

Inverse Theory Practical Exercise: A nonlinear problem.

Hugh C. Pumphrey

November 11, 2008

1 Introduction

This exercise is intended to provide some experience in applying retrieval theory to a simple nonlinear problem. Your write-up should include a clear explanation of what you did, together with clear figures. Include parts of the code you wrote where it is relevant. Be sure to answer all the questions that are asked in these instructions. Please do not repeat large amounts of material from the notes or from these instructions.

Consider the situation shown in Figure 1.

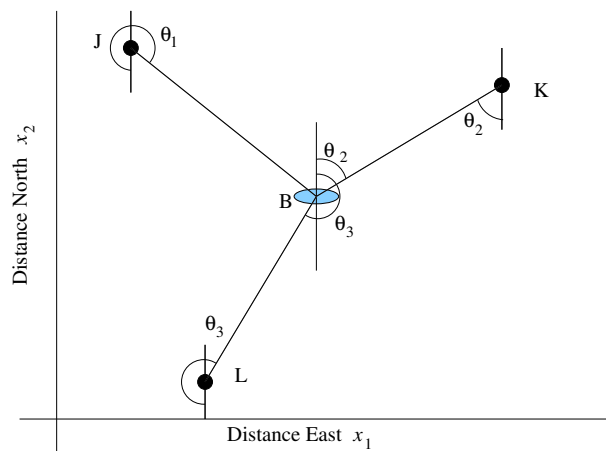


Figure 1: All at sea. The observer is in a boat at B, from where he can see three landmarks at J, K and L. His bearing compass allows him to measure the three angles θ_1 , θ_2 and θ_3 .

You are in a boat, out at sea, in the days before handheld GPS and other such wussy modern gadgets. You do, however, have an old-fashioned bearing compass. You can see some landmarks in the distance, which you can identify on your chart of the area. You can use the bearing compass to measure the direction towards each of these landmarks, that is, the angles θ_1 , θ_2 and θ_3 . In this exercise, you will use these measurements to estimate the position of the boat (x_1, x_2) .

Traditionally, this was done by drawing lines on the chart, passing through the landmarks, at the measured angles, as shown in Figure 2. Measurement error meant that the lines did not cross at a single point, but formed a small triangle or *cocked hat*. The usual assumption was that your boat was in the middle of the triangle, somewhere. In this exercise, we will treat the problem as an inverse problem of the type we have learned about in the lectures. The measurement vector \mathbf{y} is the three angles: $\mathbf{y} = (\theta_1, \theta_2, \theta_3)$. The state vector \mathbf{x} which we wish to estimate is the boat's position: $\mathbf{x} = (x_1, x_2)$. As in all the examples we have already studied, we know the forward function giving \mathbf{y} in terms of \mathbf{x} .

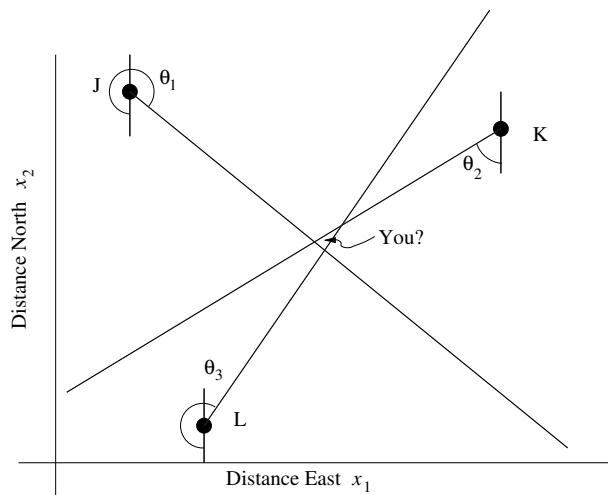


Figure 2: The traditional solution. The boat was supposed to be in the centre of the triangle or *cocked hat* as it was known, with the size of the triangle giving some clue as to the error.

2 The forward function

Use your knowledge of trigonometry to write down the forward function F which gives \mathbf{y} in terms of \mathbf{x} and of the landmark positions? Write a function in R (or your language of choice) that implements it¹.

Assume that the three landmarks have positions $\mathbf{x}_1 = (9, 1)$, $\mathbf{x}_2 = (1, 9)$ and $\mathbf{x}_3 = (9, 9)$ What value does your function return for \mathbf{y} if the position of the ship is:

1. $\mathbf{x} = (2, 3)$
2. $\mathbf{x} = (5, 10)$
3. $\mathbf{x} = (8, 5)$

You may well want to draw a scale diagrams of these three cases and check with a protractor that the answers returned by your program make sense.

3 The cost function

The problem of determining \mathbf{x} from \mathbf{y} is over-determined, so you do not need any *a priori* information to solve it. We can minimise the following cost function:

$$C(\mathbf{x}) = (\mathbf{y} - F(\mathbf{x}))^T \mathbf{S}_y^{-1} (\mathbf{y} - F(\mathbf{x}))$$

Write a function in R (or your language of choice) that calculates the cost function, assuming that:

- the three landmarks again have positions $\mathbf{x}_1 = (9, 1)$, $\mathbf{x}_2 = (1, 9)$ and $\mathbf{x}_3 = (9, 9)$.
- The three measured angles are $\theta_1 = 110^\circ$, $\theta_2 = 332^\circ$ and $\theta_3 = 43^\circ$.

Remember that $\mathbf{y} - F(\mathbf{x})$ is a difference of two *angles*, and that the difference between -179° and $+179^\circ$ is really only 2° . You may assume that you can measure a bearing with an error of $\pm 2^\circ$ and that the error in measuring one bearing is not correlated with the error in measuring another bearing.

Make a contour plot of the cost function over the region $0 < x_1 < 10$, $0 < x_2 < 10$, superimposing the lines like those in Figure 2. You may have to choose your contour lines with care — remember that they do not have to be uniformly spaced, but that non-evenly-spaced contours need to be carefully labelled.

¹Hint: to avoid sign errors, use the `atan2` function.

Note that your function should vary smoothly except in the immediate vicinity of the three landmarks. If you get vertical or horizontal lines across which your cost function changes by a large amount, then you have a mistake that you need to correct.

How well does the minimum of your cost function appear to correspond to the cocked hat?

4 Minimising the cost function

4.1 Inverse Hessian

The Gauss-Newton or Inverse Hessian formula for minimising this cost function is:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + (\mathbf{K}_i^T \mathbf{S}_y^{-1} \mathbf{K}_i)^{-1} \mathbf{K}_i^T \mathbf{S}_y^{-1} [\mathbf{y} - F(\mathbf{x}_i)] \quad (1)$$

where

$$\mathbf{K} = \left. \frac{d\mathbf{y}}{d\mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}_i}$$

Write a program to find the value of \mathbf{x} which minimises C using this formula. Again, you will need to remember that $\mathbf{y} - F(\mathbf{x})$ is a difference of two angles, and that the difference between -179° and $+179^\circ$ is really only 2° . Decide on a suitable criterion for convergence. You will need to think about how to calculate \mathbf{K} . You can always resort to perturbing each element of \mathbf{x} by a small amount and seeing how \mathbf{y} changes. But in a case like this, you may be able to use your trig and calculus skills to find a formula for \mathbf{K} . Your code should take the following steps:

1. Set \mathbf{x} to a starting point \mathbf{x}_0
2. Use your forward model to calculate $F(\mathbf{x})$ from your current value of \mathbf{x} .
3. Calculate \mathbf{K} from your current value of \mathbf{x}
4. Calculate your new value of \mathbf{x} using equation 1
5. Check to see if your new value of \mathbf{x} is significantly different from the previous value. If it is, set \mathbf{x} to the value you just calculated and go back to step 2
6. Write out your final value of \mathbf{x} and your final value of $\hat{\mathbf{S}} = (\mathbf{K}_i^T \mathbf{S}_y^{-1} \mathbf{K}_i)^{-1}$

Start the Inverse Hessian formula off at each of the following values

- $\mathbf{x}_0 = (1, 2)$
- $\mathbf{x}_0 = (10, 9)$
- $\mathbf{x}_0 = (10, 9.9)$

Does the method converge for all of these starting points (or indeed any of them)? How many iterations does it take to reach your convergence criterion in the cases that converge? For any converging cases, make a plot showing \mathbf{x}_i for all the steps, so that you can see how the solution is approached.

4.2 Steepest Descents

The steepest descents formula for our example is:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \gamma^{-1} \mathbf{K}_i^T \mathbf{S}_y^{-1} [\mathbf{y} - F(\mathbf{x}_i)] \quad (2)$$

Solve the problem using this formula for the following starting points

- $\mathbf{x}_0 = (10, 9.9)$
- $\mathbf{x}_0 = (11, 11)$

Your code should be similar to the code you wrote for the inverse Hessian case. The only difference is that you should calculate your estimate of \mathbf{x} using equation 2 instead of equation 1.

What is a suitable value of gamma? How many iterations does it take to reach your convergence criterion in the cases that converge? For both of the above cases, make a plot showing \mathbf{x}_i for all the steps, so that you can see how the solution is approached.

4.3 Marquadt-Levenberg

The Marquadt-Levenberg formula for this example is:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + (\mathbf{K}_i^T \mathbf{S}_y^{-1} \mathbf{K}_i + \gamma \mathbf{D})^{-1} \mathbf{K}_i^T \mathbf{S}_y^{-1} [\mathbf{y} - F(\mathbf{x}_i)]$$

Solve the same problem as above using this formula, starting from the same two points that you used for the steepest descents formula. Your program will have to be a bit more complicated. Each time you calculate a new \mathbf{x} you will have to evaluate the cost function and see if it got smaller or larger. If it got larger you will have to re-do the iteration with a larger γ , starting at the same \mathbf{x} as the previous step. If it got smaller, you can start the next iteration with your new value of \mathbf{x} and use a smaller γ . Does it always converge, this time? How many iterations does it take? How did you choose the parameter γ for each iteration? Remember that γ should be very small by the time you have found the solution.

5 How good is your answer?

The Marquadt-Levenberg and Gauss-Newton methods both return a covariance matrix $\hat{\mathbf{S}}$ for the solution:

$$\hat{\mathbf{S}} = (\mathbf{K}_i^T \mathbf{S}_y^{-1} \mathbf{K}_i)^{-1}$$

(If both methods converged, then they should return the same matrix $\hat{\mathbf{S}}$ as well as the same solution $\hat{\mathbf{x}}$.) What is this matrix for the solution you found to the above problem?

6 A nastier case

Investigate the problem for the case where your boat and the last landmark were in different places, so that the landmark positions are: $\mathbf{x}_1 = (9, 1)$, $\mathbf{x}_2 = (1, 9)$ and $\mathbf{x}_3 = (8, 2)$. This time, the measured angles are $\theta_1 = 128^\circ$, $\theta_2 = 322^\circ$ and $\theta_3 = 126^\circ$. In what ways is the cost function different from the previous case? Where is your boat now? How well do you know this? (Think about all of $\hat{\mathbf{S}}$ before when giving your answer.) How well do the three solution methods work in this case? What problems do they encounter?