ELE 851
Estimation and Detection Theory

Final Project

Analysis on the Detection of Sinusoidal Signals with Unknown Parameters

Yanjun Yan
yayan@syr.edu

12/5/2003
Preface
The work in this project is motivated by the analysis in 7.6 (from page 261) and Example 5.5 (from page 155) in the textbook volume II and also Example 11.1 in Volume I (from page 347). These analyses are closely related in the sense that they are dealing with quite similar signals, but they also differ in many details thus providing a broad platform for the reader to understand the sinusoidal detection scenario deeply and accurately. Computer simulations are carried out to verify their results and the performance of different detectors is compared and interpreted. Another important motivation for this work is that the sinusoidal detection is rather essential in practical applications, so the results in this project should be useful in real world applications.

I Introduction
Fourier Transform can help us decompose almost any signal into sinusoidal components, so the sinusoid detection is very important in practical applications, but sometimes the model of the sinusoidal signal or its parameters are just unknown to the detector designer, so there are different detectors and they’ll perform differently correspondingly. In 7.6, the sinusoidal signal model is assumed to be deterministic, but the amplitude, phase, frequency or the delay might be unknown. In example 5.5, the sinusoidal signal model is assumed to be random, the amplitude and phase are unknown, but there’s no delay. Then in example 11.1, the parameter vector is estimated, thus the detection test of the signal in example 5.5 or 7.6 can be regarded as the energy estimator based on the estimate of the parameter vector.

In order for fair comparison of the difference between deterministic and random model, same conditions are applied to both models such as unknown amplitude and phase, and no delay. Basically in dealing with random signals, we resort to the Bayesian philosophy. Even though the signal is random, the estimator can be interpreted as an estimate of the given realization of the signal. In contrast, in dealing with deterministic signals, we first try to find a UMP test, but if the UMP test doesn’t exist, we’ll either try to model the signal model as random or try to find a GLRT.

II Simulation for Example 5.5
In Rayleigh fading model, the amplitude and phase are assumed random, but the frequency is assumed known: $0 < f_0 < 1/2$ and there’s no delay.

During our observation interval, we deem the estimate of the parameter as a realization of the random variable, so this realization can be regarded invariant at least during this observation interval, that’s why we can use multiple data points altogether to improve the estimation performance. This assumption is understandable if we consider the slow fading channel. And even if the channel is indeed time selective fading, then we can shorten our observation interval correspondingly to be less than the coherence time, then during the observation interval the random variable can be still regarded as invariant. The worst case might be that there’s only one data point for one realization, but we can again increase the sampling rate to get more data points for one realization if we really want to get multiple data for one realization. While increasing the sampling rate, we also need to be careful about
the correlation between the noise data points.

Therefore in each Monte Carlo Trial with certain SNR ratio, only one set of realization of the amplitude and phase is generated and used.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{蒙特卡洛模拟结果.png}
\caption{Monte Carlo Simulation Results for Eq 5.5}
\end{figure}

MATLAB is not very efficient in dealing with memory, so I couldn’t use too many Monte Carlo Trials, thus the last line for the smallest false alarm rate is kind of wavy. Later I tried to use several groups of Monte Carlo Trials with the same structure and then average over the groups, the results are similar, but it took a much longer time.

**III Simulation for 7.6**

For comparison with Example 5.5, simulation is focused on the condition when the amplitude and phase are unknown. The test is the same as in 5.5, but the performance is different, that’s totally because the signal model is different. In this case, the signal is modeled as deterministic even though the parameter is unknown, thus the pdf is like an impulse, and the higher the signal energy, the more shift of the impulse from the zero value.

When false alarm rate is very low, the required Monte Carlo Trials are too many, so I had to skip those simulations to turn in this report relatively earlier therefore only three lines are shown below. But the relative relation is the same. And comparing this with the plot in the textbook, they are very close, so
if more time is given for other simulations corresponding to low false alarm rate, the curves are expected to have the same trend as in the textbook.

Comparing Figure 2 with Figure 1 in a single plot Figure 3 on next page, please note that Figure 2 considers SNR between 0 and 20 dB, and Figure 1 considers SNR between 0 and 30 dB, but in Figure 2 the detection rate has already reached 1 for large SNR, so in Figure 3, the steeper group is for deterministic model in 7.7.2, and the flatter group is for random model in example 5.5.

From comparison, it can be seen that the deterministic model has a better detection performance, which is reasonable since the pdf of the deterministic signal is concentrated on one point (impulse as we described before), and the random signal may have a much broader distribution hence even for the same mean and variance, the realized value of the signal could be very small, thus drag down the performance of the whole system.
IV Analysis

The Comparison between the Random and Deterministic Models shows that for the same average signal power, the detection performance for the deterministic model is better. However this result holds totally based on the TURE model of the real system. This is because the tests for both situations—formula (5.20) (on page 158) and formula (7.25) (on page 266) — are exactly the same, so from the detector designer’s point of view, there’s not much left we can do in designing the detector test to improve the detection ratio for a constant false alarm rate.

To appreciate this result in another way, we can say that the null hypothesis for both models have the same distribution, so for a given SNR and false alarm rate, we will get the same threshold, but because of the difference between the H1 hypothesis, the detection rate will surely go differently.

Again from the estimation point of view as in Example 11.1 in Volume I, we are estimating the signal’s energy based on the estimates of the signal amplitude and phase, and then decide whether the estimated signal’s energy has exceeded the threshold.

In practice, estimation is always the first step to do, even though sometimes in detection problems the estimation might be inexplicit. But the test is always none or less a transform of the estimate,
sometimes the estimate can be easily seen as the test is a linear or affine transform of it, but sometimes
the test might be a quadratic or even other nonlinear transform of the estimate, thus the estimation step
is not easily discernable.

Above analysis is kind of motivated by Neyman Fisher Factorization Theorem, the Sufficiency
Analysis and by comparing the tests in textbook Volume II and those estimators in textbook Volume I.
I think the idea is intuitive as well.

V Conclusion

This project has verified some results in the textbook Volume I example 11.1, Volume II example 5.5
and chapter 7.6 by Monte Carlo Simulations. And I have analyzed the results both visually and
theoretically. From this project, I’ve learned a lot on the intrinsic relation between the signal
distribution, estimation and test design.
clear all; close all;
M = 50000;   % Monte Carlo Trial Number
repM=1;
N = 16;     % Signal length
N_n0=32;    % Total observation length for delay present
Pfa = [10.^(1:-1:-3)];
var = 1;
SNRdB = [0:1:20];
SNR = 10.^(SNRdB/10);
step=1;
for snr = SNR
    % GLRT test on page 261-272
    for i = 1:repM
        [threshold(step,:,i),Pd(step,:,i)] = G1(N,M,snr,Pfa,step,i);
    end
    step=step+1;
end
Pd=mean(Pd,3);
figure
plot(SNRdB,Pd);
legend('P_F_A=10^-^1','P_F_A=10^-^2','P_F_A=10^-^3','P_F_A=10^-^4','P_F_A=10^-^5');
title('Monte Carlo Simulation Results for Eg 5.5')
xlabel('SNR (dB)')
ylabel('P_D')

step=1;
for snr = SNR
    % Bayesian NP test on page 155-161
    [threshold(step,:),Pd(step,:)] = BNP(N,M,snr,Pfa,step);
    Pd_theory(step,:) = Pfa.^(1/(1+snr/2));
    step=step+1;
end
figure
plot(SNRdB,Pd);
legend('P_F_A=10^-^1','P_F_A=10^-^2','P_F_A=10^-^3');
title('Monte Carlo Simulation Results for 7.6')
xlabel('SNR (dB)')
ylabel('P_D')
function [threshold,Pd] = BNP(N,M,snr,Pfa,step)

disp(['Step ',num2str(step),', has started.'])

f0 = 0.25; % Frequency
n0 = 0; % Delay
n = [1:N];
A = repmat(raylrnd(sqrt(pi/2),1,M),N,1); % amplitude (sigma_s^2=1)
phi = repmat(rand(1,M)*2*pi,N,1); % phase
w = randn(N,M)*sqrt(N/snr); % Noise to maintain the SNR as N*sigma_s^2/sigma^2
s = A.*cos(phi).*cos(2*pi*f0*repmat(n,1,M))-A.*sin(phi).*sin(2*pi*f0*repmat(n,1,M));
x0 = w;
x1 = s+w;

% T(1,:) is the test when signal is present
% T(2,:) is the test when signal is not present
T = [abs(sum(x1.*repmat(exp(-j*2*pi*f0*n),1,M))).^2/N;...
     abs(sum(x0.*repmat(exp(-j*2*pi*f0*n),1,M))).^2/N;];
disp(['Step ',num2str(step),', is in process....'])

for j = 1:length(Pfa)
    threshold(j) = eval(strcat('seek_threshold(Pfa(',num2str(j),'),T(2,:))'));
    Pd(j) = eval(['length(find(T(1,:)>threshold(',num2str(j),')))'/M'])
end
disp(['Step ',num2str(step),', has finished.'])
function [threshold, Pd] = G1(N, M, snr, Pfa, step, i)

    if i == 1
        disp(['Step ', num2str(step), ', has started.'])
    end

    % No delay
    % H0
    x0 = randn(N, M);
    % H1
    A = sqrt(snr * 2 / N); % Amplitude

    f0 = 0.25; % Frequency
    phi = pi / 4; % Phase
    n0 = 0; % Delay
    n = [1:N]';
    x1 = x0 + repmat(A * cos(2 * pi * f0 * n + phi), 1, M);

    % Amplitude Unknown
    A_hat1 = 2 / N * sum(x1 .* repmat(cos(2 * pi * f0 * n + phi), 1, M), 1);
    A_hat0 = 2 / N * sum(x0 .* repmat(cos(2 * pi * f0 * n + phi), 1, M), 1);

    % T(1,:) is the test when signal is present
    % T(2,:) is the test when signal is not present
    T = [A_hat1.^2; A_hat0.^2];

    if i == 1
        disp(['Step ', num2str(step), ', is in process....'])
    end

    for j = 1:length(Pfa)
        threshold(j) = eval(strcat('seek_threshold(Pfa(', num2str(j), '), T(2,:))'));
        Pd(j) = eval(['length(find(T(1,:) > threshold(', num2str(j), '),')) / M']);
    end

    if i == 1
        disp(['Step ', num2str(step), ', has finished.'])
    end
function threshold=seek_threshold(Pfa,T)

    M=length(T);
    ngam=100;
    gammamin=min(T);
    gammamax=max(T);
    gamdel=(gammamax-gammamin)/ngam;
    gamma=[gammamin:gamdel:gammamax]';
    P=zeros(length(gamma),1);
    for i=1:length(gamma)
        clear Mgam;
        Mgam=find(T>gamma(i));
        P(i)=length(Mgam)/M;
    end

    I=find(P<Pfa);
    threshold=gamma(I(1));