第1章　Matlab 简介及信号处理基础

在信息隐藏与数字水印实验过程中，主要使用 Matlab 和 VC 等工具来实现各种算法。本章通过介绍 Matlab 命令来掌握信息隐藏实验过程中所需的各种 Matlab 操作，同时了解对图像和音频等载体信号进行离散余弦、离散小波变换等基本操作的方法，为今后的其他实验打下良好基础。

1.1　Matlab 简介

【实验目的】

了解 Matlab 软件使用环境，熟悉 Matlab 使用方法，掌握信息隐藏实验中的各种常用 Matlab 命令。

【实验环境】

（1）Windows XP 或 Vista 操作系统；

（2）Matlab7.1 版本软件。

【原理简介】

Matlab 是广泛使用的一种可视化科学计算软件，它具有语法结构简单、数值计算高效、图形功能完备和图像处理方便的特点，是信号处理和信息隐藏处理中使用最多的软件。

【实验步骤】

1. 常用命令

dir：列出当前目录下的所有文件；

clc：清除命令窗口；

clear all：清除环境（从内存中清除所有变量）；

who：将内存中的当前变量以简单形式列出；

close all：关闭所有的 Figure 窗口。

2. 变量设置

1）变量命名

Matlab 的变量名以字母打头，后面最多可跟 19 个字母或数字，如 x、y、ae3 和 d3er45 等都是合法的变量名，不能使用内部函数或命令名作为变量名。Matlab 中的变量名区分大小写，ab 与 Ab 表示两个不同的变量。

2）变量赋值

（1）表达式赋值：

```
>>a = (100 * 0.02)/4.0
  a = 0.0500
```

（2）矩阵赋值。数值通常按行输入，行之间用分号隔开。

```
>>C = [ -1,0,0;1, -1,0;0,0,2];(省略最后的分号，Matlab 会回显矩阵值)
```

（3）通过引用特定的位置可以单独改变某个矩阵元素。如 S = [5,6,4]，用命令 S(2) = 8 把矩阵 S 的第二个元素值由 6 改成 8。

（4）可以引用已定义的矩阵，重新定义一个新矩阵。如 S = [5,6,4]，可定义一个新矩阵。

```
>>B = [3 S 2]
  B =
    3 5 6 4 2
```

3. 整数操作

1）fix(x)：截尾取整

```
>>fix([3.12 -3.12])
ans = 3     -3
```

2）floor(x)：不超过 x 的最大整数(高斯取整)

```
>>floor([3.12 -3.12])
ans = 3     -4
```

3）ceil(x)：大于 x 的最小整数

```
>>ceil([3.12 -3.12])
ans = 4     -3
```

4. 随机序列常用命令

1）rand：均匀分布随机矩阵

```
rand                    % 无变量输入时只产生一个随机数
y = rand(n)             % 生成 n × n 随机矩阵，其元素在(0,1)内
y = rand(m,n)           % 生成 m × n 随机矩阵，其元素在(0,1)内
```

例 1.1 产生一个 3 × 4 随机矩阵，Matlab 命令为：R = rand(3,4)。

2）randn：正态分布随机矩阵

```
randn                   % 无变量输入时只产生一个正态分布随机数
y = randn(n)            % 生成 n × n 正态分布随机矩阵
y = randn(m,n)          % 生成 m × n 正态分布随机矩阵
```

例 1.2 产生均值为 0.6、方差为 0.1 的 4 阶矩阵。

Matlab 命令为：>>mu = 0.6;sigma = 0.1;>>x = mu + sqrt(sigma) * randn(4)。

3）randsrc：产生均匀分布数组

```
randsrc                 % 无变量输入时只产生一个随机数 1 或者 -1
y = randsrc(n)          % 生成 n × n 随机数组，其元素为 1 或者 -1
y = randsrc(m,n)        % 生成 m × n 随机数组，其元素为 1 或者 -1
```

例 1.3　产生一个 2×3 随机矩阵，Matlab 命令为：R = randsrc(2,3)。

5. 矩阵常用操作命令

Matlab 的基本单位是矩阵，掌握矩阵的输入、各种数值运算以及矩阵函数是学好 Matlab 的关键。

1）矩阵的输入

（1）直接输入创建矩阵。以“[”和“]”作为首尾，同行的元素用“,”或空格隔开，不同行的元素用“;”或按 Enter 键来分隔；矩阵的元素可以是数字也可以是表达式，如果是数值计算，表达式中不可包含未知变量。

例 1.4　直接输入创建矩阵 $A=\begin{pmatrix}1 & 2 & 3\\ 4 & 15 & 60\\ 7 & 8 & 9\end{pmatrix}$。

Matlab 命令为：>>A = [1,2,3;4,15,60;7,8,9]。

（2）用矩阵函数来生成矩阵。Matlab 提供了大量的函数来创建特殊矩阵，表 1.1 给出 Matlab 常用的矩阵函数。

表 1.1　常用的矩阵函数

函数名称	函数功能	函数名称	函数功能
zero(m,n)	m 行 n 列零矩阵	rand(m,n)	m 行 n 列随机矩阵
eye(n)	n 阶方矩阵	randn(m,n)	m 行 n 列正态随机矩阵
ones(m,n)	m 行 n 列元素为 1 的矩阵	magic(n)	n 阶魔方矩阵

例 1.5　生成矩阵 $A=\begin{pmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{pmatrix}$。

Matlab 命令为：>>ones(3)。

例 1.6　生成矩阵 $A=\begin{pmatrix}0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\end{pmatrix}$。

Matlab 命令为：>>zeros(2,5)。

例 1.7　生成 3 阶魔方矩阵 $A=\begin{pmatrix}8 & 1 & 6\\ 3 & 5 & 7\\ 4 & 9 & 2\end{pmatrix}$。

Matlab 命令为：>>magic(3)。

2）操作符“:”的说明

j:k 表示步长为 1 的等差数列构成的数组：[j,j+1,j+2,…,k]。

j:i:k 表示步长为 i 的等差数列构成的数组：[j,j+i,j+2×i,…,k]。

A(i:j)表示 A(i),A(i+1),…,A(j)。

3）对矩阵元素的操作

设 A 是一个矩阵，则在 Matlab 中用如下符号表示它的元素：

A(i,j)表示矩阵 A 的第 i 行第 j 列元素。

A(:,j)表示矩阵 A 的第 j 列。

A(i,:)表示矩阵 A 的第 i 行。

A(:,:)表示 A 的所有元素构造的 2 维矩阵。

A(:)表示以矩阵 A 的所有元素按列构成的一个列矩阵。

A(i)表示矩阵 A(:)的第 i 个元素。

[ ]表示空矩阵。

4) 矩阵的运算

A + B:矩阵加法。

A - B:矩阵减法。

A * B:矩阵乘法。

A′:A 的转置。

k * A:数 k 乘以 A。

det(A):A 的行列式。

rank(A):A 的秩。

5) 数组

在 Matlab 中数组是一行或者一列的矩阵,对矩阵的输入、修改和保存都适用于数组,同时 Matlab 还提供了一些创建数组的特殊指令。

(1) 特殊数组的创建:

linspace(a,b,n)　给出区间[a,b]的 n 个等分点数据

例 1.8　给出区间[0,1]的 6 个等分点数据。

Matlab 命令为:>>linspace(0,1,6)。

(2) 数组运算。数组运算除作为 1 × n 的矩阵应遵循矩阵的运算规则外,Matlab 中还为数组提供了一些特殊的运算:乘法为:.*,乘幂为:.^。数组运算强调元素对元素的运算。

例 1.9　数组运算。

```
>>a=1:5                    % 定义数组 a
a  =1     2     3     4     5
>>b=3:2:11                 % 定义数组 b
b  =3     5     7     9    11
>>a.^2                     % 数组 a 每一个元素求平方
ans  =1     4     9    16    25
>>a.* b                    % 数组 a 每一个元素乘以对应数组 b 的元素
ans  =3    10    21    36    55
```

6. 位操作

1) bitand:按位与

C = bitand(A,B)命令将返回两个非负整数数组 A 和 B 的相应元素按位与操作的结果。为了确保 A 和 B 的元素都是整数,可以使用 ceil、fix、floor 或 round 函数来生成 A 和 B。

例 1.10　>>C = bitand(1,2)

结果为:C = 0。

2) bitor:按位或

C = bitor(A,B)命令将返回两个非负整数数组 A 和 B 的相应元素按位或操作的结果。为了确保 A 和 B 的元素都是整数,可以使用 ceil、fix、floor 或 round 函数来生成 A

和 B。

例 1.11　>>C = bitor(1,2)

结果为:C = 3。

3）bitxor:按位异或

C = bitxor(A,B)返回两个非负整数数组 A 和 B 的相应元素进行按位异或的结果。为了确保 A 和 B 的元素都是整数,可以使用 ceil、fix、floor 或 round 函数来生成 A 和 B。

例 1.12　>>C = bitxor(1,2)

结果为:C = 3。

4）bitset:设置指定位的值

C = bitset(A,bit,v)命令将 A 中元素第 bit 位设为 v,其中 v 必须为 0 或 1,A 中的元素必须为非负整数,bit 必须为 1 ~ A 中元素浮点整数表示法位数之间的一个数字。

例 1.13　>>C = bitset(1,2,1)

结果为:C = 3。

5）bitget:获取指定位的值

C = bitget(A,bit)命令将返回 A 中元素第 bit 指定位的值,A 中的元素必须为非负整数,bit 必须为 1 ~ A 中元素浮点整数表示法位数之间的一个数字。

例 1.14　>>C = bitget(1,2)

结果为:C = 0。

7. 绘图操作

1）图形标注

title('string','属性名','属性值',…)——给图形加标题

xlabel('string','属性名','属性值',…)——给 x 轴加标注

ylabel('string','属性名','属性值',…)——给 y 轴加标注

legend('string1','string2',…)——添加图例,其顺序对应于绘图指令中的顺序

axis([xmin,xmax,ymin,ymax])——控制坐标轴的刻度范围

2）二维图形

(1) plot(x,y)。

功能:以向量 x、y 为轴,绘制曲线。

注:plot(x,y)命令可用来绘制函数 f(x)图形,此时可通过向量 x 常用命令 x = a:h:b 的形式获得 f(x)函数在绘图区间[a,b]上的自变量点向量数据,对应的函数向量值取为 y = f(x)。步长 h 可以任意选取,一般步长越小,曲线越光滑,但是步长太小,会增加计算量,运算速度也要降低。通常步长 h 取值 0.1 可达到较好的绘图效果。

例 1.15　绘制函数 $y = \sin x^2$ 在 $-5 \leqslant x \leqslant 5$ 的图形。

解　Matlab 命令:

```
x = -5:.1:5;                    % 取绘图横坐标向量点 x
y = sin(x.^2);
plot(x,y);
xlable('input value');
ylable('function value');
```

```
title('曲线 y = sinx^2');
```

绘图结果如图 1.1 所示。

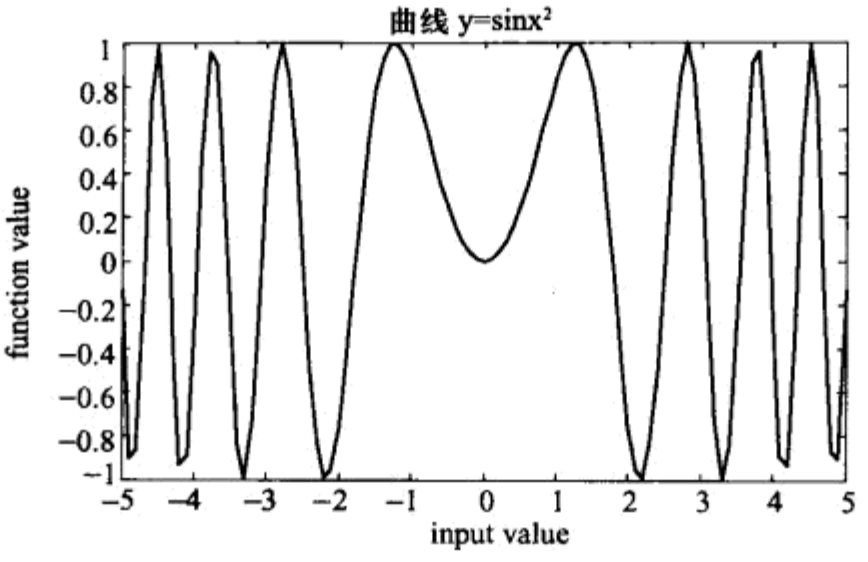

图 1.1　绘图结果

(2) plot(x1,y1,x2,y2,x3,y3,…)。

功能:在同一图形窗口绘制多条不同颜色曲线,曲线关系为

$$y_1 = f(x_1), y_2 = f(x_2), y_3 = f(x_3)$$

例 1.16　在同一图形窗口画出三个函数 $y = \cos 2x$,$y = x^2$,$y = x$ 的图形,$-2 \leqslant x \leqslant 2$。

解　Matlab 命令:

```
x = -2:.1:2;
plot(x,cos(2* x),x,x.^2,x,x)
legend('cos(2x)','x^2','x')
```

绘图结果如图 1.2 所示。

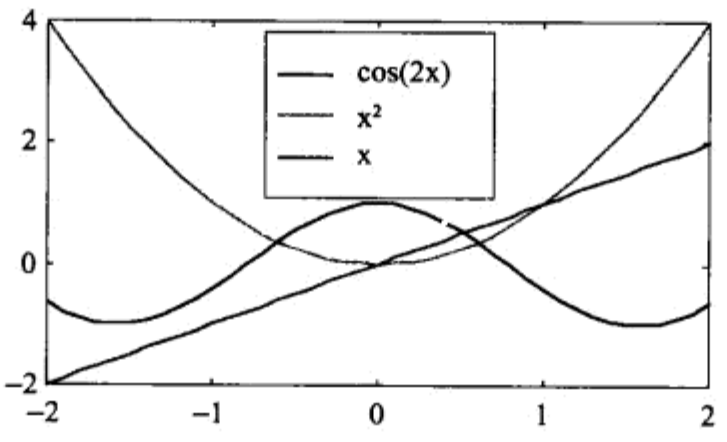

图 1.2　绘图结果

3) 二维特殊图形

(1) bar 表示的直方图。

例 1.17

```
>> x = 1:5;
>> bar(x),title('直方图');(如图 1.3 所示)
```

(2) hist 表示生成直方图。M = hist(N)表示将 N 中的最大值、最小值找出来,然后,平均取 10 个等间隔点,看以每个间隔点为中心,向两边各扩展 1/2 间隔的范围内,包括 N 的元素个数,因此 M 返回值都是 1 × 10 大小。

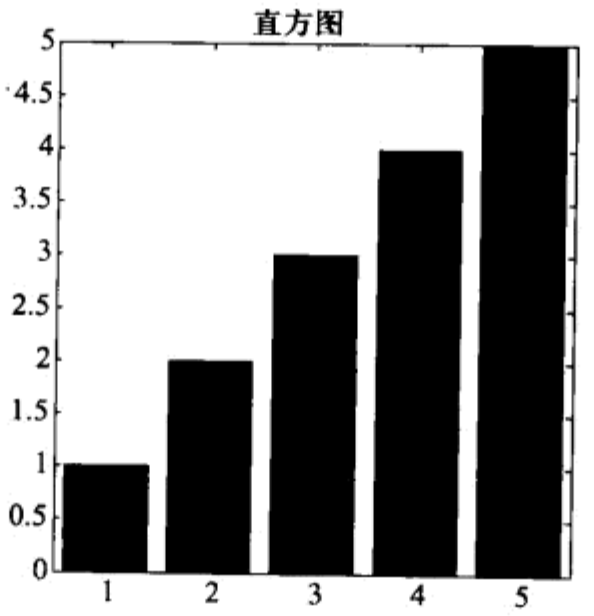

图 1.3　直方图显示结果

例 1.18

```
>> b1 = [0,1,2,3,4,5,6,7,8,9];b2 = [4,5,7];
>> M = hist(b2)
>> M = hist(b1)
```

8. 文件操作

1）fopen 打开文件

fopen 函数的调用格式为

fid = fopen(文件名,打开方式)

其中:文件名用字符串形式,表示待打开的数据文件。常见的打开方式有:r 表示对打开的文件读数据,w 表示对打开的文件写数据,a 表示在打开的文件末尾添加数据。fid 用于存储文件句柄值,句柄值用来标识该数据文件,其他函数可利用它对该数据文件进行操作。文件数据格式有两种形式:一种是二进制文件,另一种是文本文件。在打开文件时需要进一步指定文件格式类型,即指定是二进制文件还是文本文件。

2）fclose 关闭文件

文件读、写等操作完成后,应及时关闭。关闭文件用 fclose 函数,调用格式为

sta = fclose(fid)

该函数关闭 fid 所表示的文件。sta 表示关闭文件操作的返回代码,若关闭成功,返回 0;否则返回 -1。

3）二进制文件的读写操作

(1) fread 读二进制文件。fread 函数可以读取二进制文件的数据,并将数据存入矩阵。其调用格式为

[A,COUNT] = fread(fid,size,precision)

其中:A 用于存放读取的数据;COUNT 返回所读取的数据元素个数;fid 为文件句柄;size 为可选项(若不选用则表示读取整个文件内容,若选用则它的值可为以下选项:N 表示读取 N 个元素到一个列向量;Inf 表示读取整个文件;[M,N]表示读数据到 M × N 的矩阵中,数据按列存放);precision 代表读写数据的类型。

(2) fwrite 写二进制文件。fwrite 函数按照指定的数据类型将矩阵中的元素写入到文

件中。其调用格式为

COUNT = fwrite（fid,A,precision）

其中:COUNT 返回所写的数据元素个数;fid 为文件句柄;A 用来存放写入文件的数据;precision 用于控制所写数据的类型,其形式与 fread 函数相同。

9. M 文件的建立与使用

M 文件有命令文件和函数文件两种形式,这两种文件的扩展名相同,都是. m。

当用户要运行的命令较多时,直接从键盘上逐条输入较为烦琐,可利用命令文件来解决多行输入问题。用户可将一组相关命令编辑在同一个命令文件中,运行时只需输入文件名字,Matlab 就会自动按顺序执行文件中的命令。函数文件是另一种形式的 M 文件,它的第一句可执行语句是以 function 引导的定义语句,在函数文件中的变量都是局部变量。

1）命令文件

命令文件的一般形式为

< M 文件名 >. m

如 a1. m,pp. m 等都是合法的 M 文件名。

M 文件有两种运行方式:一是在命令窗口直接输入文件名,按 Enter 键; 二是在编辑窗口打开菜单 Tools,再单击 Run。M 文件保存的路径一定要在搜索路径上,否则 M 文件不能运行。

例 1.19 用 M 命令文件绘制 lena. bmp 图像。

第一步:打开 Matlab 命令窗口,单击【File】|【New】|【Mfile】,打开编辑窗口。

第二步:在编辑窗口中输入:

```
clc;
clear;
[fn,pn] =uigetfile('* .bmp','请选择图像文件');
[x,map] = imread(strcat(pn,fn),'bmp');
imshow(x);
```

第三步:保存 M 文件,并且保存在搜索路径上,文件名为 showlena. m。

第四步:运行 M 文件。在命令窗口输入 showlena,并按 Enter 键;或在编辑窗口打开菜单 Tools,再选择 Run 命令。

2）函数文件

M 函数文件的一般形式为

function <因变量> = <函数名>(<自变量>)

M 函数文件可以有多个因变量和多个自变量,当有多个因变量时用[ ]括起来。

例 1.20 读入并绘制图像。

第一步:打开 Matlab 命令窗口,单击【File】|【New】|【Mfile】,打开编辑窗口。

第二步:在编辑窗口中输入:

```
functiony = a(x)
imshow(x);
```

第三步:保存 M 函数文件,并且保存在搜索路径上,文件名为 showimage. m。

第四步:在命令窗口执行下列语句:

```
[fn,pn] =uigetfile('* .bmp','请选择图像文件');
```

```
[x,map] = imread(strcat(pn,fn),'bmp');
a(x);
```

【思考题】

1. 分别用 rand 和 linspace 生成随机矩阵和数组，练习矩阵基本运算和数组基本运算。

2. 练习指令 bar，barh，hist，pie，area。

3. 随机生成 a、b，建立一个命令文件将变量 a、b 的值互换。

4. 建立一个函数文件，求出变量 a、b 中的最大值，在其他地方调用该函数求出两个数中的最大值。

1.2 信号处理基础

【实验目的】

了解音频和图像数据系数特点，掌握音频和图像文件的离散傅里叶、离散余弦和离散小波变换等基本操作。

【实验环境】

（1）Windows XP 或 Vista 操作系统；

（2）Matlab7.1 版本软件；

（3）BMP 格式图像文件；

（4）WAV 格式音频文件。

【原理简介】

离散傅里叶、离散余弦和离散小波变换是图像、音频信号常用基础操作，时域信号转换到不同变换域以后，会导致不同程度的能量集中，信息隐藏利用这个原理在变换域选择适当位置系数进行修改，嵌入信息，并确保图像、音频信号经处理后感官质量无明显变化。

一维离散傅里叶变换对定义：

一维离散傅里叶变换：

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}$$

一维离散傅里叶逆变换：

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi kn/N}$$

一维离散余弦变换对定义：

一维离散余弦正变换：

$$C(0) = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} f(x)$$

$$C(u) = \sqrt{\frac{2}{N}}\sum_{x=0}^{N-1} f(x)\cos\frac{(2x+1)u\pi}{2N}, u = 1,2,\cdots,N-1$$

一维离散余弦反变换：

$$f(x) = \frac{1}{\sqrt{N}}C(0) + \sqrt{\frac{2}{N}}\sum_{u=1}^{N-1} C(u)\cos\frac{(2x+1)u\pi}{2N}, x = 0,1,\cdots,N-1$$

一维连续小波变换对定义：

一维连续小波变换：

$$CWT_x(\tau,a) = \frac{1}{\sqrt{|a|}}\int x(t)h^*\left(\frac{t-\tau}{a}\right)\mathrm{d}t(h(t)\text{ 是小波母函数})$$

一维连续小波逆变换：

$$x(t) = \frac{1}{C_H}\iint \frac{1}{a^2}CWT_x(\tau,a)\frac{1}{\sqrt{|a|}}h\left(\frac{t-\tau}{a}\right)\mathrm{d}a\mathrm{d}b$$

二维离散傅里叶变换对定义：

二维离散傅里叶变换：

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\mathrm{e}^{-\mathrm{j}2\pi(ux/M+vy/N)}$$

$$u = 0,1,\cdots,M-1;\ v = 0,1,\cdots,N-1$$

二维离散傅里叶逆变换：

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)\mathrm{e}^{\mathrm{j}2\pi(ux/M+vy/N)}$$

$$x = 0,1,\cdots,M-1;y = 0,1,\cdots,N-1$$

二维离散余弦变换对定义：

二维离散余弦正变换：

$$C(0,0) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)$$

$$C(u,v) = \frac{2}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\frac{(2x+1)\pi u}{2N}\cos\frac{(2y+1)\pi v}{2N}$$

二维离散余弦反变换：

$$f(x,y) = \frac{1}{N}C(0,0) + \frac{2}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} C(u,v)\cos\frac{(2x+1)\pi u}{2N}\cos\frac{(2y+1)\pi v}{2N}$$

【实验步骤】

1. 用离散傅里叶变换分析合成音频和图像

1）分析合成音频文件(图 1.4)

① 读取音频文件数据；

② 一维离散傅里叶变换；

③ 一维离散傅里叶逆变换；

④ 观察结果。

(1) 读取音频文件数据。新建一个 m 文件，另存为 example11. m，输入以下命令：

```
clc;
```

```
clear;
len =40000;
[fn,pn] =uigetfile('* .wav','请选择音频文件');
[x,fs] =wavread(strcat(pn,fn),len);
```

uigetfile 是文件对话框函数，提供图形界面供用户选择所需文件，返回目标的目录名和文件名。

函数原型：y = wavread（file）。

功能：读取微软音频格式(wav)文件内容。

输入参数：file 表示音频文件名，字符串。

返回参数：y 表示音频样点，浮点型。

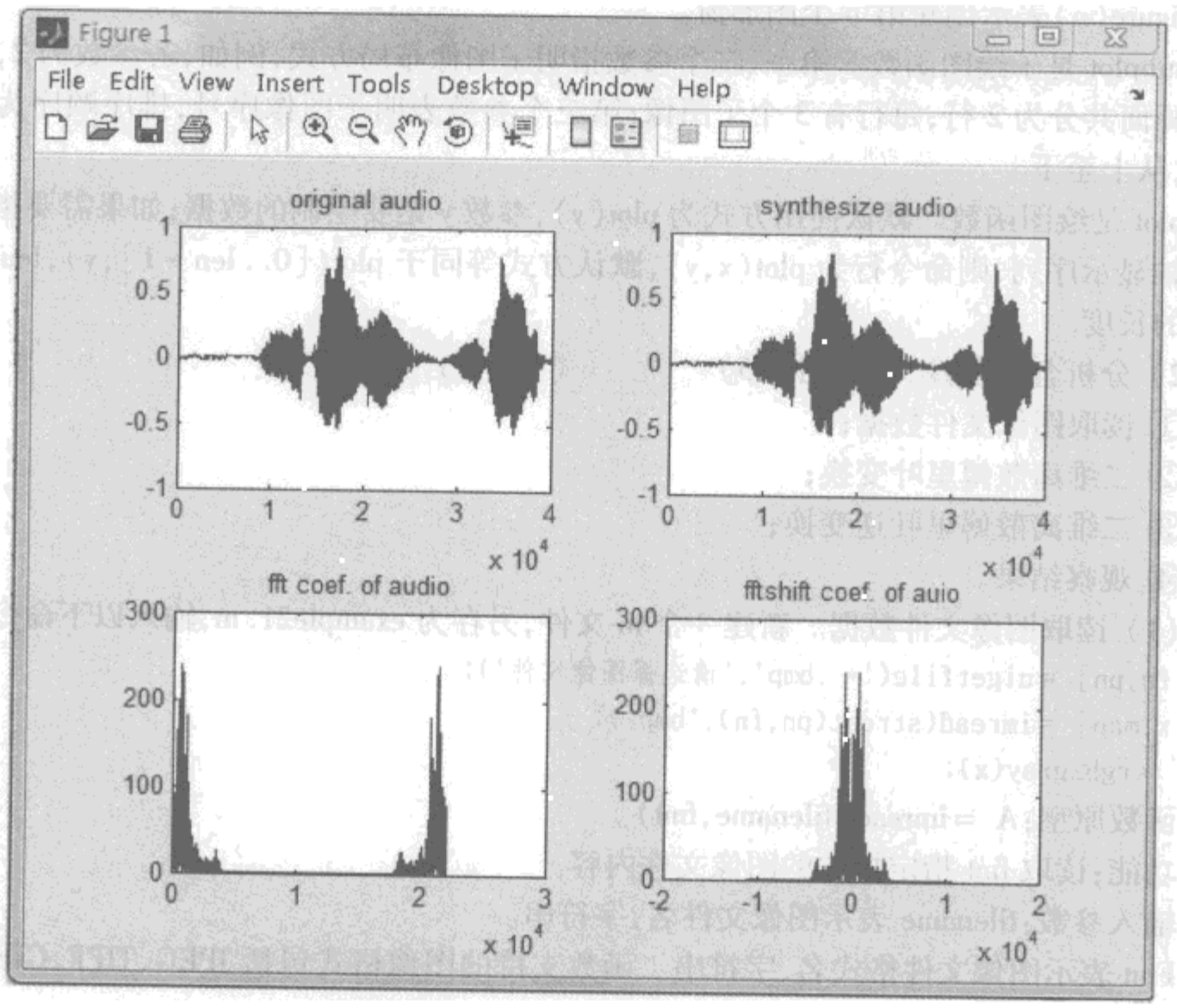

图 1.4　用离散傅里叶变换分析合成音频文件

（2）一维离散傅里叶变换。新建一个 m 文件，另存为 example12. m，输入以下命令：

```
xf =fft(x);
f1 =[0:len-1] * fs / len;
xff =fftshift(xf);
hl =floor(len / 2);
f2 =[-hl:hl] * fs / len;
```

fft 函数对输入参数进行一维离散傅里叶变换并返回其系数，对应频率从 0 至 fs（采样频率），使用 fftshift 将零频对应系数移至中央。上述代码还计算了离散样点对应的频率值，以便更好地观察频谱。

(3) 一维离散傅里叶逆变换。新建一个 m 文件,另存为 example13. m,输入以下命令:

```
xsync = ifft(xf);
```

ifft 函数对输入参数进行一维离散傅里叶逆变换并返回其系数。

(4) 观察结果。新建一个 m 文件,另存为 example14. m,输入以下命令:

```
figure;
subplot(2,2,1);plot(x);title('original audio');
subplot(2,2,2);plot(xsync);title('synthesize audio');
subplot(2,2,3);plot(f1,abs(xf));title('fft coef. of audio');
subplot(2,2,4);plot(f2(1:len),abs(xff));title('fftshift coef. of auio');
```

figure(n)表示创建第 n 个图形窗。

subplot 是子绘图函数。第一、二个参数指明子图像布局方式,例如,若参数为 2,3,则表示画面共分为 2 行,每行有 3 个子图像;第三个参数表明子图像序号,排序顺序为从左至右,从上至下。

plot 是绘图函数。默认使用方式为 plot(y),参数 y 是要绘制的数据;如果需要指明图像横轴显示序列,则命令行为 plot(x,y),默认方式等同于 plot([0..len - 1],y),len 为序列 y 的长度。

2) 分析合成图像文件(图 1.5)

① 读取图像文件数据;

② 二维离散傅里叶变换;

③ 二维离散傅里叶逆变换;

④ 观察结果。

(1) 读取图像文件数据。新建一个 m 文件,另存为 example21. m,输入以下命令:

```
[fn,pn] = uigetfile('* .bmp','请选择图像文件');
[x,map] = imread(strcat(pn,fn),'bmp');
I = rgb2gray(x);
```

函数原型:A = imread(filename,fmt)。

功能:读取 fmt 指定格式的图像文件内容。

输入参数:filename 表示图像文件名,字符串。

Fmt 表示图像文件格式名,字符串。函数支持的图像格式包括 JPEG、TIFF、GIF、BMP 等,当参数中不包括文件格式名时,函数尝试推断出文件格式。

返回参数:A 表示图像数据内容,整型。

函数 rgb2gray 将 RGB 图像转换为灰度图。

(2) 二维离散傅里叶变换。新建一个 m 文件,另存为 example22. m,输入以下命令:

```
xf = fft2(I);
xff = fftshift(xf);
```

fft2 函数对输入参数进行二维离散傅里叶变换并返回其系数,使用 fftshift 将零频对应系数移至中央。

(3) 二维离散傅里叶逆变换。新建一个 m 文件,另存为 example23. m,输入以下命令:

```
xsync =ifft2(xf);
```

ifft2 函数对输入参数进行二维离散傅里叶逆变换并返回其系数。

（4）观察结果。新建一个 m 文件，另存为 example24. m，输入以下命令：

```
figure;
subplot(2,2,1);imshow(x);title('original image');
subplot(2,2,2);imshow(uint8(abs(xsync)));title('synthesize image');
subplot(2,2,3);mesh(abs(xf));title('fft coef. of image');
subplot(2,2,4);mesh(abs(xff));title('fftshift coef. of image');
```

imshow 是二维数据绘图函数，mesh 通过三维平面显示数据。

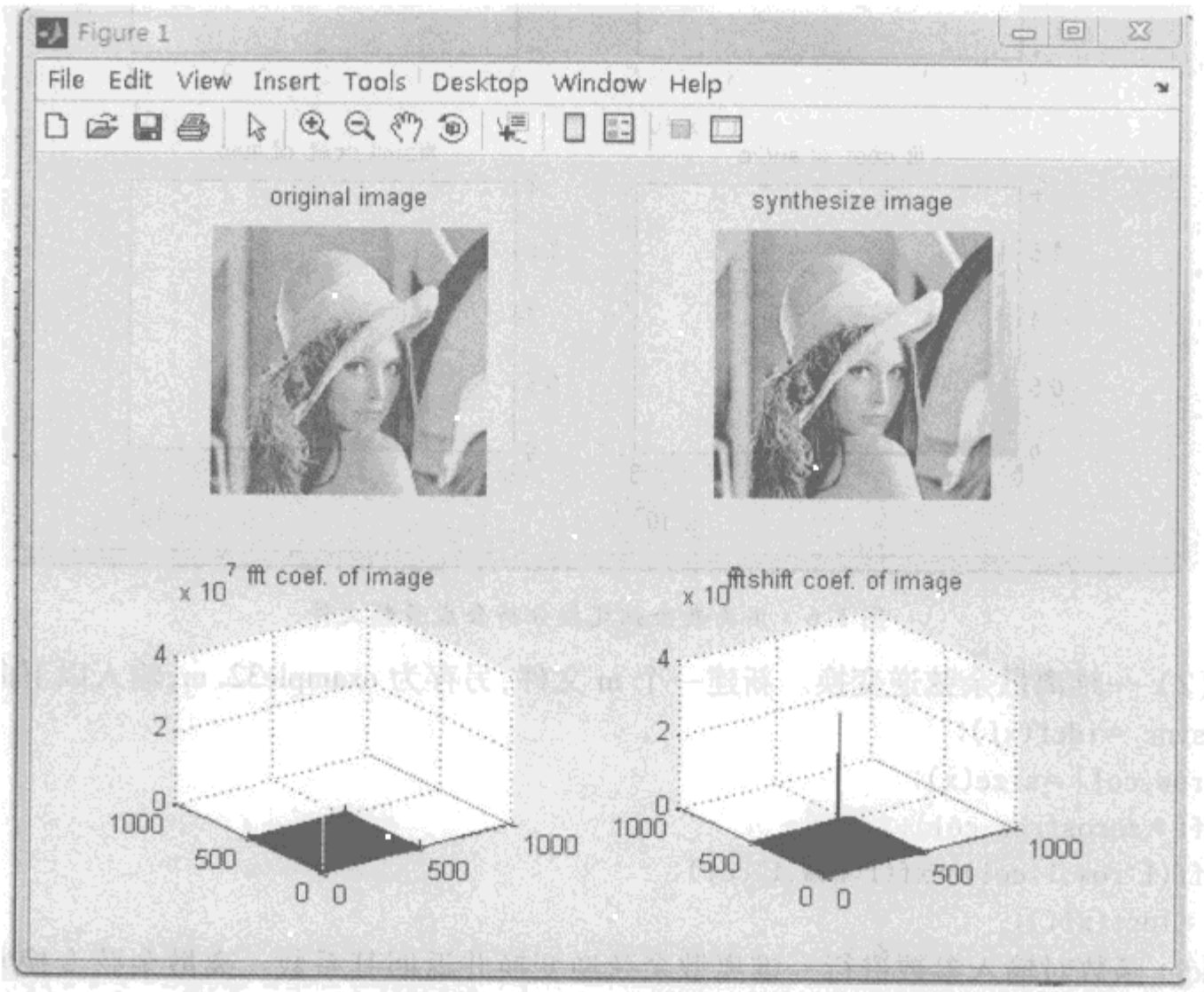

图 1.5　用离散傅里叶变换分析合成图像文件

2. 用离散余弦变换分析合成音频和图像

1）分析合成音频文件（图 1.6）

① 读取音频文件数据；

② 一维离散余弦变换；

③ 一维离散余弦逆变换；

④ 观察结果。

（1）一维离散余弦变换。新建一个 m 文件，另存为 example31. m，输入以下命令：

```
xf =dct(x);
```

dct 函数对输入参数进行一维离散余弦变换并返回其系数，对应频率从 0 至 fs（采样频率）。

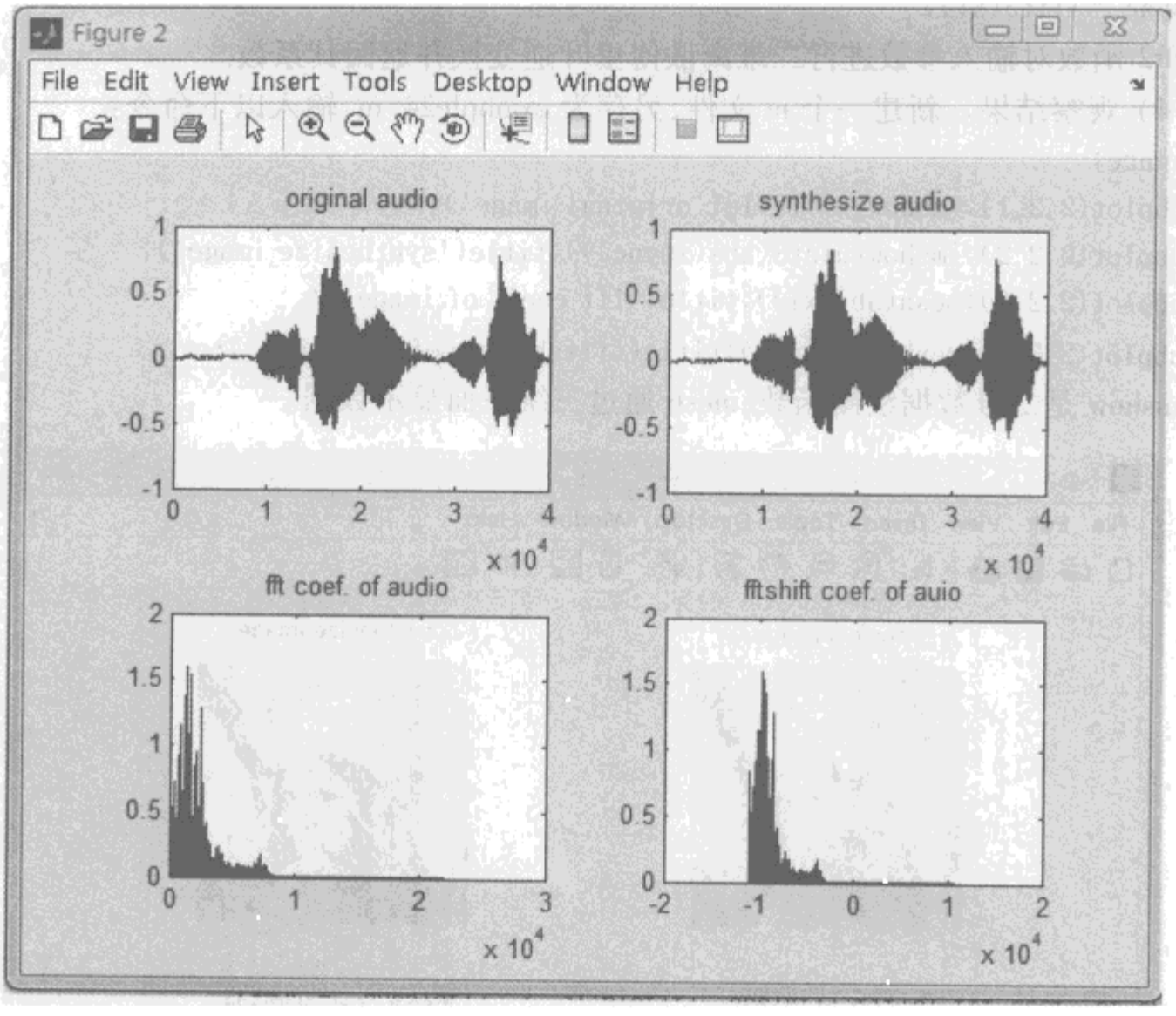

图 1.6　用离散余弦变换分析合成音频文件

（2）一维离散余弦逆变换。新建一个 m 文件，另存为 example32. m，输入以下命令：

```
xsync =idct(xf);
[row,col]=size(x);
xff=zeros(row,col);
xff(1:row,1:col)=xf(1:row,1:col);
y=idct(xff);
```

idct 函数对输入参数进行一维离散余弦逆变换并返回其系数。离散余弦变换常用于图像压缩，可以尝试只使用部分系数重构语言，通过观察可以发现，原始音频和合成后音频两者差别不大。

（3）观察结果。新建一个 m 文件，另存为 example33. m，输入以下命令：

```
figure;
subplot(2,2,1);plot(x);title('original audio');
subplot(2,2,2);plot(xsync);title('synthesize audio');
subplot(2,2,3);plot(f1,abs(xf));title('fft coef. of audio');
subplot(2,2,4);plot(f2(1:len),abs(xff));title('fftshift coef. of auio');
```

2）分析合成图像文件（图 1.7）

① 读取图像文件数据；

② 二维离散余弦变换；

③ 二维离散余弦逆变换；

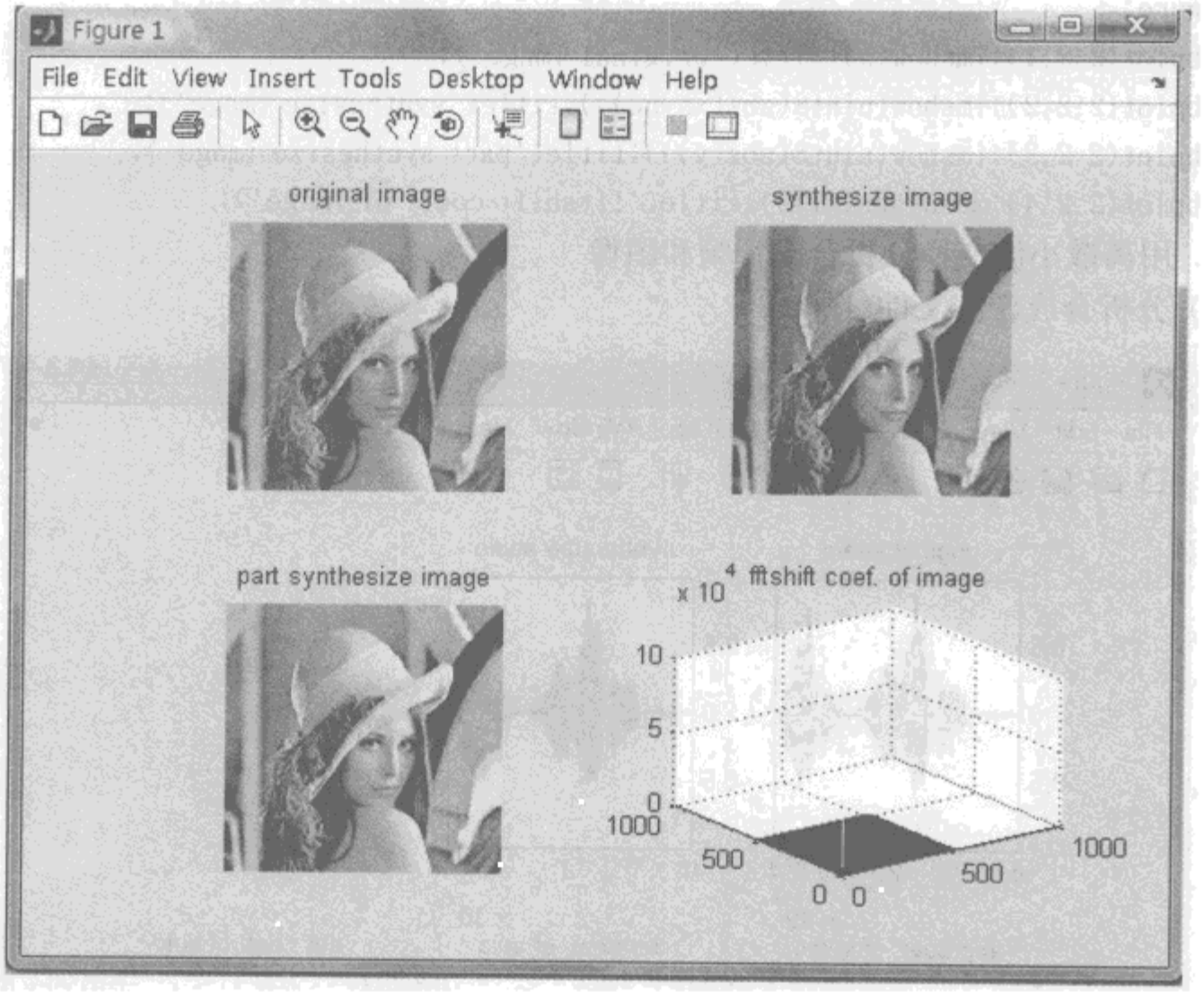

图 1.7　用离散余弦变换分析合成图像文件

④ 观察结果。

（1）二维离散余弦变换。新建一个 m 文件，另存为 example41.m，输入以下命令：

```
xf =dct2(I);
```

dct2 函数对输入参数进行二维离散余弦变换并返回其系数。

（2）二维离散余弦逆变换。新建一个 m 文件，另存为 example42.m，输入以下命令：

```
xsync =uint8(idct2(xf));
[row,col] =size(I);
lenr =round(row * 4 / 5);
lenc =round(col * 4 / 5);
xff =zeros(row,col);
xff(1:lenr,1:lenc) =xf(1:lenr,1:lenc);
y =uint8(idct2(xff));
```

idct2 函数对输入参数进行二维离散余弦逆变换并返回其系数。可以尝试使用部分系数重构图像，本例中使用了系数矩阵中 4/5 的数据，其他部分置零。为了保证图像能正确显示，使用 uint8 对重构图像原始数据进行了数据类型转换，确保其取值范围在 0 ~ 255 之间。

（3）观察结果。请输入命令显示四个子图，分别是原始图像、使用全部系数恢复的图像、使用部分系数恢复的图像和用三维立体图方式显示系数。

新建一个 m 文件，另存为 example43.m，输入以下命令：

```
figure;
subplot(2,2,1);imshow(x);title('original image');
subplot(2,2,2);imshow(uint8(abs(xsync)));title('synthesize image');
subplot(2,2,3);imshow(uint8(abs(y)));title('part synthesize image');
subplot(2,2,4);mesh(abs(xff));title('fftshift coef. of image');
```

3. 用离散小波变换分析合成音频和图像

1）分析合成音频文件(图 1.8)

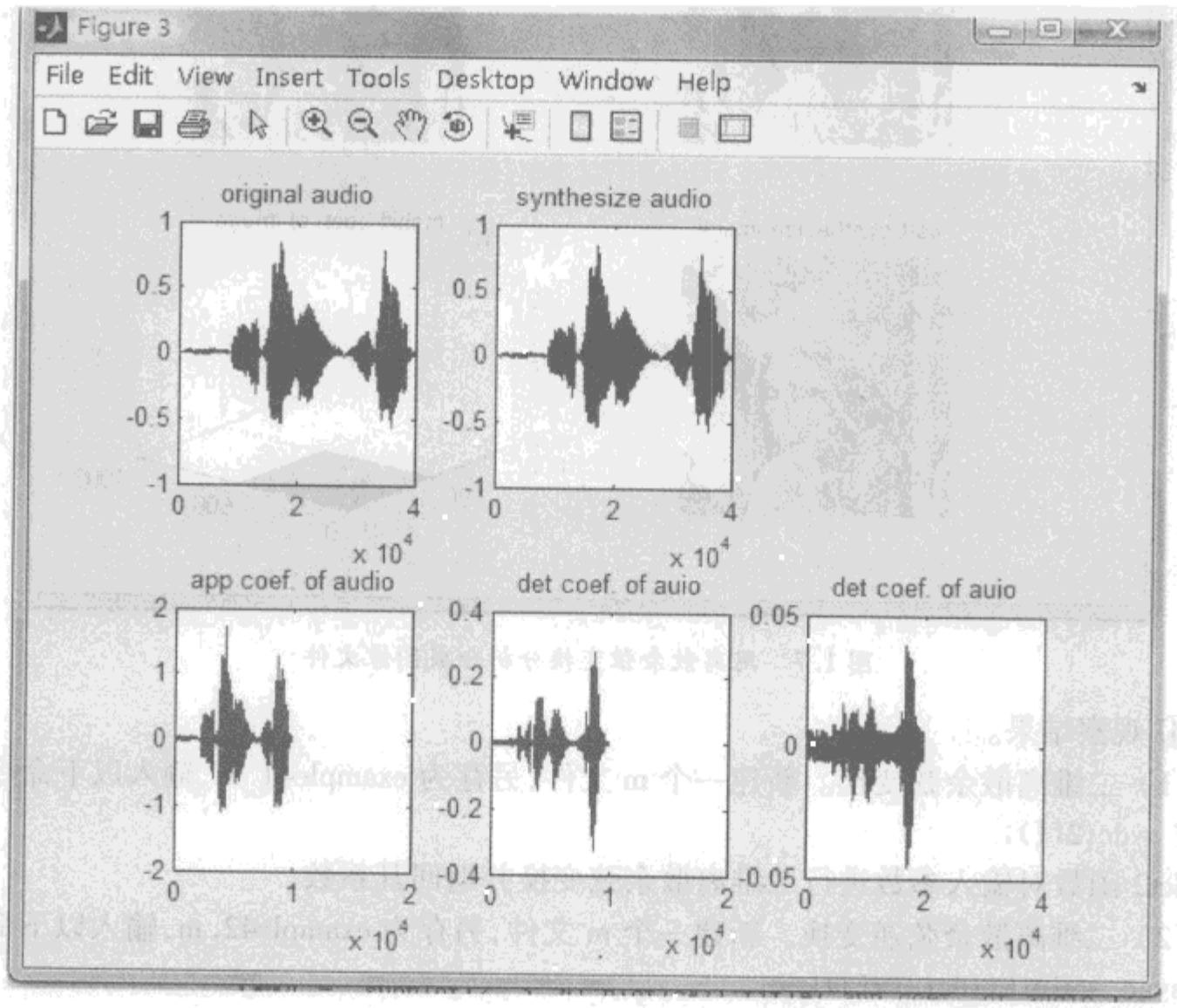

图 1.8　用离散小波变换分析合成音频文件

① 读取音频文件数据；

② 一维离散小波变换；

③ 一维离散小波逆变换；

④ 观察结果。

(1) 一维离散小波变换。新建一个 m 文件,另存为 example51. m,输入以下命令：

```
[C,L] = wavedec(x,2,'db4');
```

wavedec 函数对输入参数进行一维离散小波变换并返回其系数 C 和各级系数长度 L。第二个参数指明小波变换的级数,第三个参数指明小波变换使用的小波基名称。

(2) 一维离散小波逆变换。新建一个 m 文件,另存为 example52. m,输入以下命令：

```
xsync = waverec(C,L,'db4');
cA2 = appcoef(C,L,'db4',2);
cD2 = detcoef(C,L,2);
```

```
cD1 = detcoef(C,L,1);
```

waverec 函数对输入参数进行一维离散小波逆变换并返回其系数。appcoef 函数返回小波系数近似分量，第一个参数 C、第二个参数 L 是 wavedec 的返回参数，为各级小波系数和其长度，第三个参数指明小波基名称，第四个参数指明级数。detcoef 函数返回小波系数细节分量，第一个参数 C、第二个参数 L 是 wavedec 的返回参数，为各级小波系数和其长度，第三个参数指明级数。

（3）观察结果。新建一个 m 文件，另存为 example53. m，输入以下命令：

```
figure;
subplot(2,3,1);plot(x);title('original audio');
subplot(2,3,2);plot(xsync);title('synthesize audio');
subplot(2,3,4);plot(cA2);title('app coef. of audio');.
subplot(2,3,5);plot(cD2);title('det coef. of auio');
subplot(2,3,6);plot(cD1);title('det coef. of auio');
```

2）分析合成图像文件(图 1.9)

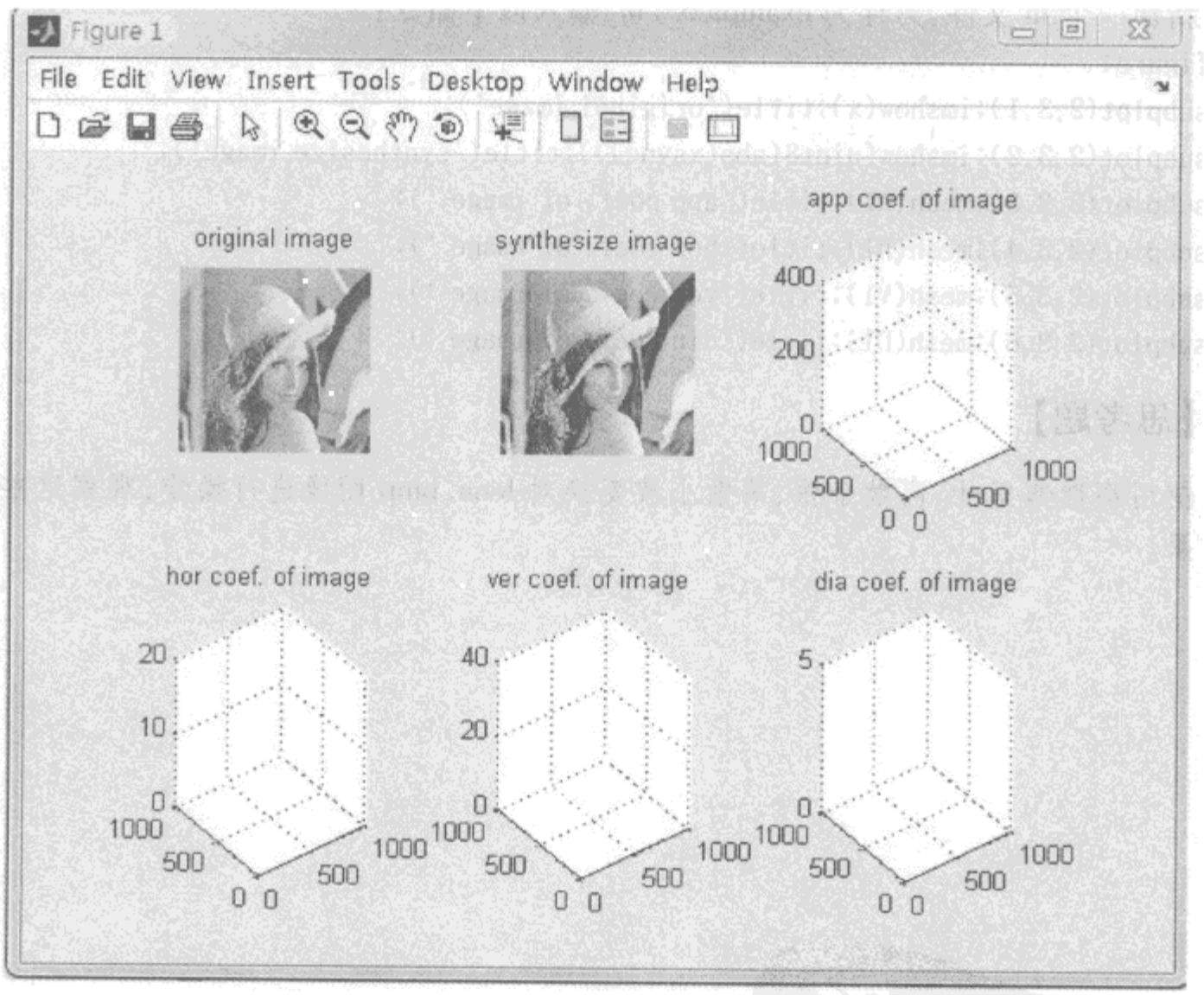

图 1.9　用离散小波变换分析合成图像文件

① 读取图像文件数据；

② 二维离散小波变换；

③ 二维离散小波逆变换；

④ 观察结果。

（1）二维离散小波变换。新建一个 m 文件，另存为 example61. m，输入以下命令：

```
sx =size(I);
[cA1,cH1,cV1,cD1] =dwt2(I,'bior3.7');
```

dwt2 函数对输入参数进行二维一级离散小波变换并返回近似分量、水平细节分量、垂直细节分量和对角线细节分量、如果要对图像进行多级小波分解,使用 wavedec2 函数。

(2) 二维离散小波逆变换。新建一个 m 文件,另存为 example62.m,输入以下命令:

```
xsync =uint8(idwt2(cA1,cH1,cV1,cD1,'bior3.7',sx));
A1 =uint8(idwt2(cA1,[],[],[],'bior3.7',sx));
H1 =uint8(idwt2([],cH1,[],[],'bior3.7',sx));
V1 =uint8(idwt2([],[],cV1,[],'bior3.7',sx));
D1 =uint8(idwt2([],[],[],cD1,'bior3.7',sx));
```

idwt2 函数对输入参数进行二维离散小波逆变换并返回其系数。可以尝试仅使用近似分量、水平细节分量、垂直细节分量或对角线细节分量重构图像。

(3) 观察结果。输入命令显示六个子图,分别是原始图像、使用全部系数恢复的图像、小波系数近似分量、水平细节分量、垂直细节分量和对角线细节分量。

新建一个 m 文件,另存为 example63.m,输入以下命令:

```
figure;
subplot(2,3,1);imshow(x);title('original image');
subplot(2,3,2);imshow(uint8(abs(xsync)));title('synthesize image');
subplot(2,3,3);mesh(A1);title('app coef. of image ');
subplot(2,3,4);mesh(H1);title('hor coef. of image ');
subplot(2,3,5);mesh(V1);title('ver coef. of image ');
subplot(2,3,6);mesh(D1);title('dia coef. of image ');
```

【思考题】

使用离散傅里叶、离散余弦、离散小波变换对 lena.bmp 图像进行操作,观察这些变换的结果。