目 录

第一章 MALTAB 基础知识 1

1.1 MATLAB 基础知识 1
1.2 MATLAB 基本运算 2
1.3 MATLAB 程序设计 7

第二章 MATLAB 计算结果可视化和确知信号分析 13

2.1 计算结果可视化 13
2.2 确知信号分析 17

第三章 随机信号与数字基带仿真 24

3.1 基本原理和实现示例 24
3.2 蒙特卡罗算法 31

第四章 模拟调制 MATLAB 实现 35

4.1 模拟调制 35
4.2 AM 调制解调的 MATLAB 实现 36

第五章 模拟信号的数字传输 45

5.1 脉冲编码调制 45
5.2 低通抽样定理 45
5.3 均匀量化原理 46
5.4 非均匀量化 48

第六章 数字频带传输系统 52

6.1 数字频带传输原理 52
6.2 信道加性高斯白噪声功率的讨论 53
6.3 仿真分析 54

第七章 通信系统仿真综合实验 68

7.1 基本原理 68
7.2 实验内容 68

第一章 MALTAB 基础知识

本章目标

- 了解 MATLAB 程序设计语言的基本特点，熟悉 MATLAB 软件运行环境
- 掌握创建、保存、打开 m 文件及函数的方法
- 掌握变量等有关概念，具备初步的将一般数学问题转化为对应的计算机模型并进行处理的能力

1.1 MATLAB 基础知识

1.1.1 MATLAB 程序设计语言简介

MATLAB，Matrix Laboratory 的缩写，是由 MathWorks 公司开发的一套用于科学工程计算的可视化高性能语言，具有强大的矩阵运算能力。与大家常用的 Fortran 和 C 等高级语言相比，MATLAB 的语法规则更简单，更贴近人的思维方方式，被称为“草稿纸式的语言”。MATLAB 软件主要由主包、仿真系统（simulink）和工具箱（toolbox）三大部分组成。

1.1.2 MATLAB 界面及帮助

MATLAB 基本界面如图 1-1 所示，命令窗口包含标题栏、菜单栏、工具栏、命令行区、状态栏、垂直和水平波动条等区域。

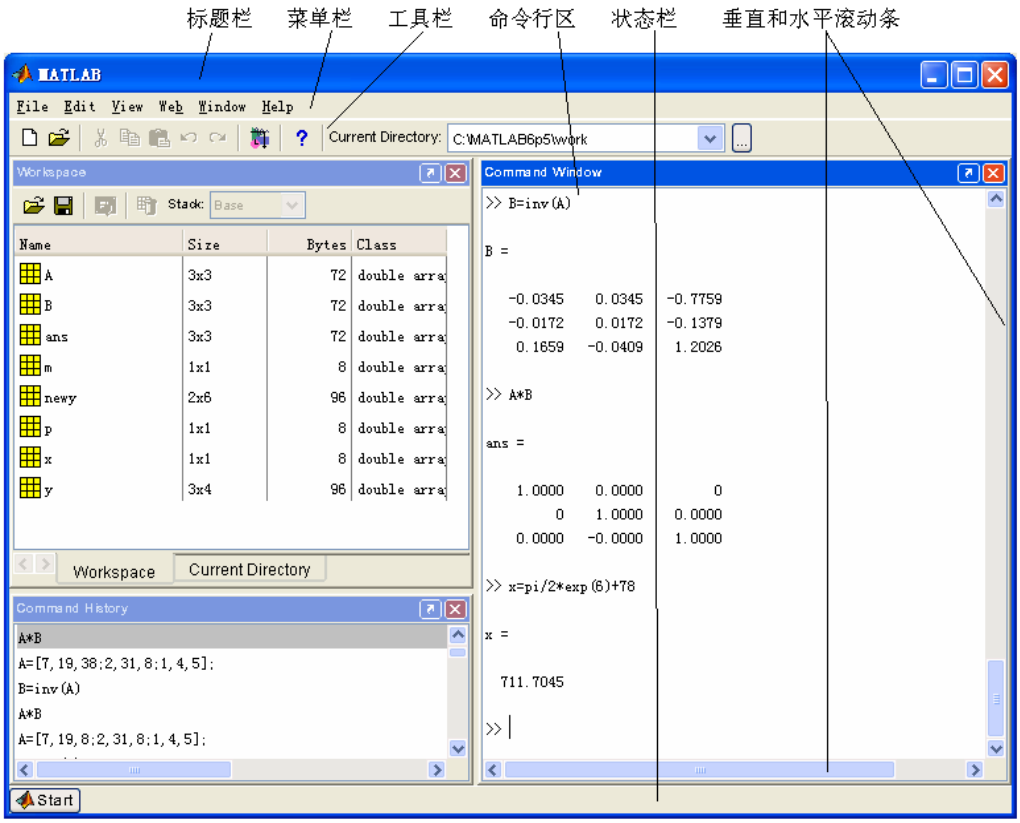


图 1-1 MATLAB 基本界面

（1）菜单栏

在 MATLAB 主窗口的菜单栏，共包含 File、Edit、View、Web、Window 和 Help 6 个菜单项。

- File 菜单项：File 菜单项实现有关文件的操作。
- Edit 菜单项：Edit 菜单项用于命令窗口的编辑操作。
- View 菜单项：View 菜单项用于设置 MATLAB 集成环境的显示方式。
- Web 菜单项：Web 菜单项用于设置 MATLAB 的 Web 操作。
- Window 菜单项：主窗口菜单栏上的 Window 菜单，只包含一个子菜单 Close all，用于关闭所有打开的编辑器窗口，包括 M-file、Figure、Model 和 GUI 窗口。
- Help 菜单项：Help 菜单项用于提供帮助信息。

（2）工具栏

MATLAB 主窗口的工具栏共提供了 10 个命令按钮。这些命令按钮均有对应的菜单命令，但比菜单命令使用起来更快捷、方便。

（3）命令行区

MATLAB 按以下顺序对输入命令进行解释：

- 检查它是否是工作空间中的变量，实则显示变量内容。
- 检查它是否是嵌入函数，是则运行之。
- 检查它是否是子函数。
- 检查它是否是私有函数。
- 检查它是否是位于 MATLAB 搜索路径范围内的函数文件或脚本文件。

MATLAB 里有以下几种方法可获得帮助：

（1）帮助命令（help）是查询函数相关信息的最直接方式，信息会直接显示在命令窗口中。键入 help sin，会显示 sin 相关信息。

（2）lookfor 命令可以从键入的关键字列出所有相关的题材，和 help 相比，lookfor 覆盖范围更广，可查找到某个主题所有词组或短语。

（3）帮助窗口（help window）提供与帮助命令相同的信息，但帮助窗口界面更为方便直接。

（4）帮助桌面（help desk）通过在命令窗口中选择帮助菜单的“help desk”选项或键入 helpdesk 命令即可进入帮助桌面。

（5）在线帮助页是帮助桌面的在线帮助均有相应的 PDF 格式文件。

（6）Mathworks 网站，对于连接入 Internet 的用户通过 Mathworks 公司的网站 www.mathworks.com 询问有关问题。

1.2 MATLAB 基本运算

1.2.1 MATLAB 内部特殊变量和常数

MATLAB 内部有很多变量和常数，用来表达特殊含义，常用的有：

- 变量 ans：指示当前未定义变量名的答案。
- 常数 eps：表示浮点相对精度，其值是从 1.0 到下一个最大浮点数之间的差值。
- 常数 Inf：表示无穷大。
- 虚数单位 i j：表示不定型值，是由 0/0 运算产生。
- 常数 pi：表示圆周率 π 。

1.2.2 变量类型

（1）变量命名规则

MATLAB 中对变量的命名应遵循以下规则：

- 变量名可以由字母、数字和下划线混合组成，但必须以字母开头。

- 字符长度不能大于 31。
- 变量命名区分大小写。

（2）局部变量和全局变量

局部变量是指那些每个函数体内自己定义的，不能从其他函数和 MATLAB 工作空间访问的变量。

全局变量是指用关键字“global”声明的变量。全局变量名应尽量大写，并能反映它本身的含义。如果需要在工作空间和几个函数中都能访问一个全局变量，必须在工作空间和几个函数中都声明该变量是全局的。

1.2.3 内存变量管理

（1）内存变量的删除与修改

clear 命令用于删除 MATLAB 工作空间中的变量。

clc 命令用于清空 MATLAB 命令窗口中的变量。

who 和 whos 这两个命令用于显示在 MATLAB 工作空间中已经驻留的变量名清单。who 命令只显示出驻留变量的名称，whos 在给出变量名的同时，还给出它们的大小、所占字节数及数据类型等信息。

（2）内存变量文件

利用 MAT 文件可以把当前 MATLAB 工作空间中的一些有用变量长久地保留下来，扩展名是.mat。MAT 文件的生成和装入由 save 和 load 命令来完成。常用格式为：

save 文件名 [变量名表] [-append][-ascii]

load 文件名 [变量名表] [-ascii]

1.2.4 MATLAB 常用数学函数

MATLAB 提供了许多数学函数，函数的自变量规定为矩阵变量，运算法则是将函数逐项作用于矩阵的元素上，因而运算的结果是一个与自变量同维数的矩阵。函数使用说明：

- 三角函数以弧度为单位计算。
- abs 函数可以求实数的绝对值、复数的模、字符串的 ASCII 码值。
- 用于取整的函数有 fix、floor、ceil、round，要注意它们的区别。
- rem 与 mod 函数的区别。rem(x,y)和 mod(x,y)要求 x,y 必须为相同大小的实矩阵或为标量。

1.2.5 MATLAB 矩阵生成

MATLAB 具有强大的矩阵运算和数据处理功能，对矩阵的处理必须遵从代数规则。

（1）矩阵的建立

- 直接输入法

最简单的建立矩阵的方法是从键盘直接输入矩阵的元素。具体方法如下：将矩阵的元素用方括号括起来，按矩阵行的顺序输入各元素，同一行的各元素之间用空格或逗号分隔，不同行的元素之间用分号分隔。

- 利用 M 文件建立矩阵

对于比较大且比较复杂的矩阵，可以为它专门建立一个 M 文件。下面通过一个简单例子来说明如何利用 M 文件创建矩阵。

- 利用冒号表达式建立一个向量

冒号表达式可以产生一个行向量，一般格式是：e1:e2:e3，其中 e1 为初始值，e2 为步长，e3 为终止值。

在 MATLAB 中，还可以用 linspace 函数产生行向量。其调用格式为：linspace(a,b,n)，其中 a 和 b 是生成向量的第一个和最后一个元素，n 是元素总数。显然，linspace(a,b,n)与

a:(b-a)/(n-1):b 等价。

- 建立大矩阵

大矩阵可由方括号中的小矩阵或向量建立起来。

（2）矩阵的拆分

- 矩阵元素

通过下标引用矩阵的元素，例如

```
A=[1,2,3;4,5,6];
A(1,2)
ans=2
```

采用矩阵元素的序号来引用矩阵元素。矩阵元素的序号就是相应元素在内存中的排列顺序。在 MATLAB 中，矩阵元素按列存储，先第一列，再第二列，依次类推。例如

```
A=[1,2,3;4,5,6];
A(3)
ans=2
```

显然，序号(Index)与下标(Subscript)是一一对应的，以 m×n 矩阵 A 为例，矩阵元素 A(i,j)的序号为(j-1)*m+i。其相互转换关系也可利用 sub2ind 和 ind2sub 函数求得。

- 矩阵拆分

1) 利用冒号表达式获得子矩阵

① A(:,j)表示取 A 矩阵的第 j 列全部元素；A(i,:)表示 A 矩阵第 i 行的全部元素；A(i,j)表示取 A 矩阵第 i 行、第 j 列的元素。

② A(i:i+m,:)表示取 A 矩阵第 i～i+m 行的全部元素；A(:,k:k+m)表示取 A 矩阵第 k～k+m 列的全部元素，A(i:i+m,k:k+m)表示取 A 矩阵第 i～i+m 行内，并在第 k～k+m 列中的所有元素。

此外，还可利用一般向量和 end 运算符来表示矩阵下标，从而获得子矩阵。end 表示某一维的末尾元素下标。

2) 利用空矩阵删除矩阵的元素

在 MATLAB 中，定义[]为空矩阵。给变量 X 赋空矩阵的语句为 X=[]。注意，X=[]与 clear X 不同，clear 是将 X 从工作空间中删除，而空矩阵则存在于工作空间中，只是维数为 0。

（3）特殊矩阵

常用的产生通用特殊矩阵的函数有：

zeros：产生全 0 矩阵(零矩阵)。

ones：产生全 1 矩阵(幺矩阵)。

eye：产生单位矩阵。

rand：产生 0～1 间均匀分布的随机矩阵。

randn：产生均值为 0，方差为 1 的标准正态分布随机矩阵。

例 1-1 分别建立 3×3、3×2 和与矩阵 A 同样大小的零矩阵。

(1) 建立一个 3×3 零矩阵。

```
zeros(3)
```

(2) 建立一个 3×2 零矩阵。

```
zeros(3,2)
```

(3) 设 A 为 2×3 矩阵，则可以用 zeros(size(A))建立一个与矩阵 A 同样大小零矩阵。

```
A=[1 2 3;4 5 6];   %产生一个 2×3 阶矩阵 A
zeros(size(A))     %产生一个与矩阵 A 同样大小的零矩阵
```

例 1-2 建立随机矩阵：(1) 在区间[20,50]内均匀分布的 5 阶随机矩阵。(2) 均值为 0.6、方差为 0.1 的 5 阶正态分布随机矩阵。

命令如下：

```
x=20+(50-20)*rand(5)
y=0.6+sqrt(0.1)*randn(5)
```

此外，常用的函数还有 reshape(A,m,n)，它在矩阵总元素保持不变的前提下，将矩阵 A 重新排成 m×n 的二维矩阵。

1.2.6 MATLAB 矩阵运算

（1）算术运算

- 基本算术运算

MATLAB 的基本算术运算有：十(加)、一(减)、*(乘)、/(右除)、\(左除)、^(乘方)。注意，运算是在矩阵意义下进行的，单个数据的算术运算只是一种特例。

1) 矩阵加减运算

假定有两个矩阵 A 和 B，则可以由 A+B 和 A-B 实现矩阵的加减运算。运算规则是：若 A 和 B 矩阵的维数相同，则可以执行矩阵的加减运算，A 和 B 矩阵的相应元素相加减。如果 A 与 B 的维数不相同，则 MATLAB 将给出错误信息，提示用户两个矩阵的维数不匹配。

2) 矩阵乘法

假定有两个矩阵 A 和 B，若 A 为 m×n 矩阵，B 为 n×p 矩阵，则 C=A*B 为 m×p 矩阵。

3) 矩阵除法

在 MATLAB 中，有两种矩阵除法运算：\和/，分别表示左除和右除。如果 A 矩阵是非奇异方阵，则 A\B 和 B/A 运算可以实现。A\B 等效于 A 的逆左乘 B 矩阵，也就是 inv(A)*B，而 B/A 等效于 A 矩阵的逆右乘 B 矩阵，也就是 B*inv(A)。对于含有标量的运算，两种除法运算的结果相同，如 3/4 和 4\3 有相同的值，都等于 0.75。又如，设 a=[10.5,25]，则 a/5=5\a=[2.1000 5.0000]。对于矩阵来说，左除和右除表示两种不同的除数矩阵和被除数矩阵的关系。对于矩阵运算，一般 A\B≠B/A。

4) 矩阵的乘方

一个矩阵的乘方运算可以表示成 A^x，要求 A 为方阵，x 为标量。

- 点运算

在 MATLAB 中，有一种特殊的运算，因为其运算符是在有关算术运算符前面加点，所以叫点运算。点运算符有.*、./、.\和.^。两矩阵进行点运算是指它们的对应元素进行相关运算，要求两矩阵的维参数相同。

（2）关系运算

MATLAB 提供了 6 种关系运算符：<(小于)、<=(小于或等于)、>(大于)、>=(大于或等于)、==(等于)、～=(不等于)。它们的含义不难理解，但要注意其书写方法与数学中的不等式符号不尽相同。关系运算符的运算法则为：

- 当两个比较量是标量时，直接比较两数的大小。若关系成立，关系表达式结果为 1，否则为 0。
- 当参与比较的量是两个维数相同的矩阵时，比较是对两矩阵相同位置的元素按标量关系运算规则逐个进行，并给出元素比较结果。最终的关系运算的结果是一个维数与原矩阵相同的矩阵，它的元素由 0 或 1 组成。
- 当参与比较的一个是标量，而另一个是矩阵时，则把标量与矩阵的每一个元素按标量关系运算规则逐个比较，并给出元素比较结果。最终的关系运算的结果是一个维数与原矩阵相同的矩阵，它的元素由 0 或 1 组成。

例 1-3 产生 5 阶随机方阵 A，其元素为[10,90]区间的随机整数，然后判断 A 的元素是否能被 3 整除。

(1) 生成 5 阶随机方阵 A。

```
A=fix((90-10+1)*rand(5)+10)
```

(2) 判断 A 的元素是否可以被 3 整除。

```
P=rem(A,3)==0
```

其中，rem(A,3)是矩阵 A 的每个元素除以 3 的余数矩阵。此时，0 被扩展为与 A 同维数的零矩阵，P 是进行等于(==)比较的结果矩阵。

（3）逻辑运算

MATLAB 提供了 3 种逻辑运算符：&(与)、|(或)和～(非)。逻辑运算的运算法则为：

- 在逻辑运算中，确认非零元素为真，用 1 表示，零元素为假，用 0 表示。
- 设参与逻辑运算的是两个标量 a 和 b，那么，

  a&b　a,b 全为非零时，运算结果为 1，否则为 0。

  a|b　a,b 中只要有一个非零，运算结果为 1。

  ~a　当 a 是零时，运算结果为 1；当 a 非零时，运算结果为 0。
- 若参与逻辑运算的是两个同维矩阵，那么运算将对矩阵相同位置上的元素按标量规则逐个进行。最终运算结果是一个与原矩阵同维的矩阵，其元素由 1 或 0 组成。
- 若参与逻辑运算的一个是标量，一个是矩阵，那么运算将在标量与矩阵中的每个元素之间按标量规则逐个进行。最终运算结果是一个与矩阵同维的矩阵，其元素由 1 或 0 组成。
- 逻辑非是单目运算符，也服从矩阵运算规则。在算术、关系、逻辑运算中，算术运算优先级最高，逻辑运算优先级最低。

例 1-4 建立矩阵 A，然后找出大于 4 的元素的位置。

```
A=[4,-65,-54,0,6;56,0,67,-45,0]
find(A>4)
```

1.2.7 MATLAB 中的矩阵分析

（1）矩阵的转置

转置运算符是单撇号’。

（2）矩阵的旋转

利用函数 rot90(A,k)将矩阵 A 旋转 90°的 k 倍，当 k 为 1 时可省略。

（3）矩阵的逆

对于一个方阵 A，如果存在一个与其同阶的方阵 B，使得：A*B=B*A=I (I 为单位矩阵)则称 B 为 A 的逆矩阵，当然，A 也是 B 的逆矩阵。求一个矩阵的逆是一件非常烦琐的工作，容易出错，但在 MATLAB 中，求一个矩阵的逆非常容易。求方阵 A 的逆矩阵可调用函数 inv(A)。

（4）方阵的行列式

把一个方阵看作一个行列式，并对其按行列式的规则求值，这个值就称为矩阵所对应的行列式的值。在 MATLAB 中，求方阵 A 所对应的行列式的值的函数是 det(A)。

（5）矩阵的秩

矩阵线性无关的行数与列数称为矩阵的秩。在 MATLAB 中，求矩阵秩的函数是 rank(A)。

1.3 MATLAB 程序设计

1.3.1 M 文件

用 MATLAB 语言编写的程序，称为 M 文件。M 文件可以根据调用方式的不同分为两类：命令文件(Script File)和函数文件(Function File)。M 文件是一个文本文件，它可以用任何编辑程序来建立和编辑，而一般常用且最为方便的是使用 MATLAB 提供的文本编辑器。

（1）建立新的 M 文件

启动 MATLAB 文本编辑器有 3 种方法：

- 菜单操作。从 MATLAB 主窗口的 File 菜单中选择 New 菜单项，再选择 M-file 命令，屏幕上将出现 MATLAB 文本编辑器窗口。
- 命令操作。在 MATLAB 命令窗口输入命令 edit，启动 MATLAB 文本编辑器后，输入 M 文件的内容并存盘。
- 命令按钮操作。单击 MATLAB 主窗口工具栏上的 New M-File 命令按钮，启动 MATLAB 文本编辑器后，输入 M 文件的内容并存盘。

（2）打开已有的 M 文件

打开已有的 M 文件，也有 3 种方法：

- 菜单操作。从 MATLAB 主窗口的 File 菜单中选择 Open 命令，则屏幕出现 Open 对话框，在 Open 对话框中选中所需打开的 M 文件。在文档窗口可以对打开的 M 文件进行编辑修改，编辑完成后，将 M 文件存盘。
- 命令操作。在 MATLAB 命令窗口输入命令：edit 文件名，则打开指定的 M 文件。
- 命令按钮操作。单击 MATLAB 主窗口工具栏上的 Open File 命令按钮，再从弹出的对话框中选择所需打开的 M 文件。

例 1-5 分别建立命令文件和函数文件，将华氏温度 f 转换为摄氏温度 c。

程序 1：

首先建立命令文件并以文件名 f2c.m 存盘。

```
clear;          %清除工作空间中的变量
f=input('Input Fahrenheit temperature：');
c=5*(f-32)/9
```

然后在 MATLAB 的命令窗口中输入 f2c，将会执行该命令文件，执行情况为：

```
Input Fahrenheit temperature：73
c =
    22.7778
```

程序 2：

首先建立函数文件 f2c.m。

```
function c=f2c(f)
c=5*(f-32)/9
```

然后在 MATLAB 的命令窗口调用该函数文件。

```
clear;
y=input('Input Fahrenheit temperature：');
x=f2c(y)
```

输出情况为：

```
Input Fahrenheit temperature：70
c =
    21.1111
```

```
x =
    21.1111
```

1.3.2 程序控制结构

（1）顺序结构

- 数据的输入，该函数的调用格式为：A=input(提示信息，选项);

例 1-5 想输入一个人的姓名，可采用命令：

```
xm=input('What''s your name?','s');  %采用's'选项，则允许用户输入一个字符串。
```

- 数据的输出，其调用格式为：disp(输出项);
- 程序的暂停可以使用 pause 函数，其调用格式为：pause(延迟秒数)；如果省略延迟时间，直接使用 pause，则将暂停程序，直到用户按任一键后程序继续执行。若要强行中止程序的运行可使用 Ctrl+C 命令。

（2）选择结构

- if 语句

在 MATLAB 中，if 语句有 3 种格式。

1) 单分支 if 语句：

```
if  条件
        语句组
    end
```

当条件成立时，则执行语句组，执行完之后继续执行 if 语句的后继语句，若条件不成立，则直接执行 if 语句的后继语句。

2) 双分支 if 语句：

```
if  条件
        语句组 1
    else
        语句组 2
    end
```

当条件成立时，执行语句组 1，否则执行语句组 2，语句组 1 或语句组 2 执行后，再执行 if 语句的后继语句。

例 1-6 计算分段函数的值。

程序如下：

```
x=input('请输入 x 的值:');
if x<=0
    y= (x+sqrt(pi))/exp(2);
else
    y=log(x+sqrt(1+x*x))/2;
end
y
```

3) 多分支 if 语句：

```
if  条件 1
        语句组 1
    elseif  条件 2
        语句组 2
        ……
    elseif  条件 m
```

```
        语句组 m
    else
        语句组 n
    end
```

语句用于实现多分支选择结构。

例 1-7 输入一个字符，若为大写字母，则输出其对应的小写字母；若为小写字母，则输出其对应的大写字母；若为数字字符则输出其对应的数值，若为其他字符则原样输出。

```
c=input('请输入一个字符','s');
if c>='A' & c<='Z'
    disp(setstr(abs(c)+abs('a')-abs('A')));
elseif c>='a'& c<='z'
    disp(setstr(abs(c)-abs('a')+abs('A')));
elseif c>='0'& c<='9'
    disp(abs(c)-abs('0'));
else
    disp(c);
end
```

- switch 语句

  switch 语句根据表达式的取值不同，分别执行不同的语句，其语句格式为：

```
switch  表达式
  case  表达式 1
    语句组 1
  case  表达式 2
    语句组 2
    ……
  case  表达式 m
    语句组 m
  otherwise
    语句组 n
end
```

当表达式的值等于表达式 1 的值时，执行语句组 1，当表达式的值等于表达式 2 的值时，执行语句组 2，…，当表达式的值等于表达式 m 的值时，执行语句组 m，当表达式的值不等于 case 所列的表达式的值时，执行语句组 n。当任意一个分支的语句执行完后，直接执行 switch 语句的下一句。

例 1-8 某商场对顾客所购买的商品实行打折销售，标准如下(商品价格用 price 来表示)：

price<200	没有折扣
200≤price<500	3%折扣
500≤price<1000	5%折扣
1000≤price<2500	8%折扣
2500≤price<5000	10%折扣
5000≤price	14%折扣

输入所售商品的价格，求其实际销售价格。

程序如下：

```
price=input('请输入商品价格');
```

```
switch fix(price/100)
    case {0,1}                  %价格小于 200
        rate=0;
    case {2,3,4}                %价格大于等于 200 但小于 500
        rate=3/100;
    case num2cell(5:9)          %价格大于等于 500 但小于 1000
        rate=5/100;
    case num2cell(10:24)        %价格大于等于 1000 但小于 2500
        rate=8/100;
    case num2cell(25:49)        %价格大于等于 2500 但小于 5000
        rate=10/100;
    otherwise                   %价格大于等于 5000
        rate=14/100;
end
price=price*(1-rate)    %输出商品实际销售价格
```

- try 语句

语句格式为：

```
try
    语句组 1
catch
    语句组 2
end
```

try 语句先试探性执行语句组 1，如果语句组 1 在执行过程中出现错误，则将错误信息赋给保留的 lasterr 变量，并转去执行语句组 2。

例 1-9 矩阵乘法运算要求两矩阵的维数相容，否则会出错。先求两矩阵的乘积，若出错，则自动转去求两矩阵的点乘。

程序如下：

```
A=[1,2,3;4,5,6]; B=[7,8,9;10,11,12];
try
    C=A*B;
catch
    C=A.*B;
end
C
lasterr                 %显示出错原因
```

（3）循环结构

- for 语句

for 语句的格式为：

```
for  循环变量=表达式 1:表达式 2:表达式 3
        循环体语句
end
```

其中表达式 1 的值为循环变量的初值，表达式 2 的值为步长，表达式 3 的值为循环变量的终值。步长为 1 时，表达式 2 可以省略。

例 1-10 一个三位整数各位数字的立方和等于该数本身则称该数为水仙花数。输出全部水仙

花数。

程序如下：

```
for m=100:999
m1=fix(m/100);                  %求 m 的百位数字
m2=rem(fix(m/10),10);     %求 m 的十位数字
m3=rem(m,10);         %求 m 的个位数字
if m==m1*m1*m1+m2*m2*m2+m3*m3*m3
disp(m)
end
end
```

- while 语句

while 语句的一般格式为：

```
while (条件)
    循环体语句
end
```

其执行过程为：若条件成立，则执行循环体语句，执行后再判断条件是否成立，如果不成立则跳出循环。

例 1-11 从键盘输入若干个数，当输入 0 时结束输入，求这些数的平均值和它们之和。

程序如下：

```
sum=0;
cnt=0;
val=input('Enter a number (end in 0):');
while (val~=0)
  sum=sum+val;
  cnt=cnt+1;
  val=input('Enter a number (end in 0):');
end
if (cnt > 0)
    sum
    mean=sum/cnt
end
```

- break 语句和 continue 语句

break 语句用于终止循环的执行。当在循环体内执行到该语句时，程序将跳出循环，继续执行循环语句的下一语句。

continue 语句控制跳过循环体中的某些语句。当在循环体内执行到该语句时，程序将跳过循环体中所有剩下的语句，继续下一次循环。

例 1-12 求[100，200]之间第一个能被 21 整除的整数。

程序如下：

```
for n=100:200
if rem(n,21)~=0
      continue
end
break
end
```

n

（4）函数文件

- **函数文件的基本结构**

函数文件由 function 语句引导，其基本结构为：

function 输出形参表=函数名(输入形参表)

注释说明部分

函数体语句

其中以 function 开头的一行为引导行，表示该 M 文件是一个函数文件。函数名的命名规则与变量名相同。输入形参为函数的输入参数，输出形参为函数的输出参数。当输出形参多于一个时，则应该用方括号括起来。

- **函数调用**

函数调用的一般格式是：

[输出实参表]=函数名(输入实参表)

要注意的是，函数调用时各实参出现的顺序、个数，应与函数定义时形参的顺序、个数一致，否则会出错。函数调用时，先将实参传递给相应的形参，从而实现参数传递，然后再执行函数的功能。

例 1-13 利用函数的递归调用，求 n！。

n!本身就是以递归的形式定义的：显然，求 n!需要求(n-1)!，这时可采用递归调用。递归调用函数文件 factor.m 如下：

```
function f=factor(n)
if n<=1
    f=1;
else
    f=factor(n-1)*n;       %递归调用求(n-1)!
end
```

第二章 MATLAB 计算结果可视化和确知信号分析

本章目标

- 掌握二维平面图形的绘制方法，能够使用这些方法进行常用的数据可视化处理
- 理解周期信号的傅里叶级数展开的物理意义
- 掌握信号的傅里叶变换及其反变换

2.1 计算结果可视化

MATLAB 在数据可视化方面的表现能力很强。它的图形处理能力不仅功能强大，而且充分考虑了不同层次用户的不同需求，系统具有两个层次的绘图指令：一个层次是直接对图形句柄进行操作的底层绘图指令；另一层次是在底层指令基础上建立的高层绘图指令。常用的 **MATLAB** 绘图语句有 **figure**、**plot**、**subplot**、**stem** 等，图形修饰语具有 **title**、**axis**、**text** 等。

（1）figure 语句

figure 有两种用法。当只有一句 **figure** 命令时，程序会创建一个新的图形窗口，并返回一个整数型的窗口编号。当采用 **figure(n)**时，表示将第 **n** 个图形窗口作为当前的图形窗口，将其显示在所有窗口的最前面。如果该图形窗口不存在，则新建一个窗口，并赋以编号 **n**。

（2）plot 语句

线形绘图函数。用法为 **plot(x,y,'s')**。参数 **x** 为横轴变量，**y** 为纵轴变量，**s** 用以控制图形的基本特征如颜色、粗细等，通常可以省略，常用方法如表 **2-1** 所示。

表 2-1　plot 命令的参数及其含义

参数	含义	参数	含义	参数	含义
y	黄色	**.**	点	**-**	实线
m	紫色	**o**	圆	**:**	虚线
c	青色	**x**	打叉	**-.**	点划线
r	红色	**+**	加号	**--**	破折线
g	绿色	*****	星号	**^**	向上三角形
b	蓝色	**s**	正方形	**<**	向左三角形
w	白色	**d**	菱形	**>**	向右三角形
k	黑色	**v**	向下三角形	**p**	五角星形

（3）subplot 语句

subplot(m,n,i)是分割显示图形窗口命令，它把一个图形窗口分为 **m** 行 **n** 列共 **m×n** 个小窗口，并指定第 **i** 个小窗口为当前窗口。

（4）二维统计分析图

在 **MATLAB** 中，二维统计分析图形很多，常见的有条形图、阶梯图、杆图和填充图等，所采用的函数分别是：

bar(x,y,选项)

stairs(x,y,选项)

stem(x,y,选项)

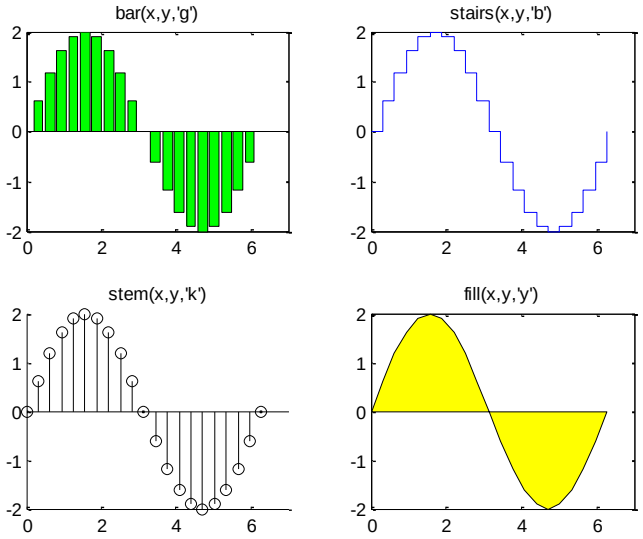
fill(x1,y1,选项 1,x2,y2,选项 2,…)

例 **2-1** 分别以条形图、阶梯图、杆图和填充图形式绘制曲线 **y=2sin(x)**。

程序如下：

```
x=0:pi/10:2*pi;
y=2*sin(x);
subplot(2,2,1);bar(x,y,'g');
title('bar(x,y,''g'')');axis([0,7,-2,2]);
subplot(2,2,2);stairs(x,y,'b');
title('stairs(x,y,''b'')');axis([0,7,-2,2]);
subplot(2,2,3);stem(x,y,'k');
title('stem(x,y,''k'')');axis([0,7,-2,2]);
subplot(2,2,4);fill(x,y,'y');
title('fill(x,y,''y'')');axis([0,7,-2,2]);
```

仿真结果：



（5）图形保持

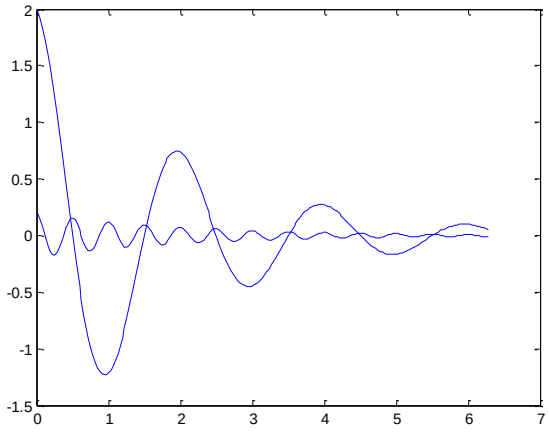
hold on/off 命令控制是保持原有图形还是刷新原有图形，不带参数的 **hold** 命令在两种状态之间进行切换。

例 2-2 采用图形保持，在同一坐标内绘制曲线 $y1=0.2e^{-0.5x}\cos(4\pi x)$ 和 $y2=2e^{-0.5x}\cos(\pi x)$。

程序如下：

```
x=0:pi/100:2*pi;
y1=0.2*exp(-0.5*x).*cos(4*pi*x);
plot(x,y1)
hold on
y2=2*exp(-0.5*x).*cos(pi*x);
plot(x,y2);
hold off
```

仿真结果：



（6）绘图修饰命令

title(图形名称)

xlabel(x 轴说明)

ylabel(y 轴说明)

text(x,y,图形说明)

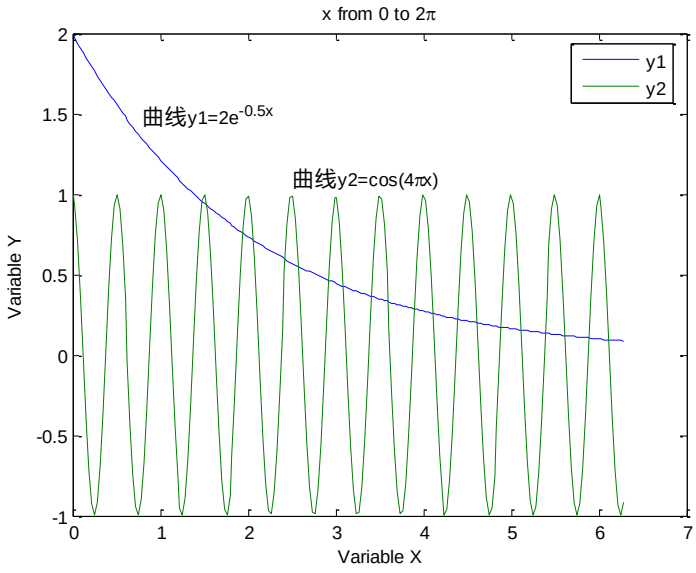
legend(图例 1,图例 2,…)

例 2-3 在 0≤x≤2π 区间内，绘制曲线 y1=2e-0.5x 和 y2=cos(4πx)，并给图形添加图形标注。

程序如下：

```
x=0:pi/100:2*pi;y1=2*exp(-0.5*x);y2=cos(4*pi*x);
plot(x,y1,x,y2)
title('x from 0 to 2{\pi}');                    %加图形标题
xlabel('Variable X');                           %加 X 轴说明
ylabel('Variable Y');                           %加 Y 轴说明
text(0.8,1.5,'曲线 y1=2e^{-0.5x}');             %在指定位置添加图形说明
text(2.5,1.1,'曲线 y2=cos(4{\pi}x)');
legend('y1','y2')                               %加图例
```

仿真结果：



（7）坐标控制

axis 函数的调用格式为：

axis([xmin xmax ymin ymax zmin zmax])

axis 函数功能丰富，常用的格式还有：

axis equal：纵、横坐标轴采用等长刻度。

axis square：产生正方形坐标系(缺省为矩形)。

axis auto：使用缺省设置。

axis off：取消坐标轴。

axis on：显示坐标轴。

给坐标加网格线用 **grid** 命令来控制。**grid on/off** 命令控制是画还是不画网格线，不带参数的 **grid** 命令在两种状态之间进行切换。

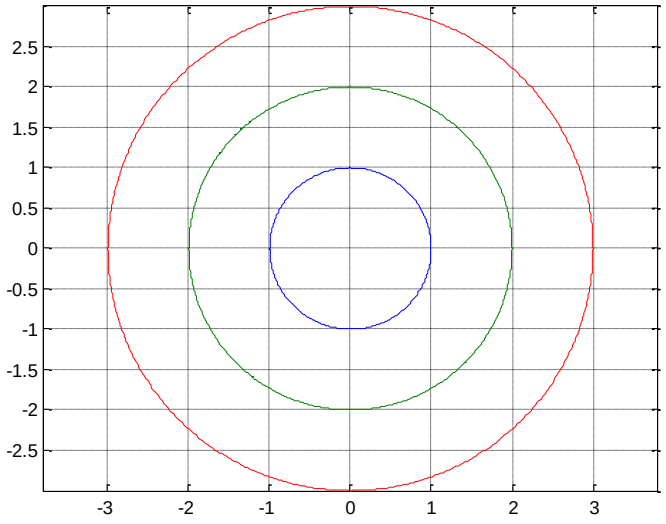
给坐标加边框用 **box** 命令来控制。**box on/off** 命令控制是加还是不加边框线，不带参数的 **box** 命令在两种状态之间进行切换。

例 2-4 在同一坐标中，可以绘制 3 个同心圆，并加坐标控制。

程序如下：

```
t=0:0.01:2*pi;
x=exp(i*t);
y=[x;2*x;3*x]';
plot(y)
grid on;                %加网格线
box on;                 %加坐标边框
axis equal              %坐标轴采用等刻度
```

仿真结果：



2.2 确知信号分析

2.2.1 周期信号的傅里叶级数

（1）基本原理

若一周期信号 $f(t)=f(t+kT)$，其中 k 为整数，T 成为信号的周期。若周期信号在一个周期内可积，则可通过傅立叶级数对该信号进行展开。其傅立叶展开式如（2－1）式所示：

$$f(t)=\sum_{n=-\infty}^{\infty}F_n e^{j2\pi nf_s t} \qquad (2-1)$$

其中，$F_n=\frac{1}{T}\int_{-T/2}^{T/2}f(t)e^{-j2\pi nf_s t}dt$，$T$ 为信号周期；$f_s=1/T$ 为信号的基波；F_n 为傅立叶展开系数，其物理意义为频率分量 nf_s 的幅度和相位。

式 2－1 表明：信号可以展开成一系列频率为 $f_s=1/T$ 的整数倍的正弦、余弦信号的加权叠加，其中相应频率分量的加权系数即为 F_n，因此可以用周期信号的傅立叶展开来重构该周期信号，其逼进程度与展开式的项数有关。

（2）举例

设周期信 号一个周期的波形为 $f(t)=\begin{cases}1, & 0\le t\le T/2\\ -1, & T/2<t\le T\end{cases}$，求该信号傅里叶级数展开式，并用 MATLAB 画出傅里叶级数展开后的波形，并通过展开式项数的变化考察其对 $f(t)$ 的逼近程度，考察其物理意义。

解：

$$\begin{aligned}F_n&=\frac{1}{T}\int_0^T f(t)e^{-j2\pi nf_s t}dt\\ &=\frac{1}{T}\left(\int_0^{\frac{T}{2}}e^{-j2\pi nf_s t}dt-\int_{\frac{T}{2}}^{T}e^{-j2\pi nf_s t}dt\right)\\ &=\frac{1}{T}\left(\frac{e^{-j\pi n}-1}{-j2\pi nf_s}-\frac{1-e^{-j\pi n}}{-j2\pi nf_s}\right)\\ &=\sin c(n/2)e^{-jn\pi/2}\end{aligned}$$

注：$\sin c(x)=\sin\pi x/\pi x=sa(\pi x)$

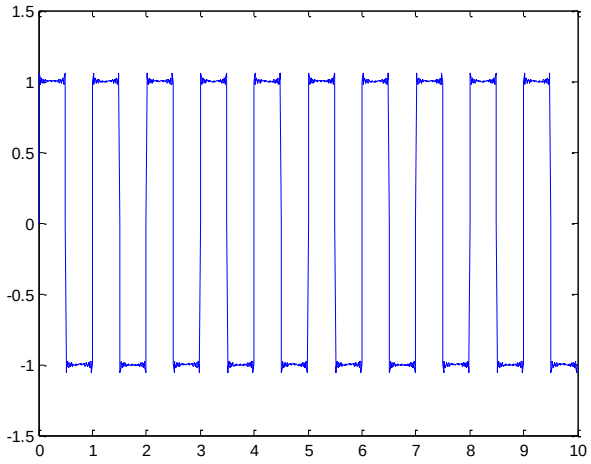
源代码：

```
clear all;
N=20;%取展开式的项数为 2N+1 项
%可以改为N=input('input N:')
T=1;%周期为 1
fs=1/T;
```

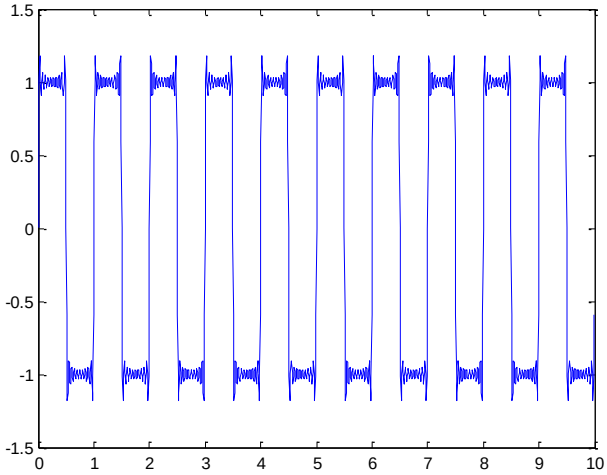
```
N_sample=128;%为了画波形，设置每个周期的采样点数
dt=1/ N_sample;%时间分辨率
t=0:dt:10*T-dt;%取 10 个周期
n=-N:N;
Fn=sinc(n/2).*exp(-j*n*pi/2);%求傅立叶系数
Fn(N+1)=0;%当 n=0 时，代入 Fn 得 Fn=0，由于数组的序号是从 1 开始的，即 n=-N 时对%应 Fn(1), n=0 时对%应 Fn(n+1)，即 n=N 时对%应 Fn(2N+1)
ft=zeros(1,length(t));%建立一个全零数组，其长度和原始信号长度相同，用来存放由傅里叶%展开恢复的信号
for m=-N:N;%一共 2N+1 项累加。
ft=ft+Fn(m+N+1)*exp(j*2*pi*m*fs*t) ;%Fn 是一个数组,而 MATLAB 中数组中元素的序%号是从 1 开始的，故 Fn 序号是从 1 开始的，到 2N＋1 结束，该语句中体现为为 Fn(m+N+1)%而当 n=0 时，Fn=0，在数组中的位置为第 N+1 个元素，故令 Fn(N+1)=0
end
plot(t,ft)
```

仿真结果：

N=100 时，



N=20 时，



可以看出：用周期信号的傅立叶展开来重构该周期信号，其逼进程度与展开式的项数有关。

2.2.2 信号的傅里叶变换及其反变换

（1）基本原理

对于非周期信号$s(t)$，满足绝对可积的条件下，可利用傅里叶变换对其进行频域分析。

$$S(f)=\int_{-\infty}^{\infty}s(t)e^{-j2\pi ft}dt\ ,\ \ s(t)=\int_{-\infty}^{\infty}S(f)e^{j2\pi ft}df$$

其中，$S(f)$称为信号$s(t)$傅里叶变换，表示了该信号的频谱特性。

在数字信号处理中，需要利用离散傅立叶变换（DFT）计算信号的傅里叶变换，现在考察一下信号$s(t)$的傅里叶变换与其离散傅立叶变换之间的关系。

将信号$s(t)$按照时域均匀抽样定理进行等间隔抽样后，得到序列$\{s_n, n=0,1,2,\cdots,N-1\}$，$s_n=s(n\Delta t)$，其中，$\Delta t$为抽样间隔，则由数字信号处理的知识可知，序列$s_n$的离散傅立叶变换为

$$S_k=\sum_{n=0}^{N-1}s_n e^{-j\frac{2\pi}{N}nk}\qquad (k=0,1,2,\cdots,N-1)$$

其中，N 为采样点数。

而$s(t)$在一段时间$[0,T]$内的傅立叶变换为

$$\begin{aligned}S(f)&=\int_0^T s(t)e^{-j2\pi ft}dt\\&=\lim_{N\to\infty}\sum_{n=0}^{N-1}s(n\Delta t)e^{-j2\pi fn\Delta t}\Delta t\\&\overset{令\Delta t=T/N}{=}\lim_{N\to\infty}\frac{T}{N}\sum_{n=0}^{N-1}s(n\Delta t)e^{-j\frac{2\pi}{N}nfT}\\&\overset{注意到s(n\Delta t)=s_n}{=}\lim_{N\to\infty}\frac{T}{N}\sum_{n=0}^{N-1}s_n e^{-j\frac{2\pi}{N}nfT}\end{aligned}$$

得到$s(t)$在一段时间$[0,T]$内的傅立叶变换是连续谱$S(f)$，而对$s(t)$进行离散傅立叶变换得到的是离散谱S_k，为了比较它们之间的关系，对$S(f)$也进行等间隔抽样，且抽样间隔为$\Delta f=1/T$，即其频率分辨率，则在频率范围$[0,(N-1)\Delta f]$内，

$$S(f)=S(k\Delta f)=\lim_{N\to\infty}\frac{T}{N}\sum_{n=0}^{N-1}s_n e^{-j\frac{2\pi}{N}nfT}$$

$$= \lim_{N\to\infty} \frac{T}{N} \sum_{n=0}^{N-1} s_n e^{-j\frac{2\pi}{N}nk}$$

$$= \lim_{N\to\infty} \frac{T}{N} S_k \qquad (k = 0,1,2,\cdots,N-1)$$

可以看到，$s(t)$的离散傅里叶变换与$s(t)$在一段时间$[0,T]$内的傅立叶变换$S(f)$的抽样$S(k\Delta f)$成正比。由于 N 点离散傅里叶变换具有$S_k = S_{k+m*N}$的性质，故信号$s(t)$连续谱的负半轴部分可以通过对S_k的平移得到。

需要注意的是信号$s(t)$的离散傅立叶变换只和信号$s(t)$在一段时间$[0,T]$内的傅立叶变换有关，而由公式 2－1，$s(t)$的频谱是在时间$[-\infty,\infty]$上得到的。所以上述计算所得到的并不是真正的信号频谱，而是信号加了一个时间窗后的频谱。当信号$s(t)$是随时间衰减的或是时限信号，只要时间窗足够长，可以通过这种方法获得信号的近似频谱。因此，用 DFT 计算的信号频谱精度依赖于信号、抽样的时间间隔和时间窗的大小。一般情况下，对于时限信号，在抽样时间间隔小，即抽样频率高的情况下能获得较为精确的信号频谱。

计算信号的离散傅里叶变换在数字信号处理中有一种高效算法，即快速傅里叶变换 FFT，Matlab 中也有专门的工具，下面简要介绍：

fft(x)，x 是离散信号，或对模拟信号取样后的离散值。

ifft(x),，x 是对信号进行快速傅里叶变换后的离散谱。

，

源代码一：

利用 fft,fftshift 定义函数 T2F 计算信号的傅立叶变换

```
function [f,sf]=T2F(t,st)%该子函数需要两个参数 t 和 st。
%t—离散时间；st—离散信号
dt=t(2)-t(1) ;%  时间分辨率
T=t(end) ;
df=1/T ;%频率分辨率
N=length(st) ;%离散傅立叶变换长度
f=-N/2*df :df :N/2*df-df ;%设定频谱区间，注意要关于原点对称，共有 N 个点，包括 0 点，%故要减去一个 df
sf=fft(st);
```

sf=T/N*fftshift(sf);%信号的频谱与离散傅立叶变换之间的关系，fftshift(x)是将信号%的频谱 x 进行移位，与原点对称。

源代码二：

利用 ifft,fftshift 定义函数 T2F 计算信号的傅立叶反变换

```
function [t,st]= F2T (f,sf)
%f 离散的频率；sf—信号的频谱
df=f(2)-f(1) ; %频率分辨率
Fmx=f(end)-f(1)+df ;%频率区间长度
dt=1/Fmx ; %已知频率区间长度时，求时间分辨率，由前面频率分辨率公式△f=df=1/T，
%T=dt*N，得到△f=df=1/ (dt*N)，故 dt=1/(df*N)=1/Fmx，即时间分辨率
N=length(sf) ;
T=dt*N; %信号持续时间
t=0:dt:T-dt;
%离散傅立叶反变换，是 T2F 的逆过程
sff=fftshift(sf); %把对称的频谱进行平移，平移后同 T2F 中的 sf
st=Fmx*ifft(sff); %由于 T2F 中求信号频谱在 DFT 基础上乘了一个因子 T/N，反变换求信
号时要乘以其倒数即 N / T＝1/dt，正好等于 Fmx。
```

（2）举例

设非周期信号 $s(t)=\begin{cases}1, & 0\le t\le T/2\\ -1, & T/2<t\le T\end{cases}$，求该信号的傅里叶变换，用 MATLAB 画出傅里叶变换后的频谱，并对频谱进行反变换，画出 $s(t)$ 的波形。

解：

$$S(f)=\int_0^{\frac{T}{2}}e^{-j2\pi ft}dt-\int_{\frac{T}{2}}^{T}e^{-j2\pi ft}dt$$

$$=\frac{e^{-j\pi fT}-1}{-j2\pi f}-\frac{e^{-j2\pi fT}-e^{-j\pi fT}}{-j2\pi f}$$

$$=\frac{1-e^{-j\pi fT}}{j2\pi f}-\frac{e^{-j\pi fT}-e^{-j2\pi fT}}{j2\pi f}=\frac{\left(1-e^{-j\pi fT}\right)^2}{j2\pi f}$$

$$=\frac{\left[e^{-j\frac{1}{2}\pi fT}\left(e^{j\frac{1}{2}\pi fT}-e^{-j\frac{1}{2}\pi fT}\right)\right]^2}{j2\pi f}$$

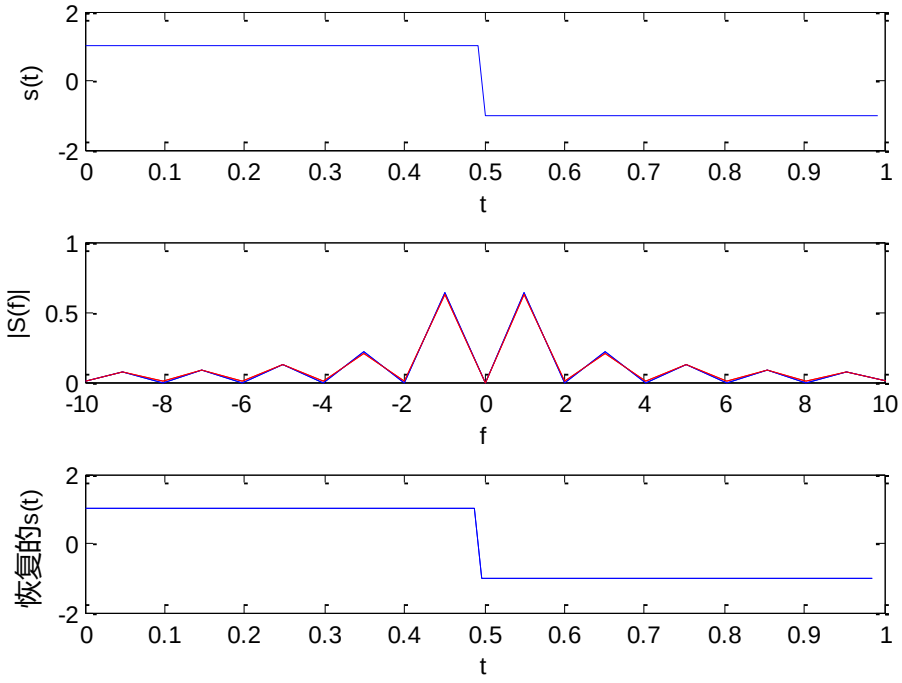
$$=e^{-j\pi fT}\frac{-4\sin^2\left(\frac{1}{2}\pi fT\right)}{j2\pi f}=e^{-j\pi fT}\frac{\sin^2\left(\frac{1}{2}\pi fT\right)}{\left(\frac{1}{2}\pi fT\right)^2}j\frac{\pi f}{2}T^2$$

$$=j\frac{\pi f}{2}T^2e^{-j\pi fT}\sin c^2\left(fT/2\right)$$

主程序：

```
clear all
T=1;
N_sample=128;%为了画波形，设置每个周期的采样点数
dt=1/ N_sample;%时间分辨率
t=0:dt:T-dt;
st=[ones(1, N_sample/2), -ones(1, N_sample/2)];%依据 T 将信号离散化
subplot(311);plot(t,st);axis([0 1 -2 2]);xlabel('t');ylabel('s(t)');
subplot(312) ;
[f,sf]=T2F(t,st) ;
plot(f,abs(sf)) ;hold on ;%画出 sf 的幅度谱，不含相位
axis([-10 10 0 1]);
xlabel('f');ylabel('|S(f)|');
sff=T^2*j*pi*f*0.5.*exp(-j*2*pi*f*T).*sinc(f*T*0.5).*sinc(f*T*0.5) ;%依据傅里叶变换求%
信号频谱
plot(f,abs(sff),'r-')
[t,st]= F2T (f,sf);%进行离散傅立叶反变换，求原始信号
subplot(313) ;
axis([0 1 -2 2]);
xlabel('t');ylabel('恢复的 s(t)');
plot(t,st) ;hold on ;
```

仿真结果：


s(t)
t
|S(f)|
f
恢复的s(t)
t

第三章 随机信号与数字基带仿真

本章目标

- 掌握库函数产生随机数方法
- 理解采用蒙特卡罗算法仿真的思想
- 基带信号波形生成和其功率谱密度

3.1 基本原理和实现示例

3.1.1.库函数产生随机数

① 均匀分布的随机数

利用 MATLAB 库函数 rand 产生。rand 函数产生（0，1）内均匀分布的随机数，使用方法如下：

1）x=rand(m)；产生一个 m×m 的矩阵，所含元素取值均为在（0，1）内均匀分布的随机数。

2）x=rand(m，n)；产生一个 m×n 的矩阵，所含元素取值均为在（0，1）内均匀分布的随机数。

3）x=rand；产生一个随机数。

② 高斯分布的随机数

randn 函数产生均值为 0，方差为 1 的高斯分布的随机数，使用方法如下：

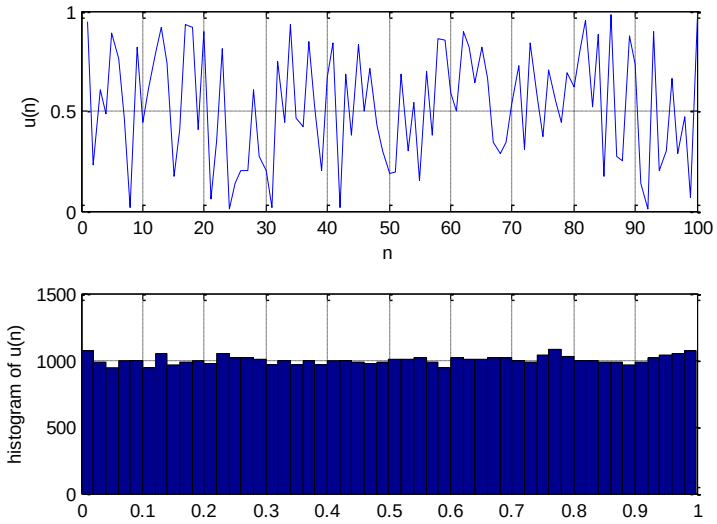
1）x=randn(m)；产生一个 m×m 的矩阵，所含元素都是均值为 0，方差为 1 的高斯分布的随机数。

2）x=randn(m，n)；产生一个 m×n 的矩阵，所含元素都是均值为 0，方差为 1 的高斯分布的随机数。

3）x=randn；产生一个均值为 0，方差为 1 的高斯分布的随机数。

例题 3-1 产生一个(0,1)上均匀分布的白噪声信号 u(n)，画出其波形，并检验其分布(exa0301_rand.m)

```
clear;%清除内存中可能保留的 MATLAB 变量
N=500000;%u(n)的长度
u=rand(1,N);%调用 rand，得到均匀分布的随机数 u(n)
u_mean=mean(u);%求 u(n)均值
power_u=var(u);%求 u(n)方差
subplot(211)
plot(u(1:100));grid on;%在一个图上分上下两个子图
ylabel('u(n) ');%给 y 轴加坐标
xlabel('n'); %给 x 轴加坐标
subplot(212)
hist(u,50);grid on;%对 u(n)做直方图，检验其分布，50 是对取值范围[0 1]均分等分 50 份，网格
ylabel('histogram of u(n) ');
```



运算结果：均值0.5,方差0.083

例题 3-2 产生一个均值为 0.01，功率为 0.1 的均匀分布的白噪声信号 u(n)，画出其波形。

解：

分析：在例题 3.1 中 x=rand(1，N)给出均值为 0.5，功率为 0.083 的白噪声，

现在要均值为 m_u，方差为 σ_u^2 (平均功率为 $P=\sigma_u^2+m_u^2$)，实施步骤：

1. 将随机变量 x 的均值调整为 m_u：即 $u=x-mean(x)+m_u$

2. 将随机变量 x 的方差调整为 σ_u^2：即 $u=a\times x=\dfrac{\sigma_u}{\sigma_x}x$

3. 随机变量 $u=cx+d$ 的均值为 m_u、方差为 σ_u^2，则 c、d 分别为：

$$m_u=E[cx+d]=cm_x+d$$

$$\sigma_u^2=E[u-m_u]^2=E[cx+d-(cm_x+d)]^2=c^2E[x-m_x]^2=c^2\sigma_x^2$$

由上面的两个方程，可以求解出： $c=\dfrac{\sigma_u}{\sigma_x}$

$$d=m_u-cm_x=m_u-m_x\times\frac{\sigma_u}{\sigma_x}$$

4. 随机变量 $u=cx+d$ 的均值为 m_u、功率为 P，则 c、d 分别为：

$$m_u=E[cx+d]=cm_x+d$$

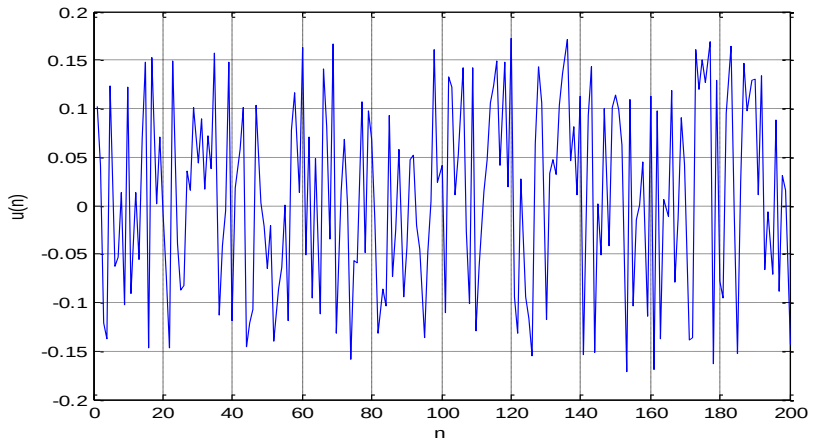
$$P=\sigma_u^2+m_u^2=c^2\sigma_x^2+m_u^2$$

由上面的两个方程，可以求解出： $c=\sqrt{\frac{P-m_u^2}{\sigma^2{}_x}}$

$$d=m_u-cm_x=m_u-m_x\times\sqrt{\frac{P-m_u^2}{\sigma^2{}_x}}$$

```
clear;
P=0.1;                          % 希望的功率
mu=0.01;                        % 希望的均值
N=50000;                        % 序列长度
x=rand(1,N);                    % 生产N个[0，1]上均匀分布的随机序列x
temp=var(x);                    % x序列的方差
c=sqrt((P-mu^2)/var(x));        % 求系数c
d=mu-c*mean(x);                 % 求系数d
u=c*x+d;                        % 求均值为ma、功率为P的随机序列u
mean_u=mean(u)                  % 求随机序列u的均值
power_u=dot(u,u)/N              % 求随机序列u的功率，检验u1(n)的功率是否满足要求dot
%是matlab内部函数，实现两个向量的乘积，该句等效于var
plot(u(1:200));grid on;
ylabel('u(n)')
xlabel('n')
```

仿真结果：



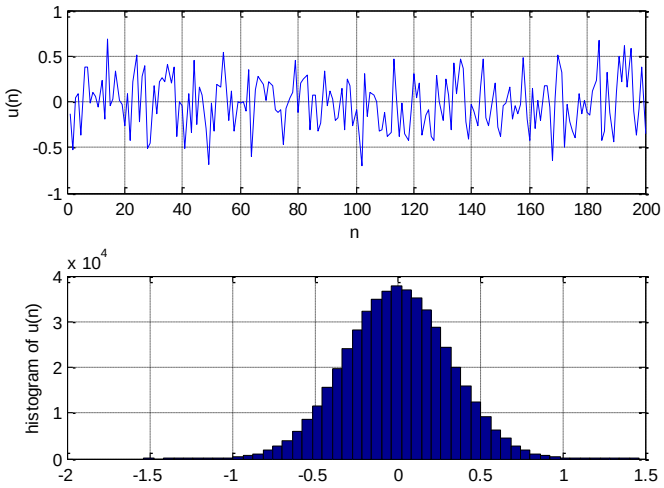
例题 3-3 产生一个均值为 0，方差为 0.1，服从高斯分布的白噪声信号 u(n)，画出其波形.(exa0303_rand.m)

解：调用**randn**，生成均值为0，方差为1，服从高斯分布的白噪声信号u(n)。调整均值：用u(n)加上均值即可；调整功率：令希望的功率为P，则需要求出常数a，$a=\sqrt{P}$，用a乘**u(n)**。

```
clear;
p=0.1;
N=500000;
u=rand(1,N);
```

```
a=sqrt(p);
u=u*a;
power_u=var(u);
subplot(211)
plot(u(1:200));grid on;
ylabel('u(n) ');
xlabel('n');
subplot(212)
hist(u,50);grid on;
ylabel('histogram of u(n) ');
```

仿真结果：



例题 3-4 通过 Matlab 中的函数 rand 产生[0,1]均匀分布的随机变量,并用其产生一个 0、1 的随机序列，其中：0 的概率为 p=0.3，1 的概率为 1-p=1-0.3=0.7。(exa0304_rand.m)

解　若能产生[0,1]均匀分布的随机变量 X,则令 $Y=\begin{cases}0, & X\le p\\ 1, & X>p\end{cases}$，则 P(Y=0)=p, P(Y=1)=1-p。

```
%产生一个(p,1-p)的 0-1 随机变量,文件 rand01.m
function s=rand01(p,m,n)
% 输入参数:
% p:0-1 分布中 1 的概率
% m,n:产生的随机变量样本个数 m×n
% 输出:产生的随机变量样本矢量
x=rand(m,n);
s=(sign(x-p+eps)+1)/2;   %  eps = 2^(-52).
```

例题 3-5 通过 Matlab 中的函数 randn 产生 N(0,1)的高斯随机变量,并用其产生 $\sigma^2=2$ 的瑞利分布随机变量。(exa0305_rand.m)

解：可以证明,两个独立同分布、均值为 0 的高斯随机变量 N(0, σ^2)的平方和开根号所得的随机变量服从功率为 $2\sigma^2$ 的瑞利分布。

```
functions=rayleigh(sigma2,m,n)
% 输入参数: sigma2，瑞利分布的功率；m,n: 输出 m×n 个样本
x=sqrt(sigma2/2)*randn(m,n);
y=sqrt(sigma2/2)*randn(m,n);
s=sqrt(x.*x+y.*y);
```

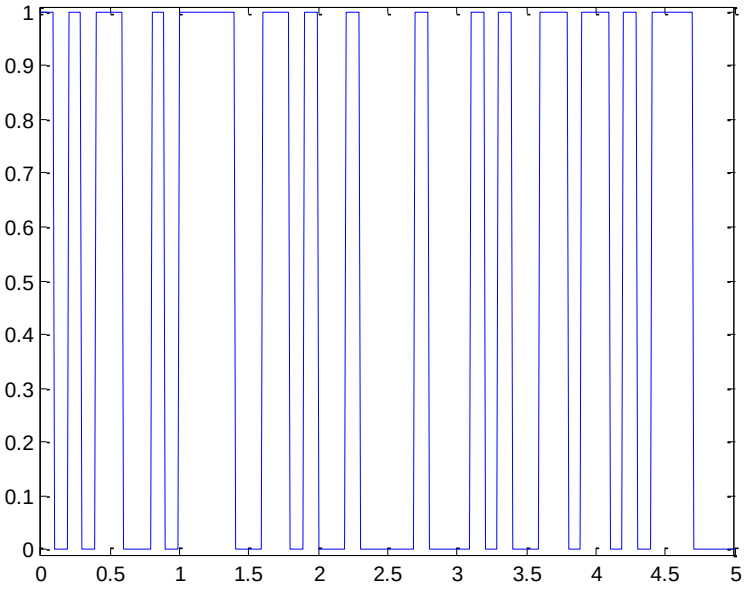
3.1.2 基带信号波形生成和其功率谱密度

1).要画出完整的波形，每一个码元要采 n 个样，如果一个 N 个码元的 0、1 序列 x，要画出它的矩形脉冲波形，可以用如下方法完成。

例题 3-6 产生一个 N 码元，每码元采样 n 个的 0、1 序列. (exa0306_rand.m)

```
N=10000;                                    % 二进制序列的长度
dsource =(sign(rand(1,N)-0.5+eps)+1)/2; % 生成 N 码元的 0、1 序列
n=10;    % 每周期采样数为10
temp1=ones(1,n);%表示1码
temp0=zeros(1,n);%表示0码
new_dsource=[];
for i=1:length(dsource)
    if dsource(i)==0
    new_dsource=[new_dsource temp0];
    else
    new_dsource=[new_dsource temp1];
    end
end
T=0.10;                                     % 每码元周期
t=0:T/n:T/n*(length(new_dsource)-1);    % 时间轴，new_dsource序号从1开始
到%(length(new_dsource)，而t是从0开始，故要减去1
plot(t,new_dsource)
axis([min(t)-0.01,max(t)+0.01,min(new_dsource)-0.01,max(new_dsource)+0.01])
```

仿真结果：



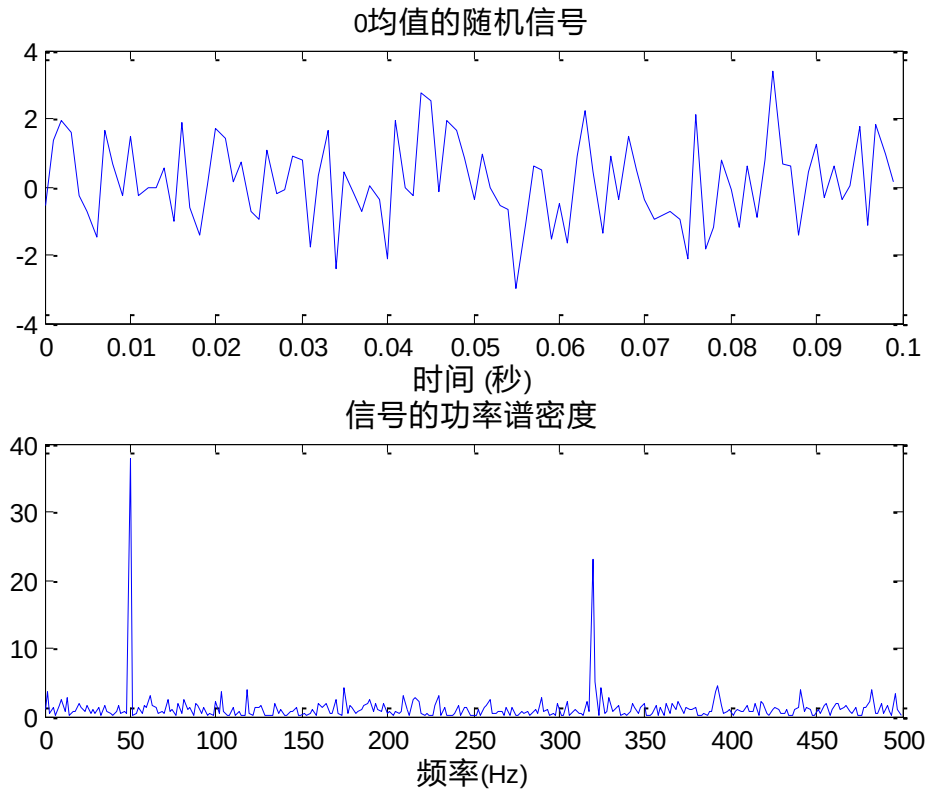
2). 信号的功率谱密度

信号 $f(t)$ 的功率谱密度为：$P(\omega)=\lim_{T\to\infty}\frac{|F_T(\omega)|^2}{T}$，因此可以用如下方法求解信号的功率谱密度。

例题 3-7 求叠加了高斯噪声的正弦信号的功率谱密度. (exa0307_rand.m)

```
clear all
t = 0:0.001:0.6; % 时域信号的时间范围
x = 0.4*sin(2*pi*50*t)+0.4*sin(2*pi*320*t);   % 正弦信号
y = x + randn(size(t));  % 正弦信号+噪声
subplot(2,1,1);
plot(t(1:100),y(1:100));
title('0均值的随机信号')
xlabel('时间 (秒)')
Nf=length(t);
Y = fft(y,Nf);% 求有限长（正弦+噪声）信号的傅里叶变换
Pyy=abs(Y).^2/Nf; % 求傅里叶变换模平方的均值
f = 1000*(0:(Nf-1)/2)/Nf;% 得到频率轴，1000=1/dt，频率区间长度，见第二章定义，这里
%只画出了正半轴，注意区间长度
subplot(2,1,2);
plot(f,Pyy(1:((Nf-1)/2+1)));%注意区间长度
title('信号的功率谱密度');
xlabel('频率(Hz)')
```

仿真结果：

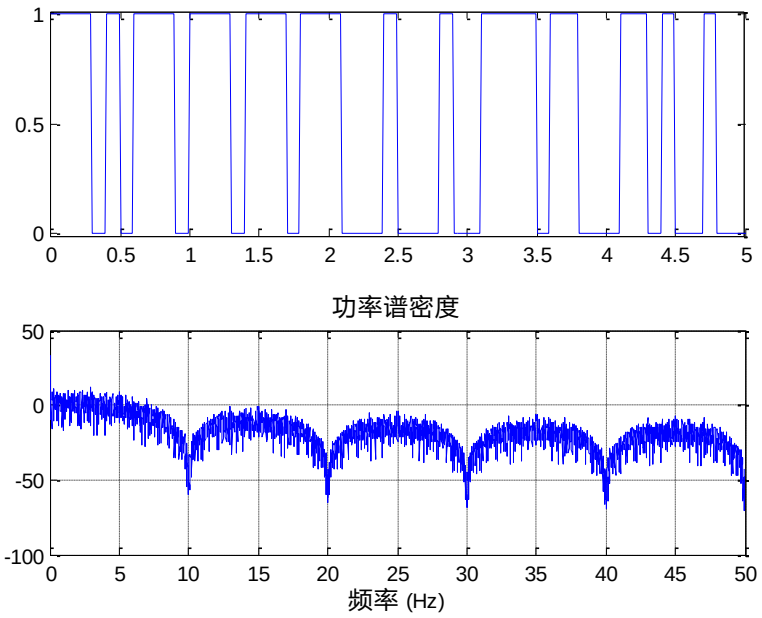


例题 3-8 求单极性 0、1 随机序列的功率谱密度. (exa0307_rand01_power.m)

```
clear all
N=1000;                                  % 0、1序列长度
dsource =(sign(rand(1,N)-0.5+eps)+1)/2; % 生成长度为N的0、1序列
n=10;                                    % 每码元采n=10个样值
temp1=ones(1,n);temp0=zeros(1,n);
new_dsource=[];
for i=1:length(dsource)
    if dsource(i)==0
        new_dsource=[new_dsource temp0];
    else
        new_dsource=[new_dsource temp1];
    end
end
T=0.10;                                  % 码元周期
t=0:T/n:T/n*(length(new_dsource)-1);     % 时间轴
subplot(2,1,1)
plot(t,new_dsource)                      % 画生成的0、1随机序列的波形
axis([min(t)-0.01,max(t)+0.01,min(new_dsource)-0.01,max(new_dsource)+
0.01])

y=new_dsource;
Nf=length(t);
Y = fft(y,Nf);                % 做FFT
```

```
Pyy=abs(Y).^2/Nf;                % 平均功率
dt=t(2)-t(1); df=1/dt;           % 频域延拓周期
f = df*(0:(Nf-1)/2)/Nf;    % 频率轴
subplot(2,1,2);
plot(f,10*log10(Pyy(1:((Nf-1)/2+1))))
title('功率谱密度');
xlabel('频率 (Hz)');grid
```

仿真结果：



3.2 蒙特卡罗算法

蒙特卡罗估计是指通过随机实验估计系统参数值的过程。蒙特卡罗算法的基本思想：由概率论可知，随机实验中实验的结果是无法预测的，只能用统计的方法来描述。故需进行大量的随机实验，如果实验次数为 N ，以 N_A 表示事件 A 发生的次数。若将 A 发生的概率近似为相对频率，定义为 N_A/N 。这样，在相对频率的意义下，事件 A 发生的概率可以通过重复无限多次随机实验来求得，即： $P(A)=\lim\limits_{N\to\infty}\dfrac{N_A}{N}$

在二进制数字通信系统中，若 N 是发送端发送的总码元数， N_A 是差错发生的次数，则总误码率可通过蒙特卡罗算法计算。

例题 3-8 利用蒙特卡罗算法仿真二进制基带通信系统的误码率

假定通信系统满足以下条件：

① 信源输出的数据符号是相互独立和等概的双极性基带信号；

② 发送端没有发送滤波器，接收端没有接收滤波器；

③ 信道是加性高斯白噪声信道。

数字基带信号传输系统模型如图 3-1 所示：

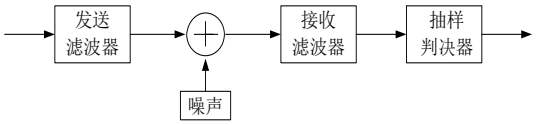


图 3-1 数字基带信号传输系统模型

对于单极性二进制数字系统，抽样判决器输入信号为：

$$r=\begin{cases}\mathrm{A}+n_1, & \text{“1”}\\ 0+n_2, & \text{“0”}\end{cases}$$

A 为判决器输入有用信号电压，n_1，n_2 为信道输入的均值为 0，方差为 σ_n^2 高斯噪声。

当 $P(1)=P(0)=1/2$ 时：最佳判决门限：$V_d^*=\mathrm{A}/2$，误码率：

$$Pe=\frac{1}{2}erfc(\frac{A}{2\sqrt{2}\sigma_n})=\frac{1}{2}erfc(\sqrt{\frac{A^2}{4\times 2\sigma^2{}_n}})=\frac{1}{2}erfc(\sqrt{\frac{r}{4}})$$

抽样判决器输入信噪比：$r=\dfrac{A^2}{2\sigma^2{}_n}$；$\sigma_n=\sqrt{\dfrac{A^2}{2r}}=\dfrac{A}{\sqrt{2r}}$

通信系统的蒙特卡罗仿真模型如图 3-2 所示。编程实现二进制基带通信系统的误码率的蒙特卡罗仿真，并和理论误码率比较。

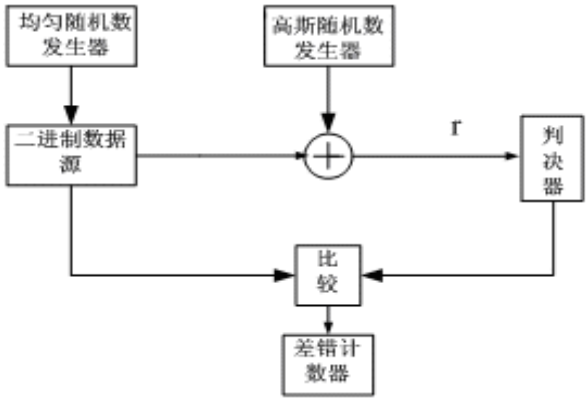
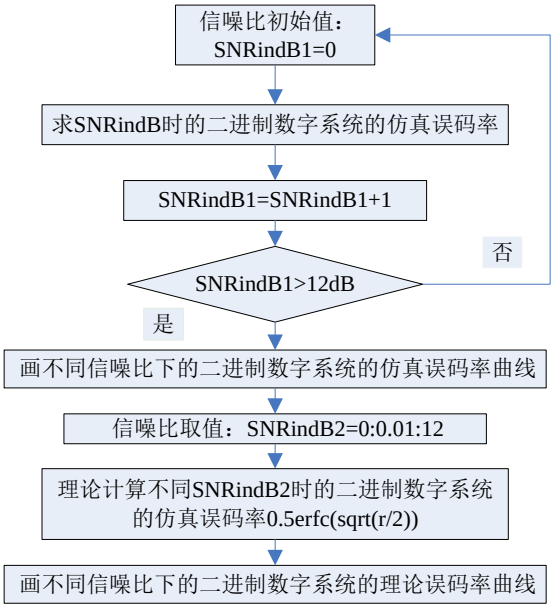


图 3-2 通信系统的蒙特卡罗仿真模型

本实验的核心内容：分别用蒙特卡罗模型仿真双极性数字基带系统在不同信噪比下的误码率曲线，和用理论公式计算双极性数字基带系统在不同信噪比下的误码率曲线，并进行比较。

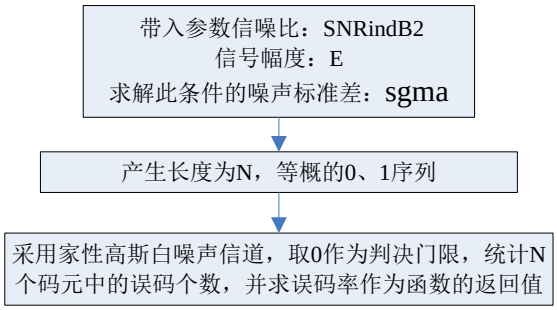
1）仿真流程图：



2）主程序：

```
echo on;
SNRindB1=0:1:12;
SNRindB2=0:0.01:12;
%计算蒙特卡洛仿真误码率
for i=1:length(SNRindB1)
    smld_err_prb(i)= exa0308_Fun_singlePe(SNRindB1(i)); % 调蒙特卡洛仿真误码率子程序
end
semilogy(SNRindB1,smld_err_prb,'r*');              % 画蒙特卡洛仿真误码率曲线
%计算理论误码率
for i=1:length(SNRindB2)
    SNR=10^(SNRindB2(i)/10);                       % 将 dB 换为比值
    theo_err_prb(i)= (1/2)*erfc(sqrt(SNR / 4)); % 理论计算该信噪比下的误码率
end
hold on
semilogy(SNRindB2,theo_err_prb);                   % 画理论误码率曲线
grid on
```

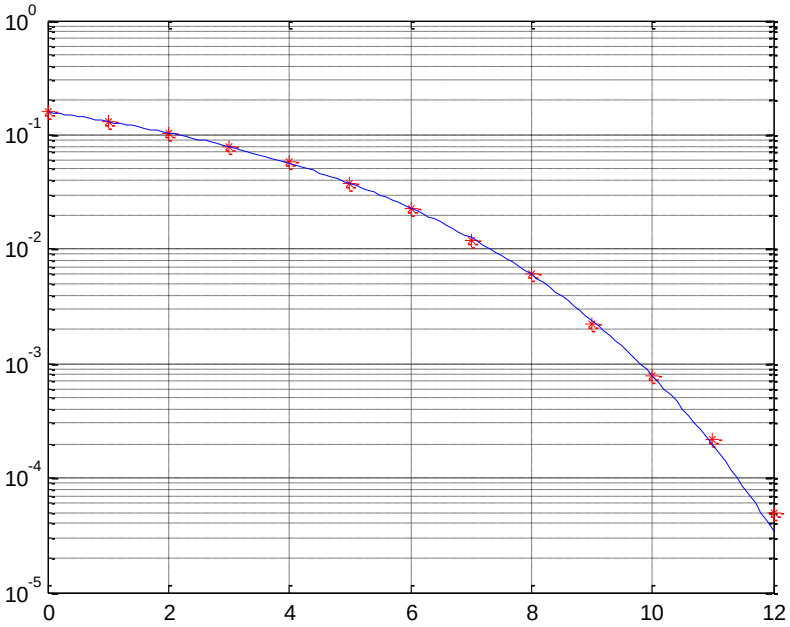
3）蒙特卡洛仿真误码率仿真流程图：




```
function [p]= exa0308_Fun_singlePe(snr_in_dB)%计算误码率
%信噪比与 dB 值的转换
```

```
E=1;    % 信号幅度为1v
SNR=10^(snr_in_dB/10);      % 求信号与噪声的比值
% 产生标准差为sgma，均值为0的N个高斯白噪声
sgma=E/sqrt(SNR*2);      % 噪声的标准差
%二进制序列的产生
N=10000;        %二进制序列的长度
dsource =(sign(rand(1,N)-0.5+eps)+1)/2;
%计算误码率
numoferr=0;             % 误码计数器
for i=1:N
 if(dsource(i)==0)
        r= 0+ sgma*randn;   % 发端发0时的，接收信号
    else
    r= E+ sgma *randn;  % 发端发1时的，接收信号
    end
    if(r<0.5*E)
        decis=0;            % 判决
    else
        decis=1;            % 判决
    end
    if(decis~=dsource(i))   % 与发端序列比较
        numoferr=numoferr+1;% 与发端序列不相同的，误码计数器加1
    end
end
p=numoferr/N;               % 求误码率
```

仿真结果：



第四章 模拟调制 MATLAB 实现

本章目标

- 掌握线性模拟调制信号的波形及产生方法；
- 掌握线性模拟调制信号的频谱特点；
- 掌握线性模拟调制信号的解调方法；
- 掌握线性模拟调制系统的 **MATLAB** 仿真实现。

4.1 模拟调制

- 模拟调制包括幅度调制（DSB，SSB，AM）和相角调制（频率和相位调制）。
- 幅度调制（线性调制）是正弦载波的幅度随着调制信号而改变的调制方案。
- 若调制信号 $m(t)$,频谱为 $M(f)$，带宽为 $B=f_m$，SSB 调制的带宽为 B，DSB 和 AM 调制的带宽为 $2B$。VSB-AM 的带宽在 B-$2B$ 区间内。

4.1.1 AM 调制和相干解调

- AM 调制与 DSB 调制在许多方面十分相似，唯一区别在于 AM 调制用 $[A+m(t)]$ 代替 DSB 的 $m(t)$。
- $s_{AM}(t)=[A+m(t)]\cos 2\pi f_c t$
- $$S_{AM}(f)=\frac{A_c}{2}[\delta(f+f_c)+\delta(f-f_c) +aM(f+f_c)+aM(f-f_c)]$$

调制器模型如图 3-1 所示

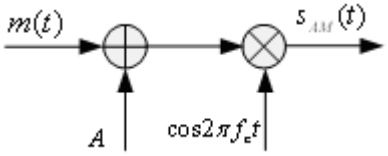


图 3-1 AM 调制器模型

- 假设被调信号 $m(t)$ 是零均值信号，则调制信号的功率为 $\overline{m^2(t)}$
- $P_{AM}=\frac{A^2+\overline{m^2(t)}}{2}=P_c+P_s$， 为调制信号平均功率，$P_s$ 为边带功率，P_c 为载波功率。
- 调制信号中用于发送信息的功率和总功率的比值称为调制效率，$\eta_{AM}=\frac{P_c}{P_c+P_s}=\frac{\overline{m^2(t)}/2}{A^2/2+\overline{m^2(t)}/2}$

- 解调器输入信噪比定义为

$r_i = \frac{S_i}{N_i} = \frac{S_i}{n_0 B_{BPF}}$，$B_{BPF}$ 为理想带通滤波器带宽，在理想信道中，当带通滤波器幅频特性为常数 1 时，$\frac{S_i}{N_i} = \frac{P_s}{n_0 B_{BPF}}$，$n_0 = \frac{P_s}{r_i B_{BPF}}$，即可由解调器输入信噪比计算出信道噪声的单边带功率谱密度，而当系统抽样速率为 f_s 时，产生的高斯白噪声带宽为 $f_s/2$，由此可计算出信道中高斯白噪声的平均功率，即方差 $\sigma_n^2 = n_0 f_s/2$，从而利用第三章的知识可以产生信道中所叠加的高斯白噪声。

4.2 AM 调制解调的 MATLAB 实现

例：调制信号为

$$m(t) = \begin{cases} 1 & , \quad 0 \le t \le t_0/3 \\ -2 & , \quad t_0/3 < t \le 2t_0/3 \\ 0 & , \quad \text{其它} \end{cases}$$

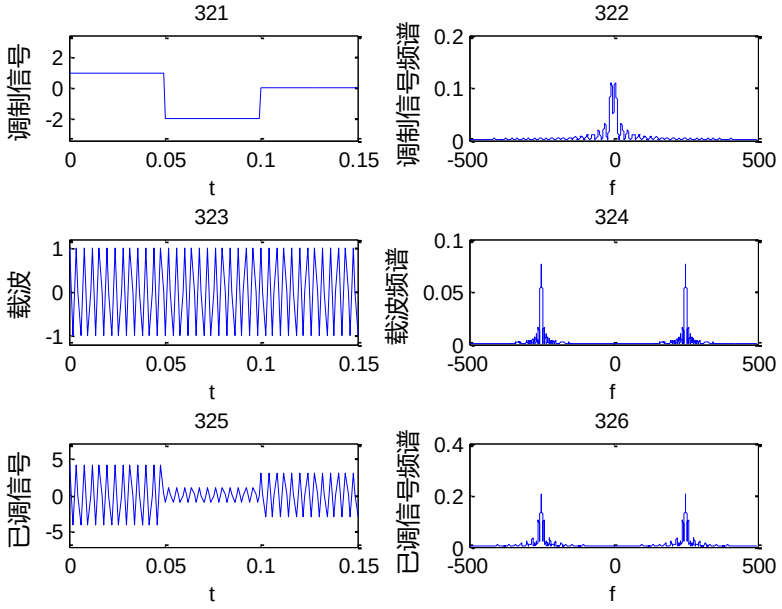
,利用 AM 调制方式调制载波 $c(t) = \cos 2\pi f_c t$，假设 $t_0 = 0.15s$, $f_c = 250Hz$, 直流分量为 3，采用相干方式解调实现 AM 信号的调制解调。

主程序

am.m　　AM 调制解调程序

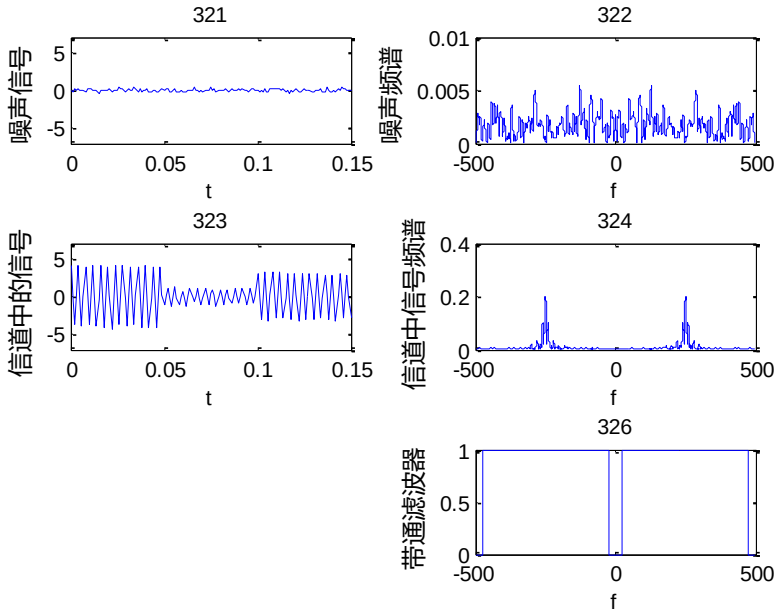
```
% AM调制解调
clear all;
close all;
echo on
%----------------系统仿真参数
 A=3;     %直流分量
fc=250;   %载波频率（Hz）
t0=0.15;%信号时长
snr=25;  %解调器输入信噪比dB
dt=0.001% 系统时域采样间隔
fs=1/dt;%系统采样频率
df = 0.2;   %所需的频率分辨率
t=0:dt:t0;
    Lt=length(t);%仿真过程中，信号长度
snr_lin = 10^(snr/10);  %信噪比
    %-------------画出调制信号波形及频谱
% 产生模拟调制信号
m=[ones(1,t0/(3*dt)),-2*ones(1,t0/(3*dt)),zeros(1,t0/(3*dt)+1)];
L=2*min(m);
R=2*max(abs(m))+A;
```

```
pause%画出调制信号波形及频谱
clf

figure(1)
subplot(321);
plot(t,m(1:length(t)));% 画出调制信号波形
axis([0 t0 -R/2 R/2]);
xlabel('t');
ylabel('调制信号');
subplot(322);
[M,m,df1,f]=T2F(m,dt,df,fs);%求出调制信号频谱
[Bw_eq]=signalband(M,df,t0);%求出信号等效带宽
f_start=fc-Bw_eq;
f_cutoff=fc+Bw_eq;
plot(f,fftshift(abs(M)))% 画出调制信号频谱
xlabel('f');
ylabel('调制信号频谱');
pause %画出载波及频谱
subplot(323);
c=cos(2*pi*fc*t);%载波
plot(t,c);
axis([0 t0 -1.2 1.2]);
xlabel('t');
ylabel('载波');
subplot(324)% 载波频谱
[C,c,df1,f]=T2F(c,dt,df,fs);
plot(f,fftshift(abs(C)))% 画出载波频谱
xlabel('f');
ylabel('载波频谱');
pause% 画已调信号及其频谱
subplot(325)% 画已调信号
u=(A+m(1:Lt)).*c(1:Lt);%已调信号
plot(t,u);% 画出已调信号波形
axis([0 t0 -R R]);
xlabel('t');
ylabel('已调信号');
subplot(326);
[U,u,df1,f]=T2F(u,dt,df,fs);
plot(f,fftshift(abs(U)))% 画出已调信号频谱
xlabel('f');
ylabel('已调信号频谱');
%----------该图为figure(1)--------%
```



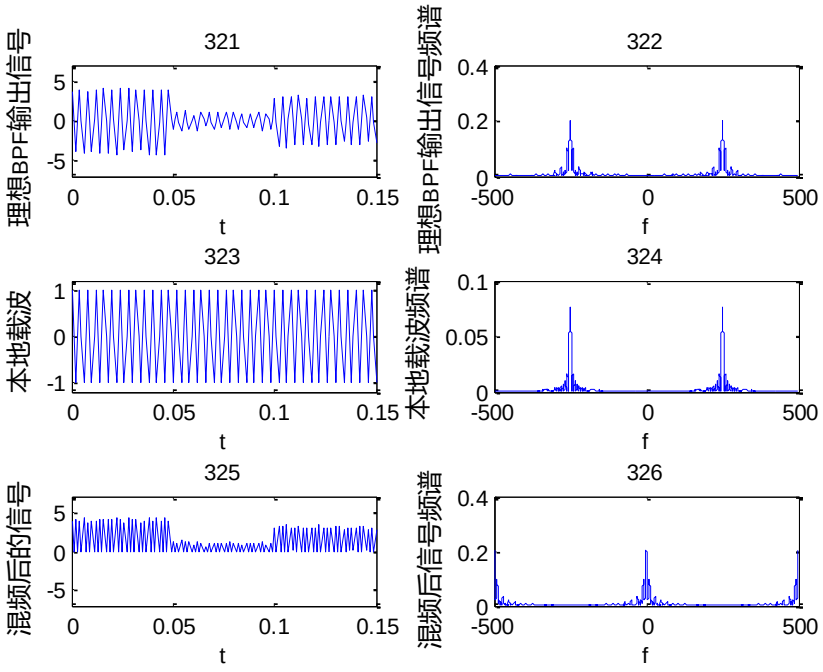

```
%------------将已调信号送入信道
%先根据所给信噪比产生高斯白噪声
signal_power = power_x(u(1:Lt));  %已调信号的平均功率
noise_power=(signal_power*fs)/(snr_lin*(2* Bw_eq));%求出噪声方差（噪声均值为0）
noise_std = sqrt(noise_power);       %噪声标准偏差
noise = noise_std*randn(1,Lt);          %产生噪声
pause%画出信道高斯白噪声波形及频谱，此时，噪声已实现，为确知信号，可求其频谱
figure(2)
subplot(321);
plot(t,noise);% 画出噪声波形
axis([0 t0 -R R]);
xlabel('t');
ylabel('噪声信号');
subplot(322);
[noisef,noise,df1,f]=T2F(noise,dt,df,fs);%噪声频谱
plot(f,fftshift(abs(noisef)))% 画出噪声频谱
xlabel('f');
ylabel('噪声频谱');
pause%画出叠加了噪声的已调信号波形及频谱
sam=u(1:Lt)+noise(1:Lt);%叠加了噪声的已调信号
subplot(323); %画出叠加了噪声的已调信号波形
plot(t,sam);
axis([0 t0 -R R]);
xlabel('t');
ylabel('信道中的信号');
subplot(324);
```

```
[samf,sam,df1,f]=T2F(sam,dt,df,fs);%求出叠加了噪声的已调信号频谱
plot(f,fftshift(abs(samf)))% 画出叠加了噪声的已调信号频谱
xlabel('f');
ylabel('信道中信号频谱');
[H,f]=bp_f(length(sam),f_start,f_cutoff,df1,fs,1);%求带通滤波器
subplot(326);
plot(f,fftshift(abs(H)))% 画出带通滤波器
xlabel('f');
ylabel('带通滤波器');
%----------该图为figure(2)--------%
```



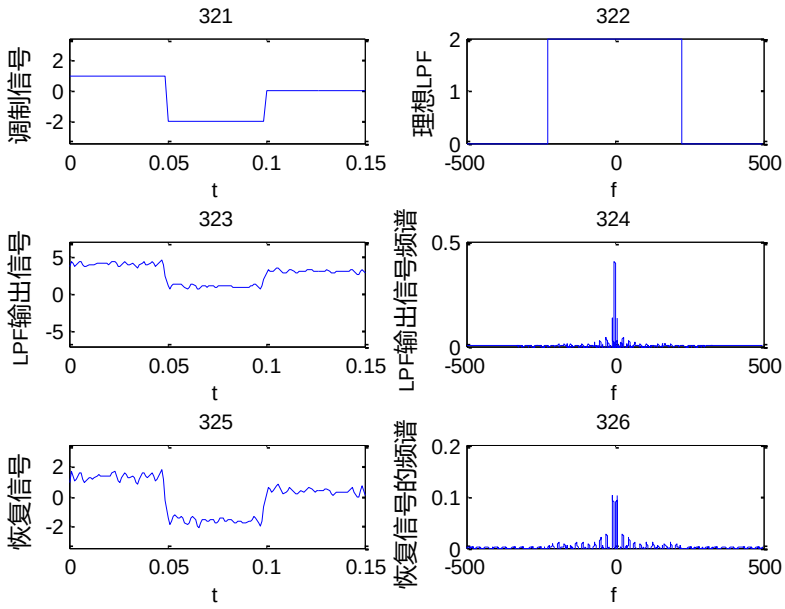

```
%-----------------经过带通滤波器
pause%经过理想带通滤波器后的信号及其频谱
DEM = H.*samf;        %滤波器输出的频谱
[dem]=F2T(DEM,fs);%滤波器的输出波形

figure(3)
subplot(321)%经过理想带通滤波器后的信号波形
plot(t,dem(1:Lt))%画出经过理想带通滤波器后的信号波形
axis([0 t0 -R R]);
xlabel('t');
ylabel('理想BPF输出信号');
[demf,dem,df1,f]=T2F(dem(1:Lt),dt,df,fs);%求经过理想带通滤波器后信号频谱
subplot(322)
plot(f,fftshift(abs(demf)));% 画出经过理想带通滤波器后信号频谱
xlabel('f');
ylabel('理想BPF输出信号频谱');

%---------------和本地载波相乘，即混频
```

```
pause%混频后的信号，先画本地载波及其频谱
subplot(323)
plot(t,c(1:Lt));
axis([0 t0 -1.2 1.2]);
xlabel('t');
ylabel('本地载波');
subplot(324)% 载波频谱
[C,c,df1,f]=T2F(c(1:Lt),dt,df,fs);
plot(f,fftshift(abs(C)))% 画出载波频谱
xlabel('f');
ylabel('本地载波频谱');
pause%再画混频后信号及其频谱
der=dem(1:Lt).*c(1:Lt);%混频
subplot(325)%画出混频后的信号
plot(t,der);
axis([0 t0 -R R]);
xlabel('t');
ylabel('混频后的信号');
subplot(326)
[derf,der,df1,f]=T2F(der,dt,df,fs);%求混频后的信号频谱
plot(f,fftshift(abs(derf)))%画出混频后的信号的频谱
xlabel('f');
ylabel('混频后信号频谱');
%----------该图为figure(3)---------%
```




```
%--------------------经过低通滤波器
```

```
pause%画出理想低通滤波器

figure(4)
[LPF,f]=lp_f(length(der),Bw_eq,df1,fs,2);%求低通滤波器
subplot(322)
plot(f,fftshift(abs(LPF)));% 画出理想低通滤波器
xlabel('f');
ylabel('理想LPF');
pause%混频信号经理想低通滤波器后的频谱及波形
DM = LPF.*derf; %理想低通滤波器输出的频谱

[dm]=F2T(DM,fs);%滤波器的输出波形

subplot(323)
plot(t,dm(1:Lt));%画出经过低通滤波器后的解调出的波形
axis([0 t0 -R R]);
xlabel('t');
ylabel('LPF输出信号');
subplot(324)
[dmf,dm,df1,f]=T2F(dm(1:Lt),dt,df,fs);%求LPF输出信号的频谱
plot(f,fftshift(dmf));%画出LPF输出信号的频谱
xlabel('f');
ylabel('LPF输出信号频谱');
axis([-fs/2 fs/2 0 0.5]);
%--------去除解调信号中的直流分量

%----------该图为 figure(4)--------%
```




```
pause%去除解调信号中的直流分量
```

```
dmd=dm(1:Lt)-mean(dm(1:Lt));
subplot(325)
plot(t,dmd);%画出恢复信号(去除直流分量)
axis([0 t0 -R/2 R/2]);
xlabel('t');
ylabel('恢复信号');
[dmdf,dmd,df1,f]=T2F(dmd,dt,df,fs);%求恢复信号的频谱
subplot(326)
plot(f,fftshift(dmdf));%画出恢复信号的频谱
xlabel('f');
ylabel('恢复信号的频谱');
axis([-fs/2 fs/2 0 0.2]);
subplot(321);
plot(t,m(1:Lt));% 画出调制信号波形
axis([0 t0 -R/2 R/2]);
xlabel('t');
ylabel('调制信号');
```

4.子函数

- 序列的傅立叶变换

```
function [M,m,df]=fftseq(m,ts,df)
%各参数含义与子函数T2F中的完全相同，完成
fs = 1/ts;
if nargin ==2
    n1 =0;
else
    n1 = fs/df;
end
n2 = length(m);
n = 2^(max(nextpow2(n1),nextpow2(n2)));
M = fft(m,n);
m = [m,zeros(1,n-n2)];
df = fs/n;
```

- 计算信号功率

```
function p=power_x(x)
%x:输入信号
%p:返回信号的x功率
p=(norm(x).^2)./length(x);
```

- 信号从频域转换到时域

```
function [m]=F2T(M,fs)

%-----------------------输入参数
```

%M：信号的频谱
%fs:系统采样频率
%--------------------输出(返回)参数
%m:傅里叶逆变换后的信号，注意其长度为2的整数次幂，利用其画波形时，要注意选取m的一部分，选取长度和所给时间序列t的长度要一致，plot(t,m(1:length(t)))，否则会出错。
m = real(ifft(M))*fs;

- 信号从时域转换到频域

function [M,m,df1,f]=T2F(m,ts,df,fs)

%------------------------输入参数

%m:信号
%ts:系统时域采样间隔
%df:所需的频率分辨率
%fs:系统采样频率
%----------------------输出(返回)参数

%M:傅里叶变换后的频谱序列

%m: 输入信号参与过傅里叶变换后对应的序列，需要注意的是，该序列与输入信号m的区别，其长度是不一样的，输入的m长度不一定是2的整数次幂，而傅里叶变换要求输入信号长度为2的整数次幂，故傅里叶变换前需对输入的m信号进行补零，其长度有所增加，故输出参数中的m为补零后的输入信号，其长度与输入参数m不一样，但与M,f长度是一样的，并且，其与时间序列t所对应的序列m(1:length(t))与输入参数中的m是一致的。
%df1:返回的频率分辨率
%f:与M相对应的频率序列
[M,m,df1]=fftseq(m,ts,df);
f = [0:df1:df1*(length(m)-1)] -fs/2;   %频率向量
M=M/fs;

- 带通滤波器

Function[H,f]=bp_f(n,f_start,f_cutoff,df1,fs,p)
%带通滤波器函数   输入设计的滤波器参数，产生带通滤波器频率特性函数H和频率向量f

%------------------------输入参数

%n 带通滤波器的输入信号长度
%f_start   通带起始频率
%f_cutoff   带通滤波器的截止频率
%df1   频率分辨率
%fs   抽样频率
%p   滤波器幅度
%----------------------输出(返回)参数
%H 带通滤波器频率响应
%f 频率向量
%设计滤波器

```
n_cutoff = floor(f_cutoff/df1);
n_start = floor(f_start/df1);
f = [0:df1:df1*(n-1)] -fs/2;    %频率向量
H = zeros(size(f));
H(n_start+1:n_cutoff) = p*ones(1,n_cutoff-n_start);
H(length(f) - n_cutoff+1:length(f)-n_start) = p*ones(1,n_cutoff-n_start);
```

- 低通滤波器

```
function [H,f]=lp_f(n,f_cutoff,df1,fs,p)
%低通滤波器函数    输入设计的滤波器参数，产生低通滤波器频率特性函数H和频率向量f
%------------------------输入参数
%n  低通滤波器的输入信号长度
%f_cutoff   低通滤波器的截止频率
%df1        频率分辨率
%fs         抽样频率
%p          滤波器幅度
%----------------------输出(返回)参数
%H  低通滤波器频率响应
%f  频率向量
n_cutoff = floor(f_cutoff/df1);      %设计滤波器
f = [0:df1:df1*(n-1)] -fs/2;    %频率向量
H = zeros(size(f));
H(1:n_cutoff) = p*ones(1,n_cutoff);
H(length(f) - n_cutoff+1:length(f)) = p*ones(1,n_cutoff);
```

- 计算信号有效带宽

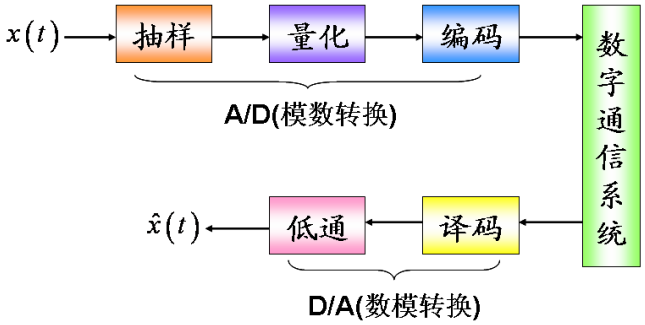
```
function [Bw_eq]=signalband(sf,df,T)
%计算信号等效带宽
%sf：信号频谱
%df:频率分辨率
%T：信号持续时间
sf_max=max(abs(sf));
Bw_eq=sum(abs(sf).^2)*df/T/sf_max.^2
```

第五章 模拟信号的数字传输

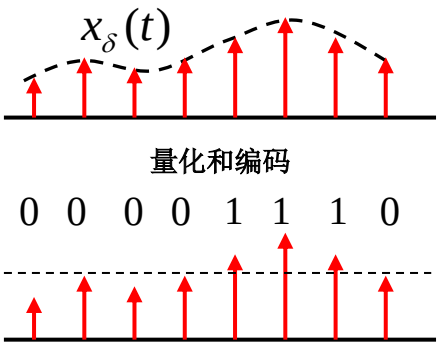
本章目标

- 掌握低通信号抽样定理
- 理解 13 折线 A 率逐次比较型 PCM 编码仿真的思想
- 掌握 13 折线 A 律逐次比较型 PCM 编，译码原理

5.1 脉冲编码调制



抽样：



5.2 低通抽样定理

一频带限制在 $(0, f_H)$ 赫内的时间连续信号 $m(t)$，若以 $f_s \geq 2f_H$ 速率对 $m(t)$ 等间隔 $T_s = 1/f_s \leq 1/2f_H$ 抽样，则 $m(t)$ 将被所得抽样函数 $m_s(t)$ 完全确定。

例1．设低通信号

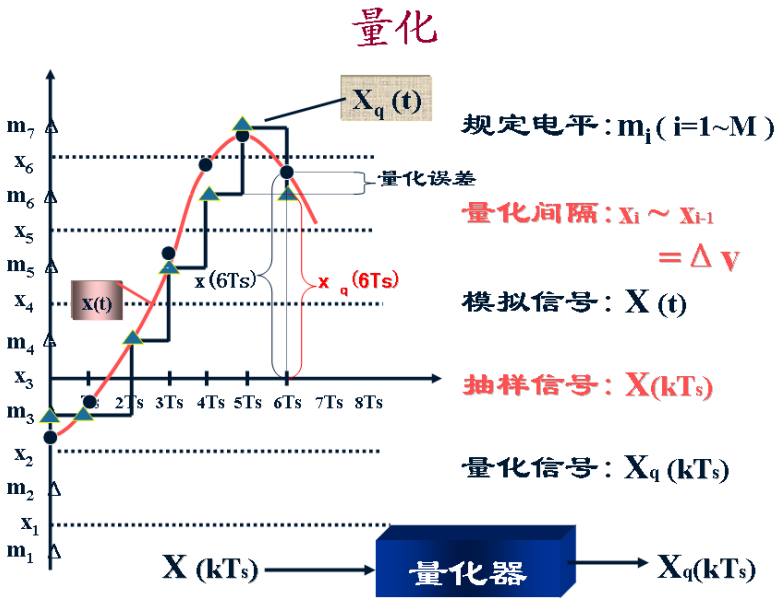
$$x(t) = 0.1\cos(0.15\pi t) + 1.5\sin 2.5\pi t + 0.5\cos 4\pi t$$

(1)画出该低通信号的波形
(2)画出抽样速率为fs=4Hz的抽样序列
(3)抽样序列恢复出原始信号

```
%低通抽样定理
clear all;
close all;
dt=0.01;
```

```
t=0:dt:10;
xt=0.1*cos(0.15*pi*t)+1.5*sin(2.5*pi*t)+0.5*cos(4*pi*t);
[f,xf]=T2F(t,xt);
%抽样信号，抽样频率为4Hz
fs=4;
sdt=1/fs;
t1=0:sdt:10;
st=0.1*cos(0.15*pi*t1)+1.5*sin(2.5*pi*t1)+ 0.5*cos(4*pi*t);;
[f1,sf]=T2F(t1,st);
%恢复原始信号
t2=-50:dt:50;
gt=sinc(fs*t2);
stt=sigexpand(st,sdt/dt);
xt_t=conv(stt,gt);
figure(1);
subplot(311);
plot(t,xt);
title('原始信号');
subplot(312);
plot(t1,st);
title('抽样信号');
subplot(313);
t3=-50:dt:60+sdt-dt;
plot(t3,xt_t);
title('抽样信号恢复');
axis([0 10 -4 4]);
```

5.3 均匀量化原理



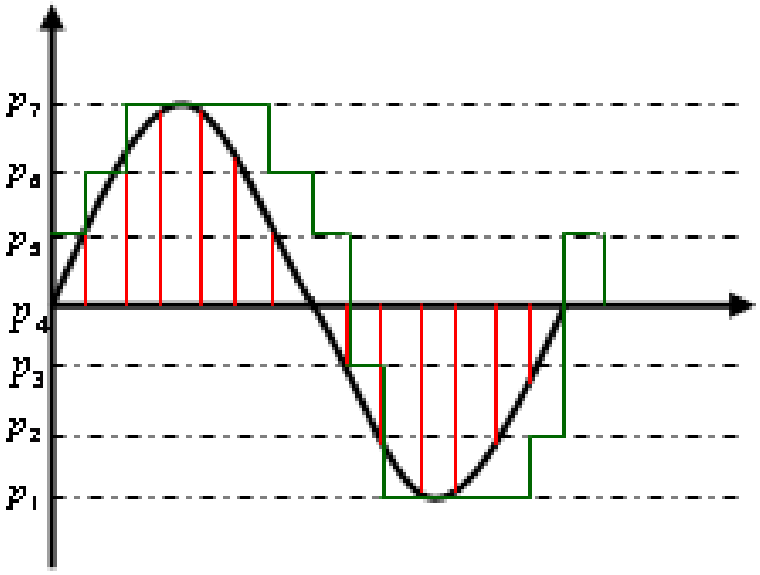
例2．均匀量化程序

输入参数：f—待量化的值；V—从+V到－V量化；L—量化的级数。

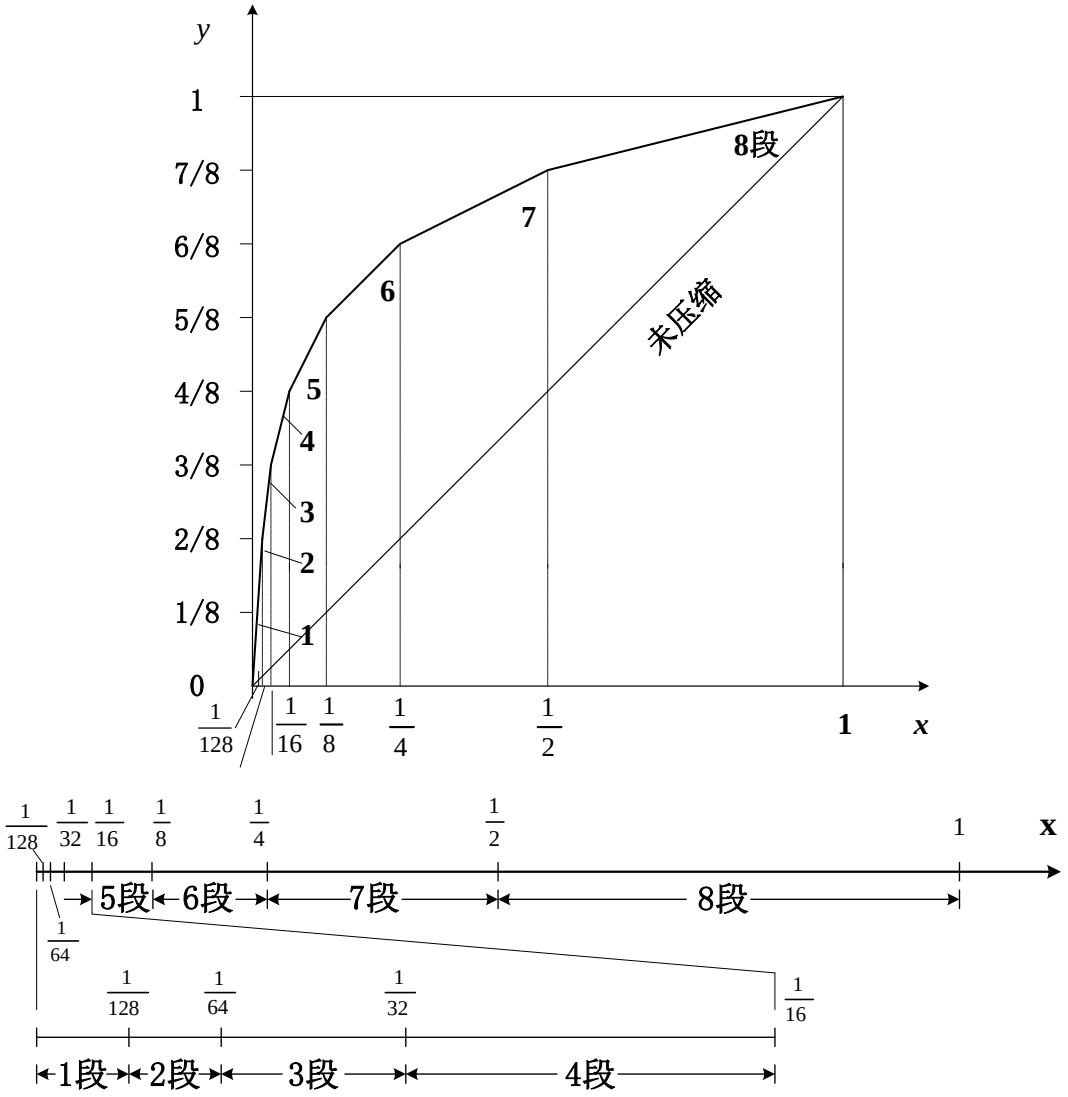
```
function h=junyun(f,V,L)
n=length(f);%抽样序列的长度
t=2*V/L;     %量化区间的宽度
p=zeros(1,L+1);
for i=1:L+1
   p(i)=-V+(i-1)*t;
end            %计算量化电平值
for i=1:n
    if f(i)>V
        h(i)=V;
    end
    if f(i)<=-V
        h(i)=-V;
   end        %处理过载情况
flag=0;
   for j=2:L/2+1
       if flag==0
            if f(i)<p(j)
                h(i)=p(j-1);
                flag=1;
             end
       end
    end           %处理小于0的抽样值
for j=L/2+2:L+1
        if(flag==0)
           if f(i)<p(j)
               h(i)=p(j);
               flag=1;
           end
        end
     end          %处理大于0的抽样值
end
```

均匀量化程序调用

```
x=[0:0.004:4*pi];
y=sin(x);
w=junyun(y,1,32);
plot(x,y,x,w);
x=[0:0.004:4*pi];
y=sin(x);
w=junyun(y,1,8);
plot(x,y,x,w);
```


p_7
p_6
p_5
p_4
p_3
p_2
p_1

5.4 非均匀量化


y
1
7/8
6/8
5/8
4/8
3/8
2/8
1/8
0
8段
7
6
5
4
3
2
1
未压缩
1/128
1/16
1/8
1/4
1/2
1
x
1/128
1/32
1/16
1/8
1/4
1/2
1
x
5段
6段
7段
8段
1/64
1/128
1/64
1/32
1/16
1段
2段
3段
4段

信号小时，Δ小，信号大时，Δ大。一般语音信号，信号幅度小出现的概率大，信号幅度大出现的概率小。通过非均匀量化，使得平均信噪比增大。

2.回忆 13 折线 A 律 PCM 的非线性编码方法。

极性码　　段落码　　段内码

C1　　C2C3C4　　C5C6C7C8

- **将量化区间［a,b］分为 4096 个小段**
- **正半轴 2048 个小段，负半轴 2048 个小段**
- **每个小段用Δ表示**

段落编码	区间范围/Δ	量化间隔/Δ	量化区间/Δ	译码器量化输出/Δ	PCM编码
000	[0, 16)	1	[0, 1)	1	1 000 0000
			[1, 2)	2	1 000 0001
			[2, 3)	3	1 000 0010
			---	---	---
			[15, 16)	15	1 000 1111
001	[16, 32)	1	[16, 17)	16	1 001 0000
			[17, 18)	17	1 001 0001
			[18, 19)	18	1 001 0010
			---	---	---
			[31, 32)	31	1 001 1111
010	[32, 64)	2	[32, 34)	33	1 010 0000
			[34, 36)	35	1 010 0001
			[36, 38)	37	1 010 0010
			---	---	---
			[62, 64)	63	1 010 1111
011	[64, 128)	4	[64, 68)	66	1 011 0000
			[68, 72)	70	1 011 0001
			[72, 76)	74	1 011 0010
			---	---	---
			[124, 128)	126	1 011 1111
100	[128, 256)	8	[128, 136)	134	1 100 0000
			[136, 144)	140	1 100 0001
			[144, 152)	148	1 100 0010
			---	---	---
			[248, 256)	252	1 100 1111
101	[256, 512)	16	[256, 272)	264	1 101 0000
			[272, 288)	280	1 101 0001
			[288, 304)	296	1 101 0010
			---	---	---
			[496, 512)	504	1 101 1111
110	[512, 1024)	32	[512, 544)	528	1 110 0000
			[544, 576)	560	1 110 0001
			[576, 608)	592	1 110 0010
			---	---	---
			[992, 1024)	1008	1 110 1111
111	[1024, 2048)	64	[1024, 1088)	1056	1 111 0000
			[1088, 1152)	1120	1 111 0001
			[1152, 1216)	1184	1 111 0010
			---	---	---
			[1984, 2048)	2016	1 111 1111

例 3．采用 13 折线 A 律编译码电路，设最小量化级为 1Δ，已知抽样脉冲值为＋635Δ
1）试求此时编码器输出码组，并计算接收端译码后的量化误差；
2）写出对应七位码（不包含极性码）的均匀量化 11 位码

```
%输入 x 参数为 0－2048 的样值，输出 out 为 8 位二进制码
function [out]=pcm_encode(x)
n=length(x);
for i=1:n
%编写段落码
    if x(i)>0
        out(i,1)=1;
    else
        out(i,1)=0;
    end
%编写断内码计算量化间隔和量化起始电平
    if abs(x(i))>0&abs(x(i))<32
%段落码
out(i,2)=0;out(i,3)=0;out(i,4)=0;
%量化间隔
step=2;
%起始电平
st=0;
    elseif 32<=abs(x(i))&abs(x(i))<64
out(i,2)=0;out(i,3)=0;out(i,4)=1;step=2;st=32;
    elseif 64<=abs(x(i))&abs(x(i))<128
out(i,2)=0;out(i,3)=0;out(i,4)=1;step=4;st=64;
    elseif 128<=abs(x(i))&abs(x(i))<256
out(i,2)=0;out(i,3)=1;out(i,4)=1;step=8;st=128;
    elseif 256<=abs(x(i))&abs(x(i))<512
out(i,2)=1;out(i,3)=0;out(i,4)=0;step=16;st=256;
    elseif 512<=abs(x(i))&abs(x(i))<1024
out(i,2)=1;out(i,3)=0;out(i,4)=1;step=32;st=512;
    elseif 1024<=abs(x(i))&abs(x(i))<2048
out(i,2)=1;out(i,3)=1;out(i,4)=0;step=64;st=1024;
    elseif 2048<=abs(x(i))&abs(x(i))<4096
out(i,2)=1;out(i,3)=1;out(i,4)=1;step=128;st=2048;
    else
out(i,2)=1;out(i,3)=1;out(i,4)=1;step=128;st=2048;
    end
    if(abs(x(i))>=4096)
%处理过载现象
        out(i,2:8)=[1 1 1 1 1 1 1];
    else
%计算段落码
```

```
            tmp=floor((abs(x(i))-st)/step);
%十进制转二进制数，%如果不减48，最后4位是ASCII数
          % 49 48 48 48
            t=dec2bin(tmp,4)-48;
            out(i,5:8)=t(1:4);
      end
end
out=reshape(out',1,8*n);
```

13 折线译码函数：

```
％输入 in 为 8 位二进制码，（－v，+v）为量化区间
function [out]=pcm_decode(in,v)
n=length(in);
in=reshape(in',8,n/8)';
slot(1)=0;
slot(2)=32;
slot(3)=64;
slot(4)=128;
slot(5)=256;
slot(6)=512;
slot(7)=1024;
slot(8)=2048;
step(1)=2;
step(2)=2;
step(3)=4;
step(4)=8;
step(5)=16;
step(6)=32;
step(7)=64;
step(8)=128;
for i=1:n/8
      ss=2*in(i,1)-1;
      tmp=in(i,2)*4+in(i,3)*2+in(i,4)+1;
      st=slot(tmp);
dt=(in(i,5)*8+in(i,6)*4+in(i,7)*2+in(i,8))*step(tmp)+0.5*step(tmp);
      out(i)=ss*(st+dt)/4096*v;
end
```

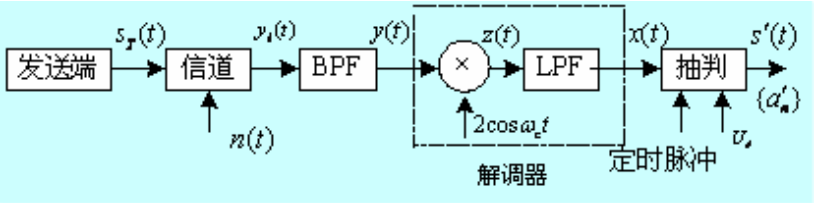
第六章　数字频带传输系统

本章目标

- 掌握数字频带传输系统调制解调的仿真过程
- 掌握数字频带传输系统误码率仿真分析方法

6.1 数字频带传输原理

本章以 2PSK 为例，仿真说明数字频带传输的整个过程



假定：信道噪声为加性高斯白噪声，其均值为 0、方差为σ^2；发射端发送的 2PSK 信号为：

$$s_T(t)=\begin{cases}A\cos\omega_c t, & \text{发 “1”}\\ -A\cos\omega_c t, & \text{发 “0”}\end{cases}$$

经信道传输，接收端输入信号为：

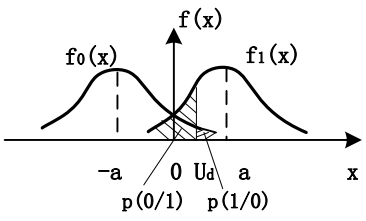
$$y_d(t)=s(t)+n(t)$$

经带通滤波器输出：

$$\begin{aligned}y(t)&=s(t)+n_i(t)\\&=\begin{cases}a\cos\omega_c t+n_c(t)\cos\omega_c t-n_s(t)\sin\omega_c t, & \text{发 “1”}\\ -a\cos\omega_c t+n_c(t)\cos\omega_c t-n_s(t)\sin\omega_c t, & \text{发 “0”}\end{cases}\end{aligned}$$

相干解调后，得

$$x(t)=\begin{cases}a+n_{c1}(t), & \text{发 “1”}\\ -a+n_{c2}(t), & \text{发 “0”}\end{cases}$$



当 1，0 独立等概出现时，2PSK 系统的最佳判决门限电平$U_d^*=0$。故判决规则为在取样时刻的判决值大于 0，判 1，小于 0，判 0。

6.2 信道加性高斯白噪声功率的讨论

信道加性高斯白噪声功率由于其均值为 **0**，故其方差 $\sigma^2 = n_0 f_s / 2$ 即为其平均功率，其中 n_0 为高斯白噪声单边带功率谱密度； f_s 为系统的采样速率，在图 **1** 中，信道噪声的功率谱密度图可以看出，当 f_s **=16Hz** 时，产生的高斯噪声带宽为 **8Hz**；在图 **2** 中，当 f_s **=32Hz** 时，产生的高斯噪声带宽为 **16Hz**；信道中的加性高斯白噪声通过理想带通滤波器后，输出的窄带噪声即为相干解调器输入噪声 N_i，其功率谱密度不变，仍为 n_0。可以看出，

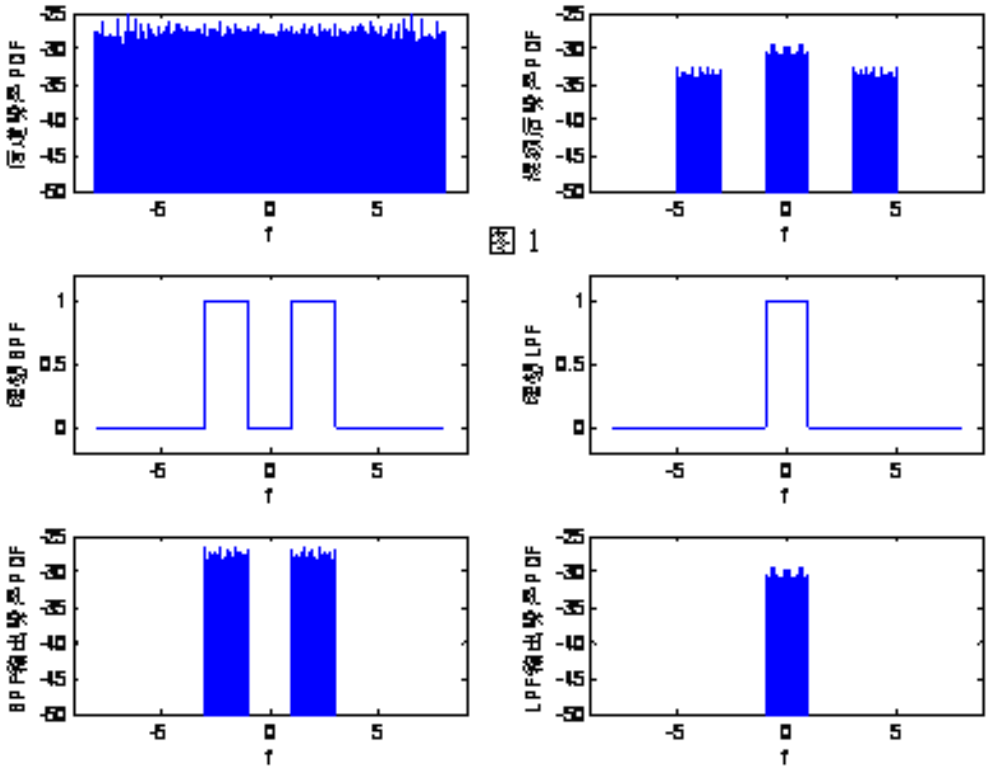


图 1

N_i 与信道加性高斯白噪声功率 σ^2 之间有一定的关系，其共同点是其功率谱密度相同，从图 **1** 和图 **2** 也可以观察出来。因此，在理想通信系统中，利用已给解调器输入信噪比及已调信号功率和带宽，可以计算出 n_0，从而算出信道加性高斯白噪声的方差，由于其均值为 **0**，故该方差为其平均功率，利用它可以生成信道加性高斯白噪声。转换关系为：

$$r = \frac{S_i}{N_i} = \frac{S_i}{n_0 B_{BPF}}, \quad n_0 = \frac{S_i}{r \bullet 2R_B},$$

因此 $\sigma^2 = \dfrac{S_i \bullet f_s}{2 \bullet r \bullet 2R_B}$， $\sigma = \sqrt{\dfrac{S_i \bullet f_s}{2 \bullet r \bullet 2R_B}} = \sqrt{\dfrac{S_i \bullet f_s}{2r \bullet B_{BPF}}}$

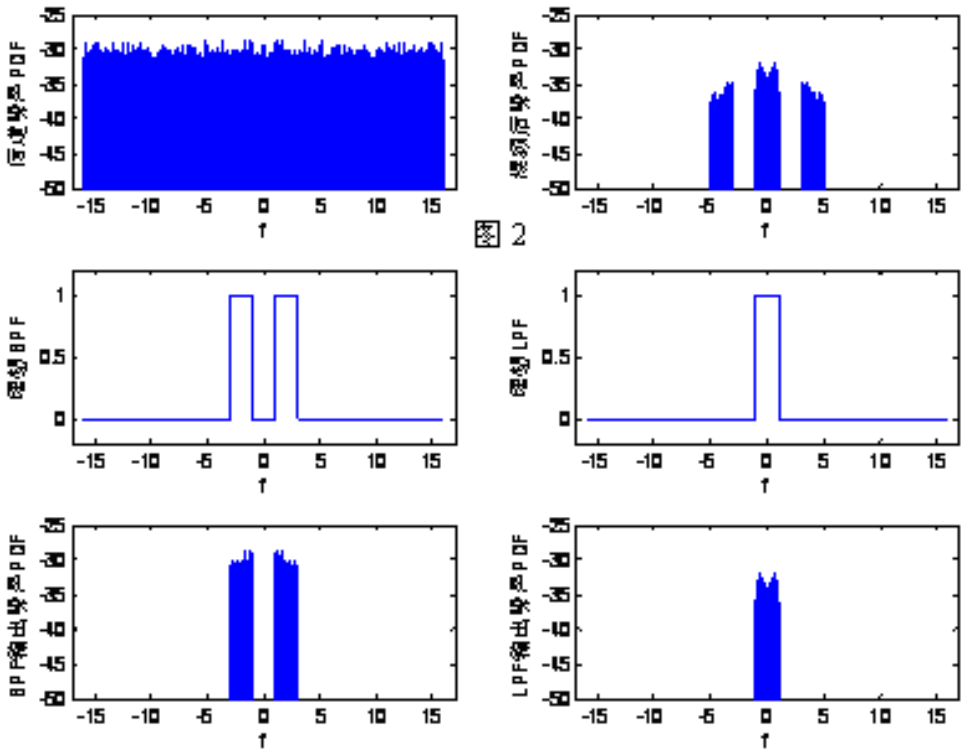


图 2

所以，产生高斯白噪声的程序为：

signal_power=power_x(s_2psk(1:Lt));%求出已调信号%s_2psk平均功率,Lt为已调信号长度。noise_power=(signal_power*fc*N_sample)/(snr_lin*2*(B2-B1));%求出%噪声方差（噪声均值为0），fc 为载波频率；N_sample 为每个码元的的采样点数，二者的乘积为系统采样频率；snr_lin为信噪比。B2-B1为已调信号带宽。

noise_std=sqrt(noise_power);%求出噪声均方差

noise=noise_std.*randn(1:Lt);%以噪声均方差作为幅度产生高斯白噪声

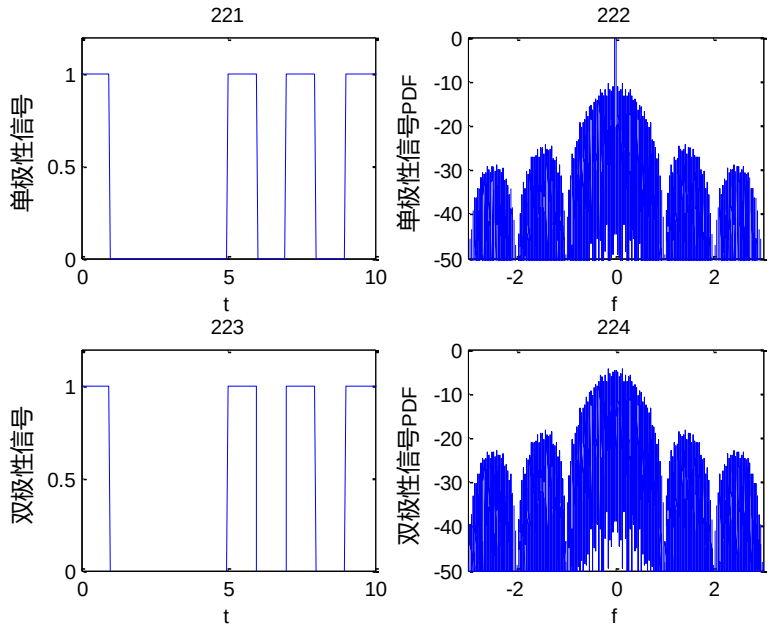
此外，图 1 和图 2 分别给出了噪声在调制信号带宽为 1Hz，已调信号带宽为 2Hz 的理想通信系统中传输其功率变化的过程，便于大家参考。

6.3 仿真分析

主程序1：调制解调过程仿真

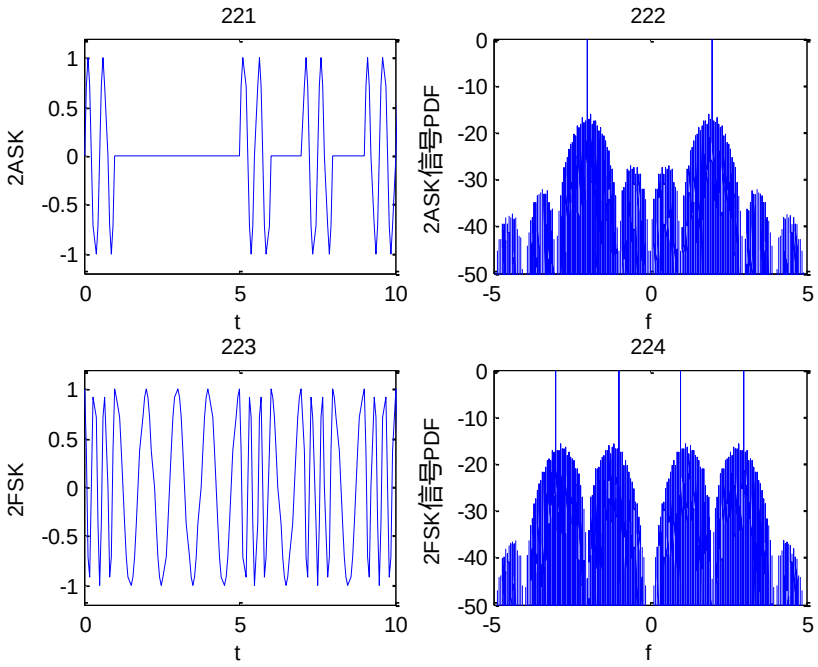
```
clear all;
close all;
echo on
%-------------------系统仿真参数
A=1;     %载波振幅
fc=2;    %载波频率（Hz）
snr=5;   %信噪比dB
N_sample=8;% 基带信号中每个码元的的采样点数

N=10000;  %  码元数
Ts=1;    % 码元宽度
df=0.01%频率分辨率
```

```
B=1/Ts;
f_start=fc-B;
f_cutoff=fc+B;
fs=fc*N_sample%系统采样频率，即考虑载波后一个码元内的采样点数。
ts=Ts/fs; % 系统采样间隔
t=0:ts:N*Ts-ts;
Lt=length(t);
%----------画出调制信号波形及功率谱
% 产生二进制信源
d=sign(randn(1,N));
dd=sigexpand((d+1)/2,fc*N_sample);
gt=ones(1,fc*N_sample); % NRZ波形
d_NRZ=conv(dd,gt);
d_NRZ1=d_NRZ(1:Lt);
pause%画出单极性NRZ波形及其功率谱
figure(1)
subplot(221);
plot(t,d_NRZ1);% 画出单极性NRZ信号波形
axis([0 10 0 1.2]);
xlabel('t');
ylabel('单极性信号');
subplot(222);
[d_NRZ1f,d_NRZ1,df1,f]=T2F(d_NRZ1,ts,df,fs);%求出单极性NRZ信号功率谱
plot(f,10*log10(abs(fftshift(d_NRZ1f).^2/length(f))));% 画出单极性NRZ信号功率
谱
 axis([-3*B 3*B -50 0]);
xlabel('f');
ylabel('单极性信号PDF');
pause%画出双极性NRZ波形及其功率谱
d_sjx=2*d_NRZ-1;%生成双极性NRZ信号
d_sjx1=d_sjx(1:Lt);
subplot(223);
plot(t,d_sjx1);% 画出双极性NRZ信号波形
axis([0 10 0 1.2]);
xlabel('t');
ylabel('双极性信号');
subplot(224);
[d_sjx1f,d_sjx1,df1,f]=T2F(d_sjx1,ts,df,fs);%求出双极性NRZ信号功率谱
plot(f,10*log10(abs(fftshift(d_sjx1f).^2/length(f))));% 画出双极性NRZ信号功率
谱
axis([-3*B 3*B -50 0]);
xlabel('f');
ylabel('双极性信号PDF');
%-------这个图为figure(1)--------%
```



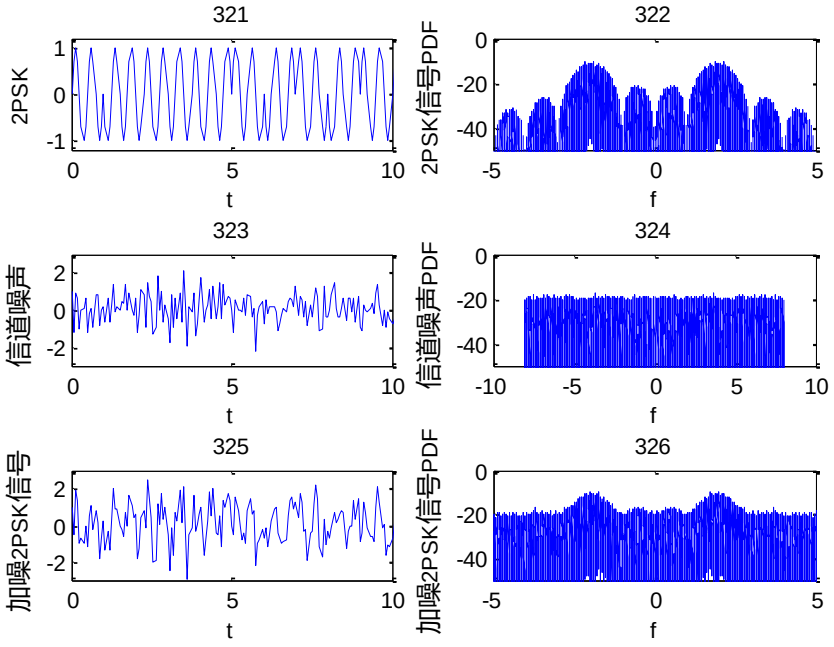

```
%---------画出数字频带信号及其功率谱
% 对数字基带信号进行2ASK调制
ht=A*sin(2*pi*fc*t);%载波
s_2ask=d_NRZ(1:Lt).*ht;%生成已调信号2ASK
pause%画出已调信号2ASK及其功率谱
figure(2)
subplot(221);
plot(t,s_2ask);%画出2ASK信号
axis([0 10 -1.2 1.2]);
xlabel('t');
ylabel('2ASK');
%求2ASK信号功率谱
[s_2askf,s_2ask,df1,f]=T2F(s_2ask,ts,df,fs);%求出单极性NRZ信号功率谱
subplot(222);
plot(f,10*log10(abs(fftshift(s_2askf).^2/length(f))));% 画出单极性NRZ信号功率
谱
axis([-fc-3*B fc+3*B -50 0]);
xlabel('f');
ylabel('2ASK信号PDF');
% 对数字基带信号进行2FSK调制
s_2fsk=A*cos(2*pi*fc*t+2*pi*d_sjx(1:Lt).*t);%生成2FSK信号
 pause%画出已调信号2FSK及其功率谱
subplot(223)
plot(t,s_2fsk);%画出2FSK波形
axis([0 10 -1.2 1.2]);
xlabel('t');
ylabel('2FSK');
subplot(224)
```

```
[s_2fskf,s_2fsk,df1,f]=T2F(s_2fsk,ts,df,fs);%求2FSK信号功率谱
plot(f,10*log10(abs(fftshift(s_2fskf).^2/length(f))));% 画出2FSK功率谱
axis([-fc-3*B fc+3*B -50 0]);
xlabel('f');
ylabel('2FSK信号PDF');
%-------这个图为figure(2)--------%
```

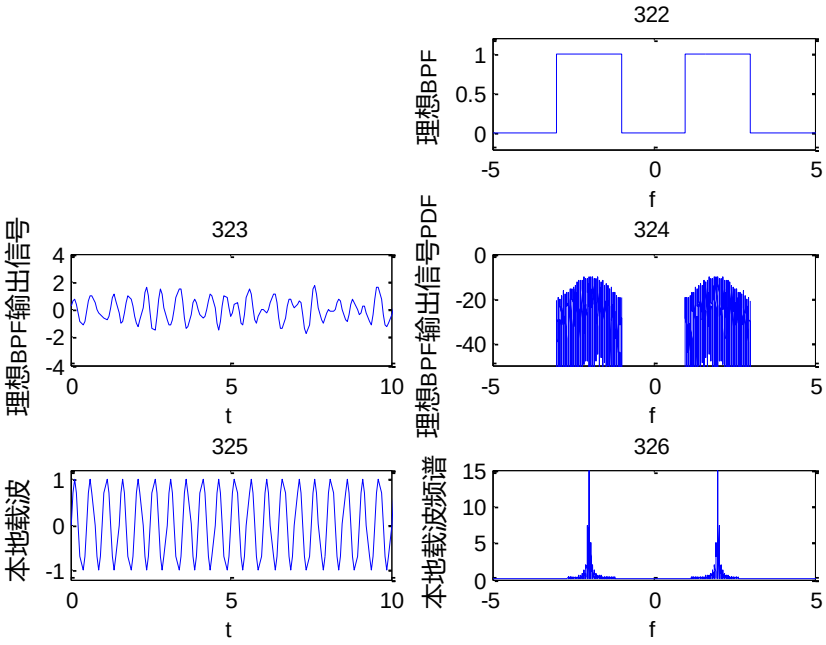



```
% 对数字基带信号进行2PSK调制
s_2psk=d_sjx(1:Lt).*ht;%生成2PSK信号
pause%画出已调信号2PSK及其功率谱
figure(3)
subplot(321)
plot(t,s_2psk);%画出2PSK波形
axis([0 10 -1.2 1.2]);
xlabel('t');
ylabel('2PSK');
subplot(322)
[s_2pskf,s_2psk,df1,f]=T2F(s_2psk,ts,df,fs);%求2PSK信号功率谱
plot(f,10*log10(abs(fftshift(s_2pskf).^2/length(f))));% 画出2PSK信号功率谱
axis([-fc-3*B fc+3*B -50 0]);
xlabel('f');
ylabel('2PSK信号PDF');
%-----------将2PSK信号送入信道进行传输，先生成信道加性高斯白噪声噪声
snr_lin=10^(snr/10); %换算成倍数
signal_power=power_x(s_2psk(1:Lt));%求出信号平均功率
noise_power=(signal_power*fs)/(snr_lin*2*(2*B));%求出噪声方差（噪声均值为0）
noise_std=sqrt(noise_power);%求出噪声均方差
noise=noise_std.*randn(1,Lt);%以噪声均方差作为幅度产生高斯白噪声
```

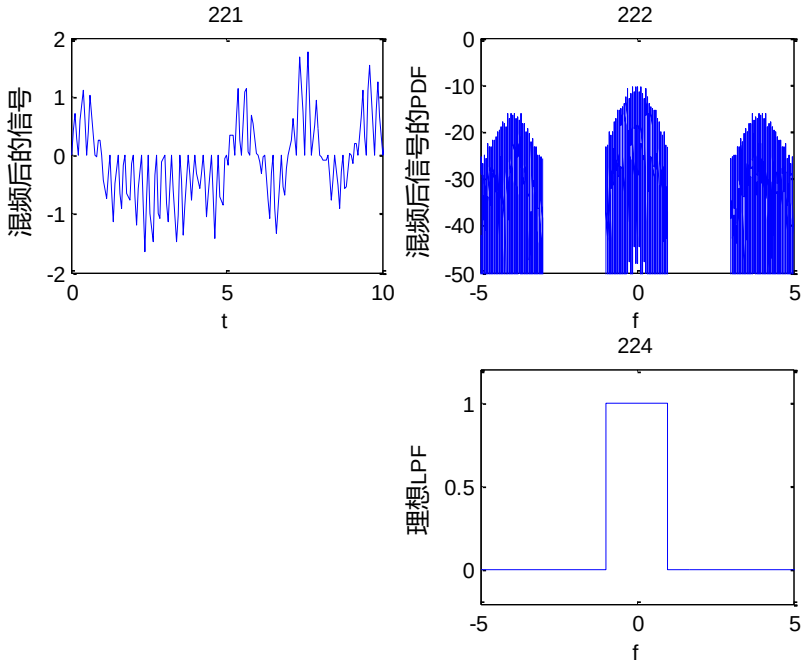
```
%----------------将已调信号送入信道
pause%画出信道中的高斯白噪声及其功率谱
subplot(323)
plot(t,noise(1:Lt));%画出噪声
xlabel('t');
ylabel('信道噪声');
axis([0 10 -3 3]);
[noisef,noise,df1,f]=T2F(noise,ts,df,fs);%求信道噪声功率谱
subplot(324)
plot(f,10*log10(abs(fftshift(noisef).^2/length(f))));% 画出信道噪声功率谱
axis([-fs/2-2 fs/2+2 -50 0]);
xlabel('f');
ylabel('信道噪声PDF');
r=s_2psk(1:Lt)+noise(1:Lt);%叠加了噪声的已调信号，相当于将已调信号送入理想信道
pause%画出加噪后的已调信号2PSK及其功率谱
subplot(325)
plot(t,r);%画出加噪后的已调信号2PSK
xlabel('t');
ylabel('加噪2PSK信号');
axis([0 10 -3 3]);
[rf,r,df1,f]=T2F(r,ts,df,fs);%求加噪后的已调信号2PSK功率谱
subplot(326)%画出加噪后已调信号的功率谱
plot(f,10*log10(abs(fftshift(rf).^2/length(f))));% 画出已调信号2PSK功率谱
axis([-fc-3*B fc+3*B -50 0]);
xlabel('f');
ylabel('加噪2PSK信号PDF');
%-------这个图为figure(3)--------%
```



```
%-----在接收端准备进行解调，先通过带通滤波器
pause%画出带通滤波器
[H,f]=bp_f(length(rf),f_start,f_cutoff,df1,fs,1);%经过理想带通滤波器
figure(4)
subplot(322)
plot(f,abs(fftshift(H)));% 画出理想带通滤波器
axis([-fc-3*B fc+3*B -0.2 1.2]);
xlabel('f');
ylabel('理想BPF');
DEM = H.*rf;       %滤波器输出的频谱
[dem]=F2T(DEM,fs);%滤波器的输出波形
dem1=dem(1:Lt)
pause%经过理想带通滤波器后的信号波形及功率谱
subplot(323)%经过理想带通滤波器后的信号波形
plot(t,dem1)%画出经过理想带通滤波器后的信号波形
axis([0 10 -4 4]);
xlabel('t');
ylabel('理想BPF输出信号');
[demf1,dem1,df1,f]=T2F(dem1,ts,df,fs);%求经过理想带通滤波器后信号功率谱
subplot(324)
plot(f,10*log10(abs(fftshift(demf1).^2/length(f))));% 画出经过理想带通滤波器后
信号功率谱
axis([-fc-3*B fc+3*B -50 0]);
xlabel('f');
ylabel('理想BPF输出信号PDF');
%-----进行相干解调，先和本地载波相乘,即混频
subplot(325)%画出同频同相的本地载波
plot(t,ht);
axis([0 10 -1.2 1.2]);
xlabel('t');
ylabel('本地载波');
subplot(326)% 本地载波频谱
[htf,ht,df1,f]=T2F(ht,ts,df,fs);
plot(f,fftshift(abs(htf)))% 画出载波频谱
axis([-fc-3*B fc+3*B 0 15]);
xlabel('f');
ylabel('本地载波频谱');
%-------这个图为figure(4)--------%
```

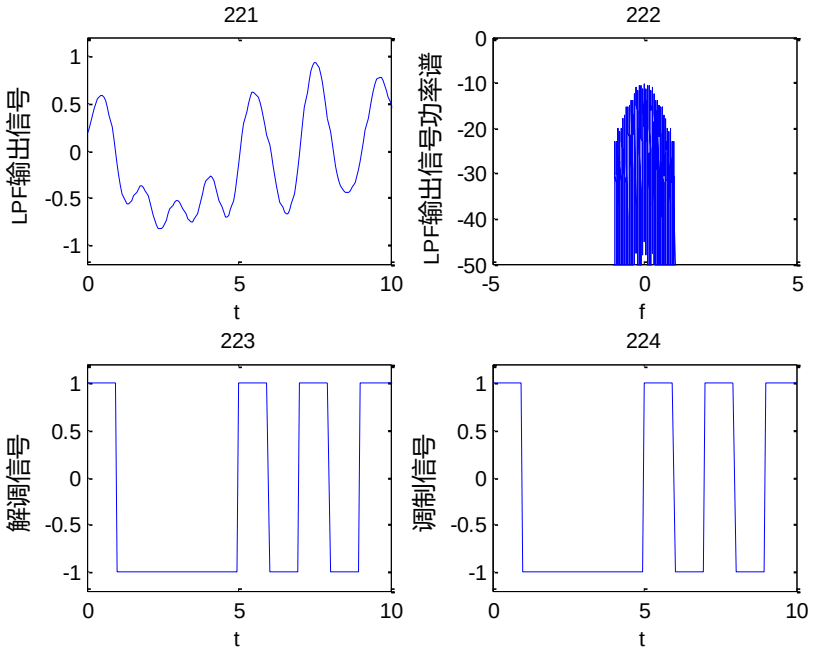



```
pause%画出混频后的信号及功率谱
figure(5)
der=deml(1:Lt).*ht(1:Lt);%和本地载波相乘，即混频
subplot(221)%画出混频后的波形
plot(t,der);
axis([0 10 -2 2]);
xlabel('t');
ylabel('混频后的信号');
[derf,der,df1,f]=T2F(der,ts,df,fs);%求混频后信号的功率谱
subplot(222)
plot(f,10*log10(abs(fftshift(derf).^2/length(f))));%画出混频后的功率谱
axis([-fc-3*B fc+3*B -50 0]);
xlabel('f');
ylabel('混频后信号的PDF');
%---------------再经过低通滤波器
pause%画出理想低通滤波器
[LPF,f]=lp_f(length(derf),B,df1,fs,1);%求低通滤波器
subplot(224) % 画出理想低通滤波器
plot(f,fftshift(abs(LPF))); axis([-fc-3*B fc+3*B -0.2 1.2]);
xlabel('f');
ylabel('理想LPF');
pause%混频信号经理想低通滤波器后的波形及功率谱
DM = LPF.*derf;  %理想低通滤波器输出信号频谱
[dm]=F2T(DM,fs); %理想低通滤波器的输出波形
%-------这个图为figure(5)--------%
```




```
figure(6)
subplot(221)
plot(t,dm(1:Lt));%画出经过低通滤波器后的解调出的波形
axis([0 10 -1.2 1.2]);
xlabel('t');
ylabel('LPF输出信号');
subplot(222)
[dmf,dm,df1,f]=T2F(dm,ts,df,fs);%求LPF输出信号的功率谱
plot(f,10*log10(abs(fftshift(dmf).^2/length(f))));%画出LPF输出信号的功率谱
axis([-fc-3*B fc+3*B -50 0]);
xlabel('f');
ylabel('LPF输出信号功率谱');
%-------最后对LPF输出信号抽样判决
panjue=zeros(1,N);%建立存储判决值的矩阵
%抽样判决，规则：大于0判1，小于判-1
for i=1:N;
    if dm(fc*N_sample*(i-1)+fc*N_sample/2+1)>0;%抽样判决时刻
        panjue(i)=1;
    else
        panjue(i)=-1;
    end
end
%----画出判决出的基带信号波形，并和调制信号比较
rr=sigexpand(panjue,fc*N_sample);
rrt=ones(1,fc*N_sample); % NRZ波形
huifu_NRZ=conv(rr,rrt);
pause%观察调制信号和解调信号波形
```

%-------这个图为figure(6)--------%

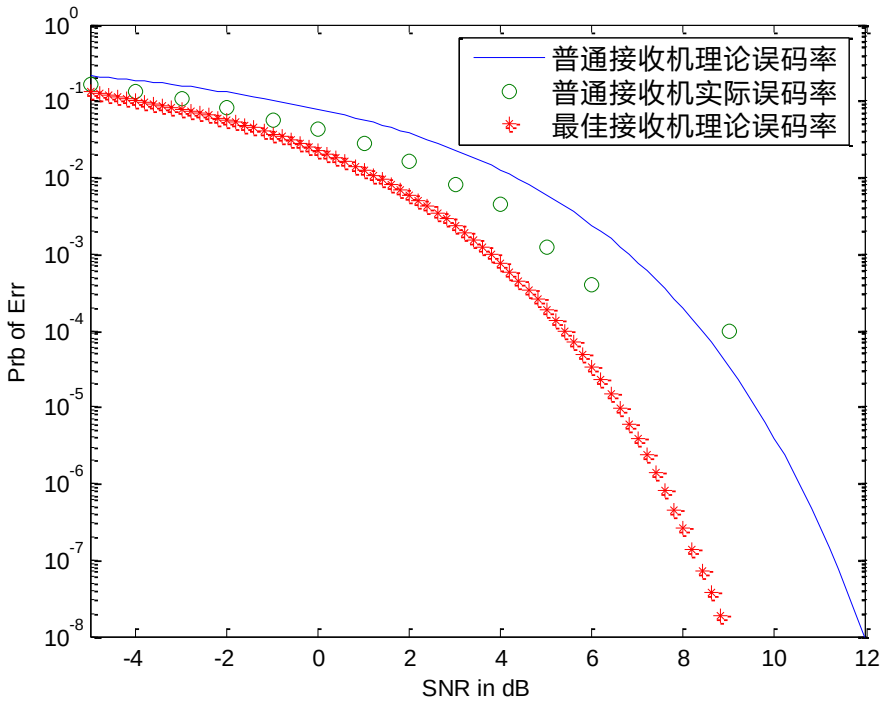



```
subplot(224)
plot(t,d_sjx(1:Lt));%调制信号波形
axis([0 10 -1.2 1.2]);
xlabel('t');
ylabel('调制信号');
subplot(223)
plot(t,huifu_NRZ(1:Lt));%解调信号波形
axis([0 10 -1.2 1.2]);
xlabel('t');
ylabel('解调信号');
%-------------------统计误码数
numoferr=sum(abs(panjue-d)/2)/N%计算出错误码元数
```

主程序2：误码率仿真

```
clear all
close all
A=1;    %载波振幅
fc=2;   %载波频率（Hz）
SNRindB1=-5:1:12;      % 信噪比取值向量，dB为单位
SNRindB2=-5:0.2:12;    % 信噪比取值向量，dB为单位
N_sample=3;%每个码元的的采样点数
N=10000;  %  码元数
Ts=1;   % 码元宽度
d=sign(randn(1,N));
df=0.01;
simu_err_prb=zeros(1,length(SNRindB1));
```

```
%理论误码率
for i=1:length(SNRindB2)
    % 计算信噪比值
    SNR=10^(SNRindB2(i)/10);
 % 计算普通接收机的理论误码率。
theo_err_prb(i)=0.5*erfc(sqrt(SNR));%一般蒙特卡洛仿真是针对最佳接收机的，对于来自同一信道的接收信号，信道高斯噪声的功率谱密度是不变的，最佳接收机性能之所以比一般接收机好，是由于其输入噪声比是一般接收机的2倍（一般接收机带通滤波器带宽按2/Ts计算）；故信道高斯白噪声的单边带功率谱密度即可按最佳接收机设计，也可按普通接收机设计。
%互补误差函数
theo_err_prb1(i)=0.5*erfc(sqrt(2*SNR));%最佳接收机理论误码率曲线
end
%计算普通接收机实际误码率
for i=1:length(SNRindB1)
[numoferr,panjue,desingal,t]=bpskberr(A,fc,SNRindB1(i),N_sample,N,Ts,d,df);
simu_err_prb(i)=numoferr;
end
% 误码率曲线图：估计值和理论值曲线对比图
semilogy(SNRindB2,theo_err_prb,SNRindB1,simu_err_prb,'o',SNRindB2,theo_err_prb1,'*');
axis([-5 12 0.00000001 1]);
xlabel('SNR in dB');
ylabel('Prb of Err');
legend('普通接收机理论误码率','普通接收机实际误码率','最佳接收机理论误码率');
```



4、子函数

①.误码率

function [numoferr,panjue,desingal,t]=bpskberr(A,fc,snr,N_sample,N,Ts,d,df)

```
%求误码率
%--------------------系统仿真参数
% A;    %载波振幅
% fc;   %载波频率（Hz）
% snr;  %信噪比dB
% N_sample;%每个码元的的采样点数
% N;  %  码元数
% Ts;   % 码元宽度
%d;输入二进制代码
%df:频率分辨率
% ---------------------输出(返回)参数
%numoferr;%误码率
%panjue 恢复的二进制代码1用1表示，0用-1表示
%desingal;%恢复的数字基带信号
%t;时域采样时序
%-----------------------生成调制信号
B=1/Ts;
f_start=fc-B;
f_cutoff=fc+B;
fs=fc*N_sample;%系统采样频率
ts=Ts/fs; % 系统采样间隔
t=0:ts:N*Ts-ts;
Lt=length(t);
% 产生二进制信源
dd=sigexpand((d+1)/2,fc*N_sample);
gt=ones(1,fc*N_sample); % NRZ波形
d_NRZ=conv(dd,gt);
d_sjx=2*d_NRZ-1;%生成双极性NRZ信号
%----对数字基带信号进行2PSK调制
ht=A*sin(2*pi*fc*t);%载波
s_2psk=d_sjx(1:Lt).*ht;%生成2PSK信号
%-----------------生成高斯白噪声噪声
snr_lin=10^(snr/10); %换算成倍数
signal_power=power_x(s_2psk(1:Lt));%求出信号平均功率
noise_power=(signal_power*fc*N_sample)/(snr_lin*2*(f_cutoff-f_start));%求出噪
声方差（噪声均值为0）
noise_std=sqrt(noise_power);%求出噪声均方差
noise=noise_std.*randn(1,Lt);%以噪声均方差作为幅度产生高斯白噪声
%---------------将已调信号送入信道
r=s_2psk(1:Lt)+noise(1:Lt);%叠加了噪声的已调信号，相当于将已调信号送入理想信道
```

```
[rf,r,df1,f]=T2F(r,ts,df,fs);
%----------在接收端先通过带通滤波器
[H,f]=bp_f(length(rf),f_start,f_cutoff,df1,fs,1);
DEM = H.*rf;       %滤波器输出的功率谱
[dem]=F2T(DEM,fs);%滤波器的输出信号波形
%进行相干解调，先和本地载波相乘,即混频
%和本地载波相乘
der=dem(1:Lt).*ht;%混频
[derf,der,df1,f]=T2F(der,ts,df,fs);%求混频后信号的功率谱
%----------------再经过低通滤波器
[LPF,f]=lp_f(length(derf),B,df1,fs,1);
DM = LPF.*derf; %理想低通滤波器输出的功率谱
[dm]=F2T(DM,fs);%理想低通滤波器的输出信号
%----------最后对LPF输出信号抽样判决
panjue=zeros(1,N);%建立存储判决值的矩阵
%抽样判决，规则：大于0判1，小于0判-1
for i=1:N
    if dm(fc*N_sample*(i-1)+fc*N_sample/2+1)>0;%抽样判决时刻
        panjue(i)=1;
    else
        panjue(i)=-1;
    end
end
%----------生成判决出的基带信号波形dd1=sigexpand(panjue,fc*N_sample);
gt1=ones(1,fc*N_sample); % NRZ波形
desinga=conv(dd1,gt1);
desingal=desinga(1:Lt);
%-------------------------统计误码数
numoferr=sum(abs(panjue-d)/2)/N;%计算出错误码元数
```

②.计算信号功率

```
function p=power_x(x)
%x：输入信号
%p：返回信号的x功率
p=(norm(x).^2)./length(x);
```

③.FFT逆变换

```
function [m]=F2T(M,fs)
%-------------------------输入参数
%M：信号的频谱
%fs:系统采样频率
%--------------------输出(返回)参数
```

%m:傅里叶逆变换后的信号，注意其长度为2的整数次幂，利用其画波形时，要注意选取m的一部分，选取长度和所给时间序列t的长度要一致，plot(t,m(1:length(t)))，否则会出错。

```
m = real(ifft(M))*fs;
```

④. FFT（主要对fftseq中的频率和频谱向量做校正）

```
function [M,m,df1,f]=T2F(m,ts,df,fs)
%------------------------输入参数
%m:信号
%ts:系统时域采样间隔
%df:所需的频率分辨率
%fs:系统采样频率
%----------------------输出(返回)参数
%M:傅里叶变换后的频谱序列
%m: 输入信号参与过傅里叶变换后对应的序列，需要注意的是，该序列与输入信号m的区别，其长度是不一样的，输入的m长度不一定是2的整数次幂，而傅里叶变换要求输入信号长度为2的整数次幂，故傅里叶变换前需对输入的m信号进行补零，其长度有所增加，故输出参数中的m为补零后的输入信号，其长度与输入参数m不一样，但与M,f长度是一样的，并且，其与时间序列t所对应的序列m(1:length(t))与输入参数中的m是一致的。
%df1:返回的频率分辨率
%f:与M相对应的频率序列
[M,m,df1]=fftseq(m,ts,df);
f = [0:df1:df1*(length(m)-1)] -fs/2;     %频率向量
M=M/fs;
```

⑤.FFT子函数

```
function [M,m,df]=fftseq(m,ts,df)
%各参数含义与子函数T2F中的完全相同。
fs = 1/ts;
if nargin ==2
    n1 =0;
else
    n1 = fs/df;
end
n2 = length(m);
n = 2^(max(nextpow2(n1),nextpow2(n2)));
M = fft(m,n);
m = [m,zeros(1,n-n2)];
df = fs/n;
```

⑥. 序列补零扩展子函数

```
function [out]=sigexpand(d,M)
% 将输入的序列扩展成间隔为N-1个0的序列
%------------------------输入参数
%M:在这里指系统采样频率，即在一个码元宽度内采多少个样点，设计时要考虑载波频率与基带信号采样率。M为二者的乘积。
%d:输入的二进制代码
%----------------------输出(返回)参数
```

```
%out：为1*(M*length(d))维的矩阵
例如d=[1 1],M=3,则out=[1 0 0 1 0 0]
N=length(d);
out=zeros(M,N);
out(1,:)=d;
out=reshape(out,1,M*N);
```

⑦．理想带通滤波器

```
Function[H,f]=bp_f(n,f_start,f_cutoff,df1,fs,p)
%带通滤波器函数     输入设计的滤波器参数，产生带通滤波器频率特性函数H和频率向量f
%-------------------------输入参数
%n  带通滤波器的输入信号长度
%f_start     通带起始频率
%f_cutoff    带通滤波器的截止频率
%df1         频率分辨率
%fs          抽样频率
%p           滤波器幅度
%-----------------------输出(返回)参数
%H  带通滤波器频率响应
%f  频率向量
n_cutoff = floor(f_cutoff/df1);       %设计滤波器
n_start = floor(f_start/df1);
f = [0:df1:df1*(n-1)] -fs/2;     %频率向量
H = zeros(size(f));
H(n_start+1:n_cutoff) = p*ones(1,n_cutoff-n_start);
H(length(f) - n_cutoff+1:length(f)-n_start) = p*ones(1,n_cutoff-n_start);
```

⑧．理想低通滤波器

```
function [H,f]=lp_f(n,f_cutoff,df1,fs,p)
%低通滤波器函数     输入设计的滤波器参数，产生低通滤波器频率特性函数H和频率向量f
%-----------------------输入参数
%n  低通滤波器的输入信号长度
%f_cutoff    低通滤波器的截止频率
%df1         频率分辨率
%fs          抽样频率
%p           滤波器幅度
%---------------------输出(返回)参数
%H  低通滤波器频率响应
%f  频率向量
n_cutoff = floor(f_cutoff/df1);       %设计滤波器
f = [0:df1:df1*(n-1)] -fs/2;     %频率向量
H = zeros(size(f));
H(1:n_cutoff) = p*ones(1,n_cutoff);
H(length(f) - n_cutoff+1:length(f)) = p*ones(1,n_cutoff);
```

第七章 通信系统仿真综合实验

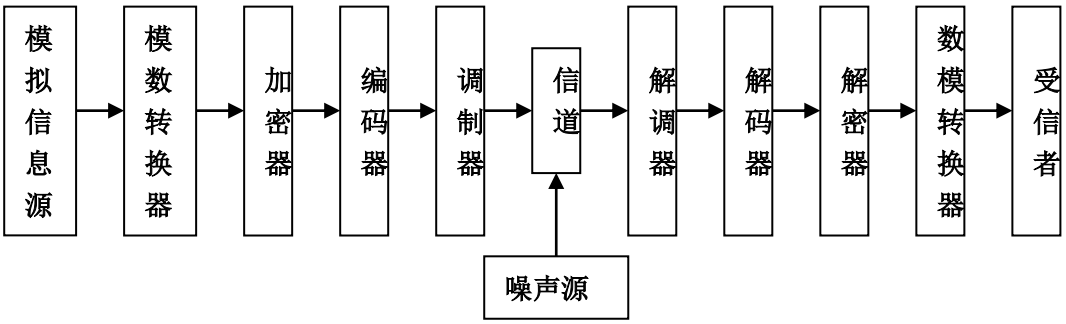
本章目标

- 巩固抽样定理、均匀量化、PCM 编码仿真的思想。
- 巩固二进制数字调制的原理。
- 在上述实验的基础上，实现综合的数字通信系统的仿真。

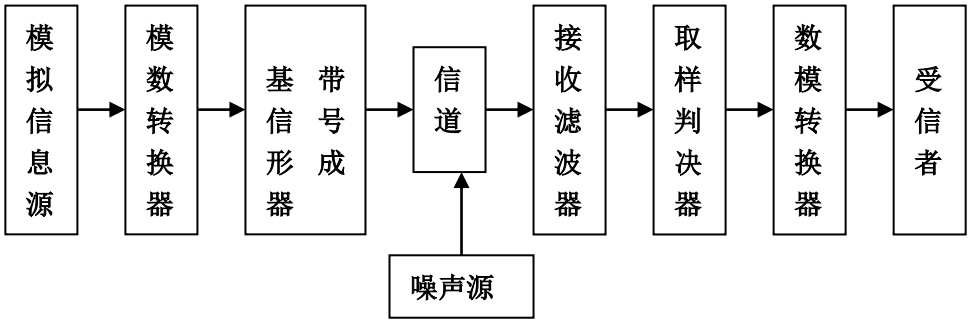
7.1 基本原理

信道中传输数字信号的系统称为数字通信系统。在日常生活中大部分信号为连续变化的模拟信号，那么要实现模拟信号在数字系统中的传输，则必须在发送端将模拟信号数字化，即 A/D 变换；在接收端需进行相反的变换，即 D/A 变换。数字通信系统可进一步细分为数字频带传输通信系统和数字基带传输通信系统，下面分别加以说明。

7.1.1 数字频带传输通信系统



7.1.2 数字基带传输通信系统



7.2 实验内容

在前四章的基础之上，合理运用第五章，第六章的主函数及子函数完成通信系统仿真综合实验，以下两个题目可任选其一。

1、 利用 MATLAB 软件完成模拟信号的数字频带传输系统的仿真，设模拟调制信号为

$$m(t)=\begin{cases}1 \quad , \quad 0\le t<t_0/3\\ -2 \quad , \quad t_0/3\le t<2t_0/3\\ 0 \quad , \quad 2t_0/3\le t<t_0\end{cases},$$

t_0 自设，要求：

①画出数字频带传输通信系统中除加密解密器外其余各点的波形及其对应的频谱或功率谱，调制方式自选，信噪比自设，编码器输出信号为双极性不归零码；同时画出对应的理想滤波器幅频特性

②分析不同信噪比下的抗噪声性能。

2、利用 MATLAB 软件完成数字基带传输系统的仿真，输入模拟信号见题 1。要求：

①画出数字基带传输通信系统中除基带信号形成器及接收滤波器以外，其余各点的波形及其对应的频谱或功率谱；同时画出信道对应的理想低通滤波器的幅频特性。

②分析不同信噪比下的抗噪声性能。