The submission includes the script, **main.m**, and the function file, **myCrossCorr.m**.

The plot is shown in Fig 1, and the computed location of the first snare in samples from the start of the drum loop is 0.5000 s, as shown in Fig 2.

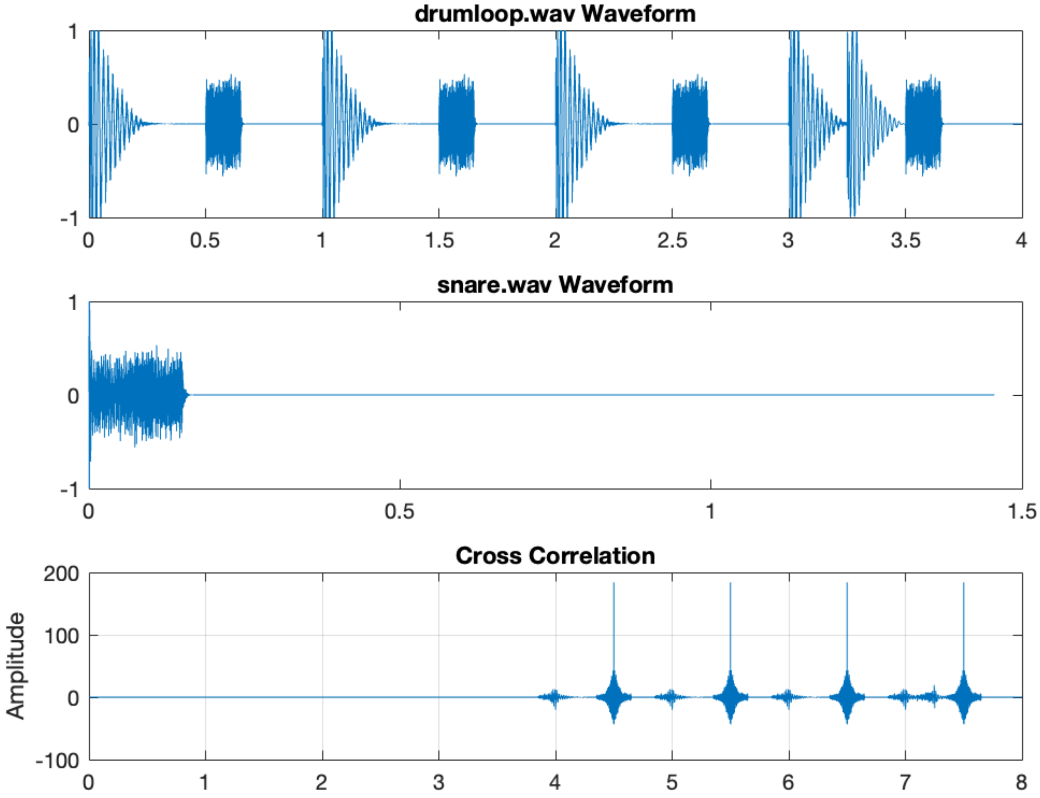
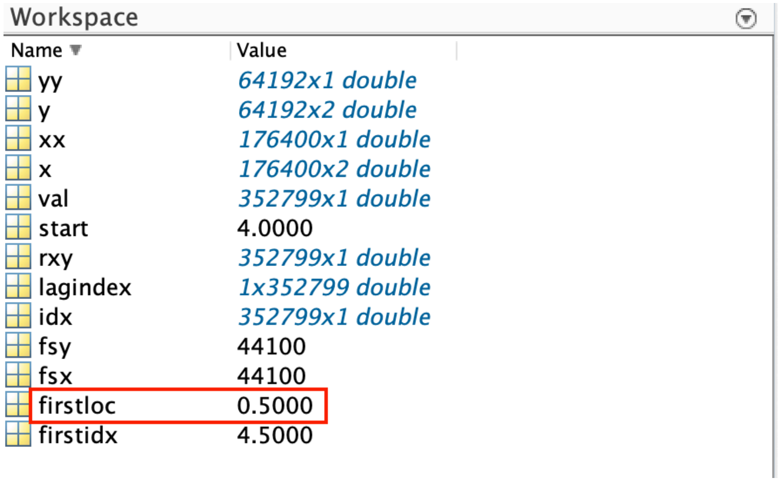


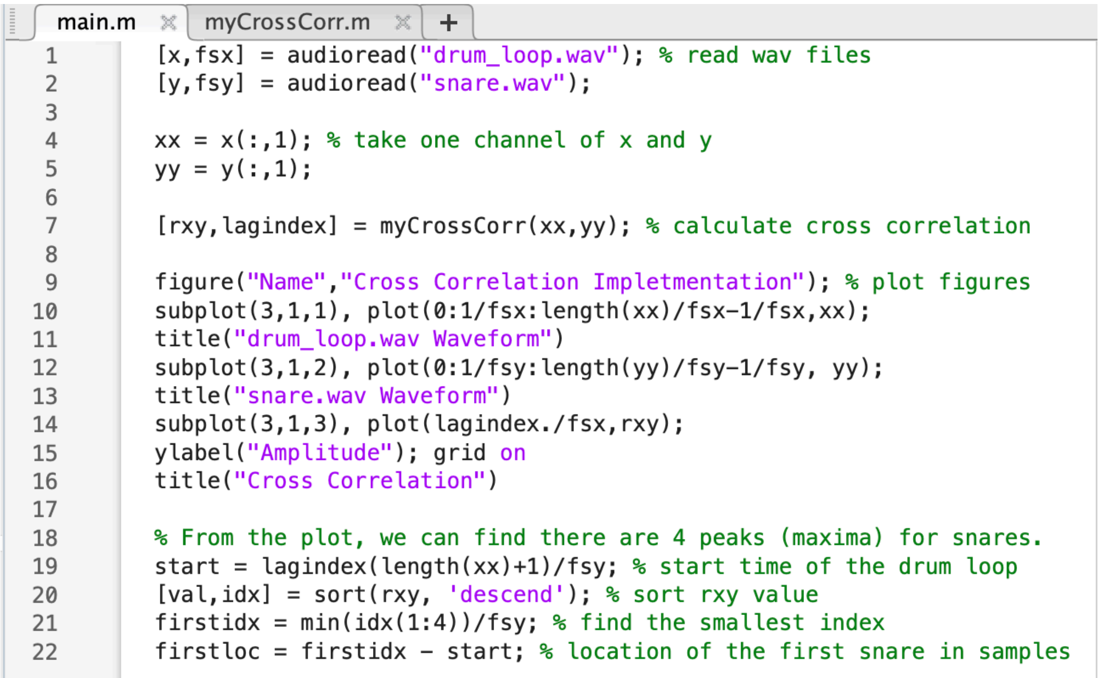
Fig 1: Cross-correlation plot



```
Name ▼          Value
yy              64192x1 double
y               64192x2 double
xx              176400x1 double
x               176400x2 double
val             352799x1 double
start           4.0000
rxy             352799x1 double
lagindex        1x352799 double
idx             352799x1 double
fsy             44100
fsx             44100
firstloc        0.5000
firstidx        4.5000
```

Fig 2: Location of the first snare

Codes will take approximately 55 seconds to run.



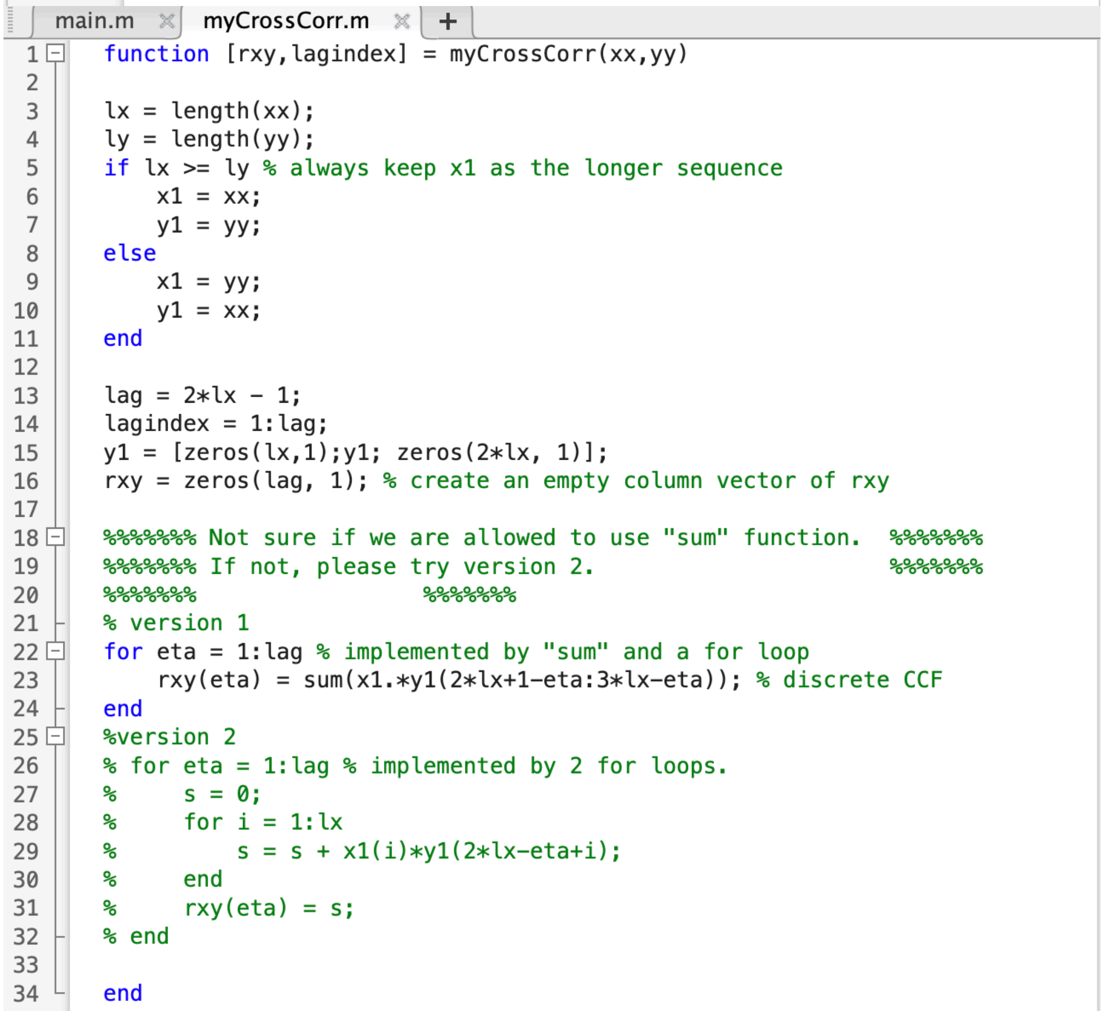
```
[x,fsx] = audioread("drum_loop.wav"); % read wav files
[y,fsy] = audioread("snare.wav");

xx = x(:,1); % take one channel of x and y
yy = y(:,1);

[rxy,lagindex] = myCrossCorr(xx,yy); % calculate cross correlation

figure("Name","Cross Correlation Impletmentation"); % plot figures
subplot(3,1,1), plot(0:1/fsx:length(xx)/fsx-1/fsx,xx);
title("drum_loop.wav Waveform")
subplot(3,1,2), plot(0:1/fsy:length(yy)/fsy-1/fsy, yy);
title("snare.wav Waveform")
subplot(3,1,3), plot(lagindex./fsx,rxy);
ylabel("Amplitude"); grid on
title("Cross Correlation")

% From the plot, we can find there are 4 peaks (maxima) for snares.
start = lagindex(length(xx)+1)/fsy; % start time of the drum loop
[val,idx] = sort(rxy, 'descend'); % sort rxy value
firstidx = min(idx(1:4))/fsy; % find the smallest index
firstloc = firstidx - start; % location of the first snare in samples
```



```
function [rxy,lagindex] = myCrossCorr(xx,yy)

lx = length(xx);
ly = length(yy);
if lx >= ly % always keep x1 as the longer sequence
    x1 = xx;
    y1 = yy;
else
    x1 = yy;
    y1 = xx;
end

lag = 2*lx - 1;
lagindex = 1:lag;
y1 = [zeros(lx,1);y1; zeros(2*lx, 1)];
rxy = zeros(lag, 1); % create an empty column vector of rxy

%%%%%%% Not sure if we are allowed to use "sum" function.  %%%%%%%
%%%%%%% If not, please try version 2.                     %%%%%%%
%%%%%%%                 %%%%%%%
% version 1
for eta = 1:lag % implemented by "sum" and a for loop
    rxy(eta) = sum(x1.*y1(2*lx+1-eta:3*lx-eta)); % discrete CCF
end
%version 2
% for eta = 1:lag % implemented by 2 for loops.
%     s = 0;
%     for i = 1:lx
%         s = s + x1(i)*y1(2*lx-eta+i);
%     end
%     rxy(eta) = s;
% end

end
```