[-2.2,-0.7,1.5,-1; 0.2,-6.3,6,-1.5; ...
      0.6,-0.9,-2,-0.5; 1.4,-0.1,-1,-3.5];
   B=[6,9; 4,6; 4,4; 8,4]; C=[1 2 3 4];
   G=ss(A,B,C,[0 0],'Ts',0.1);
   norm(G,2), norm(G,inf), abs(eig(G))
```

可以直接求解出 $||\boldsymbol{G}(z)||_2=\infty$，$||\boldsymbol{G}(z)||_\infty=45.5817$。进一步地，由 `eig()` 函数可以得出系统特征值的模为 4,4,3,3，原系统不稳定，所以得出该系统的 $\mathcal{H}_2$ 范数为无穷大。

5.2　线性系统时域响应解析解法

前面介绍过，线性系统的数学基础是线性微分方程和线性差分方程，它们在某些条件下是存在解析解的，这里将介绍两种线性系统的解析解方法：基于状态方程的解析解方法和基于传递函数的解析解方法，并将以典型二阶系统为例，引入后面将使用的一些概念，如阻尼比、超调量等。还将介绍时间延迟系统的解析解方法。

5.2.1　直接积分解析解方法

再考虑状态方程的解析解

$$\boldsymbol{x}(t)=\mathrm{e}^{\boldsymbol{A}(t-t_0)}\boldsymbol{x}(t_0)+\int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\tau)}\boldsymbol{B}\boldsymbol{u}(\tau)\mathrm{d}\tau,\ \boldsymbol{y}(t)=\boldsymbol{C}\boldsymbol{x}(t) \tag{5-2-1}$$

由于 MATLAB 的符号运算具有很强的积分运算能力，且求解矩阵指数也很容易，所以可以尝试符号运算命令求出线性系统的解析解，具体的求解语句为

```
y=C*(expm(A*(t-t0))*x0+...
    expm(A*t)*int(expm(-A*τ)*B*subs(u,t,τ),τ,t0,t))
```

其中，`subs()` 函数用来处理变量替换的运算，因为输入信号原本是 t 的函数，而在积分运算中需要 τ 的函数，所以需要用 `subs()` 函数进行变量替换。求出解析解后，有必要采用 `simplify()` 函数化简得出的结果。

例5-12 系统的状态方程模型为

$$\begin{cases} \dot{\boldsymbol{x}}(t)=\begin{bmatrix} -19 & -16 & -16 & -19 \\ 21 & 16 & 17 & 19 \\ 20 & 17 & 16 & 20 \\ -20 & -16 & -16 & -19 \end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix} 1 \\ 0 \\ 1 \\ 2 \end{bmatrix}u(t) \\ y(t)=[2,1,0,0]\boldsymbol{x}(t) \end{cases}$$

其中，状态变量初值为 $\boldsymbol{x}^{\mathrm{T}}(0)=[0,1,1,2]$，且输入信号为 $u(t)=2+2\mathrm{e}^{-3t}\sin 2t$。试求系统时域响应的解析解。

解 下面的语句可以直接得出系统时域响应的解析解。

```
>> syms t tau; u=2+2*exp(-3*t)*sin(2*t);
   A=[-19,-16,-16,-19; 21,16,17,19; 20,17,16,20; -20,-16,-16,-19];
   B=[1; 0; 1; 2]; C=[2 1 0 0]; D=0;  x0=[0; 1; 1; 2];
   y=C*(expm(A*t)*(x0+int(expm(-A*tau)*B*subs(u,t,tau),tau,0,t)))
   y=simplify(y)
```

得出的时域响应为

$$y(t)=-54+\frac{127}{4}t\mathrm{e}^{-t}+57\mathrm{e}^{-3t}+\frac{119}{8}\mathrm{e}^{-t}+4t^2\mathrm{e}^{-t}-\frac{135}{8}\mathrm{e}^{-3t}\cos 2t+\frac{77}{4}\mathrm{e}^{-3t}\sin 2t$$

5.2.2 基于增广矩阵的解析解方法

对于一般的输入信号来说，直接由式(5-1-2)求取系统的解析解并非很容易的事，因为其中积分项不好处理。如果能对状态方程进行某种变换，消去输入信号，则该方程的解析解就容易求解了。这里将对一类典型输入信号介绍状态增广的方法，将其化为不含有输入信号的状态方程，从而直接求解原来状态方程的解析解[6]。

先考虑单位阶跃信号 $u(t)=1(t)$，若假设有另外一个状态变量 $x_{n+1}(t)=u(t)$，则其导数为 $\dot{x}_{n+1}(t)=0$，这样系统的状态方程可以改写为

$$\left[\begin{array}{c} \dot{\boldsymbol{x}}(t) \\ \hdashline \dot{x}_{n+1}(t) \end{array}\right]=\left[\begin{array}{c:c} \boldsymbol{A} & \boldsymbol{B} \\ \hdashline \boldsymbol{0} & \boldsymbol{0} \end{array}\right]\left[\begin{array}{c} \boldsymbol{x}(t) \\ \hdashline x_{n+1}(t) \end{array}\right] \tag{5-2-2}$$

可见，这样就把原始的状态方程转换成直接可以求解的自治系统方程了。

$$\begin{cases} \dot{\widetilde{\boldsymbol{x}}}(t)=\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{x}}(t) \\ \widetilde{\boldsymbol{y}}(t)=\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{x}}(t) \end{cases} \tag{5-2-3}$$

式中，$\widetilde{\boldsymbol{x}}^{\mathrm{T}}(t)=\left[\boldsymbol{x}^{\mathrm{T}}(t),x_{n+1}(t)\right]$，且 $\widetilde{\boldsymbol{x}}^{\mathrm{T}}(0)=\left[\boldsymbol{x}^{\mathrm{T}}(0),1\right]$，其解析解比较容易求出

$$\widetilde{\boldsymbol{x}}(t)=\mathrm{e}^{\widetilde{\boldsymbol{A}}t}\widetilde{\boldsymbol{x}}(0) \tag{5-2-4}$$

除了阶跃信号外，下面的一类典型输入信号也可以作相应的增广。

$$u(t)=u_1(t)+u_2(t)=\sum_{i=0}^{m}c_it^i+\mathrm{e}^{d_1t}\left(d_2\cos d_4t+d_3\sin d_4t\right) \tag{5-2-5}$$

引入附加状态变量 $x_{n+1}=\mathrm{e}^{d_1 t}\cos d_4 t$, $x_{n+2}=\mathrm{e}^{d_1 t}\sin d_4 t$, $x_{n+3}=u_1(t),\cdots,x_{n+m+3}=u_1^{(m-1)}(t)$,通过推导可以得出式(5-2-3)中给出的系统增广状态方程模型,式中

$$\widetilde{\boldsymbol{A}}=\left[\begin{array}{c|cc|ccc}\boldsymbol{A} & d_2\boldsymbol{B} & d_3\boldsymbol{B} & \boldsymbol{B} & \boldsymbol{0} & \cdots\ \boldsymbol{0}\\ \hline \boldsymbol{0} & d_1 & -d_4 & \multicolumn{3}{c}{\multirow{2}{*}{$\boldsymbol{0}$}}\\ & d_4 & d_1 & & & \\ \hline \multirow{3}{*}{$\boldsymbol{0}$} & \multicolumn{2}{c|}{\multirow{3}{*}{$\boldsymbol{0}$}} & 0 & 1 & \cdots\ 0\\ & & & 0 & 0 & \cdots\ 0\\ & & & \vdots & \vdots & \ddots\ \vdots\\ & & & 0 & 0 & \cdots\ 0\end{array}\right],\widetilde{\boldsymbol{x}}(t)=\left[\begin{array}{c}\boldsymbol{x}(t)\\ x_{n+1}(t)\\ x_{n+2}(t)\\ x_{n+3}(t)\\ x_{n+4}(t)\\ \vdots\\ x_{n+m+3}(t)\end{array}\right],\widetilde{\boldsymbol{x}}(0)=\left[\begin{array}{c}\boldsymbol{x}(0)\\ \hline 1\\ 0\\ \hline c_0\\ c_1\\ \vdots\\ c_m m!\end{array}\right] \tag{5-2-6}$$

这样系统的状态方程模型的解析解同样能由式(5-2-4)求出。

作者用 MATLAB 语言编写了一个函数 ss_augment(),可以用来求取系统的增广状态方程模型,该函数的内容如下:

```
function [Ga,Xa]=ss_augment(G,cc,dd,X)
G=ss(G); Aa=G.a; Ca=G.c; Xa=X; Ba=G.b; D=G.d;
if (length(dd)>0 & sum(abs(dd))>1e-5)
   if (abs(dd(4))>1e-5)
      Aa=[Aa dd(2)*Ba, dd(3)*Ba; ...
          zeros(2,length(Aa)), [dd(1),-dd(4); dd(4),dd(1)]];
      Ca=[Ca dd(2)*D dd(3)*D]; Xa=[Xa; 1; 0]; Ba=[Ba; 0; 0];
   else, Aa=[Aa dd(2)*B; zeros(1,length(Aa)) dd(1)];
      Ca=[Ca dd(2)*D]; Xa=[Xa; 1]; Ba=[B;0];
end, end
if (length(cc)>0 & sum(abs(cc))>1e-5), M=length(cc);
   Aa=[Aa Ba zeros(length(Aa),M-1); zeros(M-1,length(Aa)+1) ...
       eye(M-1); zeros(1,length(Aa)+M)];
   Ca=[Ca D zeros(1,M-1)]; Xa=[Xa; cc(1)]; ii=1;
   for i=2:M, ii=ii*i; Xa(length(Aa)+i)=cc(i)*ii;
end, end, Ga=ss(Aa,zeros(size(Ca')),Ca,D);
```

其调用格式为 [G_1,$\boldsymbol{x}_1$]=ss_augment(G,$\boldsymbol{c}$,$\boldsymbol{d}$,$\boldsymbol{x}_0$),其中,$\boldsymbol{c}=[c_0,c_1,\cdots,c_m]$,$\boldsymbol{d}=[d_1,d_2,d_3,d_4]$,$\boldsymbol{x}_0$ 为初始状态。构造出系统的增广状态方程模型后,则可以用 MATLAB 符号运算工具箱的 expm() 函数求取各个状态变量的解析解。

例 5-13 试用增广状态法重新求解例 5-12 中的问题。

解 可以用 ss_augment() 函数得出系统的增广状态方程模型。

```
>> c=[2]; d=[-3,0,2,2]; x0=[0; 1; 1; 2];
   A=[-19,-16,-16,-19; 21,16,17,19; 20,17,16,20; -20,-16,-16,-19];
   B=[1; 0; 1; 2]; C=[2 1 0 0]; D=0; G=ss(A,B,C,D);
   [Ga,xx0]=ss_augment(G,c,d,x0); Ga.a, xx0'
```

得出的增广状态方程模型为

$$\dot{\tilde{\boldsymbol{x}}}(t)=\left[\begin{array}{cccc:cc:c}-19&-16&-16&-19&0&2&1\\21&16&17&19&0&0&0\\20&17&16&20&0&2&1\\-20&-16&-16&-19&0&4&2\\\hdashline0&0&0&0&-3&-2&0\\0&0&0&0&2&-3&0\\\hdashline0&0&0&0&0&0&0\end{array}\right]\tilde{\boldsymbol{x}}(t),\quad \tilde{\boldsymbol{x}}(0)=\left[\begin{array}{c}0\\1\\1\\2\\\hdashline1\\0\\\hdashline2\end{array}\right]$$

得出了系统的增广状态方程模型，则可以用下面的语句直接获得生成信号的解析解，得出的结果和前面的完全一致。

```
>> syms t; y=Ga.c*expm(Ga.a*t)*xx0; %求解系统的解析解
```

5.2.3 基于Laplace变换、z变换的解析解方法

1. 连续系统的解析解法

假设系统的传递函数由下式给出

$$G(s)=\frac{b_1s^m+b_2s^{m-1}+\cdots+b_ms+b_{m+1}}{s^n+a_1s^{n-1}+a_2s^{n-2}+\cdots+a_{n-1}s+a_n}\tag{5-2-7}$$

且已知系统输入信号的Laplace变换$U(s)$，则可以求出系统输出信号的Laplace变换$Y(s)=G(s)U(s)$。这样，系统输出信号的解析解可以由Laplace反变换直接求出。在MATLAB下，可以用`laplace()`、`ilaplace()`函数来直接求解函数的正反Laplace变换。如果没有安装符号运算工具箱，则可以通过$Y(s)$函数部分分式展开的方式求取时域响应的解析解[7]。下面将通过例子演示系统解析解的求解方法。

例5-14 考虑系统的传递函数模型。

$$G(s)=\frac{s^3+7s^2+3s+4}{s^4+7s^3+17s^2+17s+6}$$

系统的输入信号为单位阶跃信号，试求输出信号的解析解。

解 已知输入信号的Laplace变换为$1/s$，这样，通过下面的语句可以直接求出输出信号的解析解。

```
>> syms s; G=(s^3+7*s^2+3*s+4)/(s^4+7*s^3+17*s^2+17*s+6);
   Y=G/s; y=ilaplace(Y)
```

可以得出原问题的解析解为

$$y(t)=\frac{31}{12}\mathrm{e}^{-3t}-9\mathrm{e}^{-2t}+\frac{23}{4}\mathrm{e}^{-t}-\frac{7}{2}t\mathrm{e}^{-t}+\frac{2}{3}$$

2. 离散系统的解析解法

考虑离散系统传递函数模型$G(z)$，如果输入信号的z变换为$R(z)$，则输出信号的z变换可以表示为$Y(z)=G(z)R(z)$，这样，输出信号的解析解$y(n)$可以由$Y(z)$进行z反变换直接求出，$y(n)=\mathscr{Z}^{-1}[Y(z)]$。MATLAB的符号运算工具箱提供了$z$

变换函数 ztrans() 和 z 反变换函数 iztrans()，可以用来求取离散系统的时域响应解析解。

例 5-15 假设一个系统的离散传递函数为

$$G(z)=\frac{(z-1/3)}{(z-1/2)(z-1/4)(z+1/5)}$$

并假设系统的输入为阶跃信号，试求输出信号的解析解。

解 已知输入阶跃信号的 z 变换为 $z/(z-1)$，这样就可以用下面的语句将系统的输出在 MATLAB 环境中计算出来。

```
>> syms z; G=(z-1/3)/(z-1/2)/(z-1/4)/(z+1/5);
   R=z/(z-1); y=iztrans(G*R)
```

系统的解析解为

$$y(n)=\frac{800}{567}\left(-\frac{1}{5}\right)^n-\frac{80}{81}\left(\frac{1}{4}\right)^n-\frac{40}{21}\left(\frac{1}{2}\right)^n+\frac{40}{27}$$

例 5-16 试求离散传递函数 $G(z)=\dfrac{5z-2}{(z-1/2)^3(z-1/3)}$ 的阶跃响应解析解。

解 阶跃响应的解析解可以通过下面的命令求出。

```
>> syms z; G=(5*z-2)/(z-1/2)^3/(z-1/3);
   R=z/(z-1); y=iztrans(G*R)
```

这样经过 z 反变换即可以求出输出信号的解析解为

$$y(n)=36-108\,(1/3)^n+72\,(1/2)^n-60n\,(1/2)^n-12n^2\,(1/2)^n$$

3. 时间延迟系统的解析解法

考虑带有时间延迟的连续系统模型 $G(s)\mathrm{e}^{-Ls}$ 和离散系统传递函数 $H(z)z^{-k}$，直接对这样的式子进行部分分式展开不便，所以在使用前述的展开时可不考虑时间延迟因素，这样就可以得出不带有时间延迟的系统输出解析解，假设分别为 $y(t)$ 或 $y(n)$，这时根据 Laplace 变换和 z 变换的性质，分别用 $t-L$ 或 $n-k$ 代替得出解析解中的 t 或 n，就可以构造出时间延迟系统的解析解。

例 5-17 如果例 5-14 中给出的传递函数 $G(s)$ 含有 2 s 延迟，试求阶跃响应的解析解。

解 阶跃响应解析解可以由下面的变量替换语句直接得出。

```
>> syms s t; G=(s^3+7*s^2+3*s+4)/(s^4+7*s^3+17*s^2+17*s+6);
   Y=G/s; y=ilaplace(Y); y=subs(y,t,t-2), fplot(y,[0,10])
```

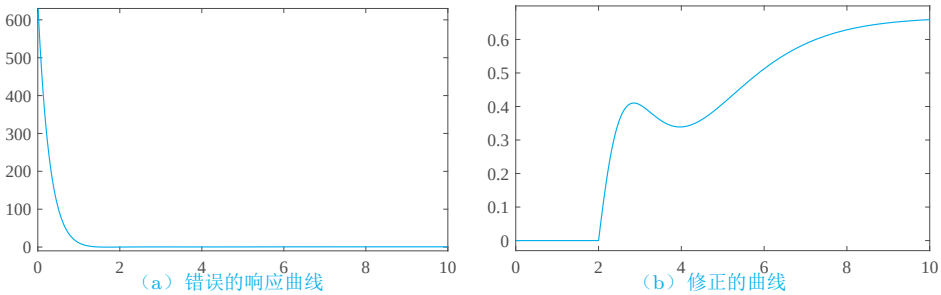
可以得出该系统的解析解如下，阶跃响应如图 5-5(a)所示。

$$y=2/3-9\mathrm{e}^{-2t+4}+31\mathrm{e}^{-3t+6}/12-\mathrm{e}^{-t+2}\,(14t-51)/4$$

观察得出的曲线可见，该结果是错误的，因为在 $t\leqslant 2$ 时系统响应本应为零，而单纯的变量替换导致非零的结果，所以应该将得出的解析解修正如下，其中，$1(\cdot)$ 为 Heaviside 函数，该系统的阶跃响应曲线可以由下面语句直接绘制，如图 5-5(b)所示。

$$y(t)=\left[\frac{2}{3}-9\mathrm{e}^{-2(t-2)}+\frac{31}{12}\mathrm{e}^{-3(t-2)}-\frac{1}{4}\,\mathrm{e}^{-(t-2)}\,(14(t-2)-23)\right]\times 1(t-2)$$

```
>> fplot(y*heaviside(t-2),[0,10])
```



（a）错误的响应曲线　　（b）修正的曲线

图 5-5　延迟系统的阶跃响应曲线

例 5-18　考虑下面的带有时间延迟的系统模型，试求该模型的单位阶跃响应解析解。

$$G(z)z^{-5}=\frac{5z-2}{(z-1/2)^3(z-1/3)}z^{-5}$$

解　可以看出，其中的 $G(z)$ 和例 5-16 中的完全一致，该例中已经得出了不带有时间延迟部分的阶跃响应解析解。对带有时间延迟的系统来说，用 $n-5$ 取代其中的 n，得出的结果就是整个系统的阶跃响应解析解为

$$\begin{aligned}y(n)&=-108\,(1/3)^{n-5}+\left[-12(n-5)^2-60(n-5)+72\right](1/2)^{n-5}+36\\&=-108\,(1/3)^{n-5}+\left(-12n^2+60n+72\right)(1/2)^{n-5}+36\end{aligned}$$

变量替换还可以用 subs() 函数处理。更严格地说，原延迟系统的解析解应该写成

$$y(n)=\begin{cases}0, & n\leqslant 5\\ -108\,(1/3)^{n-5}+\left(-12n^2+60n+72\right)(1/2)^{n-5}+36, & n>5\end{cases}$$

虽然可以用下面命令求解原问题，但得出的解可读性较差，包含 kroneckerDelta() 与 nchoosek() 等函数，需要进一步手工替换与处理。

```
>> syms z; G=(5*z-2)/(z-1/2)^3/(z-1/3)*z^(-5);
   R=z/(z-1); y=iztrans(G*R)
```

5.2.4　阶跃响应指标

线性系统典型的阶跃响应曲线示意图由图 5-6 给出，其中，人们感兴趣的阶跃响应指标包括：

（1）稳态值 $y(\infty)$。亦即系统在时间很大时的系统输出极限值，对不稳定系统来说稳态值趋于无穷大。对稳定的线性连续系统模型来说，应用 Laplace 变换中终值的性质定理，可以容易地得出系统阶跃响应的稳态值为

$$y(\infty)=\lim_{s\to 0}sG(s)\frac{1}{s}=G(0)=\frac{b_m}{a_n}\qquad(5\text{-}2\text{-}8)$$

亦即对传递函数模型来说，系统的稳态值即为分子、分母常数项的比值。如果已知系统的数学模型 G，则系统的阶跃响应稳态值可以由 dcgain(G) 直接得出。

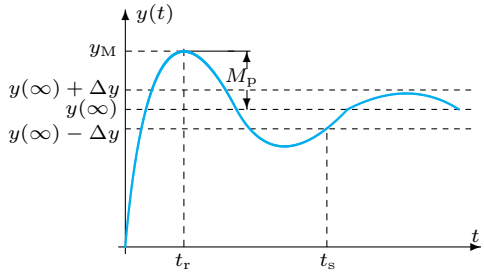


图 5-6 典型控制系统阶跃响应指标示意图

(2)**超调量** σ。定义为系统的峰值 y_p 与稳态值的差距,通常用下面的公式求出

$$\sigma = \frac{y_p - y(\infty)}{y(\infty)} \times 100\% \tag{5-2-9}$$

(3)**上升时间** t_r。一般定义为系统阶跃响应从稳态值的 10% 到 90% 的这段时间,有的定义也可以是从开始响应到阶跃响应达到稳态值所需的时间。

(4)**调节时间** t_s。一般指系统的阶跃响应进入稳态值附近的一个带中,比如 2% 或 5% 的带后不再出来时所需的时间。

对一个好的伺服控制系统来说,一般应该具有稳态误差小或没有稳态误差,超调量小或没有超调量,上升时间短,调节时间短等性能。所以这些性能指标在控制系统设计中是经常使用的。

5.3 线性系统的数值仿真分析

前面介绍了线性系统的解析解方法,并解释了可以求解的条件。严格说来,四阶以上的系统需要求解四阶以上的多项式方程,所以根据 Abel 不可能性定理,没有代数解法能得出这类一般方程的解,换言之,这类方程是没有解析解的,从而使得高阶微分方程也没有解析解。应用前面介绍的解析解和数值解的结合可以求出系统时域响应的高精度准解析表达式。

在实际应用中,并不是所有情况下都希望得出系统的解析解,有时得到系统时域响应的曲线就足够了,不一定非得得出输出信号的解析表达式。在这样的情况下可以借助于微分方程数值解的技术求取系统响应的数值解,并用曲线表示结果。

本节首先介绍阶跃响应、脉冲响应的数值解求法及响应曲线绘制方法,再介绍一般输入下系统时域响应数值解、非零初始状态响应数值解及曲线绘制等内容,最后将介绍多变量系统的时域响应分析方法。

5.3.1 线性系统的阶跃响应与脉冲响应

线性系统的阶跃响应可以通过 `step()` 函数直接求取,脉冲响应可以使用函数 `impulse()` 来获得,而在任意输入下的系统响应可以通过 `lsim()` 函 数,更复杂系

统的时域响应分析还可以通过强大的Simulink环境来直接求取。

step()函数有如下多种调用格式：

```
step(G),                 % 不返回变元将自动绘制阶跃响应曲线
[y,t]=step(G),           % 自动选择时间向量,进行阶跃响应分析
[y,t]=step(G,tn),        % 设置系统的终止响应时间tn,进行阶跃响应分析
y=step(G,t),             % 用户自己选择时间向量t,进行阶跃响应分析
```

除了经典的step()函数之外，新版MATLAB控制系统工具箱提供了stepplot()函数，也可以绘制系统的阶跃响应曲线。对一般使用者而言，该函数与step()基本通用，这里不特别推荐使用新版本的函数。当然，新版本函数是基于面向对象技术编写的，该函数调用时，允许由h=stepplot(···)格式返回阶跃响应曲线的句柄h，用户可以事后修改图形对象的属性。后面将通过例子演示图形属性的修改方法。

这里系统模型G可以为任意的线性时不变系统模型，包括传递函数、零极点、状态方程模型、单变量和多变量模型、连续与离散模型、带有时间延迟的模型等。若上述的函数调用时不返回任何参数，则将自动打开图形窗口，将系统的阶跃响应曲线直接在该窗口上显示出来，并用虚线绘制稳态值。如果想同时绘制出多个系统的阶跃响应曲线，则可以仿照plot()函数给出系统阶跃响应曲线命令，如

```
step(G1,'-',G2,'-.b',G3,':r')
```

该命令可以用实线绘制系统G_1的阶跃响应曲线，用蓝色点画线绘制G_2的阶跃响应曲线，用红色点线绘制出系统G_3的阶跃响应曲线。

与plot()函数一样，step(h,...)函数还允许在指定坐标系下绘制阶跃响应曲线，其中，h为指定坐标系的句柄。

例5-19 试求下面带有时间延迟连续模型的单位阶跃响应。

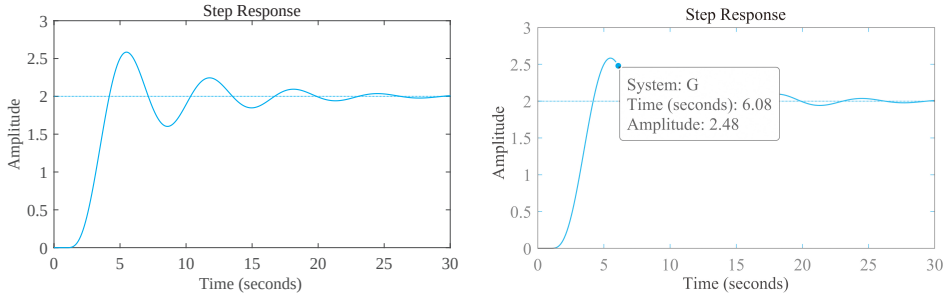
$$G(s)=\frac{10s+20}{10s^4+23s^3+26s^2+23s+10}\mathrm{e}^{-s}$$

解 可以通过下面的命令直接输入系统模型，并绘制出阶跃响应曲线，如图5-7(a)所示。

```
>> G=tf([10 20],[10 23 26 23 10],'ioDelay',1); % 系统模型
   step(G,30);     % 绘制阶跃响应曲线,终止时间为30
```

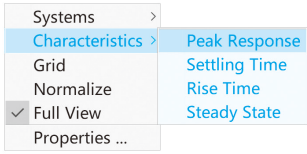
在自动绘制的系统阶跃响应曲线上，若单击曲线上某点，则可以显示出该点对应的时间信息和响应的幅值信息，如图5-7(b)所示。通过这样的方法就可以容易地分析系统阶跃响应的情况。

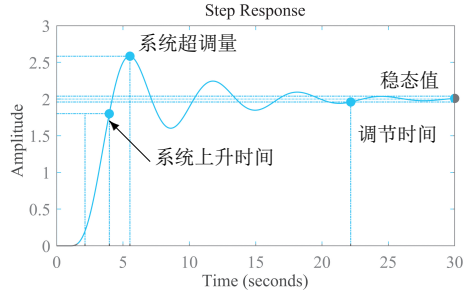
在控制理论中介绍典型线性系统的阶跃响应分析时经常用一些指标来定量描述，例如系统的超调量、上升时间、调节时间等，在MATLAB自动绘制的阶跃响应曲线中，如果想得出这些指标，只需右击鼠标键(不在响应曲线上)，则将得出如图5-8(a)所示的菜单，选择其中的Characteristics(特性)菜单项，从中选择合适的分析内容，即可以得出系统的阶跃响应指标，如图5-8(b)所示。若想获得某个指标的具体值，则需先将鼠标移动到该点上即可。



（a）自动绘制的阶跃响应曲线　　（b）获取某点的响应值

图 5-7　线性系统的阶跃响应曲线





（a）系统阶跃响应快捷菜单　　（b）阶跃响应指标显示

图 5-8　阶跃响应指标显示

用前面给出的方法,还可以容易地得出系统阶跃响应的解析解。

```
>> syms s t; G1=tf2sym(G); y1=ilaplace(G1/s)
```

这样就能得出系统的阶跃响应解析解的数学形式为

$$y(t)=2-\frac{10t}{17}\mathrm{e}^{-t}-\frac{4}{17}\left(\cos\frac{\sqrt{391}}{20}t+\frac{103}{\sqrt{391}}\sin\frac{\sqrt{391}t}{20}t\right)\mathrm{e}^{-3t/20}-\frac{30}{17}\mathrm{e}^{-t}$$

因为解析解是已知的,所以由下面的语句还可以估算出解析解的精度。

```
>> [y,t1]=step(tf(num,den));      % 用数值方法求取阶跃响应数据
   y0=subs(y1,t,t1); norm(y-y0) % 评价数值解的误差
```

可见得出的阶跃响应可以达到 9.1×10^{-14} 这样的精度级,所以结果是可信的。

例 5-20　考虑例 5-3 中描述的闭环系统,试绘制该闭环系统的阶跃响应曲线。

解　前面介绍过,这样的闭环模型不能用传递函数描述,但可以自动由带有内部延迟的状态方程模型表示。所以,即使这样复杂系统的阶跃响应也可以通过下面直观的、常规的方法绘制阶跃响应曲线,如图 5-9 所示,可见该系统是稳定的,与例 5-3 得出的结论是完全一致的。

```
>> s=tf('s'); G=(1+3*exp(-s)/(s+1))/(s+1);
   Gc=0.3+0.15/s; G1=feedback(G*Gc,1); step(G1)
```

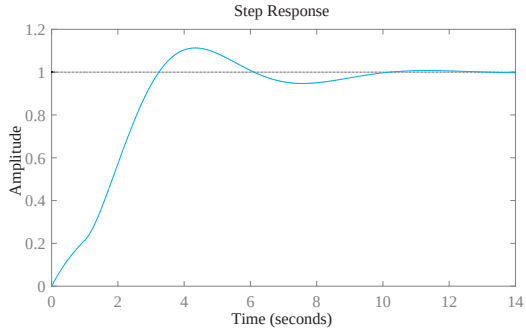


图 5-9　连续内部延迟闭环系统的阶跃响应

例 5-21　第 4 章中曾经介绍了连续系统离散化的方法。假设连续系统的数学模型为 $G(s)=\mathrm{e}^{-s}/(s^2+0.2s+1)$，试研究采样周期对系统离散化的影响。

解　选择采样周期为 $T=0.01,0.1,0.5,1.2\,\mathrm{s}$，则可以用下面的语句得出各个离散化的传递函数模型。再用 `step()` 函数进行对比分析，得出如图 5-10 所示的阶跃响应曲线。采样周期越小，离散化系统越接近原始的连续模型。若采样周期选择过大，则有可能丢失原来系统的信息。

```
>> G=tf(1,[1 0.2 1],'ioDelay',1);    %输入连续系统数学模型
   G1=c2d(G,0.01,'zoh'); G2=c2d(G,0.1);
   G3=c2d(G,0.5); G4=c2d(G,1.2);     %Tustin 变换,有时可能导致虚系数
   step(G,'-',G2,'--',G3,':',G4,'-.',10) %比较各个模型阶跃响应
```

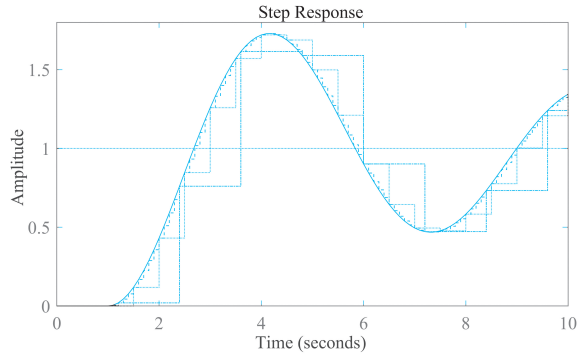


图 5-10　连续系统离散化的效果比较

这样得出的离散模型分别为

$$G_1(z)=\frac{4.997\times10^{-5}z+4.993\times10^{-5}}{z^2-1.998z+0.998}z^{-100},\quad G_2(z)=\frac{0.004963z+0.00493}{z^2-1.97z+0.9802}z^{-10}$$

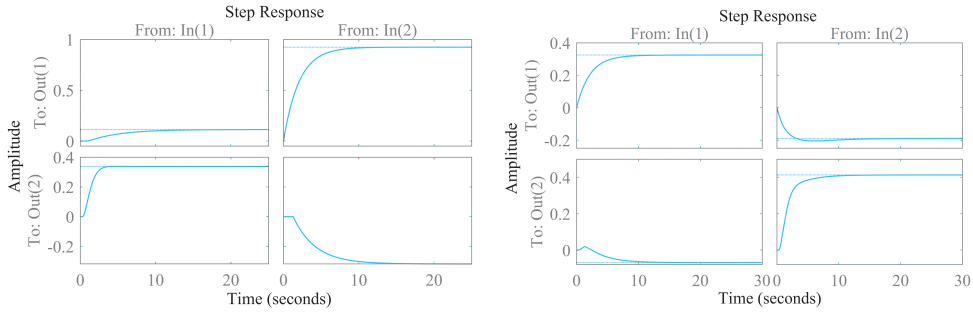
$$G_3(z)=\frac{0.1185z+0.1145}{z^2-1.672z+0.9048}z^{-2},\quad G_4(z)=\frac{0.01967z^2+0.7277z+0.3865}{z^3-0.6527z^2+0.7866z}$$

值得指出的是，`step()` 函数绘制出的离散系统阶跃响应曲线是以阶梯线的形式表示的，在该曲线上仍然可以使用右键菜单显示其响应指标。

例 5-22　试绘制例 4-10 中给出的双输入、双输出系统的阶跃响应曲线。

解　可以用下面语句直接绘制出分别在两路阶跃输入激励下系统的两个输出信号的阶跃响应曲线,如图 5-11(a)所示。

```
>> g11=tf(0.1134,[1.78 4.48 1],'ioDelay',0.72);
   g21=tf(0.3378,[0.361 1.09 1],'ioDelay',0.3);
   g12=tf(0.924,[2.07 1]); g22=tf(-0.318,[2.93 1],'ioDelay',1.29);
   G=[g11, g12; g21, g22]; step(G)  %多变量系统的阶跃响应
```



(a) 原系统阶跃响应　　(b) 加入矩阵环节后阶跃响应

图 5-11　多变量系统的阶跃响应曲线

注意,这时得出的阶跃响应曲线是在两路输入均单独作用下分别得出的。从得出的系统阶跃响应可以看出,在第 1 路信号输入时,第 1 路输出信号有响应,而第 2 路输出信号也有很强的响应。单独看第 2 路输入信号的作用也是这样,这在多变量系统理论中称为系统的耦合,在多变量系统的设计中是很不好处理的。因为若没有这样的耦合,则可以给两路信号分别设计控制器就可以了,但有了耦合,就必须考虑引入某种环节,使得耦合尽可能小,这样的方法在多变量系统理论中又称为解耦。考虑有了现成的矩阵 $\boldsymbol{K}_{\mathrm{p}}$ 对系统进行补偿

$$\boldsymbol{K}_{\mathrm{p}}=\begin{bmatrix}0.1134 & 0.924\\ 0.3378 & -0.318\end{bmatrix}$$

由于需要对传递函数进行四则运算,而其中子传递函数有的带有时间延迟,传统意义下并不能利用矩阵乘法的方式进行直接运算,采用带有内部延迟的状态方程模型则可以处理。

```
>> Kp=[0.1134,0.924; 0.3378,-0.318]; step(G*Kp)
```

上面的语句可以直接绘制出 $\boldsymbol{G}(s)\boldsymbol{K}_{\mathrm{p}}$ 系统的阶跃响应曲线,如图 5-11(b)所示。可见在矩阵的补偿下,两路输出的耦合明显降低,从而使得控制器单独设计变成可能。

系统的脉冲响应曲线可以由 MATLAB 控制系统工具箱中的 `impulse()` 函数直接绘制出来,该函数的调用格式与 `step()` 函数完全一致。

例 5-23　试求例 5-19 中系统的脉冲响应曲线。

解　可以用下面的语句直接绘制该系统的脉冲响应曲线,如图 5-12 所示。

```
>> G=tf([10 20],[10 23 26 23 10],'ioDelay',1); impulse(G,30);
```

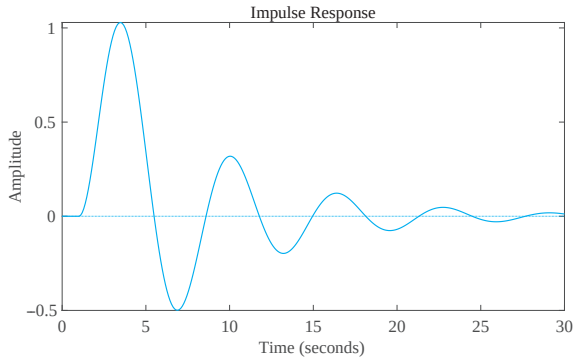


图 5-12　系统的脉冲响应曲线

5.3.2　任意输入下系统的响应

前面介绍了两种常用的时域响应求取函数，`step()` 函数和 `impulse()` 函数，应用这些函数可以很容易地绘制系统的时域响应曲线。

若输入信号的 Laplace 变换 $R(s)$ 能够表示成有理函数的形式，则输出信号可以写成 $Y(s) = G(s)R(s)$，这样系统的时域响应可以由 $Y(s)$ 的脉冲响应函数 `impulse()` 直接绘制出来，这样就可以实现系统的时域分析与仿真。

例 5-24　试绘制出例 5-19 中延迟系统的斜坡响应曲线。

解　斜坡信号的 Laplace 变换为 $1/s^2$，故系统的斜坡响应既可以由 $G(s)/s$ 系统的阶跃响应求出，也可以由 $G(s)/s^2$ 系统的脉冲响应得出，所以由下面的 MATLAB 语句可以绘制出系统的斜坡响应曲线，如图 5-13 所示。

```
>> G=tf([10 20],[10 23 26 23 10],'ioDelay',1); %系统模型
   s=tf('s'); step(G/s,50);                    %或 impulse(G/s^2,50)
```

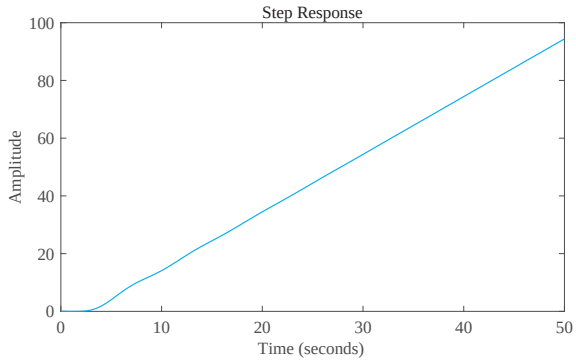


图 5-13　系统的斜坡响应曲线

如果输入信号由其他数学函数描述，或输入信号的数学模型未知，则用这两个函数就无能为力了，需要借助于 `lsim()` 函数来绘制系统时域响应曲线了。`lsim()` 函数的调

用格式与 step() 等函数的格式较类似,所不同的是,需要提供有关输入信号的函数值,该函数的调用格式为 lsim(*G*,*u*,*t*),其中,*G* 为系统模型,*u* 和 *t* 将用于描述输入信号,*u* 中的点对应于各个时间点处的输入信号值,若想研究多变量系统,则 *u* 应该是矩阵,其各行对应于 *t* 向量各个时刻的各路输入的值。调用了这个函数,将自动绘制出系统在任意输入下的时域响应曲线。

例 5-25 考虑例 5-3 中描述的闭环系统,如果输入信号为下面给出的分段函数,试绘制该闭环系统的时域响应曲线。

$$u(t)=\begin{cases} t, & t\leqslant 2 \\ 2, & 2<t\leqslant 20 \end{cases}$$

解 由下面的语句可以描述分段函数输入信号,然后直接绘制出变换系统的时域响应曲线,如图 5-14 所示。

```
>> s=tf('s'); G=(1+3*exp(-s)/(s+1))/(s+1);
   Gc=0.3+0.15/s; G1=feedback(G*Gc,1);
   t=0:0.01:20; u=t.*(t<=2)+2*(t>2); lsim(G1,u,t);
```

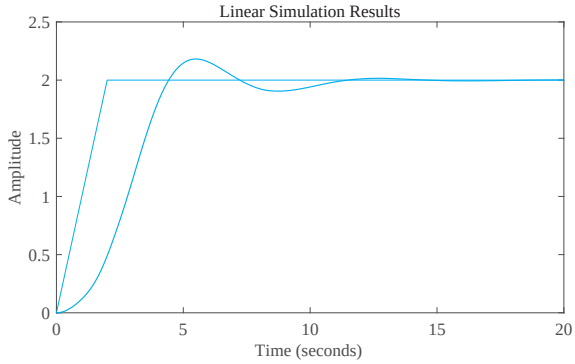


图 5-14 连续延迟系统的给定输入响应

例 5-26 考虑例 4-10 中的双输入双输出系统,假设第 1 路为 $u_1(t)=1-\mathrm{e}^{-t}\sin(3t+1)$,第 2 路输入为 $u_2(t)=\sin(t)\cos(t+2)$,试绘制系统的时域响应曲线。

解 可以用下面的语句输入系统模型,然后先定义系统的两路输入,再调用 lsim() 函数,就可以绘制出系统在这两路输入信号下系统时域响应曲线,如图 5-15 所示。

```
>> g11=tf(0.1134,[1.78 4.48 1],'ioDelay',0.72);
   g21=tf(0.3378,[0.361 1.09 1],'ioDelay',0.3);
   g12=tf(0.924,[2.07 1]); g22=tf(-0.318,[2.93 1],'ioDelay',1.29);
   G=[g11, g12; g21, g22]; t=[0:.1:15]';
   u=[1-exp(-t).*sin(3*t+1),sin(t).*cos(t+2)];
   lsim(G,u,t); %双输入信号下的时域响应曲线
```

这里的时域响应曲线和以前介绍的多变量系统阶跃响应概念是不同的,在这里是

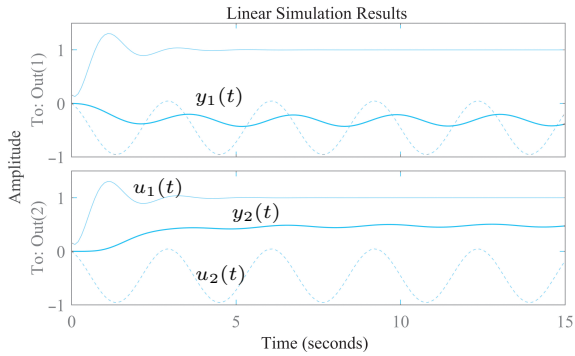


图 5-15　多变量系统的时域响应曲线

指在这两个信号共同作用下系统的时域响应，所以只需绘制两个图形，分别描述两路输出信号即可，两路输入信号也分别在时域响应曲线上绘制出来。

5.3.3　非零初始状态下系统的时域响应

前面介绍的传递函数时域响应曲线都是针对零初始状态系统的求解问题，如果系统的初始状态非零，则应该先使用 `initial()` 函数求出非零初始状态的时域响应，该函数的调用格式为 `[y,t]=initial(G,x0,tn)`，其中，t_n 为终止仿真时间，再利用叠加原理将 `lsim()` 的结果加到前面得出的结果上。

例 5-27　试用数值仿真方法重新求解例 5-12 中给出的问题。

解　在原例中得出了原系统的时域响应解析解，这里将探讨非零初始状态下时域响应曲线的绘制方法。可以给出下面的语句绘制出该系统的时域响应曲线，如图 5-16 所示。

```
>> A=[-19,-16,-16,-19; 21,16,17,19; 20,17,16,20; -20,-16,-16,-19];
   B=[1; 0; 1; 2]; C=[2 1 0 0]; G=ss(A,B,C,0);
   x0=[0; 1; 1; 2]; [y1,t]=initial(G,x0,10);
   u=2+2*exp(-3*t).*sin(2*t); y2=lsim(G,u,t); plot(t,y1+y2)
```

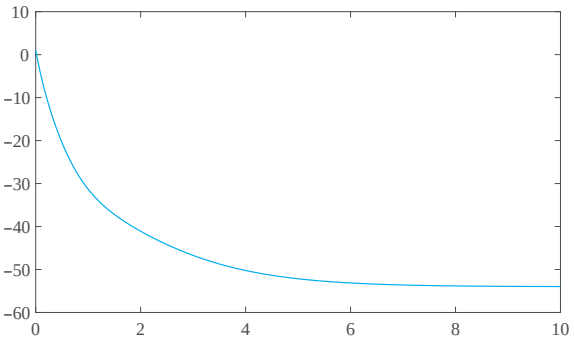


图 5-16　非零初始状态下的时域响应曲线

5.3.4 非正则系统的时域响应

值得指出的是，前面介绍的 step() 等函数只能用于正则系统的曲线绘制。对一些特定问题而言，例如，PID 控制系统，由于控制器分子的阶次高于分母的阶次，可能使得某些系统信号导致非正则现象。本节通过例子演示一个非正则系统的例子，演示 step() 函数的局限性，并演示某些非正则系统的近似数值解。

例 5-28 假设某受控对象模型与 PID 控制器模型如下，试绘制控制器输出信号的阶跃响应曲线。

$$G(s)=\frac{1}{(s+1)^3},\quad G_c(s)=2.18+\frac{0.847}{s}+1.4s$$

解 从输入信号到控制器输出信号的传递函数为 $G_c(s)/(1+G(s)G_c(s))$，从效果上等于前向通路为 $G_c(s)$，反馈通路为 $G(s)$ 的负反馈连接。由下面的语句可以尝试获得系统的阶跃响应曲线，同时得出传递函数的零极点形式。

```
>> s=tf('s'); G=1/(s+1)^3; Gc=2.18+0.847/s+1.4*s;
   G0=feedback(Gc,G); step(G0)
```

这样得出的等效传递函数如下，不过调用 step() 函数得出错误信息“Cannot simulate the time response of improper (non-causal) models（不能得出非正则系统的时域响应)”，因为传递函数的分子阶次高于分母的阶次。

$$G_0(s)=\frac{1.3958(s+1)^3(s+0.7792)^2}{(s+0.8322)(s+0.626)(s^2+1.542s+1.627)}$$

从 PID 模型看，产生非正则的原因是 $1.4s$ 算子，应该用 $1.4s/(\tau s+1)$，其中，τ 可以取很小的值，例如，$\tau=0.001$。这样，可以用下面的语句在求解控制信号时，对 τ 值依赖极大，因为 PID 控制器中的微分信号瞬时理论值为无穷大。

```
>> Gc=2.18+0.847/s;
   Gc1=Gc+1.4*s/(0.001*s+1); Gc2=Gc+1.4*s/(0.005*s+1);
   step(feedback(Gc1,G),feedback(Gc2,G),'--')
```

5.3.5 面向对象的时域响应曲线绘制

前面指出，绘制系统的阶跃响应可以使用经典的 step() 函数，也可以使用新的 stepplot() 函数绘图，它们之间的区别在于后者允许返回图形的句柄。类似地，impulse()、lsim()、initial() 等函数也有对应的面向对象版本：impulseplot()、lsimplot()、initialplot()，这里将通过例子演示这些函数与经典函数的区别。

例 5-29 试重新绘制例 5-19 系统的阶跃响应曲线，并将图形的标题字号设置为 20。

解 由例 5-19 中的 step() 函数绘制完系统的阶跃响应曲线之后，图形属性是不能修改的，因为不允许单独选择标图或其他对象。这时可以尝试 stepplot() 函数直接绘图，该命令得出与图 5-7(a) 完全一样的曲线，与此同时，还得到了该图的句柄 h。

```
>> G=tf([10 20],[10 23 26 23 10],'ioDelay',1); % 系统模型
   h=stepplot(G,30); % 绘制阶跃响应曲线,并得到句柄
```

由 p=getoptions(h) 命令可以提取该图形下一级对象的句柄，包括其中的标题对象 p.Title。该对象默认的字号为 11。用户只需将其设置为 20，然后用 setoptions() 函数实施修改，直接得出如图 5-17 所示的新阶跃响应曲线。

```
>> p=getoptions(h); p.Title %列出标题的所有属性，其中有 FontSize
   p.Title.FontSize=20; setoptions(h,p) %修改对象属性，获得新图形
```

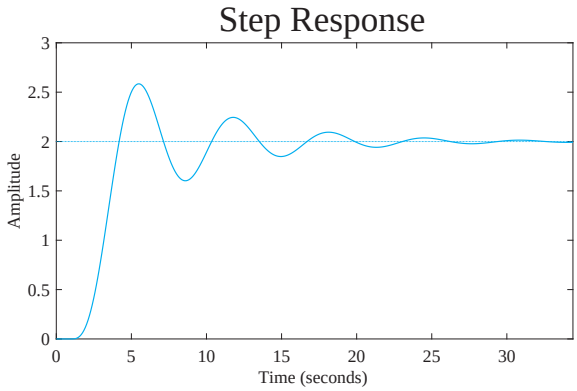


图 5-17　修改标题字号的阶跃响应曲线

5.4　根轨迹分析

系统的根轨迹分析与设计技术是自动控制理论中一种很重要的方法，根轨迹起源于对系统稳定性的研究，在以前没有很好的求特征根的方法时起到一定的作用，现在根轨迹方法仍然是一种较实用的方法。本节先给出根轨迹的概念与绘制方法，再介绍特殊系统根轨迹的绘制方法。

5.4.1　一般系统的根轨迹分析

根轨迹绘制的基本考虑是：假设单变量系统的开环传递函数为 $G(s)$，且设控制器为增益 K，整个控制系统是由单位负反馈构成的闭环系统，这样就可以求出闭环系统的数学模型为 $G_c(s) = KG(s)/(1 + KG(s))$，可以看出，闭环系统的特征根可以由下面的方程求出

$$1 + KG(s) = 0 \tag{5-4-1}$$

并可以变化为多项式方程求根的问题。对指定的 K 值，由数学软件提供的多项式方程求根方法就可以立即求出闭环系统的特征根，改变 K 的值可能得出另外的一组根。对 K 的不同取值，则可能绘制出每个特征根变化的曲线，这样的曲线称为系统的根轨迹。

MATLAB 中提供了 rlocus() 函数，可以直接用于系统的根轨迹绘制，根轨迹函数的调用方法也是很直观的，类似于 step() 函数，常用的函数调用格式为

```
rlocus(G)        %不返回变元将自动绘制根轨迹曲线
```

```
rlocus(G,K)              %给定增益向量,绘制根轨迹曲线
[R,K]=rlocus(G)          %R为闭环特征根构成的复数矩阵
rlocus(G1,'-',G2,'-.b',G3,':r') %同时绘制若干系统的根轨迹
```

该函数可以用于单变量不含有时间延迟的连续、离散系统的根轨迹绘制，也可以用于带有时间延迟的单变量离散系统的根轨迹绘制。

在绘制出的根轨迹上，如果用鼠标单击某个点，将显示出关于这个点的有关信息，包括这点处的增益值，对应的系统特征根的值和可能的闭环系统阻尼比和超调量等，可以通过这样的方法得出临界增益等实用信息。

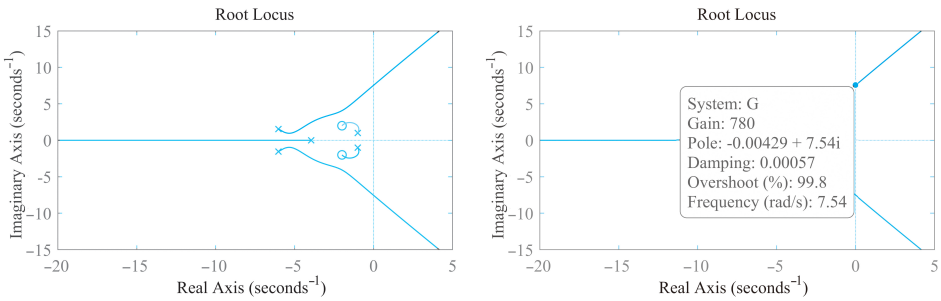
绘制了系统的根轨迹曲线，则给出 grid 命令将在根轨迹曲线上叠印出等阻尼线和等自然频率线，根据等阻尼线可以进行基于根轨迹的系统设计。用户还可以使用 rlocusplot() 函数，用面向对象的方法绘制根轨迹曲线。

例 5-30 假设系统开环传递函数如下，试绘制其根轨迹并求出临界增益。

$$G(s)=\frac{s^2+4s+8}{s^5+18s^4+120.3s^3+357.5s^2+478.5s+306}$$

解 如果不采用计算机工具，直接采用控制理论中介绍的示意图方法则无法绘制此系统的根轨迹，因为高阶系统的零极点是未知的，无法确定根轨迹的起点和终止点。这样的问题用 MATLAB 语言求解就不是难事了，可以先输入系统的传递函数模型，然后调用 rlocus() 函数可以立即绘制出精确的根轨迹，如图 5-18(a)所示。

```
>> num=[1 4 8]; den=[1,18,120.3,357.5,478.5,306];
   G=tf(num,den); rlocus(G)  %绘制系统的根轨迹曲线
```



(a) 根轨迹曲线　　(b) 由根轨迹交互得出临界点

图 5-18　控制系统根轨迹分析和阶跃响应分析

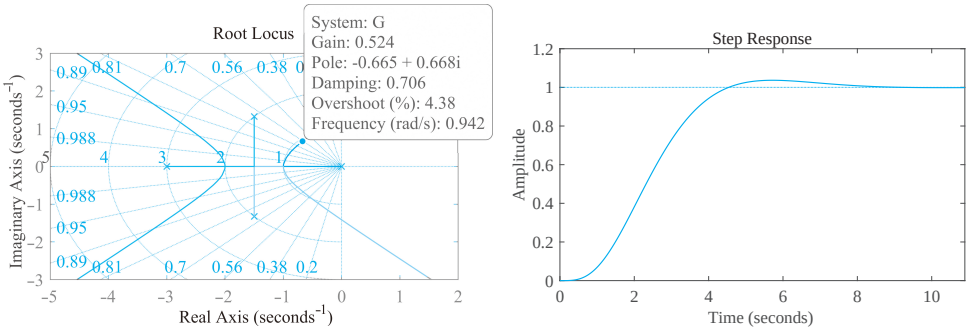
单击根轨迹上的点，则可以显示出该点处的增益值和其他相关信息。例如，若单击根轨迹和虚轴相交的点，则可以得出该点处增益的临界值为 780，如图 5-18(b)所示。可以看出，若系统的增益 $K>780$，则闭环系统将不稳定。

例 5-31 考虑如下的系统开环模型，试设计有较好性能的比例控制器。

$$G(s)=\frac{10}{s(s+3)(s^2+2s+4)}$$

解　通过下面的语句可以输入系统的数学模型，并绘制出系统的根轨迹，如图 5-19(a)所示。在该曲线中，对曲线和等阻尼线进行了处理，使得显示效果更好。

```
>> s=tf('s'); G=10/(s*(s+3)*(s^2+3*s+4));
   rlocus(G), grid  %绘制系统的根轨迹曲线，并绘制等阻尼线
```



（a）根轨迹曲线　　（b）阶跃响应曲线

图 5-19　系统根轨迹和闭环阶跃响应

根据绘制的根轨迹曲线和等阻尼线，可以单击阻尼比 ζ 在 0.707 附近的点，这样可以得出图 5-19(a)所示的结果，可以选择 $K=0.524$，这样用下面的语句可以绘制出系统的阶跃响应曲线，如图 5-19(b)所示。可以看出闭环系统动态性能比较好。

```
>> K=0.524; step(feedback(G*K,1)) %绘制闭环系统的阶跃响应曲线
```

例 5-32　已知离散系统的传递函数模型为

$$G(z)=\frac{-0.95(z+0.51)(z+0.68)(z+1.3)(z^2-0.84z+0.196)}{(z+0.66)(z+0.96)(z^2-0.52z+0.1117)(z^2+1.36z+0.7328)}$$

其采样周期为 $T=0.1\,\mathrm{s}$，试绘制其根轨迹曲线并求出临界增益。

解　可以用下面的语句输入该系统的数学模型。

```
>> z=tf('z','Ts',0.1);  %定义z变换算子
   G=-0.95*(z+0.51)*(z+0.68)*(z+1.3)*(z^2-0.84*z+0.196)/...
        ((z+0.66)*(z+0.96)*(z^2-0.52*z+0.1117)*(z^2+1.36*z+0.7328));
   rlocus(G), grid  %绘制系统的根轨迹
```

系统的根轨迹曲线如图 5-20 所示。根轨迹曲线与单位圆由三个交点，可以利用鼠标单击的方法得出图中信息，可以发现，系统的临界增益为 $K=0.099$。

例 5-33　若离散开环传递函数模型如下，且已知系统的采样周期为 $T=0.1\,\mathrm{s}$，试绘制根轨迹曲线并求出临界增益，如果系统带有 6 不延迟，试重新分析系统。

$$G(z)=\frac{0.52(z-0.49)(z^2+1.28z+0.4385)}{(z-0.78)(z+0.29)(z^2+0.7z+0.1586)}$$

解　可以用下面的语句将其输入到 MATLAB 工作空间，并由 `rlocus()` 函数直接绘制出系统的根轨迹曲线，如图 5-21(a)所示。

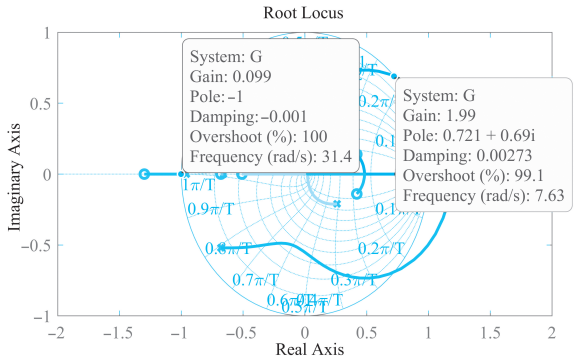
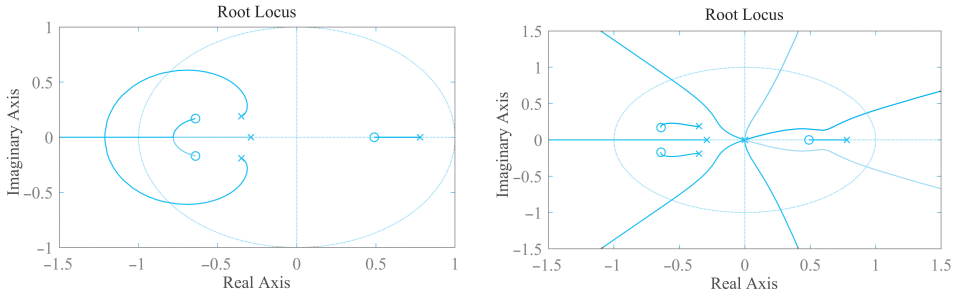


图 5-20　离散系统的根轨迹分析

```
>> z=tf('z','Ts',0.1);
   G=0.52*(z-0.49)*(z^2+1.28*z+0.4385)/...
          ((z-0.78)*(z+0.29)*(z^2+0.7*z+0.1586));
   rlocus(G)     %绘制系统的根轨迹
```



（a）离散系统的根轨迹　　（b）带有延迟系统的根轨迹

图 5-21　离散系统的分析结果

利用grid命令，可以立即得出带有等阻尼线的系统根轨迹曲线。单击左侧和单位圆相交的点还可以得出系统的临界增益值为 2.83，这样可以得出结论：只要 $K < 2.83$，则闭环系统的全部极点均位于单位圆内，这时闭环系统是稳定的。

下面考虑时间延迟的情况，假设系统的传递函数带有 6 步的纯延迟，可以用下面的语句输入系统的新模型，并绘制出时间延迟系统的根轨迹曲线，如图 5-21(b)所示。

```
>> G.ioDelay=6; rlocus(G) %绘制新系统的根轨迹
```

从新系统的根轨迹可以看出，放大倍数 $K < 1.16$，否则闭环系统将不稳定。可见，在引入了纯时间延迟之后，系统的稳定范围将缩小。

5.4.2　正反馈系统的根轨迹

前面介绍的根轨迹绘制都是负反馈系统的根轨迹，如果系统含有正反馈而不是负

反馈，则由特征方程可见

$$1-KG(s)=0 \quad \rightarrow \quad 1+K[-G(s)]=0 \tag{5-4-2}$$

所以用`rlocus(-G)`函数可以直接绘制正反馈系统的根轨迹曲线，方法也很直观。

例5-34 假设开环传递函数如下，试绘制正反馈系统的根轨迹。

$$G(s)=\frac{s^2+5s+6}{s^5+13s^4+65s^3+157s^2+184s+80}$$

解 由下面语句即可绘制出正反馈系统的根轨迹曲线，如图5-22所示。单击根轨迹曲线和虚轴的交点，则可以立即得出使闭环系统临界不稳定的K值，为13.5。亦即当$0 \leqslant K \leqslant 13.5$时闭环系统稳定。

```
>> G=tf([1 5 6],[1 13 65 157 184 80]); rlocus(-G)
```

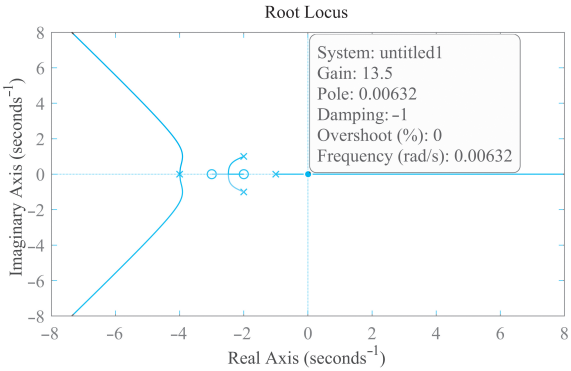


图5-22 正反馈系统的根轨迹分析

5.4.3 延迟系统的根轨迹

对连续延迟系统$G(s)=N(s)\mathrm{e}^{-Ts}/D(s)$而言，其中$N(s)$与$D(s)$为多项式，可以直接写出其特征方程为

$$N(s)\mathrm{e}^{-Ts}+kD(s)=0 \tag{5-4-3}$$

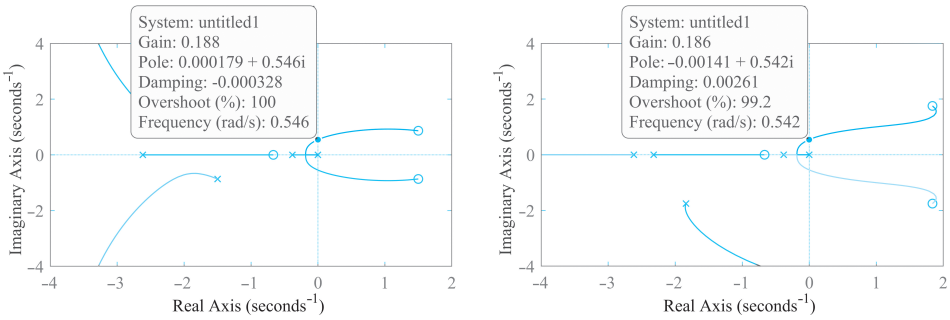
可以看出，如果$T\neq 0$，则特征方程不是多项式方程，即使采用数值方法也难于求解，所以可以考虑使用Padé近似来逼近延迟项，将特征方程转换成多项式方程，绘制出系统的近似根轨迹。由近似根轨迹可以求出系统的临界增益。如果对两个不同的Padé近似阶次临界增益相近，则可以近似认为该临界增益是原延迟系统的临界增益。

例5-35 考虑受控对象模型$G=(6s+4)\mathrm{e}^{-2s}/[s(s^2+3s+1)]$，试求出临界增益。

解 用下面语句可以立即绘制出2阶Padé近似下的近似根轨迹曲线如图5-23(a)所示，局部放大后可以得出近似的临界增益为0.188。

```
>> s=tf('s'); G=(6*s+4)/s/(s^2+3*s+1); G.ioDelay=2;
   rlocus(pade(G,2)) %可以选择不同的阶次绘制近似的根轨迹
```

增大 Padé 近似的阶次，用类似的语句可以得出 3 阶 Padé 近似根轨迹如图 5-23(b)所示，这时得出的临界增益大概等于 0.186，再进一步增加近似的阶次，得出的根轨迹分支增加，但得出的临界增益也差不多，由此可以得出近似的临界增益值。选择不同的 Padé 近似阶次，得出的临界增益都差不多，都为 0.186 左右。



（a）二阶 Padé 近似下根轨迹　　（b）三阶近似根轨迹

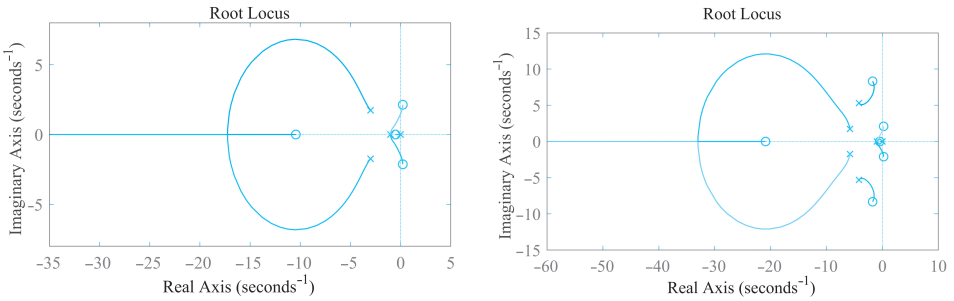
图 5-23　不同近似阶次下的根轨迹曲线

其实，利用 Padé 近似技术还可以绘制更复杂系统的近似根轨迹，不局限于 $G(s)=N(s)\mathrm{e}^{-Ts}/D(s)$ 类型的开环模型。这里将给出例子研究近似根轨迹的绘制与应用。

例 5-36　考虑例 5-3 中描述的开环系统，试绘制近似根轨迹并得出临界增益。

解　如果选择 Padé 近似的阶次为 2，则由下面的语句可以绘制开环系统的近似根轨迹曲线，如图 5-24(a)所示。局部放大根轨迹曲线与虚轴交点处的根轨迹曲线，可以得出近似临界增益为 5.09。

```
>> s=tf('s'); G=(1+3*exp(-s)/(s+1))/(s+1);
   Gc=0.3+0.15/s; rlocus(pade(G*Gc,2))
```



（a）二阶 Padé 近似下根轨迹　　（b）四阶近似根轨迹

图 5-24　不同近似阶次下的根轨迹曲线

如果将 Padé 近似的阶次增加到 4，则得出的近似根轨迹如图 5-24(b)所示，这时可以通过局部放大得出临界增益为 5.05，尝试再高阶次的 Padé 近似，则能得出类似的结论。这样的结论还可以通过仿真证实，因为若选择增益 $K=5.05$，则可以得出闭环等幅

振荡的阶跃响应曲线(曲线从略),说明这样的方法是可行的。

```
>> K=5.05; step(feedback(K*G*Gc,1))
```

5.4.4　系统对参数的根轨迹

假设某传递函数模型含有参数 a，则可以把考虑外部增益 K，写出新的特征方程 $1+G(s)=0$。由于 $G(s)$ 含有参数 a 且为有理函数，总是可以通过手工变换将方程变换成 $1+a\widetilde{G}(s)=0$，这样就可以借助于 `rlocus`($\widetilde{G}$) 函数绘制关于参数 a 的根轨迹了。本节将通过例子演示参数根轨迹的绘制方法。

例 5-37　考虑下面给出的传递函数,试绘制出关于参数 a 的根轨迹。

$$G(s)=\frac{5(s+5)(s^2+6s+12)}{(s+a)(s^3+4s^2+3s+2)}$$

解　记 $N_1(s)=5(s+5)(s^2+6s+12)$, $D_1(s)=s^3+4s^2+3s+2$,则可以将系统的特征方程直接改写为

$$1+\frac{N_1(s)}{(s+a)D_1(s)}=0 \;\Rightarrow\; N_1(s)+(s+a)D_1(s)=0 \;\Rightarrow\; \big[N_1(s)+sD_1(s)\big]+aD_1(s)=0$$

可以推导出 $1+a\widetilde{G}(s)=0$,其中

$$\widetilde{G}(s)=\frac{D_1(s)}{N_1(s)+sD_1(s)}$$

这样就可以使用下面的语句绘制出关于参数 a 的根轨迹曲线,如图 5-25 所示。可以看出,因为根轨迹曲线与虚轴没有交点,所以无论 $a\geqslant 0$ 取何值,闭环系统都是稳定的。

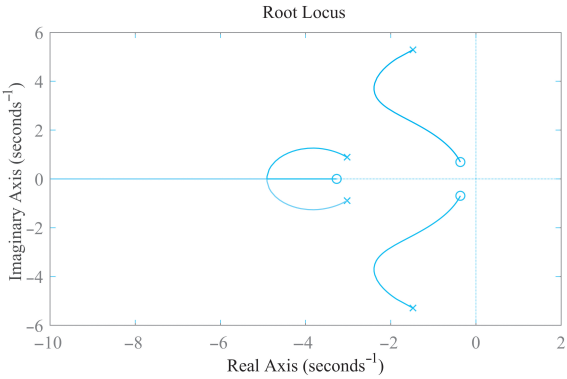


图 5-25　关于参数 a 的根轨迹

```
>> s=tf('s'); N1=5*(s+5)*(s^2+6*s+12); D1=s^3+4*s^2+3*s+2;
   G1=D1/(N1+s*D1); rlocus(G1)
```

5.5　线性系统频域分析

系统的频域分析是控制系统分析中一种重要的方法,早在 1932 年,Nyquist 提出了一种频域响应的绘图方法,并提出了可以用于系统稳定性分析的 Nyquist 定理[8],Bode

提出了另一种频率响应的分析方法，同时可以分析系统的幅值相位与频率之间的关系，又称为Bode图[9]，Nichols在Bode图的基础上又进行了重新定义，构成了Nichols图[10]。这些方法曾经是单变量系统频域分析中最重要的几种方法，在系统的分析和设计中起着重要的作用。由于多变量系统的信号之间相互耦合，如果想对某对输入输出信号单独设计控制器不是件容易的事，需要引入解耦。本节将介绍单变量系统的频域分析，基于Nyquist定理的稳定性分析，多变量系统的逆Nyquist阵列分析与对角占优的概念，并将介绍频域稳定性裕度的分析。

5.5.1 单变量系统的频域分析

对系统的传递函数模型$G(s)$来说，若用频率$\mathrm{j}\omega$取代复变量s，则可以将$G(\mathrm{j}\omega)$看成增益，这个增益是复数量，是ω的函数。描述这个复数变量有几种方法，根据表示方法的不同，就可以构造出不同的频域响应曲线：

1. 实虚部形式

可以将复数分解为实部和虚部，它们分别是频率ω的函数，这时

$$G(\mathrm{j}\omega)=P(\omega)+\mathrm{j}Q(\omega) \tag{5-5-1}$$

若用横轴表示实数，纵轴表示虚数，则可以将增益$G(\mathrm{j}\omega)$在复数平面上表示出来，这样的曲线称为Nyquist图，该图是分析系统稳定性和一些性能的有效工具，现在仍然在使用。传统Nyquist图中未提供频率信息，这不能不说是传统Nyquist图的缺陷，因为某些点的频率信息在系统设计中是有用的。

在MATLAB下提供了一个`nyquist()`函数，可以直接绘制系统的Nyquist图。该函数的常用调用格式为

```
nyquist(G)                  % 不返回变元将自动绘制 Nyquist 图
nyquist(G,{ωm,ωM})          % 给定频率范围绘制 Nyquist 图
nyquist(G,ω)                % 给定频率向量 ω 绘制 Nyquist 图
[R,I,ω]=nyquist(G)          % 计算 Nyquist 响应数值
nyquist(G1,'-',G2,'-.b',G3,':r') % 绘制几个系统的 Nyquist 图
```

用户可以单击Nyquist图上的点，显示该点处增益与频率之间的关系，MATLAB提供的工具给传统的Nyquist图又赋予了新的特色。改写的`grid`命令可以在Nyquist图上叠印出等M圆。

和`step()`函数类似，`nyquist()`函数也允许在指定坐标系下绘制曲线。也可以用面向对象的`nyquistplot()`函数绘制Nyquist图。

2. 幅值相位形式

复数量$G(\mathrm{j}\omega)$可以分解为幅值和相位的形式，即

$$G(\mathrm{j}\omega)=A(\omega)\mathrm{e}^{-\mathrm{j}\phi(\omega)} \tag{5-5-2}$$

这样，以频率ω为横轴，幅值$A(\omega)$为纵轴，则可以构造出幅值和频率之间的关系曲线，又称为幅频特性。若以频率ω为横轴，幅值$\phi(\omega)$为纵轴，则可以构造出相位和频率之间的关系曲线，又称为相频特性。在实际系统分析中，常用对数形式表示横轴，其单位常用rad/s，幅频特性中幅值进行对数变换，即$M(\omega)=20\lg[A(\omega)]$，其单位是分贝（dB），相频特性中，相位的单位常取作角度，这样绘制出的图形称为系统的Bode图。

MATLAB的控制系统工具箱中提供了`bode()`函数，可以直接绘制系统的Bode图。该函数的常用调用格式为

```
bode(G)                          % 不返回变元将自动绘制 Bode 图
bode(G,{ωm,ωM})                  % 给定频率范围绘制 Bode 图
bode(G,ω)                        % 给定频率向量 ω 绘制 Bode 图
[A,φ,ω]=bode(G)                  % 计算 Bode 响应数值
bode(G1,'-',G2,'-.b',G3,':r')    % 同时绘制若干系统的 Bode 图
```

和Nyquist图不同的是，Bode图可以同时绘制出系统增益、相位与频率之间的关系，所以相比之下，Bode图提供的信息量更大。

3. 其他描述

还是采用幅值、相位的描述方法，用横轴表示相位，用纵轴表示单位为dB的幅值，就可以绘制出另一种图形，这样的图形称为Nichols图。

在MATLAB控制系统工具箱中，用`nichols()`函数可以绘制出系统的Nichols图，该函数的调用格式与`bode()`完全一致。这时的`grid`函数可以叠印出等幅值曲线和等相位曲线。也可以使用面向对象的`bodeplot()`和`nicholsplot()`函数绘制系统的频域响应曲线。

对离散系统$H(z)$来说，可以将$z=\mathrm{e}^{\mathrm{j}\omega T}$代入传递函数模型，就可以得出频率和增益$\widehat{H}(\mathrm{j}\omega)$之间的关系。MATLAB中提供的各种频域响应分析函数，如`nyquist()`等，同样直接适用于离散的系统模型。

例5-38 考虑连续线性系统的传递函数模型，试绘制其Nyquist图。

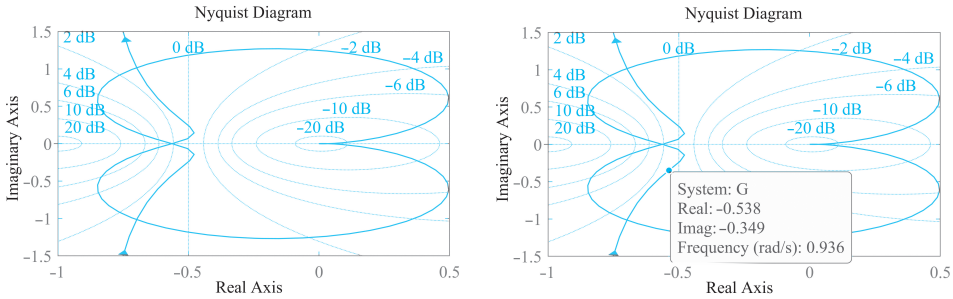
$$G(s)=\frac{s+8}{s(s^2+0.2s+4)(s+1)(s+3)}$$

解 可以通过下面的命令绘制出系统的Nyquist图，并叠印等幅值圆。

```
>> s=tf('s'); G=(s+8)/(s*(s^2+0.2*s+4)*(s+1)*(s+3));
   nyquist(G), grid %绘制 Nyquist 图并叠印等幅值圆
   ylim([-1.5 1.5]) %根据需要手动选择纵坐标范围
```

由于系统含有位于$s=0$处的极点，所以若ω较小时，增益的幅值很大，远离单位圆，因此单位圆附近的Nyquist图形看得不是很清楚，因此应该给出相应的语句对得出的Nyquist图进行局部放大，如图5-26(a)所示。

传统的Nyquist图不能显示出增益幅值和频率ω之间的关系，而用MATLAB提供的工具允许用户用单击的方式选择Nyquist图上的点，这时将同时显示该点处的频率、



(a) 系统的 Nyquist 图　　(b) 从 Nyquist 图上读取附加信息

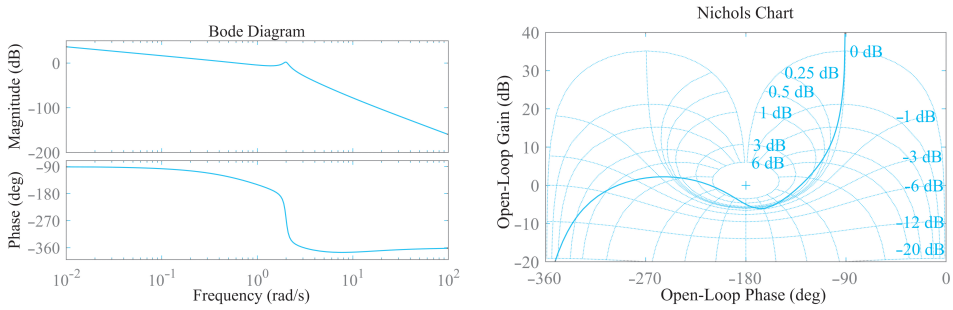
图 5-26　系统的频域响应分析结果

增益以及闭环系统超调量等信息,如图 5-26(b)所示。这样的工具为 Nyquist 图这一传统的工具赋予了新的功能,将有助于系统的频域分析。

例 5-39　考虑例 5-38 中给出的传递函数模型,试绘制 Bode 图与 Nichols 图。

解　若给出下面的命令,则将绘制出系统的 Bode 图和 Nichols 图,如图 5-27 所示。可以看出,这样的函数对系统的频域分析提供了很多的方便。

```
>> s=tf('s'); G=(s+8)/(s*(s^2+0.2*s+4)*(s+1)*(s+3)); bode(G);
   figure; nichols(G), grid  %绘制系统的 Nichols 图,并叠印等幅值线
```



(a) 系统的 Bode 图　　(b) 系统的 Nichols 图

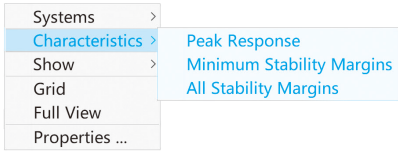
图 5-27　系统的频域响应分析结果

MATLAB 提供的这些函数都允许用户选择特性分析功能,例如,在系统的 Bode 图上,若右击鼠标则得出快捷菜单,其 Characteristics 菜单项的内容如图 5-28(a)所示,从中可以选择稳定性相关的菜单项,则将得出如图 5-28(b)所示的 Bode 图。其他的几个函数如 `nyquist()` 和 `nichols()` 等,都支持自己的 Characteristics 菜单选择。

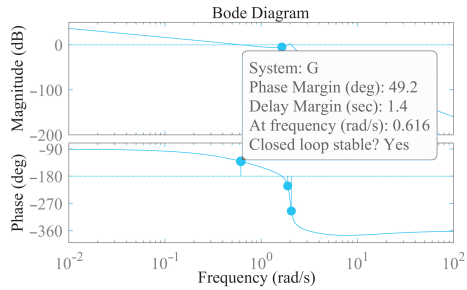
例 5-40　再考虑前面例子中的连续系统,选择采样周期 $T = 0.1$ s,试比较原系统与离散化系统的 Bode 图。

解　给出下面的命令,则可以得出离散化模型,该模型的 Bode 图可以用同样的命令直接绘制出来,如图 5-29(a)所示。

```
>> s=tf('s'); G=(s+8)/(s*(s^2+0.2*s+4)*(s+1)*(s+3));
   G1=c2d(G,0.1); bode(G,'-',G1,'--')
```



（a）频率响应特性显示菜单

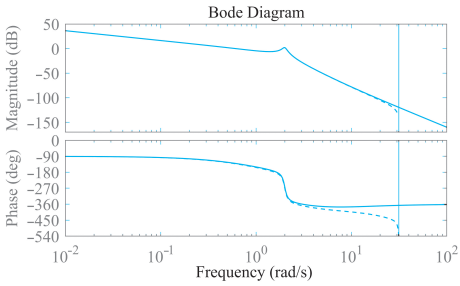


（b）系统的 Bode 图

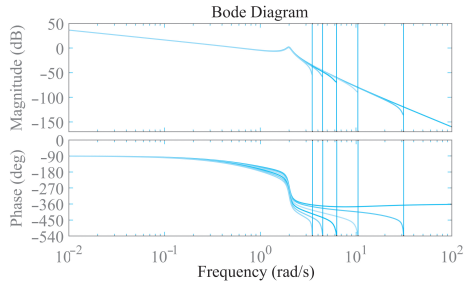
图 5-28　系统的频域响应分析结果

选择不同的采样周期，则可以得出如图 5-29（b）所示的 Bode 图。随着采样周期的不同选择，可以得出不同的 Bode 图。可见，低频时离散模型接近连续模型。采样周期越大，则高频响应与连续模型的差异越大。因为高频段对应于时域的初始响应，所以采样周期越大，开始时段系统的时域响应越不精确。

```
>> bode(G), hold on; for T=[0.1:0.2:1], bode(c2d(G,T)); end
```



（a）$T=0.1$ 时离散化系统的 Bode 图



（b）不同采样周期下系统的 Bode 图

图 5-29　离散化系统的 Bode 图

例 5-41　考虑离散系统的传递函数模型

$$G(z)=\frac{0.2(0.3124z^3-0.5743z^2+0.3879z-0.0889)}{z^4-3.233z^3+3.9869z^2-2.2209z+0.4723}$$

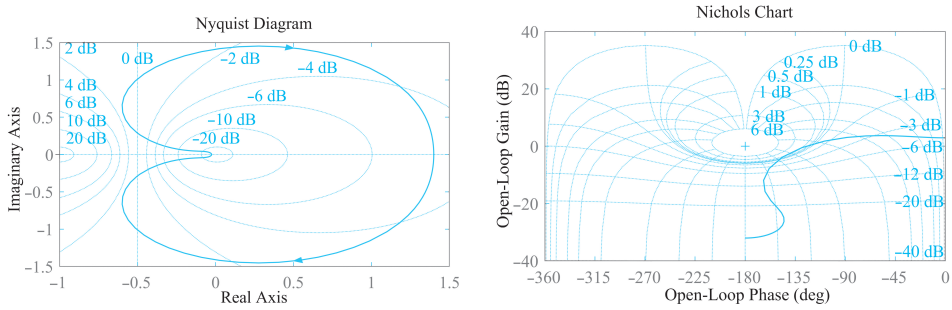
且已知系统的采样周期为 $T=0.1\,\mathrm{s}$，试绘制 Nyquist 与 Nichols 图。

解　可以用下面的语句将其输入到 MATLAB 工作空间，并将系统的 Nyquist 图、Nichols 图直接绘制出来，如图 5-30 所示。从这个例子可以看出，绘制离散系统的频域响应曲线也是很容易的。

```
>> num=0.2*[0.3124 -0.5743 0.3879 -0.0889];
   den=[1 -3.233 3.9869 -2.2209 0.4723];
   G=tf(num,den,'Ts',0.1); nyquist(G); grid %绘制系统的 Nyquist 图
   figure, nichols(G), grid                %绘制系统的 Nichols 图
```

例 5-42　试绘制带有时间延迟传递函数模型 $G(s)=\mathrm{e}^{-2s}/(s+1)$ 的 Nyquist 图。

解　若只想获得 $\omega\in[0.1,10000]$ 区间的频域点，则不能再依赖 `nyquist()` 函数的默认调



（a）系统的 Nyquist 图　　（b）系统的 Nichols 图

图 5-30　离散系统的频域响应分析结果

用，而需要自己选定频率向量，从而得到一个分支的 Nyquist 图，以便更好地观测时间延迟系统的 Nyquist 图。可以给出如下的 MATLAB 语句。

```
>> G=tf(1,[1 1],'ioDelay',2); %输入系统的传递函数模型
   w=logspace(-1,4,2000);      %按照对数等分的原则选择 2000 个频率点
   [x,y]=nyquist(G,w); plot(x(:),y(:)) %绘制系统的 Nyquist 曲线
```

这样就可以绘制出系统的 Nyquist 图，如图 5-31 所示。在这样得出的 Nyquist 图中，`grid` 命令并不能给出等幅值圆，因为这个图形不是 `nyquist()` 函数自动绘制的。另外应该注意本图所示的时间延迟系统 Nyquist 图的典型形状。

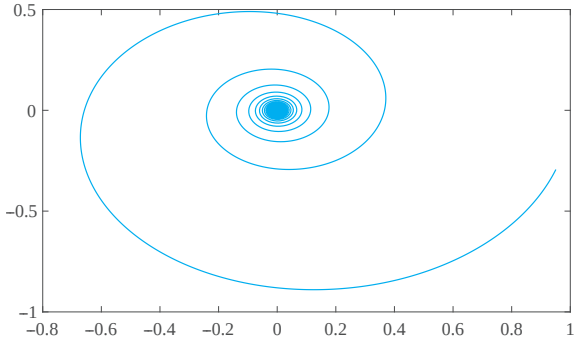


图 5-31　时间延迟系统的 Nyquist 图

5.5.2　带有内部延迟模型的频域响应分析

正如前面指出的那样，只要单变量系统可以用 LTI 的模型形式描述处理，即使该模型带有内部延迟等难以处理的环节，也可以完全采用相同的 `bode()` 等函数，直接对系统进行频域响应分析。这里将给出一个简单例子演示内部延迟模型的 Bode 图绘制与分析方法。

例 5-43　考虑例 5-3 中描述的开环系统，试绘制该系统的 Bode 图。

解　输入系统的开环模型就可以绘制出系统的 Bode 图，如图 5-32 所示。由于该系统对应的是延迟微分方程模型，所以其 Bode 图的走行方式与一般无延迟模型的 Bode 图看起

来有明显的区别。

```
>> s=tf('s'); G=(1+3*exp(-s)/(s+1))/(s+1);
   Gc=0.3+0.15/s; bode(G*Gc), K0=10^(14.1/20)
```

其实，由图中得出的幅值裕度 14.1dB 也可以推算出系统的临界增益为 $K_0 = 5.0699$ 左右，与前面得出的根轨迹结果是吻合的。不过从信息显示看，尽管找到了剪切点信息，但用现有的工具无法判定带有内部延迟的闭环系统的稳定性。

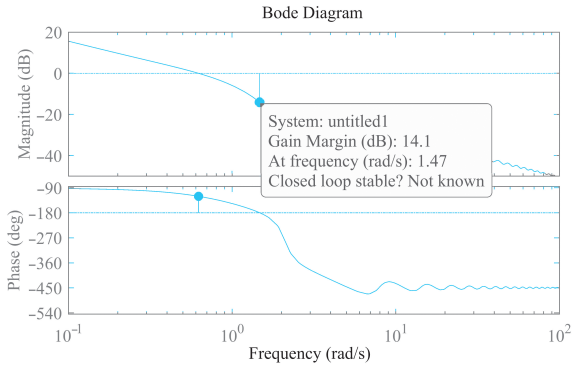


图 5-32　含有内部延迟系统的 Bode 图

5.5.3　利用频率特性分析系统的稳定性

频域响应的分析方法最早应用就是利用开环系统的 Nyquist 图来判定闭环系统的稳定性，其稳定性分析的理论基础是 Nyquist 稳定性定理。Nyquist 定理的内容是：如果开环模型含有 m 个不稳定极点，则单位负反馈下单变量闭环系统稳定的充要条件是开环系统的 Nyquist 图逆时针围绕 $(-1, \mathrm{j}0)$ 点 m 周。

Nyquist 定理可以分下面两种情况进一步解释为：

（1）若系统的开环模型 $G(s)H(s)$ 为稳定的，则当且仅当 $G(s)H(s)$ 的 Nyquist 图不包围 $(-1, \mathrm{j}0)$ 点，闭环系统为稳定的。如果 Nyquist 图顺时针包围 $(-1, \mathrm{j}0)$ 点 p 次，则闭环系统有 p 个不稳定极点。

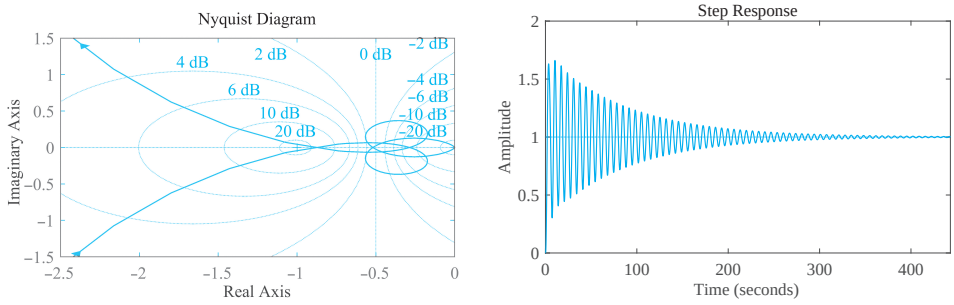
（2）如果系统的开环模型 $G(s)H(s)$ 是不稳定的，且有 p 个不稳定极点，则当且仅当 $G(s)H(s)$ 的 Nyquist 图逆时针包围 $(-1, \mathrm{j}0)$ 点 p 次，闭环系统为稳定的。若 Nyquist 图逆时针包围 $(-1, \mathrm{j}0)$ 点 q 次，则闭环系统有 $p - q$ 个不稳定极点。

例 5-44　试绘制下面连续传递函数模型的 Nyquist 图，并绘制闭环系统的阶跃响应曲线。

$$G(s) = \frac{2.7778(s^2 + 0.192s + 1.92)}{s(s+1)^2(s^2 + 0.384s + 2.56)}$$

解　用下面的语句即可输入系统模型，并绘制出系统的 Nyquist 曲线，如图 5-33(a)所示。

```
>> s=tf('s');
   G=2.7778*(s^2+0.192*s+1.92)/(s*(s+1)^2*(s^2+0.384*s+2.56));
   nyquist(G); axis([-2.5,0,-1.5,1.5]); grid %绘制 Nyquist 图
```



（a）系统的 Nyquist 图　　（b）闭环系统的阶跃响应

图 5-33　给定系统的分析

从得出的 Nyquist 图可以看出，尽管该图走向较复杂，但可以看出，整个 Nyquist 图并不包围 $(-1, \mathrm{j}0)$ 点，且因为开环系统不含有不稳定极点，所以根据 Nyquist 定理可以断定，闭环系统是稳定的。可以绘制出闭环系统的阶跃响应曲线，如图 5-33(b)所示。

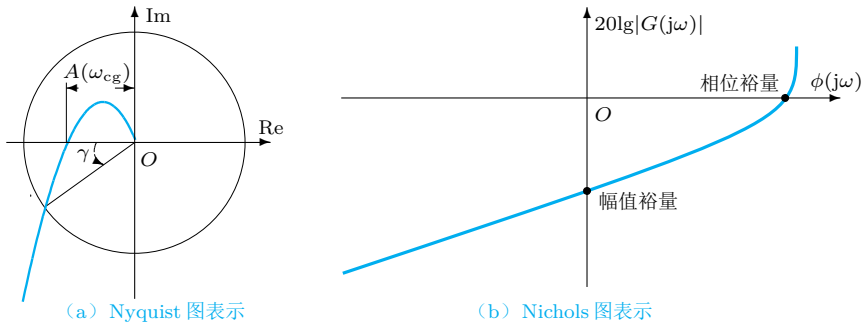
```
>> step(feedback(G,1))  % 闭环系统阶跃响应
```

可以看出，虽然闭环系统是稳定的，但其阶跃响应的振荡是很强的，所以，该系统并不是很令人满意的，对这样的系统需要给其设计一个控制器改善其性能。

5.5.4　系统的幅值裕度和相位裕度

从前面给出的例子可以看出，系统的稳定性固然重要，但它不是唯一刻画系统性能的准则，因为有的系统即使稳定，但其动态性能表现为很强的振荡，也是没有用途的。另外，如果系统的增益出现变化，比如增大很小的值，都可能使该模型的 Nyquist 图发生延伸，最终包围 $(-1, \mathrm{j}0)$ 点，导致闭环系统不稳定。基于频域响应裕度的定量分析方法是解决这类问题的一种比较有效的途径。

在图 5-34(a)、(b)中分别给出了在 Nyquist 图和 Nichols 图上幅值裕度与相位裕度的图形表示，在 Bode 图上也应该有相应的解释。



（a）Nyquist 图表示　　（b）Nichols 图表示

图 5-34　系统幅值相位裕度的图形表示

若当系统的 Nyquist 图在频率 ω_{cg} 时与负实轴相交，则将该频率下幅值的倒数，

即 $G_{\mathrm{m}}=1/A(\omega_{\mathrm{cg}})$，定义为系统的幅值裕度。若假设系统的 Nyquist 图与单位圆在频率 ω_{cp} 处相交，且记该频率下的相位角度为 $\phi(\omega_{\mathrm{cp}})$，则系统的相位裕度定义为 $\gamma=\phi(\omega_{\mathrm{cp}})-180^\circ$。

可以看出，一般若幅值裕度 G_{m} 的值越大，则对扰动的抑制能力就越强。如果 $G_{\mathrm{m}}<1$，则闭环系统是不稳定的。同样，若相位裕度的值越大，则系统对扰动的抑制能力也越强。如果 $\gamma<0$，则闭环系统不稳定。下面再考虑几种特殊的情形：

（1）如果系统的 Nyquist 图不与负实轴相交，则系统的幅值裕度为无穷大。

（2）如果系统的 Nyquist 图与负实轴在 $(-1,\mathrm{j}0)$ 与 $(0,\mathrm{j}0)$ 这两个点之间有若干交点，则系统的幅值裕度以离 $(-1,\mathrm{j}0)$ 最近的点为准。

（3）如果系统的 Nyquist 图不与单位圆相交，则系统的相位裕度为无穷大。

（4）如果系统的 Nyquist 图在第三象限与单位圆有若干交点，则系统的相位裕度以与离负实轴最近的为准。

MATLAB 控制系统工具箱中提供了 `margin()` 函数，可以直接用于系统的幅值与相位裕度的求取，该函数的调用格式为 `[`G_{m}`,`γ`,`ω_{cg}`,`ω_{cp}`]=margin(`G`)`。在得出的结果中，如果某个裕度为无穷大，则返回 `Inf`，相应的频率值为 `NaN`。

例 5-45　考虑例 5-44 中研究的开环对象模型，试求复制与相位裕度。

解　可以用下面语句输入系统模型，并对系统的频域响应裕度进行分析。

```
>> s=tf('s');
   G=2.7778*(s^2+0.192*s+1.92)/(s*(s+1)^2*(s^2+0.384*s+2.56));
   [gm,pm,wg,wp]=margin(G) %计算系统的频域响应裕度
```

可以得出系统的幅值裕度为 1.105，频率为 0.962 rad/s，相位裕度为 2.0985°，剪切频率为 0.926 rad/s，由于幅值、相位裕度偏小，系统的闭环响应将有强振荡。

5.6　多变量系统的频域分析

前面的系统分析一般均侧重于单变量系统，随着控制理论的发展和过程控制的实际需要，多变量系统分析与设计成了 20 世纪 70～80 年代控制理论领域的热门研究主题，出现了各种各样的分析与设计方法。这里将着重探讨多变量频域分析方法及其 MATLAB 语言解决方法，介绍 Nyquist 图与奇异值曲线的绘制方法。

5.6.1　多变量系统频域分析概述

在开始介绍控制系统理论中的多变量系统频域分析方法之前，将先通过例子来演示用 MATLAB 的控制系统工具箱函数的直接使用与分析的结果。

例 5-46 考虑下面给出的多变量系统模型[11]，试绘制其 Nyquist 图。

$$\boldsymbol{G}(s)=\begin{bmatrix}\dfrac{0.806s+0.264}{s^2+1.15s+0.202} & \dfrac{-15s-1.42}{s^3+12.8s^2+13.6s+2.36}\\ \dfrac{1.95s^2+2.12s+0.49}{s^3+9.15s^2+9.39s+1.62} & \dfrac{7.15s^2+25.8s+9.35}{s^4+20.8s^3+116.4s^2+111.6s+18.8}\end{bmatrix}$$

解 可以通过下面语句直接输入系统的传递函数矩阵，并用 MATLAB 控制系统工具箱提供的函数 `nyquist()` 直接绘制出该多变量系统的 Nyquist 图，如图 5-35 所示。

```
>> g11=tf([0.806 0.264],[1 1.15 0.202]);
   g12=tf([-15 -1.42],[1 12.8 13.6 2.36]);
   g21=tf([1.95 2.12 0.49],[1 9.15 9.39 1.62]);
   g22=tf([7.15 25.8 9.35],[1 20.8 116.4 111.6 18.8]);
   G=[g11, g12; g21, g22]; nyquist(G) %绘制 Nyquist 图
```

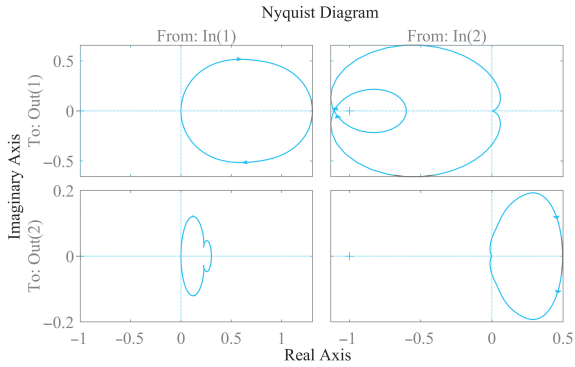


图 5-35 多变量系统的 Nyquist 图

上述的 `nyquist()` 等函数事实上不大适用于多变量系统的频域分析，虽然它们可以直接绘制出一种 Nyquist 曲线，但对多变量系统的分析没有太大的帮助。针对多变量系统的频域分析，英国学者 Howard H Rosenbrock[12]、Alistair G J MacFralane[13] 等教授分别提出了不同的多变量频域分析与设计算法，形成了有重要影响的英国学派（British School），其中以 Rosenbrock 教授为代表的一类利用逆 Nyquist 阵列（inverse Nyquist array，INA）的方法是其中有影响的方法。

英国剑桥大学学者 Boyel 和 Maciejowski 等推出的多变量频域设计（Multivariable Frequency Design，MFD）工具箱[14] 很适合于求解频域设计问题，它提供了一系列函数来对频域模型进行分析。在 MFD 工具箱中，很多函数需要已知多变量传递函数矩阵的公分母，所以直接求解起来较困难，故可以用 MFD 工具箱中的 `mvss2tf()` 函数直接求出 `[N,d]=mvss2tf(A,B,C,D)`，其中，$\boldsymbol{d}$ 为传递函数矩阵的公分母，$\boldsymbol{N}$ 为传递函数矩阵的分子，而系统的状态方程模型可以由 `ss()` 函数得出。

例5-47 试求出下面2输入2输出传递函数矩阵的公分母模型。

$$\boldsymbol{G}(s)=\begin{bmatrix}\dfrac{s+4}{(s+1)(s+5)} & \dfrac{1}{5s+1}\\ \dfrac{s+1}{s^2+10s+100} & \dfrac{2}{2s+1}\end{bmatrix}$$

解 由上面的模型可以很容易地求出系统的公分母和传递函数矩阵。

```
>> s=tf('s'); g11=(s+4)/((s+1)*(s+5)); g21=(s+1)/(s^2+10*s+100);
   g12=1/(5*s+1); g22=2/(2*s+1); G1=ss([g11 g12; g21 g22]);
   G1=minreal(G1); [N,d]=mvss2tf(G1.a,G1.b,G1.c,G1.d) %建议最小实现
```

可以得出传递函数矩阵的公分母为

$$d(s)=s^6+16.7s^5+176.3s^4+767.1s^3+971.5s^2+415s+50$$

且分子多项式矩阵$\boldsymbol{N}(s)$的数学形式为

$$\begin{bmatrix}s^5+14.7s^4+149.9s^3+499.4s^2+294s+40 & 0.2s^5+3.3s^4+34.6s^3+146.5s^2+165s+50\\ s^5+7.7s^4+16s^3+13.4s^2+4.6s+0.5 & s^5+16.2s^4+168.2s^3+683s^2+630s+100\end{bmatrix}$$

注意，这样的变换方式只适用于不带时间延迟的模型。如果某传递函数矩阵含有延迟，则可以先用不含有时间延迟的状态方程模型先表示出来，延迟时间常数由一个单独的延迟矩阵描述。

5.6.2 多变量系统对角占优分析

假设多变量反馈系统的前向通路传递函数矩阵为$\boldsymbol{Q}(s)$，反馈通路的传递函数矩阵为$\boldsymbol{H}(s)$，则闭环系统的传递函数矩阵为

$$\boldsymbol{G}(s)=\left[\boldsymbol{I}+\boldsymbol{Q}(s)\boldsymbol{H}(s)\right]^{-1}\boldsymbol{Q}(s) \tag{5-6-1}$$

其中$\boldsymbol{I}+\boldsymbol{Q}(s)\boldsymbol{H}(s)$称为系统的回差（return difference）矩阵。因为稳定性分析利用回差矩阵的逆矩阵性质，所以在频域分析中用逆Nyquist分析更方便，由此出现了在多变量频域分析系统中的逆Nyquist阵列[12]方法。

Gershgorin定理是基于Nyquist阵列的多变量设计方法的核心。假设

$$\boldsymbol{C}=\begin{bmatrix}c_{11} & \cdots & c_{1k} & \cdots & c_{1n}\\ \vdots & \ddots & \vdots & \ddots & \vdots\\ c_{k1} & \cdots & c_{kk} & \cdots & c_{kn}\\ \vdots & \ddots & \vdots & \ddots & \vdots\\ c_{n1} & \cdots & c_{nk} & \cdots & c_{nn}\end{bmatrix} \tag{5-6-2}$$

为复数矩阵，其特征根λ满足

$$|\lambda-c_{kk}|\leqslant\sum_{j\neq k}|c_{kj}|,\quad \text{且}\quad |\lambda-c_{kk}|\leqslant\sum_{j\neq k}|c_{jk}| \tag{5-6-3}$$

换句话说，该矩阵的特征值位于一族以c_{kk}为圆心，以不等式右面的表达式为半径的圆构成的并集内，而这些圆又称为Gershgorin圆。这两个不等式表示的关系分别称为列Gershgorin圆和行Gershgorin圆。Gershgorin定理的示意图如图5-36所示。

其实,对传统的 Gershgorin 定理直接拓展,就可能得出更小半径的圆。

$$|\lambda - c_{kk}| \leqslant \min\left(\sum_{j\neq k}|c_{kj}|, \sum_{j\neq k}|c_{jk}|\right) \tag{5-6-4}$$

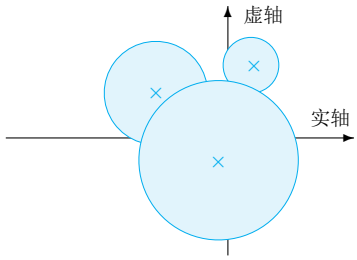


图 5-36 Gershgorin 定理示意图

对于频率响应的所有数据来说,将由一系列 Gershgorin 圆的包络线可以构成 Gershgorin 带,若对全部的 ω 来说,各个对角元素的 Gershgorin 带均不包含原点,则称原系统为对角占优系统。

选定了频率向量 $\boldsymbol{w}$,并已知系统的多变量系统模型,则可以用多变量频域设计工具箱中提供的 `mv2fr()` 函数直接获得系统的频域响应数据。

$\boldsymbol{H}$=mv2fr($\boldsymbol{N}$,d,$\boldsymbol{w}$), $\boldsymbol{H}$=mv2fr($\boldsymbol{A}$,$\boldsymbol{B}$,$\boldsymbol{C}$,$\boldsymbol{D}$,$\boldsymbol{w}$)

其中,返回的 $\boldsymbol{H}$ 是由多变量频率响应数据构成的矩阵,是多变量频域设计工具箱的基本数据格式。该工具箱提供了多变量系统的 Nyquist 图形绘制函数 `plotnyq()` 和 Gershgorin 带绘制的函数 `fgersh()`,但由于调用过程较烦琐,所以对输入个数与输出个数相等的系统来说,我们编写了一个新的函数 gershgorin($\boldsymbol{H}$),可以直接绘制出系统带有 Gershgorin 带的 Nyquist 图,该函数的内容如下:

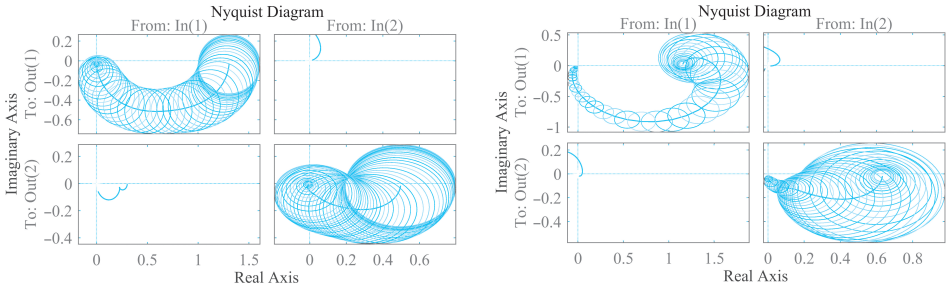
```
function gershgorin(H,key)
if nargin==1, key=0; end
t=[0:.1:2*pi,2*pi]'; [nr,nc]=size(H); nw=nr/nc; ii=1:nc;
for i=1:nc, circles{i}=[]; end
for k=1:nw  %计算各个频率下的Nyquist阵列
   G=H((k-1)*nc+1:k*nc,:);
   if nargin==2 && key==1, G=inv(G); end, H1(:,:,k)=G;
   for j=1:nc, ij=find(ii~=j);
      v=min([sum(abs(G(ij,j))),sum(abs(G(j,ij)))]);
      x0=real(G(j,j)); y0=imag(G(j,j));
      r=sum(abs(v)); %计算Gershgorin圆盘的半径
      circles{j}=[circles{j} x0+r*cos(t)+sqrt(-1)*(y0+r*sin(t))];
end, end
hold off; nyquist(tf(zeros(nc)),'w'); hold on;
h=get(gcf,'child'); h0=h(end:-1:2);
for i=ii, for j=ii
   axes(h0((j-1)*nc+i)); NN=H1(i,j,:); NN=NN(:);
   if i==j  %对角元素绘制Gershgorin圆
      cc=circles{i}(:); x1=min(real(cc)); x2=max(real(cc));
      y1=min(imag(cc)); y2=max(imag(cc)); plot(NN)
      plot(circles{i}), plot(0,0,'+'), axis([x1,x2,y1,y2])
```

```
    else, plot(NN), end     %非对角元素绘制
end, end, hold off
```

例5-48 再考虑例5-46中的多变量系统模型，试重新绘制Nyquist曲线。

解 用下面的语句可以绘制叠印Gershgorin带的Nyquist曲线，如图5-37(a)所示。

```
>> g11=tf([0.806 0.264],[1 1.15 0.202]);
   g12=tf([-15 -1.42],[1 12.8 13.6 2.36]);
   g21=tf([1.95 2.12 0.49],[1 9.15 9.39 1.62]);
   g22=tf([7.15 25.8 9.35],[1 20.8 116.4 111.6 18.8]);
   G=[g11, g12; g21, g22]; w=logspace(-2,1.5);
   G=ss(G); H=mv2fr(G.a,G.b,G.c,G.d,w); gershgorin(H);
```



（a）系统的Nyquist图　　（b）$\boldsymbol{K}_{\mathrm{p}}$补偿后的Nyquist图

图5-37 多变量系统的Nyquist阵列图

从图形可以看出，尽管闭环系统稳定，但由于Gershgorin带太宽，覆盖原点，不能保证为对角占优系统，所以在设计时有很多困难。

考虑前置静态补偿矩阵

$$\boldsymbol{K}_{\mathrm{p}}=\begin{bmatrix}0.3610 & 0.4500\\ -1.1300 & 1.0000\end{bmatrix}$$

则可以用下面语句绘制补偿系统的带有Gershgorin带的Nyquist曲线，如图5-37(b)所示。可见这时得出的Gershgorin带明显变窄，系统为对角占优系统，易于设计与进一步分析。

```
>> Kp=[0.3610,0.4500; -1.1300,1.0000];
   G=ss(G*Kp); H=mv2fr(G.a,G.b,G.c,G.d,w); gershgorin(H);
```

多变量频域设计（MFD）工具箱还提供了多变量系统频域响应数据的运算函数。例如，两个串联的多变量传递函数矩阵$\boldsymbol{G}_1(s)$和$\boldsymbol{G}_2(s)$的频域响应数据可以调用函数$\boldsymbol{H}$=fmulf($\boldsymbol{w}$,$\boldsymbol{H}_2$,$\boldsymbol{H}_1$)求出，如果其中用$\boldsymbol{K}$矩阵乘以传递函数的频域响应数据，则用$\boldsymbol{H}$=fmul($\boldsymbol{w}$,$\boldsymbol{H}_1$,$\boldsymbol{K}$)或$\boldsymbol{H}$=fmul($\boldsymbol{w}$,$\boldsymbol{K}$,$\boldsymbol{H}_1$)直接求出。在多变量系统运算中应该注意模块相乘运算的顺序。

函数$\boldsymbol{H}$=faddf($\boldsymbol{w}$,$\boldsymbol{H}_1$,$\boldsymbol{H}_2$)可以计算出多变量系统$\boldsymbol{G}_1(s)$和$\boldsymbol{G}_2(s)$并联时频域

响应的数据，而函数 `H=faddf(w,K,H1)` 可以求出模块频域响应数据和矩阵 $\boldsymbol{K}$ 相加的频域响应数据。

MFD 工具箱中描述受控对象的函数不能直接处理时间延迟项，所以可以采用该工具箱中 `H=fdly(w,H1,D)` 函数直接求出，其中 $\boldsymbol{D}$ 为延迟矩阵。利用 MFD 工具箱，还可以由 `H=finv(w,H1)` 函数求出逆 Nyquist 响应数据[1]。

从函数调用方式看，这样处理复杂结构多变量系统的频域响应还是比较麻烦的。为此，我们编写了直接求取多变量系统的频域响应的函数 `H=mfrd(G,w)`。该函数利用控制系统工具箱支持的带有内部延迟状态方程模型，事先计算出系统的 LTI 模型 $\boldsymbol{G}$，然后计算其在频率向量点 $\boldsymbol{w}$ 处的频域响应数据 $\boldsymbol{H}$。该函数清单如下：

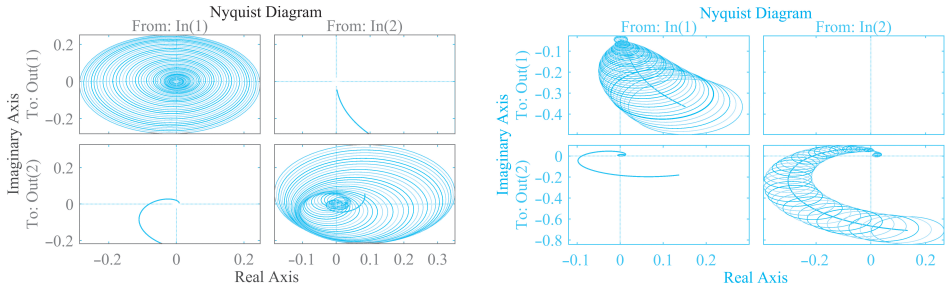
```
function H=mfrd(G,w)
H1=frd(G,w); h=H1.ResponseData; H=[];
for i=1:length(w); H=[H; h(:,:,i)]; end
```

例 5-49　考虑带有时间延迟模型的 Nyquist 曲线的绘制方法，假设系统模型为 [11]，试分析其对角占优性。

$$\boldsymbol{G}(s)=\begin{bmatrix}\dfrac{0.1134}{1.78s^2+4.48s+1}\mathrm{e}^{-0.72s} & \dfrac{0.924}{2.07s+1}\\ \dfrac{0.3378}{0.361s^2+1.09s+1}\mathrm{e}^{-0.3s} & \dfrac{-0.318}{2.93s+1}\mathrm{e}^{-1.29s}\end{bmatrix}$$

解　用下面的语句可以直接绘制出系统的 Nyquist 曲线，如图 5-38(a)所示。显然，由于有 Gershgorin 带覆盖原点，这样的系统不是对角占优的系统。

```
>> G=[tf(0.1134,[1.78 4.48 1]), tf([0.924],[2.07,1]);
      tf(0.3378,[0.361,1.09,1]), tf(-0.318,[2.93 1])];
   G=ss(G); D=[0.72 0; 0.3 1.29]; w=logspace(0,1);
   H=mv2fr(G.a,G.b,G.c,G.d,w); H1=fdly(w,H,D); gershgorin(H1);
```



（a）系统的 Nyquist 图　　（b）$\boldsymbol{G}^{-1}(0)$ 补偿后的 Nyquist 图

图 5-38　系统带有 Gershgorin 带的 Nyquist 图

[1] 函数 `finv()` 与统计工具箱中 F 分布逆概率分布函数重名，如果同时安装了这两个工具箱，应该在路径顺序上加以安排，确保调用正确的函数。

在多变量系统频域设计理论中，一种最直接的对角占优补偿方法 [12] 是引入前置静态增益矩阵 $\boldsymbol{K}_{\mathrm{p}}=\boldsymbol{G}^{-1}(0)$，这样将得出补偿后的 Nyquist 图，如图 5-38(b) 所示。可见，这样设计的系统改善了对角占优的性能。后面的内容将系统介绍多变量系统设计理论。

```
>> H0=mv2fr(G.a,G.b,G.c,G.d,0);  % 求出 Kp=G^{-1}(0)
   Kp=inv(H0); H2=fmul(w,H1,Kp); gershgorin(H2);
```

利用前面介绍的 `mfrd()` 函数，则上述语句可以简化成

```
>> G.ioDelay=D; G1=G*Kp; H2=mfrd(G1,w); gershgorin(H2)
```

5.6.3　多变量系统的奇异值曲线绘制

单变量系统用 Bode 图可以很容易描述其特性，多变量系统不适于用 Bode 图表示，而可以采用奇异值的形式表示。多变量系统的传递函数矩阵在 ω 处存在奇异值 $\sigma_1(\omega)$, $\sigma_2(\omega)$, $\cdots$, $\sigma_m(\omega)$，这样当频率 ω 变化时，传递函数矩阵的奇异值可以作为轨迹绘制出来，称为奇异值曲线。这些奇异值曲线可以看成是多变量系统的 Bode 图。奇异值曲线是多变量系统鲁棒控制中的重要指标，将在第 10 章进一步介绍其基本内容。

鲁棒控制工具箱中 [15] 提供了 `sigma()` 函数可以直接绘制多变量系统的奇异值曲线，该函数的调用格式与 `bode()` 等函数完全一致。还可以使用面向对象的 `sigmaplot()` 函数绘制多变量系统的奇异值曲线。

例 5-50　仍考虑例 5-49 中给出的带有时间延迟的多变量模型，试绘制其奇异值曲线。

解　该延迟多变量系统的奇异值曲线可以由下面的语句直接绘制出来，如图 5-39 所示。

```
>> G=[tf(0.1134,[1.78 4.48 1],'ioDelay',0.72),tf([0.924],[2.07,1]);
      tf(0.3378,[0.361,1.09,1],'ioDelay',0.3), ...
      tf(-0.318,[2.93 1],'ioDelay',1.29)];
   sigma(G)          % 直接绘制系统的奇异值曲线
```

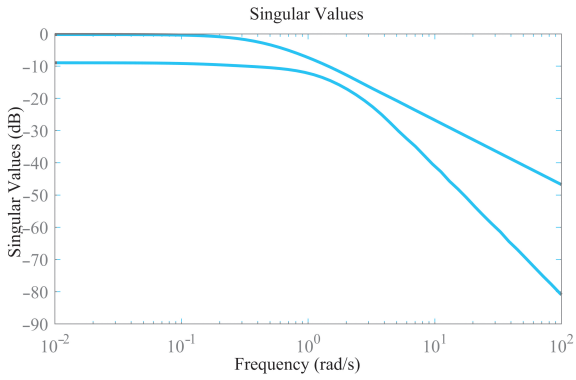


图 5-39　多变量系统的奇异值曲线

5.7 习题

(1) 判定下列连续传递函数模型的稳定性。

① $\dfrac{1}{s^3+2s^2+s+2}$ ② $\dfrac{1}{6s^4+3s^3+2s^2+s+1}$ ③ $\dfrac{1}{s^4+s^3-3s^2-s+2}$

④ $\dfrac{3s+1}{s^2(300s^2+600s+50)+3s+1}$ ⑤ $\dfrac{0.2(s+2)}{s(s+0.5)(s+0.8)(s+3)+0.2(s+2)}$

(2) 判定下面采样系统的稳定性。

① $H(z)=\dfrac{-3z+2}{z^3-0.2z^2-0.25z+0.05}$

② $H(z)=\dfrac{3z^2-0.39z-0.09}{z^4-1.7z^3+1.04z^2+0.268z+0.024}$

③ $H(z)=\dfrac{z^2+3z-0.13}{z^5+1.352z^4+0.4481z^3+0.0153z^2-0.01109z-0.001043}$

④ $H(z^{-1})=\dfrac{2.12z^{-2}+11.76z^{-1}+15.91}{z^{-5}-7.368z^{-4}-20.15z^{-3}+102.4z^{-2}+80.39z^{-1}-340}$

(3) 由下面给出的控制系统传递函数模型写出状态方程实现的可控标准型和可观测标准型。

$$G(s)=\frac{0.2(s+2)}{s(s+0.5)(s+0.8)(s+3)+0.2(s+2)}$$

(4) 给出一个八阶系统模型 $G(s)$

$$G(s)=\frac{18s^7+514s^6+5982s^5+36380s^4+122664s^3+222088s^2+185760s+40320}{s^8+36s^7+546s^6+4536s^5+22449s^4+67284s^3+118124s^2+109584s+40320}$$

并假定系统具有零初始状态，请求出单位阶跃响应和脉冲响应的解析解。若输入信号变为正弦信号 $u(t)=\sin(3t+5)$，请求出零初始状态下系统时域响应的解析解，并用图形的方法进行描述，和数值解进行比较。

(5) 给出连续系统的状态方程模型，请判定系统的稳定性。

① $\dot{\boldsymbol{x}}(t)=\begin{bmatrix}-0.2 & 0.5 & 0 & 0 & 0\\ 0 & -0.5 & 1.6 & 0 & 0\\ 0 & 0 & -14.3 & 85.8 & 0\\ 0 & 0 & 0 & -33.3 & 100\\ 0 & 0 & 0 & 0 & -10\end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix}0\\0\\0\\0\\30\end{bmatrix}u(t)$

② $\boldsymbol{x}[(k+1)T]=\begin{bmatrix}17 & 24.54 & 1 & 8 & 15\\ 23.54 & 5 & 7 & 14 & 16\\ 4 & 6 & 13.75 & 20 & 22.5889\\ 10.8689 & 1.2900 & 19.099 & 21.896 & 3\\ 11 & 18.0898 & 25 & 2.356 & 9\end{bmatrix}\boldsymbol{x}(kT)+\begin{bmatrix}1\\2\\3\\4\\5\end{bmatrix}u(kT)$

(6) 考虑下面给出的多变量系统，试求出该系统的零点和极点，并判定系统的稳定性。

$$\begin{cases}\dot{\boldsymbol{x}}(t)=\begin{bmatrix}-3 & 1 & 2 & 1\\ 0 & -4 & -2 & -1\\ 1 & 2 & -1 & 1\\ -1 & -1 & 1 & -2\end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix}1 & 0\\ 0 & 2\\ 0 & 3\\ 1 & 1\end{bmatrix}\boldsymbol{u}(t)\\ \boldsymbol{y}(t)=\begin{bmatrix}1 & 2 & 2 & -1\\ 2 & 1 & -1 & 2\end{bmatrix}\boldsymbol{x}(t)\end{cases}$$

注意，多变量系统零点的概念和单变量系统不同，不能由单独求每个子传递函数零点的方式求取，应该由tzero()函数得出，另外，pzmap()函数同样适用于多变量系统。

（7）判定下列系统的可控、可观测性，求出它们的可控、可观测及Luenberger标准型实现，并求出系统的2-范数和无穷范数。

① $\boldsymbol{A}=\begin{bmatrix}0&1&1&1\\0&0&0&1\\0&1&0&0\\0&0&1&1\end{bmatrix}$，$\boldsymbol{B}=\begin{bmatrix}1&0\\0&0\\0&1\\1&0\end{bmatrix}$，$\boldsymbol{C}=\begin{bmatrix}1&0&0&0\\0&1&0&0\end{bmatrix}$

② $\boldsymbol{A}=\begin{bmatrix}0&2&0&0\\0&1&-2&0\\0&0&3&1\\1&0&0&0\end{bmatrix}$，$\boldsymbol{B}=\begin{bmatrix}2&0\\1&2\\0&1\\0&0\end{bmatrix}$，$\boldsymbol{C}=\begin{bmatrix}0&1&0&0\\0&0&1&0\end{bmatrix}$

（8）求出下面状态方程模型的最小实现。

$$\dot{\boldsymbol{x}}(t)=\begin{bmatrix}0&-3&0&0\\1&-4&0&0\\0&0&0&0\\0&0&1&-2\end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix}3&2\\1&2\\1&1\\1&1\end{bmatrix}\boldsymbol{u}(t),\quad \boldsymbol{y}(t)=\begin{bmatrix}0&1&0&0\\0&0&0&1\end{bmatrix}\boldsymbol{x}(t)$$

（9）请求出下面自治系统状态方程的解析解。

$$\dot{\boldsymbol{x}}(t)=\begin{bmatrix}-5&2&0&0\\0&-4&0&0\\-3&2&-4&-1\\-3&2&0&-4\end{bmatrix}\boldsymbol{x}(t),\quad \boldsymbol{x}(0)=\begin{bmatrix}1\\2\\0\\1\end{bmatrix}$$

并和数值解得出的曲线比较。

（10）假设PI和PID控制器的结构分别为

$$G_{\mathrm{PI}}(s)=K_{\mathrm{p}}+\frac{K_{\mathrm{i}}}{s},\quad G_{\mathrm{PID}}(s)=K_{\mathrm{p}}+\frac{K_{\mathrm{i}}}{s}+K_{\mathrm{d}}s$$

请说明为什么PI或PID控制器可以消除稳定闭环系统的阶跃响应稳态误差，不稳定系统能用PI或PID控制器消除稳态误差吗，为什么？

（11）请绘制下面状态方程模型的单位阶跃响应曲线。

$$\dot{\boldsymbol{x}}(t)=\begin{bmatrix}-0.2&0.5&0&0&0\\0&-0.5&1.6&0&0\\0&0&-14.3&85.8&0\\0&0&0&-33.3&100\\0&0&0&0&-10\end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix}0\\0\\0\\0\\30\end{bmatrix}u(t)$$

且输出方程为$y(t)=[1,0,0,0,0]\boldsymbol{x}(t)$。绘制出所有状态变量的曲线。选择不同的采样周期$T$，对该系统进行离散化，绘制出离散系统的阶跃响应曲线，和连续系统进行比较，并说明超调量、调节时间等指标的变化规律。

（12）假设连续系统传递函数模型如下给出，试选择不同的采样周期$T=0.01,0.1,1\mathrm{s}$等对其进行离散化，试对比连续系统及离散化系统的时域响应曲线，你能从中得出什么结论？

$$G(s)=\frac{-2s^2+3s-4}{s^3+3.2s^2+1.61s+3.03}$$

(13) 试绘制下列开环系统的根轨迹曲线,并确定使单位负反馈系统稳定的 K 值范围。

① $G(s)=\dfrac{(s+6)(s-6)}{s(s+3)(s+4-4\mathrm{j})(s+4-4\mathrm{j})}$ ② $G(s)=\dfrac{s^2+2s+2}{s^4+s^3+14s^2+8s}$

③ $G(s)=\dfrac{1}{s(s^2/2600+s/26+1)}$ ④ $G(s)=\dfrac{800(s+1)}{s^2(s+10)(s^2+10s+50)}$

⑤ $H(z)=\dfrac{5(z-0.2)^2}{z(z-0.4)(z-1)(z-0.9)+0.6}, T=0.1\,\mathrm{s}$

⑥ $H(z^{-1})=\dfrac{(z^{-1}+3.2)(z^{-1}+2.6)}{z^{-5}(z^{-1}-8.2)},\ T=0.05\,\mathrm{s}$

(14) 绘制下面状态方程系统的根轨迹,确定使单位负反馈系统稳定的 K 值范围。

$$\dot{\boldsymbol{x}}(t)=\begin{bmatrix}-1.5 & -13.5 & -13 & 0\\ 10 & 0 & 0 & \\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix}1\\0\\0\\0\end{bmatrix}u(t),\quad y(t)=[0,0,0,1]\boldsymbol{x}(t)$$

(15) 假设连续延迟系统的传递函数如下给出,试求出能使得单位负反馈系统稳定的 K 值范围。

$$G(s)=\frac{K(s-1)\mathrm{e}^{-2s}}{(s+1)^5}$$

(16) 假设系统的开环模型如下给出,并假设系统由单位负反馈结构构成,试用根轨迹找出能使得闭环系统主导极点有大约 $\zeta=0.707$ 阻尼比的 K 值。

$$G(s)=\frac{K}{s(s+10)(s+20)(s+40)}$$

(17) 已知离散系统的受控对象模型如下给出,试绘制其根轨迹,并得出使得单位负反馈闭环系统稳定的 K 值范围。选择一个能使闭环系统稳定的 K,绘制闭环系统的阶跃响应曲线,并求出阶跃响应的超调量、调节时间等指标。

$$H(z)=K\frac{1}{(z+0.8)(z-0.8)(z-0.99)(z-0.368)}$$

(18) 若上述系统带有时间延迟,即 $\widetilde{H}(z)=H(z)z^{-8}$,试重复上题的分析过程。改变系统的延迟时间常数再进行分析,得出相应的结论。

(19) 考虑开环传递函数模型如下给出,试绘制出该系统关于 a 的根轨迹,求出使得单位负反馈闭环系统稳定的 a 的范围。

$$G(s)=\frac{0.3(s+2)(s^2+2.1s+2.23)}{s^2(s^2+3s+4.32)(s+a)}$$

(20) 对下列各个开环模型进行频域分析,绘制出 Bode 图、Nyquist 图及 Nichols 图,并求出系统的幅值裕度和相位裕度,在各个图形上标注出来。假设闭环系统由单位

负反馈构造而成，试由频域分析判定闭环系统的稳定性，并用阶跃响应来验证。

① $G(s)=\dfrac{8(s+1)}{s^2(s+15)(s^2+6s+10)}$　② $G(s)=\dfrac{4(s/3+1)}{s(0.02s+1)(0.05s+1)(0.1s+1)}$

③ $\begin{cases}\dot{\boldsymbol{x}}(t)=\begin{bmatrix}0&2&1\\-3&-2&0\\1&3&4\end{bmatrix}\boldsymbol{x}(t)+\begin{bmatrix}4\\3\\2\end{bmatrix}u(t)\\ y(t)=[1,2,3]\boldsymbol{x}(t)\end{cases}$

④ $H(z)=0.45\dfrac{(z+1.31)(z+0.054)(z-0.957)}{z(z-1)(z-0.368)(z-0.99)}$，

⑤ $G(s)=\dfrac{6(-s+4)}{s^2(0.5s+1)(0.1s+1)}$　⑥ $G(s)=\dfrac{10s^3-60s^2+110s+60}{s^4+17s^3+82s^2+130s+100}$

(21) 假设典型单位负反馈控制系统的各个模型如下

$$G(s)=\frac{2}{s\left[(s^4+5.5s^3+21.5s^2+s+2)+20(s+1)\right]},\quad G_c(s)=K\frac{1+0.1s}{1+s}$$

并假定 $K=1$，请绘制出系统的 Bode 图、Nyquist 图与 Nichols 图，请判定这样设计出来的反馈系统是否为较好设计的系统，画出闭环系统的阶跃响应曲线做出说明，并指出如何修正 K 的值来改进系统的响应。

(22) 试对下面的时间延迟系统进行频域分析，绘制出系统的各种频域响应曲线及各种裕度，判定单位负反馈下闭环系统的稳定性，用时域响应验证得出的结论。

① $G(s)=\dfrac{(-2s+1)\mathrm{e}^{-3s}}{s^2(s^2+3s+3)(s+5)(s^2+2s+6)}$

② $H(z)=\dfrac{z^2+0.568}{(z-1)(z^2-0.2z+0.99)}z^{-5},\ T=0.05\,\mathrm{s}$

(23) 假设系统的对象模型为 $G(s)=1/s^2$，某最优控制器模型为

$$G_c(s)=\frac{5620.82s^3+199320.76s^2+76856.97s+7253.94}{s^4+77.40s^3+2887.90s^2+28463.88s+2817.59}$$

并假设系统由单位负反馈结构构成，请绘制出叠印有等 M 线和等 N 线的 Nyquist 图、Nichols 图，并由之分析闭环系统的动态性能，绘制闭环系统阶跃响应曲线来证实你的推断。

(24) 假设受控对象模型与由某种方法设计出串联控制器模型如下给出，试用频域响应的方法判定闭环系统的性能，并用时域响应检验得出的结论。

$$G(s)=\frac{100(1+s/2.5)}{s(1+s/0.5)(1+s/50)},\quad G_c(s)=\frac{1000(s+1)(s+2.5)}{(s+0.5)(s+50)}$$

(25) 假设带有时间延迟的系统传递函数矩阵为

$$\boldsymbol{G}(s)=\begin{bmatrix}\dfrac{0.06371}{s^2+2.517s+0.5618}\mathrm{e}^{-0.72s} & \dfrac{0.4464}{s+0.4831}\\ \dfrac{0.9357}{s^2+3.019s+2.77}\mathrm{e}^{-0.3s} & \dfrac{-0.1085}{s+0.3413}\mathrm{e}^{-1.29s}\end{bmatrix}$$

试绘制其带有 Gershgorin 带的 Nyquist 阵列，分析其是否为对角占优的系统，绘制系统的开环阶跃响应，该响应是否符合你的结论？

(26) 考虑下面给出的双输入双输出系统。

$$\boldsymbol{G}(s)=\begin{bmatrix}\dfrac{0.806s+0.264}{s^2+1.15s+0.202} & \dfrac{-(15s+1.42)}{s^3+12.8s^2+13.6s+2.36}\\ \dfrac{1.95s^2+2.12s+4.90}{s^3+9.15s^2+9.39s+1.62} & \dfrac{7.14s^2+25.8s+9.35}{s^4+20.8s^3+116.4s^2+111.6s+188}\end{bmatrix}$$

绘制出带有 Gershgorin 带的 Nyquist 曲线，并在该曲线上标出各个频率下的特征值，验证这些特征值满足 Gershgorin 定理，并绘制该系统的阶跃响应曲线来演示结果系统是不是较好解耦的系统。

(27) Bode 增益曲线描述的是系统模型 $G(s)$ 的幅值与频率之间的关系，即 $|G(\mathrm{j}\omega)|$ 与 $s=\mathrm{j}\omega$ 之间的关系。MATLAB 语言提供了强大的绘图功能，试用三维表面图的方式绘制出下面函数的增益曲面，其中 $s=x+\mathrm{j}y$。

① $G(s)=\dfrac{3s+1}{s^2(300s^2+600s+50)+3s+1}$

② $G(s)=\dfrac{(-2s+1)\mathrm{e}^{-3s}}{s^2(s^2+3s+3)(s+5)(s^2+2s+6)}$

参考文献

[1] 王万良. 自动控制原理 [M]. 北京：科学出版社，2001.

[2] Kailath T. Linear systems[M]. Englewood Cliffs：Prentice-Hall，1980.

[3] Kalman R E. On the general theory of control systems[C]// Proceedings of 1st IFAC Congress. Moscow，1960：521–547.

[4] 郑大钟. 线性系统理论 [M]. 北京：清华大学出版社，1980.

[5] 薛定宇. 控制系统仿真与计算机辅助设计 [M]. 北京：机械工业出版社，2005.

[6] 薛定宇，任兴权. 连续系统的仿真与解析解法 [J]. 自动化学报，1992，19(6)：694–702.

[7] 薛定宇. 控制系统计算机辅助设计—— MATLAB 语言与应用 [M]. 2 版. 北京：清华大学出版社，2006。

[8] Nyquist H. Regeneration theory[J]. The Bell System Technical Journal，1932，11(1)：126–147.

[9] Bode H. Network analysis and feedback amplifier design[M]. New York：D Van Nostrand，1945.

[10] James H M，Nichols N B，Phillips R S. Theory of servomechanisms[M]. New York：McGraw-Hill，1947.

[11] Munro N. Multivariable control 1: the inverse Nyquist array design method[C]// Lecture notes of SERC vacation school on control system design. UMIST，Manchester，1989.

[12] Rosenbrock H H. Computer-aided control system design[M]. New York：Academic Press，1974.

[13] MacFarlane A G J，Postlethwaite I. The generalized Nyquist stability criterion and multivariable root loci[J]. International Journal of Control，1977，25(1)：81–127.

[14] Boyel J M，Ford M P，Maciejowski J M. A multivariable toolbox for use with MATLAB[J]. IEEE Control Systems Magazine，1989，9(1)：59–65.

[15] MathWorks. Robust control toolbox user's manual[Z]. Natick: MathWorks，2005.