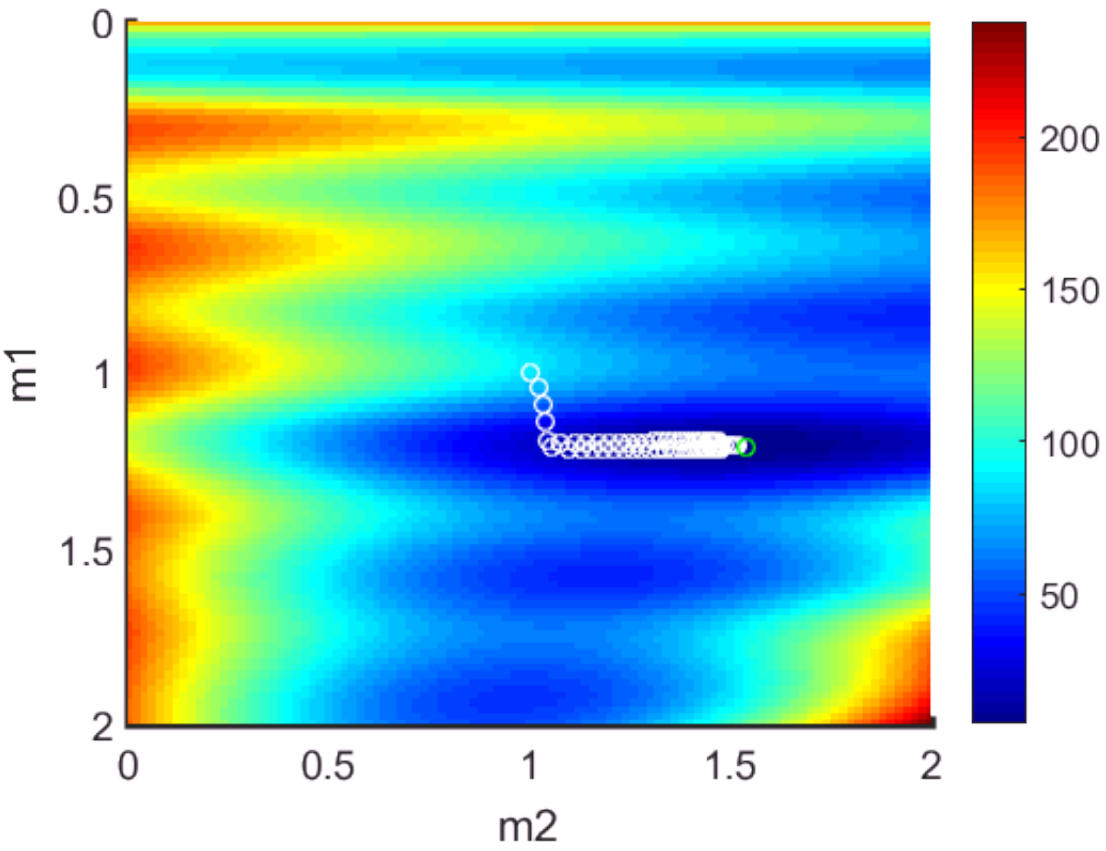
```
% gda09_15
% Gradient method applied to inverse problem
% d(x)=g(x, m1, m2) with g = sin(w0*m(1)*x) + m(1)*m(2)
% Supports Figure 9.13

clear all;

% Auxially variable x
N=40;
x = [0:N-1]'/N;

% true model parameters
mt = [1.21, 1.54]';

% true data
w0=20;
dtrue = sin(w0*mt(1)*x) + mt(1)*mt(2);

% observed data
sd=0.4;
dobs = dtrue + random('Normal',0,sd,N,1);

% plot true and observed data
figure(1);
clf;
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
plot(x,dobs,'ko','LineWidth',3);
plot(x,dtrue,'k-','LineWidth',3);
xlabel('x');
ylabel('d(x)');
hold on;

% 2D (m1,m2) grid
M = 101;
dm = 0.02;
m1min=0;
m2min=0;
m1a = m1min+dm*[0:M-1]';
m2a = m2min+dm*[0:M-1]';
m1max = m1a(M);
m2max = m2a(M);

% tabulate E(m1,m2) on grid (for plotting purposes only)
E = zeros(M,M);
for j = [1:M]
for k = [1:M]
    yest = sin(w0*m1a(j)*x) + m1a(j)*m2a(k);
    E(j,k) = (dobs-yest)'*(dobs-yest);
end
end

% plot error surface
figure(2);
clf;
set(gca,'LineWidth',3);
hold on;
set(gca,'FontSize',14);
```

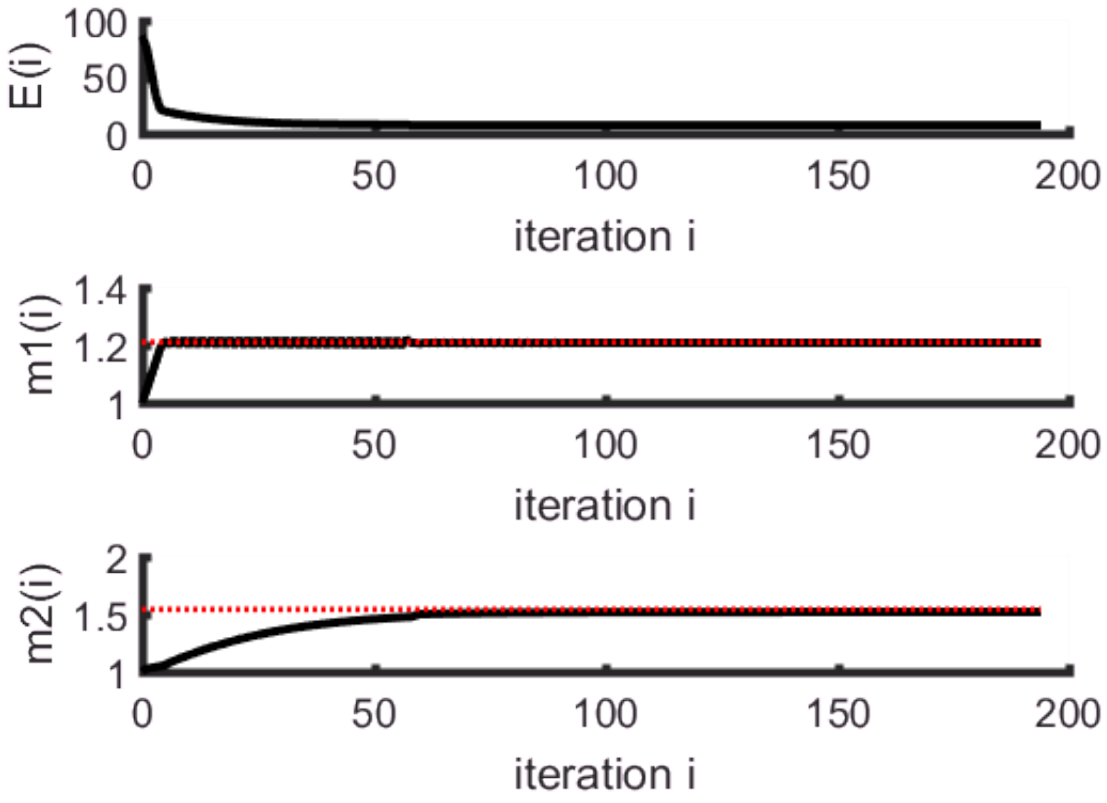
```
colormap('jet');
xlabel('m2');
ylabel('m1');
axis( [m2min, m2max, m1min, m1max] );
axis ij;
imagesc( [m2min, m2max], [m1min, m1max], E);
colorbar;

% parameters for gradient method
alpha = 0.05;
c1 = 0.0001;
c2 = 0.9;
tau = 0.5;

% trial solution
mgo=[1,1]';
plot( mgo(2), mgo(1), 'wo' );

% save history of Error and model parameters
Niter=500;
Ehis=zeros(Niter+1,1);
m1his=zeros(Niter+1,1);
m2his=zeros(Niter+1,1);

% error and its gradient at the trial solution
ygo = sin( w0*mgo(1)*x) + mgo(1)*mgo(2);
Ego = (ygo-dobs)'*(ygo-dobs);
dydmo = zeros(N,2);
dydmo(:,1) = w0 * x .* cos( w0 * mgo(1) * x ) + mgo(2);
dydmo(:,2) = mgo(2)*ones(N,1);
dEdmo = 2*dydmo'*(ygo-dobs);

for k = [1:Niter]

    % downhill direction
    v = -dEdmo / sqrt(dEdmo'*dEdmo);

    % save history
    if( k==1 )
        Ehis(1)=Ego;
        m1his(1)=mgo(1);
        m2his(1)=mgo(2);
    end

    % backstep
    for kk=[1:10]
        mg = mgo+alpha*v;
        yg = sin( w0*mg(1)*x) + mg(1)*mg(2);
        Eg = (yg-dobs)'*(yg-dobs);
        dydm = zeros(N,2);
        dydm(:,1) = w0 * x .* cos( w0 * mg(1) * x ) + mg(2);
        dydm(:,2) = mg(2)*ones(N,1);
        dEdm = 2*dydm'*(yg-dobs);
        if( (Eg<=(Ego + c1*alpha*v'*dEdmo)) )
            break;
        end
        alpha = tau*alpha;
    end

    % change in solution
    Dmg = sqrt( (mg-mgo)'*(mg-mgo) );
```

```
    % update
    mgo=mg;
    ygo = yg;
    Ego = Eg;
    dydmo = dydm;
    dEdmo = dEdm;
    Ehis(1+k)=Eg;
    m1his(1+k)=mg(1);
    m2his(1+k)=mg(2);
    plot( mg(2), mg(1), 'wo' );

    if( Dmg < 1.0e-6 )
        break; % terminate iterations when change
               % in solution is suffiently small
    end
end

% plot true soluton
plot( mt(2), mt(1), 'go' );
```



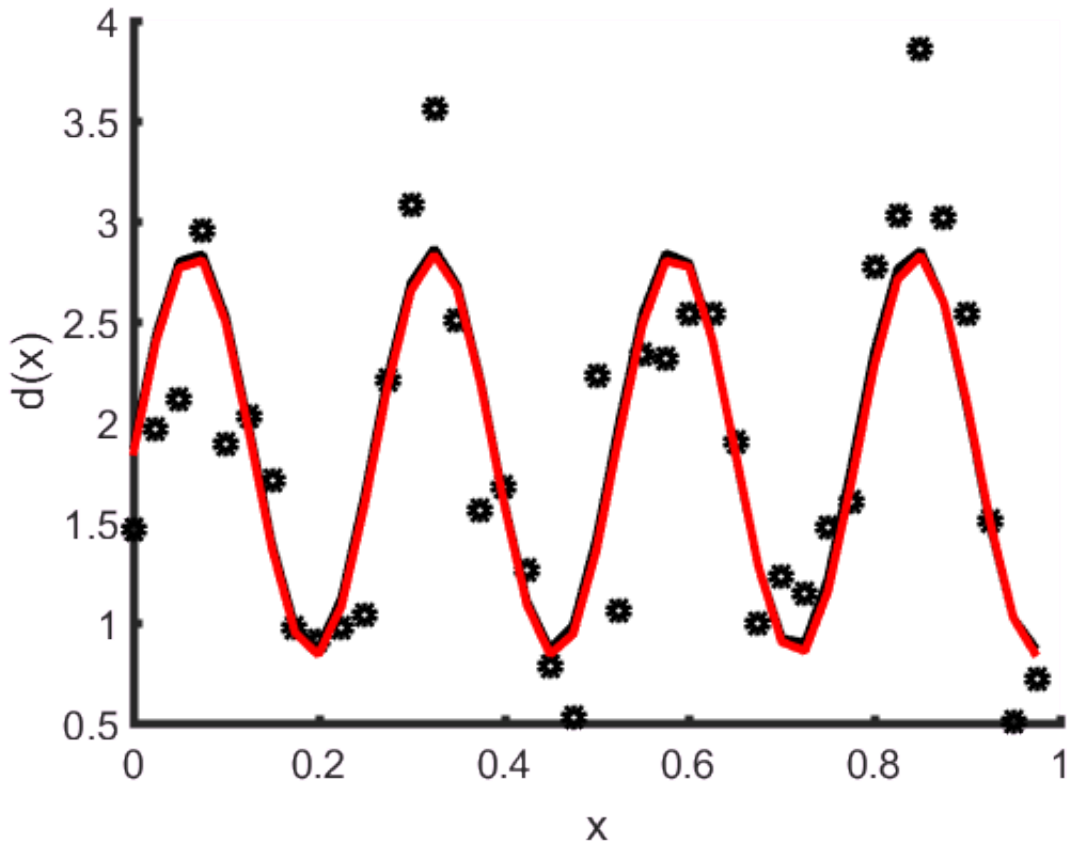

```
% Figure 9.13 Gradient method is used to solve the same nonlinear curve-fitting problem as in
% Figure 9.5. (B) Error surface (colors), showing true solution (green dot), and a series of i
% (white circles) determined by the method. (C) Plot of error E and model parameters m1 and m2
% a function of iteration number. MatLab script gda09_15.

% plot history of E, m1, m2
figure(3);
clf;
subplot(3,1,1);
set(gca,'LineWidth',3);
```

```
set(gca,'FontSize',14);
hold on;
plot( [0:k], Ehis(1:k+1), 'k-', 'LineWidth', 3 );
xlabel('iteration i');
ylabel('E(i)');

subplot(3,1,2);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
plot( [0:k], m1his(1:k+1), 'k-', 'LineWidth', 3 );
plot( [0, k], [mt(1), mt(1)], 'r:', 'LineWidth', 2);
xlabel('iteration i');
ylabel('m1(i)');

subplot(3,1,3);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
plot( [0:k], m2his(1:k+1), 'k-', 'LineWidth', 3 );
plot( [0, k], [mt(2), mt(2)], 'r:', 'LineWidth',2 );
xlabel('iteration i');
ylabel('m2(i)');
```




```
% Figure 9.13 Gradient method is used to solve the same nonlinear curve-fitting problem as in
% Figure 9.5. (C) Plot of error E and model parameters m1 and m2 as
% a function of iteration number. MatLab script gda09_15.

% evaluate prediction and plot it
figure(1);
dpre = sin(w0*mgo(1)*x) + mgo(1)*mgo(2);
```

```
plot(x,dpre,'r-','LineWidth',3);
```




```
% Figure 9.13 Gradient method is used to solve the same nonlinear curve-fitting problem as in
% Figure 9.5. (A) The observed data (black circles) are computed from the true data (black cur
% by adding random noise. The predicted data (red curve) are based on the results of the metho
```