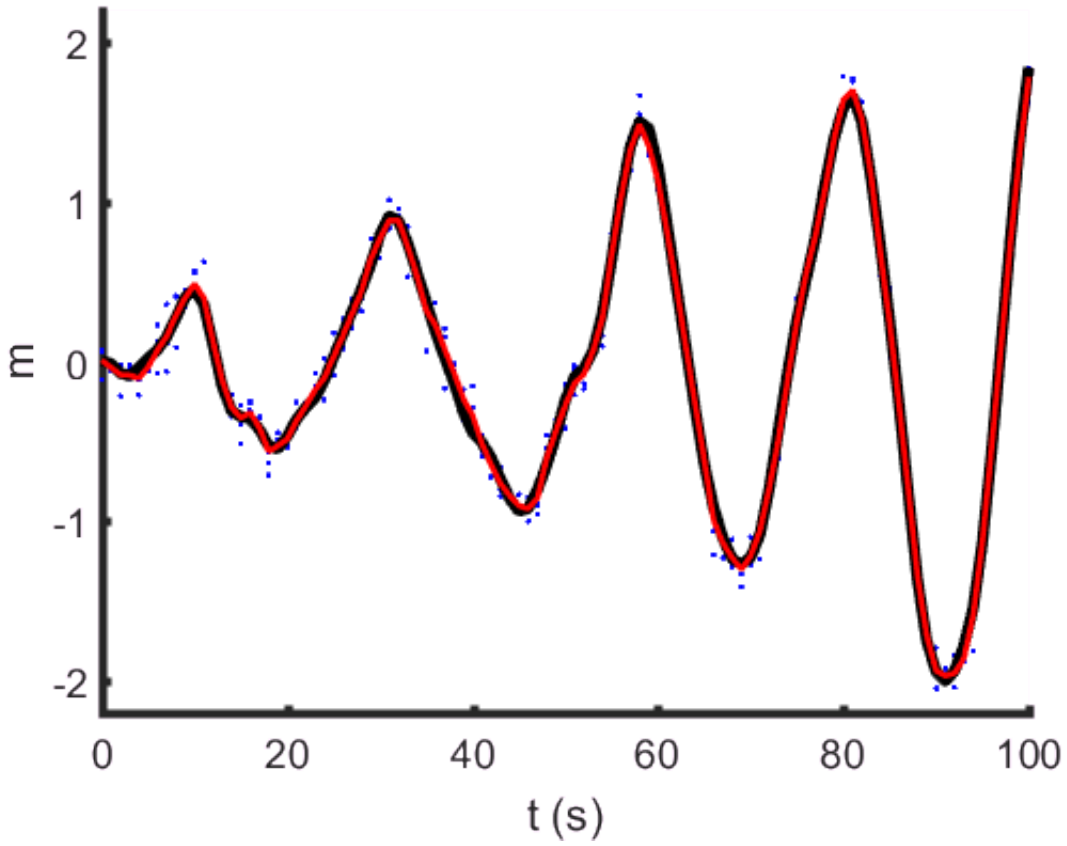
```
clear all;

% gda_05_17
% supports Figure 5.17
% sample chi-squared analysis
% distribution of errors P (Phi), E and L
% for a problem of a band-limited timeseries
% with redundant noisy observations of the
% time series and with a priori smoothness
% information

% time t variable
M=101;
Dt=1;
t = Dt*[0:M-1]';

% make a wiggly curve m(t) by starting out with random noise
% and bandpass filtering it around a narrow range of angular
% frquencies centered on w0.  The bandpass filtering is
% accomplished by the gda_chebyshevfilt() function, which
% though not mentioned in the text (sorry about that) is pretty
% easy to use and has many other applications.  Its arguments are
% output_timeseries = gda_chebyshevfilt( input_timeseries, Dt, f1, f2 )
% where Dt is the sampling of the timeseries (say in s) and
% (f1,f2) are the (low,high) size of the frequency band (say in Hz)
n = 10;
A = 2.0;
w0 = n*pi/t(end);
Dw = w0/10;
mtrue = gda_chebyshevfilt( random('Normal',0,1,M,1), Dt, (w0-Dw)/(2*pi), (w0+Dw)/(2*pi) );
mtrue = A*mtrue/max(abs(mtrue));

% components of data kernel, all normalize to produce
% data of about the same value

% data kernel is just 3 independent observations of same point
N = 3*M;
G = [eye(M,M); eye(M,M); eye(M,M)];
dtrue = G*mtrue;
sigma_d = 0.1;
dobs = dtrue + random('Normal',0,sigma_d,N,1);
sqrtcovdi = eye(N,N)/sigma_d;

% prior information: second derivative close to zero
H = (1/Dt^2)*toeplitz( [1; zeros(M-3,1)], [1, -2, 1, zeros(1,M-3) ] );
h = zeros(M-2,1);
K = M-2;
mddtrue = H*mtrue;
sigma_mdd = (w0^2)*A/sqrt(2);
sigma_mdd2 = std(mddtrue);  % for test purposes
sqrtcovhi = eye(K,K)/sigma_mdd;

% overall least squares equation and its solution
F = [ sqrtcovdi*G; sqrtcovhi*H ];
f = [ sqrtcovdi*dobs; sqrtcovhi*h ];
FTFinv = inv(F'*F);
mest = FTFinv*(F'*f);

mddest = H*mest;
```

```
sigma_mdd3 = std(mddest);  % for test purposes

% plot m
figure(1);
clf;
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [0, t(end), -1.1*A, 1.1*A] );
plot( t, dobs(1:M), 'b.', 'LineWidth', 2 );
plot( t, dobs(M+1:2*M), 'b.', 'LineWidth', 2 );
plot( t, dobs(2*M+1:3*M), 'b.', 'LineWidth', 2 );
plot( t, mtrue, 'k-', 'LineWidth', 4 );
plot( t, mest, 'r-', 'LineWidth', 2 );
xlabel('t (s)');
ylabel('m');
```




```
% Fig. 5.17  Test for the compatibility of the prior and posterior errors. (A) The smooth mode
% is measured by three different observers at a suite of closely-spaced points (blue dots).  T
% reconstructed (red) using the data together with the prior information that its second deriv

% chi-squared values
e = sqrtcovdi*(G*mest-dobs); % normalized prediction error
l = sqrtcovhi*(H*mest-h); % normalized prior error
E = e'*e; % Eobs
L = l'*l; % Lobs
P = E+L;  % PHIobs
vP = N+K-M; % degrees of freeedom in PHI
vE = vP*N/(N+K); % approx degrees of freeedom in E
vL = vP*K/(N+K); % approx degrees of freeedom in L
```

```
% Chi-squared test of the Null Hypothesis
fprintf('Analytic 95%% confidence tests\n');
```

```
Analytic 95% confidence tests
```

```
if ( P>(vP-2*sqrt(2*vP)) && P<(vP+2*sqrt(2*vP)) )
    fprintf('P %.2f vP %.2f null hypothesis cannot be rejected\n', P, vP);
else
    fprintf('P %.2f vP %.2f null hypothesis can be rejected\n', P, vP);
end
```

```
P 273.87 vP 301.00 null hypothesis cannot be rejected
```

```
if ( E>(vE-2*sqrt(2*vE)) && E<(vE+2*sqrt(2*vE)) )
    fprintf('E %.2f vE %.2f null hypothesis cannot be rejected\n', E, vE);
else
    fprintf('E %.2f vE %.2f null hypothesis can be rejected\n', E, vE);
end
```

```
E 225.11 vE 226.87 null hypothesis cannot be rejected
```

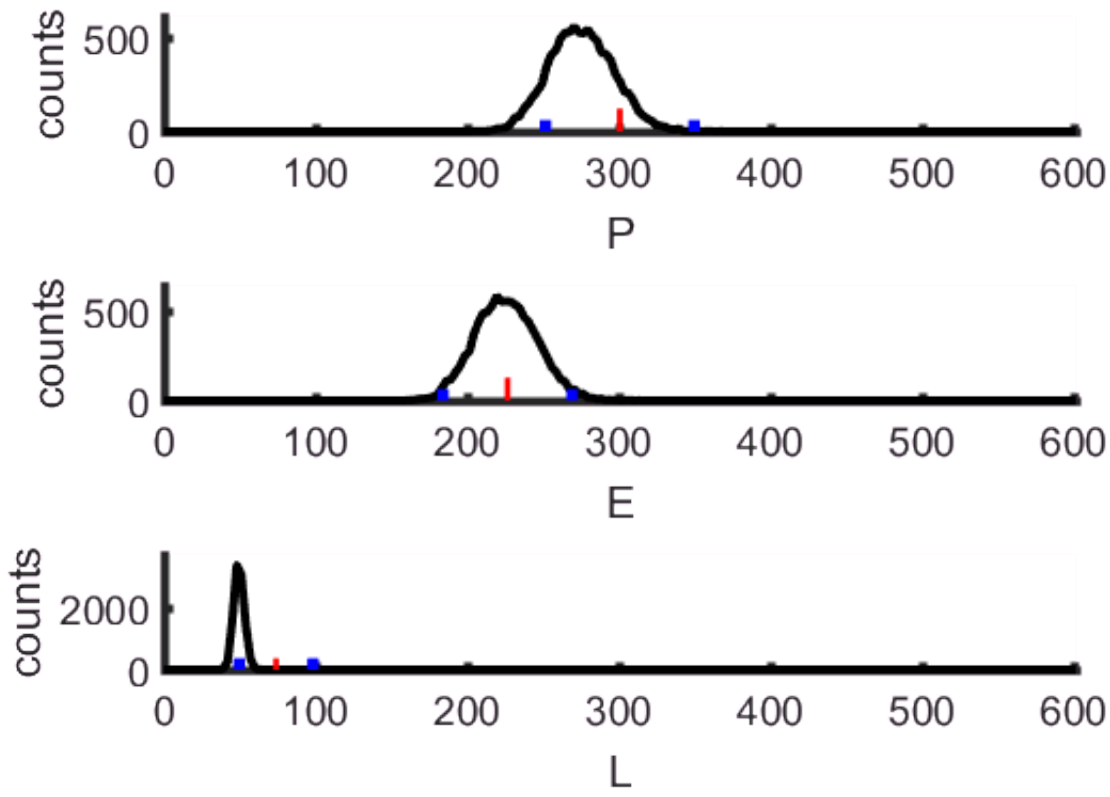
```
if ( L>(vL-2*sqrt(2*vL)) && L<(vL+2*sqrt(2*vL)) )
    fprintf('L %.2f vL %.2f null hypothesis cannot be rejected\n', L, vL);
else
    fprintf('L %.2f vL %.2f null hypothesis can be rejected\n', L, vL);
end
```

```
L 48.76 vL 74.13 null hypothesis can be rejected
```

```
fprintf('\n');
```

```
% Now do it lots of times to create empirical pdf's
% p(PHI), p(E) and p(L)
Nreal = 10000;
Pvec = zeros(Nreal,1);
Evec = zeros(Nreal,1);
Lvec = zeros(Nreal,1);
for ireal = [1:Nreal]
    dobs = dtrue + random('Normal',0,sigma_d,N,1);
    f = [ sqrtcovdi*dobs; sqrtcovhi*h ];
    mest = FTFinv*(F'*f);
    e = sqrtcovdi*(G*mest-dobs);
    l = sqrtcovhi*(H*mest-h);
    Evec(ireal) = e'*e;
    Lvec(ireal) = l'*l;
    Pvec(ireal) = Evec(ireal)+Lvec(ireal);
end
Evec = sort(Evec);
Lvec = sort(Lvec);
Pvec = sort(Pvec);

% empirical pdf's via histograms
Nbins = 201;
binmin = 0;
binmax = 2*vP;
Dbin = (binmax-binmin)/(Nbins-1);
```

```
bins = binmin + Dbin*[0:Nbins]';
Phist = hist( Pvec, bins );
Ehist = hist( Evec, bins );
Lhist = hist( Lvec, bins );

% plot histigram
figure(2);
clf;

% p(PHI)
subplot(3,1,1);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
top = 1.1*max(Phist);
axis( [binmin, binmax, 0, top ] );
plot( bins,Phist, 'k-', 'LineWidth', 3 );
plot( [vP, vP], [0, top/5], 'r-', 'LineWidth', 2 );
plot( [vP-2*sqrt(2*vP), vP-2*sqrt(2*vP)], [0, top/10], 'b-', 'LineWidth', 4 );
plot( [vP+2*sqrt(2*vP), vP+2*sqrt(2*vP)], [0, top/10], 'b-', 'LineWidth', 4 );
xlabel('P');
ylabel('counts');

% p(E)
subplot(3,1,2);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
top = 1.1*max(Ehist);
axis( [binmin, binmax, 0, top ] );
plot( bins,Ehist, 'k-', 'LineWidth', 3 );
plot( [vE, vE], [0, top/5], 'r-', 'LineWidth', 2 );
plot( [vE-2*sqrt(2*vE), vE-2*sqrt(2*vE)], [0, top/10], 'b-', 'LineWidth', 4 );
plot( [vE+2*sqrt(2*vE), vE+2*sqrt(2*vE)], [0, top/10], 'b-', 'LineWidth', 4 );
xlabel('E');
ylabel('counts');

% p(L)
subplot(3,1,3);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
top = 1.1*max(Lhist);
axis( [binmin, binmax, 0, top ] );
plot( bins,Lhist, 'k-', 'LineWidth', 3 );
plot( [vL, vL], [0, top/10], 'r-', 'LineWidth', 2 );
plot( [vL-2*sqrt(2*vL), vL-2*sqrt(2*vL)], [0, top/10], 'b-', 'LineWidth', 4 );
plot( [vL+2*sqrt(2*vL), vL+2*sqrt(2*vL)], [0, top/10], 'b-', 'LineWidth', 4 );
xlabel('L');
ylabel('counts');
```




```
% Fig. 5.17  Test for the compatibility of the prior and posterior errors. (B) Histogram (blac
% the total error Φ^est for 10,000 realizations of the data, compared to mean (red bar) and 95%
% interval (blue bars) of the corresponding chi-squared distribution. (C) Same as (B), but for
% error E^est. (D) Same as (C), but for a priori error L^est. MatLab script gda05_17.

% empirical confidence limits based on the empirical pdf's
fprintf('Numerical 95%% confidence tests\n');
```

```
Numerical 95% confidence tests
```

```
i=find(P<Pvec,1);
p = i/Nreal;
if ( p>0.025 && p<0.975 )
    fprintf('P %.2f vP %.2f null hypothesis cannot be rejected\n', P, vP);
else
    fprintf('P %.2f vP %.2f null hypothesis can be rejected\n', P, vP);
end
```

```
P 273.87 vP 301.00 null hypothesis cannot be rejected
```

```
i=find(E<Evec,1);
p = i/Nreal;
if ( p>0.025 && p<0.975 )
    fprintf('E %.2f vE %.2f null hypothesis cannot be rejected\n', E, vE);
else
    fprintf('E %.2f vE %.2f null hypothesis can be rejected\n', E, vE);
end
```

```
E 225.11 vE 226.87 null hypothesis cannot be rejected
```

```
i=find(L<Lvec,1);
p = i/Nreal;
if ( p>0.025 && p<0.975 )
    fprintf('L %.2f vL %.2f null hypothesis cannot be rejected\n', L, vL);
else
    fprintf('L %.2f vL %.2f null hypothesis can be rejected\n', L, vL);
end
```

```
L 48.76 vL 74.13 null hypothesis cannot be rejected
```