Czech Technical Univerzity in Prague

Faculty of electrical engineering

Diploma thesis

2014

Petr Tománek

Czech Technical Univerzity in Prague

Faculty of electrical engineering

Department of Radio Engineering

Synchronization and Hierarchical Pilot Signal Design for Wireless Physical Layer Network Coding

May 2014

Author: Petr Tománek

Supervisor: prof. Ing. Jan Sýkora, CSc.

Statutory declaration

I declare that I have worked on my diploma thesis on my own using only the sources mentioned in the bibliography. I agree that my thesis or it´s parts can be borrowed and provided to public.

Date: 25.5 2012

....

Signature

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Electromagnetic Field

DIPLOMA THESIS ASSIGNMENT

Student: **Bc. Petr Tománek**

Study programme: Communications, Multimedia and Electronics
Specialisation: Wireless Communication

Title of Diploma Thesis: **Synchronization and Hierarchical Pilot Signal Design for Wireless Physical Layer Network Coding**

Guidelines:

Student will get acquainted with WPNC fundamentals and basics of estimation theory and statistical signal processing. A proper attentions should be given to the impact of the channel parametrization (relative, common) on the performance of Network Coded Modulation. Then student focuses on the development of hierarchical pilot signals and related receiver synchronization and parameter estimation processing that would work in WPNC scenarios. The work should start with simple orthogonal pilots and simple single-stage scenarios. Then it should be generalized into the non-orthogonal (true hierarchical) pilot design and possibly also into a large scale multi/mixed stage distributed synchronization. Results should be analyzed analytically where possible and/or supported by numerical or simulation based approach where appropriate. A selected simple scenarios should be also implemented into the experimental TxR cloud framework and verified by a real radio hop experiment.

Bibliography/Sources:

[1] Jan Sykora, Alister Burr. Advances in Wireless Network Coding - The Future of Cloud Communications, IEEE PIMRC 2013 tutorial, London, September/2013
[2] E. Biglieri: Coding for Wireless Channels, Springer, 2005
[3] S. Kay: Statistical Signal Processing, Vol. I&II, Prentice Hall 1993
[4] Internal materials of FP7/DIWINE project

Diploma Thesis Supervisor: prof.Ing. Jan Sýkora, CSc.

Valid until: Summer Semester 2014/2015

v. z.

prof. Ing. Miloš Mazánek, CSc.
Head of Department

L.S.

v. z.

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, December 30, 2013

Acknowledgement :

I would like to thank prof.Ing. Jan Sýkora, CSc. for his time he spent giving me long and exhausting consultations. I would also like to thank Ing. Tomáš Hynek and Bc. Jan Hejtmánek for their incredibly helpful comments and ideas during the whole time.

But most of all I would like to thank my family for their incredible support and encouragement that helped me in the toughest moments.

Summary:

Student will get acquainted with WPNC fundamentals and basics of estimation theory and statistical signal processing with the emphasis on their applications in the synchronization. A proper attentions should be given to the impact of the channel parametrization (relative, common) on the performance of Network Coded Modulation. Then student focuses on the development of hierarchical pilot signals and related receiver synchronization and parameter estimation processing that would work in WPNC scenarios. The work should start with simple orthogonal pilots and simple single-stage scenarios. Then it should be generalized into the non-orthogonal (true hierarchical) pilot design and possibly also into a large scale multi/mixed stage distributed synchronization. Results should be analyzed analytically where possible and/or supported by numerical or simulation based approach where appropriate. A selected simple scenarios should be also implemented into the experimental TxR cloud framework and verified by a real radio hop experiment.

Key words:

Network Coding, Estimation theory, Synchronization, Pilot signal design

Abstrakt:

Student se seznámí se základy WPNC, teorie odhadu a statistického zpracování signálu se zaměřením na jejich aplikaci pro synchronizaci. Naležitá pozornost by měla být věnována dopadu parametrizace kanálu (společný, relativní) na fungování Síťově kódované modulace. Poté se student zaměří na vývoj hierarchického pilotního signálu a s tím spojené synchronizaci a odhadu kanálu, který by fungoval ve WPNC scénáři. Práce začne jednoduchými ortogonalními piloty a jednoduchou jedno etapovou ukázkou. Poté by se měla zobecnit v návrh noortogonálních (pravých hierarchických) pilotů and pokud možno ve vice stupňovou distribuovanou synchronizaci. Výsledky by měly být analyzovány analiticky a pokud to půjde, tak předvedeny na simulaci. Vybrané scénáře by měly být také implementovány v expetimantálnám TxR cloud frameworku a vyzkoušeny na realném radiovém přenosu.

Klíčová slova:

Sítˇové kódování, Teorie odhadu, Synchronizace, Návrh pilotních signálů

Contents

Notation

ARMA	Auto-Regressive Moving Average
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CAZAC	Constant Amplitude Zero Auto-Correlation
CFO	Carrier Frequency OUset
CRLB	Cramer- Rao Lower Bound
CSE	Channel State Estimator
DA	Data Aided
DD	Decision Directed
DF	Decode and Forward
HNC	Hierarchical Network Code
LOS	Line-Of-Sight
LS	Least Square
LTE	Long-Term Evolution
MAP	Maximum A posteriori Probability
MIMO	Multiple Input Multiple Output
ML	Maximum Likelyhood
MM	Method of Moments
MSE	Mean Square Error
NC	Network Coding
NDA	Non-Data Aided
NSTA	Non-Symbol Timing Aided
OFDM	Orthogonal Frequency-Division Multiplexing

PDF	Power spectral density
PHY	Physical Layer Additive
PiAcq	Acquisition Pilot
PiCSE	CSE Pilot
PiHrc	Hierarchy Pilot
PL	Payload
SDF	Selective Decode and Forward
SNR	Signal-to-Noise Ratio
STA	Symbol Timing Aided
var	Variance
WNC	Wireless Network Coding
WPNC	Wireless Physical Layer Network Coding

List of Figures

1. Introduction to Wireless Network coding

Wireless, in its various forms, is an increasingly dominant communication medium. It provides the means for mobility, city-wide Internet connectivity, distributed sensing, etc. Right now, all over the world, mobile access to the internet is becoming wholly fundamental to doing business in all industries. Flexible working practices facilitated by mobile networks and devices are already essential, and are allowing enterprises to conduct operations across boundaries that previously inhibited growth. The current generation of mobile networks continues to transform the way people communicate and access information. Further developing and implementing technologies that enable true human-centric and connected machine-centric networks will come to redefine end user mobility along with the entire landscape of the global telecoms industry. Integration of mass-scale cloud architectures will infuse mobile networks with capabilities for flexibly delivering services at unprecedented speeds while meeting forecasts for tremendous growth in mobile data traffic, IoT connectivity, and security. A more massive capacity for managing connections will better enable a greater widespread adoption of M2M services and interactions, and will facilitate innovation in localized mobile service delivery. Wireless networks will increasingly become the primary means of network access for person-to-person and person-to-machine connectivity. These networks will need to match advances in wired networking in terms of delivered quality of service, reliability and security.

1.1. Wireless Physical Layer

So far wireless network have been designed using wired network as the model. By using wireless channel as a point-to-point link and applying wired network protocols like shortest path routing on them, while ignoring the fact that we are dealing with wireless environment, makes whole structure flawed by design for future improvement. Such design has worked well for wired networks, but is not sufficient enough for for the unreliable and unpredictable wireless medium.

Today wireless networks face a lot of problems like low throughput, dead spots, and inadequate mobility support. As mentioned before it is mainly because of fact, that wireless medium is different from the wired one. Wired links are unicast, meaning that any transmissions in a wired network does not interfere with other, on the other hand, most of the wireless communication is broadcast with omni-directional antennas, that leads to the problem. Instead of having clear information about which node is connected through which link, we are receiving superposition of electromagnetic waves from multiple sources. This kind of interference is dealt with by using different kind of ortogonality

- time, frequency, spreading code sequence, polarization, space etc. Another critical difference is the very idea of wired node being mostly considered static, where as wireless systems are build to be mobile and portable. All of those criteria show that wired network design is far from being optimal for wireless medium.

1.2. Wireless Network coding

A significant breakthrough came in 2000 when Network Coding (NC) was introduced in [1]. An important change was introduced. In classical routing network sollution we simply stored and forward information, in NC solution intermediate nodes can mix information in different messages in order to achieve multicast capacity. Putting information together is called encoding, we denote it χ_k for node k. Such function can be for example simple XOR. In order to retrieve information in destination node, we need to perform inverse operation χ_k^{-1}. This new approach had enormous impact on the way wired network were built and has a lot of implementations in wide area of applications.

However, in wireless networks as mentioned before, we are not able to distinguish incoming messages sharing same resource, unlike in wired networks where we know which data line came from what node. WNC is located at PHY of communication link, because the superpossition of the signals at a receiving antenna is not something we can get rid of.

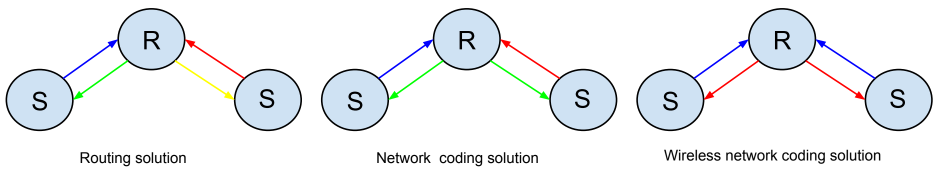

Figure 1.1.: Two-way relay channel

Figure 1.1 shows all the scenarios mentioned up to now on a simple two-way relay channel. First one is classical routed solution where delivering 2 packets takes 4 resource slots. Each source takes different slot to transmit the message and relay itself has to transmit individual massages in one slot. Making its throughput 1/2. Second and enhanced option is network coded scenario where the sources use separated time slot to avoid a collision when communicating with the relay but relay can forward those messages in one slot so it takes only 3 resource slots and upgrades the throughput up to 2/3. And the last option is case of wireless network coding where we use the fact that signals are combined in wireless environment by default to our advantage. This results in throughput of 1 packet per channel use.

Huge disadvantage of wireless network is amount of the transformation signal is going to get on its way to receiver. Attenuation of the signal, multipath spreading, dispersion in the frequency, phase rotation, time delay, all of those can be fought by channel estimation and synchronization. This work focuses on the area of introducing basic principals of estimation theory and its application in WNC as well as synchronization and design of pilot signals for such usage.

1.3. Relaying Strategies

At this point we can divide the way signals are processed after their superposition directly at PHY into several groups. Such actions are called Relaying Strategies and we separate them by the fact whether there is decision to be made by the relay about incoming signals or not.

Amplify & Forward (AF) [3] and Analogue Network Coding (ANC) [4] are strategies where no decision is to be made about incoming signals. Only action made by the relay is scaling or simply amplifying the superposition who received signals and transmit it towards the other nodes. This method is very simple but also has disadvantages because by amplifying received signal we also increase its noise that is spread throughout the network.

By Decode & Forward (DF) we mean that relay is making some decision about incoming superposition. There are kind of decisions based of which we recognize several following strategies.

- Compute & Forward (CmpF) [9,10] uses properties of lattice codes in order to process superimposed signals at PHY.
- Joint Decode & Forward (JDF) and Physical Layer Network Coding (PLNC) [5,6]. Relay tries to bring back the situation to dedicated channel per each source by decoding incoming superposition and applying network code on those estimates.
- Hierarchical Decode & Forward (HDF) [7,8] Unlike in case of JDF, HDF works with superposition of codewords in signal representation.
- De-Noise & Forward (DNF) [6] is very similar to HDF but more focused on symbol by symbol processing.

Part I.

Theoretical Background

2. Estimation theory

2.1. Parameter model

We assume that all parameters could be represented as a vector parameter θ. This holds if the parameter itself is constant (either scalar or vector). But we also know that all time dependent parameters like time-dependent phase, time-dependent attenuation can be expressed using signal expansion in constellation or sampling space. So we can still represent whatever we like as a series of coefficients, which would be ordered into the vector. So whatever appears as a θ is considered as a vector of parameters. Such notation covers any application we can think of.

Parameters can be classified into stochastic and deterministic domains. And inside the class of stochastic parameters we can have variety of options. First off all, the "most friendly" option is that there is a full knowledge of Probability Density Function of the parameter, that means random parameter with known PDF. The second option is that we know that parameter is random but we are not given the knowledge of the PDF. But sometimes we can know at least what is the class of PDF. For example we can say about parameter that it has Gaussian PDF and we do not have any knowledge about variance and mean value. So we are sure that It is type of Gaussian class of PDF but we do not have full parametrization of the PDF.

Sometimes we know just the range of the values. For example that the parameter could be only positive values (It is random, we are restricted to the positive numbers that cannot say what is the PDF). All that knowledge will be utilized by the estimator. Another important example is that parameter is random with unknown PDF. But we know that PDF is symmetric. That means the probability of being below or above some given value is the same. Which is sometimes very useful.

For the deterministic parameter the only knowledge we can have is the parameter range.

We have enormous variety of possibilities of how to model the dynamics of the parameter. If we constrain ourselves to the linear models then we can describe the parameter for example by Power Spectrum density (PSD), correlation properties. We can even describe it using State Space Description for by defining the output and state equation for continuous time

$$\frac{\partial \theta}{\partial t} = X(\theta(t), u(t), t)$$

or in discrete time

$$\theta[k+1] = X(\theta[k], u[k], k)$$

where u is excitation.

One of the options of how to model dynamics is also ARMA (auto-regressive Moving average) model.

All of mentioned options are valid possibilities how to model dynamics of the parameter. The advantage of modeling inside of linear models is that whatever we are given under some mild conditions we can usually find all the other descriptions of the parameter. So for example given the knowledge of the ARMA model we can quite easily describe power spectrum density, given the knowledge of PSD we know directly what are the correlation properties and vice versa.

2.2. Estimator

When designing estimator, we essentially perform three steps. The first one called Estimator criterion is the step where we must define what is the criterion fidelity of the estimate. That means how would we measure the quality of estimate. It is usually directly given in terms of some performance goal or metric. So for example we say, that our estimator would be maximazing A posteriori Probability or minimize mean square error. Different goals require different parameter models so we cannot apply every possible criterion for every possible parameter. For example if we want to minimize mean square error then the word *mean* means that we must have a stochastic model for a parameter, because otherwise unless we have a stochastic model of a parameter we cannot evaluate averages or a mean value, so no combinations are possible and for different parameter classes we define a different possible fidelity criterions. The major distinction is between random and deterministic parameters.

Once we set the goal in terms of what are we supposed to maximizes then we need to capture that in mathematical form. This is what we call estimator equation. Usually the equation itself is formulated in terms of some function ρ, which is the utility or performance goal. And we maximize it

$$\hat{\theta} = arg\,max_{\check{\theta}}\rho(\check{\theta})$$

or minimize it

$$\hat{\theta} = arg\,min_{\check{\theta}}\rho(\check{\theta})$$

The argument which maximizes or minimizes the function is the estimate.

Finally once we create equation of the estimator we need to solve that equation, but that doesnt mean that we are able to solve it. In some easy cases of $arg\,max\rho(\theta)$ we can set first derivative of θ to be zero and call that function $\mu(\theta)$ and then we can solve that equation to provide estimate. But this is only in very rare set of cases. On the other hand if it happens and equation is simple enough to be implemented this is definitely the goal.

But sometimes, we must solve the equation in a approximate iterative way. Most typical is to use iterative solver. This off course adds a completely new layer of imperfections in solver.

So we define the criterion of fidelity, then we set the equation and then the solver itself can sometimes introduce a brutal approximation. So something that we would get from the equation directly does not necessarily has to be the one that we obtain from the solver. The most typical example is iterative solver, where we solve the equation in iterative way and we still got some marginal error of the solver and hope that the solver would converge to the solution that we would otherwise get by direct solution.

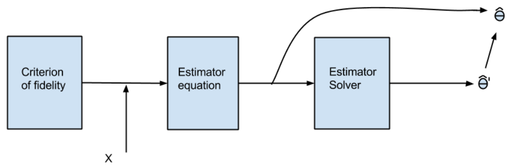

Figure 2.1.: Estimator criterion

2.2.1. Classification of the criterions

Criterions could be classified into three major classes depending on stachastic or deterministic nature or possibly no nature at all.

First one is the criterion which sits in stochastic domain. Typically the performance metric is something define in a stochastic way. For example mean square error, maximum probability of any type or sometimes it is likehood - stochastic based value.

The second possibility are the criteria which are defined regardless of any stochastic assumptions. Those are called time domain or series type of a criteria, because usually most typical domain of the parameter over the indices of θ is the time. Those criteria are import by the fact that the goal metric must not utilize in any way the stochastic knowledge of the parameter because we do not have that available and it is usually in a form where we got some parametric approximation of the function and we are just trying to fit that parametric approximation to what we observe. The second condition is that we have available only the single realization of the signal or measurement observation.

The third class is class on its own, which has absolutely no relationship to any interpretable performance criterion. It is completely Ad-hoc method. Example of that is Method of moments.

2.2.2. Examples of estimators- Maximum Likelyhood

First estimator is Maximum Likelihood (ML). The goal we are trying to achieve is to maximize is the conditional PDF. The ML estimator is one that maximizes the probability that we get what we receive conditioned on the parameter itself. So we must know the stochastic input/output of the channel. Then we observe some received signal x and we try to choose the one that causes the most probably the one we obtain

$$\hat{\theta} = arg\, max_{\check{\theta}} p(x \mid \check{\theta})$$

Maximum Likelyhood estimator is one of the most popular. The first reason is that it can always be constructed, because in order to even start doing anything with estimator on receiver side, we need to know the observation model, so knowledge of this observation model is starting point of everything. Once we have an observation model we have got directly a mathematical form of the estimator. ML has also very good performance because it is asymptotically unbiased and efficient. If we have long enough observation then we won't make a systematic error, we would get a correct value of a mean and we are capable of attaining Cramer-Rao Lower Bound (CRLB), which is the minimum possible variance of any estimator. That means that ML touches the best performance among any estimators if the observation is long enough. In asymptotically regime, the mean of the estimate is given by the value of the parameter itself and moreover the distribution of the estimate is Gaussian. So the estimator touches the best performance it gives the true value of the parameter in mean it has the minimum possible variance among all other estimators and we know what is the distribution of the error. For all those reason the ML estimator is very frequent and very popular choice (around 99 percent of estimators in digital communication is based on Maximum Likelyhood)

2.2.3. Examples of estimators- Bayesian estimator

The second is the whole class of estimators. It is class of Bayesian estimators.

We define so called Loss function

$$L(\theta, \check{\theta})$$

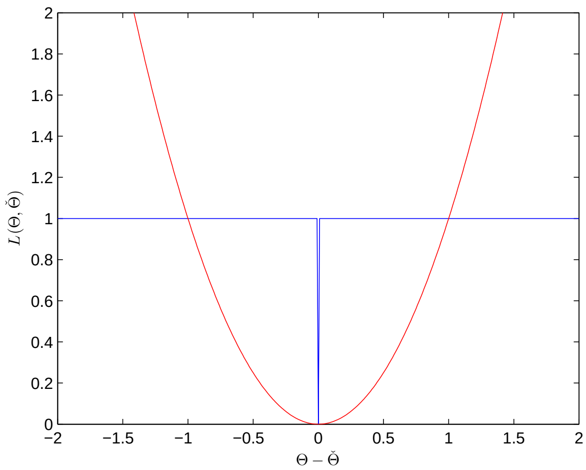

Figure 2.2.: Example of Loss functions

This is in fact performance metric, which says how much " it hurts to make error". As the figure 2.2 shows on the horizontal axis is difference between estimator and the true value and vertical axis indicates Loss function. If we hit directly parameter to true value the Loss is zero. So every Loss function is zero at zero difference between estimate and true parameter. The figure 2.2 shows two examples the blue function is "hit or miss" uniform function typical used in coding in digital communication. The second one is quadratic one which is related to the energy of error of the estimation.

Then we define Bayesian risk

$$R(\check{\theta} = E_{x,\theta}[L(\theta,\check{\theta})])$$

it is mean value of the Loss function over all random influences of the system. There are influences cause by signal observation but there are also random influences caused by parameter itself because the parameter itself could be random, for example phase of the system or transmitted data.

Minimizing the average risk with the uniform function is the same as minimizing the conditional risk and it is the same as maximizing the A posteriori probability of the parameter and for that reason the resulting estimator is called Maximum A posteriori Probability (MAP) estimator.

$$\hat{\theta} = arg\, min\, R(\check{\theta}) = arg\, max\, p(\theta\check{|}x)$$

In all situations when we do not care how far we get with the estimator (hit or miss situation), the MAP estimator is a good choice

If the A priory probability of the parameter is a uniform one then formaly the MAP estimator becomes ML estimator. So if we are sure that the parameter itself is uniformly distributed ML and MAP should give exactly the same results or vice -versa we can see what is the additional advantage of MAP estimator against ML because Maximum Likelyhood cannot reflect uneven a priory probability of a parameter, while MAP estimator reflects that correctly.

The second example of a loss function is the one which is directly related to the energy $||\theta - \check{\theta}||^2$ and is something we can directly interpret as the energy of a difference and Bayesian risk is then nothing else than mean energy of the difference, which is generally called Mean Square Error (MSE).

MSE estimator can be proved to be equal to calculating conditional mean of the parameter.

$$\hat{\theta} = E\left[\theta|x\right]$$

An example of an estimator, where we do not need any stochastic knowledge so it is completely base on observing and fitting the model in some way to be as close as possible to what we observe and the measure of such fitting, is in term

$$\hat{\theta} = arg\, min\, ||x - s(\check{\theta})||^2$$

so we observe x then we got the model of useful part of the observation and we try to fiddle the parameter θ in such a way that the observation and the model of the observation would be as close as possible in terms of norm of the difference. The observation model could we absolutely arbitery. We dont even need to know what is the fluctuation part of the model. But in order to make this type of estimator providing reasonable results, the fluctuation of the model must be symmetric so in order to make Least Square (LS) solution viable we need to assume that the model fluctuations are symmetric around the system model, because otherwise the results are producing systematic error and we don't know what is the systematic error because we don't know what is the model of the observation.

The last method, which has absolutely no utility target at all and is completely ad hoc method, is a Method of Moments (MM). The idea is very simple. Let us assume that we have some parameter that is observed through x and we take it and evaluate some moment of x. That means evaluating mean of some function which is typically polynomial function. So the first moment is mean value, second moment is mean value of the square, third moment is the mean value of third power and so on, generally it can be any function. We assume that moment itself has some known relationship to the parameter and we can denote that relationship by function $h(\theta)$

$$\mu = h(\theta)$$

Then obviously θ would be inverse function which has the argument of the moment

$$\theta = h^{-1}(\mu)$$

A real estimator is then taking the same function but instead of a true moment we substitute it with estimate.

For example we can imagine that some parameter θ is related to the mean value of the signal

$$\mu = E(x)$$

then the true Theta would be

$$\theta = h^{-1}(\mu)$$

and Theta estimate would be

$$\hat{\theta} = h^{-1}(\hat{\mu})$$

where is $\hat{\mu}$ is estimate that is calculated by averaging in time domain

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} Y_i$$

2.3. Performance parameters of the estimator

This section is denoted to measuring the performance of estimator, which means saying which one is better than the other. There is large variance of criteria and this work aims to cover only the most basic and important ones. In addition to those we can of course evaluate more other criteria for example the performance of the estimator given the amount of A priory known information we need in order to build the estimator. If we allowed ourselves a luxury long pilot then the performance of the estimator would be wonderful, but the price we pay is length of pilot. So it is something we need to make a trade of.

Another important one would be robustness. We need to have an estimator that works regardless on the channel model assumption.

Sometimes the estimators can work quite nicely for almost zero error of other parameter. But if we allow that there is nonzero error of that parameter then estimator collapses so it is important performance feature to say what is the allowed range of other parameters which are potentially influencing the performance of our estimator.

2.3.1. Bias - Deterministic parameter

This is parameter that shows whether the estimate on mean gives the correct value of the parameter. This is the behavior we usually expect. No real estimator makes sense unless it is unbiased.

The formal definition of the bias is that It is mean between the estimate and the true value of the parameter

$$b = E[\hat{\theta} - \theta]$$

in case of deterministic parameter the mean of parameter is parameter itself so then

$$b = E[\hat{\theta}] - \theta$$

And we say that $\hat{\theta}$ is unbiased if $b = 0$

2.3.2. Estimator variance - Deterministic parameter

We define two entities. The first one is Mean Square Error (MSE). The definition of MSE is as name suggests mean value of a square error, where error is difference between true parameter value and it's estimate.

$$MSE = E[|\hat{\theta}_i - \theta_i|^2]$$

It says how close we are with our estimate to true value. The further we are the more penalty we pay. If the parameter itself has some kind of interpretation related to the energy of the difference, then it is specifically interpretable because it is energy of the difference and we can measure it. So the MSE is very natural performance indicator. But in order to evaluate MSE we need estimate and the real value of the parameter which we do not have at the receiver side because real value of the parameter is something we do not have. So we have something that is very nicely interpretable but we cannot find nor measure it. On the other hand the next entity is variance which is something we can easily obtain from observation. So on the one side we have something that is difficult to get and is very useful, and on the other side something we can measure quite easily but does not say anything useful. Fortunately there is circumstance that relates those two together. It is under condition of zero mean. We can find relationship between mean square error and variance where they equal each other if estimator is unbiased.

2.3.3. Random Parameter

For the deterministic parameter there was a true value and the estimate was jumping around so we measured whether it jumps symmetrically around the true value and then what is the square difference. For a random parameter the parameter itself jumps so the mean bias and mean square error is not the useful measure of the performance anymore because those two are loosing their original interpretation as they have for the deterministic parameter.

There are some ways how to solve it. The easiest one is to return the situation back to the deterministic by evaluating a conditional characteristics that means that we can now evaluate what is the conditional square error,

$$MSE(\theta_i) = E[|\hat{\theta} - \theta|^2|\theta_i]$$

conditional bias,

$$b(\theta) = E[\hat{\theta}|\theta] - \theta$$

as there is of course same relationship between conditional MSE and conditional variance as It were before. All of them are define by expectations conditioned by the true value of the parameter so we are asking what would be the performance If we stopped the randomness of the parameter for fixed value.

The interpretation is now that we can do that but there has to be awareness of what we are evaluating. We are evaluating the performance if the true value was a given value so we get an additional degree of freedom.

2.3.4. Acquisition time

In simple terms Acquisition time says how long it would take the synchronizer to get synchronized. But we need to keep in mind that everything is random so the time to get a synchronization is not a deterministic value. It would be time depending on noise, initial state of the synchronizer and all of those are random so we need to use a probability description. The second issue is that do we understand by the word synchronized, meaning where is the point we can say we are synchronized.

So we define what is the probability that the estimation error would be within some region which we call Locked state region at time t so we can say that synchronizer reaches the synchronization locked region in some time with given probability, provided that higher probability takes longer time.

$$T_a(P_a): \quad Pr\left\{(\theta(t) - \hat{\theta}(t)) \in A, \forall t > T_a\right\} = P_a$$

On the other hand, event that could potentially happen is called Synchronization failure, and the time it takes - Synchronization failure time. This is the procedure when we leave the synchronization mode and we usually define that as a mean time between synchronizer locked state exits. That means how much time takes between two occurrences of loosing the synchronization.

2.3.5. PDF of estimation error

The final and ultimate goal is the infulance on Bit Error Rate (BER), which is what we are doing the synchronization for. Let us assume that we can evaluate the probability of a decoder as a function of actual error of the parameter estimator.

$$p(\theta_\epsilon), \quad \text{where } \theta_\epsilon = \hat{\theta}(x) - \theta$$

To get the avarage probability of the error we simply avarage this bit error rate over the density function of the error of the parameter.

$$P_e = \int_{\{\theta_\epsilon\}} P_e(\theta_\epsilon) p(\theta_\epsilon) \mathrm{d}\ \theta_\epsilon$$

where

$$P_e(\theta_\epsilon) = Pr\left\{\hat{d}(x) \neq d | \theta_\epsilon\right\}$$

is conditional probability of data detection error.

But in order to do that we need to have a stochastic description of the error of the parameter. If we use ML estimator, meaning the estimator is having a long observation, then we can just calculate what is the mean and we evaluate variance and we use the Gaussian PDF.

2.4. Performance limits

Essentially the performance limits means a single extremely important theorem which is called Cramer-Rao Lower Bound (CRLB). This theorem says that if some condition holds then the variance of any unbiased estimator is lower bounded by some value. So we can find what is the best possible performance which cannot be beaten by any estimator at all on condition that this estimator is unbiased. Since we do not like any other estimator then unbiased ones, this is a generally applicable theorem that says what is the performance limit. So this is ultimate bound for the quality of any unbiased estimator. The condition under which this holds is called regularity condition

$$E\left[\frac{\partial \ln p(x|\theta)}{\partial \theta}\right] = 0$$

The lower bound isself is defined in terms of Fisher information matrix.

$$\mathrm{J}_{k,i}(\theta) = -E\left[\frac{\partial^2 ln\, p(x|\theta)}{\partial \theta_k \partial \theta_i}\right]$$

This is the matrix that has individual components on k-th row and i-th collumn evaluated as a second order derivative of the logarithm of the obsevation model, then we evaluate the mean value over all random influences which are in the system. If we want to the estimator of i-th component of the vector θ, then this expression would be lower bounded in variance by the component sitting on the main diagonal on the i-th position, which is calculated from the inversed Fisher information matrix. So if we have multiple parameters we can calculate the lower bound of variance for each of them.

$$\mathrm{var}(\hat{\theta}) \geq \left[\mathrm{J}^{-1}(\theta)\right]_{i,i}$$

This holds for deterministic parameter. In case of random parameter we need to make the same operation with conditioning as in the case of MSE and bias with random parameter.

2.5. Sufficient statistics

Sufficient statistics is a core idea which is at beginning of any derivation of any synchronizer or estimator. The idea is as follows let us assume the parameter θ observed through observation x. Let us also assume that we build estimator. The question is if we can take the observation $\overrightarrow{x}$ and perform some function $T(.)$ to get vector $\overrightarrow{y}$ and of course to build some other estimator but such that those two estimators provide together the same value. The motivation is that we would like to have such a function which reduces the dimensionality.

So the estimator based on the original observation should give the same as the estimator which uses preprocessed observation. The $T(x)$ is sufficient statistics if the conditional observation model conditioned by $T(x)$ is equal to the observation model additional conditioned by the parameter itself

$$\hat{\theta}(x) = \hat{\theta}(T(x))$$

$$p(x|T(x), \theta) = p(x|T(x))$$

$p(x|T(x))$ does not depend directly on the parameter, because every piece of information about the parameter is already in $T(x)$. There is no additional piece of information that would be needed and that means that everything about the parameter θ is contained in $T(x)$ and then it is equal to the situation where we would be conditioning the observation model additionally by the parameter itself. Unfortunately there is not a systematic way how to derive $T(.)$ which would be fulfilling same goals. The only way we can do it is procedure where we try some function and then we verify whether this function is sufficient statistics. This is done by Nyman-Fisher Factorization theorem.

$$\exists g, h \;:\; p(x|\theta) = g(T(x), \theta)h(x)$$

The function $T(x)$ is sufficient statistics if and only the observation model can be factorized in two functions $g(.)$ and $h(.)$. That function which contains the parameter θ contains the observation only wrapped up in the sufficient statistics and the part that does not fit into the wrapping sufficient statistics must not contain the parameter θ.

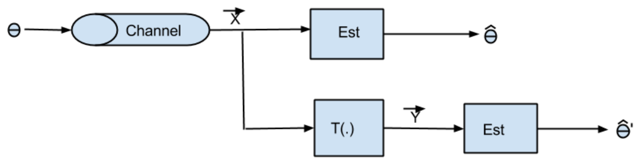

Figure 2.3.: Sufficient statistics

2.6. Estimator equation and solver

Most of the mathematical formulations of the estimation problem finishes in the form to find the minimum or argument of the minimum of the function ρ or find the argument of the maximum of some function ρ of the parameter of the interest. Finding minimum is equivalent to finding one minus function which was originally supposed to be a maximum. That makes the criterion in form

$$\hat{\theta} = arg\, min\, \rho(\check{\theta})$$

or

$$\hat{\theta} = arg\, max\, \rho(\check{\theta})$$

This equation can be solved by defining the first derivative and then let that first derivative be zero and find the solution of that equation. And of course depending on whether it is maximum or minimum the second derivative should be negative or positive.

$$\mu(\hat{\theta}) = \rho(\hat{\theta}) = \frac{\partial \dot{\rho}(\theta)}{\partial \theta} \mid_{\theta=\hat{\theta}} = 0$$

We have several ways how to solve this expression. The first one implies directly solving it. It is simply called a direct solution or a closed-form expression

$$\hat{\theta} = \hat{\theta}(x)$$

and because there is no need for any feedback in contrast with the other solution we call that a Feed-Forward or single-shot, because it is simply enough to know what is the observation x and then we calculate the solution and it is done so in single step. This solution is straightforward but is strongly depends on our capability of finding the solution of Feed-Forwards type of the solver. Unfortunately it exists only in rare cases.

In all other cases we need to find another form of solution and another form of solution is relying on the iterative solver. The iterative solution is trial and error approach. The principal is to maximize or minimize the objective function, so when we start somewhere and we find out that the function μ (first derivative of objective function) is nonzero we need to correct towards the lower or higher values depending on μ being negative of positive number, provided we have the knowledge about utility function having maximum or minimum extreme. This type of solving depends of good behavior of function μ which is called and indicator function, because it indicates the direction we should correct the next step in solution.

Mathematical formulation of the problem can be divided in two way, for discrete time and continuous time. In discrete time let us assume that a current estimate at time k. So at time k we are trying to find how should we correct the estimate based on result of indicator function. We are moving in relative way against the current guess. The question is how much should we correct the initial estimate. For that reason we simply define the operator G that can be any number.

$$\hat{\theta}'[k+1] = \hat{\theta}'[k] + G[\dot{\rho}(\hat{\theta}'[k])]$$

At most simple case the operator G can be a unity. That means taking the exact value we were obtaining by indicating function. If the value G is very small that means the correction is very small and shift against previous guess will be some very small scaling value then we will be stepping towards solution in small increments, but we can be sure that it converges. In other case of way big value it can happen that we will be diverging from correct solution.

3. Synchronization

3.1. System model

The purpose of this section is to find out the likelihood function describing our system from the perspective of the variable we are interested in. Once we got the likelihood function then we apply the knowledge of the estimator on the particular case.

3.1.1. Digital modulation signal

We will assume that our modulation is general modulation

$$s(t) = A\sum_{n} h(d_n, \sigma_n, t - nT_S)$$

with modulation function $h(d_n, \sigma_n, t - nT_S)$.
It could be expanded into the complete basis

$$A_S = \{\{\zeta_k(t - nT_S)\}_{k=1}^{N}\}_n$$

We also assume that modulation is Nyquist one, meaning no inter-symbol interference.

$$\int_{-\infty}^{\infty} h(d_n, \sigma_n, t - nT_S)h * (d_{n'}, \sigma_{n'}, t - n'T_S)\mathrm{d}t = 0, \quad for\, n{\neq}n'$$

3.1.2. Linear AWGN channel model

Let the channel be linear channel model so the received signal with delay shift, phase shift and attenuation is

$$x = \alpha e^{j\Psi} s(t - \tau_0) + \omega(t)$$

all the variables in the model are assumed to be time dependent. The Gaussian noise is linear model of complex envelope of the Gaussian noise.

$$P_{S_\omega}(f) = 2N_0$$

The delay of the model is split into the fractional part and the integer part in terms of the integer number of the symbol duration.

Amplitude itself is random parameter with unknown PDF. For the phase we introduce so called restricted phase. This is the phase including the frequency offset. In digital communication we always need to consider presence of the non-zero frequency offset so we cannot assume that there is zero frequency offset.

3.1.3. Equivalent channel model

Assuming observed signal to be

$$x = u(t) + \omega(t)$$

where useful signal is

$$u(t) = \alpha e^{j\Psi'} s(t - \tau_0)$$

unrestricted phase is given

$$\Psi = 2\pi f_s t + \varphi$$

and delay

$$\tau = \tau_s + \tau_f T_S$$

we can assume that all the parameters except for the unrestricted phase has the zero order dynamic model that means there is no drift or the drift is very small. In case of unrestricted phase we must assume that the dynamic model is first order meaning we presume significant drift.

3.1.4. Likelihood function

Assuming additive Gaussian channel we have input-output relationship

$$x(t) \approx \sum_n \alpha_n e^{j\varphi_n} e^{j2\pi f_{s,n} t} A h(d_n, \sigma_n, t - \tau_{0,n} - nT_S) + \omega(t)$$

then we expand the signal into the ortogonal basis that is complete with respect to useful signal

$$A_u = \left\{ \{\xi_{n,k}(t)\}_{k=1}^{N} \right\}_n = \left\{ \left\{ e^{j\varphi_n} e^{j2\pi f_{s,n} t} \xi_k(t - \tau_{0,n} - nT_S) \right\}_{k=1}^{N} \right\}_n$$

then we complement that with the part with respect to the Gaussian noise

$$A_x = A_u + A'$$

The resulting likelihood function is

$$\Lambda(\alpha, \varphi, f_s, \tau_0, D) = e^{-\frac{1}{2N_0} \int_{-\infty}^{\infty} |x(t) - u(t, \alpha, \varphi, f_s, \tau_0, D)|^2 \mathrm{d}t}$$

expanding into the constellation space we get

$$\Lambda(\alpha, \varphi, f_s, \tau_0, D) = e^{-\frac{1}{2N_0} ||x - u(\alpha, \varphi, f_s, \tau_0, D)||^2}$$

and the final form after expansion, the likelihood function which is suitable for everything which is related to all parameters is

$$\Lambda(\alpha, \varphi, f_s, \tau_0, D) = e^{-\frac{1}{2N_0}\sum_n \sum_{k=1}^{N} R[x_{n,k}(\varphi_n, f_{s,n}, \tau_{0,n}) u^*_{n,k}(\alpha_n, d_n)] - \frac{1}{2}\sum_{k=1}^{N} |u_{n,k}(\alpha_n, d_n)|^2}$$

So the knowledge of this likelihood function is enough to design anything we want on the receiver side if the channel is the linera one with the delay, rotation and amplitude shift.

As a sufficient statistics we can do the expansion of the continuous value signal in to the samples if the signal is bandwidth limited.

3.2. Frequency offset Synchronization

3.2.1. Synchronizer classification

Typical division is if the frequency offset is greater or equal of 10 percent of the symbol timing, then we cannot rely on anything else, that means all algorithms must be self-contained algorithms that means either Non-Data Aided (NDA) or Data Aided (DA) which means we know the transmitted data (for example pilot of the frame). On the other side if it is lower than 10 percent of symbol timing we are in tracking mode, that means the decoder is running and we can have Decision Directed (DD) algorithm, we can have Symbol Timing Aided (STA) algorithms, because the symbol timing is of course running. So the top level hierarchy is Symbol Timing Aided and Non-Symbol Timing Aided (NSTA).

3.2.2. Symbol Timing Aided algorithms

Everything can be time dependent (phase, frequency shift, delay). In model we assume that changes are slower than symbol period. Our interest is to get development of the estimate of the parameter in time. In order to do that we can develop a dynamic model for a desired parameter, for example using ARMA model, which is quite complicated. However we usually simplify the situation by assuming that the parameter is piecewise constant. The advantage of this approach is that we do not need to use any dynamic model for the estimator which would have the observation constrained by the width of the window. Of course if we do the estimate piecewise per blocks we would get set of values for indices of parameter and we would be completely neglecting the mutual dependence between the blocks. So we get the mutual dependence between the blocks back in to the game by two mechanisms.

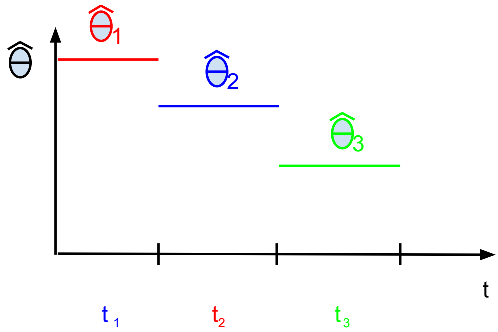

Figure 3.1.: Development of the estimate of the parameter in time

The first solution (feed forward) is following the processing of individual blocks by post processing so we simply take those per-block estimates and we post process them for example by moving average smoothing.

The second general approach is that if our algorithm for solving the inner block estimate if of a feedback type, we can build the slow smoothing dynamic in to the feedback algorithm by adjusting the loop filter.

First estimator is symbol timing aided and data or decision directed so the likelihood function is

$$\Lambda(\alpha, \varphi, f_s, \hat{\tau}_S, \hat{Q}) = a_1 e^{\frac{1}{N_0}\sum_{n=0}^{W-1}\Re[x_{n,1}(\varphi, f_s, \hat{\tau}_S)u^*_{n,1}(\alpha, \hat{q_n})]}$$

where

$$u_{n,1} = \alpha A \hat{q_n}$$

This likelihood function depends on several parameters, some of them are assumed to be given. As said this is data aided or decision directed so $\hat{Q}$ will be given. The data aided means that we are absolutely sure what are the data for example we agree with transmitter side what would be the pilot signal. The decision directed means that we feed the estimator with the decision. Which means that phase φ needs to be eliminated from likelihood function.

where

$$x_{n,1}(\varphi, f_s, \hat{\tau}_S) = e^{-j\varphi}\int_{-\infty}^{\infty} x(t)e^{-j2\pi f_s t}h*(t - \hat{\tau}_S - nT_S)\mathrm{d}t$$

is matched filter output including the phase rotation and same output but without the phase rotation is

$$z_n(f_s, \hat{\tau}_S) = \int_{-\propto}^{\propto} x(t)e^{-j2\pi f_s t}h*(t - \hat{\tau}_S - nT_S)\mathrm{d}t$$

then the likelihood function would be

$$\Lambda(\alpha, \varphi, f_s, \hat{\tau}_S, \hat{Q}) = a_1 e^{\frac{1}{N_0}\Re[e^{-j\varphi}\sum_{n=0}^{W-1} z_n(f_s,\hat{\tau}_S)\alpha A \hat{q}_n^*]}$$

If we take only real part of the considered expression we get

$$\Lambda(\alpha, \varphi, f_s, \hat{\tau}_S, \hat{Q}) = a_1 e^{\frac{1}{N_0}\cos(\beta-\varphi)}$$

In case of elimination of phase we take the likelihood function and integrate it over all phases

$$\Lambda(\alpha, \varphi, f_s, \hat{\tau}_S, \hat{Q}) = \frac{a_1}{2\pi}\int_{-\pi}^{\pi} e^{\frac{1}{N_0}\cos(\beta-\varphi)}\mathrm{d}\varphi$$

which leads to Bessel function of zero order and the amplitude of summation of $z_n(f_s, \hat{\tau}_S)$.

Then we can write the ML estimator which is one that maximazes the utility objective function which is

$$\hat{f}_s = arg\,max_{\check{f}_s}|\sum_{n=0}^{W-1} z_n(\check{f}_s, \hat{\tau}_S)\alpha A \hat{q}_n^*|^2$$

3.2.3. Non-Symbol Timing Aided Algorithm

In this case we do not have available symbol timing information. Typically these algorithms are used when we are acquiring the synchronization at the very beginning of data receiving process.

First case is Data Aided algorithm that means algorithm which relies on the knowledge of the data that were transmitted. This is typically part of the transmitted frame. Still we do not have the information on the symbol timing so we have two ways how to deal with this.

Option one is standard elimination of symbol timing from the likelihood function.

The second option is to take a look what is the shape of the likelihood function from the perspective of the symbol timing. If the shape of the likelihood function would be more or less constant we do not need to do the elimination explicitly. The estimation is given then

$$\hat{f}_s = arg\,max_{\check{f}_s}|\sum_{n=0}^{W-1} z_n(\check{f}_s, 0)|^2$$

3.3. Carrier Phase Synchronization

3.3.1. Synchronizer classification

First of all we need to distinguish whether we have got the Frequency Offset Aided (FOA) or Non Frequency Offset Aided (NFOA). In other words whether some other estimator already corrected the frequency error or not. If we know the frequency offset or the frequency offset was already corrected by other estimator then we can build the phase synchronizer, there is no residual frequency offset and our estimator could be of first order. In opposite case there is full frequency offset incoming to our phase estimator and phase estimator must cope together with the phase and frequency. then it must definitely be the second order.

The secondary type of classification is if we have the symbol timing available or not, eventually if we have data or decisions available or not.

Another thing which is common to all phase synchronizers is phase ambiguity. All standard codes and modulations are rotation and phase ambiguous, that means we cannot distinguish the situation where we have got the zero phase and the rotations of 360 degrees. So we need additional tools for resolving the phase ambiguity either by differential modulation or inserting rotation invariant signals.

3.3.2. Non-Symbol Timing Aided Algorithms

In case of NSTA algorithms the main idea of procedure is same as in frequency estimator. We need to eliminate the symbol timing from the likelihood function which is done in exactly same way as in frequency synchronization.

In the case of Non-Data Aided situation, several Ad-hoc algorithms is developed to make use of modulation removal. This is mostly done by Squaring loop or Costas loop.

3.3.3. Symbol Timing Aided Algorithms

First option is STA-FOA-NDA algorithm which means that frequency is given, phase is our target of the estimation, symbol timing is given and we need to make the elimination over data. Provided that our modulation is constant energy constalation the likelyhood function is again

$$\Lambda(\alpha, \varphi, \hat{f}_S, \hat{\tau}_S, Q) = a_1 e^{\frac{1}{N_0} \sum_{n=0}^{W-1} \Re[x_{n,1}(\varphi, \hat{f}_S, \hat{\tau}_S) u^*_{n,1}(\alpha, q_n)]}$$

if the likelihood function contains an exponential of the sum it is a product of individual per-symbol likelihood functions

$$\Lambda_1(\alpha, \varphi, \hat{f}_S, \hat{\tau}_S, Q) = \prod_{n=0}^{W-1} \Lambda_{1,n}(\alpha, \varphi, \hat{f}_S, \hat{\tau}_S, q_n)$$

where each per-symbol likelihood function is the exponential containing single expression for a single symbol.

$$\Lambda_{1,n}(\alpha, \varphi, \hat{f}_S, \hat{\tau}_S, q_n) = e^{\frac{\alpha A}{N_0} \Re[e^{-j\varphi} z_n(\hat{f}_s, \hat{\tau}_S) q^*_n]}$$

Data elimination means averaging over all data symbols, if we realize that each individual component of the likelihood function depends on single data symbol. If the transmitted data symbols are Independent Identical Distributed then the average likelihood function is product of averages

$$\Lambda_1(\alpha, \varphi, \hat{f}_S, \hat{\tau}_S) = \prod_{n=0}^{W-1} \Lambda_{1,n}(\alpha, \varphi, \hat{f}_S, \hat{\tau}_S)$$

$$\Lambda_{1,n}(\alpha, \varphi, \hat{f}_S, \hat{\tau}_S) = \frac{1}{M} \sum_{\{q_n\}} e^{\frac{\alpha A}{N_0} \Re[e^{-j\varphi} z_n(\hat{f}_s, \hat{\tau}_S) q_n^*]}$$

On the other hand when we decide that our algorithm will be Data Aided or Decision Directed we simply substitute the data where we originally eliminated the data from the likelihood function. Because there is nothing to be eliminated STA,FOA-DA/DD synchronizer has everything known so there is no elimination at all. We simply find the maximum of the likelihood function over the phase.

3.4. Symbol Timing synchronization

3.4.1. Symbol Timing Synchronizer

Synchronizers could be divided by several criteria. This chapter aims to introduce those basic classifications that are used for such division.

On top of expected division of synchronizers based on amount of information known a priori, there is whole other category which is called synchronous and Non-synchronous sampling, which is very specific to symbol timing synchronization.

We are supposed to take the samples of the signal. The time instant, when we are supposed those samples, is controlled from synchronizer. So let us assumed that we have received continuous time domain signal and our synchronizer instructs us when to take the samples. We must not forget that the real time scale and the instructed point where we are supposed to take samples, are not co-linear. The problem is that the entity we work with is time axis and we have got one real time axis and the other one which is the virtual one of received signal. Our synchronizer has the output that instructs us what are supposed to be time instances where we should take the case samples. t_p as a function of k does not need to be and is not linear function of k , because it depends on symbol timing which is function of deviation of clock and many others. The time scale of how we jump further or back depends on symbol timing estimate. So we would take the sample at that given point. We do that in natural way by having controlled Clock Generator, which is fed control signal and the resulting sample is really the one taken at the time instant where we are instructed to take it. This approach is really natural but is has big disadvantage that we need to mix digital and analog domain, first part must live in analog continuous time domain and it is typically fed from DSP where the algorithm of the estimator exists so there must be D/A converter which translated the result of the estimator which is digital one into the analog domain control signal. Also analog part of hardware is a bit more complicated.

So the alternative solution is non - synchronous or full digital solution. It means that we take the samples regularly at kT_p. We do not care where the sample is supposed to be taken, we simply take it at kT_p at linear scale and then from now on it is fully in digital domain of DSP. We recalculate those obtained samples to the ones we are interested in. This is provided by device called interpolator. The sampling rate of signal in interpretation is exactly the same one as the original but it is shifted to the point where we need the sample. Advantage of this structure is the fact that sampler is free running so it is quite simple device and the whole loop of synchronizer is closed inside of DSP so there is no need interact from DSP outside to the analog domain. Nowadays all symbol timing synchronizer are done in this way.

Now we have a look on how to build a device which performs the interpolation which is controlled in delay by input from synchronizer. Our task is: given the samples of continuous time domain signal taken at kT_p we need to get a new samples $s\acute{}$ which are taken kT_p plus some given shift τ and the shift is controlled from the synchronizer.

$$s\acute{}(\tau)[k] = s(kT_p + \tau)$$

It is useful to split the complete delay τ into the integer multiple of the sampling period m_p and fractional part of the sampling period where fractional part is from the range $\tau_p \in [0, T_p]$ and introducing μ_p as a normalized value of the fractional part

$$\mu_p = \tau_p / T_p \qquad \mu_p \in [0, 1]$$

So the ideal situation we are given continuous time domain signal reconstructed from from samples

$$s(t) = \sum_k s(kT_p)\mathrm{sinc}(\frac{t - kT_p}{T_p})$$

and we need to take samples in different point

$$s\acute{}(\tau)[n] = s(nT_p + \tau)$$

so after substituting integer and fractional parts of τ we are given

$$s\acute{}(\tau)[n] = \sum_i s[n + m_p - i]c_i(\mu_p)$$

where c_i is substitution of $\mathrm{sinc}(i + \mu_p)$ function.

So if we want to recalculate the samples then the operation which does this is linear filter with coefficients that depend on μ_p that means on the value we need to delay our samples. However we need to realize that μ_p is controlled by the synchronizer of symbol timing, It changes all the time and the coefficients must be calculated in real time in receiver. This operation runs on the sampling rate that means this is the fastest operation of the receiver, because the Sufficient statistics, unless we know the symbol timing, is a fractional space, that means several samples per symbol. So we need to design the algorithm that is efficient to be run through time which has a finite number of coefficients and still having acceptable approximation.

First we define the fidelity criterion which in our case is the power of the error signal

$$\bar{P}_{\triangle}(\mu_p, \{c_i(\mu_p)\}_i) = \mathrm{AvE}[|s_i\acute{}[k] - s\acute{}[k]|^2]$$

The first brute force solution is simply to take the filter and truncate the sinc function for finite number of coefficients for left hand side and right hand side $c_i(\mu_p) = \mathrm{sinc}(i+\mu_p)$, $i \in \{-K, \ldots, (K-1)\}$

$$s\acute{}(\tau)[n] \approx \sum_{i=-K}^{K-1} s[n + m_p - i]c_i(\mu_p)$$

The problem with this brute force implementation is that the sinc function decays quite slowly and we still need relatively huge number of coefficients to produce reasonable fidelity of the approximation.

Somewhat better solution, which targets only the fidelity of the approximation, is to forget the fact that c_i were originally supposed to be the samples of sums of XOR function and let us try to optimize those coefficients in such a way that we minimize the mean error power of the approximation. So for finite number of coefficients we want to have optimal coefficients which are minimizing the mean error power

$$\{c_i(\mu_p)\}_{i=-K}^{K-1} = arg\ min \frac{\bar{P}_{\triangle}(\mu_p, \{c_i(\mu_p)\}_i)}{\bar{P}_s}$$

However we still did not address the problem of calculating those coefficients in a real time so it is very complicated optimization procedure which needs to be run for each individual value of μ_p.

The second solution, which targets the simplicity of the calculation of the coefficients, is to assume that those coefficients are piecewise polynomial approximations of the ideal interpolator. So this is the solution where on the other side we do not care about the fidelity of the approximation but we target the simplicity of the calculation. Of course we can combine both approaches and take polynomial approximation and optimize the coefficients in such a way that it minimizes the mean error power of the approximation. Result would provide good fidelity but still be relatively simple to calculate.

Another thing that is very specific for symbol timing synchronization is evaluation of first derivative. As mentioned above, whatever solver we use for solving the utility function (either feed-forward or feed-back) we must calculate first derivative. There are two possibilities.

First one is doing that in analytic way. Whatever function is the objective function we still need to calculate first derivative. That is obtained by filter called Derivative Matched Filter. It is similar to matched filter but instead of basis function there is a first derivative and the output of this filter is the one which contains the derivative over the delay. So we can obtain the first derivative of the objective function by simply changing the matched filter which is used for calculating this utility function and that filter is now Derivative Matched Filter.

Second possibility is to calculate first derivative as an approximation. It is based on replacement of the derivative by finite difference and because this finite difference contains the value of the utility function taken a bit later or bit early it is called Early-Late (EL) type of algorithm. We do not define what the utility function should be, it can be minimum means square error, maximum likelihood, ad-hoc etc.

$$\frac{\partial\rho(\tau_S)}{\partial\tau_S} \approx \frac{\rho(\tau_S + \triangle\tau_S) - \rho(\tau_S - \triangle\tau_S)}{2\triangle\tau_S}$$

3.4.2. Carrier Aided algorithm

By carrier aided algorithm we mean that the phase and frequency is available from outside. The further division is again whether the algorithm is Non-Data Aided or Data Aided.

In Non-Data aided we will eliminate over data Q and our target is τ_S. Equation for likelihood function remains the same. Even the elimination remains the same, the likelihood function is product of individually eliminated likelihood functions over the individual data q_n.

$$\hat{\tau_S} = arg\, max_{\check{\tau_S}} L_1(\alpha, \hat{\varphi}, \hat{f_S}, \check{\tau_S})$$

where L_1 is Log-likelihood function.

The difference is that in the carrier phase synchronizer we assume that τ_S is available and we search for φ, now exactly vice versa, we assume that φ is available and we search for τ_S. But the likelihood function now looks exactly the same as for the phase synchronizer.

In Carrier Aided, Data Aided/Decision Directed Algorithm, there is no elimination at all, because we provide all parameters and even the likelihood function stays the same. The only question is how to make the solver for this function.

$$\hat{\tau_S} = arg\, max_{\check{\tau_S}} \left\{ \Re[e^{-j\hat{\varphi}} z_n(\hat{f}_s, \tau_S)\hat{q}_n^*] \right\}$$

The solver would need to calculated derivative which is done by derivative matched filter or by Early-Late approximation.

3.4.3. Non Carrier Aided

In this class we do not assume the knowledge of frequency and the phase.

In the case of Non Carrier Phase Aided, Frequency Offset Aided, Non Data Aided algorithm we obviously eliminate over phase which results in Bessel function. Before we do the elimination over the data we need to do approximation of Bessel function which works only for low SNR. The elimination of the data leads to the expression which contains additive and scaling constants which are not important but also the sum of magnitude square of the matched filter output with variable τ_S which is the one we look for the maximum.

$$\rho(\hat{f_S}, \tau_S) = \sum_{n=0}^{W-1} |z_n(\hat{f_S}, \tau_S)|^2$$

3.5. Frame Synchronization

In frame synchronization we cannot use Non Data Aided algorithms, because unless we know the pattern in signal we cannot say where it begins. So all the algorithms are Data Aided which is equivalent to transmitting the pilot. We typically transmit the known sequence which is buried either in the noise or in random payload data. If we set up standard procedure of Maximum Likelihood delay estimate of the known sequence of the signal, we would get, because the noise is additive and Gaussian, a correlation receiver. This is the consequence of the fact that the likelihood function for Gaussian noise contains square of the difference of the two signals.

The receiver would receive the signal and correlate it with the known sequence on the receiver side. The second step of discussion is what should be the optimal pilot sequence which would provide the best performance of this estimator, provided the performance is measured in mean square error of the estimation. The answer is that the pilot sequence should have the Dirac delta type of the correlation function. We have huge number of sequences which are mimicking more or less perfect shape of correlation function but all of them have one disadvantage. The desired correlation property holds only for cyclic correlation. The price we pay is that we must emulate the cyclic correlation. The way how we do that is simple, instead of one period we must send two periods of the signal. It is typically done that we send the whole period of pilot signal and then half period and then of course on the receiver side if we correlate that with the single period of the sequence. What we calculate is something that emulates cyclic correlation function. The result is that the sequence occupies twice as much space in comparison with the perfect aperiodic sequence.

Part II.

Thesis Contribution

4. Interface protocol

4.1. Goals

The objective of introducing Interface Protocol is to create a container for PHY solutions specific to the Wireless PHY Layer Network Coding based communication. It should be generic enough to serve for all possible solutions and scenarios using a modular implementation.

- The functionality must provide frame synchronization that properly addresses multiple cloud stages
- Slot synchronization for a packet based operation
- Hierarchy signalisation carrying the information about network graph fragments and other relays operations
- Node-local Channel State Estimation (CSE)
- Actual information carrying payload (PL) codeword transfer.

4.2. Assumtions and constrains

When we are considering Ad-hoc network with no central authority or base-station to provide pilot beacon, we must reflect that on the requirement of scheduling, synchronization and resource management. All those functionality has to be directly on PHY since we are trying to avoid the usage of upper layers. Specifically if we are building network based on non-ortogonal sharing because from the real HW implementation point of view,we are approaching the constraint of half-duplex of Tx/Rx. However there is at least one thing that can be simplified. Network synchronization in terms of the frame timing and the carrier frequency is way less restricting. Such synchronization algorithms could by applied without consideration of hierarchical stage scheduling meaning that all receivers are in sync. Advantage of this approach is the possible separation of designing, where implementation of synchronization is done in different stages than the WPNC related aspects, resulting in important simplification of such task because synchronization of symbol timing and carrier frequency could be done by ortogonal resources. Then the all-including hierarchical joint WPNC and synchronisation design can be viewed as a specific advanced solution. But with that said, It cannot be forgotten that initial idea of the interface is to be connected to hierarchical structure of the network. This feature is essential from the WPNC coding and processing point of view and therefore it makes it mandatory from this viewpoint.

4.3. Frame structure

The structure of frame was proposed in [11] as follows. Main entity is called Superframe that contains forward and backward frames used for bidirectional communication between sources and destinations or vice-versa. The reason for such design is to define Stage Scheduling inside frames by slots.

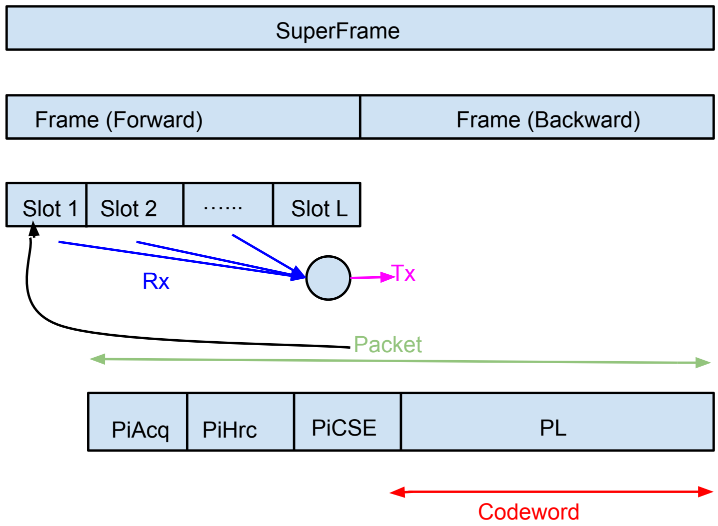

Figure 4.1.: Frame structure

In fact the slots themselves are used for scheduling of the stage. If viewed in simplified case, there is L slots and in case of the activity of Tx node in stage l, slot number l ($l \in \{1, \ldots, L\}$) is assigned to this node. The node which is active (Tx) at the stage l can collect all Rx packets from stages $1, \ldots, (l-1)$. By this definition, the first slot $l = 1$ is reserved for source S. In this design we are able to track and follow the information propagation through the network. The amount of delay is given by the length of the frame which is given by number of stages unlike for classical routing where the delay is also dependent on the total number of the nodes.

4.4. Pilots

4.4.1. General requirements on pilots

Because in WPNC enviroment the payloads share the medium in a non-ortogonal manner, we assume that even the pilots should be able to work in non-ortogonal way. Those pilots are supposed to provide all information that are needed in given stage sharing. Only way of simplification is using orthogonal pilots for synchronization to be decoupled from actual

PL data performance.

The fact that codeword might be reliably decodable only at final destination forces us to separate pilots from actual payload which might not be possible to decode under relay operation. We expect the pilots to serve for synchronization and signalization functionality we need in WPNC PHY layer processing and have very small amount of information in contrast with potentially huge length of codeword we would need for reliable decoding, creating large overhead and latency in case of pilot and payload cooperation.

4.4.2. Acquisition Pilot (PiAcq)

PiAcq serves for three main purposes [11] :

- Identification of the node. It must have a unique signature per node.
- Identification of Tx activity. It serves other nodes for decisions of the radio visibility of the given node and also as a radio beacon for for distributed stage/slot synchronization.
- Synchronization of frame/slot. The pilot must have delay resolution capabilities. The frame synchronization corresponds to the Tx activity stage slot. The slot synchronization means the alignment of the packet inside the slot.

4.4.3. Hieratchy Pilot (PiHrc)

PiHrc is a pilot signal very specific to WPNC. It should carry the information about hierarchical functions of the node itself and all preceding nodes. It means [11]

- identifying the contributing nodes, for example: (R2,R1); R1(SA,SB); R2(SB,SC).
- It must identify the hierarchical functions used by the node operations, e.g. ID of HNC map, type of relay operation.

4.4.4. Channel State Estimation Pilot (PiCSE)

PiCSE pilots serve for node-local CSI estimator. However there is an important phenomenon which distinguishes this from the classical point-to-point situation. The channel parametrization contains relative (nonlinear) and absolute (linear) part in the multisource setup (see the hierarchical pilot example in the following text). The WPNC receiver operations are strongly dependent on the relative channel coefficients. Therefore the pilots must be optimized for this. This includes the design of the pilots resistant to the non-orthogonal superposition while still providing a high quality relative channel fading coefficient estimates. [11]

4.5. Resources for pilots

In proposed design both ortogonal and non-ortogonal pilots are assumed. Pilots that are using ortogonal resources are not that much efficient but on the other side, it makes the design more robust and easier to implement.

- Ortogonal Pilots - each node has its dedicated ortogonal resource, It can be Time Domain slot, OFDM sub-carrier or Code. The difference between classical scheduling where each link has its own resource, we now dedicate resources to nodes, so final number of needed resources corresponds to number of nodes.

- Non-Ortogonal Pilots - This kind of pilots are used when we expect Rx to receive superposition of pilots that are not ortogonal to each other in any known way. Such pilots we call Hierarchical Pilots and are used to provide only hierarchical related information. in case of Channel State Estimation it would be

$$u = h_A s_A + h_B s_B$$

where we are not interested in separate channel responses h_A, h_B but only in $\tilde{h} = h_A$ (common channel) and $h = h_B/h_A$ (relative channel)

4.6. CAZAC codes

Constant Amplitude Zero Auto-Correlation sequence (CAZAC) is a periodic complex-valued signal with modulus one and out-of-phase periodic (cyclic) auto-correlation equal to zero. CAZAC sequences find application in wireless communication systems. The CAZAC code for space-time stream is generated by cyclically shifting the basic CAZAC code of a chosen length. Zero AutoCorrelation means that a CAZAC code is always orthogonal with its cyclic shifted versions. Major advantages of the code is: reduced inter-symbol interference, helps avoiding interferences between multiple antennas and lowers peak-to-average power ratio. As a result, CAZAC codes are regarded as optimum training sequence for channel estimation in MIMO-OFDM systems.One of the well known CAZAC codes with special properties are Zadoff-Chu sequences.

4.6.1. Zadoff-Chu seqeuence

A Zadoff–Chu sequence is a complex-valued mathematical sequence. As shown in figure 4.2, such sequence has constant amplitude of unity. Cyclically shifted versions of the sequence imposed on a signal result in zero correlation with one another at the receiver. Therefore belongs to CAZAC codes. A generated Zadoff–Chu sequence that has not been shifted is called "root sequence".

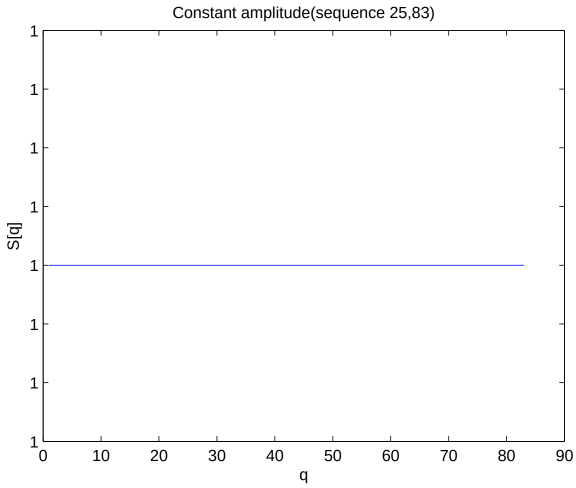

Figure 4.2.: Amplitude of sequence with length 83 and root 25

These sequences exhibits the mentioned property that cyclically shifted versions of itself are orthogonal to one another, provided, that is, that each cyclic shift, when viewed within the time domain of the signal, is greater than the combined propagation delay and multi-path delay-spread of that signal between the transmitter and receiver.

The complex value at each position n of each root Zadoff–Chu sequence parameterized by K is given by

$$c = \begin{cases} e^{\frac{jK\pi q^2}{L_P}} & When\, L_P\, is\, even \\ e^{\frac{jK\pi q(q+1)}{L_P}} & When\, L_P\, is\, odd \end{cases}$$

Where

$$q = 0, 1, \cdots L_{P-1}$$

and L_p is length of the sequence and K is called root and is relative prime to L_p.

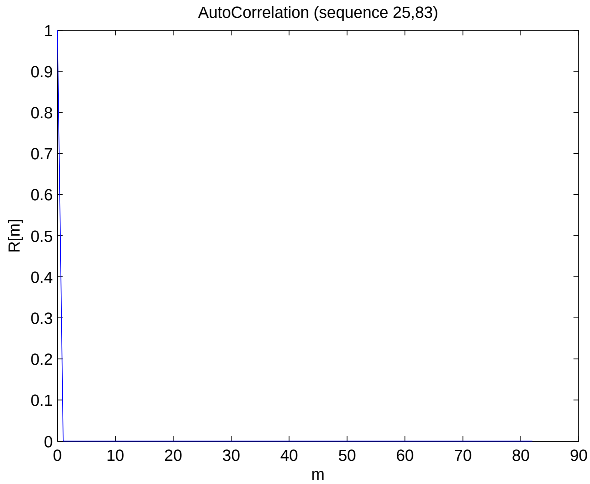

Figure 4.3.: Cyclic AutoCorrelation of sequence with length 83 and root 25

Properties of Zadoff-Chu sequences are :

1. If length of sequence is prime number, Discrete Fourier Transform of Zadoff-Chu sequence is another Zadoff-Chu sequence conjugated, scaled and time scaled

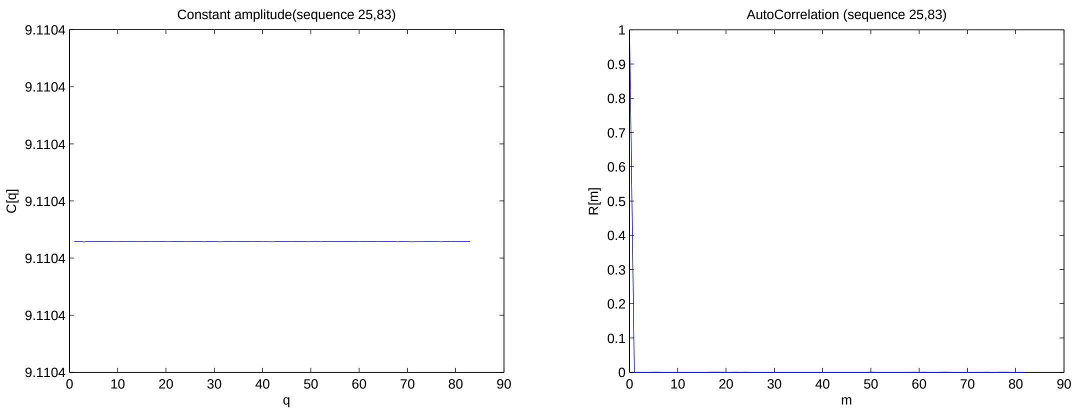

Figure 4.4.: Discrete Fourier Transform of Zadoff-Chu sequence

2. They are periodic with period L_P if $L_P$is odd.

$$c(q + L_P) = c(q)$$

3. The auto correlation of a prime length Zadoff-Chu sequence with a cyclically shifted version of itself is zero, i.e., it is non-zero only at one instant which corresponds to the cyclic shift

4. The cross correlation between two prime length Zadoff-Chu sequences, i.e. different values of K, is constant $\frac{1}{\sqrt{L_P}}$, provided that $K_1 - K_2$ is relative prime to L_P

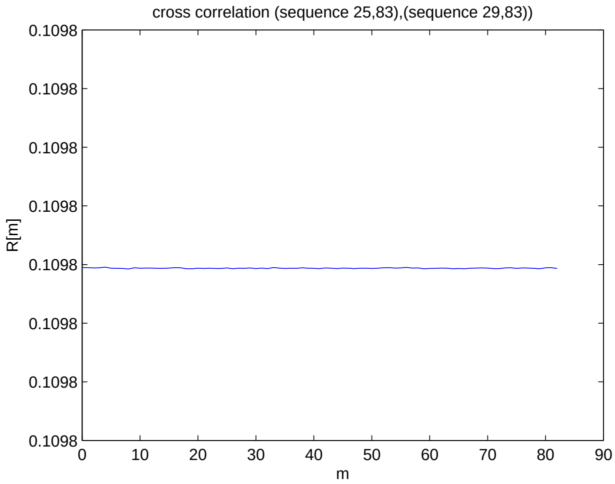

Figure 4.5.: Cross correlation of two sequences

It is worth noticing that if the condition of difference in two roots of sequences does not hold up, then we get cross correlation that does not have small correlation values nor constant value. Figure 4.6 shows two sequences where difference in root is not prime to their length. $\frac{1}{\sqrt{L_P}}$ equals to 0.109, but the graph shows that computed cross correlation between sequences 25,63 and 34,63(first number indicates root, second length of sequence) has peaks three times bigger than mentioned value, because difference between roots is 9 which is not prime to length of 63.

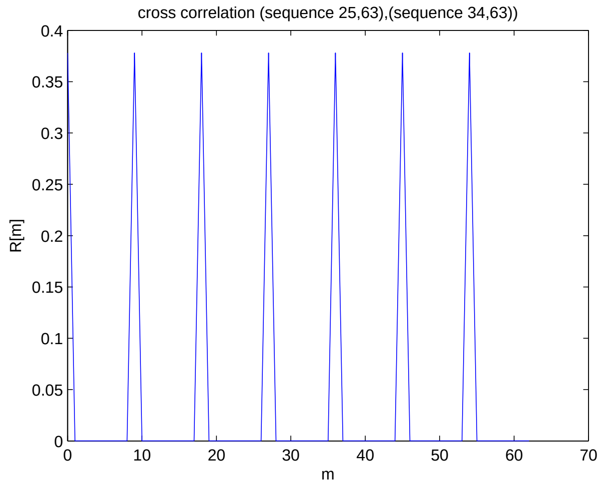

Figure 4.6.: Croess correlation of wrong two sequences

As mentioned before Zadoff-Chu sequences have very good anti-noise ability. The figure 4.7 shows amplitude of such sequence with Additive White Gaussian Noise (AWGN) of -10 dB signal -noise ratio (SNR). Red dashed line shows average value where as black line indicates its peak value.

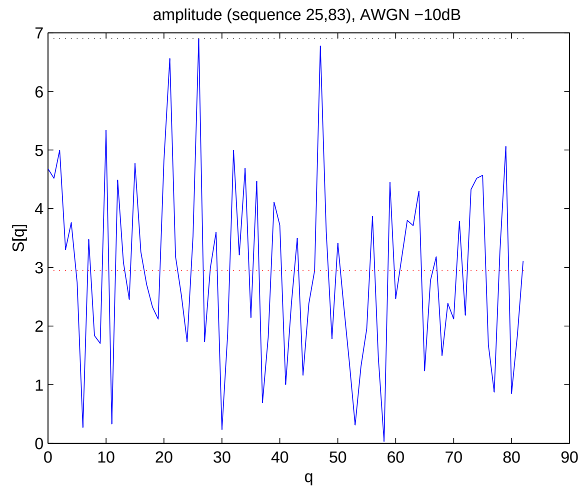

Figure 4.7.: Amplitude of sequence with AWGN

Figure 4.8 shows desired extraordinary properties of the cyclic auto-correlation even with AWGN. Peak value is still detectable even with a smaller SNR ration.

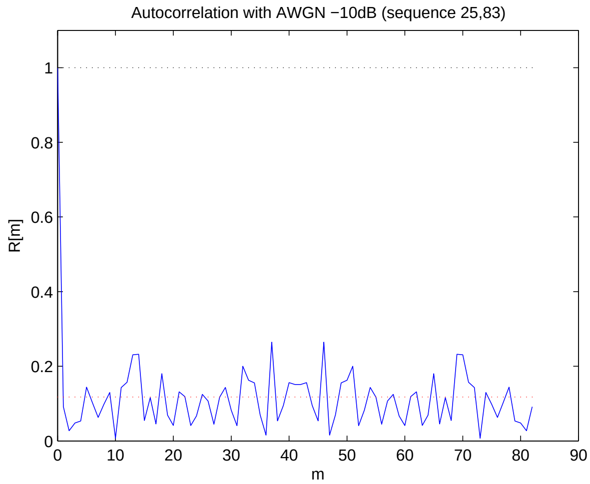

Figure 4.8.: Autocorrelation of sequence with AWGN

Anti-noise ability is noticable even in cross-correlation between two sequences where both are with AWGN.

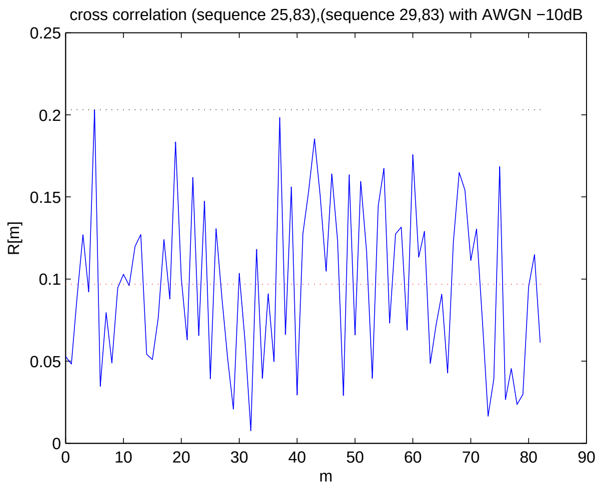

Figure 4.9.: Cross-correlation of sequence with AWGN

5. Implementation

Following Chapter describes the implementation of synchronization. The simulation was done by Ettus Research USRP, the scalable software defined radio (SDR) platform. This device was controlled via MATLAB. The modulation chosen for demonstration is QPSK with REC filter. Sampling frequency is set to 200 kHz and sampling 8 samples per symbol. Data are split into packets and transmitted in loop. Receiving side receives signal, which was then processed in “offline mode”. Several scenarios were investigated including classic point to point and leading to one receiver receiving simultaneously signal from two transmitters at the same time and performing synchronization of superposed signal of both transmitters.

An algorithm is proposed for synchronization of incoming signal and is also applied in mentioned scenarios. The channel estimator with ortogonal pilots is then proposed as well as applied in synchronization. Mentioned estimators for frequency offset and phase are then evaluated and compared to theoretical limits and their performance. Lastly an introduction to Non-Ortogonal Channel Estimation describes an idea of pilots for such performance and their usage.

5.1. Packet Synchronization

The synchronization of packets is based on idea of frame synchronization. We use data aided algorithm. As proposed in DIWINE project [11], the CAZAC sequences, namely Zadoff-Chu is being used as a pilot signals. These pilots are added to modulated data to complete packets. The important thing to mention is that the pilot signals that are transmitted have higher energy than data itself so that the correlation peaks are assured to be way above signal level of data. In tested case, we choose to transmit pilots with ten times more power.

Algoritmus 5.1 Making packet- adding unmodulated pilots to modulated data

```
for i = 0:nPacket-1
Pilot(:,i+1) = [cazac;cazac;tx_CE(:,i+1)];
end
```

So every transmitter has its own sequence that is ortogonal to others. This sequence as mentioned is repeated several times due to demonstrated good properties of cyclic correlation of CAZAC sequences.

Chapter 3.4.2 shows that correlator output can be used for ML estimation of time shift.

As shown in figure 5.1. The receiver contains set of correlators for every CAZAC sequence of transmitters. So when the incoming signal is correlated in all of those correlators, the correlated signal has the peaks in correlator that matches the right sequence and zero in all others.

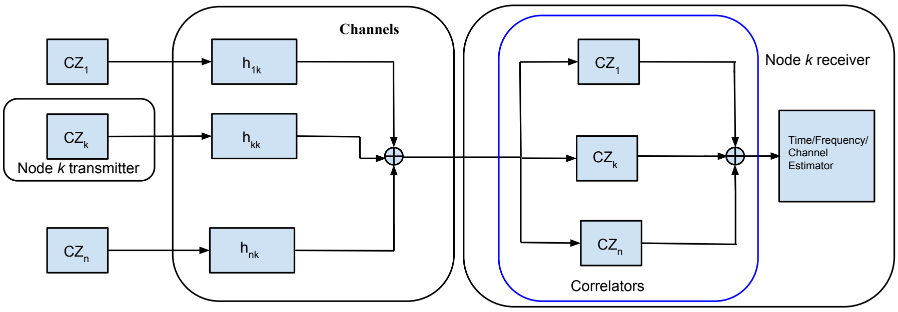

Figure 5.1.: Tranmitter-receiver Block diagram

The likelihood function of received continuous signal is parameterized by phase, frequency and time shift, so we take likelihood function

$$p(y|\tau, f, \varphi) = \alpha \cdot exp\left(\frac{1}{\sigma_n^2}\sum_n x(n)e^{-j\varphi}e^{-j2\pi fn}s*(n-\tau)|^2\right)$$
$$p(y|\tau, f, \varphi) = \alpha \cdot exp\left(\frac{1}{\sigma_n^2}R\sum_n [x(n)e^{-j\varphi}e^{-j2\pi fn}s*(n-\tau)]\right) \tag{5.1}$$

and assuming constant phase we can eliminate it from likelihood function knowing PDF (uniform one).

$$p(x|\tau, f) = \int_{-\infty}^{\infty} p(y|\tau, f, \varphi)p(\varphi)\mathrm{d}\varphi$$

So let the $y_k[m]$ be the received discrete signal, which enters the bank of K correlators, each matched to one of the n CZ sequences CZ_i (for $i = 1, \cdots, n$). The output of the k-th correlator is.

$$Rcv_k[l] = \sum_{m=0}^{N-1} y_k[m+l]CZ_k^*[m]$$

Then the Time offset estimation is described as

$$\tau_{kk} = arg\ max_l|Rcv_k[l]|$$

and is correct only if we assume phase to be constant so it can be eliminated.

MATLAB implementation of the algorithm is shown below. Incoming signal is correlated with matching sequence, then search for peaks in such correlation takes place. To ensure that right peaks are found, we restrict search for peaks in a way that those peaks must be greater than chosen ”MINPEAKHEIGHT” and be separated from each other at least number of ”MINPEAKDISTANCE” samples.

Lastly we know that each packet has 2 peaks, so we need to find where the right couple of peaks starts, which is solved by simple IF conditioning. Resulting value tau_est says for how many samples is start of the packet shifted

Algoritmus 5.2 Coarse estimation of Time offset

```
correlation=xcorr(sig,cazac);
[pks,locs] = findpeaks(abs(correlation),'MINPEAKHEIGHT',0.01,..
..'MINPEAKDISTANCE',110);
if(locs(1)+150<locs(2))
   tau_est=locs(2);
end
if(locs(1)+150>locs(2))
tau_est=locs(1);
  end
```

The corralated received signal is then shown in figure 5.2. The figure shows correlation of CAZAC sequence with the packet that has two CAZAC sequences as a pilot signal. There are two dominant peaks indicating shift of the packet.

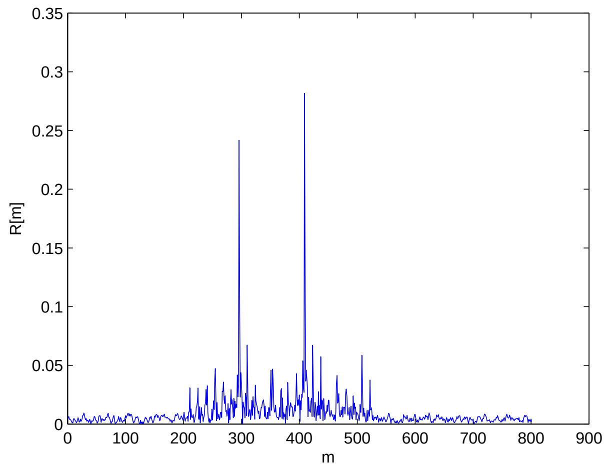

Figure 5.2.: Correlation of the packet

However, due to the existence of the carrier frequency offset (CFO). That makes the estimation of delay less accurate because now our estimate has certain error and we are not able to point ourselves to the beginning of the packet. We call this type of synchronization coarse one because we can estimate timing but with certain error but eventually with frequency offset big enough, the synchronization fails. The figure 5.3 shows that correlation peak is in correct position if the correlated sequence has no frequency offset, however we if shift the sequence in frequency, the peak indicating correct shift is attenuated and dominant peak shows up indicating wrong shift. On this figure the correct shift is 222 samples, which is correctly indicated by correlation function with zero frequency shift, however if we shift the signal for 1400 Hz, new peak, which is clearly dominant, shows up and falsely indicates time shift for only 208 samples instead of 222. This behavior is quite different from standard PN sequences, where with growing frequency offset, the dominant peak fades away and is buried in rest of the correlation with no other dominant peaks indicating false time shift, this behavior is shown in figure 5.4.

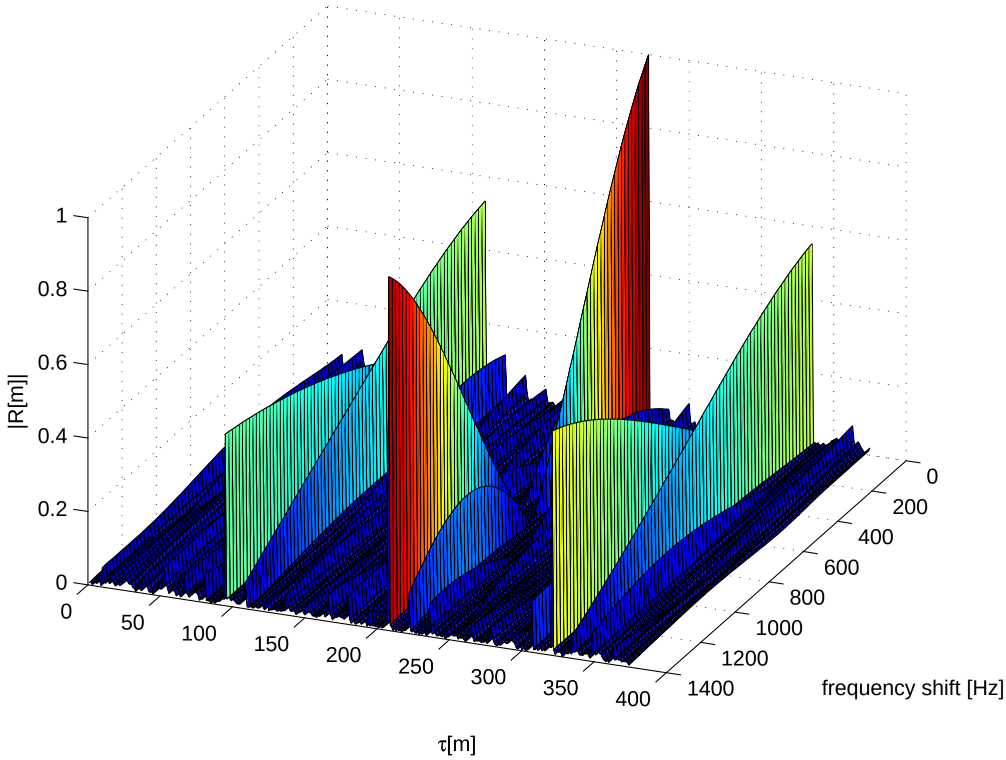

Figure 5.3.: Coarse timing synchronization of CAZAC sequence

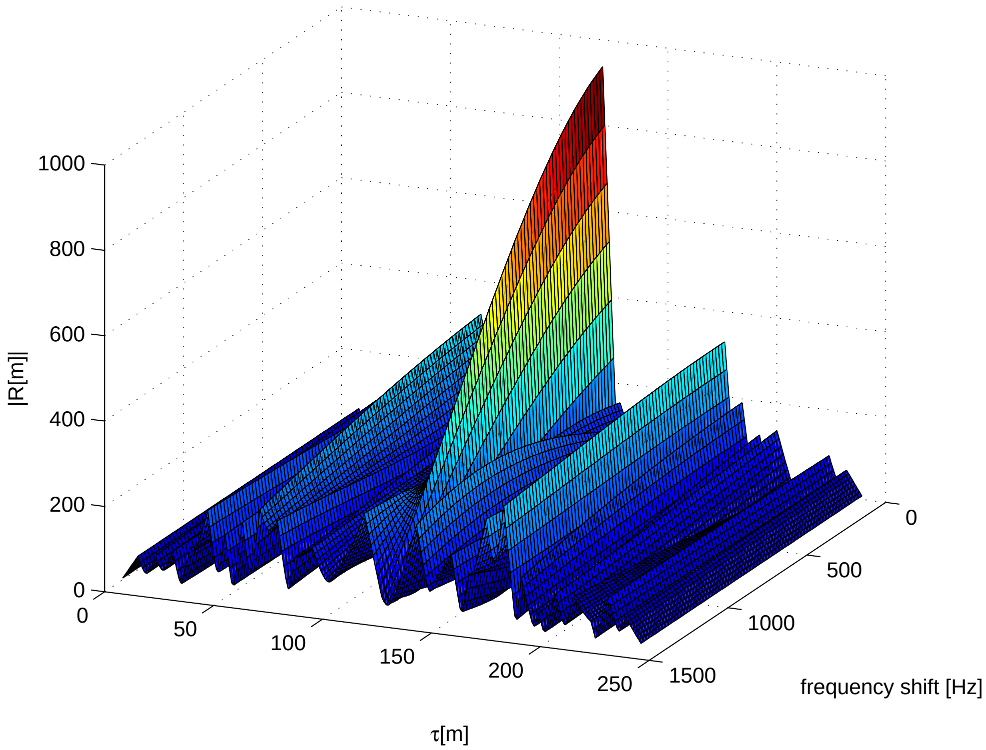

Figure 5.4.: Correlation function of PN sequence with frequency shift

The solution to this problem is to eliminate the CFO from likelihood function.

$$p(x|\tau) = \int_{-\infty}^{\infty} p(x|\tau, f)p(f)\mathrm{d}f$$

where

$$p(x|\tau, f) = \frac{I_0}{\sigma_n^2}\left[|x(n), e^{j2\pi fn}s*(n-\tau)|\right] \approx 1 + |x(n), e^{j2\pi fn}s*(n-\tau)|^2$$

We approximate the integral of the marginalized likelihood function by the sum over few neighbouring frequencies.

$$p(x|\tau) = \sum_{f_i} p(x|\tau, f_i)p(f_i)$$

Algoritmus 5.3 Elimination of Frequency Offset

```
for i = 1:length(data_rx)-pilot_len
   test(1,i) = data_rx(i:i+pilot_len -1)'*Pilot
end
for i = 1:length(data_rx)-pilot_len
  test(2,i) = data_rx1(i:i+pilot_len -1)'*(Pilot .*offset1);
end
for i = 1:length(data_rx)-pilot_len
   test(3,i) = data_rx(i:i+pilot_len -1)'*(Pilot .*offset2);
end
test = abs(test).^2;
data_no_CFO= sum(test);
```

Figure 5.5 shows correlation done by marginalization. Dominant peak shows the correct time shift with no frequency shit and with frequency shift is still the most dominant one and indicate shift for correct amount of samples.

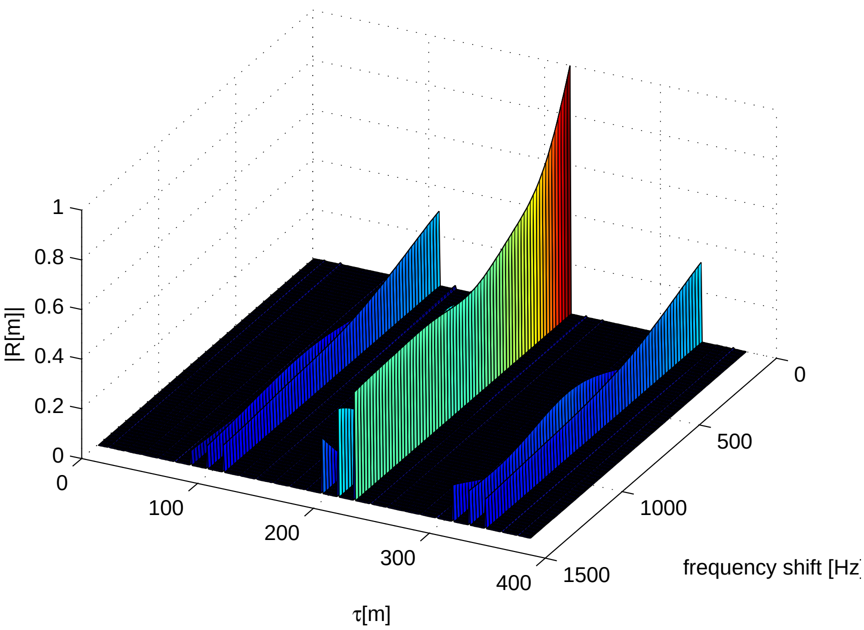

Figure 5.5.: Fine timing synchronization

5.2. Proposed Synchronization Scheme

Timing synchronization leads us to design of synchronizing procedure. Although fine timing synchronization let us correctly estimate beginning of packet, we loose all information about phase of the correlation peaks, because in process of marginalization we calculate absolute value of the square power. Information about phase of the correlation peaks is critical to the further estimation of the CFO and channel response, so in this case we are not able to use fine timing.

Assuming we are at the point of estimating phase offset, we have been provided the estimate of the time offset and the Log-likelihood function (5.1) is in the form

$$p(y|f,\varphi) = \frac{1}{\sigma_n^2} \sum_n R[x(n)e^{-j\varphi}e^{-j2\pi fn}s*(n)]$$

so the ML estimator is

$$\hat{\varphi} = arg\, max_{\varphi} \sum_n R[x(n)e^{-j\varphi}e^{-j2\pi fn}s*(n)]$$

Potentially good usage of this method is assumed to be in synchronization of larger packet block that are transmitted in the loop by source. In that case we only require timing synchronization and we do not need correlation peaks for any further estimations.

However in our case we need to settle for coarse timing estimate. Figure 5.6 shows block diagram of whole process of synchronization. We perform coarse timing estimate after which we can estimate and compensate the CFO. Once we do that there is no longer need for marginalization in order to get fine timing estimate. Finally we estimate channel response and compensate the received signal.

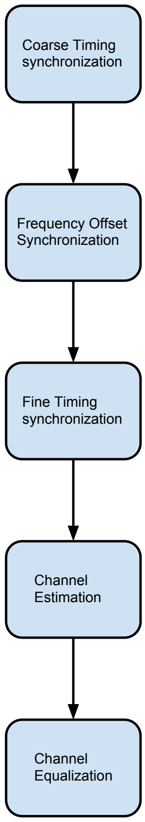

Figure 5.6.: Block diagram of synchronization process

5.3. Frequency Offset Synchronization

The frequency offset estimation is done with usage of CAZAC sequences. We know exactly what does the peak of correlation function look like. Every packet has two CAZAC sequences which gives us 2 correlation peaks when correlated with one sequence. By subtracting the difference of the phase of those two peaks and with known distance. We are able to estimate how fast the phase grows for two points that are supposed to have same phase which ultimately gives us frequency offset because frequency is derivative of phase.

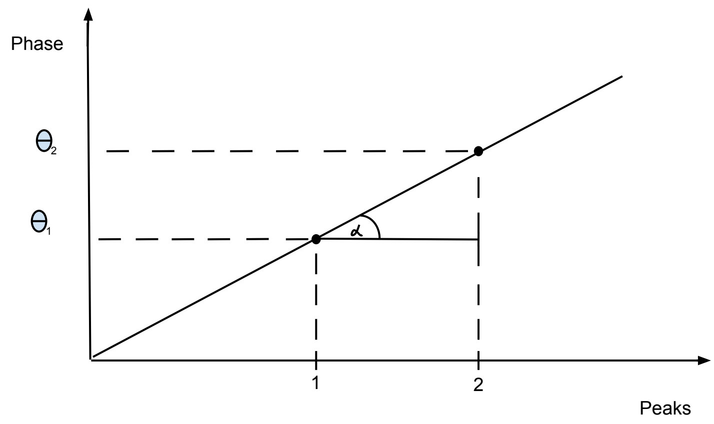

Figure 5.7.: Estimation of frequency by evaluating phase difference

We can calculate the angle h-th correlation peak

$$\theta_k^h = \Im(log(Rcv_k^h[l]))$$

which can be used to estimate $\widehat{\triangle\theta_k}$

$$\widehat{\triangle\theta_k} = \theta_k^{h+1} - \theta_k^h$$

The estimation of the phase assumes to have precise estimate of the frequency, so we can consider the phase to be constant. However, as shown in figure 5.8 if our estimate of the frequency has reaches certain error we cannot assume to have constant phase anymore. Which means the estimation works only for small $\triangle f$.

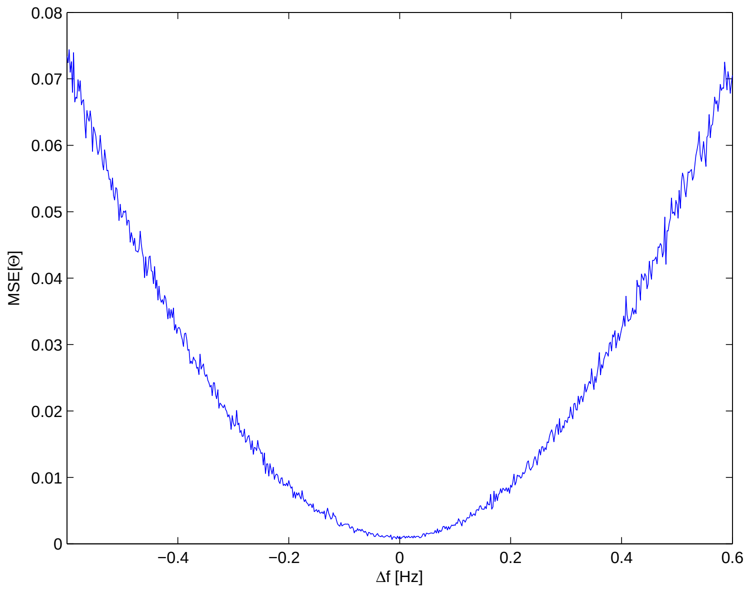

Figure 5.8.: dependence of MSE of the phase on frequency estimate error

The MATLAB implementation shown below is built around the idea that we divide the signal into individual packets where we estimate their frequency offset and compensate. We calculate the phase of each correlation peak and calculate the frequency offset estimate as a difference of two consecutive peak. The packet is then compensated by such offset.

Algoritmus 5.4 Algorithm of frequency offset estimation and compensation

```
Step= len_packet;
while 1
  start = tau_est+ o*Step;
  if length(correlation(start+1:end))<Step_1;
      break;
  end
  correlation_packet=(correlation(start:start+Step-1));
  packet=sig(start:start+Step_1-1);
  [peaks,locs]=findpeaks(abs(correlation_packet),'MINPEAKHEIGHT',0.01,...
...'MINPEAKDISTANCE',110);
  phase(1)=imag(log(correlation_packet(locs(1))));
phase(2)=imag(log(correlation_packet(locs(2))));
  offset_est=phase(2)-phase(1);
  offset_est=offset_est/(len_pilot*2*pi);
  offset(o+1)=offset_est;
  tt = 1:length(packet);
  tt = tt + start-1;
  tt = tt(:);
  compansated_signal(:,o+1)=packet.*(exp(j*2*pi*offset_est*tt));
  o=o+1
end
```

The only problem is that the phase is moving between the values of minus Pi and plus Pi, which ruins our estimation of growth if the phase jump from positive value to lower and even negative value.

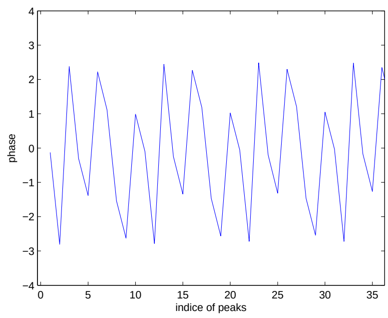

Figure 5.9.: Phase of correlation peaks

This is the reason we need to unwrap phase so that when value jumps back to lower values, there needs to be two Pi added.

Algoritmus 5.5 Unwraping of the phase

```
for(i=1:length(phase)-1)
  if(phase(i+1)+2*pi*clk<phase(i))
    phase(i+1)=phase(i+1)+2*(pi*(clk+1));
    clk=clk+1;
  else
    phase(i+1)=phase(i+1)+2*(pi*(clk));
  end
end
```

The resulting phase growth after phase unwrap is shown in figure 5.10

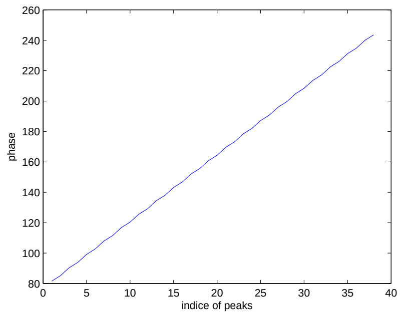

Figure 5.10.: Unwraped phase

5.4. Flad - fading channel Estimation

The channel estimation abuses that fact that we know exactly how how the correlation peak looked like before the signal was send by a transmitter. Provided that signal is considered narrow-band we can simplify the channel response

$$y(k) = x(k)h(k)$$

so by comparing correlation peak of incoming signal with theoretical one we get the channel response

$$\frac{y(k)}{x(k)} = h$$

which contains the information about the phase shift and magnitude attenuation. We can use this channel response to compensate received packet and get the original one.

$$x(k) = \frac{y(k)}{h}$$

Figure 5.11 shows the constellation diagram of the received signal after the synchronization is done

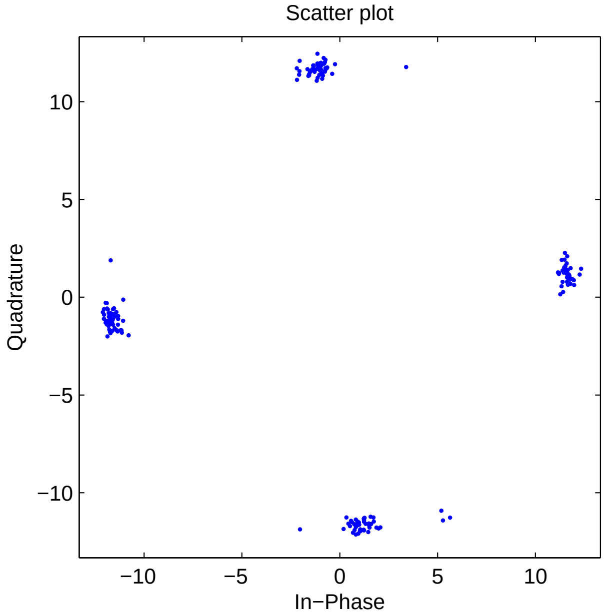

Figure 5.11.: Constellation diagram

5.5. Performance Limits and Evaluation

In this section we are considering received signal with noise to be

$$y_n = exp(j2\pi ftn)s(n-\tau) + \omega_n \tag{5.2}$$

Where f is carrier frequency offset and ω_n AWGN.

We derive the CRLB assuming perfectly decoupled problem with the received signal having either timing offset or frequency offset but not both simultaneously.

5.5.1. CRLB for Timing Offset

If we consider f to be zero, then y_n in the presence of noise is given

$$y(n) = s(n-\tau) + \omega_n$$

Denoting $p(y;\tau)$ to be the joint Probability Density Function for whole observation parameterized by τ, we derive from

$$p(y|\tau) = \alpha \cdot exp\left(\frac{1}{\sigma_n^2}\sum_n R[y(n)s^*(n-\tau)]\right) \tag{5.3}$$

Calculating natural logarithm and second order partial derivative with respect to τ and we get

$$\left[\frac{\partial^2 \ln p(y;\tau)}{\partial \tau^2}\right] = -\frac{1}{2\sigma_n^2} \cdot \sum_n R[(y(n)s^*(n-\tau))''] \tag{5.4}$$

and substituting $y(n)$

$$\left[\frac{\partial^2 \ln p(y;\tau)}{\partial \tau^2}\right] = -\frac{1}{2\sigma_n^2} \cdot \sum_n R[(s(n-\tau))''(s^*(n-\tau)) + (s(n-\tau))(s^*(n-\tau))' +$$

$$+(s(n-\tau))'(s^*(n-\tau))' + (s(n-\tau)+\omega_n)(s^*(n-\tau))]$$

calculating expectation value we can simplify $E[\omega_n] = 0$ and then get finally

$$CRLB(\tau) = -\mathrm{E}\left[\frac{\partial^2 \ln p(y;\tau)}{\partial \tau^2}\right]^{-1} \tag{5.5}$$

5.5.2. CRLB for Frequency Offset

If we put $\tau = 0$ then the receiver signal in presence of noise is

$$y_n = exp(j2\pi ftn)s(n) + \omega_n \tag{5.6}$$

Denoting $p(y;f)$ to be the joint Probability Density Function for whole observation parameterized by f, we derive

$$p(y|f) = \alpha \cdot exp\left(\frac{1}{\sigma_n^2}\sum_n R[y(n)s^*(n) \cdot e^{-j2\pi ftn}]\right)$$

Calculating natural logarithm and second order partial derivative with respect to f and we get

$$\left[\frac{\partial^2 \ln p(y;f)}{\partial f^2}\right] = -\frac{1}{2\sigma_n^2} \cdot \sum_n R[(y(n)s^*(n) \cdot e^{-j2\pi ftn})'']$$

and substituting $y(n)$

$$\left[\frac{\partial^2 \ln p(y;f)}{\partial f^2}\right] = -\frac{1}{2\sigma_n^2} \cdot \sum_n R[(s(n)e^{-j2\pi fn}(-j2\pi n)^2)s(n) * e^{-j2\pi fn} +$$

$$+(s(n)e^{-j2\pi fn} + \omega_n)s(n) * e^{-j2\pi fn}(-j2\pi n) +$$

$$+(s(n)e^{-j2\pi fn} + (-j2\pi n))s(n) * e^{-j2\pi fn}(-j2\pi n) +$$

$$+(s(n)e^{-j2\pi fn} + \omega_n)s(n) * e^{-j2\pi fn}(-j2\pi n)^2]$$

calculating expectation value we can simplify $E[\omega_n] = 0$ and then get finally

$$CRLB(\tau) = -\mathrm{E}\left[\frac{\partial^2 \ln p(y;f)}{\partial f^2}\right]^{-1} \tag{5.7}$$

5.5.3. Evaluation of the Implementation

The results of the implemented synchronization are shown in this section. We evaluate the performance by the distribution of the individual estimated parameters for which the CRLB was calculated in previous section.

Figure 5.12 shows the distribution of the time offset error of real incoming signal. As shown, the error is zero for vast majority of nearly thousand estimated time offsets. About 20 of those are estimated with the error of two samples.

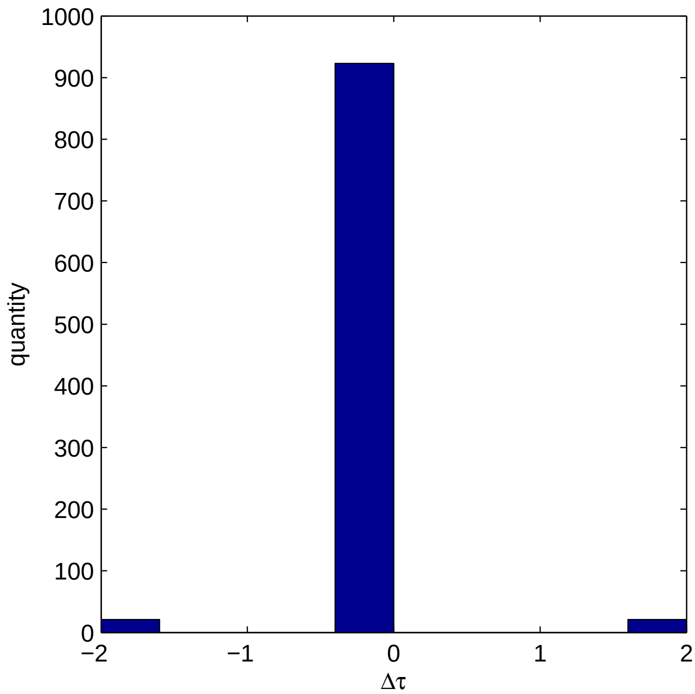

Figure 5.12.: Distribution of the Time offset estimate error

In case of the lower SNR, MSE of the time shift estimate gets bigger. In higher SNR environment, the correlation peaks are still high enough to be correctly recognized, but if we lower SNR to small enough values, the correct peaks become harder to recognize and often the wrong time shift is estimated. Resulting in large MSE of the estimate.

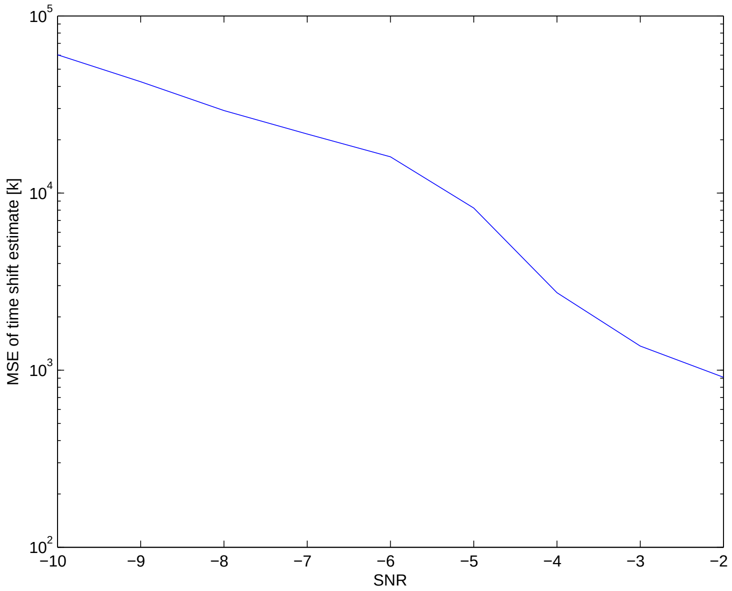

Figure 5.13.: MSE of time shift estimate.

Figure 5.14 shows the distribution of the error of frequency offset estimate. It indicates Gaussian distribution with certain value of variance.

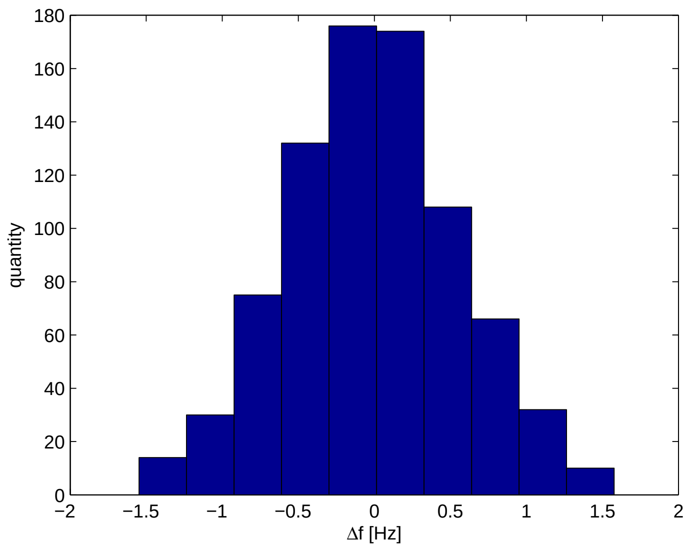

Figure 5.14.: Distribution of the error of frequency offset estimate

As figure 5.15 shows MSE of the estimate raises for lower SNR

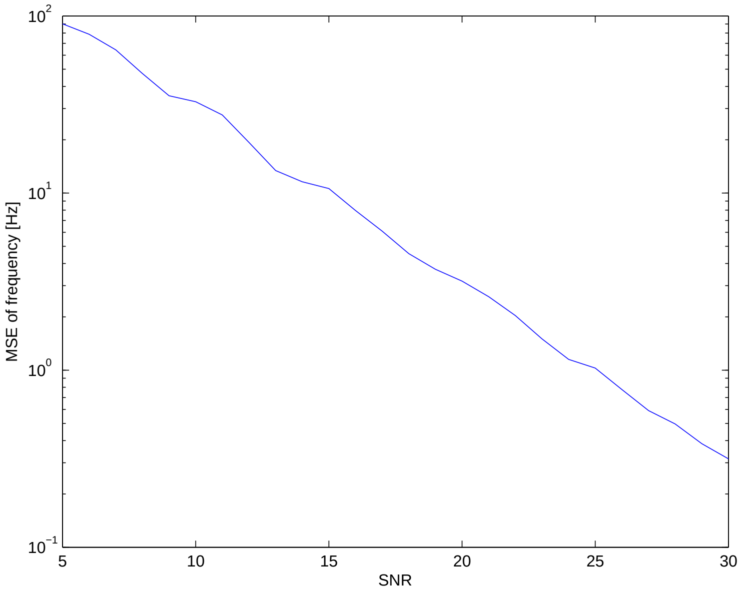

Figure 5.15.: MSE of frequency offset estimate.

Figure (5.16) shows the distribution of the error of estimated phase. It also has Gaussian distribution.

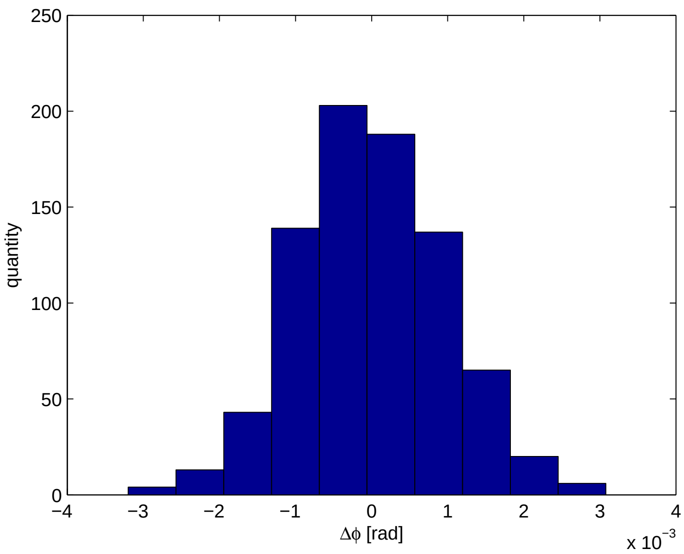

Figure 5.16.: Distribution of the error of phase offset estimate

Figure 5.17 shows how much the MSE growths by lowering SNR.

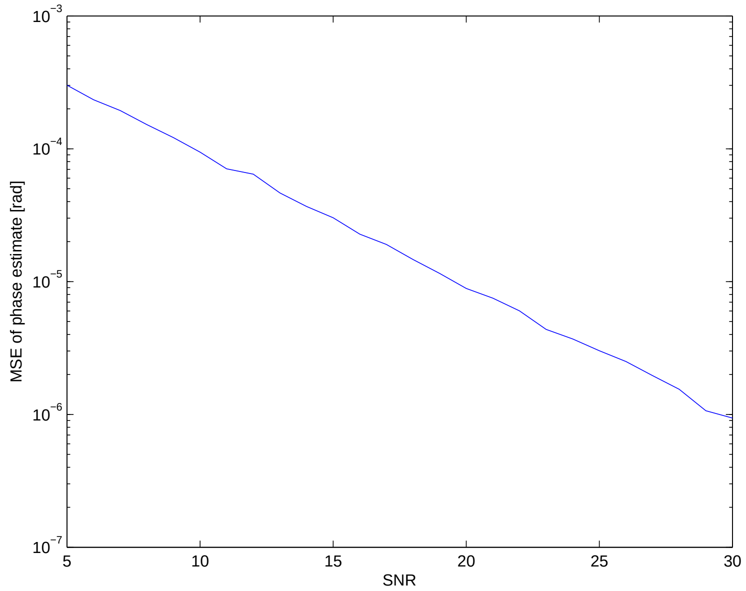

Figure 5.17.: MSE of phase offset estimate.

5.6. Non-Ortogonal Channel Estimation

In multiple access channel (MAC), we assume that channels from the sources A and B to the destination are subject to frequency-flat and time invariant fading with fade coefficients h_A and h_B. That means that the received signal r at the destination is given by

$$y = h_A x_A + h_B x_B + \omega = r_0 + \omega$$

where ω is AWGN and y_0 is noiseless superimposed signal at the receiver.

We can write the individual transmitted signals as $x_A = M(s_A)$ and $x_B = M(s_B)$ where $M(\cdot)$ denotes a modulation mapping function and s_A, s_B are the source symbols. So after substitution we get

$$y_0(s_A, s_B) = h_A M(s_A) + h_B M(s_B)$$

In general of course each pair of symbols (s_A, s_B) result in a different value of y_0, so that the cardinality of the received constellation at the destination is M^2. However there are certain values of h_A and h_B for which some of these constellation points are coincident. We refer to these sets of values as singular fade states.

So a pair of channel fade coefficients (h_A, h_B) constitutes a singular fade state if and only if for some pair of symbols (s_A, s_B) there is at least one pair of symbols $(s_{1A}, s_{1B}) \neq (s_{2A}, s_{2B})$ for which $r_0\,(s_{1A}, s_{1B}) = r_0\,(s_{2A}, s_{2B})$ [11]

So we can write that singular fade state occurs if and only if :

$$h_A x_{1A} + h_B x_{1B} = h_A x_{2A} + h_B x_{2B}$$

$$x_{1A} + \frac{h_B}{h_A} x_{1B} = x_{2A} + \frac{h_B}{h_A} x_{2B}$$

We call $\frac{h_B}{h_A}$ a relative channel. Nowadays the relative and common channels are estimated by ortogonal pilots. In this work is shown the example of this case as well as implementation. Pilots can be ortogonal in many ways (frequency,time, code). However, as proposed in DIWINE group, a design of the pilots is required that would not use any ortogonal resources. Such pilot does not need to be able to directly estimate single channel coefficients. It just has to make estimate of a relative and common channel.

Additionally from the perspective of the receiver we do not have same requirements on the quality of the estimate of h_A and the quality of the estimate of $\frac{h_B}{h_A}$. Because relative channel only deforms decision regions and the error rate of this deformation is different than in the case of rotation of whole constellation.

6. Conclusion

The goal of this thesis was to introduce basic principles of wireless network coding, give an overview about estimation theory, channel parametrization and ultimately synchronization. The task was to design hierarchical and non hierarchical pilot signals for channel estimation of the relative and common channel response. The studied theory was supposed to be implemented and showcased in few selected scenarios.

This work does introduce principles of Wireless Network Coding and on a simple example highlights its advantage over simple routing solutions and even over Network Coding.

Estimation theory was thoroughly investigated and basic overview of parameter model introduced. In the same chapter, the purpose of estimators is explained in the sense of setting up criterion on fidelity, Estimator equation and solver. Based on the criteria, several estimators are derived as well as performance parameters for arbitrary estimator. Performance limits are also explained in a form of Cramer- Rao Lower Bound. Finally, introduction how to make a solver for estimator is shortly hinted.

Chapter three introduces various algorithms for frequency offset synchronization, carrier phase synchronization and symbol timing synchronization. Those algorithms are divided based on the fact how much information we have available. In all cases we use likelihood function to derive those algorithms and have to eliminate all those parameters that are not available. Frame synchronization is briefly discussed in terms of design of pilot signal.

Contribution part starts with chapter which sets up the goals in terms of interface protocol and choice of pilot signals for various demands. Short introduction to to Cazac sequences, which are chosen as pilot signals for ortogonal synchronization, is provided in terms of basic correlation properties.

Last chapter describes the implementation of synchronization algorithms with designed pilot sequences. The simulation is done by software radio platform, controlled in MATLAB environment and showed in point to point scenario. First is analyzed packet timing synchronization where we investigate properties of correlation of Cazac sequences with influence of frequency shift offset. This is compared to behavior of PN sequence in the same scenario. Then the timing offset estimation with elimination of the frequency offset is introduced. Based on the mentioned properties is proposed the whole synchronization algorithm that first executes coarse timing synchronization followed by frequency offset synchronization where we make use of the difference of the phases of two consecutive correlation peaks, after which the fine timing synchronization is done. Last step is the estimation of the channel.

Performance limits are calculated in a form of Cramer-Rao Lower Bound and actual performance of the proposed synchronizer evaluated by measuring mean square error of the estimation.

Last section gives short introduction to the channel estimation with non-ortogonal pilot signals. This design of the pilots and estimation algorithm for that is not yet solved as this field is still the subject of ongoing research .

Bibliography

[1] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," Information Theory, IEEE Transactions on, vol. 46, no. 4, pp. 1204 –1216, Jul. 2000.

[2] S. Zhang, S. C. Liew, and P. P. Lam, "Hot topic: physical-layer network coding," in Proceed- ings of the 12th annual international conference on Mobile computing and networking, ser. MobiCom '06. New York, NY, USA: ACM, 2006, pp. 358–365.

[3] J. Laneman, D. Tse, and G. W. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior," Information Theory, IEEE Transactions on, vol. 50, no. 12, pp. 3062–3080, 2004.

[4] S. Katti, S. Gollakota, and D. Katabi, "Embracing wireless interference: analog network coding," SIGCOMM Comput. Commun. Rev., vol. 37, no. 4, pp. 397–408, Aug. 2007.

[5] B. Rankov and A. Wittneben, "Achievable rate regions for the two-way relay channel," in Information Theory, 2006 IEEE International Symposium on, 2006, pp. 1668–1672.

[6] P. Popovski and H. Yomo, "Physical network coding in two-way wireless relay channels," in Communications, 2007. ICC '07. IEEE International Conference on, 2007, pp. 707–712.

[7] J. Sykora and A. Burr, "Layered design of hierarchical exclusive codebook and its capacity regions for HDF strategy in parametric wireless 2-WRC," Vehicular Technology, IEEE Transactions on, vol. 60, no. 7, pp. 3241 –3252, Sep. 2011.

[8] J. Sykora and A. Burr, "Network coded modulation with partial side-information and hierarchical decode and forward relay sharing in multi-source wireless network," in Wireless Conference (EW), 2010 European, 2010, pp. 639–645.

[9] B. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference through structured codes," Information Theory, IEEE Transactions on, vol. 57, no. 10, pp. 6463 –6486, Oct. 2011.

[10] R. Zamir, "Lattices are everywhere," in Information Theory and Applications Workshop, 2009, 2009, pp. 392–421.

[11] "DIWINE - Dense cooperative wireless cloud network," http://www.diwine-project.eu

[12] T. Koike-Akino, P. Popovski, and V. Tarokh, "Denoising maps and constellations for wireless network coding in two-way relaying systems," in Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE, 2008, pp. 1–5.

[13] T. Koike-Akino, P. Popovski, and V. Tarokh, "Optimized constellations for two-way wireless relaying with physical network coding," Selected Areas in Communications, IEEE Journal on, vol. 27, no. 5, pp. 773–787, 200

[14] G. Potnis, D. Jalihal, "Novel polyphase training sequence based synchronization estimator for OFDM " in Communications (NCC), 2011 National Conference on, 2011, pp. 3–4.

[15] D. Sen et. al., "An Efficient Frequency Offset Estimation Scheme for Multi-Band OFDM Ultra-Wideband Systems," 2nd International Symposium on Advanced Networks and Telecommunication Systems, pp. 1-3, Dec. 2008.

[16] STEVEN M. KAY. Fundamentals of statistical signal processing: Estimation Theory. Englewood Cliffs, New Jersey: PTR Prentice-Hall, Inc, 1993. ISBN 0-13-042268-1.

[17] M. M. U. Gul, S. Lee and M. Xiaoli, "Robust synchronization for OFDM employing Zadoff-Chu sequence", Proceedings of the 46th Annual Conference on Information Sciences and Systems (CISS), pp.1,6, 21-23 March 2012.

Appendix

Algoritmus 6.1 Coarse timing synchronization and frequency offset synchronization

```
clear all;
close all;
fs=2e5;
%% searching for peaks and correlation
sig1=load('caz8len113amp10');
sig1=sig1.r(2100001:2700000);
cazac1=zadoff(8,113);
nSamp=8;
REC_filter = ones(nSamp,1)*sqrt(nSamp);
fn1=sig1;
vysl_kor1=xcorr(fn1,cazac1);
%% searching for peaks
[pks,locs] = findpeaks(abs(vysl_kor1(length(fn1):length(fn1)+90000)),...
...'MINPEAKHEIGHT',0.1,'MINPEAKDISTANCE',110);
bonus=-2;
if(locs(1)+150<locs(2)) tau_est1=locs(2)+bonus; peaks1=locs(2:end); end
if(locs(1)+150>locs(2)) tau_est1=locs(1)+bonus; peaks1=locs(1:end); end
signl1=(sig1(tau_est1:end));
%% sync freq offset
vysl_kor1=vysl_kor1(length(fn1):length(fn1)+7000);
o = 0;
Step_1 = packet_len;
while 1 %freq. offset estimator
start = tau_est1+ o*Step_1;
if length(vysl_kor1(start+1:end))<Step_1 break;
end
data_rx =(vysl_kor1(start:start+Step_1-1));
sig_rr=fn1(start:start+Step_1-1);
[piks,loks]=findpeaks(abs(data_rx),'MINPEAKHEIGHT',0.1,...
...'MINPEAKDISTANCE',110);
%% unwrap
if(angle(data_rx(loks(1)))>angle(data_rx(loks(2))))
phase_cor=angle(data_rx(loks(2)))+2*pi;
else phase_cor=angle(data_rx(loks(2)));
end
%%
offset_est=angle(data_rx(loks(1)))-phase_cor;
offset_est=offset_est/(pilot_len*2*pi);
offset_frac(o+1)=offset_est;
  tt = 1:length(packet);
  tt = tt + start-1;
  tt = tt(:);
  compansated_signal(:,o+1)=packet.*(exp(j*2*pi*offset_est*tt));
  o=o+1
end
```

Algoritmus 6.2 Fine timing synchronization and Channel estimation

```
fn1c=reshape(compansated_signal,size(compansated_signal,1)...
...*size(compansated_signal,2),1);
%% fine time sync
sekv=[cazac1,cazac1];
krlc=xcorr(fn1c,sekv);
[pks1,locs1] = findpeaks(abs(krlc(length(fn1c):end)),...
...'MINPEAKHEIGHT',0.05,'MINPEAKDISTANCE',600);
vzor=xcorr(sekv,sekv);
prenos=(krlc(length(fn1c)+locs1))/(vzor(226));
tau_est2=locs1(1);
signalfine=(fn1c(tau_est2:end));
dl=round(length(signalfine)/packet_len);
for(i=0:1:dl-3)
t=(1:packet_len-pilot_len)+packet_len*i;
bez_caz(:,i+1)=(signalfine(packet_len*i+2...
...*pilot_len:packet_len+packet_len*i)) /prenos(i+1);
end
bez_caz_r=reshape(bez_caz,size(bez_caz,1)*size(bez_caz,2),1);
signal1=conj(REC_filter);
sm = filter(signal1,1,bez_caz_r);
smd = downsample(sm,nSamp,7);
scatterplot(smd)
```