Table of Contents

Create 3D realistic analytical model

Figure_300

```
% Create realistic geometry that is used in models #12, #13, #14
% Model #12 - v = 1500;
% Model #13 - v = 1500 + 0.5*z
% Model #14 - if z < 250 => v = 1500,
%             if z > 250 => v = 1500 + 0.5*(z-250)
```

Introduction

Author: Abakumov Ivan

Publication date: 27th August 2016

Define working folder, add links to Library and SeisLab

```
clear all; close all; clc;
mlibfolder = '/home/zmaw/u250128/Desktop/MLIB';
path(path, mlibfolder);
addmypath;
```

Read 3D SEG salt model

see file /home/zmaw/u250128/Desktop/CRS/00_MATLAB/analize_SEG_3D_geometry.m

```
segyvelfile = '/scratch/local1/ivan/CRS_data/SEG_C3_WA_data/
SEG_C3NA_Velocity.sgy';
veldata = read_segy_file(segyvelfile);
```

```
inline = veldata.headers(6, :)/200+1;     % (20 m x 10 scaling factor)
xline  = veldata.headers(7, :)/200+1;

G=GridClass;

%   [m]               [m]          [m]              [s]
G.x0 = 0;        G.y0 = 0;    G.z0 = 0;       G.t0 = 0.00;             %
 initial point
G.nx = 676;      G.ny = 676;  G.nz = 201;     G.nt = 625;              %
 grid size
G.dx = 20;       G.dy = 20;   G.dz = 20;      G.dt = 0.008;            %
 grid step (meter)

G.gridInfo;
G.setGrid;

Information about grid:
x0=0, dx=20, Nx=676.
y0=0, dy=20, Ny=676.
z0=0, dz=20, Nz=201.
t0=0, dt=0.008, Nt=625.
```

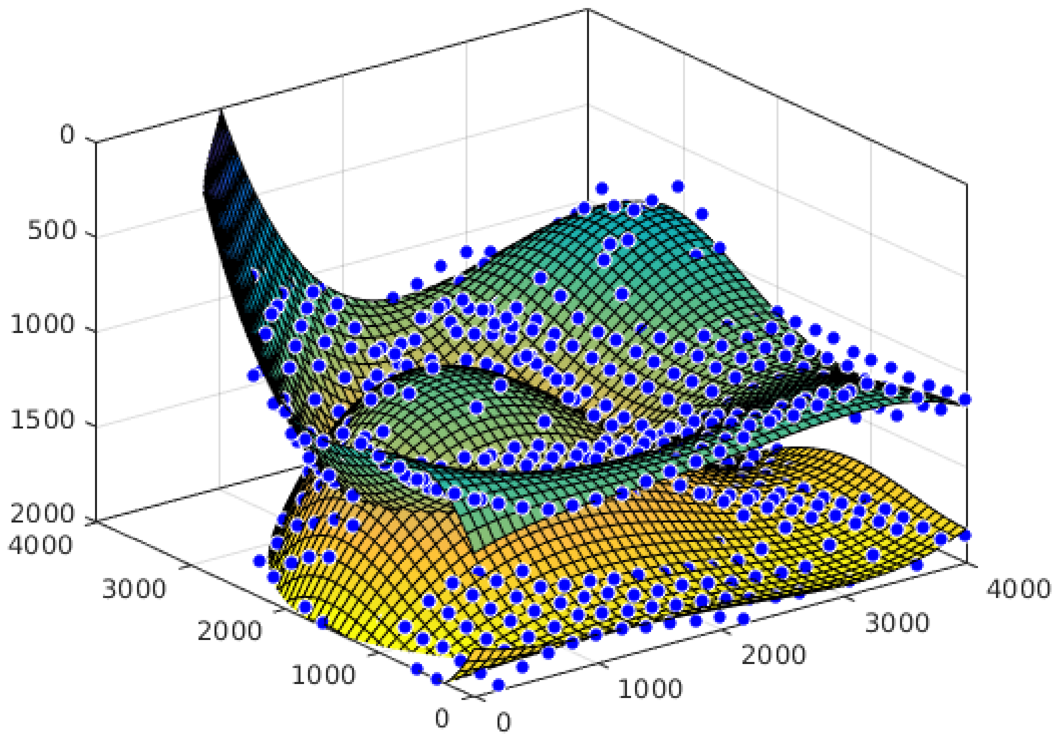
Make 3D SEG salt velocity cube

```
velmod = zeros(G.nx,G.ny,G.nz);
for i=1:length(inline)
    velmod(inline(i), xline(i), :) = veldata.traces(:, i);
end
```

Make salt body

```
salt = velmod>4000;
salt_border = diff(salt,1,3);
salt_top    = zeros(676,676);
salt_bot    = zeros(676,676);

for i=1:676
    for j=1:676
        [border, k] = max(salt_border(i,j,:));
        if border==1
            salt_top(i,j) = G.z0 + (k-1)*G.dz;
        end
        clear border k
        [border, k] = min(salt_border(i,j,:));
        if border==-1
            salt_bot(i,j) = G.z0 + (k-1)*G.dz;
        end
        clear border k
    end
end
```

```
indx = 280:10:480;
indy = 180:10:380;
ind  = (salt_top(indx, indy)~=0);

X = G.xx(indx)- min(G.xx(indx));
Y = G.yy(indy) - min(G.yy(indy));
[XX, YY] = meshgrid(X, Y);

figure(1)
salt_top = salt_top(indx,indy)+500;
salt_bot = 3000 - salt_top;
%top.ind = reshape(salt_top>0, 441, 1);
top.x   = reshape(XX, 441, 1);
top.y   = reshape(YY, 441, 1);
top.ind = reshape(ind, 441, 1);
top.z   = reshape(salt_top, 441, 1);
top.x   = top.x(ind);
top.y   = top.y(ind);
top.z   = top.z(ind);
top.f = fit( [top.x, top.y], top.z, 'poly44' );
plot(top.f, [top.x,top.y], top.z)

bot.ind = reshape(ind, 441, 1);
bot.x   = reshape(XX, 441, 1);
bot.y   = reshape(YY, 441, 1);
bot.z   = reshape(salt_bot, 441, 1);
bot.x   = bot.x(bot.ind);
bot.y   = bot.y(bot.ind);
bot.z   = bot.z(bot.ind);
hold on
bot.f = fit( [bot.x, bot.y], bot.z, 'poly44' );
plot(bot.f, [bot.x,bot.y], bot.z)
set(gca, 'ZDir', 'reverse')
axis([0 4000 0 4000 0 2000])
caxis([0 2000])

Warning: Equation is badly conditioned. Remove repeated data points or
 try
centering and scaling.
Warning: Equation is badly conditioned. Remove repeated data points or
 try
centering and scaling.
```



Make subset of 3D SEG salt model

```
sG=GridClass;

%   [m]                [m]           [m]               [s]
sG.x0 = 0;        sG.y0 = 0;    sG.z0 = 0;        sG.t0 = 0.00;
   % initial point
sG.nx = 201;      sG.ny = 201;  sG.nz = 201;      sG.nt = 1000;
    % grid size
sG.dx = 20;       sG.dy = 20;   sG.dz = 10;       sG.dt = 0.001;
   % grid step (meter)

sG.gridInfo;
sG.setGrid;
sGold = oldGrid(sG);

svelmod = zeros(sG.nx, sG.ny, sG.nz);

for k=1:25
     svelmod(:,:,k) = 1500;
end

for k=26:201
     svelmod(:,:,k) = svelmod(:,:,k-1) + sG.dz*0.5;
end
```

```
[sXX, sYY] = meshgrid(sG.xx, sG.yy);

clear salt
salt.gtop = x2grid(top.f(sXX, sYY), sG.z0, sG.dz, sG.nz);
salt.gbot = x2grid(bot.f(sXX, sYY), sG.z0, sG.dz, sG.nz);

svelmod_no_salt = svelmod;
for i=1:sG.nx
    for j=1:sG.ny
        svelmod(i,j,salt.gtop(i,j):end) = 4400;
    end
end

clear sXX sYY
[sXX, sYY, sZZ] = meshgrid(sG.xx, sG.yy, sG.zz);

% save this model

f = top.f;

save([mlibfolder '/CRS/models/model_13_reflector.mat'], 'f');
save([mlibfolder '/CRS/models/model_13_G_file.mat'], 'sG');
```

Information about grid:
x0=0, dx=20, Nx=201.
y0=0, dy=20, Ny=201.
z0=0, dz=10, Nz=201.
t0=0, dt=0.001, Nt=1000.

Create central ray

```
X0 = [ 2000; 2000; 0];

[ t0, xref ]  = Get_model_exact_traveltime(X0, X0, 14);

tti  = FSM3D(sGold, xref, svelmod_no_salt);
xi = x2grid(X0(1), sG.x0, sG.dx, sG.nx);
yi = x2grid(X0(2), sG.y0, sG.dy, sG.ny);
zi = x2grid(X0(3), sG.z0, sG.dz, sG.nz);

dt = tti(xi,yi,zi)/sG.nx/1.002;
ray = zeros(3, sG.nx+1);
ray(:,1) = X0;

for i=1:sG.nx
    xi = x2grid(ray(1,i), sG.x0, sG.dx, sG.nx);
    yi = x2grid(ray(2,i), sG.y0, sG.dy, sG.ny);
    zi = x2grid(ray(3,i), sG.z0, sG.dz, sG.nz);
    pp(1,1) = (tti(xi+1,yi,zi) - tti(xi,yi,zi))/sG.dx;
    pp(2,1) = (tti(xi,yi+1,zi) - tti(xi,yi,zi))/sG.dy;
    pp(3,1) = (tti(xi,yi,zi+1) - tti(xi,yi,zi))/sG.dz;
    ray(:,i+1) = ray(:,i) - pp*dt/(norm(pp))^2;
```

```
end
ray(:,end) = xref;
```

Make simplified model

```
%[ t0, w, M, N ] = Get_model_stacking_parameters( X0, 14 );

CRS_param = MLD('/home/zmaw/u250128/Desktop/MLIB/CRS/models/
model_14_CRS_param.mat');

t0 = CRS_param.t0;
w  = CRS_param.w;
M  = CRS_param.M;
N  = CRS_param.N;
v0 = 1500;

smodel = Get_simplified_model(X0,v0,t0,w,M,N);

% Make central ray

cray = Get_smodel_central_ray(smodel);

% Make parabolic and ellipsoidal reflectors

[XX, YY] = meshgrid(-2000:100:2000, -2000:100:2000);
[TH, PH] = meshgrid(-pi:pi/100:pi, -pi/2:pi/100:pi/2);

ref_par = Get_smodel_reflector_parabolic(smodel, XX, YY);

ref_elc = Get_smodel_reflector_ellipsoidal_cart(smodel, XX, YY);

ref_ela = Get_smodel_reflector_ellipsoidal_polar(smodel, TH, PH);
```

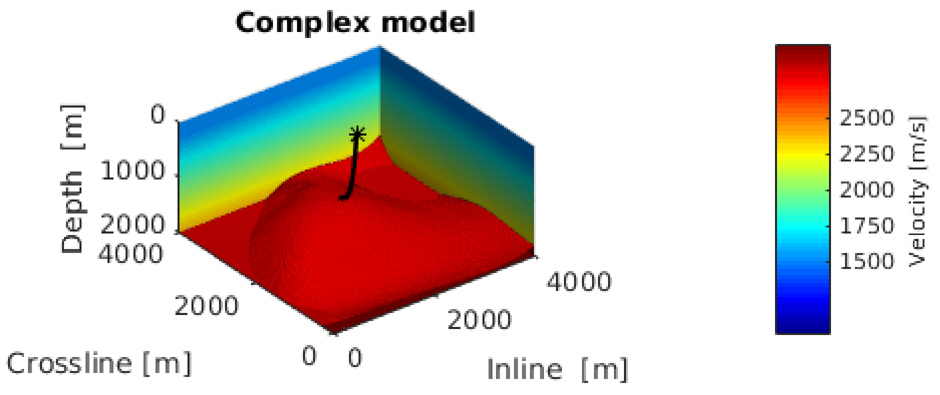
Create simplified velmod

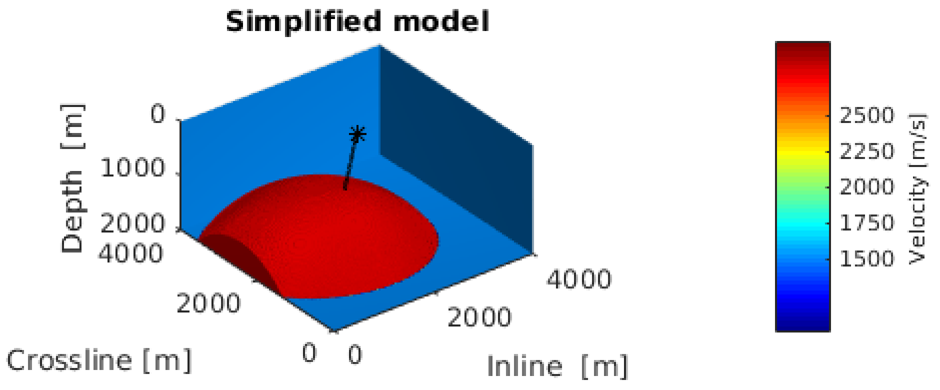
```
avelmod = ones(size(svelmod))*v0;

[XX, YY, ZZ] = meshgrid(sG.xx, sG.yy, sG.zz);
ind = Get_smodel_reflector_ellipsoidal_exist(smodel, XX, YY, ZZ);

avelmod(ind) = 4400;
```

Plot simplified 3D velocity cube

```
f3 = figure(3);
subplot(2,1,1);
xs = 4000;
ys = 4000;
zs = 2000;
h=slice(sXX,sYY,sZZ,svelmod,xs,ys,zs);
set(h,'FaceColor','interp','EdgeColor','none','DiffuseStrength',.8)
xlabel('Inline  [m]');
```

```
ylabel('Crossline [m]');
zlabel('Depth  [m]');
title('Complex model')
colormap('jet')
caxis([1000 3000])
camlight(100,100)
lighting gouraud
hold on
p = patch(isosurface(sXX,sYY,sZZ,svelmod,4200));
isonormals(sXX,sYY,sZZ,svelmod,p)
p.FaceColor = 'red';
p.EdgeColor = 'none';
daspect([1,1,1])
view(3);
hold on
plot3(ray(1,:),ray(2,:),ray(3,:),'-black', 'LineWidth',2  );
plot3(X0(1),X0(2),X0(3),'*black' );
plot3(xref(1), xref(2), xref(3), '.black','LineWidth',2 )
axis tight
set(gca, 'ZDir', 'reverse')
axis([0 4000 0 4000 0 2000])
c = colorbar('Ticks', [1500,1750 2000, 2250, 2500]);
c.Label.String = 'Velocity [m/s]';

subplot(2,1,2);
h=slice(sXX,sYY,sZZ,avelmod,xs,ys,zs);
set(h,'FaceColor','interp','EdgeColor','none','DiffuseStrength',.8)
xlabel('Inline  [m]');
ylabel('Crossline [m]');
zlabel('Depth  [m]');
title('Simplified model')
colormap('jet')
caxis([1000 3000])
camlight(100,100)
lighting gouraud
hold on
p = patch(isosurface(sXX,sYY,sZZ,avelmod,4200));
isonormals(sXX,sYY,sZZ,avelmod,p)
p.FaceColor = 'red';
p.EdgeColor = 'none';
daspect([1,1,1])
view(3);
hold on
plot3(cray.xx,cray.yy,cray.zz,'-black', 'LineWidth',2  );
plot3(X0(1),X0(2),X0(3),'*black' );
axis tight
set(gca, 'ZDir', 'reverse')
axis([0 4000 0 4000 0 2000])
c = colorbar('Ticks', [1500,1750 2000, 2250, 2500]);
c.Label.String = 'Velocity [m/s]';
```





Published with MATLAB® R2015a