```
% gda12_03
% Supports Figure 12.5
% correcting gravity cross-over errors

clear all;

% load gravity(lat,lon) data
D=load('../data/gravity.txt');
lon=D(:,1);
lat=D(:,2);
grav=D(:,3);

% setup axes
mingrav = min(grav);
maxgrav = max(grav);
meangrav = mean(grav);
Ng=length(lon);
lonmin=min(lon);
lonmax=max(lon);
latmin=min(lat);
latmax=max(lat);
Nlon=length(find(lat==lat(1)));
Nlat=length(find(lon==lon(1)));
Dlon = (lonmax-lonmin)/(Nlon-1);
Dlat = (latmax-latmin)/(Nlat-1);

% I tested the coordinate system by
% resetting a specific datum to a value
% that I could easily pick out, and then
% comparing its position on the plot
% with the position displayed below
if( 0 )
    i=1500;
    fprintf('%f %f\n', lon(i), lat(i) );
    grav(i)=mingrav;
end

% put data into grid
% Note the messing about with trasnposes and
% flipup() to make it come out rightside up
LON=reshape(lon,Nlon,Nlat);
LAT=reshape(lat,Nlon,Nlat);
GRAV=reshape(grav,Nlon,Nlat);
LAT=flipud(LAT');
LON=flipud(LON');
GRAV=flipud(GRAV');

% examne order
debg = 0;
if( debg )
    fprintf('LON(1,1)=%f LON(1,N)=%f\n', LON(1,1), LON(1,end) );
    fprintf('LON(N,1)=%f LON(N,N)=%f\n', LON(end,1), LON(end,end) );
    fprintf('\n');
    fprintf('LAT(1,1)=%f LAT(1,N)=%f\n', LAT(1,1), LAT(1,end) );
    fprintf('LAT(N,1)=%f LAT(N,N)=%f\n', LAT(end,1), LAT(end,end) );
    fprintf('\n');
end

% at this point, ordering is
```

```
% LON(1,1)=0.041667 LON(1,N)=9.958330
% LON(N,1)=0.041667 LON(N,N)=9.958330
%
% LAT(1,1)=0.041667 LAT(1,N)=0.041667
% LAT(N,1)=9.958330 LAT(N,N)=9.958330

% set a small (i,1) block of Grav(1,j) to a low value
% and visually check where it plots on the map to
% confirm that the ordering is right
debg = 0;
if( debg )
    GRAV(1:10,1:20)=mingrav;
end

% plot gravity data
figure(1);
clf;
colormap('jet');

% plot true gravity
subplot(2,2,1);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [lonmin, lonmax, latmin, latmax] );
axis xy;
imagesc([lonmin, lonmax], [latmin, latmax], GRAV);
title('true gravity');

% plot gravity and prepare to superimpose tracks
subplot(2,2,2);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [lonmin, lonmax, latmin, latmax] );
axis xy;
imagesc([lonmin, lonmax], [latmin, latmax], GRAV);
title('tracks');

% the tracks are synthetic; that is, constructed in this program
Ntracks = 35;
Ntlon = Nlon;
Dtlon = (lonmax-lonmin)/(Ntlon-1);
tlon = lonmin + Dtlon*[0:Ntlon-1]';

% construct ascending tracks
NA = Ntracks;
Atracklat = zeros(Ntlon, NA);
C = 1;
L = 2*(lonmax-lonmin);
for i = [1:NA]
   Atracklat(:,i) = tlon + C*cos(2*pi*tlon/L) + i/2 - 9;
   plot( tlon, Atracklat(:,i), 'k-', 'LineWidth', 2 );
end

% construct descending tracks
ND = Ntracks;
Dtracklat = zeros(Ntlon, NA);
C = 1;
L = 2*(lonmax-lonmin);
for i = [1:NA]
```

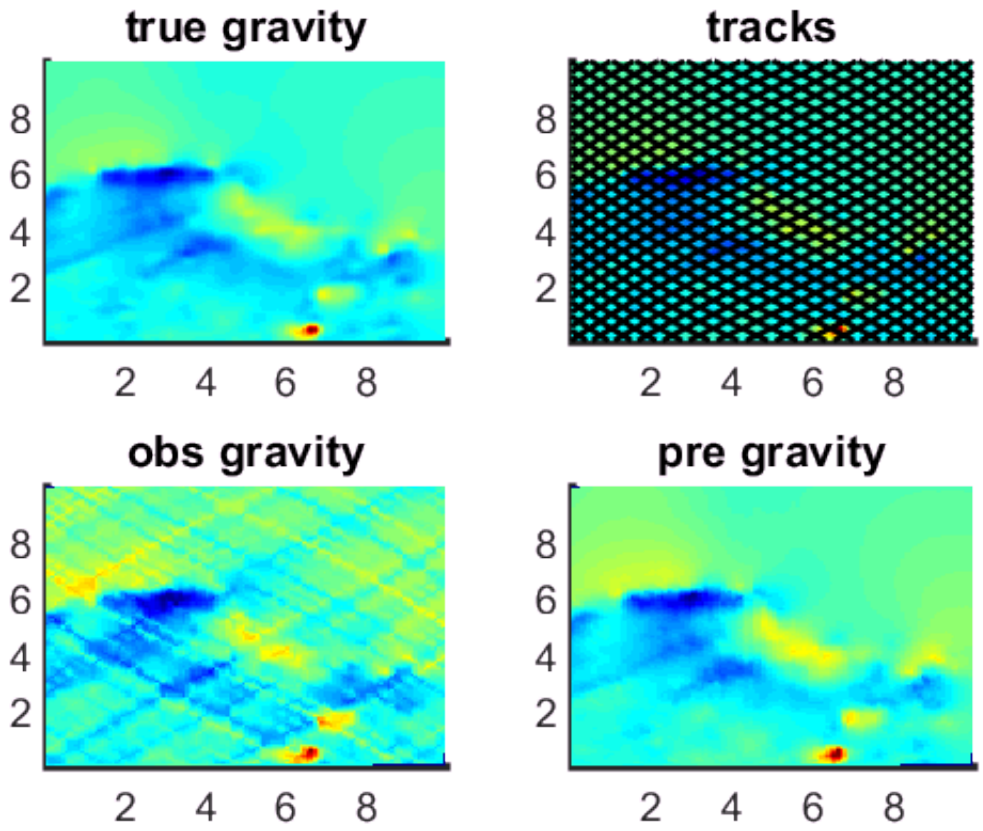
```
   Dtracklat(:,i) = latmax - (tlon + C*cos(2*pi*tlon/L) + i/2 -9);
   plot( tlon, Dtracklat(:,i), 'k-', 'LineWidth', 2 );
end

M = NA+ND; % number of model parameters are number of tracks

% model parameters are gravity shifts (offets) along each track.
% they are randomply chosen.
% store the shifts in vector with ascending tracks first
sm=0.1;
mtrue = random('Normal',0,sm*(maxgrav-mingrav)/2,M,1);

% create a dataset of all the data beneath the tracks,
% but offset by the shifts
Nsamples=0;
slonv = zeros(M*Ntlon,1);
slatv = zeros(M*Ntlon,1);
sgrav = zeros(M*Ntlon,1);
slon = tlon;
for i = [1:NA]  % ascending tracks
   slat = Atracklat(:,i);
   ii = floor((slat-latmin)/Dlat)+1;
   jj = floor((slon-lonmin)/Dlon)+1;
   for kk = [1:Ntlon]
       if( (ii(kk)>=1) && (ii(kk)<=Nlat) && (jj(kk)>=1) && (jj(kk)<=Nlon) )
           Nsamples=Nsamples+1;
           slonv(Nsamples) = slon(kk);
           slatv(Nsamples) = slat(kk);
           sgrav(Nsamples) = GRAV(ii(kk),jj(kk))+mtrue(i);
       end
   end
end
for j = [1:ND]  % decending tracks
   slat = Dtracklat(:,j);
   ii = floor((slat-latmin)/Dlat)+1;
   jj = floor((slon-lonmin)/Dlon)+1;
   for kk = [1:Ntlon]
       if( (ii(kk)>=1) && (ii(kk)<=Nlat) && (jj(kk)>=1) && (jj(kk)<=Nlon) )
           Nsamples=Nsamples+1;
           slonv(Nsamples) = slon(kk);
           slatv(Nsamples) = slat(kk);
           sgrav(Nsamples) = GRAV(ii(kk),jj(kk))+mtrue(NA+j);
       end
   end
end

% interpolate data along tracks to a 2D image
% Note: the code was originally
% ginterp = TriScatteredInterp( slonv, slatv, sgrav );
% however called complained that there were duplicate
% points.  This patch explicitly removes duplicates
sorted = sortrows( [slonv, slatv, sgrav], [1,2] );
Nsort=length(sorted);
flag = ones(Nsort,1);
for i=[2:Nsort]
    if( (sorted(i,1)==sorted(i-1,1)) && (sorted(i,2)==sorted(i-1,2)) )
        flag(i)=0;
    end
end
k = find( flag==1 );
ginterp = TriScatteredInterp( sorted(k,1), sorted(k,2), sorted(k,3) );
```

```
obsgrav = ginterp(LON(:),LAT(:));
OBSGRAV = reshape(obsgrav,Nlon,Nlat);

% plot gravity data (without cross-0ver corrections)
subplot(2,2,3);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [lonmin, lonmax, latmin, latmax] );
axis xy;
imagesc([lonmin, lonmax], [latmin, latmax], OBSGRAV);
title('obs gravity');

% find all track intersections (cross-overs)
subplot(2,2,2);
% Xlon(i,j) is lon where ascending track i and descending track j cross
Xlon = NaN(NA,ND);
% Xlat(i,j) is lat where ascending track i and descending track j cross
Xlat = NaN(NA,ND);
% Xgrav(i,j) is grav where ascending track i and descending track j cross
Xgrav = NaN(NA,ND);
% iiX, jjX are nearest indices in GRAV(iiX,iiJ) array
iiX = zeros(NA,ND);
jjX = zeros(NA,ND);
count=0;
for i = [1:NA]
for j = [1:ND]
    signAD = sign(Atracklat(:,i)-Dtracklat(:,j));
    sign1 = signAD(1);
    k2 = find( signAD ~= sign1, 1 );
    if( length(k2) == 0 ) % no intersection
        continue;
    end
    if( k2 == 1 ) % intersects at left edge; ignore
        continue;
    end
    % linearly interpolate
    k1 = k2-1;
    A = Atracklat(k1,i)-Dtracklat(k1,j);
    B = Atracklat(k2,i)-Dtracklat(k2,j);
    slope = (B-A)/Dtlon;
    deltalon = -A/slope;
    Xlon(i,j) = tlon(k1)+deltalon;
    Xlat(i,j) = Xlon(i,j) + C*cos(2*pi*Xlon(i,j)/L) + (i/2-9);
    jjXp=floor((Xlon(i,j)-lonmin)/Dlon)+1;
    iiXp=floor((Xlat(i,j)-latmin)/Dlat)+1;
    if( (iiXp>=1) && (iiXp<=Nlat) && (jjXp>=1) && (jjXp<=Nlon) )
        count=count+1;
        iiX(i,j) = iiXp;
        jjX(i,j) = jjXp;
        % check if these are right by making a dot on the
        % image and comparing a displayed version of the lat, lon
        % with the position of the dot
        debg = 0;
        if( debg && (count==20) )
            GRAV( iiXp, jjXp ) = mingrav;
            fprintf('lon %f lat %f\n', Xlon(i,j), Xlat(i,j) );
        end
        Xgrav(i,j) = GRAV( iiXp, jjXp );
        % plot intersection point on map
        debg = 0;
```

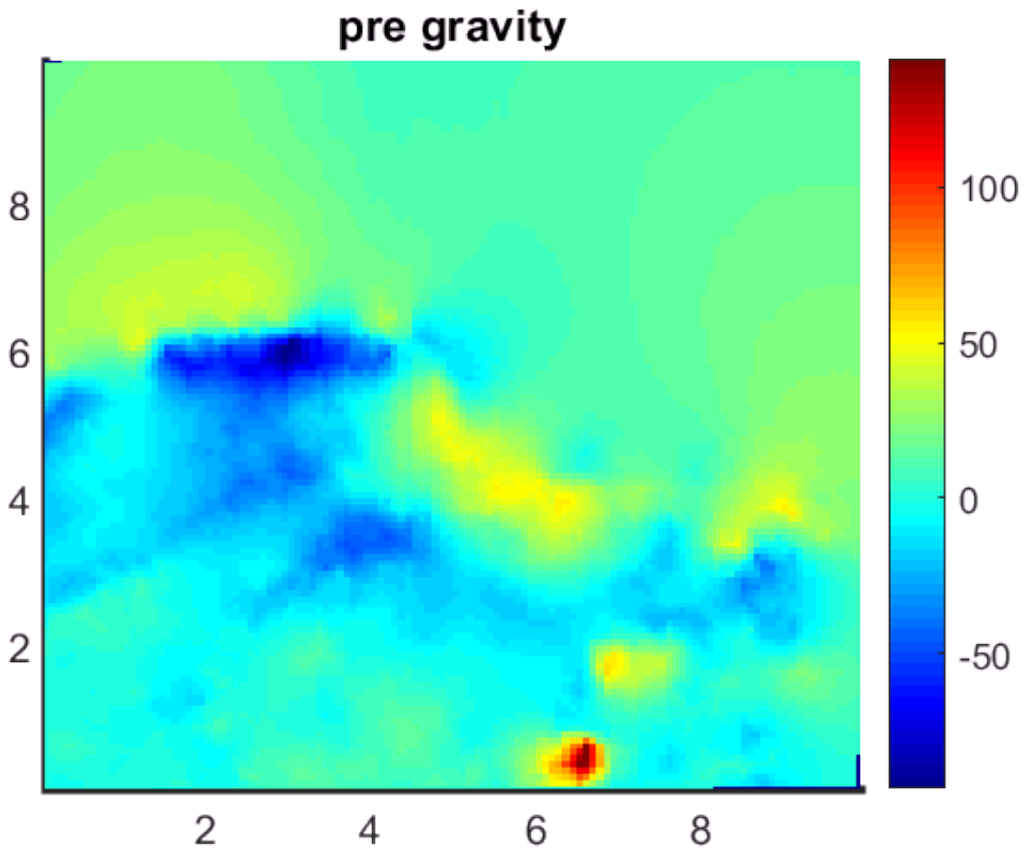
```
        if( debg )
            plot( Xlon(i,j), Xlat(i,j), 'ko', 'LineWidth', 2);
        end
    end
end
end

debg = 0;
if( debg ) % part of a check that indiced iXlon and iXlat are correct
           % the position of the little spot in the image should match
           % the (lon, lat) position displayed
    figure(3);
    clf;
    set(gca,'LineWidth',2);
    hold on;
    axis( [lonmin, lonmax, latmin, latmax] );
    axis xy;
    imagesc([lonmin, lonmax], [latmin, latmax], GRAV);
    title('for debugging');
end

N = count; % number of data is number of intersections

% gravity values for each track at its intersection point
Agrav = zeros(N,1);
Dgrav = zeros(N,1);
% back indices to i,j
iA = zeros(N,1);
jB = zeros(N,1);
k=0;
for i = [1:NA]
for j = [1:ND]
    if( iiX(i,j) == 0 )
        continue;
    end
    k=k+1;
    Agrav(k) = Xgrav(i,j)+mtrue(i);
    Dgrav(k) = Xgrav(i,j)+mtrue(j+NA);
    iA(k)=i;
    jD(k)=j;
end
end

% data equations are that, at an intersection point,
% grav measured by ascending track - offset of that track =
% grav measured by decending track - offset of that track
G = spalloc( N, M, 2*N );
d = zeros(N,1);
k=0;
for k = [1:N]  % loop row-wise
    G(k,iA(k)) = 1;
    G(k,NA+jD(k)) = -1;
    d(k) = Agrav(k)-Dgrav(k);
    debg = 0;
    if( debg )
     fprintf('%d   %d %d   %f\n', k, iA(j), jD(k), d(k) );
    end
end

% solve with damped least squares
% damping is necessary, since the solution
```

```
% is only determined up to an additive constant
epsilon = 1.0e-5*max(max(G));
mest = (G'*G + epsilon*speye(M))\(G'*d);

% recreate the dataset of all the data beneath the tracks,
% but correct the shifts
Nsamples=0;
slonv = zeros(M*Ntlon,1);
slatv = zeros(M*Ntlon,1);
sgrav = zeros(M*Ntlon,1);
slon = tlon;
for i = [1:NA]  % ascending tracks
   slat = Atracklat(:,i);
   ii = floor((slat-latmin)/Dlat)+1;
   jj = floor((slon-lonmin)/Dlon)+1;
   for kk = [1:Ntlon]
       if( (ii(kk)>=1) && (ii(kk)<=Nlat) && (jj(kk)>=1) && (jj(kk)<=Nlon) )
           Nsamples=Nsamples+1;
           slonv(Nsamples) = slon(kk);
           slatv(Nsamples) = slat(kk);
           sgrav(Nsamples) = GRAV(ii(kk),jj(kk))+mtrue(i)-mest(i);
       end
   end
end
for j = [1:ND]  % decending tracks
   slat = Dtracklat(:,j);
   ii = floor((slat-latmin)/Dlat)+1;
   jj = floor((slon-lonmin)/Dlon)+1;
   for kk = [1:Ntlon]
       if( (ii(kk)>=1) && (ii(kk)<=Nlat) && (jj(kk)>=1) && (jj(kk)<=Nlon) )
           Nsamples=Nsamples+1;
           slonv(Nsamples) = slon(kk);
           slatv(Nsamples) = slat(kk);
           sgrav(Nsamples) = GRAV(ii(kk),jj(kk))+mtrue(NA+j)-mest(NA+j);
       end
   end
end

% interpolate data along tracks to make 2D image
% Note: same issue as above with duplicate points
% ginterp = TriScatteredInterp( slonv, slatv, sgrav );
sorted = sortrows( [slonv, slatv, sgrav], [1,2] );
Nsort=length(sorted);
flag = ones(Nsort,1);
for i=[2:Nsort]
    if( (sorted(i,1)==sorted(i-1,1)) && (sorted(i,2)==sorted(i-1,2)) )
        flag(i)=0;
    end
end
k = find( flag==1 );
ginterp = TriScatteredInterp( sorted(k,1), sorted(k,2), sorted(k,3) );
pregrav = ginterp(LON(:),LAT(:));
PREGRAV = reshape(pregrav,Nlon,Nlat);

% plot predicted gravity data (with cross-over corrections)
figure(1);
subplot(2,2,4);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [lonmin, lonmax, latmin, latmax] );
```

```
axis xy;
imagesc([lonmin, lonmax], [latmin, latmax], PREGRAV);
title('pre gravity');
```




```
% Figure 12.5 Example of crossover error adjustment of satellite gravity data.
% (A) True gravity anomaly data for the equatorial Atlantic Ocean. It reflects
% variations in the depth of the seafloor and density variations within the oceanic
% crust. (B) Hypothetical satellite tracks, along which the gravity is measured.
% The measurements along each track have a constant offset reflecting errors in the
% assumed altitude of the satellite. (C) Reconstructed gravity anomaly without crossover
% correction. Artifacts parallel to the tracks are clearly visible. (D) Reconstructed
% gravity anomaly with crossover correction. The artifacts are eliminated. Data courtesy
% of Bill Haxby, Lamont-Doherty Earth Observatory. MatLab script gda12_03.


% plot gravity data
figure(2);
clf;
colormap('jet');
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [lonmin, lonmax, latmin, latmax] );
axis xy;
imagesc([lonmin, lonmax], [latmin, latmax], PREGRAV);
title('pre gravity');
colorbar();
```




```
% Figure. Predicted gravity after cross-over correction
```