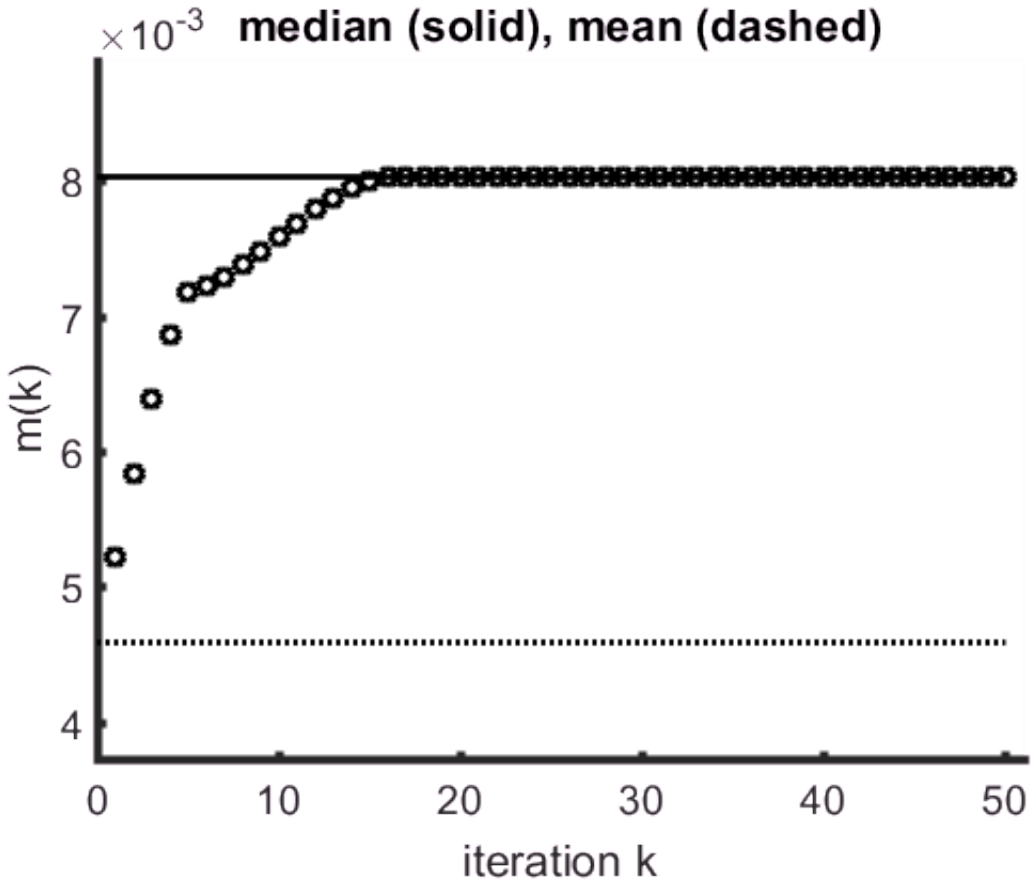
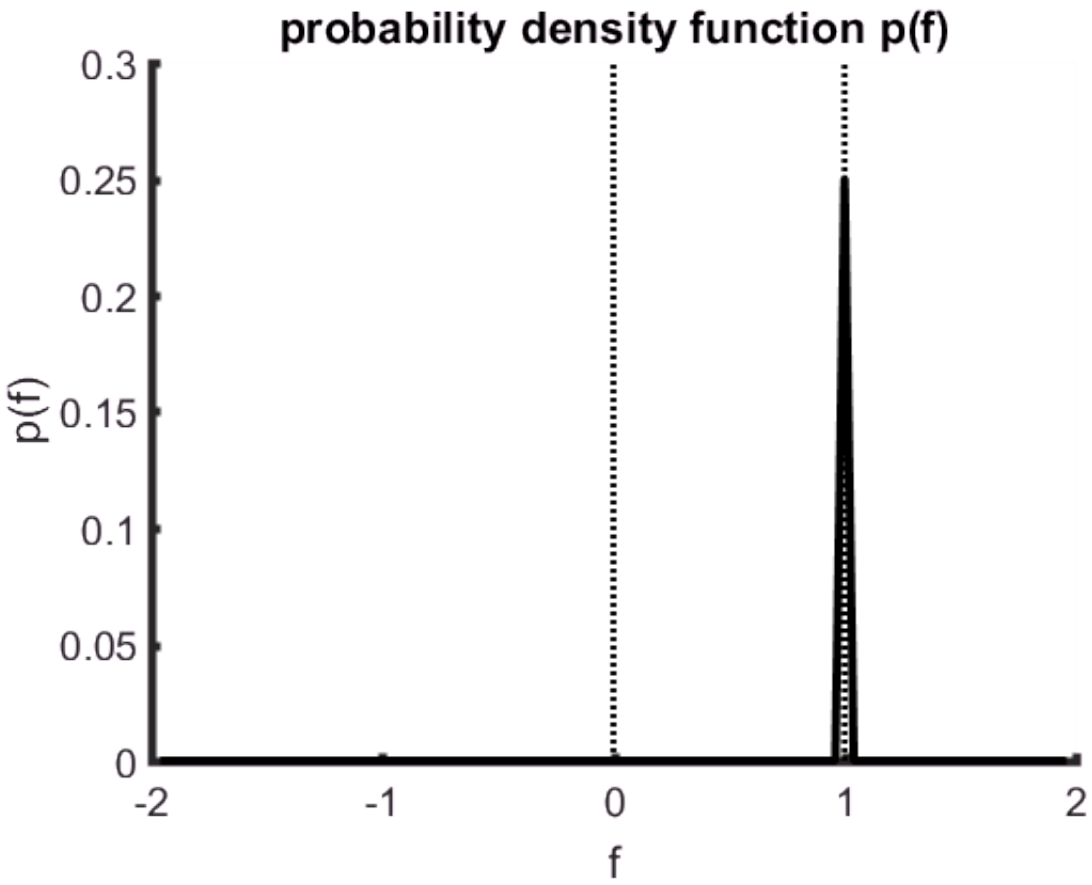
```
% gda08_04
% supports Figure 8.4
% example of solving an L1 problem by iterating
% a weighted solution to an L2 problem.  The example chosen here is
% [d1 d2 ... dN]' = [1 1 ... 1]' m
% since we know that the solution is the mean when the dobs is
% Normally-distributed, that is, mest=mean(dobs) and that the
% solution is the median when dobs is exponentially-distributed,
% that is mest=median(dobs).  The quantity f = (dmean-mest)/(dmean-dmedian)
% is used as a quality of solution indicator.  It measures how
% much closer the solution is to the median than the mean.
% The scripts creates the empirical probability density function p(f).

% Note: The script is a bit sluggish because the L1 problem
% is being solved many time to generate a histogram

clear all;

% Allows you to toggle between different ways of initializing
% the solution.
USEL2 = 1;
USETRUE = 0;
USEZERO = 0;
USEONE = 0;

% Allows you to toggle between simpe formula for weigths and
% Frommlet & Nuel's (2016) more complicated (but equivalent and
% more numerically stable version)
USEFN = 1;

% true solution
mtrue = 0;

% setup for histogram
Nbins=101;
binleft = (-2);
binright = 2;
Dbin=(binright-binleft);
bins = binleft + Dbin*[0:Nbins-1]'/(Nbins-1);
figure(2);
clf;
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
pmax = 0.3;
axis( [binleft, binright, 0, pmax] );
plot( [0;0], [0;pmax], 'k:', 'LineWidth', 2 );
plot( [1;1], [0;pmax], 'k:', 'LineWidth', 2 );

Nreal = 100; % number of realizations
f = zeros(Nreal,1); % (mest-mmedian)/(mmean-mmedian)
for j=[1:Nreal]  % loop over realizations

% exponentially-distributed noise
N = 101; % must be odd so that median is unique

% generate exponentially-distributed random noise
smp = 0.1;
mu = smp/sqrt(2);
```

```
rsign = (2*(random('unid',2,N,1)-1.5));
dn = rsign.*random('exponential',mu,N,1);

% set up problem
G = ones(N,1);     % data kernel says d(i) = 1 m(1)
dtrue = G*mtrue;   % true data
dobs = dtrue + dn; % observed data is true data plus noise
dmean=mean(dobs);  % sample mean is poor estimate of m(1)
dmedian = median(dobs);  % median is good estimate of m(1)
H = abs( dmean - dmedian ); % for plotting purposes

Ni=50;  % number of iterations of solution
if( j==1 ) % plot solution as a function of iteration,
           % but only for first realization
    figure(1);
    clf;
    set(gca,'LineWidth',3);
    set(gca,'FontSize',14);
    hold on;
    axis( [0, Ni+1, min(dmean,dmedian)-H/4, max(dmean,dmedian)+H/4] );
    plot( [0; Ni], [dmean, dmean], 'k:', 'LineWidth', 2 );
    plot( [0; Ni], [dmedian, dmedian], 'k-', 'LineWidth', 2 );
    xlabel('iteration k');
    ylabel('m(k)');
    title('median (solid), mean (dashed)');
end

% setup for iterative solution
delta = 1e-5;
gamma = 2;
n = 1;
% initial solution
if( USEL2 == 1 )
    W = eye(N);
    mest =  (G'*W*G)\(G'*W*dobs);  % L2 solution
elseif( USETRUE == 1 )
    mest = mtrue;                  % true solution
elseif( USEZERO == 1 )
    mest = 0;                      % zero solution
elseif( USEONE == 1 )
    mest = 1;                      % unity solution
end
for i = [1:Ni] % iterative solution
  e = dobs-G*mest;  % prediction error
  if( USEFN ) % Frommlet & Nuel's (2016) weight formula
    w = zeros(N,1);
    ae = abs(e);
    for k=[1:N]
        if( ae(k)>=delta )
           w(k)=(delta^(n-2))*exp(((n-2)/gamma)*log1p((ae(k)/delta)^gamma));
        else
           w(k)=(ae(k)^(n-2))*exp(((n-2)/gamma)*log1p((delta/ae(k))^gamma));
        end
    end
  else % simple weight formula
    w = ((ae.^gamma) + (delta^gamma) ).^((n-2)/gamma);  % weight matrix
  end
  W = diag(w);
  mest = (G'*W*G)\(G'*W*dobs); % L2 solution
  if( j==1 ) % plot, but only first realization
      plot( [i], [mest], 'ko', 'LineWidth', 2 );
```

```
  end
end % end loop over iterations in solution

% f is a quality of solution indicator
% the solution should be the median, not the mean
% f is normalized closeness of solution from medium
f(j) = (dmean-mest)/(dmean-dmedian);

end  % end loop over realizations
```




```
% Figure 8.4 (A) When applied to the first exemplary problem, the iterative algorithm rapidly
% converges to the median of the data (red line, the correct value). (B) An empirical probabil
% density function p(f) for the solution quality factor f is strongly peaked around f=1 (red l
% the correct value). probability density for f

p = hist( f, bins );
p = p/(Dbin*sum(p));
figure(2);
plot( bins(2:end-1), p(2:end-1), 'k-', 'LineWidth', 3 );
xlabel('f');
ylabel('p(f)');
title('probability density function p(f)');
```




```
% Figure 8.4 (B) An empirical probability density function p(f) for the solution quality facto
% f is strongly peaked around f=1 (red line, the correct value). probability density for f
```