承蒙 Thomas F.Weiss.允许使用

麻省理工大学

电气工程与计算机科学系

信号与系统——6.003

MATLAB 导论——1999 年秋季课程

Thomas F.Weiss

1999 年 9 月 9 日最后更改

目录

1 绪论

MATLAB是一个对信号处理和系统分析非常有效的编程语言和数据可视化软件包。本文档是对MATLAB的一个简介，主要介绍课程 6.003 [1]中特别重要的特点。假定读者熟知雅典娜工程（Project Athena），并有雅典娜账号，但对MATLAB有很少或没有使用经验。其他的帮助可以通过雅典娜咨询获得，它提供了很多指导性材料和短期课程（电话分机号3-4435），在线咨询（在雅典娜命令行输入olc）和雅典娜在线帮助（在雅典娜命令行输入help）。论述MATLAB的书籍有很多，比如，Engineering Problem Solving with Matlab，D. M. Etter著，Prentice-Hall出版社出版（1997）和Mastering MATLAB，Hanselman和Littlefield著，Prentice-Hall出版社出版（1996）。K. Sigmon写的平装本MATLAB Primer，CRC出版社出版（1994），是MATLAB指令的一个便捷的总结。关于MATLAB的更多信息可以参考商家（MathWorks公司）主页，网址是http://www.mathworks.com。完整版文档可以联系MathWorks公司购买。

2 准备开始

雅典娜工程中，可以从 Dashboard（你登录雅典娜工程后屏幕上方的菜单）中进入MATLAB，用层级菜单，引导路径如下：

Numerical/Math//Analysis and Plotting//MATLAB

然后 MATLAB 会打开一个命令窗口，上面有 MATLAB 命令提示符“>>”。

MATLAB 有很多与 UNIX 命令相似的有用的命令，比如，“ls”，“pwd”和“cd”。用它们可以方便地列出 MATLAB 操作目录，查看操作目录的路径，更改操作目录。用命令“path”控制可以查看某些目录下的 MATLAB 文件。“path”命令列出了 MATLAB 搜速路径中的目录。用命令 path(path, p)或 path(p, path)可以将某个目录设置到 MATLAB 的搜索路径中，其中 p 某个新的目录，比如可以包括使用者写的函数。

用下面的引导路径可以从雅典娜工程的菜单中进入一个特别设计的软件：

Coursewar//Electrical Engineering and Computer Science//
6.003 Signals and Systems//MATLAB.

这些命令显示了浏览 6.003 中几个重要主题的一个图形用户界面。本软件也用于课程演示。

3 在 MATLAB 中获得帮助

如果你想知道一个函数的用法，并且知道它的函数名，可以用“help”命令：

```
>> help functionname
```

这个命令可以显示该函数的一个描述，通常还包括一系列相关函数。如果你不记得函数名，可以用“lookfor”命令加上这个函数的关键词：

```
>> lookfor keyword
```

这个命令可以显示描述中包含该关键词的一系列函数。

其他你可能觉得有用的帮助命令有“info”，“what”和“which”。这些命令的描述可以用帮助命令查看。MATLAB 还有大量的演示程序，可以用“demo”命令查看。

[1] 本文档的修订版会放置在 6.003 的课程主页上

4 MATLAB 变量——标量，向量，矩阵

MATLAB 以 $M \times N$ 的形式保存了大量的矩阵，其中 M 是行数，N 是列数。一个 1×1 矩阵是一个标量；1×N 矩阵是一个行向量，M×1 矩阵是一个列向量。矩阵的全部元素可以是实数，也可以是复数；如果用户没有重新定义，那么 $\sqrt{-1}$ 可以写作“i”或“j”。方括号“[]”表示一个矩阵，空格把相邻列元素分开，分号把相邻行分开。例如，考虑如下对变量 x 的赋值

实数量 `>> x = 5`

复数量 `>> x = 5 + 10j`（或者`>> x = 5 + 10i`）

行向量 `>> x = [1 2 3]`（或者 `x = [1, 2, 3]`）

列向量 `>> x = [1; 2; 3]`

3×3 矩阵`>> x = [1 2 3; 4 5 6; 7 8 9]`

有几点需要说明的地方。矩阵的复数元素不能打空格，比如“-1+2j”可以作为一个矩阵元素，而“-1 + 2j”就不行。还有，“-1+2j”可以正确地解释，然而“-1+j2”就不行。（MATLAB 把“j2”解释为一个变量名。你可以写成“-1+j*2”。）

4.1 复数运算

下面说明了一些重要的复数运算：

复数量 `>> x = 3+4j`

x 的实部 `>> real(x)` $\Rightarrow 3$

x 的虚部 `>> imag(x)` $\Rightarrow 4$

x 的幅值 `>> abs(x)` $\Rightarrow 5$

x 的相角 `>> angle(x)` $\Rightarrow 0.9273$

x 的共扼 `>> conj(x)` $\Rightarrow 3-4i$

4.2 生成向量

向量可以用“：”命令生成。比如，以增量 0.5 生成一个从 0 到 10 的向量 x，下列命令生成了一个 1×21 矩阵

```
>> x = [0:0.5:10];
```

其他生成向量的命令有“linspace”和“logspace”，前者通过指定第一个值、最后一个值和它们之间的值的个数来生成向量，后者的方法一样，但第一个值和最后一个值之间的项数以对数空间排列。

4.3 访问向量元素

向量元素可以通过指定行和列来访问。比如在矩阵 $A=[1 \quad 2 \quad 3;4 \quad 5 \quad 6;7 \quad 8 \quad 9]$ 中，第一行第三列的元素可以这样访问

[1] 本文档的修订版会放置在 6.003 的课程主页上

>> x = A(1, 3)，结果是 3

整个第二行这样访问

>> y = A(2, :)，结果是[4 5 6]

其中“:”的意思是“取该栏的所有项”。由第 1 行、第 2 行和全部 3 列组成的 A 的一个子矩阵这样来生成

>> z = A(1:2, 1:3)，结果是[1 2 3; 4 5 6]

5 矩阵运算

MATLAB 对矩阵的运算包括算术运算，关系运算和逻辑运算。

5.1 算术矩阵运算

矩阵的基本算术运算（当然标量是矩阵的特殊情况）有：

+ 加法
- 减法
* 乘法
/ 右除
\ 左除
^ 取幂（幂）
, 转置

如果矩阵大小对于运算不合适，就会出现错误信息。除法定义如下：如果 A 可逆且矩阵大小合适，则 $A*x=b$ 的解是 $x=A\backslash b$， $x*A=b$ 的解是 $x=b/A$。

加法和减法涉及的是元素对元素的运算；而乘除法不是。但是 MATLAB 提供了元素对元素的运算，要在运算符前面加“.”，如下：

.* 乘法
./ 右除
.\ 左除
.^ 取幂（幂）
.’ 转置（不共扼）

矩阵乘法和元素对元素乘法的不同可以通过下面的例子看到：

```
>> A = [1 2; 3 4]
A =
    1   2
    3   4
>> B = A*A
B =
    7   10
    15  22
>> C = A.*A
C =
    1   4
    9   16
```

5.2 关系运算

关系运算定义如下：

<	小于
<=	小于或等于
>	大于
>=	大于或等于
==	等于
~=	不等于

这些都是元素对元素的运算，它返回一个 1 和 0 矩阵（1=真，0=假）。注意“=”和“==”的区别。

5.3 流程控制运算

MATLAB 包含了一组常用的流程控制结构，例如，for，while 和 if，还有逻辑运算符，比如&（与），|（或）和~（非）。

5.4 数学函数

MATLAB 给出了大量的内置函数，它们对矩阵进行元素对元素的运算，包括：

sin	正弦
cos	余弦
tan	正切
asin	反正弦
acos	反余弦
atan	反正切
exp	指数运算
log	自然对数运算
log10	普通对数
sqrt	平方根
abs	绝对值
sign	符号运算

6 MATLAB 文件

几种 MATLAB 文件包括 MATLAB 命令脚本文件，像内置 MATLAB 函数一样调用的用户编写的 MATLAB 函数文件，包含数字解和图像的文件。

6.1 M 文件

MATLAB 是解释型语言，就是说 MATLAB 命令行中敲入的命令在当前 MATLAB 进程

中被解释运行。但是，每次执行一个任务时敲入长长的命令序列是很烦人的。有两种方法可以使 MATLAB 的力量得到扩展——脚本和函数。这两种方法都用像 emacs 一样的文本编辑器中编写的 m 文件（因为扩展名是.m 所以这样命名，m 文件还称点 m 文件）。m 文件的好处在于它可以保存命令，还可以轻易地修改命令而无需重新敲入整个命令行。

6.1.1 脚本

MATLAB 脚本是在编辑器中敲入的一个命令序列，并保存为 m 文件。用 emacs 编写一个 m 文件，你可以在 Athena 命令行中敲入

```
athena% emacs filename.m &
```

或者在 MATLAB 中输入

```
>> ! emacs filename.m &
```

注意“!”允许直接执行 UNIX 命令。在 emacs 编辑器中按执行顺序敲入 MATLAB 命令。在 MATLAB 命令行中敲入文件名就可以执行这些命令，比如，m 文件 filename.m 的执行可以敲入：

```
>> filename
```

执行 m 文件等于在 MATLAB 命令行中输入整个命令序列。m 文件用到的所有变量都被存放在 MATLAB 工作空间中。工作空间（在 MATLAB 初始化时是空的）包含了 MATLAB 进程中定义的所有变量。

6.1.2 函数

第二中 m 文件是函数文件，它和脚本文件一样在编辑器中生成，但有如下的形式：

```
function [output 1, output 2] = functionname(input1, input2)
%
%[output 1, output 2] = functionname(input1, input2) Functionname
%
%Some comments that explain what the function does go here.
%
MATLAB command 1;
MATLAB command 2;
MATLAB command 3;
```

该函数的 m 文件名是 functionname.m，它在 MATLAB 命令行中或被另外一个 m 文件调用，如下：

```
>> [output1, output2] = functionname(input1, input2)
```

注意 MATLAB 忽略了“%”后面的所有文字，可以用这个符号写注释。以“;”结束一行可以停止输出打印，在一行的最后输入“...”可以续行，以便在下一行继续输入指令。

6.2 Mat 文件

Mat 文件（因为有扩展名.mat 所以这样命名，因此也称为点 mat 文件）是存储数值结果的压缩二进制文件。这种文件可以存储由一系列 MATLAB 指令生成的结果。例如，在名为 filename.mat 的文件中存储两个变量 variable1 和 variable2，可以输入

```
>> save filename.mat variable1 variable2
```

保存所以当前变量，可以输入

```
>> save filename.mat
```

稍后可以载入 mat 文件，输入

```
>> load filename(或者 load filename.mat)
```

6.3 脚本文件

MATLAB 生成的图形可以保存到脚本文件，这样就可以日后打印了（比如用标准 UNIX 命令“lpr”）。例如保存当前绘图，输入

>> print –dps filename.ps

绘图也可以在 MATLAB 中直接绘制，输入

>> print –Pprintername

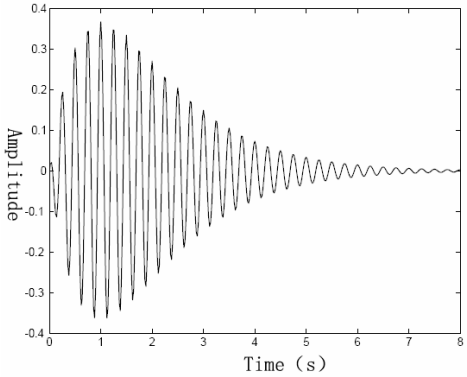

图1：函数 $x(t) = te^{-t}\cos(2\pi 4t)$ 的绘图举例

输入“help print”查看附加选项。

6.4 Diary 文件

一个 MATLAB 进程的书面记录可以用 diary 命令保存在一个 diary 文件中。在 MATLAB 进程中开始记录一个 diary 文件，并把它保存在文件 filename 中，输入

>> diary filename

结束信息记录，关闭文件，输入

>> diary off

7 绘图

MATLAB 包含了大量的二维和三维绘图命令。其中最基本的命令是“plot”，它有多个可选参数。用这个命令画一个时间函数作为一个简单的例子。

```
t = linspace(0, 8, 401);          %Define a vector of times from 0 to 8 s with 401 points
x = t.*exp(-t).*cos(2*pi*4*t);    %Define a vector of x values
plot(t, x);                       %Plot x vs t
xlabel('Time(s)');                %Label time axis
ylabel('Amplitude');              %Label amplitude axis
```

该脚本生成的图如图 1 所示。

7.1 简单绘图命令

简单的二维绘图命令包括

plot	连续函数在线性坐标上绘图
stem	离散采样在线性坐标上绘图
loglog	对数 x 和 y 轴
semilogx	线性 y 轴和对数 x 轴
semilogy	线性 x 轴和对数 y 轴
bar	条线图
errorbar	误差条线图
hist	柱状图
polar	极坐标图

7.2 自定义绘图

自定义绘图有很多命令，用注释，标题，坐标轴名称等等。一些最常用的命令有

xlabel	为 x 轴命名
ylabel	为 y 轴命名
title	为绘图命名
grid	为绘图加网格
gtext	允许用鼠标定位文本
text	允许在图的指定坐标放置文本
axis	运行改变 x 和 y 轴
figure	生成一个图形对象
figure（n）	使当前图像的图像句柄为 n
hold on	允许在同一坐标轴上绘制多个图
hold off	释放保持当前绘图
close（n）	关闭第 n 号图像
subplot（a，b，c）	生成 a×b 矩阵的绘图，当前图形为第 c 个
orient	指定图形方位

8 信号与系统命令

下面的命令以信号与系统中主题的顺序组织。每个命令都有很多选项可以其应用得更加广泛。

8.1 多项式

系统论中经常会出现多项式。MATLAB 用行向量来表示多项式系数。比如多项式 s^2+4s-5 在 MATLAB 中表示为>> p = [1 4 -5]。下面是控制多项式的一些更重要的命令列表。

roots(p)	用列向量表示多项式的根
polyval(p, x)	估值向量 x 处的多项式 p
conv(p1, p2)	计算多项式 p1 和 p2 的乘积

deconv(p1, p2)	计算 p1 除以 p2 的商
poly2str(p, ‘s’)	用 s 将多项式显示为等式
poly(r)	给定根为 r 的一个列向量，计算多项式的值

8.2 拉普拉斯变换与 Z 变换

拉普拉斯变换是分析连续时间动态系统的一个很重要的工具，而 Z 变换是分析离散时间动态系统的一个很重要的工具。下面的列表包括了实施变换的重要命令。

residue(n, d)	计算多项式之比 n(s)/d(s)的部分分式展开
lsim（SYS, u）	计算/绘制系统 SYS 对输入向量 u 的响应
step（SYS）	计算/绘制系统 SYS 的阶跃响应
impulse（SYS）	计算/绘制系统 SYS 的冲击响应
pzmap（n, d）	计算/绘制系统 SYS 的零极点图
residuez(n, d)	计算多项式之比n(z)/d(z)的部分分式展开，写成z^{-1}的函数
dlsim(n, d, u)	计算系统函数为 n(z)/d(z)的系统对输入向量 u 的时间响应
dstep(n, d)	计算系统函数为 n(z)/d(z)的系统的阶跃响应
dimpulse(n, d)	计算系统函数为 n(z)/d(z)的系统的冲击响应
zplane(z, p)	由极点零点向量 p 和 z 绘制零极点图

这些命令中很多都是对 LTI 系统的一些说明有效的。其中一个说明是关于传递函数的，“SYS”由“TF(num, den)”代替，“num”和“den”分别是系统函数分子分母的系数向量。

8.3 频率响应

对于以多项式之比的方式给定的连续或离散时间系统的系统函数，计算和绘制频率响应由几种有用的命令。

bode(n, d)	绘制一个 CT 系统的波德图，系统函数是多项式比 n(s)/d(s)
freqs(n, d)	计算系统函数为 n(s)/d(s)的一个 CT 系统的频率响应
freqz(n, d)	计算系统函数为 n(z)/d(z)的一个 DT 系统的频率响应

8.4 傅立叶变换与滤波

关于滤波有一系列丰富的命令。这里列出了一些基本命令。

fft(x)	计算向量 x 的离散傅立叶变换
ifft(x)	计算向量 x 的反离散傅立叶变换
fftshift	将 fft 输出从离散频率范围（0，2π）转化为（-π，+π）弧度
filter(n, d, x)	用过滤器过滤向量 x，过滤器系统函数为 n(z)/d(z)，包含一些输出延迟
filtfilt(n, d, x)	除了不含输出延迟以外与 filter 相同

除此之外还有很多过滤器函数，包括 firls，firl1，firl2，invfreqs，invfreqz，remez 和 butter。还有很多窗口函数，包括 boxcar，hanning，hamming，bartlett，blackman，kaiser 和 chebwin。

9 应用举例

9.1 由系统函数求零极点图，波德图和阶跃响应

给定系统函数

$$H(s)=\frac{s}{s^2+2s+101},$$

MATLAB 可以求出零极点图，波德图和阶跃响应。

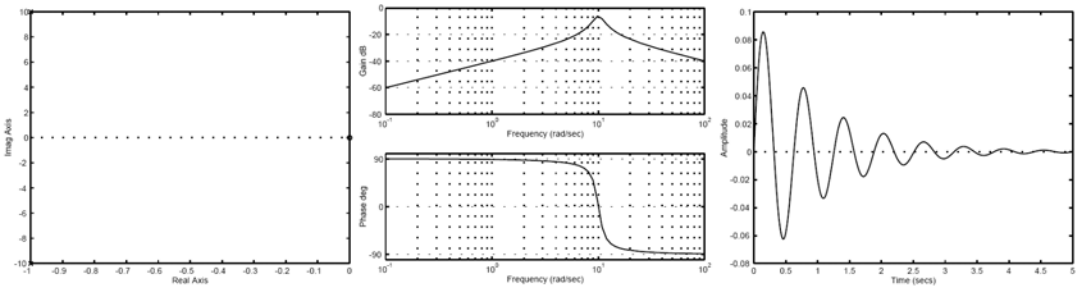

图 2：上图由脚本生成，每幅图都保存为一个封装的脚本文件。
这三幅图由比例 0.3 确定，包含在本文档中。

9.1.1 简单解

获得所需的图形的最简单的方法是用下面的脚本让 MATLAB 选择所以的尺寸、名称和注释。

```
num = [1 0];            %Define numerator polynomial
den = [1 2 101];        %Define denominator polynomial

%%Pole-zero diagram
figure(1)               %Create figure1
pzmap(num,den);         %Plot pole-zero diagram in figure1

%%Bode diagram
figure(2)               %Create figure2
bode(num,den);          %Plot the Bode diagram in figure2

%%Step response
figure(3);              %Create figure3
step(num,den);          %Plot the step response in figure3
```

这个脚本生成了图 2 所示的各个图形。

9.1.2 自定义解

有时可能需要显示一个对图形外观有更多控制的结果。下面的脚本绘制的图形与上面相同，但表明了如何加入名称，图题，定义坐标轴尺度等等。

```
%%Define variables
t = linspace(0, 5, 201);        %Define a time vector with 201 equally-spaced…
                                points from 0 to 5 s.
```

```
w = logspace(-1, 3, 201);                %Define a radian frequency vector with 201…
                                         logarithmically-spaced points from 10^-1 to…
                                         10^3 rad/s
num = [1 0];                             %Define numerator polynomial
den = [1 2 101];                         %Define denominator polynomial
[poles, zeros] = pzmap(num, den);        %Define poles to be a vector of the poles and zeros…
                                         to be a vector of zeros of the system function
[mag, angle] = bode(num, den, w);        %Define mag and angle to be the magnitude and…
                                         angle of the frequency response at w
[y, x] = step(num, den, t);              %Define y to be the step response of the system…
                                         function at t


%%Pole-zero diagram
figure(1)                                %Create figure1
subplot(2, 2, 1)                         %Define figure1 to be a 2 X 2 matrix of plots and…
                                         the next plot is at position (1, 1)
plot(real(poles), imag(poles), 'x', real(zeros), imag(zeros), 'o');  %Plot pole-zero diagram…
                                         with x for poles and o for zeros
title('Pole-Zero Diagram');              %Add title to plot
xlabel('Real');                          %Label x axis
ylabel('Imaginary');                     %Label y axis
axis([-1.1 0.1 -12 12]);                 %Define axis for x and y
grid;                                    %Add a grid

%%Bode diagram magnitude
subplot(2, 2, 2);                        %Next plot goes in position (1,2)
semilogx(w, 20*log10(mag));              %Plot magnitude logarithmically in w and in…
                                         decibels in magnitude
title('Magnitude of Bode Diagram');
ylabel('Magnitude (dB)');
xlabel('Radian Frequency (rad/s)');
axis([0.1 1000 -60 0]);
grid;
subplot(2, 2, 4);                        %Next plot goes in position (2, 2)
semilogx(w, angle);                      %Plot angle logarithmically in w and linearly in…
                                         angle
title('Angle of Bode Diagram');
ylabel('Angle(deg)');
xlabel('Radian Frequency (rad/s)');
axis([0.1 1000 -90 90]);
gird;
```

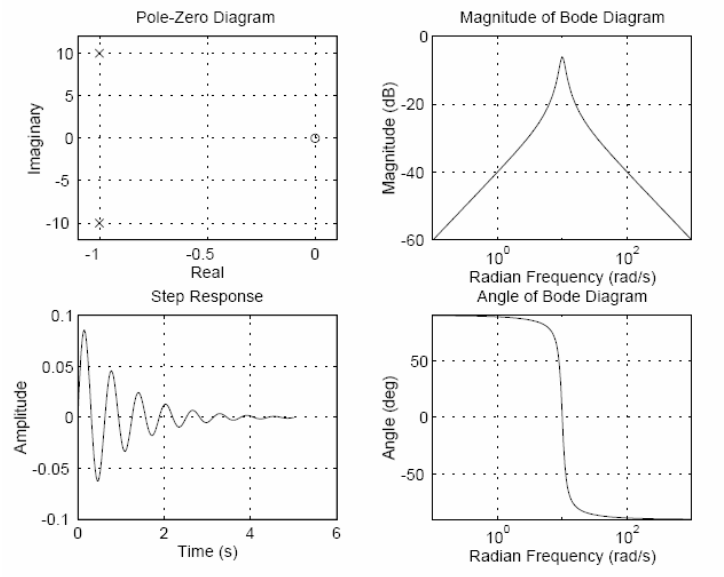

图 3：本组合图像由以上脚本生成并保存为一个封装脚本文件，

按 0.6 确定比例并包含在本文档中

```
%%Step Response                    %Next plot goes in position (2, 1)
subplot(2, 2, 3);
plot(t, y);                        %Plot step response linearly in t and y
title('Step Response');
xlabel('Time(s)');
ylabel('Amplitude');
grid;
```

该脚本生成了如图 3 所示的图像。

9.2 多项式的根轨迹

分析一个系统时，人们常常对确定某个参数变换时多项式根的轨迹很感兴趣。当开环增益变换时绘制闭环反馈系统的极点轨迹是一个常见的例子。比如，在图 4 所示系统中，闭环增益由 Black 的公式给出，如下

$$H(s)=\frac{Y(s)}{X(s)}=\frac{G(s)}{1+KG(s)}。$$

如果 G（s）是 s 的一个有理函数，则可以表示为

$$G(s)=\frac{N(s)}{D(s)}$$

其中 N（s）和 D（s）是多项式。解 H（s）得到

$$H(s)=\frac{N(s)}{D(s)+KN(s)}。$$

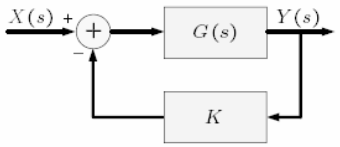

图4：开环增益为G（s）的反馈系统

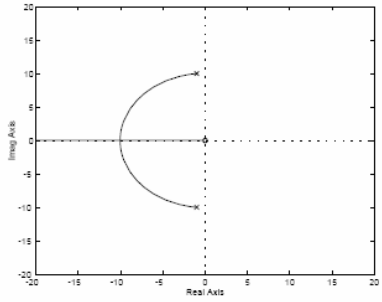

图5：对于G（s)=s/(s^2+2s+101)，
当K变换时H（s）的极点的根轨迹

当 K 变换时求 H（s）极点就意味着求出当 K 变换时多项式 $D(s)+KN(s)$ 的根。因为这是一个常用的计算过程，MATLAB 为计算根轨迹提供了方便的函数，称为“rlocus”。一下的脚本对于 G（s）绘制了根轨迹

$$G(s)=\frac{s}{s^2+2s+101}。$$

```
num = [1 0];                %Define numerator polynomial
den = [1 2 101];            %Define denominator polynomial
figure(1);
rlocus(num,den)             %Plot root locus
```

如图 5 所示。与上例相似，根轨迹也可以自定义，见“help rlocus”。

命令 rlocus 也可用于其他环境下。比如，假设我们有一个 RLC 电路，其导纳为

$$Y(s)=\frac{1}{L}\frac{s}{s^2+Rs+2},$$

我们期望得到当电阻 R 变换时的极点（分母多项式的根）轨迹。只能将分母多项式解析成两部分 $N(s)=s$ 和 $D(s)=s^2+2$，然后用“rlocus”绘制图像。

9.3 LTI 系统对输入的响应

MATLAB 命令 lsim 使给定系统函数 H（s），计算一个 LTI 系统对输入 x（t）的响应更加方便。设

$$H(s)=\frac{5s}{s^2+2s+101},$$

我们要求出对输入 $x(t)=\cos(2\pi t)u(t)$ 的响应。下面的脚本计算了响应，并绘制出图像。

```
figure(1);
num = [5 0];                %Define numerator polynomial
```

```
den = [1 2 101];                %Define denominator polynomial
t = linspace(0, 10, 401);       %Define a time vector
u = cos(2*pi*t);                %Compute the cosine input function
[y, x] = lsim(num, den, u, t);  %Compute the cosine input function
plot(t, y, 'r', t, u, 'b');     %Plot the output in red and the input in blue
xlabel('Time(s)');
ylabel('Amplitude');
```

图像如图 6 所示。

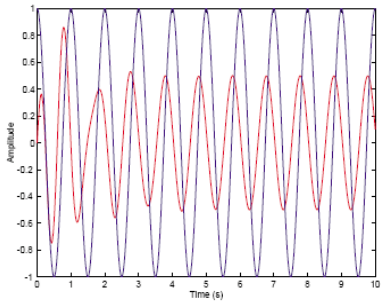

图6：输入函数为余弦函数，起始时间t=0。系统函数为H（s）=5s/(s^2+2s+101)的LTI系统的输出如红线所示。

10 致谢

本文档使用了 Deron Jackson 和 Alan Gale 编写的早期文档。