TTT4275 Detection Project Report

Hanna Lille
Kristin Herø

April 27, 2021

TTT4275 Estimation, Detection and Classification
NTNU

Summary

In this project, a Neyman-Pearson detector was designed in order to determine spectrum availability. The detector determines if there are active primary users in a cognitive radio system. If not, it is safe for secondary users to use the spectrum.

The statistical properties and distributions of the signals were used in order to compute the detector. It was shown that the detector could be accurately modeled with a Gamma distribution. In addition, an approximation of the detector with Gaussian distribution was implemented, which let us compute its complexity more easily.

Lastly, the detector was applied to a given data set to determine if primary users were present or not. The probability of detection was estimated to be ≈ 1, so we are fairly confident with the performance of the detector.

Contents

1 Introduction

This project deals with spectrum sensing in OFDM based wireless communication systems. The goal is to design a Neyman-Pearson (NP) detector to determine the spectrum availability.

The motivation for choosing this task is the use of cognitive radio to exploit underutilized spectral resources in an attempt to match the constantly growing demands for high data rates. In wireless communication systems, spectrum is a scarce resource. Therefore, this project is of high relevance to society.

The theory needed to understand the results and implement the task is described in section 2. The task itself is described in section 3, while the implementation and discussion of the results are given in section 4. Finally, the conclusion is given in section 5.

2 Theory

This chapter is a presentation of the theory needed to understand the task.

2.1 Estimating the expected value of a Gaussian random variable

The Minimum Variance Unbiased estimator for the expected value of a Gaussian variable is:

$$\hat{\mu_x} = \frac{1}{N}\sum_{n=0}^{N-1} x(n) \tag{1}$$

.[1, p.10]

2.2 Estimating the variance of random variable

The variance of a collection of n equally likely values can be written as

$$Var(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_x)^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2 - \mu_x^2 \tag{2}$$

, where μ_x is the expected value.[2]

To estimate the variance, the expected value can be estimated by the mean, (1).

2.3 Gamma distribution

The mean of a gamma distributed variable is $k\theta$, and the variance is $k\theta^2$[3].

For gamma distributed variables, the cdf, cumulative distribution function, is defined as:

$$F(x; k, \theta) = \frac{\gamma(k, \frac{x}{\theta})}{\Gamma(k)}, \tag{3}$$

. [4]

The sum of gamma distributed random variables $X_i \sim \Gamma(k_i, \theta)$ results in a variable that is also gamma distributed, with $\sum_{k=0}^{K-1} X_i \sim \Gamma(\sum_{k=0}^{K-1} k_i, \theta)$ [5].

2.4 Chi-square and Gamma distribution

If $X \sim \chi^2(\nu)$ and $c > 0$, then $cX \sim \Gamma(k = \frac{\nu}{2}, \theta = 2c)$ [6]. In other words, a random variable that is a constant multiplied with a chi-squared distributed variable is Gamma distributed with the parameters presented here.

2.5 Neyman-Pearson Detector

The Neyman-Pearson Lemma shows that if it is desired to increase the probability of detection of an LRT, likelihood ratio test, one must also accept the consequence of an increased false-alarm rate [1, p.47]. The likelihood ratio test is on the form

$$L(x) = \frac{p_1(x)}{p_0(x)} \begin{cases} \geq & \lambda & \Rightarrow & H_1 \\ < & \lambda & \Rightarrow & H_0 \end{cases}, \tag{4}$$

and is what is used for the Neyman-Pearson detection scheme. "The Neyman-Pearson detection criterion is aimed to maximize the power under the constraint that the false-alarm rate be upper bounded by, say, α_0"[1, p.47]. To derive the NP detector, a cost function is defined as

$$J = (1 - \beta) + \lambda(\alpha - \alpha_0) \tag{5}$$

,

where β is the power of the detection scheme, probability of detection, and α is the probability of false alarm. Furthermore, it can be proved that an LRT, (4), minimizes (5) for any positive λ. To satisfy the constraint and maximize the power, choose $\alpha = \alpha_0$, and determine the threshold λ from solving the equation

$$\lambda = P_{FA}^{-1}(\alpha_0) \tag{6}$$

. [1, p.47]

2.6 The Central Limit Theorem

The central limit theorem states that "under certain general conditions, the distribution F(x) of x approaches a normal distribution with the same mean and variance: $F(x) \approx G(\frac{x-\eta}{\sigma})$, as n increases" [7, p.278]. This can be used to approximate random variables with other distributions as Gaussian random variables.

3 Tasks

This project deals with designing a Neyman-Pearson detector in order to determine spectrum availability in OFDM cognitive radios. In the cognitive radio system, secondary users (SU), non-paying customers, will detect when the spectrum can be used without interfering with primary users (PU), who are paying customers. This spectrum sensing can be formulated as a binary hypothesis testing problem:

$$H_0 : x(n) = w(n), n = 0, 1, ..., N-1$$
$$H_0 : x(n) = s(n) + w(n), n = 0, 1, ..., N-1$$

, where s(n) is the signal sequence and w(n) is white Gaussian noise. Orthogonal frequency-division multiplexing (OFDM) is used to encode digital information and construct s(n), where each information symbol $S(k)$, for $k = 0, 1, ..., N-1$, is allocated to one out of N carrier frequencies. The time-domain signal is given by the inverse discrete Fourier transform of $S(k)$ allocated to a frequency. Therefore, $s(n)$ can be obtained by:

$$s(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} S(k) e^{\frac{j2\pi nk}{N}}, n = 0, 1, ..., N-1 \tag{7}$$

This project will design models to be used for spectrum sensing and performance guarantees, so that an SU can detect when PUs are idle and hence not interfere on the spectrum.

3.1 Task 1: Model Building

To develop a suitable detector, we need a model that explains how the observed data is generated. For this, we need the PDFs accociated with each hypothesis, $p_0(x)$ and $p_1(x)$. For this task, we will verify that the PDF of a single sample of the signal sequence $s(n)$ can be modeled as a complex normal variable. To do this, we plot histograms of $s_R(n)$ and $s_I(n)$ and compute estimates of $E\{s_R(n)s_I(n)\}$ and $E\{s_R(n)\} + jE\{s_I(n)\}$. Two sets of data for the PU data symbols are provided: one with samples from a standard Gaussian distribution and one from binary phase shift keying with $P(S(k) = 1) = P(S(k) = -1) = 0.5$

3.2 Task 2: One-Sample-Detector

In this task, we derive the Neyman-Pearson detector when only a single sample is observed, $N = 1$. We assume that $w(0)$ is a zero-mean complex Gaussian random variable with variance σ_w and that σ_w and the variance of signal $s(n)$, σ_s, is known.

3.3 Task 3: Performance of the One-Sample-Detector

For this task, we verify that

$$\frac{2|x(0)|^2}{\sigma_w^2} \tag{8}$$

under H_0, and

$$\frac{2|x(0)|^2}{\sigma_w^2 + \sigma_s^2} \tag{9}$$

under H_1, can be modeled as chi-square random variables with two degrees of freedom. For this, we plot the histograms of (8) and (9) and compare to the true PDF of a chi-square random variable with two degrees of freedom. Then we compute the probability of detection, P_D, and the probability of false alarm, P_{FA}.

3.4 Task 4: NP Detector With Data Set of K Samples

In this task, we derive the Neyman-Pearson detector for a data set of K samples. Then, we compute the associated threshold, λ', that maximizes the probability of detection, P_D, while ensuring that the probability of false alarm, P_{FA}, is less than a given limit, α_0.

3.5 Task 5: Performance of a General NP Detector

In this task, the distribution of the test statistic under the different hypotheses is computed. The task then asks to plot the receiver operating characteristics. The probability of false alarm is plotted against the probability of detection for different amount of samples to show how the performance of the detector varies.

3.6 Task 6: Approximate Performance of a General NP Detector

The task is to approximate the test statistic as a Gaussian random variable by using the central limit theorem. The probability density function of the approximated variable is then computed. Finally, the probability of detection and false alarm is plotted as a function of the threshold, as well as compared to the exact probabilities.

3.7 Task 7: Complexity of Detector

This task asks to compute the number of samples needed to obtain the given probabilities of detection and false alarm. This should be done using the approximated detector from the previous task.

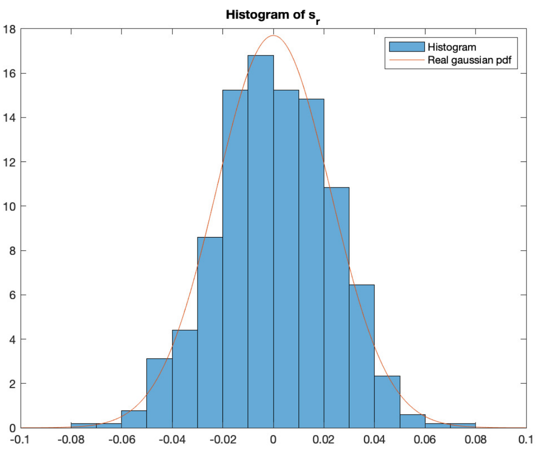
3.8 Task 8: Numerical Experiments in PU Detection

In this task, we are given a data set containing 100 realizations of signals observed at the SU. The given parameters are N=256, $\sigma_w^2 = 1$ and $\sigma_s^2 = 5$. Using the given data set and parameters, the tasks is to apply the NP detector to decide whether a PU is present or not for two different probabilities of false alarm: $P_{FA} = 0.1$ and $P_{FA} = 0.01$.

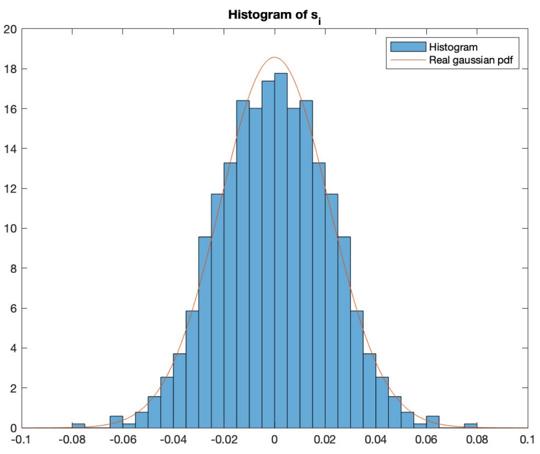
4 Implementation and Results

This chapter contains details of how we solved the problem and the results we obtained, as well as a discussion of the results. Matlab code is appended in section 6.

4.1 Task 1: Model Building

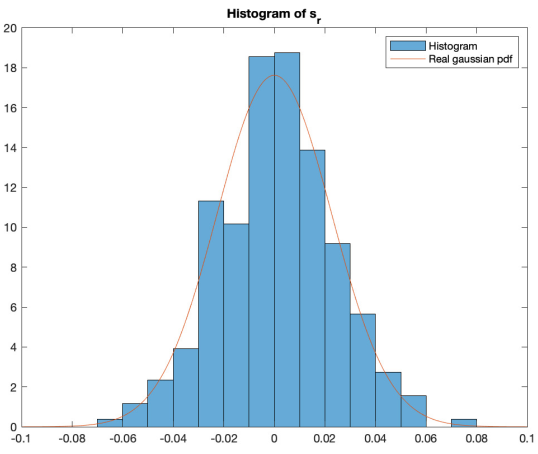


(a) Histogram of s_r

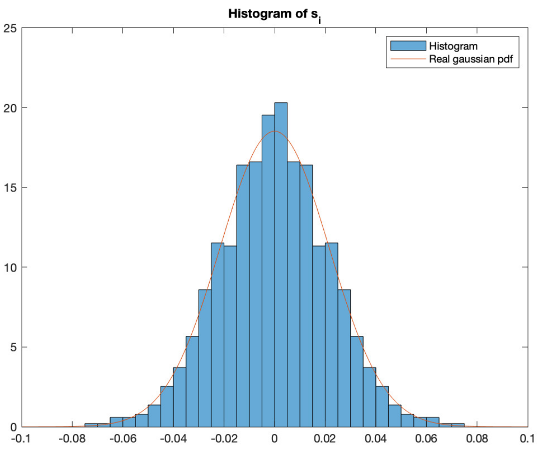


(b) Histogram of s_i

Figure 1: Histograms of the standard normal Gaussian distributed samples



(a) Histogram of s_r



(b) Histogram of s_i

Figure 2: Histograms of the binary phase shift keying (BPSK) symbols

Histograms of s_r and s_i from the standard normal Gaussian distributed samples and from the binary phase shift keying are shown in Figure 1 and Figure 2, respectively. It is clear from the Gaussian pdf plotted in the histograms, that the signals are accurately modeled as i.i.d. Gaussian distributed random variables. From this, we can conclude

Table 1: Expectation values

Expectation value	Gaussian	BPSK
$E\{s_r(n)s_i(n)\}$	-9.9923e-21	7.9409e-21
$E\{s_r(n)\} + jE\{s_i(n)\}$	5.2507e-4 - j2.9138e-19	9.7656e-04 + j1.4230e-19

that $s(n) = s_R(n) + js_I(n)$ can be accurately modeled as a complex normal variable with probability density function given by

$$p(s) = p(s_R)p(s_I) = \frac{1}{\pi\sigma_s^2}e^{\frac{1}{\sigma_s^2}|s|^2} \tag{10}$$

,

where $s_R(n) \sim N(0, \frac{\sigma^2}{2})$ and $s_I(n) \sim N(0, \frac{\sigma^2}{2})$.

Furthermore, estimates of the expectation value of $s(n)$, μ_s, are shown in Table 1. A suitable estimator for the expectation value is the mean, as discussed in subsection 2.1:

$$\hat{\mu_x} = \bar{x} = \sum_{i=0}^{N-1} x_i. \tag{11}$$

It is clear from the results that the expected value of $s(n)$ is ≈ 0. The Matlab code for this task is shown in Listing 1.

4.2 Task 2: One-sample-detector

Both the signal sequence and the noise are zero-mean complex Gaussian random variable with probability density functions given by (10) and

$$p(w) = \frac{1}{\pi\sigma_w^2}e^{\frac{1}{\sigma_w^2}|w|^2} \tag{12}$$

, respectively. Furthermore, the PDF of $x = x(0)$ under H_1 is given by

$$p_1(x) = \frac{1}{\pi(\sigma_w^2 + \sigma_s^2)}e^{-\frac{1}{\sigma_w^2+\sigma_s^2}|x(0)|^2}, \tag{13}$$

and the PDF under H_0 is given by

$$p_1(x) = \frac{1}{\pi\sigma_w^2}e^{-\frac{1}{\sigma_w^2}|x(0)|^2}. \tag{14}$$

This leads to the following likelihood ratio test:

$$L(x) = \frac{p_1(x)}{p_0(x)} = \frac{\frac{1}{\pi(\sigma_w^2+\sigma_s^2)} e^{-\frac{1}{\sigma_w^2+\sigma_s^2}|x(0)|^2}}{\frac{1}{\pi\sigma_w^2} e^{\frac{1}{\sigma_w^2}|x(0)|^2}} > \lambda \tag{15}$$

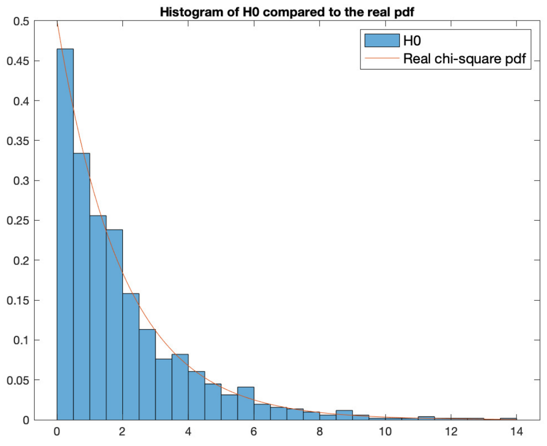
$$\ln L(x) = \ln(\frac{\sigma_w^2}{\sigma_s^2+\sigma_w^2}) + \frac{\sigma_s^2|x(0)|^2}{\sigma_w^2(\sigma_w^2+\sigma_s^2)} \geq \ln\lambda \tag{16}$$

From the log-likelihood ratio test, (16), we determine the following test statistic:

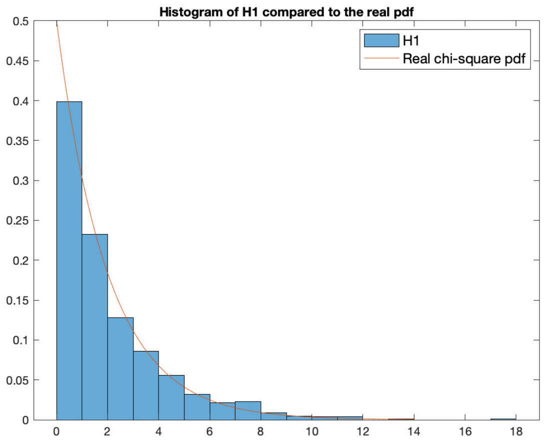
$$T(x) = |x(0)|^2 \geq \frac{\sigma_w^2(\sigma_w^2+\sigma_s^2)}{\sigma_s^2}(\ln\lambda - \ln(\frac{\sigma_w^2}{\sigma_s^2+\sigma_w^2})) = \lambda' \tag{17}$$

The NP detector given by (17) is referred to as the energy detector [1, p.56].

4.3 Task 3: Performance of the One-Sample-Detector



(a) Histogram of H0



(b) Histogram of H1

Figure 3: Histograms of the chi-squared random variables compared to the real pdf

Table 2: Variances

Variance	**Value**
σ_s^2	0.9917
σ_w^2	0.9957

Figure 3 shows the histograms of the variables given by (8) and (9), compared to the true pdf of a chi-suare random variable with two degrees of freedom. To compute the variables from the data $x(0)$, we use a suitable estimator for the variance, σ_w and σ_s, as discussed in subsection 2.2:

$$\hat{\sigma^2} = \frac{1}{N}\sum_{i=0}^{N-1} |x_i|^2 \tag{18}$$

.

From the histograms, it is clear that the variables can be accurately modeled by a chi-square distribution with two degrees of freedom. The test statistic, $T(x)$, under H_0 is given by

$$T(x) = |x(0)|^2 = \frac{\sigma_w^2}{2}\frac{2|x(0)|^2}{\sigma_w^2} \tag{19}$$

and can therefore be modeled as gamma distributed variable with shape $k = \frac{2}{2} = 1$ and scale $\theta = 2\frac{\sigma_w^2}{2} = \sigma_w^2$, i. e. $T(x) \sim \Gamma(k = 1, \theta = \sigma_w^2)$ under H_0, as explained in subsection 2.3. The same line of argument leads to $T(x)$ being gamma distributed with $k = 1$ and $\theta = \sigma_s^2 + \sigma_w^2$ under H_1.

This leads to the following expression for the probability of false alarm:

$$P_{FA} = Prob\{T(x) > \lambda'; H_0\} = 1 - F(\lambda'; 1, \sigma_w^2) = e^{\frac{-\lambda'}{\sigma_w^2}}$$

, where $F(x; k, \theta)$ is the cumulative distribution function of a gamma distributed variable, as introduced in subsection 2.3.

For the probability of detection, the expression is:

$$P_D = Prob\{T(x) > \lambda'; H_1\} = 1 - F(\lambda'; 1, \sigma_w^2 + \sigma_s^2) = e^{\frac{-\lambda'}{\sigma_w^2+\sigma_s^2}}$$

4.4 Task 4: NP Detector with Data Set of K Samples

The PDF of $\boldsymbol{x}$ under H_1 for K samples is given by

$$p_1(\boldsymbol{x}) = (\frac{1}{\pi(\sigma_w^2 + \sigma_s^2)})^K \exp\{-\sum_{k=0}^{K-1} \frac{1}{\sigma_w^2 + \sigma_s^2}|x(k)|^2\}$$

and for H_0

$$p_0(\boldsymbol{x}) = (\frac{1}{\pi(\sigma_w^2)})^K \exp\{-\sum_{k=0}^{K-1} \frac{1}{\sigma_w^2}|x(k)|^2\}$$

.

The likelihood ratio test becomes

$$L(x) = \frac{p_1(\boldsymbol{x})}{p_0(\boldsymbol{x})} = \frac{(\frac{1}{\pi(\sigma_w^2+\sigma_s^2)})^K \exp\{-\sum_{k=0}^{K-1} \frac{1}{\sigma_w^2+\sigma_s^2}|x(k)|^2\}}{(\frac{1}{\pi(\sigma_w^2)})^K \exp\{-\sum_{k=0}^{K-1} \frac{1}{\sigma_w^2}|x(k)|^2\}}$$

and from this we derive the test statistic in the same manner as for the one-sample detector:

$$T(x) = \sum_{k=0}^{K-1} |x(k)|^2 \geq \frac{\sigma_w^2(\sigma_w^2+\sigma_s^2)}{\sigma_s^2}(\ln\lambda - K(\frac{\sigma_w^2}{\sigma_s^2+\sigma_w^2})) = \lambda'$$

As explained in subsection 2.5, the threshold, λ', that maximizes the probability of detection while ensuring that the probability of false alarm is under a given limit, α_0, is given by

$$\lambda' = P_{FA}^{-1}(\alpha_0)$$

.

4.5 Task 5: Performance of a General NP Detector

It is known from Task 3 that $T(x) = |x(0)|^2 \sim \Gamma(1, \sigma^2)$. As stated in subsection 2.3, the sum of gamma distributed random variables is also gamma distributed, i.e. :

$$T(x) = \sum_{k=0}^{K-1} |x(k)|^2 \sim \Gamma(K, \sigma^2)$$

The test statistic under the different hypotheses will then have the following distributions:

$$T(x) \overset{H_0}{\sim} \Gamma(K, \sigma_w^2)$$

$$T(x) \overset{H_1}{\sim} \Gamma(K, \sigma_w^2 + \sigma_s^2)$$

The probabilities, P_D and P_{FA}, are computed using the Matlab function "gamcdf". The implementation of the function and the plotting is shown in Listing 3 in the appendix. The receiver operating characteristics for different amount of samples are plotted in Figure 4, using the estimated variances from task 3. A larger number of samples gave a very good P_D, since the probability rapidly becomes 1 for both K=100 and K=1000. This means that the more samples we have, the more likely is it that the detector will detect PUs, with few false alarms.

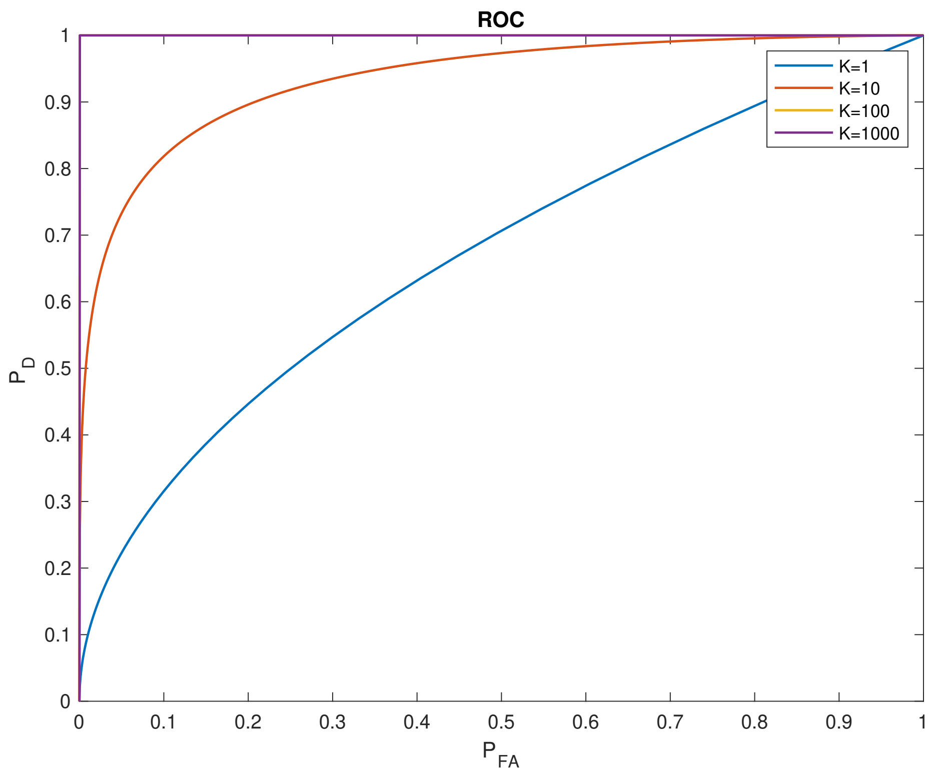


Figure 4: ROC for different K's

4.6 Task 6: Approximate Performance of a General NP Detector

From subsection 2.3, we know that the mean and variance of a gamma distributed variable is $k\theta$ and $k\theta^2$, respectively. Hence the mean and variance of $T(X)$ is given by:

$$\mu = K\sigma^2$$

,

$$\sigma^2 = K(\sigma^2)^2$$

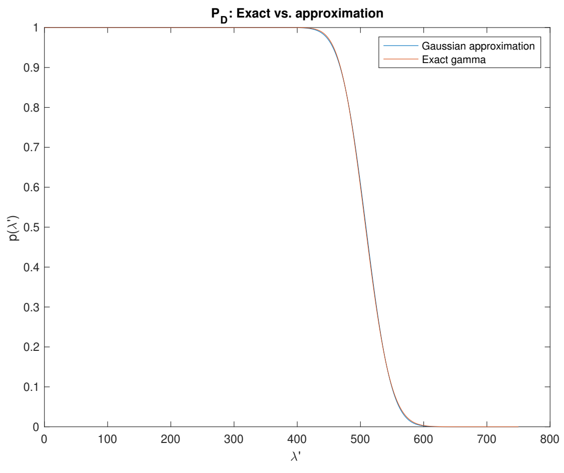
The test statistic T(x) can be approximated as a Gaussian random variable by using the central limit theorem described in subsection 2.6.

The approximated Gaussian variable under different hypotheses will therefore have these distributions:

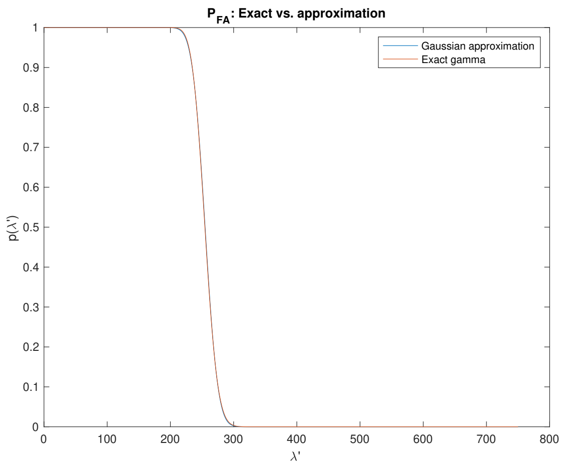
$$T(x) \overset{H_0}{\sim} N(K\sigma_w^2, K\sigma_w^4)$$

$$T(x) \overset{H_1}{\sim} N(K(\sigma_w^2 + \sigma_s^2), K(\sigma_w^2 + \sigma_s^2)^2)$$

The probabilities were computed using the Matlab function "normcdf". The implementation of this and the plotting of the probabilities against the threshold is shown in Listing 4 in the appendix. The comparisons of the approximated probabilities from the Gaussian variables and the exact probabilities are plotted in Figure 5. The approximations of both the probabilities are quite accurate, as there is little to no deviation from the exact probabilities in the plots. This proves that it is acceptable to approximate the performance of the detector with a Gaussian variable.



(a) P_D



(b) P_{FA}

Figure 5: Comparison of approximated and exact probabilities

4.7 Task 7: Complexity of Detector

$P_{FA} = \alpha$ =constant and $P_D = \beta$ = constant. The equations for β and α will be used to isolate K.

$$P_D = Prob\{T > \lambda' | H_1\} = Prob\{Z > \frac{\lambda - \mu_T | H_1}{\sigma_t | H_1}\} = Q(\frac{\lambda' - K(\sigma_w^2 + \sigma_s^2))}{\sqrt{K(\sigma_w^2 + \sigma_s^2)^2}}) = \beta, \quad (20)$$

where Z is a standard Gaussian normal distributed variable. Equation 20 can be rewritten:

$$\frac{\lambda' - K(\sigma_w^2 + \sigma_s^2))}{\sqrt{K(\sigma_w^2 + \sigma_s^2)^2}} = Q^{-1}(\beta) \quad (21)$$

An expression for λ' is computed by using the equation for α.

$$P_{FA} = Q(\frac{\lambda' - K\sigma_w^2}{\sqrt{K\sigma_w^2}}) = \alpha$$

$$\frac{\lambda' - K\sigma_w^2}{\sqrt{K\sigma_w^2}} = Q^{-1}(\alpha)$$

$$\lambda' = K\sigma_w^2 + Q^{-1}(\alpha)\sqrt{K}\sigma_w^2 \quad (22)$$

Inserting equation 22 into equation 21 yields:

$$\frac{K\sigma_w^2 + Q^{-1}(\alpha)\sqrt{K}\sigma_w^2 - K(\sigma_w^2 + \sigma_s^2))}{\sqrt{K(\sigma_w^2 + \sigma_s^2)^2}} = Q^{-1}(\beta)$$

Solving for K gives:

$$K = (\frac{Q^{-1}(\alpha)\sigma_w^2 - Q^{-1}(\beta)(\sigma_w^2 + \sigma_s^2)}{\sigma_s^2})^2 \tag{23}$$

Equation 23 is the expression to compute the number of samles, K, needed to attain a given α and β.

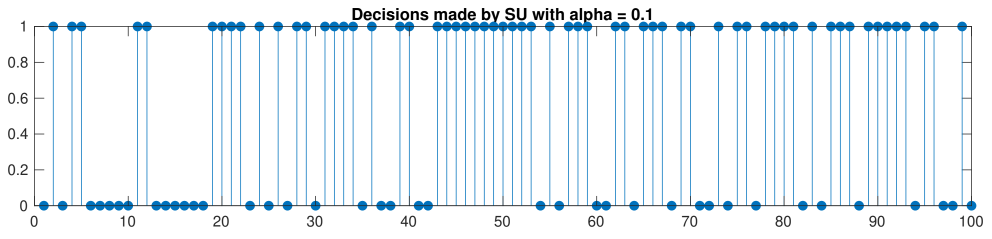
4.8 Task 8: Numerical Experiments in PU Detection

The NP detector was applied to a given dataset to decide whether PUs were present in 100 realizations. The Matlab code for implementing this is shown in Listing 5.

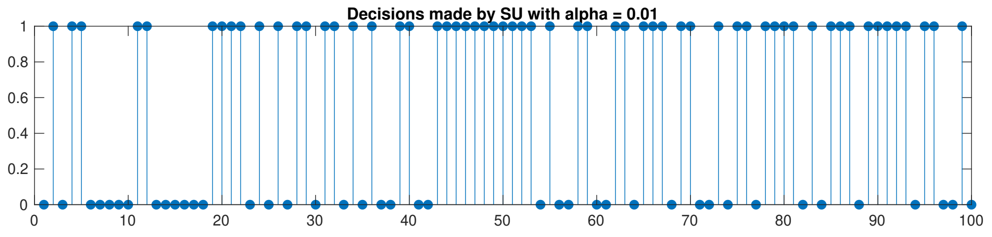
The results obtained after applying the detector are shown in Table 3 and Figure 6. We are quite confident that we detected all PUs, since we also computed the probability of detection, which was rounded up to 1 by Matlab. On the other hand, we expect some of the detected PUs to be false alarm, since the P_{FA} is larger than zero.

Table 3: NP Detector Results

P_{FA}	**PU present**	**PU absent**
0.1	61	39
0.01	59	41



(a) $P_{FA} = 0.1$



(b) $P_{FA} = 0.01$

Figure 6: Decisions made by the SU for different P_{FA}

5 Conclusion

In this project we used Matlab to design a Neyman-Pearson detector to determine spectrum availability. We learned that detection is a trade-off between wanting to increase the probability of detection while wanting to lower the probability of false alarm. We saw that even without any a-priori information, we can exploit statistical properties and known variances to make a detector with high probability of detection. Matlab has a large set of useful functions for this project. However, we still had to derive many functions on our own, which is not really necessary since these are well known functions that could have been implemented in a standard library. The project is based on several approximated PDFs. This works well for our purpose, but it leads to some inaccuracies that could cause problems on a larger scale. However, the simplifications make the problem more understandable and illustrate the basic techniques of detection. In conclusion, the project has taught us a great deal of detection theory through implementation and experimenting of the NP-detector.

References

[1] S. W. Tor A. Myrvoll and M. H. Johnsen, *Estimation, Detection and Classification Theory.*

[2] Wikipedia, “Variance,” [Online]. Available: `https://en.wikipedia.org/wiki/Variance`.

[3] WolframMathWorld, “Gamma distribution,” [Online]. Available: `https://mathworld.wolfram.com/GammaDistribution.html`.

[4] Wikipedia, “Gamma distribution,” [Online]. Available: `https://en.wikipedia.org/wiki/Gamma_distribution`.

[5] ——, “Relationships among probability distributions,” [Online]. Available: `https://en.wikipedia.org/wiki/Relationships_among_probability_distributions`.

[6] ——, “Chi-square distribution,” [Online]. Available: `https://en.wikipedia.org/wiki/Chi-square_distribution`.

[7] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill Higher Education, 2002.

6 Appendix

```
clear all;
% Gaussian data set
Sk = importdata('data/T1_data_Sk_Gaussian.mat');
s = ifft(Sk);
s_r = real(s);
s_i = imag(s);
N = length(Sk);
sigma_r_gaussian = std(s_r);
sigma_i_gaussian = std(s_i);
x = linspace(-0.1,0.1,1000);
pdf_r_gaussian = normpdf(x,0,sigma_r_gaussian);
pdf_i_gaussian = normpdf(x,0,sigma_i_gaussian);

% Plotting of histograms Gaussian
figure(1)
histogram(s_r, 'Normalization', 'pdf')
hold on;
plot(x,pdf_r_gaussian)
legend('Histogram', 'Real gaussian pdf')
title('Histogram of s_r')
figure(2)
histogram(s_i, 'Normalization', 'pdf')
hold on;
plot(x,pdf_i_gaussian)
legend('Histogram', 'Real gaussian pdf')
title('Histogram of s_i')

% BPSK data set
Sbk = importdata('data/T1_data_Sk_BPSK.mat');
sb = ifft(Sbk);
sbr = real(sb);
sbi = imag(sb);
sigma_r_bpsk = std(sbr);
sigma_i_bpsk = std(sbi)
pdf_r_bpsk = normpdf(x,0,sigma_r_bpsk);
pdf_i_bpsk = normpdf(x,0,sigma_i_bpsk);

%% Plotting of histograms BPSK
figure(3)
histogram(sbr, 'Normalization', 'pdf')
hold on;
plot(x,pdf_r_bpsk)
legend('Histogram','Real gaussian pdf')
title('Histogram of s_{r}')
figure(4)
histogram(sbi, 'Normalization', 'pdf')
hold on;
plot(x,pdf_i_bpsk)
legend('Histogram','Real gaussian pdf')
title('Histogram of s_{i}')

%% Expected values Gaussian
avg_r = 0;
avg_i = 0;
```

```matlab
avg_ri = 0;
for i = 1:N
    avg_r = avg_r + sr(i)/N;
    avg_i = avg_i + si(i)/N;
    avg_ri = avg_ri + sr(i)*si(i)/N;
end
exp_s = avg_r + 1i*avg_i;

%% Expected values BPSK
b_avg_r = 0;
b_avg_i = 0;
b_avg_ri = 0;
for i = 1:N
    b_avg_r = b_avg_r + sbr(i)/N;
    b_avg_i = b_avg_i + sbi(i)/N;
    b_avg_ri = b_avg_ri + sbr(i)*sbi(i)/N;
end
b_exp_s = b_avg_r + 1i*b_avg_i;
```

Listing 1: Matlab code task 1

```matlab
% Importing data
data_H0      = importdata('data/T3_data_x_H0.mat');
data_H1      = importdata('data/T3_data_x_H1.mat');
data_sigma_s = importdata('data/T3_data_sigma_s.mat');
data_sigma_w = importdata('data/T3_data_sigma_w.mat');

N = length(data_sigma_s);

% Calculating variances
sigma_s_sq = 0;
sigma_w_sq = 0;
for i = 1:N
    sigma_s_sq = sigma_s_sq + abs(data_sigma_s(i))^2/N;
    sigma_w_sq = sigma_w_sq + abs(data_sigma_w(i))^2/N;
end

chi_square_H0 = zeros(1,N);
chi_square_H1 = zeros(1,N);

% Computing variables
for i = 1:N
    chi_square_H0(i) = 2*abs(data_H0(i))^2/sigma_w_sq;
    chi_square_H1(i) = 2*abs(data_H1(i))^2/(sigma_s_sq + sigma_w_sq);
end

% Computing real pdf
x = linspace(0,14);
pdf_chi_sq = pdf('Chisquare',x, 2);

% Plotting histograms
figure(1)
h = histogram(chi_square_H0, 'Normalization', 'pdf'); hold on;
plot(x, pdf_chi_sq); hold off;
legend({'H0', 'Real chi-square pdf'}, 'FontSize', 12)
title('Histogram of H0 compared to the real pdf')

```

```matlab
figure(2)
histogram(chi_square_H1, 'Normalization', 'pdf'); hold on;
plot(x, pdf_chi_sq); hold off;
legend({'H1', 'Real chi-square pdf'}, 'FontSize', 12)
title('Histogram of H1 compared to the real pdf')
```

Listing 2: Matlab code task 3

```matlab
lambda = 0:0.1:3000;
K = [1,10,100,1000];

% Computing and plotting P_D and P_FA for different K
for i=1:length(K)
    P_FA = 1 - gamcdf(lambda, K(i), sigma_w_sq);
    P_D = 1 - gamcdf(lambda, K(i), sigma_w_sq + sigma_s_sq);
    plot(P_FA,P_D, 'LineWidth',1.2)
    hold on
    title('ROC')
end

legend('K=1','K=10','K=100','K=1000')
xlabel('P_{FA}'); ylabel('P_{D}')
```

Listing 3: Matlab code task 5

```matlab
lambda = 0:0.1:750;
K = 256;

% Computing approximated P_D and P_FA
P_FA_gaussian = 1 - normcdf(lambda, K*sigma_w_sq, sqrt(K*sigma_w_sq^2));
P_D_gaussian = 1 - normcdf(lambda, K*(sigma_w_sq+sigma_s_sq), sqrt(K*(
    sigma_s_sq+sigma_w_sq)^2));

% Computing exact P_D and P_FA
P_FA = 1 - gamcdf(lambda, K, sigma_w_sq);
P_D = 1 - gamcdf(lambda, K, sigma_w_sq + sigma_s_sq);

% Plot to compare
figure(1)
plot(lambda, P_FA_gaussian)
hold on;
plot(lambda, P_FA)
legend('Gaussian approximation', 'Exact gamma')
xlabel("\lambda'"); ylabel("p(\lambda')")
title('P_{FA}: Exact vs. approximation')

figure(2)
plot(lambda, P_D_gaussian)
hold on;
plot(lambda, P_D)
legend('Gaussian approximation', 'Exact gamma')
xlabel("\lambda'"); ylabel("p(\lambda')")
title('P_D: Exact vs. approximation')
```

Listing 4: Matlab code task 6

```matlab
data = importdata('data/T8_numerical_experiment.mat');
```

```
% Initializing parameters
K = 256;
realizations = 100;
sigma_w_sw = 1;
sigma_s_sq = 5;
alpha = [0.1;0.01];
q_inv = [1.2816;2.3263]; % Looked up in advance
lambda = (K*sigma_w_sq) + (q_inv * sqrt(K)*sigma_w_sq);
PU_present = zeros(2,1);
T = zeros(1,realizations);
decisions = zeros(2, realizations);

% Looping through realizations and checking if PUs present
for i=1:realizations
    T(i) = sum(abs(data(:,i)).^2);
    if T(i)>lambda(1)
        PU_present(1) = PU_present(1) + 1;
        decisions(1,i) = 1;
    end
    if T(i)>lambda(2)
        PU_present(2) = PU_present(2) + 1;
        decisions(2,i) = 1;
    end
end
PU_absent = realizations - PU_present;

% Computing P_D to establish confidence
beta = 1- normcdf(lambda, K*(sigma_w_sq+sigma_s_sq), sqrt(K*(sigma_s_sq+
    sigma_w_sq)^2));

% Plotting decisions
figure(1)
stem(decisions(1,:),'filled')
title('Decisions made by SU with alpha = 0.1')

figure(2)
stem(decisions(2,:),'filled')
title('Decisions made by SU with alpha = 0.01')
```
Listing 5: Matlab code task 8