```
% gda07_08
% least squares with inequality constraints
% exemplary stright line problem
% supports Figure 7.9
clear all;

% three inequality constraints
% m1-m2>=0          [1   -1] [0]
% 0.5*m1+m2>=1      [0.5  1] [1]
% m1>=0.2           [1    0] [0.2]
% in the last constraint, the 0.2 can be diddled
% right now this constraint does not affect the
% feasible region, but changing 0.2 to 1.2 cnages
% the region considerably
H = [ [1, 0.5, 1]', [-1, 1, 0]' ];
h = [0, 1, 0.2]';

% straight line problem
% in feasible region: mtrue = [1, 0.8]';
% out of feasbile region mtrue = [1, 1.3]';
mtrue = [1.0, 0.2]';
N=11;
z = [0:N-1]'/(N-1);
G = [ones(N,1), z];
dtrue = G*mtrue;
sd=0.05;
dobs = dtrue + random('Normal',0,sd,N,1);

% solve by ordinary least squares, just for check
mestls = (G'*G)\(G'*dobs);
dprels = G*mestls;
fprintf('ordinary least squares:\n');
```

```
ordinary least squares:
```

```
fprintf('   mtrue %f %f\n', mtrue(1), mtrue(2) );
```

```
   mtrue 1.000000 0.200000
```

```
fprintf('   mest  %f %f\n', mestls(1), mestls(2) );
```

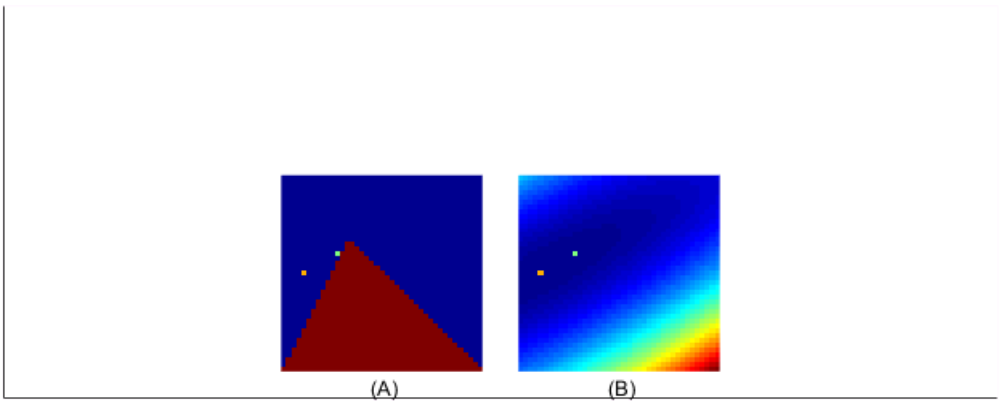
```
   mest  0.960430 0.253704
```

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

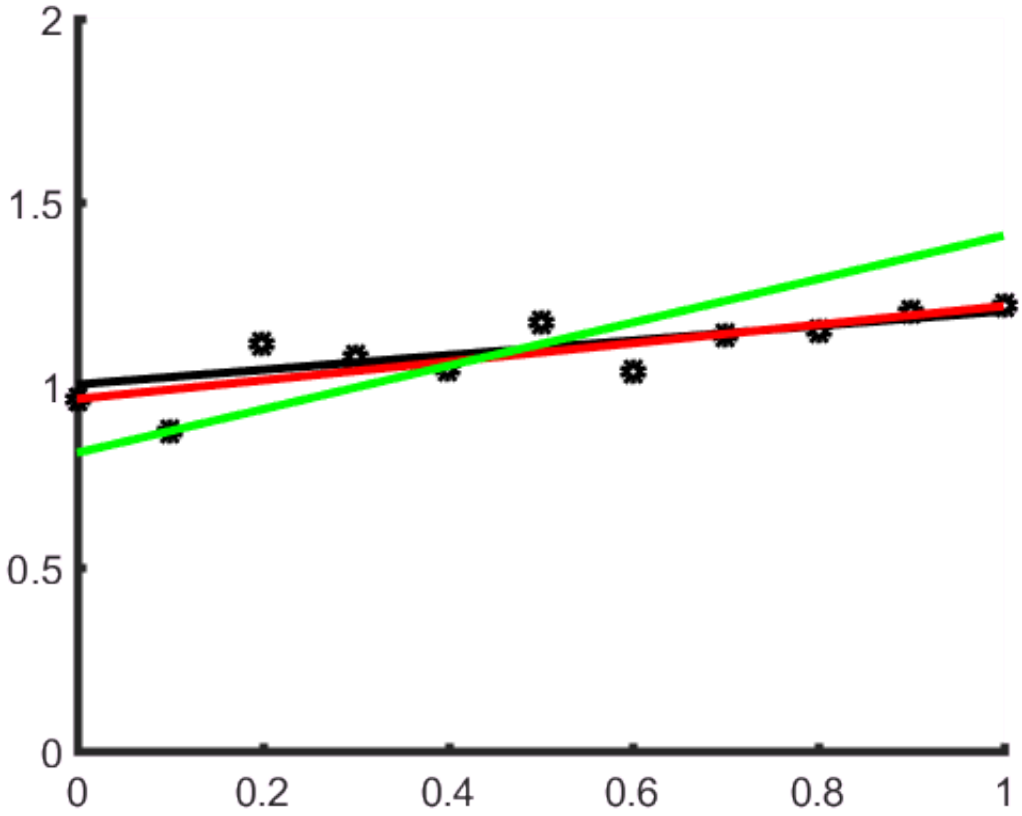
```
% (m1, m2) grid for plotting purposes
m1min=0;
m1max=2;
Nm1 = 41;
Dm1 = (m1max-m1min)/(Nm1-1);
m1 = m1min + Dm1*[0:Nm1-1]';
m2min=0;
m2max=2;
Nm2 = 41;
Dm2 =(m2max-m2min)/(Nm2-1);
```

```
m2 = m2min + Dm2*[0:Nm2-1]';
X = m1*ones(Nm2,1)';
Y = ones(Nm1,1)*m2';

% Error surface, for plotting purposes
E = zeros(Nm1,Nm2);
for i=[1:Nm1]
for j=[1:Nm2]
    m = [ m1(i), m2(j) ]';
    e = dobs - G*m;
    E(i,j)=e'*e;
end
end
Emin = min(min(E));
Emax = max(max(E));

% evaluate the constraints, for plotting purposes
% constraint 1
p=1;
C1 = zeros(Nm1,Nm2);
for i=[1:Nm1]
for j=[1:Nm2]
    m = [ m1(i), m2(j) ]';
    C1(i,j)=H(p,:)*m-h(p);
end
end
C1a = (C1>=0)+0.001;
% constraint 2
p=2;
C2 = zeros(Nm1,Nm2);
for i=[1:Nm1]
for j=[1:Nm2]
    m = [ m1(i), m2(j) ]';
    C2(i,j)=H(p,:)*m-h(p);
end
end
C2a = (C2>=0)+0.001;
% constraint 3
p=3;
C3 = zeros(Nm1,Nm2);
for i=[1:Nm1]
for j=[1:Nm2]
    m = [ m1(i), m2(j) ]';
    C3(i,j)=H(p,:)*m-h(p);
end
end
C3a = (C3>=0)+0.001;
% union of constraints
CT = ((C1>=0)&(C2>=0)&(C3>=0))+0.001;

% plot true solution by making dot in CT
im1est = floor((mtrue(1)-m1min)/Dm1)+1;
jm2est = floor((mtrue(2)-m2min)/Dm2)+1;
CT(im1est,jm2est)=0.7;
E(im1est,jm2est)=0.7*(Emax-Emin);

% SVD
% note that G must be overdetermined so V=Vp
% economy size (flag 0) SVD so U is Up
[Up, Lp, Vp] = svd(G,0);
lambda = diag(Lp);
```

```
rlambda = 1./lambda;
Lpi = diag(rlambda);

% transformation 1
Hp = -H*Vp*Lpi;
hp = h + Hp*Up'*dobs;

% transformation 2
Gp = [Hp, hp]';
dp = [zeros(1,length(Hp(1,:))), 1]';
mpp = lsqnonneg(Gp,dp);
ep = dp - Gp*mpp;
mp = -ep(1:end-1)/ep(end);

% take mp back to m
mest = Vp*Lpi*(Up'*dobs-mp);
dpre = G*mest;

% plot solution by making dot in CT
im1est = floor((mest(1)-m1min)/Dm1)+1;
jm2est = floor((mest(2)-m2min)/Dm2)+1;
CT(im1est,jm2est)=0.5;
E(im1est,jm2est)=0.5*(Emax-Emin);

gda_draw(' ',CT,'caption (A)',' ',E,'caption (B)');
```




```
% Figure 7.9 Problem of minimizing the prediction error E subject to inequality constraints, a
% to the straight line problem. (A) Feasible region of the model parameters (the intercept m1
% slope m2 of the straight line). The unconstrained solution (orange circle) is outside the fe
% region, but the constrained solution (green circle) is on its boundary. (B) The unconstraine
% solution (orange circle) is at the global minimum of the prediction error E, while the const
% solution is not. MatLab script gda07_08.

figure(2);
clf;
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [0, 1, 0, 2 ] );
plot( z, dobs, 'ko', 'LineWidth', 3 );
plot( z, dtrue, 'k-', 'LineWidth', 3);
plot( z, dprels, 'r-', 'LineWidth', 3);
plot( z, dpre, 'g-', 'LineWidth', 3);
```




```
% Figure 7.9 Problem of minimizing the prediction error E subject to inequality constraints, a
% to the straight line problem. (C) Plot of the data d(z) showing true data (black line), obse
% circles), unconstrained prediction (red line), and constrained prediction (green line). MatL

fprintf('least squares with inequality constraint:\n');
```

```
least squares with inequality constraint:
```

```
fprintf('   mtrue %f %f\n', mtrue(1), mtrue(2) );
```

```
   mtrue 1.000000 0.200000
```

```
fprintf('   mest  %f %f\n', m(1), mestls(2) );
```

```
   mest  2.000000 0.253704
```

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