Code Tutorial

April 1, 2022

This is a tutorial for running the codes to produce forward gravity anomalies. In the manuscript, we have discussed four different algorithms to estimate the forward gravity anomalies due to any bounded irregular topography surfaces having any functional form of density distributions. These four algorithms are as follows

1. True layered model
2. Gauss-FFT based quadrature model
3. Standard FFT based quadrature model
4. Gausss-FFT based analytic model

For each of those four algorithms, few common inputs are required to estimate the forward gravity anomalies,

(i) 2D gridded topography data of shallower layer in m.

(ii) 2D gridded topography data of deeper layer in m.

(iii) 3D functional form of density distribution for bounded topography data in kg/m^3.

(iv) One dimensional grids having fixed-step interval along the horizontal x-direction corresponding to the topography layer.

(v) One dimensional grids having fixed-step interval along the horizontal y-direction corresponding to the topography layer.

(vi) The height of the the observation plane at which forward model has to evaluate.

In figure 1, all commands for input data for a generalized exponential density model are given. For plotting of input topography surfaces is shown in figure 2.

Finally, three different subroutines are developed for different algorithms,

(I) grav_layer_gaussfft: It will produce gravity anomalies due to any irregular topography bound mass source by vertically slicing the entire mass source in horizontal layers.

(II) grav_quadrature_fft: It will produce gravity anomalies by expanding the edges using the standard FFT algorithm.

(III) grav_quadrature_gaussfft: The Gauss FFT algorithm will produce gravity anomalies by numerically evaluating the density integral.

After providing all input information, users can use the above subroutines to produce the forward anomalies. But grav_quadrature_gaussfft is recommended as it provides a more accurate and efficient forward model. grav_layer_gaussfft is the computationally expensive algorithm and requires one extra input, i.e., the number of layers. In figures 3, 4, and 5, all algorithms for the exponential density model are implemented. After the evaluation, all model outputs are stored in the specified output folder for plotting. The example code for plotting the output data is also given. Matlab file plllt1.m is used for plotting the topography and corresponding gravity anomalies for different density distributions. By changing the input data, as shown in figure 1, any user can evaluate the forward gravity anomalies as per the choice of forward algorithm. A generalized code is added named 'generalized.m' to produce gravity anomalies by any user by providing input data as per the choice.

To generate output data used in the manuscript, separate codes are developed for all different density models using different forward anomaly evaluation algorithms. A detailed readme file is provided for a better understanding. All output data for the different models were stored in the output folder, and finally, 'plllt1.m' and 'plllt2.m' are used for visualizing all input and output data.

```
%%Inputs%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%exponential density model
%importing topography data

%shallower surface
data1=importdata(fullfile('.',
'input','synthetic_topo_exp_density_shallower_layer.txt'));
%deeper surface
data2=importdata(fullfile('.',
'input','synthetic_topo_exp_density_deeper_layer.txt'));

%Depth grids in meter
xx=importdata(fullfile('.',
'input','synthetic_x_exp_density.txt')); %along x axis
yy=importdata(fullfile('.',
'input','synthetic_y_exp_density.txt')); %along y axis

%observation point at z=0;
z0=0;

%density contrast
rho=@(x,y,z) -500.*(2.32.*10^-5.*x+1.5.*10^-5.*y).*exp(-
0.0187.*z.*10^-2);     %exponential

%number of gauss quadrature node
Mx=2; My=2;

%number of layers for grav_layer_gaussfft
layers=150;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 1: Code for providing input data to evaluate forward gravity anomalies.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%plotting the input data
%colormaps
ccmap1=makecolormap({'green','yellow','khaki','olive','brow
n','silver'}, 128);
[XX,YY]=meshgrid(xx,yy);
XX=XX./1000; YY=YY./1000;
figure(1)

surf(XX,YY,data1./10^3)
hold on;
[C,h] = contour3(XX,YY,data1./10^3,20,'k');
surf(XX,YY,data2./10^3)
[C,h] = contour3(XX,YY,data2./10^3,20,'k');
shading interp
set(gca, 'Zdir', 'reverse')

colormap(ccmap1)
view(3);
xlabel('x (km)')
ylabel('y (km)')
zlabel('Depth (km)')
%title('depth data plot')
set(gca, 'ZDir','reverse')
grid on;
box on;
c = colorbar;
c.Label.String = 'Depth (km)';
xlim([min(XX(:)) max(XX(:))])
ylim([min(YY(:)) max(YY(:))])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 2: Code for plotting topography surfaces as per input data.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Ouputs%%%
%%Gauss-fft quadrature based model%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%gravity anomaly for given data using gauss-fft quadrature
based model
[XX1, YY1, gz, delta1, delta2,
N]=grav_quadrature_gaussfft(data1,data2,xx,yy,rho,z0,Mx,My)
; %gravity anomaly

%saving the output data
save(fullfile('.',
'output','gravity_exp_density_quadrature1.txt'), 'gz', '-
Ascii') %gravity anomaly
save(fullfile('.', 'output','x_meshgrid_exp_density.txt'),
'XX1', '-Ascii') %x grid
save(fullfile('.', 'output','y_meshgrid_exp_density.txt'),
'YY1', '-Ascii') %y grid

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%plotting the output data
%colormaps
ccmap2=makecolormap({'navy','firebrick','red','orange','vio
let','darkorchid','mediumspringgreen'}, 128);

XX1=XX1*10^-3; YY1=YY1*10^-3;
figure(2)
surf(XX1,YY1,gz)
shading interp
xlabel('x (km)')
ylabel('y (km)')
zlabel('Gravity anomaly (mGal)')
view(2)
grid on;
box on;
colormap(ccmap2)
c = colorbar;
c.Label.String = 'Gravity anomaly (mGal)';
xlim([min(XX1(:)) max(XX1(:))])
ylim([min(YY1(:)) max(YY1(:))])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 3: Code for evaluating gravity anomalies using using Gauss-FFT quadrature based model.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Ouputs%%%
%%Standard fft quadrature based model%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%gravity anomaly for given data using standard fft
quadrature based model
[XX1, YY1, gz, delta1, delta2,
N]=grav_quadrature_fft(data1,data2,xx,yy,rho,z0); %gravity
anomaly

%saving the output data
save(fullfile('.',
'output','gravity_exp_density_quadrature2.txt'),'gz','-
Ascii')
save(fullfile('.', 'output','x_meshgrid_exp_density.txt'),
'XX1', '-Ascii') %x grid
save(fullfile('.', 'output','y_meshgrid_exp_density.txt'),
'YY1', '-Ascii') %y grid

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%plotting the output data
%colormaps
ccmap1=makecolormap({'green','yellow','khaki','olive','brow
n','silver'}, 128);
ccmap2=makecolormap({'navy','firebrick','red','orange','vio
let','darkorchid','mediumspringgreen'}, 128);

XX1=XX1*10^-3; YY1=YY1*10^-3;
figure(3)
surf(XX1,YY1,gz)
shading interp
xlabel('x (km)')
ylabel('y (km)')
zlabel('Gravity anomaly (mGal)')
view(2)
grid on;
box on;
colormap(ccmap2)
c = colorbar;
c.Label.String = 'Gravity anomaly (mGal)';
xlim([min(XX1(:)) max(XX1(:))])
ylim([min(YY1(:)) max(YY1(:))])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 4: Code for evaluating gravity anomalies using using Standard FFT quadrature based model.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Ouputs%%%
%%layer based model%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
gravity anomaly for given data using layer based model
[XX1, YY1,
gz]=grav_layer_gaussfft(data1,data2,xx,yy,rho,z0,Mx,My,laye
rs); %gravity anomaly

%saving the output data
save(fullfile('.',
'output','gravity_exp_density_layer.txt'),'gz','-
Ascii')%gravity anomaly
save(fullfile('.', 'output','x_meshgrid_exp_density.txt'),
'XX1', '-Ascii')%x grid
save(fullfile('.', 'output','y_meshgrid_exp_density.txt'),
'YY1', '-Ascii')%y grid

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%colormaps
ccmap1=makecolormap({'green','yellow','khaki','olive','brow
n','silver'}, 128);
ccmap2=makecolormap({'navy','firebrick','red','orange','vio
let','darkorchid','mediumspringgreen'}, 128);

XX1=XX1*10^-3; YY1=YY1*10^-3;
figure(4)
surf(XX1,YY1,gz)
shading interp
xlabel('x (km)')
ylabel('y (km)')
zlabel('Gravity anomaly (mGal)')
view(2)
grid on;
box on;
colormap(ccmap2)
c = colorbar;
c.Label.String = 'Gravity anomaly (mGal)';
xlim([min(XX1(:)) max(XX1(:))])
ylim([min(YY1(:)) max(YY1(:))])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 5: Code for evaluating gravity anomalies using layering based model.