```
% gda09_03
%
% An inverse problem to determine the model parameters
% in a model with an exponential function, d(z)=m1*exp(m2 z)
% using (Case 1) a linearizing transformation
% and (Case 2) Newton's Method.
% Supports Figure 9.2

clear all;

% auxiallary variable z
N=20;
zmin = 0;
zmax = 1;
Dz = (zmax-zmin)/(N-1);
z = zmin + Dz*[0:N-1]';

% data, a exponential d=m1*exp(m2 z)
M=2;
mtrue = [0.9, -4]';
dtrue = mtrue(1)*exp(mtrue(2)*z);

% additive noise
sd=0.02;
bottom= 0.02;
dobs = dtrue + random('Normal',0,sd,N,1);
% but don't let data become negative
i=find(dobs<0);
dobs(i)=bottom;

% linearizing transformation, solve by standard least squares
% d     = m1 exp( m2 z)
% ln(d) = ln(m1) + m2 * z
G=zeros(N,M);
G=[ones(N,1), z];
lndobs = log(dobs);
mestlin = (G'*G)\(G'*lndobs);
mestlin(1) = exp(mestlin(1));
dprelin = mestlin(1)*exp(mestlin(2)*z);

% Newton's Method iterative solution
% d     = m1 exp( m2 z)
% dd/dm1 = exp( m2 z)
% dd/dm2 = m1 z exp( m2 z)

% initial guess is previous solution
mest = mestlin;
Nit=10; % maximum number of iterations
sdmmin = 1e-5; % terminate iteration early when increment dm
               % has sqrt(variance(dm))<sdmmin
for it=[1:Nit]
    % Newton's Method
    dd = dobs - (mest(1)*exp(mest(2)*z));
    G = [exp(mest(2)*z), mest(1)*z.*exp(mest(2)*z)];
    dm = (G'*G)\(G'*dd);
    mest = mest+dm;
    sdm = sqrt(sum(dm'*dm)/M);  % sqrt(variance(dm))
    if( sdm < sdmmin )
        break; % terminate iterations early
```

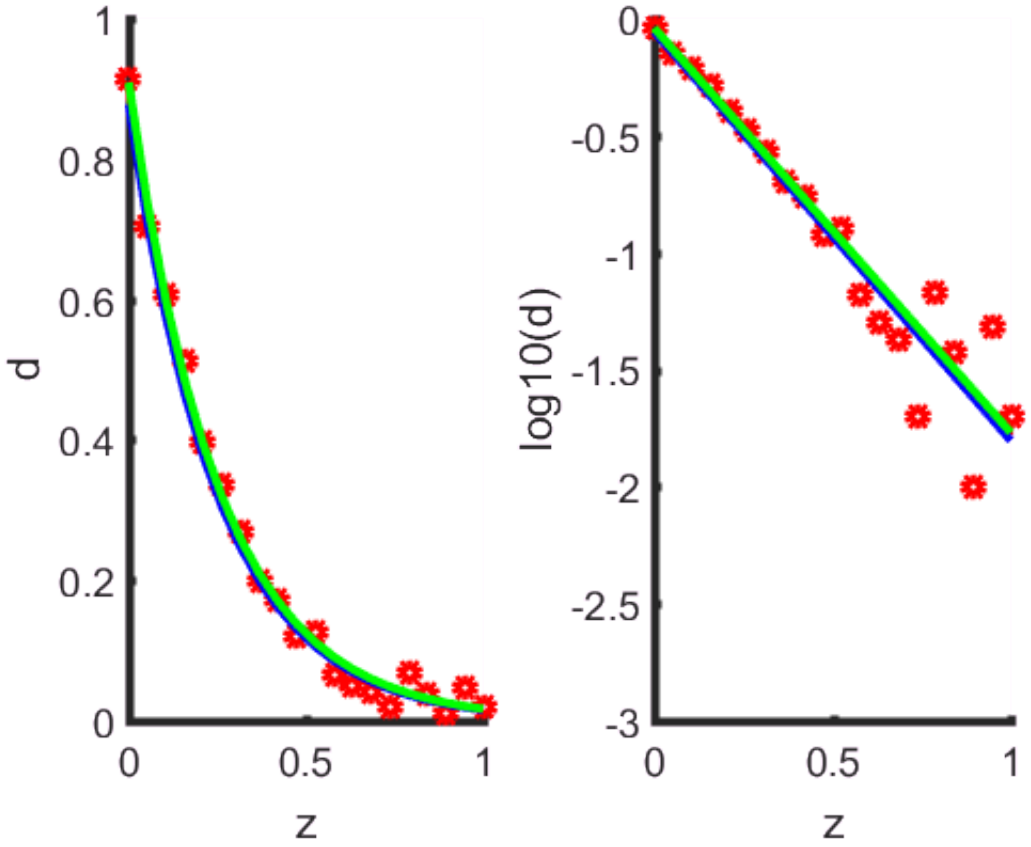
```
    end
end

% least squares prediction
dpre = mest(1)*exp(mest(2)*z);

figure(1);
clf;

% linear plot
subplot(1,2,1);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [zmin, zmax, 0, 1] );
plot( z, dobs, 'ro', 'LineWidth', 3 );
plot( z, dtrue, 'r-', 'LineWidth', 3 );
plot( z, dprelin, 'b-', 'LineWidth', 3 );
plot( z, dpre, 'g-', 'LineWidth', 3 );
xlabel('z');
ylabel('d');

% log plot
subplot(1,2,2);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [zmin, zmax, -3, 0] );
plot( z, log10(dobs), 'ro', 'LineWidth', 3 );
plot( z, log10(dtrue), 'r-', 'LineWidth', 3 );
plot( z, log10(dprelin), 'b-', 'LineWidth', 3 );
plot( z, log10(dpre), 'g-', 'LineWidth', 3 );
xlabel('z');
ylabel('log10(d)');
```




```
% Figure 9.2 (A) Least-squares fit to the exponential function,d_i=m_1  exp#(m_2 z_i ).
% (Red curve) The true function,d(z), for(m_1,m_2 )=(0.7,-4) (Red circles) Data that
% include Normally distributed random noise with zero mean and variance,σ_d^2=(0.2)^2.
% (Green curve) Nonlinear least-squares solution using Newton's method. (Blue curve)
% Least-squares solution using the linearizing transformation,
% log#〖d_i 〗=log#〖m_1 〗+m_2 z_i. Note that the two solutions are different.
% (B) Log-linear version of the graph in (A). Note that the scatter of the data
% increases with z, and that the solution based on the linearizing transformation
% is strongly affected by outliers associated with it. MatLab script gda09_03.

fprintf('True        m1: %.3f m2: %.3f\n', mtrue(1), mtrue(2) );
```

```
True        m1: 0.900 m2: -4.000
```

```
fprintf('Newtons     m1: %.3f m2: %.3f\n', mest(1), mest(2) );
```

```
Newtons     m1: 0.909 m2: -3.978
```

```
fprintf('Linearizing m1: %.3f m2: %.3f\n', mestlin(1), mestlin(2) );
```

```
Linearizing m1: 0.878 m2: -4.021
```