Detection Project Report

Group 61
Student Kenny Hoang Nguyen 478374
Student Martin Lund Haug 766881

April, 2020

NTNU – Trondheim
Norwegian University of
Science and Technology

Abstract

In this project a detector for a binary hypothesis problem is derived. Its main task is to determine if there are primary users, i.e paying customers, in part of the spectrum in a cognitive radio system. If not, the secondary users can utilize the spectrum when it is not in use by paying customers, thus taking increasing use of the radio system. Using knowledge about the statistical distributions to the signals in the network which in addition is drowned in noise, a detector with desired performance properties is derived. The exact detector is derived for this specific problem using the known distribution and properties of the data.

Further, an approximate detector is calculated, which can be used to easily determine the decision rule, without needing to know all the details of the signals distribution. Another fine property of this approximate detector is that the amount of samples needed for the detector to perform well can be easily found because of its known properties.

Starting with the signals which is proved to be of a complex Gaussian distribution, a detector with the likelihood ratio test is derived. This detector uses a test statistic with another distribution which is the square of a complex Gaussian which belongs to χ^2 distribution which essentially is a special case of the Gamma distribution. And as it turns out the detector can be approximated as a normal Gaussian variable because it utilizes the central limit theorem. This turns out to be good approximate, and much easier to handle because of its many known properties. The results of this detector shows that it manages to correctly detect in 98% of the cases with as few as 9 samples for each time step.

Contents

1 Introduction

Detection problems occurs in many places in everyday life, in your computer you have electrical signals which the CPU has to detect as 1 or 0, or in other words a current is present or not. Another example is in air defense where the military has radars to detect if there is an hostile air action present or not. In this project we are going to look at a cognitive radio system with primary users (PU) and secondary users (SU), where the SUs are going to detect if there are PUs in the network so that they can decide if they can utilize the frequency spectrum without interfering with the quality of service (QoS) of the PUs.
This problem where the SUs use the frequency spectrum in an opportunistic manner whenever there are idle PUs is interesting since in wireless communication systems, spectrums are a scarce resource that service providers pay a substantial amount of money to the government in order to license the spectrum. This cost is covered by the customers in their monthly mobile subscription cost. Paying customers demands a certain QoS, which in the mentioned case is the PUs, and any interference appearing on the communication channel should be kept at a minimum to deliver the promised QoS.
This project starts by introducing the theoretical background necessary to understand and solve this problem in section 2. Then the tasks that needs to be solved for this problem is in section 3 followed by the implementation and results in section 4 then finally it is all wrapped up in 5.

2 Theory

2.1 Gaussian Distribution

The Gaussian distribution or the normal distribution is an important distribution that is often used in natural and social sciences for real-valued random variables when their distributions are not known [1]. The importance of this distribution comes from the central limit theorem, that states that for some conditions the sum of many (enough) observations of a random variable with finite mean and variance converges to a normal distribution even if the random variable comes from another distribution [2]. An important function here is the probability density function of a normal distribution with mean μ and variance σ^2 is

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \tag{1}$$

which gives the probability to obtain any value $x \in \mathbb{R}$ from this distribution. In this particular project we use the complex gaussian distribution which for our random variables has the probability density function

$$f(x) = \frac{1}{\pi\sigma^2} e^{-\frac{1}{\sigma^2}|x-\mu|^2} \tag{2}$$

Which in this case take in any value $x \in \mathbb{C}$.

2.2 Chi-squared distribution

In this project the χ^2-distribution is also an important distribution that is going to be used. The reason for this is because the χ^2-distribution is the resulting distribution from the sum of k independent squared standard normal variables. This resulting distribution will for these k normal variables have k degrees of freedom for the χ^2-distribution [3] Which will result in the point distribution function:

$$p(x;k) = \frac{x^{\frac{k}{2}-1} e^{-\frac{x}{2}}}{2^{\frac{k}{2}}\Gamma(\frac{k}{2})} \tag{3}$$

2.3 Gamma distribution

Furthermore the Gamma distribution is also a distribution that is going to be used here. It is mainly used because χ^2 is a special case of the Gamma distribution. The main property that is going to be used is if $Q \sim \chi^2(\nu)$ and c is a positive constant, then $cQ \sim Gamma(\nu/2, 2c)$ [4]. Resulting in the cumulative distribution function:

$$f(x;\nu/2,2c) = \frac{1}{(2c)^{\frac{\nu}{2}}} \int_0^x t^{\frac{\nu}{2}-1} e^{-\frac{t}{2c}} dt \tag{4}$$

where ν is the distributions degrees of freedom.

2.4 Estimators

The definition of expected value of discrete random variables is

$$\mu = \mathbb{E}\{X\} = \sum_{i=0}^{\infty} x_i p(x_i) \tag{5}$$

Which gives some nice properties for the estimators.
Furthermore the variance of discrete random variables is defined by

$$\sigma^2 = \mathrm{Var}\{X\} = \mathbb{E}\{X^2\} - \mathbb{E}\{X\}^2 \tag{6}$$

When we with probable cause can say something about the distribution that the random variables are sampled from, but not their mean and/or variance, we can estimate these distributions properties. Assumed that the random variables are sampled independent from the same identical distribution (iid), then the mean can be estimated with the average of the samples, which is unbiased.

$$\begin{aligned}
\hat{\mu} &= \mathbb{E}\{\frac{1}{N}\sum_{n=0}^{N-1} x\} \\
&= \frac{1}{N}\sum_{n=0}^{N-1} \mathbb{E}\{x\} \\
&= \frac{1}{N}\sum_{n=0}^{N-1} \mu \\
&= \mu
\end{aligned} \tag{7}$$

Meaning that the expected value of the mean of the samples will, with the number of samples taken, converge to the actual expected value of the distribution.
This is also the case if there are samples of the variance of the data available

$$\begin{aligned}
\hat{\sigma^2} &= \mathbb{E}\{\frac{1}{N}\sum_{n=0}^{N-1} \sigma^2\} \\
&= \frac{1}{N} N\sigma^2 \\
&= \sigma^2
\end{aligned} \tag{8}$$

These results are used when solving the problems later on in this report.

2.5 (Binary) Hypothesis Testing

Detection problems are often formulated to be about if a signal is present or not in conditions which masks the signal that we desire to detect, for

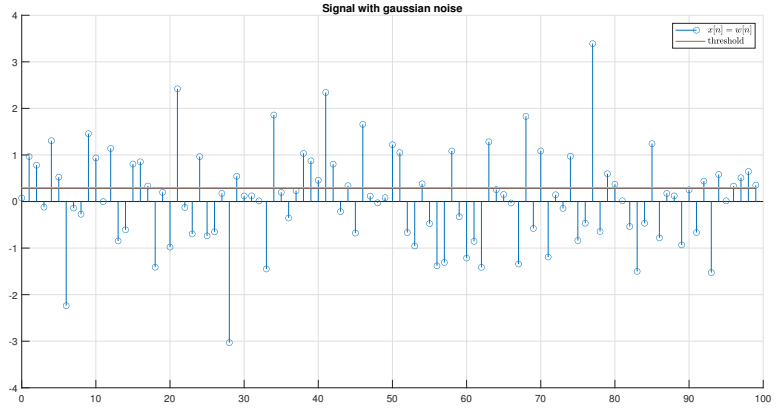


Figure 1: 100 samples from a gaussian distribution $\mathcal{N}(0,1)$

instance white gaussian noise. The hypothesises that could be formulated is:

$$\begin{aligned} \text{Null hypothesis} H_0 &: x[n] \sim \mathbb{P}_0 \\ \text{Alternative hypothesis} H_1 &: x[n] \sim \mathbb{P}_1 \end{aligned} \tag{9}$$

where the null hypothesis is "no signal present" and alternative hypothesis is "signal present", and $\mathbb{P}_0$ and $\mathbb{P}_1$ are two arbitrary distributions.
Let $[x[0]x[1]\dots x[N-1]]$ be sampled random variables from a sensor where it is constant "1" when detecting an object and constant "0" when not detecting anything. A simple detector in this case could be by setting a threshold at $\lambda = 0.3$ that detects when the threshold is broken. The problem with this simple detector appears when the signal from our sensor is noisy. Let now the samples from our sensor be

$$\begin{aligned} H_0 &: x[n] = w[n] \\ H_1 &: x[n] = A + w[n] \end{aligned}$$

where $w[n]$ $\mathcal{N}(0,1)$. Then under the null hypothesis in a non-noisy environment $w[n] = 0 \implies x[n] = 0$, however we have a noisy envirnoment so $x[n]$ obtains random values from the normal distribution with zero mean and unit variance. In figure 1 we can see how the data we obtain from 100 samples may look like. From these 100 samples there are 39 samples that are above the set threshold meaning that from these 100 samples we will get 39 false positives/alarms which will alert the user that there is an object detected when there really is not. Our goal when solving detection problems is to develop a detector that from the samples can find a decision rule so that the number of false alarms is minimized but at the same time manages to detect correctly when there is an object present [5].

2.6 Neyman-Pearson detector

The Neyman-Pearson detector utilizes the likelihood ratio test (LRT)

$$L(\mathbf{x}) \triangleq \frac{p_1(\mathbf{x})}{p_0(\mathbf{x})} \begin{cases} \geq \lambda \implies H_1 \\ < \lambda \implies H_0 \end{cases} \tag{10}$$

where $p_1(x)$ and $p_0(x)$ is the point distribution functions or the likelihood functions of the distributions under H_1 and H_0 respectively. With the Neyman-Pearson detector the threshold λ is chosen to satisfy the constraint and to maximize the power such that

$$P_{FA} = \alpha = \int_{L(\mathbf{x})>\lambda} p_0(\mathbf{x}) d\mathbf{x} \tag{11}$$

where α_0 is the tuning factor. This goes hand in hand with the detection rate since the Neyman-Pearson lemma states that to increase the power of the likelihood ratio test, the false alarm rate is also increased.

3 The tasks

The main problem that is going to be solved in this project is the detection problem

$$\begin{aligned} H_0 &: x(n) = w(n), n = 0, 1, \ldots, N-1 \\ H_1 &: x(n) = s(n) + w(n), n = 0, 1, \ldots, N-1 \end{aligned} \tag{12}$$

where $s(n)$ is a waveform sequence of the PU and $w(n)$ is an additive white complex Gaussian noise. To construct the sequence $s(n)$, that is transmitted over the wireless channel, the modulation method orthogonal frequency-division multiplexing (OFDM) is used. Each information symbol $S(k)$, $k = 0, 1, \ldots, N-1$ is allocated on each N carrier frequencies. The unique time-domain signal $s(n)$ corresponding to the sampled spectrum is obtained by inverse discrete Fourier transform. Thus is the PUs time-domain signal given by:

$$s(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} S(k) e^{\frac{j2\pi nk}{N}}, n = 0, 1, \ldots, N-1 \tag{13}$$

which we notice is a complex-valued quantity.

3.1 Task 1: Model building

Here the data $x[n]$ is generated, which can be used in our analysis to obtain a suitable detector. To generate $x[n]$ we need do to verify that the complex-valued time-domain OFDM signal sequence $s(n) = s_R(n) + js_I(n)$ is independent and identically distributed, in addition to verify that it is accurately modelled with a complex Gaussian distribution.
The data sets T1 are used, they contain samples that are taken from a standard normal Gaussian distribution and binary phase shift keying respectively.

3.2 Task 2: One-sample detector

In this task only a single sample is used to generate the NP-detector. That is, there needs to be done calculations to retrieve the decision rule that decides for when to choose null hypothesis over the alternative hypothesis and vice versa.

3.3 Task 3: Performance of the one-sample detector

Here the data sets T3 are given and are to be used to verify that

$$H_0 : \frac{2x_R^2(0)}{\sigma_w^2} + \frac{2x_I^2(0)}{\sigma_w^2} = \frac{2|x(0)|^2}{\sigma_w^2} \tag{14}$$

$$H_1 : \frac{2x_R^2(0)}{\sigma_w^2 + \sigma_s^2} + \frac{2x_I^2(0)}{\sigma_w^2 + \sigma_s^2} = \frac{2|x(0)|^2}{\sigma_w^2 + \sigma_s^2} \tag{15}$$

are χ^2-distributed with 2 degrees of freedom. This can be used to show that $\widetilde{x} = \frac{2|x(0)|^2}{\sigma^2}$ has the point-distribution function $f(\widetilde{x}) = \frac{1}{2}e^{-\frac{\widetilde{x}}{2}}$ for $\widetilde{x} > 0$. Further, the result can then be used to calculate the probability for false alarm and the probability for detection for this one-sample detector.

3.4 Task 4: General NP detector

In this task the one-sample detector derived in the previous tasks are generalized so it can be used when there are $K > 1$ samples. Since $|x(n)|^2$ is a sum of two standard normal Gaussian squared, then in the case where there are multiple samples, the χ^2-distribution has $2K$ degrees of freedom which can be used to derive the threshold λ' that maximizes P_D and ensure that $P_{FA} < \alpha$.

3.5 Task 5: Performance of the general NP detector

In this task the distribution obtained in task 4 is used to plot the receiver operating characteristics, to visualize how the performance of the Neyman-Pearson detector changes with the power of the test and the false alarm rate.

3.6 Task 6: Approximate performance of the general NP detector

Here the central limit theorem is used to approximate the test statistic. The PDF of the test statistic is calculated, and thus can P_D and P_{FA} be plotted as a function of the threshold where the Gaussian is used instead of the pdf from the gamma distribution.

3.7 Task 7: Complexity of the detector

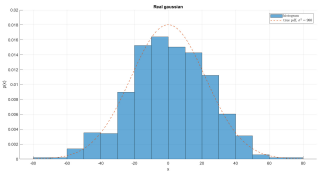
In the former task the detector was approximated to be from the Gaussian distribution. Using this knowledge, an expression for how many samples that are needed to attain a given P_D and P_{FA} is derived.

3.8 Task 8: Numerical experiments in PU detection

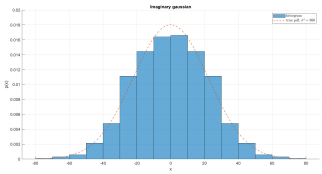
The NP detector that has been found is used on a numerical experimental data that contains 100 realizations of a PU signal. The task is to see if the detector will detect the PU.

4 Implementation and results

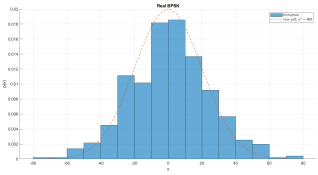
4.1 Task 1: Model development



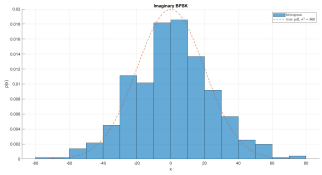
(a) The histogram of Re(S(k)) Gaussian



(b) The histogram of Im(S(k)) Gaussian



(c) The histogram of Re(S(k)) BPSK



(d) The histogram of Im(S(k)) BPSK

Figure 2: The histograms compared to pdf of Gaussian

In the calculations of the variance and expected value for the four distributions the results were:

Gaussian		BPSK	
$\mathbb{E}\{s[n]\}$	$\mathbb{E}\{s_r[n]s_i[n]\}$	$\mathbb{E}\{s[n]\}$	$\mathbb{E}\{s_r[n]s_i[n]\}$
$0.53767 - j2.0817 \cdot 10^{-16}$	$1.707 \cdot 10^{-15}$	$1 - j6.5919 \cdot 10^{-17}$	$2.4425 \cdot 10^{-15}$

Table 1: Shows the expected value for each distributions

This illustrates that even if we expect that the signal has expected value at 0 it can still be a little bit skewed. The histogram shows that the variance is so high that the expected value is approximately at 0 in comparison anyways. The code for this problem is appended in A.1

4.2 Task 2: One-sample-detector

In this task only one sample is used meaning that the detection problem is in this case:

$$H_0 : x[0] = w[0]$$
$$H_1 : x[0] = s[0] + w[0]$$

It is given from the problem that $w \sim \mathbb{C}\mathcal{N}(0, \sigma_w^2)$ and $s \sim \mathbb{C}\mathcal{N}(\mu_s, \sigma_s^2)$ which means that under H_0, x is from same distribution as w and thus a complex gaussian with same mean and variance as w. Under H_1, which is a

sum of two complex gaussian distribution, x also is from a complex gaussian distribution. However the mean and variance needs to be calculated to adjust for both distributions. The mean of $x[0]$ under H_1 is

$$\begin{aligned}\mathbb{E}\{x[0]\} &= \mathbb{E}\{s[0] + w[0]\} \\ &= \mathbb{E}\{s[0]\} + \mathbb{E}\{w[0]\} \\ &= \mu_s + 0 = \mu_s\end{aligned}$$

The variance of $x[0]$ under H_1 is

$$\begin{aligned}\mathrm{Var}\{x[0]\} &= \mathrm{Var}\{s[0] + w[0]\} \\ &= \mathrm{Var}\{s[0]\} + \mathrm{Var}\{w[0]\} \\ &= \sigma_s^2 + \sigma_w^2\end{aligned}$$

Meaning that

$$p(x; H_0) = p_0(x) = \frac{1}{\sigma_w^2 \pi} e^{-\frac{|x|^2}{\sigma_w^2}} \tag{16}$$

$$p(x; H_1) = p_1(x) = \frac{1}{(\sigma_w^2 + \sigma_s^2)\pi} e^{-\frac{|x-\mu_s|^2}{\sigma_s^2+\sigma_w^2}} \tag{17}$$

Setting up the LRT:

$$\begin{aligned}L(x) = \frac{p_1(x[0])}{p_0(x[0])} &= \frac{\frac{1}{(\sigma_w^2+\sigma_s^2)\pi} e^{-\frac{|x[0]-\mu_s|^2}{\sigma_s^2+\sigma_w^2}}}{\frac{1}{\sigma_w^2\pi} e^{-\frac{|x[0]|^2}{\sigma_w^2}}} \\ &= \frac{\sigma_w^2}{\sigma_w^2 + \sigma_s^2} e^{-\frac{1}{\sigma_w^2+\sigma_s^2}|x[0]-\mu_s|^2 + \frac{1}{\sigma_w^2}|x[0]|^2}\end{aligned}$$

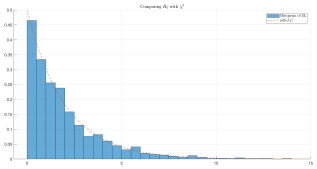
Since it is given that $\mu_s \approx 0$ this can be used to simplify the caluclations. The decision rule is to choose H_1 when $L(x[0]) \geq \lambda$, since $L(x[0])$ is a monotonically increasing function then the inequality holds for $\ln L(x[0]) \geq \ln \lambda$.

$$\begin{aligned}\ln L(x[0]) = \ln(\sigma_w^2) - \ln(\sigma_w^2 + \sigma_s^2) - \frac{1}{\sigma_w^2 + \sigma_s^2}|x[0]|^2 + \frac{1}{\sigma_w^2}|x[0]|^2 &\geq \ln \lambda \\ \left(\frac{1}{\sigma_w^2} + \frac{1}{\sigma_s^2 + \sigma_w^2}\right)|x[0]|^2 &\geq \ln \lambda - \ln \frac{\sigma_w^2}{\sigma_w^2 + \sigma_s^2} \\ |x[0]|^2 = x_R[0]^2 + x_I[0]^2 &\geq \frac{\sigma_w^2(\sigma_w^2 + \sigma_s^2)}{\sigma_s^2}\left(\ln \lambda - \ln \frac{\sigma_w^2}{\sigma_w^2 + \sigma_s^2}\right) = \lambda'\end{aligned}$$

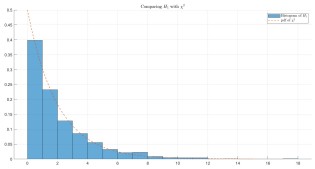
This also means that the decision rule is to choose H_1 when $|x[0]|^2 \geq \lambda'$ and choose H_0 when $|x[0]|^2 < \lambda'$ where

$$\lambda' = \frac{\sigma_w^2(\sigma_w^2 + \sigma_s^2)}{\sigma_s^2}\left(\ln \lambda - \ln \frac{\sigma_w^2}{\sigma_w^2 + \sigma_s^2}\right) \tag{18}$$

4.3 Task 3: Performance of the one-sample detector



(a) The histogram of (14)



(b) The histogram of (15)

The histogram in 3a and 3b shows that $|x[0]|^2$ is χ^2-distributed with 2 degrees of freedom both under H_0 and H_1 and the difference between H_0 and H_1 is essentially the constant scaling factor given by the variances σ_w^2 and σ_s^2. Calculating the probability for the false alarm gives

$$\begin{aligned} P_{FA} &= \text{Prob}\{|x[0]| \geq \lambda' \big| H_0\} \\ &= \int_{\lambda'}^{\infty} \frac{e^{-\frac{2x[0]}{2\sigma_w^2}}}{\sigma_w^2} dx[0] \\ &= -\frac{\sigma_w^2}{\sigma_w^2} e^{-\frac{x[0]}{\sigma_w^2}} \Big|_{\lambda'}^{\infty} \\ &= e^{-\frac{\lambda'}{\sigma_w^2}} \end{aligned}$$

Which only holds when $\lambda' > 0$.
Furthermore the probability for detection is

$$\begin{aligned} P_D &= \text{Prob}\{|x[0]| \geq \lambda' \big| H_1\} \\ &= \int_{\lambda'}^{\infty} \frac{e^{-\frac{x[0]}{\sigma_w^2+\sigma_s^2}}}{\sigma_w^2+\sigma_s^2} dx[0] \\ &= -\frac{\sigma_w^2+\sigma_s^2}{\sigma_w^2+\sigma_s^2} e^{-\frac{x[0]}{\sigma_w^2+\sigma_s^2}} \Big|_{\lambda'}^{\infty} \\ &= e^{-\frac{\lambda'}{\sigma_w^2+\sigma_s^2}} \end{aligned}$$

Which also only holds for $\lambda' > 0$. The code for this problem can be found in A.2

4.4 Task 4: Generalized for multiple samples

We take the one-sample-detector a step further and generalize it to use it for multiple samples. That is that the likelihood function is used to calculate

the LRT so that we get

$$\begin{aligned}\ln L(x) &= \ln \prod_{n=0}^{K-1} \frac{1}{(\sigma_w^2+\sigma_s^2)\pi} e^{-\frac{|x(n)|^2}{\sigma_s^2+\sigma_w^2}} - \ln \prod_{n=0}^{K-1} \frac{1}{\sigma_w^2\pi} e^{-\frac{|x(n)|^2}{\sigma_w^2}} \\ &= \sum_{n=0}^{K-1} \left(\ln \frac{\sigma_w^2}{\sigma_w^2+\sigma_s^2} + \frac{|x(n)|^2}{\sigma_w^2} - \frac{|x(n)|^2}{\sigma_w^2+\sigma_s^2} \right) \\ &= \sum_{n=0}^{K-1} \left(\ln \frac{\sigma_w^2}{\sigma_w^2+\sigma_s^2} + \frac{\sigma_s^2}{\sigma_w^2(\sigma_w^2+\sigma_s^2)} |x(n)|^2 \right) \geq \ln \lambda\end{aligned}$$

So the decision rule ends up being to choose H_1 when

$$T(\mathbf{x}) = \sum_{n=0}^{K-1} |x(n)|^2 \geq \frac{\sigma_w^2(\sigma_w^2+\sigma_s^2)}{\sigma_s^2} \left(\ln \lambda - K \ln \frac{\sigma_w^2}{\sigma_w^2+\sigma_s^2} \right) = \lambda' \tag{19}$$

Here $T(\mathbf{x})$ is a sum of K complex Gaussian squared, meaning the pdf in this case is a χ^2-distribution with $2K$ degrees of freedom. This means that

$$\begin{aligned}P_D &= \int_{\lambda'}^{\infty} \frac{x^{K-1} e^{-\frac{x}{\sigma_w^2+\sigma_s^2}}}{(\sigma_w^2+\sigma_s^2)^K \Gamma(K)} dx \\ &= \frac{1}{(\sigma_w^2+\sigma_s^2)^K \Gamma(K)} \int_{\lambda'}^{\infty} x^{K-1} e^{-\frac{x}{\sigma_w^2+\sigma_s^2}} dx \\ &= Q\left(\frac{\lambda'}{\sigma_w^2+\sigma_s^2} \right)\end{aligned} \tag{20}$$

Differentiating the integral with respect to λ' it shows that P_D is strictly decreasing and has extremal points at $\lambda' = 0 \vee \lambda' = \infty$. Which essentially means that P_D is maximized when $\lambda' \in [0, \infty)$ is as close to 0 as possible. For the false alarm with the upper limit α_0, its cumulative distribution function is

$$\begin{aligned}P_{FA} &= \frac{1}{(\sigma_w^2)^K \Gamma(K)} \int_{\lambda'}^{\infty} x^{K-1} e^{-\frac{x}{\sigma_w^2}} dx \\ &= Q\left(\frac{\lambda'}{\sigma_w^2} \right) \leq \alpha_0\end{aligned} \tag{21}$$

Which means that for the inequality to hold in addition to maximize the power of the test, need to choose $\lambda' >= \sigma_w^2 Q^{-1}(\alpha_0)$. Essentially choosing $\lambda' = \sigma_w^2 Q^{-1}(\alpha_0)$ to minimize P_D. Inserting this to get the expression for P_D.

$$P_D = Q(\frac{\sigma_w^2}{\sigma_w^2+\sigma_s^2} Q^{-1}(\alpha_0)) \tag{22}$$

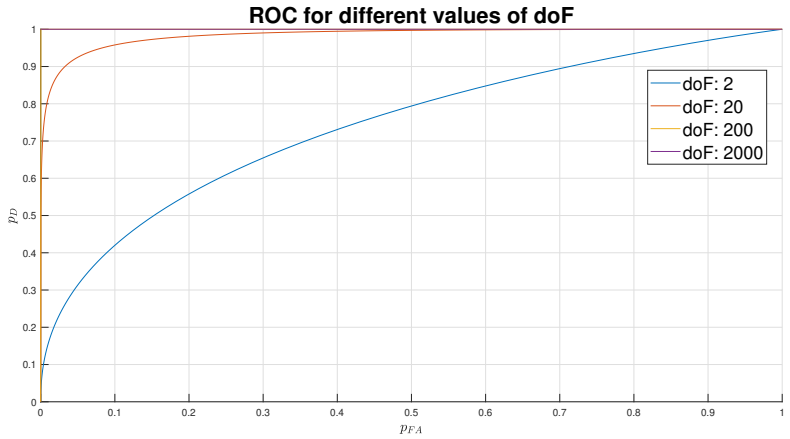


Figure 4: The receiver operating characteristics with different doF

4.5 Task 5: Performance of the general detector

$\chi^2(k)$ is a special case of the Gamma distribution where it has $k/2$ degrees of freedom. Using this and the fact that $Q \sim \chi^2$ has the property that $cQ \sim Gamma(k/2, 2c)$. $T(\mathbf{x})$ has K degrees of freedom in the Gamma distribution. Furthermore under H_0, $T(\mathbf{x})$ is scaled such that $c = \frac{\sigma_w^2}{2}$ which gives the point distribution function

$$p(x) = \frac{x^{K-1} e^{-\frac{x}{\sigma_w^2}}}{\Gamma(K)\sigma_w^{2K}} \tag{23}$$

Similarly for $T(\mathbf{x})$ under H_1 has the scaling constant $c = \frac{\sigma_w^2+\sigma_s^2}{2}$ and thus the point distribution function

$$p(x) = \frac{x^{K-1} e^{-\frac{x}{\sigma_w^2+\sigma_s^2}}}{\Gamma(K)(\sigma_w^2+\sigma_s^2)^K} \tag{24}$$

.

In figure 4 the curves become better for increasing doF. This means that if the test statistic $T(\mathbf{x})$ has more samples, the power of the test gets better. The code to compute and draw this is in A.3

4.6 Task 6: Central limit theorem

By the central limit theorem that states that for $T(\mathbf{x}) = \sum_{n=0}^{K-1} |x[n]|^2$, that $T(\mathbf{x})$ converges to a Gaussian distribution when $|x[n]|^2$ comes from a independent identical distribution with finite expected value μ and finite variance σ^2. Already know that $|x[n]|$ is complex gaussian and iid for each sample $n = 0, \ldots, K-1$, so $|x[n]|^2$ is also iid.

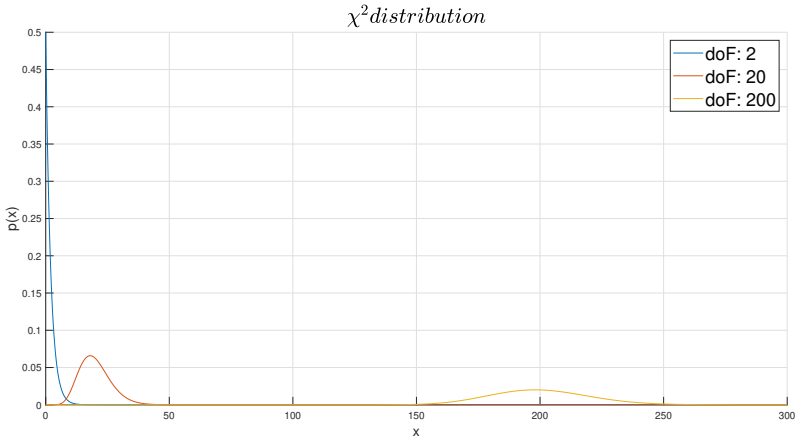


Figure 5: The pdf of χ^2 for different degrees of freedom

In figure 5 notice that for higher degrees of freedom, which essentially means for more samples, the pdf of the χ^2 distributions looks more and more like the Gaussian distribution with mean equal to the degrees of freedom.
In this case the mean is

$$\begin{aligned}\hat{\mu} = \mathbb{E}\left\{T(\mathbf{x})\right\} &= \mathbb{E}\{\sum_{n=0}^{K-1} |x[n]|^2\} \\ &= \sum_{n=0}^{K-1} \mathbb{E}\{|x[n]^2|\} \\ &= \sum_{n=0}^{K-1} (\mathrm{Var}\{|x[n]|\} + \mathbb{E}\{|x[n]|\}^2) \\ &= \begin{cases} K\sigma_w^2, & \mathrm{under} H_0 \\ K(\sigma_w^2 + \sigma_s^2), & \mathrm{under} H_1 \end{cases}\end{aligned} \tag{25}$$

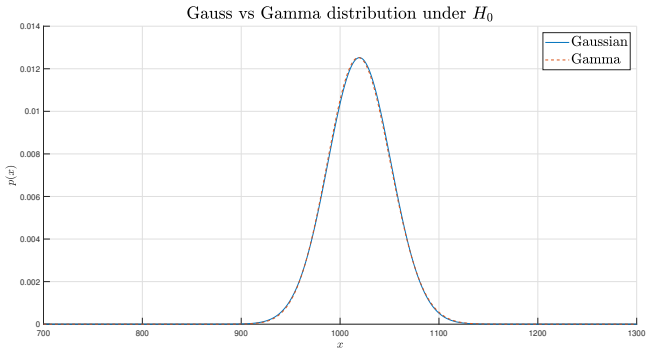
and the variance is

$$\begin{aligned}\hat{\sigma^2} = \mathrm{Var}\left\{T(\mathbf{x})\right\} &= \mathrm{Var}\{\sum_{n=0}^{K-1} |x[n]|^2\} \\ &= \sum_{n=0}^{K-1} \mathrm{Var}\{|x[n]|^2\} \\ &= \sum_{n=0}^{K-1} (\mathbb{E}\{|x[n]|^4\} - \mathbb{E}\{|x[n]|^2\}^2) \\ &= \begin{cases} K\sigma_w^4, & \mathrm{under} H_0 \\ K(\sigma_s^2 + \sigma_w^2)^2, & \mathrm{under} H_1 \end{cases}\end{aligned} \tag{26}$$

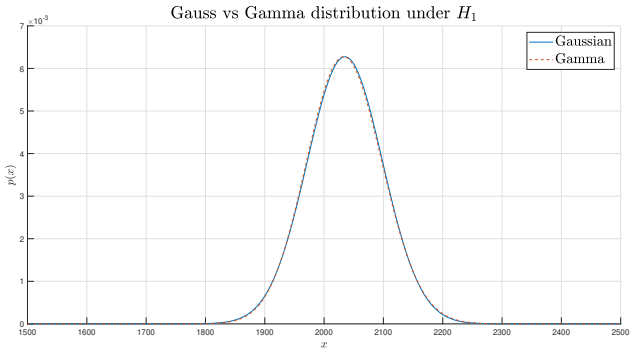
Which gives the pdf of a normal Gaussian for each of the cases as

$$f(x) = \frac{1}{\sqrt{2\pi\hat{\sigma}^2}} e^{-\frac{1}{2}\left(\frac{x-\hat{\mu}}{\hat{\sigma}}\right)^2} \tag{27}$$

This results in the two figures in for $T(\mathbf{x})$ under H_0 and H_1 in figure 6 Resulting in more general distribution that can be used for detectors on the



(a) Normal vs Gamma distribution under H_0



(b) Normal vs Gamma distribution under H_1

Figure 6

same form as $T(\mathbf{x})$, when it is a sum of iid random variables. Using this as the pdf when calculating the P_{FA} and P_D results in 7 One can see that the results are quite similar to the exact P_D and P_{FA}. The code for this problem is found in A.4

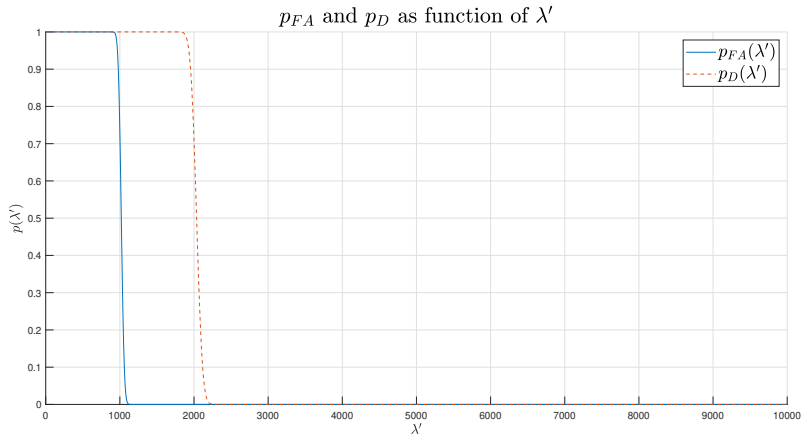


Figure 7: Showing how P_{FA} and P_D from normal distribution change with λ'

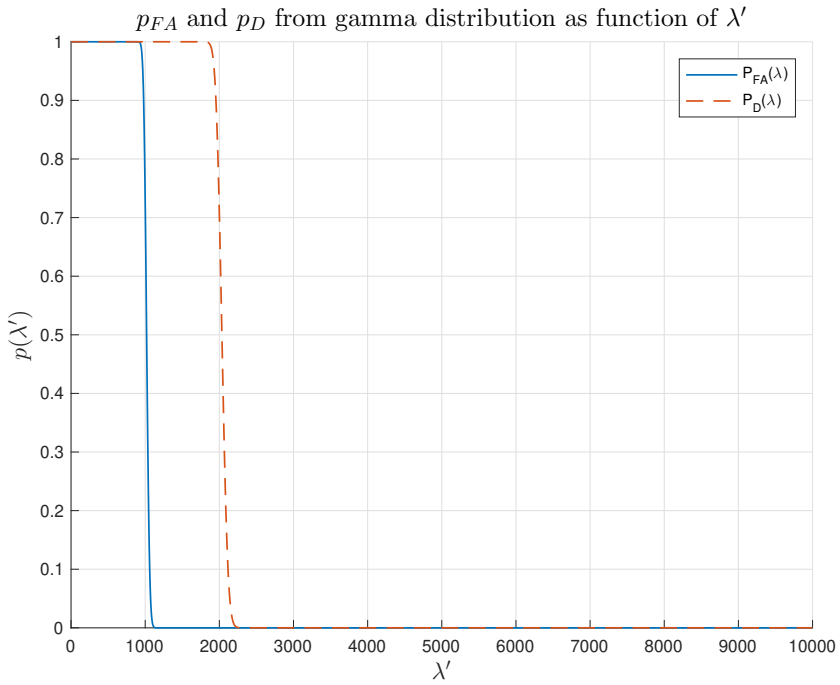


Figure 8: Showing how the exact P_{FA} and P_D change with λ'

4.7 Task 7: Detector complexity

Using the pdf for Gaussian distribution, to find an expression for K.

$$\begin{aligned} P_{FA} &= \int_{\lambda'}^{\infty} \frac{1}{\sqrt{2\pi K\sigma_w^4}} e^{-\frac{1}{2}\frac{\left(x-K\sigma_w^2\right)^2}{K\sigma_w^4}} \\ &= Q\left(\frac{\lambda' - K\sigma_w^2}{\sqrt{K}\sigma_w^2}\right) = \alpha \end{aligned} \tag{28}$$

Solve (28) for λ' to retrieve

$$\lambda' = \left(Q^{-1}(\alpha)\sqrt{K} + K\right)\sigma_w^2 \tag{29}$$

Do the same for P_D

$$\begin{aligned} P_D &= \int_{\lambda'}^{\infty} \frac{1}{\sqrt{2\pi K(\sigma_w^2+\sigma_s^2)^2}} e^{-\frac{1}{2}\frac{(x-K(\sigma_w^2+\sigma_s^2))^2}{K(\sigma_w^2+\sigma_s^2)^2}} \\ &= Q\left(\frac{\lambda' - K(\sigma_w^2+\sigma_s^2)}{\sqrt{K(\sigma_w^2+\sigma_s^2)^2}}\right) = \beta \end{aligned} \tag{30}$$

Solve (30) for K, and utilize the result from (29) as substitute for λ'

$$\begin{aligned} Q^{-1}(\beta) &= \frac{\lambda' - K(\sigma_w^2+\sigma_s^2)}{\sqrt{K(\sigma_w^2+\sigma_s^2)^2}} \\ &= \frac{(Q^{-1}(\alpha)\sqrt{K}+K)\sigma_w^2 - K(\sigma_w^2+\sigma_s^2)}{\sqrt{K}(\sigma_w^2+\sigma_s^2)} \\ &= \frac{Q^{-1}(\alpha)\sqrt{K}\sigma_w^2 - K\sigma_s^2}{\sqrt{K}(\sigma_w^2+\sigma_s^2)} \\ &= \frac{Q^{-1}(\alpha)\sqrt{K}\sigma_w^2}{\sqrt{K}(\sigma_w^2+\sigma_s^2)} - \frac{K\sigma_s^2}{\sqrt{K}(\sigma_w^2+\sigma_s^2)} \\ \frac{K}{\sqrt{K}}\sigma_s^2 &= Q^{-1}(\alpha)\sigma_w^2 - Q^{-1}(\beta)(\sigma_w^2+\sigma_s^2) \\ K &= \left(\frac{Q^{-1}(\alpha)\sigma_w^2 - Q^{-1}(\beta)(\sigma_w^2+\sigma_s^2)}{\sigma_s^2}\right)^2 \end{aligned} \tag{31}$$

Which gives the amount of samples K needed to attain the limit $P_{FA} = \alpha$ and $P_D = \beta$. Furthermore an observation that can be done is that Q is cumulative function for the standard Gaussian distribution where the mean is 0 and standard deviation is 1.

4.8 Task 8: Test with data

Trying to detect using the numerical data the result in table 2 is obtained showing how many time steps a PU was detected out of $N = 256$ time steps.

α \ β	0.98	0.99
0.1	247	253
0.01	253	251

Table 2: Showing how many PUs were detected out of 256

α \ β	0.98	0.99
0.1	158	256
0.01	253	248

Table 3: Showing how many PUs were detected out of 256

The code A.5 was used. Another run gave the result in 3 Which means it is not always perfect, but by increasing K manually instead of finding the most efficient, it was obtained 100% detection for it all. The reason the results vary are because K signals from each time steps are chosen randomly from the dataset given. However the detector is created using those randomly chosen variables. The probability of false alarm and detection does not mean the results will match perfect, but it comes close to it and it is hard to separate them when the difference between them are so small.

5 Conclusion

This project focused on Spectrum Sensing in OFDM Cognitive Radios, which is an important problem in modern communication as the spectrum is a scarce resource. After modeling how the signal data was generated, a suitable detector for the binary hypothesis testing was implemented. This was a result of using well known statistical features of the Gamma and Gaussian distribution, as well as the Neyman-Person Lemma. After testing the performance of the one-sample-detector, a more generalized K-sample detector was implemented. This new NP detector for K samples was then approximated as a Gaussian random variable which showed to be quite accurate to the original gamma distribution. This detector based on K samples showed to be useful in the way that one could choose a P_D and P_{FA} only at the cost of more samples K. In the end, this whole detector was put to a test when a data set of 100 realizations of a signal from a PU was tested on the detector. As the result in task 8 have shown, the detector did very well, and detected the signal almost every time, however increasing K manually gave a detection rate of 100%.
The project has been very educational and it has been specifically interesting to see how probability distributions are used in modern communications systems for hypothesis testing. With an increasing number of communication devices this problem is highly relevant.

References

[1] *Normal distribution*. https://en.wikipedia.org/wiki/Normal_distribution. Accessed: 2020-04-19.

[2] *Central limit theorem*. https://en.wikipedia.org/wiki/Central_limit_theorem. Accessed: 2020-04-29.

[3] *Chi-squared distribution*. https://en.wikipedia.org/wiki/Chi-squared_distribution. Accessed: 2020-04-28.

[4] *Gamma distribution*. https://en.wikipedia.org/wiki/Gamma_distribution. Accessed: 2020-04-29.

[5] Tor A. Myrvoll, Stefan Werner, and Magne H. Johnsen. *Estimation, Detection and Classification Theory*. 2020.

A MATLAB code

A.1 Problem 1

```
clc; close all; clear all;


%% Confirm that our data is iid and normal gaussian
S_k_gauss          = load(['Dataset/'...
    'T1_data_Sk_Gaussian.mat']).T1_data_Sk_Gaussian;
S_k_bpsk           = load(['Dataset/'...
    'T1_data_Sk_BPSK.mat']).T1_data_Sk_BPSK';

[N, one]         = size(S_k_gauss);

% The formula given is basically a fourier transform
s_gauss              = fft(S_k_gauss);
s_bpsk               = fft(S_k_bpsk);

s_gauss_real         = real(s_gauss);
s_gauss_imag         = imag(s_gauss);

s_bpsk_real          = real(s_bpsk);
s_bpsk_imag          = imag(s_bpsk);


%% Estimate the expected values
sr_expected_gauss            = sum(s_gauss_real)/N;
si_expected_gauss            = sum(s_gauss_imag)/N;
s_expected_gauss             = sr_expected_gauss +...
    1i*si_expected_gauss;


sr_expected_bpsk             = sum(s_bpsk_real)/N;
si_expected_bpsk             = sum(s_bpsk_imag)/N;
s_expected_bpsk              = sr_expected_bpsk +...
    1i*si_expected_bpsk;

% When samples are taken uniformly from the same population with any
% distribution the expected value of the mean is always unbiased
product_s_expected_gauss     = ...
    sum(s_gauss_real.*s_gauss_imag)/N;
product_s_expected_bpsk      = ...
    sum(s_bpsk_real.*s_bpsk_imag)/N;
```

```
41
42 disp(['The expected value of s_n' ...
43     ' with gaussian samples: '...
44     num2str(s_expected_gauss)]);
45
46
47 disp(['The expected value of s_n'...
48     ' with bpsk samples: '...
49     num2str(s_expected_bpsk)]);
50
51 disp(['The expected value of product'...
52     ' of real and imaginary part' ...
53     ' with gaussian samples: '...
54     num2str(product_s_expected_gauss)]);
55
56 disp(['The expected value of product'...
57     ' of real and imaginary part' ...
58     ' with bpsk samples: '...
59     num2str(product_s_expected_bpsk)]);
60
61 %% Gaussian Create a generic complex point distribution
62 sigma_s_sq      = 980;
63
64 px_r            = makedist('Normal', 'mu',...
65     0, 'sigma', sqrt(sigma_s_sq/2));
66 px_i            = makedist('Normal', 'mu',...
67     0, 'sigma', sqrt(sigma_s_sq/2));
68
69 x = (-80:0.1:80)';
70
71
72 %% Plot histograms of the sampled data Gaussian
73 figure(1);
74 title('Real gaussian', 'fontsize', 22);
75 hold on
76 histogram(s_gauss_real, 'Normalization', 'pdf');
77 hold on
78 plot(x, px_r.pdf(x), '--', 'Linewidth', 1)
79 hold on
80 xlabel('x', 'Interpreter', 'latex', 'fontsize', 14);
81 hold on
82 ylabel('p(x)', 'Interpreter', 'latex', 'fontsize', 14);
83 hold on
84 grid on;
```

```
85 legend('histogram', 'true pdf, $\sigma^2 = 980$',...
86     'Interpreter', 'latex', 'fontsize', 18);
87 hold off;
88
89 figure(2);
90 title('Imaginary gaussian', 'fontsize', 22);
91 hold on
92 histogram(s_gauss_imag, 'Normalization', 'pdf');
93 hold on
94 plot(x, px_i.pdf(x), '--', 'Linewidth', 1)
95 hold on
96 xlabel('x', 'Interpreter', 'latex', 'fontsize', 14);
97 hold on
98 ylabel('p(x)', 'Interpreter', 'latex', 'fontsize', 14);
99 hold on
100 grid on;
101 legend('histogram', 'true pdf, $\sigma^2 = 980$',...
102     'Interpreter', 'latex', 'fontsize', 18);
103 hold off;
104
105 %% BPSK Create a generic complex point distribution
106
107 sigma_s_sq      = 800;
108
109 px_r            = makedist('Normal', 'mu', 0,...
110     'sigma', sqrt(sigma_s_sq/2));
111 px_i            = makedist('Normal', 'mu', 0,...
112     'sigma', sqrt(sigma_s_sq/2));
113
114 x = (-80:0.1:80)';
115
116 figure(3);
117 title('Real BPSK', 'fontsize', 22);
118 hold on
119 histogram(s_bpsk_real, 'Normalization', 'pdf');
120 hold on
121 plot(x, px_r.pdf(x), '--', 'Linewidth', 1)
122 hold on
123 xlabel('x', 'Interpreter', 'latex', 'fontsize', 14);
124 hold on
125 ylabel('p(x)', 'Interpreter', 'latex', 'fontsize', 14);
126 hold on
127 grid on;
128 legend('histogram', 'true pdf, $\sigma^2 = 800$',...
```

```
    'Interpreter', 'latex', 'fontsize', 18);
hold off;

figure(4);
title('Imaginary BPSK', 'fontsize', 22);
hold on
histogram(s_bpsk_real, 'Normalization', 'pdf');
hold on
plot(x, px_i.pdf(x), '--', 'Linewidth', 1)
hold on
xlabel('x', 'Interpreter', 'latex', 'fontsize', 14);
hold on
ylabel('p(x)', 'Interpreter', 'latex', 'fontsize', 14);
hold on
grid on;
legend('histogram', 'true pdf, $\sigma^2 = 800$',...
    'Interpreter', 'latex', 'fontsize', 18);
hold off;
```

A.2 Problem 3

```
close all; clc; clear all


%% Load data
x_h0        = load(['Dataset/'...
    'T3_data_x_H0.mat']).T3_data_x_H0; % x=w
x_h1        = load(['Dataset/'...
    'T3_data_x_H1.mat']).T3_data_x_H1; % x=w+s

sigma_w  = load('Dataset/T3_data_sigma_w.mat').w;
sigma_s  = load('Dataset/T3_data_sigma_s.mat').s_t;

[N, one]   = size(x_h0);

%% Estimate the variances sigma_w_sq and sigma_s_sq
sigma_w_sq_hat  = sum(abs(sigma_w).^2)/N;
sigma_s_sq_hat  = sum(abs(sigma_s).^2)/N;


%% Calculate the histogram for our data
chi_sq_h0   = zeros(N, 1);
chi_sq_h1   = zeros(N, 1);

```

```
for i = 1:N
    chi_sq_h0(i) = 2*abs(x_h0(i))^2/sigma_w_sq_hat;
    chi_sq_h1(i) = 2*abs(x_h1(i))^2/...
        (sigma_s_sq_hat+sigma_w_sq_hat);
end



%% Generate an arbitrary chi-square random variable with two DoF

x           = 0:0.1:15;
doF         = 2;
chi_sq      = pdf('Chisquare', x, doF)';


figure(1);
title('Comparing $H_0$ with $\chi^2$',...
    'Interpreter', 'latex', 'fontsize', 22);
hold on
histogram(chi_sq_h0, 'Normalization', 'pdf');
hold on
plot(x, chi_sq, '--', 'Linewidth', 1);
hold on
grid on;
hold on
legend('Histogram of $H_0$','$pdf of \chi^2$',...
    'Interpreter', 'latex', 'fontsize', 18);
hold on
xlabel('x', 'Interpreter', 'latex', 'fontsize', 14)
hold on
ylabel('p(x)', 'Interpreter', 'latex', 'fontsize', 14)
hold off

figure(2);
title('Comparing $H_1$ with $\chi^2$',...
    'Interpreter', 'latex', 'fontsize', 22);
hold on
histogram(chi_sq_h1, 'Normalization', 'pdf');
hold on
plot(x, chi_sq, '--', 'Linewidth', 1);
hold on
grid on;
hold on;
legend('Histogram of $H_1$', 'pdf of $\chi^2$',...
```

```
    'Interpreter', 'latex', 'fontsize', 18);
hold on
xlabel('x', 'Interpreter', 'latex', 'fontsize', 14)
hold on
ylabel('p(x)', 'Interpreter', 'latex', 'fontsize', 14)
hold off
```

A.3 Problem 5

```
clc; clear all; close all

%% Data handling
sigma_w  = load(['Dataset/' ...
    'T3_data_sigma_w.mat']).w;
sigma_s  = load(['Dataset/'...
    'T3_data_sigma_s.mat']).s_t;

[K, one]    = size(sigma_w);

%% Estimate
sigma_w_sq_hat  = sum(abs(sigma_w).^2)/K;
sigma_s_sq_hat  = sum(abs(sigma_s).^2)/K;


%% Figures
x           = 0:0.01:300;

figure(1);
title('$\chi^2 distribution$',...
    'Interpreter', 'latex', 'fontsize', 22);
hold on
for k = (0:2)
    doF         = 2*10^k;
    chi_sq      = pdf('Chisquare', x, doF)';
    plot(x, chi_sq, 'Linewidth', 1,...
        'DisplayName', ['doF: '...
        num2str(doF)]);
    hold on
end
legend('show', 'fontsize', 18);
hold on
xlabel('x', 'fontsize', 14);
hold on
ylabel('p(x)', 'fontsize', 14);
```

```
hold on
grid on;
hold off


%% Calculate and plot ROC
figure(2);
title('ROC for different values of doF', 'fontsize', 22);
hold on

for i = (0:3)
    doF      = 2*10^i;
    x        = 0:0.1:2500;
    p_FA     = 1-gamcdf(x, doF, sigma_w_sq_hat);
    p_D      = 1-gamcdf(x, doF, ...
        (sigma_w_sq_hat+sigma_s_sq_hat));
    plot(p_FA, p_D, 'Linewidth', 1,...
        'DisplayName', ['doF: ' num2str(2*10^i)]);
    hold on

end
legend('show', 'fontsize', 18);
hold on
grid on;
hold on
xlabel('$p_{FA}$', 'Interpreter', 'latex',...
    'fontsize', 14);
hold on
ylabel('$p_{D}$', 'Interpreter', 'latex',...
    'fontsize', 14);
hold off;
```

A.4 Problem 6

```
clc; clear all; close all;

%% Data handling
sigma_w  = load(['Dataset/' ...
    'T3_data_sigma_w.mat']).w;
sigma_s  = load(['Dataset/'...
    'T3_data_sigma_s.mat']).s_t;

[K, one]    = size(sigma_w);

```

```
%% Estimate the variances sigma_w_sq and sigma_s_sq
sigma_w_sq_hat  = sum(abs(sigma_w).^2)/K;
sigma_s_sq_hat  = sum(abs(sigma_s).^2)/K;

%% Calculating p_D and p_FA
lambda_prime    = (0:0.1:10000);

mu_h0           = K*sigma_w_sq_hat;
sigma_h0        = sqrt(K)*sigma_w_sq_hat;
p_fa            = 1 - normcdf(lambda_prime,...
    mu_h0, sigma_h0);

mu_h1           = K*(sigma_w_sq_hat+sigma_s_sq_hat);
sigma_h1        = sqrt(K)*(sigma_w_sq_hat+sigma_s_sq_hat);
p_d             = 1 - normcdf(lambda_prime,...
    mu_h1, sigma_h1);


figure(1);
title('$p_{FA}$ and $p_D$ as function of $\lambda''$',...
    'Interpreter', 'latex', 'fontsize', 22);
hold on
plot(lambda_prime, p_fa, 'Linewidth', 1);
hold on
plot(lambda_prime, p_d, '--', 'Linewidth', 1);
hold on
grid on;
hold on
legend('$p_{FA}(\lambda'')$','$p_D(\lambda'')$',...
    'Interpreter', 'latex', 'fontsize', 18);
hold on
xlabel('$\lambda''$',...
    'Interpreter', 'latex', 'fontsize', 14);
ylabel('$p(\lambda'')$',...
    'Interpreter', 'latex', 'fontsize', 14);
hold off

figure(4);
title('$p_D$ as a function of $p_{FA}$',...
    'Interpreter', 'latex', 'fontsize', 22);
hold on
plot(p_fa, p_d, 'Linewidth', 1);
hold on
grid on;
```

```
hold on
legend('$K = 1024$',...
    'Interpreter', 'latex', 'fontsize', 18);
hold on
xlabel('$\lambda''$',...
    'Interpreter', 'latex', 'fontsize', 14);
ylabel('$p(\lambda'')$',...
    'Interpreter', 'latex', 'fontsize', 14);
hold off

%% Check out the gaussian distribution up against the gamma distributio
x       = 700:0.1:1300;
gaussh0 = normpdf(x, mu_h0, sigma_h0);
gammah0 = gampdf(x, K, sigma_w_sq_hat);

figure(2);
title('Gauss vs Gamma distribution under $H_0$',...
    'Interpreter', 'latex', 'fontsize', 22);
hold on
plot(x, gaussh0, 'Linewidth', 1);
hold on
plot(x, gammah0, '--', 'Linewidth', 1);
hold on
grid on;
hold on
legend('Gaussian','Gamma',...
    'Interpreter', 'latex', 'fontsize', 18);
hold on
xlabel('$x$',...
    'Interpreter', 'latex', 'fontsize', 14);
ylabel('$p(x)$',...
    'Interpreter', 'latex', 'fontsize', 14);
hold off


%% Under H1
x         = 1500:0.1:2500;
gaussh1   = normpdf(x, mu_h1, sigma_h1);
gammah1   = gampdf(x, K, (sigma_w_sq_hat+sigma_s_sq_hat));

figure(3);
title('Gauss vs Gamma distribution under $H_1$',...
    'Interpreter', 'latex', 'fontsize', 22);
hold on
```

```
plot(x, gaussh1, 'Linewidth', 1);
hold on
plot(x, gammah1, '--', 'Linewidth', 1);
hold on
grid on;
hold on
legend('Gaussian','Gamma',...
    'Interpreter', 'latex', 'fontsize', 18);
hold on
xlabel('$x$',...
    'Interpreter', 'latex', 'fontsize', 14);
ylabel('$p(x)$',...
    'Interpreter', 'latex', 'fontsize', 14);
hold off
```

A.5 Problem 8

```
clc; clear all; close all;

%% Load data
signal  = load(['Dataset/T8_numerical'...
    '_experiment.mat']).T8_numerical_experiment;

%% Statistics data
sigma_w_sq  = 1;
sigma_s_sq  = 5;

%alpha_0     = 0.1;  % Choose maxlimit p_FA = 0.1
alpha_0     = 0.01; % Choose maxlimit p_FA = 0.01

%beta_0      = 0.98; % choose minlimit p_D = 0.99
beta_0      = 0.99; % choose minlimit p_D = 0.98

K           = round(((norminv(alpha_0)*sigma_w_sq-...
    norminv(beta_0)*(sigma_w_sq+sigma_s_sq))/sigma_s_sq)^2);

signal      = signal(:,randi([1, 100], [K, 1]));

[N, K]      = size(signal);
T_x         = sum(abs(signal).^2, 2);

%% Detector
% Calculate the threshold
mu_h0           = K*sigma_w_sq;
```

```
28 sigma_h0         = sqrt(K)*sigma_w_sq;
29 lambda_prime     = norminv(1-alpha_0, mu_h0, sigma_h0);
30
31 disp(['The chosen threshold was lambda_prime: ' num2str(lambda_prime)])
32 disp(['Here you get the test power beta: ' num2str(beta_0)]);
33 disp(['Here you get the false alarm rate alpha: ' num2str(alpha_0)]);
34 disp(['Number of samples used for test statistic: ' num2str(K)]);
35 % With this threshold, at which timesteps has PUs
36 results          = gt(T_x, lambda_prime);
37 disp(['Of ' num2str(N) ' timesteps, in '...
38     num2str(nnz(results>0))...
39     ' of them, PU was detected']);
```