Simulation of the Propagation of Terahertz Waves in Linear & Nonlinear Isotropic Media

A Major Qualifying Project Report

Submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Bachelor of Science

By

Miguel A. Aranda R.

April 26, 2018

Approved by:

Professor Alex Zozulya, Advisor

Physics Department

Professor Reinhold Ludwig, Advisor

Electrical & Computer Engineering Department

Professor Lyubov Titova, Co-Advisor

Physics Department

Abstract

Terahertz (THz) spectroscopy offers the possibility to study and characterize the dielectric response of materials. Novel developments in the generation of THz pulses allow the creation of high peak electric fields which enable the observation of nonlinear effects. This MQP deals with the development of MATLAB code using the split-step method to simulate the propagation of THz pulses in linear and nonlinear isotropic media, more specifically, liquids. Programs were developed to determine the dielectric properties of a sample through the analysis of a pulse propagated through the sample, and to calculate threshold electric field values for observing nonlinearities. The code was tested with experimental data of propagation through water samples.

Acknowledgements

I would like to acknowledge and thank my advisors, Professors Alex Zozulya, Reinhold Ludwig and Lyubov Titova, for their help and guidance throughout the completion of this Major Qualifying Project. I also want to thank Kateryna Kushnir for her feedback and support.

Table of Contents

Introduction

The study of terahertz (THz) waves and their interaction with matter has become a very productive scientific field, contributing advances in many fields. Ruben's experiments on blackbody emission (at about 6 THz) at the start of the twentieth century led to Planck's blackbody radiation law, and subsequently to the formulation of quantum mechanics. Nevertheless, progress in the field was delayed until the 1970s, when researchers discovered the use of THz frequencies for astronomical research [1]. Since then, discoveries, inventions and advances have been compounding to increase the depth and breadth of THz analysis.

The goal of this project is the modelling of the propagation of THz waves in linear and nonlinear isotropic media and use this ability to compare it to experimental results, analyze data, and obtain a deeper understanding of the phenomena observed in the lab. This can be achieved through computer modelling, and mores specifically, the use of the split-step method for nonlinearities. For this implementation, the development of an appropriate model requires an understanding of the experimental setup used in the laboratory, the pieces of software used, in particular Matlab, the underlying mathematics involved in the program (partial differential equations, Fourier transforms, and various discretization issues), and the interaction between these components.

The terahertz range

THz waves are electromagnetic radiation between the milimeter wave and infrared portions of the EM spectrum, and ranges between 0.3THz-20THz (10^{12} Hz), or the corresponding wavelengths of $1\ \text{mm} - 15\ \mu\text{m}$ and energies of $1.2\ -\ 8.3\ \text{meV}$, as seen in figure 1. The use of THz provides a high bandwidth and good resolution while remaining non-invasive and non-destructive. THz analysis has become the method of choice for study in various fields. In material science and manufacturing it is used for property identification and defect detection [2]. In medicine, for non-invasive imaging [3] (even 3D imaging). For security purposes, THz are particularly useful, since many of the materials used in weapon manufacturing have a characteristic, identifiable resonance spectrum in the THz domain which can be taken advantage of for concealed weapon detection [4]. In the field of museum conservation, THz waves are used

to analyze the making of objects of interest without having to break them apart, and has provided a new method of studying painting techniques and restoration efforts [5]. THz waves are also being studied as an alternative to current communication frequencies, to provide higher bandwidths [6]. In particular, at WPI's Ultrafast THz and Optical Spectroscopy Lab, THz waves are used to study nanomaterials for photovoltaic applications, the role of intermolecular dynamics in chemical reactions, and the properties of cells [7].

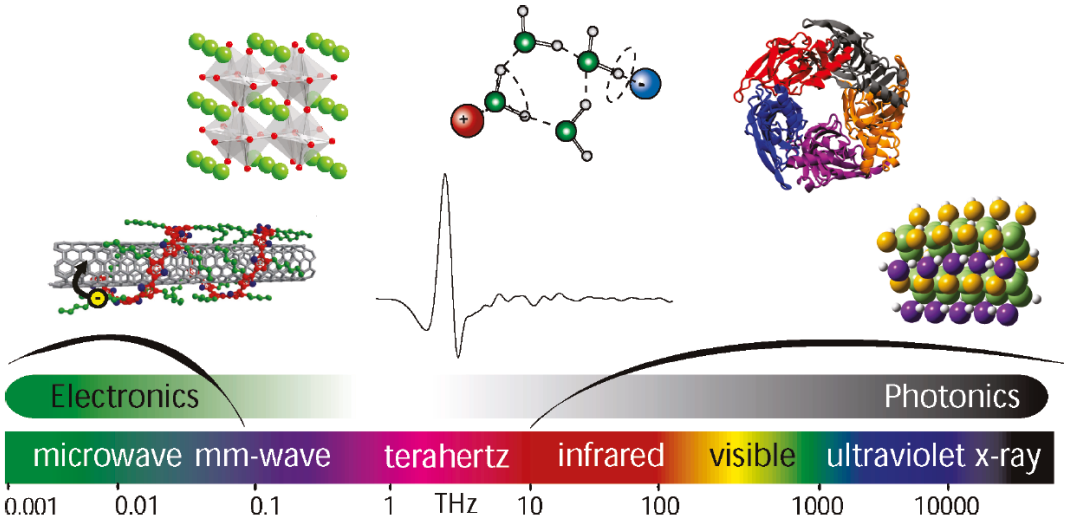

Figure 1: EM spectrum highlighting the phenomena analyzed with THz (semiconductors, nanomaterials, modes of liquids and proteins) from [8].

Before its recent popularity, the use of THz radiation was unavailable to researchers, since it was beyond what was accessible for electronic circuity and below what could be accessed optically [8]. However, THz radiation can now be generated through a variety of methods, most notably optical rectification and electronic oscillators. Optical rectification is a nonlinear process in which a polarization is generated in a nonlinear medium through the use of an intense optical beam, and is the method capable of working at the highest efficiency and generating the strongest pulses [9]. Thanks to this technology it is possible to generate short, nearly single-cycle THz pulses using ultrafast optical and infrared lasers, most notably Ti:Sapphire lasers. Other systems include synchrotrons, quantum cascade lasers and free electron lasers [8]. The short duration and broad bandwidth of THz pulses lend themselves to application in spectroscopy. There are various specific spectroscopic techniques involving THz, some of which are time-domain spectroscopy (TDS), time-resolved spectroscopy (TRTS), emission spectroscopy (TES) and imaging. These distinct implementations allow the investigation of different features of interest. For instance, TDS

is used to determine the static properties of a sample (like complex permittivity), while TRTS helps to analyze dynamic properties of a sample (which is of use in semiconductor physics). A benefit of the use of THz is that coherent time domain detection allows researchers to determine the amplitude and phase of the pulse, and can be used to compute the complex frequency-resolved permittivity of the sample without Kramers-Kronig analysis [1]. A sample permittivity spectrum is shown in figure 2, which also outlines processes observed at each different section of the electromagnetic spectrum with atomic resonances characterizing the THz gap.

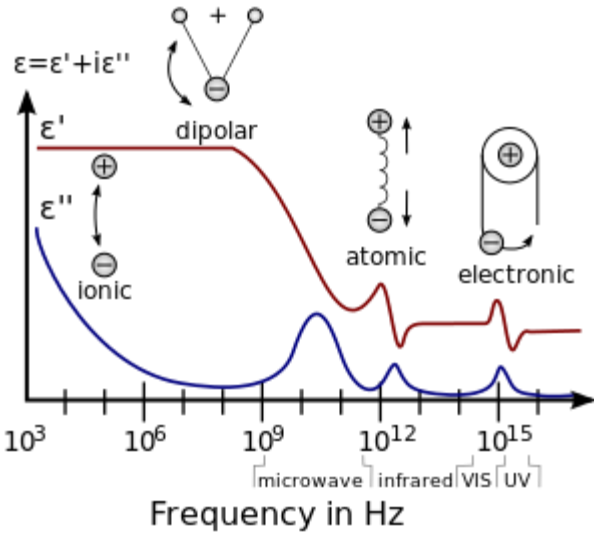

Figure 2: Atomic interactions and frequency (borrowed from [10])

In typical THz spectroscopy experiments (as seen in figure 3) Ti:sapphire laser pulses traverse the sample and are then measured in the time domain. THz-TDS can be used for the analysis of solids, gases, and liquids, though this project focuses on the latter. For liquids, dielectric properties are determined by the formation, interaction and relaxation of dipoles [8], and water is of paramount importance to this area. Water has high absorption in the THz range, with an absorption coefficient of about $250\ \mathrm{cm}^{-1}$ at 1 THz [11]. This property leads to many practical applications, such as quantifying hydration in medicine or the analysis of dielectric properties of solutions. However, penetration into biological tissues is limited to a superficial layer of skin, and the propagation through air is hindered by the presence of water molecules in air.

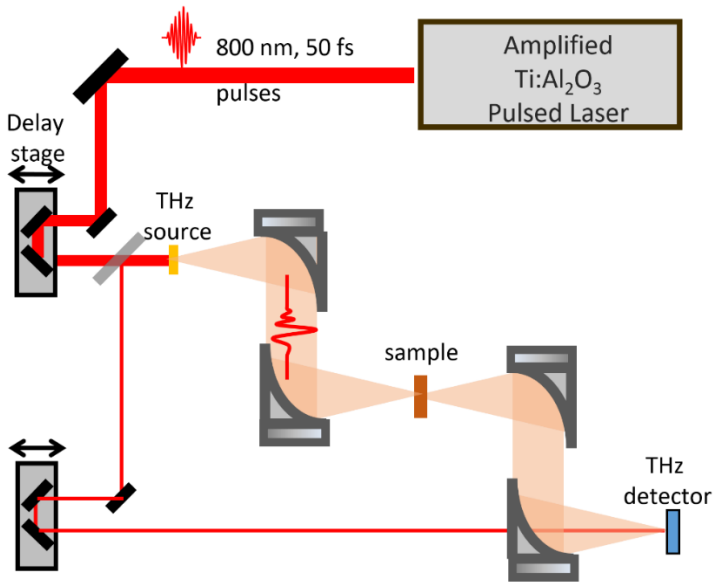

Figure 3: A THz-TDS experimental setup using a broadband pulsed laser.

Motivation

This project has two main aims. First, to simulate linear propagation through liquids, to model results and extract their complex dielectric functions. Second, to simulate nonlinear effects for high peak electric fields, which will become more relevant through the use of new THz sources with peak field values of MV/cm. This will provide insight into the new phenomena that can be observed. The resulting simulation can be used to estimate the threshold electric field value for which the effects become pronounced.

The study of THz spectroscopy and nonlinearities has many interesting applications. The THz range is coincident with the frequency of many important vibrational modes, as seen in figure 4. 2D materials can be used for integrated photonic circuits operating at THz wavelengths, including the possibility of developing graphene based emitters, detectors, and modulators [12]. THz imaging can be used to investigate subsurface damages in solar cells [13]. In the realm of biotechnology, THz radiation can be used to recognize and characterize biomolecules since it coincides with the low-frequency vibration, rotation, and translation biological molecules [14]. In particular, for liquids, THz can be used to detect liquid explosives [15], improving airport security, to quantify refined oil mixtures [16], and multiple biological uses, such as drug analysis and

characterization and skin cancer imaging, amongst others [17]. These and many others show the potential that THz has as an area of interdisciplinary research and applications.

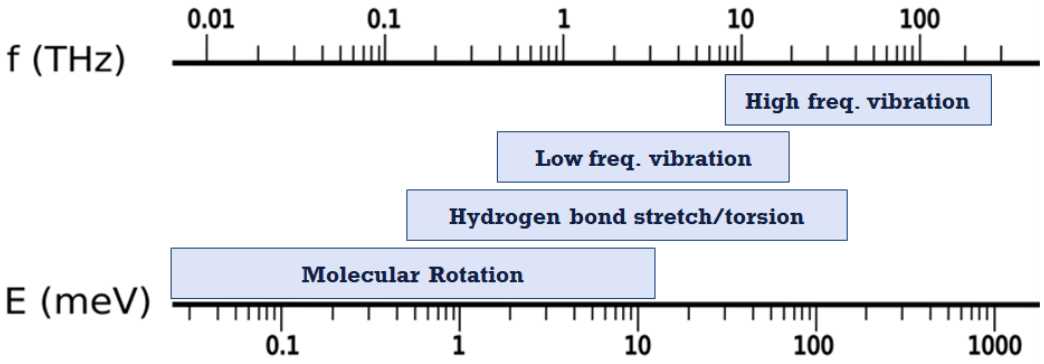

Figure 4: The THZ spectrum and molecular modes, from [18].

Propagation of electromagnetic radiation in dielectric media

The analysis of the phenomena observed in the lab requires the study of non-linear optics, the development of a pulse propagation model, an algorithm to implement the model in a computer, and a model to describe material properties in the THz domain. The mathematical formulation of the aforementioned aspects is outlined and discussed in the following sections.

Pulse propagation model

The behavior of electromagnetic radiation is governed, at its most fundamental level, by Maxwell's Equations [19] (1-4):

$$\nabla \cdot \boldsymbol{D} = \rho_f \tag{1}$$

$$\nabla \cdot \boldsymbol{B} = 0 \tag{2}$$

$$\nabla \times \boldsymbol{E} = -\frac{\partial \boldsymbol{B}}{\partial t} \tag{3}$$

$$\nabla \times \boldsymbol{H} = \boldsymbol{J}_f + \frac{\partial \boldsymbol{D}}{\partial t} \tag{4}$$

where ρ_f is the free charge density and $\boldsymbol{J}_f$ is the free current density, $\boldsymbol{E}$ is the electric field, $\boldsymbol{D}$ is the electric displacement field, $\boldsymbol{B}$ and $\boldsymbol{H}$ are the magnetic fields. Fields $\boldsymbol{D}$ and $\boldsymbol{H}$ are defined through the constitutive relations:

$$\boldsymbol{D} \equiv \varepsilon_0 \boldsymbol{E} + \boldsymbol{P} \tag{5}$$

$$\boldsymbol{H} \equiv \frac{1}{\mu_0} \boldsymbol{B} - \boldsymbol{M} \tag{6}$$

here ε_0 and μ_0 are the permittivity and permeability of free space ($\varepsilon_0 \mu_0 = c^{-2}$, where c is the speed of light in vacuum), while $\boldsymbol{P}$ and $\boldsymbol{M}$ are polarization and magnetization, respectively.

In the following we will be working with dielectric ($\rho_f = 0, \boldsymbol{J}_f = 0$) and nonmagnetic ($\boldsymbol{M} = 0$) media. We can obtain a single equation for the propagation of the electric field $\boldsymbol{E}$ by taking the curl of eq. (3):

$$\nabla \times \nabla \times \boldsymbol{E} = -\frac{\partial}{\partial t} \nabla \times \boldsymbol{B} \tag{7}$$

To transform eq. (7), we first use $\boldsymbol{M} = 0$ and eq. (6), to set

$$\boldsymbol{B} = \mu_0 \boldsymbol{H}$$

Then, given that $\boldsymbol{J}_f = 0$, eq. (4), becomes

$$\nabla \times \boldsymbol{B} = \mu_0 \frac{\partial \boldsymbol{D}}{\partial t}$$

which is now used to get

$$\nabla^2 \boldsymbol{E} - \nabla(\nabla \cdot \boldsymbol{E}) = -\mu_0 \frac{\partial^2 \boldsymbol{D}}{\partial t^2} \tag{8}$$

Furthermore, eq. (6) can be used to replace the electric displacement field in favor of $\boldsymbol{E}$ and $\boldsymbol{P}$:

$$\nabla^2 \boldsymbol{E} - \nabla(\nabla \cdot \boldsymbol{E}) = \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \left(\boldsymbol{E} + \frac{\boldsymbol{P}}{\varepsilon_0} \right) \tag{9}$$

In an isotropic linear medium, $\nabla \cdot \boldsymbol{E} = 0$. In the presence of nonlinearity and transverse spatial effects, this is not necessarily the case. Nevertheless, in the following we will assume that the transverse effects are sufficiently small and set $\nabla \cdot \boldsymbol{E} = 0$. Additionally, we will assume that the electromagnetic field is polarized in the same direction and drop vector notation.

$$\nabla^2 E = \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \left(E + \frac{P}{\varepsilon_0} \right) \tag{10}$$

In general, the jth component of the polarization field is related to the electric field as

$$\frac{1}{\varepsilon_0} P_j = \int_{-\infty}^{t} \chi_{jk}^{(1)}(t-\tau) E_k(\tau) d\tau +$$

$$\int_{-\infty}^{t} \chi_{jkl}^{(2)}(t-\tau_1, t-\tau_2) E_k(\tau_1) E_l(\tau_2) d\tau_1 d\tau_2 + \tag{11}$$

$$\int_{-\infty}^{t} \chi_{jklm}^{(3)}(t-\tau_1, t-\tau_2, t-\tau_3) E_k(\tau_1) E_l(\tau_2) E_m(\tau_3) d\tau_1 d\tau_2 d\tau_3 + \cdots$$

where $\chi_{jk}^{(1)}$ is the linear susceptibility and $\chi^{(n)}$ is the nth order nonlinear susceptibility. We will consider the simplest case of interest, where the optical medium is centrosymmetric and isotropic. In this case, $\chi^{(2)} = 0$, and eq. (4) can be simplified to

$$\frac{1}{\varepsilon_0}P = \int_{-\infty}^{t} \chi^{(1)}(t-\tau)E(\tau)d\tau +$$

$$\int_{-\infty}^{t} \chi^{(3)}(t-\tau_1, t-\tau_2, t-\tau_3)E(\tau_1)E(\tau_2)E(\tau_3)d\tau_1 d\tau_2 d\tau_3 \tag{12}$$

$$= \frac{1}{\varepsilon_0}(P_L + P_{NL})$$

where we have separated the polarization into linear and nonlinear parts.

The Fourier transform of the electric field $E(t)$ defined by the following relations:

$$E(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} E(t)e^{i\omega t}dt,$$
$$E(t) = \int_{-\infty}^{\infty} E(\omega)e^{-i\omega t}d\omega \tag{13}$$

Fourier transforms of $D(\omega)$ and $P_{NL}(\omega)$ in (7) are defined analogously.

By applying the transform to eq. (5), we arrive at:

$$D(\omega) = \varepsilon(\omega)E(\omega) + P_{NL}(\omega) \tag{14}$$

The quantity $\varepsilon(\omega)$ in (6) is the linear dielectric constant of the medium, defined by the relations

$$\varepsilon(\omega) = 1 + \chi^{(1)}(\omega) \tag{15}$$

where

$$\chi^{(1)}(\omega) = \int_{0}^{\infty} \chi^{(1)}(\omega)e^{i\omega t}dt$$

is the Fourier transform of the linear susceptibility.

In the following we assume that the electric field $E(t)$ can be represented as

$$E(t) = e(t)e^{-i\omega_0 t} + c.c. \tag{16}$$

where $c.c.$ stands for the complex conjugate and $e(t)$ is a complex amplitude that changes slowly as compared to $\exp(-i\omega_0 t)$:

$$\left|\frac{de}{dt}\right| \ll \omega_0|e| \tag{17}$$

Similarly, we write

$$P_{NL}(t) = p_{NL}(t)e^{-i\omega_0 t} + c.c \tag{18}$$

Assumption (16) means that $E(\omega)$ and $P_{NL}(\omega)$ are both localized around $\omega = \pm\omega_0$.

Fourier transform of eq. (10) reads

$$\nabla^2 E(\omega) = \frac{\omega^2}{c^2}[\varepsilon(\omega)E(\omega) + P_{NL}(\omega)] \tag{19}$$

Inverse Fourier transform of eq. (19) allows one to obtain an equation for the slowly varying envelope $e(t)$ given by eq. (16).

We can represent $k^2(\omega) = \left(\frac{\omega}{c}\right)^2 \varepsilon(\omega)$ as a power series around $\omega = \omega_0$:

$$k^2(\omega) = \sum_{m=0}^{\infty} \frac{1}{m!}\frac{d^m k^2}{d\omega^m}\bigg|_{\omega_0} (\omega - \omega_0)^m$$

This allows us to get (see [20]):

$$\nabla^2 e + \left[\sum_{m=0}^{\infty} \frac{i^m}{m!}\frac{d^m k^2}{d\omega^m}\bigg|_{\omega_0} \frac{\partial^m}{\partial t^m}\right] e + \frac{\omega_0}{c^2\varepsilon_0}\left(1 + \frac{i}{\omega_0}\frac{\partial}{\partial t}\right) p_{NL} = 0 \tag{20}$$

Further representing the envelope $e(t)$ as

$$e(r,t) = A(r,t)\exp[ik(\omega_0)z] \tag{21}$$

where $k(\omega) = \frac{\omega}{c}\sqrt{\varepsilon(\omega)}$, substituting it into eq. (14) and dropping the subscript 0 on ω_0 allows us to get

$$2ik\left(\frac{\partial}{\partial z} + k'\frac{\partial}{\partial t}\right)A + \frac{\partial^2}{\partial z^2}A + \nabla_\perp^2 A \tag{22}$$

$$+\left[\sum_{m=2}^{\infty}\frac{i^m}{m!}\frac{d^m k^2}{d\omega^m}\bigg|_{\omega_0}\frac{\partial^m}{\partial t^m}\right]A+\frac{\omega^2}{c^2\varepsilon_0}\left(1+\frac{i}{\omega}\frac{\partial}{\partial t}\right)^2 p_{NL}=0$$

where $k'=dk(\omega)/d\omega$.

In the following we will neglect the second derivative with respect to z in eq. (22) assuming that

$$\left|\frac{\partial}{\partial z}A\right| \ll |k||A|.$$

We will also omit the transverse Laplacian $\nabla_\perp^2$ assuming that the electromagnetic field is sufficiently wade in the transverse dimension. We will also introduce the following notation for the Taylor expansion of k:

$$\beta_m=\mathrm{Re}\left[\left(\frac{\partial k(\omega)}{\partial\omega}\right)_{\omega_0}\right]$$

$$\alpha_m=\mathrm{Im}\left[\left(\frac{\partial k(\omega)}{\partial\omega}\right)_{\omega_0}\right]$$

$$\widehat{D}=i\frac{\alpha_0}{2}-\frac{\alpha_1}{2}\partial_t+\sum_{m=2}^{\infty}\frac{\beta_m+\frac{i\alpha_m}{2}}{m!}(i\partial_t)^m$$

This reduces eq. (22) to

$$\partial_z A=i\widehat{D}A+\frac{\omega^2}{c^2\varepsilon_0}\left(1+\frac{i}{\omega}\frac{\partial}{\partial t}\right)^2 p_{NL} \tag{23}$$

Furthermore, we will only consider the lowest order and thus simpler contribution to p_{NL}. We know that polarization is related to the electric field in the third order (eq. (12)), so we will combine its coefficients into a single nonlinear parameter γ as follows:

$$\partial_z A=i\widehat{D}A+i\gamma|A|^2A \tag{24}$$

Finally, for most simulations we do not need to consider every term in the Taylor expansion of k, so we will limit ourselves to three, which makes the final version of eq. (24) become:

$$\partial_z A=\left(-\frac{\alpha_0}{2}-i\frac{\alpha_1}{2}\partial_t+\left(-i\frac{\beta_2}{2}+\frac{\alpha_2}{4}\right)(\partial_t)^2+\left(\frac{\beta_3}{3!}+i\frac{\alpha_3}{2*3!}\right)(\partial_t)^3+i\gamma|A|^2\right)A \tag{25}$$

The result of the propagated envelope can be transformed back into the oscillating electric field by multiplying the envelope by $e^{-i(\omega_0 t)}$. Furthermore, we are working in a reference frame moving at the speed of the wave, so we use $\tau = t + \beta_1 z$ to get back to static reference frame and account for the different propagation time inside the medium.

Kerr effect

The main source of nonlinear effects observed in THz regime experiments is termed the Kerr effect, a change in the refractive index of a medium by an amount proportional to $|\boldsymbol{E}|^2$. First of all, the complex refractive index of a medium, n, is related to the complex dielectric function by the following relations

$$\begin{gathered} \varepsilon = n^2 \\ \mathrm{Re}[\varepsilon] = \mathrm{Re}[n]^2 - \mathrm{Im}[n]^2 \\ \mathrm{Im}[\varepsilon] = 2\mathrm{Re}[n]\mathrm{Im}[n] \\ \mathrm{Re}[n] = \sqrt{\frac{|\varepsilon| + \mathrm{Re}[\varepsilon]}{2}} \\ \mathrm{Im}[n(\omega)] = \sqrt{\frac{|\varepsilon| - \mathrm{Re}[\varepsilon]}{2}} \end{gathered} \tag{26}$$

Recall, from eq. (14) the displacement field is separated using the nonlinear polarization:

$$D = \varepsilon E + P_{NL} \tag{27}$$

The ε in eq. (27) can be now rewritten in terms of electric susceptibilities:

$$\varepsilon = 1 + 4\pi\chi^{(1)} \tag{28}$$

where $\chi^{(1)}$ is the linear electric susceptibility. Likewise, the nonlinear polarization can also be expressed using electric susceptibilities:

$$\begin{gathered} P_{NL} = \varepsilon_0 \chi_{nl} E \\ P_{NL} = \varepsilon_0 (\chi^{(1)} E + \chi^{(2)} E^2 + \chi^{(3)} E^3 + \cdots) \end{gathered} \tag{29}$$

where $\chi^{(n)}$ is called the nth order susceptibility. These values can be calculated by analyzing the potential well produced by the electric field. The optical Kerr effect causes the refractive index of a medium to become dependent on beam intensity. To express this, consider the nonlinear dielectric constant of a medium in which $\chi^{(2)} = 0$:

$$\varepsilon_{r_{nl}} = 1 + \chi^{(1)} + \chi^{(3)}E^2 = \varepsilon_r + \Delta\varepsilon \tag{30}$$

from eq. (30), and by using eq. (26), n must be:

$$n = \sqrt{\varepsilon_r + \Delta\varepsilon} = \sqrt{\varepsilon_r} + \frac{\Delta\varepsilon}{2\sqrt{\varepsilon_r}} = n_0 + \frac{\chi_3|\boldsymbol{E}|^2}{2n_0} \tag{31}$$

by assuming that $\varepsilon_r \ll \Delta\varepsilon$ in the second equality. This shows that n changes proportionally to the 3rd order nonlinearity and the square of the electric field. Furthermore, since $\chi^{(3)}$ and n_0 are constants, we can set:

$$n = n_0 + K|E|^2 \tag{32}$$

where K is called the Kerr constant of a medium. Notice that this parallels the nonlinear term in eq. (25), in which γ is the nonlinear parameter multiplying $|A|^2$.

Split step method

The split step method is a simulation scheme to account for nonlinearities in which propagation is performed in multiple steps. Each step is divided into two, half of the step is used for the linear propagation, and the other half for the non-linear. In mathematical terms, the differential equation employed must be divisible into linear and nonlinear operators [21]:

$$\frac{\partial A}{\partial z} = \left[\widehat{D} + \widehat{N}\right]A \tag{33}$$

with our particular differential equation the linear operator $\widehat{D}$ is the dispersion operator, while $\widehat{N} = i\gamma|A|^2$. To perform the propagation, we separate the space we need to traverse into small steps of size h. We first propagate the pulse a distance of $h/2$ and act on it only with the linear operator. After that, we take a complete step of h and act on the pulse with the nonlinear operator. Finally, we take another half step and act with the linear operator to get:

$$A(z+h) = e^{\frac{\hat{D}h}{2}} e^{\hat{N}h} e^{\frac{\hat{D}h}{2}} A(z) \tag{34}$$

This symmetry helps to increase the precision of the result, notice that performing multiple steps in order produces eq. (35):

$$A(z+nh) = e^{\frac{\hat{D}h}{2}} e^{\hat{N}h} e^{\frac{\hat{D}h}{2}} e^{\frac{\hat{D}h}{2}} e^{\hat{N}h} e^{\frac{\hat{D}h}{2}} \dots A(z) = e^{\frac{\hat{D}h}{2}} e^{\hat{N}h} e^{\hat{D}h} e^{\hat{N}h} \dots e^{\frac{\hat{D}h}{2}} A(z) \tag{35}$$

Since linear half steps are performed one after the other, they can be combined into a single full-length step. The exact answer is given by acting on the pulse with both linear and nonlinear operators at the same time, which yields eq. (36):

$$A(z+h) = e^{(\hat{D}+\hat{N})h} A(z) \tag{36}$$

If the two operators (linear and nonlinear) were commutative, the split-step result would be exact. Nevertheless, to analyze the error involved in this approximation we can Taylor expand the exponentials in eq. (34) [21] for each part of the propagation to obtain eqs. (37-38):

$$e^{\frac{\hat{D}h}{2}} = 1 + \frac{h}{2}\hat{D} + \frac{h^2}{8}\hat{D}^2 + O(h^3) \tag{37}$$

$$e^{\hat{N}h} = 1 + h\hat{N} + \frac{h^2}{2}\hat{N}^2 + O(h^3) \tag{38}$$

where $O(h^3)$ indicates that the remaining terms in the expansion are of the order of h^3 or higher. The multiplication of these Taylor expansions results in:

$$e^{\frac{\hat{D}h}{2}} e^{\hat{N}h} e^{\frac{\hat{D}h}{2}} = 1 + h\hat{D} + h\hat{N} + \frac{h^2}{2}\hat{D}^2 + \frac{h^2}{2}\hat{N}^2 + \frac{h^2}{2}\hat{D}\hat{N} + \frac{h^2}{2}\hat{N}\hat{D} + O(h^3) \tag{39}$$

In contrast, the Taylor expansion of the exact solution is given by:

$$e^{(\hat{D}+\hat{N})h} = 1 + h\left(\hat{D}+\hat{N}\right) + \frac{1}{2}h^2\left(\hat{D}+\hat{N}\right)^2 + O(h^3) \tag{40}$$

As seen in above, the Taylor expansions of the exact solution (eq. (36)) and the split step solution (eq. (39)) are identical up to the h^2 terms. This implies that the error in the approximation of the propagation of the wave will be a factor of the cube of the step size h. Therefore, the split step method is second-order accurate.

The split step method was chosen in favor of other simulation schemes like the finite difference method because it relies on FFTs which can be efficiently computed. Furthermore, it's second order accuracy is enough for the desired precision of the simulation.

The Debye model

The Debye model is used to represent the (frequency-dependent) dielectric function of liquids by representing them as damped oscillations. In general, the dielectric function is divided into relaxation processes and vibrational modes. These are used to represent various physical phenomena, including molecular stretch, rotational modes and collective vibrational modes. The characteristic oscillatory modes of water are shown in figure 5. In eq. (41), these are represented through sums of up to N for relaxation processes and M vibrational modes:

$$\varepsilon(\omega) = \sum_{j=1}^{N} \frac{\Delta\varepsilon_j}{1 - i\omega\tau_j} + \sum_{j=1}^{M} \frac{A_j}{\omega_j^2 - \omega^2 - i\omega\gamma_j} + \varepsilon_\infty \tag{41}$$

where the coefficients τ_j are the characteristic relaxation time, ω_j the pulse carrier frequency, A_j the vibrational amplitudes, and ε_∞ the permittivity at infinite frequency [22]. In the computer simulations, this model is used to obtain the values for the α and β coefficients in the dispersion operator.

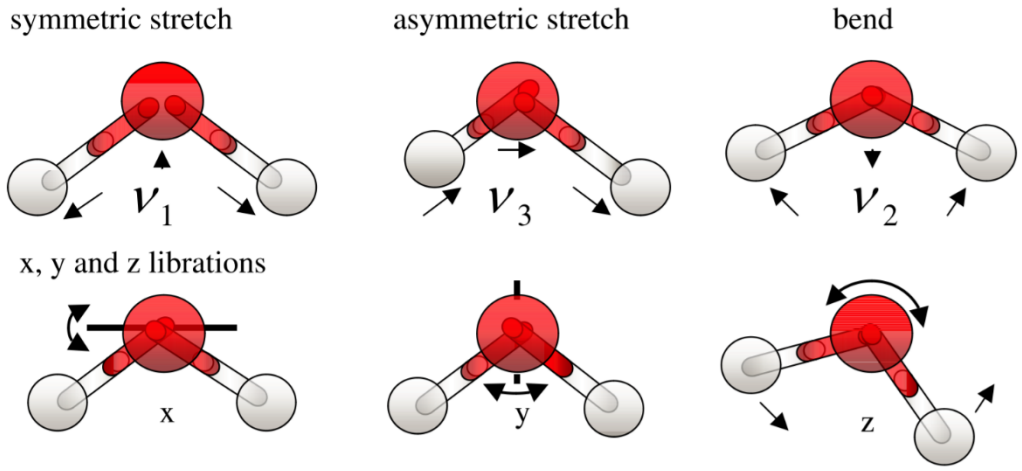

Figure 5: Main vibration modes of water, from [17].

Using THz-TDS to extract dielectric properties

THz-TDS can be used to extract the properties of a material through which a wave has propagated. While mimicking the experimental setup shown in figure 6, the Fourier-space amplitude and phase of the reference pulse and sample pulse can be compared to obtain the propagation characteristics of the material.

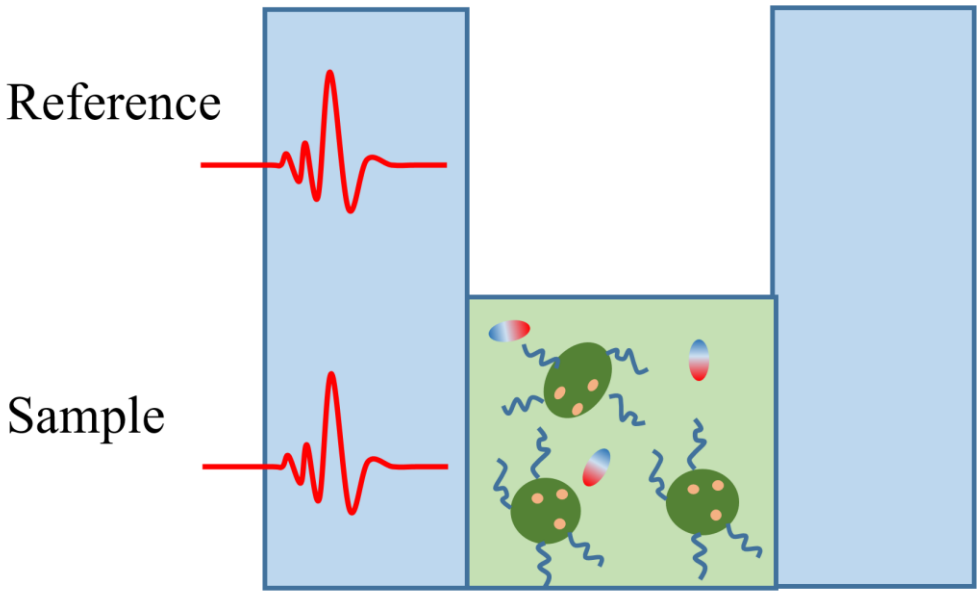

Figure 6: Experimental setup for dielectric property extraction

To do so, first we recall that from the propagation section, we split the complex propagation constant k into its real components β and complex components α, and used those as sum of ω_0-centered terms. Therefore, we can now define

$$k(\omega) = \beta(\omega) + i\alpha(\omega) \tag{42}$$

where

$$\beta(\omega) = \sum_{m=0}^{\infty} \frac{\beta_m}{m!}(\omega - \omega_0)^m \ , \qquad \alpha(\omega) = \sum_{m=0}^{\infty} \frac{\alpha_m}{m!}(\omega - \omega_0)^m$$

Furthermore, from the solution of the linear part of eq. (25) in the Fourier domain, we can see that a propagated pulse will differ from the initial pulse by:

$$E_{in}(\omega) = E_{out}(\omega)e^{\frac{-\alpha(\omega)d}{2}}e^{i(\beta)d} \tag{43}$$

where d is the distance covered inside the medium. We first separate the ratio of electric fields into amplitude and phase:

$$\frac{E_{in}(\omega)}{E_{out}(\omega)} = T(\omega)e^{i\phi(\omega)} \tag{44}$$

This relation can allow us to extract the $\alpha(\omega)$ and $\beta(\omega)$ of a medium by using the following,

$$T(\omega) = e^{-\frac{\alpha(\omega)d}{2}}$$

$$\alpha(\omega) = -\left(\frac{2}{d}\right)\ln[T(\omega)] \tag{45}$$

$$e^{i\phi(\omega)} = e^{i\left(\beta(\omega)+\frac{\omega_0}{c}\right)d}$$

$$\beta(\omega) = \frac{\phi(\omega)}{d} + \frac{\omega_0}{c} \tag{46}$$

The results for α and β in equations (45-46) describe the dielectric properties of the propagation medium, and can also be used, if needed, to calculate the dielectric function $\varepsilon(\omega)$ or the refractive index $n(\omega)$ of the material.

Objective

Since THz spectroscopy is a new and rapidly developing field, experimental methods, and equipment are frequently improved and upgraded. More intense laser sources and more complex phenomena are being able to be studied The goal of this MQP is to develop MATLAB simulations in order to aid the development and understanding of the experiments performed at WPI's Terahertz lab, in particular, when stronger sources allow the observation of nonlinear propagation. It is desired that the simulation should fit certain functionality goals. First of all, its performance should be accurate. THz experimental data has a large portion of noise components around the main pulse oscillations, and the experimental setups are fairly complex, with various transmission and reflection factors involved. The program is intended to serve as a guide for nonlinear effects and thresholds rather than a perfect linear propagation computation, so errors of up to 0.3 are considered acceptable. On the other hand, it is very useful to be able to determine the dielectric function of a medium through the comparison of two pulses, and this method allows for high accuracy, so a relative error of under 0.1 is expected.

In terms of run-time, this type of simulation is typically not very computationally intensive, so long run-times are not the norm. Nevertheless, the simulation should be optimized for more efficient running (looking ahead for the future inclusion of more complex features), and runs should complete in a matter of seconds. Another requirement for the program is ease of use and extendibility. Since the program is intended to be used in the laboratory (as opposed to being a standalone result) ease of use and data visualization can be very helpful for comparing to experimental data. Furthermore, the extendibility of the code will allow for future improvements and the modelling of more complex media and phenomena.

Design

The main concerns in developing the simulation were precision, ease of use and modification, and efficiency. The split-step method, which was chosen in favor of other simulation methods like the finite-difference method, guarantees a precision that the error is at most the cube of the step size. Another benefit of the split-step is its generality and widespread use, which ensure it is appropriate for the current design and for future changes. The split-step is also a very generalized procedure to solve nonlinear equations, so that means that if we wished to alter eq. (25) to account for other effects or factors, that would require minimal work to do in code, since the solution and analysis procedure would remain the same. Furthermore, the ability to simulate the propagation of the wave through a medium, and to double-check the results by extracting the dielectric properties, is helpful to discover mistakes.

Programming Language

The two principal options for programming language were C++ and Matlab. C++, as a compiled language, is generally faster than Matlab. Nevertheless, Matlab has various benefits to its own. It supports complex number arithmetic, which would have to be independently written in C++. Matlab's implementation of the Fast Fourier Transform (FFT) is based upon the highly-efficient hand-optimized FFTW library, while in C++ it would have to be included through an external library. Since the Fourier transform is an integral part of the simulation process, given that every evaluation of the linear propagation requires an FFT and an inverse FFT, there is a speed increase brought by the FFT's optimization. Matlab has various built-in features such as curve-fitting, and envelope and phase calculation which ease the building process. Moreover, its plotting capabilities allow for fast and effective visualization of results [24].

Numerical Implementation

The simulations were built using Matlab scripts to calculate the dispersion factor, obtain a pulse envelope, propagate it, and then study the results. The final result was modularized to ease the creation of possible additions to the current program. For ease of use, the numbers used in the implementation were normalized to picoseconds, Terahertz, MV/mm and millimeters.

As explained in the Propagation chapter, Kerr nonlinearity is the most dominant nonlinear term for the setup concerned. There are two main factors that define whether nonlinear effects can be observed in this system and the peak electric field size of the pulse the value of the Kerr constant. The measurement of the value of the Kerr constant in the THz domain has been performed in [25] using single-cycle pulses with electric field strengths of 250 kV/cm, which determined the upper bound for the Kerr constant as $1.5 \times 10^{-16}\mathrm{m/V^2}$, which, normalized to the units used in the simulation is $0.15\ \mathrm{mm/MV^2}$.

Debye model fit

The Debye model for water requires two relaxation processes and one vibrational mode, the coefficients used are in Table 1, and the dielectric function waveform obtained for the 0.2-2.5 THz range in figure 7:

$\Delta\varepsilon_1$	τ_1 [ps]	$\Delta\varepsilon_3$	τ_3 [ps]	$\frac{A}{(2\pi)^2}$ [THz]^2	$\frac{\gamma}{2\pi}$ [THz]	ε_∞
72.3	8.34	2.12	0.36	28.4	7.06	2.68

Table 1: Debye Model parameters of water from [22]

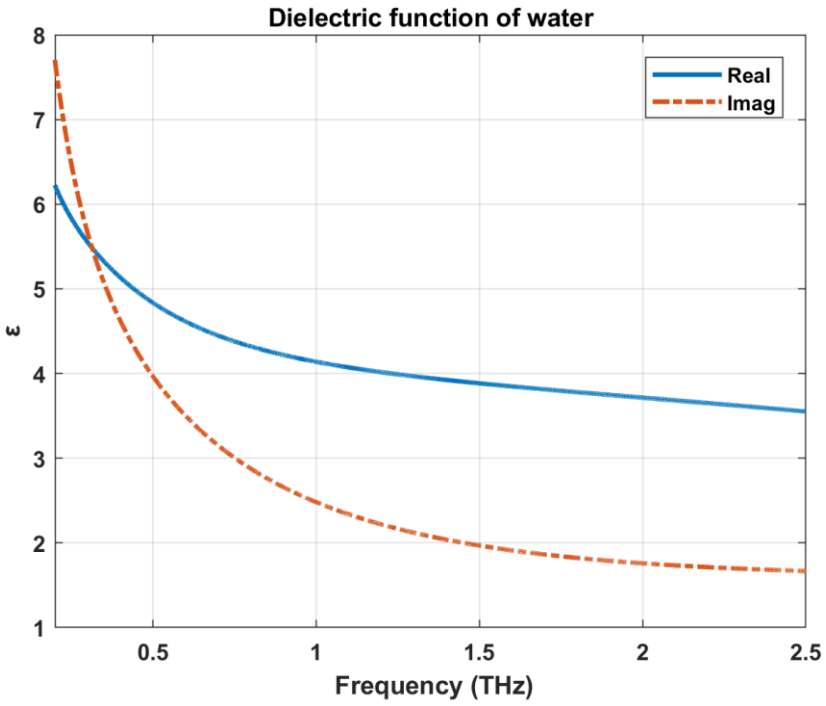

Figure 7: Debye model dielectric function of water.

In the simulation process, the effects of ε are accounted for through the use of the complex propagation constant $k = \frac{\omega\sqrt{\varepsilon}}{c}$, which is used as a sum of α and β terms. To ensure agreement between this decomposition of k and the original function, rather than using its Taylor expansion, curve fitting was used to calculate α and β parameters. Matlab's Curve Fitting Tool was used to fit both $\alpha(\omega)$ and $\beta(\omega)$ to an equation of the form:

$$\alpha(\omega) = \alpha_0 + \alpha_1(\omega - \omega_0) + \frac{\alpha_2}{2!}(\omega - \omega_0)^2 + \frac{\alpha_3}{3!}(\omega - \omega_0)^3 \qquad (47)$$

In addition to improving the accuracy of the simulations, the curve fitting shortens the time needed for computation, since it does not require the calculation of various derivatives of k for its Taylor expansion. The values used in the simulation, and their comparison to the original wave are shown in table 2 and figures 8-9.

	α_0	α_1	α_2	α_3	β_0	β_1	β_2	β_3	ω_0 [THz]
Linear test	0.0005	0	0	0	0	0	0.1	0	0
Water	9.178	1.275	-0.133	0.0262	26.8	6.403	-0.0569	0.0022	0.55

Table 2: Simulation parameters

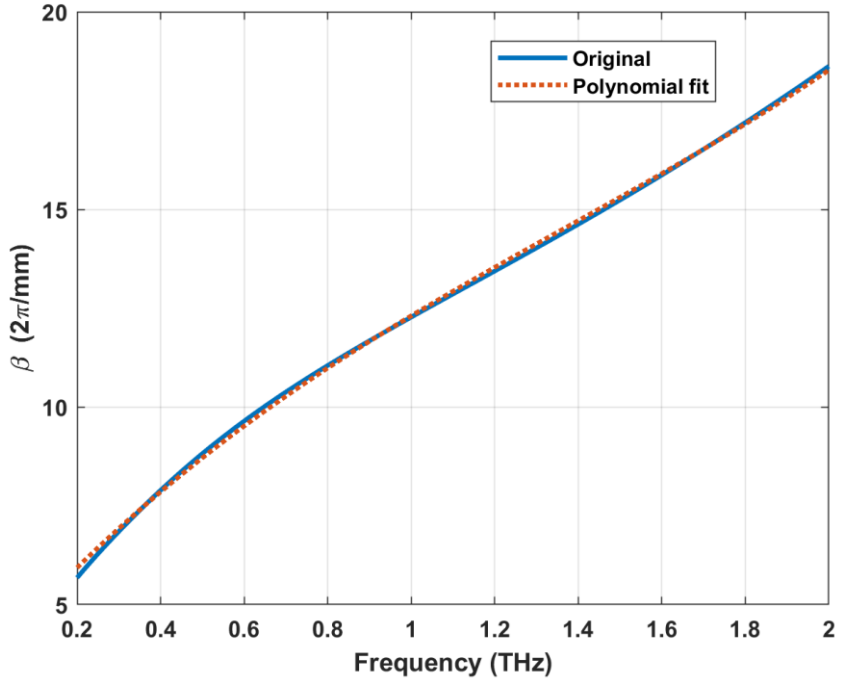

Figure 8: Original wave and polynomial fit of $\beta(\omega)$

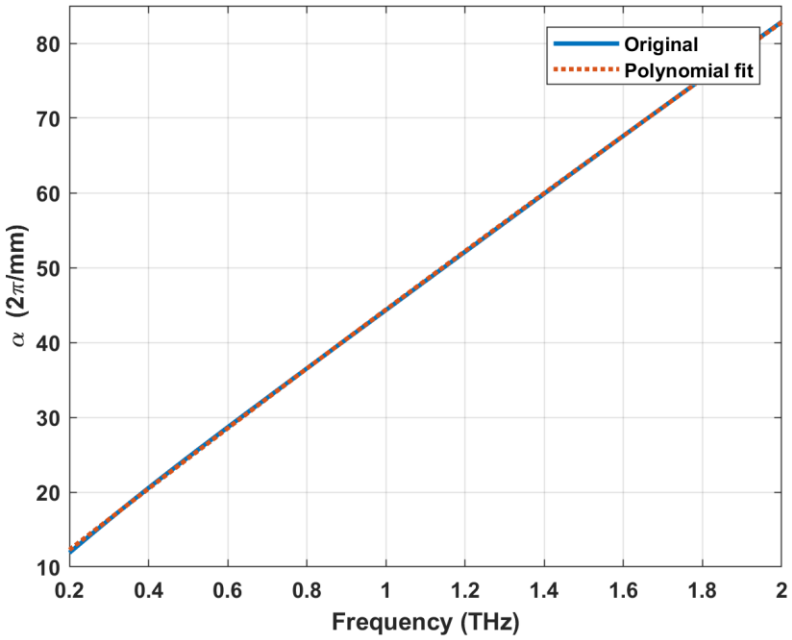

Figure 9: Original and polynomial fit of $\alpha(\omega)$

Input Pulse

A typical wave produced by the equipment at the lab has a time-domain waveform similar to that of Figure 10:

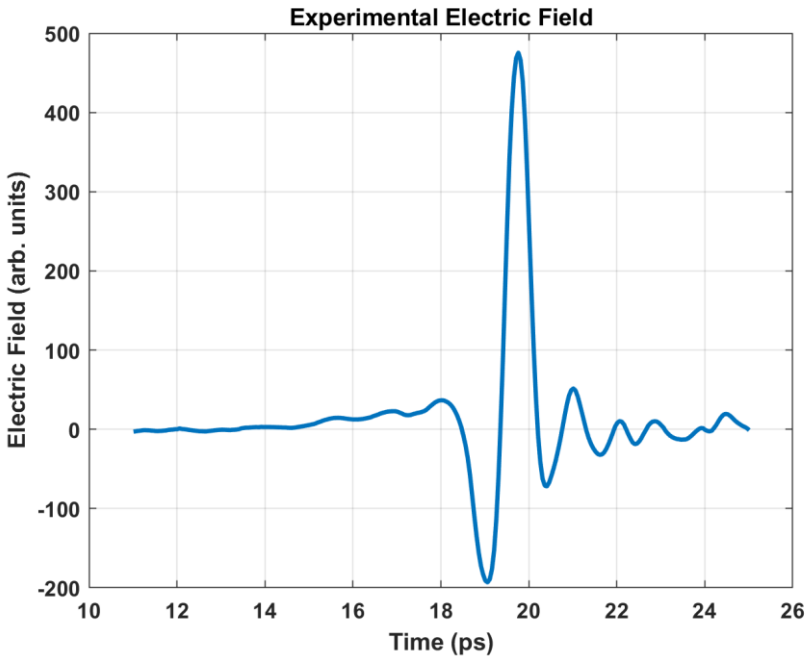

Figure 10: Electric Field sent through samples in the lab.

For some uses of the program it can be useful to simplify this waveform. It was chosen to fit this curve to a sinusoid modulated by a Gaussian:

$$E(t) \approx E_0 \cos(\omega_0 * t + \varphi) * \exp\left(-\frac{(t-t_0)^2}{2\sigma^2}\right) \tag{48}$$

This fitting eases the computation in two manners, it reduces noise around the peak electric field, and the Gaussian can be used in itself as the propagated envelope without needing to perform extra calculations. Furthermore, a simple input curve can make it easier to visualize certain nonlinear propagation features in order to study nonlinear thresholds or extract dielectric properties. The results of the fitting, using Matlab's Curve Fitting Toolbox (which relies on the Levenberg-Marquardt Method [26]) are shown in table 3, with the waveform comparison in figure 11:

E_0	501.2
ω_0 [THz $* 2\pi$]	3.584

$\varphi\ [rad]$	-1.799
$t_0\ [ps]$	19.62
σ	0.4694

Table 3: Electric Field fitting parameters

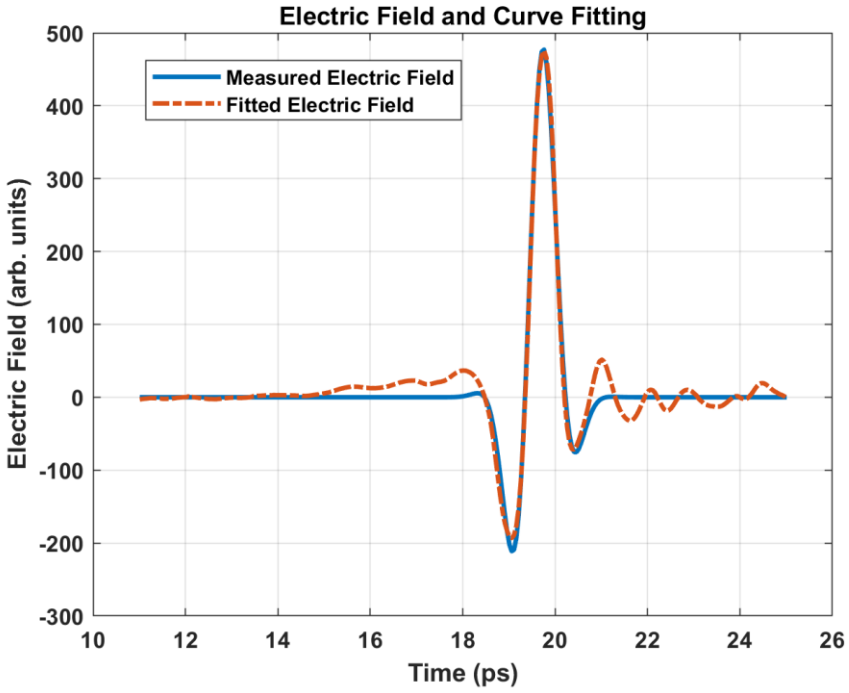

Figure 11: Original Electric Field and fit.

The frequency-domain waveforms of the original and fitted curves are in figure 12:

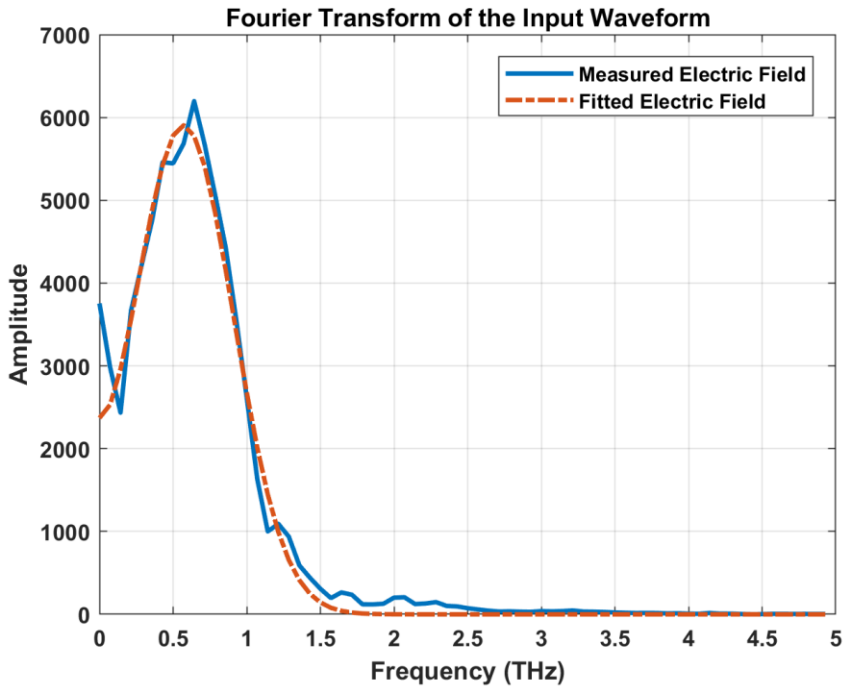

Figure 12: Fourier Transform of the original pulse and fit.

Other methods of equation fitting were also attempted, including Taylor Expansions, Gaussian sums, and fitting the curve with a similar equation to eq. (45), but with a chirp added through the β parameter in eq. (46):

$$E(t) \approx E_0 \cos(\omega_0 * t + \beta * t^2 + \varphi) * \exp\left(-\frac{(t-t_0)^2}{2\sigma^2}\right) \tag{49}$$

Nevertheless, the results did not show much difference upon the equation used previously.

A fitting process was also created in order to obtain electric field envelopes from experimental data without using Matlab's Curve Fitting Toolbox. This was necessary in order to have fitting results readily available for any input waveform. The process assumes that the characteristic frequency of the electric field oscillation (typically close to 0.6 THz) is approximately known. As a first step, Matlab's envelope function is used, which works by transforming the input pulse into Hilbert-space [27]. Yet since this maintains a low of the experimental noise around the main function peak, the pulse obtained from the previous step can be fit to a Gaussian by using the fit function, which results in a smooth, noiseless envelope. The phase of the original electric field, which is necessary to transform back to the output field after

propagation, is determined by using a stochastic gradient descent algorithm. An example result of this fitting process are shown in figure 13, and the code is available in Appendix B.

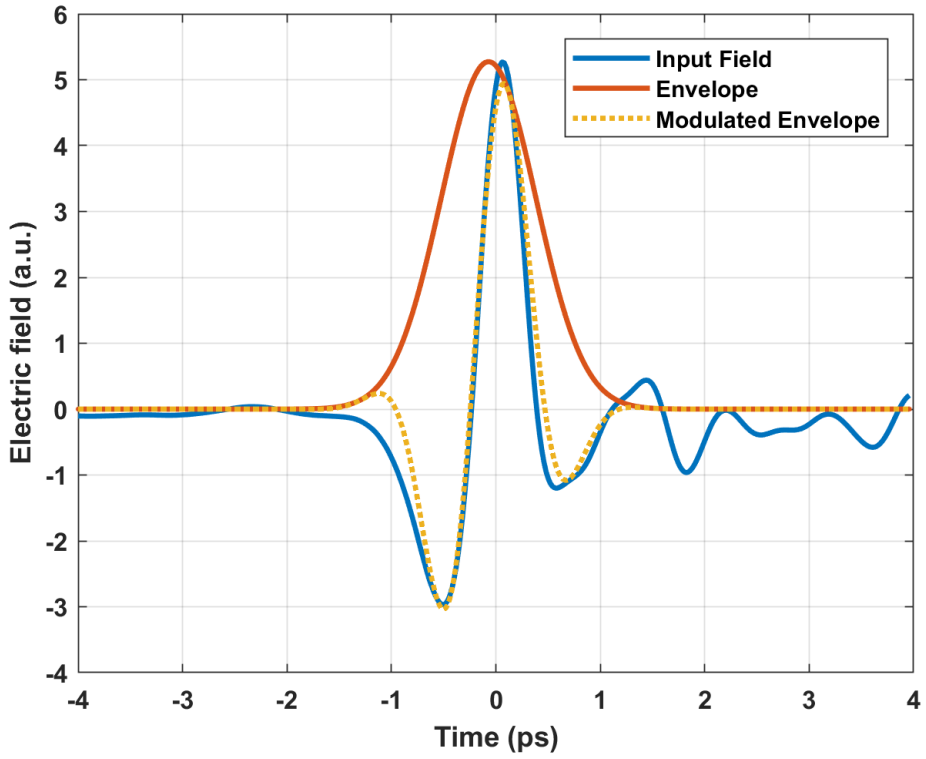

Figure 13: Envelope calculation example

Results

Linear Test

The first step of testing the code was to run it and compare the result with a known analytical solution. This of course, requires the non-linear operator to be set to zero. Furthermore, this requires an input which has a known simple transformation to and from the frequency domain, which is why the Gaussian pulse was chosen. The simplest equation describing the Gaussian pulse:

$$\psi(0,t) = e^{-t^2} \tag{50}$$

Besides that, a simple definition can be used for the dispersion operator. Since this test is intended to test the correctness of the split-step approach, using a dispersion operator that models a real-life material is not required. Thus, we only need to use the lower-order parameters as in eq. (51)

$$\widehat{D} = -\frac{\alpha_0}{2} + i\frac{\beta_2}{2}\frac{\partial^2}{\partial t^2} \tag{51}$$

in which α_0 and β_2 were numbers chosen arbitrarily (to 0.005 and 0.1) in order to produce a model in which attenuation and dispersion were visible. To propagate this Gaussian pulse, it first must be transformed to Fourier-space, which produces:

$$\psi(0,\omega) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\exp(-t^2 - i\omega t)\,dt = \sqrt{\pi}\exp\left(-\frac{\omega^2}{4}\right) \tag{52}$$

Now the Gaussian pulse in eq. (50) must be acted on by the operator to propagate it, which yields the eq. (53):

$$\psi(z,\omega) = \frac{1}{\sqrt{2}}\exp\left(-\frac{\omega^2}{4}\right) * \exp\left(\widehat{D}z\right) = \frac{1}{\sqrt{2}}\exp\left(-\frac{\omega^2}{4} + i\frac{\beta_2}{2}\omega^2 z - \frac{\alpha_0}{2}z\right) \tag{53}$$

The propagated pulse, eq. (53), can once again be transformed back to the time-domain through the inverse Fourier transform:

$$\psi(z,t) = \int_{-\infty}^{\infty}\psi(z,\omega)\exp(2\pi i\omega t)\,d\omega$$

$$\psi(z,t) = \frac{1}{\sqrt{2}}\exp\left(-\frac{\alpha_0}{2}z\right)\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\exp\left(-\frac{\omega^2}{4} + i\frac{\beta_2}{2}\omega^2 z + i\omega t\right)dt$$

$$\psi(z,t) = \frac{\exp\left(-\frac{\alpha_0}{2} - \frac{t^2}{1-2i\beta_2 z}\right)}{2\sqrt{\frac{1}{4} + i\frac{\beta_2}{2}z}} \tag{54}$$

The result, in eq. (54) was tested against the simulated output from the code with an input waveform shown in figure 14:

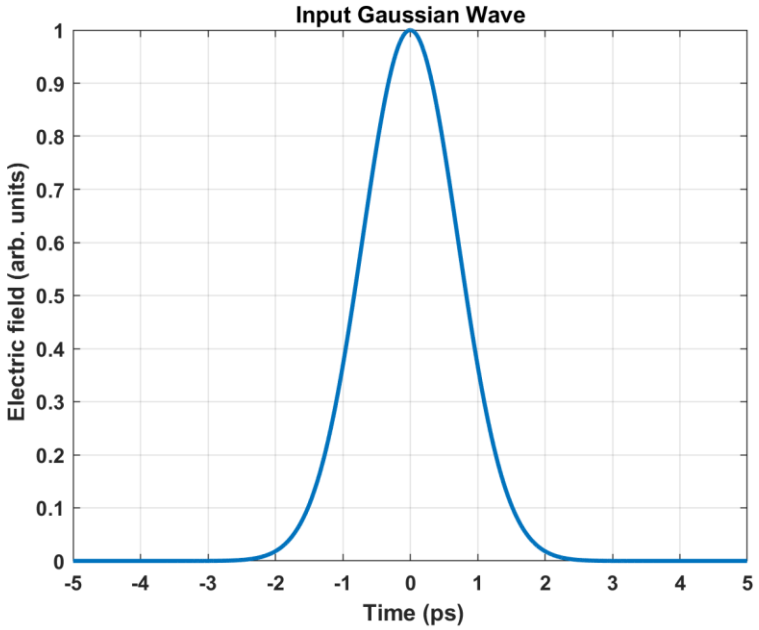

Figure 14: Input Gaussian wave to the linear simulation

The resulting waveform for propagation across 5 mm can be seen in figure 15.

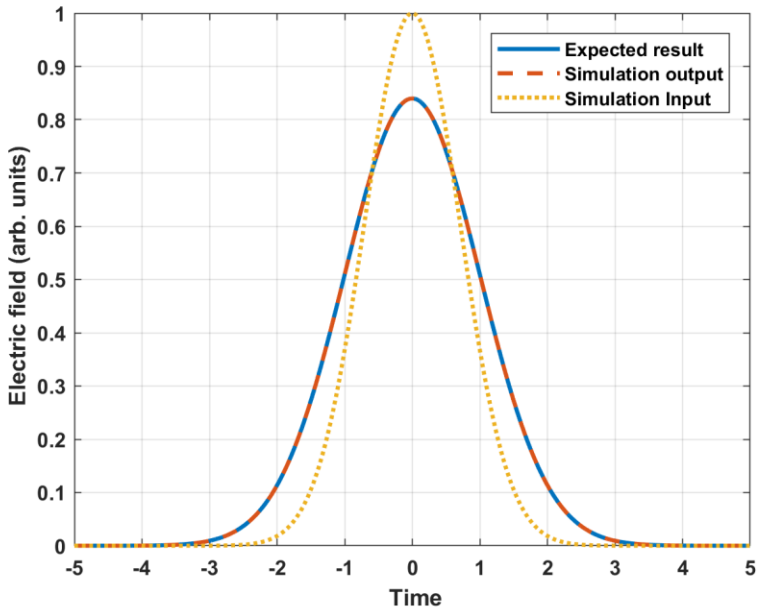

Figure 15: Simulation input, output and expected result. Note that the expected result and output are almost perfectly overlapped.

As seen in figure 15, the expected output matches the simulation output almost perfectly. Since the operation performed is linear, the number of steps performed does not affect the relative error. The code used to perform this simulation can be seen in Appendix C.

Comparison to experimental data

Another method of testing the simulation was by comparing the simulation output to actual experimental data from a test in which a THz pulse was propagated through 0.1mm of water. Simulations will never be able to fully replicate real-life experiments due to various experimental considerations like refraction, phase changes, and transmission through the quartz containing the water, or the effects of the measurement equipment. Noise and the envelope approximation are additional sources of error. Nevertheless, these comparisons still ensure that the output from the simulation corresponds to the real-life behavior of pulses.

The experimental data used was for propagation across 0.1 mm of water. As a first step, and as discussed in the previous chapter, we must calculate an envelope of the initial pre-propagation pulse, shown in figure 16.

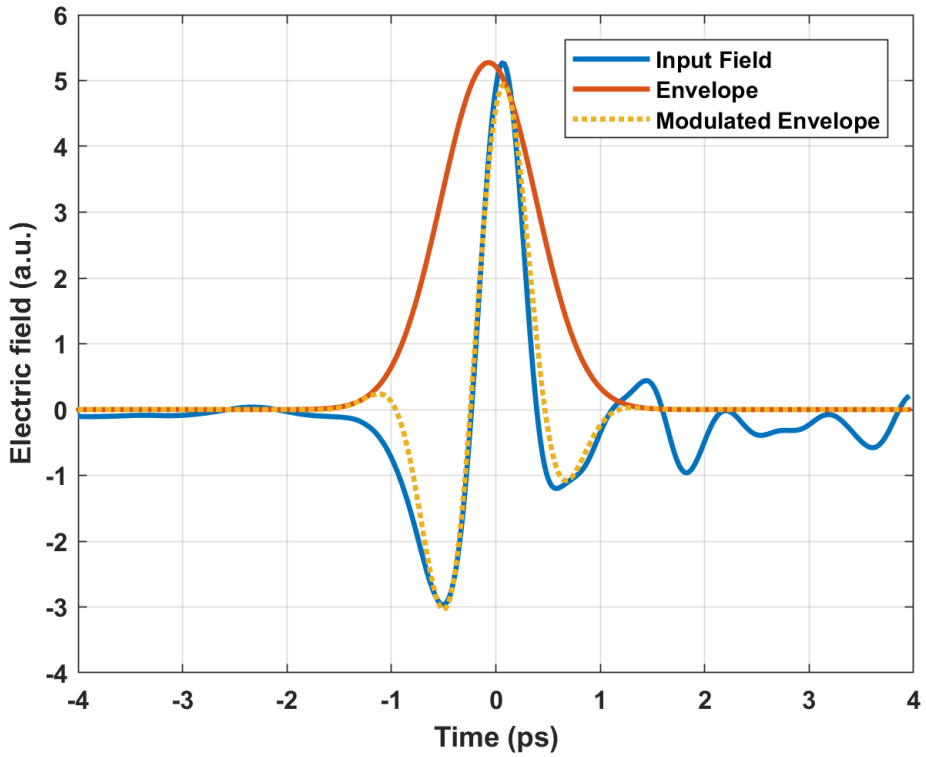

Figure 16: Envelope calculation example

The field is then propagated and transformed back into an oscillating waveform, which results in what is seen in figure 17. Though the simulation is unable to match the noisy waveforms surrounding the main peak, yet it is a fairly good match for the main oscillation. The relative error calculated from the comparison of these waveforms was 0.18, which can be mostly attributed to the noisy sidebands, and fits the design goals. The code used for this part of the project is shown in Appendix D. The average time to run the program (including the various plots used) was 3.2 seconds, well within the design goal.

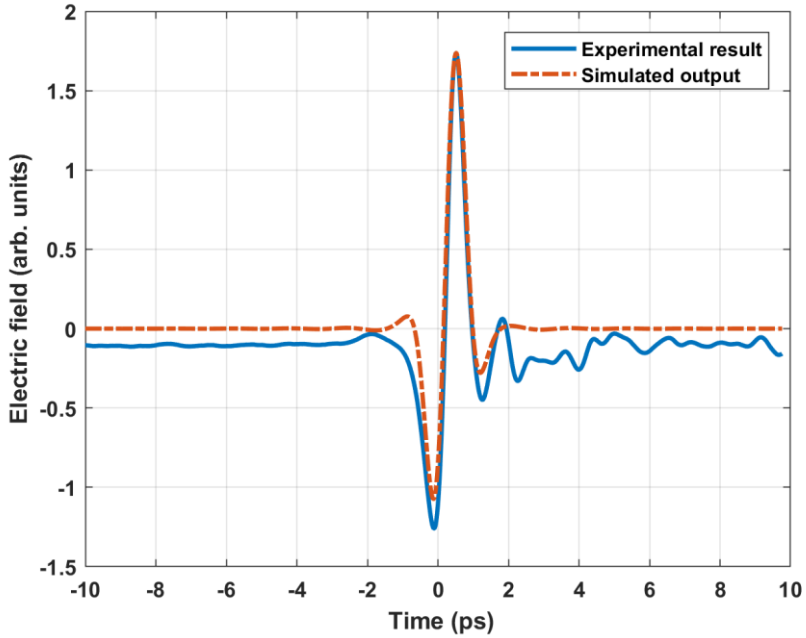

Figure 17: Comparison of experimental data & simulation

Extraction of the complex refractive index

As discussed in the propagation chapter, it is possible to extract the complex refractive index of a material by comparing a pulse propagated through air and another through the sample. This process can be implemented in the code by using the output of the pulse propagation, and compering the result to the waveform used for propagation. This was performed by propagating an input approximation pulse (as that of figure 11) through 0.1 mm, the result of which is shown in figure 18.

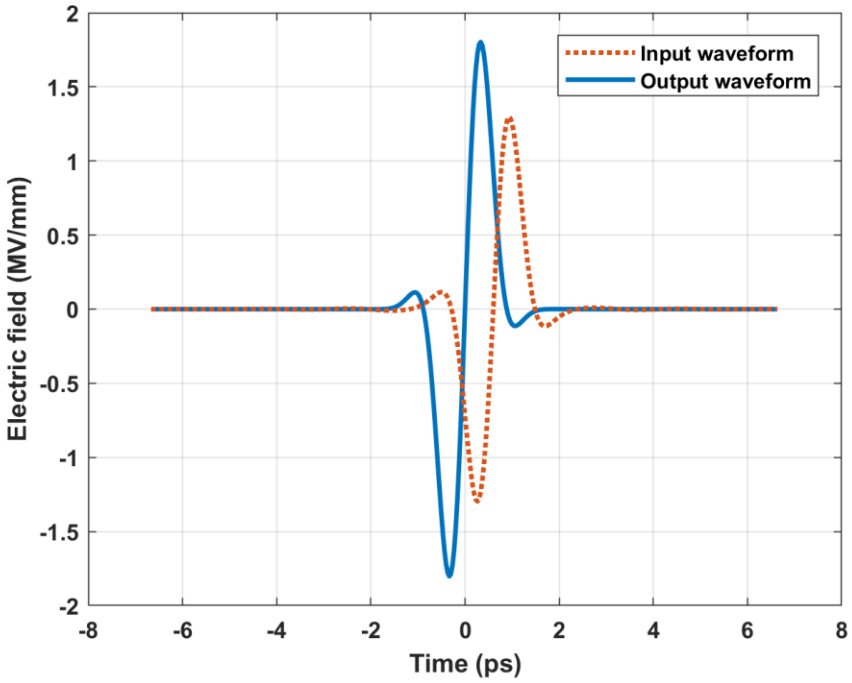

Figure 18: propagation across 0.1 mm of water

The resulting electric field in the Fourier domain can be used to extract the $\alpha(\omega)$ and $\beta(\omega)$ values for the material. The extraction of α was very close to the expected value, with a relative error of $3.7 \cdot 10^{-4}$, and the waveform comparison can be seen in figure 19.

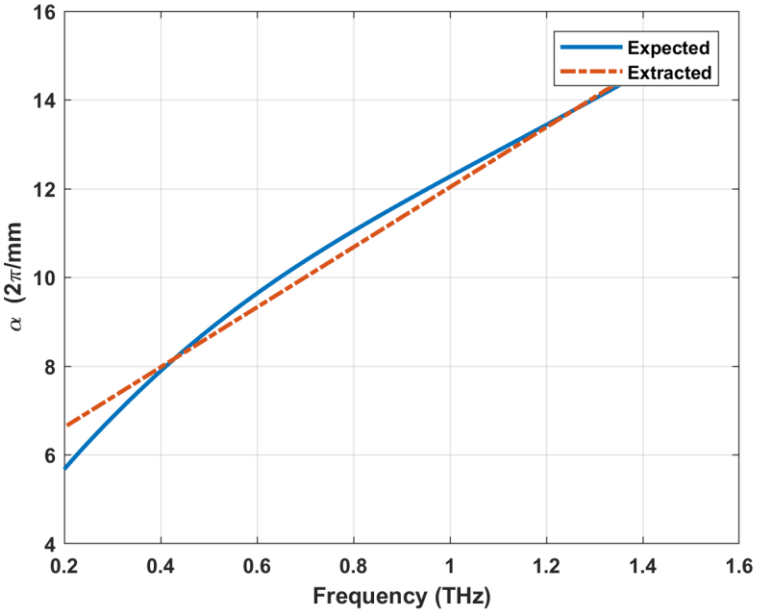

Figure 19: $\alpha(\omega)$ extracted waveform

On the other hand, and since eq. (25) does not account for the value of β_0, the extracted waveform for β has an offset with respect to the original. Nevertheless, the relative error of 0.018 is still within the design goals, and the waveform can be seen in figure 20.

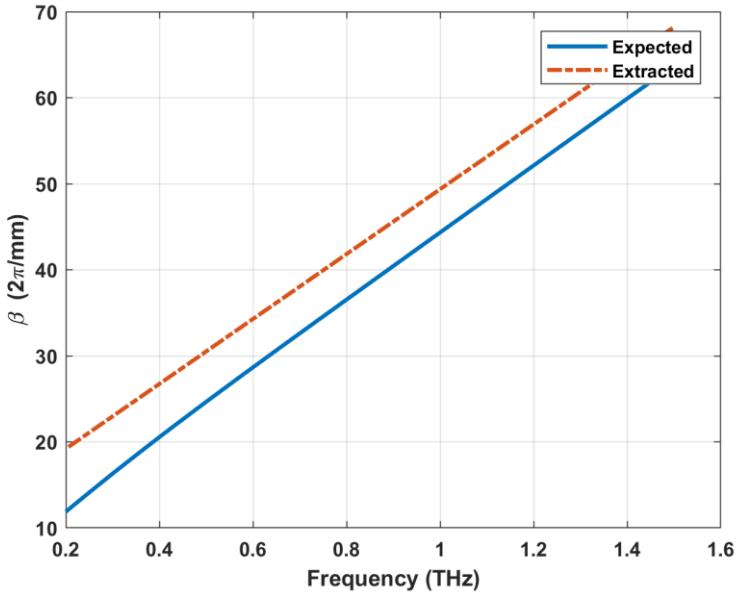

Figure 20: $\beta(\omega)$ extracted function

These results can be used to calculate the refractive index or dielectric function of the medium. The extraction of ε is shown in figures 21-22 The code used for dielectric property extraction is shown in Appendix E.

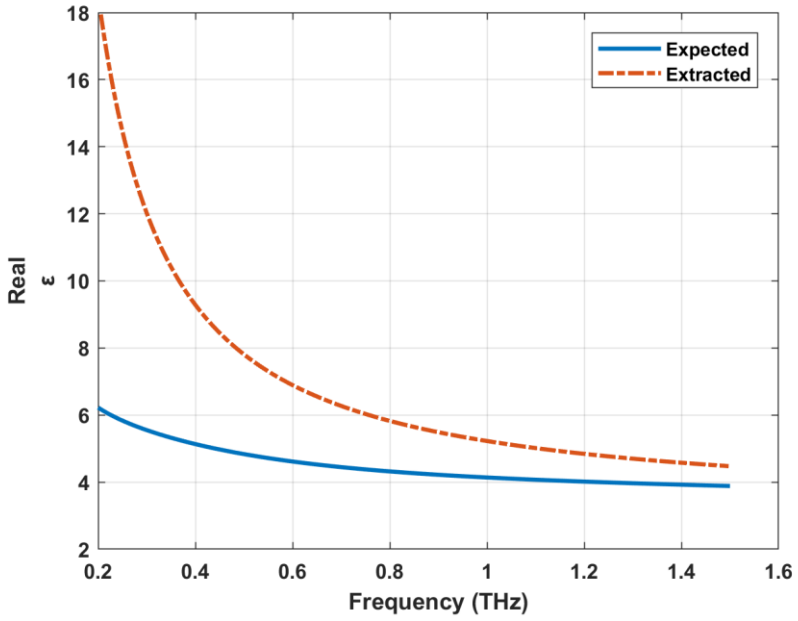

Figure 21: Comparison of the real part of the dielectric function and the extracted value.

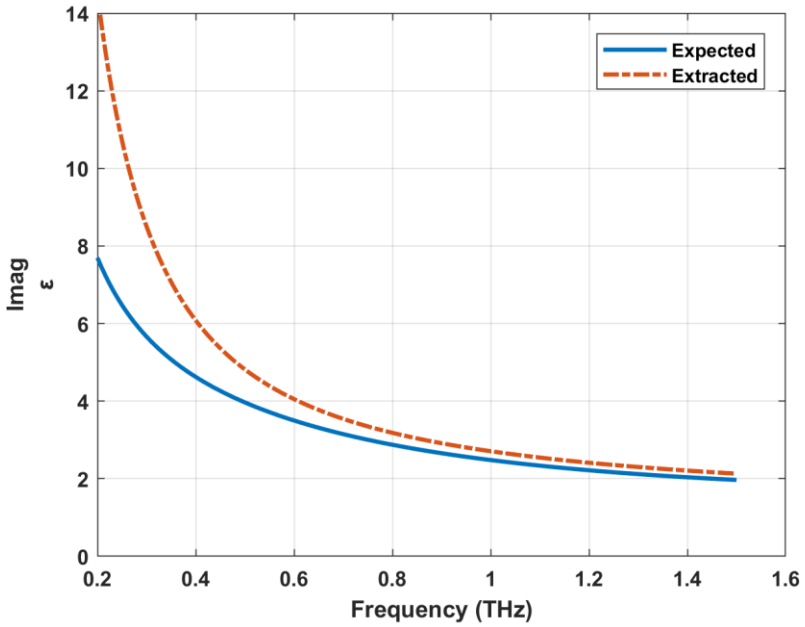

Figure 22: Comparison of the imaginary part of the dielectric function and the extracted value.

Nonlinear effect thresholds

Due to the solution of the nonlinearity in eq. (25), which must be of the form $\exp(i\gamma|A|^2h)$, there are two ways in which we are interested in measuring the effect of nonlinearities, through

comparing the change in peak electric field, and the change in phase shift. Figure 23, compares the result of propagating a wave using two different nonlinear parameters to illustrate the effects.

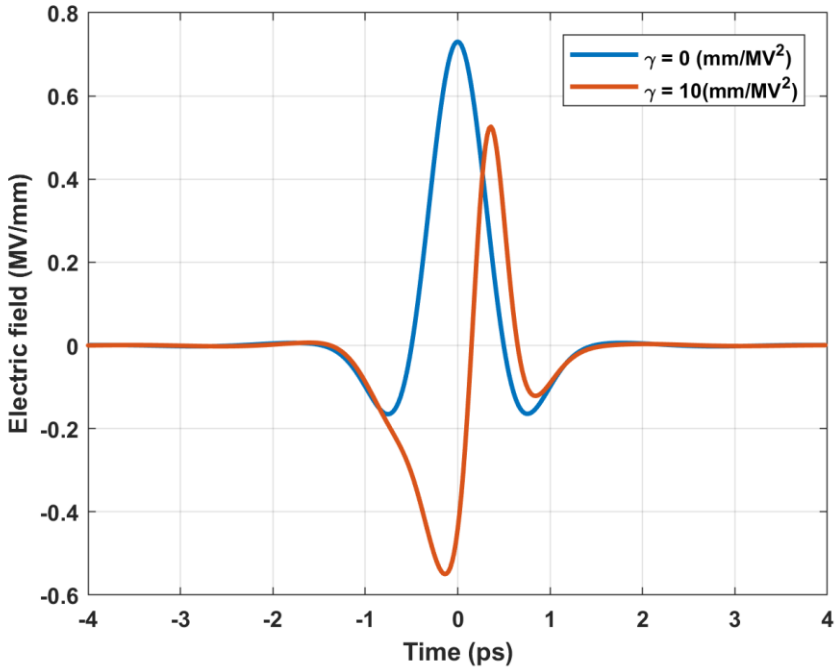

Figure 23: Propagation through 0.1 mm of water with different nonlinear parameters.

It is known, as discussed previously, that the two main contributions to nonlinear effects are the initial peak electric field intensity and the nonlinear parameter. We wanted to determine the threshold for nonlinear effects for a given initial electric field. To do so, iterations of the program were evaluated by varying the initial peak field and nonlinear parameter to calculate the resulting produced time shift and change in peak field. The simulations were performed in a non-dispersive medium in order to focus solely on the nonlinearities, and the results are shown in figures 24-25.

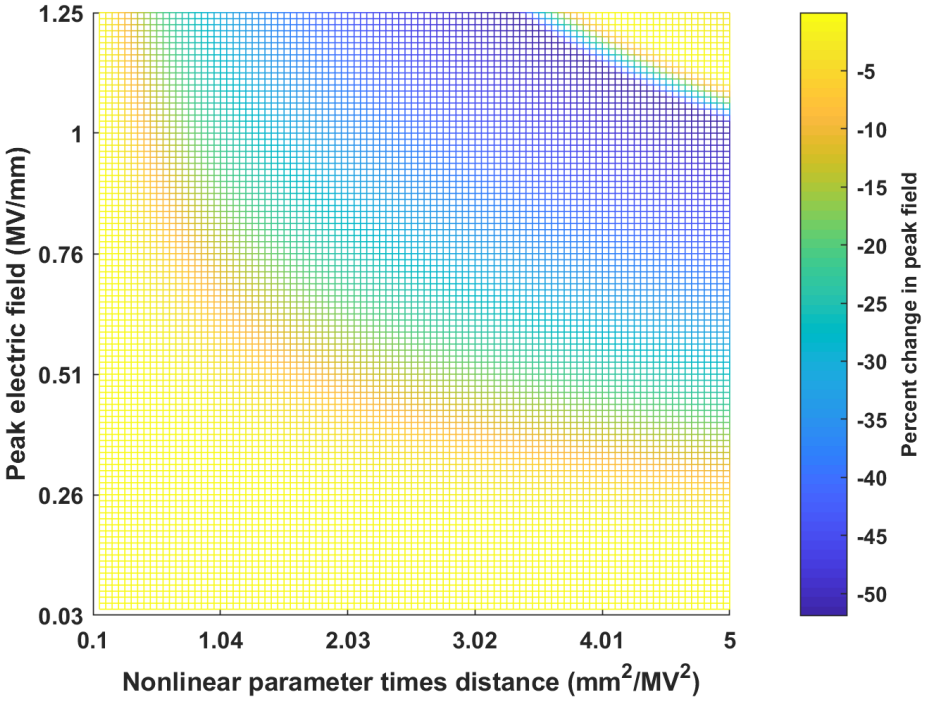

Figure 24: Change in peak field due to nonlinearities.

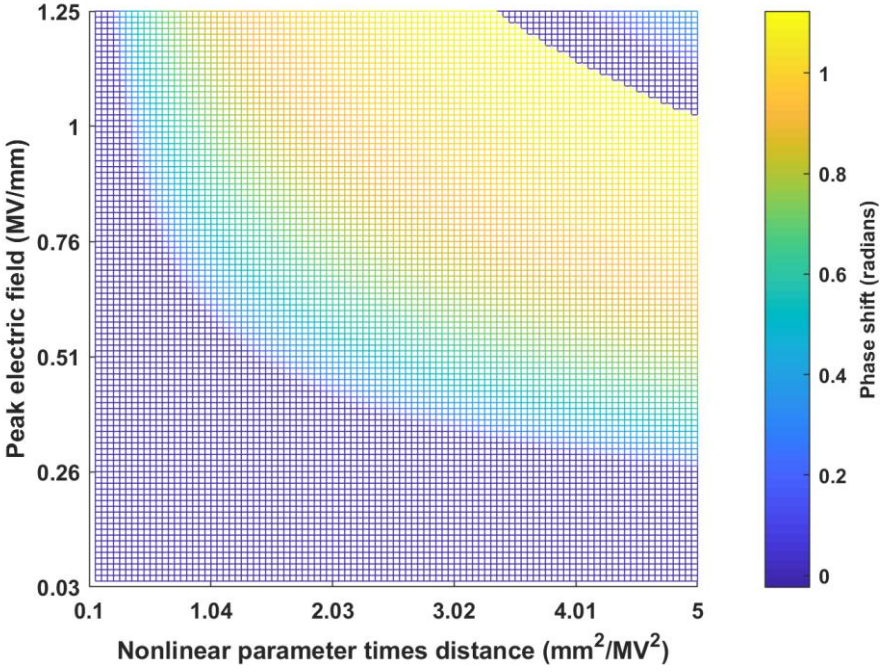

Figure 25: Time shift due to nonlinear effects.

As seen in the figures, there is a threshold indicated by the change of color in the mesh, a transition from purely linear propagation to having observable nonlinear effects. Furthermore, both effects start to be seen at the same time. This tool can be used to determine the threshold electric field needed to observe nonlinearities in a given material. The code used to calculate these graphs is shown in Appendix F.

Conclusion & Future Work

The simulations developed in their current state can serve to approximate and study the phenomena observed at the Ultrafast THz and Optical Spectroscopy Lab, including the appropriate modelling of water. The tool can also help to determine the threshold electric field and parameter values for observing nonlinear effects. Furthermore, it can be useful for determining the dielectric function (or refractive index) of a sample. Another extension of the simulation in which the nonlinear parameter of a sample can be evaluated through multiple iterations is already a work in progress. The program has been designed for adaptability in order to be easily used and easily expanded in the future, with some extra features already in place, such as the inclusion of higher order Taylor expansion terms, or the simulation of Raman scattering. Furthermore, it is very advantageous to use the split-step to perform the propagation because if needed, we could change the linear or nonlinear part of the equation to account for other effects, yet the solution procedure would remain the same. It is also possible to extend the program to simulate more complex media and to include other nonlinear effects.

In addition, there is still a lot of testing that is needed to be performed on the code. When the experimental setup for observing nonlinearities is available, the results should be compared to those of the program. Furthermore, other materials should also be simulated and the results compared to the expected value. It is my hope that it will prove itself useful for the study performed at the Lab and that it can provide a unique insight into nonlinearities when they are observable.

Bibliography

[1] E. Bründermann, H. Hübers, and M. F. Kimmitt, *Terahertz techniques* (Springer, Heidelberg [u.a.], 2012), 151.

[2] Toshihiko Kiwa, Masayoshi Tonouchi, Masatsugu Yamashita, and Kodo Kawase, Optics Letters **28**, 2058 (2003).

[3] C. B. Reid, E. Pickwell-MacPherson, J. G. Laufer, A. P. Gibson, J. C. Hebden, and V. P. Wallace, Physics in Medicine and Biology **55**, 4825 (2010).

[4] J. F. Federici, B. Schulkin, F. Huang, D. Gary, R. Barat, F. Oliveira, and D. Zimdars, Semiconductor Science and Technology **20** (2005).

[5] C. Koch Dandolo, M. Picollo, C. Cucci, and P. Jepsen, Appl. Phys. A **122**, 1 (2016).

[6] T. Kürner and S. Priebe, J Infrared Milli Terahz Waves **35**, 53 (2014).

[7] L. Titova, *WPI - Terahertz Lab*, 2018.

[8] J. B. Baxter and G. W. Guglietta, Analytical chemistry **83**, 4342 (2011).

[9] K. Ravi, W. R. Huang, S. Carbajo, E. A. Nanni, D. N. Schimpf, E. P. Ippen, and F. X. Kärtner, Optics express **23**, 5253 (2015).

[10] Wikipedia, *Permittivity*, 2018.

[11] C. B. Reid, E. Pickwell-MacPherson, J. G. Laufer, A. P. Gibson, J. C. Hebden, and V. P. Wallace, Physics in Medicine and Biology **55**, 4825 (2010).

[12] M. Wang and E. Yang, Nano-Structures & Nano-Objects (2017).

[13] U. Blumröder, H. Hempel, K. Füchsel, P. Hoyer, A. Bingel, R. Eichberger, T. Unold, and S. Nolte, physica status solidi (a) (2016).

[14] X. Yang, X. Zhao, K. Yang, Y. Liu, Y. Liu, W. Fu, and Y. Luo, Trends in Biotechnology **34**, 810 (2016).

[15] N. Krumbholz, C. Jansen, M. Scheller, T. Müller-Wirts, S. Lübbecke, R. Holzwarth, R. Scheunemann, R. Wilk, B. Sartorius, H. Roehle, *et al*, in , Sep 17, 2009), p. 748504.

[16] Y. Li, J. Li, Z. Zeng, J. Li, Z. Tian, and W. Wang, Applied optics **51**, 5885 (2012).

[17] E. Pickwell and V. P. Wallace, Journal of Physics D: Applied Physics **39** (2006).

[18] H. Fröhlich, *Biological coherence and response to external stimuli* (Springer, Berlin u.a, 1988).

[19] J. D. Jackson, *Classical electrodynamics* (Wiley, New York [u.a.], 1999).

[20] R. W. Boyd, *Nonlinear Optics* (Elsevier, Acad. Press, Amsterdam [u.a.], 2008).

[21] S. MacNamara and G. Strang, *Operator Splitting* (Springer, 2016), p. 1.

[22] Uffe Møller, David G. Cooke, Koichiro Tanaka, and Peter Uhd Jepsen, Journal of the Optical Society of America B **26** (2009).

[23] P. U. Jepsen, D. G. Cooke, and M. Koch, Laser & Photonics Reviews **5**, 124 (2011).

[24] A. Gilat, *MATLAB* (Wiley, Hoboken, NJ, 2015).

[25] Peter Zalden, Xiaojun Wu, Liwei Song, Haoyu Huang, Oliver D. Mucke, Christian Bressler, and Franz X. Kartner, in (The Institute of Electrical and Electronics Engineers, Inc. (IEEE), Piscataway, Jan 1, 2016), p. 1.

[26] MathWorks, *Least-Squares (Model Fitting) Algorithms*, 2018.

[27] MathWorks, *Envelope extraction*.

Appendix A: General functions

```
%% Propagate
function [A_out] = propagate(A_in, dispersion, gamma, distance,
steps, time)
%% Performs symmetric split step propagation with steps number
of steps across a length of distance
%Inputs: -----
% A_in: electric field envelope
% dispersion: zero-centered array of the frequency dependent
effect of
% dispersion
% gamma: nonlinearity parameter
% distance: length traversed
% steps: number of steps to be taken
% time: array of time domain values in which the field exists
(only used for plotting)
% ------
% Output: -----
% A_out: final electric field envelope
% -----

deltaz = distance/(steps); % step size in z
deltahalf = deltaz/2; %half step size
transform = fftshift(fft(A_in)); % Take Fourier transform
transform = exp(deltahalf*(dispersion)).*transform; % Advance
half step linearly in Fourier space
A_out = (ifft(fftshift(transform))); % Fourier transform back to
timme Space

for n=1:(steps-1)
    A_Field_nl = exp(1i*deltaz*gamma*(abs(A_out)).^2).*A_out; %
Propagate non-linear part of NLSE
    transform = fftshift(fft(A_Field_nl)); % Take Fourier
transform
    transform = exp(deltaz*(dispersion)).*transform; % Advance
in Fourier space
    A_out = ifft(fftshift((transform))); % Fourier transform
back to Physical Space
    %---- Uncomment the following for plotting the evolution of
the field
```

```
%     figure(1);
%     plot(time, real(A_in), time, real(A_out));

end
A_Field_nl = exp(1i*deltaz*gamma*(abs(A_out)).^2).*A_out; %
Propagate non-linear part of NLSE
transform = fftshift(fft(A_Field_nl)); % Take Fourier transform
transform = exp(deltahalf*(dispersion)).*transform; % Advance
half step linearly in Fourier space
A_out = (ifft(fftshift(transform))); % Fourier transform back to
time Space
```

```
function [eps_w] = epsilon_water2(omega)
%% Calculates the dielectric function of water
%Dielectric function of water parameters
% From 'Terahertz reflection spectroscopy of Debye
% relaxation in polar liquids' by Moller, Cooke, Tanaka & Jepsen
% Inputs: -----
% omega: frequency domain grid used in simulation in (THz*2*pi)
(assumed to
% be zero centered)
% ------
% Outputs: -----
% eps_w: dielectric function of water for the given frequency
array
% -----
len = length(omega);
%% Factors
%Dielectric function of water parameters (2*pi) factor for
normalization to
%THz*(2*pi)
del_epsilon1 = 72.3;
tau_1 = 8.34/(2*pi);
del_epsilon3 = 2.12;
tau_3 = 0.36/(2*pi);
gammav = 7.06*2*pi;
epsilon_inf = 2.68;
omegav = 5.01*2*pi;
Av = 28.4*4*pi^2;

%% General dielectric function
```

```
vibration = (omegav^2-omega.^2-1i*omega*gammav);
eps_w = del_epsilon1./(1-1i*omega*tau_1)+del_epsilon3./(1-
1i*omega*tau_3)+ Av./vibration+epsilon_inf;
eps_w(1:len/2) = eps_w(len:-1:len/2+1); % make it symmetric
```

```
%% Plot & compare
function err = plot_compare(Exp_in, Simul_in, time)
%% Plots and computes the relative error between Exp_in and
Simul_in
% Note: only considers real parts of the waveforms
% Note: assumes vectors are the same length, but matches the
indexes of maxima
% Inputs: -----
% Exp_in: Expected waveform input
% Simul_in: Simulated waveform input
% time: time-domain grid to which electric field values
correspond
% -----
% Output: -----
% err: relative error of Simul_in with respect to exp_in
% -----

len = length(Exp_in);
Exp = real(Exp_in);
Simul = real(Simul_in);
[max1,indexexp] = max(Exp); % Calculate locations of maxima
[max2,indexsim] = max(Simul);
startexp = indexexp > indexsim; % check which index is lower

%% Calculate ranges for plotting
rangeexp = 1+startexp*(indexexp -
indexsim):len+not(startexp)*(indexexp - indexsim);
rangesim = 1+not(startexp)*(indexsim - indexexp):len-
startexp*(indexexp - indexsim);
if ((rangeexp(1) < 1) | (rangeexp(end) > len) | (rangesim(1) <
1) | (rangesim(end) > len))
    rangeexp(1)
    rangeexp(end)
    rangesim(1)
    rangesim(end)
end
```

```
Exp_r = Exp(rangeexp);
Sim_r = Simul(rangesim);
timerange = 1:length(rangeexp); % calculate time range of the
length needed

%% Plotting
% figure(1);
% pl2 = plot(time(timerange), Exp_r); grid;  hold on;
% pl1 = plot (time(timerange),Sim_r,"-.");
% pl1(1).LineWidth = 2;
% pl2(1).LineWidth = 2;
% set(gca,'FontWeight','bold');
% legend([pl2,pl1],["Experimental result","Simulated output"])
% xlabel("\bf Time (ps)"); ylabel("Electric field (arb.
units)");

%% Calculate error
err = sum((Exp_r-Sim_r).^2)/sum(Exp_r.^2)
```

```
%% Get Dispersion
function [d_factor, alphas, betas] = getkdisp(omega, eps_in)
% Calculates the dispersion to be used from a curve fitting of k
% Inputs: -----
% omega: frequency-domain grid
% eps_in: dielectric function of water input (only used for
comparison to
% fit)
% -----
% Outputs: -----
% d_factor: dispersion for omega
% alphas: coefficients for imag(k) fit
% betas: coefficients for real(k) fit
% d_factor: dispersion term
% -----
%% Matlabs curve fitting results
% General model:
%      f(omega) = ka0+ka1*(omega-3.5)+ka2*(omega-
3.5)^2+ka3*(omega-3.5)^3
% Coefficients (with 95% confidence bounds):
%        ka0 =       9.178  (9.161, 9.196)
%        ka1 =       1.275  (1.265, 1.285)
```

```
%        ka2 =    -0.133  (-0.06917, -0.06374)
%        ka3 =    0.0262  (0.004192, 0.004545)
%
%        General model:
%      f(omega) = kb0+kb1*(omega-3.5)+kb2*(omega-
3.5)^2+kb3*(omega-3.5)^3
% Coefficients (with 95% confidence bounds):
%        kb0 =        26.8
%        kb1 =       6.403
%        kb2 =    -0.0569
%        kb3 =   0.0022

% General model:
%      f(omega) = ka0+ka1*(omega-3.5)
% Coefficients (with 95% confidence bounds):
%        ka0 =        9.04  (8.991, 9.089)
%        ka1 =       1.076  (1.068, 1.083)
% General model:
%      f(omega) = kb0+kb1*(omega-3.5)
% Coefficients (with 95% confidence bounds):
%        kb0 =       26.95  (26.88, 27.02)
%        kb1 =       6.158  (6.147, 6.168)

%% Initialization
alphas = [9.04, 1.076];
betas = [26.95, 6.158];
omega0 = 3.5;
c = 0.3;
len = length(eps_in);
[a,ind1] = min(abs(omega-0.2*2*pi));
[a,ind2] = min(abs(omega-2*2*pi));
range1 = ind1:ind2;
k = omega.*sqrt(eps_in)/c;
terms = length(alphas);
k_app = zeros(1,len);
d_factor = zeros(1,len);

%% Dispersion calculation
for num=1:terms
    k_app = k_app+(1i*alphas(num)+betas(num))*(omega-
omega0).^(num-1)/factorial(num-1);
```

```
    d_factor = d_factor - alphas(num)*(omega-omega0).^(num-
1)/(2*factorial(num-1));
    if num > 2
        d_factor = d_factor + 1i*betas(num)*(omega).^(num-
1)/(factorial(num-1));
    end
end
d_factor(1:len/2) = d_factor(len:-1:len/2+1);

%% Comparison plots
figure(13);
p1 = plot(omega(range1)/(2*pi), real(k(range1))); hold on;
p2 = plot(omega(range1)/(2*pi), real(k_app(range1)),":"); grid;
xlabel("Frequency (THz)"); ylabel("\alpha  (2\pi/mm)");
p1(1).LineWidth = 2;
p2(1).LineWidth = 2;
set(gca,'FontWeight','bold');
axis([0.2 2 10 85]);
legend([p1 p2], ["Original" "Polynomial fit"])

figure(14);
p1 = plot(omega(range1)/(2*pi), imag(k(range1))); hold on;
p2 = plot(omega(range1)/(2*pi), imag(k_app(range1)),":"); grid;
xlabel("Frequency (THz)"); ylabel("\beta  (2\pi/mm)");
p1(1).LineWidth = 2;
p2(1).LineWidth = 2;
set(gca,'FontWeight','bold');
axis([0.2 2 5 20]);
legend([p1 p2], ["Original" "Polynomial fit"])
```

```
function [E1, E2,dt] = get_fields_from_files()
%% Promts the user to select two files and returns the first
colums as vector
% Returns two vectors of the same size and even length.
% Note: it is assumed that the sampling rate for both
experimental inputs
% is the same.
% Outputs: -----
% E1: Electric field, the pre-propagation value
% E2: experimental propagated electric field
```

```
% dt: time between samples (in ps)
% -----
% Note: it is assumed that the field is in units of MV/mm
%Parameters for your data (technical):
%Where your files are (relative or absolute addresses are fine)
        defaultPath = './data/';

%The spacing character used in the file to separate entries
        fileDelimiter = '\t';           %(default is tab, \t )

%The columns in the file that have each set of information
        colTime = 1;                    %time in picoseconds
        colData = 2;                    %signed E field


%Open file selection boxes (Empty cell 1 then Empty cell 2)
try
    if ~isempty(successfulLoad) && successfulLoad &&
~askForFiles
        getFiles = false;
    else
        getFiles = true;
    end
catch
    getFiles = true;
end
if getFiles
    successfulLoad = false;
    clc;
    fprintf('Select Cell_1 file');
    [dataC1Name, dataC1Path] = uigetfile('*','Select the file
with your Cell_1 data\n',defaultPath);
    clc;
    fprintf('Select Cell_2 file ');
    [dataC2Name, dataC2Path] = uigetfile('*','Select the file
with your Cell_2 data\n',defaultPath);
    %throwing error
    if isequal(dataC1Name,0) || isequal(dataC1Path, 0) ||...
            isequal(dataC1Name, '') || isequal(dataC1Path, '')
||...
```

```
            isequal(dataC2Name,0) || isequal(dataC2Path, 0)
||...
            isequal(dataC2Name, '') || isequal(dataC2Path, '')
        clear dataC1Name dataC2Name dataC1Path dataC2Path;
        clc;
        error('File selection failed. Are you sure you selected
both files?');
    else
        clc;
        fprintf('Opening ''%s''\n   and
''%s''\n',dataC1Name,dataC2Name);
        successfulLoad = true;
    end
else
    clc;
    fprintf('Using ''%s''\n  and
''%s''\n',dataC1Name,dataC2Name);
end
clear askForFiles;

%% Read the selected file and extract pertinent data
rawC1Data =
dlmread(strcat(dataC1Path,dataC1Name),fileDelimiter);
rawC2Data =
dlmread(strcat(dataC2Path,dataC2Name),fileDelimiter);
T1 = rawC1Data( :, colTime);
T2 = rawC2Data( :, colTime);
E3 = rawC1Data( :, colData);
E4 = rawC2Data( :, colData);
clear colTime colData

%% Make lenght even and equal
len1 = length(E3);
len2 = length(E4);
lused = min(len1,len2);
lused = lused -mod(lused,2);
E1 = E3(1:lused)';
E2 = E4(1:lused)';
dt = T1(2)-T1(1);
```

Appendix B: Envelope calculation

```
%% Get Envelope
function [A_Field, in_phase] = get_envelope(E1, omega0, time,
e_thres)
% calculates the electric field envelope and phase of the input
field
% Inputs: -----
% E1: electric fied
% omega0: center frequency of the input electric field
% time: time domain array in which the field is represented
% e_thres: error threshold accepted for ending the loop
% -----
% Outputs: -----
% A_Field: electric field envelope
% in_phase: initial phase of the input electric field
% -----
% Note: Change step size and threshold error value to affect the
rate of
% convergence

%% Fitting
A_Field= envelope(E1); % as a first approximation use matlabs
envelope function
A_fit = fit(time',A_Field','gauss1'); % afterwards this is fit
to a gaussian function
A_Field = (A_fit.a1)*exp(-((time-(A_fit.b1))/(A_fit.c1)).^2); %
Gaussian fit

%% Phase calculation
in_phase = 0; % phase guess
step_size = pi/6; % step size
err = sum((E1 -
A_Field.*cos(omega0*time+in_phase)).^2)/sum(abs(E1).^2); %
relative error
while (abs(err) > e_thres)
    in_phase = in_phase+err*step_size; % change the phase guess
according to the error
    err = sum((E1 -
A_Field.*cos(omega0*time+in_phase)).^2)/sum(abs(E1).^2); % calc
error
end
```

Appendix C: Linear test

```
%% Linear test
% Propagates a gaussian pulse in the linear domain and compares
the reuslt
% to the expected analytical result
close all; clc; clear all;

distance = 5;
step_num = 100; % No. of z steps
deltaz = distance/step_num % step size in z
deltahalf = deltaz/2;
time_points = 3000;
dtau = 10;
time = (-time_points*dtau/2:dtau:((time_points-
1)*dtau/2))/time_points; % array of time points

%% Propagation constant factors, change these for different
results
% arbitrarily chosen for easily visible results, all other
factors are assumed to be 0.
alpha0 =0.0005; % Dispersion operator coefficients
beta2 = 0.1;

E_Field=  exp(-time.^2); % Initial gaussian pulse
E_Field1 = E_Field;

omega = 2*pi*[-(time_points/2):time_points/2-1]/10; %fourier
space array

%% Propagation
transform = fftshift(fft(E_Field)); % Take Fourier transform
transform = exp(deltahalf*(0.5i*beta2*omega.^2-
alpha0/2)).*transform; % Advance half step in Fourier space
E_Field = (ifft(fftshift(transform)));

for n=1:(step_num-1)
    E_Field_nl = E_Field; % Propagate non-linear part of NLSE
    transform = fftshift(fft(E_Field_nl)); % Take Fourier
transform
    transform = exp(deltaz*(0.5i*beta2*omega.^2-
alpha0/2)).*transform; % Advance in Fourier space
```

```
    E_Field = (ifft(fftshift(transform))); % Fourier transform
back to Physical Space
end

transform = fftshift(fft(E_Field)); % Take Fourier transform
transform = exp(deltahalf*(0.5i*beta2*omega.^2-
alpha0/2)).*transform; % Advance half step in Fourier space
E_Field = (ifft(fftshift(transform)));

%% Expected output calculation
exp1 = exp(-distance*alpha0*0.5);
factor2 = sqrt(1/4-0.5i*beta2*distance);
exp3 = exp(-time.^2/(1-2i*beta2*distance));
expected2 = pi*exp1*exp3/(factor2*2*pi);

%% Result plotting and comparison
figure(3) % Plot expected and result
plot1 = plot(time, abs(expected2)); hold on;
plot2 = plot(time, abs(E_Field),"--");
plot3 = plot(time, abs(E_Field1), ":"); grid;
plot1(1).LineWidth = 2;
plot2(1).LineWidth = 2;
plot3(1).LineWidth = 2;
xlabel("\bf Time (ps)"); ylabel("\bf Electric field (arb.
units)");
legend([plot3,plot1, plot2],["Simulation Input","Expected
result","Simulation output"])
set(gca,'FontWeight','bold');
```

Appendix D: Comparison to experimental data

```
%% Experimental Propagation
% Propagates the approximation of a THz electric field
% Units normalized to mm, THz*2*pi, ps, MV/mm
close all; clc; clear all;

%% Initialization
[E1, E2, dt] = get_fields_from_files(); %get electric fields
% E1 is the field propagated and compared to E2
tic
N1 = length(E1);
time = -dt*N1/2:dt:dt*(N1/2-1);
omega = 2*pi*(-N1/2:N1/2-1)/(dt*N1);
omega0 = 0.6*2*pi;
[a,omega0_index] = min(abs(omega-omega0));

distance = .1; % In mm
step_num = 100; % No. of z steps
deltaz = distance/(step_num); % step size in z
deltahalf = deltaz/2;
c = 0.3; % Speed of light, mm/ps
gamma = 0; % nonlinearity value

%% Dielectric model calculation
epsilon_w = epsilon_water2(omega);% dielectric function of water
[d_factor, alphas, betas] = getkdisp(omega, epsilon_w);
%% Propagation
E1_init = E1; % copy of init electric field

[A_Field, in_phase] = get_envelope(E1, omega0, time,0.205); %
get the envelope of the field

figure(1);
plot(time, A_Field, time, E1, time,
A_Field.*cos(omega0*time+in_phase));

[A_Field] = propagate(A_Field, d_factor, gamma, distance,
step_num, time); % propagate, see propagate.m
```

```
E_res = real(A_Field.*exp(-1i*(omega0*time+in_phase))); % get
back to the electric field

time2 = time+betas(2)*distance; % going back to a static
reference frame
[a,zi1] = min(abs(time));
[a,zi2] = min(abs(time2));
E_res = circshift(E_res,zi1-zi2);
%% Plotting
figure(3);
p1 = plot(time, E_res, time, E1_init, time, E2); hold on;  grid;
p1(1).LineWidth = 2;
p1(2).LineWidth = 2;
p1(3).LineWidth = 2;
set(gca,'FontWeight','bold'); xlabel("Time (ps)");
ylabel("Electric field (MV/mm)");

plot_compare(E2, E_res, time);
toc
```

Appendix E: Dielectric property extraction

```
%% Propagate approx
% Propagates the approximation of a THz electric field
% Units normalized to mm, THz*2*pi, ps, MV/mm
close all; clc; clear all;

%% Initialization
time_points = 3600; % Granularity in the time domain
points = [-time_points/2:time_points/2-1];
dtau = 120; % Time in ps
time = (-time_points*dtau/2:dtau:((time_points-
1)*dtau/2))/time_points; % Array of time points
c = 0.3; % Speed of light, mm/ps
dt = time(2)-time(1); % time between samples

% Curve fitting of initial pulse
E0 =      2.506; % pulse amplitude
in_phase =      -pi/2; % Initial phase
sig =      0.4694; % Standard Deviation
w0 =       3.548; %Pulse resonant freq.
E1 = E0*cos(w0*time+in_phase).*exp(-0.5*(time/sig).^2); %
Electric field

%FFT
N1 = length(E1); %length of pulse
omega = 2*pi*(-N1/2:N1/2-1)/(dt*N1); % Frequency domain grid
omega0 = w0; % center frequency
[a,omega0_index] = min(abs(omega-omega0)); % calculate index of
center frequency

distance = .1; % In mm
step_num = 100; % No. of z steps
deltaz = distance/(step_num); % step size in z
deltahalf = deltaz/2;
c = 0.3; % Speed of light, mm/ps
gamma = 0; % nonlinearity value

%% Dielectric model calculation
epsilon_w = epsilon_water2(omega);% dielectric function of water

[d_factor, alphas, betas] = getkdisp(omega, epsilon_w);
```

```
%% Propagation
E1_init = E1; % copy of init electric field

A_field1 = E0*exp(-0.5*(time/sig).^2); % gaussian envelope of
the field

[A_field] = propagate(A_field1, d_factor, gamma, distance,
step_num, time); % propagate, see propagate.m

E_res = real(A_field.*exp(1i*(omega0*time+in_phase))); % return
to oscillating electric field

time2 = time+betas(2)*distance;
[a,zi1] = min(abs(time));
[a,zi2] = min(abs(time2));
A_field = circshift(A_field,zi1-zi2);
E_res = circshift(E_res,zi1-zi2);
%% Plotting
p_range = N1/2-199:N1/2+200; % plotting range
figure(3);
p2 = plot(time(p_range), E1_init(p_range)); hold on;  grid;
p1 = plot(time(p_range), E_res(p_range),":");
p1(1).LineWidth = 2;
p2(1).LineWidth = 2;
set(gca,'FontWeight','bold'); xlabel("Time (ps)");
ylabel("Electric field (MV/mm)");
legend([p1 p2] , ["Input waveform" "Output waveform"]);


%% Dielectric function extraction
extract_n(A_field1,A_field,dt,epsilon_w,distance, omega); %
extraction of dielectric function
compare_eps(alphas,betas,epsilon_w,omega)
```

```
function extract_n(Ref_in, Sam_in, dt,eps_in,d, omega)
%% Extracts the refractive index and dielectric function of a
sample
%% by comparing pulses propagated through air and the sample
```

```
%Note: assumes both pulses have the same sampling frequency and
length
% Compares and plots extracted value to eps_in
% Inputs: -----
% Ref_in: Reference electric field
% Sam_in: Electric field propagated trough a sample
% dt: time between seconds of the electric field values (in ps)
% eps_in: dielectric function of the sample (if known), used for
% comparison. (assumed to be zero-centered)
% d: distance traveled by the pulse in the sample
% omega: frequency-domain array corresponding to eps_in values
(used only
% for plotting) (in THz*2*pi)
% -----

% Outputs: -----
% n: complex refractive index of the sample
% eps: complex dielectric  function of the sample
% errorn: relative error in the refractive index (by
% comparing to eps_in)
% -----
% Note: output waveforms are only calculated for the relevant
part of the
% spectrum (0.2-2.5 THz) and will have different frequency
domain spacing
% than the input dielectric function to attain higher
resolution.
% Note: Error calculations are based on spline interpolations of
the input
% dielectric function in order to match the spacing of the
calculated
% refractive index
% -----

c=0.3; % speed of light in mm/ps
len = length(Ref_in); % length of inputs
omega = omega/(2*pi);% transform to THz
omega0 = 3.5;

%adding zeros to have a better definition in the fourier domain
Ref = zeros(3*len,1);
```

```
Sam = zeros(3*len,1);
Ref(1:len,1) = Ref_in;
Sam(1:len,1) = Sam_in;


len2 = length(Ref); % New length
R_fft = fft(Ref); %compute fft
S_fft = fft(Sam);

dw1 = 1/(dt*len2); %step of frequency in THz

Ref_fft = abs(R_fft(1:len2/2)); %We only care about half of the
spectrum, the other is just a  repetition
Sim_fft = abs(S_fft(1:len2/2));

% Get phase angles Ref: help unwrap
phsref_unwr = unwrap(angle(R_fft),pi);
phssim_unwr = unwrap(angle(S_fft),pi);

w = 1.5; %max frequency for readable data
W1 = 0:dw1:w; % Frequency grid of readable data in THz
W_len = length(W1); % Length of the grid

% Indexes of relevant frequency range bounds
[a,indexw02] = min(abs(W1-0.2));
[a,indexo02] = min(abs(omega-0.2));
[a,indexo25] = min(abs(omega-w));

% Ranges for the relevant part of the spectrum
range1 = indexo02:indexo25;
range2 = 2+indexw02:W_len;

% Fit phases to polynomials (see help polyval and polyfit)
fitref = polyfit(W1,phsref_unwr(1:W_len)',1);
phsref_fit = polyval(fitref,W1);
fitsim = polyfit(W1,phssim_unwr(1:W_len)',1);
phssim_fit = polyval(fitsim,W1);

phs_diff = (-phsref_unwr(1:W_len)+phssim_unwr(1:W_len)); %Phase
difference
phs_diff= abs(phs_diff');
```

```
T = abs(Sim_fft(1:W_len)')./abs(Ref_fft(1:W_len)'); % Ratio of
amplitudes

beta_calc = phs_diff/d+omega0/c; % beta calculated
alpha_calc = -(2/d)*log(T); % calculate absorption coefficient
eps_calc = ((beta_calc+1i*alpha_calc)*c./(W1*2*pi)).^2; %
extracted dielectric function

kappa_in = 2*pi*omega.*sqrt(eps_in)/c; % input propagation
constant

%% Plot values (comment if needed)
figure(7)
p1 = plot(omega(range1),real(kappa_in(range1))); hold on;
p2 = plot(W1(range2), beta_calc(range2),"-."); grid;
xlabel("Frequency (THz)"); ylabel("\beta  (2\pi/mm)");
p1(1).LineWidth = 2;
p2(1).LineWidth = 2;
set(gca,'FontWeight','bold');
legend([p1 p2], ["Expected" "Extracted"]);
figure(8);
p1 = plot(omega(range1),imag(kappa_in(range1))); grid; hold on;
p2 = plot(W1(range2), alpha_calc(range2),"-.");
xlabel("Frequency (THz)"); ylabel("\alpha  (2\pi/mm");
p1(1).LineWidth = 2;
p2(1).LineWidth = 2;
legend([p1 p2], ["Expected" "Extracted"]);
set(gca,'FontWeight','bold');

figure(17)
p1 = plot(omega(range1),real(eps_in(range1))); hold on;
p2 = plot(W1(range2), real(eps_calc(range2)),"-."); grid;
xlabel("Frequency (THz)"); ylabel(["Real" char(949)]);
p1(1).LineWidth = 2;
p2(1).LineWidth = 2;
set(gca,'FontWeight','bold');
legend([p1 p2], ["Expected" "Extracted"]);
figure(18);
p1 = plot(omega(range1),imag(eps_in(range1))); grid; hold on;
p2 = plot(W1(range2), imag(eps_calc(range2)),"-.");
```

```
xlabel("Frequency (THz)"); ylabel(["Imag" char(949)]);
p1(1).LineWidth = 2;
p2(1).LineWidth = 2;
legend([p1 p2], ["Expected" "Extracted"]);
set(gca,'FontWeight','bold');

%% Interpolation & error calc
alpha_interp =
interp1(omega(range1),(imag(kappa_in(range1))),W1(range2),'splin
e');
beta_interp =
interp1(omega(range1),(real(kappa_in(range1))),W1(range2),'splin
e');

eps_interp =
interp1(omega(range1),eps_in(range1),W1(range2),'spline');

alpha_err = sum((alpha_interp-
alpha_calc(range2)).^2/(alpha_interp).^2);
beta_err = sum((beta_interp-
beta_calc(range2)).^2/(beta_interp).^2);
eps_err = sum((eps_interp-eps_calc(range2)).^2/(eps_interp).^2);
```

Appendix F: Nonlinear effect threshold

```
%% Gamma graph
% Plots graphs of kerr nonlinearity thresholds by varying
initial pulse
% intensity and nonlinear parameter gamma
close all; clc; clear all;

distance = 0.1; % In m change to mm
time_points = 1500; % Granularity in the time domain
points = [-time_points/2:time_points/2-1];
dtau = 10; % Time in ps
time = (-time_points*dtau/2:dtau:((time_points-
1)*dtau/2))/time_points; % Array of time points
c = 0.3; % Speed of light, mm/ps

% Curve fitting of initial pulse Several Gaussians??
E0 =       501.2/200;
sig =      0.4694; % Standard Deviation
t0 =       0; % I centered it around 0
w0 =       3.584; %Pulse resonant freq.
E_Field =  E0*exp(-0.5*((time-t0)/sig).^2)/10; % Electric field
envelope (oscillation is added after propagation

%% Change these two to vary the input field and gamma
multiples = linspace(0.1,5,100); % multiples of the initial
pulse amplitudes
nld = linspace(0.1,5,100);

maxi1 = time_points/2; % initial index of max value
dt = time(2)-time(1); % time step

%% Propagation
for  count1=1:100
    for count2=1:100
        curr_field = E_Field*multiples(count1); % change field
amplitude
        curr_field =
exp(1i*nld(count2)*(abs(curr_field)).^2).*curr_field; %
propagate nonlinearly
        curr_field = real(curr_field.*cos(-1i*w0*time)); % Add
oscillation and take real part
        [a,maxic] = max(curr_field); %index of max value
        time_c = (maxi1-maxic)*dt; % calculate time shift
```

```
        max_change(count1,count2) = (max(curr_field)-
max(E_Field*multiples(count1)))*100/max(E_Field*multiples(count1
)); % percent change in peak
        phase_shift(count1,count2) = time_c*w0; % phase shift
    end
end
%% Plotting
ticksx=multiples*max(E_Field);
printx={num2str(round(nld(1),2))};
printy={num2str(round(ticksx(1),2))};
for count3=1:5
    printx=[ printx, {num2str(round(nld(count3*20),2))} ];
    printy=[ printy, {num2str(round(ticksx(count3*20),2))} ];
end
figure(4);
mesh(max_change);
c = colorbar;
c.Label. String = 'Percent change in peak field';
ylabel("Peak electric field (MV/mm)")
xlabel("Nonlinear parameter times distance (mm^2/MV^2)")
set(gca,'FontWeight','bold');
set(gca, 'xtick', 0:20:100);
set(gca, 'ytick', 0:20:100);
xticklabels(printx);
yticklabels(printy);
figure(7);
mesh(phase_shift);
c = colorbar;
c.Label.String = 'Phase shift (radians)';
ylabel("Peak electric field (MV/mm)")
xlabel("Nonlinear parameter times distance (mm^2/MV^2)")
set(gca,'FontWeight','bold');
set(gca, 'xtick', 0:20:100);
set(gca, 'ytick', 0:20:100);
xticklabels(printx);
yticklabels(printy);
```