SIO227A Homework 4 (Eric Gallimore)

Table of Contents

Do some setup 1
Do line fitting, find parameters tau, p. 1
Inversion 3
Make a table of velocity vs. depth 3
Find the Pn crossover distance 4
How thick is the crust? 4
How much error is in this estimate? 5

Do some setup

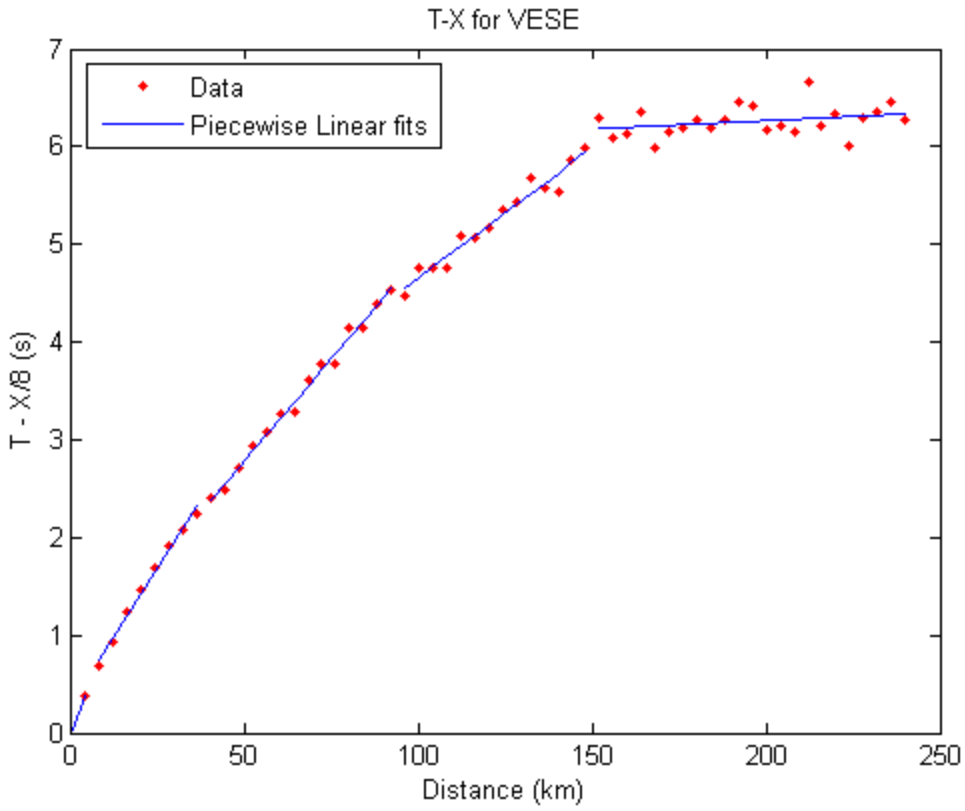
```
load('tx_points.txt')
close all;

% Check what we imported
%plot(tx_points(:,1), tx_points(:,2))
%hold on;
%plot(tx_points(:,1), tx_points(:,3), 'r.');

% Make the data easier to use
x = tx_points(:,1);
t = tx_points(:,3);

reduction_v = 8; %km/s

% Add a point at 0,0, since we know it would exist...
x = [0; x];
t = [0; t];

% Define regions of roughly linear slope by eye... there is surely a better
% way to do this?
sections{1} = 1:2;
sections{2} = 3:find(x==36); % x 8 to 36
sections{3} = sections{2}(end)+1:find(x==92); % x 40 to 92
sections{4} = sections{3}(end)+1:find(x==148); % x 96 to 148
sections{5} = sections{4}(end)+1:length(x); % x 152 to end
```

Do line fitting, find parameters tau, p.

```
figure();
plot(x, t, 'r.');
hold on;

% Fit a line to each sections
```

```
% This doesn't do a good job of drawing a line through the origin, but
% doing a good job would require hard work or using the Optimization Toolbox
% So, handle the 0 case as a special case contianing two points.
for i = 1:length(sections)
    pf(i,:) = polyfit(x(sections{i}), t(sections{i}), 1);
    l{i} = polyval(pf(i,:), x(sections{i}));

    plot(x(sections{i}), l{i});

    % slowness is the slope of each line.  Don't forget reduction velocity
    % ray parameter = slowness = 1/v
    p(i) = pf(i,1) + 1/reduction_v;

    % Now, get delay time
    % This is the y-intercept of each line
    tau(i) = pf(i,2);
end

% finish up the plot
title('T-X for VESE');
xlabel('Distance (km)');
ylabel('T - X/8 (s)');
legend('Data', 'Piecewise Linear fits', 'Location', 'NorthWest');

% Show these values
disp(p);
disp(tau);
```

```
    0.2195    0.1812    0.1664    0.1517    0.1268

         0    0.2924    0.7316    1.9953    5.9119
```



Inversion

First, build G matrix (using eqn 5.13) There are 5 sections, so this will be 5x5

```
G = zeros(length(sections));

% There's probably a way to do this in one line, but I keep screwing it up.
for i = 1:length(sections)
    % We need the absolute value to avoid getting imaginary numbers in our
    % result...
    G(i, :) = abs(2*sqrt(p(i)^2 - p.^2));
end

% G is only the lower triangular part of what we just built
G = tril(G);


% Now, find h (tau = Gh)
%h = G \ tau'; % <-- doesn't work, matrix is singular?
h = pinv(G) * tau';
```

Make a table of velocity vs. depth

```
v = 1./p;
% We need to include velocities at upper and lower boundaries.
```

```
v_vs_depth = zeros(2*length(h) - 1, 2);

% do edge case
v_vs_depth(1,1) = 0;
v_vs_depth(1,2) = v(1);

for i = 1:length(h)
    % find the depths
    depth_idx = 2*(i-1)+2;
    v_vs_depth(depth_idx, 1) = sum(h(1:i));
    v_vs_depth(depth_idx, 2) = v(i);
    if i < length(h)
        v_vs_depth(depth_idx+1, 1) = sum(h(1:i));
        v_vs_depth(depth_idx+1, 2) = v(i+1);
    end
end

fprintf('Depth (km)\tVelocity (km/s)\n');
disp(v_vs_depth);
```

```
        Depth (km) Velocity (km/s)
                 0    4.5558
            1.1796    4.5558
            1.1796    5.5201
            3.9336    5.5201
            3.9336    6.0082
           11.7817    6.0082
           11.7817    6.5928
           30.2950    6.5928
           30.2950    7.8881
           30.2950    7.8881
```

Find the Pn crossover distance

For this, we just look at the plot and notice the point where it seems like the slope experiences the greatest transition. I'm sure there is a better way to do this, but by identifying the points to either side of this inflection and taking the average, we can find that:

```
Pn = (152 + 148) / 2;

fprintf('Pn = %d km\n', Pn);
```

```
        Pn = 150 km
```

How thick is the crust?

Based on the same somewhat-qualitative explanation, we determine that the crust is everything "before" the crossover point (the first 4 layers). So, using the model, we see that:

```
fprintf('Crust thickness = %d km\n', sum(h(1:4)));
```

```
        Crust thickness = 3.029495e+001 km
```

How much error is in this estimate?

First and foremost, we should consider that the layers in the model were selected by visually identifing features on a plot. This is obviously a significant source of error, and is not easily quantified. We could probably develop a quantitative or iterative method to define layers, and then we would be able to assign uncertainty bounds to our layer definitions. The linear fits to the data in each layer do have quantified misfit, and these should also be taken into account.

Published with MATLAB® 7.12