Technique for Numerical Computation of Cramér-Rao Bound using MATLAB

Hing Cheung So

http://www.ee.cityu.edu.hk/~hcso

Department of Electronic Engineering
City University of Hong Kong

The purpose of this note is to provide an effective means to compute the Cramér-Rao lower bound (CRLB) numerically with the use of symbolic computation in MATLAB.

CRLB is the performance bound in terms of minimum achievable variance provided by any unbiased estimators.

For deterministic parameter estimation, its derivation requires knowledge of the noise probability density function (PDF) which must have closed-form so that analytical differentiation is allowed.

Although there are other variance bounds, CRLB is simplest.

Suppose the PDF of $\mathbf{r} = \mathbf{f}(\mathbf{x}) + \mathbf{w}$ is $p(\mathbf{r}; \mathbf{x})$. Here, $\mathbf{f}$ is a known linear/nonlinear function, $\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_L]^T \in \mathbb{R}^L$ is the parameter vector, and $\mathbf{w}$ is noise vector.

The CRLB for $\mathbf{x}$ can be obtained in two steps:

- Compute the Fisher information matrix $\mathbf{I}(\mathbf{x})$ where

$$\mathbf{I}(\mathbf{x}) = \begin{bmatrix} -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial^2 x_1}\right\} & -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial x_1 \partial x_2}\right\} & \cdots & -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial x_1 \partial x_L}\right\} \\ -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial x_2 \partial x_1}\right\} & -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial^2 x_2}\right\} & \cdots & -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial x_2 \partial x_L}\right\} \\ \vdots & \vdots & \vdots & \vdots \\ -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial x_L \partial x_1}\right\} & \cdots & \cdots & -E\left\{\frac{\partial^2 \ln p(\mathbf{r};\mathbf{x})}{\partial^2 x_L}\right\} \end{bmatrix}$$

- CRLB for x_l is the (l,l) entry of $\mathbf{I}^{-1}(\mathbf{x})$, $l = 1, 2, \cdots, L$.

Consider $r[n] = A + w[n]$, $n = 1, \cdots, N$, where $A \in \mathbb{R}$ is the parameter to be estimated and $w[n] \in \mathbb{R}$ is zero-mean white Gaussian noise with variance σ_w^2. The CRLB is derived as:

$$p(\mathbf{r}; A) = \frac{1}{(2\pi\sigma_w^2)^{N/2}} e^{-\frac{1}{2\sigma_w^2}\sum_{n=1}^{N}(r[n]-A)^2}$$

$$\Rightarrow \ln p(\mathbf{r}; A) = -\ln((2\pi\sigma_w^2)^{N/2}) - \frac{1}{2\sigma_w^2}\sum_{n=1}^{N}(r[n]-A)^2$$

$$\Rightarrow \frac{\partial^2 \ln p(\mathbf{r}; A)}{\partial^2 A} = -\frac{N}{\sigma_w^2}$$

$$\Rightarrow \mathbf{I}(A) = -E\left\{\frac{\partial^2 \ln p(\mathbf{r}; A)}{\partial^2 A}\right\} = \frac{N}{\sigma_w^2}$$

$$\Rightarrow \mathbf{I}^{-1}(A) = \frac{\sigma_w^2}{N}$$

The CRLB can be obtained easily in closed-form for this simple example.

How about for more complicated problems? We use another simple example for its numerical calculation as follows.

Consider

$$r[n] = A_1 e^{j(\omega_1 n + \phi_1)} + A_2 e^{(j\omega_2 n + \phi_2)} + w[n], \quad n = 1, \cdots, N$$

where $\mathbf{x} = [\omega_1 \quad \omega_2 \quad \phi_1 \quad \phi_2 \quad A_1 \quad A_2]^T \in \mathbb{R}^6$ and $w[n] \in \mathbb{C}$ is zero-mean white complex Gaussian noise with variance σ_w^2.

Ignoring the irrelevant term, the log PDF is:

$$\begin{aligned}
&\ln p(\mathbf{r};\mathbf{x}) \\
&= -\frac{1}{\sigma_w^2} \sum_{n=1}^{N} \left(r[n] - A_1 e^{j(\omega_1 n + \phi_1)} - A_2 e^{j(\omega_2 n + \phi_2)} \right) \\
&\left(r^*[n] - A_1 e^{-j(\omega_1 n + \phi_1)} - A_2 e^{-j(\omega_2 n + \phi_2)} \right)
\end{aligned}$$

We can see that the main computation is to find the second-order derivatives for:

$$p = \left(r[n] - A_1 e^{j(\omega_1 n+\phi_1)} - A_2 e^{j(\omega_2 n+\phi_2)}\right)\left(r^*[n] - A_1 e^{-j(\omega_1 n+\phi_1)} - A_2 e^{-j(\omega_2 n+\phi_2)}\right)$$

With the use of symbolic computation of MATLAB, we define:

```
syms w1 w2 p1 p2 A1 A2 n s1 s2 p;
p=(s1-A1*exp(j*w1*n)*exp(j*p1)-
A2*exp(j*w2*n)*exp(j*p2))*(s2-A1*exp(-
j*w1*n)*exp(-j*p1)-A2*exp(-j*w2*n)*exp(-j*p2));
```

Here, the first 6 symbols correspond to $\mathbf{x}$, and the remaining symbols are time index (n), $r[n]$ (s1), $r^*[n]$ (s2) and p (p).

For example, in computing $\frac{\partial^2 p}{\partial^2 \omega_1}$ in analytical form, we use:

```
p11=simple(diff(diff(p,w1),w1))
```

Note that `simple` is used to simplify the resultant expression.

In a similar manner, we compute all other required second-order derivatives. Note also that we only need to determine the upper triangular elements of $\mathbf{I}(\mathbf{x})$ as the matrix is symmetric.

For detail, please refer to `comp_diff.m` or see below:

http://www.ee.cityu.edu.hk/~hcso/comp_diff.m

comp_diff.m

```
%Parameter vector consists of symbols:
%frequencies: w1 and w2;
%initial phases: p1 and p2;
%real-valued amplitudes: A1 and A2

%Other defined symbols are:
%r[n] and its conjugate: s1 and s2;
%time index: n;
%log PDF: p

%pij stands for the (i,j) entry of the FIM (individual and without expectation)
%e.g., p11 corresponds to differentiation of p w.r.t. w1 twice

syms w1 w2 p1 p2 A1 A2 n s1 s2 p;
p=(s1-A1*exp(j*w1*n)*exp(j*p1)-A2*exp(j*w2*n)*exp(j*p2))*(s2-A1*exp(-j*w1*n)*exp(-j*p1)-A2*exp(-j*w2*n)*exp(-j*p2));
p11=simple(diff(diff(p,w1),w1))
p12=simple(diff(diff(p,w1),w2))
p13=simple(diff(diff(p,w1),p1))
p14=simple(diff(diff(p,w1),p2))
p15=simple(diff(diff(p,w1),A1))
p16=simple(diff(diff(p,w1),A2))
p22=simple(diff(diff(p,w2),w2))
p23=simple(diff(diff(p,w2),p1))
p24=simple(diff(diff(p,w2),p2))
p25=simple(diff(diff(p,w2),A1))
p26=simple(diff(diff(p,w2),A2))
p33=simple(diff(diff(p,p1),p1))
p34=simple(diff(diff(p,p1),p2))
p35=simple(diff(diff(p,p1),A1))
p36=simple(diff(diff(p,p1),A2))
p44=simple(diff(diff(p,p2),p2))
p45=simple(diff(diff(p,p2),A1))
p46=simple(diff(diff(p,p2),A2))
p55=simple(diff(diff(p,A1),A1))
p56=simple(diff(diff(p,A1),A2))
p66=simple(diff(diff(p,A2),A2))
```

Then we copy the resultant expressions to another MATLAB file for $\mathbf{I}(\mathbf{x})$ computation.

Here, we assign:

```
s(n)=A1*exp(j*w1*n)*exp(j*p1)+A2*exp(j*w2*n)*exp(j*p2);
```

and replace `s1` and `s2` by `s(n)` and `conj(s(n))`, respectively. We need to assign $\mathbf{x}$ for numerical computation, and without loss of generality, here all parameters are set to one.

Note also that the expected value of $r[n]$ is `s(n)`.

For detail, please refer to `comp_fim.m` or see below:

http://www.ee.cityu.edu.hk/~hcso/comp_fim.m

comp_fim.m

```
%Finding FIM for two cisoids
%As an illustration:
%A1=A2=w1=w2=p1=p2=1
%Observation length is N=100
%Standard derivation of zero-mean white noise power is sigma=1

N=100; % data length
sigma=1; % noise standard deviation
fim=zeros(6,6); %initialize FIM
A1=1;
A2=1;
w1=1;
w2=2;
p1=1;
p2=2;
for n=1:N

s(n)=A1*exp(j*w1*n)*exp(j*p1)+A2*exp(j*w2*n)*exp(j*p2);

fim(1,1)=fim(1,1) + A1*n^2*(-2*A2*cos(w1*n+p1-w2*n-p2)+exp(i*(w1*n+p1))*conj(s(n))+exp(-i*(w1*n+p1))*s(n));
fim(1,2)=fim(1,2) +2*A1*n^2*A2*cos(w1*n+p1-w2*n-p2);
fim(1,3)=fim(1,3) +A1*n*(-2*A2*cos(w1*n+p1-w2*n-p2)+exp(i*(w1*n+p1))*conj(s(n))+exp(-i*(w1*n+p1))*s(n));
fim(1,4)=fim(1,4) +2*A1*n*A2*cos(w1*n+p1-w2*n-p2);
fim(1,5)=fim(1,5)-n*(2*A2*sin(w1*n+p1-w2*n-p2)+i*conj(s(n))*exp(i*(w1*n+p1))-i*s(n)*exp(-i*(w1*n+p1)));
fim(1,6)=fim(1,6)-2*n*A1*sin(w1*n+p1-w2*n-p2);

fim(2,2)=fim(2,2) +A2*n^2*(-2*A1*cos(w1*n+p1-w2*n-p2)+exp(i*(w2*n+p2))*conj(s(n))+exp(-i*(w2*n+p2))*s(n));
fim(2,3)=fim(2,3)+2*A1*n*A2*cos(w1*n+p1-w2*n-p2);
fim(2,4)=fim(2,4)+A2*n*(-2*A1*cos(w1*n+p1-w2*n-p2)+exp(i*(w2*n+p2))*conj(s(n))+exp(-i*(w2*n+p2))*s(n));
fim(2,5)=fim(2,5)+2*A2*n*sin(w1*n+p1-w2*n-p2);
fim(2,6)=fim(2,6)-n*(-2*A1*sin(w1*n+p1-w2*n-p2)+i*conj(s(n))*exp(i*(w2*n+p2))-i*s(n)*exp(-i*(w2*n+p2)));

fim(3,3)=fim(3,3)+A1*(-2*A2*cos(w1*n+p1-w2*n-p2)+exp(i*(w1*n+p1))*conj(s(n))+exp(-i*(w1*n+p1))*s(n));
fim(3,4)=fim(3,4)+2*A1*A2*cos(w1*n+p1-w2*n-p2);
fim(3,5)=fim(3,5)-2*A2*sin(w1*n+p1-w2*n-p2)-i*conj(s(n))*exp(i*(w1*n+p1))+i*s(n)*exp(-i*(w1*n+p1));
fim(3,6)=fim(3,6)-2*A1*sin(w1*n+p1-w2*n-p2);
```

```
fim(4,4)=fim(4,4)+A2*(-2*A1*cos(w1*n+p1-w2*n-p2)+exp(i*(w2*n+p2))*conj(s(n))+exp(-i*(w2*n+p2))*s(n));
fim(4,5)=fim(4,5)+2*A2*sin(w1*n+p1-w2*n-p2);
fim(4,6)=fim(4,6)+2*A1*sin(w1*n+p1-w2*n-p2)-i*conj(s(n))*exp(i*(w2*n+p2))+i*s(n)*exp(-i*(w2*n+p2));

fim(5,5)=fim(5,5)+2;
fim(5,6)=fim(5,6)+2*cos(w1*n+p1-w2*n-p2);

fim(6,6)=fim(6,6)+2;

end

fim(2,1)=fim(1,2);

fim(3,1)=fim(1,3);
fim(3,2)=fim(2,3);

fim(4,1)=fim(1,4);
fim(4,2)=fim(2,4);
fim(4,3)=fim(3,4);

fim(5,1)=fim(1,5);
fim(5,2)=fim(2,5);
fim(5,3)=fim(3,5);
fim(5,4)=fim(4,5);

fim(6,1)=fim(1,6);
fim(6,2)=fim(2,6);
fim(6,3)=fim(3,6);
fim(6,4)=fim(4,6);
fim(6,5)=fim(5,6);

fim=fim./sigma^2;
```