SOEE3250/SOEE5675M/SOEE5116M 2018/19

Inverse Theory
Andy Hooper

Practical 1: Inversion by trial and error

In this practical you will learn how to set up an inverse problem, and use Matlab to 'solve' the inverse problem by running the associated forward problem multiple times.

There is a Matlab primer in the 'Practicals' folder on the VLE.

Background
You are tasked with monitoring surface deformation at a potential sinkhole. You set up a benchmark at a point (P_0) in the centre of the sinkhole and using a laser range finder, you measure the distance to P_0 from 4 points with known coordinates (P_1 to P_4). You would like to know the initial position of P_0 – this is an inverse problem. Your measurements are not your parameters of interest, but are related to them.

Point	X(m)	Y(m)	Z(m)	Range(m)
P_1	110	-65	20	370.2
P_2	172	423	25	392.8
P_3	826	-143	40	482.4
P_4	698	354	35	360.8

1) What are the parameters of interest (model parameters)? Is this a discrete or continuous inverse problem?
2) What are the data (i.e. measurements)?
3) Set up the forward problem: first,write down the equation that relates the coordinates of the unknown point (P_0) to the range from point P_1. Second, amend the equation to give the equation for a general point, P_i. What assumptions are you making in this forward operation?
4) Is this a linear or non-linear problem (i.e. can you reformulate it in the form **d**=**Gm**, where **d** is the vector of data and **m** is the vector of the model parameters)?
5) Start up Matlab. Create a function (**forward_operator.m**) that takes a column vector, **m,** as input and outputs a column vector of ranges (**d_hat**) that would be expected at points P_1 to P_4. To create the function, type `>>edit forward_operator.m`, and on the first line of the file that is opened, enter:

   ```
   function [d_hat]=forward_operator(m)
   ```

   Then enter the commands to do the actual calculations.

   You can add comments to the file by typing '%' – everything in the rest of the line is treated as a comment.
6) Plot the X, Y positions of P_1 to P_4 as blue crosses (use the Matlab function 'plot'; type `>>help plot` for info on how to use it). Add a title and label the axes using `>>title, >>xlabel and >>ylabel`. Make a rough guess at the position of P_0.

and plot this position as a red point (hint: type `>>hold on` to avoid the figure being cleared first).

7) Set up a column vector of observations (**d**) using the actual measurements.
8) Set up a vector (**m_trial**) with your guessed coordinates of P_0 (you will need to guess the elevation too). Run **forward_model.m** on this vector to get **d_hat** and subtract the result from **d**. These values are known as residuals. Calculate the sum of the residuals squared. This is known as the residual sum of squares (RSS) and is a measure of how well your model parameters fit the data (lower is better).
9) Adjust your guess for the coordinates of P_0, add this new point to your plot, then repeat Step 8. Has the RSS increased or decreased? Keep trying new guesses and see how low you can make the RSS. Submit your best guess for the coordinates to the VLE ('Location of P0' assignment under 'Practicals').