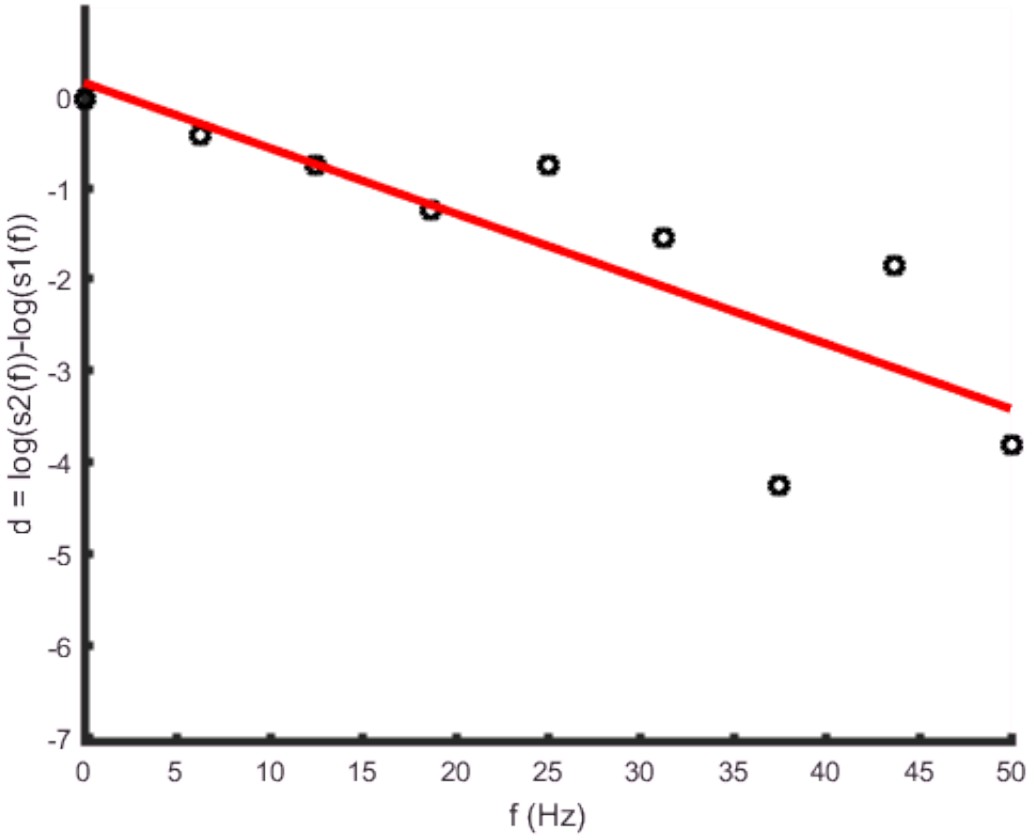
```
% gda14_06
% This code parallels "Method Summary 5,
%                             Bootstrap Confidence Intervals"

clear all;

% Background:
% This example relies on syntheic data created by the
% make_pulses_for_3.m script.  That script makes two
% true pulses p1(t) and p2(t) that have an amplitude
% spectral density of exactly
%         s1(f)=A1*exp(-pi*abs(f)*tstar1) and
%         s2(f)=A2*exp(-pi*abs(f)*tstar2)
% with tstar1=0.1 and tstar2=0.03.  However, random noise
% is added to the pulses before computing their spectrum,
% so the observed pulses do not have exactly this spectrum.
%
% Note that if we subtracted the logarithms of the spectra
% we would get the straight line
% dobs(f) = log(s2)-log(s1) = (A2-A1) + (-pi*(tstar2-tstar1))*f
% So we could determine Dtstar=tstar2-tstar1 simply by
% fitting a straight line to the data.  However, the
% 95% confidence of this slope might not be the usual
% least squares confidence, because the process of taking
% the logarithm affects the statics of the noise; it
% will not in general be Normal.
%
% So we will calculate the 95% confidence of Dtstar by
% bootstrapping and compare it to the usual least squares
% value
%
% the data file 'pulses.txt' has three columns, frequency,
% log(s1(f)) and log(s2(f))

% Step 1: Identify the parameter for which confidence intervals are needed
% The parameter q is the differntial tstar; that is, q=Dtstar

% Step 2: Identify the data and a method of estimating ? from the data
% the data are the difference in log spectra; that is,
% dobs(f) = log(s2)-log(s1)

% load data
D = load('pulses.txt');
f = D(:,1);
logs1obs = D(:,2);
logs2obs = D(:,3);
dobs = logs2obs - logs1obs;
N = length(dobs);

% plot data
figure(1);
clf;
hold on;
set(gca,'LineWidth',3);
dmax = max(dobs);
axis( [f(1), f(end), dmax-7, dmax+1] );
plot( f, dobs, 'ko', 'LineWidth', 2 );
xlabel('f (Hz)');
ylabel('d = log(s2(f))-log(s1(f))');
```

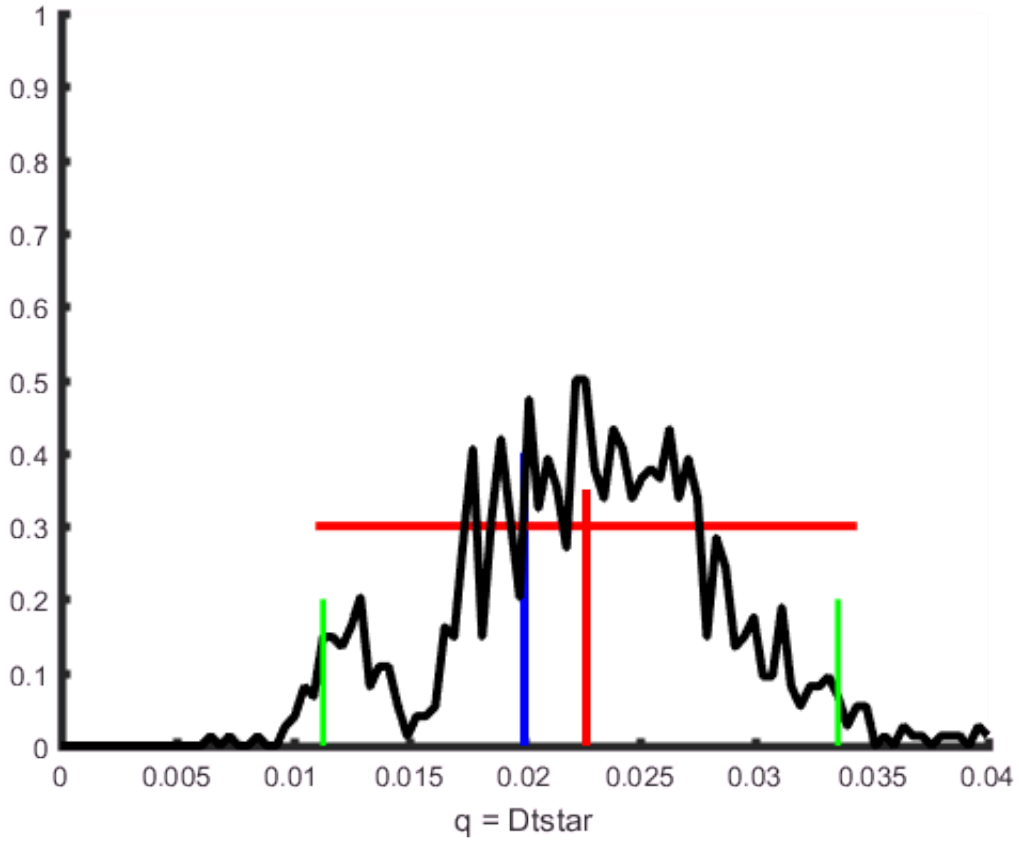
```
% true Dtstar, know because the data are synthetic
Dtstartrue = 0.02;
qtrue = Dtstartrue;

% Step 3: For reference, estimate solution mest
% from the observed data dobs

% do an initial least squares fit to determine
% tstar and the least squares estimate of its
% 95% confidence
M = 2;
G = [ones(N,1), f];
mest = (G'*G)\(G'*dobs);
dpre = G*mest;

% plot the fit (to check that it looks right)
figure(1);
plot( f, dpre, 'r-', 'LineWidth', 3 );
```




```
% Figure 1. Log-linear plot of the data (circles) and the fit (line)

% estimate the sqrt(variance) of Dtstar
% in the normal least squares way; this estimate
% can be compared to the bootstrap result
% (such an estimate may not always be available)
e = dobs-dpre;
E = e'*e;
sigma2d = E/(N-M);
covm = sigma2d*inv(G'*G);
```

```
Dtstarest = -mest(2)/pi;
qest = Dtstarest;
sigmaDtstar = sqrt(covm(2,2))/pi;
sigmaq = sigmaDtstar;

% now plot the estimate of Dtstar on a plot
% whose horizontal axis is Dtstar
% the true value of Dtstar is the vertical blue bar
% the least squares estimate of Dtstar is the vertical red bar
% and its 95% confidence is the horizontal red bar
figure(2);
clf;
hold on;
set(gca,'LineWidth',3);
axis( [0, 2*qtrue, 0, 1] );
plot( [qtrue, qtrue]', [0, 0.4], 'b-', 'LineWidth', 3 );
plot( [qest, qest]', [0, 0.35], 'r-', 'LineWidth', 3 );
plot( [qest-2*sigmaq, qest+2*sigmaq]', [0.3, 0.3], 'r-', 'LineWidth', 3 );
xlabel('q = Dtstar');

% now use the bootstrap method to compute an empirical
% probability density function of Dtstar and plot it
Nr = 1000;  % number of realizations
qrs = zeros(Nr,1);  % tabulate the q (=Dtstar) of each realization
for ir = [1:Nr]  % loop over realizations
    % randomly resample the data
    j = unidrnd( N, N, 1 );
    frs = f(j); % resampled frequencies
    drs = dobs(j); % resampled data
    G = [ones(N,1), frs]; % construct the data kernel
    mrs = (G'*G)\(G'*drs); % solve the least squares problem
    qrs(ir) = -mrs(2)/pi; % tabulate the Dtstar
end

% Now construct a histogram of the Dtstar values
% and scale it to an empirical probability density function
Nbin = 100; % number of bins in histogram
binmin = 0; % range of histogram
binmax = 2*qtrue;
Dbin = (binmax-binmin)/(Nbin-1);
bins = binmin + Dbin*[0:Nbin-1]';
p = hist(qrs,bins); % histogram = empirical pdf p(q)
p = p/(Dbin*sum(p));  % normalize to unit area
pmax = max(p); % find maximum for plotting purposes
plot( bins, 0.5*p/pmax, 'k-', 'LineWidth', 3 );
P = Dbin*cumsum(p); % integrate to an empirical cumulative distribution P(q)

% use the cumulative distribution to find the left and right
% edges of the 95% confidence interval and plot then as
% green bars.  The upshot is that they agree pretty well with
% the least squares estimate (red horizontal bar).  Because of
% the tendency of the logarithm to preferentially amplify
% small values, the center of the pdf is centered a little
% to the left of the true Dtstar value.  However, the displacement
% is small compared to the width of the confidence interval.
il = find( P>0.025, 1 );
qL = bins(il);
ir = find( P>0.975, 1 );
qR = bins(ir);
plot( [qL, qL]', [0, 0.2]', 'g-', 'LineWidth', 2 );
```

```
plot( [qR, qR]', [0, 0.2]', 'g-', 'LineWidth', 2 );
```




```
% Figure 2.  Histogram of the values of q (black curve), which we
% interpret as an empirical probability denisty function p(q).  The
% 95% confidence interval (green bars) encompasses 95% or the
% area of the p.d.f. The true value of q (ble bar) lies within this
% interval.  The linearized prediction of the mean (vertical red
% bar) and 95% confidence interval (red bar) are close to those
% of the empirical p.d.f., indicating that the bootstrapping is
% not necessary for this particular scenario.
```