Traineeship on Fiber-Optic Communications
Linear Propagation Effects in Optical Fibers

Author: Joris Steinberg
Location: ITBA (Instituto Tecnológico de Buenos Aires), Buenos Aires, Argentina
Training Coordinator (ARG): Dr. D. F. Grosz
Training Coordinator (NED): Dr. I. Tafur Monroy
Supervising Professor: Prof. ir. A. M. J. Koonen
Department: Eindhoven University of Technology, Faculty of Electrical Engineering, TTE-ECO
Time Period: April – June, 2004
Preface

This paper reports on the traineeship on Fiber-Optic Communications performed at the Instituto Tecnológico de Buenos Aires (ITBA) located in Buenos Aires, Argentina in the time period April-June 2004.

I enjoyed performing research on fiber optic communications systems, especially since it is a state-of-the art research area driven by the ever-increasing growth of voice and data traffic. Furthermore I greatly enjoyed working with Professor Diego Grosz, to whom my special thanks go out for his everlasting enthusiasm and great help regarding the traineeship. Furthermore I would like to thank Professor Pablo Fierens (ITBA) and Professor I. Tafur Monroy (TUE) who made it possible for me to conduct this research at ITBA.

Buenos Aires, July 7, 2004

Joris Steinberg
# Contents

**Preface** .................................................................................................................. 3

**Summary** .................................................................................................................. 5

**List of Symbols and Abbreviations** ........................................................................ 6

**1 Introduction** .......................................................................................................... 8

1.1 Group-Velocity Dispersion .................................................................................... 8

1.2 Material Dispersion ............................................................................................. 10

1.3 Waveguide Dispersion .......................................................................................... 10

1.4 Higher-Order Dispersion ...................................................................................... 12

1.5 Dispersion-Induced Limitations .......................................................................... 13

**2 Pulse Broadening** ................................................................................................. 16

2.1 Chirped Gaussian Pulses ...................................................................................... 16

2.2 Chirped Super-Gaussian Pulses ........................................................................... 19

2.3 Dispersion Lengths ............................................................................................. 26

**3 Pulse Sequences** .................................................................................................. 27

3.1 Dispersion of Gaussian Pulse Sequences (RZ) .................................................... 27

3.2 Dispersion of Super-Gaussian Pulse Sequences (NRZ) ...................................... 28

**4 Optical Noise & Filtering** .................................................................................... 31

4.1 Optical Noise ..................................................................................................... 31

4.2 Filtering .............................................................................................................. 32

**5 System Performance** ............................................................................................ 34

5.1 Introduction ....................................................................................................... 34

5.2 Eye-Patterns and BER ......................................................................................... 36

5.2.1 RZ-sequences ............................................................................................... 37

5.2.2 NRZ-sequences ............................................................................................. 38

5.3 System Performance Metrics ............................................................................. 40

5.4 1-dB OSNR Penalty Point .................................................................................... 42

5.5 Conclusions ....................................................................................................... 44

5.6 Recommendations for Further Investigation .................................................... 44

**Appendices** ............................................................................................................. 45

Appendix A: Analytical Broadening .......................................................................... 45

Appendix B: Calculated Broadening .......................................................................... 46

Appendix C: Super Gaussians .................................................................................... 49

Appendix D: Spectral Width Super Gaussian Pulse .................................................. 51

Appendix E: $\omega T$ versus $L_D$ and $d(\omega T)/dL_D$ versus $L_D$ for $m = 1, 2$ and $3$ ........................................................................................................... 52

Appendix F: 3D-plot time frequency Super-Gaussian ............................................. 57

Appendix G: 8 bit RZ-sequence plus Dispersion ..................................................... 60

Appendix H: 8 bit NRZ-sequence plus Dispersion ...................................................... 62

Appendix I: Routine-check on building-code for NRZ-sequences ............................. 65

Appendix J: Adding Noise (OSNR) ........................................................................... 71

Appendix K: Filtering of an RZ-sequence .................................................................. 73

Appendix L: Eye-pattern of an RZ-sequence ............................................................ 75

Appendix M: Eye-pattern of an NRZ-sequence ........................................................ 78

Appendix N: BER vs. OSNR for RZ-sequence ......................................................... 82

Appendix O: BER vs. OSNR for NRZ-sequence ....................................................... 85

Appendix P: Software ............................................................................................... 89

**References** ............................................................................................................... 90

**Suggestions for Further Reading** ........................................................................... 91
Summary

This report summarizes the research activities performed as part of the traineeship at ITBA, Buenos Aires, Argentina. It is divided as follows:

- Chapter One deals with the different types of dispersion effects that occur upon propagation in an optical fiber. Expressions for calculating the order of magnitude for the various dispersion types are presented. This theory is primarily based on the book by Govind. P. Agrawal, “Fiber-Optic Communication Systems”, second edition.

- Chapter Two discusses pulse broadening of chirped Gaussian and Super-Gaussian pulses due to dispersion. The definition for dispersion length of a Gaussian pulse $L_D = T_0^2 / |\beta_2|$ is shown to be not adequate for Super-Gaussian pulses. Thus, a new definition for the dispersion length of Super-Gaussian pulses is derived.

- Chapter Three deals with the building and propagation of Gaussian and Super-Gaussian pulse sequences and the effect of chromatic dispersion.

- Chapter Four introduces the concept of Optical Signal-to-Noise-Ratio (OSNR), photo-detection and signal-filtering both in the optical and electrical domains.

- Chapter Five introduces metrics for system performance by means of the Bit-Error-Rate (BER) and the OSNR. Operational margins and dispersion tolerances are calculated for a fiber type with its relevant system parameters.
# List of Symbols and Abbreviations

## Constants

### Physical Constants

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>SI Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>speed of light in vacuum</td>
<td>(3.0 \times 10^8 \text{ m/s})</td>
</tr>
</tbody>
</table>

## Variables

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>SI Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\beta)</td>
<td>GVD-parameter</td>
<td>[rad·sm(^{-1})]</td>
</tr>
<tr>
<td>(b)</td>
<td>normalized propagation constant</td>
<td>-</td>
</tr>
<tr>
<td>(B)</td>
<td>bit-rate</td>
<td>[bits·s(^{-1})]</td>
</tr>
<tr>
<td>(B_j)</td>
<td>oscillator strength</td>
<td>-</td>
</tr>
<tr>
<td>(B_e)</td>
<td>electrical bandwidth</td>
<td>[Hz]</td>
</tr>
<tr>
<td>(B_o)</td>
<td>optical bandwidth</td>
<td>[Hz]</td>
</tr>
<tr>
<td>(C)</td>
<td>linear frequency chirp</td>
<td>-</td>
</tr>
<tr>
<td>(\sigma)</td>
<td>rms width</td>
<td>[s]</td>
</tr>
<tr>
<td>(\sigma)</td>
<td>standard deviation</td>
<td>-</td>
</tr>
<tr>
<td>(D)</td>
<td>dispersion parameter</td>
<td>[ps·km(^{-1})·nm(^{-1})]</td>
</tr>
<tr>
<td>(E)</td>
<td>electric field</td>
<td>[V·m(^{-1})]</td>
</tr>
<tr>
<td>(\phi)</td>
<td>phase derivative</td>
<td>[rad·s(^{-1})]</td>
</tr>
<tr>
<td>(F)</td>
<td>field distribution</td>
<td>-</td>
</tr>
<tr>
<td>(\Delta f)</td>
<td>frequency window</td>
<td>[rad·s(^{-1})]</td>
</tr>
<tr>
<td>(I)</td>
<td>field intensity</td>
<td>-</td>
</tr>
<tr>
<td>(\lambda)</td>
<td>wavelength</td>
<td>[nm]</td>
</tr>
<tr>
<td>(L)</td>
<td>length</td>
<td>[m]</td>
</tr>
<tr>
<td>(m)</td>
<td>super-gaussian order</td>
<td>-</td>
</tr>
<tr>
<td>(n)</td>
<td>refractive index</td>
<td>-</td>
</tr>
<tr>
<td>(p(.))</td>
<td>probability of received bit</td>
<td>-</td>
</tr>
<tr>
<td>(P(.))</td>
<td>probability of deciding wrongly</td>
<td>-</td>
</tr>
<tr>
<td>(Q)</td>
<td>Q-factor</td>
<td>-</td>
</tr>
<tr>
<td>(S)</td>
<td>differential dispersion parameter</td>
<td>[ps·km(^{-1})·nm(^{-2})]</td>
</tr>
<tr>
<td>(t)</td>
<td>time</td>
<td>[s]</td>
</tr>
<tr>
<td>(T)</td>
<td>1/e-intensity half width</td>
<td>[s]</td>
</tr>
<tr>
<td>(v)</td>
<td>speed</td>
<td>[m·s(^{-1})]</td>
</tr>
<tr>
<td>(V)</td>
<td>normalized frequency</td>
<td>-</td>
</tr>
<tr>
<td>(\omega)</td>
<td>frequency</td>
<td>[rad·s(^{-1})]</td>
</tr>
<tr>
<td>(X)</td>
<td>x-factor</td>
<td>-</td>
</tr>
<tr>
<td>(\hat{x})</td>
<td>polarization vector</td>
<td>-</td>
</tr>
<tr>
<td>(z)</td>
<td>propagation distance</td>
<td>[km]</td>
</tr>
</tbody>
</table>
Subscripts

<table>
<thead>
<tr>
<th>Subscript</th>
<th>Refers to…</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>zero-bit</td>
</tr>
<tr>
<td>1</td>
<td>one-bit</td>
</tr>
<tr>
<td>D</td>
<td>dispersion</td>
</tr>
<tr>
<td>g</td>
<td>group</td>
</tr>
<tr>
<td>m</td>
<td>material</td>
</tr>
<tr>
<td>min</td>
<td>minimum</td>
</tr>
<tr>
<td>Th</td>
<td>threshold</td>
</tr>
<tr>
<td>w</td>
<td>waveguide</td>
</tr>
<tr>
<td>zd</td>
<td>zero dispersion</td>
</tr>
</tbody>
</table>

Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASE</td>
<td>Amplified Spontaneous Emission</td>
</tr>
<tr>
<td>BER</td>
<td>Bit Error Rate</td>
</tr>
<tr>
<td>DSF</td>
<td>Dispersion-Shifted Fiber</td>
</tr>
<tr>
<td>EDFA</td>
<td>Erbium Doped Fiber Amplifier</td>
</tr>
<tr>
<td>F</td>
<td>Fourier Transform</td>
</tr>
<tr>
<td>F^{-1}</td>
<td>Inverse Fourier Transform</td>
</tr>
<tr>
<td>FEC</td>
<td>Forward Error Correcting</td>
</tr>
<tr>
<td>GVD</td>
<td>Group Velocity Dispersion</td>
</tr>
<tr>
<td>ISI</td>
<td>Intersymbol Interference</td>
</tr>
<tr>
<td>N</td>
<td>Noise</td>
</tr>
<tr>
<td>NRZ</td>
<td>Non Return-to-Zero</td>
</tr>
<tr>
<td>NZ-DSF</td>
<td>Non Zero Dispersion-Shifted Fiber</td>
</tr>
<tr>
<td>OOK</td>
<td>On-Off Keying</td>
</tr>
<tr>
<td>OSNR</td>
<td>Optical Signal-to-Noise Ratio</td>
</tr>
<tr>
<td>RMS</td>
<td>Root Mean Square</td>
</tr>
<tr>
<td>RZ</td>
<td>Return-to-Zero</td>
</tr>
<tr>
<td>S</td>
<td>Signal</td>
</tr>
<tr>
<td>SSMF</td>
<td>Standard Single Mode Fiber</td>
</tr>
</tbody>
</table>
1 Introduction

Single-mode fibers suffer from group-velocity dispersion (GVD), or simply referred to as fiber dispersion, originating from the frequency dependence of the fiber’s core refractive index. In the presence of dispersion different spectral components of a transmitted pulse travel at different speeds, leading to pulse broadening and, as it will be shown, limiting system performance.

1.1 Group-Velocity Dispersion

Consider a single-mode fiber of length $L$. A specific spectral component at frequency $\omega$ will arrive at the output end of the fiber with a time delay $T = L / v_g$, where $v_g$ is the group velocity, defined as

$$v_g = (d\beta / d\omega)^{-1},$$

where $\beta$ is the propagation constant. By using $\beta = \tilde{n}_g = \tilde{n}_0 / c$ in Eq. 1.1.1, one can show that $v_g = c / \tilde{n}_g$, where $\tilde{n}_g$ is the group index given by

$$\tilde{n}_g = n + \omega (d\tilde{n} / d\omega).$$

The frequency dependence of the group velocity leads to pulse broadening simply because different spectral components of the pulse disperse during propagation and do not arrive simultaneously at the fiber output. If $\Delta\omega$ is the spectral width of the pulse, the extent of pulse broadening for a fiber of length $L$ is given by

$$\Delta T = \frac{dT}{d\omega} \Delta\omega = \frac{d}{d\omega}\left(\frac{L}{v_g}\right) \Delta\omega = L \frac{d^2\beta}{d\omega^2} \Delta\omega = L\beta_2 \Delta\omega,$$

where Eq. (1.1.1) was used. The parameter $\beta_2 = d^2\beta / d\omega^2$ is known as the GVD parameter. It determines how much an optical pulse will broaden upon propagation in an optical fiber.

In some cases, the frequency spread $\Delta\omega$ is determined by the range of wavelengths $\Delta\lambda$ emitted by the optical source. It is customary to use $\Delta\lambda$ in place of $\Delta\omega$. By using $\omega = 2\pi c / \lambda$ and $\Delta\omega = (-2\pi c / \lambda^2) \Delta\lambda$, Eq. (1.1.3) can be written as

$$\Delta T = \frac{d}{d\lambda}\left(\frac{L}{v_g}\right) \Delta\omega = DL\Delta\omega,$$

where
Traineeship on Fiber-Optic Communications

\[ D = \frac{d}{d\lambda} \left( \frac{1}{v_g} \right) = -\frac{2\pi c}{\lambda^2} \beta_2; \quad (1.1.5) \]

\( D \) is called the dispersion parameter and is expressed in units of ps/(km-nm).

The effect of dispersion on the bit rate \( B \) can be estimated by using the criterion \( B\Delta T < 1 \), i.e., by asking that the bit remains within the allocated time slot. By substituting \( \Delta T \) from Eq. (1.1.4) this condition becomes

\[ BL|D|\Delta\lambda < 1. \quad (1.1.6) \]

Eq. (1.1.6) provides an order-of-magnitude estimate of the \( BL \) product (i.e., the system capacity) offered by single-mode fibers.

For standard silica fibers, \( D \) is relatively small in the wavelength region near 1.3 \( \mu \)m. However, this parameter can vary considerably when the operating wavelength is shifted from this region. The wavelength dependence of \( D \) is governed by the frequency dependence of the mode index \( n \). From Eq. (1.1.5), \( D \) can be written as

\[ D = -\frac{2\pi c}{\lambda^2} \frac{d}{d\omega} \left( \frac{1}{v_g} \right) = -\frac{2\pi}{\lambda^2} \left( 2 \frac{d\tilde{n}}{d\omega} + \frac{d^2\tilde{n}}{d\omega^2} \right), \quad (1.1.7) \]

where Eq. (1.1.2) was used. The mode index is expressed as

\[ \tilde{n} = n_2 + b(n_1 - n_2) \approx n_2(1 + b\Delta), \quad (1.1.8) \]

where \( b \) is the normalized propagation constant and \( \Delta \) is the relative difference between core and cladding refractive indices. By introducing the normalized frequency \( V \) defined as

\[ V = k_0 a(n_1^2 - n_2^2)^{1/2} \approx (2\pi / \lambda)an_1\sqrt{2\Delta}, \quad (1.1.9) \]

where \( a \) is the fiber core radius, we can now express the total dispersion \( D \) as the sum of two terms

\[ D = D_m + D_w, \quad (1.1.10) \]

where the material dispersion \( D_m \) and the waveguide dispersion \( D_w \) are given by

\[ D_m = -\frac{2\pi}{\lambda^2} \frac{dn_2g}{d\lambda} = \frac{1}{c} \frac{dn_2g}{d\lambda} \quad \text{and} \quad (1.1.11) \]

\[ ^1 \text{The normalized frequency } V \text{ determines the condition for single-mode propagation. In particular the fiber becomes single mode when } V < 2.405. \text{ The } V \text{ parameter is also related to the number of modes that a multimode fiber will support. This number is approximately given by } 4V^2/\pi^2. \]
\[ D_w = -\frac{2\pi\Delta}{\lambda^2} \left[ \frac{n_{2g}^2 V_d^2 (V_b)}{n_2 \omega} \frac{dn_{2g}}{d\omega} + \left( \frac{d(V_b)}{d\omega} \right) \right]. \]  

Eq. (1.1.12)

Here \( n_{2g} \) is the group index of the cladding material. In deriving Eqs. (1.1.10)-(1.1.12) the parameter \( \Delta \) was assumed to be frequency independent. A third term known as differential material dispersion should be added to Eq. (1.1.10) when \( d\Delta / d\omega \neq 0 \). Its contribution is, however, negligible in practice.

### 1.2 Material Dispersion

Material dispersion occurs because the refractive index of silica, the material used for fiber fabrication, changes with the optical frequency \( \omega \). On a fundamental level, the origin of material dispersion is related to the characteristic resonance frequencies at which the material absorbs the electromagnetic radiation. Far from the medium resonances, the refractive index \( n(\omega) \) is well approximated by the Sellmeier equation

\[
n^2(\omega) = 1 + \sum_{j=1}^{M} \frac{B_j \omega_j^2}{\omega^2 - \omega_j^2},
\]

Eq. (1.2.1)

where \( \omega_j \) is the resonance frequency and \( B_j \) is the oscillator strength. Here \( n \) stands for \( n_1 \) or \( n_2 \), depending on whether the dispersive properties of the core or the cladding are considered. The sum of Eq. (1.2.1) extends over all material resonances that contribute in the frequency range of interest. In the case of optical fibers, the parameter \( B_j \) and \( \omega_j \) are obtained empirically by fitting the measured dispersion curves Eq. (1.2.1) with \( M = 3 \).

Material dispersion \( D_m \) is related to the slope of \( n_g \) by the relation \( D_m = c^{-1}(dn_g / d\lambda) \). It turns out that \( dn_g/d\lambda = 0 \) for pure silica at \( \lambda = 1.276 \mu m \). This wavelength is referred to as the zero-dispersion wavelength \( \lambda_{zd} \), since \( D_m = 0 \) at \( \lambda = \lambda_{zd} \). The dispersion parameter \( D_m \) is negative below \( \lambda_{zd} \) and becomes positive above that. In the wavelength range 1.25-1.66 \( \mu m \) it can be approximated by an empirical relation

\[
D_m \approx 122(1 - \frac{\lambda_{zd}}{\lambda}).
\]

Eq. (1.2.2)

It should be stressed that \( \lambda_{zd} = 1.276 \mu m \) only for pure silica. It can vary in the range 1.27-1.29 \( \mu m \) for optical fibers whose core and cladding are doped to vary the refractive index. The zero-dispersion wavelength of optical fibers also depends on the core radius \( a \) and the index step \( \Delta \) through the waveguide contribution to the total dispersion.

### 1.3 Waveguide Dispersion

The contribution to waveguide dispersion \( D_w \) to the dispersion parameter \( D \) is given by Eq. (1.1.12) and depends on the \( V \) parameter of the fiber. Both derivatives \( d(V_b)/dV \) and
$Vd^2(Vb)/dV^2$ are positive in the 0-1.6 μm. Therefore $D_w$ is negative in this entire wavelength range. $D_m$ is negative for wavelengths below $\lambda_{zd}$ and becomes positive above that. Figure 1.1 shows $D_m$, $D_w$ and their sum $D = D_m + D_w$, for a typical single-mode fiber. The main effect of waveguide dispersion is to shift $\lambda_{zd}$ by an amount of 30-40 nm so that the total dispersion is zero near 1.31 μm. It also reduces $D$ from its material value $D_m$ in the wavelength range 1.3-1.6 μm that is of interest for optical communication systems, since the minimum fibers loss is near 1.55 μm. High values of $D$ limit the performance of 1.55 μm lightwave systems.

![Figure 1-1: Total dispersion $D$ and relative contributions of material dispersion $D_M$ and waveguide dispersion $D_W$ for a conventional single-mode fiber.](image)

Since the waveguide contribution $D_w$ depends on fiber parameters such as the core radius $a$ and the refractive index difference $\Delta$, it is possible to design the fiber such that $\lambda_{zd}$ is shifted into the vicinity of 1.55 μm. Such fibers are called dispersion-shifted fibers. It is also possible to tailor the waveguide contribution such that the total dispersion $D$ is relatively small over a wide wavelength range extending from 1.3 to 1.6 μm. Such fibers are called dispersion-flattened fibers. Figure 1.2 shows typical examples of the wavelength dependence of $D$ for standard (conventional), dispersion-shifted, and dispersion-flattened fibers. The design of dispersion-modified fibers involves the use of multiple cladding layers and a tailoring of the refractive-index profile.
1.4 Higher-Order Dispersion

It appears from Eq. (1.1.6) that the $BL$ product of a single-mode fiber can be increased indefinitely by operating at the zero-dispersion wavelength $\lambda_{zd}$ where $D = 0$. Dispersive effects, however, do not disappear completely at $\lambda = \lambda_{zd}$. Optical pulses still experience broadening because of higher-order dispersive effects. This feature can be understood by noting that $D$ cannot be made zero at all wavelengths contained within the pulse spectrum centered at $\lambda_{zd}$. Clearly, the wavelength dependence of $D$ will play a role in pulse broadening. Higher-order dispersive effects are governed by the dispersion slope $S = dD/d\lambda$. The parameter $S$ is also called differential-dispersion parameter or second-order dispersion parameter. By using Eq. (1.1.5) it can be written as

$$S = (2\pi c / \lambda^3) \beta_3 + (4\pi c / \lambda^3) \beta_2,$$

(1.4.1)

where $\beta_3 = d\beta_2/d\omega = d^2\beta/d\omega^2$. At $\lambda = \lambda_{zd}$, $\beta_2 = 0$, and $S$ is proportional to $\beta_3$. For a source of spectral width $\Delta\lambda$, the effective value of dispersion parameter becomes $D = S\Delta\lambda$. The limiting bit rate-distance product can be estimated by using Eq. (1.1.6) with this value of $D$, or by using

$$BL|S|\langle\Delta\lambda\rangle^2 < 1.$$

(1.4.2)
1.5 Dispersion-Induced Limitations

Each frequency component of the optical field propagates in a single-mode fiber as

\[ \tilde{E}(r, \omega) = \hat{x} F(x, y) \tilde{B}(0, \omega) \exp(i\beta z), \]  

(1.5.1)

where \( \hat{x} \) is the polarization unit vector, \( \tilde{B}(0, \omega) \) is the initial amplitude, and \( \beta \) is the propagation constant. \( F(x, y) \) is the field distribution of the fundamental fiber mode that is often approximated by a Gaussian distribution. In general, \( F(x, y) \) also depends on \( \omega \), but this dependence can be ignored for pulses whose spectral width \( \Delta\omega \ll \omega_0 \), a condition generally satisfied in practice. Here \( \omega_0 \) is the frequency at which the pulse spectrum is centered; it is referred to as the center frequency or carrier frequency. Different spectral components propagate inside the fiber according to the simple relation

\[ \tilde{B}(z, \omega) = \tilde{B}(0, \omega) \exp(i\beta z). \]  

(1.5.2)

The amplitude in the time domain can be obtained by taking the inverse Fourier transform of Eq. (1.5.2) and is given by

\[ B(z, t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \tilde{B}(z, \omega) \exp(-i\omega t) d\omega. \]  

(1.5.3)

The initial spectrum amplitude \( \tilde{B}(0, \omega) \) is just the Fourier transform of the input amplitude \( B(0, t) \).

Pulse broadening results from the frequency dependence of \( \beta \). For quasi-monochromatic pulses with \( \Delta\omega \ll \omega_0 \), it is useful to expand \( \beta(\omega) \) in a Taylor series around the carrier frequency \( \omega_0 \) and retain terms up to the third order

\[ \beta(\omega) = \bar{n}(\omega) \frac{\omega}{c} \approx \beta_0 + \beta_1 (\Delta\omega) + \frac{1}{2} \beta_2 (\Delta\omega)^2 + \frac{1}{6} \beta_3 (\Delta\omega)^3, \]  

(1.5.4)

where \( \Delta\omega = \omega - \omega_0 \) and \( \beta_m = (d^m \beta / d\omega^m)_{\omega=\omega_0} \). Note that from Eq. (1.1.1) \( \beta_1 = 1/\nu_g \), where \( \nu_g \) is the group velocity. The GVD coefficient \( \beta_2 \) is related to the dispersion parameter \( D \) by Eq. (1.1.5), whereas \( \beta_3 \) is related to the dispersion slope \( S \) by Eq. (1.4.1). By substituting Eqs. (1.5.2) and (1.5.4) in Eq. (1.5.3) and introducing a slowly-varying amplitude \( A(z, t) \) of the pulse envelope by the relation

\[ B(z, t) = A(z, t) \exp[i(\beta_0 z - \omega_0 t)], \]  

(1.5.5)

the amplitude \( A(z, t) \) is found to be given by

\[ A(z, t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \tilde{A}(\omega) \tilde{B}(0, \omega) \times \]
Traineeship on Fiber-Optic Communications

\[
\exp \left[ i \beta_1 z \Delta \omega + \frac{i}{2} \beta_2 z (\Delta \omega)^2 + \frac{i}{6} \beta_3 z (\Delta \omega)^3 - i \Delta \omega t \right],
\]

(1.5.6)

where \( \tilde{A}(0, \Delta \omega) = G(0, \omega - \omega_0) \) is the Fourier transform of \( A(0,t) \).

By calculating \( \partial A / \partial z \) and noting that \( \Delta \omega \) is replaced by \( i(\partial A / \partial t) \) in the time domain, Eq. (1.5.6) can be written as

\[
\frac{\partial A}{\partial z} + \beta_1 \frac{\partial A}{\partial t} + \frac{i}{2} \beta_2 \frac{\partial^2 A}{\partial t^2} - \frac{1}{6} \beta_3 \frac{\partial^3 A}{\partial t^3} = 0.
\]

(1.5.7)

Eq. (1.5.7) is the basic propagation equation that governs pulse evolution inside a single-mode fiber. In the absence of dispersion \( (\beta_2 = \beta_3 = 0) \), the optical pulse propagates without change in its shape such that \( A(z,t) = A(0,t-\beta_1 z) \). By making the transformation to a reference frame moving with the pulse and introducing the new coordinates

\[
t' = t - \beta_1 z \quad \text{and} \quad z' = z,
\]

(1.5.8)

Eq. (1.5.7) can be written as

\[
\frac{\partial A}{\partial z'} + \frac{i}{2} \beta_2 \frac{\partial^2 A}{\partial t'^2} - \frac{1}{6} \beta_3 \frac{\partial^3 A}{\partial t'^3} = 0.
\]

(1.5.9)

For simplicity of notation, the prime over \( z' \) and \( t' \) is dropped resulting in the following equation

\[
\frac{\partial A}{\partial z} + \frac{i}{2} \beta_2 \frac{\partial^2 A}{\partial t^2} - \frac{1}{6} \beta_3 \frac{\partial^3 A}{\partial t^3} = 0.
\]

(1.5.10)

Performing a Fourier transform on Eq. (1.5.10) and noting that \( A \leftrightarrow \tilde{A} \) and \( \partial / \partial t \leftrightarrow i\omega \) results in

\[
\frac{\partial \tilde{A}}{\partial z} = \left( \frac{i\beta_2 \omega^2}{2} + \frac{i\beta_3 \omega^3}{6} \right) \tilde{A}.
\]

(1.5.11)

Solving the simple first order differential Eq. (1.5.11) results in

\[
\tilde{A}(z,\omega) = \tilde{A}(0,\omega) \exp \left( \frac{i}{2} \beta_2 \omega^2 z + \frac{i}{6} \beta_3 \omega^3 z \right).
\]

(1.5.12)

The amplitude in the time domain can be obtained by taking the inverse Fourier transform and is given by
\[ A(z,t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \tilde{A}(0, \omega) \exp \left( i \frac{\beta_2}{2} \omega^2 z + i \frac{\beta_3}{6} \omega^3 z - i\omega t \right) d\omega. \] (1.5.13)

Eq. (1.5.13) allows to calculate the distortion experienced by optical pulses as they propagate along the fiber.
2 Pulse Broadening

2.1 Chirped Gaussian Pulses

Gaussian pulses can be described by taking the initial amplitude as

\[ A(0,t) = A_0 \exp \left[ -\frac{1+iC}{2} \left( \frac{t}{T_0} \right)^2 \right], \quad (2.1.1) \]

where \( A_0 \) is the peak amplitude. The parameter \( T_0 \) represents the half-width at the 1/e intensity point. The parameter \( C \) governs the linear frequency chirp imposed on the pulse. A pulse is said to be chirped if its carrier frequency changes with time. The frequency change is related to the phase derivative and is given by

\[ \delta \omega(t) = -\frac{\partial \phi}{\partial t} = \frac{C}{T_0^2} t, \quad (2.1.2) \]

where \( \phi \) is the phase of \( A(0,t) \). The time-dependent frequency shift \( \delta \omega \) gives rise to frequency chirp. Its inclusion is important since semiconductor lasers that are driven directly generally emit pulses that are considerably chirped. The Fourier spectrum of a chirped pulse is broader than that of an unchirped pulse. This can be seen by taking the Fourier transform of Eq. (2.1.1), so that

\[ \tilde{A}(0,\omega) = A_0 \left( \frac{2\pi T_0^2}{1+iC} \right)^{1/2} \exp \left[ -\frac{\omega^2 T_0^2}{2(1+iC)} \right]. \quad (2.1.3) \]

The spectral half-width (at the 1/e-intensity point) is given by

\[ \Delta \omega_0 = (1 + C^2)^{1/2} T_0^{-1}. \quad (2.1.4) \]

In the absence of frequency chirp \((C = 0)\), the spectral width satisfies the relation \( \Delta \omega_0 T_0 = 1 \). Such a pulse has the narrowest spectrum and is called transform-limited. The spectral width is enhanced by a factor of \((1+C^2)^{1/2}\) in the presence of linear chirp, as seen in Eq. (2.1.4). Such pulses are said to carry excess bandwidth.

Solving Eq. (1.5.13) for the input field of Eq. (2.1.3), considering the case in which the carrier wavelength is far away from the zero dispersion wavelength so that the contribution of the \( \beta_3 \) term is negligible, results in

\[ A(z,t) = \left( \frac{A_0 T_0}{T_0^2 - i\beta_2 z(1+iC)} \right)^{1/2} \exp \left[ -\frac{(1+iC)^2}{2[T_0^2 - i\beta_2 z(1+iC)]} \right]. \quad (2.1.5) \]
Eq. (2.1.5) shows that a Gaussian pulse remains Gaussian on propagation. The pulse width changes with $z$ as

$$\frac{T_1}{T_0} = \left[ \left( 1 + \frac{C\beta_2 z}{T_0^2} \right)^2 + \left( \frac{\beta_2 z}{T_0^2} \right)^2 \right]^{1/2}, \quad (2.1.6)$$

where $T_1$ is the half-width at the $1/e$-intensity point. Figure 2.1 shows the broadening factor $T_1/T_0$ as a function of the propagation distance $z/L_D$, where $L_D = T_0^2/|\beta_2|$ is called the dispersion length. An unchirped pulse ($C = 0$) broadens as $[1 + (z/L_D)^2]^{1/2}$ and its width increases by a factor of $\sqrt{2}$ at $z = L_D$. (Script: Appendix A)

![Figure 2-1: Graphical representation of the Analytical broadening factor $T_1/T_0$ as function of the propagation distance $z/L_D$ for a chirped Gaussian input pulse.](image)

The chirped pulse, on the other hand, may broaden or compress depending whether $\beta_2$ and $C$ have the same or opposite signs. For $\beta_2C > 0$ the chirped Gaussian pulse broadens monotonically at a rate faster than the unchirped pulse. For $\beta_2C < 0$, the pulse width initially decreases and becomes minimum at a distance

$$z_{\text{min}} = \left[ \frac{|C|}{1 + C^2} \right] L_D. \quad (2.1.7)$$

The minimum value depends on the chirp parameter as

$$T_{1\text{min}} = T_0 \left/ (1 + C^2) \right.^{1/2}. \quad (2.1.8)$$

Eq. (2.1.6) can be generalized to include higher-order dispersion governed by $\beta_3$ in Eq. (1.5.13). A Gaussian input pulse does not remain Gaussian on propagation and develops a long tail with an oscillatory structure. Such pulses cannot be properly characterized by
their half-width at the $1/e$-intensity point. A proper measure of the pulse width is the RMS width of the pulse defined by

$$\sigma = \left[ \langle t^2 \rangle - \langle t \rangle^2 \right]^{1/2}, \quad (2.1.9)$$

where angle brackets denote the averaging with respect to the intensity profile, i.e.,

$$\langle t^n \rangle = \frac{\int_{-\infty}^{\infty} t^n |A(z,t)|^2 dt}{\int_{-\infty}^{\infty} |A(z,t)|^2 dt}. \quad (2.1.10)$$

The broadening factor, defined as $\sigma/\sigma_0$, where $\sigma_0$ is the RMS width of the input Gaussian pulse ($\sigma_0 = T_0/\sqrt{2}$) is given by

$$\frac{\sigma}{\sigma_0} = \left[ \left(1 + \frac{C\beta_3^2L}{2\sigma_0^2} \right)^2 + \left(\frac{\beta_3L}{2\sigma_0^2} \right)^2 \left(1 + C^2 \right) \frac{1}{2} \left(\frac{\beta_3L}{4\sigma_0^3} \right)^2 \right]^{1/2}, \quad (2.1.11)$$

where $L$ is the fiber length.

Figure 2.1 can also be obtained by simulating propagation in a fiber of a (chirped) Gaussian pulse. This can be done by defining a Gaussian pulse in Matlab and keep track of the RMS pulse-width while it propagates through the fiber. The following procedure is used (Script: Appendix B):

First, a Gaussian pulse as defined in Eq. (2.1.1) is defined. $A_0$ is set to one for simplicity reasons. Next, the Fourier Transform of the input field is calculated by means of a FFT algorithm. The effect of dispersion at a propagation length $z$ is calculated by multiplying the Fourier Transform by the following factor:

$$\exp\left[\frac{i}{2} \beta_3 \omega^2 z \right], \quad (2.1.12)$$

where the contribution of the $\beta_3$ term is neglected by assuming that the carrier wavelength lies far away from the fiber’s zero-dispersion wavelength. The resulting field is then transferred back to the time-domain and the RMS pulse-width is calculated.

The resulting pulse broadening for different chirp factors of a Gaussian pulse is shown in figure 2.2.
The simulated results shown in figure 2.2 completely match those obtained by analytical calculation. For $C = -2$ the numerical minimum of $\sigma_1/\sigma_0 \approx 0.447$ occurs at a distance of $z = 0.4L_D$. These values are equal to the values for the analytical minimum that can be obtained from Eqs. (2.1.7) and (2.1.8).

### 2.2 Chirped Super-Gaussian Pulses

The complete match between analytical and simulated results obtained in section 2.1 tells us that the routine for calculating the broadening factor is correct. Therefore the routine can be expanded to calculate pulse broadening for Super-Gaussian pulses. These pulses have the following form

$$A(0, t) = A_0 \exp \left[ -\frac{1 + iC}{2} \left( \frac{t}{T_0} \right)^{2m} \right]$$

(2.2.1)

and only differ from a normal Gaussian pulse in the factor $m$ in the exponent. An increasing $m$-order results in a “more square” pulse shape. Figure 2.3 shows a Super-Gaussian pulse for $m = 2, 3$ and 4. Figure 2.4 shows the accompanying spectra of the pulses in the frequency domain. As can be seen from the graphs the spectral width of the pulses increases with increasing $m$-order. This is verified by calculating the RMS spectral-width. This results in $\Delta \omega_{rms} = 5.05, 6.17$ and 7.11 rad/s for $m = 2, 3$ and 4 respectively.
Calculating the broadening factor as a function of dispersion length for $m = 2$ and 3 gives the following results shown in figure 2.5. (Script: Appendix B)
Figure 2-5: Simulated broadening factor $\sigma_1/\sigma_0$ as function of the propagation distance $z/L_D$ for a chirped Super Gaussian input pulse with $m = 2, 3$.

As can be seen from this figure for $C = -2$ the distance at which the minimum broadening factor occurs decreases with increasing $m$-order. Furthermore, the steepness of the graphs increases with increasing $m$-order. These effects can be explained by looking at the pulse shape propagated along the fiber. For increasing $m$-order the squareness of the pulse increases and therefore the spectrum of the pulse broadens. An increasing spectral width suffers more from dispersive effects due to the greater number of frequencies contained within the pulse. Therefore, minima occur at shorter propagation lengths and the minimum broadening factor increases with increasing $m$-order. Taking the minimum value for $m = 1$ calculated before and calculating the minimum values for $m = 2$ and $3$ results in the following table 2.1

<table>
<thead>
<tr>
<th>$m$</th>
<th>Minimum Broadening Factor $\sigma_1/\sigma_0$</th>
<th>Distance $z/L_D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.447</td>
<td>0.400</td>
</tr>
<tr>
<td>2</td>
<td>0.645</td>
<td>0.200</td>
</tr>
<tr>
<td>3</td>
<td>0.766</td>
<td>0.133</td>
</tr>
</tbody>
</table>

Table 2-1: Minimum broadening factor $\sigma_1/\sigma_0$ and corresponding distance $z/L_D$ for $m = 1, 2$ and $3$

According to these values a new equation for the minimum dispersion as defined in Eq. (2.1.7) can be derived. This new equation takes the $m$-order into account and yields
Traineeship on Fiber-Optic Communications

\[ z_{\text{min}} = \frac{|C|}{m(1 + C^2)} L_D. \] (2.2.2)

Let us take a closer look at what happens with the broadening factor for \( C = 0 \) (no chirp). A first rough approximation analyses is performed below. For the rise-time \( T_{\text{rise}} \) of a Super-Gaussian pulse holds [1].

\[ T_{\text{rise}} \approx \frac{T_0}{m}. \] (2.2.3)

The spectral width of a single pulse can be estimated by

\[ \Delta \omega_{\text{rms, super, gaussian}} \propto \frac{1}{T_{\text{rise}}}. \] (2.2.4)

From Eqs. (2.2.3) and (2.2.4) we have

\[ \Delta \omega_{\text{rms, super, gaussian}} \propto \frac{m}{T_0}. \] (2.2.5)

Since the product \( \Delta \omega_{\text{rms, gaussian}} * T_0 \) is a constant, we obtain a relation between the spectral width of a Gaussian and a Super-Gaussian pulse given by

\[ \Delta \omega_{\text{rms, super, gaussian}} \propto m * \Delta \omega_{\text{rms, gaussian}}. \] (2.2.6)

How does this affect the expression for the dispersion length \( L_D = T_0^2/|\beta_2| \) as stated before? Put in another form this expression becomes

\[ L_{D, \text{gaussian}} = \frac{T_0^2}{|\beta_2|} = \frac{1}{2\Delta \omega_{\text{rms, gaussian}}^2 |\beta_2|}. \] (2.2.7)

Combined with the expression found in Eq. (2.2.6) this results in a new dispersion length equation that takes the \( m \)-order into account

\[ L_{D, \text{super, gaussian}} = \frac{1}{2m^2 \Delta \omega_{\text{rms, gaussian}}^2 |\beta_2|} = \frac{T_0^2}{m^2 |\beta_2|} = \frac{L_{D, \text{gaussian}}}{m^2}. \] (2.2.8)

This result is valid for dispersion lengths shorter than \( L_{D, \text{gaussian}} \) of a Super-Gaussian pulse. However, for increasing dispersion lengths this \( m^2 \) turns into \( m \). This is true because upon propagation a Super-Gaussian pulse will become Gaussian. (Proof of this fact will be given later on.) Analytically this can be shown as follows; for dispersion
lengths longer than $10L_{D,\text{gaussian}}$ of a Super-Gaussian pulse with a certain $m$-order the following equation holds

$$\Delta \omega_{\text{rms, super gaussian}} \propto \frac{\sqrt{m}}{T_0}.$$  \hfill (2.2.9)

Moreover, the product $\Delta \omega_{\text{rms, gaussian}} \star T_0$ is a constant and therefore

$$\Delta \omega_{\text{rms, super gaussian}} \propto \sqrt{m} \star \Delta \omega_{\text{rms, gaussian}}.$$  \hfill (2.2.10)

(Eq. (2.2.10) can be verified by performing some calculations. These are shown in Appendix D.) Just like it was done before, this results in a new definition for the dispersion length of a Super-Gaussian pulse of a given $m$-order:

$$L_{D,\text{super gaussian}} = \frac{1}{2m\Delta \omega^2_{\text{rms, gaussian}} \left| \beta_2 \right|} = \frac{T_0^2}{m\left| \beta_2 \right|} = \frac{L_{D,\text{gaussian}}}{m}.$$  \hfill (2.2.11)

To summarize, we have the following definitions

$$L_{D,\text{super gaussian}} = \frac{L_{D,\text{gaussian}}}{m^2} \text{ for } z < L_{D,\text{gaussian}},$$

$$L_{D,\text{super gaussian}} = \frac{L_{D,\text{gaussian}}}{m} \text{ for } z > 10L_{D,\text{gaussian}}.$$  \hfill (2.2.12)

From figures 2.2 and 2.5 it can be noted that the pulse first starts broadening in a non-linear way but after a certain distance broadens in linear fashion. The left graph in figure 2.6 shows the product of the RMS spectral-width and RMS time-width plotted against the normalized propagation length for a single pulse and for different $m$-orders. The right graph shows the derivative of the broadening factor with respect to this normalized propagation length. (Script: Appendix E) At short propagation lengths this derivative increases but settles to a stable value after approximately three dispersion lengths. These values are 0.5, 1.0 and 1.5 for $m = 1, 2$ and 3 respectively. It can be noted that this behavior is therefore proportional to the $m$-order.
Figure 2-6: left: frequency-time RMS product versus normalized distance $z/L_D$, right: $d(\omega T)/d(z/L_D)$ versus distance $z/L_D$.

How much faster does a Super-Gaussian pulse disperse as compared to a Gaussian pulse according to these results? To visualize this, the following formula is solved for $X$

$$m^X = \frac{(d(\omega T)/dL_D)_m}{(d(\omega T)/dL_D)_{m=1}}.$$  \hspace{2cm} (2.2.13)

The results are shown in the following table 2.2

<table>
<thead>
<tr>
<th>$z/L_D$</th>
<th>$X$</th>
<th>$z/L_D$</th>
<th>$X$</th>
<th>$z/L_D$</th>
<th>$X$</th>
<th>$z/L_D$</th>
<th>$X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>1.81</td>
<td>1.2</td>
<td>1.24</td>
<td>2.4</td>
<td>1.09</td>
<td>0.0</td>
<td>1.72</td>
</tr>
<tr>
<td>0.1</td>
<td>1.78</td>
<td>1.3</td>
<td>1.21</td>
<td>2.5</td>
<td>1.08</td>
<td>0.1</td>
<td>1.68</td>
</tr>
<tr>
<td>0.2</td>
<td>1.73</td>
<td>1.4</td>
<td>1.19</td>
<td>3.0</td>
<td>1.06</td>
<td>0.2</td>
<td>1.63</td>
</tr>
<tr>
<td>0.3</td>
<td>1.67</td>
<td>1.5</td>
<td>1.18</td>
<td>3.5</td>
<td>1.05</td>
<td>0.3</td>
<td>1.57</td>
</tr>
<tr>
<td>0.4</td>
<td>1.60</td>
<td>1.6</td>
<td>1.16</td>
<td>4.0</td>
<td>1.05</td>
<td>0.4</td>
<td>1.50</td>
</tr>
<tr>
<td>0.5</td>
<td>1.54</td>
<td>1.7</td>
<td>1.15</td>
<td>4.5</td>
<td>1.04</td>
<td>0.5</td>
<td>1.44</td>
</tr>
<tr>
<td>0.6</td>
<td>1.48</td>
<td>1.8</td>
<td>1.14</td>
<td>5.0</td>
<td>1.04</td>
<td>0.6</td>
<td>1.38</td>
</tr>
<tr>
<td>0.7</td>
<td>1.42</td>
<td>1.9</td>
<td>1.13</td>
<td>5.5</td>
<td>1.03</td>
<td>0.7</td>
<td>1.33</td>
</tr>
<tr>
<td>0.8</td>
<td>1.37</td>
<td>2.0</td>
<td>1.12</td>
<td>6.0</td>
<td>1.03</td>
<td>0.8</td>
<td>1.29</td>
</tr>
<tr>
<td>0.9</td>
<td>1.33</td>
<td>2.1</td>
<td>1.11</td>
<td>6.5</td>
<td>1.03</td>
<td>0.9</td>
<td>1.26</td>
</tr>
<tr>
<td>1.0</td>
<td>1.30</td>
<td>2.2</td>
<td>1.10</td>
<td>7.0</td>
<td>1.03</td>
<td>1.0</td>
<td>1.23</td>
</tr>
<tr>
<td>1.1</td>
<td>1.26</td>
<td>2.3</td>
<td>1.10</td>
<td>7.5</td>
<td>1.03</td>
<td>1.1</td>
<td>1.20</td>
</tr>
</tbody>
</table>

Table 2-2: $X$-factor as mentioned in Eq. (2.2.7) for $m = 2$ and 3.
At short propagation lengths $X \sim 2$. It then decreases to become approximately 1 at a normalized propagation length of 3 and bigger. Taking this into account an equation for the broadening factor of a single Super-Gaussian pulse can be obtained. To do that, first Eq. (2.1.6), $L_D = T_0^2/|\beta_2|$ and $C = 0$ are combined to form

$$\frac{T_1}{T_0} = 1 + \left( \frac{z}{L_D} \right)^{2^{1/2}}.$$  \hspace{1cm} (2.2.14)

The results of table 2.2 turn Eq. (2.2.14) into

$$\frac{T_{L_{\text{super gaussian}}}}{T_{0_{\text{super gaussian}}}} \approx m^X \left[ 1 + \left( \frac{z}{L_D} \right)^{2^{1/2}} \right], \text{ for } z < 3L_{D_{\text{gaussian}}} \text{ with } 1 < X < 2,$$

$$\frac{T_{L_{\text{super gaussian}}}}{T_{0_{\text{super gaussian}}}} \approx m \left[ 1 + \left( \frac{z}{L_D} \right)^{2^{1/2}} \right], \text{ for } z > 3L_{D_{\text{gaussian}}}.$$  \hspace{1cm} (2.2.15)

Taking the $m$-order within the brackets results in

$$\frac{T_{L_{\text{super gaussian}}}}{T_{0_{\text{super gaussian}}}} \approx \left[ m^2^X + \left( \frac{z}{L_D / m^X} \right)^{2^{1/2}} \right], \text{ for } z < 3L_{D_{\text{gaussian}}}, \text{ where } 1 < X < 2,$$

$$\frac{T_{L_{\text{super gaussian}}}}{T_{0_{\text{super gaussian}}}} \approx \left[ m^2 + \left( \frac{z}{L_D / m} \right)^{2^{1/2}} \right], \text{ for } z > 3L_{D_{\text{gaussian}}}.$$  \hspace{1cm} (2.2.16)

This results in a new definition of the dispersion length of Super-Gaussian pulses:

$$L_{D_{\text{super gaussian}}} = \frac{L_{D_{\text{gaussian}}}}{m^X} \text{ for } z < 3L_{D_{\text{gaussian}}}, \text{ where } 1 < X < 2,$$

$$L_{D_{\text{super gaussian}}} = \frac{L_{D_{\text{gaussian}}}}{m} \text{ for } z > 3L_{D_{\text{gaussian}}}.$$  \hspace{1cm} (2.2.17)

These results match with the results obtained in Eq. (2.2.12). Looking at the pulse shapes after propagation over many dispersion lengths explains these results as well. The left graph in figure 2.7 shows a single Super-Gaussian pulse with $m = 2$ after propagation over long lengths (Script: Appendix F). As seen in this figure, dispersion causes a Super-Gaussian pulse to become Gaussian after some propagation. The pulse (initially Super-
Gaussian) will then broaden like a Gaussian pulse. Eq. (2.2.17) for $z > 3L_D$ will then hold. The $m$-order in this equation represents the extra dispersive effects undergone by the pulse at short propagation lengths. The right graph of figure 2.7 shows the spectrum of the pulse for different propagation lengths. As expected, the frequency spectrum does not change due to dispersion.

![Figure 2-7: left: Pulse shape of a propagated Super-Gaussian pulse with $m = 2$. right: Corresponding spectra](image)

### 2.3 Dispersion Lengths

To put things in perspective, actual propagation lengths in kilometers are calculated and listed in table 2.3. A Gaussian pulse with a $T_0$ of 25 ps (a duty cycle of 50 % at a bit rate of 10 Gbit/s) is considered. SSMF, DSF and NZ-DSF stand for Standard Single-Mode Fiber, Dispersion-Shifted Fiber and Non-Zero Dispersion-Shifted Fiber respectively.

| Fiber type | $|\beta_2|$ [ps$^2$/km] | Distance for a single dispersion length [km] | Distance for three dispersion lengths [km] |
|------------|-------------------|------------------------------------------|-----------------------------------------|
| SSMF       | 20                | 31.2                                     | 93.6                                    |
| DSF        | 2                 | 312                                      | 936                                     |
| NZ-DSF     | 2 to 8            | 78.1 to 312                              | 234 to 936                              |

**Table 2-3:** Actual propagation lengths in kilometers for different fibers. Duty cycle = 50% and $B = 10$ GHz.
3 Pulse Sequences

High-capacity fiber-optic communication systems, in their vast majority, transmit information by amplitude modulation. In the digital domain this is characterized by two states, or bits, “1” and “0”. The presence of a light pulse within the allocated bit-slot represents a “1” whereas the absence of such pulse represents a “0”. This modulation is referred to as On-Off Keying (OOK).

Within OOK transmission two different modulation formats are usually adopted. First, there is the modulation format where the transmitted “1” is characterized by a Gaussian pulse. At the end of the bit-slot the pulse intensity returns to zero and so the modulation format is called Return-to-Zero (RZ). Such RZ pulses are characterized by their duty cycle, defined as the 1/e-intensity full-width time divided by the bit-slot.

There is another modulation format where the transmitted “1” is characterized by a Super-Gaussian pulse (“square” pulse). In this case, the pulse intensity does not return to zero within the bit-slot. Whenever there are two consecutives “1’s”, the signal will not return to zero in between these two bits, but remain in a “high” state. This modulation format is called Non Return-to-Zero (NRZ).

3.1 Dispersion of Gaussian Pulse Sequences (RZ)

Figure 3.1 shows an 8-bit RZ sequence and its corresponding spectrum. (Script: Appendix G). The routine for creating this sequence simply consists of attaching bit-slots to form a total sequence. Due to the fact that the pulse returns to zero within the bit-slot modulation side-bands will arise at the bit-rate frequency.

![Figure 3-1: Top: 8 bit sequence of a RZ-system with \( B = 10 \) Gbit/s and duty cycle = 30\% Bottom: corresponding spectrum.](image-url)
Figure 3.2 shows four plots of the same sequence of bits propagated over several dispersion lengths (Script: Appendix G). Since dispersion will only cause a phase-shift to the Fourier transform of the input field, the spectrum will not change due to dispersion. This is also apparent in the right plot of figure 2.7.

As can be seen from this figure the pulses lose peak power due to dispersion-induced broadening. At a certain propagation distance adjacent pulses will start to interfere. This phenomenon is known as Inter-Symbol-Interference (ISI). In the presence of too much ISI a receiver may not be able to distinguish the “1’s” from the “0’s” anymore thus leading to detection errors.

3.2 Dispersion of Super-Gaussian Pulse Sequences (NRZ)

Figure 3.3 shows an 8-bit NRZ sequence and its corresponding spectrum. (Script: Appendix H). It can be noted (comparing the spectra of figures 3.1 and 3.3) that the modulation side-bands at the bit-rate frequency are lost for the NRZ-sequence. This is because the “1” bits will not return to zero within the bit-slot. The routine for creating this sequence works slightly different from the routine for creating an RZ-sequence. First the sequence is built in a similar way as for RZ. This will create a sequence where the field will become zero in between two adjacent “1’s”. Therefore these “dips” will be filtered out and replaced with a “high” value. The result is shown in figure 3.3. The bit-slots from
3 to 6 ps contain three “1’s” and in between two consecutive “1’s” the signal does not return to zero.

**Figure 3-3:** Top: 8-bit NRZ sequence with $B = 10$ Gbit/s, duty cycle = 95% and $m = 3$. Bottom: corresponding spectrum.

Figure 3.4 shows four plots of the same sequence of bits propagated over several kilometers. (Script: Appendix H). The dispersion length increases, since the duty cycle is higher than that of the RZ sequence. Therefore the pulse is only propagated for $z$-values within one dispersion length. The pulse retains its shape even after 67.7 km whereas the RZ pulse sequence already degrades at a propagated distance of 33.75 km. The explanation for this is as follows. The fact that the NRZ-system contains more square pulses implies a broader spectrum. However, the fact that the pulses for NRZ are much broader reduces the spectral width. Overall, the NRZ spectrum is narrower than the RZ spectrum. This causes NRZ to be less sensitive to dispersion than RZ (this is because dispersion goes with the square of the spectrum, $\omega^2$ in Eq. (2.1.12)). To make sure that the algorithm for creating an NRZ-sequence is correct and dispersion results can be trusted a check is performed and described in Appendix I.
Figure 3-4: NRZ bit sequence propagated over $z = 0$, $1/5L_D$, $2/5L_D$, and $3/5L_D$. ($|\beta_2| = 20 \text{ ps}^2/\text{km}$)
4 Optical Noise & Filtering

4.1 Optical Noise

As a signal propagates along a fiber link it needs to be amplified a certain number of times in order to keep an adequate power level. This can be done by means of in-line Erbium Doped Fiber Amplifiers (EDFA’s). A schematic drawing is shown in figure 4.1.

![Figure 4-1: Schematic drawing of an optical amplified fiber link with transmitter and receiver Tx and Rx respectively.](image-url)

What an EDFA practically does is resetting the signal power to its original level, i.e. the launch input power provided by the booster amplifier at the transmitter side. Amplified Spontaneous Emission (ASE) broadband noise is added every time the signal is amplified. The ASE noise is characterized by its spectral density. As it is customary, the optical noise power is calculated in a frequency window $\Delta f = 12.5$ GHz. [2]. At a transmission wavelength of $\lambda = 1.55 \mu m$ this corresponds to a wavelength window $\Delta \lambda \sim 0.1$ nm. The average optical noise level can be specified by defining the Optical Signal-to-Noise Ratio (OSNR). The noise distribution is assumed to be Gaussian. Figure 4.2 shows an RZ sequence for different levels of OSNR. (Script: Appendix J)

![Figure 4-2: RZ sequence for OSNR = 40, 35, 30 and 25 dB](image-url)
Note that, since the signal is polarized and the optical noise is not, on average half of the noise power will beat with the signal. This leads to an effective noise level that is 3 dB lower ( = factor \( \frac{1}{2} \)) than the specified value. This factor has already been accounted for in figure 4.2 and in all simulations that follow.
In this report only linear propagation effects are considered therefore it does not matter if the optical noise is added to the signal before or after propagation. We chose to set the OSNR level after propagation.

4.2 Filtering

The fiber link depicted in figure 4.1 represents a single-wavelength system. In Wavelength Division Multiplexed (WDM) systems many different wavelengths, or channels, are transmitted through the same optical fiber. A schematic of a WDM system is shown in figure 4.3, where we now have multiplexers and demultiplexers at the end terminals.

![Figure 4-3: Schematic drawing of a Multiplexer and De-Multiplexer](image)

The de-multiplexer selects a single channel by means of band pass filtering. In this report the de-multiplexer is modelled by a Butterworth filter with a 3-dB bandwidth of 25 GHz, a typical number found in 10 Gbit/s systems.
The influence of such optical filtering can be appreciated from the following discussion. If the signal plus noise is represented by \((S + N)\), then the intensity of the detected field is proportional to:

\[
|S + N|^2 = S^2 + 2NS + N^2. \tag{4.2.1}
\]

The cross-product term gives rise to Signal-ASE beating, whereas the \(N\)-square term gives rise to ASE-ASE beating\(^2\). Being broadband the ASE-ASE beating is strongly suppressed by the de-multiplexer.
There is another filtering process involved in photo-detection. This filtering has its origin in the inherently slow electronics found at the receiver. We model such an electrical filter by means of a Butterworth filter with a typical 3-dB bandwidth of 7 GHz for an RZ sequence and 8 GHz for an NRZ sequence [3]. Figure 4.4 shows a filtered and a non-

\(^2\) When using an EDFA as an optical preamplifier in front of the receiver, the Signal-ASE beating will be by far the dominant noise contribution as compared to both shot and thermal noise. For this reason these last two noise terms are neglected in this report.
filtered RZ-sequence with 30-dB OSNR. Note that the filtered signal now represents a voltage or a current and not the optical power anymore. Figure 4.5 shows the corresponding spectra before and after filtering. (Script: Appendix K) Observe that due to the electrical filter, the modulation side-bands decrease.

![Figure 4-4: RZ sequence with OSNR = 30 dB before and after filtering.](image)

![Figure 4-5: Frequency spectrum of the non-filtered and filtered RZ sequence.](image)
5 System Performance

5.1 Introduction

The OSNR level determines how much the transmitted signal is degraded. A measure to express this degradation after transmission is to calculate the Bit-Error-Rate (BER). Let $I$ represent the (fluctuating) signal received by the decision circuit at the receiver. The sampled value $I$ fluctuates from bit to bit around an average value $I_1$ of $I_0$ depending on whether the bit corresponds to 1 or 0 in the bit-sequence. The decision circuit compares the sampled value with a threshold value $I_{Th}$ and “decides” “1” if $I > I_{Th}$ or “decides” “0” if $I < I_{Th}$. An error occurs if $I < I_{Th}$ when a bit “1” is transmitted. An error also occurs if $I > I_{Th}$ when a bit “0” is transmitted. In our case detection errors are caused by OSNR degradation and dispersion-induced pulse distortion. Both sources of errors can be included in an equation for the BER

$$BER = p(1)P(0/1) + p(0)P(1/0),$$  \hspace{2cm} (5.1.1)

where $p(1)$ and $p(0)$ are the probabilities of receiving bits “1” and “0” respectively, $P(0/1)$ is the probability of deciding “0” when “1” is received, and $P(1/0)$ is the probability of deciding “1” when “0” is received. Since “1” and “0” bits are equally likely to occur, $p(1) = p(0) = \frac{1}{2}$, and the BER becomes

$$BER = 1/2[P(0/1) + P(1/0)].$$  \hspace{2cm} (5.1.2)

Figure 5.1 shows how $P(0/1)$ and $P(1/0)$ depend on the probability density function $p(I)$ of the sampled value $I$. This distribution is considered to be Gaussian since the added is Gaussian as well.

![Figure 5-1: Gaussian probability densities of “1” and “0” bits. The dashed region shows the probability of incorrect decision.](image)
The two dashed regions show the probability of an incorrect decision. If $\sigma_1$ and $\sigma_0$ represent the standard deviation of the received signal power of a “1” bit and a “0” bit respectively then the conditional probabilities are given by

\[
P(0/1) = \frac{1}{\sigma_1 \sqrt{2\pi}} \int_{-\infty}^{I_{th}} \exp\left(-\frac{(I - I_1)^2}{2\sigma_1^2}\right) dI = \frac{1}{2} \text{erfc}\left(\frac{I_1 - I_{th}}{\sigma_1 \sqrt{2}}\right), \tag{5.1.3}
\]

\[
P(1/0) = \frac{1}{\sigma_0 \sqrt{2\pi}} \int_{I_{th}}^{\infty} \exp\left(-\frac{(I - I_0)^2}{2\sigma_0^2}\right) dI = \frac{1}{2} \text{erfc}\left(\frac{I_{th} - I_0}{\sigma_0 \sqrt{2}}\right), \tag{5.1.4}
\]

where \( \text{erfc} \) stands for the complementary error function, defined as

\[
\text{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_{x}^{\infty} \exp(-y^2) dy. \tag{5.1.5}
\]

By substituting Eqs. (5.1.3) and (5.1.4) in Eq. (5.1.2), the BER is given by

\[
BER = \frac{1}{4} \left[ \text{erfc}\left(\frac{I_1 - I_{th}}{\sigma_1 \sqrt{2}}\right) + \text{erfc}\left(\frac{I_{th} - I_0}{\sigma_0 \sqrt{2}}\right) \right]. \tag{5.1.6}
\]

Equation (5.1.6) shows that the BER depends on the decision threshold $I_{th}$. In practice, $I_{th}$ is optimized to minimize the BER. The minimum occurs when $I_{th}$ is chosen such that

\[
(I_1 - I_{th}) / \sigma_1 = (I_{th} - I_0) / \sigma_0 \equiv Q. \tag{5.1.7}
\]

An explicit expression of $I_{th}$ is

\[
I_{th} = \frac{\sigma_0 I_1 + \sigma_1 I_0}{\sigma_0 + \sigma_1}. \tag{5.1.8}
\]

The BER with the optimum setting of the decision threshold is obtained by using Eqs. (5.1.6) and (5.1.7) and is given by

\[
BER = \frac{1}{2} \text{erfc}\left(\frac{Q}{\sqrt{2}}\right). \tag{5.1.9}
\]
5.2 Eye-Patterns and BER

After an RZ or an NRZ bit sequence is created, propagated (dispersion), noise added and optically and electrically filtered, a resulting Eye-Pattern can be created. An Eye-Pattern shows the detected signal when all the bit-slots are plotted on top of each other. Figure 5.2 shows an example of an Eye-Pattern. $I_1$ and $I_0$ are determined as indicated in this figure. The variance around these values (for a small band around the decision time window) can be calculated, thus finding $\sigma_1$ and $\sigma_0$. With these four values the BER can be calculated using Eqs. (5.1.7) and (5.1.9). Increasing the propagation distance and decreasing the OSNR will make the eye close and the BER become worse.

![Figure 5-2: Example of an Eye-Pattern with the one and zero levels indicated.](image)
5.2.1 RZ-sequences

Figure 5.3 shows the eye-patterns for 6 different combinations of OSNR and propagation length. (Script: Appendix L) The duty cycle = 30% and $|\beta_2| = 20 \text{ ps}^2/\text{km}$. The calculated BER is shown on every graph.

![Eye-Patterns for six different combinations of OSNR and propagation lengths. Duty cycle = 30%, $|\beta_2| = 20 \text{ ps}^2/\text{km}$.](image)

Note that as the propagation length increases the “$I_1$” level decreases due to dispersion. Also, as the OSNR decreases, the BER increases.
5.2.2 NRZ-sequences

Figure 5.4 shows the eye-patterns for six different combinations of OSNR and propagation lengths. (Script: Appendix M) The order of the Super-Gaussian pulses is \( m = 2 \), the duty cycle = 90\% and \( |\beta_2| = 20 \text{ ps}^2/\text{km} \). The calculated BER is shown on every graph.

Due to ISI the eye will close horizontally. The OSNR makes the eye close vertically.
Figure 5.5 shows the eye-patterns for six different combinations of OSNR and propagation lengths. (Script: Appendix M) The order of the Super-Gaussians is $m = 3$, the duty cycle $= 95\%$ and $|\beta|^2 = 20$ ps$^2$/km. The calculated BER is shown on every graph.
5.3 **System Performance Metrics**

In order to characterize the system performance and obtain a more quantitative idea of the effect of dispersion we can calculate the BER for different OSNR levels. In the presence of dispersion we expect to require more OSNR to achieve a certain level of BER due to pulse distortion. The extra amount of OSNR that is required is referred to as an *OSNR penalty*. In this way we can fully characterize the system performance by comparing the *required OSNR* after propagation with the required OSNR in *back-to-back* configuration.

The back-to-back configuration baseline is obtained by connecting the transmitter directly to the receiver. This can be simulated by introducing a propagation length of $z = 0$. Figure 5.6 shows a BER versus OSNR curve for an RZ-sequence. (Script: Appendix N) $B_o$ and $B_e$ stand for the optical and electrical Butterworth filter bandwidths respectively.

![Figure 5-6: BER versus OSNR for different propagation lengths for an RZ sequence](image)
A graph like figure 5.6 can also be obtained for an NRZ-sequence. Figure 5.7 shows this graph for an NRZ-sequence built with Super-Gaussian pulse of $m = 2$ order. Figure 5.8 shows the same graph for Super-Gaussian $m = 3$ pulses. (Script: Appendix O)

![Figure 5-7: BER versus OSNR for different propagation lengths of an NRZ sequence with Super-Gaussian Order $m = 2$.](image)

![Figure 5-8: BER versus OSNR for different propagation lengths of an NRZ sequence with Super-Gaussian Order $m = 3$.](image)
5.4 1-dB OSNR Penalty Point

From the previous section we note that the OSNR requirements in back-to-back configuration are similar for RZ and NRZ bit sequences. In a real system the back-to-back baseline will change depending on system parameters such as the transmitter extinction ratio, the optical and electrical filter shapes and bandwidths, the data modulation format, etc. [2].

There are two relevant BER points that can be used for quantitative estimations of system penalties. These points correspond to a BER of approximately $10^{-5}$ and $10^{-9}$. The latter is an arbitrary reference point that is commonly used to characterize system performance. The former represents a more physical point since by using forward error correction (FEC) codes a BER of $10^{-5}$ can be corrected to $10^{-16}$. A BER of $10^{-16}$ corresponds to the error-free criterion currently adopted for high-capacity optical communication systems.

Let us take a closer look at the graphs obtained in section 5.3. Figure 5.9 shows a more detailed version of figure 5.6. The 1-dB OSNR penalty point at a BER of $10^{-5}$ occurs at $1.3L_{D\text{, gaussian}}$. Therefore the definition of a dispersion length ($L_{D\text{, gaussian}}$) for Gaussian pulses is a good indication for the propagation length at which dispersive effects will become a problem.

![Figure 5-9: More detailed version of figure 5.6. (BER versus OSNR for an RZ sequence)](image-url)
Figure 5.10 shows a more detailed version of figure 5.7. It can be noted that the 1-dB OSNR penalty point at a BER of $10^{-5}$ for an NRZ-system built with Super-Gaussian pulses occurs at $0.3L_{D_{\text{gaussian}}}$. The definition of a dispersion length of Gaussian pulses is therefore not a good indication of the length at which dispersive effects will become a problem for an NRZ-system. The new dispersion length, introduced in section 2.2 and given by Eq. (2.2.12) for short propagation lengths is

$$L_{D_{\text{super gaussian}}} = \frac{L_{D_{\text{gaussian}}}}{m^2} = 0.25L_{D_{\text{gaussian}}},$$  \hspace{1cm} (5.4.1)

and provides a more suitable definition that allows us to calculate beforehand at which propagation distance dispersive effects will become significant.

![Figure 5-10: More detailed version of figure 5.7. (BER versus OSNR for a NRZ sequence)](image-url)
5.5 Conclusions

In this report we addressed dispersion effects in linear optical communication links. We first started by explaining how dispersion can be calculated analytically. We continued by deriving a new way to define a dispersion length for the case of Super-Gaussian pulses. The usefulness of this definition was confirmed by simulations. Gaussian and Super-Gaussian pulse sequences were built and dispersive effects were imposed on them. Then, a certain amount of noise was added to the signal which allowed us to set a given OSNR level. To represent the photodetection process of the signal at the receiver we optically and electrically filtered the bit sequence with Butterworth filters. Then, the corresponding eye pattern was built and the respective BER calculated. This allowed us to plot graphs of BER versus OSNR and determine the amount of dispersion that led to a 1-dB OSNR penalty. These results confirmed that our new definition for the dispersion length is better suited when applied to NRZ bit sequences as compared to the usual definition based on Gaussian pulses.

5.6 Recommendations for Further Investigation

In this report we limited ourselves to calculating the influence of linear effects on system performance. However, in high-capacity WDM optical systems, non-linear effects arising from the intensity dependence of the refractive index become important, and may lead to severe pulse distortion compromising system performance [4].

A recommendation for further investigation includes taking into account non-linear effects in our system simulations to validate some of the concepts introduced in this report. Note that even in the presence of non-linear effects we can still evaluate system performance using the metrics introduced in this report. Curves of BER versus OSNR and the required OSNR concept will prove to be useful tools for estimating system penalties arising from non-linear pulse distortion.
Appendices

Appendix A: Analytical Broadening

% This script calculates and plots the broadening factor of a Gaussian pulse with different chirp factors for a range of zero to two dispersion lengths.

figure(1);
axes('fontsize',14);

z=[0:0.001:2];

for i=1:3,
    if i == 1,
        C=0;
    elseif i == 2,
        C=-2;
    elseif i == 3,
        C=2;
    end

    broadening = ((1+C*z).^2 + (z).^2).^(1/2);
    plot(z,broadening,'b','linewidth',1.5);
    axis([0 2 0 4]);
    grid on;
    hold on;

    if i == 1,
        gtext('C = 0','color','r','fontsize',15);
    elseif i == 2,
        gtext('C = -2','color','r','fontsize',15);
    elseif i == 3,
        gtext('C = 2','color','r','fontsize',15);
    end
end

gtext('\beta_2 > 0','color','r','fontsize',15);
xlabel('DISTANCE, z/Ld','fontsize',14);
ylabel('BROADENING FACTOR, \sigma_1 / \sigma_0','fontsize',14);
hold off;
Appendix B: Calculated Broadening

% This script calculates and plots the simulated broadening factor
% for different chirp and m-factors for a range of zero to two
% dispersion lengths.

% First a suitable time and accompanying frequency vector are created

B = 10e9;
Nsamples = 8192;
Nbits = 8;
duty_cycle = 0.50;
beta = 1;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Nsamples/2-1);
minw = -dw * Nsamples/2;
w = linspace(minw,maxw,Nsamples);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

E = [0 0 0 0 1 0 0 0]

max_t_bit = (dt * Nsamples/2) + (Nbits-4)*(Nsamples*dt) - dt/2;
min_t_bit = -(dt * Nsamples/2)-(4-1)*(Nsamples*dt) + dt/2;
t_bit = linspace(min_t_bit,max_t_bit,Ntotal);

% Second a routine for calculating the broadening factor after
% a number of dispersion lengths is created. During these
% calculations the results are plotted.

for m = 1:3,
    for v = 1:3,
        if v == 1,
            C = 0;
        elseif v == 2,
            C = -2;
        else v == 3,
            C = 2;
        ...
    end
end
C = 2;
end

h = exp(-(1+i*C)/2).*((t_bit./T0).^((2*m)));
absh = abs(h);
h_power = absh.^2;

H = fft(h);
G = fftshift(H);

for n = 0:100,
e = n/50;
g(n+1) = n;
z = Ld * e;
J = G .* exp((j/2) .* beta .* (w.^2) .* z);
k = ifft(J);
h = abs(k);
h_power = h.^2;
t2 = sum((t_bit.^2).* h_power) / sum(h_power);
t_2 = sum(t_bit .* h_power) / sum(h_power);
sigma(n+1) = sqrt (t2 - (t_2)^2);
end

if v == 2,
[b u]=min(sigma);
calculated_minimum(m,1) = b/sigma(1);
calculated_minimum(m,2) = (g(u))/50;
end

figure(m);
if v == 1,
axes('fontsize',14);
end
plot(g/50,(sigma./sigma(1)),'b','linewidth',1.5);
hold on;
grid on;
axis([0 2 0 4]);
xlabel('DISTANCE, z/Ld','fontsize',14);
ylabel('BROADENING FACTOR, \sigma_1 / \sigma_0','fontsize',14);

if v == 1,
gtext('C = 0','color','r','fontsize',15);
elseif v == 2,
gtext('C = -2','color','r','fontsize',15);
elseif v == 3,
gtext('C = 2','color','r','fontsize',15);
end

if m == 2,
figure(4);
subplot(1,2,1);
plot(g/50,(sigma./sigma(1)),'b','linewidth',1.5);
xlabel('DISTANCE, z/Ld','fontsize',14);
ylabel('BROADENING FACTOR,\sigma_1 / \sigma_0','fontsize',14);
axis([0 2 0 4]);
grid on;
hold on;
elseif m == 3,
    figure(4),
    subplot(1,2,2);
    plot(g/50, (sigma./sigma(1)),'b','linewidth',1.5);
    xlabel('DISTANCE, z/Ld','fontsize',14);
    axis([0 2 0 4]);
    grid on;
    hold on;
end

if m == 2 && v == 1,
    gtext('C = 0','color','r','fontsize',13);
elseif m == 2 && v == 2,
    gtext('C = -2','color','r','fontsize',13);
elseif m == 2 && v == 3,
    gtext('C = 2','color','r','fontsize',13);
elseif m == 3 && v == 1,
    gtext('C = 0','color','r','fontsize',13);
elseif m == 3 && v == 2,
    gtext('C = -2','color','r','fontsize',13);
elseif m == 3 && v == 3,
    gtext('C = 2','color','r','fontsize',13);
end

end

figure(m);
if m == 1,
    gtext('Gaussian Pulse, \beta_2 > 0','color','r','fontsize',13);
elseif m == 2,
    gtext('Super-Gaussian Pulse with m = 2, \beta_2 > 0','color','r','fontsize',13);
else m == 3,
    gtext('Super-Gaussian Pulse with m = 3, \beta_2 > 0','color','r','fontsize',13);
end

figure(4);
if m == 2 && v ==3,
    gtext('m = 2','color','r','fontsize',13);
elseif m == 3 && v ==3,
    gtext('m = 3','color','r','fontsize',13);
end
% This script plots the different pulse shapes and accompanying
% spectra for Super Gaussian pulses with m = 2, 3 and 4

% First a suitable time and accompanying frequency vector is created

B = 1;
Nsamples = 2^13;
Nbits = 64;
duty_cycle = 0.4;
beta = 1;
C = 0;
m = 2;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Ntotal/2)-dt/2;
min_t_bit = -(dt * Ntotal/2)+dt/2;

t_total = linspace(-deltaT/2+dt/2,(deltaT/2-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

% Second a routine for calculating the pulse shapes and accompanying
% spectra is created. During these calculations the results are
% plotted.

for m = 2:4,

    h = exp(-(1/2).*((t_total./T0).^(2*m)));
h_power = (abs(h)).^2;
H = fft(h);
G = fftshift(H);
G_power = (abs(G)).^2;

    figure(1);
subplot(1,3,(m-1));
plot(t_total,h_power,'b','linewidth',1.5);
gtext('m = 2','color','r','fontsize',15);
end
gtext('m = 3','color','r','fontsize',15);
xlabel('time [ps]','fontsize',14);
elseif m == 4,
gtext('m = 4','color','r','fontsize',15);
end

figure(2);
subplot(1,3,(m-1));
plot(w,G_power,'b','linewidth',1.5);
gtext('m = 2','color','r','fontsize',15);
elseif m == 3,
gtext('m = 3','color','r','fontsize',15);
xlabel('frequency [rad/s]','fontsize',14);
elseif m == 4,
gtext('m = 4','color','r','fontsize',15);
end

w2 = sum((w.^2).* G_power) / sum(G_power);
w_2 = sum(w .* G_power) / sum(G_power);
sigmaw(m-1) = sqrt (w2 - (w_2)^2);
end
Appendix D: Spectral Width Super Gaussian Pulse

Figure D-1 below shows the pulse shape and accompanying frequency spectrum for different $m$-orders.

![Figure D-1: left: Pulse shapes for several Super Gaussians. Right: Accompanying spectra of the Super Gaussian Pulses](image)

Table D-1 lists the RMS spectral-width for the different pulse shapes. Furthermore $\omega_{\text{rms},m=1}/\omega_{\text{rms},m=1}$ is calculated. As can be seen from the table the following equation for the RMS spectral-width of a Super-Gaussian pulse holds

$$\Delta \omega_{\text{rms, super gaussian}} \propto \sqrt{m} \Delta \omega_{\text{rms, gaussian}}$$  \hspace{1cm}(D.1)

<table>
<thead>
<tr>
<th>$m$</th>
<th>$\omega_{\text{rms}}$ [rad/s]</th>
<th>$\omega_{\text{rms}}/\omega_{\text{rms},m=1}$</th>
<th>$\sqrt{m}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5,657E+10</td>
<td>1,00</td>
<td>1,00</td>
</tr>
<tr>
<td>2</td>
<td>8,056E+10</td>
<td>1,42</td>
<td>1,14</td>
</tr>
<tr>
<td>3</td>
<td>9,866E+10</td>
<td>1,74</td>
<td>1,73</td>
</tr>
<tr>
<td>4</td>
<td>1,138E+11</td>
<td>2,01</td>
<td>2,00</td>
</tr>
<tr>
<td>5</td>
<td>1,272E+11</td>
<td>2,25</td>
<td>2,24</td>
</tr>
<tr>
<td>6</td>
<td>1,392E+11</td>
<td>2,46</td>
<td>2,45</td>
</tr>
<tr>
<td>7</td>
<td>1,503E+11</td>
<td>2,66</td>
<td>2,65</td>
</tr>
<tr>
<td>8</td>
<td>1,606E+11</td>
<td>2,84</td>
<td>2,83</td>
</tr>
</tbody>
</table>

Table D-1: Spectral widths ($\omega_{\text{rms}}$) for several Super Gaussian Pulse shapes.
Appendix E: $\omega T$ versus $L_D$ and $d(\omega T)/dL_D$ versus $L_D$ for $m = 1, 2$ and 3

% This script calculates and plots the product of the RMS time-width
% and the RMS spectral-width, furthermore it calculates and plots the
% derivative of this graph with respect to the propagation length

% First a suitable time and accompanying frequency vector are created

B = 10e9;
Nsamples = 2^9;
Nbits = 32;
beta = 1;
dutycycle = 0.25;
C = 0;

T0 = dutycycle/(2*B);
Ld = (T0.^2)/beta;
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

for m=1:3,
    h_low = 0;
    h_high = exp(-((1+i*C)/2).*((-dt/2./T0).^(2*m)));
    h_rise = exp(-((1+i*C)/2).*((t_bit_1st_half./T0).^(2*m)));
    h_fall = exp(-((1+i*C)/2).*((t_bit_2nd_half./T0).^(2*m)));
    h_low_after_fall = exp(-((1+i*C)/2).*((t_bit_3rd_half./T0).^(2*m)));
    h_rise_after_low = exp(-

% Second a routine for calculating the product stated above, after
% the pulse has propagated a certain number of dispersion lengths,
% is created.

for m=1:3,
Traineeship on Fiber-Optic Communications

\[
\frac{(1+iC)}{2} \cdot \left(\frac{t_{bit\_0th\_half}}{T_0}\right)^{2m}
\]

for \( f=1:Nbits \),
if \( E(f) == 0 \),
\[ q = [(f-1)\times Nsamples + 1:1:Nsamples*f] \]
\[ h(q) = h_{\text{low}} \]
elseif \( E(f) == 1 \),
\[ q_{\text{1st\_half}} = [(f-1)\times Nsamples+1:1:(f-1)\times Nsamples+Nsamples/2] \]
if \( f > 1 \),
\[ q_{\text{0th\_half}} = q_{\text{1st\_half}} - Nsamples/2 \]
\[ h(q_{\text{0th\_half}}) = h_{\text{rise\_after\_low}} \]
end
\[ h(q_{\text{1st\_half}}) = h_{\text{rise}} \]
break
end
end

for \( p=f:(Nbits-1) \),
\[ q_{\text{minus\_half}} = [p\times Nsamples-(Nsamples/2-1):1:p\times Nsamples] \]
\[ q_{\text{plus\_half}} = q_{\text{minus\_half}} + Nsamples/2 \]
if \( E(p) == 0 \) \&\& \( E(p+1) == 0 \),
\[ h(q_{\text{plus\_half}}) = h_{\text{low}} \]
\[ h(q_{\text{minus\_half}}) = h_{\text{low}} \]
elseif \( E(p) == 0 \) \&\& \( E(p+1) == 1 \),
\[ h(q_{\text{plus\_half}}) = h_{\text{rise}} \]
\[ h(q_{\text{minus\_half}}) = h_{\text{rise\_after\_low}} \]
elseif \( E(p) == 1 \) \&\& \( E(p+1) == 0 \),
\[ h(q_{\text{plus\_half}}) = h_{\text{low\_after\_fall}} \]
\[ h(q_{\text{minus\_half}}) = h_{\text{fall}} \]
elseif \( E(p) == 1 \) \&\& \( E(p+1) == 1 \),
\[ h(q_{\text{plus\_half}}) = h_{\text{high}} \]
\[ h(q_{\text{minus\_half}}) = h_{\text{high}} \]
end
end

einde = [Nsamples*Nbits-(Nsamples/2-1):1:Nsamples*Nbits];
if \( E(Nbits) == 0 \),
\[ h(einde) = 0 \]
elseif \( E(Nbits) == 1 \),
\[ h(einde) = h_{\text{fall}} \]
end

abs\_h = abs(h);
h\_power = abs\_h.^2;
H = fft(h);
G = fftshift(H);
for \( i=0:36 \),
if \( i == 0 \),
\[ z = 0 \]
elseif \( i == 1 \),
\[ z = 0.1 \times Ld \]
elseif \( i == 2 \),
\[ z = 0.2 \times Ld \]
elseif i == 3,
z = 0.3 * Ld;
elseif i == 4,
z = 0.4 * Ld;
elseif i == 5,
z = 0.5 * Ld;
elseif i == 6,
z = 0.6 * Ld;
elseif i == 7,
z = 0.7 * Ld;
elseif i == 8,
z = 0.8 * Ld;
elseif i == 9,
z = 0.9 * Ld;
elseif i == 10,
z = 1.0 * Ld;
elseif i == 11,
z = 1.1 * Ld;
elseif i == 12,
z = 1.2 * Ld;
elseif i == 13,
z = 1.3 * Ld;
elseif i == 14,
z = 1.4 * Ld;
elseif i == 15,
z = 1.5 * Ld;
elseif i == 16,
z = 1.6 * Ld;
elseif i == 17,
z = 1.7 * Ld;
elseif i == 18,
z = 1.8 * Ld;
elseif i == 19,
z = 1.9 * Ld;
elseif i == 20,
z = 2.0 * Ld;
elseif i == 21,
z = 2.1 * Ld;
elseif i == 22,
z = 2.2 * Ld;
elseif i == 23,
z = 2.3 * Ld;
elseif i == 24,
z = 2.4 * Ld;
elseif i == 25,
z = 2.5 * Ld;
elseif i == 26,
z = 3 * Ld;
elseif i == 27,
z = 3.5 * Ld;
elseif i == 28,
z = 4 * Ld;
elseif i == 29,
z = 4.5 * Ld;
elseif i == 30,
z = 5 * Ld;
elseif i == 31,
\[ z = 5.5 \times L_d; \]

\[
\text{elseif } i == 32, \\
z = 6 \times L_d; \\
\text{elseif } i == 33, \\
z = 6.5 \times L_d; \\
\text{elseif } i == 34, \\
z = 7 \times L_d; \\
\text{elseif } i == 35, \\
z = 7.5 \times L_d; \\
\text{elseif } i == 36, \\
z = 8 \times L_d; \\
\]

\[ v = \text{linspace}(i,i,N_{total}); \]

\[ J = G \times \exp((j/2) \times \beta \times (w \times z)); \]

\[ Y = \text{abs}(J); \]

\[ \text{power}_Y = Y \times z; \]

\[ h = \text{ifi}(J); \]

\[ \text{habs} = \text{abs}(h); \]

\[ h_{power} = \text{habs} \times z; \]

\[ t_2 = \text{sum}(t_{total} \times z) \times \text{sum}(h_{power}); \]

\[ t_2 = \text{sum}(t_{total} \times z) \times \text{sum}(h_{power}); \]

\[ \text{sigmat}(m, i+1) = \sqrt{t_2 - (t_2 \times z)^2}; \]

\[ w_2 = \text{sum}(w \times z) \times \text{sum}(power_Y); \]

\[ w_2 = \text{sum}(w \times z) \times \text{sum}(power_Y); \]

\[ \text{sigmaw}(m, i+1) = \sqrt{w_2 - (w_2 \times z)^2}; \]

\[ \text{product}(m,:) = \text{sigmat}(m,: \times \text{sigmaw}(m,:); \]

\[ \text{ends} \]

\[ \text{widths} = [0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5 8; \]

\[ \text{product}]; \]

\% Here the results are plotted.

\[ \text{figure}(1); \]

\[ \text{subplot}(1,2,1); \]

\[ \text{plot}(\text{widths}(1,:),\text{widths}(2,:),'bx-','linewidth',2); \]

\[ \text{hold on}; \]

\[ \text{gtext('m = 1','fontsize',15);} \]

\[ \text{plot}(\text{widths}(1,:),\text{widths}(3,:),'rx-','linewidth',2); \]

\[ \text{gtext('m = 2','fontsize',15);} \]

\[ \text{plot}(\text{widths}(1,:),\text{widths}(4,:),'gx-','linewidth',2); \]

\[ \text{gtext('m = 3','fontsize',15);} \]

\[ \text{xlabel('DISTANCE, z/Ld','fontsize',14);} \]

\[ \text{ylabel('\omegaT','fontsize',14);} \]

\[ \text{hold off;} \]

\[ \text{subplot}(1,2,2); \]

\[ \text{dLd} = \text{diff}(\text{transpose}(\text{widths}(1,:))); \]

\[ \text{dwT}_m1 = \text{diff}(\text{transpose}(\text{widths}(2,:))); \]

\[ \text{dwT}_m2 = \text{diff}(\text{transpose}(\text{widths}(3,:))); \]

\[ \text{dwT}_m3 = \text{diff}(\text{transpose}(\text{widths}(4,:))); \]

\[ \text{dwT_dLd}_m1 = \text{dwT}_m1 / \text{dLd}; \]
\[
dwT_{dLd_m2} = dwT_{m2}/dLd;
\]
\[
dwT_{dLd_m3} = dwT_{m3}/dLd;
\]
\[
\text{lengths} = [0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5];
\]

```
plot(lengths,dwT_dLd_m1,'bx-','linewidth',2);
hold on;
gtext('m = 1','fontsize',15);
plot(lengths,dwT_dLd_m2,'rx-','linewidth',2);
gtext('m = 2','fontsize',15);
plot(lengths,dwT_dLd_m3,'gx-','linewidth',2);
gtext('m = 3','fontsize',15);
xlabel('DISTANCE, z/Ld','fontsize',14);
ylabel('d(\omegaT)/dLd','fontsize',14);
hold off;
```

```matlab
factor2 = (log(dwT_dLd_m2./dwT_dLd_m1))/log(2);
factor3 = (log(dwT_dLd_m3./dwT_dLd_m1))/log(3);
factors = [transpose(factor2); transpose(factor3)];
```

```matlab
% Results are stored in an appropriate file
fid = fopen('factors','w');
fprintf(fid,'%6.5f	 %6.5f
',factors);
fclose(fid);
```
Appendix F: 3D-plot time frequency Super-Gaussian

% This script calculates and makes a 3D plot of a Super-Gaussian pulse
% that travels through a fiber. Furthermore the 3D plot of the
% accompanying spectrum is made

% First a suitable time and accompanying frequency vector are created

B = 10e9;
Nsamples = 2^10;
Nbits = 32;
dutycycle = 0.25;
C = 0;
m = 2;

T0 = dutycycle/(2*B);
Ld = (T0.^2)/beta;
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t_bit_0th_half = linspace(min_t_bit - (dt * Nsamples/2),min_t_bit – dt,Nsamples/2);
t_bit_1st_half = linspace(min_t_bit,-dt/2,Nsamples/2);
t_bit_2nd_half = linspace(dt/2,max_t_bit,Nsamples/2);
t_bit_3rd_half = linspace(max_t_bit + dt,max_t_bit + (dt * Nsamples/2),
Nsamples/2);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

E = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

h_low = 0;
h_high = exp(-(1i*C)/2).*((-dt/2./T0).^((2*m)));
h_rise = exp(-(1i*C)/2).*((t_bit_1st_half./T0).^((2*m)));
h_fall = exp(-(1i*C)/2).*((t_bit_2nd_half./T0).^((2*m)));
h_low_after_fall = exp(-(1i*C)/2).*((t_bit_3rd_half./T0).^((2*m)));
h_rise_after_low = exp(-(1i*C)/2).*((t_bit_0th_half./T0).^((2*m)));

% Second the pulse is created

for f=1:Nbits,
if E(f) == 0,
q = [(f-1)*Nsamples + 1:1:Nsamples*f];
h(q) = h_low;
end
end
elseif E(f) == 1,
    q_1st_half = [(f-1)*Nsamples+1:1:(f-1)*Nsamples+Nsamples/2];
    if f > 1,
        q_0th_half = q_1st_half - Nsamples/2;
        h(q_0th_half) = h_rise_after_low;
    end
    h(q_1st_half) = h_rise;
    break
end
end
for p=f:(Nbits-1),
    q_minus_half = [p*Nsamples-(Nsamples/2-1):1:p*Nsamples];
    q_plus_half = q_minus_half + Nsamples/2;
    if E(p) == 0 && E(p+1) == 0,
        h(q_plus_half) = h_low;
        h(q_minus_half) = h_low;
    elseif E(p) == 0 && E(p+1) == 1,
        h(q_plus_half) = h_rise;
        h(q_minus_half) = h_rise_after_low;
    elseif E(p) == 1 && E(p+1) == 0,
        h(q_plus_half) = h_low_after_fall;
        h(q_minus_half) = h_fall;
    elseif E(p) == 1 && E(p+1) == 1,
        h(q_plus_half) = h_high;
        h(q_minus_half) = h_high;
    end
end
einde = [Nsamples*Nbits-(Nsamples/2 -1):1:Nsamples*Nbits];
if E(Nbits) == 0,
    h(einde) = 0;
elseif E(Nbits) == 1,
    h(einde) = h_fall;
end
absh = abs(h);
h_power = absh.^2;
H = fft(h);
G = fftshift(H);

% Here the pulse is propagated through the fiber. During the
% calculations the results are plotted in a 3D graph.
for i=0:4,
    if i == 0,
        z = 0;
    elseif i == 1,
        z = 1/2 * Ld;
    elseif i == 2,
        z = 1 * Ld;
    elseif i == 3,
        z = 3/2 * Ld;
    elseif i == 4,
        z = 2 * Ld;
end
```matlab
elseif i == 4,
    z = 5/2 * Ld;
end

v = linspace(i/2,i/2,Ntotal);
J = G .* exp((j/2) .* beta .* (w.^2) .* z);
Y = abs(J);
power_Y = Y.^2;
h = ifft(J);
habs = abs(h);
h_power = habs.^2;

figure(1);
subplot(1,2,1);
plot3(v,t_total,h_power,'linewidth',2);
hold on;

subplot(1,2,2);
plot3(v,w,power_Y,'linewidth',2);
hold on;
end

subplot(1,2,1);
xlabel('DISTANCE, z/Ld','fontsize',12);
ylabel('time [ps]','fontsize',12);
zlabel('field power','fontsize',12);
view(40,-20);
axis([0 2.5 1.6e-9 1.7e-9 0 1]);
hold off;

subplot(1,2,2);
xlabel('DISTANCE, z/Ld','fontsize',12);
ylabel('frequency [rad/s]','fontsize',12);
zlabel('field power','fontsize',12);
view(40,-20);
axis([0 2.5 -5e11 5e11 0 8e4]);
hold off;
```
Appendix G: 8 bit RZ-sequence plus Dispersion

% This script creates a RZ-sequence of 8 bits and propagates it over a
% range of dispersion lengths.

% First a suitable time and accompanying frequency vector is created

B = 10e9;
Nsamples = 2^8;
Nbits = 8;
duty_cycle = 0.30;
beta = 20e-24;
C = 0;
m = 1;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t_bit = linspace(min_t_bit,max_t_bit,Nsamples);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

E = [0 1 0 1 1 1 0 1];
q = [1:1:Nsamples];
for p=1:Nbits,
    if E(p) == 1;
        h(q) = exp(-(1/2).*((t_bit./T0).^2));
    else
        h(q) = 0;
    end
    q = q + Nsamples;
end

% Second the power of the pulse and accompanying spectrum is plotted.

figure(1);
subplot(2,1,1);
absh = abs(h);
h_power = absh.^2;
plot(t_total,h_power,'linewidth',2);
Traineeship on Fiber-Optic Communications

axis([0 dt*Ntotal 0 1.1]);
ylabel('field power','fontsize',14);
xlabel('time [s]','fontsize',14);
H = fftshift(fft(h));
H_power = (abs(H)).^2;

subplot(2,1,2);
plot(w,H_power,'linewidth',2);
xlabel('frequency [rad/s]','fontsize',14);
ylabel('field power','fontsize',14);
axis([-2e11 2e11 0 2.5e5]);

% Here the pulse is propagated over several dispersion lengths.

figure(2)
for i=0:3,
    subplot(2,2,i+1);
    z = i*Ld;
    J = H .* exp((j/2) .* beta .* (w.^2) .* z);
    h  = ifft(J);
    h_power = (abs(h)).^2;
    plot(t_total,h_power,'linewidth',2);
    axis([0 Ntotal*dt 0 1.1]);
    xlabel('time [s]','fontsize',12);
ylabel('field power','fontsize',12);
end

gtext('z = 0','color','r','fontsize',15);
gtext('z = 11.25 km (L_D)','color','r','fontsize',15);
gtext('z = 22.5 km (2L_D)','color','r','fontsize',15);
gtext('z = 33.75 km (3L_D)','color','r','fontsize',15);
Appendix H: 8 bit NRZ-sequence plus Dispersion

% This script creates a NRZ-sequence of 8 bits and propates it over a
% range of dispersion lengths.

% First a suitable time and accompanying frequency vector is created

B = 10e9;
Nsamples = 2^8;
nbits = 8;
duty_cycle = 0.95;
beta = 20e-24;
C = 0;
m = 3;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t_bit_0th_half = linspace(min_t_bit - (dt * Nsamples/2),min_t_bit -
dt,Nsamples/2);
t_bit_1st_half = linspace(min_t_bit,-dt/2,Nsamples/2);
t_bit_2nd_half = linspace(dt/2,max_t_bit,Nsamples/2);
t_bit_3rd_half = linspace(max_t_bit + dt,max_t_bit + (dt * Nsamples/2),
Nsamples/2);

E = [0 1 0 1 1 1 0 1];
h_low = 0;
h_high = exp(-((1+i*C)/2).*(-(dt/2./T0).^(2*m)));
h_rise = exp(-((1+i*C)/2).*((t_bit_1st_half./T0).^((2*m))));
h_fall = exp(-((1+i*C)/2).*((t_bit_2nd_half./T0).^((2*m))));
h_low_after_fall = exp(-((1+i*C)/2).*((t_bit_3rd_half./T0).^((2*m))));
h_rise_after_low = exp(-((1+i*C)/2).*((t_bit_0th_half./T0).^((2*m))));

for f=1:Nbits,
    if E(f) == 0,
        q = [(f-1)*Nsamples + 1:1:Nsamples*f];
        h(q) = h_low;
    end
end
elseif E(f) == 1,
    q_1st_half = [(f-1)*Nsamples+1:1:(f-1)*Nsamples+Nsamples/2];
    if f > 1,
        q_0th_half = q_1st_half - Nsamples/2;
        h(q_0th_half) = h_rise_after_low;
    end
    h(q_1st_half) = h_rise;
    break
end
end
for p=f:(Nbits-1),
    q_minus_half = [p*Nsamples-(Nsamples/2-1):1:p*Nsamples];
    q_plus_half = q_minus_half + Nsamples/2;
    if E(p) == 0 && E(p+1) == 0,
        h(q_plus_half) = h_low;
        h(q_minus_half) = h_low;
    elseif E(p) == 0 && E(p+1) == 1,
        h(q_plus_half) = h_rise;
        h(q_minus_half) = h_rise_after_low;
    elseif E(p) == 1 && E(p+1) == 0,
        h(q_plus_half) = h_low_after_fall;
        h(q_minus_half) = h_fall;
    elseif E(p) == 1 && E(p+1) == 1,
        h(q_plus_half) = h_high;
        h(q_minus_half) = h_high;
    end
end
einde = [Nsamples*Nbits-(Nsamples/2 -1):1:Nsamples*Nbits];
if E(Nbits) == 0,
    h(einde) = 0;
elseif E(Nbits) == 1,
    h(einde) = h_fall;
end
% Second the power of the pulse and accompanying spectrum is plotted.
figure(1);
subplot(2,1,1);
abh=abs(h);
H_power = abh.^2;
plot(t_total,H_power,'linewidth',2);
ylabel('field power','fontsize',14);
xlabel('time [s]','fontsize',14);
axis([0 dt*Ntotal 0 1.1]);
H = fftshift(fft(h));
H_power = (abs(H)).^2;
subplot(2,1,2);
plot(w,H_power,'linewidth',2);
xlabel('frequency [rad/s]','fontsize',14);
ylabel('field power','fontsize',14);
axis([-2e11 2e11 0 2e6]);

% Here the pulse is propagated over a range of kilometers.
figure(2);
for i=0:3,
    subplot(2,2,i+1);
    z = i*Ld/5;
    J = H .* exp((j/2) .* beta .* (w.^2) .* z);
    h = ifft(J);
    h_power = (abs(h)).^2;
    plot(t_total,h_power,'linewidth',2);
    axis([0 Ntotal*dt 0 1.8]);
    xlabel('time [s]', 'fontsize', 12);
    ylabel('field power', 'fontsize', 12);
end

gtext('z = 0', 'color', 'r', 'fontsize', 15);
gtext('z = 22.6 km (1/5L_D)', 'color', 'r', 'fontsize', 15);
gtext('z = 45.2 km (2/5L_D)', 'color', 'r', 'fontsize', 15);
gtext('z = 67.7 km (3/5L_D)', 'color', 'r', 'fontsize', 15);
Appendix I: Routine-check on building-code for NRZ-sequences

This appendix will shortly discuss the validity of the routine for creating a NRZ-sequence. First of all the correctness of building of and performing dispersion calculations on a single Super-Gaussian pulse is assumed. Figure I.1 shows 2 pulses. The blue pulse is created by a single broad steep Super-Gaussian pulse. The red pulse is created by the attaching routine of Super-Gaussians which is also used for creating a sequence of NRZ-pulses.

Since both pulses are almost equal they should also look equal after propagation. Figure I.2 shows the pulse-shape of the Single broad steep Super-Gaussian and its corresponding spectrum after propagating 0, 1/5L_D, 1/2L_D, L_D, 2L_D, 3L_D, 4L_D and 10L_D. Figure I.3 shows the pulse-shape of 4 consecutive one’s attached after propagating the same lengths. Both graphs are equal and therefore the conclusion can be drawn that the routine for attaching Super-Gaussian pulse-shapes is correct and dispersion results can be trusted.
Figure I-3: Pulse-shape of 4 consecutive “1’s” attached after propagation and its accompanying spectrum.

Another effect that can be noted is that the pulse-shape of a sequence of attached Super-Gaussians first picks up in intensity and narrows after propagation. After a while the pulse-shape will become Gaussian and the intensity will reduce along propagation.

Below the code for creating figures I.1 – I.3 is listed.

```matlab
% This script plots creates 2 almost equal pulses but with different % routines. After that both pulses are propagated and results are plotted
% First a single broad steep Super-Gaussian is created

T = 128;
Nsamples = 2^15;
beta = 1;
C = 0;
m = 9;

T0 = 8;
Ld = (2.^2)/beta;
dt = T / Nsamples;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t = linspace(min_t_bit,max_t_bit,Nsamples);
t_total = linspace(dt/2,Nsamples*dt-dt/2,Nsamples);

dw = 2*pi / T;
deltaw = 2*pi / dt;
maxw = dw * (Nsamples/2-1);
minw = -dw * Nsamples/2;
w = linspace(minw,maxw,Nsamples);
```
Traineeship on Fiber-Optic Communications

\[ h = \exp\left(-\frac{1}{2}\cdot\left(\frac{t}{T0}\right)^{2m}\right); \]

\[ h\_power = (\text{abs}(h))^2; \]

\[ H = \text{fftshift}(\text{fft}(h)); \]

\[ H\_power = (\text{abs}(H))^2; \]

```matlab
figure(1);
plot(t_total,h_power,'b','linewidth',1.5);
hold on;
axis off;
```

% The pulse-shape is propagated.

```matlab
for i=0:7,
    if i == 0,
        z = 0;
    elseif i == 1,
        z = 1/5 * Ld;
    elseif i == 2,
        z = 1/2 * Ld;
    elseif i == 3,
        z = 1 * Ld;
    elseif i == 4,
        z = 2 * Ld;
    elseif i == 5,
        z = 3 * Ld;
    elseif i == 6,
        z = 4 * Ld;
    elseif i == 7,
        z = 10 * Ld;
    end

    v = linspace(i,i,Nsamples);
    J = H .* \exp((j/2) .* beta .* (w.^2) .* z);
    J_power = (\text{abs}(J))^2;
    h = \text{ifft}(J);
    h_power = (\text{abs}(h))^2;

    figure(2);
    subplot(1,2,1);
    plot3(v,t_total,h_power,'linewidth',2);
    hold on;

    subplot(1,2,2);
    plot3(v,w,J_power,'linewidth',2);
    hold on;
end
```

```matlab
figure(2);
subplot(1,2,1);
view(30,-15);
axis([0 8 34 95 0 2]);
axis off;
hold off;
```
% Here almost the same pulse is created but now with the routine of % attaching Super-Gaussian pulses.

B = 0.25;
Nsamples = 2^10;
Nbits = 32;
beta = 1;
C = 0;
m = 2;

T0 = 2;
Ld = (T0.^2)/beta;
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t_bit_0th_half = linspace(min_t_bit - (dt * Nsamples/2),min_t_bit -
dt,Nsamples/2);
t_bit_1st_half = linspace(min_t_bit,-dt/2,Nsamples/2);
t_bit_2nd_half = linspace(dt/2,max_t_bit,Nsamples/2);
t_bit_3rd_half = linspace(max_t_bit + dt,max_t_bit + (dt * Nsamples/2),
Nsamples/2);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

E = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

h_low = 0;
h_high = exp(-((1+i*C)/2).*((-dt/2./T0).^2*m)));
h_rise = exp(-((1+i*C)/2).*((t_bit_1st_half./T0).^2*m)));
h_fall = exp(-((1+i*C)/2).*((t_bit_2nd_half./T0).^2*m)));
h_low_after_fall = exp(-((1+i*C)/2).*((t_bit_3rd_half./T0).^2*m)));
h_rise_after_low = exp(-((1+i*C)/2).*((t_bit_0th_half./T0).^2*m)));

for f=1:Nbits,
    if E(f) == 0,
        q = [(f-1)*Nsamples + 1:1:Nsamples*f];
        h(q) = h_low;
    elseif E(f) == 1,
Traineeship on Fiber-Optic Communications

q_1st_half = [(f-1)*Nsamples+1:1:(f-1)*Nsamples+Nsamples/2];
if f > 1,
    q_0th_half = q_1st_half - Nsamples/2;
    h(q_0th_half) = h_rise_after_low;
end
    h(q_1st_half) = h_rise;
break
end

for p=f:(Nbits-1),
    q_minus_half = [p*Nsamples-(Nsamples/2-1):1:p*Nsamples];
    q_plus_half = q_minus_half + Nsamples/2;
    if E(p) == 0 && E(p+1) == 0,
        h(q_plus_half) = h_low;
        h(q_minus_half) = h_low;
    elseif E(p) == 0 && E(p+1) == 1,
        h(q_plus_half) = h_rise;
        h(q_minus_half) = h_rise_after_low;
    elseif E(p) == 1 && E(p+1) == 0,
        h(q_plus_half) = h_low_after_fall;
        h(q_minus_half) = h_fall;
    elseif E(p) == 1 && E(p+1) == 1,
        h(q_plus_half) = h_high;
        h(q_minus_half) = h_high;
end
end

einde = [Nsamples*Nbits-(Nsamples/2-1):1:Nsamples*Nbits];
if E(Nbits) == 0,
    h(einde) = 0;
elseif E(Nbits) == 1,
    h(einde) = h_fall;
end

h_power = (abs(h)).^2;
H = fftshift(fft(h));
H_power = (abs(H)).^2;

figure(1);
plot(t_total,h_power,'r','linewidth',1.5);
axis([54 75 0 1.1]);
gtext('Single broad steep Super Gaussian','color','b','fontsize',16);
gtext('4 consecutive one''s attached','color','r','fontsize',16);
hold off;

% The second pulse shape is being propagated.

for i=0:7,
    if i == 0,
        z = 0;
    elseif i == 1,
        z = 1/5 * Ld;
    elseif i == 2,
        z = 1/2 * Ld;
end
\begin{verbatim}
elseif i == 3,
z = 1 * Ld;
elseif i == 4,
z = 2 * Ld;
elseif i == 5,
z = 3 * Ld;
elseif i == 6,
z = 4 * Ld;
elseif i == 7,
z = 10 * Ld;
end

v = linspace(i,i,Ntotal);
J = H .* exp((j/2) .* beta .* (w.^2) .* z);
J_power = (abs(J)).^2;
h = ifft(J);
h_power = (abs(h)).^2;

figure(3);
subplot(1,2,1);
plot3(v,t_total,h_power,'linewidth',2);
hold on;

subplot(1,2,2);
plot3(v,w,J_power,'linewidth',2);
hold on;
end

figure(3);
subplot(1,2,1);
view(30,-15);
axis([0 8 34 95 0 2]);
axis off;
hold off;

subplot(1,2,2);
view(30,-15);
axis([0 8 -1.5 1.5 0 2e7]);
axis off;
hold off;
\end{verbatim}
% This script adds a specified level of white Gaussian noise to a RZ-bitsequence

% First a RZ-bitsequence is created

B = 10e9;
Nsamples = 2^8;
Nbits = 256;
duty_cycle = 0.25;
beta = 1;
C = 0;
m = 1;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t_bit = linspace(min_t_bit,max_t_bit,Nsamples);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

Z = randsrc(1,Nbits-8,[0,1]);
E = [0 1 1 0 0 1 0 1 Z];

q = [1:1:Nsamples];
for p=1:Nbits,
    if E(p) == 1;
        h(q) = exp(-(1/2).*((t_bit./T0).^2*m));
    else
        h(q) = 0;
    end
    q = q + Nsamples;
end

h_power = (abs(h)).^2;
H = fftshift(fft(h));
H_power = H.^2;

% then the noise is added to the RZ-bitsequence
signal_power = sum(h_power)
Res_bandwidth_f = 12.5e9
Res_bandwidth_w = 2*pi*Res_bandwidth_f

for i = 1:4,
    SNR = 48 - 5*i;
    Pn = signal_power/(10^(SNR/10));
    delta_Pn = Pn/Res_bandwidth_w;
    Pn_total = delta_Pn * deltaw;

    SNR_new = 10 * log10(signal_power/Pn_total);
    h_noise(i,:) = awgn(h,SNR_new,'measured');
end

h_noise_power = h_noise.^2;

% the bitsequence plus noise is plotted for different noise levels

figure(1);
subplot(2,2,1);
plot(t_total,h_noise_power(1,:),'linewidth',1.5);
axis([0 Nsamples*dt*8 0 1.8]);
xlabel('time [s]', 'fontsize', 13);
ylabel('field intensity', 'fontsize', 13);
subplot(2,2,2);
plot(t_total,h_noise_power(2,:),'linewidth',1.5);
axis([0 Nsamples*dt*8 0 1.8]);
xlabel('time [s]', 'fontsize', 13);
ylabel('field intensity', 'fontsize', 13);
subplot(2,2,3);
plot(t_total,h_noise_power(3,:),'linewidth',1.5);
axis([0 Nsamples*dt*8 0 1.8]);
xlabel('time [s]', 'fontsize', 13);
ylabel('field intensity', 'fontsize', 13);
subplot(2,2,4);
plot(t_total,h_noise_power(4,:),'linewidth',1.5);
axis([0 Nsamples*dt*8 0 1.8]);
xlabel('time [s]', 'fontsize', 13);
ylabel('field intensity', 'fontsize', 13);
gtext('OSNR = 40 dB','color','r','fontsize',12);
gtext('OSNR = 35 dB','color','r','fontsize',12);
gtext('OSNR = 30 dB','color','r','fontsize',12);
gtext('OSNR = 25 dB','color','r','fontsize',12);
Appendix K: Filtering of an RZ-sequence

% This scripts filters a RZ-sequence electrically and optically with a
% Butterworth filter

% First a RZ-sequence of pulses is created

B = 10e9;
Nsamples = 2^8;
Nbits = 256;
duty_cycle = 0.25;
beta = 1;
C = 0;
m = 1;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t_bit = linspace(min_t_bit,max_t_bit,Nsamples);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

Z = randsrc(1,Nbits-8,[0,1]);
E = [0 1 1 0 0 1 0 1 Z];
q = [1:1:Nsamples];
for p=1:Nbits,
    if E(p) == 1;
        h(q) = exp(-(1/2).*((t_bit./T0).^((2*m))));
    else
        h(q) = 0;
    end
    q = q + Nsamples;
end

h_power = (abs(h)).^2;
H = fftshift(fft(h));
H_power = H.^2;

% Noise is added to the RZ-sequence
signal_power = sum(h_power)
Res_bandwidth_f = 12.5e9
Res_bandwidth_w = 2*pi*Res_bandwidth_f
SNR = 33;
Pn = signal_power/(10^(SNR/10));
delta_Pn = Pn/Res_bandwidth_w;
Pn_total = delta_Pn * deltaw;
SNR_new = 10 * log10(signal_power/Pn_total);
h_noise = awgn(h,SNR_new,'measured');
h_noise_power = h_noise.^2;

H_noise = fftshift(fft(h_noise));
H_noise_power = (abs(H_noise)).^2;

% Then the signal is filtered optically and electrically

Bo = 25e9;
[d,c] = butter(3,(2*pi*Bo/w(length(w))));
Be = 7e9;
[b,a] = butter(3,(2*pi*Be/w(length(w))));
h_noise_filtered = filter(d,c,h_noise);
h_noise_filtered_power = (abs(h_noise_filtered)).^2;
h_noise_filtered_power_filtered = filter(b,a,h_noise_filtered_power);
H_noise_filtered = fftshift(fft(h_noise_filtered_power_filtered));
H_noise_filtered_power = (abs(H_noise_filtered)).^2;

figure(1);
subplot(2,1,1)
plot(t_total,h_noise,'linewidth',1.5);
axis([0 Nsamples*8*dt 0 1.5]);
ylabel('field intensity','fontsize',13);
subplot(2,1,2)
plot(t_total,h_noise_filtered_power_filtered,'linewidth',1.5);
axis([0.6*Nsamples*dt Nsamples*8.5*dt -0.1 1.4]);
xlabel('time [s]','fontsize',13);
ylabel('field intensity','fontsize',13);
gtext('OSNR = 30 dB','color','r','fontsize',14);

figure(2);
subplot(1,2,1);
plot(w,H_noise_power,'linewidth',1.5);
xlabel('frequency','fontsize',13);
ylabel('field intensity','fontsize',13);
axis([-2e11 2e11 0 10e7]);
subplot(1,2,2);
plot(w,H_noise_filtered_power,'linewidth',1.5);
xlabel('Frequency','fontsize',13);
axis([-2e11 2e11 0 5e7]);
Appendix L: Eye-pattern of an RZ-sequence

% This script calculates and plots the eye-pattern of a RZ-sequence. Furthermore it calculates the BER that goes with this eye-pattern.
% First a suitable time and accompanying frequency vector are created

B = 10e9;  
Nsamples = 2^8;  
Nbits = 512;  
duty_cycle = 0.3;  
 beta = 20e-24;  
 C = 0;  
  
m = 1;  
 
T0 = (duty_cycle/(B*2));  
Ld = (T0.^2)/beta;  
sigma_nul = T0/sqrt(2);  
  
T = 1/B;  
Ntotal = Nsamples * Nbits;  
dt = T / Nsamples;  
  
deltaT = dt * Nsamples * Nbits;  
  
max_t_bit = (dt * Nsamples/2)-dt/2;  
min_t_bit = -(dt * Nsamples/2)+dt/2;  
  
t_bit = linspace(min_t_bit,max_t_bit,Nsamples);  
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);  
  
dw = 2*pi / deltaT;  
deltaw = 2*pi / dt;  
maxw = dw * (Ntotal/2-1);  
minw = -dw * Ntotal/2;  
w = linspace(minw,maxw,Ntotal);  
  
Z = randsrc(1,Nbits-4,[0,1]);  
E = [0 1 0 Z 0]  
% The RZ-sequence is created

q = [1:1:Nsamples];  
for p=1:Nbits,  
  if E(p) == 1;  
    h(q) = exp(-(1/2).*((t_bit./T0).^(2*m)));  
  else  
    h(q) = 0;  
  end  
  q = q + Nsamples;  
end  
  
H = fftshift(fft(h));  
H_power = (abs(H)).^2;  
% The sequence is being propagated
\[ z = L_d; \]
\[ J = H \cdot \exp((j/2) \cdot \beta \cdot (w^2) \cdot z); \]
\[ J_{\text{power}} = \left| \text{abs}(J) \right|^2; \]
\[ h = \left| \text{abs}(\text{ifft}(J)) \right; \]
\[ h_{\text{power}} = \left| \text{abs}(h) \right|^2; \]

\% Optical Noise is added to the propagated sequence

signal_power = sum(h_power);

SNR = 14;
Res_bandwidth_f = 12.5e9;
Res_bandwidth_w = 2*pi*Res_bandwidth_f;
Pn = signal_power/(10^((SNR/10)));
delta_Pn = Pn/Res_bandwidth_w;
Pn_total = delta_Pn * deltax;

SNR_new = 10 * log10(signal_power/Pn_total);

h_noise = awgn(h,SNR_new,'measured');

\% The signal is filtered

Bo = 25e9;
[d,c] = butter(3,(2*pi*Bo/w(length(w))));
h_noise_filtered = filter(d,c,h_noise);
h_noise_power = (abs(h_noise_filtered)).^2;

Be = 7e9;
[b,a] = butter(3,(2*pi*Be/w(length(w))));
h_noise_power_filtered = filter(b,a,h_noise_power);
H_noise_power = fftshift(fft(h_noise_power));
H_noise_power_filtered = fftshift(fft(h_noise_power_filtered));

\% The bitslots are plotted on top of eachother, so an eye-pattern is \% created.

T = 1;
while h_noise_power_filtered(T) < 0.1,
    T = T + 1;
end
while h_noise_power_filtered(T) < h_noise_power_filtered(T+1),
    T = T + 1;
end

P = h_noise_power_filtered(T);

figure(1);
subplot(2,3,1)
for e = 1:Nbits,
    c = [(e-1)*Nsamples+1+T-Nsamples/2):1:e*Nsamples+T-Nsamples/2];
    if (e*Nsamples+T-Nsamples/2) > Ntotal,
        break
    end
    plot(t_bit,h_noise_power_filtered(c));
hold on;
Traineeship on Fiber-Optic Communications

```
r = T + (e-1)*Nsamples;
g = t_bit(Nsamples/2);
v = h_noise_power_filtered(r);
%plot(g,v,'rx','linewidth',2)
end
hold off;

% The I1, I0, sigma0 and signal values are calculated for a range of
% thresholds.

gemiddelde = mean(h_noise_power_filtered);
U = 0;
for q = 1:1,
treshold = gemiddelde * (0.5+(q)/100);
i_one = 0;
i_zero = 0;

for u = 1:(Nbits-2),
r = T + (u-1)*Nsamples;
    if h_noise_power_filtered(r) > treshold,
        i_one = i_one + 1;
        %h_one(i_one) = h_noise_power_filtered(r);
        for l = (r + ceil(-Nsamples*0.01)):(r+floor(Nsamples*0.01)),
            U = U + h_noise_power_filtered(l);
        end
        h_one(i_one) = U/((r+floor(Nsamples*0.01))-(r + ceil(-
            Nsamples*0.01))+1);
        U = 0;
    elseif h_noise_power_filtered(r) < treshold,
        i_zero = i_zero + 1;
        %h_zero(i_zero) = h_noise_power_filtered(r);
        for l = (r + ceil(-Nsamples*0.01)):(r+floor(Nsamples*0.01)),
            U = U + h_noise_power_filtered(l);
        end
        h_zero(i_zero) = U/((r+floor(Nsamples*0.01))-(r + ceil(-
            Nsamples*0.01))+1);
        U = 0;
    end
end
h1= mean(h_one);
h0 = mean(h_zero);

sigma1 = std(h_one)/sqrt(2);
sigma0 = std(h_zero)/sqrt(2);

Q(q) = (h1-h0)/((sigma1+sigma0));
Ber(q) = (1/2)*erfc(Q(q)/sqrt(2));
clear h_one;
clear h_zero;
end

% The minimum BER with accompanying threshold is displayed.

[B W] = min(Ber);
BER = B
Treshold = gemiddelde * (0.5+(W)/100)
```
Appendix M: Eye-pattern of an NRZ-sequence

% This script calculates and plots the eye-pattern of a NRZ-sequence. Furthermore it calculates the BER that goes with this eye-pattern.

% First a suitable time and accompanying frequency vector are created

B = 10e9;
Nsamples = 2^8;
Nbits = 512;
duty_cycle = 0.95;
beta = 20e-24;
C = 0;
m = 2;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;
t_bit = linspace(min_t_bit,max_t_bit,Nsamples);
t_bit_0th_half = linspace(min_t_bit - (dt * Nsamples/2),min_t_bit – dt,Nsamples/2);
t_bit_1st_half = linspace(min_t_bit,-dt/2,Nsamples/2);
t_bit_2nd_half = linspace(dt/2,max_t_bit,Nsamples/2);
t_bit_3rd_half = linspace(max_t_bit + dt,max_t_bit + (dt * Nsamples/2),Nsamples/2);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

Z = randsrc(1,Nbits-4,[0,1]);
E = [0 1 0 Z 0]

% The RZ-sequence is created

h_low = 0;
h_high = exp(-((1+i*C)/2).*((-dt/2./T0).^2*m)));
h_rise = exp(-((1+i*C)/2).*((t_bit_1st_half./T0).^2*m));
h_fall = exp(-((1+i*C)/2).*((t_bit_2nd_half./T0).^2*m));
h_low_after_fall = exp(-((1+i*C)/2).*((t_bit_3rd_half./T0).^2*m));
h_rise_after_low = exp(-((1+i*C)/2).*((t_bit_0th_half./T0).^2*m));

for f=1:Nbits,
    if E(f) == 0,
        q = [(f-1)*Nsamples + 1:1:Nsamples*f];
\[
\begin{align*}
    h(q) &= h_{\text{low}}; \\
    \text{elseif } E(f) &= 1, \\
    q_{1st\_half} &= [(f-1)\ast N\text{samples}+1:1:(f-1)\ast N\text{samples}+N\text{samples}/2]; \\
    \text{if } f > 1, \\
    q_{0th\_half} &= q_{1st\_half} - N\text{samples}/2; \\
    h(q_{0th\_half}) &= h_{\text{rise after low}}; \\
    h(q_{1st\_half}) &= h_{\text{rise}}; \quad \text{break}
    \end{align*}
\]

\[
\begin{align*}
    \text{end}
    \end{align*}
\]

\[
\begin{align*}
    \text{for } p &= f:(N\text{bits}-1), \\
    q_{\text{minus half}} &= [p\ast N\text{samples}-(N\text{samples}/2-1):1:p\ast N\text{samples}]; \\
    q_{\text{plus half}} &= q_{\text{minus half}} + N\text{samples}/2; \\
    \text{if } E(p) &= 0 \&\& E(p+1) == 0, \\
    h(q_{\text{plus half}}) &= h_{\text{low}}; \\
    h(q_{\text{minus half}}) &= h_{\text{low}}; \\
    \text{elseif } E(p) &= 0 \&\& E(p+1) == 1, \\
    h(q_{\text{plus half}}) &= h_{\text{rise after low}}; \\
    h(q_{\text{minus half}}) &= h_{\text{rise}}; \\
    \text{elseif } E(p) &= 1 \&\& E(p+1) == 0, \\
    h(q_{\text{plus half}}) &= h_{\text{low after fall}}; \\
    h(q_{\text{minus half}}) &= h_{\text{fall}}; \\
    \text{elseif } E(p) &= 1 \&\& E(p+1) == 1, \\
    h(q_{\text{plus half}}) &= h_{\text{high}}; \\
    h(q_{\text{minus half}}) &= h_{\text{high}};
    \end{align*}
\]

\[
\begin{align*}
    \text{end}
    \end{align*}
\]

\[
\begin{align*}
    \text{einde} &= [N\text{samples}\ast N\text{bits}-(N\text{samples}/2-1):1:N\text{samples}\ast N\text{bits}]; \\
    \text{if } E(N\text{bits}) &= 0, \\
    h(\text{einde}) &= 0; \\
    \text{elseif } E(N\text{bits}) &= 1, \\
    h(\text{einde}) &= h_{\text{fall}}; \\
    \text{end}
    \end{align*}
\]

\[
\begin{align*}
    H &= \text{fftshift}(\text{fft}(h)); \\
    H_{\text{power}} &= (\text{abs}(H))^{\ast 2}; \\
    \text{\% The sequence is being propagated}
    \end{align*}
\]

\[
\begin{align*}
    z &= 67.8; \\
    J &= H \ast \text{exp}((j/2) \ast \text{beta} \ast (w^{\ast 2}) \ast z); \\
    J_{\text{power}} &= (\text{abs}(J))^{\ast 2}; \\
    h &= \text{abs}(\text{ifft}(J)); \\
    h_{\text{power}} &= (\text{abs}(h))^{\ast 2}; \\
    \text{\% Optical Noise is added to the propagated sequence}
    \end{align*}
\]

\[
\begin{align*}
    \text{signal\_power} &= \text{sum}(h_{\text{power}}); \\
    \text{SNR} &= 14 \\
    \text{Res\_bandwidth\_f} &= 12.5e9; \\
    \text{Res\_bandwidth\_w} &= 2\ast \text{pi} \ast \text{Res\_bandwidth\_f};
    \end{align*}
\]
Pn = signal_power/(10^(SNR/10));
delta_Pn = Pn/Res_bandwidth_w;
Pn_total = delta_Pn * deltaw;

SNR_new = 10 * log10(signal_power/Pn_total);

h_noise = awgn(h,SNR_new,'measured');

% The signal is filtered
Bo = 25e9
[d,c] = butter(3,(2*pi*Bo/w(length(w))));
h_noise_filtered = filter(d,c,h_noise);

h_noise_power = (abs(h_noise_filtered)).^2;

Be = 8e9
[b,a] = butter(3,(2*pi*Be/w(length(w))));
h_noise_power_filtered = filter(b,a,h_noise_power);

H_noise_power = fftshift(fft(h_noise_power));
H_noise_power_filtered = fftshift(fft(h_noise_power_filtered));

% The bitslots are plotted on top of eachother, so an eye-pattern is % created.
T = 1;
while h_noise_power_filtered(T) < 0.1,
    T = T + 1;
end
while h_noise_power_filtered(T) < h_noise_power_filtered(T+1),
    T = T + 1;
end

P = h_noise_power_filtered(T);

figure(5);
subplot(2,3,6);
for e = 1:Nbits,
    c = [((e-1)*Nsamples+1+T-Nsamples/2):1:e*Nsamples+T-Nsamples/2];
    if (e*Nsamples+T-Nsamples/2) > Ntotal,
        break
    end
    plot(t_bit,h_noise_power_filtered(c));
    hold on;
    r = T + (e-1)*Nsamples;
    g = t_bit(Nsamples/2);
    v = h_noise_power_filtered(r);
    %plot(g,v,'rx','linewidth',2)
end
hold off;

% The I1, I0, sigma0 and signal values are calculated for a range of % thresholds.
gemiddelde = mean(h_noise_power_filtered);
U = 0;
for q = 1:1,
    treshold = gemiddelde * (0.5+(q)/100);
i_one = 0;
i_zero = 0;
    for u = 1:(Nbits-2),
        r = T + (u-1)*Nsamples;
            if h_noise_power_filtered(r) > treshold,
                i_one = i_one + 1;
                    #h_one(i_one) = h_noise_power_filtered(r);
                    for l = (r + ceil(-Nsamples*0.01)):r+floor(Nsamples*0.01)),
                        U = U + h_noise_power_filtered(l);  
                    end
                h_one(i_one) = U/((r+floor(Nsamples*0.01))-(r + ceil(-
                        Nsamples*0.01))+1);
                U = 0;
            elseif h_noise_power_filtered(r) < treshold,
                i_zero = i_zero + 1;
                    #h_zero(i_zero) = h_noise_power_filtered(r);
                    for l = (r + ceil(-Nsamples*0.01)):r+floor(Nsamples*0.01)),
                        U = U + h_noise_power_filtered(l);  
                    end
                h_zero(i_zero) = U/((r+floor(Nsamples*0.01))-(r + ceil(-
                        Nsamples*0.01))+1);
                U = 0;
            end
            end
    end
    total_bits = i_zero + i_one;
    hl = mean(h_one);
    h0 = mean(h_zero);
    sigma1 = std(h_one)/sqrt(2);
    sigma0 = std(h_zero)/sqrt(2);
    Q(q) = (hl-h0)/((sigma1+sigma0));
    Ber(q) = (1/2)*erfc(Q(q)/sqrt(2));
    clear h_one;
    clear h_zero;
end
% The minimum BER with accompanying threshold is displayed.
[B W] = min(Ber);
BER = B
Treshold = gemiddelde * (0.5+(W)/100)
Appendix N: BER vs. OSNR for RZ-sequence

% This script calculates the BER for various OSNR's and propagation
% lengths of a RZ-sequence.

% First a suitable time and accompanying frequency vector are created

B = 10e9;
Nsamples = 2^8;
Nbits = 512;
duty_cycle = 0.30;
beta = 20e-24;
C = 0;
m = 1;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;

t_bit = linspace(min_t_bit,max_t_bit,Nsamples);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

Z = randsrc(1,Nbits-4,[0,1]);
E = [0 1 0 Z 0]

% The RZ-sequence is created

g = [1:1:Nsamples];
for p=1:Nbits,
    if E(p) == 1;
        h(q) = exp(-(1/2).*((t_bit./T0).^((2*m))));
    else
        h(q) = 0;
    end
    q = q + Nsamples;
end

H = fftshift(fft(h));
H_power = (abs(H)).^2;
% The sequence is being propagated, noise is added and the minimum BER % is calculated

for e = 1:4,
    z = (e-1)*Ld
    J = H .* exp((j/2) .* beta .* (w.^2) .* z);
    J_power = (abs(J)).^2;
    h  = abs(ifft(J));
    h_power = (abs(h)).^2;
    signal_power = sum(h_power);
    Bo = 25e9;
    [d,c] = butter(3,(2*pi*Bo/w(length(w))));
    Be = 7e9;
    [b,a] = butter(3,(2*pi*Be/w(length(w))));
    for k = 1:19,
        SNR = 11.5 + k*0.5
        Signal_to_noise_ratio(k,e) = SNR-3;
        Res_bandwidth_f = 12.5e9;
        Res_bandwidth_w = 2*pi*Res_bandwidth_f;
        Pn = signal_power/(10^(SNR/10));
        delta_Pn = Pn/Res_bandwidth_w;
        Pn_total = delta_Pn * deltaw;
        SNR_new = 10 * log10(signal_power/Pn_total);
        h_noise = awgn(h,SNR_new,'measured');
        h_noise_filtered = filter(d,c,h_noise);
        h_noise_power = (abs(h_noise_filtered)).^2;
        h_noise_power_filtered = filter(b,a,h_noise_power);
        H_noise_power = fftshift(fft(h_noise_power));
        H_noise_power_filtered = fftshift(fft(h_noise_power_filtered));
        T = 1;
        while h_noise_power_filtered(T) < 0.1,
            T = T + 1;
        end
        while h_noise_power_filtered(T) < h_noise_power_filtered(T+1),
            T = T + 1;
        end
    end
    P = h_noise_power_filtered(T);
    gemiddelde = mean(h_noise_power_filtered);
    U = 0;
    for q = 1:100,
        treshold = gemiddelde * (0.5+(q)/100);
i_one = 0;
i_zero = 0;

for u = 1:(Nbits-2),
r = T + (u-1)*Nsamples;
    if h_noise_power_filtered(r) > treshold,
        i_one = i_one + 1;
        h_one(i_one) = h_noise_power_filtered(r);
        for l = (r + ceil(-Nsamples*0.01)):((r+floor(Nsamples*0.01))),
            U = U + h_noise_power_filtered(l);
        end
        h_one(i_one) = U/((r+floor(Nsamples*0.01))-(r +
            ceil(-Nsamples*0.01))+1);
        U = 0;
    end
    elseif h_noise_power_filtered(r) < treshold,
        i_zero = i_zero + 1;
        h_zero(i_zero) = h_noise_power_filtered(r);
        for l = (r + ceil(-Nsamples*0.01)):((r+floor(Nsamples*0.01))),
            U = U + h_noise_power_filtered(l);
        end
        h_zero(i_zero) = U/((r+floor(Nsamples*0.01))-(r +
            ceil(-Nsamples*0.01))+1);
        U = 0;
    end
end

total_bits = i_zero + i_one;

h1 = mean(h_one);
h0 = mean(h_zero);
sigma1 = std(h_one)/sqrt(2);
sigma0 = std(h_zero)/sqrt(2);

Q(q) = (h1-h0)/((sigma1+sigma0));
Ber(q) = (1/2)*erfc(Q(q)/sqrt(2));
clear h_one;
clear h_zero;
end

[B1 W] = min(Ber);

BER(k,e) = B1;
Treshold(k,e) = gemiddelde * (0.5+(W)/100);
end

% The results are stored so the can be plotted in Microcal Origin

Matrix = transpose([Signal_to_noise_ratio(:,1) BER]);

fid = fopen('BER_Gaussian','w');
fprintf(fid,'%6.5e	 %6.5e	 %6.5e	 %6.5e	 %6.5e
',Matrix);
fclose(fid);
Appendix O: BER vs. OSNR for NRZ-sequence

% This script calculates the BER for various OSNR's and propagation
% length of a NRZ-sequence.

% First a suitable time and accompanying frequency vector are created

B = 10e9;
Nsamples = 2^8;
Nbits = 512;
duty_cycle = 0.95;
beta = 20e-24;
C = 0;
m = 2;

T0 = (duty_cycle/(B*2));
Ld = (T0.^2)/beta;
sigma_nul = T0/sqrt(2);
T = 1/B;
Ntotal = Nsamples * Nbits;
dt = T / Nsamples;
deltaT = dt * Nsamples * Nbits;

max_t_bit = (dt * Nsamples/2)-dt/2;
min_t_bit = -(dt * Nsamples/2)+dt/2;
t_bit = linspace(min_t_bit,max_t_bit,Nsamples);

t_bit_0th_half = linspace(min_t_bit - (dt * Nsamples/2),min_t_bit -
dt,Nsamples/2);
t_bit_1st_half = linspace(min_t_bit,-dt/2,Nsamples/2);
t_bit_2nd_half = linspace(dt/2,max_t_bit,Nsamples/2);
t_bit_3rd_half = linspace(max_t_bit + dt,max_t_bit + (dt * Nsamples/2),
Nsamples/2);
t_total = linspace(dt/2,(deltaT-dt/2),Ntotal);

dw = 2*pi / deltaT;
deltaw = 2*pi / dt;
maxw = dw * (Ntotal/2-1);
minw = -dw * Ntotal/2;
w = linspace(minw,maxw,Ntotal);

Z = randsrc(1,Nbits-4,[0,1]);
E = [0 1 0 0 Z 0]

% The NRZ-sequence is created

h_low = 0;
h_high = exp(-(1+i*C)/2).*((-dt/2./T0).^((2*m)));
h_rise = exp(-(1+i*C)/2).*((t_bit_1st_half./T0).^((2*m)));
h_fall = exp(-(1+i*C)/2).*((t_bit_2nd_half./T0).^((2*m)));
h_low_after_fall = exp(-(1+i*C)/2).*((t_bit_3rd_half./T0).^((2*m)));
h_rise_after_low = exp(-(1+i*C)/2).*((t_bit_0th_half./T0).^((2*m)));

for f=1:Nbits,
    if E(f) == 0,
q = [(f-1)*Nsamples + 1:1:Nsamples*f];
h(q) = h_low;
elseif E(f) == 1,
    q_1st_half = [(f-1)*Nsamples+1:1:(f-1)*Nsamples+Nsamples/2];
    if f > 1,
        q_0th_half = q_1st_half - Nsamples/2;
        h(q_0th_half) = h_rise_after_low;
    end
    h(q_1st_half) = h_rise;
    break
end
end
for p=f:(Nbits-1),
    q_minus_half = [p*Nsamples-(Nsamples/2-1):1:p*Nsamples];
    q_plus_half = q_minus_half + Nsamples/2;
    if E(p) == 0 && E(p+1) == 0,
        h(q_plus_half) = h_low;
        h(q_minus_half) = h_low;
    elseif E(p) == 0 && E(p+1) == 1,
        h(q_plus_half) = h_rise;
        h(q_minus_half) = h_rise_after_low;
    elseif E(p) == 1 && E(p+1) == 0,
        h(q_plus_half) = h_low_after_fall;
        h(q_minus_half) = h_fall;
    elseif E(p) == 1 && E(p+1) == 1,
        h(q_plus_half) = h_high;
        h(q_minus_half) = h_high;
    end
end
einde = [Nsamples*Nbits-(Nsamples/2 -1):1:Nsamples*Nbits];
if E(Nbits) == 0,
    h(einde) = 0;
elseif E(Nbits) == 1,
    h(einde) = h_fall;
end
H = fftshift(fft(h));
H_power = (abs(H)).^2;
% The sequence is being propagated, noise is added and the minimum BER % is calculated
for e = 1:4,
    z = (e-1)*33
    J = H .* exp((j/2) .* beta .* (w.^2) .* z);
    J_power = (abs(J)).^2;
    h  = abs(ifft(J));
    h_power = (abs(h)).^2;
    signal_power = sum(h_power);
    Bo = 25e9;
    [d,c] = butter(3,(2*pi*Bo/w(length(w))));
Be = 8e9;
[b,a] = butter(3,(2*pi*Be/w(length(w))));

for k = 1:19,

SNR = 11.5 + k*0.5
Signal_to_noise_ratio(k,e) = SNR-3;
Res_bandwidth_f = 12.5e9;
Res_bandwidth_w = 2*pi*Res_bandwidth_f;
Pn = signal_power/(10^(SNR/10));
delta_Pn = Pn/Res_bandwidth_w;
Pn_total = delta_Pn * deltaw;

SNR_new = 10 * log10(signal_power/Pn_total);

h_noise = awgn(h,SNR_new,'measured');

h_noise_filtered = filter(d,c,h_noise);

h_noise_power = (abs(h_noise_filtered)).^2;

h_noise_power_filtered = filter(b,a,h_noise_power);

H_noise_power = fftshift(fft(h_noise_power));
H_noise_power_filtered = fftshift(fft(h_noise_power_filtered));

T = 1;
while h_noise_power_filtered(T) < 0.1,
    T = T + 1;
end

while h_noise_power_filtered(T) < h_noise_power_filtered(T+1),
    T = T + 1;
end

P = h_noise_power_filtered(T);

gemiddelde = mean(h_noise_power_filtered);
U = 0;

for q = 1:1,
    treshold = gemiddelde * (0.5+(q)/100);
i_one = 0;
i_zero = 0;

    for u = 1:(Nbits-2),
        r = T + (u-1)*Nsamples;
        if h_noise_power_filtered(r) > treshold,
            i_one = i_one + 1;
            h_one(i_one) = h_noise_power_filtered(r);
            for l = (r + ceil(-Nsamples*0.01)): (r+floor(Nsamples*0.01)),
                U = U + h_noise_power_filtered(l);
            end
            i_one = U/((r+floor(Nsamples*0.01))-(r + ceil(-Nsamples*0.01))+1);
            U = 0;
        elseif h_noise_power_filtered(r) < treshold,
i_zero = i_zero + 1;
%h_zero(i_zero) = h_noise_power_filtered(r);
for l = (r + ceil(-
    Nsamples*0.01)): (r+floor(Nsamples*0.01)),
    U = U + h_noise_power_filtered(l);
end
h_zero(i_zero) = U/((r+floor(Nsamples*0.01))-(r +
    ceil(-Nsamples*0.01)))+1;
U = 0;
end
end

total_bits = i_zero + i_one;

h1 = mean(h_one);
h0 = mean(h_zero);
sigma1 = std(h_one)/sqrt(2);
sigma0 = std(h_zero)/sqrt(2);

Q(q) = (h1-h0)/((sigma1+sigma0));
Ber(q) = (1/2)*erfc(Q(q)/sqrt(2));
clear h_one;
clear h_zero;
end

[B W] = min(Ber);

BER(k,e) = B;
Treshold(k,e) = gemiddelde * (0.5+(W)/100);
end
end

% The results are stored so the can be plotted in Microcal Origin

Matrix = transpose([Signal_to_noise_ratio(:,1) BER]);

fid = fopen('BER_Super_Gaussian_m2','w');
fprintf(fid,'%6.5e\t %6.5e\t %6.5e\t %6.5e\t %6.5e\n',Matrix);
fclose(fid);
Appendix P: Software

- Matlab 6.5
- Microcal Origin 6.0
- Microsoft Office Word 2002
- Microsoft Office Excel 2002
References


Suggestions for Further Reading


