第 3 章　MATLAB 弹出窗口定义

MATLAB 弹出窗口有很多，主要方便用户进行程序调试状态查看，例如报错窗口，能够较迅速地获取程序运行到哪个地方报错了；文本打开窗口用于提示用户打开哪一个文件进行操作，因此 MATLAB 弹出窗口定义的学习显得尤为必要。

3.1　文件打开操作 uigetfile

MATLAB 文件打开操作命令为 uigetfile，用户可以根据该命令进行文件打开操作，具体的 uigetfile 命令的使用帮助如下：

```
>> help uigetfile
uigetfile - Open standard dialog box for retrieving files

   This MATLAB function displays a modal dialog box that lists files in the
current
   folder and enables you to select or enter the name of a file.

   filename = uigetfile
   [FileName,PathName,FilterIndex] = uigetfile(FilterSpec)
   [FileName,PathName,FilterIndex] = uigetfile(FilterSpec,DialogTitle)
   [FileName,PathName,FilterIndex]                                    =
uigetfile(FilterSpec,DialogTitle,DefaultName)
   [FileName,PathName,FilterIndex]                                    =
uigetfile(...,'MultiSelect',selectmode)
```

MATLAB 提供 5 种 uigetfile 打开功能，每一种打开方式适应一类打开模式，此 5 类打开方式能满足用户所有打开文件操作的需求，下面将对 uigetfile 函数具体的用法进行详细介绍。

（1）直接使用 uigetfile 函数，即在 MATLAB 命令窗口输入 uigetfile 函数，如图 3-1 所示。

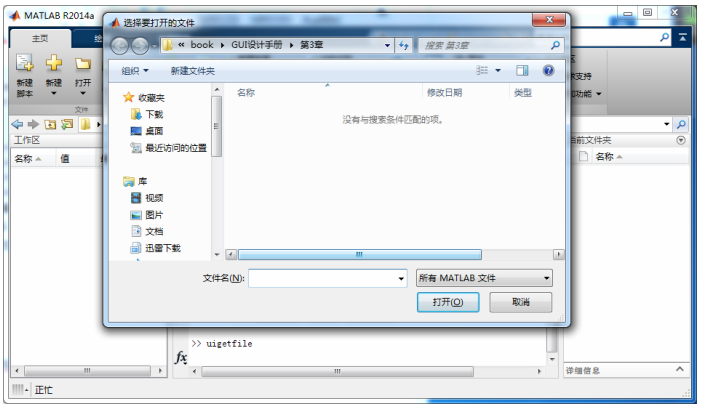

图 3-1　uigetfile 打开操作

如图 3-1 所示，直接在 MATLAB 命令行窗口输入 uigetfile 命令，弹出一个文件选择窗口，默认文件格式为所有 MATLAB 文件格式，并且默认路径为当前路径，用户可以在这个窗口下进行路径选择，并在设定好的路径下选择相应的要打开的文件。

（2）用 uigetfile 函数打开指定格式的文件，方便用户快速打开文件，例如打开.m 文件，具体的操作如图 3-2 所示。

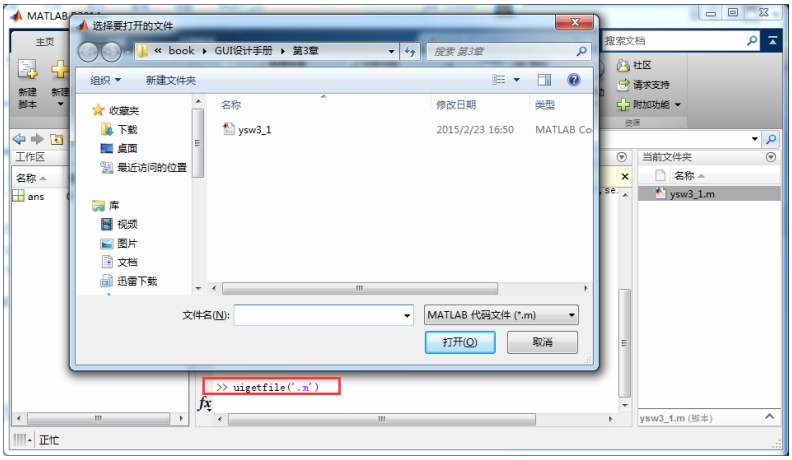

图 3-2　m 文件打开操作

如图 3-2 所示，在命令窗口输入 uigetfile('.m')，则弹出默认为.m 文件的打开界面，在当前路径下，ysw3_1.m 是唯一一个以.m 为后缀的文件，则用户可以很快速地选择。

当用户单击【打开】按钮，则返回打开的文件的文件名，具体如下：

```
>> uigetfile('.m')  % 选择.m 文件

ans =

ysw3_1.m
```

用户也可以对打开界面提示的文件进行修改，如图 3-2 中的【选择要打开的文件】为默认说明，用户也可以进行修改，具体操作如图 3-3 所示。

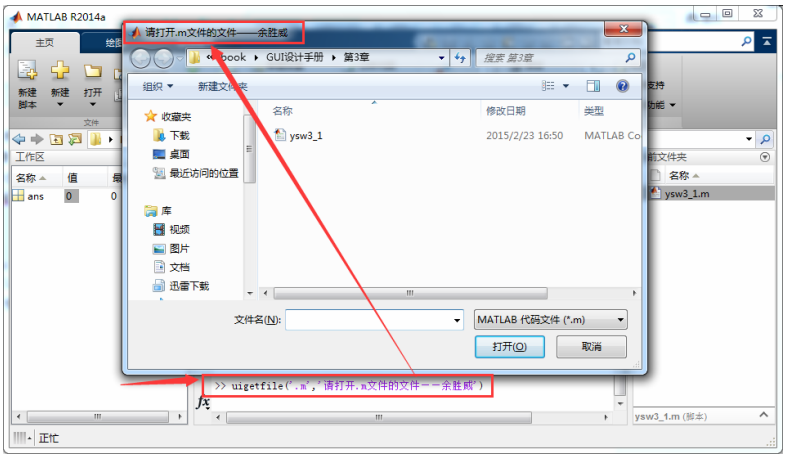

图 3-3　提示文字修改操作

具体的代码如下：

```
>> uigetfile('.m','请打开.m 文件的文件——余胜威')

ans =

ysw3_1.m
```

能够获取文件名称后，那么能不能继续获取打开的文件所在的路径呢？如何获取打开的文件所在的路径呢？具体的方法如下：

```
>> [filename, pathname] = uigetfile('.m','请打开.m 文件的文件——余胜威')

filename =

ysw3_1.m


pathname =

F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章\
```

（3）多种后缀文件进行选择性的打开操作。我们知道，不同格式的文件有很多，怎么样去选择打开多种不同后缀的文件呢？uigetfile 函数提供了相应的操作功能，具体的使用方法如下：

```
[filename, pathname] = ...
    uigetfile({'*.m';'*.slx';'*.mat';'*.*'},'File Selector');
```

运行程序弹出如图 3-4 所示的窗口，可以进行不同格式文件的选择操作。

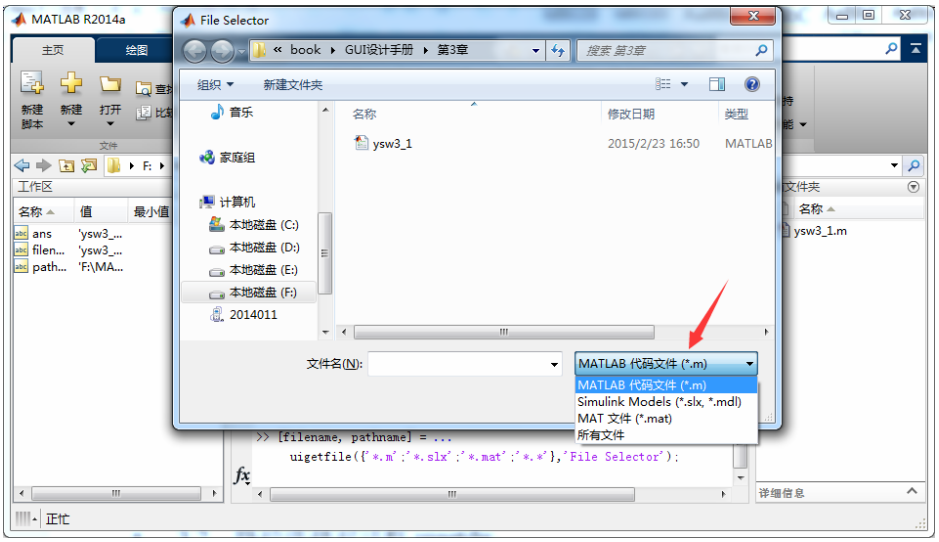

图 3-4　多种格式文件选择

如图 3-4 所示，单击【MATLAB 代码文件（*.m）】，弹出下拉菜单，下拉菜单含有代码指定的几种格式文件，如 Simulink Models（*.slx，*.mdl）文件、MAT 文件（*.mat）以及所有文件（*.*）等，用户选择不同的格式文件，则只显示该格式文件，从而方便用户进行盲操作，即适应不同的文件的操作，从而增强代码的可执行性。

增加不同格式文件的操作，具体如下：

```
[filename, pathname] = uigetfile( ...
{'*.m;*.fig;*.mat;*.slx;*.mdl',...
 'MATLAB Files (*.m,*.fig,*.mat,*.slx,*.mdl)';
   '*.m',  'Code files (*.m)'; ...
   '*.fig','Figures (*.fig)'; ...
   '*.mat','MAT-files (*.mat)'; ...
   '*.mdl;*.slx','Models (*.slx, *.mdl)'; ...
   '*.*',  'All Files (*.*)'}, ...
   'Pick a file');          % 设置显示不同格式的文件，.m 为 MATLAB 脚本文件的后缀名
```

对于不同的图像格式的文件操作，具体如下：

```
uigetfile({'*.jpg;*.tif;*.png;*.gif','All Image Files';...
          '*.*','All Files' ;
         '*.jpg','.jpg 图像' ;
         '*.tif','.tif 图像' ;
         '*.png','.png 图像' ;
         '*.gif','.gif 图像' ;
         '*.bmp','.bmp 图像' ;})    % 设置显示图像后缀名
```

（4）指定路径的文件打开操作。

最简单的操作，就是直接打开某一个文件，例如：

```
[filename, pathname, filterindex] = uigetfile({'*.*','All Files' },
'mytitle',...
         'F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3
         章\ysw3_1.m')
```

运行该文件得到：

```
filename =
ysw3_1.m

pathname =
F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章\

filterindex =
     1
```

打开文件窗口如图 3-5 所示。

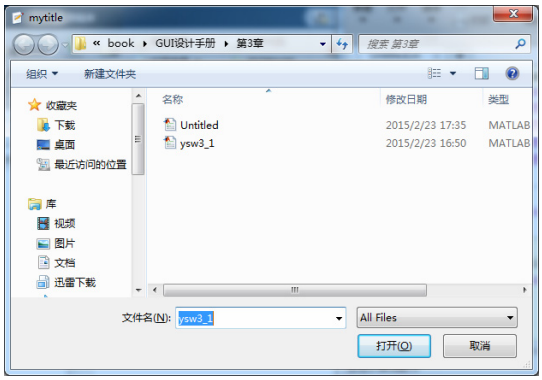

图 3-5　指定文件

如图 3-5 所示，ysw3_1.m 文件被指定，用户只需要单击【打开】按钮即可。

当用户不需要指定具体的文件名，而需要打开某一类型文件时，代码如下：

```
uigetfile('F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册
\','.m')
```

打开文件之后，如何执行该文件呢？MATLAB 有 run 函数可供调用，具体的使用如下：

```
clc,clear,close all   % 清屏和清除工作区
warning off          % 取消警告
[filename, pathname, filterindex] = uigetfile({'*.*','All Files' },
'mytitle',...
        'F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3
        章\ysw3_1.m')
run(filename)        % 执行 filename 文件
```

其中 ysw3_1.m 文件如下：

```
% Designed by Yu Shengwei From SWJTU University
% 2015 年 2 月 23 日
clc,clear,close all          % 清理命令区、清理工作区、关闭显示图形
warning off                  % 消除警告
feature jit off              % 加速代码运行
format short                 % 数据类型
tic                          % 运算计时
ysw = [1,1,7,4,0,6,3,0,8,7];
toc  % 计时结束
```

运行程序之后，得到如图 3-6 所示结果。

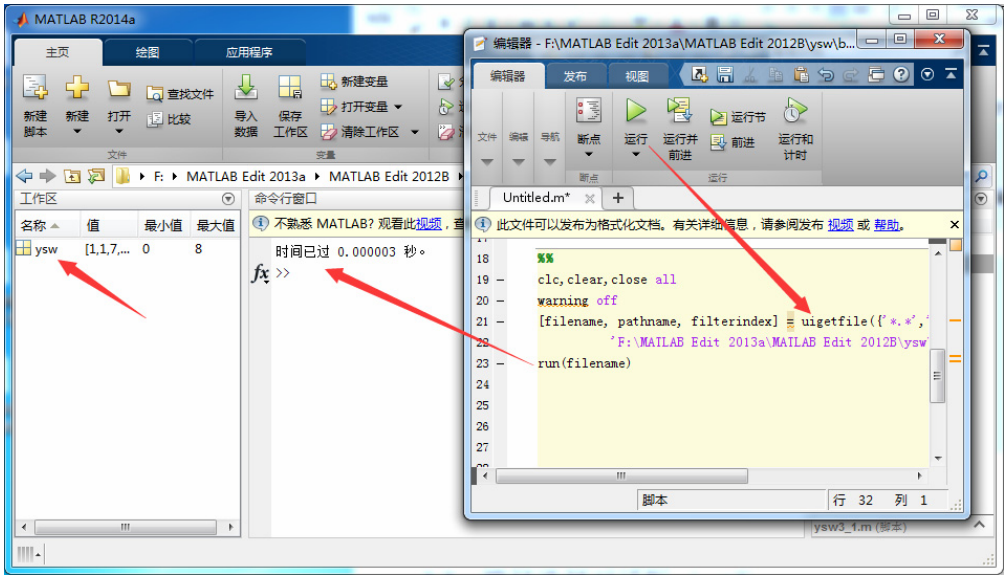

图 3-6　.m 文件执行操作

学习和掌握 uigetfile 函数之后，读者朋友们将更加对 MATLAB 的快捷性和简便性所吸引。uigetfile 函数介绍之后，接下来让我们来学习一下路径选择对话框 uigetdir 函数。

3.2　路径选择对话框 uigetdir

对于路径选择对话框，是文件读取操作过程中的一个环节，MATLAB 单独提供了路径选择对话框 uigetdir 函数，主要应对大数据处理时，实现文件路径的实时切换，以提高程序执行的效率。

MATLAB 中 uigetdir 函数的使用帮助如下：

```
>> help uigetdir
uigetdir - Open standard dialog box for selecting directory

    This MATLAB function displays a modal dialog box enabling you to navigate
the
    folder hierarchy and select a folder or type the name of a folder.

    folder_name = uigetdir
    folder_name = uigetdir(start_path)
    folder_name = uigetdir(start_path,dialog_title)
```

从帮助文档可知，uigetdir 函数是供用户选择路径所用，使用该 uigetdir 函数，会弹出一个路径选择对话框，供用户较快较好地使用。uigetdir 主要分为 3 种使用格式：直接使用、设置路径使用以及修改对话框标题的路径选择对话框。具体的使用如下。

（1）uigetdir 的直接使用。uigetdir 的直接使用，顾名思义就是在 MATLAB 命令行窗口输入 uigetdir 函数，按键盘回车键，即可实现路径选择对话框的选择命令，具体如下：

```
uigetdir    % 直接打开选择文件
```

运行程序，得到如图 3-7 所示结果。

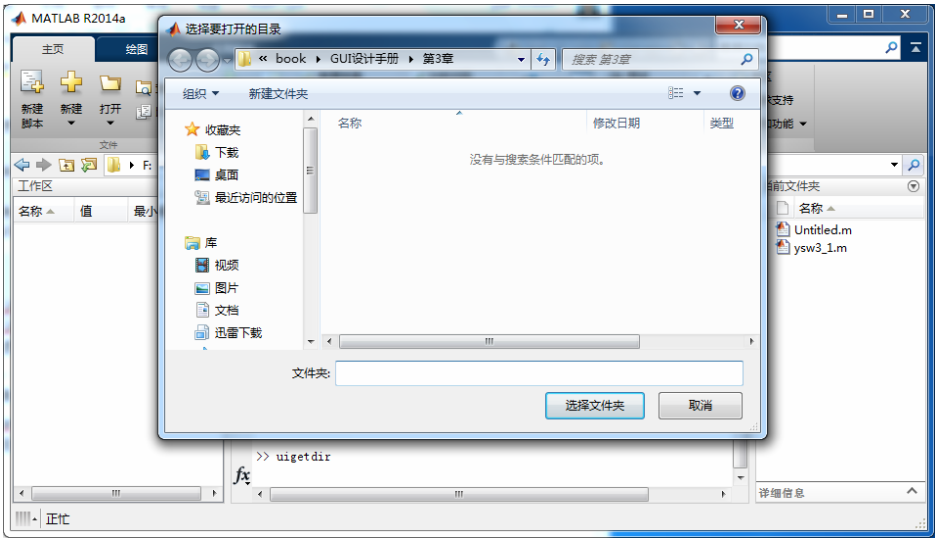

图 3-7　uigetdir 使用

如图 3-7 所示，弹出一个路径选择对话框，标题为【选择要打开的目录】，用户可以

选择数据所在的文件夹即可。

（2）指定某个盘符下的文件夹路径。uigetdir 函数提供可供路径输入的功能，即定位到某一个路径下，用户在该路径下进行文件夹选择，具体的代码如下：

```
clc,clear,close all          % 清理命令区、清理工作区、关闭显示图形
warning off                  % 消除警告
feature jit off              % 加速代码运行
dname = uigetdir('C:\')      % 打开 C 盘文件
```

该程序指定为 C 盘符路径下，用户在 C 盘下进行文件夹选择，从而避免用户回到【我的电脑】界面去选择盘符，从而提高使用效率。

运行程序得到如图 3-8 所示结果。

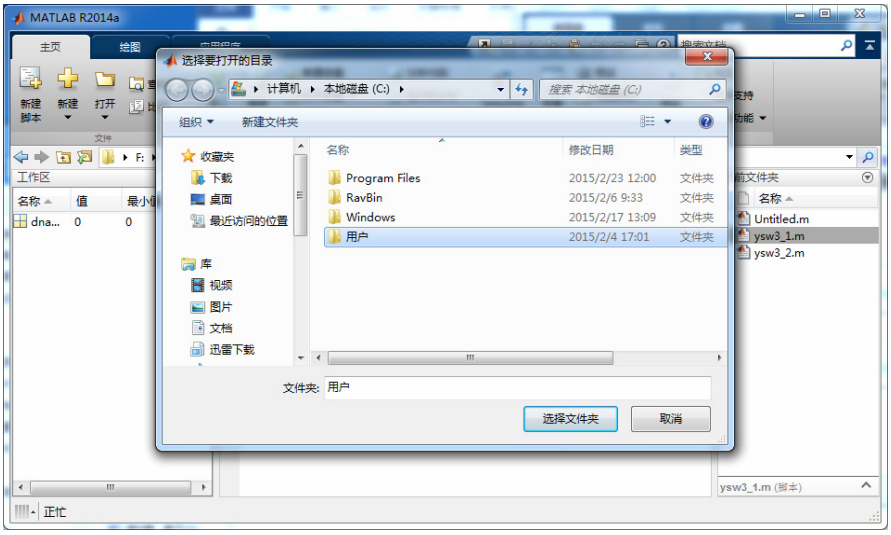

图 3-8　C 盘路径

得到文件夹名称如下：

```
dname =

C:\Users
```

同样，对 C 盘下“用户文件夹”下的文件夹进行读取，该如何操作呢？直接将“用户文件夹”输入为路径即可，程序如下：

```
>> dname = uigetdir('C:\Users\')

dname =

C:\Users\ysw
```

由此可知，采用 uigetdir 在指定路径下进行文件夹选择操作显得较简便易行。

（3）MATLAB 根路径文件夹操作。读取 MATLAB 根目录路径，在根目录路径下进行文件夹操作。MATLAB 提供了简便的根目录函数，具体的使用如下：

```
clc,clear,close all % 清理命令区、清理工作区、关闭显示图形
dname = uigetdir(matlabroot,'MATLAB 根目录路径')
```

运行程序输出图形如图 3-9 所示。

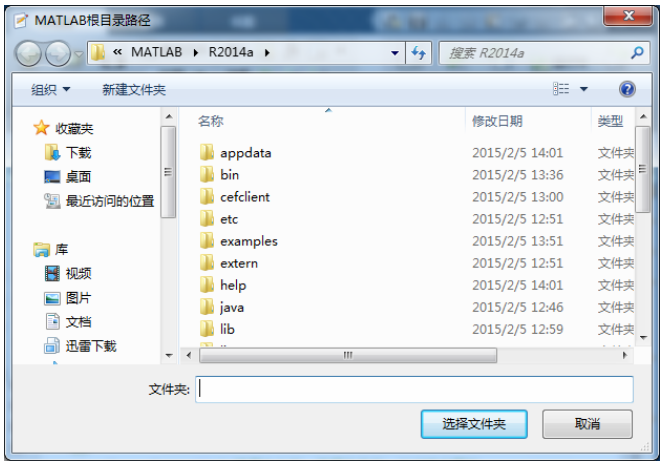

图 3-9　MATLAB 根目录路径

MATLAB 命令行显示根目录路径如下：

```
dname =

C:\Program Files\MATLAB\R2014a\bin
```

由此可知道，该 MATLAB 版本为 2014a，安装在 C 盘 Program Files 文件夹下。

（4）其他命令。对于获取当前工作路径，MATLAB 提供更加简便的路径提取函数进行路径读取，具体的使用如下：

```
>> cd    % 获取当前工作路径

F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章
```

当输入 C 盘盘符时，cd 函数直接将工作路径指向 C 盘路径下，具体如图 3-10 所示。

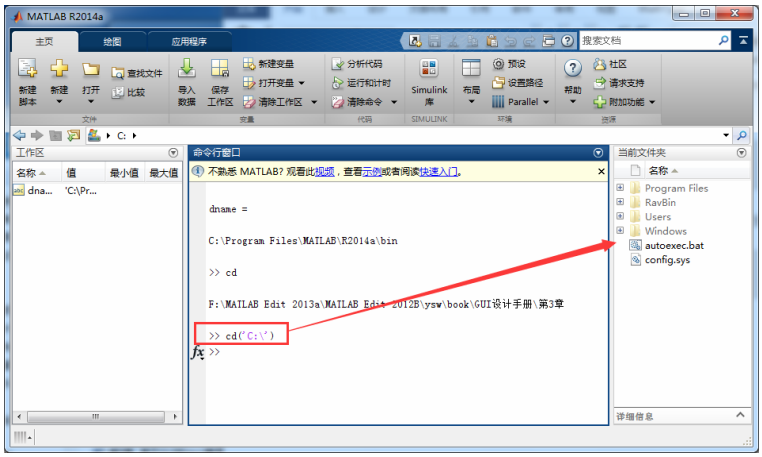

图 3-10　C 盘工作路径

返回到上一级工作路径，程序如下：

```
>> cd('F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章')
```

文件路径选择功能在大数据处理中，应用较广泛，本节比较系统而全面地介绍了路径选择功能，用户完全可在此基础上游刃有余地进行拓展操作。

3.3　文件保存操作 uiputfile

Uiputfile 用于实现文件保存操作，在实际程序应用设计中，时常应用到，具体的 uiputfile 命令含义如下：

```
>> help uiputfile
uiputfile - Open standard dialog box for saving files

    This MATLAB function displays a modal dialog box for selecting or
    specifying a
    file you want to create or save.

    FileName = uiputfile
    [FileName,PathName] = uiputfile
    [FileName,PathName,FilterIndex] = uiputfile(FilterSpec)
    [FileName,PathName,FilterIndex] = uiputfile(FilterSpec,DialogTitle)
    [FileName,PathName,FilterIndex] = uiputfile(FilterSpec,DialogTitle,
    DefaultName)

    uiputfile 的参考页

    另请参阅 save, uigetdir, uigetfile, uisave
```

由 MATLAB 的帮助文档可知，uiputfile 有 5 种使用方式，并且 uiputfile 命令常和 save、uigetdir、uigetfile 和 uisave 等命令连用。

1．使用uiputfile命令返回文件名

直接使用 uiputfile 命令即可返回文件名，具体的使用方法就是直接在 MATLAB 命令行窗口输入 uiputfile 命令，具体如图 3-11 所示。

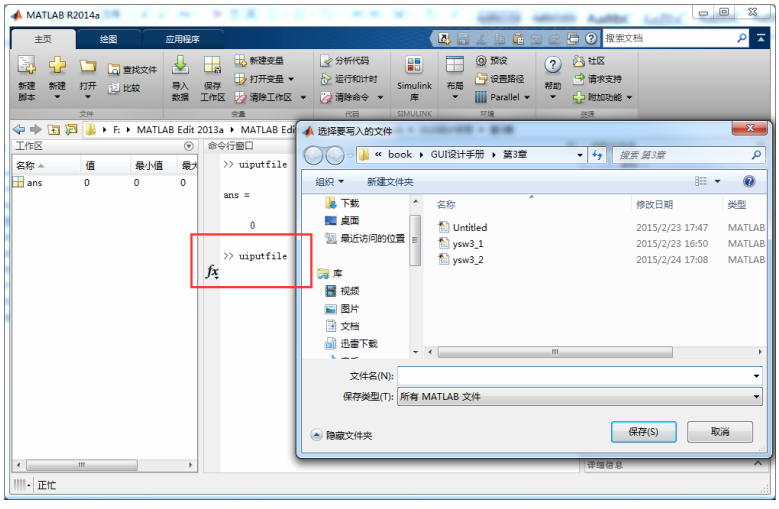

图 3-11　输入 uiputfile 命令并回车

如图 3-11 所示为直接输入 uiputfile 命令并回车的结果，得到当前路径下的文件夹，在此处如果关掉弹出的【选择要写入的文件】窗口，则返回 0，如果输入 ysw3_3，再单击【保存】按钮，则返回如下结果：

```
>> uiputfile   % 输入文件

ans =

ysw3_3.rpt
```

如果输入 ysw3_3.m 则返回 ysw3_3.m，具体如下：

```
>> uiputfile   % 输入文件

ans =

ysw3_3.m
```

由此可知，MATLAB 中 uiputfile 命令默认的文件名后缀为.rpt，需要用户注意。

2．直接输入uiputfile

返回文件名和路径。该命令也是在 MATLAB 命令行直接输入 uiputfile 然后返回文件夹名称以及路径，具体如图 3-12 所示。

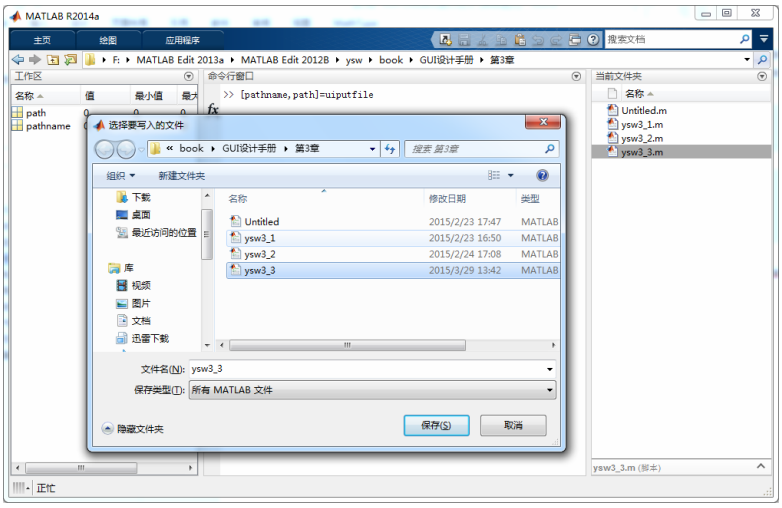

图 3-12　选择存在的 ysw3_3.m 文件

如图 3-12 所示，选择已知的 ysw3_3.m 文件，则返回相应的文件名和路径，具体如下：

```
>> [pathname,path]=uiputfile   %获取输入文件名和路径

pathname =

ysw3_3.m


path =

F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章\
```

```
>>
```

在选中某文件的时候，会提示如图 3-13 所示的对话框，然而这个对话框需要选择【是】，即替换 ysw3_3.m 脚本文件，该替换操作不会改变 ysw3_3.m 脚本文件里面的内容，而仅仅是对文件名称以及文件所在路径的提取。

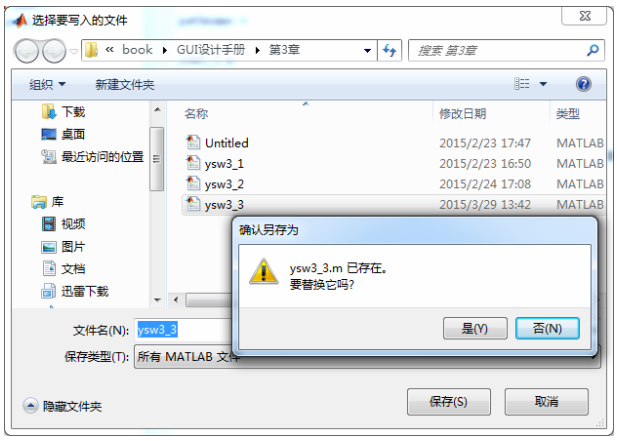

图 3-13　选择提示对话框

如图 3-13 所示，单击【是】按钮，返回文件名称以及文件所在路径。

3．输入要读取的文件的后缀

从而定位文件名和路径，具体的使用如下：

```
[FileName,PathName,FilterIndex] = uiputfile(FilterSpec)  % 定位文件名和路径
```

（1）当不指定文件名的后缀时，可直接输入如下代码：

```
[FileName,PathName,FilterIndex] = uiputfile('')          % 定位文件名和路径
```

运行程序输出图形如图 3-14 所示。

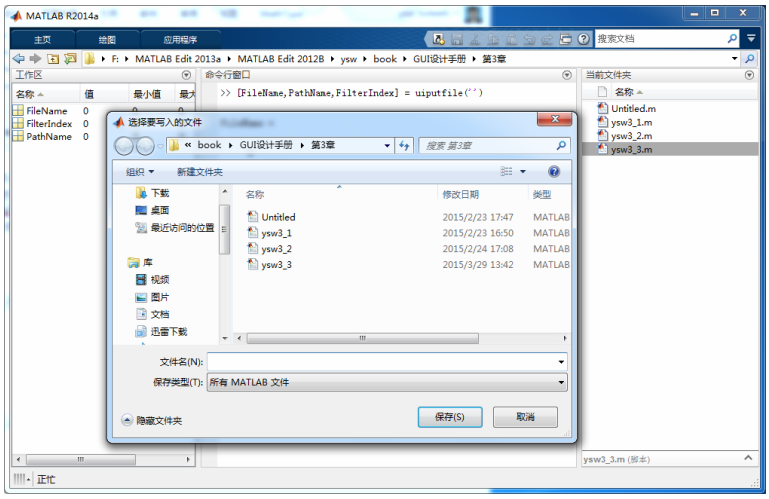

图 3-14　带输入的 uiputfile 使用

当选择 ysw3_3.m 文件时，得到的结果如下：

```
>> [FileName,PathName,FilterIndex] = uiputfile('')    %  定位文件名和路径

FileName =
ysw3_3.m

PathName =
F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章\

FilterIndex =
     1
```

（2）当指定文件名的后缀，例如.mat 文件时，则对话框只提示相应的.mat 文件供用户选择，具体的使用如下：

```
>> [FileName,PathName,FilterIndex] = uiputfile('.mat')
                                                    % 定位，mat 文件名和路径

FileName =
data.mat

PathName =
F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章\

FilterIndex =
     1
```

运行程序输出图形如图 3-15 所示。

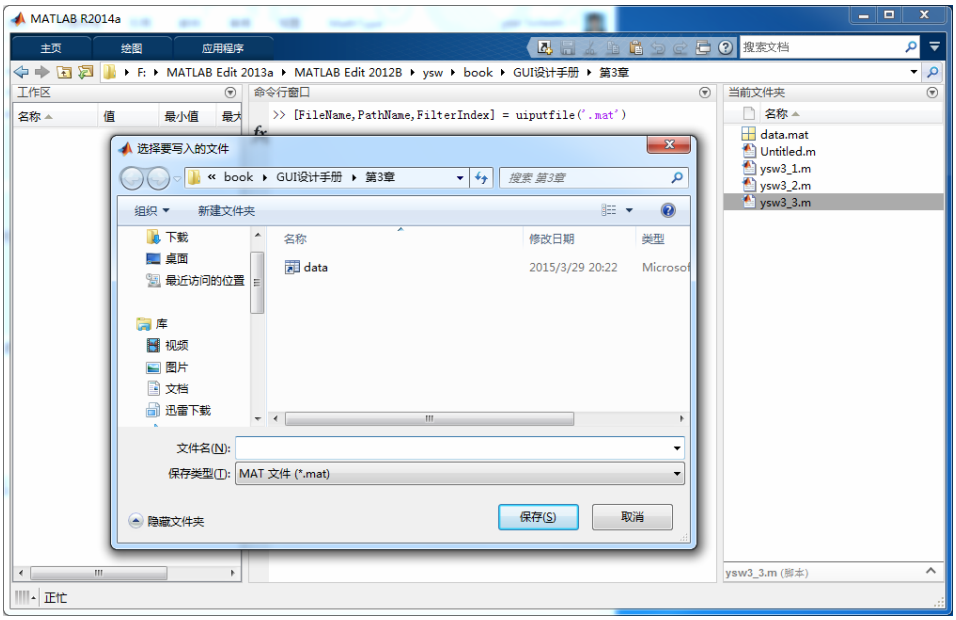

图 3-15　.mat 文件获取

4．采用uiputfile修改提示对话框标题

MATLAB 提供用于修改对话框标题的操作，具体的使用格式如下：

```
[FileName,PathName,FilterIndex] = uiputfile(FilterSpec,DialogTitle)
                                                    % 修改输入文件选择标题
```

其中 DialogTitle 为对话框的标题，具体的使用如下。

（1）采用默认格式，具体如下：

```
[FileName,PathName,FilterIndex] = uiputfile('.mat','')    % 选择.mat 文件
```

具体如图 3-16 所示。

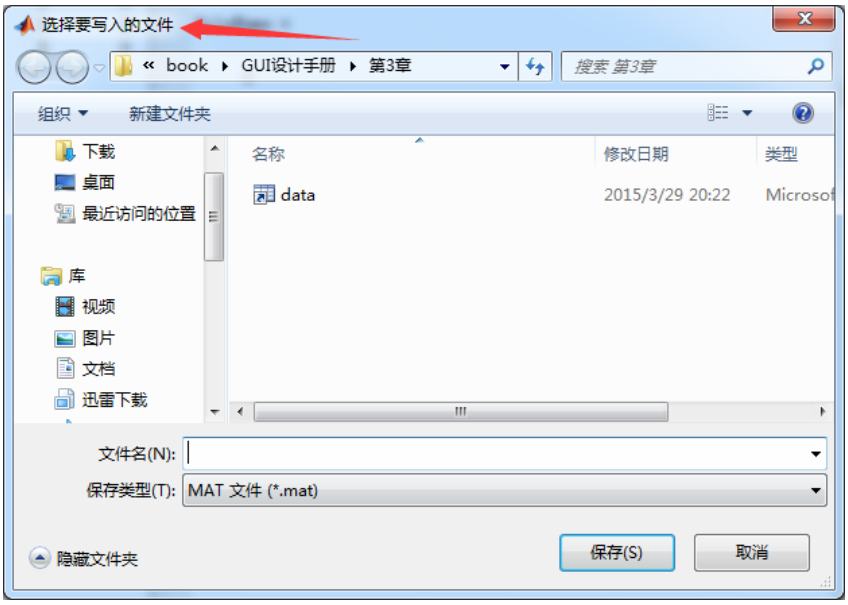

图 3-16　标题“选择要写入的文件”

如图 3-16 可知，MATLAB 默认的对话框标题为【选择要写入的文件】。

（2）当修改标题时，具体操作如下：

```
>> [FileName,PathName,FilterIndex] = uiputfile('.mat','请选择文件')

FileName =
data.mat

PathName =
F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章\

FilterIndex =
     1
```

运行程序输出图形如图 3-17 所示。

5．指定uiputfile函数要选择的文件名

具体的使用格式如下：

```
  [FileName,PathName,FilterIndex] = uiputfile(FilterSpec,DialogTitle,
  DefaultName) % 选择指定的文件
```

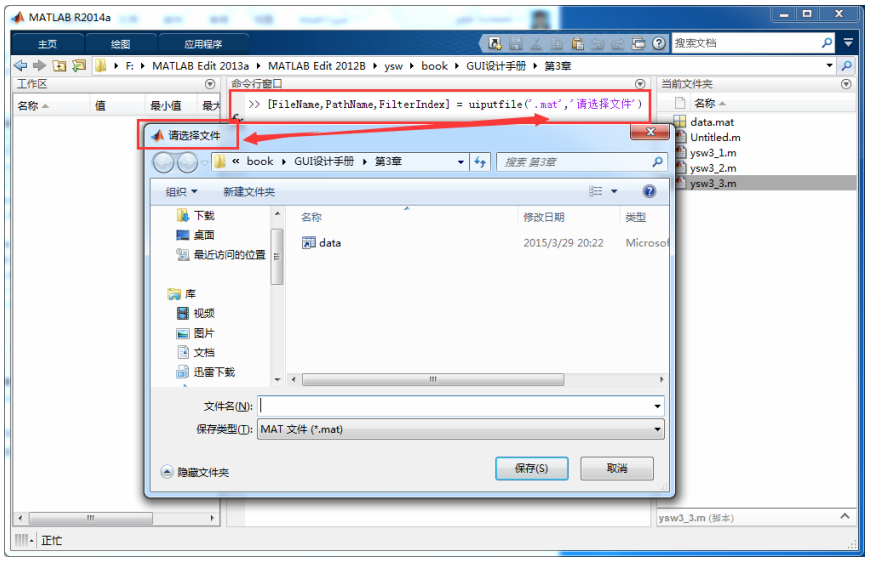

图 3-17　标题修改为“请选择文件”

DefaultName 为要选择的文件的文件名，用户可以直接写入选择对话框，具体的使用格式如下：

```
>> [FileName,PathName,FilterIndex] = uiputfile('.m','请选择文件','ysw3_3')

FileName =
ysw3_3.m

PathName =
F:\MATLAB Edit 2013a\MATLAB Edit 2012B\ysw\book\GUI 设计手册\第 3 章\

FilterIndex =
     1
```

运行程序输出对话框如图 3-18 所示。

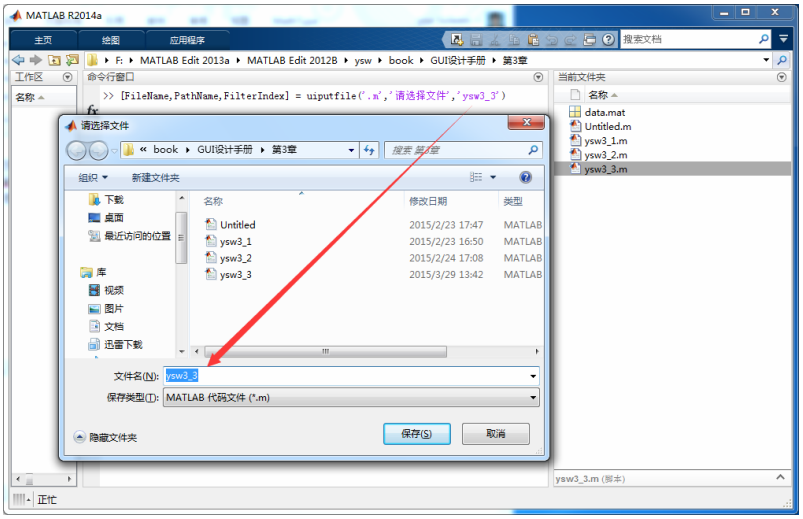

图 3-18　指定文件名名称

如图 3-18 所示，选择对话框直接写入文件名名称“ysw3_3”，用户此时不需要用鼠标选中某一个文件，直接单击【保存】按钮即可。

MATLAB 提供 uiputfile 函数供用户使用，用户可以通过指定某个文件名，然后执行，从而达到程序的高效率执行，增强人机互动性能。

3.4　程序运行进度条 waitbar

waitbar 函数为 MATLAB 程序运行进度条，也就是说，使用 waitbar 函数，可以看到程序的运行进度，从而增强可视化效果，让用户了解运行的进度。

具体的 waitbar 函数使用如下：

```
>> help waitbar
waitbar - Open or update wait bar dialog box

    This MATLAB function displays a wait bar of fractional length x.

    h = waitbar(x,'message')
    waitbar(x,'message','CreateCancelBtn','button_callback')
    waitbar(x,'message',property_name,property_value,...)
    waitbar(x)
    waitbar(x,h)
    waitbar(x,h,'updated message')

    waitbar 的参考页

    另请参阅 close, delete, dialog, getappdata, msgbox, setappdata

>>
```

waitbar 函数能够增加用户对于程序执行位置的判断，具体的函数使用有 6 种模式，以下一一进行介绍。

（1）采用图形句柄，进行进度查看，具体的使用格式如下：

```
h = waitbar(x,'message')     % 程序运行状态显示
```

其中 h 为图形句柄，x 为 0~1 之间，message 为显示信息，具体使用如下：

```
>> h = waitbar(1,'message')   % 程序运行状态显示

h =

    0.0016
```

运行程序输出结果如图 3-19 所示。

如图 3-19 所示，运行程序得到红色的对话框，进度对话框显示为“message”，x=1，在这一步运算中，waitbar 直接为运行完毕，运行时间为 0.0016 秒。

例如修改运行提示信息，具体代码如下：

```
h = waitbar(1,'程序运行中')
```

运行程序输出结果如图 3-20 所示。

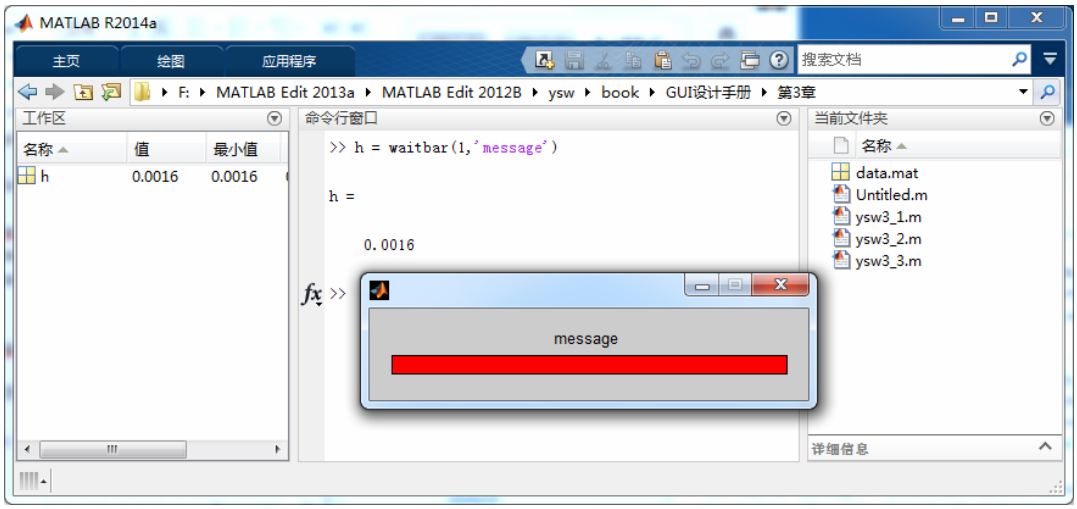

图 3-19　waitbar 使用模式

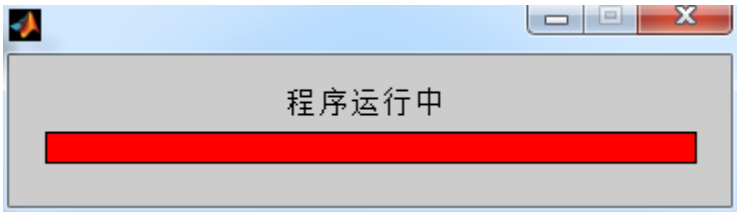

图 3-20　“程序运行中”提示框

对于如图 3-19~图 3-20 中的程序运行对话框，可用程序关闭图形句柄 h，如下：

```
>> close(h)  % 关闭 h
```

close 函数表示关闭图形句柄 h。

同样，MATLAB 还提供 delete 函数删除 h 句柄，释放 MATLAB 执行内存，具体的使用如下：

```
delete(h)   % 删除 h
```

为了动态显示 waitbar 进度显示效果，采用如下程序进行执行显示，具体代码如下：

```
clc,clear,close all  % 清屏和清理工作区
warning off          % 取消警告
h = waitbar(0,'Please wait...');  % 提示等待
steps = 1000;   % 计算次数
for step = 1:steps
    % 程序运行状态进度
    waitbar(step / steps)    % 程序执行状态，即执行位置
end
close(h)
```

运行程序可看到进度条的动态显示过程，如图 3-21~图 3-22 所示。

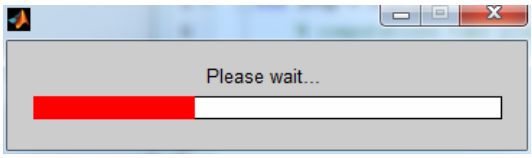

图 3-21　进度条状态 1

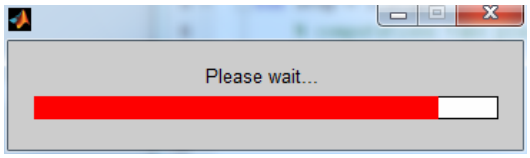

图 3-22　进度条状态 2

（2）wairbar 对话框提供取消运行按钮，具体的使用如下：

```
waitbar(x,'message','CreateCancelBtn','button_callback')  % 增加取消按钮
```

CreateCancelBtn 为取消按钮，button_callback 为取消该 waitbar 函数执行状态。可以关闭 waitbar 进度条，也可以删除 waitbar 进度条，具体的使用如下：

```
>> h=waitbar(0.1,'message','CreateCancelBtn','delete_h(h)')
```

相应的取消函数程序如下：

```
function delete_h(h)  % 删除 h 句柄
delete(h)
```

执行该程序，得到如图 3-23 所示结果。

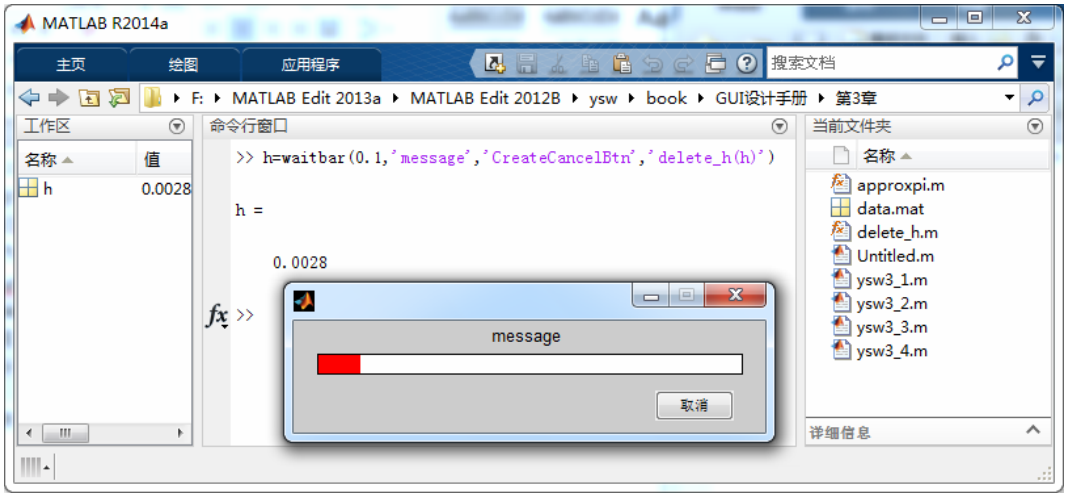

图 3-23　带取消按钮的进度条

如图 3-23 所示，单击【取消】按钮，即可删除该 waitbar 进度条。

也可以不用函数进行删除，而是直接输入，程序如下：

```
>> h=waitbar(0.1,'message','CreateCancelBtn','button_callback')
                                                  % 增加取消按钮 button_callback

h =

    6.0029
```

此时得到相应的对话框如图 3-24 所示。

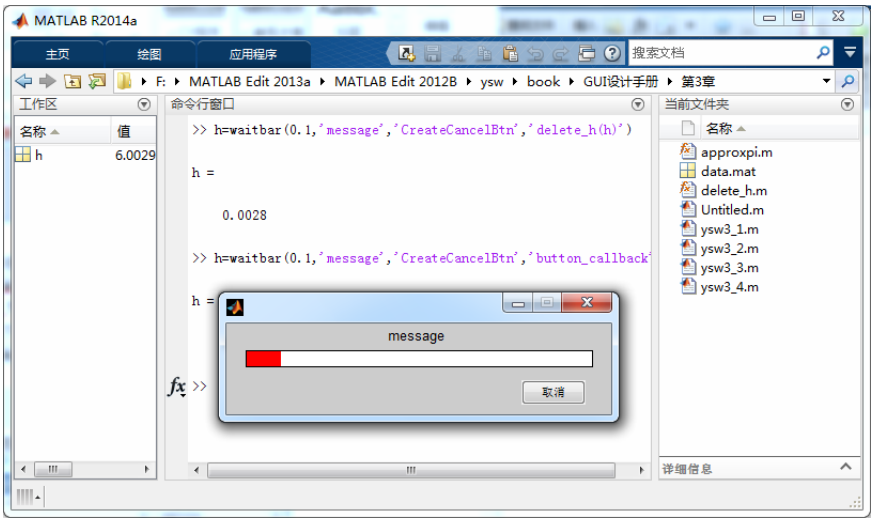

图 3-24　不添加函数的 waitbar 进度条

此时如果单击关闭如图 3-24 的进度条，系统会提示如下错误：

```
未定义函数或变量 'button_callback'。

Error while evaluating figure CloseRequestFcn
```

此时我们没有指定 button_callback 函数，因此会出现报错的提示，然而 waitbar 进度条依然不能关闭，此时应该使用 delete 函数才能关闭该进度条，具体如下：

```
>> delete(h)
```

（3）waitbar 进度条标题修改。

```
waitbar(x,'message',property_name,property_value,...)
```

其中，property_name 为进度条的标题名字，property_value 为标题的名称，具体的使用如下：

```
>> h=waitbar(0.1,'message','name','余胜威')

h =

    6.0032
```

运行程序输出图形如图 3-25 所示。

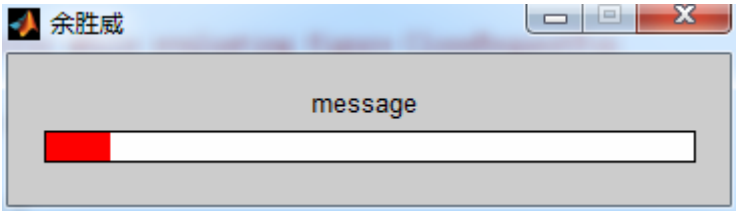

图 3-25　对话框标题修改

（4）直接使用 waitbar 函数。

具体的使用格式如下：

```
waitbar(x)
```

x 为 0~1 之间，具体使用如下：

```
>> waitbar(0.1)    % 默认显示
```

运行程序输出图形如图 3-26 所示。

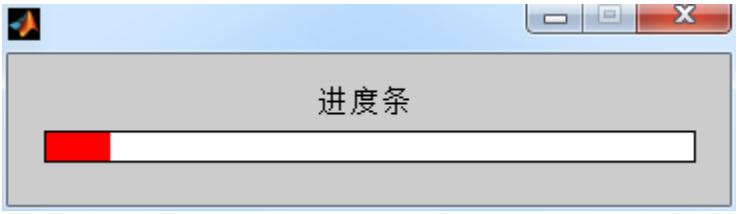

图 3-26　waitbar 默认信息

如图 3-26 所示的 waitbar 进度条信息显示，默认为“进度条”。

（5）修改 waitbar 显示信息。

```
waitbar(x,h)    % 等待
```

具体的使用如下：

```
waitbar(0.1,'2*x');  % 显示为 2*x
```

得到的结果如图 3-27 所示。

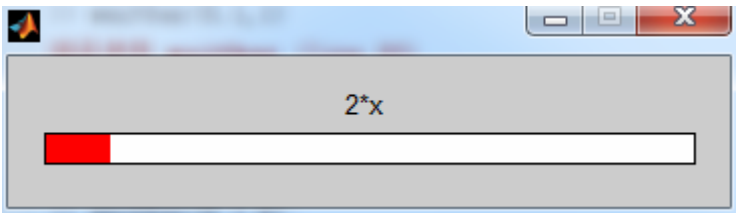

图 3-27　进度条信息修改

（6）更新进度条信息显示值。

```
waitbar(x,h,'updated message')
```

在此显示 pi 的计算值的变化情况，编写函数如下：

```
function [valueofpi step] = approxpi(steps)

h = waitbar(0,'1','Name','Approximating pi...',...
          'CreateCancelBtn',...
          'setappdata(gcbf,''canceling'',1)');
                                        % 程序执行状态设置，增加【取消】按钮
setappdata(h,'canceling',0)             % 是否单击了【取消】按钮
% Approximate as pi^2/8 = 1 + 1/9 + 1/25 + 1/49 + ...  % 逼近结果
pisqover8 = 1;                                         % 初始化 1
denom = 3;                                             % 步长 1/3^2
valueofpi = sqrt(8 * pisqover8);   % pi 值
```

```
for step = 1:steps                    % 步数
    % 是否单击【取消】按钮
    if getappdata(h,'canceling')      % 是否单击了【取消】按钮
        break                         % 暂停
    end
    % Report current estimate in the waitbar's message field
    waitbar(step/steps,h,sprintf('%12.9f',valueofpi))  % 显示计算值
    % Update the estimate
    pisqover8 = pisqover8 + 1 / (denom * denom);       % 更新
    denom = denom + 2;                % 更新，1/9 ，  1/25 ，1/49 ，...
    valueofpi = sqrt(8 * pisqover8);
                              % 估计值，pi^2/8 = 1 + 1/9 + 1/25 + 1/49 + ...
end
delete(h)       % DELETE the waitbar; don't try to CLOSE it
```

相应的调用主程序如下：

```
[estimated_pi steps] = approxpi(10000)
```

运行程序得到如图 3-28~图 3-31 所示的结果。

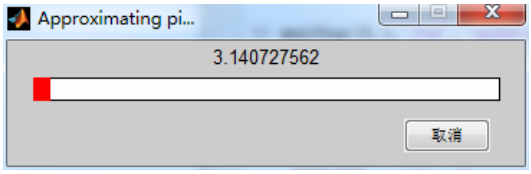

图 3-28　进度条信息更新 1

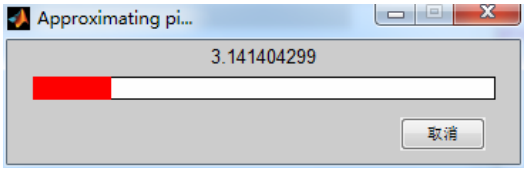

图 3-29　进度条信息更新 2

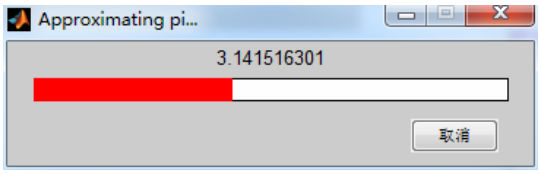

图 3-30　进度条信息更新 3

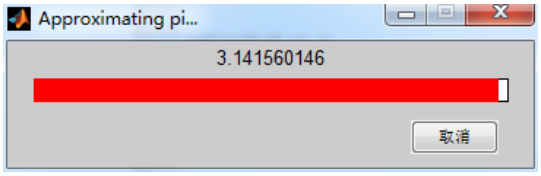

图 3-31　进度条信息更新 4

由此可知，MATLAB 提供的 waitbar 函数使用功能非常丰富，用户可以根据自己的需求进行相应的进度条设计。

3.5　错误提示对话框 errordlg

MATLAB 提供了 errordlg 错误对话框，用于显示程序的错误提示及报警信息，具体的 errordlg 的使用格式如下：

```
>> help errordlg
errordlg - Create and open error dialog box

   This MATLAB function creates and displays a dialog box with title Error
Dialog
   that contains the string This is the default error string.
```

```
    h = errordlg
    h = errordlg(errorstring)
    h = errordlg(errorstring,dlgname)
    h = errordlg(errorstring,dlgname,createmode)

    errordlg 的参考页

    另请参阅 dialog, figure, helpdlg, inputdlg, listdlg, msgbox, questdlg,
uiresume, uiwait, warndlg

>>
```

errordlg 用于提示程序错误信息，对于程序的调试很重要。

（1）errordlg 的直接使用。

MATLAB 提供了 errordlg 的直接使用功能，具体的使用格式如下：

```
>> h=errordlg  % 直接使用

h =

0.0029
```

直接运行 h = errordlg 时，得到如图 3-32 所示结果。

如图 3-32 所示为错误提示，错误窗口界面显示“这是默认错误字符串”，对于该对话框的出现，可以直接单击【确定】按钮关闭该对话框。

（2）修改错误提示字符串。

```
h = errordlg(errorstring)    % 带字符串的错误提示
```

用户在进行程序调试时，可以对程序错误来源进行定性分析，即指定程序可能存在的问题，具体的使用如下：

```
>> h=errordlg('矩阵的维数不同')

h =

    0.0034
```

运行程序得到如图 3-33 所示对话框。

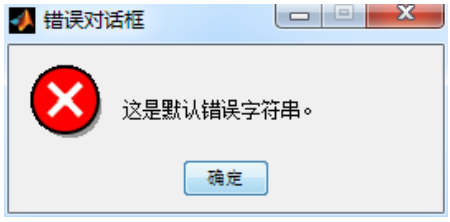

图 3-32　错误提示

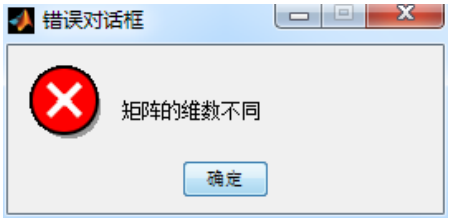

图 3-33　指定错误类型对话框

如图 3-33 所示，对话框提示“矩阵的维数不同”。

当然，用户可以分开输入，具体使用如下：

```
>>  h=errordlg('矩阵的''维数不同')
```

运行程序仍然得到如图 3-33 所示对话框。

如果需要多行显示，则输入：

```
>> errordlg({'程序' '警告'})
```

运行程序得到如图 3-34 所示对话框。

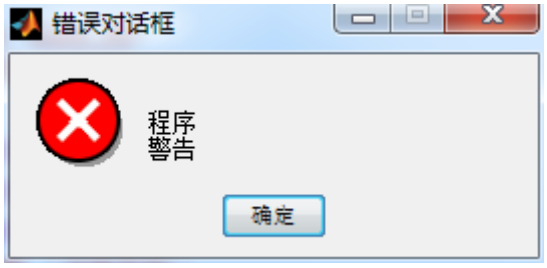

图 3-34　多行显示

（3）指定错误的类型，并修改对话框标题。

```
h = errordlg(errorstring,dlgname)  %指定错误的类型
```

如图 3-32~图 3-34 所示，错误对话框的标题默认为【错误对话框】，在此可以对对话框标题进行修改，具体的使用如下：

```
>> h=errordlg('矩阵的维数不同','余胜威-程序纠错')  %指定错误的类型

h =

    0.0039
```

运行程序得到如图 3-35 所示对话框。

（4）修改对话框信息显示。

```
h = errordlg(errorstring,dlgname,createmode)      % 修改对话框信息显示
```

creatmode 可以为字符串，也可以为一个结构体。如果 creatmode 为字符串，则 creatmode 为 modal、non-modal (default)或 replace 三者之一，具体的使用如下：

```
mode = struct('WindowStyle','non-modal',...
      'Interpreter','tex');                % 显示格式设置
h = errordlg('Try this equation instead: f(x) = x^2',...
                                           % 修改对话框信息显示
    'Equation Error', mode);
```

运行程序得到如图 3-36 所示对话框。

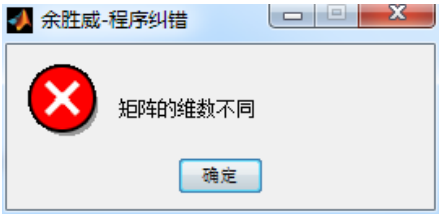

图 3-35　标题修改

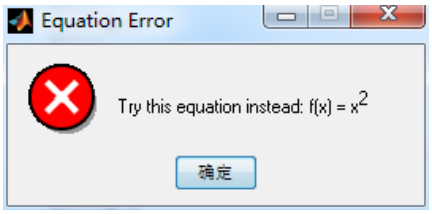

图 3-36　non-modal (default)显示模式

修改 mode 如下：

```
>> mode = struct('WindowStyle','modal',...
      'Interpreter','tex');              % 显示样式设置
>> h = errordlg('Try this equation instead: f(x) = x^2',...
    'Equation Error', mode);             % 修改对话框信息显示
```

运行程序得到如图 3-37 所示对话框。

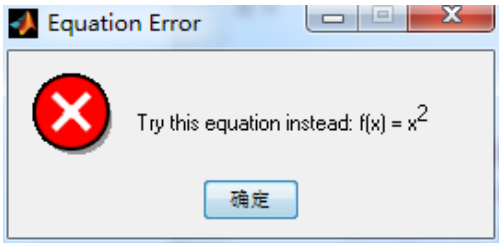

图 3-37　modal 显示模式

由此可知，MATLAB 提供的 errordlg 对话框模式是很丰富的，用户可以在程序中指定可能的错误来源，从而更好地定位程序的 debug 源头。

3.6　警告提示对话框 warndlg

warndlg 警告提示对话框和错误对话框类似，用于警告提示之用。具体的 warndlg 使用格式如下：

```
>> help warndlg
warndlg - Open warning dialog box

    This MATLAB function displays a dialog box named Warning Dialog containing
    the
    string This is the default warning string.

    h = warndlg
    h = warndlg(warningstring)
    h = warndlg(warningstring,dlgname)
    h = warndlg(warningstring,dlgname,createmode)

    warndlg 的参考页

    另请参阅 dialog, errordlg, figure, helpdlg, inputdlg, listdlg, msgbox,
    questdlg, uiresume, uiwait, warning

>>
```

从帮助信息可看出，warndlg 警告提示对话框主要包括 4 种使用方式，以下将一一介绍。

（1）warndlg 的直接使用。

MATLAB 提供了 warndlg 的直接使用功能，具体的使用格式如下：

```
>> h=warndlg    % 直接使用
h =
```

```
    0.0032

>>
```

直接运行 h = warndlg 时，得到如图 3-38 所示结果。

如图 3-38 所示为警告提示对话框，警告提示对话框窗口界面显示“这是默认警告字符串”，对于该对话框的出现，可以直接单击【确定】按钮关闭该对话框。

（2）修改警告提示字符串。

```
h = warndlg(warningstring)  % 修改警告提示字符串
```

用户在进行程序调试时，可以对程序警告来源进行定性分析，即指定程序可能存在的问题，具体的使用如下：

```
>> h = warndlg('程序警告')  % 修改提示字符为"程序警告"

h =
0.0037
```

运行程序得到如图 3-39 所示对话框。

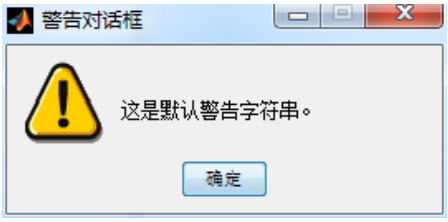

图 3-38　错误提示

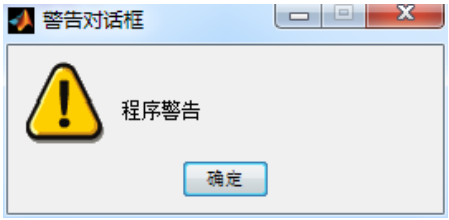

图 3-39　指定警告类型对话框

如图 3-39 所示，对话框提示“程序警告”。

当然，用户可以分开输入，具体使用如下：

```
>> warndlg('程序''警告')
```

运行程序仍然得到如图 3-39 所示对话框。

如果需要多行显示，则输入：

```
warndlg({'程序' '警告'})
```

运行程序得到如图 3-40 所示对话框。

（3）指定警告的类型，并修改对话框标题。

```
h = warndlg (errorstring,dlgname)   % 指定警告的类型
```

如图 3-38~图 3-40 所示，警告提示对话框的标题默认为“警告对话框”，在此可以对对话框标题进行修改，具体的使用如下：

```
>> h = warndlg('程序警告','余胜威的程序设计')

h =
    0.0042
```

运行程序得到如图 3-41 所示对话框。

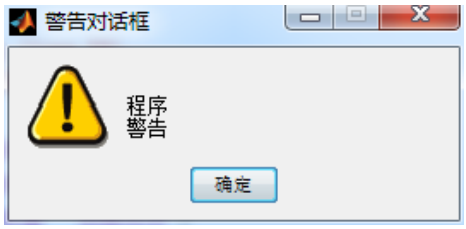

图 3-40　多行显示

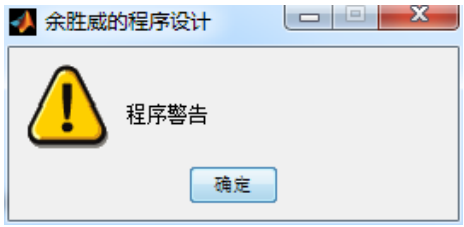

图 3-41　标题修改

（4）修改警告对话框信息显示。

```
h = warndlg(errorstring,dlgname,createmode)
```

creatmode 可以为字符串，也可以为一个结构体。如果 creatmode 为字符串，则 creatmode 为 modal、non-modal (default)或 replace 三者之一，具体的使用如下：

```
mode = struct('WindowStyle','non-modal',...
      'Interpreter','tex');     % 显示样式
>> h = warndlg ('程序警告，请检查',...
'余胜威提示', mode);              % 文字提示
```

运行程序得到如图 3-42 所示对话框。

修改 mode 如下：

```
>> mode = struct('WindowStyle','modal',...
      'Interpreter','tex');          % 显示样式
>> h = warndlg ('程序警告，请检查一下，避免可能的 debug!!!',...
'余胜威提示', mode);                  % 字符串设置
```

运行程序得到如图 3-43 所示对话框。

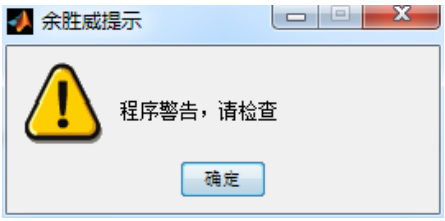

图 3-42　non-modal (default)显示模式

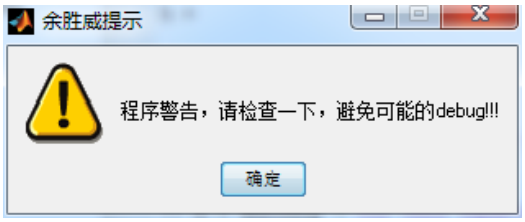

图 3-43　modal 显示模式

由此可知，MATLAB 提供的 warndlg 对话框模式是很丰富的，用户可以在程序中指定可能的警告来源，从而更好地定位程序的 debug 源头。

3.7　用户提示对话框 msgbox

msgbox 为用户提示对话框，用户可以对程序的执行状态进行提示，以至于让用户可以进一步地执行下一步，msgbox 在 gui 设计中常用到。具体的 msgbox 使用格式如下：

```
>> help msgbox
msgbox - Create and open message dialog box

    This MATLAB function creates a message dialog box that automatically wraps
    Message to fit an appropriately sized figure.

    h = msgbox(Message)
    h = msgbox(Message,Title)
    h = msgbox(Message,Title,Icon)
    h = msgbox(Message,Title,'custom',IconData,IconCMap)
    h = msgbox(___,CreateMode)

    msgbox 的参考页

    另请参阅 errordlg, helpdlg, imread, warndlg
```

由此可知，msgbox 有 5 种使用格式，5 种使用格式基本满足用户的设计要求，具体介绍如下。

（1）直接输入提示信息。

msgbox 提示对话框直接输入提示信息，具体使用如下：

```
>> msgbox('测试数据训练完毕')
```

运行程序得到如图 3-44 所示对话框。

当一行不能完好地表示信息时，可以在第二行写入，具体的操作如下：

```
msgbox({'测试数据' '训练完毕'})
```

运行程序得到如图 3-45 所示对话框。

同理，若三行显示，则为：

```
msgbox({'测试数据' '训练' '完毕'})
```

运行程序得到如图 3-46 所示对话框。

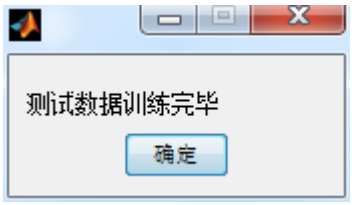

图 3-44　提示信息输入

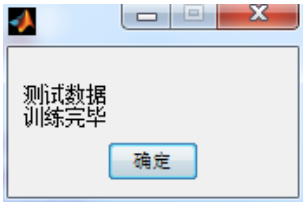

图 3-45　两行显示

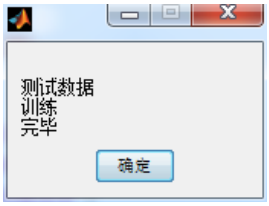

图 3-46　三行显示

（2）修改显示标题。

MATLAB 中 msgbox 对话框的标题可以修改，具体的使用如下：

```
>> h = msgbox('执行成功','余胜威');
```

运行程序得到如图 3-47 所示对话框。

（3）msgbox 显示具体的图标。

Msgbox 还可以显示具体的图标，例如错误图标、警告图标等，其使用格式如下：

```
h = msgbox(Message,Title,Icon)     % 修改显示图标
```

其中 Icon 为提示图标的特殊函数字符，具体使用如下：

```
>> h = msgbox('程序错误', '余胜威','error');
```

运行程序得到如图 3-48 所示对话框。

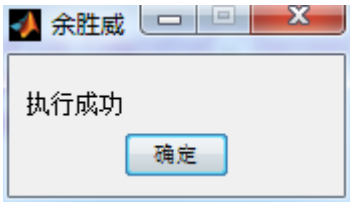

图 3-47　标题修改

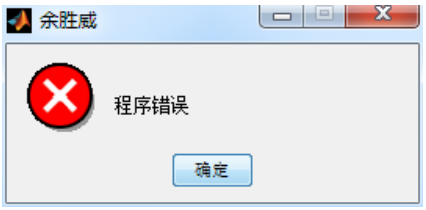

图 3-48　添加错误图标

例如添加警告图标，具体的使用如下：

```
>> h = msgbox('程序错误', '余胜威','warn');    % 添加警告图标
```

运行程序得到如图 3-49 所示对话框。
同理还有帮助图标，具体的使用如下：

```
>> h = msgbox('程序错误', '余胜威','warn');    % 添加帮助图标
```

运行程序得到如图 3-50 所示对话框。

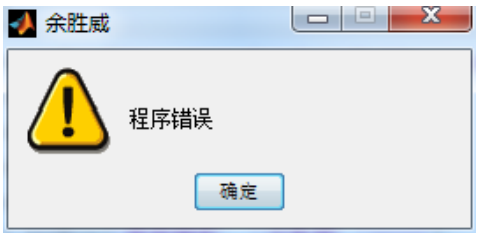

图 3-49　添加警告图标

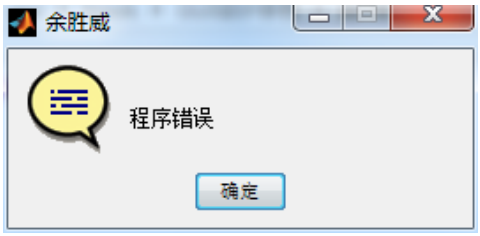

图 3-50　添加帮助图标

（4）设定用户自己的图标。
Msgbox 中用户可以设定自己的图标进行显示，具体的使用格式如下：

```
h = msgbox(Message,Title,'custom',IconData,IconCMap)  % 设定用户自己的图标
```

具体的应用如下：

```
clc,clear,close all   % 清屏
warning off        % 取消警告
feature jit off       % 加速通道
copyfile(fullfile(matlabroot,...
        'help','includes','product',...
        'images','global','ico_large_info.png'));  % 复制图像句柄
[cdata] = imread('DSC06314.jpg');                  % 读入图像
h=msgbox('程序完成',...
        '余胜威','custom',cdata);                  % 显示用户自己设定的图标
```

运行程序得到如图 3-51 所示对话框。

或者直接操作如下：

```
clc,clear,close all
[cdata,map] = imread('DSC06314.jpg');
h=msgbox('余胜威',...
        '执行自己的图标','custom',cdata,map);  % 修改标题
```

运行程序得到如图 3-52 所示对话框。

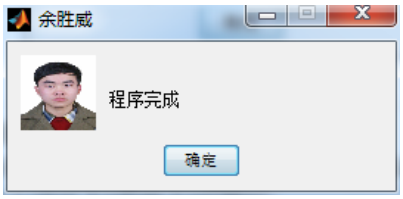

图 3-51　用户自己设定图标

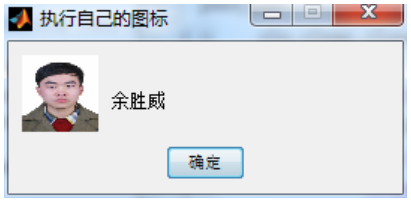

图 3-52　自己设定的图标显示

（5）修改警告对话框信息显示。

```
h = msgbox(___,CreateMode)  % 修改警告对话框
```

creatmode 可以为字符串，也可以为一个结构体。如果 creatmode 为字符串，则 creatmode 为 modal、non-modal (default)或 replace 三者之一，具体的使用如下：

```
uiwait(msgbox('Operation Completed','Success','modal'));
                                % 标题为 Success，显示为 Operation Completed
```

运行程序得到如图 3-53 所示对话框。

修改 mode 如下：

```
clc,clear,close all                                    % 清屏
CreateStruct.Interpreter = 'tex';                      % 显示样式
CreateStruct.WindowStyle = 'modal';                    % 设置样式
h=msgbox('Z = X^2 + Y^2','Value',CreateStruct);        % 显示 Z=X^2+Y^2
```

运行程序得到如图 3-54 所示对话框。

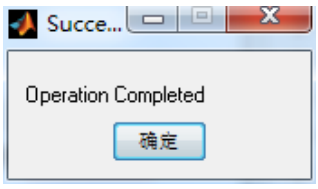

图 3-53　modal 显示模式 1

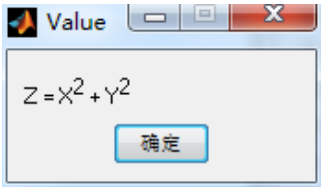

图 3-54　modal 显示模式 2

由此可知，MATLAB 提供的 msgbox 对话框功能是很丰富的，用户可以在程序中指定显示的文字以及图案，从而更好地优化设计界面。

3.8　提问对话框 questdlg

questdlg 为提问对话框，用于用户选择哪个程序进行执行，例如是否继续执行，是否

现在关闭程序等等，具体的提问对话框 questdlg 的使用格式如下：

```
>> help questdlg
questdlg - Create and open question dialog box

    This MATLAB function displays a modal dialog box presenting the question
    'qstring'.

    button = questdlg('qstring')
    button = questdlg('qstring','title')
    button = questdlg('qstring','title',default)
    button = questdlg('qstring','title','str1','str2',default)
    button = questdlg('qstring','title','str1','str2','str3',default)
    button = questdlg('qstring','title', ..., options)

    questdlg 的参考页

    另请参阅 dialog, errordlg, figure, helpdlg, inputdlg, listdlg, msgbox,
    textwrap, uiresume, uiwait, warndlg

>>
```

由此可知，提问对话框 questdlg 有 6 种使用格式，具体如下。

（1）直接使用字符串输出。

直接弹出提示对话框，具体的使用如下：

```
>> button = questdlg('qstring')  % 设置为 qstring
```

运行程序得到如图 3-55 所示界面。

图 3-55　提问对话框

单击【是】按钮，则 button 返回 Yes，单击【否】按钮，则返回 No，单击【取消】按钮，则返回 Cancel，具体如下：

```
>> button = questdlg('qstring')  % 提问对话框
button =
Yes

>> button = questdlg('qstring')
button =
No

>> button = questdlg('qstring')
button =
Cancel
>>
```

（2）修改提问对话框标题。

具体的使用格式如下：

```
>> button = questdlg('qstring','余胜威')  % 设置标题

button =

Yes
```

运行程序得到如图 3-56 所示界面。

（3）设置弹出对话框的按键。

通过设置弹出对话框的按键，可以事先设置该界面上的按钮状态，如图 3-56 所示界面默认为【是】状态，修改默认显示按钮，其具体的使用格式如下：

```
button = questdlg('qstring','title',default)
```

默认设置为【否】，程序如下：

```
>> button = questdlg('qstring','title','No')  % 默认设置为“否”

button =

Cancel
```

运行程序得到如图 3-57 所示界面。

图 3-56　修改提问对话框标题

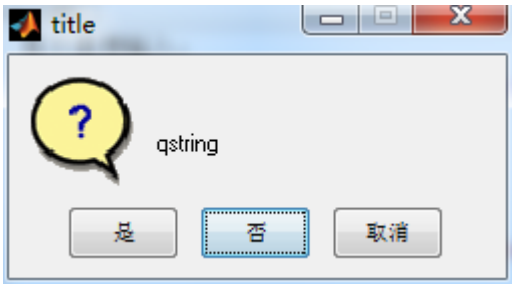

图 3-57　默认对话框选择【否】

例如设定默认为【取消】，则程序如下：

```
>> button = questdlg('qstring','title','Cancel')  % 默认设置为“取消”

button =

Cancel
```

运行程序得到如图 3-58 所示界面。

（4）修改选择按钮的显示字符串。

具体的 MATLAB 调用格式如下：

```
button = questdlg('qstring','title','str1','str2',default)
```

如图 3-58 所示，界面按钮默认为【是】、【否】、【取消】，用户可以对该字符串进行修改，具体如下：

```
>> button = questdlg('提问对话框','余胜威','Yes','No','Cancel')
                                                    % 修改选择按钮的显示字符串

button =

Yes
```

运行程序得到如图 3-59 所示结果。

图 3-58　默认对话框选择【取消】

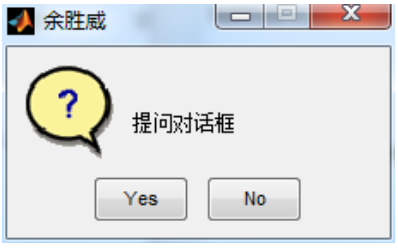

图 3-59　修改显示按钮字符

在代码中，questdlg('提问对话框','余胜威','Yes','No','Cancel')，默认按钮选择为 cancel，但是字符设置为【Yes】、【No】，因此没有选中某一个按钮。如果选择【Yes】按钮，则程序修改为：

```
>> button = questdlg('提问对话框','余胜威','Yes','No','Yes')
                                        % 默认设置为 Yes，且只有 Yes 和 No 两个按钮

button =

Yes
```

运行程序得到如图 3-60 所示结果。

（5）设置对话框为 3 个可选择按钮。

具体的 MATLAB 调用格式如下：

```
button = questdlg('qstring','title','str1','str2','str3',default)
```

其中，default 为默认按钮，一般选择 str1 字符串，具体的使用如下：

```
>> button = questdlg('提问对话框','余胜威','Yes','No','Cancel','Yes')

button =

Yes
```

运行程序得到如图 3-61 所示结果。

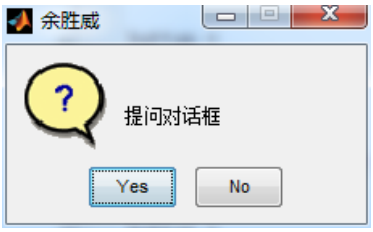

图 3-60　修改为默认按钮

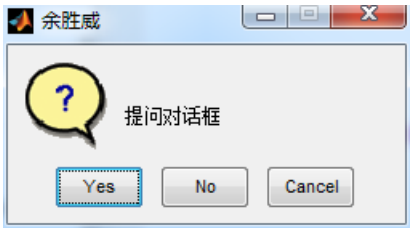

图 3-61　设置 3 个选择按钮

通过【Yes】、【No】、【Cancel】3 个选项，用户可以选择不同的按钮，执行相应的程序功能，例如代码如下：

```
% Construct a questdlg with three options
choice = questdlg('Would you like a dessert?', ...
    'Dessert Menu', ...
    'Ice cream','Cake','No thank you','No thank you'); % 创建 3 个按钮：Ice
    cream、Cake、No thank you
% Handle response
switch choice
   case 'Ice cream'
      disp([choice ' coming right up.'])      % 显示输出字符串
      dessert = 1;
   case 'Cake'
      disp([choice ' coming right up.'])      % 显示输出字符串
      dessert = 2;
   case 'No thank you'
      disp('I''ll bring you your check.')     % 显示输出字符串
      dessert = 0;
end
```

运行程序得到如下结果：

```
I'll bring you your check.
```

得到的界面如图 3-62 所示。

图 3-62　选项下对应的程序

（6）显示公式文本选项等。

具体的调用格式如下：

```
button = questdlg('qstring','title', ..., options)     % 显示公式文本
```

options 可以设置为显示文本的格式，例如当在提问对话框输入公式时，可以用 options 来进行控制，具体的使用如下：

```
options.Interpreter = 'tex';                    % 解析器
% Include the desired Default answer            % 默认设置
options.Default = 'Don''t know';
% Create a TeX string for the question
qstring = 'Is \Sigma(\alpha - \beta) < 0?';     % 提示字符串
choice = questdlg(qstring,'Boundary Condition',...
    'Yes','No','Don''t know',options)           % 提问对话框
```

运行程序得到如下结果：

```
choice =

Yes
```

得到的界面如图 3-63 所示。

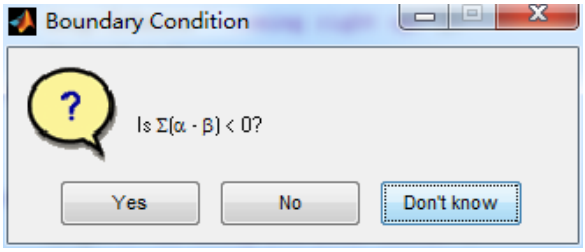

图 3-63　选项设置

3.9　数据输入对话框 inputdlg

inputdlg 为数据输入对话框，顾名思义就是用户可以在弹出的对话框中输入数值，从而被程序所读取，达到友好的修改参数的目的。具体的数据输入对话框 inputdlg 的使用格式如下：

```
>> help inputdlg
inputdlg - Create and open input dialog box

   This MATLAB function creates a modal dialog box and returns user input
for
   multiple prompts in the cell array.

   answer = inputdlg(prompt)
   answer = inputdlg(prompt,dlg_title)
   answer = inputdlg(prompt,dlg_title,num_lines)
   answer = inputdlg(prompt,dlg_title,num_lines,defAns)
   answer = inputdlg(prompt,dlg_title,num_lines,defAns,options)

   inputdlg 的参考页

   另请参阅 dialog, errordlg, figure, helpdlg, input, listdlg, msgbox,
questdlg, str2num, uiresume, uiwait, warndlg

>>
```

数据输入对话框 inputdlg 有 5 种输入模式，具体使用方法介绍如下。

（1）直接采用默认方式，弹出输入数值对话框。

具体的使用如下：

```
>> answer = inputdlg('')   % 数值对话框

answer =

   '100'
```

运行程序得到如图 3-64 所示结果。

如图 3-64 所示，该简单界面直接让用户输入数值。

可以在用户输入数值前，给出提示信息，具体如下：

```
>> answer = inputdlg('a=?')   % 输入 a 的取值

answer =

    {}
```

运行程序得到如图 3-65 所示结果。

（2）输入对话框标题修改。

可以对提示信息框的标题进行修改，具体的使用如下：

```
>> answer = inputdlg('a=?','数据输入')  % 标题为“数据输入”

answer =

   '100'
```

运行程序得到如图 3-66 所示结果。

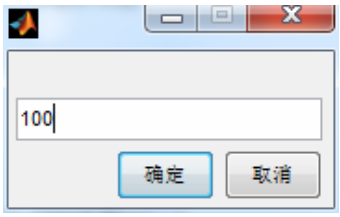

图 3-64　数值输入

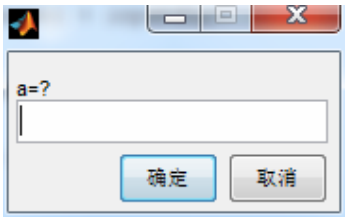

图 3-65　提示信息显示

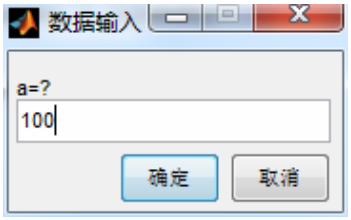

图 3-66　标题显示修改

特别地，当输入为数值 100 时，可将结果直接读取，读取方式如下：

```
>> data = str2num(answer{:})   % 转化为数值型

data =

  100
```

（3）增加输入的行数。

如图 3-65~图 3-66 所示为输入一个数值的情况，inputdlg 可直接输入多个变量值，具体如下：

```
clc,clear,close all                          % 清屏
warning off                                  % 取消警告
prompt = {'矩阵维数:','颜色空间名称:'};       % 输入对话框说明符
dlg_title = '余胜威';                        % 标题
answer = inputdlg(prompt,dlg_title)          % 显示
```

运行程序得到如图 3-67 所示结果。

当输入如图 3-68 所示，单击【确定】按钮时，返回值如下：

```
answer =
```

```
    '3'
    'rgb'
```

返回值为一个 cell 细胞体数组，并且按照行进行保存。

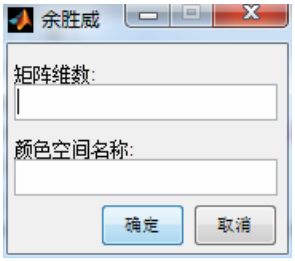

图 3-67　多行输入

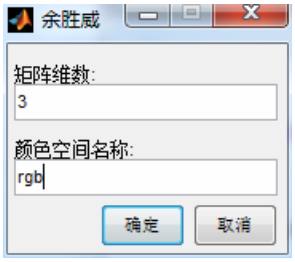

图 3-68　输入量

（4）修改输入的维数。

如图 3-68 所示，矩阵维数输入为一个实数，也可以输入为一个矩阵，具体的程序如下：

```
clc,clear,close all                                    % 清屏
warning off                                            % 取消警告
prompt = {'矩阵维数:','颜色空间名称:'};
dlg_title = '余胜威';                                   % 标题
num_lines = 2;                                         % 两栏输入
answer = inputdlg(prompt,dlg_title,num_lines)  % 显示
```

运行程序得到如图 3-69 所示对话框。

用户可以在如图 3-69 所示对话框中输入数值和颜色空间名称，针对每一行可以输入多个数值和字符串，但只能输入 num_lines 行，当输入如图 3-70 所示时，程序返回如下结果：

```
answer =

    [2x7 char]
[2x4 char]

>> answer{1,1}
ans =
2 3 1 2
1 2 3 4

>> answer{2,1}
ans =
rgb
yuv
```

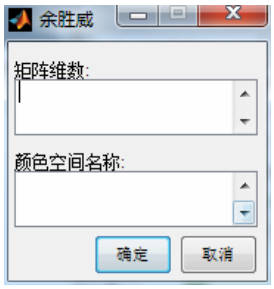

图 3-69　多数据输入

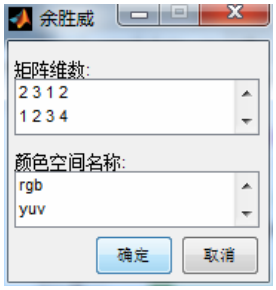

图 3-70　输入字符串

同样可以设定输入的字符的个数，例如：

```
x = inputdlg({'Name','Telephone','Account'},...
             'Customer', [1 50; 1 12; 1 7]);
```

运行程序得到如图 3-71 所示结果。

（5）设定输入对话框的默认值。

有时用户知道初始状态，可以直接输入相应的默认值，具体的使用格式如下：

```
answer = inputdlg(prompt,dlg_title,num_lines,defAns)  % 输入对话框
```

具体的案例分析如下：

```
clc,clear,close all                 % 清屏
warning off                         % 取消警告
prompt = {'矩阵维数:','颜色空间名称:'};
dlg_title = '余胜威';
num_lines = 1;                      % 一栏显示
def = {'20','hsv'};                 % 初始化设置
answer = inputdlg(prompt,dlg_title,num_lines,def);
```

运行程序得到如图 3-72 所示结果。

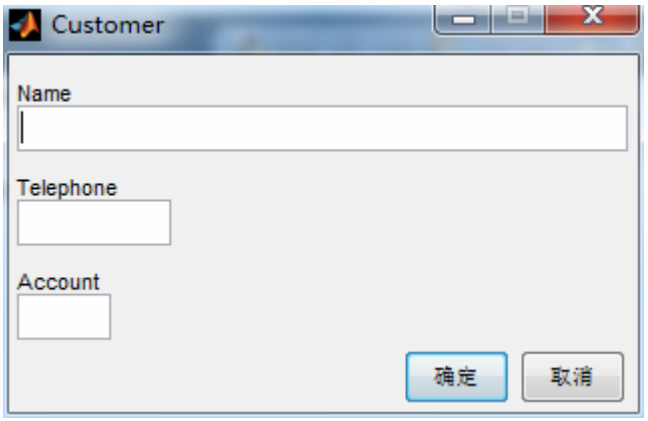

图 3-71　输入字符串长度

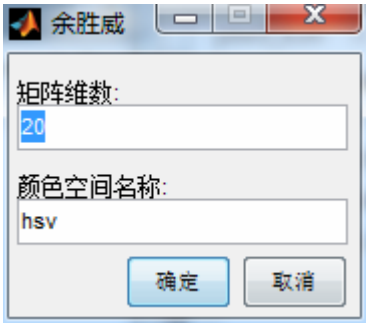

图 3-72　默认输入值

（6）修改显示字体。

Inputdlg 输入对话框，提供各种字体的输入，具体的使用格式如下：

```
answer = inputdlg(prompt,dlg_title,num_lines,defAns,options)% 修改显示字体
```

相应的实际应用如下：

```
clc,clear,close all                 % 清屏和清除工作区
warning off                         % 取消警告
prompt = {' x^2:','颜色空间名称:'};
dlg_title = '余胜威';
numlines=1;                         % 一栏
defaultanswer={'20','RGB'};         % 默认设置
options.Resize='on';                % 矢量化窗口打开
options.WindowStyle='normal';       % 窗口可以拉大缩小
```

```
options.Interpreter='tex';          % 解析器设置
answer=inputdlg(prompt,dlg_title,numlines,...
              defaultanswer,options);   % 输入窗口显示
```

运行程序得到如图 3-73 所示结果。

图 3-73　字体样式

3.10　本 章 小 结

本章对常用的对话框进行了系统的说明，具体涉及到 uigetfile、uigetdir、uiputfile、waitbar、errordlg、warndlg、msgbox、questdlg 和 inputdlg 等功能的阐述，内容详细且易学易用。