Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Disertační práce

2019 Ing. Karel Dudáček

Západočeská univerzita v Plzni
Fakulta aplikovaných věd

MĚŘENÍ ZPOŽDĚNÍ SIGNÁLŮ S POUŽITÍM NEEKVIDISTANTNÍHO VZORKOVÁNÍ

Ing. Karel Dudáček

disertační práce
k získání akademického titulu doktor
v oboru Informatika a výpočetní technika

Školitel: doc. Ing. Vlastimil Vavřička, CSc.

Katedra informatiky a výpočetní techniky

Plzeň 2019

University of West Bohemia

Faculty of Applied Science

SIGNAL DELAY MEASUREMENT USING NON-UNIFORM SAMPLING

Ing. Karel Dudáček

Doctoral thesis
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in specialization
Computer Science and Engineering

Supervisor: doc. Ing. Vlastimil Vavřička, CSc.

Department of Computer Science and Engineering

Pilsen 2019

Abstract

A goal of this thesis is development of methods for decreasing number of samples required for digital measurement of delays between fast analogue signals. After introduction of a problem, contemporary delay measurement methods are studied and compared. Comparison of methods is partially based on the analysis and also on results of experimental evaluation. I found that lots of methods exist, but they are mainly based on cross-correlation. These methods require high sampling frequency, because their resolution is equal to a sampling period, therefore, these methods are not good starting point for reduction of number of samples. Due to this fact, development of a new method was a preferred option.

My novelty method uses nonuniform sampling and nonuniform Fourier transform (NDFT) for computation of a frequency and phase shift of the signals. As we have computed phase shift and frequency of signals, it is easy to compute their delay. Nonuniform sampling have "theoretically infinite" sampling frequency; this means that we can reduce sample amount by decreasing an average sampling frequency, resulting in reduction of total data amount. All this without risk of an aliasing.

However, as the number of samples decreased, statistical features of the sampling got worse and spurious peaks caused by fuzzy aliasing appeared in the spectrum. General methods for fuzzy aliasing removal were found hardly usable, therefore, development of a new method was preferred. My novelty Trellis post-processing method uses the fact that signal frequency changes slowly between particular measurements. Several greatest peaks in each spectrogram are selected as suspicious peaks; a sequence of these suspicious peaks is interpreted as a weighted graph. As the spurious peaks are at "random" positions, correct peaks could be selected by finding the lightest path in the graph. Trellis post-processing significantly simplifies identification of the frequency of signal, and makes possible measurements using NDFT with much more limited number of samples.

The goals of the Ph.D. thesis are described in the chapter 3, their fulfillment is evaluated in the chapter 8.3. Both *NDFT delay measurement method* and *Trellis post-processing method* are novelty results of the author's research. Chapter 8.2 describes topics for the future work.

Keywords: Delay measurement, phase shift, sampling frequency, non-uniform sampling, non-uniform discrete Fourier transform (NDFT), aliasing, fuzzy aliasing, Trellis post-processing, spectrogram, weighted graph, frequency tracking.

Declaration of Authenticity

I hereby confirm, that this doctoral thesis is my original and sole work and that only sources listed in the bibliography were used.

Čestně prohlašuje, že tato dizertační práce je původní a vypracoval jsem ji samostatně, a že jsem použil pouze citované zdroje.

In Pilsen on August 15, 2019 / V Plzni dne 15. srpna 2019

*

Contents

Chapter 1

Introduction

In many applications, precise measurement of short delays between analogue signals is required. A number of methods were developed for this purpose, supposing measurement of signals with sharp edges. However, if the signal has no sharp edge, it is common to use some of the analogue methods which are based on various principles. Usage of analogue measurement methods requires highly precise production and calibration of each manufactured piece. More recently, there has been a great tendency towards digital signal processing making product development faster and production cheaper. For high precision of measurement, it is required to have signals with special properties or use high speed signal acquisition and processing. But in many cases signal features are fixed and signal processing resources are limited.

Many devices, e.g. radar and sonar sensors, laser rangefinders, ultrasonic flowmeters, etc., measure delay between two short pulses. These devices use signals with a high propagation speed for measurement, and as development continues, shorter and shorter ranges are measured. Therefore, the delay between transmitted impulse and received echo is decreasing and shorter pulses are used. This results in more and more stringent requirements for precision of delay measurement between pulses.

Combination of the both above mentioned tendencies brings difficulties since signals must be processed as fast as possible, with high precision, and low processing demands. This work tries to tackle this problem and find proper measurement method. It is focused on a subset of use cases – short periodic pulses with similar features, but different delays.

In the beginning, this work was motivated by development of the digital delay measurement unit for an ultrasonic flowmeter as a replacement of the analogue one. The initial examination of the problem showed that the problem is broader and more general than initially considered, especially if high speed digital signal processing units should have been eliminated as much as possible. As the problem was not easy to be solved by common methods, it became a motivation to start my research in this domain.

Chapter 2

Problem Specification

Problem described in the introduction is too general and its description is poor there. As this work focuses on a small subset of delay measurement use cases, this chapter formally describes this subset. In the first part of this chapter, features of processed signals are described. The second part describes requirements posed on the hypothetical good-enough measurement method.

2.1 Signal Features

The main problem is to measure delay between two signals (signal pair) acquired from a sort of the sensors, for example from the ultrasonic detectors. In the whole work, it is supposed that this signal pair has following features:

- Signals appear in bursts of finite waveform length L (usually short).
- Both waveforms in the pair have similar envelope shape but are time-shifted by Δt.
- Signals in burst are harmonically oscillating at unknown frequency f_p.
- Analytical description of signal envelope is not known.
- Bursts are periodic, but each burst is unique.
- Frequency f_p is slowly changing between bursts (no or negligible change during one burst).
- There is no useful signal between bursts (e.g. all samples are zeros or there are no samples at all). Length of this interval is marked D.

An example of such waveform can be seen in figures 2.1 (whole waveform) and 2.2 (one burst).

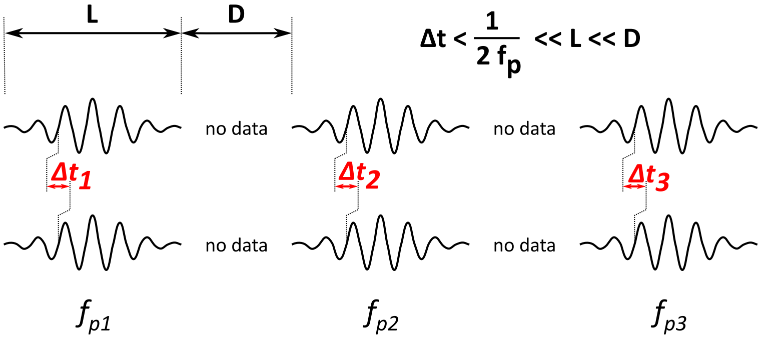

Figure 2.1: Waveform example (whole waveform).
Burst length is marked L, length of time interval between bursts is marked D, time shift is marked Δt_i.

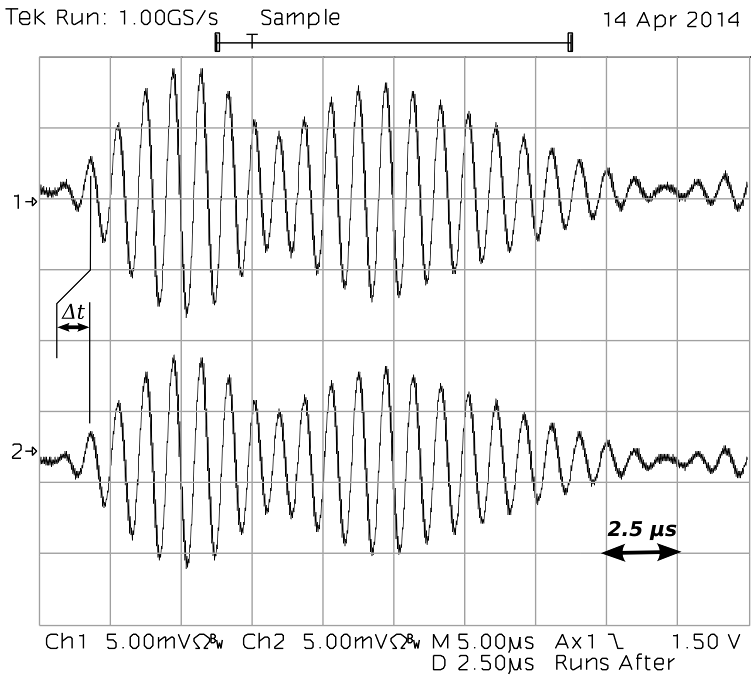

Figure 2.2: Waveform example (one burst).

An example of one burst recorded on ultrasonic flow-meter prototype. Time shift is hard to notice because $\Delta t \approx 0.01 \cdot f_p^{-1}$.

2.2 Properties of the Measurement

Suppose the above-described signal pair is acquired in some appliance regardless of whether it is radar, a laser rangefinder or something else. The basic principle of many devices is a measurement of the time shift Δt between the two waveforms in the signal pair with high precision. The measured time shift is usually very short. In many applications, very fine measurement resolution is also required.

The desired measurement method should have following properties:

- Measure delay between signals in particular burst.
- Work without a priori knowledge of the signal frequency f_p.
- Provide results regardless slow changes of signal frequency f_p.
- Ability to measure very short time delays ($\Delta t \approx f_p^{-1}/2 \ \div\ f_p^{-1}/100$).
- Measurement resolution in the order of $f_p^{-1}/1000$.
- High accuracy of measurement.

Chapter 3

Goals of the Ph.D. Thesis

3.1 Goals of the Ph.D. Thesis

Goals of the Ph.D. thesis are following:

1. To study and compare the methods for time delay measurement.
2. To develop new procedures for decreasing the sampling frequency and amount of sampled data necessary for time delay measurement.
3. To verify properties of the proposed method.

The hypothesis on the background of the research is: *Using non-uniform sampling and the non-uniform Fourier transform, it is possible to decrease the number of samples and the sampling frequency needed for measurement of the time shift between analogue signals compared to methods using periodic sampling.*

3.2 Objectives to be met

In the chapter 3.1, the main goals are presented. In this section, those goals are refined into several objectives planned for further studies in order to accomplish the goals.

1. **Investigate the current state of the art of delay measurement and examine principles of those methods.** For the delay measurement exist many methods, many of them are generally known or use generally known principles. Understanding the basics is the first task to be accomplished when improving any method.
2. **Compare the existing methods each another. Evaluate their suitability for improvements described in the goal 2.** Features of measurement methods are described in many articles, but comparison of methods is very rare. Therefore, the first step must be comparison of the contemporary methods and selection of the one that provides acceptable results and also promises the possibility of modification to the non-uniform sampling.
3. **Propose a method for time shift computation of waveforms using non-uniform sampling in order to decrease the number of samples and sampling frequency.** Usage of non-uniform sampling could probably decrease the amount of data to be stored and also enlarge the bandwidth. Several methods of non-uniform sampling exist, therefore, the one suitable for our problem must be chosen. Common processing methods do not work for non-uniformly sampled data; therefore, new method must be developed.
4. **Implement and test the method in a near-to-real appliance. Compare the new method with contemporary delay measurement methods.** Features of the method must be evaluated in various conditions including real data.

5. **Examine the relation between the number of samples, their resolution and accuracy of the results.** As the amount of input data decreases, accuracy of the results probably decreases, too. It must be investigated, whether accuracy is proportional to the number of samples, or whether there are some threshold limiting the usability. The effect of quantization must be also evaluated. This can be done either analytically or using numerical experiments.

6. **Adapt the method for the use of general-purpose AD converters.** According to the literature, general purpose AD converters can be used for non-uniform sampling. This makes possible usage of the common of-the-shelf components, but also brings some limitations, that must be considered.

Chapter 4

State of the Art of Delay Measurement Methods

The problem of measurement of short time delays is very old. Thus, many measurement methods were developed in the past. The first methods were fully analogue. Then came methods using analogue preprocessing for digital measurement. In recent years, there is a great tendency to use fully digital signal processing. This chapter describes the state of the art of the delay measurement methods focusing on the digital methods.

4.1 Basic Time Delay Measurement Methods

A few basic measurement methods exist for time delay measurement. These methods are easy to use but they are unsuitable for many applications. Basic delay measurement methods are:

- the cross-correlation method,
- the pulse counting method,
- the phase shift method,
- the delay line method.

In this section, there is a brief description of these basic methods.

4.1.1 Cross-correlation method

A common method for signal delay computation is finding the maximum of their cross-correlation function (4.1). This maximum corresponds to the time shift with the best similarity between signals.

$$\begin{aligned} R_{xy}(k) &= \sum_m x(m) \cdot y(k+m) \\ L_{xy}(k) &= k \end{aligned} \tag{4.1}$$

Each point of cross-correlation function $R_{xy}(k)$ is a dot product of discrete signals $x(m)$ and $y(m)$ mutually shifted by k samples. Time shift k is known as a *lag*; dependence of lag k on time (sample number) is linear increasing function $L_{xy}(k)$.

The distance between samples is sampling period $T_s = f_s^{-1}$, where f_s is sampling frequency. The distance between points in the cross-correlation function is also T_s. This means that maximal resolution of this method is equal to the sampling period T_s. Many applications need resolution that corresponds to very high sampling frequencies ($10Gs/s$). In these applications, this method is not usable without modifications providing finer resolution.

4.1.2 Pulse counting method

The second basic time shift measurement method is the pulse counting method. A reference clock generator produces pulses with period T_r. These pulses are routed to the counter. When the first event (e.g. signal transition on signal x) occurs, the counter starts counting reference pulses; when the second event occurs (e.g. transition on signal y), the counter stops counting. The delay between events is $\Delta t = n \cdot T_r$, where n is the counter value.

Maximal measurement error is $E_{max} = 2 \cdot T_r$, mean error is $E = T_r$.

4.1.3 Phase shift method

Let us consider a pair of discrete (attenuated) harmonic signals $x(t)$ and $y(t)$. Both signals have frequency f_p. If phase shift $\Delta\varphi$ is known, it is possible to compute time shift Δt of waveforms using (4.2).

$$\Delta t = \frac{\Delta\varphi}{\omega_p} = \frac{\Delta\varphi}{2 \cdot \pi \cdot f_p} \tag{4.2}$$

The phase shift and frequency f_p can be easily computed using the discrete Fourier transform. The phase shift computed using the Fourier transform is in range $< -\pi, +\pi >$; the time shift is then in range $\left\langle -\frac{1}{2 \cdot f_p}, +\frac{1}{2 \cdot f_p} \right\rangle$.

When using this method, frequency f_p of the signal must be found. It is possible to use a priori knowledge of the frequency or determine the frequency from the signal spectra. When the ratio of sampling frequency f_s and signal frequency f_p is not an integer (4.3), no point in the discrete spectra corresponds to the frequency of the signal, but more points are influenced by this frequency.

$$f_s = k \cdot f_p; \quad k \in \mathbb{N} \tag{4.3}$$

4.1.4 Delay line method

Usage of a delay line is another basic delay measurement method. This method is suitable only for signals with edges or similar synchronization points. One signal is connected to the input of the delay line, the other signal is used as a clock signal for memory elements (e.g. flip-flops). When an event occurs on the first signal, it propagates along the delay line. Propagation is delayed in each element by its propagation delay. When an event occurs on the second signal, it triggers memory elements capturing a state of the delay elements, i.e. the position of the first event can be found. The time difference between events can be computed from events positions and the propagation delay of delay elements. The basic delay line concept can be seen in the Figure 4.1.

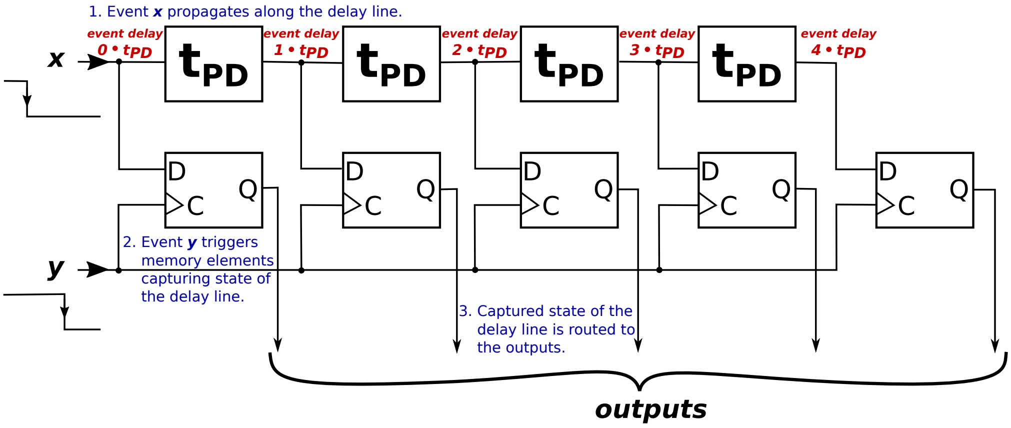
1. Event *x* propagates along the delay line.
event delay 0 • tPD
event delay 1 • tPD
event delay 2 • tPD
event delay 3 • tPD
event delay 4 • tPD
tPD
x
y
D
C
Q
2. Event *y* triggers memory elements capturing state of the delay line.
3. Captured state of the delay line is routed to the outputs.
outputs

Figure 4.1: Basic delay line concept.
An event on signal x propagates along the delay line; an event on signal y triggers memory elements (D flip–flops) and captures a state of the delay line.

4.2 Various Measurement Methods Described in the Literature

In the literature, many methods for measurement of the time shift of signals are described. In this section, there is a brief description of these methods.

4.2.1 Using a harmonic reference clock for the pulse counting method

Paper [1], describes a method for increasing the precision of the pulse counting method using a harmonic reference clock signal instead of a rectangular clock signal. For a coarse measurement of the time interval between two events, counting reference clock pulses (periods of sinus) is used. The distance between the *start* event and the first pulse (and between the last pulse and the *stop* event) is estimated using measurement of a phase of the reference clock at the moment of the event. This means that delay measurement precision is in fractions of the reference clock period.

4.2.2 Edge transformation using a SAW filter

Paper [2] describes a method for time measurement using a transversal surface acoustic wave (SAW) filter. When signals are sampled, it is problematic to measure the precise position of sharp narrow peaks or transitions because of the sampling theorem (a sharp edge has an infinite spectrum; thus, sampling frequency must also be infinite — practically, it means very high). The authors of this paper used a SAW filter for transformation of the sharp edge into finite deterministic oscillations. A SAW filter's oscillations have a narrow spectral line at a known fixed frequency; thus, waveforms can be easily sampled and reconstructed from the samples. Due to the known behavior of the SAW filter, the time of arrival of the excitation pulses can be computed.

4.2.3 Dividing delay line into coarse and fine blocks

The authors of paper [3] presented an improved delay line method for measurement of on-chip signal delays. According to the paper, it is a problem to produce a long on-chip delay line with high precision. The method presented in the paper divides the delay line into a *coarse* block and a *fine* block. When the signal pair arrives, it is routed to the coarse block. In the coarse block, a coarse delay (t_C) between signals as c multiples of coarse block elements' delay T_C is measured. The delay between signals is also reduced by t_C in the coarse block. From the coarse block the signals are routed into the fine block. In the fine block, a fine delay (t_F) between signals as f multiples of fine block elements' delay T_F is measured. The measured delay between signals is computed as a sum of fine and coarse delay $\Delta t = t_C + t_F = c \cdot T_C + f \cdot T_F$. The values c and f are integers not greater than the number of delay elements in the appropriate block.

4.2.4 Multichannel cross-correlation

Paper [4] introduces an improvement of the cross-correlation method . The time resolution of the basic cross-correlation method is equal to the sampling period T_s. The authors of this paper increased the resolution of the method to T_s/N where N is a small integer.

One of the signals (x) in the pair is directly sampled, the other signal (y) is routed to an analogue delay line with $N-1$ stages. The delay of each stage is T_s/N. The original signal and outputs of all stages are sampled giving N delayed waveforms y_n where n is the number of delay stages in the path ($n = 0, 1, \ldots, N-1$). Time shifts Δt_n are computed using the correlation method. Linear approximation function $\widehat{\Delta t}(n)$ is fitted using the least squares method and final time shift Δt is computed as $\widehat{\Delta t}(0)$.

4.2.5 Interpolation methods for the cross-correlation method

Paper [5] describes four interpolation methods for resolution improvement of the cross-correlation method . In the article, following interpolation methods are described:

- Parabolic fitting with bias compensation.
- Parabolic-fit interpolation combined with linear filter interpolation.
- Parabolic interpolation to the complex correlation function envelope.
- Matched filtering for interpolation.

4.2.6 Design of an optimal filter for the cross-correlation method

The authors of paper [6] derived an optimal filter for use in cross-correlators. They derived a mathematical model of cross-correlator for measurement of the time shift between two signals in a noisy environment. According to the model, two optimization criteria were found:

- The ratio between the cross-correlation function peak and background noise must be maximized at a position (lag) corresponding to the time shift of the signals.
- The mean-square difference between the original signal (signal before addition of noise) and the filtered signal must be minimized.

Using these two criteria, the paper's authors derived an optimal filter for filtering the cross-correlation function in order to minimize the influence of the noise on the peak detector.

4.2.7 Time delay estimation based on SINC filtering

Papers [7] and [8], describe a method for time delay estimation using a *sinc* function coefficients estimation. The papers describe the relation between two delayed signals x_k and y_k as a function $y_k = \sum x_{k-n} \cdot \mathrm{sinc}(n + m + a)$. The authors have shown that the delay between signals is equal to the peak position (in time axis) of the *sinc* function. According to paper [7], this peak position can be estimated using an LMS adaptive algorithm.

4.2.8 Signal filtering using an auto-adjusting Wiener filter

The authors of paper [9] used a Wiener filter for reconstruction of a noisy signal from an ultrasonic receiver. For adjusting the Wiener filter, it is necessary to the know system's impulse response and the power spectra of the signal and noise. The authors of the paper used the part of the received signal with high amplitude for measurement of the signal parameters and the part of the signal with low amplitude for measurement of noise parameters. The impulse response of the system is obtained from the transmitter—receiver acoustic crosstalk.

4.2.9 Cross-wavelet transformation

The method described in paper [10] uses wavelet and cross-wavelet transforms for time shift computation. Received signals $x(s, \tau)$ and $y(s, \tau)$ are transformed from the time domain into the wavelet domain using wavelet transform ψ giving their wavelet domain images $X_\psi(s, \tau)$ and $Y_\psi(s, \tau)$. In the wavelet domain the image of their cross-wavelet function $C_\psi = X_\psi \cdot Y_\psi^*$ is computed. The cross-wavelet function is transformed back to the time domain (i.e. we get $c(s, \tau)$) and its maximum $c_{max}(s_{max}, \tau_{max})$ is found. The time shift between signals corresponds to τ_{max} and the ratio between amplitudes of signals corresponds to s_{max}.

4.2.10 Third order cumulant method

Paper [11], describes two time delay computation methods: the *Third order cumulant method* and the *Bispectrum method*[1]. The third cumulant of the normal distribution is zero. The third order cumulant method computes third order cumulant $C_{xxx}(m_1, m_2)$ of signal $x(t)$ and cross-cumulant $C_{xxx}(m_1, m_2)$ of signals $x(t)$ and $y(t) = x(t - \tau)$ (i.e. τ is the time shift between signals). These cumulants do not contain white Gaussian noise. Then the cumulants are sliced at $m_2 = 0$ and the cross-correlation of the slices is computed $R(m_1) = \text{xcorr}\{C_{xxx}(m_1, 0),\ C_{xyx}(m_1, 0)\}$. The maximum of the cross-correlation function is at the position $m_1 = \tau$, i.e. $\max\{R(m_1)\} = R(\tau)$.

4.2.11 Bispectrum method

The Bispectrum method is the second method described in paper [11]. The bispectrum is the Fourier transform of the third order cumulant (it suppresses white noise). According to this paper, the bispectra $B_{xxx}(\omega_1, \omega_2) = \mathcal{FFT}\{C_{xxx}(m_1, m_2)\}$ and $B_{xyx}(\omega_1, \omega_2) = \mathcal{FFT}\{C_{xyx}(m_1, m_2)\}$ are computed. From these bispectra function $T(t) = \iint (B_{xyx}/B_{xxx})\, d\omega_1\, d\omega_2$ is computed, having its peak at position $t = \tau$ (signals' time shift).

[1] The bispectrum method is described in subsection 4.2.11.

4.3 Summary of the Contemporary Methods.

The previous two sections briefly describe various time delay estimation methods whether well-known or described in literature. These methods can be divided into two groups:

1. Methods for measuring time delay between sharp edges in the signals.
2. Methods for measuring time delay between signals without sharp edges.

Methods described in subsections 4.1.2, 4.1.4, 4.2.1, 4.2.3 belong to the first group. These methods are intended for measurement of delay between digital signals. For measurement of delay between analogue signals they are suitable only in a limited number of cases. These methods are useless for processing of signals described in Chapter 2. The method described in subsection 4.2.2 belongs to this group, but only partially, because it converts a problem of measuring delay between sharp edges to the problem belonging to the other group.

Methods described in subsections 4.1.1, 4.1.3, 4.2.4, 4.2.5, 4.2.6, 4.2.7, 4.2.8, 4.2.9, 4.2.10 and 4.2.11 belong to the second group. Method 4.2.2 partially belongs to this group too. These methods are intended for measurement of delay between analogue signals.

Many of these methods are based on the basic cross-correlation method, improving some of its features. The basic cross-correlation method has two main disadvantages. The first disadvantage of this method is coarse time resolution and the second disadvantage is the problem with precise peak identification when the cross-correlation function is flat and signals are noisy.

- Time resolution of the basic cross-correlation method is equal to the sampling period of processed signals. Methods described in subsections 4.2.4, 4.2.5 and 4.2.2 improve the time resolution of the cross-correlation method in various ways. Method 4.2.4 uses an analogue delay line and multiple AD converters, the methods described in 4.2.5 use various interpolation methods and method 4.2.2 uses computation in a continuous time domain.

- When the amplitude of the processed signals changes slowly, their cross-correlation function has a flat peak. Then the maximum detection is difficult and can be easily influenced by noise. Methods described in subsections 4.2.6, 4.2.8, 4.2.9 and 4.2.10 enhance the cross-correlation function in order to get sharper maxima.

Method 4.2.9 uses a wavelet transform instead of the Fourier transform to compute cross-correlation. The main idea behind this method is a description of the signal using functions more similar to the signal (in paper [10] short oscillating impulses are processed) than sinusoids are.

Methods not based on the cross-correlation method are phase shift method 4.1.3, *sinc* filtering method 4.2.7 and bispectrum method 4.2.11.

4.4 Methods Based on Cross-correlation

The basics of the cross-correlation methods are described in subsection 4.1.1, but for a better understanding of the main advantages and drawbacks of this method and its improved versions, it is necessary to provide a wider introduction to cross-correlation methods.

4.4.1 Continuous time cross-correlation method

Let us consider a pair of time-shifted continuous time signals (4.4). Their cross-correlation function is in (4.5). The time shift between these signals can be computed by maximizing the cross-correlation function. Substituting (4.4) into (4.5) we get (4.6).

$$\begin{aligned} & x(t) \\ & y(t) = x(t - \Delta t) \end{aligned} \tag{4.4}$$

$$\begin{aligned} R_{xy}(\tau) &= \int_{-\infty}^{+\infty} x(t) \cdot y(t + \tau) \quad dt \\ L_{xy}(\tau) &= \tau \end{aligned} \tag{4.5}$$

$$R_{xy}(\tau) = \int_{-\infty}^{+\infty} x(t) \cdot x(t + \tau - \Delta t) \quad dt \tag{4.6}$$

The maximum of the cross-correlation function R_{xy} is at the position defined by equation (4.7). Thus, the time shift between signals can be found as an argument of the cross-correlation function at its maximum (4.8).

$$\max\{R_{xy}\} = \int_{-\infty}^{+\infty} x^2(t) dt = \int_{-\infty}^{+\infty} x(t) \cdot x(t + \underbrace{\tau - \Delta t}_{=0}) \quad dt \tag{4.7}$$

$$\Delta t = \arg\left\{\max_{\tau}\{R_{xy}(\tau)\}\right\} = \arg\left\{\max_{\tau}\left\{\int_{-\infty}^{+\infty} x(t) \cdot x(t + \underbrace{\tau - \Delta t}_{=0}) \quad dt\right\}\right\} \tag{4.8}$$

If the signals x and y are zero outside the interval $\langle A, B \rangle$, the cross-correlation function can be normalized (4.9).

$$R_{xy}(\tau) = \frac{1}{(B - A)} \int_{2A-B}^{2B-A} x(t) \cdot x(t + \tau - \Delta t) \quad dt \tag{4.9}$$

If the signals are noisy, equation (4.9) can be rewritten as (4.10), where $r(t)$ and $s(t)$ are noises with zero mean value ($\mathrm{E}(s) = 0$ and $\mathrm{E}(r) = 0$).

$$\widehat{R_{xy}}(\tau) = \frac{1}{(B - A)} \int_{2A-B}^{2B-A} (x(t) + s(t)) \cdot (x(t + \tau - \Delta t) + r(t + \tau)) \quad dt \tag{4.10}$$

If the signal length increases $(B - A) \to \infty$, the cross-correlation function of the noisy signals approaches the cross-correlation function of the original signals $\widehat{R_{xy}}(\tau) \to R_{xy}(\tau)$.

$$\begin{aligned} \widehat{R_{xy}}(\tau) &= \tfrac{1}{(B-A)} \int_{2A-B}^{2B-A} x(t) \cdot x(t + \tau - \Delta t) \quad dt + & \overset{(B-A)\to\infty}{\longrightarrow} R_{xy}(\tau) \\ &+ \tfrac{1}{(B-A)} \int_{2A-B}^{2B-A} x(t) \cdot r(t + \tau) \quad dt + & \overset{(B-A)\to\infty}{\longrightarrow} 0 \\ &+ \tfrac{1}{(B-A)} \int_{2A-B}^{2B-A} x(t + \tau - \Delta t) \cdot s(t) \quad dt + & \overset{(B-A)\to\infty}{\longrightarrow} 0 \\ &+ \tfrac{1}{(B-A)} \int_{2A-B}^{2B-A} s(t) \cdot r(t + \tau) \quad dt & \overset{(B-A)\to\infty}{\longrightarrow} 0 \end{aligned} \tag{4.11}$$

4.4.2 Discrete time cross-correlation method

If the signal is sampled with sampling period T_s, the equations in (4.4) change to (4.12).

$$\begin{aligned} x(t) &= x(n \cdot T_s) \\ y(t) &= x(t - \Delta t) = x(n \cdot T_s - \Delta t) \\ n &= -\lfloor N/2 \rfloor, \ldots -2, -1, 0, 1, 2, \ldots, \left\lfloor \frac{N-1}{2} \right\rfloor \end{aligned} \tag{4.12}$$

The cross-correlation function of sampled signals is in (4.13). Substituting (4.12) to (4.13), we get (4.14). The n is the sample number; N is the number of samples of each signal. If the signal is infinite, $N \to \infty$.

$$\begin{aligned} R_{xy}(k \cdot T_s) &= \sum_n x(n \cdot T_s) \cdot y((n+k) \cdot T_s) \\ L_{xy}(k \cdot T_s) &= k \cdot T_s \\ k &= -(N-1), \ldots, -2, -1, 0, 1, 2, \ldots, (N-1) \end{aligned} \tag{4.13}$$

$$R_{xy}(k \cdot T_s) = \sum_n x(n \cdot T_s) \cdot x(n \cdot T_s + k \cdot T_s - \Delta t) \tag{4.14}$$

The time shift between sampled signals can be computed using (4.15) (modification of (4.8) for sampled signal). The position of the maximum of the cross-correlation function of sampled signals corresponds to the position of the maximum of the cross-correlation function of original (i.e. continuous) signals only if the time shift between signals is equal to the integer multiple of the sampling period (4.16).

$$\widehat{\Delta t} = \arg\left\{\max_{k \cdot T_s}\{R_{xy}(k \cdot T_s)\}\right\} = \arg\left\{\max_{k \cdot T_s}\left\{\sum_n x(n \cdot T_s) \cdot x(n \cdot T_s + \underbrace{k \cdot T_s - \Delta t}_{\substack{=0 \text{ if} \Delta t = m \cdot T_s \\ m \in \mathbb{Z} \\ \neq 0 \text{ otherwise}}})\right\}\right\} \tag{4.15}$$

$$\widehat{\Delta t} \begin{cases} = \Delta t & \text{if} \quad \Delta t = m \cdot T_s; \quad m \in \mathbb{Z} \\ \in \langle \Delta t - T_s, \Delta t + T_s \rangle & \text{otherwise} \end{cases} \tag{4.16}$$

It is possible to normalize the cross–correlation function of sampled signals (4.17).

$$R_{xy}(k \cdot T_s) = \frac{1}{N} \sum_n x(n \cdot T_s) \cdot x(n \cdot T_s + k \cdot T_s - \Delta t) \tag{4.17}$$

4.4.3 Cross-correlation of the harmonic functions

If the signals $x(n)$ and $y(n)$ are harmonic functions with angular frequency ω (4.18), then their cross-correlation function is a harmonic function with the same frequency (4.19).

$$\begin{aligned} x(n) &= A \cdot \sin(\omega n) \\ y(n) &= B \cdot \sin(\omega n + \omega \underbrace{\frac{\Delta t}{T_s}}_{\overset{\mathrm{s}}{=} \Delta n}) = B \cdot \sin(\omega(n + \Delta n)) \end{aligned} \tag{4.18}$$

$$
\begin{aligned}
R_{xy}(k) &= \sum_n x(n)\cdot y(n+k) = \sum_n A\cdot\sin(\omega n)\cdot B\cdot\sin(\omega(n+k+\Delta n)) = \\
&= AB\sum_n \sin(\omega n)\cdot\sin(\omega(n+k+\Delta n)) = \\
&= \underbrace{\frac{AB}{2}}_{\overset{\mathrm{s}}{=}C}\sum_n\left(\underbrace{\cos(\omega\Delta n-\omega k)}_{\text{const. in}\, n} - \cos(2\omega n+\omega k-\omega\Delta n)\right) = \\
&= CN\cos(\omega k-\omega\Delta n) - C\sum_n\cos(\omega k-\omega(2n+\Delta n)) = \\
&= CN\cos(\omega k-\underbrace{\omega\Delta n}_{\psi}) - C\cos(\omega k-\underbrace{\omega(2n_0+\Delta n)}_{\varphi_0})- \\
&\quad - C\cos(\omega k-\underbrace{\omega(2n_1+\Delta n)}_{\varphi_1}) - \cdots \\
&\quad \cdots - C\cos(\omega k-\underbrace{\omega(2n_{N-1}+\Delta n)}_{\varphi_{N-1}}) = \\
&= -\frac{C}{2}\Big(e^{j(\omega k+\varphi_0)} + e^{-j(\omega k+\varphi_0)} + e^{j(\omega k+\varphi_1)} + e^{-j(\omega k+\varphi_1)} + \cdots \\
&\quad \cdots + e^{j(\omega k+\varphi_{N-1})} + e^{j(\omega k+\varphi_{N-1})} - Ne^{j(\omega k+\psi)} - Ne^{j(\omega k+\psi)}\Big) = \\
&= -\frac{C}{2}\,e^{j\omega k}\underbrace{\left(e^{j\varphi_0}+e^{j\varphi_1}+\cdots+e^{j\varphi_{N-1}}-Ne^{j\psi}\right)}_{\overset{\mathrm{s}}{=}D} - \\
&\quad - \frac{C}{2}\,e^{-j\omega k}\underbrace{\left(e^{-j\varphi_0}+e^{-j\varphi_1}+\cdots+e^{-j\varphi_{N-1}}-Ne^{-j\psi}\right)}_{\overset{\mathrm{s}}{=}E} = \\
&= -\frac{C}{2}\left(De^{j\omega k}-Ee^{-j\omega k}\right) = -\frac{C}{2}\left((D+E)\cos(\omega k)+j(D-E)\sin(\omega k)\right) = \\
&= \underbrace{C\left(N\cos(\omega\Delta n)-\sum_n\cos(2\omega n+\omega\Delta n)\right)}_{\overset{\mathrm{s}}{=}F\ \text{(constant in } k)}\cos(\omega k)+ \\
&\quad \underbrace{C\left(N\sin(\omega\Delta n)-\sum_n\sin(2\omega n+\omega\Delta n)\right)}_{\overset{\mathrm{s}}{=}G\ \text{(constant in } k)}\sin(\omega k) = \\
&= \frac{AB}{2}\left(F\cos(\omega k)+G\sin(\omega k)\right) \qquad A,B,F,G\in\mathbb{R}
\end{aligned}
\tag{4.19}
$$

If the signals have finite lengths, the cross-correlation function is a harmonic function modulated by the cross-correlation function of their envelopes (it is trapezoid for rectangular envelopes because of the summation of different numbers of signal points at different positions on k axis).

When the time shift computation is done according to equation (4.15), the value $\widehat{\Delta t}$ occurs at any point in the interval $\langle \Delta t - \frac{T_s}{2},\ \Delta t + \frac{T_s}{2}\rangle$ with same probability (always valid because (4.20) holds for all cross-correlation functions).

$$
\begin{gathered}
f(x_{max}+\Delta x) = f(x_{max}-\Delta x) \\
x_{max} = \arg\left\{\max_x\{f(x)\}\right\} \\
\Delta x \le \varepsilon \quad \varepsilon\in\mathbb{R}_+
\end{gathered}
\tag{4.20}
$$

Thus, the more precise estimation of value Δt should be done using knowledge of the shape of cross-correlation function and using more points of the cross-correlation function than only $\widehat{\Delta t}$. Some methods for such estimations were described in the literature and briefly described in subsections 4.2.4, 4.2.5 and 4.2.2.

4.4.4 Computation complexity of the cross-correlation method

Computation of the cross-correlation function according to the definition in 4.13 requires $N \cdot K = N(2N-1) = 2N^2 - N$ of the multiplications and the same number of summations. The localization of the maximum requires $K = 2N - 1$ comparisons. Thus, the computation complexity of the cross-correlation method is $O(N^2)$.

The cross-correlation function can be also computed using a fast algorithm (4.21). Computation complexities of FFT and IFFT are $O(N \cdot \log N)$, thus the computation complexity of the cross-correlation method is decreased to $O(N \cdot \log N)$.

$$\begin{aligned} R_{xy}(t) &= \mathcal{IFFT}\{\mathbf{R_{xy}}(f)\} \\ \mathbf{R_{xy}}(f) &= \mathbf{X}(f) \cdot \mathbf{Y}^*(f) \\ \mathbf{X}(f) &= \mathcal{FFT}\{X(t)\} \\ \mathbf{Y}(f) &= \mathcal{FFT}\{Y(t)\} \\ Y^* &\text{ is the complex conjugate of } Y \end{aligned} \tag{4.21}$$

4.4.5 Maxima detection

The problem of localization of maxima of the discrete cross-correlation function seems to be easy, but when sampling frequency decreases to the Nyquist limit, it became to be tricky.

Let us consider hypothetical cross-correlation function described by (4.22). This function is used just for demonstration; any real cross-correlation function of finite length signals is more complicated.

$$x(t) = t \cdot sin(\omega t) \tag{4.22}$$

If this function is continuous, the value of any local maximum (4.23) of this function is greater than the values of all previous local maxima (4.24).

$$\begin{aligned} x_{max} &= \max_D\{x(t)\} = x(t_{max}) \\ D &= (-\infty, \tau\rangle \\ \tau &\in \mathbb{R} \\ \text{i.e. } t_{max} &= \arg\{x_{max}\} \end{aligned} \tag{4.23}$$

$$\begin{aligned} x_{max} &> \max_{D_\varepsilon}\{x(t)\} \\ D_\varepsilon &= (-\infty, t_{max}) \end{aligned} \tag{4.24}$$

If this function is sampled, i.e. (4.25), the relation (4.24) isn't valid for some points, even if the Nyquist's condition is met.

$$x(n) = nT_s \cdot sin(\omega n T_s) \quad n \in \mathbb{Z} \tag{4.25}$$

The violation is unavoidably caused by sample timing if the sampling frequency is not an integer multiple of the signal frequency. See Figure 4.2. If the sampling frequency $1/T_s$ decreases, the relation (4.24) is violated more and more often. Thus, the sampling frequency should be much higher than Nyquist's limit or the localization of the maxima of the cross-correlation function cannot be done using a basic search for the highest value of the cross-correlation function.

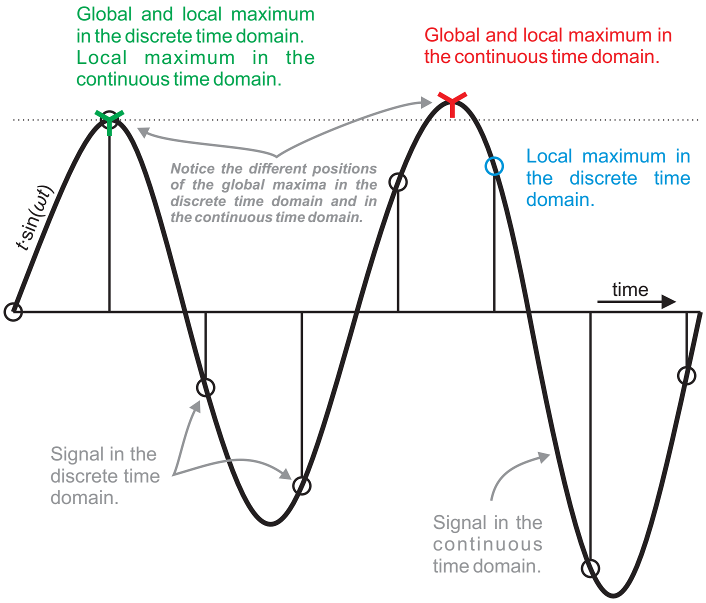
Global and local maximum in the discrete time domain. Local maximum in the continuous time domain.
Global and local maximum in the continuous time domain.
Notice the different positions of the global maxima in the discrete time domain and in the continuous time domain.
Local maximum in the discrete time domain.
t·sin(ωt)
time
Signal in the discrete time domain.
Signal in the continuous time domain.

Figure 4.2: False peak detection
Local and global maxima of the signal $x(t) = t \cdot sin(\omega t)$*. Notice the different positions of global maxima for the continuous time domain and the discrete time domain.*

4.5 Common Features of Contemporary Methods

In the literature, many methods for short time delay measurement are described. When the methods requiring sharp edges in the signal are eliminated, the rest of the methods have the following common features:

1. Requires high sampling frequency (in the sense of multiples of signal bandwidth). Thus, high speed AD converters are required, e.g. for sampling of a 1 MHz signal, 20 MS/s AD converter is necessary.

2. When the signals are sampled, a large amount of data is produced and must be stored or transmitted for processing; thus, high speed communication is required.

3. For real-time computations, fast computation units (e.g. digital signal processors) are required.

Chapter 5

Experimental Evaluation of Methods

As shown in the previous chapter, there is a lot of delay measurement methods described in the literature, but many of them share the similar principles and have similar features. For better understanding and comparison of these methods, several ones were implemented, and their features were experimentally evaluated. At first, various variants of the particular method were compared. Afterwards, the methods (their best variants) were compared each another.

As an input signals for experiments, a pair of the signals imitating typical signals from the real appliance were used. These signals are according to the chapter 2.1.

During the experiments, following methods were implemented and compared:

1. interpolation followed by correlation,
2. approximation of the cross-correlation function,
3. time shift computation from the phase shift,
4. signal approximation using the least squares method and statistical evaluation.

5.1 Description and Evaluation of the Methods

In this section, details of the evaluated time delay measurement methods are described, and different versions of these methods are compared for various sampling frequencies and noise levels.

A numeric method evaluation was carried out using the above-described signals. The main features of the simulated signal are in accordance with the signal features described in the Chapter 2.1, $f_p = 1\,MHz$. White Gaussian noise at various signal to noise ratios was added to the simulated signal in order to evaluate the method's noise sensitivity.

Time shift computations for various time shifts were carried out for many times (5 to 1000 times, depending on computation complexity) and errors were statistically evaluated. The computations were carried out in double precision IEEE 754 floats.

Two different error behaviors were observed regarding dependence between time shift and error magnitude:

- The error depends only on the method's parameters. In the following chapters the error is considered as absolute (5.1).
- The error depends on the method's parameters and the time shift. In the following chapters the error is considered as relative (5.2).

$$E_{abs} = \widehat{\Delta t} - \Delta t \tag{5.1}$$

$$E_{rel} = \left| \frac{\widehat{\Delta t} - \Delta t}{\Delta} t \right| \tag{5.2}$$

The results for various combinations of parameters are compared in the following statements and in the next chapter (Chapter 5.2) the methods are compared to one another.

5.1.1 Interpolation followed by correlation

This method is based on the common correlation method. Analogue signals are sampled (sampling period is less than the required resolution of the result) and interpolated to the required sampling rate. The common correlation method is applied to these interpolated (i.e. up-sampled) signals .

The interpolation ratio can be computed from equation (5.3) where f_s is the sampling frequency of the original signal, f_{interp} is the sampling frequency of the interpolated signal and res is the required resolution.

$$\frac{1}{n} = \frac{f_s}{f_{interp}} = res \cdot f_s \tag{5.3}$$

The interpolation can be carried out in different ways:

- Lagrange's (Newton's) polynomial interpolation,
- Hermitian polynomial interpolation,
- interpolation using zero–value samples insertion and filtration,
- spline interpolation.

Because the interpolation ratio can be very high, the usability of this method is limited due to very high computation complexity.

Interpolation methods comparison

Comparison of the following interpolation methods was carried out: interpolation using zero-valued samples insertion and filtration [12], linear interpolation and spline interpolation using natural cubic spline [13] and pchip spline [14]. Lagrangian and Hermitian interpolation are not suitable for interpolating oscillating signals due to their properties.

The interpolation method using zero-valued samples insertion and filtration was found to get much better results than the other methods, see Figure 5.1.

Effect of filtration

The effect of signal filtration was analyzed. As signal disturbed by white Gaussian noise at various levels was filtered using a bandpass filter with cut-off frequencies $0.5 \cdot f_p$ and $2 \cdot f_p$. The signals were filtered before or after interpolation. Experiments confirmed that filtration after interpolation has no effect on results, see Figure 5.2. When filtration is carried out before interpolation, error magnitude increases rapidly.

5.1.2 Approximation of the correlation function

This method is another modification of the correlation method increasing its resolution. As signal pair is sampled and then cross-correlation of signals is computed. The correlation function is approximated using the least square method near its maximum. The maximum of the approximation function is analytically computed and corresponds to the time shift between signals in the pair.

The signal (maximal frequency f_{max}) is sampled at sampling frequency f_s ($f_s > 2 \cdot f_{max}$) and the correlation function R_{xy} is computed according to the equation (4.1). Its maximum $R_{max} = \max(R_{xy})$ and corresponding lag L_{max} are found. The lag corresponds to the time shift with resolution T_s. In the vicinity of L_{max} the correlation function is approximated by function $\widehat{R}_{xy}$ and its maximum $\widehat{R}_{max} = \max(\widehat{R}_{xy})$ is found. Having corresponding lag $\widehat{L}_{max}$ time shift $\widehat{\Delta t}$ can be computed with a resolution much higher than T_s, see equation (5.4).

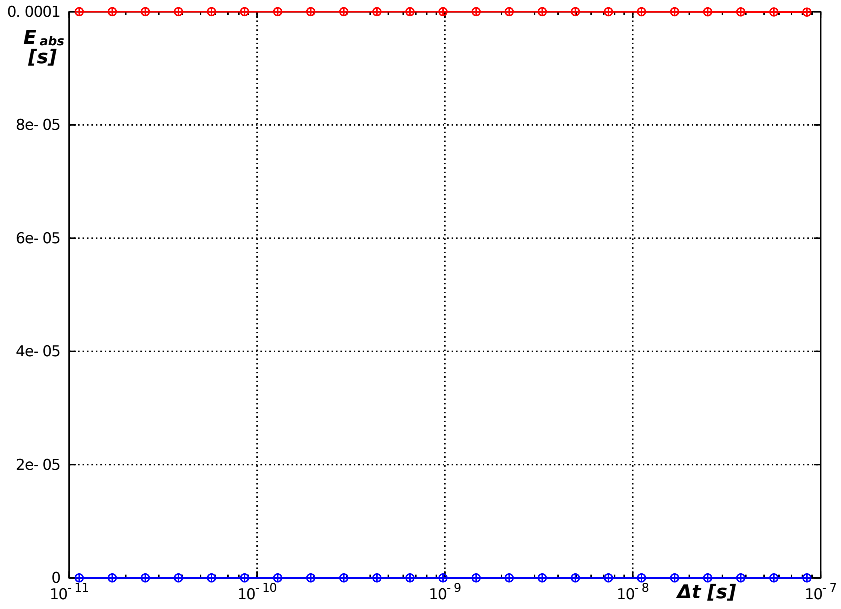

Figure 5.1: Interpolation methods comparison.
Absolute computation error for various interpolation methods, $snr = 20\,dB$, $f_s = 50\,Msample/s$. Interpolation using zero-valued samples insertion and filtration; interpolation using natural cubic and pchip splines.

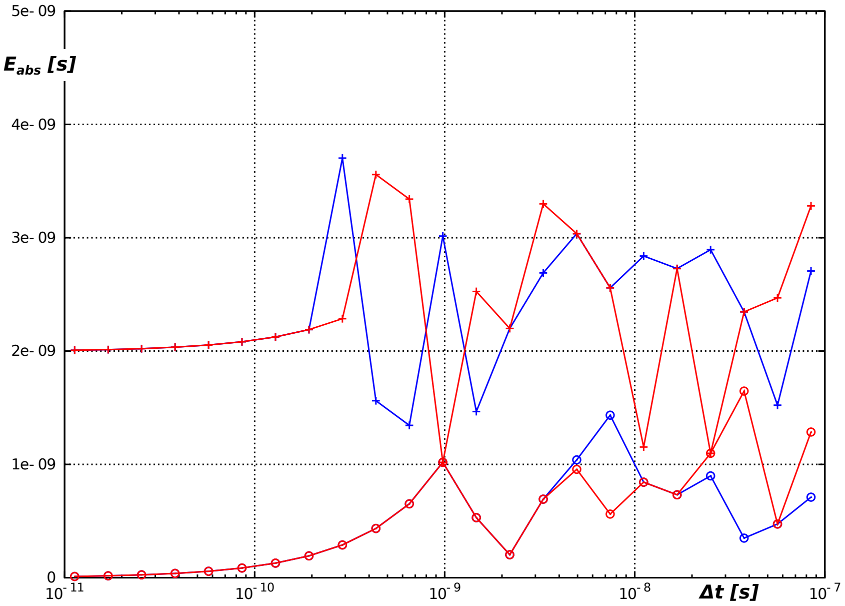

Figure 5.2: Effect of filtration after interpolation.
Absolute computation error for method of interpolation followed by correlation. Filtration after interpolation, $f_p = 1\,MHz$, passband $0.5 \cdot f_p$ to $2 \cdot f_p$, $snr = 20\,dB$, $f_s = 5\,Msample/s$.
Without filtration in blue, with filtration in red.
o . . . mean error, + . . . maximal error

For approximation of the correlation function it is advantageous to use a polynomial approximation using the least squares method (at least in the case of cross-correlation of signals specified in Chapter 2.1). In the neighborhood of correlation maxima, n points in both directions are picked up. Coordinates of these points are $[L_{max-n}, R_{max-n}], \ldots,$ $[L_{max}, R_{max}], \ldots, [L_{max+n}, R_{max+n}]$. These points are polynomialy fitted by polynomial $P_m(L)$ of degree m (in the least-squares sense). Local maxima of this polynomial can be computed analytically. Local maximum $[\widehat{L}_{max}, \widehat{R}_{max}]$ nearest (in the sense of $\min |L_{max} - \widehat{L}_{max}|$) to the cross-correlation function maximum $[L_{max}, R_{max}]$ determines the time shift $\widehat{\Delta t}$, see equation (5.4).

$$\widehat{\Delta t} = \hat{L}_{max} \cdot T_s = \frac{\hat{L}_{max}}{f_s} \tag{5.4}$$

Appropriate degree of the approximation polynomial

The appropriate number of points n for polynomial fitting depends on the degree of the polynomial (m). If the quadratic curve is fitted, it is a good choice to fit it between inflexion points of the correlation function. For the harmonic function, the inflexion points are situated at zero crossings. In this case, n is given by equation (5.5). For a higher polynomial degree, more points can be used.

$$n = \left\lfloor \frac{f_s}{4 \cdot f_p} \right\rfloor \tag{5.5}$$

Increasing the polynomial degree also brings increasing ripple and approximation error in the intervals between nodes (samples). An extreme case arises when $m + 1$ points ($n = m/2$) are fitted with a polynomial of degree m. In this case the least square method changes into Lagrange interpolation with all its consequences.

Experimental comparison of approximation polynomials

Comparison of results has been done for approximation of the cross-correlation function by polynomials of degree two and four. A polynomial of degree two gives better results for some combinations of sampling frequency and noise level and a polynomial of degree four is better for the other combinations (see Figure 5.3 and 5.4). The results of the experiment are summarized in Table 5.1.

Effect of filtration

The effect of signal filtering has been evaluated using low-pass filter with cut-off frequency $2 \cdot f_p$ (for definition of f_p, see Chapter 2.1). It was found that filtration has no effect on computation error.

Table 5.1: Appropriate degree of approximation polynomial.

Appropriate degree of approximation polynomial for method of approximation of correlation function for various sampling frequencies and noise levels. $f_p = 1\,MHz$

	$50\,Ms/s$	$10\,Ms/s$	$5\,Ms/s$
Without noise	4	4	2
$snr = 20\,dB$	4	4	2
$snr = 10\,dB$	2	2	2
$snr = 0\,dB$	2	x	x

2 or 4 ... appropriate degree of polynomial is 2 or 4;
x ... differences between degrees 2 and 4 are insignificant.

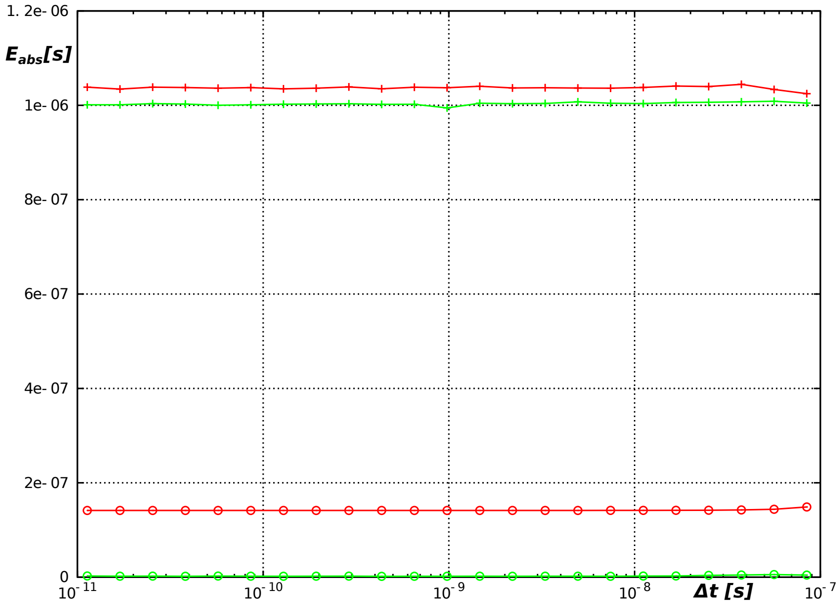

Figure 5.3: Computation error — approximation of correlation function.
Absolute computation error for method of approximation of correlation function, $snr = 10\,dB$, $f_s = 5\,Msample/s$.
Polynomial of degree two in green, of degree four in red.
o . . . mean error, + . . . maximal error

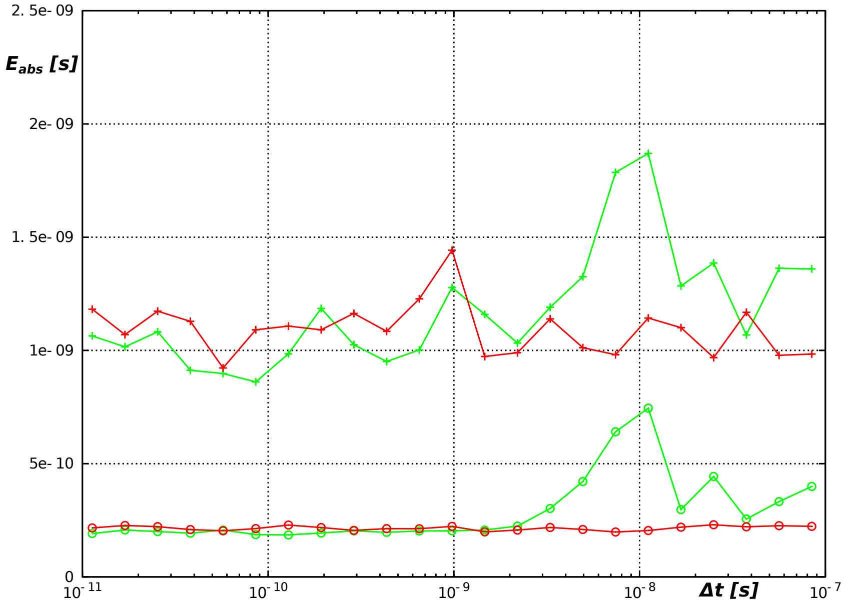

Figure 5.4: Computation error — approximation of correlation function.
Absolute computation error for method of approximation of correlation function, $snr = 20\,dB$, $f_s = 50\,Msample/s$.
Polynomial of degree two in green, of degree four in red.
o . . . mean error, + . . . maximal error

5.1.3 Signal approximation using the least squares method and statistical evaluation

Sampled signals are divided into small elements (e.g. half-waves) and each element is approximated using the least square method. Time shifts are analytically computed for corresponding pairs of elements. This set of elementary time shifts is statistically evaluated.

Let us consider a pair of time shifted signals, x_1 and x_2. Both of them are divided in some way to n elements $x_{11}, x_{12}, \ldots, x_{1n}$ and $x_{21}, x_{22}, \ldots, x_{2n}$. Each element is approximated (in the sense of least squares) by an appropriate function (a polynomial is a good choice); we get n functions for each signal $\widehat{x}_{11}, \widehat{x}_{12}, \ldots, \widehat{x}_{1n}$ and $\widehat{x}_{21}, \widehat{x}_{22}, \ldots, \widehat{x}_{2n}$. For each pair of corresponding functions, their time shift $\widehat{\Delta t}_1 \cdots \widehat{\Delta t}_n$ is computed. The algorithm for computation of time shifts depends on the approximation function used; for a quadratic approximation it may be the distance between peaks, for linear approximation it may be the distance between zero-crossings, etc. The set of elementary time shifts $\{\widehat{\Delta t}_i\}$ is statistically evaluated in order to get final time shift $\widehat{\Delta t}$. Basic methods of statistical evaluation are computations of the arithmetic mean or median of the set $\{\widehat{\Delta t}_i\}$.

Experimental comparison of approximation polynomials

This method was evaluated in versions using polynomials of degree two or four as an approximation function $\widehat{x_{ik}}$. For a quadratic function the time shift $\widehat{\Delta t_k}$ was computed as a distance of vertexes in t–axis. Local extrema were found for polynomials of degree four. Time shifts (in t–axis) were computed for extrema laying at particular intervals. These time shifts were used as $\widehat{\Delta t_k}$.

The comparison of both versions of the method can be found in Table 5.2. In the table, it can be seen the polynomial of degree of two is better than the one of degree of four, except for noiseless signals sampled at low frequency. Dependence of computation error on sampling frequency can be seen in the Figure 5.5.

Statistical evaluation of elementary time shifts

For statistical evaluation of the set of time shifts $\left\{\widehat{\Delta t_1}, \widehat{\Delta t_2}, \ldots, \widehat{\Delta t_n}\right\}$, the arithmetic mean and median were used. Computation error is many times $(10^2 - 10^{10})$ higher when using the arithmetic mean than when using the median.

Table 5.2: Appropriate degree of approximation polynomial.

Appropriate degree of approximation polynomial for method using approximation and statistical evaluation for various sampling frequencies and noise levels. $f_p = 1\,MHz$

	$50\,Ms/s$	$10\,Ms/s$	$5\,Ms/s$
Without noise	2	4	4
$snr = 20\,dB$	2	2	2
$snr = 10\,dB$	2	2	2
$snr = 0\,dB$	2	2	2

2 or 4 . . . polynomial of degree two or four gives a smaller computation error.

5.1.4 Phase shift methods

In this subsection, there two variants of the method for time shift computation from the phase shift are compared; the basic phase shift method (see 4.1.3) and the phase shift method with phase interpolation (described below).

Phase shift method with phase interpolation

This method is based on the basic phase shift method (see 4.1.3). The phase spectrum of the signal is computed using the Fourier transform and n points of the phase spectrum is chosen in the neighborhood of the frequency f_p. Then an approximation polynomial $\widehat{\varphi}(f)$ is fitted through these points. The phase shift between signals $\widehat{\Delta\varphi}$ is computed according to equation (5.6). The time shift is computed according to (4.2).

$$\widehat{\Delta\varphi} = \widehat{\varphi_1}(f_p) - \widehat{\varphi_2}(f_p) \tag{5.6}$$

Comparison of methods with and without phase interpolation

A comparison of the phase shift method with and without phase interpolation was carried out. Methods were compared for signals with signal to noise ratios $20, 10 and 0\,dB$ sampled at frequencies $50, 10$ and $5\,Msample/s$. The method without phase interpolation was better in all evaluated cases. See Figures 5.6 and 5.7.

Influence of the windowing function

The influence of the windowing function used for sample weighting during Fourier transform computation was evaluated. It was found that differences between results when using rectangular or different windows are negligible and thus can be ignored, see Figure 5.8.

5.2 Comparison of Methods

Various modifications of methods were compared in the previous section (section 5.1) and for each method the best variant was found. In this section, the above-mentioned methods are compared with one another for various combinations of noise level and sampling frequency. The results are summarized in Table 5.3. Because the computation complexity of *interpolation followed by the correlation method* is much higher than the computation complexity of the other methods, also the next best method has been indicated in the cases when this method was the best.

In Table 5.3, it can be seen that the choice of method for a particular application depends on the signal to noise ratio and is independent of the sampling frequency. The error magnitude depends on the sampling frequency. (Excluding cases of very low sampling frequency — near the Nyquist's frequency.)

Error magnitudes for various combinations of noise level and sampling frequency can be found in Table 5.4. Methods used for computations are summarized in Table 5.3.[1] All computations were carried out in double precision IEEE 754 floating point arithmetic.

[1] Methods comparison was published at *Applied Electronics Conference 2015*, see [15].

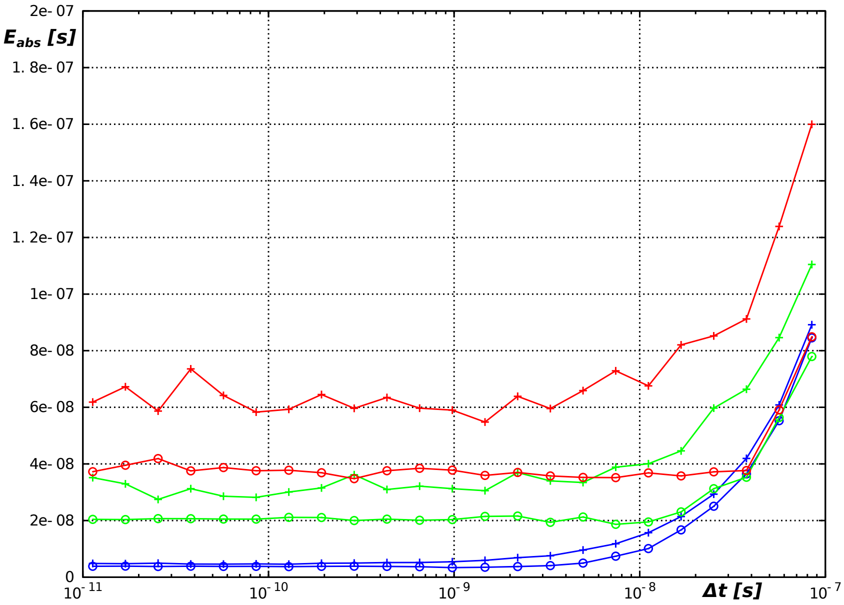

Figure 5.5: Signal approximation using the least squares method — dependence of error on sampling frequency.

Absolute computation error for signal approximation and statistical evaluation method. Degree of approximation polynomial is two, $snr = 0\,dB$.

Sampling frequency $f_s = 50\,Msample/s$ in blue, $f_s = 10\,Msample/s$ in green and *$f_s = 5\,Msample/s$ in red*.

o . . . mean error, + . . . maximal error.

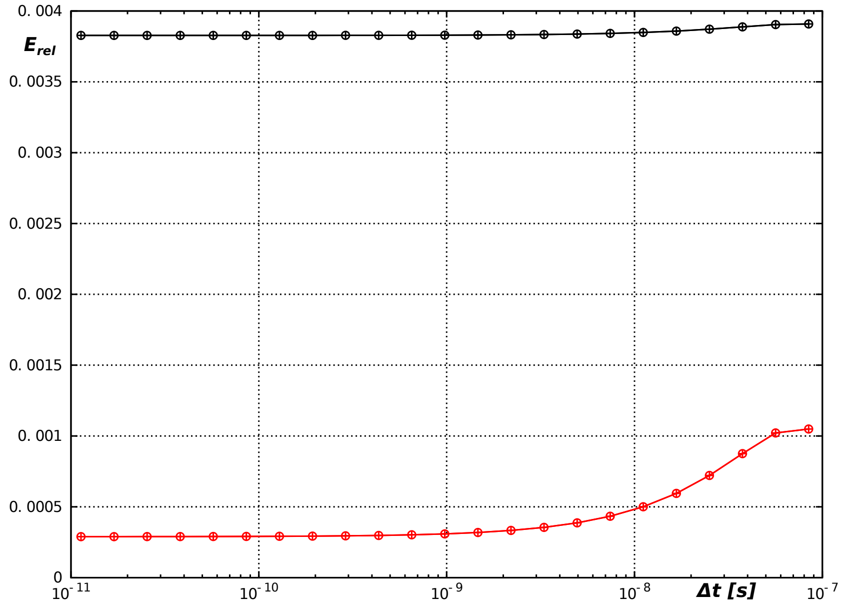

Figure 5.6: Time shift computation from the phase shift I.

Relative computation error for time shift computation from the phase shift with and without phase interpolation. Without noise, $f_s = 10\,Msample/s$.

Without phase interpolation in red, with phase interpolation in black.

o . . . mean error, + . . . maximal error

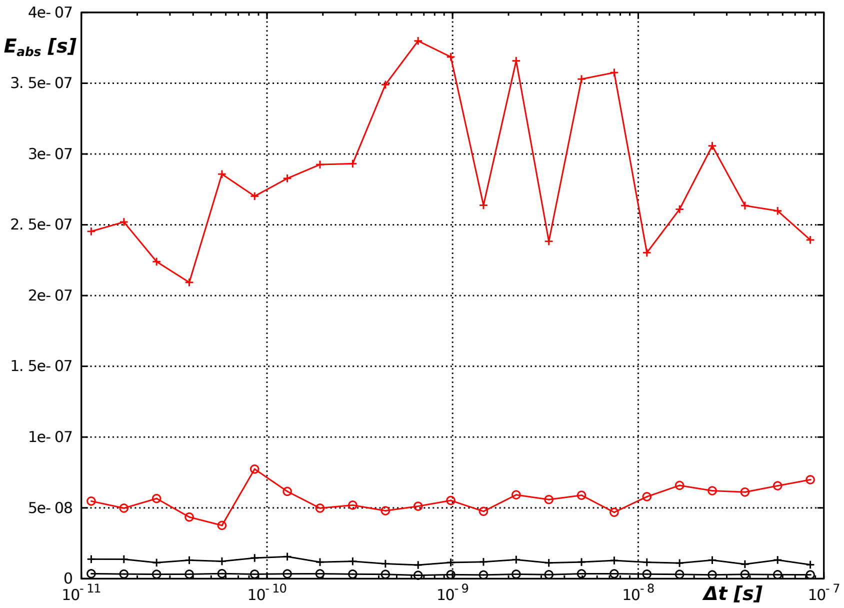

Figure 5.7: Time shift computation from the phase shift II.
Absolute computation error for time shift computation from the phase shift with and without phase interpolation. snr = $10\,dB$, $f_s = 10\,Msample/s$.
Without phase interpolation in red, with phase interpolation in black.
o . . . mean error, + . . . maximal error

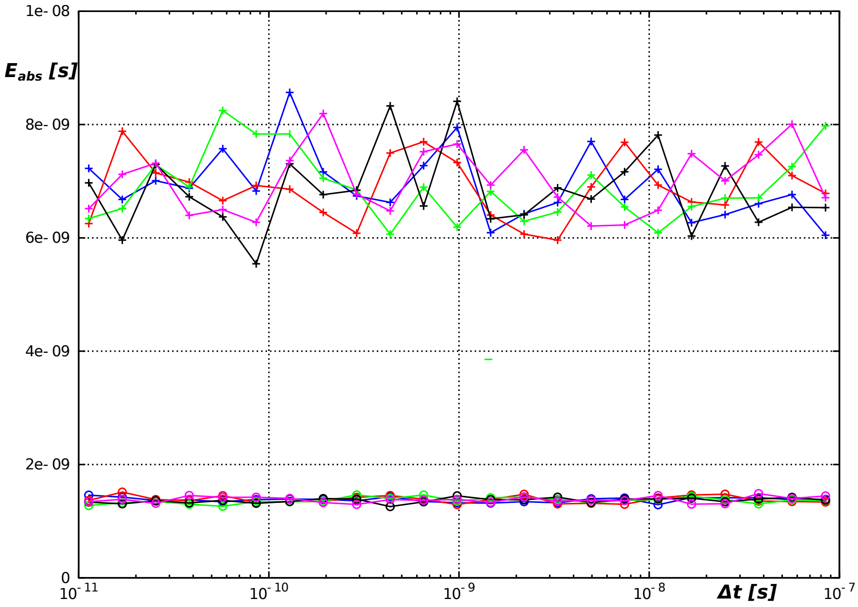

Figure 5.8: Time shift computation from the phase shift - influence of the window.
Absolute computation error for time shift computation from the phase shift. snr = $20\,dB$, $f_s = 5\,Msample/s$.
Windowing function: rectangular in blue, triangular in red, hamming in green, hann in black and kaiser in magenta.
o . . . mean error, + . . . maximal error.

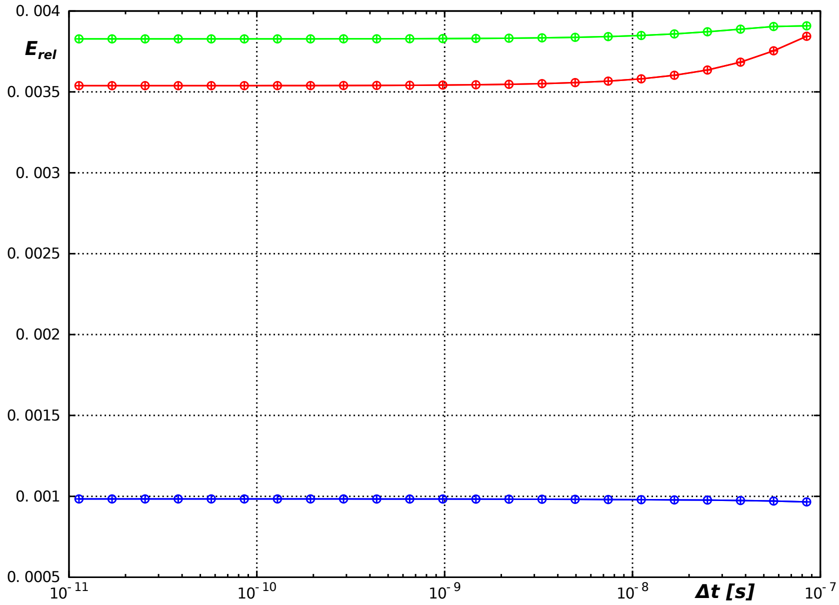

Figure 5.9: Computation error for signal time shift computation using the Fourier transform — without noise.

Relative computation error for signal time shift computation using the Fourier transform. $f_p = 1\,MHz$*, without noise.*

Sampling frequency $f_s = 50\,Msample/s$ *in blue,* $f_s = 10\,Msample/s$ *in green,* $f_s = 5\,Msample/s$ *in red.*

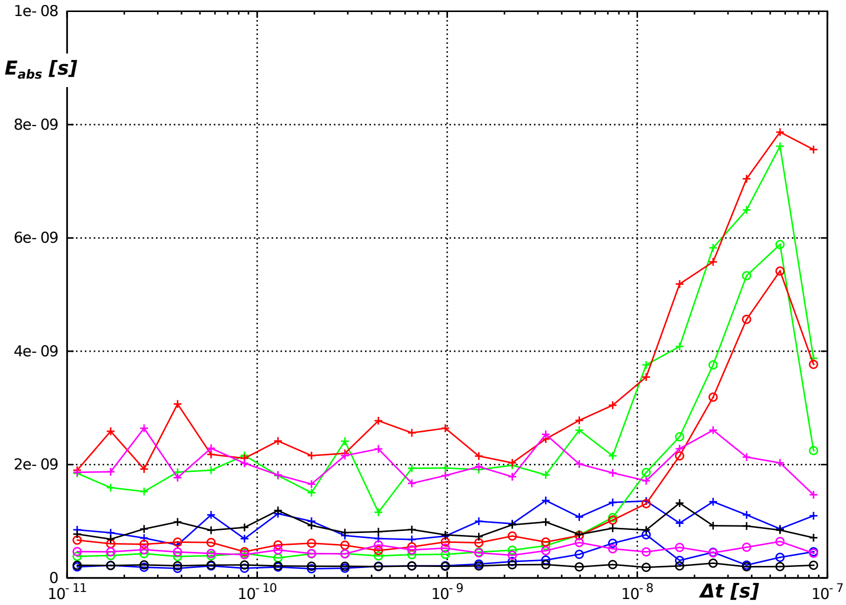

Figure 5.10: Computation error for method of approximation of correlation function I.

Absolute computation error for method of approximation of correlation function. $f_p = 1\,MHz$*,* $snr = 20\,dB$*.*

Approximation polynomial of degree 2: Sampling frequency $f_s = 50\,Msample/s$ *in blue,* $f_s = 10\,Msample/s$ *in green,* $f_s = 5\,Msample/s$ *in red.*

Approximation polynomial of degree 4: Sampling frequency $f_s = 50\,Msample/s$ *in black,* $f_s = 10\,Msample/s$ *in magenta.*

o . . . mean error, + . . . maximal error.

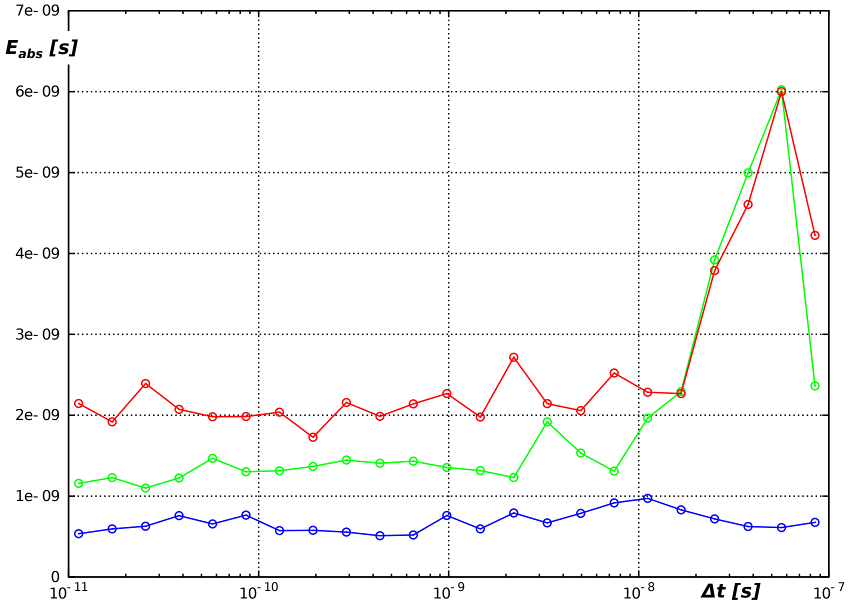

Figure 5.11: Computation error for method of approximation of correlation function II.
Absolute computation error for method of approximation of correlation function. $f_p = 1\,MHz$, $snr = 10\,dB$.
Approximation polynomial of degree 2. Sampling frequency $f_s = 50\,Msample/s$ *in blue*, $f_s = 10\,Msample/s$ *in green*, $f_s = 5\,Msample/s$ *in red*.

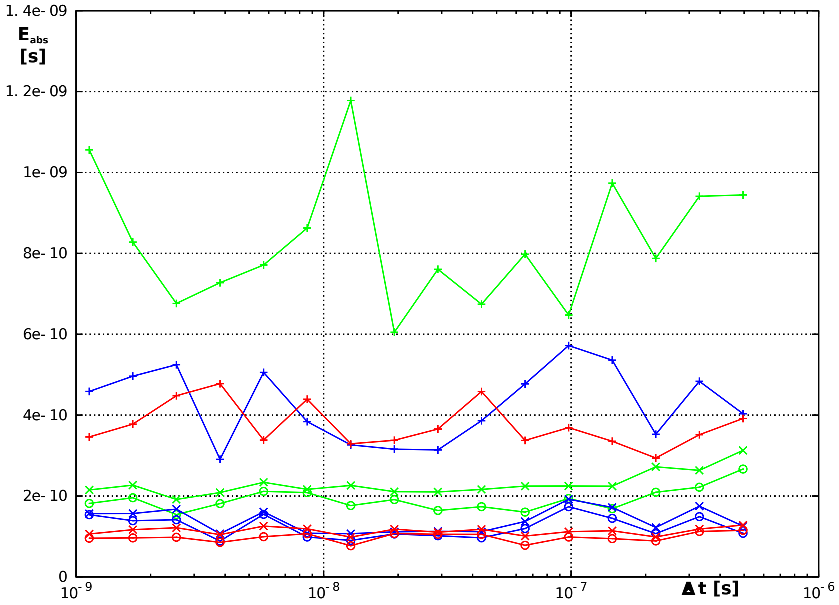

Figure 5.12: Comparison of absolute computation errors for $f_s = 100\,Msample/s$.
$f_s = 100 \cdot f_p$, $SNR = 20\,dB$, $f_p = 1\,MHz$,
Approximation of the correlation function using quadratic polynomial in blue, using polynomial of order of four in red. Phase shift method in green.
Computation run repeatedly:
"x"... mean error, "o"... median of error, "+"... maximal error.

Table 5.3: Comparison of methods.

This table summarizes the best methods for various combinations of noise level and sampling frequency. $f_p = 1\,MHz$.

	$50\,Ms/s$	$10\,Ms/s$	$5\,Ms/s$
Without noise	FP	FP	FP
$snr = 20\,dB$	IK (KC4)*	KC4	KC2
$snr = 10\,dB$	KC2	KC2	KC2
$snr = 0\,dB$	PS	PS	PS

IK . . . Interpolation followed by correlation.

FP . . . Time shift computation using Fourier transform.

PS . . . Signal approximation using least squares and statistical evaluation.

KC2 or KC4 . . . Approximation of correlation function using polynomial of degree two or four.

* *This method has much higher computation complexity than the other methods. The next best method is KC4.*

Table 5.4: Magnitude of computation error.

This table summarizes magnitudes of computation errors for methods from the Table 5.3. Computations were carried out in double precision floating point arithmetic. $f_p = 1\,MHz$.

	$50\,Ms/s$	$10\,Ms/s$	$5\,Ms/s$	Figure number
Without noise	0.1 %	0.38 %	0.35 %	5.9
$snr = 20\,dB$	170 ps / 220 ps [1] [2]	450 ps / 400 ps [1]	700 ps	5.10
$snr = 10\,dB$	0.5 ns	1.5 ns	2 ns	5.11
$snr = 0\,dB$	4 ns	20 ns	37 ns	5.5

[1] *Error magnitudes for methods KC4 / KC2.*

[2] *Error magnitude for interpolation followed by correlation (IK) is 200 ps. (Interpolation using zero–valued samples insertion and filtration. Interpolation rate 1:100.) Computation time for this method is much higher than for other methods.*

5.3 Consequences of the Method Comparison

The above compared methods could be successfully used for delay measurement under various conditions, but the common disadvantage of all mentioned methods is the requirement that the sampling frequency must be much higher than the Nyquist's limit, especially for the correlation methods. This means that the sampling frequency of the AD converter must be high even for relatively low-frequency signals. Therefore, a large amount of data is produced, requiring high speed communication with the processing unit, a large memory for data storage, and a high-speed processing unit. These properties cause resulting devices complex and development of low power devices difficult. For wider usage of digital delay measurement, it is necessary to reduce the complexity of delay measurement devices. **Therefore, development of a method for decreasing the number of samples is required.**

The improved method should give the results at least as good as the contemporary methods, but with reduced number of samples. The preferred solution is to somehow decrease the sampling rate in order to make possible the usage of built-in AD converters of low-cost low-power microcontrollers for processing the signals, that are nowadays not possible to be processed by such microcontrollers.

The resolution of cross-correlation method is dependent on the sampling frequency of the signal, therefore, this method is not good candidate for decreasing the sample amount, even though some amount of data could be saved by decreasing the number of quantization levels (bits per sample). Contrary, the phase shift method is able to work with the low sampling frequency and the Fourier transform could be computed even with the samples unevenly spaced. Thus, phase shift method seems to be the best candidate for the improvements.

Chapter 6

Non-uniform Sampling for Sampling Frequency Reduction

As a development of all kinds of devices continues, digital processing of signals with higher and higher frequencies is required. Sampling of a high frequency signal requires high sampling frequency; therefore, a large amount of data is, produced increasing demands posed on the data transmission and storage. If the signal is relatively narrow band, sampling out of the baseband is the common approach to reduce the sampling frequency. However, it doesn't work for frequencies changing in a wider range, therefore, other methods must be used. In such cases, advantages of the *randomized sampling* could be used. Randomized sampling is not one method, but a heterogeneous group of improvements for obtaining more precise sampling results using stochastic processes.

Using non-uniform sampling, infinite sampling frequency could be reached, making possible alias free processing of the signal with infinite bandwidth. However, infinite sampling frequency is possible only for infinite signals in ideal world. For finite signals, imperfections of the sampling process limit the usable bandwidth. But even in this case, average sampling frequency decreases below Nyquist's limit and the sampled signal is still characterized by samples.

My novelty method uses non-uniform sampling for reduction of sampling frequency in delay measurement applications. If the measured signals are non-uniformly sampled at relatively low average sampling frequency, using my new method, it is possible to compute the delay between these signals. All without the need of high-speed data processing and large memories, that are required when using ordinary methods, where the signals are sampled uniformly with much higher sampling frequency. My method is not based on the cross-correlation, but on the phase shift computation, therefore, it has not disadvantages of the cross-correlation methods (e.g. resolution equal to the sampling period). Conversely, we compute Fourier transform from the non-uniformly distributed samples of the signal, and then use the phase shift for computation of the delay between signals. Due to real world imperfections, non-uniform sampling is also influenced by the Nyquist's theorem, but in a slightly different way than uniform sampling. This limitation is also taken into account in my method. *Having an appropriate AD converter, my method would made possible measurement of delays between signals using low-cost microcontrollers.* This novelty method for decreasing sampling frequency using non-uniform sampling is described in this chapter. Method was also published in author's article [16].

6.1 Difficulties when using the Fourier Transform for Signals Time Shift Measurement

Suppose we have a pair of signals as described in Chapter 2.1. For time shift computation, signals have to be sampled at least at Nyquist's sampling rate $f_s = 2 \cdot f_p$. Therefore, if f_p is very high, the signal must be sampled at a very high sampling frequency. If the sampled segment of the waveform is long at the same time, we get a huge amount of sampled data to process. It could require a large amount of memory, high speed communication for real-time data transfers etc. To reduce the amount of data, we can consider reducing the sampling frequency or sampling only a part of the waveform. Both approaches have disadvantages: Reducing sampling frequency can cause problems with aliasing. If, on the other hand, only a short part of the waveform is sampled, short-time noise could influence the results[1]. This problem can be solved by usage of a non-uniform sampling.

If a non-uniform sampling is used, we use a time measurement resolution dt instead of sampling period $\frac{1}{f_s}$, and thus the average sampling rate can be lower than $2 \cdot f_p$. This means that a signal can be sampled along the whole waveform length, while providing a lower amount of data without the danger of aliasing. Using a non-uniform Fourier transform, the signals time shift can be computed.

6.2 Non-uniform Sampling

Let us consider a band limited signal (bandwidth B). A well-known Nyquist's theorem says that sampling frequency must be at least $2 \cdot B$ to avoid aliasing. Using an infinite sampling frequency, we can get an infinite bandwidth. In a real system, infinite frequency is unreachable, but we can approximate it. If we sampled a signal of infinite length composed of harmonic waveforms at random sample intervals, we would reach an infinite bandwidth [17]. But this is also impossible for two reasons – firstly no real signal is of infinite length and, secondly, we cannot measure time with infinite precision. However, it can be done using an approximation, if we comply with several conditions. If the signal is of finite length and we take enough samples, it will be statistically similar to an infinite length signal. Precision of finite time measurement resolution becomes evident in a limited bandwidth and, therefore, *latent sampling frequency.*

6.2.1 Latent sampling frequency

If the signal is sampled at random points in time, it is important to specify not only the sample number m, but also the time t_m, when each one of the samples is taken. Time intervals between sampling points can be measured in multiples of a time quantum dt. As a result, time measurement has limited resolution dt. (Note that time quantum does not represent a time measurement accuracy but the time measurement resolution.) Sampling times t_m can be written as multiplies of this time quantum (6.1).

$$\begin{aligned} t_m &= n \cdot dt \\ &dt \ldots \text{time quantum} \\ &n \ldots \text{random integer} \\ &m \ldots \text{sample number} \end{aligned} \tag{6.1}$$

Having this equation, the non-uniform sampling can be regarded as uniform sampling at frequency dt^{-1} with some samples missing. If the sampling frequency is known, the Nyquist's theorem can be applied to figure out the allowed signal bandwidth (6.2).

$$\begin{aligned} B &= \frac{1}{2 \cdot dt} \\ &B \ldots \text{allowed bandwidth} \end{aligned} \tag{6.2}$$

[1] It also influences the result in case of whole waveform sampling, but in this case an influence is reduced by the amount of other samples.

6.2.2 Sampling points

For distribution of sampling points, it is convenient to use (6.3). This distribution assures a flat probability mass function when time goes to infinity.

$$\begin{aligned} &t_{n+1} = t_n + r_n \cdot dt \\ &r_n \sim Po(\frac{\overline{f_s}}{dt}); \\ &r_n \ldots \text{random variable} \\ &\overline{f_s} \ldots \text{average sampling frequency} \end{aligned} \tag{6.3}$$

Suppose we have an AD converter with maximal sampling frequency f_{max}. For sampling (uniform or non-uniform), we must ensure $f_s \leq f_{max}$. Because sampling times are random, we cannot ensure it for each sample, but we can set the probability of violation to acceptable level p.

From cumulative distribution function (6.4) of the Poisson distribution [18, pp. 32–38], we can find r_{min}.[2] At probability level $1 - p$, the value r_{min} is the minimum of the set $\{r_n\}$; $n = 0, 1, \ldots N-1$. Therefore, the sampling period is greater than $r_{min} \cdot dt$ and the maximal frequency can be enumerated.

$$\begin{aligned} &\frac{p}{N} > \sum_{k=0}^{r} \frac{\lambda^k}{k!} e^{-\lambda} = F(r) \\ &N \ldots \text{number of samples} \\ &\lambda = \frac{\overline{f_s}}{dt} \end{aligned} \tag{6.4}$$

The AD converter sampling frequency must comply with (6.5).

$$f_m > \frac{1}{r_{min} \cdot dt} \tag{6.5}$$

For values of λ greater than 9 approximation (6.6) can be used [18, pp. 104].

$$Po(\lambda) \approx N(\mu = \lambda, \sigma^2 = \lambda) \tag{6.6}$$

6.3 Non-uniform Fourier Transform

Let us consider a signal $x(t)$. The definition of Fourier transform $\mathbb{X}(k)$ of $x(t)$ is in (6.7).[3] [19] N is the total number of samples, t_n is the sampling time, and $\Delta\omega$ is the circular frequency step.

$$\mathbb{X}(k) = \sum_{n=0}^{N-1} x(t_n) \cdot e^{-j \cdot k \cdot \Delta\omega \cdot t_n} \tag{6.7}$$

Assigning frequency step (6.8) and sampling time $t_n = n$, we get the standard DFT definition (6.9).

$$\Delta\omega = \frac{2 \cdot \pi}{N} \tag{6.8}$$

$$\mathbb{X}(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-j 2\pi k \frac{n}{N}} \tag{6.9}$$

Assigning frequency step (6.10) into (6.7), we get non-uniform DFT definition (6.11).

$$\begin{aligned} &\Delta\omega = \frac{2 \cdot \pi}{T} \\ &T \ldots \text{signal (window) length} \end{aligned} \tag{6.10}$$

[2] Values of $F(r)$ are tabulated for various values of λ, see [18, pp. 104].

[3] In this and all following equations we have omitted normalization considering the number of samples.

Table 6.1: Fourier transform modifications comparison.

	DFT	NDFT	Padded DFT[a]
Frequency resolution	$\frac{1}{T}$	$\frac{1}{T}$	$\frac{1}{l \cdot T}$
Maximal non-aliased frequency	$\frac{fs}{2}$	$\frac{1}{2 \cdot dt}$	$\frac{fs}{2}$

[a] Zero padded to l times original length.

$$\mathbb{X}(k) = \sum_{n=0}^{N-1} x(t_n) \cdot e^{-j2\pi k \frac{t_n}{T}} \tag{6.11}$$

This definition expects continuous time, but we can measure time only in discrete time quanta (clock ticks) lasting dt. Then we can use the number of clock ticks instead of the time (6.12).

$$m = \frac{t}{dt} \tag{6.12}$$

Substituting (6.12) into (6.11), we get (6.13). Informally said, we sample the signal in N points of M possible.

$$\begin{aligned} \mathbb{X}(k) &= \sum_{n=0}^{N-1} x(m_n) \cdot e^{-j2\pi k \frac{m_n}{M}} \\ M &= \frac{T}{dt}; \quad m_n = \frac{t_n}{dt} \end{aligned} \tag{6.13}$$

$M \ldots$ signal (window) length in clock ticks dt

$m_n \ldots$ sampling time in clock ticks dt

Frequency resolution and maximal non-aliased frequency of standard (DFT) and non-uniform (NDFT) Fourier transform versions can be found in Table 6.1.

The non-uniform Fourier transform can be computed from its definition with computational complexity $O(N^2)$, using the FFT algorithm with computational complexity $O(M \cdot \log(M))$,[4] or faster using an approximation algorithm, for example [20–22].

6.4 Experimental Evaluation of the NDFT Method

Numerical experiments were carried out to prove desired features of the NDFT method. Pairs of time shifted waveforms were generated, and noise of a defined level was added. Then the time shift of waveforms was computed, and relative and absolute errors were evaluated. This procedure was repeated for many combinations of sampling frequency and signal to noise ratio. For each combination of parameters, the computation was repeated many times and results were statistically evaluated.

When SNR is high (in the context of this article, $SNR \approx 20\,dB$) and sampling frequency is high ($fs \approx 50 \cdot f_p$), the NDFT method is inferior to the correlation or polynomial fit methods. But as a noise gets stronger ($SNR \to 0\,dB$) and/or sampling frequency decreases, Fourier transform methods start to perform better than the others. See Figures 6.1 and 6.2 for details. When sampling frequency approaches the Nyquist's limit ($fs = 2 \cdot f_p$) and then decreases, NDFT is preferable to classical (uniform) DFT, see Figures 6.3 and 6.4.

During my experiments, as the signal got shorter, identification of frequency f_p became difficult due to false peaks in spectrograms. In almost all examined cases, there was local maximum at the correct position corresponding to the frequency f_p, but several higher peaks appeared at „random" positions. These peaks are caused by effect called *fuzzy aliasing* or *secondary aliasing*. This effect causes various artifacts in sampled signal and in its spectrum due to poor statistical features of the sampling process. For details

[4] Each clock tick, we have not used for sampling, is used as zero-valued sample for computation using FFT.

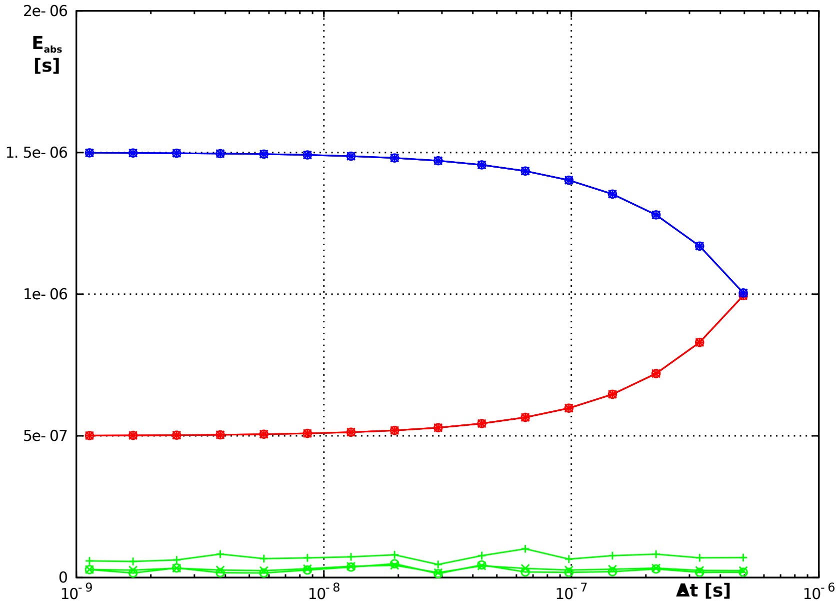

Figure 6.1: Methods comparison I.

$f_s = 2 \cdot f_p$, $dt = (100 \cdot f_p)^{-1}$, $f_p = 1\,MHz$, without noise

Approximation of the cross-correlation function in blue, time shift computation from the phase shift (classical) in red, non-uniform Fourier transform method in green.

Computation run repeatedly:

"x"... mean error, "o"... median of error, "+"... maximal error.

about fuzzy aliasing see [23, chapter 9]. Fuzzy aliasing is always present in results of the non-uniform Fourier transform. If the non-uniform Fourier transform is used for spectral analysis, fuzzy aliasing must be considered even in case of long signals, especially for low power components. In our case it got much stronger for short signals due to the small amount of the samples.

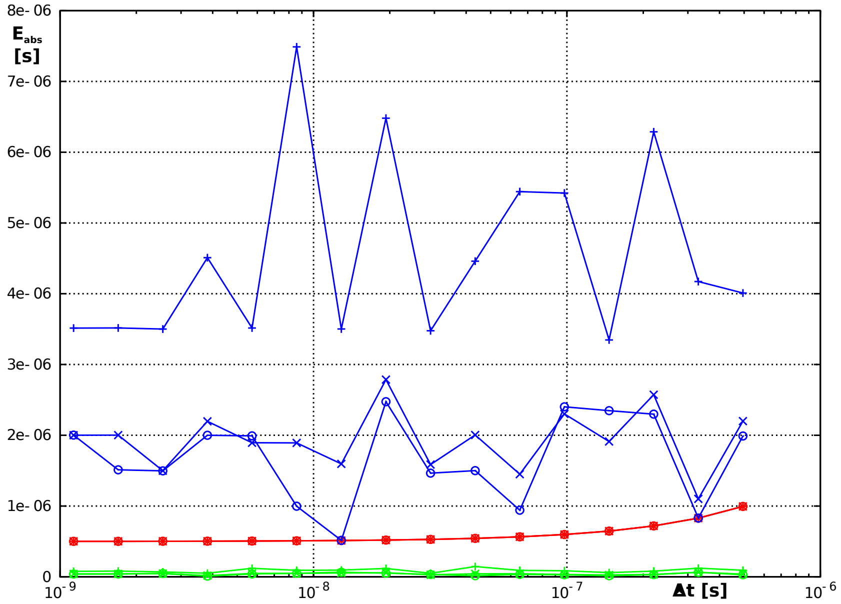

Figure 6.2: Methods comparison II.

$f_s = 2 \cdot f_p$, $dt = (100 \cdot f_p)^{-1}$, $f_p = 1\,MHz$, $SNR = 0\,dB$
Approximation of the cross-correlation function in blue, time shift computation from the phase shift (uniform) in red, non-uniform Fourier transform method in green.
Computation run repeatedly:
"x"... mean error, "o"... median of error, "+"... maximal error.

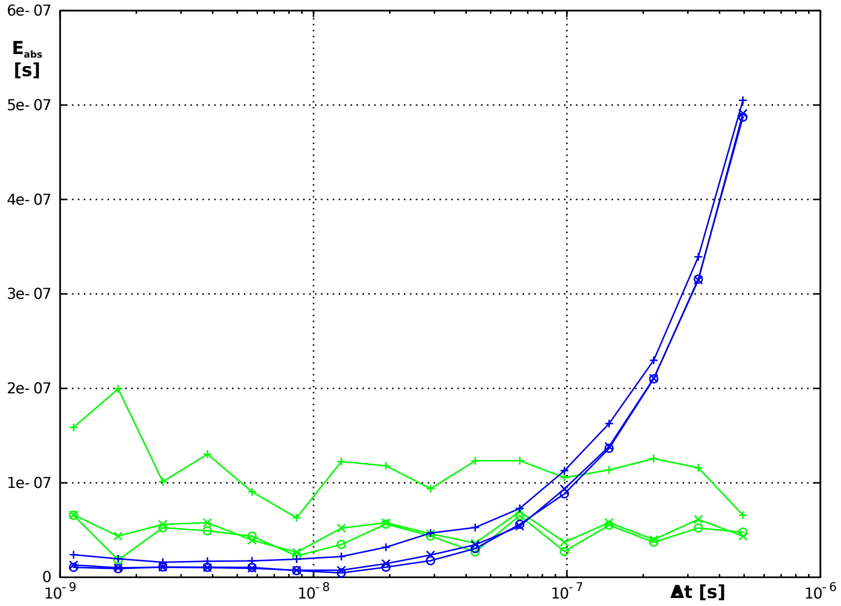

Figure 6.3: NDFT method's absolute error I.

$f_s = 1 \cdot f_p$, $dt = (500 \cdot f_p)^{-1}$, $SNR = 0\,dB$, $f_p = 1\,MHz$, $N \doteq 125$, $M = 50\,000$
Signal approximation using least squares (see 5.1.3) in blue, non-uniform Fourier transform method in green. Classical (uniform) phase shift method failed.
Computation run repeatedly:
"x"... mean error, "o"... median of error, "+"... maximal error.

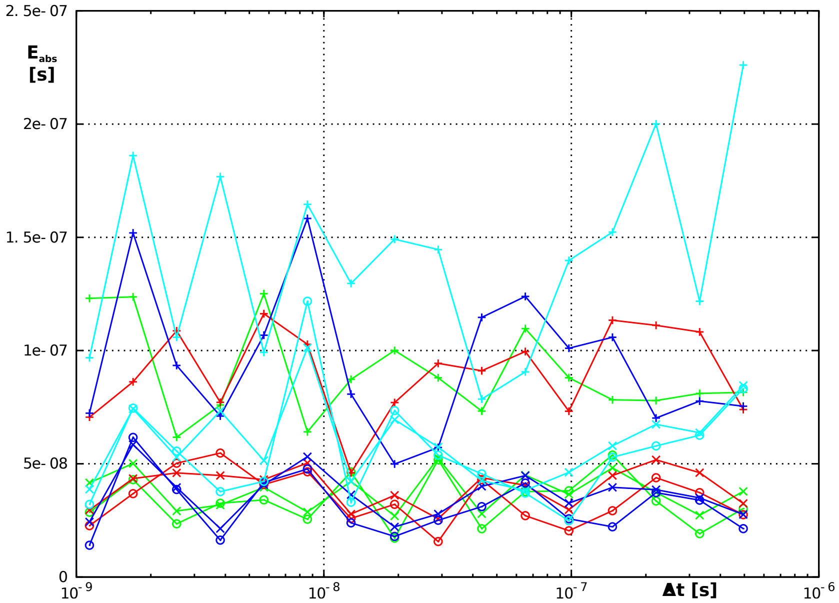

Figure 6.4: NDFT method's absolute error II.

$f_s = 0.5 \cdot f_p$, $dt = (100 \cdot f_p)^{-1}$, $f_p = 1\,MHz$, $N \doteq 62$, $M = 10\,000$

Diagram only for non-uniform Fourier shift method, other methods failed due to lack of samples.

Without noise in green, $SNR = 20\,dB$ in red, $SNR = 10\,dB$ in blue, $SNR = 0\,dB$ in cyan.

Computation run repeatedly:

"x"... arithmetic mean of the error, "o"... median of the error, "+"... maximal error.

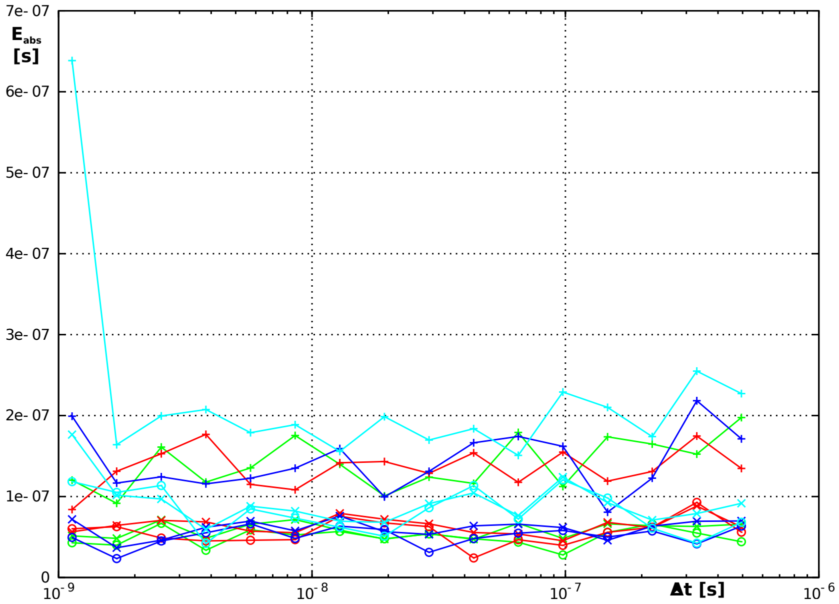

Figure 6.5: NDFT method's absolute error III.

$f_s = 0.2 \cdot f_p$, $dt = (100 \cdot f_p)^{-1}$, $f_p = 1\,MHz$, $N \doteq 25$, $M = 10\,000$

Diagram only for non-uniform Fourier shift method, other methods failed due to lack of samples.

Without noise in green, $SNR = 20\,dB$ in red, $SNR = 10\,dB$ in blue, $SNR = 0\,dB$ in cyan.

Computation run repeatedly:

"x"... arithmetic mean of the error, "o"... median of the error, "+"... maximal error.

6.5 NDFT Method – Conclusion

My novelty NDFT method was shown to be prefferable to the other methods for measurement of delays between signals sampled at low sampling frequency (see chapter 6.4 for details). If the signal frequency f_p is known, my NDFT method is directly usable for delay measurement. However, a signal frequency is unknown in our case, therefore, it must be identified from a spectrogram. Unfortunately, spectrograms contain spurious peaks due to *fuzzy aliasing*. Therefore, **some method for dealing with fuzzy aliasing must be used** in order to identify the signal frequency.

The main advantage of my novelty NDFT method is its usability with very low average sampling rates. It works even with average sampling frequency lower than $2 \cdot f_p$, since the non-aliased bandwidth does not depend on the average sampling rate, but on the time measurement resolution. This feature makes an advantage of my method against the other methods, because **sampling rate can be reduced without narrowing a bandwidth of an input**. It can be used for reduction of a sample amount by decreasing an average sampling rate, it makes possible delay measurements using low cost microcontrollers for data processing, or it could even make possible to carry out the whole delay measurement using slow internal AD converters of the microcontroller for signal sampling.

Chapter 7

Trellis Post-processing Method

If the non-uniform Fourier transform is used for spectra computation, some distortion of the results may be present. This distortion is caused by finite length of the signal and the sampling function. As the signal gets shorter, the number of samples taken decreases and statistical features of the sampling process gets worse causing various artifacts in sampled signal and in its spectrum. This distortion is called *fuzzy aliasing* or *secondary aliasing*. It is always present in results of the non-uniform Fourier transform, but in our case, it gets much stronger and spurious peaks in spectrum appear. That causes identification of oscillation frequency f_p more difficult. Low signal to noise ratio makes the situation even worse.

Fuzzy aliasing is caused by cross-interference between frequencies in the spectrum due to statistical imperfections of a sampling process, see [23, chapter 9, 15.2, and 18] for details. The problem of fuzzy aliasing is not new, therefore, several methods for its elimination exist. Main principles and features of fuzzy aliasing elimination methods are described in this chapter. Both methods solve the problem without usage of any side information. However, in our case we have some sideband information – we know, that the frequency of the useful peak slowly changes between bursts (successive instances of the signal). Therefore, it could be advantageous to evaluate the sequences of bursts to use this information.

If we had continuous discrete time signal (i.e. not bursts but uninterrupted signal), probably good solution would be to derive non-uniform version of some time-frequency analysis method (e.g. some method from Cohen class of methods). However, time-frequency analysis methods are not appropriate in our case since we have bursts with negligible frequency change during burst and step change between bursts. During the burst, time-frequency analysis gives no extra information in comparison with spectrogram (there is nearly no frequency change) but increases computation complexity. Between the bursts, there is frequency change, but there is no need (neither opportunity) for time-frequency analysis because there are no samples. Therefore, it seems to be beneficial to develop a new method for spectrogram post-processing using our sideband information, and without the computationally demanding elimination of the fuzzy aliasing.

In the first part of the chapter, there is a short overview of the current methods for dealing with fuzzy aliasing including their pros and cons. I found these methods unsuitable for our problem, therefore, I had to develop a new method. I noticed, that the positions of spurious peaks are ”randomly” changing between spectrograms, but the correct peak is always present. Thus, I decided not to remove the fuzzy aliasing, but just to select the correct peak in spectrogram. Using this approach, I developed a method, that converted difficult signal processing problem to much easier graph problem and solved it. My method not only solves the problem of frequency identification in presence of fuzzy aliasing in my delay measurement method (that was the motivation), but also simplifies tracking of the useful/interesting frequency in similar use-cases (e.g. monitoring of spectrum). If needed, it seems also possible to modify Trellis post-processing method for tracking of multiple frequencies.

My novelty *Trellis post-processing method* is described in this chapter. It was also published in author's article [24].

7.1 Fuzzy Aliasing

Base functions of the continuous Fourier transform are orthogonal, suppressing influence between spectral lines. For the uniform Discrete Fourier transform, the base functions are orthogonal for frequencies fulfilling the Nyquist's sampling theorem. If the signal length is an integer multiple of periods of the both signal and base function (or infinite length), the particular Fourier coefficient is zero for the base function with frequency differing from the signal frequency. For another signal lengths, windowing function has to be used in order to reduce the effects of unfinished signal periods (spectral leakage) and thus reduce the Fourier coefficients not corresponding to the signal frequency. If the Nyquist's sampling theorem is not fulfilled, (normal) aliasing occurs.

Let us have a general DFT equation

$$\mathbb{X}(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(t_n) \cdot e^{-j \cdot \omega_k \cdot t_n} \tag{7.1}$$

After rewriting the exponentional form of the base function $e^{-j \cdot \omega_k \cdot t_n}$ to sum of the components $\cos(\omega_k \cdot t_n) - j \cdot \sin(\omega_k \cdot t_n)$, we get

$$\mathbb{X}(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(t_n) \cdot (\cos(\omega_k \cdot t_n) - j \cdot \sin(\omega_k \cdot t_n)) \tag{7.2}$$

Splitting the equation 7.2, we get coefficients for sinus and cosinus part of the base function (7.3)

$$\begin{aligned} \mathbb{X}(k) &= a(k) - j \cdot b(k) \\ a(k) &= \frac{1}{N} \sum_{n=0}^{N-1} x(t_n) \cdot \cos(\omega_k \cdot t_n) \\ b(k) &= \frac{1}{N} \sum_{n=0}^{N-1} x(t_n) \cdot \sin(\omega_k \cdot t_n) \end{aligned} \tag{7.3}$$

Having the signal $x(t) = \cos(\omega_p \cdot t)$ we got dot product of two harmonic functions with circular frequencies ω_p and ω_k

$$\begin{aligned} a(k) &= \frac{1}{N} \sum_{n=0}^{N-1} \cos(\omega_p \cdot t_n) \cdot \cos(\omega_k \cdot t_n) \\ b(k) &= \frac{1}{N} \sum_{n=0}^{N-1} \cos(\omega_p \cdot t_n) \cdot \sin(\omega_k \cdot t_n) \end{aligned} \tag{7.4}$$

From this equation, it is clear that particular Fourier coefficient in DFT is a normalized dot product of the sampled signal and the sampled base function (see also the figure 7.1). Therefore, the indefinite properly sampled monofrequency signal is not orthogonal with the only one base function. This means that the coefficients $a(k)$ and $b(k)$ are non-zero for the frequency $\omega_k = \omega_p$, only. (Nyquist's condition must be valid also for the base functions, otherwise this statement is not valid due to normal aliasing.)

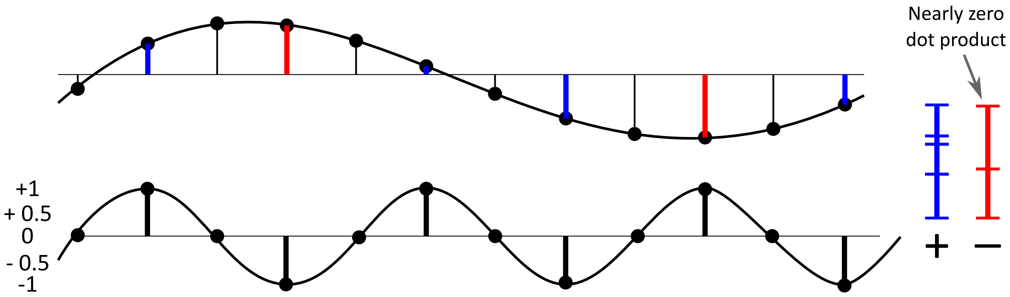

Figure 7.1: Dot product of two base functions (equidistant sampling).

For the properly sampled infinite length signal, the dot product of two sampled base functions is zero. If the harmonic monofrequency signal has non-zero dot product with the one base function, there will be zero dot product with the other one, therefore, there is no influence of the one spectral line to the other.

For the finite length signal, there is influence of the incomplete period at the beginning and at the end of the signal (spectral leakage) if the signal length is not the multiple of the whole periods of the signal and the base function and the windowing function is not used for its suppression.

Positive terms are in blue, negative in red. In the right side of the picture, there is a sum of the terms forming the final dot product (it is not shown since it is zero).

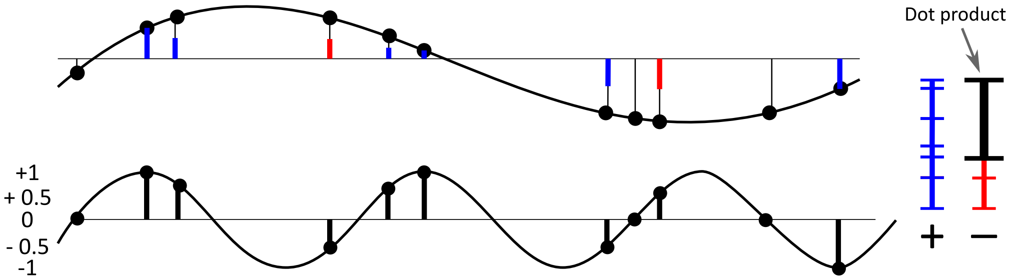

Figure 7.2: Dot product of two base functions (nonuniform sampling, few samples).

If the non-uniform sampling is used, the dot product of two sampled base functions depends also on the positions of samples in the particular instance of the sampling process. However, having the small number of samples, these samples could be improperly spread, making the dot product non-zero. Therefore, the base functions are not orthogonal each other and due to transitivity, harmonic monofrequency signal influence two spectral lines.

Positive terms are in blue, negative in red. In the right side of the picture, there is a sum of the terms forming the final dot product. Dot product is in black.

If non-uniform sampling is used, samples are positioned randomly and enough samples is required to make the statistical features of the sampling representative[1]. If the number of samples decrease, statistical features of the non-uniform sampling get worse and it could happen that the samples are distributed in the way, that a dot product of two base functions is non-zero[2], causing that these base functions are not orthogonal for this instance of the sampling process. Due to this fact, harmonic monofrequency signal is non-orthogonal with more than one base function, therefore, it influences more than one spectral line, see figure 7.2. This distortion is called *fuzzy aliasing*.

As the number of samples increases, statistical features of the sampling process get better and the dot product of the base functions decreases, therefore, the harmonic monofrequency signal influences mainly the correct spectral line (see figure 7.3).

[1] Nonuniform sampling works ideally only for infinite length signal, therefore, infinite number of samples.

[2] It would have been zero if the infinite signal length was used.

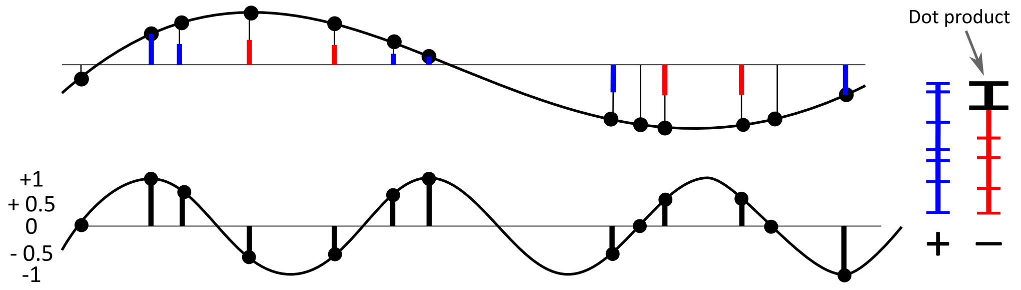

Figure 7.3: Dot product of two base functions (nonuniform sampling, more samples).

Having the larger number of non-uniformly distributed samples (due to higher sampling rate or longer signal), the statistical features of the sampling process get better, and the dot product of base functions decreases. Therefore, the base functions are better orthogonal and the harmonic monofrequency signal less influences the other spectral lines.

Positive terms are in blue, negative in red. In the right side of the picture, there is a sum of the terms forming the final dot product (it is not shown since it is zero).

7.2 Current Methods for Fuzzy Aliasing Elimination

The problem of fuzzy aliasing is not new, therefore, several methods for its elimination exist in the literature. One method is based on the computation and subtraction of the cross-interference between spectral lines. This method is described in [23, chapter 18]. Another method is based on iterative approximation of the signal, it is described in [23, chapter 20.2.2] and [25, chapter 4.1]. Main principles and features of both methods are described in the following sections.

7.2.1 Cross interference method

According to [23, chapter 18.1.1], cross-interference can be computed and then subtracted from the results of the Fourier transform.

The estimation of particular Fourier coefficient is influenced by other coefficients as described by equation 7.5.

$$\begin{aligned}
\widehat{a_i} &= \sum_{k=0}^{K-1} \left(a_k \left(A_i C_k\right) + b_k \left(A_i S_k\right)\right) \\
\widehat{b_i} &= \sum_{k=0}^{K-1} \left(a_k \left(B_i C_k\right) + b_k \left(B_i S_k\right)\right) \\
i &= 0, 1 \ldots (K-1) \\
K &\ldots \text{ number of spectral components} \\
\widehat{a_i},\ \widehat{a_i} &\ldots \text{ estimations of Fourier coefficients} \\
a_k,\ b_k &\ldots \text{ Fourier coefficients} \\
A_i C_k,\ A_i S_k,\ B_i C_k,\ B_i S_k &\ldots \text{ cross} - \text{interference coefficients}
\end{aligned} \tag{7.5}$$

This influence is characterized by the cross-interference coefficients (equation 7.6). Each coefficient characterizes the influence of two spectral lines to each another. Coefficients are dependent on the particular instance of the sampling process, but independent on the sampled signal.

$$
\begin{aligned}
(A_iC_k) &= \frac{2}{N}\sum_{n=0}^{N-1}\left(\cos\left(2\pi f_k t_n\right)\cdot\cos\left(2\pi f_i t_n\right)\right)\\
(B_iC_k) &= \frac{2}{N}\sum_{n=0}^{N-1}\left(\cos\left(2\pi f_k t_n\right)\cdot\sin\left(2\pi f_i t_n\right)\right)\\
(A_iS_k) &= \frac{2}{N}\sum_{n=0}^{N-1}\left(\sin\left(2\pi f_k t_n\right)\cdot\cos\left(2\pi f_i t_n\right)\right)\\
(B_iS_k) &= \frac{2}{N}\sum_{n=0}^{N-1}\left(\sin\left(2\pi f_k t_n\right)\cdot\sin\left(2\pi f_i t_n\right)\right)\\
&N\ldots\text{ number of samples}\\
&t_n\ldots\text{ sampling instants}
\end{aligned}
\tag{7.6}
$$

System of equations 7.5 can be rewritten to the matrix form (7.7), (7.8) and solved (7.9), giving values of the Fourier coefficients without effect of the cross-interference.

$$
\begin{pmatrix} \widehat{a_1} \\ \widehat{b_1} \\ \widehat{a_2} \\ \widehat{b_2} \\ \ldots \\ \widehat{a_K} \\ \widehat{b_K} \end{pmatrix} = \mathbf{C}\cdot\begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \\ \ldots \\ A_k \\ b_K \end{pmatrix}
\tag{7.7}
$$

$$\mathbf{C}\ldots\text{ matrix of cross} - \text{interference coefficients}$$

$$
\mathbf{C} = \begin{pmatrix} A_iC_k & B_iC_k & A_{i+1}C_k & B_{i+1}S_k \\ A_iS_k & B_iS_k & A_{i+1}S_k & B_{i+1}S_k \\ A_iC_{k+1} & B_iC_{k+1} & A_{i+1}C_{k+1} & B_{i+1}S_{k+1} \\ A_iC_{k+1} & B_iC_{k+1} & A_{i+1}C_{k+1} & B_{i+1}S_{k+1} \end{pmatrix}
\tag{7.8}
$$

$$
\begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \\ \ldots \\ a_K \\ b_K \end{pmatrix} = \mathrm{inv}(\mathbf{C})\cdot\begin{pmatrix} \widehat{a_1} \\ \widehat{b_1} \\ \widehat{a_2} \\ \widehat{b_2} \\ \ldots \\ \widehat{a_K} \\ \widehat{b_K} \end{pmatrix}
\tag{7.9}
$$

$$\mathbf{C}\ldots\text{ matrix of cross} - \text{interference coefficients}$$

Computation complexity of this method can be estimated in the following way (N – number of samples, M – length of the signal in clock ticks dt and number of frequency steps, K – number of spectral components/lines):

- Computation of NDFT: $O(M\cdot\log(M))$ (computation using FFT) or $O(N^2)$ (according to definition).

- Computation of one cross-interference coefficient: $O(N)$ (there are four coefficients for one frequency pair).

- There are M^2 frequency pairs. It is necessary to compute approximately half of the coefficients since cross-interference between frequencies in the pairs $[f_k, f_m]$ and $[f_m, f_k]$ is the same.
- Inverting the matrix of cross-interference coefficients: $O(M^3)$ (generally known Gauss-Jordan elimination method; a bit faster method exist).
- Matrix multiplication: $O(M^2)$.
- **Total:** $O(N^2 + 4 \cdot M^2 \cdot N + M^3 + M^2)$.

This method promises good results since it analyses the principles of the cross-interference and using inverse transform it removes the distortion of results. Main disadvantage of this method is inversion of the large matrix causing high computation complexity (especially if the cross-interference matrix is not precomputed) and high memory requirements. Issues with precision of numeric computations are also highly probable.

Experiments were performed with this method in order to check its usability for our problem. The method was implemented in GNU Octave and tested with our spectrograms. For solving equation 7.9, Octave operator backslash[3] was used. Very often, it gave warning about numerical instability, even though, it should behave more stable than matrix inversion followed by matrix multiplication. The results of computations were the nearly same spectrograms as the input ones (no spurious peaks disappeared), therefore, **the method was recognized as not suitable for solving our problem**.

7.2.2 Iterative method

In paper [25, chapter 4.1], there is described iterative method for fuzzy aliasing removal. This method uses forward and inverse Fourier transforms for approximation of the signal and better alias suppression.

1. Signal is non-uniformly sampled.
2. Fourier transform is computed from samples (either NDFT or FFT with zeros inserted instead of ”missing” samples).
3. Following steps are repeated:
   (a) Peaks higher than some threshold are selected, other peaks are replaced by zeros. This threshold is lowered in each iteration.
   (b) Optional cross-interference suppression is done (according to [23, chapter 18])
   (c) Inverse (equidistant) Fourier transform is computed from the results of the previous step. It produces (equidistant) approximation of the signal.
   (d) Instead of approximated samples are inserted samples from step 1 where available.
   (e) Fourier transform (equidistant) of this modified signal approximation is computed.
4. Computations are interrupted when energy of peaks lower than threshold in step 3.a is low enough.
5. Results of Fourier transform from the last iteration are used as final results.

Computation complexity of this method can be estimated in the following way (N – number of samples, M – length of the signal in clock ticks dt, K – number of iterations):

- First computation of NDFT: $O(M \cdot \log(M))$ (computation using FFT) or $O(N^2)$ (according to definition).

[3] Operator backslash in GNU Octave is optimized implementation of matrix inversion and multiplication
$\mathrm{inv}(\mathbf{A}) \times y = \mathbf{A} \setminus y$ with improved stability.

- Repeated computation of IDFT (IFFT): $O(K \cdot M \cdot \log(M))$ (K iterations).
- Repeated computation of DFT (FFT): $O(K \cdot M \cdot \log(M))$ (K iterations).
- **Total:** $O((2K+1) \cdot M \cdot \log(M))$.

This method suppresses the influence of the strong peaks to the weaker peaks. It can be advantageous for processing of the signals with many peaks with various amplitudes. However, in our case it gives no advantage since we need only a few of the greatest peaks. If we had used this method, we would have obtained results in the first iteration, but these results would have been the same as we would have obtained without usage of this method. Therefore, **this method is completely and definitely not usable for our problem**.

7.3 Trellis Post-processing Method

As shown above, spurious peaks caused by the fuzzy aliasing occur in the spectrograms computed using NDFT. For short signals, identification of the useful signal frequency f_p is problematic due to these spurious peaks. Removing them could be somehow possible, but fuzzy aliasing elimination methods described in the literature were found inappropriate in our case. However, those methods were originally intended for spectral analysis, therefore, they try to remove fuzzy aliasing from the whole spectrogram. Contrary, for my method is sufficient to select the correct peak without the removal of the others. This allows development of a simpler method because the useful frequency f_p changes slowly between bursts (spectrogram instances). Therefore, selection of the useful frequency f_p can be simplified using the knowledge of history. Such selection method is proposed in this chapter.

The problem is formally defined in subsection 7.3.1, solved in subsection 7.3.2 using *Trellis post-processing method*, and features of this new method are investigated in following three subsections.

7.3.1 Problem definition

Let us have a set of signal bursts $x(t)$ sampled according to conditions described in chapter 2. After computation of non-uniform Fourier transform (NDFT), we get a set of spectrograms (amplitude spectra) $|\mathbb{X}_i(f)| = \mathfrak{Abs}(\mathbb{X}_i(f))$ (one spectrogram per burst per signal). Each of these spectrograms consists of:

- Several peaks originating from fuzzy aliasing. These peaks are at random positions.
- One peak $|\mathbb{X}_i(f_p)|$ corresponding to signal frequency f_p. Position of this peak is slowly changing from spectrogram to the next one.

Each peak $|\mathbb{X}_i(f_m)|$ is a local maximum in spectrogram $|\mathbb{X}_i(f)|$, and one of the peaks is a global maximum (it can be at f_p or elsewhere) — see figure 7.4. We have chosen N greatest local maxima.

For easier notation, we mark the peaks as $P_{i,j}$ where i is number of the spectrogram and j is number of the maximum in spectrogram; we mark the position (i.e. frequency) of the peak as $|P_{i,j}|$ and value of the peak as $||P_{i,j}||$.

The main task is to select a sequence of peaks $\{P_{0,j}, P_{1,j}, P_{2,j}, \dots\}$ that has $\{|P_{0,j}|, |P_{1,j}|, |P_{2,j}|, \dots\}$ as similar as possible to sequence of signal frequencies $\{f_{p0}, f_{p1}, f_{p2}, \dots\}$. Signal frequency f_p is slowly changing.

Informally said, we have a sequence of wanted peaks at slowly changing positions and more unwanted peaks at random positions, and we want to identify the wanted peaks even though unwanted ones are higher.

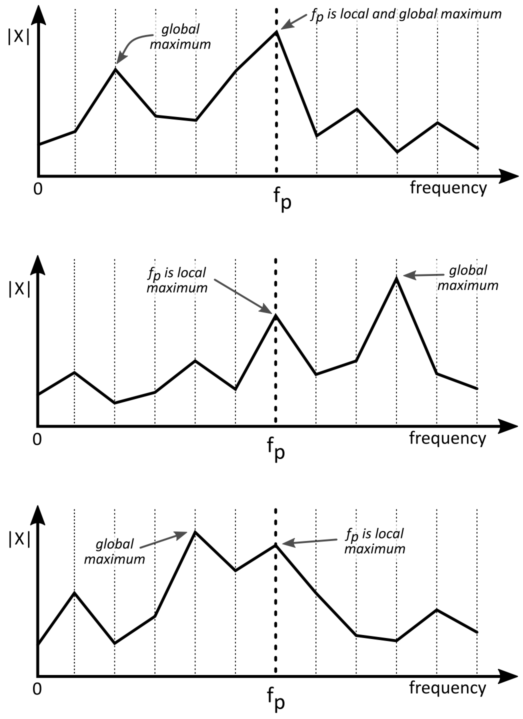

Figure 7.4: Frequency f_p corresponds to local maximum in spectrogram.

7.3.2 The Method

Let us have sets of peaks as described in chapter 7.3.1. We can draw the set of peaks in a manner that spectrogram number is on the x-axis and peak positions are on the y-axis. This way we can think about it as about directed acyclic graph. Peaks are nodes of the graph; edges are connections between peaks $P_{i,j}$ and $P_{i+1,k}$ (see figure 7.5). The edges are weighted by distance $d_{i,j:k} = \mathrm{norm}(|P_{i,j}|, |P_{i+1,k}|)$. As a norm for edge weighting, any norm $\mathrm{norm}(|Pi, j|, |Pi + 1, k|) = (\mathrm{abs}(|Pi, j|, |Pi + 1, k|))^p$; $p \geq 1$ can be used (see chapter 7.3.4 for further details)[4]. The nodes are weighted by cumulative sum of the edge weights traveled along the way to this node $w_{i+1,k} = w_{i,j} + d_{i,j:k}$; $w_{0,*} = 0$. [5] [6]

If $i < j$ than we say that peak/node $P_{i,*}$ is older than $P_{j,*}$ and spectrogram i is older than spectrogram j.

If position of the wanted peaks changes slowly and position of other peaks is random, the path going along wanted peaks would probably be the lightest path in the graph, i.e. $\min(w_{\infty,*})$. This path (we call it *correct path*) can be identified by backtracking the path from the node with the least cumulative weight.

Note: This is valid only if peaks are sparse distributed. Details are further studied in section 7.3.3.

Let us have N peaks/nodes $P_{i,*}$ and N succeeding nodes $P_{i+1,*}$; there are N^2 edges between them. These edges go to each node $P_{i+1,*}$ from all preceding nodes (see figure 7.6). If we are looking for the lightest path, it is sufficient to track only N edges – for each node $P_{i+1,*}$, only one edge that is part of the lightest path to this node is enough (see solid lines in figure 7.6). These edges can be found by evaluating

$$w_{i+1,k} = \min_j(w_{i,j} + d_{i,j:k}) = \min(w_{i,1} + d_{i,1:k}; w_{i,2} + d_{i,2:k}; \ldots; w_{i,n} + d_{i,n:k}) \tag{7.10}$$

[4] Value $p = 1/2$ was alo tried.

[5] Asterisk ($*$) in indexes (e.g. $P_{i,*}$ or $w_{0,*}$) is used as a placeholder saying *any valid value.*

[6] It would be formally more proper to weight the nodes by $\sqrt[p]{w_{i+1,k}}$, but for purposes of my method, it is enough to use $w_{i+1,k}$ since for any $a < b < c; a, b, c > 0$ holds $a^p < b^p < c^p; p > 1$.

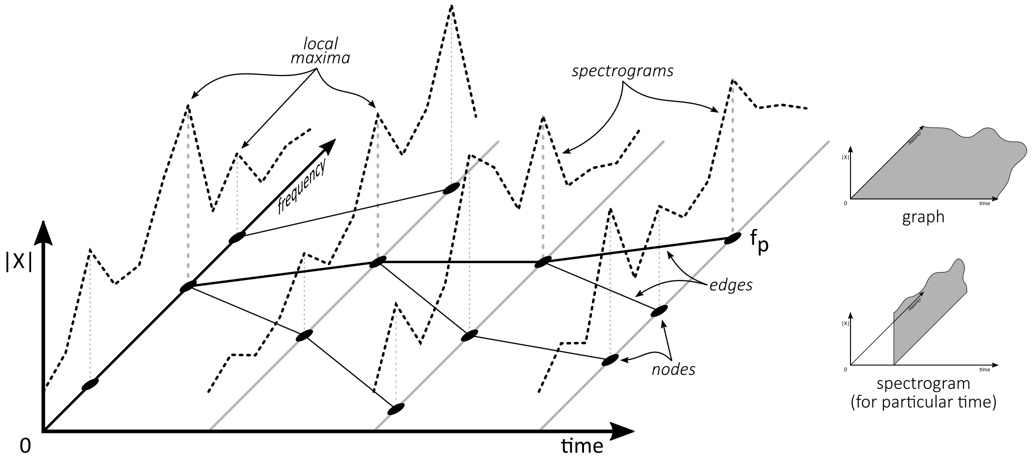

Figure 7.5: Set of peaks can be seen as a graph.

$w_{i,j}$ . . . weight of node $P_{i,j}$

$d_{i,j:k}$ . . . weight of the edge from node $P_{i,j}$ to $P_{i+1,k}$

$(w_{i,j} + d_{i,j:k})$ . . . weight of the node $P_{i+1,k}$ using the path via node $P_{i,j}$

n . . . number of nodes (peaks) per spectrogram

If there is no edge from peak $P_{i,j}$ to $P_{i+1,*}$, no path leading to $P_{i,j}$ need to be tracked any more, and paths leading from $P_{i+1,*}$ to the past starts to converge.

This way it is enough to track at most N paths in each step. If we backtrack all the tracked paths (i.e. all paths from $P_{now,*}$), we usually find, that all these paths converges to one paths somewhere (at $P_{conv,p}$) in the past.[7] Lightest path for all spectrograms older than *conv* is unambiguous, therefore, the most probable signal frequency f_p is identified for all spectrograms older than *conv* (see bold line in the figure 7.6). Path through spectrograms older than *conv* need not be tracked any more, and cumulative weight $w_{conv,p}$ can be subtracted from all tracked paths.

The convergence length $now - conv$ could theoretically be infinite (i.e. non-converging paths), but usually is much shorter and depends on particular nodes and on the norm used for computation of distances.

It is difficult to look for convergence point $P_{conv,p}$ in each step (backtrack N paths and look for their intersections). Therefore, for practical implementation is easier to assume that convergence length is a constant. This constant should be set a bit greater than the value of real convergence length in most cases. Using this simplification, f_p can be identified in each step by backtracking from any node. Advantage of this simplification is lower computation complexity and constant delay of results. Disadvantage is the fact that delay of results is in many cases greater than a real convergence length.

7.3.3 Probability of False path

Let us carry out computations as described higher in this chapter. We mark maximal allowed change of frequency f_p in one step as λ and signal bandwidth as B. Cumulative weight w_c of the correct path with length k is limited by[8]

$$\begin{aligned} w_c &= k \cdot \text{norm}(\lambda) \\ \text{norm}(\lambda) &= (\lambda)^p \ ; \ p \geq 1 \end{aligned} \tag{7.11}$$

For easier computation, we describe λ as a fraction of bandwidth $\lambda = \frac{B}{2 \cdot r}$. Let's assume uniform distribution of unwanted peaks.

[7] There could be more non-converging paths, but it is improbable due to random distribution of the most of the nodes.

[8] It would be formally more appropriate to use $w_c = \sqrt[p]{k \cdot (\lambda)^p}$, but we must use the same norm as we use for path weighting, i.e. $w_c = k \cdot (\lambda)^p$.

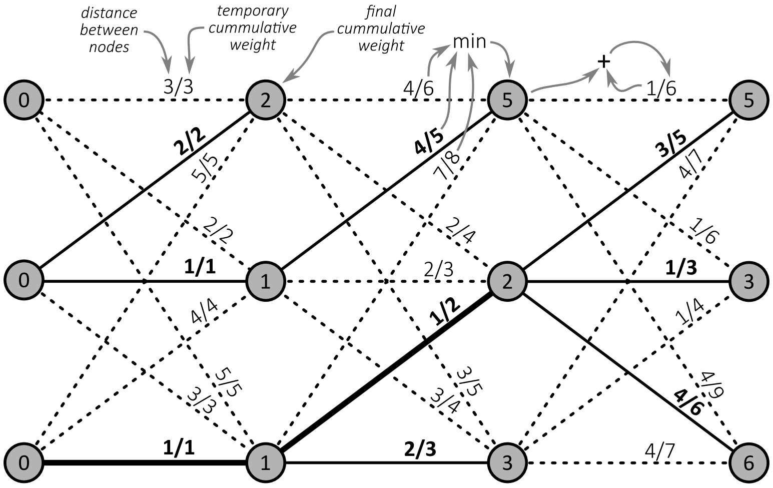

Figure 7.6: Edges between nodes $P_{i,*}$ and $P_{i+1,*}$.
Nodes $P_{i,*}$ and $P_{i+1,*}$ are connected by N^2 edges. It is enough to track only one edge to each node (solid line). Unambiguous part of the path (i.e. older than conv) is drawn bold.

We say that path is *false path* if it contains unwanted peaks, and distances between peaks are at most λ. False path is indistinguishable from correct path. Cumulative weight of the false path is limited by the same value as for the correct path.

Probability that at least one unwanted peak appears in band $2 \cdot \lambda$ wide is

$$p(1) = n \cdot \frac{2 \cdot \lambda}{B} = n \cdot \frac{\frac{B}{r}}{B} = \frac{n}{r} \tag{7.12}$$

$B \ldots$ signal bandwidth

$n \ldots$ number of peaks in one step

$r \ldots$ parameter used for description of λ as a fraction of B ($r = \dfrac{B}{2 \cdot \lambda}$)

Probability of existence of false path with length k is probability that at least one peak falls to the band wide $2 \cdot \lambda$ in all k successive steps (there is n starting nodes of the false path):

$$p(k) = n \cdot \left(\frac{n}{r}\right)^k \tag{7.13}$$

Table 7.1 contains probabilities of false path occurrence for various numbers of peaks.

Another unwanted type of the path is *light bad path.* This path has smaller cumulative weight than correct path, but distance between at least two peaks is greater than λ. This path is shown in the picture 7.7. Light bad path is distinguishable from correct path during backtracking (due to distance between peaks greater than λ).

7.3.4 Selection of proper norm

The norm used for computation of $d_{i,j:k}$ shall be carefully selected. If this norm prefers small distances, e.g. $\text{norm}(|P_{i,j}|, |P_{i+1,k}|) = (|P_{i,j}| - |P_{i+1,k}|)^4$, long edges increment cumulated weight too much. Therefore, convergence length is too long. If norm used for computation of $d_{i,j:k}$ prefers long distances, e.g. $\text{norm}(|P_{i,j}|, |P_{i+1,k}|) = \sqrt[4]{(|P_{i,j}| - |P_{i+1,k}|)}$, paths quickly converge to the one with the least cumulated weight, regardless if this path is the best. Results could be incorrect due to higher initial (slowly cumulated in past) cumulative weight. Therefore, norm must be selected somewhere between these extreme cases; generally, most common norms are $\text{abs}(b - a)$ and $(b - a)^2$.

$\mathbf{r = 10}$	$\mathbf{n = 2}$	$\mathbf{n = 3}$	$\mathbf{n = 5}$	$\mathbf{n = 7}$
$\mathbf{k = 2}$	0.08	0.3	1	1
$\mathbf{k = 5}$	10^{-3}	0.007	0.15	1
$\mathbf{k = 10}$	10^{-7}	10^{-5}	0.004	0.2
$\mathbf{k = 20}$	10^{-14}	10^{-10}	10^{-5}	0.06

$\mathbf{r = 50}$	$\mathbf{n = 3}$	$\mathbf{n = 5}$	$\mathbf{n = 10}$	$\mathbf{n = 15}$
$\mathbf{k = 2}$	0.01	0.05	0.4	1
$\mathbf{k = 5}$	10^{-6}	10^{-4}	0.003	0.04
$\mathbf{k = 10}$	10^{-12}	10^{-9}	10^{-6}	10^{-4}
$\mathbf{k = 20}$	10^{-24}	10^{-19}	10^{-13}	10^{-9}

Table 7.1: Probability of false path.

Value $r = 10$ means that f_p changes at most by 5% of B in one step, $r = 50$ means 1% of B, n is number of peaks in one step, k is the length of false path.

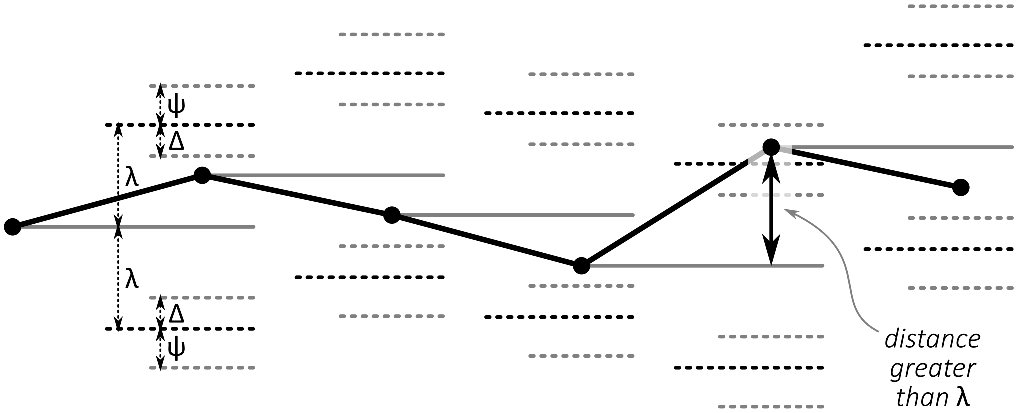

Figure 7.7: Light bad path

Light bad path is lighter than correct path, but distance between at least two peaks is greater than λ.

$$4 \cdot \mathrm{norm}(\lambda - \Delta) + \mathrm{norm}(\lambda + \psi) < 5 \cdot \mathrm{norm}(\lambda)$$

7.3.5 Computation complexity

Computation complexity of extending trellis by one new spectrogram can be computed in the following way: During computation of the best path to one new node, paths from all preceding nodes are compared, i.e. N paths. This computation is done for each new node, i.e. N times. Therefore, computation complexity is $O(N^2)$.

Computation complexity of searching minimum of N values is $O(N)$. Computation complexity of backtracking K steps to history along one path is $O(K)$.

7.4 Experimental Evaluation of Trellis Post-processing Method

To confirm properties of the trellis post-processing method, various numerical experiments were carried out. Data for these experiments were generated using following two approaches:

- Array of random numbers (uniform distribution) was generated. These numbers represent simulated positions of unwanted peaks ($|P_{i,j}|$). Then the positions of the wanted peaks (i.e. $f_{p,i}$) are generated and added to this array. This array is used instead of a result of spectra computation and maxima identification. Main advantage of this method is the fact, that properties of the input data can be precisely controlled.

- Pair of input signals is generated and sampled, NDFT is computed, peaks in spectrograms identified and used as input data for method evaluation. This approach gives almost real data since it corresponds to intended use-case of method, however, input data of the trellis method cannot be so preciously controlled. It also requires much more computations than the other approach.

With growing number of nodes (per spectrogram) amplitude of error decreases, therefore, squared error between proper path and identified path cannot be used for comparison of results. Instead of it, the number of incorrectly identified nodes is used as a metric. In following sections, number of mistakes is normalized by the number of spectrograms – results are in range $\langle 0, 1 \rangle$.

7.4.1 Number of peaks versus λ

If λ (max. distance between nodes along the correct path) grows up to $\frac{B}{2 \cdot N}$ (N is number of peaks per spectrogram), the probability of false paths grows. Dependency between these values and error was evaluated. One result of this experiment is shown in the figure 7.8. Results of this experiment confirmed theoretical assumption, that the best results are the ones with the lowest possible λ and N. Experiments showed, that for $\lambda \approx \frac{B}{2 \cdot N}$ (false path surely occurs), approximately one third of nodes is incorrectly identified. If λ is half of this critical value ($\lambda \approx \frac{B}{4 \cdot N}$), there is approximately 10% of mistakes.

7.4.2 Proper norm

For computation of distance between two peaks could be used various norms. Computations with same data and conditions[9] were carried out for several norms. One of the results is shown in the figure 7.9. Experiments showed that well-known norm $(b-a)^2$ is appropriate for usage in this method. Basic distance between nodes $\mathrm{abs}(b-a)$ gave worse results in all cases. Norm $\sqrt{\mathrm{abs}(b-a)}$ gave results very similar to $\mathrm{abs}(b-a)$. Norms $(b-a)^q$ with exponent $q = 4$ or greater gave results negligibly better than $(b-a)^2$.

[9] Generator of pseudo-random numbers was also initiated to the same state.

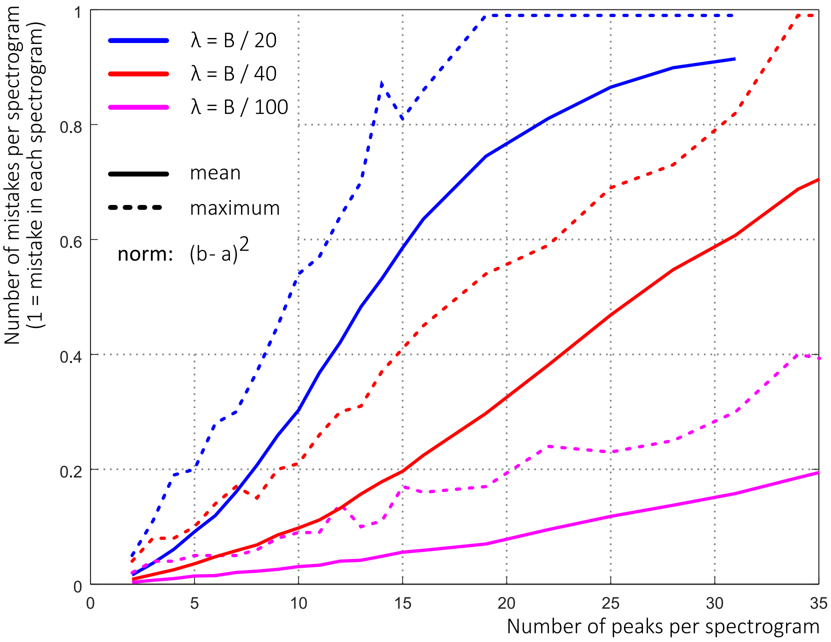

Figure 7.8: Dependency between ratio $\frac{B}{2 \cdot N}$ and number of mistakes for constant λ. *Fraction* $\frac{B}{2 \cdot N}$ *is a half of the average bandwidth occupied by one node,* B *is a signal bandwidth,* N *is the number of nodes, and* λ *is maximal distance between nodes on the correct path.*

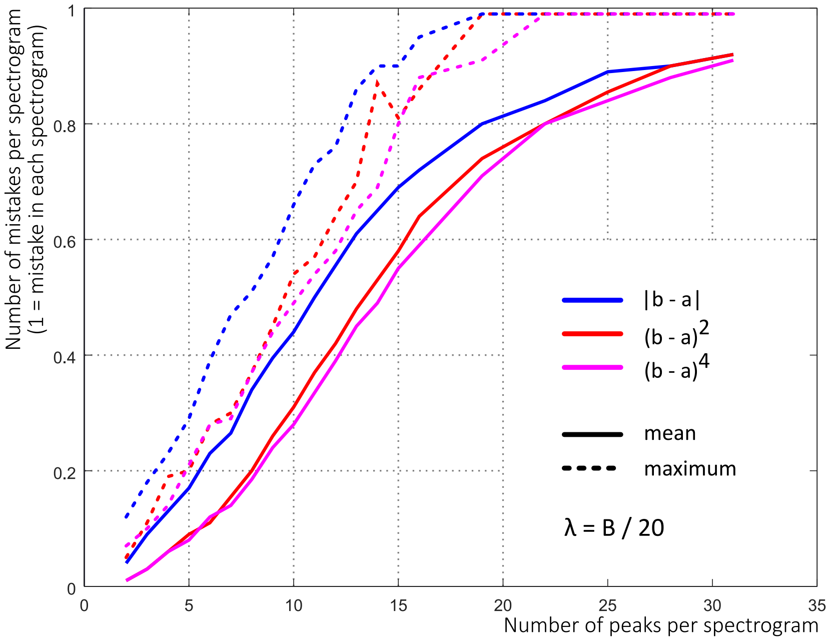

Figure 7.9: Number of mistakes for various norms. *Norm* $\sqrt{\mathrm{abs}(b-a)}$ *was also evaluated giving results very similar to* $\mathrm{abs}(b-a)$. *Norms* $(b-a)^q$ *with exponent* q *greater than 4 gave results very similar to* $(b-a)^4$. B *is a signal bandwidth.*

7.4.3 Real data

Method was many times applied on data, that were simulation of data from real application. Resulting graphs were compared with correct results in order to evaluate whether the method gives appropriate results in real conditions. Results confirmed that the method works as expected in the cases when peaks corresponding to the frequency f_p are present in spectrograms. Example can be seen in the figure 7.10. Although this figure is quite similar to figure 7.5, it is a result of computations and not a drawing.

If nodes corresponding to f_p are missing[10], method often gives incorrect results even for several neighbor spectrograms. This is inherent feature of the trellis postprocessing since no path can be routed via the correct node in case this node/peak does not exist. Instead, the path is routed via existing nodes minimizing its weight, see figure 7.11. Absence of the node could be caused by one of the following reasons:

- Parameter N (number of processed peaks per spectrogram) is to low and correct peaks are not added to set of N greatest nodes.[11]
- Values of the cross-interference coefficients caused accidental suppression of the correct peak.
- A random error or bad settings in the signal processing chain caused than spectrogram does not show properties of the signal.

7.5 Trellis Post-processing Method – Conclusion

If the non-uniform Fourier transform is used for spectra computation, some distortion of the results may be present. This distortion is called *fuzzy aliasing* and causes spurious peaks in spectrum. The general fuzzy aliasing removal methods were evaluated and found not to be suitable for our problem. Nevertheless, situation in our case was simplified by the fact, that position of the correct peak changes slowly between spectrograms. This motivated me to develop a new method, that does not remove fuzzy aliasing from spectrogram, but selects the correct peak. Using this approach, I converted difficult signal processing problem to easier graph problem and solved it. My novelty Trellis post-processing method not only solves the problem of frequency identification in presence of fuzzy aliasing in NDFT delay measurement method (that was the motivation), but also simplifies tracking of the useful/interesting frequency in similar use cases (e.g. monitoring of spectrum). It also seems possible to modify Trellis post-processing method for tracking of multiple frequencies, if needed.

The experiments proved, that my Trellis post-processing method can track the correct frequency in the sequence of spectrograms. This makes possible usage of the NDFT delay processing method for measurement of delays between signals with unknown frequency f_p.

[10] In the spectrogram, there is no local maximum for useful frequency f_p, or local maximum is too small.

[11] In case of increasing N, take care about ratio of λ and $\frac{B}{2 \cdot N}$.

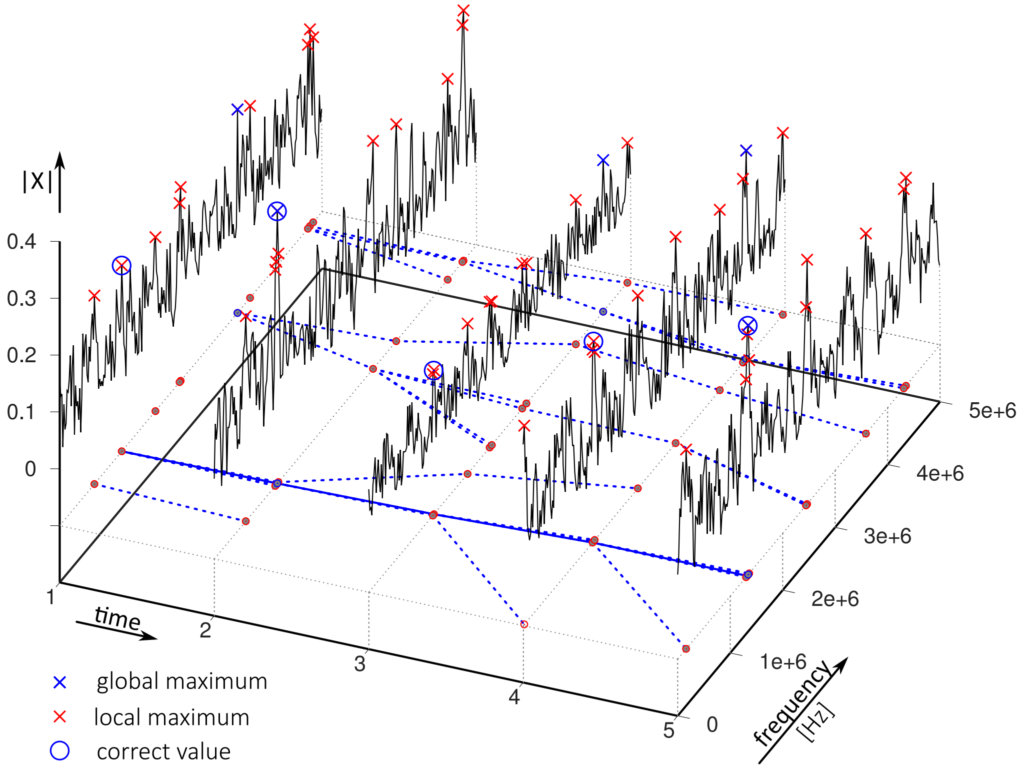

Figure 7.10: Graph computed from real data.

Signal was simulated according to chapter 2.1, frequency f_p was slowly changing around 1.25 MHz. Signal was noisy with SNR = 6 dB. Sampling was non-uniform with average sampling frequency f_s = 700 kHz and time resolution Δt = 100 ns. Number of processed local maxima $N = 10$.

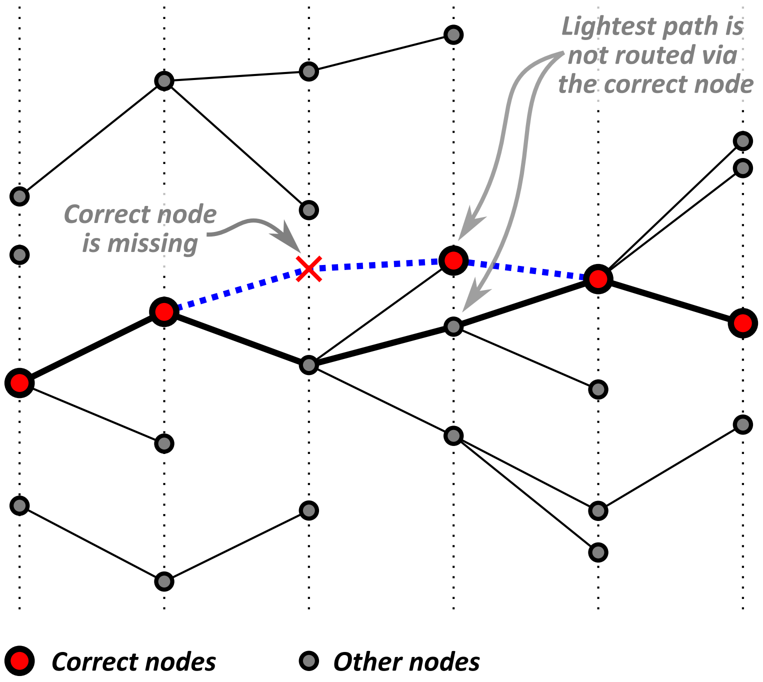

Figure 7.11: Node corresponding to the frequency f_p is missing.

Peak/node corresponding to the useful frequency is missing in one spectrogram, therefore, the lightest path is routed via other nodes, incorrectly identifying the useful frequency in this and in the next spectrogram. Dotted line corresponds to the path via missing node, bold line is the lightest path.

Chapter 8

Conclusion

8.1 Results of Research

The main goal of this work is digital measurement of delays between analogue signals using low number of samples. As the first step, contemporary methods were studied and evaluated. It was found that only several main principles are used by the most of methods. As the best candidate for my improvement was recognized the phase shift method – both signals are sampled, and their frequencies and phase shifts are found using the Fourier transform. From the phase shifts and frequencies, a time shift is computed. Computed time shift is shorter than a half period of the signal. For the longer shifts, the computed result could differ from the real result by an integer multiple of periods.

Using a non-uniform Fourier transform, the phase shift method can be used with non-equidistant sampling. Usage of the non-uniform sampling brings an advantage of a theoretically infinite sampling frequency. Considering finite resolution of a time measurement, latent sampling frequency is introduced to describe an unaliased signal bandwidth. Shortest measurable time interval is called *clock tick*. In Nyquist's theorem, this clock tick can be used instead of the sampling period. An average sampling frequency has no effect on the aliasing. A position of samples can be described as a number of clock ticks from the beginning. Using this definition and a general definition of the discrete Fourier transform, equation of the non-uniform discrete Fourier transform is derived.

The phase shift method using the non-uniform discrete Fourier transform can be used for delay measurement using decreased amount of the samples and low average sampling frequency. But as a number of samples decrease, statistic features of the sampling process get worse and *fuzzy aliasing* appears. It is caused by a random coincidence of the sampling instants mostly in the positive or negative parts (e.g. half-periods) of the signal. This effect gets more probable with decreasing number of samples. Fuzzy aliasing is not a new problem, therefore, methods for its elimination exist in the literature. Nevertheless, these methods were found not suitable for our use-case. But our use-case has a feature, that could be used for elimination of the effects of the fuzzy aliasing – we are processing a sequence of the waveforms, whose frequency changes slowly between following ones.

The fuzzy aliasing appears as a one or several spurious peaks in the spectrogram, making identification of the signal frequency more difficult. Precisely said, it disturbs all components in spectrum, but in our use-case, it is problem only if it causes large spurious peak (problem with identification of the signal frequency). It also could have influenced the correct peak causing the phase shift measurement error. This error is small (high power signal influenced by the one with lower power), less probable than spurious peaks, and harder to eliminate. Therefore, I focus on elimination of the spurious peaks, only.

Having the sequence of the spectrograms computed from the sequence of waveforms, the position of the correct peak (the one corresponding to the signal frequency) is changing slowly between spectrograms. Positions of spurious peaks are "random". For identification of the correct peaks, the trellis post-processing method can be used. From each spectrogram, a set of suspicious peaks is selected and used as nodes of a graph. The nodes (peaks) of the following spectrograms are connected, the edges are weighted by frequency change between nodes (peaks). As positions of most of the peaks are "random", the lightest path will go along the correct peaks since it corresponds to the smallest frequency changes. The method was tested with simulated data from real appliance and results showed, that method performs well with exception of cases with suppressed correct peak in spectrogram. In that case the method might influence also results from several neighboring spectrograms, because path cannot be routed via non-existent peaks.

The construction of a dedicated AD converter for non-uniform sampling is improbable, but some microcontrollers are equipped with several synchronized general-purpose AD converters. Author's conference paper [26] analyzes one example of such microcontroller and possibilities to use it for non-uniform sampling. It was found that ADCs in this microcontroller (STM32-F427) triggered by an internal timer are able to perform non-uniform sampling, widening an unaliased input bandwidth of the ADC (in comparison to the bandwidth obtained by periodic (equidistant) sampling). It was found that in case of non-uniform sampling, a limitation of input bandwidth is not caused by sampling itself (i.e. by an aliasing), but by an attenuation of higher frequencies in the sample&hold circuits of ADC. These results are also in contradiction with [23, pp. 58] for this microcontroller (STM32-F427). There is written that a bandwidth of analogue inputs of common AD converters is typically 4 to 8 times higher than half of the maximal (periodic) sampling frequency of the AD converter, but it is not true in this case.[1]

This thesis showed that phase shift method with non-uniform sampling and non-uniform discrete Fourier transform with trellis post-processing could be effectively used for measurement of delays between fast signals using the reduced average sampling frequency and reduced amount of the samples. Main principles and features of NDFT phase shift method with trellis post-processing are described in this thesis, but more work must be done to make it widely usable. The main topics for future work are described in the following chapter.

8.2 Topics for Further Work

Previous chapter summarizes the results of the research that are described in this work. Even though this work studies and describes the main principles and features of NDFT phase shift method with trellis post-processing, more work must be done to make the method widely usable. The main topics for future research are described in this chapter.

One main topic for future work is necessity to optimize implementation of the method. For the experiments, method was implemented in GNU Octave, sometimes using 'naive' implementations, and for computations was used floating point arithmetic. Therefore, for real use, better and more optimized implementation is necessary. We used computation of non-uniform discrete Fourier transform (NDFT) according to the definition, but faster approximation algorithms exist, for examples see [20–22]. These algorithms need to be evaluated and eventually modified for our use case. In order to use the method in low-cost devices, it could be also beneficial to use integer arithmetic.

Author's theoretical study [26] shows non-uniform sampling using general-purpose STM32-F4xx microcontroller. But in some applications, it could be better to use another microcontroller type/family. In such cases, similar study will be needed for respective microcontroller. For particular application, the theoretically computed features must be proven by measurement, preferably on samples of microcontrollers from several manufacturing batches. In case of non-uniform sampling using STM32-F4xx microcontroller, bandwidth limitation is caused by analogue features of ADC inputs (and not by aliasing, that occurs at higher frequency). This means that it might (or might not) be possible to get even wider bandwidth, than theoretically computed, by constraining operating conditions (e.g. temperature, operating voltage),

Many microcontrollers are equipped with AD converters working at a sampling frequency of hundreds of kilosamples. Using my methods, it seems to be possible to measure the delay between signals at frequencies of several megahertz using these cheap devices.

[1] Analysis mentioned in this paragraph is not part of this thesis, for a draft of the conference paper [26] see Appendix.

8.3 Fulfilment of the Objectives

For this work, several objectives were set-up. In this chapter, fulfillment of these objectives is evaluated.

1. **Investigate the current state of the art of delay measurement and examine principles of those methods.**

   and

2. **Compare the existing methods each another. Evaluate their suitability for improvements described in the goal 2.**

   Description of the state-of-art is in chapter 4, brief description of the interesting methods is also there. It was found, that there are lots of methods described in the literature, however, nearly all of them are slight modifications of one of the well-known basic method. Therefore, these methods were classified into several groups and basic methods on their background were examined. Some groups of methods promised usability for our problem; these methods were deeper examined, implemented, tested on our data, and results of particular methods were compared each another. Results of comparison were published in author's conference paper [15]. The most promising method – the *phase shift method* – was chosen as a basic method for my modifications.

3. **Propose a method for time shift computation of waveforms using non-uniform sampling in order to decrease the number of samples and sampling frequency.**

   Delay measurement method based on phase shift computation was modified for non-uniform sampling. Non-uniform Fourier transform was derived. Method was published in author's conference paper [16]. In this thesis, it is described in chapter 6.

4. **Examine the relation between the number of samples, their resolution and accuracy of the results.**

   The properties of the non-uniform sampling and non-uniform Fourier transform were theoretically studied in chapter 6.2; formula for computation of latent sampling frequency was derived. Experimental evaluation of these properties is described in chapter 6.4. These results were published in [16].

5. **Implement and test the method in a near-to-real appliance. Compare the new method with contemporary delay measurement methods.**

   The method was implemented in Octave and tested on simulated data from real appliance. During the tests it was found, that there are problems with identification of the correct frequency of the signal in cases with low sampling rate and small number of samples. It was investigated that this issue is caused by *fuzzy aliasing* that introduces random spurious peaks to the spectrograms. This problem was analyzed and solved using new *Trellis postprocessing*, see chapter 7. Trellis postprocessing method was published in author's journal article [24].

6. **Adapt the method for the use of general-purpose AD converters.**

   It was found that some microcontrollers are equipped with synchronized ADCs. Author's conference paper [26] studies such microcontroller and shows that nonuniform sampling could be beneficially used to widen the input bandwidth of microcontroller's ADCs. A required sampling function is also specified in the paper, taking into account timing properties of a microcontroller and ADC.

For this Ph.D. thesis, following goals were established in chapter 3. Their fulfillment is evaluated in this section.

1. **To study and compare the methods for time delay measurement.** – Finished in objectives nr. 1 and 2.
2. **To develop new procedures for decreasing the sampling frequency and amount of sampled data necessary for time delay measurement.** – Finished in objectives nr. 3 and 5.
3. **To verify properties of the proposed method.** – Finished in objectives nr. 4 and 5.

Chapter 9

Summary

In many applications, precise measurement of short delays between analogue signals is required. But in many cases signal features are fixed and signal processing resources are limited, therefore, my goal was development of methods, that allows such measurements without expensive devices.

In the beginning of the thesis, the problem is specified (chapter 2) and contemporary delay measurement methods are studied (chapters 4 and 5). In the analysis of the contemporary methods the common features of the methods were described. Based on these results, I developed a novelty method using the non-uniform sampling and non-uniform discrete Fourier transform (NDFT) for computation of a phase shift and delay between signals (chapter 6). Numerical experiments were performed, and my NDFT method was shown to be preferable to the other methods for measurement of delays between signals sampled at low average sampling frequency. The main advantage of the method is its usability with very low average sampling rates. It works even with average sampling frequency lower than double of a signal frequency. This feature makes an advantage of my method against the other methods, because sampling rate can be reduced without narrowing a bandwidth of an input.

But as a number of samples decreased, problems with fuzzy aliasing appeared and made the frequency identification difficult. Therefore, I developed the Trellis post-processing method for frequency identification and tracking (chapter 7). Using this method, slowly changing frequency can be tracked in series of spectrograms containing ”random” spurious peaks. This novelty method not only solved the problem, but it could be probably used also for similar problems of frequency tracking.

Using my novelty methods, the delays between fast analogue signals can be measured using non-uniform sampling, that reduces average sampling frequency and amount of produced data. These methods bring theoretical background, that, if appropriately implemented and optimized, could make possible a great step forward towards usage of low-cost microcontrollers for processing of fast analogue signals.

Shrnutí

V mnoha aplikacích je třeba měřit krátká zpoždění mezi rychlými analogovými signály. Protože vlastnosti signálů jsou v mnoha případech dané, a zároveň máme omezené výpočetní prostředky, cílem této práce byl vývoj metod, které umožní provádět taková měření bez nákladných zařízení.

Na začátku práce je popsán problém (kapitola 2) a soudobé metody pro měření zpoždění (kapitoly 4 a 5). V analýze soudobých metod jsou určeny společné vlastnosti těchto metod, a na jejich základě je vyvinuta nová metoda využívající neekvidistantní vzorkování a neekvidistantní diskrétní Fourierovu transformaci (NDFT) pro výpočet fázového posuvu a zpoždění mezi signály (kapitola 6). Provedené numerické experimenty potvrdily, že při měření zpoždění mezi signály navzorkovanými s velmi nízkou průměrnou vzorkovací frekvencí je moje metoda s NDFT lepší než ostatní metody. Vyvinutá metoda je použitelná při velmi nízkých vzorkovacích frekveních, funguje dokonce i při vzorkovací frekvenci nižší než dvojnásobek frekvence signálu. Tato vlastnost je její hlavní konkurenční výhodou proti jiným metodám, neboť průměrná vzorkovací frekvence může být snížena bez omezení šířky pásma na vstupu.

Při snížení počtu vzorků ovšem nastává *fuzzy aliasing*, který komplikuje určování frekvence signálu, a proto byla následně vyvinuta *mřížková metoda* (*Trellis post-processing method*)[1] pro určení a sledování frekvence (kapitola 7). Tato metoda umožňuje sledování pomalu se měnící frekvence v sérii spektrogramů obsahujících „náhodně” rozmístěné klamné spektrální čáry. Tato nová metoda nejen vyřešila problém, ale pravděpodobně ji půjde použít též pro řešení obdobných problémů se sledováním frekvence.

Tyto nové metody slouží k měření zpoždění mezi rychlými analogovými signály s použitím neekvidistantního vzorkování a neekvidistantní diskrétní Fourierovy transformace, která umožňuje snížit průměrnou vzorkovací frekvenci a díky tomu též množství vyprodukovaných dat. Vyvinuté metody přinášejí teoretické základy, které, pokud budou vhodně implementovány a zoptimalizovány, mohou umožnit velký krok vpřed k použití levných mikrokontrolérů ke zpracování rychlých analogových signálů.

[1] *Trellis* je anglicky mříž nebo rošt pro upevnění popínavých rostlin.

Zussamenfassung

In vielen Anwendungen ist eine präzise Messung kurzer Verzögerungen zwischen Analogsignalen erforderlich. Dabei sind in vielen Fällen die Signaleigenschaften gegeben und die Ressourcen für die Signalverarbeitung begrenzt. Das Ziel dieser Arbeit war die Entwicklung von Methoden, die solche präzise Messungen ohne kostbare Geräte ermöglichen.

Am Anfang der Doktorarbeit wurde das Problem spezifiziert (siehe Kapitel 2) und die gegenwärtigen Verzögerungsmessungsmethoden beschrieben (siehe Kapitel 4 und 5). Bei der Analyse der gegenwärtigen Methoden wurden die gemeinsamen Eingenschaften dieser Methoden beschrieben. Auf Basis dieser Ergebnisse haben wir eine neue Methode entwickelt. Diese Methode benutzt nicht-äquidistante Abtastung und die nicht-äquidistante diskrete Fourier-Transformation (NDFT) zur Berechnung einer Phasenverschiebung und Verzögerung zwischen den Signalen (siehe Kapitel 6). Wir haben numerische Experimente mit unserer neuartigen NDFT-Methode durchgeführt. Die Messungen haben gezeigt, dass die NDFT Methode bessere Ergebnisse im Vergleich mit anderen Methoden erreicht hat, wenn das Signal mit einer niedrigeren durchschnittlichen Abtastrate abtastet wird. Der Hauptvorteil der Methode ist ihre Anwendbarkeit bei sehr geringen durchschnittlichen Abtastraten. Sie funktioniert sogar mit einer durchschnittlichen Abtastrate, die unter der doppelten Signalfrequenz liegt. Diese Eigenschaft bietet einen Vorteil der NDFT Methode gegenüber den anderen Methoden, da die Abtastrate reduziert werden kann ohne eine Einschränkung der Eingangsbandbreite.

Mit abnehmender Anzahl von Samples traten jedoch Probleme mit dem *Fuzzy-Aliasing* auf und erhöhten die Schwierigkeiten mit der Frequenzidentifikation. Aus diesem Grund wurde eine *Gitter-Nachbearbeitungsmethode* (*Trellis post-processing method*)[2] für die Frequenzidentifikation und einen Nachlauf entwickelt (siehe Kapitel 7). Mit dieser Methode kann eine sich langsam verändernde Frequenz in einer Reihe von Spektrogrammen mit "Random" Störspitzen verfolgt werden. Diese neue Methode löste nicht nur das Problem der NDFT Methode auf, sondern sie kann wahrscheinlich auch für ähnliche Probleme des Frequenznachlaufs verwendet werden.

Mit der neuen Methoden können die Verzögerungen zwischen schnellen Analogsignalen mit nicht-äquidistanten Abtastung gemessen werden, die die durchschnittliche Abtastrate und die Menge von produzierten Daten reduziert. Die hier entwickelten Methoden liefern eine theoretische Basis. Bei einer passenden Umsetzung und Optimierung können sie einen großen Schritt nach vorne auf dem Weg zum Einsatz von kostengünstigen Mikrocontrollern zur Verarbeitung schneller Analogsignale ermöglichen.

[2] *Trellis* bedeutet im Englisch ein Gitter für Festigung der Kriechpflanzen.

References

[1] Xiangwei Zhu, Guangfu Sun, Shaowei Yong, and Zhaowen Zhuang. A high-precision time interval measurement method using a phase-estimation algorithm. *Instrumentation and Measurement, IEEE Transactions on*, 57(11):2670–2676, Nov 2008. ISSN 0018-9456. doi: 10.1109/TIM.2008.925025.

[2] P. Pánek. Time-interval measurement based on SAW filter excitation. *Instrumentation and Measurement, IEEE Transactions on*, 57(11):2582–2588, Nov 2008. ISSN 0018-9456. doi: 10.1109/TIM.2008.925014.

[3] Ming-Chien Tsai and Ching-Hwa Cheng. A full-synthesizable high-precision built-in delay time measurement circuit. In *Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific*, pages 123–124, Jan 2009. doi: 10.1109/ASPDAC.2009.4796463.

[4] Xu Wei, Chen Qian, and Gu Guo-hua. Research on laser ranging system based on time delay estimation. In *Photonics and Optoelectronics (SOPO), 2011 Symposium on*, pages 1–3, May 2011. doi: 10.1109/SOPO.2011.5780683.

[5] Xiaoming Lai and H. Torp. Interpolation methods for time-delay estimation using cross-correlation method for blood velocity measurement. *Ultrasonics, Ferroelectrics, and Frequency Control, IEEE Transactions on*, 46(2):277–290, March 1999. ISSN 0885-3010. doi: 10.1109/58.753016.

[6] Joseph C. Hassab and R. Boucher. Optimum estimation of time delay by a generalized correlator. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 27 (4):373–380, Aug 1979. ISSN 0096-3518. doi: 10.1109/TASSP.1979.1163269.

[7] Wen Ma and Jianguo Huang. Accurate time delay estimation based on SINC filtering. In *Signal Processing, 2002 6th International Conference on*, volume 2, pages 1621–1624 vol.2, Aug 2002. doi: 10.1109/ICOSP.2002.1180109.

[8] Y.T. Chan, J. Riley, and J. Plant. A parameter estimation approach to time-delay estimation and signal detection. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 28(1):8–16, Feb 1980. ISSN 0096-3518. doi: 10.1109/TASSP.1980.1163361.

[9] Fan Di, Cao Maoyong, and Sun Nongliang. Time delay estimation based on wiener filter in ultrasonic detection of sediments in drilling hole. In *Computer Science and Engineering, 2009. WCSE '09. Second International Workshop on*, volume 2, pages 582–585, Oct 2009. doi: 10.1109/WCSE.2009.880.

[10] C. Guetbi, D. Kouame, A. Ouahabi, and J.P. Chemla. Methods based on wavelets for time delay estimation of ultrasound signals. In *Electronics, Circuits and Systems, 1998 IEEE International Conference on*, volume 3, pages 113–116 vol.3, 1998. doi: 10.1109/ICECS.1998.813947.

[11] Shiyuan Zhou, Yao Xu, Hongbo Wang, and Chunguang Xu. Time delay estimation via third-order cumulant. In *Nondestructive Evaluation/Testing: New Technology Application (FENDT), 2013 Far East Forum on*, pages 77–81, June 2013. doi: 10.1109/FENDT.2013.6635533.

[12] Jiří Jan. *Číslicová filtrace, analýza a restaurace signálů*. Vysoké učení technické v Brně, VUTIUM, Brno, Czech Republic, 2002. ISBN 80-214-1558-4.

[13] Dana Černá. *Interpolace*. [online], April 2010. URL `kmd.fp.tul.cz/lide/cerna/MA3_FS/interpolace.pdf`. [cit. 2013-06-20].

[14] David Kahaner, Cleve Moler, and Stephen Nash. *Numerical Methods and Software*. Prentice Hall, Phoenix, New Jersey, 1988. ISBN 0-13-627258-4.

[15] Karel Dudáček, jr., Karel Dudáček, and Vlastimil Vavřička. Comparison of short delay measurement methods. In *International Conference on Applied Electronics (AE 2015)*, pages 27–32, Pilsen (Czech Republic), September 2015. Západočeská univerzita v Plzni. ISBN 978-80-261-0385-1.

[16] Karel Dudáček, jr., Karel Dudáček, and Vlastimil Vavřička. Short delay measurement using non-uniform Fourier transform. In *Proceedings of the 14th Biennial Baltic Electronics Conference*, pages 165–168, Tallinn (Estonia), October 2014. Tallinn University of Technology. ISBN 978-9949-23-672-5. Awarded with *Best Paper Award* at the conference.

[17] Jian-Jiun Ding. *Non-uniform Sampling*. [online]. URL `http://djj.ee.ntu.edu.tw/Nonuniform_Sampling.docx`. [cit. 2014-04-03].

[18] Zdeněk Kobeda and Jiří Reif. *Úvod do pravděpodobnosti a spolehlivosti*. University of West Bohemia, Pilsen, Czech Rep., 2nd edition, 2004. ISBN 80-7043-333-7.

[19] Jae-Jeong Hwang, Sang-Gyu Cho, Joon Moon, and Jae-Wan Lee. Nonuniform DFT based on nonequispaced sampling. *WSEAS Transactions on Information Science and Applications*, 2(9):1403–1408, 2005.

[20] Alok Dutt. *Fast Fourier Transform for Nonequispaced Data*. PhD thesis, Yale University, Connecticut, 1993.

[21] Alok Dutt and V. Rokhlin. *Fast Fourier Transform for Nonequispaced Data II*. Technical report, Yale University, Connecticut, 1993. Research Report 980.

[22] Daniel Potts. *NFFT*. [Online], January 2014. URL `http://www-user.tu-chemnitz.de/~potts/nfft/`. [cit. 2014-04-03].

[23] Ivars Bilinskis. *Digital Alias-free Signal Processing*. John Wiley & Sons, Ltd, Chichester, England, 2007. ISBN 978-0-470-02738-7.

[24] Karel Dudáček, jr. and Karel Dudáček. Trellis-based postprocessing for short delay measurement using NDFT. *Automatic Control and Computer Sciences*, 53(3):270–280, July 2019. ISSN 0146-4116. doi: 10.3103/S0146411619030039. URL `https://rdcu.be/bLwKA`.

[25] Yuri Artyukh, Ivars Bilinskis, Eugene Boole, Alexander Rybakov, and Vadim Vedin. Wideband rf signal digititising for high purity spectral analysis. In *Proceedings of the 2005 International Workshop on Spectral Methods and Multirate Signal Processing (SMMSP 2005)*, pages 123–128, Riga, Latvia, 06 2005. Tampere University of Technology. ISBN 952-15-1366-7.

[26] Karel Dudáček, jr. Non-uniform sampling using synchronised ADCs. In *XXVII International Conference on Information, Communication and Automation Technologies (ICAT 2019)*, pages ???–???, Sarajevo (Bosnia-Herzegovina), October 2019. IEEE. ISBN ??? Submitted for publication.

Author's publications

[1] Karel Dudáček, jr. and Karel Dudáček. Trellis-based postprocessing for short delay measurement using NDFT. *Automatic Control and Computer Sciences*, 53(3):270–280, July 2019. ISSN 0146-4116. doi: 10.3103/S0146411619030039. URL `https://rdcu.be/bLwKA`.

[2] Karel Dudáček, jr. Non-uniform sampling using synchronised ADCs. In *XXVII International Conference on Information, Communication and Automation Technologies (ICAT 2019)*, pages ???–???, Sarajevo (Bosnia-Herzegovina), October 2019. IEEE. ISBN ??? Submitted for publication.

[3] Karel Dudáček, jr., Karel Dudáček, and Vlastimil Vavřička. Short delay measurement using non-uniform Fourier transform. In *Proceedings of the 14th Biennial Baltic Electronics Conference*, pages 165–168, Tallinn (Estonia), October 2014. Tallinn University of Technology. ISBN 978-9949-23-672-5. Awarded with *Best Paper Award* at the conference.

[4] Karel Dudáček, jr., Karel Dudáček, and Vlastimil Vavřička. Comparison of short delay measurement methods. In *International Conference on Applied Electronics (AE 2015)*, pages 27–32, Pilsen (Czech Republic), September 2015. Západočeská univerzita v Plzni. ISBN 978-80-261-0385-1.

[5] Karel Dudáček, jr. Měření krátkých zpoždění s použitím neekvidistantní Fourierovy transformace. In *Počítačové architektury a diagnostika PAD 2014*, pages 151–156, Liberec (Czech Republic), September 2014. Technická univerzita v Liberci. ISBN 978-80-7494-026-2.

[6] Karel Dudáček, jr. Metody měření vzájemného posuvu neperiodických analogových signálů. In *Počítačové architektury a diagnostika PAD 2013*, pages 93–98, Pilsen (Czech Republic), September 2013. Západočeská univerzita v Plzni. ISBN 978-80-261-0270-0.

[7] Karel Dudáček, jr. Implementace mapovací tabulky pro paměti flash. In *Počítačové architektury a diagnostika 2012*, pages 49–54, Milovy (Czech republic), September 2012. ČVUT v Praze. ISBN 978-80-01-05106-1.

[8] Karel Dudáček. *Short Time Delay Measurement.* Technical report, University of West Bohemia, Pilsen, Czech Republic, 2015. DCSE/TR-2015-03.

Acknowledgement

At the first place, I thank my wife Táňa for patience with me, when I was spending evenings and weekends by research, writing articles, and finishing this thesis. I thank her also for encouraging me, when I wanted to give up the writing. I thank doc. Ing. Vlastimil Vavřička, CSc. (my supervisor), doc. Ing. Stanislav Racek, CSc., and my father Dr. Ing. Karel Dudáček for support, and for useful discussions regarding practical and formal aspects of my research.

Some years ago, when I was at the beginning of my research, we had a Christmas party. Small boy called Marigold was trying to make us mad by playing his flute. He was not successful, but his flute fifing brought me to the idea to use non-uniform sampling. I also thank him.

This work has been supported by UWB grant SGS–2013–029 Advanced Computer and Information Systems and by the European Regional Development Fund (ERDF) — project NTIS (New Technologies for the Information Society), European Centre of Excellence, CZ.1.05/ 1.1.00/ 0.2.0090.

Appendix

At the time of writing of this thesis, author's conference paper [26] was not published, yet; therefore, draft of the paper is attached in this appendix to allow reader to read it.

Non-uniform sampling using synchronised ADCs

Karel Dudáček jr.
University of West Bohemia
Pilsen, Czech Republic
Email: karlos@kiv.zcu.cz

***Abstract*—Some microcontrollers of the STM32 family are equipped with two or three synchronised ADCs, that could be utilized for non-uniform sampling. Conversion timing was studied, and non-uniform sampling function proposed in order to exploit the ADC as most as possible. It was found that attenuation of a sample&hold circuit limits an input bandwidth earlier than aliasing, nevertheless, it is still significantly wider bandwidth than without usage of non-uniform sampling.**

***Index Terms*—Nonuniform sampling, analog-digital conversion, microcontrollers, frequency response.**

I. Introduction

Many modern microcontrollers are equipped with one or more analog-to-digital converters (ADCs), usually intended for single-shot or periodic sampling. Nowadays, methods using non-uniform sampling appear, but their implementation in low-end devices is limited also by the lack of appropriate ADCs. Microcontrollers of STM32-F4xx family and STM32-F303 family are equipped with ADCs that seem to be usable for non-uniform sampling. In this article, we will focus on these ADCs and perform theoretical study regarding their usability for non-uniform sampling.

STM32 microcontrollers, including ADCs, are rather complex, in this article we will describe only functionalities important for non-uniform sampling. There are differences between ADCs of F303 family and F4xx family, in this article we will focus on F4xx family, and on STM32-F427 device in case of features differing across the particular family.

II. Theory of Non-uniform sampling

Let us have a band limited signal (bandwidth B). A well-known Nyquist's theorem says that, sampling frequency has to be at least $2 \cdot B$ to avoid aliasing. Using infinite sampling frequency, we can get infinite bandwidth. In real system, infinite frequency is unreachable, but we can approximate it. If we sample signal of infinite length composed of harmonic waveforms in random sample intervals, we would reach infinite bandwidth [1]. But this is also impossible due to two reasons — no available signal is of infinite length. The other reason is we can't measure time at infinite precision. However, it can be used as approximation, if we comply with several conditions. If the signal is of finite length and we take enough samples, it will be statistically similar to the infinite length signal. Finite time measurement resolution limits bandwidth and therefore introduces latent sampling frequency.

A. The latent sampling frequency

If the signal is sampled at random points in time, it is important to specify not only the sample number m, but also the time t_m when each one of the samples is taken. Time intervals between sampling time points can be specified as multiplies of a time quanta dt. As a result, time measurement has limited resolution dt. (Note that time quanta does not represent the time measurement accuracy but the time measurement resolution.) Sampling times t_m can be written as multiplies of this time quanta (1).

$$\begin{aligned} t_m &= n \cdot dt \\ &n \ldots \text{random integer } (n > 0) \\ &m \ldots \text{sample number} \end{aligned} \tag{1}$$

Having this equation, the non-uniform sampling can be regarded as uniform sampling at frequency dt^{-1} with some samples missing. If the sampling frequency is known, Nyquist's theorem can be applied to figure out the allowed signal bandwidth (2).

$$\begin{aligned} B &= \frac{1}{2 \cdot dt} \\ &B \ldots \text{allowed bandwidth} \end{aligned} \tag{2}$$

B. Sampling points

For sampling points distribution it is convenient to use (3). This distribution assures flat probability mass function when time goes to infinity.

$$\begin{aligned} &t_{n+1} = t_n + r_n \cdot dt \\ &r_n \sim Po(\frac{1}{\overline{f_s} \cdot dt}); \\ &r_n \ldots \text{random variable} \\ &\overline{f_s} \ldots \text{average sampling frequency} \end{aligned} \tag{3}$$

Suppose we have an AD converter with maximal sampling frequency f_{max}. For sampling (uniform or non-uniform) we must ensure $f_s \leq f_{max}$. Because sampling times are random, we can't ensure it for each sample, but we can set probability of violation to acceptable level p.

From cumulative distribution function (4) of Poisson distribution [2, pp. 32–38] we can find r_{min} [1]. At probability level $1 - p$ the value r_{min} is minimum of the set $\{r_n\}$; $n =$

[1] Values of $F(r)$ are tabulated for various values of λ, see [2, pp. 104].

$0, 1, \dots N-1$. Therefore the sampling period is greater then $r_{min} \cdot dt$ and the maximal frequency can be enumerated.

$$\begin{aligned} \frac{p}{N} &> \sum_{k=0}^{r} \frac{\lambda^k}{k!} e^{-\lambda} = F(r) \\ N &\dots \text{number of samples} \\ \lambda &= \frac{1}{f_s \cdot dt} \end{aligned} \tag{4}$$

The AD converter sampling frequency has to comply with (5).

$$f_{max} > \frac{1}{r_{min} \cdot dt} \tag{5}$$

For values of λ greater then 9 approximation (6) can be used [2, pp. 104].

$$Po(\lambda) \approx N(\mu = \lambda, \sigma^2 = \lambda) \tag{6}$$

III. Synchronized ADCs in STM32-F4xx

The STM32-F427 microcontroller is equipped with three ADCs, that could be used either independently or in dual/triple mode for synchronized operation. Each ADC is successive approximation analogue-to-digital converter with configurable resolution 6, 8, 10, or 12 bits. Conversion could be started either by software trigger or hardware trigger (from timers or external input). On the ADC input, there is analogue multiplexer with 16 external and 3 internal channels. Channels are organized to *regular group* and *injected group* with separate triggers. Request for conversion of an injected group during conversion of a regular group suspends the regular group conversion (similar behavior as an interrupt on processor). Detailed description of channel sequencing is out of scope of this article, for detailed description see reference manual [3] and datasheet [4].

In the dual/triple mode, two/three ADCs work together either for simultaneous conversion of two/three channels, or for overlapping conversions of a single channel (next ADC starts sampling before the previous one has finished conversion). Both/all three ADCs are controlled by the single control unit, therefore conversions are synchronized and could be controlled by a single trigger. *Interleaved mode* is intended for increasing sampling rate of uniform sampling by interleaved operation of two/three ADCs. *Alternate trigger mode* is similar, but ADCs are started by trigger, this mode seems to be well suited for non-uniform sampling. Several other modes exist, see [3] for details.

Direct memory access (DMA) could be used for reading data from ADC in single, dual, and triple operation modes.

Clock signal for ADCs is generated from peripheral clock divided by integer prescaler, maximal clock frequency of the ADC clock $f_{ADC} = 36MHz$ (for STM32-F427) [4, table 74]. The whole AD conversion is divided to *sampling phase* and *conversion phase*. Sampling phase has programmable length of 3, 15, 28, 56, 84, 112, 144, or 480 ADC clock cycles [3, chapter 11.5 and 11.13.4]. Conversion phase takes one ADC clock cycle per bit of resolution. In dual/triple mode, any channel could not be sampled concurrently by more ADCs, the next sampling of the channel could start at least two clock cycles after the end of the previous sampling[2]. Conversion phases could overlap. [3, chapter 11.9.3]

In the *triple ADC alternate trigger mode*, one trigger input is used for starting sampling of all ADCs (one ADC per event). The first trigger event starts the first ADC, the second trigger event starts the second ADC, the third trigger event starts the third ADC, the fourth trigger event starts the first ADC, and so on. See figure 1. If the *dual ADC alternate trigger mode* is used, sequence is similar to the triple ADC mode, but the third trigger event starts the first ADC, the fourth trigger event starts the second ADC, and so on. Alternate trigger mode can be only used for converting channels in the injected group. If the discontinuous conversions are enabled, one channel is converted, otherwise, all channels in the group are converted. Delay between trigger events must be at least one ADC clock cycle.

IV. Computation of ADC Limits

Maximal sampling rate of ADC is specified in datasheet [4, table 74]. At maximal ADC clock speed ($f_{ADC} = 36\ MHz$), sampling frequency is 2.4 MS/s with 12-bit resolution or 4 MS/s with 6-bit resolution. In interleaved mode, maximal sampling frequency is 4.5 MS/s for dual ADC and 7.2 MS/s for triple ADC.

According to the Nyquist's theorem, sampling rate limits an unaliased bandwidth of the sampled signal. If the periodic sampling is used, maximal signal frequency shall be at most half of this sampling rate in order to prevent aliasing. But sampling rate tells nothing about analogue features of the ADC, it could be able to sample even faster signals, but not fast enough to prevent aliasing, or it could be unable to accurately sample even signals that fulfills Nyquist's limit.

Bandwidth of the system depends on the whole signal path from the source to the end of processing, but for basic analysis of ADC features, it can be simplified. The signal source is approximated by an ideal voltage source, resistance, and capacitance of the signal source and signal path. ADC is approximated by capacitance and resistance of the sample&hold circuit, that could be found in datasheet ([4, table 74] for STM32-F427). For simplified circuit diagram see figure 2. For computations in this chapter, we will expect ideal signal source with zero resistance and capacitance, therefore, only parameters of ADC will be taken into account.

The sample&hold circuit is RC integration circuit, therefore, it is the first order low-pass filter. In order to minimize the influencing of the measurement results, changes of the filter's gain/attenuation must be lower then a half of the least significant bit of the converted value across the whole bandwidth.

The complex transfer function of the RC circuit is

[2] In reference manual [3], this rule is not mentioned for alternate trigger mode, but just for the other modes. It probably holds also for alternate trigger mode, because concurrent sampling of the single channel by two ADCs connect two sample&hold circuits in parallel to one channel, therefore, input capacity is doubled.

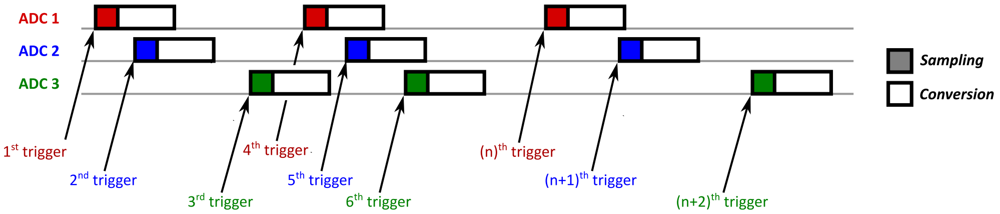

Fig. 1. Alternate trigger mode according to [3, figure 50].

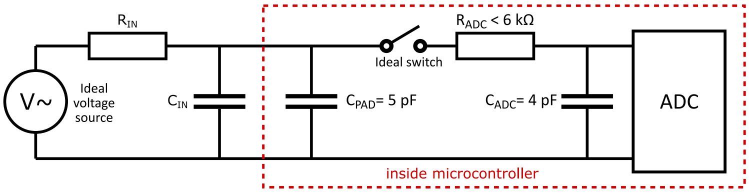

Fig. 2. Simplified circuit diagram.
R_{IN} is resistance of voltage source and signal path, C_{IN} is parasitic capacitance of the signal path, C_{PAD} is parasitic capacitance of a microcontroller pad, R_{ADC} is resistance of the sampling switch, C_{ADC} is capacitance of the sample&hold circuit.

$$F(j\omega) = \frac{1}{1 + j\omega RC} \tag{7}$$

therefore the voltage gain is

$$|F(\omega)| = \sqrt{\frac{1}{1 + (\omega RC)^2}} \tag{8}$$

For DC signal, the gain is $|F(0)| = 1$. For the maximal frequency, signal attenuation must be less than a half of the least significant bit $\frac{1}{d}$, where $d = 2^{N+1}$ and N is number of bits.

$$|F(\omega)| = 1 - \frac{1}{d} \tag{9}$$

This means, that maximal frequency will be

$$f_{max} = \frac{\sqrt{\left(\frac{d}{d-1}\right)^2 - 1}}{2\,\Pi \cdot RC} \tag{10}$$

According to the datasheet [4, table 74], these values are $C_{ADC} = 4\ pF$ (typical) / $7\ pF$ (maximal) and $1.5\ k\Omega < R_{ADC} < 6\ k\Omega$. The maximal value of R_{ADC} is given for the lowest operating voltage of MCU (1.7 V), minimal one for highest voltage (3.3 V) [4, table 74 note 4]. Maximal signal frequencies computed according to the equation (10) using datasheet values are summarized in the table I for various ADC resolutions[3].

TABLE I
MAXIMAL SIGNAL FREQUENCIES.

ADC resolution	Max. signal frequency	
N	@6 $k\Omega$	@1.5 $k\Omega$
12-bit	104 kHz	416 kHz
10-bit	207 kHz	828 kHz
8-bit	415 kHz	1.6 MHz
6-bit	834 kHz	3.3 MHz
5-bit	1.2 MHz	4.8 MHz
4-bit	1.7 MHz	6.8 MHz
3-bit	2.5 MHz	10 MHz

Values in this table are computed for typical value $C_{ADC} = 4\ pF$, and for both minimal and maximal value of $R_{ADC} = 1.5\ k\Omega/6\ k\Omega$.

V. Analysis of ADC Timing

The fastest signal sampling takes 3 periods of ADC clock signal [3, chapter 11.5], for STM32-F427 at maximal ADC clock frequency (36 MHz), it is 83 ns – it is one period of a 12 MHz signal. This means that a length of the sampling interval must be taken into account for computation of the trigger instants, otherwise, there would be very large aperture error. According to the datasheet [4, table 74], it takes three periods of ADC clock signal between trigger event and start of the sampling (valid for injected channels[4]). This means, that it have taken 6 periods of ADC clock signal since trigger event to start of the conversion. Because this delay is constant

[3] 5-bit, 4-bit, and 3-bit resolutions are not directly supported by ADC, it must be switched to 6-bit mode and results truncated.

[4] External trigger sources has additional latency due to clock resynchronisation.

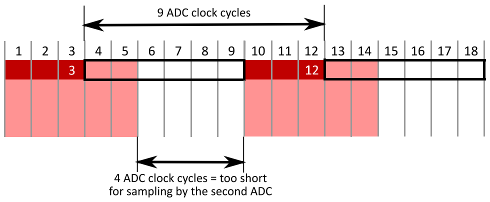

Fig. 3. Short delay between conversions prevents usage of the second ADC. *Two conversions at the first ADC are done as fast as possible, delay between two conversions of first ADC is 9 ADC clock cycles. After taking into account sampling and hold times, there is insufficient time for sampling phase of the second ADC.* ***Resulting sampling period is 9 ADC clock cycles.***

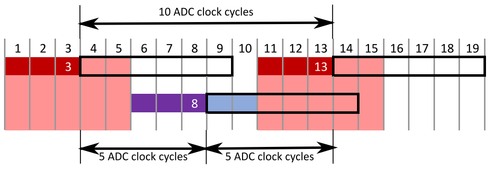

Fig. 4. Fastest sampling, but uniform.
Conversions are done as fast as possible, but respecting a sampling time of the second ADC. Delay between two conversions of first ADC is 10 ADC clock cycles, there is just sufficient time for sampling phase of the second ADC. ***Resulting sampling period is 5 ADC clock cycles.*** *This sampling is the fastest possible, but it is an uniform sampling.*

for all conversions, relative positions of samples (i.e. delays between two samples) are not affected, but it must be taken into account for absolute positions of samples (i.e. delay since time ”0” is trigger time + 6 ADC clock cycles).

As has been shown in the previous chapter, effective ADC resolution for high speed signals is limited to just few bits, therefore, it is sufficient to use 6-bit ADC resolution without any loss of information. Conversion with 6-bit resolution takes 6 ADC clock cycles. Shortest sampling phase takes 3 ADC clock cycles. Thus a whole conversion cycle takes 9 ADC clock cycles (3 cycles sampling and 6 cycles conversion). After sampling phase of one ADC, there must be at least two ADC clock cycles long *hold time* before the start of the sampling phase of another ADC. This means that shortest delay between two samples is 5 ADC clock cycles. Because two sampling periods (including hold times) take 10 ADC clock cycles and a sampling&conversion period of one ADC is 9 ADC clock cycles, a bottleneck limiting sampling frequency is a timing of the sampling phases. As the first ADC finishes conversion before the end of sampling phase (including hold time) of the second ADC, it is enough to use dual ADC mode. Usage of triple ADC mode brings no advantage in this case. See figures 3, 4, 5, and 6 for analysis of conversion timing, legend for these figures could be found in the figure 7.

As could be seen from the analysis, **a distance between two**

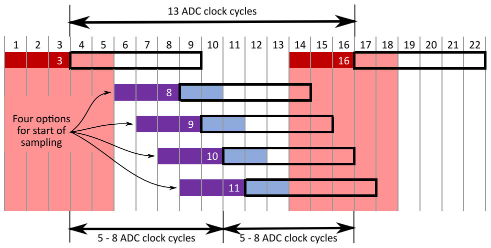

Fig. 5. Non-uniform sampling.
Conversions are done slower and with respect to a sampling time of the second ADC. Delay between two conversions of first ADC is 13 ADC clock cycles. There are four options of a trigger instant for the second ADC. ***Resulting sampling period could be varied in range of 5 – 8 ADC clock cycles.*** *Sampling could be performed as non-uniform sampling by choosing each time a random option from the four trigger instants.*

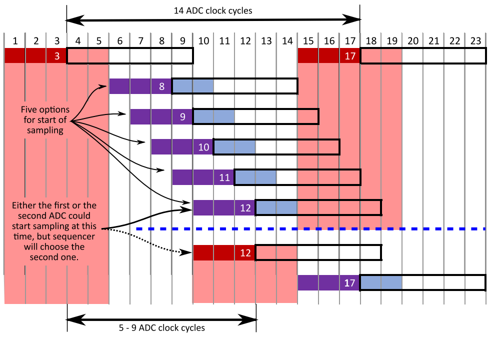

Fig. 6. Non-uniform sampling, the longest period.
Delay between two conversions of the first ADC is 14 ADC clock cycles. There are five options of a trigger instant for the second ADC. ***Resulting sampling period could be varied in range of 5 – 9 ADC clock cycles.*** *A sampling period of single ADC is 9 ADC clock cycles, therefore, the latest option is the same time instant as the earliest instant for triggering of the first ADC. It is also the longest period, that could not be obtained as a combination of two shorter ones.*

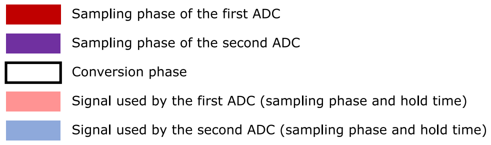
🟥 Sampling phase of the first ADC
🟪 Sampling phase of the second ADC
□ Conversion phase
🟥 Signal used by the first ADC (sampling phase and hold time)
🟦 Signal used by the second ADC (sampling phase and hold time)

Fig. 7. Legend for figures 3, 4, 5, and 6.

following samples could be 5, 6, 7, 8, or 9 ADC clock cycles. Intervals shorter than 5 clock cycles are not possible. Usage of intervals longer than 9 cycles is possible, nevertheless, it is not reasonable to do so since these intervals could be composed from the shorter ones, taking also intermediate samples.

VI. Non-uniform Sampling Function

For non-uniform sampling, equation (2) defines bandwidth of the signal, that could be processed without aliasing. This frequency depends on the length of the shortest measurable time quantum dt. In our case, this time quantum is one period of ADC clock $dt = {f_{ADC}}^{-1}$, therefore, maximal unaliased signal frequency is $f_{max} = 18\ MHz$ (for STM32-F427). This is a theoretical maximum for the non-uniform sampling, but analogue features of ADC also restrict the maximal frequency, see table I for these limitations.

It is reasonable to have average sampling period in range $(5 \ldots 9) \cdot dt$. (dt corresponds to ADC clock period ${f_{ADC}}^{-1}$). As described in the chapter II-B, a set of the sampling periods $\{r_n \cdot dt\}$ should have Poisson distribution $r_n \sim Po\left(\left(\overline{f_s} \cdot dt\right)^{-1}\right)$. But at the same time, sampling period must not be shorter than $5 \cdot dt$. This means that a set of random numbers with Poisson distribution can not be directly used, because values down to zero might occur. If the signal is short, it is also a bad idea to have excessively long interval between samples, because the amount of samples would be too low. Longer intervals could be composed from two shorter ones taking intermittent samples.

To fit our requirements, all numbers must not be less than 5 and as most numbers as possible should be in the range $[5 \ldots 9]$. Poisson distribution is characterized by parameter λ, mean value of the Poisson distribution is equal to this parameter $E\left(Po\left(\lambda\right)\right)\lambda$, therefore, it is reasonable to have this parameter "somewhere" in the range $[5 \ldots 9]$. Probability $P(5 \leq r_n \leq 9)$ that random number is in our range is summarized in table II for various values of parameter λ. As could be seen from the table, this probability is highest for $\lambda = 7$.

TABLE II
Best parameter for Poisson distribution.

λ	$P(r_n < 5)$	$P(5 \leq r_n \leq 9)$	$P(r_n > 9)$
5	0.4405	0.5277	0.0318
6	0.2851	0.6310	0.0839
6.5	0.2237	0.6537	0.1226
7	**0.1730**	**0.6575**	**0.1695**
7.5	0.1321	0.6443	0.2236
8	0.0996	0.6170	0.2834
9	0.0550	0.5324	0.4126

Having these requirements for sampling periods $\{r_n \cdot dt\}$, clipped Poisson distribution with probability function $g(x)$ is defined, see equation (11) and figure 8.[5]

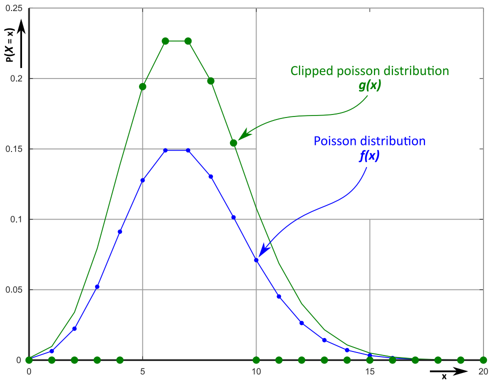

Fig. 8. Probability function of clipped Poisson distribution.

$$g(x) = \begin{cases} 0 & \text{for } x < 5 \\ \dfrac{f(x)}{F(9) - F(4)} & \text{for } 5 \leq x \leq 9 \\ 0 & \text{for } x > 9 \end{cases} \tag{11}$$

$$\begin{aligned} f(x) &= P(X = x) = \lambda^x \, \frac{\mathrm{e}^{-\lambda}}{x!} \\ F(x) &= P(X \leq x) = \sum_{k=0}^{x} \left(\lambda^k \, \frac{\mathrm{e}^{-\lambda}}{k!} \right) \\ X &\sim Po(\lambda) \\ \lambda &= 7 \\ x, k &\ldots \text{integers; } x \geq 0 \end{aligned}$$

As we are discarding some values, mean value $E(X)$ of the distribution will be affected. A new value could be computed according to the equation (12).[6]

$$E(X) = \sum_{k=5}^{9} \left(k \cdot g(k)\right) = \frac{1}{F(9) - F(4)} \cdot \sum_{k=5}^{9} \left(k \cdot f(k)\right) \tag{12}$$

$$\overline{f_s} = \left(E\left(X\right) \cdot dt\right) = \frac{f_{ADC}}{E(X)} \tag{13}$$

Using equation (12), we get mean value of our distribution $E(X) = 6.89$. Using equation (13), an average sampling frequency of ADC could be computed, it is 5.22 MHz for maximal ADC clock frequency 36 MHz.

Trigger events for ADC will be generated by some timer in microcontroller (it is equipped by several timers of different types) configured to count r_n clock periods. Sampling sequence $\{r_n\}$ could be either precomputed sequence, or it could be generated by on-chip random number generator and

[5] In the equation, discarding all values greater than 9 is assumed. Setting this threshold a bit higher is allowed and works similarly.

[6] Discarding of values greater than 9 and lower than 5 is assumed.

transformation function from uniform distribution to clipped Poisson distribution.

VII. Conclusion

Features of synchronized ADCs in STM32-F4xx microcontrollers were analyzed, and it was found, that they could be used for non-uniform sampling of signals with frequency too high for periodic sampling. Theoretical unaliased bandwidth of the signal, that could be non-uniformly sampled without aliasing, is 18 MHz. Unaliased bandwidth for uniform sampling is 3.6 MHz. However, properties of analogue part of an ADC restrict the maximal signal frequency and also a conversion resolution to lower value due to excessive attenuation at high frequencies. Due to this fact, it is possible to sample 10 MHz signal with 3-bit resolution or 4.8 MHz signal with 5-bit resolution.

Bandwidth of an uniformly sampled signal in microcontroller with synchronized ADCs is limited by an aliasing (i.e. insufficient sampling frequency). On the contrary, usage of non-uniform sampling widened the unaliased bandwidth behind the limits set by an analogue input stage of ADCs, therefore, full potential of the ADCs could be used.

Acknowledgment

I want to thank Adam Bařtipán and Petr Hodina for valuable remarks regarding architecture of STM32 processors.

References

[1] Jian-Jiun Ding. *Non-uniform Sampling*. [online]. Available: http://djj.ee.ntu.edu.tw/Nonuniform_Sampling.docx [2014, Apr. 3].

[2] Zdeněk Kobeda and Jiří Reif. *Úvod do pravděpodobnosti a spolehlivosti*. University of West Bohemia, Pilsen, Czech Rep., 2nd edition, 2004. ISBN 80-7043-333-7.

[3] *RM0090 Reference manual – STM32F40xxx, STM32F41xxx, STM32F42xxx, STM32F43xxx advanced ARM-based 32-bit MCUs*. ST Microelectronics, 2013. Doc ID 018909 Rev 4.

[4] *STM32F427xx, STM32F429xx datasheet*. ST Microelectronics, 2018. DocID024030 Rev 10.