EC5.406 Signal Detection & Estimation Theory
Assignment 3

Niharika Vadlamudi
2018122008

December 3rd 2021

1 Problem I

Assume that the two sources transmit different signals $s_1(n) = p$ and $s_2(n) = np$ where p is the baseline transmission power. Received signal at the i-th receiving node is :

$$R_i(n) = \sum_{k=1}^{2} d_{ik}^{-2} s_k(n) + w(n), \quad \text{for } n = 1, \dots, N$$

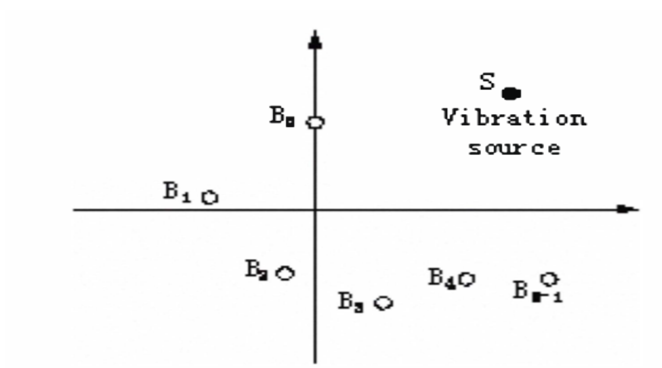
where d_{ik} is the distance from the i-th receiving node to the k-th source and $w(n)$ s are the samples of WGN process of variances σ^2. Assuming the locations of receiving nodes are known, estimate the locations of sources using Netwon-Raphson (or scoring method) and expectation maximization algorithms. Also, implement these source localization algorithms in Matlab. The Matlab codes should provide the plots (for comparing these two algorithms) of (a) average convergence time (b) mean square error of localization (individually for Source 1 and Source 2)

---

2 Newton Raphson Method

The Newton Raphson Method , is an iterative method to solve estimation of parameter problem . The basic underlying theory is we perform a grid search over the parameters in suitable step-size while satisfying the desired maxima/minima criteria . Here , we use Scoring Method , where we make a small assumption while performing gradient descent to save on computation . We essentially replace the Hessian matrix at every kth step, to a fixed Fisher Information matrix . A generic formula for this method :

$$\theta_{k+1} = \theta_k + I^{-1}(\theta) \frac{\partial \ln p(\mathbf{x};\theta)}{\partial \theta}\Big|_{\theta=\theta_k}$$



Given our setting , we have 2 sources $s_i(n)$, and M receiver nodes . It can be observed that the power received at R_i is simply the summation of source node powers , so they can be separated and optimised in nature . Now, a generic Receiver Strength Signal model can be considered as :

$$y_n = 10\log_{10}\left(\frac{\Psi}{|\boldsymbol{r}_n - \boldsymbol{l}|^{\alpha}}\right) + v_n, n = 1, 2, \ldots, N. \tag{1}$$

Here , y_n will be the received strength $(r_i(n))$, α is the attenuation constant and $|\boldsymbol{r}_n - \boldsymbol{l}|^{\alpha}$, is the distance between r_n and source vector l. So , in the following derivation we observe the behavior between 1 receiver node and 1 source node , and derive the equations . Note that ψ will be a function of n wrt source node 2 . Note that here , N is the number of receiver node and M is the length of time steps.

$$p(\mathbf{y};\boldsymbol{\theta}) = \left(\tfrac{1}{\sqrt{2\pi\sigma^2}}\right)^N \exp\left\{-\tfrac{1}{2\sigma^2}\textstyle\sum_{n=1}^{N}\left[y_n - f_n(\boldsymbol{\theta})\right]^2\right\} \tag{2}$$

$$f_n(\boldsymbol{\theta}) = 10\log_{10}\left[\frac{\Psi(n)}{g_n(\boldsymbol{\theta})}\right] \tag{3}$$

$$g_n(\boldsymbol{\theta}) = g_n(x, y) = |\mathbf{r}_n - \mathbf{l}|^{\alpha} = \left(r_{n_x} - x\right)^2 + \left(r_{n_y} - y\right)^2 \tag{4}$$

Then the vector log-likelihood function is,

$$\ln p(\mathbf{y};\boldsymbol{\theta}) = \ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^N + \left[-\frac{1}{2\sigma^2}\sum_{n=1}^{N}\left(y_n - f_n\right)^2\right] \tag{5}$$

Now , we start deriving the partial derivatives of log-likelihood function w.r.t to x and y variables .

$$\frac{\partial \ln p}{\partial x} = \frac{\partial}{\partial x}\left[-\frac{1}{2\sigma^2}\sum_{n=1}^{N}\left(y_n - f_n\right)^2\right] = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}\underbrace{\left\{\frac{\partial}{\partial x}\left[\left(y_n - f_n\right)^2\right]\right\}}_{\mathcal{A}} \tag{6}$$

$$\mathcal{A} = \frac{\partial}{\partial x}\left[\left(y_n - f_n\right)^2\right] = 2\left[y_n - f_n(\boldsymbol{\theta})\right]\underbrace{\left[-\frac{\partial f_n(x, y)}{\partial x}\right]} \tag{7}$$

$$\begin{aligned}\mathcal{B} &= \frac{\partial f_n(x, y)}{\partial x} = \frac{\partial}{\partial x}\left\{10\log_{10}\left[\frac{\Psi(n)}{g_n(x, y)}\right]\right\} \\ &= \frac{\partial}{\partial x}\left[10\log_{10}\Psi(n) - 10\log_{10}g_n(x, y)\right] = \frac{\partial}{\partial x}\left[10\log_{10}g_n(x, y)\right] \\ &= -10\frac{\partial}{\partial x}\left[\log_{10}g_n(x, y)\right] = -\frac{10}{\ln 10}\frac{\left[\partial g_n(x, y)/\partial x\right]}{g_n(x, y)} \\ &= \frac{20}{\ln 10}\frac{\left(r_{n_x} - x\right)}{g_n}\left(\because \frac{\partial g_n}{\partial x} = -2\left(r_{n_x} - x\right)\right)\end{aligned} \tag{8}$$

After plugging $\mathcal{A}$ and $\mathcal{B}$ in the above result , we get :

$$\begin{aligned}\frac{\partial \ln p}{\partial x} &= -\frac{1}{2\sigma^2}\sum_{n=1}^{N}\left\{2\left[y_n - f_n(\boldsymbol{\theta})\right]\left[-\frac{20}{\ln 10}\frac{\left(r_{n_x} - x\right)}{g_n(x, y)}\right]\right\} \\ &= \frac{20}{\sigma^2\ln 10}\sum\left\{\left[y_n - f_n(\boldsymbol{\theta})\right]\left[\frac{\left(r_{n_x} - x\right)}{g_n(x, y)}\right]\right\} \\ &= \frac{20}{\sigma^2\ln 10}\sum\left\{\frac{\left[y_n - f_n(\boldsymbol{\theta})\right]\left(r_{n_x} - x\right)}{g_n(x, y)}\right\}\end{aligned} \tag{9}$$

Finally , we obtain the equations as :

$$\frac{\partial \ln p}{\partial x}=\frac{20}{\sigma^2 \ln 10} \sum\left\{\frac{\left[y_n-f_n(\boldsymbol{\theta})\right]\left(r_{n_x}-x\right)}{g_n(x, y)}\right\} . \tag{10}$$

$$\frac{\partial \ln p}{\partial y}=\frac{20}{\sigma^2 \ln 10} \sum\left\{\frac{\left[y_n-f_n(\boldsymbol{\theta})\right]\left(r_{n_y}-y\right)}{g_n(x, y)}\right\} \tag{11}$$

On double differentiation of equations 10 & 11 , we obtain :

$$\frac{\partial^2 \ln p}{\partial x^2}=\frac{20}{\sigma^2 \ln 10} \sum\left[\frac{2\left(r_n-x\right)^2\left(y_n-f_n\right)-\left(y_n-f_n\right) \cdot g_n-\frac{20}{\ln 10} \cdot\left(r_{n_x}-x\right)^2}{{g_n}^2}\right] \tag{12}$$

$$\frac{\partial^2 \ln p}{\partial y^2}=\frac{20}{\sigma^2 \ln 10} \sum\left[\frac{2\left(r_n-y\right)^2\left(y_n-f_n\right)-\left(y_n-f_n\right) \cdot g_n-\frac{20}{\ln 10} \cdot\left(r_{n_y}-y\right)^2}{{g_n}^2}\right] \tag{13}$$

We can say the $\frac{\partial^2 \ln p}{\partial y \partial x}$ is equivalent to $\frac{\partial^2 \ln p}{\partial x \partial y}$ via symmetry statement , hence we get :

$$\frac{\partial^2 \ln p}{\partial x \partial y}=\frac{20}{\sigma^2 \ln 10} \sum\left[\frac{2\left(r_{n_x}-x\right)\left(r_{n_y}-y\right)\left(y_n-f_n\right)-\frac{20}{\ln 10}\left(r_{n_x}-x\right)\left(r_{ny}-y\right)}{g_n^2}\right] \tag{14}$$

To find the Fisher information matrix,

$$\mathbf{I}(\boldsymbol{\theta})=\left[\begin{array}{cc} -E\left(\frac{\partial^2 \ln p}{\partial x^2}\right) & -E\left(\frac{\partial^2 \ln p}{\partial x \partial y}\right) \\ -E\left(\frac{\partial^2 \ln p}{\partial y \partial x}\right) & -E\left(\frac{\partial^2 \ln p}{\partial y^2}\right) \end{array}\right]$$

we make the assumption that $E(f)=y_n$, so :

$$E\left(\frac{\partial^2 \ln p}{\partial x \partial y}\right)=\frac{20}{\sigma^2 \ln 10} \sum\left[\frac{-\frac{20}{\ln 10} \cdot\left(r_{n_x}-x\right)\left(r_{n_y}-y\right)}{g_n^2}\right]=-\left(\frac{20}{\sigma \ln 10}\right)^2 \sum\left[\frac{\left(r_{n_x}-x\right)\left(r_{n_y}-y\right)}{g_n^2}\right] \tag{15}$$

Therefore,

$$\mathbf{I}(\boldsymbol{\theta})=\left(\frac{20}{\sigma \ln 10}\right)^2 \cdot\left[\begin{array}{cc} \sum\left[\frac{\left(r_{n_x}-x\right)^2}{{g_n}^2}\right] & \sum\left[\frac{\left(r_{n_x}-x\right)\left(r_{n_y}-y\right)}{{g_n}^2}\right] \\ \sum\left[\frac{\left(r_{n_x}-x\right)\left(r_{n_y}-y\right)}{{g_n}^2}\right] & \sum\left[\frac{\left(r_{n_y}-y\right)^2}{{g_n}^2}\right] \end{array}\right] \triangleq\left(\frac{20}{\sigma \ln 10}\right)^2\left[\begin{array}{cc} a & b \\ c & d \end{array}\right] \tag{16}$$

After taking inverse of the matrix , we get :

$$\mathbf{I}^{-1}(\boldsymbol{\theta})=\left(\frac{20}{\sigma \ln 10}\right)^2 \times \frac{1}{a d-b c}\left[\begin{array}{cc} d & -b \\ -c & a \end{array}\right] \tag{17}$$

Now , we plug it the equations $17, 10, 17$ to form the NR iterative formula as :

$$\boldsymbol{\theta_{k+1}}=\boldsymbol{\theta_k}+\mathbf{I}^{-1}(\boldsymbol{\theta}) * \left.\frac{\partial \ln p(\mathbf{y} ; \theta)}{\partial \theta}\right|_{\theta=\theta_k} \tag{18}$$

2.1 Assumptions

- As mentioned before , we will isolate the procedure for each source node , and receiver node as well . If there are M nodes , the we get M number of estimations of (xs, ys) and an average of these vectors will be computed . So, the more nodes will be able to localise with positions with greater precision .
- All the co-ordinates of receiver nodes R_i will be chosen within a 10x10 grid. The total number of time steps - N will be ranging from 20 , 40 , 80 and we fix the number of receiver nodes M to be 5 .
- The σ^2 parameter will be set 100dB and the value of p will be 1 .
- Based on the value of p, we can say that $s_1(n)=1$ & $s_2(n)=n$ for all n .

2.2 Source Code for Newton Raphson Method

The following code is for source 1 localisation . There are 2 scripts attached , under the name $source_{2n}etwon_{r}aphson.m$ in the source folder , only change occurs w.r.t signal function . A brief description about the code block :

- Lines 1- 13 Global Variables declaration
- Lines 15 - 42 we try out the experiment for different N value , but fix other parameters like SNR .
- Lines 47-64 Generic functions to be defined - s1(n), res(s,l)(computes the distance between source and receiver vector) .
- Lines 66 - 72 Here we generate the random position values and receiver power values for a given N M configuration .
- Lines 83-43 A set of helper functions written to simplify the Fisher Matrix computation .
- Lines 145 - 170 Newton Raphson Method code - we decide convergence based criterio based on error.

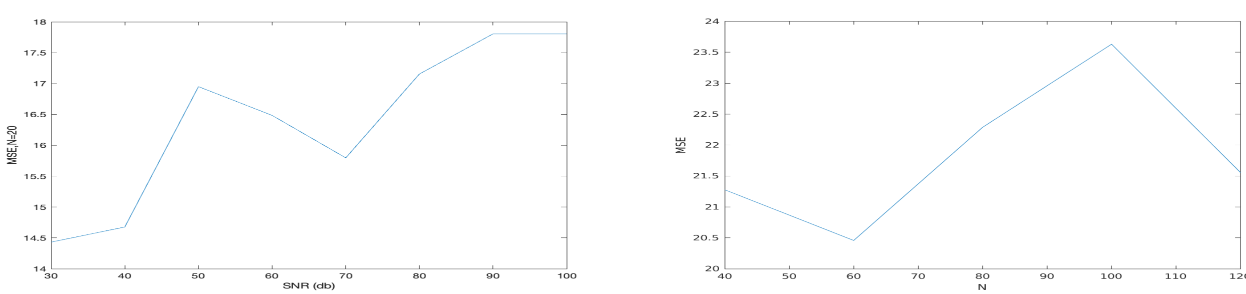
```
1 %% Script for Source 1 Localisation
2 %%Global Variables.
3 p = 1 ;
4 SNR=80;
5 sigma=1/db2pow(SNR);
6 beta=20/sigma*(log(10));
7 beta_sq=(20./sqrt(sigma)*log(10));
8 eps=0.005;
9 % Grid size of 10 x 10 .
10 radius=50;
11 % Fixing source 1
12 s1_pos_gd=[14;11];
13
14 M=10;
15 meanErrorList=[];
16 N_list=[40,60,80,100,120];
17 Rpos=generateRandomPosition(M,radius);
18 L=length(N_list);
19 for i=1:1:L
20     N=N_list(i);
21     thetha_sum=[0;0];
22     thetha_init=[0;0];
23     for n=1:1:N
24         yR_n=recieverPower(M,sigma,radius);
25         thetha_opt=newtonRaphson(M,Rpos,thetha_init,eps,yR_n,beta,beta_sq,p,radius);
26         thetha_sum=thetha_sum+thetha_opt;
27     end
28     thetha_final=(1/N)*(thetha_sum);
29     meansqerr=norm(thetha_final-s1_pos_gd);
30     meanErrorList(i)=meansqerr;
31     fprintf('Iter : %d , MSE : %f Thetha ->[%d %d]\n:',i,meansqerr,int16(thetha_final))
32     fprintf('\n')
33 end
34
35 %% Plotting- Computing for multiple values of SNR figure
36 f1 = figure();
37 set(f1, 'Visible', 'off');
38 title('mse vs snr ')
39 plot(N_list,meanErrorList)
40 xlabel('N')
41 ylabel('MSE')
42 saveas(gcf,'res1.png')
43
44 %% Helper Functions
45
46 % Source 1
47 function f1=s1(p)
48     f1=p;
49 end
50
51 % g(x,r) function
52 function res=g(r,l)
53     rx=r(1);
```

```
    ry=r(2);
    lx=l(1);
    ly=l(2);
    res=(rx-lx).^2+(ry-ly).^2;
end

% f function
function fval = ft(p,r,l)
    fval=10*log10(s1(p)./g(r,l));
end

% Random Position Generator - Rx nodes.
function R=generateRandomPosition(M,radius)
    R_x=randi([0 radius],1,M)+ randn([1 M]);
    R_y=randi([0 radius],1,M)+ randn([1 M]);
    R=[R_x;R_y];
end

% Random Signal Value generator for Rx
function yR=recieverPower(M,sigma,radius)
    w=sigma*randn([1 M]);
    dterm=radius.^2/2*randn([1 M])+0.25*radius.^2;
    p=1;
    for i=1:1:M
        yR(i)=10*log10(p*1/dterm(i))+w(i)*(sigma)*rand(1);
    end
    yR=transpose(yR);
end

% d(lnP)/dx function
function val = dlnpx(r,y,l,factor,p)
    rx=r(1);
    lx=l(1);
    num= (y-ft(p,r,l))*(rx-lx);
    den=g(r,l);
    val=factor*(num./den);
end

% d(lnP)/dy function for s1(n) signal .
function val = dlnpy(r,y,l,factor,p)
    ry=r(2);
    ly=l(2);
    num=(y-ft(p,r,l))*(ry-ly);
    den=g(r,l);
    val=factor*(num./den);
end

% @S1First derivative of log-likelihood function [ d(ln(p)/dx d(ln(p)/dy ]
function p1=lnpvec(M,R,Y,l,factor,p)
    fx=0;
    fy=0;
    for i=1:1:M
        r_i=R(:,i);
        y_i=Y(i);
        fx=fx+dlnpx(r_i,y_i,l,factor,p);
        fy=fy+dlnpy(r_i,y_i,l,factor,p);
    end
    p1=[fx;fy];
end

% Single Fisher Matrix .
function I_m=fisher_m(r,l,factor)
 rx=r(1);
 ry=r(2);

 lx=l(1);
 ly=l(2);

 g_sq= g(r,l).^2;

 % Matrix Components
```

```
 I_a=(rx-lx).^2/g_sq;
 I_b=(rx-lx)*(ry-ly)/g_sq;
 I_c=I_b;
 I_d=(ry-ly).^2/g_sq;

 I_m=factor*[[I_a I_b];[I_c I_d]];
end

% Complete Fisher Matrix - I for M observations .
function f=I(M,R,l,factor)
    f=[[0 0];[0 0]];
    for i=1:1:M
        r_i=R(:,i);
        f=f+fisher_m(r_i,l,factor);
    end
    % Perform Inverse of the matrix
%     f=(1/M)*f;
    f=inv(f);
end

% Newton Raphson Algorithm.
function thetha_opt=newtonRaphson(M,R,thetha_init,eps,Y,beta,beta_sq,p,radius)
    thetha_old=thetha_init;
    factor1=beta;
    factor2=beta_sq;
    err=inf;
    count=0;
    while(err>=eps)
        Q=mtimes(I(M,R,thetha_old,factor1),lnpvec(M,R,Y,thetha_old,factor2,p));
        if(isinf(Q)|isnan(Q))
            thetha_new=thetha_old;
            err=0;
            fprintf('Breaking due to Q-inf\n')
            break
        end
        % Update the vector.
        thetha_new=thetha_old+Q;
        err=norm(thetha_new-thetha_old);
        thetha_old=thetha_new;
        count=count+1;
    end
    % Now , the while loop has broken .
    thetha_opt=thetha_new;
    fprintf('Iterations Taken : %d\n',count)
end
```

2.3 Results

In the following figure , we have source signal $s1(n)$'s analysis of varying N vs MSE ; and effects of fixing SNR value (=100 dB) & varying N . One common observation is that by increasing M (i.e the number of receiver nodes) the performance increases ; as more data is being added to the equation.

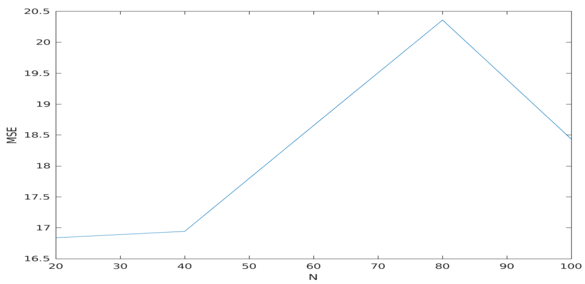


(a) SNR vs N

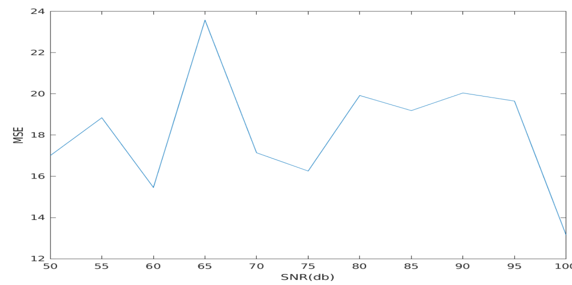
(b) N vs MSE

Figure 1: Source I

Same analysis performed on Source 2 , results in :



(a) MSE vs N , SNR = 100



(b) MSE vs SNR , N = 20

Figure 2: Source 2

On right-hand side , we can observe that the MSE value reduces as we increase the SNR value , this is true since the noise variance is being decreases hence more precision for localisation of source node .

3 References

- Vibratory Source's Search and Localization Algorithm of Newton Iteration Based on Method of Weighted Least Squares
- Stephen Kay's Estimation Theory Volume - I (MLE Chapter)
- Iterative Maximum Likelihood Locating Method Based on RSS Measurement-Jaechan Le.
- Received Signal Strength (RSS) Location Estimation with Nuissance Parameters in Correlated Shadow Fading .