Solutions to Problem Set 4

1. Let W denote the plim of $T^{-1}\Sigma w_{t+1}w'_{t+1}$ and let J be the zero-frequency spectral density of $x_t\varepsilon_{t+h}^{(h)}$, which can be written as $J=\gamma(0)+2\Sigma_{j=1}^{h-1}\gamma(j)$ under the stated assumptions, where $\gamma(j)$ is the autocovariance function of $x_t\varepsilon_{t+h}$. To prove the result, we only need to show that $W=J$. Now

$W=E(\varepsilon_{t+1}^2x_t^{(h)2})=E(\varepsilon_{t+1}^2(\Sigma_{s=1}^h x_{t+1-s})^2)=E(\varepsilon_{t+1}^2(\Sigma_{i=1}^h\Sigma_{j=1}^h x_{t+1-i}x_{t+1-j}))$

$\therefore W=E(\varepsilon_{t+1}^2(x_t^2+x_{t-1}^2...+x_{t-h+1}^2))+2\Sigma_{j=1}^{h-1}E(\varepsilon_{t+1}^2(x_tx_{t-j}+x_{t-1}x_{t+1-j}...+x_{t+1-h+j}x_{t+1-h}))$

Meanwhile

$\gamma(j)=E(r_{t+h}^{(h)}x_tr_{t+h-j}^{(h)}x_{t-j})=E((\varepsilon_{t+h}+\varepsilon_{t+h-1}...\varepsilon_{t+1})x_t(\varepsilon_{t+h-j}+\varepsilon_{t+h-j-1}...\varepsilon_{t-j+1})x_{t-j})$

$=E(\varepsilon_{t+1}^2x_tx_{t-j}+\varepsilon_{t+2}^2x_tx_{t-j}...+\varepsilon_{t+h-j}^2x_tx_{t-j})$

$=E(\varepsilon_{t+1}^2x_tx_{t-j}+\varepsilon_{t+1}^2x_{t-1}x_{t-1-j}...+\varepsilon_{t+1}^2x_{t+1-h+j}x_{t+1-h})$

Hence $J=E(\varepsilon_{t+1}^2x_t^2+\varepsilon_{t+1}^2x_{t-1}^2...+\varepsilon_{t+1}^2x_{t+1-h}^2)+2\Sigma_{j=1}^{h-1}E(\varepsilon_{t+1}^2(x_tx_{t-j}+x_{t-1}x_{t+1-j}...+x_{t+1-h+j}x_{t+1-h}))=W$ as required.

2. $E(r_t^4)=E(E(\sigma_t^4\varepsilon_t^4\mid\sigma_t))=3E(\sigma_t^4)$

$E(\sigma_t^2)=\omega/(1-\alpha)$

$\sigma_t^4=\omega^2+\alpha^2r_{t-1}^4+2\omega\alpha r_{t-1}^2$

$\therefore E(\sigma_t^4)=\omega^2+3\alpha^2E(\sigma_t^4)+2\omega\alpha\dfrac{\omega}{1-\alpha}$

$\therefore E(\sigma_t^4)(1-3\alpha^2)=\omega^2+\dfrac{2\omega^2\alpha}{1-\alpha}=\dfrac{\omega^2}{1-\alpha}(1-\alpha+2\alpha)=\dfrac{\omega^2(1+\alpha)}{1-\alpha}$

$\therefore E(\sigma_t^4)=\dfrac{1}{1-3\alpha^2}\dfrac{\omega^2(1+\alpha)}{1-\alpha}$

$\therefore E(r_t^4)=3\dfrac{1}{1-3\alpha^2}\dfrac{\omega^2(1+\alpha)}{1-\alpha}$, so long as $|\alpha|<1/\sqrt{3}$

$\therefore \dfrac{E(r_t^4)}{E(r_t^2)^2}=3\dfrac{1}{1-3\alpha^2}\dfrac{\omega^2(1+\alpha)}{1-\alpha}\dfrac{(1-\alpha)^2}{\omega^2}=\dfrac{3(1-\alpha^2)}{1-3\alpha^2}$

is the unconditional kurtosis.

3. These data were artificially generated by adding together a random walk and an iid time series:

$$y_t=x_t+u_t$$
$$x_t=x_{t-1}+\varepsilon_t$$

where the shocks are standard normal and mutually uncorrelated. Here is the program (adapted from James Lake) that answers the question (the data were saved in a file called data.mat).

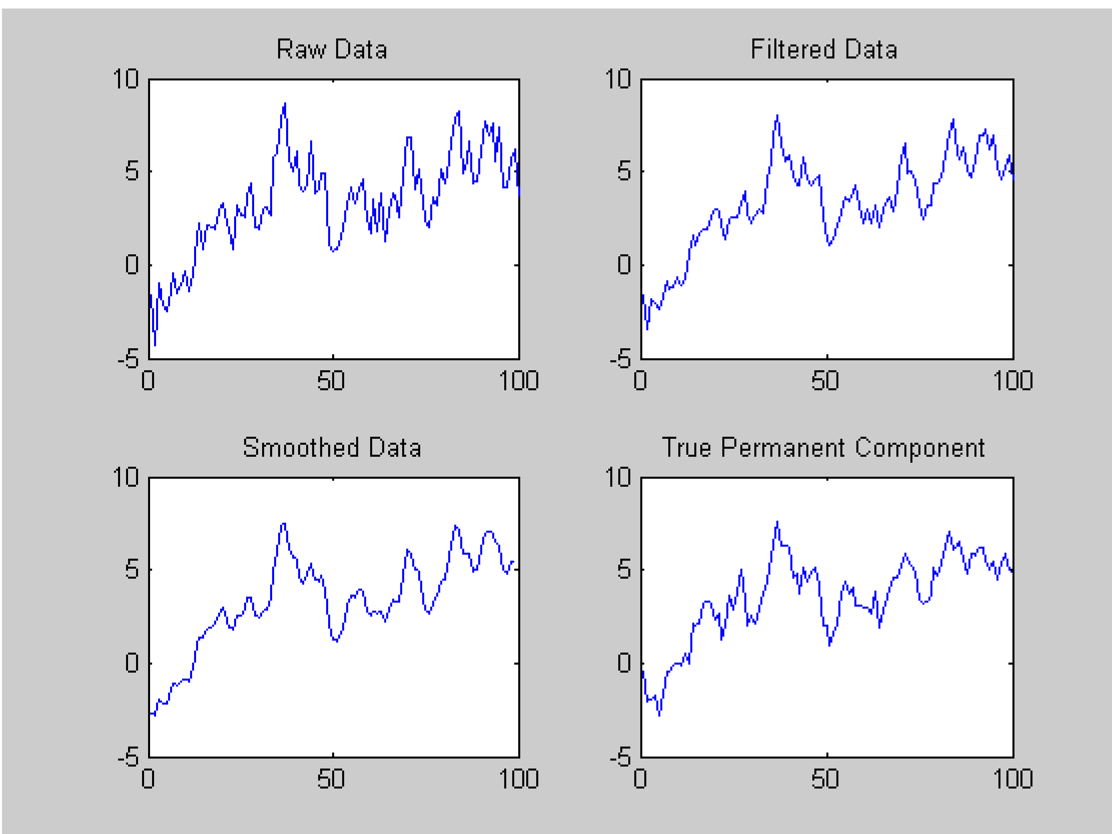
```
clear all;
clc;

%set up stuff
load data; %data are in y


%run MLE using KF_likelihood function
param0=[1;1];
param_est=fminsearch(@(param)KF_likelihood(param),param0);
sig_eps=param_est(1), sig_eta=param_est(2)  %variances of temporary and
permanent components, respectively

%use KF with above MLE estimates to estimate the state variable at t given
%info at t
[param_est_filter mu_u_filter p_u_filter
p_p_filter]=KF_likelihood(param_est);

%use Kalman smoother to estimate the state variabe at t given info a T
muT_u=mu_u_filter;
pT_u=p_u_filter;
pT_p=p_p_filter;
T=length(muT_u);
    %run backwards
for t=T-1:-1:1;
    p=pT_u(t)*inv(pT_p(t+1));
    muT(t)=muT_u(t)+ p*(muT_u(t+1)-muT_u(t));
    pT_u(t)=pT_u(t)+(p*(pT_u(t+1) - pT_p(t+1))*p);
end;

%plot all the data
figure(1);
subplot(2,2,1);
plot(y);
title('Raw Data');
subplot(2,2,2);
plot(mu_u_filter);
title('Filtered Data');
subplot(2,2,3);
plot(muT);
title('Smoothed Data');
subplot(2,2,4);
plot(trueperm);
title('True Permanent Component');
```

The program also calls this function

```
%likelihood function for Kalman filtering problem
function [log_L mu_u P_u P_p]=KF_likelihood(param)

%set up stuff
load data;
```

```
T=length(y);

%build up data
for t=1:T+1;
    if t==1; mu_p(t)=0;P_p(t)=1000;v(t)=0;F(t)=0;s_j(t)=0;  %initial values
    else F(t)=P_p(t-1)+param(1);
         v(t)=y(t-1)-mu_p(t-1);          %1 step ahead prediction error
         mu_u(t)=mu_p(t-1)+P_p(t-1)*inv(F(t))*v(t); %updating equation
P_u(t)=P_p(t-1)*(1-inv(F(t))*P_p(t-1));  %updating equation for variance
         mu_p(t)=mu_u(t);                %prediction equation for mean
         P_p(t)=P_u(t)+param(2);         %prediction equation for variance
         s_j(t)=-(T/2)*log(2*pi)-.5*log(F(t))- .5*(v(t)*inv(F(t))*v(t));
end
end
mu_u=mu_u(2:end)';
P_u=P_u(2:end)';
P_p=P_p(2:end)';
log_L=-sum(s_j(2:end));
```

The output of the program gives the estimates of the temporary and permanent variances, which are 0.7608 and 0.804, respectively. Then the program graphs (a) the actual data, (b) the filtered data, (c) the smoothed data, and (d) the true permanent component. Note that (d) is included as an extra just for reference—you could not have known this in the exercise.



Also notice that the filtered data is less "jagged" than the raw data, which makes sense because it is stripping out the temporary piece. The smoothed data is smoother again; actually a little smoother than the true random walk component.

4. I can rewrite the model as

$$\log(y_t^2) = \mu + h_t + \log(\varepsilon_t^2)$$
$$h_t = \phi h_{t-1} + \xi u_t$$
$$\mu = \omega / (1-\phi)$$
$$h_t = \log(\sigma_t^2) - \mu \Rightarrow \sigma_t = \sqrt{\exp(h_t)\exp(\mu)}$$

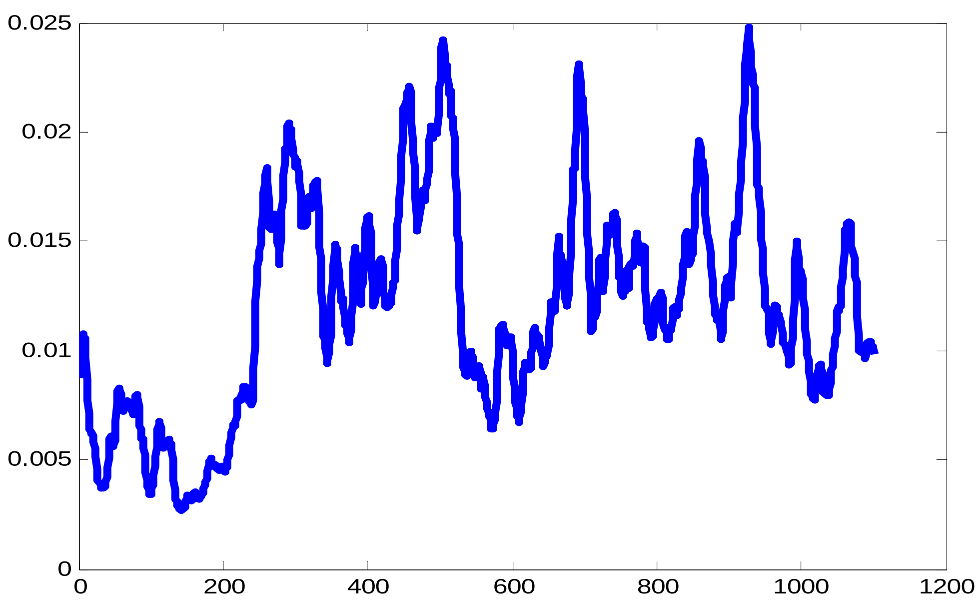
Here's the program for doing the smoothed estimates and the results.

```
y=xlsread('exrtlnd4.xls','Sheet1');
y=y(:,1);
n=length(y);
randn('seed',123); rand('seed',123);
burnin=100;
ndraw=5100;
r_p=0.086; r_m1=-7.472; r_m2=-0.698; r_sig2=1.411; r_sig=sqrt(r_sig2);
%Mixture parameters
omega=-0.17; phi=0.98; xi=0.2;
mu=(omega/(1-phi));   %Mean of log(sigs)
r_pt_eps = r_p*ones(n,1);
valpha=toeplitz(0.98.^[0:1:n]');  %Variance-covariance matrix of log(sigs)
valpha=valpha*(xi^2)/(1-(phi^2));

for idraw=1:ndraw;
[var_eps_n,r_pt_eps]=draw_var(y,r_pt_eps,mu,valpha);
     if idraw > burnin;
      sd_eps_n=sqrt(var_eps_n);
      sd_eps_save(:,idraw-burnin)=sd_eps_n;
     end;
end;


for i=1:n;
sd_eps(i,1)=median(sd_eps_save(i,:));
end;

sd_eps=sd_eps*sqrt(exp(mu));
plot(sd_eps,'Linewidth',3);
```

This program calls the following function

```
function [vardraw,r_pt]=draw_var(x,r_pt,mu,valpha);
r_p=0.086; r_m1=-7.472; r_m2=-0.698; r_sig2=1.411; r_sig=sqrt(r_sig2);
%Mixture parameters
n=size(x,1);
lnres2=log(x.^2)-mu;

%Step 1 -- initial draws of Indicator Variables %
 tmp=rand(n,1);
 ir = tmp<r_pt;
```

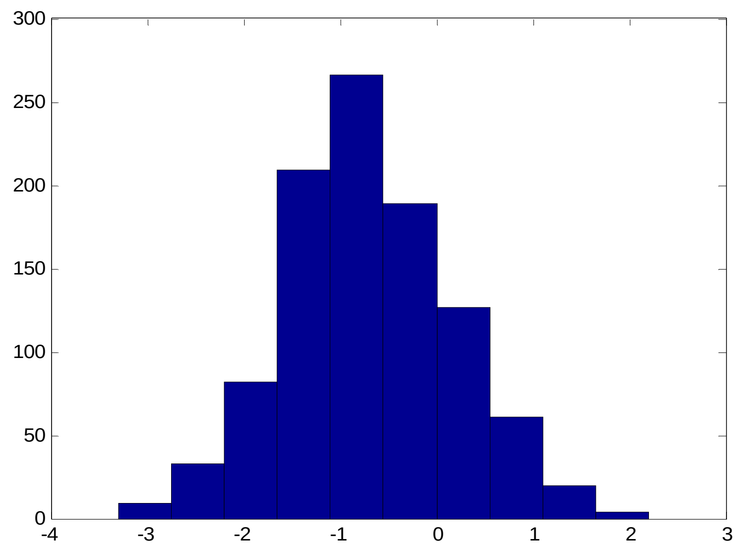
```
 tmp=rand(n,1);

% Step 2; compute system parameters given indicators %
 mut = (ir*r_m1) + ((1-ir)*r_m2);

% Compute Covariance Matrix  %
 vy=valpha(2:n+1,2:n+1);
 cy=valpha(1:n+1,2:n+1);
 diagvy=diag(vy)+r_sig2;
 for i=1:n; vy(i,i)=diagvy(i); end;
 vyi=inv(vy);
 kgain=cy*vyi;

%Compute draws of state and shocks %
 ye=lnres2-mut;
 ahat0=kgain*ye;
 ahat1=ahat0(2:n+1);
 vhat0=valpha-kgain*cy';
 cvhat0=chol(vhat0);
 adraw0=ahat0+cvhat0'*randn(n+1,1);
 adraw1=adraw0(2:n+1);
 edraw=lnres2-adraw1;

% Compute Mixture Probabilities %
  f1=exp(   (-0.5)* (((edraw-r_m1)./r_sig).^2)  );
  f2=exp(   (-0.5)* (((edraw-r_m2)./r_sig).^2)  );
  fe= r_p*f1 + (1-r_p)*f2;
  r_pt=(r_p*f1)./fe;


vardraw = exp(adraw1);
```



5. Here is the program for simulating the limiting distribution

```
n=1000;
lambda=0.5;
randn('seed',123);
for imc=1:1000;
    v=cumsum(randn(n,1))/sqrt(n);
    v=v(n*lambda:n);  %The BM from lambda to 1
    s=[lambda:1/n:1]';
    num1=sum(v(1:end-1).*(v(2:end)-v(1:end-1))./s(1:end-1));
    num2=sum((v.^2)./(s.^2))/n;
    num=(2*num1)-num2;
    denom=2*sqrt(num2);
    dmstat(imc)=num/denom;
end;
dmstat=sort(dmstat);
hist(dmstat)
[dmstat(25) dmstat(975)]
```

The histogram is



and the 2.5 and 97.5 percentiles are -2.30 and 1.09.

(b) If the goal is to test the hypothesis that $\beta_2 = 0$, it is a strange way of doing the test because an F-test will be more powerful.