大连理工大学实验预习报告

学院（系）：信息与通信工程学院 专业：电子信息工程 班级：电信 1703

姓　　名：李翰廷 学号：201746013 组：

实验时间： 实验室： 实验台：

指导教师：

实验 II：系统对随机信号响应的统计特性分析、功率谱分析及应用实验

一、 实验目的和要求

掌握直接法估计随机信号功率谱的原理和实现方法；掌握间接法估计随机信号功率谱的原理和实现方法；掌握系统对随机信号响应的统计特性分析及仿真实现方法。熟悉 MATLAB 信号处理软件包的使用。

二、 实验原理和内容

（一）实验原理：

1. 直接法估计随机信号功率谱原理

直接法又称为周期图法，它是把随机信号 x(n)的 N 点观察数据 xN(n)视为一能量有限信号，直接取 xN(n)的傅里叶变换，得到 XN(ejω)，然后取其模值的平方，并除以 N，作为对 x(n)真实的功率谱 P(ejω)的估计。工程上，常使用离散 Fourier 变换（DFT，编程上使用其快速算法 FFT），即 $P_X(k)=\frac{1}{N}|X_M(k)|^2$，进行计算。

2. 间接法估计随机信号功率谱

间接法的理论基础是 Wiener-Khintchine 定理，具体的实现方法是先由 xN(n)估计出自相关函数 r`(m)，然后对 r`(m)求傅里叶变换得到 xN(n)的功率谱，记之为 XN(ejω)，并以此作为对真实功率谱 P(ejω)的估计。工程上，常使用离散 Fourier 变换（DFT，编程上使用其快速算法 FFT），即：

$$P_X(k)=\sum_{m=-M}^{M}\hat{r}(m)e^{-j\frac{2\pi km}{2M+1}},\quad |M|\le N-1$$

进行计算。因为由这种方法求出的功率谱是通过自相关函数间接得到的，所以又称为间接法或 Blackman-Tuckey(BT)法，该方法是 FFT 出现之前常用的谱估计方法。

3. 时域中系统对随机信号响应的统计特性分析及仿真

根据系统卷积性质，计算系统输出信号的统计特性。有如下性质：

$$m_Y = m_X \sum_n h(n)$$

$$R_Y(m) = \sum_j \sum_k R_X(m+j-k)h(j)h(k)$$

4. 频域中系统对随机信号响应的统计特性分析及仿真

根据卷积定理，输入、输出信号功率谱的关系为:

$$R_Y(e^{j\omega}) = R_X(e^{j\omega})|H(e^{j\omega})|^2$$

在计算系统输 出信号功率谱时，如果在时域时计算困难，可以按照上式在频域计算。

（二）实验内容：

1. 直接法估计随机信号功率谱

(1) 生成 1024 点数据的随机信号

$$X(n) = 2\cos(2\pi f_1 t + \varphi_1) + 5\cos(2\pi f_2 t + \varphi_2) + N(n)$$

其中$f_1 = 30Hz$，$f_2 = 100Hz$，φ_1，φ_2为在$[0,2\pi]$内的均匀分布的随机变量，$N(n)$是数学期望为 0，方差为 1 的高斯白噪声。

(2) 用周期图法计算$X(n)$的功率谱，并绘图。

(3) 用 MATLAB 函数 periodogram 重新计算$X(n)$的功率谱，并与(2)做比较。

2．间接法估计随机信号功率谱

(1) 计算以上$X(n)$的自相关函数。

(2) 通过计算自相关函数的 Fourier 变换，求$X(n)$的功率谱并绘图。

(3) 利用 MATLAB 函数 psd、pwelch 重新计算$X(n)$的功率谱，并与(2)做比较。

3．系统对随机信号响应的统计特性分析及仿真

(1) 生成含 500 点数据的高斯分布白噪声随机信号$X(n)$。

(2) 设计一个带通系统$H(e^{j\omega})$，其上、下截止频率分别为 4KHz 和 3KHz.

(3) 计算$X(n)$通过以上带通滤波器的自相关函数和功率谱密度。

三、 实验步骤

1. 直接法估计随机信号功率谱

(1) 准备实验环境。
(2) 生成 1024 点高斯白噪声。
(3) 显示随机信号时域波形。
(4) 通过 Fourier 变换取上述高斯白噪声 1024 点计算对应的功率谱 PX(k)，并绘图。
(5) 与内置函数对比，并尝试改变信号、噪声幅度和频率观察效果。

2. 间接法估计随机信号功率谱

(1) 准备环境。
(2) 生成 1024 点高斯白噪声。
(3) 显示随机信号时域波形。
(4) 计算高斯白噪声的自相关函数。
(5) 通过 BT 法计算高斯白噪声的功率谱 Px(k)，并绘图。
(6) 与内置函数对比，并尝试改变信号、噪声幅度和频率观察效果。

3. 系统对随机信号响应的统计特性分析及仿真

(1) 准备环境。
(2) 生成均匀分布的随机信号。
(3) 设计滤波器。
(4) 计算输出信号均值、方差、自相关函数、功率谱等统计量。
(5) 图形展示系统输出观察效果。

四、实验数据记录表格

1. 直接法估计随机信号功率谱

	实验点 1
直接法功率谱	
内置函数功率谱	

2. 间接法估计随机信号功率谱

	实验点 1
间接法功率谱	
内置函数功率谱	

3. 系统对随机信号响应的统计特性分析及仿真

输入信号功率谱密度	
系统的功率传输函数	
输出信号的功率谱密度	
输出信号的自相关函数	

大连理工大学实验报告

学院（系）：信息与通信工程学院 专业：_____班级：_____

姓　　名：_____学号：_____组：_____

实验时间：_____实验室：_____实验台：_____

指导教师：_____

实验 II：系统对随机信号响应的统计特性分析、功率谱分析及应用实验

一、 实验目的和要求

掌握直接法估计随机信号功率谱的原理和实现方法；掌握间接法估计随机信号功率谱的原理和实现方法；掌握系统对随机信号响应的统计特性分析及仿真实现方法。熟悉 MATLAB 信号处理软件包的使用。

二、 实验原理和内容

2.1 直接法估计随机信号功率谱原理

直接法又称为周期图法，它是把随机信号 $x(n)$的 N 点观察数据 $x_N(n)$视为一能量有限信号，直接取 $x_N(n)$的傅里叶变换，得到 $X_N(e^{j\omega})$，然后取其模值的平方，并除以 N，作为对 $x(n)$真实的功率谱 $P(e^{j\omega})$的估计。工程上，常使用离散 Fourier 变换（DFT，编程上使用其快速算法 FFT），即 $P_X(k)=1/N|X_N(k)|^2$，进行计算。

2.2 间接法估计随机信号功率谱

间接法的理论基础是 Wiener-Khintchine 定理，具体的实现方法是先由 $x_N(n)$估计出自相关函数 $r^(m)$，然后对 $r^(m)$求傅里叶变换得到 $x_N(n)$的功率谱，记之为 $X_N(e^{j\omega})$，并以此作为对真实功率谱 $P(e^{j\omega})$的估计。工程上，常使用离散 Fourier 变换（DFT，编程上使用其快速算法 FFT），即 $P_X(k)=\sum_{m=-M}^{M}\hat{r}(m)e^{-j\frac{2\pi km}{2M+1}}$，$|M|\le N-1$进行计算。因为由这种方法求出的功率谱是通过自相关函数间接得到的，所以又称为间接法或 Blackman-Tuckey(BT)法，该方法是 FFT 出现之前常用的谱估计方法。

2.3 时域中系统对随机信号响应的统计特性分析及仿真

根据系统卷积性质，计算系统输出信号的统计特性。 有如下性质：

$m_Y = m_X \sum_n h(n)$， $R_Y(m) = \sum_j \sum_k R_X(m+j-k)h(j)h(k)$。

2.4 频域中系统对随机信号响应的统计特性分析及仿真

根据卷积定理,输入、输出信号功率谱的关系为 $R_Y(e^{j\omega}) = R_X(e^{j\omega})|H(e^{j\omega})|^2$。 在计算系统输出信号功率谱时，如果在时域时计算困难，可以按照上式在频域计算。

三、 主要仪器设备

Windows10(64 位)，MATLABR2018a，MATLAB2016b

四、 实验步骤与操作方法

4.1 直接法估计随机信号功率谱

先生成 1024 点数据的随机信号，再用 fft 函数估计功率谱，并与 periodog 函数对比估计功率谱

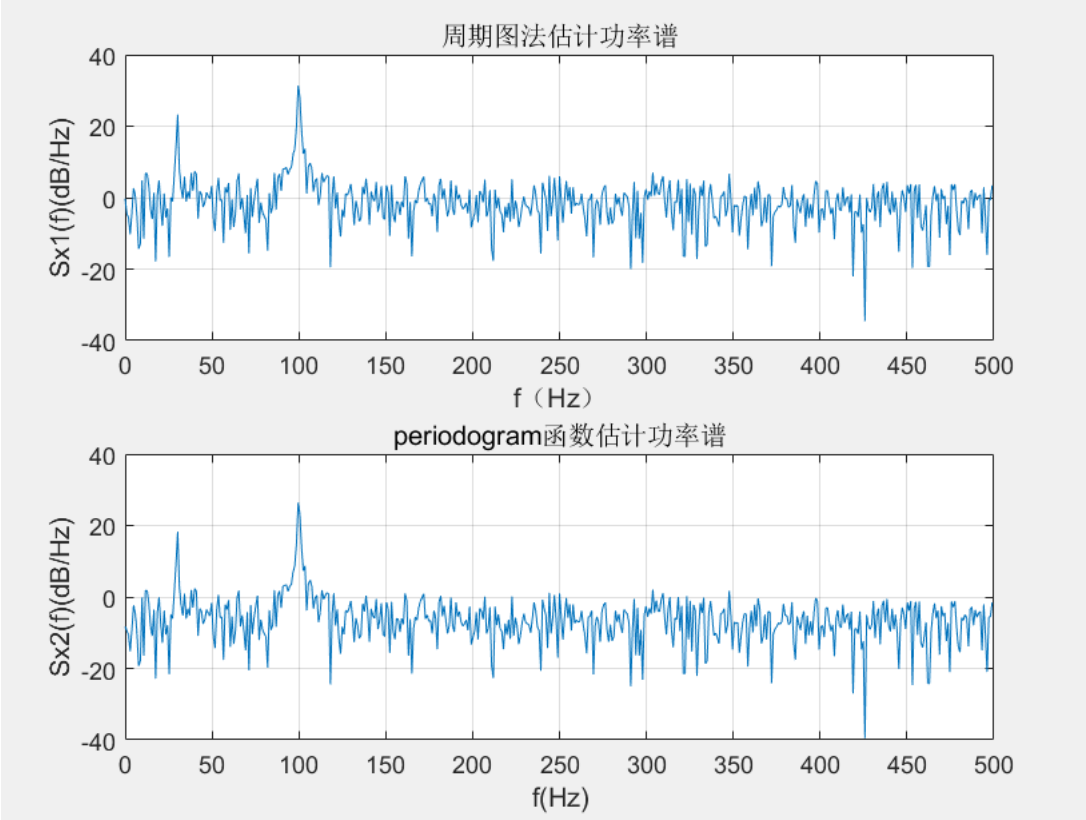
4.2 间接法估计随机信号功率谱

计算 1 中$X(n)$的自相关函数并进行 Fourier 变换，求$X(n)$的功率谱并绘图再利用 MATLAB 函数 psd、pwelch 重新计算$X(n)$的功率谱，并与(2)做比较。

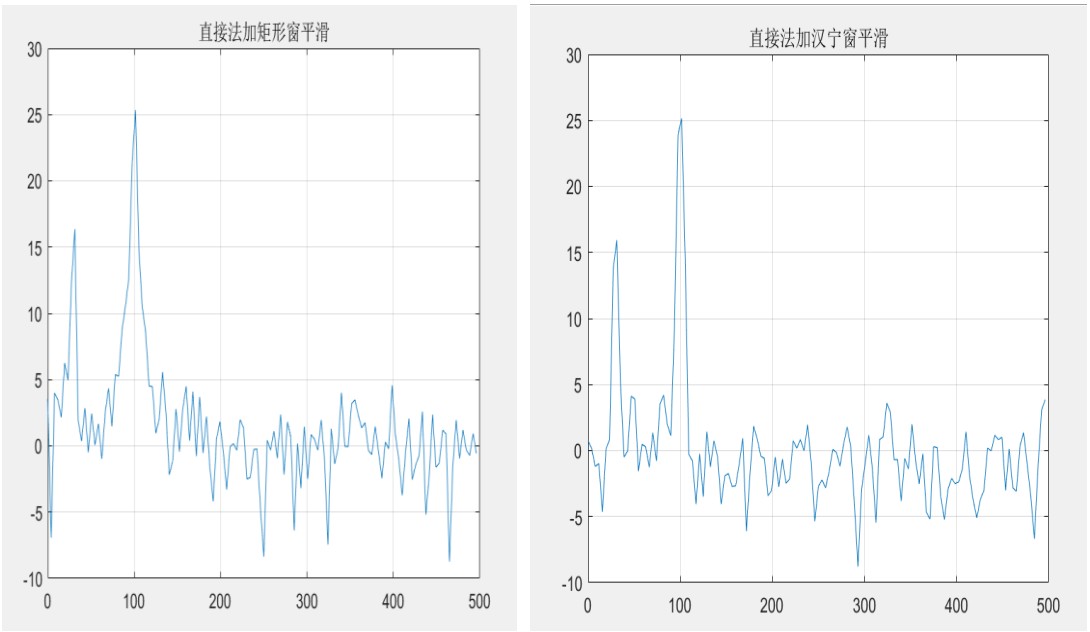
4.3 系统对随机信号响应的统计特性分析及仿真

生成含 500 点数据的高斯分布白噪声随机信号$X(n)$，设计一个上、下截止频率分别为 4KHz 和 3KHz 的带通系统$H(e^{j\omega})$,再用内置公式计算$X(n)$通过以上带通滤波器的自相关函数和功率谱密度。

五、 实验数据记录和处理

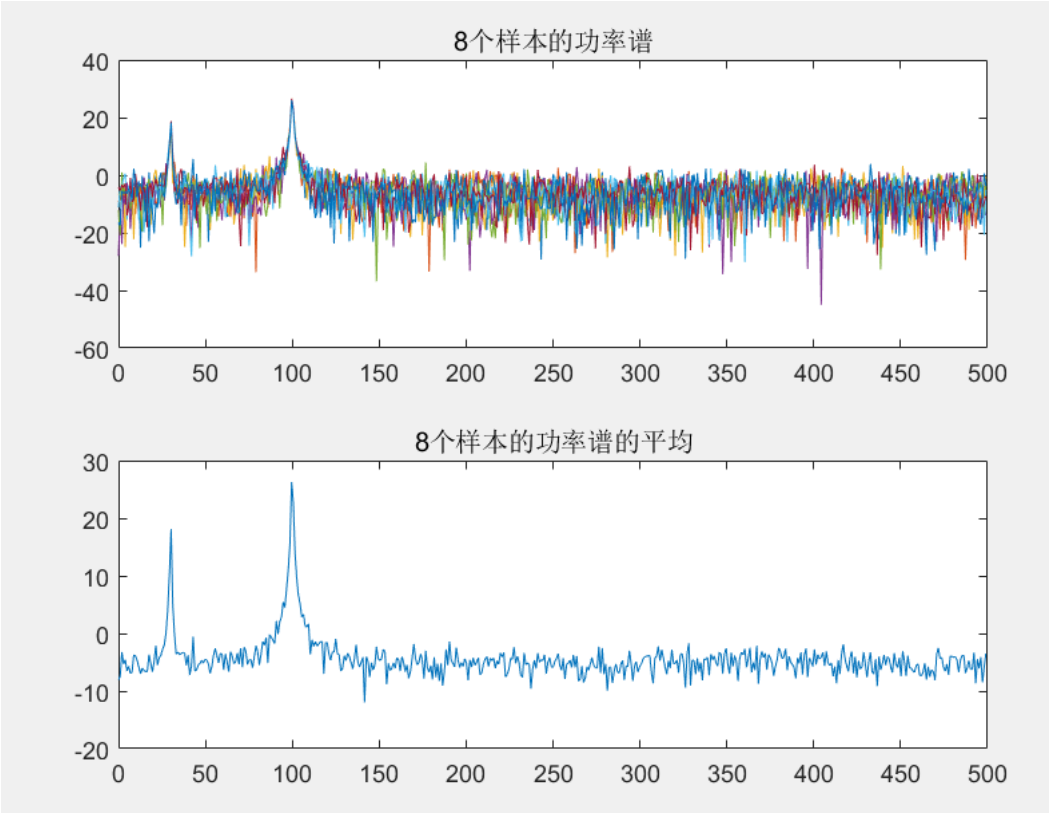


如图所示首先用自编的周期图法生成了功率谱，然后又调用函数 periodogram 函数生成了功率谱观察二者，发现区别不大。但发现两个问题，周期图法生成的功率谱一是毛刺太多，二是叠加的白噪声部分不是常数功率谱。

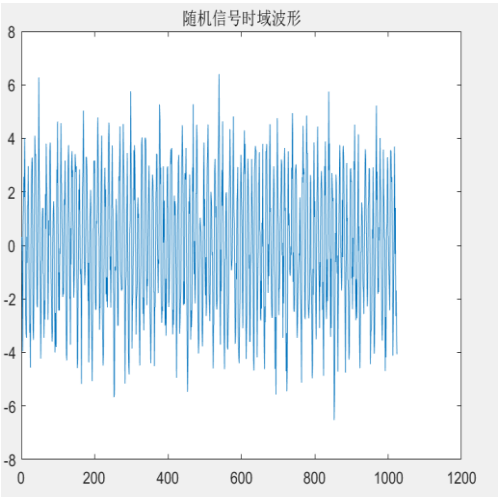


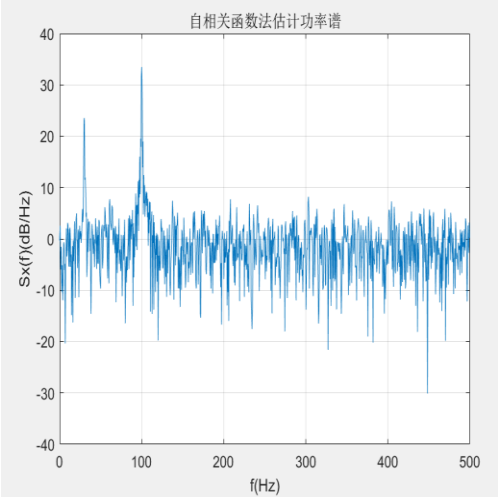
如图所示，当数据长度太大或者太小，可能引起功率谱曲线起伏加剧或者降低分辨

率，最终导致平滑程度低，而分别采用加矩形窗和汉宁窗的方法，改进了直接法生成的功率谱，使其平滑程度增高

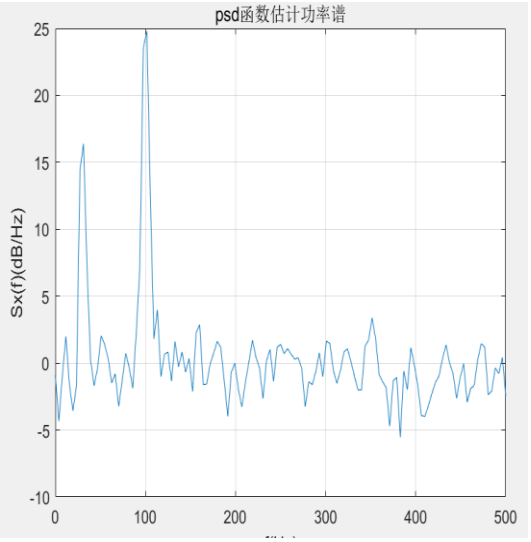


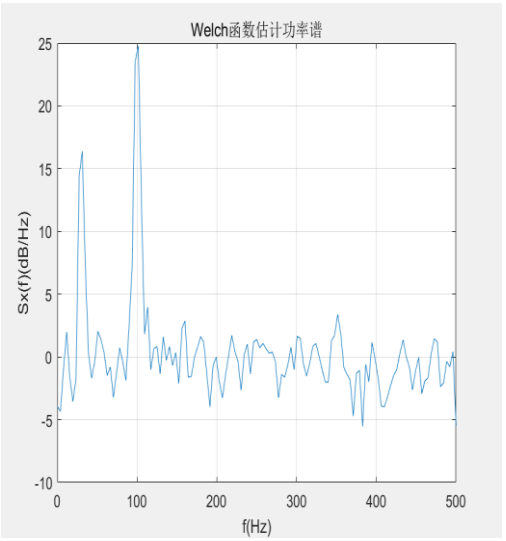
如图所示，不是常数的原因是估计功率谱时只用了一个样本序列，另外用了有限个观测数据，为改善这种情况，采用多个样本估计功率谱再取统计平均的方法，使叠加部分更接近理论上的白噪声常数功率谱。



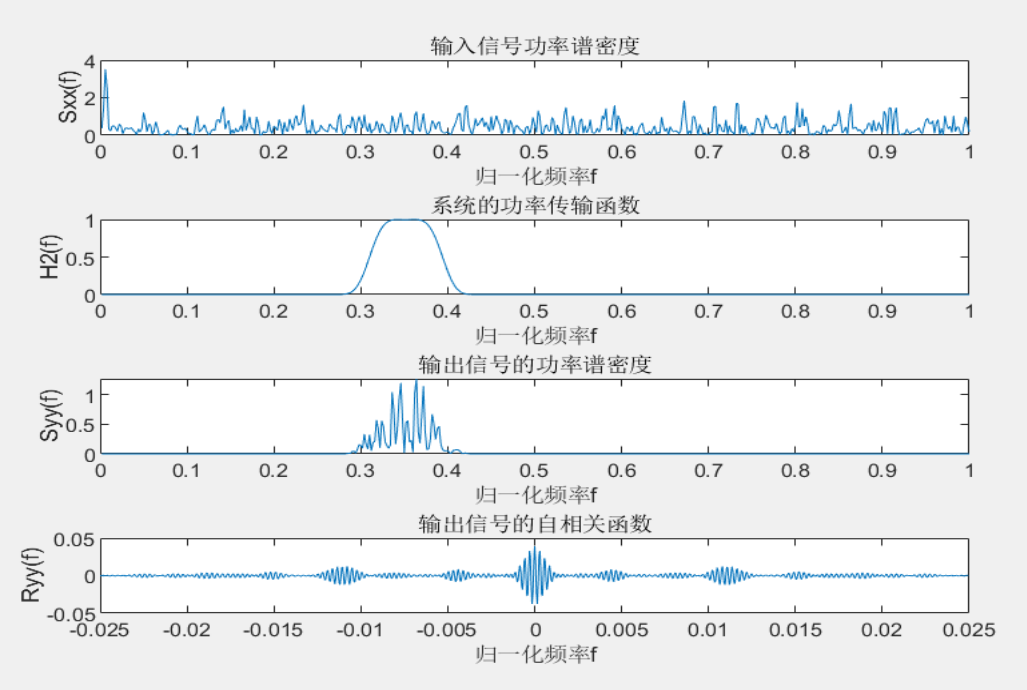


如图所示用间接法即自相关函数傅里叶变换求功率谱





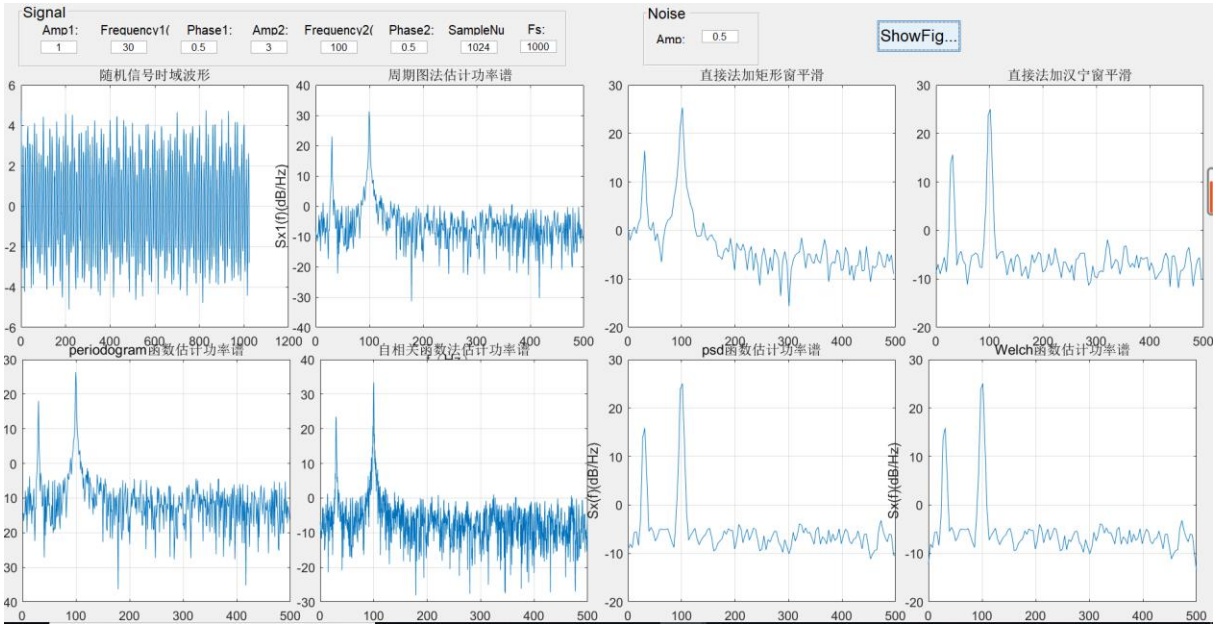
如图所示调用了系统函数 psd 和 welch 生成功率谱，它们与直接法的改进有类似之处，分别是矩形窗平滑和汉宁窗平滑。



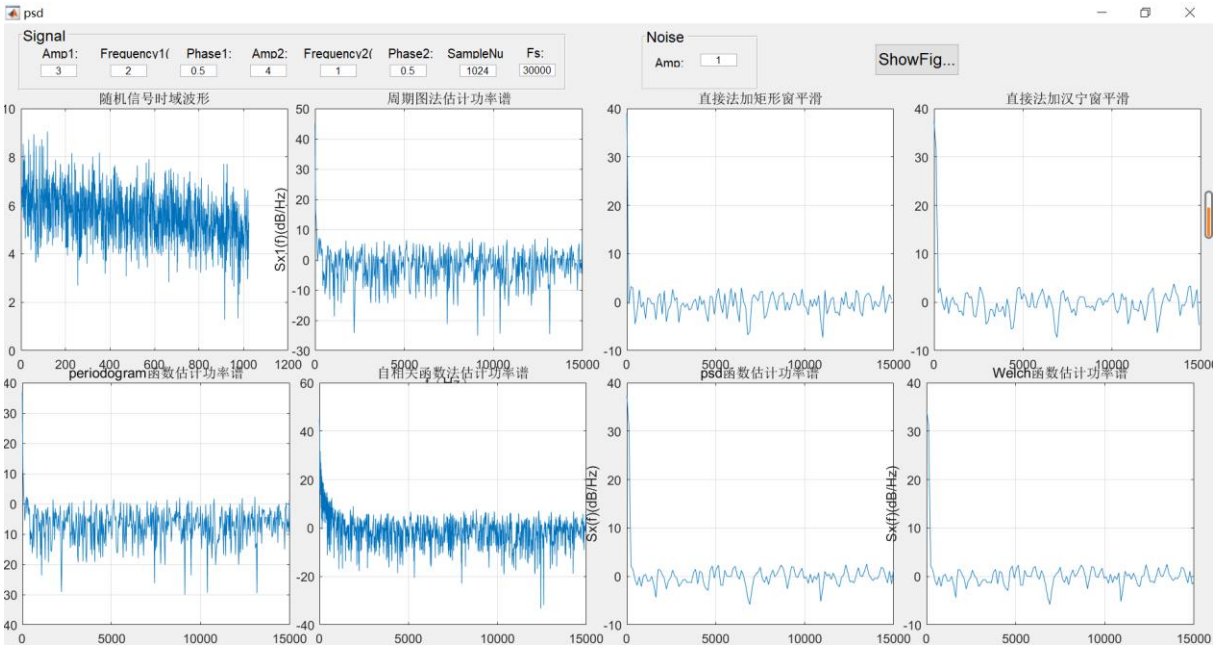
如图所示，生成含 500 点数据的高斯分布白噪声随机信号$X(n)$，设计一个上、下截止频率分别为 4KHz 和 3KHz 的带通系统$H(e^{j\omega})$，再用内置公式计算$X(n)$通过以上带通滤波器的自相关函数和功率谱密度。

六、 实验结果与分析

为比较任务二中三种方法的不同之处和优劣所在，查阅资料做了 GUI，可以调整两个乘积信号的参数和噪声的幅度，以及采样点和频率



查阅资料显示 psd 的原理是求功率谱而 pwelch 是求功率谱密度，没能弄懂于是想用 GUI 具体看一看，输入信号参数如图所示，差别不明显，尝试改变了其中的参数，当频率增大到 30kHz 的时候能看出来 psd 的谱线更加狭窄与突出。查阅资料显示 psd 的谱线在输入信号更复杂的情况下不只信号源频率处谱线突出，其他频率也会有突出，本实验可能由于输入信号还不够复杂没能显示出来。



比较任务二的三种方法，普通的间接法没有任何优化，毛刺严重，显然不如后两种，而 psd 函数在新版的 matlab 中已经不能调用，所以 pwelch 应该是最好的方法，查阅资料显示的确 pwelch 为从宽带噪声中检测窄带信号的首选，不过很遗憾我的 GUI 没能显示出来。

七、讨论、建议、质疑

首先本次实验由于使用的是自己的电脑，没有 psd 函数，后来通过从 2016 版 matlab 的 m 文件中拷贝的方法解决了这个问题，不过这也让我更深刻地意识到了每次实验报告中写的实验设备和操作环境的重要性，这不是形式而已，反而的确决定了实验成果能否实现。

此外最后设计了 GUI 想去仔细比较 psd 和 pwelch 的区别，不过可以说是以失败告终，查阅资料显示用更复杂的信号，二者的区别才能显现。

八、MATLAB 代码

8.1 直接法：

```
N=1024;fs=1000;                          %序列长度和采样频率
t=(0:N-1)/fs;                            %时间序列
fai=random('unif',0,1,1,2)*2*pi;         %产生 2 个[0，2pi]内均匀随机数
xn=cos(2*pi*30*t+fai(1))+3*cos(2*pi*100*t+fai(2))+randn(1,N);
                                         %产生含噪声的随机序列
figure,plot(xn);
title('随机信号时域波形')
Sx1=abs(fft(xn)).^2/N;                   %估计功率谱
f=(0:N/2-1)*fs/N;                        %频率轴坐标
figure
subplot(211);
plot(f,10*log10(Sx1(1:N/2)));grid on; %用 dB/Hz 做功率谱单位，画图
xlabel('f（Hz）');
ylabel('Sx1(f)(dB/Hz)');
title('周期图法估计功率谱');
Sx2=periodogram(xn);
subplot(212);
plot(f,10*log10(Sx2(1:N/2)));grid on;
```

```
xlabel('f(Hz)');
ylabel('Sx2(f)(dB/Hz)');
title('periodogram 函数估计功率谱');
```

8.2 矩形窗和汉宁窗：

```
N=1024;
fs=1000;
t=(0:N-1)/fs;
fai=random('unif',0,1,1,2)*2*pi;       %产生 2 个[0，2pi]内均匀随机数
xn=cos(2*pi*30*t+fai(1))+3*cos(2*pi*100*t+fai(2))+randn(1,N);
Nseg=256;
win=rectwin(256)';%注意符号
Sx1=abs(fft(win.*xn(1:256),Nseg).^2)/norm(win)^2;
Sx2=abs(fft(win.*xn(257:512),Nseg).^2)/norm(win)^2;
Sx3=abs(fft(win.*xn(513:768),Nseg).^2)/norm(win)^2;
Sx4=abs(fft(win.*xn(769:1024),Nseg).^2)/norm(win)^2;
Sx=10*log10((Sx1+Sx2+Sx3+Sx4)/4);
f=(0:Nseg/2-1)*fs/Nseg;
figure(8),plot(f,Sx(1:Nseg/2));grid on;
title('直接法加矩形窗平滑');
```

8.3 周期图法多个求平均改进

```
clear;
N=1024;fs=1000;
t=(0:N-1)/fs;
fai=random('unif',0,1,1,8)*2*pi;
xln=random('norm',0,1,N,8);
for k =1:8
    xn(:,k)=sin(2*pi*30*t(:)+fai(k))+3*sin(2*pi*100*t(:)+fai(k))+xln(:,k);
    Sx(:,k)=periodogram(xn(:,k));
end
ESx=mean(Sx(1:N/2,:),2);
```

```
f=(0:N/2-1)*fs/N;
subplot(211);plot(f,10*log10(Sx(1:N/2,:)));title("8 个样本的功率谱")
subplot(212);plot(f,10*log10(ESx));title("8 个样本的功率谱的平均")
```

8.4 间接法，psd 和 pwelch

```
N=1024;fs=1000;                                    %序列长度和采样频率
t=(0:N-1)/fs;                                      %时间序列
fai=random('unif',0.1,1,2)*2*pi;                   %产生 2 个[0，2pi]内均匀随机数
xn=cos(2*pi*30*t+fai(1))+3*cos(2*pi*100*t+fai(2))+randn(1,N);
                                                   %产生含噪声的随机序列
figure(1),plot(xn);
title('随机信号时域波形')
Rxx=xcorr(xn,'biased');                            %估计自相关函数 Rxx
Sx1=abs(fft(Rxx));                                 %对 Rxx 进行 FFT 得到功率谱
f=(0:N-1)*fs/N/2;                                  %频率轴坐标
figure(2);
plot(f,10*log10(Sx1(1:N)));grid on;                %用 dB/Hz 做功率谱单位，画图
xlabel('f(Hz)');
ylabel('Sx(f)(dB/Hz)');
title('自相关函数法估计功率谱')
Nseg=256;                                          %分段间隔为 256
window=hanning(Nseg);                              %汉宁窗
noverlap=Nseg/2;                                   %重叠点数为 128
f=(0:Nseg/2)*fs/Nseg;                              %频率轴坐标
Nseg=256;                                          %分段间隔为 256
window=hanning(Nseg);                              %汉宁窗
noverlap=Nseg/2;                                   %重叠点数为 128
f=(0:Nseg/2)*fs/Nseg;                              %频率轴坐标
Sx2=mypsd(xn,Nseg,fs,window,noverlap,'none');    %psd 函数估计功率谱
figure(3);
plot(f,10*log10(Sx2));grid on;
```

```
xlabel('f(Hz)');
ylabel('Sx(f)(dB/Hz)');
title('psd 函数估计功率谱');
Sx3=pwelch(xn,window,128,Nseg,fs,'onesided')*fs/2;
figure(4);                                          %Welch 函数估计功率谱
plot(f,10*log10(Sx3));grid on;
xlabel('f(Hz)');
ylabel('Sx(f)(dB/Hz)');
title('Welch 函数估计功率谱');
```

8.5 系统特性分析

```
N=500;                                              %样本长度 N=500，对应时长 25ms
xt=random('norm',0,1,1,N);                          %产生 1*N 个高斯随机数
%  显示时域波形
figure,plot(xt);
title('随机信号时域波形')
%冲激响应
ht=fir1(101,[0.3 0.4]);                             % 101 阶带通滤波器，数字截止频率为 0.3 和 0.4
%  显示冲激响应函数 ht
figure,plot(ht)
title('冲激响应函数 ht')
%传递函数
HW=fft(ht,2*N);                                     %2N 点滤波器频率响应（系统传输函数）
%  显示传递函数 HW
figure,plot((1:N)/N,abs(HW(1:N)));
title('传递函数 HW')
Rxx=xcorr(xt,'biased');                             %直接法估计白噪声的自相关函数
%  显示自相关函数 Rxx
figure,stem(Rxx)
title('输入自相关函数 Rxx')
Sxx=abs(fft(xt,2*N).^2)/(2*N);                      %周期图法估计白噪声的功率谱
```

```
HW2=abs(HW).^2;                                    %系统的功率传输函数
Syy=Sxx.*HW2;                                      %输出信号的功率谱
Ryy=fftshift(ifft(Syy));                           %用 IFFT 求输出信号的自相关函数
                                                   %函数 fftshift 对数组进行移
w=(1:N)/N;                                         %功率谱密度横轴坐标
t=(-N:N-1)/N*(N/20000);                            %自相关函数横轴坐标
subplot(4,1,1);plot(w,abs(Sxx(1:N)));              %输入信号功率谱密度
xlabel('归一化频率 f');ylabel('Sxx(f)');title('输入信号功率谱密度');
subplot(4,1,2);plot(w,abs(HW2(1:N)));              %系统的功率传输函数
xlabel('归一化频率 f');ylabel('H2(f)');title('系统的功率传输函数');
subplot(4,1,3);plot(w,abs(Syy(1:N)));              %输出信号的功率谱密度
xlabel('归一化频率 f');ylabel('Syy(f)');title('输出信号的功率谱密度');
subplot(4,1,4);plot(t,Ryy);                        %输出信号的自相关函数
xlabel('归一化频率 f');ylabel('Ryy(f)');title('输出信号的自相关函数');
```

8.6GUI

```
function varargout = psd(varargin)
% PSD MATLAB code for psd.fig
%      PSD, by itself, creates a new PSD or raises the existing
%      singleton*.
%
%      H = PSD returns the handle to a new PSD or the handle to
%      the existing singleton*.
%
%      PSD('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in PSD.M with the given input arguments.
%
%      PSD('Property','Value',...) creates a new PSD or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before psd_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
```

```
%           stop.    All inputs are passed to psd_OpeningFcn via varargin.
%
%           *See GUI Options on GUIDE's Tools menu.    Choose "GUI allows only one
%           instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help psd

% Last Modified by GUIDE v2.5 12-Nov-2019 10:34:09

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',              mfilename, ...
                                'gui_Singleton',    gui_Singleton, ...
                                'gui_OpeningFcn', @psd_OpeningFcn, ...
                                'gui_OutputFcn',    @psd_OutputFcn, ...
                                'gui_LayoutFcn',    [] , ...
                                'gui_Callback',      []);
if nargin && ischar(varargin{1})
      gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
      [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
      gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before psd is made visible.
function psd_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to psd (see VARARGIN)

% Choose default command line output for psd
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
% UIWAIT makes psd wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = psd_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
Amp1 = eval(get(handles.Amp1,'String'));
Pha1 = eval(get(handles.Pha1,'String'));
Freq1 = eval(get(handles.Freq1,'String'));
Amp2 = eval(get(handles.Amp2,'String'));
Pha2 = eval(get(handles.Pha2,'String'));
Freq2 = eval(get(handles.Freq2,'String'));
Num =eval( get(handles.Num,'String'));
AmpN = eval(get(handles.AmpN,'String'));
Fs = eval(get(handles.Fs,'String'));
Data = data_process(Amp1,Pha1,Freq1,Amp2,Pha2,Freq2,Num,AmpN,Fs);
axes(handles.axes1);
hold off;
plot(Data.A(:,1),Data.A(:,2));grid on;
title('随机信号时域波形')
axes(handles.axes2);
hold off;
plot(Data.B(:,1),Data.B(:,2));grid on;
xlabel('f（Hz）');
ylabel('Sx1(f)(dB/Hz)');
title('周期图法估计功率谱');
axes(handles.axes3);
hold off;
plot(Data.C(:,1),Data.C(:,2));grid on;
title('直接法加矩形窗平滑');
axes(handles.axes4);
hold off;
plot(Data.D(:,1),Data.D(:,2));grid on;
title('直接法加汉宁窗平滑');

axes(handles.axes5);
```

```
hold off;
plot(Data.E(:,1),Data.E(:,2));grid on;
xlabel('f(Hz)');
ylabel('Sx2(f)(dB/Hz)');
title('periodogram 函数估计功率谱');
axes(handles.axes6);
hold off;
plot(Data.F(:,1),Data.F(:,2));grid on;
title('自相关函数法估计功率谱');
axes(handles.axes7);
hold off;
plot(Data.G(:,1),Data.G(:,2));grid on;
xlabel('f(Hz)');
ylabel('Sx(f)(dB/Hz)');
title('psd 函数估计功率谱');
axes(handles.axes8);
hold off;
plot(Data.H(:,1),Data.H(:,2));grid on;
xlabel('f(Hz)');
ylabel('Sx(f)(dB/Hz)');
title('Welch 函数估计功率谱');
function AmpN_Callback(hObject, eventdata, handles)
% hObject        handle to AmpN (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles        structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AmpN as text
%              str2double(get(hObject,'String')) returns contents of AmpN as a double
```

```
% --- Executes during object creation, after setting all properties.
function AmpN_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AmpN (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function Amp1_Callback(hObject, eventdata, handles)
% hObject    handle to Amp1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Amp1 as text
%        str2double(get(hObject,'String')) returns contents of Amp1 as a double
% --- Executes during object creation, after setting all properties.
function Amp1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Amp1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function Freq2_Callback(hObject, eventdata, handles)
% hObject    handle to Freq2 (see GCBO)
```

```
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Freq2 as text
%        str2double(get(hObject,'String')) returns contents of Freq2 as a double
% --- Executes during object creation, after setting all properties.
function Freq2_CreateFcn(hObject, eventdata, handles)
% hObject      handle to Freq2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function Freq1_Callback(hObject, eventdata, handles)
% hObject      handle to Freq1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Freq1 as text
%        str2double(get(hObject,'String')) returns contents of Freq1 as a double
% --- Executes during object creation, after setting all properties.
function Freq1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to Freq1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
function Pha1_Callback(hObject, eventdata, handles)
% hObject        handle to Pha1 (see GCBO)
% eventdata     reserved - to be defined in a future version of MATLAB
% handles       structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Pha1 as text
%               str2double(get(hObject,'String')) returns contents of Pha1 as a double
% --- Executes during object creation, after setting all properties.
function Pha1_CreateFcn(hObject, eventdata, handles)
% hObject        handle to Pha1 (see GCBO)
% eventdata     reserved - to be defined in a future version of MATLAB
% handles       empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%             See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function Amp2_Callback(hObject, eventdata, handles)
% hObject        handle to Amp2 (see GCBO)
% eventdata     reserved - to be defined in a future version of MATLAB
% handles       structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Amp2 as text
%               str2double(get(hObject,'String')) returns contents of Amp2 as a double
% --- Executes during object creation, after setting all properties.
function Amp2_CreateFcn(hObject, eventdata, handles)
% hObject        handle to Amp2 (see GCBO)
% eventdata     reserved - to be defined in a future version of MATLAB
% handles       empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
```

```
%           See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Pha2_Callback(hObject, eventdata, handles)
% hObject    handle to Pha2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Pha2 as text
%        str2double(get(hObject,'String')) returns contents of Pha2 as a double

% --- Executes during object creation, after setting all properties.
function Pha2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Pha2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%           See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Num_Callback(hObject, eventdata, handles)
% hObject    handle to Num (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Num as text
%        str2double(get(hObject,'String')) returns contents of Num as a double

% --- Executes during object creation, after setting all properties.
function Num_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Num (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles        empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%               See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes during object creation, after setting all properties.
function pushbutton1_CreateFcn(hObject, eventdata, handles)
% hObject        handle to pushbutton1 (see GCBO)
% eventdata     reserved - to be defined in a future version of MATLAB
% handles        empty - handles not created until after all CreateFcns called
function Fs_Callback(hObject, eventdata, handles)
% hObject        handle to Fs (see GCBO)
% eventdata     reserved - to be defined in a future version of MATLAB
% handles        structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Fs as text
%               str2double(get(hObject,'String')) returns contents of Fs as a double
% --- Executes during object creation, after setting all properties.
function Fs_CreateFcn(hObject, eventdata, handles)
% hObject        handle to Fs (see GCBO)
% eventdata     reserved - to be defined in a future version of MATLAB
% handles        empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%               See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```