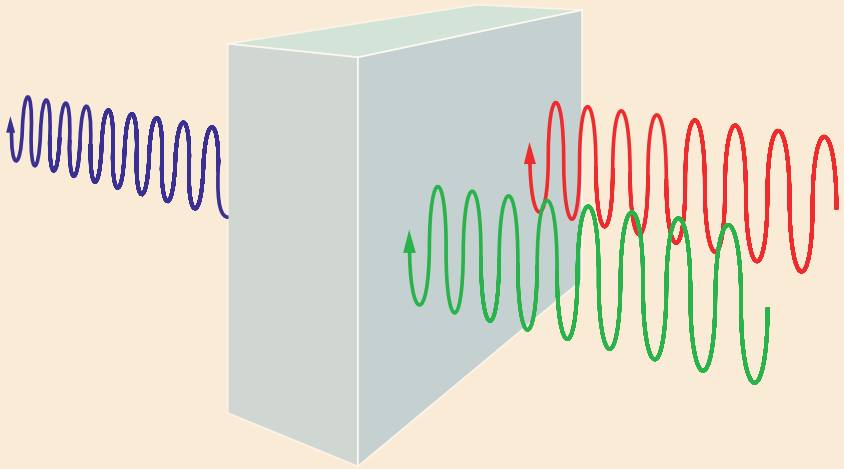
NUMERICAL MODELING OF OPTICAL PARAMETRIC FREQUENCY CONVERSION

ANDERS C. BILFELDT



for temporally confined pulses and spatially confined modes

July 2014

Anders C. Bilfeldt: *Numerical modeling of optical parametric frequency conversion,* for temporally confined pulses and spatially confined modes,
© July 2014

It is of great advantage to the student of any subject
to read the original memoirs on that subject, for science is
always most completely assimilated when it is in the nascent state...

— James Clerk Maxwell [24, Preface, p.xiii]

ABSTRACT

This thesis analyses *Parametric Frequency Conversion* arising from second order non-linear effects in crystals, namely a 5% Mg-doped PPLN crystal.

The analysis starts with a detailed analytical framework discussing the simplifications and approximations used in developing the governing equations later simulated. The theoretical framework constitutes the fundamental backbone of the simulations presented afterwards and the validity of several assumptions is analysed with the simulations as a basis. The effect of accounting for spatial and temporal confinement and depletion are all analysed.

The simulations include; *quasi-monochromatic continuous plane waves*, *quasi-monochromatic continuous Gaussian waves* and *quasi-monochromatic plane wave pulses*. Simulation results are compared to analytical results, to previously published numerical results and to new experimental results.

Results show that the computation time of the simple plane wave theory is significantly less than that of the *Gaussian basis* model used to describe spatially confined CW field and the *Split-step Fourier* method used to describe plane wave pulses. The plane wave theory is seen to be applicable for spatially confined pulses in the *slightly confined*, *weak coupling* regime and for *temporally long* pulses.

Sensible mathematics involves neglecting a quantity when it is small – not neglecting it just because it is infinitely great and you do not want it!

— Paul Dirac [21]

ACKNOWLEDGEMENT

This thesis is submitted in partial fulfilment of the requirements for a Danish Masters's Degree in Physics and Nanotechnology at the Technical University of Denmark.

Experiments and work presented in this thesis have been carried out at the Department of Photonics Engineering, Technical University of Denmark, Campus Risø, Optical Sensor Technology Group.

The work could not have been completed without the help of Associate Professor Peter Tidemand-Lichtenberg and Ph.D. Student Lasse Høgstedt to whom I am very grateful for help and guidance alongside the work presented in this thesis.

The thesis is submitted as a result of work carried out from February 2014 to June 2014 and as such constitutes 30 ECTS points.

This thesis and all simulation scripts presented is available for download at

http://www.Anders.Bilfeldt.dk/thesis/

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

LIST OF ACRONYMS

SHG Second Harmonic Generation

SFG Sum Frequency Generation

DFG Difference Frequency Generation

QPM Quasi-phase Matching

LIST OF SYMBOLS

---

t Time [s]

x Cartesian coordinate

y Cartesian coordinate

z Cartesian or cylindrical coordinate

r Radial cylindrical coordinate

θ Angular cylindrical coordinate

$\mathbf{r}$ Coordinate vector

$\mathbf{H}$ Magnetizing field [A/ m]

$\mathbf{B}$ Magnetic field [T]

$\mathbf{E}$ Electric field [N/ C]

$\mathbf{P}$ Polarization density [C/ m^2]

$\mathbf{D}$ Displacement field [C/ m^2]

$\mathbf{T}$ Material response

I Irradiance (often called intensity) [W/ m^2]

ω Angular frequency [rad/ s]

χ Susceptibility

$\chi^{(n)}$ Susceptibility expansion [$(m/ V)^{n-1}$]

ϵ_0 Vacuum permittivity [$8.854\,187\,817 \times 10^{-12}$ F/ m]

μ_0 Vacuum permeability [$4\pi \times 10^{-7}$ Vs/ Am]

1 MOTIVATION

Second harmonic generation, a special case of *three wave mixing*, is of great importance and is widely used in research as well as industrial application such as generation of 532 nm lasers and measurement of the retardation induced by optical elements [10]. Within recent years a strong emphasis has been put on the development of clinical applications for optical imaging of cells and tissues using second harmonic generation [8].

Generation of mid and far infrared light using parametric frequency conversion is of great interest for frequencies where no lasers are available. Applications of interest range from spectroscopy, medical applications, remote sensing and others [30]. Detection of mid and far infrared light by converting the infrared light to the visible range is also desirable, since existing detection technologies within the mid and far infrared spectrum are expensive and has a significant noise level due to the small band gap of the used semiconductors.

Thus parametric frequency conversion is of great importance and potential applications span the entire range of pulse formats and power levels [27]. A fundamental understanding of the physical processes governing parametric frequency conversion is needed to optimise the technology and facilitate the production of new applications. The fundamental understanding includes a set of solid experimental data that describes the effect of the physical properties of light and material. It is meanwhile of equal importance to have a solid theoretical understanding.

It is of great advantage to simulate the dependence on a variety of different physical parameters before verifying with expensive and time consuming experiments. This thesis aims at creating a numerical tool for analysing the effect of CW, *spatially confined* modes and *temporal confined* plane wave pulses.

The numerical tool is developed in Matlab and provided to the research group *Optical Sensor Technology Group* at *DTU Fotonik*, such that further improvements of the simulation and implementation of more aspects can be added later on.

Part I

THEORY

2

WAVE EQUATION

The *macroscopic Maxwell equations* governing the propagation of light is a set of coupled, non-linear, partial differential equations describing the components of vector fields. Combined with the appropriate boundary conditions, both continuous and discontinuous, these equation constitute the fundamental theory of light propagation. Only few highly idealistic situations have known analytical solutions and even fewer have closed-form analytical solutions. Most analytical solutions build upon a series of simplifications of more or less severe nature. Numerical calculations became widely used with the advent of modern computers and constitute an essential tool where no analytical solution is known.

The following chapter summarizes some of the most common simplifications used to develop the wave equation that describes the propagation of optical electromagnetic fields in non-linear materials. The focus will be on non-linear materials experiencing a second order non-linear effect, the effect utilized in *Parametric Frequency Conversion* processes.

2.1 FUNDAMENTAL MAXWELL EQUATIONS

An infinite material is considered in what follows so that no boundary condition couples the contact surfaces between materials. We are thereby also excluding reflections and singularities arising at the material corners [36]. The famous Maxwell equations governing the propagation of electromagnetic waves in a non magnetic material, valid for most materials at optical frequencies, with no free charges are given by [18]

$$\nabla \times \mathbf{H}(\mathbf{r}, t) = \frac{\partial \mathbf{D}(\mathbf{r}, t)}{\partial t} \tag{1}$$

$$\nabla \times \mathbf{E}(\mathbf{r}, t) = -\frac{\partial \mathbf{B}(\mathbf{r}, t)}{\partial t} \tag{2}$$

$$\nabla \cdot \mathbf{D}(\mathbf{r}, t) = 0 \tag{3}$$

$$\nabla \cdot \mathbf{B}(\mathbf{r}, t) = 0 \tag{4}$$

The relation combining the *displacement field*, $\mathbf{D}$, the *electric field*, $\mathbf{E}$, and the *polarization*, $\mathbf{P}$, together with the relationship combining the *magnetic field*, $\mathbf{B}$, and *magnetizing field*, $\mathbf{H}$, is given by

$$\mathbf{D}(\mathbf{r},t) = \epsilon_0 \mathbf{E}(\mathbf{r},t) + \mathbf{P}(\mathbf{r},t) \tag{5}$$

$$\mathbf{B}(\mathbf{r},t) = \mu_0 \mathbf{H}(\mathbf{r},t) \tag{6}$$

Taking the curl of Equation (2) and using Equation (6) gives

$$\nabla \times \nabla \times \mathbf{E}(\mathbf{r},t) = -\mu_0 \frac{\partial}{\partial t} \left[\nabla \times \mathbf{H}(\mathbf{r},t)\right] = -\mu_0 \frac{\partial}{\partial t} \left[\frac{\partial \mathbf{D}(\mathbf{r},t)}{\partial t}\right] \tag{7}$$

where Equation (1) was used in the last step. Inserting Equation (5) into Equation (7) yields

$$\nabla \times \nabla \times \mathbf{E}(\mathbf{r},t) = -\mu_0 \epsilon_0 \frac{\partial^2 \mathbf{E}(\mathbf{r},t)}{\partial t^2} - \mu_0 \frac{\partial^2 \mathbf{P}(\mathbf{r},t)}{\partial t^2} \tag{8}$$

Equation (8) is the most general form of the wave equation that describes electromagnetic fields in non magnetic materials [31].

Rewriting the left hand side of Equation (8) using vector calculus[1] one gets

$$\nabla \times \nabla \times \mathbf{E}(\mathbf{r},t) = \nabla \left(\nabla \cdot \mathbf{E}(\mathbf{r},t)\right) - \nabla^2 \mathbf{E}(\mathbf{r},t) \tag{9}$$

In a linear isotropic material the first term on the right hand side of equation (9) is zero. This is however not the case for non-linear optics due to the general relation between the displacement field and the electric field, Equation (5). For most non-linear materials the term can however be shown to be small [5] and is as a consequence neglected in the following, meaning that we rewrite

$$\nabla \times \nabla \times \mathbf{E}(\mathbf{r},t) = -\nabla^2 \mathbf{E}(\mathbf{r},t) \tag{10}$$

Substituting Equation (10) into the general non-linear wave equation, Equation (8), gives the simplified wave equation in the time domain [5]

$$-\nabla^2 \mathbf{E}(\mathbf{r},t) = -\mu_0 \epsilon_0 \frac{\partial^2 \mathbf{E}(\mathbf{r},t)}{\partial t^2} - \mu_0 \frac{\partial^2 \mathbf{P}(\mathbf{r},t)}{\partial t^2} \tag{11}$$

2.2 WAVE EQUATION IN FREQUENCY DOMAIN

Until now we have not dealt with the speed at which the fields vary. In the following the notation of a tilde is introduced to denote rapidly

1 Valid for *rectilinear coordinate systems* like the *Cartesian coordinate system* [32]

varying quantities. We will start from the wave equation in the time domain, Equation (11), and use the Fourier expansions of the electric field, the displacement field and the non-linear polarization given by [5].

$$\tilde{\mathbf{E}}(\mathbf{r}, t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \mathbf{E}(\mathbf{r}, \omega) e^{-i\omega t}\, d\omega \tag{12}$$

$$\tilde{\mathbf{P}}(\mathbf{r}, t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \mathbf{P}(\mathbf{r}, \omega) e^{-i\omega t}\, d\omega \tag{13}$$

$$\mathbf{E}(\mathbf{r}, \omega) = \int_{-\infty}^{\infty} \tilde{\mathbf{E}}(\mathbf{r}, t) e^{i\omega t}\, dt \tag{14}$$

$$\mathbf{P}(\mathbf{r}, \omega) = \int_{-\infty}^{\infty} \tilde{\mathbf{P}}(\mathbf{r}, t) e^{i\omega t}\, dt \tag{15}$$

Inserting Equation (14) and (15) into Equation (11) we get

$$\begin{aligned} -\frac{1}{2\pi}\nabla^2 \mathbf{E}(\mathbf{r}, \omega) &= -\frac{(-i\omega)^2}{2\pi c^2}\mathbf{E}(\mathbf{r}, \omega) - \frac{(-i\omega)^2}{2\pi\epsilon_0 c^2}\mathbf{P}(\mathbf{r}, \omega) \\ \Leftrightarrow \nabla_\perp^2 \mathbf{E}(\mathbf{r}, \omega) + \frac{\partial^2}{\partial z^2}\mathbf{E}(\mathbf{r}, \omega) + \frac{\omega^2}{c^2}\mathbf{E}(\mathbf{r}, \omega) &= -\frac{\omega^2}{\epsilon_0 c^2}\mathbf{P}(\mathbf{r}, \omega) \end{aligned} \tag{16}$$

where it was used that $\nabla^2 \equiv \nabla_\perp^2 + \frac{\partial^2}{\partial z^2} = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ and $c^2 = \mu_0\epsilon_0$. Note that Equation (16) must be fulfilled for all frequencies ω.

3 INDUCED POLARIZATION

The relationship between the induced polarization and the electric field is of a dynamic nature rather than an instantaneous nature. The electric field causes the bound electrons to oscillate affectively creating dipole oscillations that radiate electromagnetic waves that in turn drive the dipole oscillations. This process suggests an iterative solution [32].

The induced polarization is often considered to have a linear, parallel relation to the electric field. The macroscopic non-linear effects are for many applications negligible, but for the case of *parametric frequency conversion* the non-linear induced polarization is of essence, namely the second order non-linear polarization.

Let now the induced polarization, $\mathbf{P}$, be given by the perturbation series

$$\tilde{\mathbf{P}}(\mathbf{r},t) = \sum_{k=0}^{\infty} \tilde{\mathbf{P}}^{(k)}(\mathbf{r},t) = \tilde{\mathbf{P}}^{(0)}(\mathbf{r},t) + \tilde{\mathbf{P}}^{(1)}(\mathbf{r},t) + \tilde{\mathbf{P}}^{NL}(\mathbf{r},t) \tag{17}$$

where $\tilde{\mathbf{P}}^{(0)}(\mathbf{r},t)$ represent the static polarization, $\tilde{\mathbf{P}}^{(1)}(\mathbf{r},t)$ the linear induced polarization and $\tilde{\mathbf{P}}^{NL}(\mathbf{r},t)$ represents the non-linear induced polarization.

In the following we restrict our attention to materials with no static polarization and consider only the local material response, where it is assumed that the induced polarization is determined solely by the electric field at the actual observation point in space effectively neglecting nonlocal responses. This is known as the *dipole approximation* [12].

3.1 MATERIAL RESPONSE, TIME-DOMAIN

In general the linear induced polarization at time t is determined by the material response $\mathbf{T}^{(1)}(t;\tau)$ and the electric field at all times τ, meaning that [31]

$$\tilde{\mathbf{P}}^{(1)}(\mathbf{r},t) = \epsilon_0 \int_{-\infty}^{\infty} \mathbf{T}^{(1)}(t;\tau)\tilde{\mathbf{E}}(\mathbf{r},\tau)\,d\tau \tag{18}$$

where $\mathbf{T}^{(1)}(t;\tau)$ is a rank-two tensor that describes the linear material response at time t due to an applied electric field at time τ.

Causality requires that no optically induced polarization can occur before an electric field is applied, meaning that $\mathbf{T}^{(1)}(t;\tau) = 0$ for $t < \tau$. It can be showed, using *time invariance*[1] and *causality*[2], that the material response can depend only on the time difference, $t - \tau$. We define the material response functions as

$$\mathbf{T}^{(1)}(t;\tau) \equiv \mathbf{R}^{(1)}(t-\tau) \tag{19}$$

Causality requires that $\mathbf{R}^{(1)}(t-\tau) = 0$ for $t < \tau$ while reality requires that $\mathbf{R}^{(1)}(t-\tau)$ is real for any time t. The second order non-linear induced polarization is then given by

$$P_\mu^{(2)}(\mathbf{r},t) = \epsilon_0 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} R^{(2)}_{\mu\alpha_1\alpha_2}(t-\tau_1, t-\tau_2) E_{\alpha_1}(\mathbf{r},\tau_1) E_{\alpha_2}(\mathbf{r},\tau_n)\, d\tau_1\, d\tau_2 \tag{20}$$

while the general non-linear induced polarization terms is given likewise by the relation

$$P_\mu^{(n)}(\mathbf{r},t) = \epsilon_0 \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} R^{(n)}_{\mu\alpha_1\cdots\alpha_n}(t-\tau_1,\cdots,t-\tau_n) E_{\alpha_1}(\mathbf{r},\tau_1)\cdots E_{\alpha_n}(\mathbf{r},\tau_n)\, d\tau_1\cdots d\tau_n \tag{21}$$

The notation of $P_\mu^{(n)}$ assumes a summation over coordinates μ, such that $\mathbf{P}^{(n)} = P_x^{(n)}\hat{\mathbf{x}} + P_y^{(n)}\hat{\mathbf{y}} + P_z^{(n)}\hat{\mathbf{z}}$. Since the electric field is a vector and the polarization is likewise a vector, the material response function, $\mathbf{R}^{(n)}$ is a tensor. This is discussed in more detail in Section 3.3.

3.2 MATERIAL RESPONSE, FREQUENCY-DOMAIN

In many situations it is more convenient to describe the polarization by the response function in the frequency domain rather than in the time domain.

Taking the induced polarization series given by Equation (17) and Fourier transforming according to Equation (15) shows

$$\begin{aligned}\mathbf{P}(\mathbf{r},\omega) &= \int_{-\infty}^{\infty} \tilde{\mathbf{P}}(\mathbf{r},t) e^{i\omega t}\, dt = \int_{-\infty}^{\infty}\left[\sum_{k=0}^{\infty}\tilde{\mathbf{P}}^{(k)}(\mathbf{r},t)\right] e^{i\omega t}\, dt \\ &= \sum_{k=0}^{\infty}\mathbf{P}^{(k)}(\mathbf{r},\omega) = \mathbf{P}^{(0)}(\mathbf{r},\omega) + \mathbf{P}^{(1)}(\mathbf{r},\omega) + \mathbf{P}^{NL}(\mathbf{r},\omega)\end{aligned} \tag{22}$$

1 If an input signal $\alpha(t)$ produces an output $\beta(t)$ then shifting the input time δ , $\alpha(t+\delta)$, results in a time-shifted output $\beta(t+\delta)$

2 It is considered to be fundamental that no effect can precede the cause of the effect.

Starting from the linear induced polarization and Fourier transforming the electric field using Equation (14) we get

$$\begin{aligned}
\tilde{P}^{(1)}_{\mu}(\mathbf{r},t) &= \epsilon_0 \sum_{\alpha\in(x,y,z)} \int_{-\infty}^{\infty} R^{(1)}_{\mu\alpha}(\tau)\tilde{E}_{\alpha}(\mathbf{r},t-\tau)\,d\tau \\
&= \epsilon_0 \sum_{\alpha\in(x,y,z)} \int_{-\infty}^{\infty} R^{(1)}_{\mu\alpha}(\tau)\left[\frac{1}{2\pi}\int_{-\infty}^{\infty} E_{\alpha}(\mathbf{r},\omega)\right. \\
&\qquad\qquad \left. e^{-i\omega t}e^{i\omega\tau}\,d\omega\right]\,d\tau \\
&= \epsilon_0 \sum_{\alpha\in(x,y,z)} \int_{-\infty}^{\infty} \chi^{(1)}_{\mu\alpha}(\omega;\omega)E_{\alpha}(\mathbf{r},\omega)e^{-i\omega t}\,d\omega \qquad (23)
\end{aligned}$$

where $\chi^{(1)}(\omega;\omega)$ is the linear susceptibility tensor defined by

$$\chi^{(1)}_{\mu\alpha}(\omega;\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} R^{(1)}_{\mu\alpha}(\tau)e^{i\omega\tau}\,d\tau \qquad (24)$$

The second order non-linear induced polarization in the frequency domain is likewise given by

$$\begin{aligned}
\tilde{P}^{(2)}_{\mu}(\mathbf{r},t) &= \epsilon_0 \sum_{\substack{\alpha_1,\alpha_2\\ \in(x,y,z)}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} R^{(2)}_{\mu\alpha_1,\alpha_2}(\tau_1,\tau_2)\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\right. \\
&\qquad E_{\alpha_1}(\mathbf{r},\omega_1)E_{\alpha_2}(\mathbf{r},\omega_2)e^{i(\omega_1\tau_1+\omega_2\tau_2)}e^{i(\omega_1+\omega_2)t} \\
&\qquad\qquad \left. d\omega_1\,d\omega_2\right]\,d\tau_1\,d\tau_2 \\
&= \epsilon_0 \sum_{\substack{\alpha_1\alpha_2\\ \in(x,y,z)}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \chi^{(2)}_{\mu\alpha_1\alpha_2}(\omega_\sigma;\omega_1,\omega_2) \\
&\qquad E_{\alpha_1}(\mathbf{r},\omega_1)E_{\alpha_2}(\mathbf{r},\omega_2)e^{-i\omega_\sigma t}\,d\omega_1\,d\omega_2 \qquad (25)
\end{aligned}$$

while the general non-linear induced polarisation terms are given by

$$\begin{aligned}
\tilde{P}^{(n)}_{\mu}(\mathbf{r},t) &= \epsilon_0 \sum_{\substack{\alpha_1,\cdots,\alpha_n\\ \in(x,y,z)}} \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} R^{(n)}_{\mu\alpha_1,\cdots,\alpha_n}(\tau_1,\ldots,\tau_n) \\
&\left[\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} E_{\alpha_1}(\mathbf{r},\omega_1)\cdot E_{\alpha_n}(\mathbf{r},\omega_n)e^{i(\omega_1\tau_1+\cdot+\omega_n\tau_n)}\right. \\
&\qquad \left. e^{i(\omega_1+\cdot+\omega_n)t}\,d\omega_1\cdot d\omega_n\right]\,d\tau_1\cdots d\tau_n \\
&= \epsilon_0 \sum_{\substack{\alpha_1,\cdots,\alpha_n\\ \in(x,y,z)}} \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \chi^{(n)}_{\mu\alpha_1\cdots\alpha_n}(\omega_\sigma;\omega_1,\ldots,\omega_n) \\
&\qquad E_{\alpha_1}(\mathbf{r},\omega_1)\cdots E_{\alpha_n}(\mathbf{r},\omega_n)e^{-i\omega_\sigma t}\,d\omega_1\cdots d\omega_n \qquad (26)
\end{aligned}$$

To convert the linear and non-linear induced polarisation to frequency domain we insert Equation (26) into Equation (15) giving

$$\begin{aligned} P_\mu^{(n)}(\mathbf{r},\omega) = \epsilon_0 \sum_{\substack{\alpha_1,\cdots,\alpha_n \\ \in(x,y,z)}} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \Bigg[\int_{-\infty}^{\infty} \chi^{(n)}_{\mu\alpha_1\cdots\alpha_n}(\omega_\sigma;\omega_1,\ldots, \\ \omega_n) E_{\alpha_1}(\mathbf{r},\omega_1)\cdots E_{\alpha_n}(\mathbf{r},\omega_n)\, e^{-i\omega_\sigma t} e^{i\omega t}\, d\omega \Bigg]\, d\omega_1 \cdots d\omega_n \\ = \epsilon_0 \sum_{\substack{\alpha_1,\cdots,\alpha_n \\ \in(x,y,z)}} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \chi^{(n)}_{\mu\alpha_1\cdots\alpha_n}(\omega_\sigma;\omega_1,\ldots,\omega_n) \\ E_{\alpha_1}(\mathbf{r},\omega_1)\cdots E_{\alpha_n}(\mathbf{r},\omega_n)\delta(\omega-\omega_\sigma)\, d\omega_1 \cdots d\omega_n \end{aligned} \tag{27}$$

$\delta(\omega-\omega_\sigma)$ ensures that only frequencies satisfying $\omega = \omega_\sigma = \omega_1 + \omega_2 + \cdots + \omega_n$ contribute. The linear induced polarization in the frequency domain then becomes

$$\begin{aligned} P_\mu^{(1)}(\mathbf{r},\omega) =& \epsilon_0 \sum_{\alpha_1\in(x,y,z)} \int_{-\infty}^{\infty} \chi^{(1)}_{\mu\alpha_1}(\omega_\sigma;\omega_1) \\ & E_{\alpha_1}(\mathbf{r},\omega_1)\delta(\omega-\omega_\sigma)\, d\omega_1 \\ =& \epsilon_0 \sum_{\alpha_1\in(x,y,z)} \chi^{(1)}_{\mu\alpha_1}(\omega;\omega) E_{\alpha_1}(\mathbf{r},\omega) \end{aligned} \tag{28}$$

While the second order non-linear induced polarization in the frequency domain then becomes

$$\begin{aligned} P_\mu^{(2)}(\mathbf{r},\omega) = \epsilon_0 \sum_{\substack{\alpha_1,\alpha_2 \\ \in(x,y,z)}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \chi^{(2)}_{\mu\alpha_1\alpha_2}(\omega_\sigma;\omega_1,\omega_2) \\ E_{\alpha_1}(\mathbf{r},\omega_1)E_{\alpha_2}(\mathbf{r},\omega_2)\delta(\omega-\omega_\sigma)\, d\omega_1\, d\omega_2 \end{aligned} \tag{29}$$

This means that the second order non-linear induced polarization has frequency components at all possible frequencies ω_σ that is a sum of two incoming frequencies, $\omega_\sigma = \omega_1 + \omega_2$. The complicated induced polarization in the frequency domain is greatly simplified in the case of monochromatic fields as will be discussed in more detail in Section 4.

3.3 EFFECTIVE SUSCEPTIBILITY

The n'th order *electric susceptibility*, $\chi^{(n)}$, is a tensor of rank $n+1$ that couples the different components of the electric field to the different components of the induced polarization. The notation $\chi^{(n)}_{\mu,\alpha_1\cdots\alpha_n}(\omega_\sigma;\omega_1,\cdots,\omega_n)$

refers to the coupling between the μ-component of the induced polarization as a result of the α_1-component of the electric field at frequency ω_1, the α_2-component of the field at frequency ω_2 and so on.

The susceptibility is a material parameter. The tensor simplifies from its 3^{n+1} elements due to various symmetries [14] that is not considered here, although it is worth noticing that materials used for second order polarization effects must be non centrosymmetric materials since otherwise all susceptibility tensors of orders with even parity is zero tensors.

The summation in Equation (27) can be simplified by introducing an *effective susceptibility*. The effective susceptibility, χ_{eff}, is introduced for a fixed polarization of the electric fields and a fixed propagation [5]. The assumption that the polarization of the electric fields is fixed is justified by the *phase match condition* that is described in great detail in Section 5.4. The phase match condition means that the induced polarization arising at a different direction than the electric field at the same frequency will not be phase matched and is therefore not a macroscopic phenomena.

Assuming a fixed polarization of the electric field at the different frequencies we get

$$\mathbf{E}(\mathbf{r},\omega_j) = E(\mathbf{r},\omega_j)\hat{\mathbf{e}}^{(j)} = E(\mathbf{r},\omega_j)\sum_{\mu\in(x,y,z)}\hat{\mathbf{e}}_\mu^{(j)} \tag{30}$$

Inserting Equation (30) into Equation (27) gives

$$\begin{aligned}\mathbf{P}^{(n)}(\mathbf{r},\omega) = \epsilon_0 \sum_{\mu\in(x,y,z)} \sum_{\substack{\alpha_1,\cdots,\alpha_n\\ \in(x,y,z)}} \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \\ \chi^{(n)}_{\mu\alpha_1\cdots\alpha_n}(\omega_\sigma;\omega_1,\ldots,\omega_n)E(\mathbf{r},\omega_1)\hat{\mathbf{e}}^{(1)}_{\alpha_1}\cdots E(\mathbf{r},\omega_n)\hat{\mathbf{e}}^{(n)}_{\alpha_n} \\ \delta(\omega-\omega_\sigma)\,d\omega_1\cdots d\omega_n\hat{\mathbf{e}}_\mu \\ = \epsilon_0\hat{\mathbf{e}}_p\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\chi^{(n)}_{eff}(\omega_\sigma;\omega_1,\ldots,\omega_n) \\ E(\mathbf{r},\omega_1)\cdots E(\mathbf{r},\omega_n)\delta(\omega-\omega_\sigma)\,d\omega_1\cdots d\omega_n\end{aligned} \tag{31}$$

where

$$\begin{aligned}\chi^{(n)}_{eff}(\omega_\sigma;\omega_1,\ldots,\omega_n) = \\ \sum_{\substack{\mu,\alpha_1,\cdots,\alpha_n\\ \in(x,y,z)}} \chi^{(n)}_{\mu\alpha_1\cdots\alpha_n}(\omega_\sigma;\omega_1,\ldots,\omega_n)\hat{\mathbf{e}}_\mu\hat{\mathbf{e}}^{(1)}_{\alpha_1}\cdots\hat{\mathbf{e}}^{(n)}_{\alpha_n}\end{aligned} \tag{32}$$

Usually it is of interest to achieve the largest induced polarization. Applying an electric field with frequency components polarized along

given directions results in general in an induced polarization along all directions but not equally efficiently and only the phase matched direction accumulates to a macroscopic field as will be discussed later.

3.3.1 *Magnesium doped Lithium Niobate crystal*

In the simulations discussed in this project we are focusing on *magnesium doped (5%) Periodically Poled Lithium Niobate* (MgO:PPLN) crystals. The highest *effective susceptibility* for this type of crystals is for the coupling between electric fields all polarized along the extraordinary axis of the crystal, corresponding to the y-axis in laboratory coordinates according to Figure 1. This means that the highest second order susceptibility tensor element is the χ_{ccc} according to Figure 1. This is known as *type III-phase matching*.

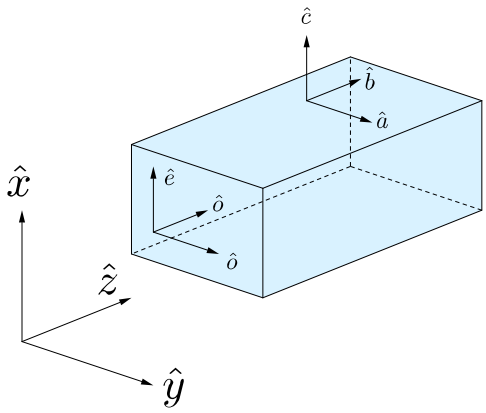


Figure 1: Illustration of the laboratory coordinate system, the crystalline coordinate system and the ordinary and extraordinary axis of crystal. This is the choice of axis that is used in the formulas presented in this thesis.

The χ_{ccc} tensor element is utilized by polarizing all the interacting fields along the extraordinary axis, the y-axis with the choice of axis seen in Figure 1. The susceptibility tensor has elements coupling the different components of the incoming field effectively inducing a polarization that is not generally parallel to the incoming fields. This non-parallel components of the polarization is ignored in the following, since these will generally not be phase matched. Phase match essentially ensures the proper phase relation between the interacting waves such that the non-linear effect arising from different locations within the crystal coherently adds up creating a macroscopic non-linear effect. This is discussed in more detail in Section 5.4.

The effective susceptibility of MgO:PPLN crystals vary in the literature [31]. The manufacture *Covision* states a value of $\chi_{eff} \approx 22\,\mathrm{pm/V}$ which is the value used in the simulation presented later [13].

The reason to use PPLN crystals is mainly due to their superior non-linearity while the MgO doping is to minimize the *photorefractive effect* of PPLN crystals [2]. Interesting parameters of the crystals chosen for given conversion processes include, but is not limited to, *transmission spectrum*, *phase match conditions* (Birefringent phase match or QMP - discussed in detail in Section 5.4), *non-linear coefficients* and *damage threshold*. The parameters cannot be chosen freely and the optimal available materials is thus determined for the precise application. MgO:PPLN crystals are chosen in this thesis since it fits the requirement of DFG at pump wavelength 1064 nm, idler wavelength 1570 nm and signal wavelength 1064 nm since QPM phase match is achievable and hence the highest order non-linear coefficient can be utilized, the crystal has relatively low transmission loss and high damage threshold [4, 6].

4 SIMPLIFICATIONS

The wave equation (11) and (16), combined with the polarization, (26) and (27), constitute the equation governing the propagation of electromagnetic fields in matter. These equations are a set of coupled, non-linear, partial differential equations describing the relationship between the different components of the vector fields. This problem can only be solved analytically for a few highly ideal situations [36]. Firstly we imagine an infinite crystal meaning that we can ignore solutions diverging at the corners of the crystal, neglect reflections and coupling in and out of the crystal.

It would be of great interest to have a model describing pulses and continuous fields having a spatial distribution that propagates either on- or off-axis in arbitrary directions. This is however not easily obtained and certainly not within the scope of this project. In the following focus will be on monochromatic fields. The propagation of plan wave monochromatic CW fields are compared to the propagation of plane wave quasi monochromatic pulses and the propagation of monochromatic, CW, circular symmetric Gaussian beams.

To solve the non-linear wave equation two fundamental approximation are called upon. The first approximation is the *Born approximation*. The second approximation is a *coupled wave theory* where the electric field is considered as a sum of monochromatic fields and the non-linear wave equation is used to derive linear coupled differential equations that govern the interacting monochromatic waves [32]. Numerous other simplification is described in the following.

4.1 LINEAR POLARIZATION

In the following we will be starting from Equation (16) and splitting the polarization into its linear and non-linear part, in accordance with Equation (22). We will be focusing on the second order induced polarization by neglecting higher order terms in the polarization expansion given by Equation (17) and (22).

The linear polarization at frequency ω_σ is only dependent on the same frequency while the second order non-linear polarization at frequency ω_σ is dependent on all possible combinations of ω_1 and ω_2 that satisfies $\omega_\sigma = \omega_1 + \omega_2$. Dealing with an electric field consisting

of a finite number of monochromatic fields giving rise to a appreciable simplification of the second order non-linear polarization as will be shown in Chapter 5.

The linear susceptibility given by Equation (28) is a second order tensor [14]. This means that the electric field in one direction can induce a polarization in another direction. If we are dealing with an electric field polarized along one of the principal axis of the crystal then the linear susceptibility becomes a diagonal matrix. In the following we consider the linear susceptibility as a scalar number since the laboratory coordinate system is placed parallel to the crystal axis, as can be seen in figure 1, and the polarization is chosen along only one of these axis.

Utilizing this gives the following wave equation

$$\begin{aligned}\nabla_\perp^2 \mathbf{E}(\mathbf{r},\omega) + \frac{\partial^2}{\partial z^2}\mathbf{E}(\mathbf{r},\omega) + \frac{\omega^2}{c^2}\mathbf{E}(\mathbf{r},\omega) + \frac{\omega^2}{\epsilon_0 c^2}\epsilon_0\chi^{(1)}\mathbf{E}(\mathbf{r},\omega) \\ = \left[\nabla_\perp^2 + \frac{\partial^2}{\partial z^2} + \frac{n(\omega)^2\omega^2}{c^2}\right]\mathbf{E}(\mathbf{r},\omega) \\ = \left[\nabla_\perp^2 + \frac{\partial^2}{\partial z^2} + k(\omega)^2\right]\mathbf{E}(\mathbf{r},\omega) = -\frac{\omega^2}{\epsilon_0 c^2}\mathbf{P}^{(NL)}(\mathbf{r},\omega) \quad (33)\end{aligned}$$

where it was used that the refractive index, n is given by equation (34) and the propagation constant, k by (35)

$$n(\omega)^2 = 1 + \chi^{(1)}(\omega;\omega) \quad (34)$$

$$k(\omega) = \frac{n(\omega)\omega}{c} \quad (35)$$

4.2 CARRIER WAVE APPROXIMATION

In the following we will consider on-axis propagation meaning that all interacting beams are propagating along the same direction, in this case the z-direction. It is clear that an off-axis propagation will severely increase the complexity of the situation. The spacial overlap of the fields is reduced and so is the effective length of the crystal. The relative phase of the beams is different from the on-axis situation and the polarization arising from the fields must be calculated using the complete susceptibility tensors.

Solutions to the linear electromagnetic wave equation where $\mathbf{P}^{(NL)}(\mathbf{r},t) = 0$ is known to have solutions of monochromatic waves of the form [35].

$$\tilde{\mathbf{E}}(\mathbf{r},t) = \frac{1}{2}\left[\mathbf{A}(\mathbf{r})e^{i(k_0 z - \omega_0 t)} + \mathbf{A}(\mathbf{r})^* e^{-i(k_0 z - \omega_0 t)}\right] \quad (36)$$

where $\mathbf{A}(\mathbf{r})$ is constant in time. In the following we assume that the electric field consists of a carrier wave at frequency ω_0 with a time dependent amplitude modulation, $\tilde{\mathbf{A}}(\mathbf{r},t)$, giving an electric field of the form

$$\tilde{\mathbf{E}}(\mathbf{r},t) = \frac{1}{2}\left[\tilde{\mathbf{A}}(\mathbf{r},t)e^{i(k_0 z-\omega_0 t)} + \tilde{\mathbf{A}}^*(\mathbf{r},t)e^{-i(k_0 z-\omega_0 t)}\right] \tag{37}$$

where $k_0 = \sqrt{n(\omega_0)\omega_0^2/c^2}$. An example of an amplitude modulated wave is illustrated in Figure 2. The amplitude modulation is a 20 fs Gaussian envelope function, $A(t)$, and the carrier wave is at 1064 nm resulting in a electric field $E(t)$. The *Matlab* is seen in Appendix D.

An electromagnetic pulse at a carrier frequency ω has $2T_0\omega/(2\pi)$ cycles counted to the e^{-1} width [31]. This means that a carrier wavelength of 1064 nm in air has 4 oscillations for a 7 fs pulse, approximately 4×10^3 oscillations for a 7 ps pulse and approximately 4×10^6 oscillations for a 7 ns. The assumption of *an amplitude modulated carrier*

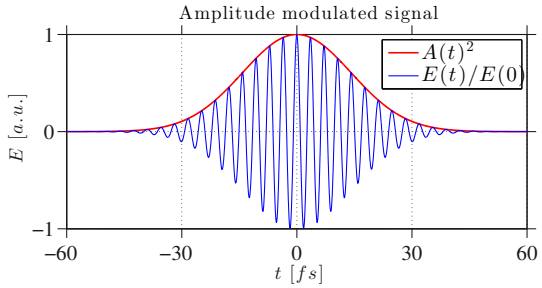


Figure 2: A 20 fs Gaussian envelope function, $A(t)$, and a carrier wave at 1064 nm resulting in a electric field $E(t)$. The *Matlab* is seen in Appendix D

wave is not fully justified since this neglects chirp and group velocity dispersion.

It is seen from Equation (37) that

$$\begin{aligned}
\mathbf{E}(\mathbf{r},\omega) &= \int_{-\infty}^{\infty} \tilde{\mathbf{E}}(\mathbf{r},t)e^{i\omega t}\,dt \\
&= \int_{-\infty}^{\infty} \frac{1}{2}\left[\tilde{\mathbf{A}}(\mathbf{r},t)e^{i(k_0 z-\omega_0 t)} + \tilde{\mathbf{A}}^*(\mathbf{r},t)e^{-i(k_0 z-\omega_0 t)}\right] e^{i\omega t}\,dt \\
&= \int_{-\infty}^{\infty} \frac{1}{2}\left[\tilde{\mathbf{A}}(\mathbf{r},t)e^{ik_0 z}e^{i(\omega-\omega_0)t} + \tilde{\mathbf{A}}^*(\mathbf{r},t)e^{ik_0 z}e^{i(\omega+\omega_0)t}\right] dt \\
&= \frac{1}{2}\left[\mathbf{A}(\mathbf{r},\omega-\omega_0)e^{ik_0 z} + \mathbf{A}^*(\mathbf{r},\omega+\omega_0)e^{-ik_0 z}\right]
\end{aligned} \tag{38}$$

where it was used that $\mathbf{A}(\mathbf{r},\omega) = \int_{-\infty}^{\infty} \tilde{\mathbf{A}}(\mathbf{r},t)e^{i\omega t}\,dt$. Inserting Equation (38) into Equation (16) and using that $\nabla^2 \equiv \nabla_\perp^2 + \frac{\partial^2}{\partial z^2}$ one gets

$$\frac{1}{2}e^{ik_0 z}\left[\nabla_\perp^2 + \frac{\partial^2}{\partial z^2} + 2ik_0\frac{\partial}{\partial z} + (k^2(\omega) - k_0^2)\right]$$
$$\mathbf{A}(\mathbf{r},\omega-\omega_0) + \text{c.c.} = \frac{-\omega^2}{\epsilon_0 c^2}\mathbf{P}^{(NL)}(\mathbf{r},\omega) \tag{39}$$

Taylor expanding $k(\omega)$ around ω_0 we get

$$k(\omega) = k_0 + k_1(\omega-\omega_0) + D \tag{40}$$

where $D = \sum_{n=2}^{\infty}\frac{1}{n!}k_n(\omega-\omega_0)^n$. This means that

$$k^2(\omega) = k_0^2 + k_1^2(\omega-\omega_0)^2 + 2k_0k_1(\omega-\omega_0)$$
$$+D^2 + 2k_0D + 2Dk_1(\omega-\omega_0) \tag{41}$$

Inserting Equation (41) into Equation (39) and neglecting all terms with D gives

$$e^{ik_0 z}\left[\nabla_\perp^2 + \frac{\partial^2}{\partial z^2} + 2ik_0\frac{\partial}{\partial z} + 2k_0k_1(\omega-\omega_0)\right.$$
$$\left.+k_1^2(\omega-\omega_0)^2\right]\mathbf{A}(\mathbf{r},\omega-\omega_0) + \text{c.c.} = -2\frac{\omega^2}{\epsilon_0 c^2}\mathbf{P}^{(NL)}(\mathbf{r},\omega) \tag{42}$$

Converting Equation (42) back to time space by multiplying the left hand side with $\frac{1}{2\pi}e^{-i\omega t}$ and integrating over all ω, the left hand side becomes

$$\begin{aligned}
&e^{ik_0 z}\frac{1}{2\pi}\int_{-\infty}^{\infty}\left[\nabla_\perp^2+\frac{\partial^2}{\partial z^2}+2ik_0\frac{\partial}{\partial z}\right]\mathbf{A}(\mathbf{r},\omega-\omega_0)e^{-i\omega t}\,d\omega\\
&+e^{ik_0 z}\frac{2k_0k_1}{2\pi}\int_{-\infty}^{\infty}(\omega-\omega_0)\mathbf{A}(\mathbf{r},\omega-\omega_0)e^{-i\omega t}\,d\omega\\
&+e^{ik_0 z}\frac{k_1^2}{2\pi}\int_{-\infty}^{\infty}(\omega-\omega_0)^2\mathbf{A}(\mathbf{r},\omega-\omega_0)e^{-i\omega t}\,d\omega+\text{c.c.}\\
=&e^{ik_0 z}e^{-i\omega_0 t}\frac{1}{2\pi}\int_{-\infty}^{\infty}\left[\nabla_\perp^2+\frac{\partial^2}{\partial z^2}+2ik_0\frac{\partial}{\partial z}\right]\tilde{\mathbf{A}}(\mathbf{r},\delta)\,d\delta\\
&+e^{ik_0 z}e^{-i\omega_0 t}\frac{2k_0k_1}{2\pi}\int_{-\infty}^{\infty}\delta\mathbf{A}(\mathbf{r},\delta)e^{-i\delta t}\,d\delta\\
&+e^{ik_0 z}e^{-i\omega_0 t}\frac{k_1^2}{2\pi}\int_{-\infty}^{\infty}\delta^2\mathbf{A}(\mathbf{r},\delta)e^{-i\delta t}\,d\delta+\text{c.c.}\\
=&e^{ik_0 z}e^{-i\omega_0 t}\left[\nabla_\perp^2+\frac{\partial^2}{\partial z^2}+2ik_0\frac{\partial}{\partial z}\right]\tilde{\mathbf{A}}(\mathbf{r},t)\\
&+e^{ik_0 z}e^{-i\omega_0 t}\frac{2k_0k_1}{2\pi}\int_{-\infty}^{\infty}i\frac{\partial}{\partial t}\mathbf{A}(\mathbf{r},\delta)e^{-i\delta t}\,d\delta\\
&+e^{ik_0 z}e^{-i\omega_0 t}\frac{k_1^2}{2\pi}\int_{-\infty}^{\infty}i^2\frac{\partial^2}{\partial t^2}\mathbf{A}(\mathbf{r},\delta)e^{-i\delta t}\,d\delta+\text{c.c.}\\
=&e^{ik_0 z}e^{-i\omega_0 t}\left[\nabla_\perp^2+\frac{\partial^2}{\partial z^2}+2ik_0\frac{\partial}{\partial z}+2ik_0k_1\frac{\partial}{\partial t}\right.\\
&\left.-k_1^2\frac{\partial^2}{\partial t^2}\right]\tilde{\mathbf{A}}(\mathbf{r},t)+\text{c.c.}
\end{aligned}\tag{43}$$

where the substitution $\delta=\omega-\omega_0$ was used. Repeating the procedure for the right hand side of Equation (42) yields

$$\begin{aligned}
&\frac{-2}{2\pi\epsilon_0c^2}\int_{-\infty}^{\infty}\omega^2\mathbf{P}^{(\mathrm{NL})}(\mathbf{r},\omega)e^{-i\omega t}\,d\omega\\
=&\frac{-2}{2\pi\epsilon_0c^2}\int_{-\infty}^{\infty}i^2\frac{\partial^2}{\partial t^2}\mathbf{P}(\mathbf{r},\omega)e^{-i\omega t}\,d\omega=\frac{2}{\epsilon_0c^2}\frac{\partial^2}{\partial t^2}\tilde{\mathbf{P}}^{(\mathrm{NL})}(\mathbf{r},t)
\end{aligned}\tag{44}$$

Combining Equation (43) and (44) gives the wave equation in the timedomain

$$\begin{aligned}
&e^{ik_0 z}e^{-i\omega_0 t}\left[\nabla_\perp^2+\frac{\partial^2}{\partial z^2}+2ik_0\frac{\partial}{\partial z}+2ik_0k_1\frac{\partial}{\partial t}\right.\\
&\left.-k_1^2\frac{\partial^2}{\partial t^2}\right]\tilde{\mathbf{A}}(\mathbf{r},t)+\text{c.c}=\frac{2}{\epsilon_0c^2}\frac{\partial^2}{\partial t^2}\tilde{\mathbf{P}}^{(\mathrm{NL})}(\mathbf{r},t)
\end{aligned}\tag{45}$$

If the electric field consists of a sum of monochromatic waves (37) meaning that

$$\begin{aligned}\tilde{\mathbf{E}}(\mathbf{r},t) &= \sum_{j\geqslant 0}\tilde{\mathbf{E}}_j(\mathbf{r},t)\\ &= \sum_{j\geqslant 0}\frac{1}{2}\left[\tilde{\mathbf{A}}_j(\mathbf{r},t)e^{i(k_{0,j}z-\omega_{0,j}t)}+\tilde{\mathbf{A}}_j^*(\mathbf{r},t)e^{-i(k_{0,j}z-\omega_{0,j}t)}\right]\end{aligned} \tag{46}$$

then the wave equation in the time domain, Equation (45), is given as a sum

$$\begin{aligned}\sum_{j\geqslant 0}e^{ik_{0,j}z}e^{-i\omega_{0,j}t}\Bigg[\nabla_\perp^2+\frac{\partial^2}{\partial z^2}+2ik_{0,j}\frac{\partial}{\partial z}+2ik_{0,j}k_{1,j}\frac{\partial}{\partial t}\\ -k_{1,j}^2\frac{\partial^2}{\partial t^2}\Bigg]\tilde{\mathbf{A}}_j(\mathbf{r},t)+\text{c.c.}=\frac{2}{\epsilon_0 c^2}\frac{\partial^2}{\partial t^2}\tilde{\mathbf{P}}^{(NL)}(\mathbf{r},t)\end{aligned} \tag{47}$$

Using now that for an electric field consisting of a sum of monochromatic fields the polarization can upon insertion of Equation (46) into Equation (27) be shown [31] to be given as

$$\tilde{\mathbf{P}}^{(n)}(\mathbf{r},t)=\frac{1}{2}\sum_{j\geqslant 0}\left[\mathbf{P}_j^{(n)}(\mathbf{r},\omega_{0,j})e^{-i\omega_{0,j}t}+\mathbf{P}_j^{(n)*}(\mathbf{r},\omega_{0,j})e^{i\omega_{0,j}t}\right] \tag{48}$$

where

$$\begin{aligned}\mathbf{P}_j^{(n)}(\mathbf{r},\omega_{0,j}) = 2\epsilon_0\chi^{(2)}(\omega_{0,j};\omega_1,\omega_2,\cdots,\omega_n)\frac{1}{2}\tilde{\mathbf{A}}_1(\mathbf{r},t)e^{ik_{0,1}z}\\ \frac{1}{2}\tilde{\mathbf{A}}_2(\mathbf{r},t)e^{ik_{0,2}z}\frac{1}{2}\tilde{\mathbf{A}}_n(\mathbf{r},t)e^{ik_{0,n}z}\\ +2\epsilon_0\chi^{(2)}(\omega_{0,j};\omega_2,\omega_1,\cdots,\omega_n)\frac{1}{2}\tilde{\mathbf{A}}_2(\mathbf{r},t)e^{ik_{0,2}z}\\ \frac{1}{2}\tilde{\mathbf{A}}_1(\mathbf{r},t)e^{ik_{0,1}z}\frac{1}{2}\tilde{\mathbf{A}}_n(\mathbf{r},t)e^{ik_{0,n}z}\\ +\quad\text{all other distinguishable terms}\end{aligned} \tag{49}$$

In the case of the second order induced polarization this becomes

$$\begin{aligned}\mathbf{P}_j^{(2)}(\mathbf{r},\omega_{0,j}) = \epsilon_0 2\chi^{(2)}(\omega_{0,j};\omega_1,\omega_2)\frac{1}{2}\tilde{\mathbf{A}}_1(\mathbf{r},t)e^{ik_{0,1}z}\frac{1}{2}\tilde{\mathbf{A}}_2(\mathbf{r},t)e^{ik_{0,2}z}\\ +\epsilon_0 2\chi^{(2)}(\omega_{0,j};\omega_2,\omega_1)\frac{1}{2}\tilde{\mathbf{A}}_2(\mathbf{r},t)e^{ik_{0,2}z}\frac{1}{2}\tilde{\mathbf{A}}_1(\mathbf{r},t)e^{ik_{0,1}z}\end{aligned} \tag{50}$$

where the second term on the right hand side is only to be included if $\omega_1 \neq \omega_2$ since only distinguishable terms in the sum of Equation (49) is to be included.

Inserting Equation (17) and (48) into Equation (47) gives

$$\begin{aligned}\sum_{j\geqslant 0} e^{ik_{0,j}z}e^{-i\omega_{0,j}t}\left[\nabla_\perp^2 + \frac{\partial^2}{\partial z^2} + 2ik_{0,j}\frac{\partial}{\partial z}\right.\\ \left. +2ik_{0,j}k_{1,j}\frac{\partial}{\partial t} - k_{1,j}^2\frac{\partial^2}{\partial t^2}\right]\tilde{\mathbf{A}}_j(\mathbf{r},t) + \text{c.c.}\\ = \sum_{j\geqslant 0}\frac{2}{\epsilon_0 c^2}\frac{\partial^2}{\partial t^2}\sum_{n\geqslant 2}\frac{1}{2}\mathbf{P}_j^{(n)}(\mathbf{r},\omega_{0,j})e^{-i\omega_{0,j}t} + \text{c.c.}\end{aligned} \tag{51}$$

Note that the complex conjugate of Equation (51) can be dropped on both sides of the equation without loss of generality [5, Ch. 2.2, p. 75].

Using that Equation (51) must be valid for each term in the series of j in order for the equation to hold for all times, t, and in all space, $\mathbf{r}$, while carrying out the time derivative, rearranging the terms and neglecting higher than second order terms in the polarization we end with

$$\begin{aligned}\left[\nabla_\perp^2 + \frac{\partial^2}{\partial z^2} + 2ik_{0,j}\frac{\partial}{\partial z} + 2ik_{0,j}k_{1,j}\frac{\partial}{\partial t} - k_{1,j}^2\frac{\partial^2}{\partial t^2}\right]\tilde{\mathbf{A}}_j(\mathbf{r},t)\\ = \frac{-\omega_{0,j}^2}{\epsilon_0 c^2}\mathbf{P}_j^{(2)}(\mathbf{r},\omega_{0,j})e^{-ik_{0,j}z}\end{aligned} \tag{52}$$

Equation (52) in combination with Equation (49) constitute the starting starting point for all the simulations presented in this thesis.

4.3 BORN APPROXIMATION

The Born approximation simplifies the relationship between the polarization of the material and the electric field. The electric field is thought of as an input field that induces oscillation of the bound electrons in the atoms of the medium. These oscillating dipoles collectively give rise to an induced polarization. The time delay between the input and the output manifests as a *material memory* meaning that the electric field at all prior times $\tau < t$ affects the polarization at time t [32].

The simplification is that the generated polarization is only considered as an output effectively neglecting the effect of the electromagnetic field arising from the dipole oscillations. This is known as *first Born approximation* in contrast to the n'th Born approximation where n iterations are preformed. The process is as follows:

1. Determine input electric field
2. Calculate polarization from the electric field
3. Determine the electric field that serves as a solution with the new polarization

where step 2 and 3 are carried out n times in the n'th Born approximation [32]. The first order Born approximation therefore neglects *dipole-dipole interactions*.

In the approach where propagation along the z-direction is discretized the electric field arising from dipole oscillation in the i'th slap is considered as an input field in the i'th+1 slap.

The *First Born approximation* is a widely used model that greatly simplifies the complexity of the problems at hand, but the validity region is not well defined and it should be verified that the modes is valid for the present case [37]. This has been omitted in the following and the model is implemented through out the models presented.

4.3.1 *Second harmonic generation*

Imagine a monochromatic electric field oscillating at frequency ω_1. This field induces, according to equation (31), a second order non-linear polarization that oscillates at the double frequency, $2\omega_1$. The amplitude of the emitted electric field at the double frequency, $2\omega_1$, is proportional to the square of the amplitude of the electric field at the incoming frequency ω_1 and proportional to the susceptibility according to equation (32).

The field emitted with frequency $2\omega_1$ can then mix with the field oscillating at frequency ω_1 and emit light at frequency $3\omega_1$ but this is neglected in *the first order Born approximation*.

4.3.2 *Three wave mixing*

Second harmonic generation is a special case of the general *three wave mixing* where $\omega_2 = \omega_1$. Imagine an electric field given as the sum of two monochromatic electric fields oscillating at frequency ω_1 and ω_2 respectively, where $\omega_2 > \omega_1$. This field induces frequency components of the second order non-linear polarization at the frequencies 0, $2\omega_1$, $2\omega_2$, $\omega_1 + \omega_2$ and $\omega_2 - \omega_1$. These fields can then in turn interact with all the other frequencies but this is neglected in *the first order Born approximation*. The field at 0 is a DC field and is thus not oscillating.

4.4 SPECIAL CASES

The complex wave equation in the time domain, Equation (52), can be simplified in several situations.

4.4.1 *Quasi monochromatic wave*

Monochromatic waves consist spectrally of a single temporal frequency while non-monochromatic waves consists spectrally of a continuum of temporal frequencies. Quasi-monochromatic waves have a small temporal frequency spread centred around a strong carrier temporal frequency [15].

$$\tilde{\mathbf{E}}(z,t) = \frac{1}{2}\left[\mathbf{A}_1(z,t)e^{i(k_{0,1}z-\omega_{0,1}t)} + \mathbf{A},\mathbf{t}_1^*(z)e^{-i(k_{0,1}z-\omega_{0,1}t)}\right] \tag{53}$$

4.4.2 *Continuous wave*

A continuous wave (CW) is a wave of constant amplitude and frequency that, in principle, is of infinite temporal duration. This means that a continuous wave has no time dependence

$$\frac{\partial}{\partial t}\mathbf{E}(\mathbf{r},t) = 0 \tag{54}$$

4.4.3 *Plane wave*

A plane wave means that the electromagnetic wave has no transverse dependence meaning that

$$\nabla_\perp \mathbf{A}(\mathbf{r},t) = 0 \tag{55}$$

A plane wave is an idealized, mathematical, non physical situation since wave extending to infinity would carry infinite power. It is worth noting that a plane wave is a good approximation to Gaussian beams experiencing small confinement.

4.4.4 *Slowly Varying Envelope Approximation*

The *Slowly Varying Envelope Approximation* (SVEA) is that the envelope, $\mathbf{A}(\mathbf{r},t)$, has a slowly varying z-dependence meaning that

$$\frac{\partial^2}{\partial z^2}\mathbf{A}(\mathbf{r},t) \ll 2ik_0\frac{\partial}{\partial z}\mathbf{A}(\mathbf{r},t) \tag{56}$$

4.5 MONOCHROMATIC SPATIAL WAVE

Applying the *Slowly Varying Envelope Approximation* (56) to a monochromatic wave given by Equation (53) and inserting this into the wave equation, (52), yields

$$\left[\nabla_{\perp}^{2} + 2ik_{0,j}\frac{\partial}{\partial z}\right]\tilde{\mathbf{A}}_{j}(\mathbf{r},t) = \frac{-\omega_{0,j}^{2}}{\epsilon_{0}c^{2}}\mathbf{P}_{j}^{(2)}(\mathbf{r},\omega_{0,j})e^{-ik_{0,j}z} \tag{57}$$

In the case of no non-linear induced polarization, $\mathbf{P}_{j}^{(2)}(\mathbf{r},\omega_{0,j}) = 0$, Equation (57) reduces to the well known *paraxial wavequation*.

The solutions to the *paraxial wavequation* is known to be *Gaussian beams*, where it is assumed that the beam is sufficiently collimated along the z-axis that the *Slowly Varying Envelope Approximation* can be applied. There is no known general analytical solution to Equation (57) and this equation is therefore solved using numerical calculations.

The choice of orthogonal Gaussian basis functions used to expand the solution to the *paraxial wavequation* is usually determined by the material symmetry in the plane perpendicular to the propagation axis [29].

4.5.1 *Orthonormal basis of Laguerre-Gaussian modes*

Figure 3 illustrates the propagation of an astigmatic Gaussian beam where the beam waist in the x- and y-direction is seen to lie at different locations along the z-axis. In the case of a non-astigmatic beam with cylindrically symmetry, $l = 0$, the solutions of the *paraxial wave equation*, (57), are *Laguerre-Gaussian modes*. The monochromatic field can be expressed by the amplitude functions $A_{j}(\mathbf{r})$ and can then be expanded by the mode given by

$$\begin{aligned} u_{j,p,l}(r,\theta,z) = c_{p,l}\frac{w_{0,j}}{w_{j}(z)}\left(\frac{r\sqrt{2}}{w_{j}(z)}\right)^{l}\exp\left(-\frac{r^{2}}{w_{j}^{2}(z)}\right)L_{p}^{l}\left(\frac{2r^{2}}{w_{j}^{2}(z)}\right) \\ \cos(l\theta)\exp\left(ik_{j}\frac{r^{2}}{2R_{j}(z)}\right)\exp\left[il\theta + i(2p+l+1)\zeta(z)\right] \end{aligned} \tag{58}$$

where $p, l \geqslant 0$ and j describes the field.

L_{p}^{l} are the generalized Laguerre polynomials

$p \geqslant 0$ is the radial index

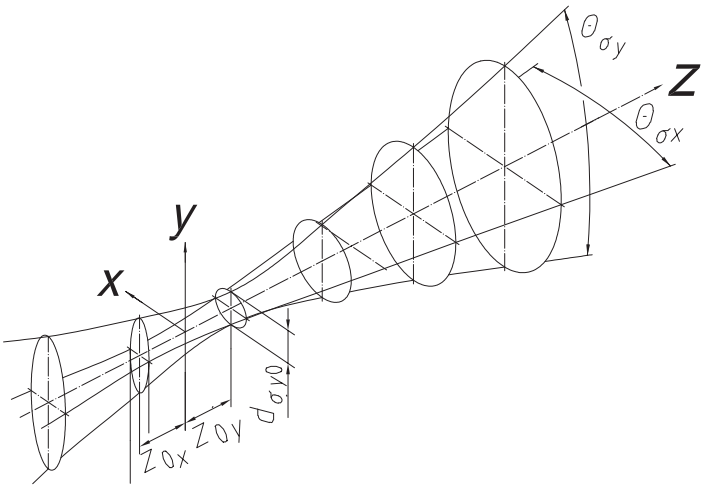


Figure 3: Illustration of the propagation of astigmatic Gaussian beam. This beam waist in the x- and y-direction is seen to lie at different locations along the z-axis. This illustration is borrowed from the ISO 11146-1:2005 standard with slight label modifications [19].

$l \geqslant 0$ is the azimuthal index

C_{lp}^{LG} is an appropriate normalization constant

$w(z)$ is the radius at which the field amplitude and intensity drop to $1/e$ and $1/e^2$ of their axial values, respectively. $w_0 \equiv w(0)$ is thus the beam waist.

$R(z)$ is the radius of curvature of the beam's wavefronts.

$\zeta(z) = \arctan\left(\frac{z}{z_R}\right)$ is the Gouy phase shift, an extra contribution to the phase that is seen in Gaussian beams.

Focusing on fields with no azimuthal dependence, $l = 0$ we get that the amplitude function is expanded as

$$A_j(r,\theta,z) = \sum_{p\geqslant 0} u_{j,p,0}(r,\theta,z) \tag{59}$$

by the basis functions

$$u_{j,p,0}(r,\theta,z) = c_{p,l}\frac{\omega_{0,j}}{\omega_j(z)}\exp\left(-\frac{r^2}{w_j^2(z)}\right)\left(\frac{2r^2}{w_j^2(z)}\right) \exp\left(ik_j\frac{r^2}{2R_j(z)}\right)\exp\left[i(2p+1)\zeta(z)\right] \tag{60}$$

When dealing with Cartesian symmetric structures, the solutions of the *paraxial wave equation*, (57), are *Hermite-Gaussian modes*, but these are omitted in the simulations presented later.

The wave vector, $\mathbf{k}$, for a Gaussian beam given by Equation (60) is perpendicular to the wavefront. The paraxial approximation is to neglect the spatial dependence of the wave vector. This is valid for small curvatures (large radius of curvature) since the wave front is then approximately a plane wave. If a Gaussian beam is highly focused the beam waist and the radius of curvate is small thereby failing the validity of the paraxial approximation.

With the paraxial approximation we write the Gaussian fields as a carrier wave given by equation (37) where the amplitude function is given by Equation (59) and (60).

4.6 SLOWLY VARYING, MONOCHROMATIC, PLANE WAVE

The *Helmholtz equation* (57) is seen to simplify for *plane waves* given by Equation (55), since they have no transverse dependence yielding the most simple equation governing the propagation of electric fields

$$2ik_{0,j}\frac{\partial}{\partial z}\tilde{\mathbf{A}}_j(\mathbf{r},t) = \frac{-\omega_{0,j}^2}{\epsilon_0 c^2}\mathbf{P}_j^{(2)}(\mathbf{r},\omega_{0,j})e^{-ik_{0,j}z} \tag{61}$$

5 PARAMETRIC FREQUENCY CONVERSION

Parametric frequency conversion is the coupling of energy between frequency components of an incoming electrical field caused by non-linear material response. In the following we are focusing on the effect of second order non-linear effects and higher order non-linear affects are ignored.

The process of second order non-linear parametric frequency conversion is of interest within current research and in industrial application where it has found use as laser sources [20], in measurement systems of the retardation induced by optical elements [10] and spectroscopy [26]. Within resent years a strong emphasis has been on the development of clinical applications for optical imaging of cells and tissues using second harmonic generation [8].

5.1 SECOND HARMONIC

Second Harmonic Generation is the generation of frequency component $2\omega_1$ by applying an electric field with frequency component ω_1 that induces a second order non-linear polarization at frequency $2\omega_1$. This process was briefly mentioned in Section 4.3.1.

5.1.1 *Slowly varying, monochromatic, plane wave*

A slowly varying, monochromatic, plane wave propagating in the z-direction given by Equation (53) will cause the second order non-linear polarization to oscillate at two frequency components, $\omega_{0,1}$ and $2\omega_{0,1}$, according to Equation (29) where we have explicitly used the first order Born approximation. We therefore look for solutions of the form

$$\begin{aligned}\tilde{\mathbf{E}}(z,t) = \frac{1}{2}\Big[&\mathbf{A}_1(z)e^{i(k_{0,1}z-\omega_{0,1}t)} + \mathbf{A}_2(z)e^{i(k_{0,2}z-\omega_{0,2}t)} \\ &+\tilde{\mathbf{A}}_1^*(\mathbf{r})e^{-i(k_{0,1}z-\omega_{0,1}t)} + \tilde{\mathbf{A}}_2^*(\mathbf{r})e^{-i(k_{0,2}z-\omega_{0,2}t)}\Big]\end{aligned} \tag{62}$$

Using the effective susceptibility definition of Equation (31) the second order non-linear polarization given by Equation (50) becomes

$$P^{(2)}(z,\omega_{0,1}) = 2\epsilon_0 2\chi^{(2)}_{eff}(\omega_{0,1};2\omega_{0,1},-\omega_{0,1}) \frac{1}{2}A_2(z)e^{ik_{0,2}z}\frac{1}{2}A_1^*(z)e^{-ik_{0,1}z} \tag{63}$$

$$P^{(2)}(z,2\omega_{0,1}) = \epsilon_0 2\chi^{(2)}_{eff}(2\omega_{0,1};\omega_{0,1},\omega_{0,1}) \frac{1}{2}A_1(z)e^{ik_{0,1}z}\frac{1}{2}A_1(z)e^{ik_{0,1}z} \tag{64}$$

Inserting the polarization given by Equation (63) and (64) into Equation (61) yields the governing equations

$$2ik_{0,1}\frac{d}{dz}A_1(z) = \frac{-\omega_{0,1}^2}{\epsilon_0 c^2}2\epsilon_0 2\chi^{(2)}_{eff}(\omega_{0,1};2\omega_{0,1},-\omega_{0,1}) \frac{1}{2}A_2(z)e^{ik_{0,2}z}\frac{1}{2}A_1^*(z)e^{-ik_{0,1}z}e^{-ik_{0,1}z} \tag{65}$$

$$2ik_{0,2}\frac{d}{dz}A_2(z) = \frac{-\omega_{0,2}^2}{\epsilon_0 c^2}\epsilon_0 2\chi^{(2)}_{eff}(2\omega_{0,1};\omega_{0,1},\omega_{0,1}) \frac{1}{2}A_1(z)e^{ik_{0,1}z}\frac{1}{2}A_1(z)e^{ik_{0,1}z}e^{-ik_{0,2}z} \tag{66}$$

Utilising now that $k_j = \frac{n(\omega_{0,j})\omega_{0,j}}{c}$, $c = \sqrt{1/\mu_0\epsilon_0}$ and rewriting $A_j(z) = \sqrt{2\eta_j\hbar\omega_{0,j}}a_j(z)$ we get

$$\sqrt{2\eta_1\hbar\omega_{0,1}}\frac{\partial}{\partial z}a_1(z) = \frac{i\omega_{0,1}}{n(\omega_{0,1})c}2\chi^{(2)}_{eff}(\omega_{0,1};2\omega_{0,1},-\omega_{0,1}) \frac{1}{2}\sqrt{2\eta_2\hbar\omega_{0,2}}a_2(z)\frac{1}{2}\sqrt{2\eta_1\hbar\omega_{0,1}}a_1^*(z)e^{i\Delta kz} \tag{67}$$

$$\sqrt{2\eta_2\hbar\omega_{0,2}}\frac{\partial}{\partial z}a_2(z) = \frac{i\omega_{0,2}}{n(\omega_{0,2})c}\chi^{(2)}_{eff}(2\omega_{0,1};\omega_{0,1},\omega_{0,1}) \frac{1}{2}\sqrt{2\eta_1\hbar\omega_{0,1}}a_1(z)\frac{1}{2}\sqrt{2\eta_1\hbar\omega_{0,1}}a_1(z)e^{-i\Delta kz} \tag{68}$$

where we rewrote $\Delta k = k_{0,2} - 2k_{0,1}$. Using that $\eta_j^2 = \frac{\mu_0}{\epsilon_0}\frac{1}{n(\omega_{0,j})^2}$ we get

$$\frac{d}{dz}a_1(\mathbf{r},t) = i\epsilon_0\omega_{0,1}\eta_1\chi_{eff}\frac{1}{2}\sqrt{\eta_2\hbar\omega_{0,2}}a_2(z)a_1^*(z)e^{i\Delta kz} \tag{69}$$

$$\frac{d}{dz}a_2(\mathbf{r},t) = i\epsilon_0\omega_{0,2}\eta_2\chi^{(2)}_{eff}\frac{1}{4}\sqrt{\eta_1\hbar\omega_{0,1}}a_1(z)a_1(z)e^{-i\Delta kz} \tag{70}$$

where it is assumed that $\chi^{(2)}_{eff} = \chi^{(2)}_{eff}(2\omega_{0,1};\omega_{0,1}) = \chi^{(2)}_{eff}(\omega_{0,1};2\omega_{0,1},-\omega_{0,1})$. Introducing the gain parameter $g = \epsilon_0\chi^{(2)}_{eff}\sqrt{\frac{1}{2}\eta^3\omega^3_{0,1}\hbar}$ and rewriting $\eta^3 = \eta_1\eta_1\eta_2$ yields the final equations

$$\frac{d}{dz}a_1(z) = i2ga_2(z)a_1^*(z)e^{i\Delta kz} \tag{71}$$

$$\frac{d}{dz}a_2(z) = iga_1(z)a_1(z)e^{-i\Delta kz} \tag{72}$$

Note that we went from a vector notation of $\mathbf{A}$ to a scalar A notation. This is justified by the phase match mentioned in Section 3.3.1 and described in more detail in Section 5.4, since all interacting fields are polarized along the extraordinary axis of the crystal, the y-axis according to Figure 1.

5.2 THREE WAVE MIXING

We imagine now that the input field consists of two carrier frequencies, $\omega_{0,1}$ and $\omega_{0,2}$ where $\omega_{0,2} > \omega_{0,1}$. The induced polarization will then oscillate at the frequency $\omega_{0,1} + \omega_{0,2}$ acting as a source term and creating a field with the frequency component $\omega_{0,3} = \omega_{0,1} + \omega_{0,2}$. This process is known as *sum frequency generation* in contrast to *difference frequency generation* that is the generation of frequency $\omega_{0,3} = \omega_{0,2} - \omega_{0,1}$. The two processes is illustrated in Figure 4 where

$$\omega_3 > \omega_2 > \omega_1 \tag{73}$$

It is customary to name the three frequency components *signal*, *idler*

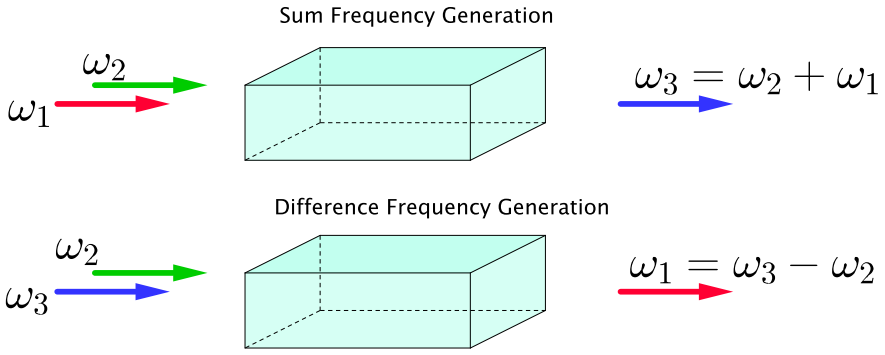


Figure 4: Illustration of three wave mixing processes. Sum frequency generation is the process where two incoming fields, ω_1 and ω_2 generates a field of the sum frequency $\omega_3 = \omega_1 + \omega_2$. Difference Frequency Generation is the process of two incoming fields, ω_1 and ω_3 generates a field of the difference frequency $\omega_2 = \omega_3 - \omega_1$.

and *pump* where the pump is the high energy frequency, $\omega_{0,3}$, while the signal is the low energy frequency component, $\omega_{0,1}$, leaving the idler as $\omega_{0,2}$.

5.2.1 *Slowly varying, monochromatic, plane wave*

For the case of slowly varying, monochromatic, plane wave the governing equations can be derived as presented in Section 5.1.1.

The induced polarization is given by

$$P^{(2)}(z,\omega_{0,1}) = \epsilon_0 4\chi^{(2)}_{eff}(\omega_{0,1};\omega_{0,3},-\omega_{0,2}) \frac{1}{2}A_3(z)e^{-ik_{0,3}z}\frac{1}{2}A_2^*(z)e^{ik_{0,2}z} \tag{74}$$

$$P^{(2)}(z,\omega_{0,2}) = \epsilon_0 4\chi^{(2)}_{eff}(\omega_{0,2};\omega_{0,3},-\omega_{0,1}) \frac{1}{2}A_3(z)e^{-ik_{0,3}z}\frac{1}{2}A_1^*(z)e^{ik_{0,1}z} \tag{75}$$

$$P^{(2)}(z,\omega_{0,3}) = \epsilon_0 4\chi^{(2)}_{eff}(2\omega_{0,1};\omega_{0,1},\omega_{0,1}) \frac{1}{2}A_1(z)e^{ik_{0,1}z}\frac{1}{2}A_2(z)e^{ik_{0,2}z} \tag{76}$$

Insertion of Equation (74), (75) and (76) into Equation (61) yields the equations governing *sum frequency generation* and *difference frequency generation*

$$\frac{d}{dz}a_1(z) = iga_3(z)a_2^*(z)e^{i\Delta kz} \tag{77}$$

$$\frac{d}{dz}a_2(z) = iga_3(z)a_1^*(z)e^{i\Delta kz} \tag{78}$$

$$\frac{d}{dz}a_3(z) = iga_1(z)a_2(z)e^{-i\Delta kz} \tag{79}$$

where

$$\Delta k = k_{0,3} - k_{0,2} - k_{0,1} \tag{80}$$

$$g = \epsilon_0\chi^{(2)}_{eff}\sqrt{\frac{1}{2}\eta^3\hbar\omega_1\omega_2\omega_3} \tag{81}$$

$$\eta^3 = \frac{\mu_0}{\epsilon_0}\frac{1}{n(\omega_{0,1})n(\omega_{0,2})n(\omega_{0,3})} \tag{82}$$

Se Appendix C for the full derivation.

For the case of *sum frequency generation* then a field of frequency ω_1 and a field of frequency ω_2 induces a field at frequency $\omega_3 = \omega_1 + \omega_2$. For the case of *difference frequency generation* then a field of frequency ω_3 and a field of frequency ω_2 induces a field at frequency $\omega_3 - \omega_2 = \omega_1$.

5.3 DEPLETION

The *weak coupling approximation* is the assumption that the pump wave only transfers a small fractions of energy to the idler and signal meaning that the amplitude is approximately constant along the crystal, $\frac{da_{\omega_1}}{dz} = 0$. Most cases significantly simplify if depletion can be ignored since the coupled differential equations can be decoupled.

The energy transfer is a significant fraction of the pump power if the interacting fields are of high intensity and depletions has to be accounted for. For pulses the average power is low but the peak power can be extremely high and depletion is therefore essential for most pulses.

The effect of accounting for depletion is analysed in different cases by running simulations with and without depletion implemented.

5.4 PHASE MATCH

The phase mismatch parameter $\Delta k = k_{0,3} - k_{0,2} - k_{0,1}$ is the difference in wave vector between the interacting fields. For the degenerate case of Second Harmonic Generation the phase match condition becomes $\Delta k = k_{2\omega} - 2k_{\omega}$ which in physical, non-dispersion materials is automatically fulfilled. In dispersive materials however the refractive index depends on frequency and the phase match condition is not automatically fulfilled. The phase mismatch essentially means that the relative phase of the fields differs along the crystal and the field generated in one slap is not in phase with the field generated in the next slap. Figure 5 illustrates the amplitude changes in incremental elements. The highest conversion is with phase match.

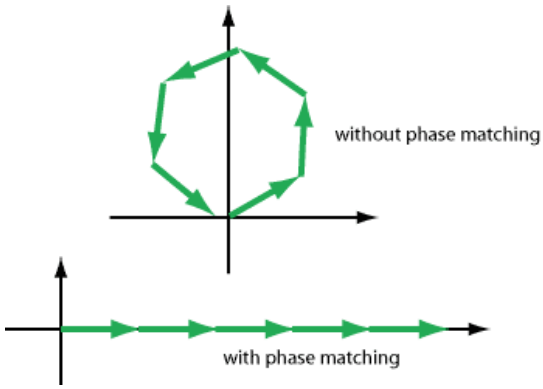


Figure 5: Amplitude changes contributed from incremental elements at different position in the crystal. Only with phase match do they add constructively. Borrowed from [28]

Since the complex wave function given by Equation (37) oscillates as $\omega t - kz$ the frequency matching condition $\omega_3 = \omega_1 + \omega_2$ along with the phase match condition, $\Delta k = k_{0,3} - k_{0,2} - k_{0,1}$, ensures both temporal and spatial phase matching of the interacting beams.

Phase match essentially ensures the proper phase relation between the interacting waves such that the non-linear effect arising from different locations within the crystal coherently adds up creating a macroscopic non-linear effect. This is discussed in more detail in Section 5.4.

5.4.1 *Birefringent phase match*

Birefringent Phase Matching is the technique of achieving phase match by exploiting the birefringence of the material. Many materials, including MgO:PPLN crystals, experience birefringence, meaning that the refractive index is polarization dependent. MgO:PPLN crystals are uniaxial crystals meaning that the refractive index along one axis, namely the extraordinary axis, differer from that along the two other axis, namely the ordinary axis.

Polarizing the interacting fields in different directions enables phase matching by tuning the temperature of the crystal since the refractive index of both the extraordinary and the ordinary axis is temperature dependent, as will be mentioned in Section 5.4.3. Rotating the crystal for a fixed temperature also enables to find the angle such that the birefringent compensates exactly the dispersion [32].

The largest second order susceptibility tensor element is, as mentioned in Section 3.3.1, the χ_{ccc} element that couples fields all polarized along the extraordinary axis. Since the polarization of the interacting field cannot be in the same direction using *Birefringent phase match* only the smaller susceptibility tensor elements can be used, effectively lowering the non-linear coupling.

5.4.2 *Quasi-Phase Matching*

Quasi Phase Matching is the technique of achieving phase matching using a crystal with spatially modulated non-linear properties. MgO:PPLN crystals are non-centrosymmetric meaning that an inversion of the crystal changes sign of the susceptibility. Introducing a periodically inversion along the crystals creates *domains* having similar susceptibility but inverted. Back conversion due to phase difference is avoided if the period of the domain walls, Λ, (domain thickness is then $\Lambda/2$) equals an odd integer multiple of the coherence length, L_c.

Using Fourier theory it can be shown that a periodically polling period of an odd integer times the coherence length, $\Lambda = mL_c$, is equivalent to adding a term $k_{qpm} = \frac{2\pi}{\Lambda}$ to the phase match condition, $\Delta k = k_{0,3} - k_{0,2} - k_{0,1} \pm k_{qpm}$ and multiplying the non-linear coefficient with a factor of $2/m\pi$ [32]. A periodically polling period of an even integer times the coherence length has no effect since the inversion is spatially at points where the fields completely cancel.

5.4.3 *Sellmeier equations*

A detailed knowledge of the frequency dependence of the refractive index is needed in order to fulfil either the *birefringent phase match* or the *quasi phase match*. The *magnesium doped (5%) Periodically Poled Lithium Niobate* (MgO:PPLN) crystals mentioned in Section 3.3.1 and simulated in this project is a uniaxial crystal meaning that it has one extraordinary axis and two ordinary. The empiric Sellmeier equations describe the refractive index as function of temperature and frequency. The largest second order non-linearity is, as mentioned in Section , achieved by the interaction of field all polarized along the extraordinary axis of the crystal. *Quasi-Phase Matching* enables to utilize the largest second order non-linearity while *Birefringent phase matching* cannot utilize this since all interacting field are not polarized in the same direction.

The Sellmeier equation for the refractive index along the extraordinary axis is [17]

$$n_e^2 = a_1 + b_1 f + \frac{a_2 + b_2 f}{\lambda^2 - (a_3 + b_3 f)^2} + \frac{a_4 + b_4 f}{\lambda^2 - a_5^2} - a_6 \lambda^2 \tag{83}$$

where

$$f = (T - 24.5\,^\circ C)(T + 570.82\,^\circ C) \tag{84}$$

The coefficients of the Sellmeier equation (83) is determined experimentally and the values used are seen in Table 1 [17].

5.4.4 *Solve* $\Delta k = 0$

In the following we have for different pooling periods solved for the temperature satisfying the phase match condition

$$\Delta k = k_3 - k_2 - k_1 \pm k_{qpm} = 0 \tag{85}$$

The wave vectors have been determined by the Sellmeier equation from Section 5.4.3 and the wavelengths satisfying phase match have

SYMBOL	n_e	n_o
a_1	5.756	5.653
a_2	0.0983	0.1185
a_3	0.2020	0.2091
a_4	189.32	89.61
a_5	12.52	10.85
a_6	1.32×10^{-2}	1.97×10^{-2}
b_1	2.860×10^{-6}	7.941×10^{-7}
b_2	4.700×10^{-8}	3.134×10^{-8}
b_3	6.113×10^{-8}	-4.641×10^{-9}
b_4	1.516×10^{-4}	-2.188×10^{-6}

Table 1: Experimentally determined constant that fits the Sellmeier Equation (83) to measured values [17]

been found for temperatures ranging from 25 °C to 150 °C for different pooling periods. The thermal expansion of the QPM structure is neglected. For a 1064 nm pump the solution to the phase match condition, given by Equation (85), is plotted in Figure 6 for the case of three wave mixing (SFG and DFG) while the degenerate case of second harmonic generation is seen in Figure 7.

5.4.5 *Bandwidth*

Phase matching is of great importance, as mentioned above, since the conversion efficiency decrease significantly as the phase mismatch increases.

An analytical expression of the phase mismatch effect on wave-mixing efficiency can be shown in the simple case of monochromatic, CW, plane wave interaction in the non-depleted case explained in Section 5.3. Starting from Equation (77) and assuming the pump, a_3, and idler, a_2 undepleted the equation becomes

$$\frac{d}{dz}a_1(z) = iga_3a_2^* e^{i\Delta kz} \tag{86}$$

having the the solution

$$a_1(z) = \int_0^L iga_3a_2^* e^{i\Delta kz}\, dz \tag{87}$$

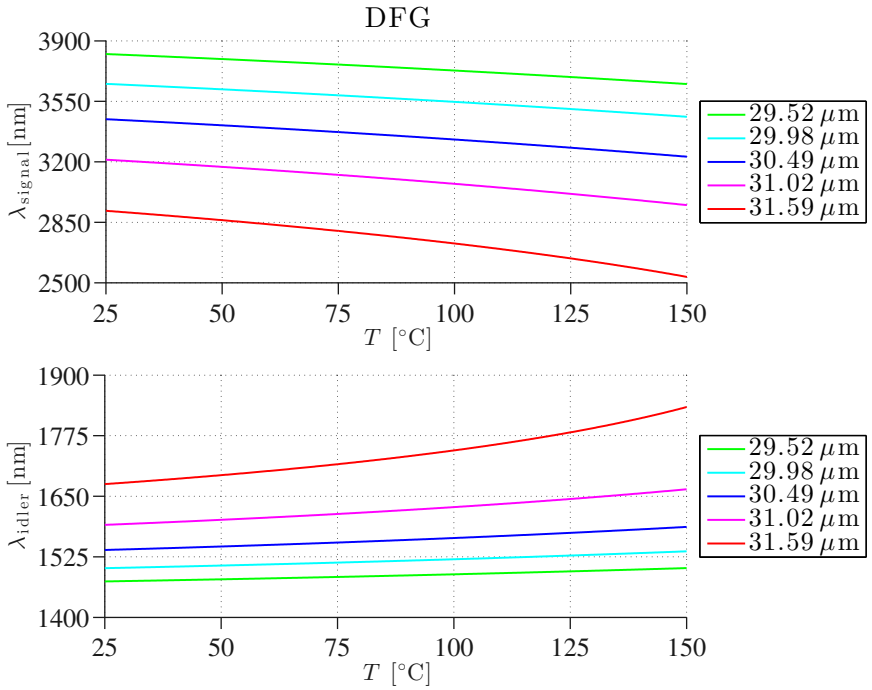


Figure 6: Wavelength satisfying Sum/Difference Frequency Conversion phase match condition as function of temperature for different pooling periodes.

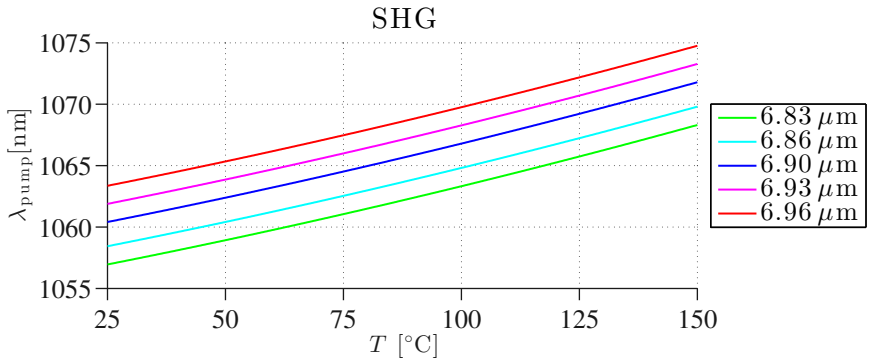


Figure 7: Wavelength satisfying Second Harmonic Generation phase match condition as function of temperature for different pooling periodes.

The relative intensity is then given as

$$\frac{I(L)}{I(0)} \propto \frac{|a_1(L|^2}{|a_1(0)|^2} \propto L^2 \frac{\sin^2(\Delta kL/2)}{(\Delta kL/2)^2} = [L \cdot \mathrm{sinc}(\Delta kL/2)]^2 \tag{88}$$

It follows from Equation (88) that in the presence of a phase mismatch the intensity of the generated signal is reduced by a factor of $\mathrm{sinc}^2(\Delta kL/2)$. The intensity is thus largest for phase match, $\Delta k = 0$ and decreases as Δk increases.

The *coherence length*, L_c is a measure of the length within which the addition of field is constructive. It is defined as [32]

$$L_c = \frac{2\pi}{|\Delta k|} \tag{89}$$

The *acceptance bandwidth* of the phase mismatch is defined as the phase mismatched, Δk for which the product of the phase mismatch and the finite interaction length, L, giving [32]

$$|\Delta k| \leqslant \frac{2\pi}{L} \tag{90}$$

5.5 PULSE PROPAGATION

The propagation of pulses is of great interest since pulses have a wide use both in research and commercial application. Pulsed lasers can reach extremely large peak powers while still maintaining a low average power. The non-linear effect can because of the extremely high peak power be of great significance.

It was already shown in Section 4.2 that a pulse consists of a continuum of frequencies around the carrier frequency in contrast to monochromatic CW fields that consists solely of the carrier frequency. Propagating a pulse in a dispersive material thus affects the pulse shape since each frequency experiences a different refractive index. This effect is called *chirp* where each frequency component is phase shifted differently.

For a simplification we consider only *plane wave* pulses since a full analysis of temporal and spatial pulses is beyond the scope of this project.

5.5.1 *Retarded time frame*

Starting from Equation (52), neglecting the transverse dependence, using the *Slowly Varying Envelope Approximation* and multiplying with $-i/2k_{0,j}$ we get

$$\left[\frac{\partial}{\partial z} + k_{1,j}\frac{\partial}{\partial t} + i\frac{k_{1,j}^2}{2k_{0,j}}\frac{\partial^2}{\partial t^2}\right]\tilde{\mathbf{A}}_j(\mathbf{r},t) = \frac{i\omega_{0,j}^2}{2k_{0,j}\epsilon_0 c^2}\mathbf{P}_j^{(2)}(\mathbf{r},\omega_{0,j})e^{-ik_{0,j}z} \tag{91}$$

In order to simplify the expression we introduce a retarded time frame by introducing

$$\tau = t - k_m z \tag{92}$$
$$z' = z \tag{93}$$

where k_m is the largest wave vector of the three interacting waves. This means that the wave travelling with wave vector k_m is stationary in the newly introduced coordinates. With this change of variables we have

$$\begin{aligned}\frac{\partial}{\partial z}\mathbf{A}_j(\tau,z') &= \frac{\partial}{\partial z'}\mathbf{A}_j(\tau,z')\frac{\partial z'}{\partial z} + \frac{\partial}{\partial \tau}\mathbf{A}_j(\tau,z')\frac{\partial \tau}{\partial z}\\ &= \frac{\partial}{\partial z'}\mathbf{A}_j(\tau,z') - k_m\frac{\partial}{\partial \tau}\mathbf{A}_j(\tau,z')\end{aligned} \tag{94}$$

$$\begin{aligned}\frac{\partial}{\partial t}\mathbf{A}_j(\tau,z') &= \frac{\partial}{\partial z'}\mathbf{A}_j(\tau,z')\frac{\partial z'}{\partial t} + \frac{\partial}{\partial \tau}\mathbf{A}_j(\tau,z')\frac{\partial \tau}{\partial t}\\ &= \frac{\partial}{\partial \tau}\mathbf{A}_j(\tau,z')\end{aligned} \tag{95}$$

$$\begin{aligned}\frac{\partial^2}{\partial t^2}\mathbf{A}_j(\tau,z') &= \frac{\partial}{\partial t}\left[\frac{\partial}{\partial t}\mathbf{A}_j(\tau,z')\right] = \frac{\partial}{\partial t}\left[\frac{\partial}{\partial \tau}\mathbf{A}_j(\tau,z')\right]\\ &= \frac{\partial^2}{\partial \tau^2}\mathbf{A}_j(\tau,z')\end{aligned} \tag{96}$$

Inserting this into Equation (91) gives

$$\begin{aligned}&\left[\frac{\partial}{\partial z'} + (k_{0,j} - k_m)\frac{\partial}{\partial \tau} + i\alpha_j\frac{\partial^2}{\partial \tau^2}\right]\mathbf{A}_j(\tau,z')\\ &\qquad = \frac{i\omega_{0,j}^2}{2k_{0,j}\epsilon_0 c^2}\mathbf{P}_j^{(2)}(z',\omega_{0,j})e^{-ik_{0,j}z'}\end{aligned} \tag{97}$$

where $\alpha_j = \frac{k_{1,j}^2}{2k_{0,j}}$ and the polarization is given by the well known equations (63) and (64) for *Second Harmonic Generation* and equation (74), (75) and (76) for *Difference Frequency Generation* and *Sum Frequency Generation*. The first term on the left hand side of Equation (97)

describes the propagation of the pulse. The second term describes dispersion (remember that $k = 1/v_g$) and the third term on the left hand side describes the group velocity dispersion. The third term is usually small and can be neglected for ns and ps pulses [34] yelding the system of coupled first order partial differential equation

$$\left[\frac{\partial}{\partial z'} + (k_{0,1} - k_m)\frac{\partial}{\partial \tau}\right] A_1(\tau, z') = \frac{i\omega_{0,1}^2}{2k_{0,1}c^2} e^{-ik_{0,1}z'}$$
$$4\chi^{(2)}_{eff}(\omega_{0,1}; -\omega_{0,3}, \omega_{0,2})\frac{1}{2}A_3(z)e^{ik_{0,3}z}\frac{1}{2}A_2^*(z)e^{-ik_{0,2}z} \tag{98}$$

$$\left[\frac{\partial}{\partial z'} + (k_{0,2} - k_m)\frac{\partial}{\partial \tau}\right] A_2(\tau, z') = \frac{i\omega_{0,2}^2}{2k_{0,2}c^2} e^{-ik_{0,2}z'}$$
$$4\chi^{(2)}_{eff}(\omega_{0,2}; -\omega_{0,3}, \omega_{0,1})\frac{1}{2}A_3(z)e^{ik_{0,3}z}\frac{1}{2}A_1^*(z)e^{-ik_{0,1}z} \tag{99}$$

$$\left[\frac{\partial}{\partial z'} + (k_{0,3} - k_m)\frac{\partial}{\partial \tau}\right] A_3(\tau, z') = \frac{i\omega_{0,3}^2}{2k_{0,3}c^2} e^{-ik_{0,3}z'}$$
$$4\chi^{(2)}_{eff}(2\omega_{0,1}; \omega_{0,1}, \omega_{0,1})\frac{1}{2}A_1(z)e^{ik_{0,1}z}\frac{1}{2}A_2(z)e^{ik_{0,2}z} \tag{100}$$

Simplifying the equations above by recalling that $k_j = \frac{n(\omega_{0,j})\omega_{0,j}}{c}$ and $\Delta k = k_{0,3} - k_{0,2} - k_{0,1}$ we get the final equations

$$\left[\frac{\partial}{\partial z'} + (k_{0,1} - k_m)\frac{\partial}{\partial \tau}\right] A_1(\tau, z') = \frac{i\omega_{0,1}}{2n(\omega_{0,1})c}\chi^{(2)}_{eff}A_3^*(z)A_2(z)e^{i\Delta k z'} \tag{101}$$

$$\left[\frac{\partial}{\partial z'} + (k_{0,2} - k_m)\frac{\partial}{\partial \tau}\right] A_2(\tau, z') = \frac{i\omega_{0,2}}{2n(\omega_{0,1})c}\chi^{(2)}_{eff}A_3^*(z)A_1(z)e^{i\Delta k z'} \tag{102}$$

$$\left[\frac{\partial}{\partial z'} + (k_{0,3} - k_m)\frac{\partial}{\partial \tau}\right] A_3(\tau, z') = \frac{i\omega_{0,3}}{2n(\omega_{0,1})c}\chi^{(2)}_{eff}A_1(z)A_2(z)e^{-i\Delta k z'} \tag{103}$$

Notice that we went from a vector notation to a scalar notation. The argumentation is the same as in Section 5.1.1 and we assume that all field are polarized along the y-direction according to laboratory coordinates as defined in Figure 1.

5.5.2 *Split-step Fourier method*

Split-step Fourier method is one method to solve the system of first order partial differential Equations (101), (102) and (103).

The process of the *Split-step Fourier Method* is to first determine the solution to the equation without the non-linear polarization and

next adding the contribution from the non-linear polarization. The first step is not as simple as the CW case previously discussed. First we notice that Fourier transforming Equation (97) without the non-linear polarization and neglecting the second order derivative of τ yields [34]

$$\left[\frac{\partial}{\partial z'} + i\Delta\omega_{0,j}\left(k_{0,j} - k_m\right)\right] A_j(\Delta\omega_{0,j}, z') = 0 \tag{104}$$

where

$$A_j(\Delta\omega_{0,j}, z') = \int_{-\infty}^{\infty} A_j(\tau, z') e^{i\Delta\omega_{0,j}\tau}\, d\tau \tag{105}$$

It is seen that propagation of Δz in frequency domain is equivalent to shifting the phase of each frequency shift, $\Delta\omega_{0,j}$, by an amount $\Delta z\Delta\omega_{0,j}\left(k_{0,j} - k_m\right)$ [34]. Each frequency is therefore added a different phase. The next step in the *Split-step Fourier Method* is to Fourier transform back to time domain and adding the non-linear contribution.

5.6 QUANTUM FLUCTUATION

Second harmonic generation depends only on the presence of the pump frequency and the process of generating the double frequency initializes even without the presence of the double second harmonic wave. Three wave mixing on the other hand depends on the presence of both the pump and either the idler or the signal waves. the *Parametric frequency conversion* process thus only initializes if either of the latter two fields are present. This is clearly seen in Equation (77), (78) and (79) where there is no conversion if $a_2(0) = a_3(0)$.

It is experimentally seen that the parametric frequency conversion initializes even without seed in contrast to the description above and Equation (77), (78) and (79). The reason for the spontaneous parametric frequency conversion is the quantum fluctuations present at all frequencies act as seed [9]. The quantum fluctuations are of random nature and should in a fully quantum mechanical description be treated as such. This would mean that the result should be given by an ensemble of simulations each initialized with a random seed frequency and random phase. Such a description is outside the scope of this project and the quantum fluctuations are approximated with a well defined seed at a fixed frequency and with a fixed phase.

5.6.1 *Estimating the quantum noise level*

If we look at the CW, plane wave equations (74), (75) and (76), in the case of no depletions and complete phase match we have the coupled ordinary differential equations

$$\frac{d}{dz}a_1(z) = -igA_2(z)^* a_3(0) \tag{106}$$

$$\frac{d}{dz}a_2(z) = -igA_1(z)^* a_3(0) \tag{107}$$

If we assume no seed on the idler wave, $a_2(0) = 0$, the solutions are of the form [32]

$$\phi_1(z) = \phi_1(0)\cosh(gA_3(0)z) \tag{108}$$

$$\phi_2(z) = \phi_1(0)\sinh(gA_3(0)z) \tag{109}$$

$$\tag{110}$$

where ϕ is the photon flux $\phi_j(z) = \frac{I_j(z)}{\hbar\omega_j} = |a_j(z)|^2$. If then the pump power and the output signal power (at $z = L$) is known we can estimate the quantum fluctuations as

$$P_{quantum} \equiv P_1(0) = \frac{P_1(L)}{\cosh(gA_3(0)z)} \tag{111}$$

where g is the gain parameter and $A_{pump}(z)$ it the pump photon flux as defined in Section 5.1.1.

This approach is obvious only an estimate and should be subject to a much more detailed analysis but this is not the subject of this thesis. A full analysis of the quantum noise would be of statistical nature and include that both the phase and the amplitude of the quantum noise is random. Several groups has presented work regarding the effects of quantum noise in parametric frequency conversion processes [1, 9].

Using the results presented later in Section 9.3 we estimate that an average pump power of 175 mW for a 7 ns pulse at a repetition rate of 20 µs gives a pump peak power of approximate 1.1 mW. This pump creates a signal pulse of peak power 180 mW. Inserting this into Equation 111 estimates the quantum fluctuations to 1.1 mW.

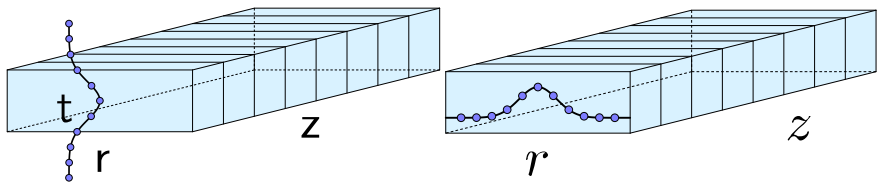
This seams rather high in comparison to the values described in other papers where it is proposed that one uses a quantum noise level of half a photon in each supported mode [34]. This has not been tested but would be a natural next step in order to simulate DFG arising from quantum noise fluctuations.

Part II

SIMULATIONS

6 SIMULATION

The approach for simulating pulses is different than that for simulating Gaussian beams as discussed earlier. For both the case of pulses and Gaussain beams the crystal is divided into slaps along the z-direction and the electric field is calculated in each of these slaps. Pulses are discretized on a time vector where it is assumed that there is no transverse dependence of the electric field, see Figure 8a, while Gaussian pulses are discretized on a radial vector describing the transverse dependence of the electric field while no time dependence of the field is assumed (other than the time dependence of the carrier waves as mentioned in Section 4.2), see Figure 8b.



(a) (b)

Figure 8: Illustration of the discretization of (a) time and (b) transverse spatial part of the E-field along the crystal.

6.1 CONVERGENCE

The accuracy of numerical simulations depend critically on the number of discrete points in which the equations are explicitly calculated. As the number of discrete points is increased towards the limit of infinite number of points, the step size goes towards zero and the result converges towards the true solution. In reality the number of points is limited by the calculation power of the computers used. It is desirable to archive a result that is within gives accuracy of the real solution while having as small computation time as possible.

In the following the effect of the step size and the effect of introducing an effective QPM structure is analysed.

6.1.1 *Step size*

The usual approach to numerically solve a set of ordinary differential equation is using *Runge–Kutta methods* and a widely used and extremely optimized algorithm is the *Runge–Kutta 45 method*. Usual implementations of the *Runge–Kutta 45 method* uses *adaptive step-sizes* where the step-size is adapting to the variation of the function. In this approach a *relative* and a *absolute* tolerance is defined that then determines the step-size.

In the case of *fixed step-size* where no tolerances are defined then the validity of the calculated values must be verified. Inaccurate or wrong results occur if the step-size is large in a region where the function varies rapidly. Small step-sizes in regions of small variation on the contrary give only slightly more precise values while the computational time increases.

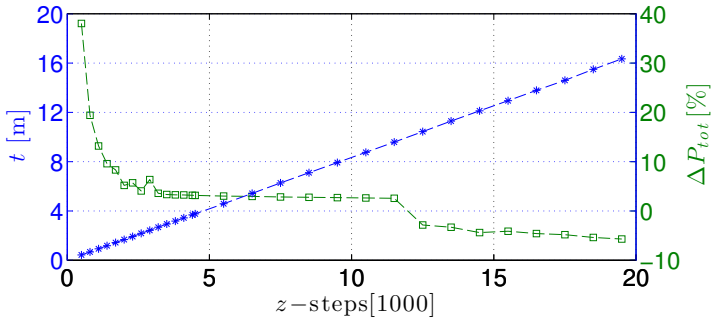


Figure 9: Computational time, t, and the maximal loss in power along the crystal, ΔP, as function of the number of steps in the z-direction. Simulated using 5003 r-steps and a pump power of 1 kW.

Figure 9 illustrates how the number of steps in the z-direction affects the computation time, t, and the maximal loss in power along the crystal, ΔP while the effect of the number of steps in the r-direction is seen in Figure 10. The simulation used is the Gaussian basis model described in Section 6.5 using a QPM domain inversion structure as will be discussed in Section 6.1.2.

It is clearly seen from Figure 9 that a small number of z-steps gives a large numerical error and hence a large power/gain. The relatively high loss in power for a high number of z-steps probably originate from the QPM structure used for the simulations. The domain inversion is only precise if a high number of steps within each domain region of thickness $\approx 30\,\mu m$ while the stepsize in the case of 20.000 z-steps is $2\,\mu m$. An even higher number of z-steps would correct this

error but increase the computation time significantly. Instead the effective QPM structure is implemented as described below.

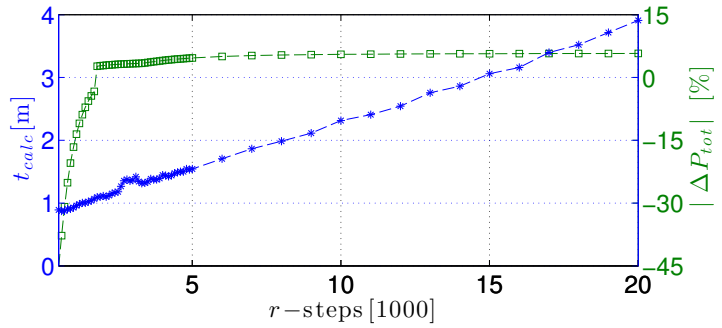


Figure 10: Computational time, t, and the maximal loss in power along the crystal, ΔP, as function of the number of steps in the r-direction. Simulated using 2500 z-steps and a pump power of 1 kW.

6.1.2 *QPM domain structure*

The *Quarter Phase Matching* approach of domain inversion, described in Section 5.4.2, is implemented by changing the sign of $\chi^{(2)}$ in neighbouring domains. The domains are a few μm wide and should contain enough steps that a converging result is simulated. This means that an extremely high number of z-steps is required. One way of avoiding the high amount of steps is to set $\Delta k = 0$ and replacing the domain inversion with an effective $\chi^{(2)}_{eff} = \frac{2}{m\pi}\chi^{(2)}$ where m is the QPM order [31]. By introducing the effective QPM structure a much lower number of z-steps is needed for the simulation to converge.

Figure 11 shows plane wave simulations of 1st, 2nd and 3rd order QPM structures using both a QPM domain inversion and using an effective susceptibility. It is seen that the simulation using an effective susceptibility approximates the QPM domain inversion approach very well even with much smaller step size.

Figure 12 shows a simulation of a plane wave, CW fields made with domain inversion using Equation (77), (78) and (79), see Figure 12a and 12b, and with the use of an effective susceptibility, Figure 12c and 12d using different *absolute tolerances* for the *Runge-Kutta 45* solver. It is clearly seen that the tolerances, and hence the z-step size, is of great significance for the case of domain inversion while the results are unaffected at the same tolerances using an effective susceptibility. With this argumentation the effective susceptibility approach is implemented in all the simulation presented in this thesis.

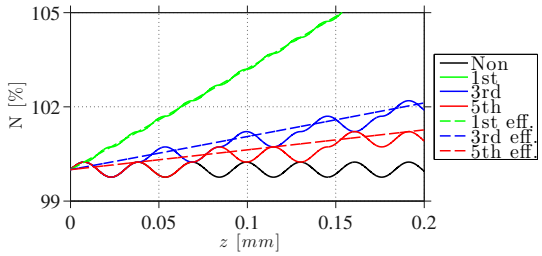


Figure 11: Comparing plane wave simulations using QPM domain inversion to the approach of an effective susceptibility. A 1st, 2nd and 3rd order WPM stucture is analysed and the number of signal photons relative to the seed photons is plotted as function of distance within the crystal.

Figure 13 compares the power output at the crystal for the simulations presented in Figure 12. From this it is easy to see that the result starts to converge for a QPM domain structure for tolerances of $\approx 1 \times 10^{-4}$ while the result converges for an effective susceptibility method already for tolerances $\approx 1 \times 10^{-2}$. The calculation time for the simulations is also significantly decreased using the effective susceptibility method. The calculation time for the simulation used to produce Figure 13a was ≈ 9 min while the calculation time used to produce Figure 13b was only ≈ 12 s.

The method of an effective QPM structure has been used in the following simulation, meaning that an effective second order nonlinear susceptibility of $\frac{2}{\pi}\chi^{(2)}_{\mathrm{eff}}$ is used while the phase mismatch is set to zero, $\Delta k = 0$.

The simulations used to create Figure 12 and 13 uses a pump power of 16 kW, a seed of 20 mW and an effective mode area of $\pi(40\,\mu\mathrm{m})^2$. The Matlab script is seen in Appendix F.

6.2 MATLAB

Simulations are written in *Matlab R2013a* and divided into subscripts whenever possible. The simulations are run on DTU Computing Center's *HPC Computer Cluster*. For further information see [11]. *Matrix calculations* have been implemented widely in order to minimize the computation time[1] but has been omitted intentionally for ease of understanding at selected sections in the simulation code.

1 Matlab preforms *matrix operations* much faster than *loop operations* [22].

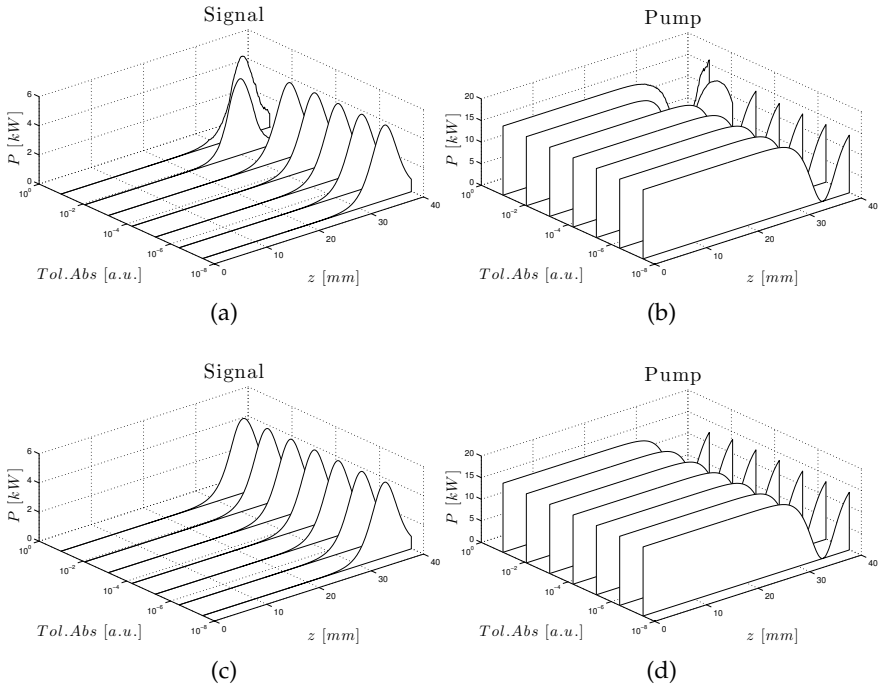


Figure 12: Simulations of DFG for monochromatic, CW, plane waves. Propagation of the signal and pump is plotted for different *absolute tolerances*. It is seen that for (a) (b) QPM domain structure the tolerances must be extremely rigid to get a precision like (c) (d) using an effective non-linear susceptibility and setting $\Delta k = 0$.

Structures are the most common way to save heterogeneous data[2] in Matlab and group data to a parent structure [23]. In the following all parameters are found in the structure *Param* while all quantities associated with the signal (Electric field, irradiance, output power etc.) is found in the structure *Signal*. One obtains the irradiance (I) of the signal using the matlab command

```
Signal.I
```

6.3 NUMERICAL VALUES

A number of constants is used in the simulations presented. Table 2 shows the constants used.

2 *Heterogeneous data*: Data that is not of the same type and of same length, e.g. *strings*, *vectors*, *matrices*, *integers*, *doubles* and many other formats.

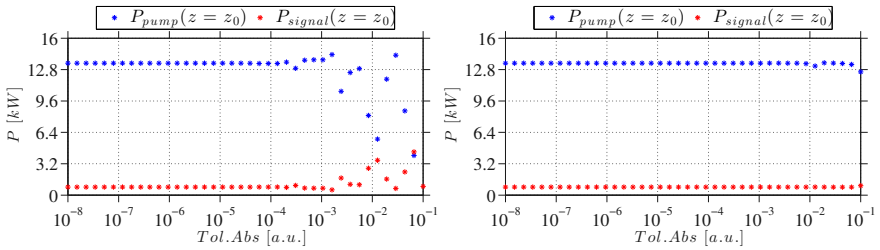


Figure 13: Simulations of DFG for monochromatic, CW, plane waves. The power output at the end of the crystal for (a) (b) a QPM domain structure and (c) (d) using an effective non-linear susceptibility and setting $\Delta k = 0$.

SYMBOL	NAME	VALUE
c	Speed of light	$299\,792\,458\,\mathrm{m\,s^{-1}}$
$\hbar$	Reduced Planck constant	$1.054\,571\,73 \times 10^{-34}\,\mathrm{J\,s}$
ϵ_0	Vacuum permittivity	$8.854\,187\,817 \times 10^{-12}\,\mathrm{F\,m^{-1}}$
μ_0	Vacuum permeability	$4\pi \times 10^{-7}\,\mathrm{V\,s\,A^{-1}\,m^{-1}}$
$\chi^{(2)}_{eff}$	Effective second order susceptibility	$22\,\mathrm{pm\,V^{-1}}$
L	Crystal length	$40\,\mathrm{mm}$

Table 2: Physical constants used in the simulations presented.

6.4 SLOWLY VARYING, MONOCHROMATIC, PLANE CW WAVE

Simulation of difference frequency generation for plane waves is implemented into Matlab using Equation (77), (78) and (79).

The simulations are implemented using an *adaptive step-sizes* to minimize calculation time and increase precision in accordance with Section 6.1.1. The Matlab ODE solver is implemented as

```
%% Solve set of ODEs
% Sets the global and local maximum fault tolerance.
options = odeset('AbsTol',TolAbs,'RelTol',TolRel);
% Solve ODE: ode45(solve for , interval , init.values , options).
[z, A]  = ode45(@(z,A) ode45equation(z,A,g,deltak,Param.QPM, ...
    lcoh),linspace(0,Param.Length,Param.zsteps), ...
    [Signal.A, Idler.A, Pump.A], options);
```

where the code

```
[Signal.A, Idler.A, Pump.A]
```

defines a vector of the initial condition of the ODE given by Equation (77), (78) and (79).

The entire Matlab script can be seen in Appendix H and the implementation of the *ode45equation* script is seen in Appendix I.

6.5 SLOWLY VARYING, MONOCHROMATIC, SPATIAL CW WAVE

A number of different approaches to solve equation (57) has been proposed [7, 3]. In the following the approach presented by [7] where the solution to the *paraxial wave equation* is expanded in *Laguerre-Gaussian modes* in accordance with Section 4.5.1.

The simulations have been implemented in *Matlab* and compared to the simulation of the more simple case of *Slowly varying, monochromatic, plane wave* described in Section 4.6 and simulated in Section 6.4.

6.5.1 *Matlab implementation of Gaussian beams*

The approach of spanning the field as a sum of Gaussian modes was described in detail in Section 4.5.1 and has been implemented in Matlab to analyse the parametric frequency conversion in MgO:PPLN crystals for monochromatic, slowly varying CW waves having a transverse, circular symmetric profile. The Matlab implementation is sketched as a flowchart in Figure 14

6.5.2 *Basis*

Each interacting waves have the same mutual basis in cylindrical coordinates while each wave has an individual Gaussian basis due to their different beam waists. We name the n-dimensional r space for V and span it using a basis of functions ψ_n. We name the p dimensional Gaussian space as W and span it using functions $u_{p,0}$ given by Equation (60). Is is well known from linear algebra that the linear transformation from V space to W is given by

$$T: \quad V \to W \tag{112}$$

where T is a p x n matrix given by

$$T = 2\pi\Delta r \begin{bmatrix} r_1 u^*_{0,0}(r_1, z) & \cdots & r_n u^*_{0,0}(r_n, z) \\ \vdots & \ddots & \vdots \\ r_n u^*_{p-1,0}(r_1, z) & \cdots & r_n u^*_{p-1,0}(r_n, z) \end{bmatrix} \tag{113}$$

In Matlab this basis transformation matrix must be made for each z-step and is made with the matlab function *GaussianBasis.m* seen in Appendix J

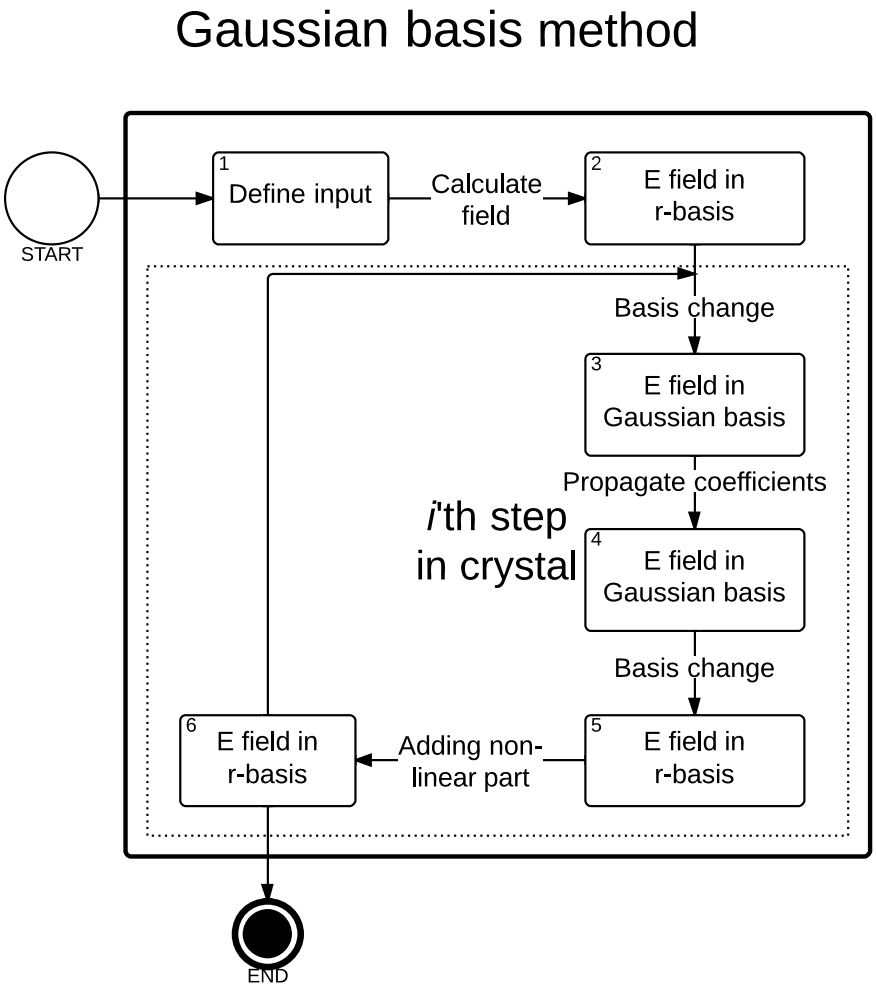


Figure 14: Flowchart illustrating the procedure implemented in the Matlab simulation using a *Gaussian Basis Method* to simulate propagation of Gaussian beams.

To describe the electric field in V it is expanded using the eigenfunctions of W as:

$$E(r) = \sum_{j=0\cdots n-1} a_j u_{j,0} \tag{114}$$

where a_j is the coordinates in W space.

6.5.3 *Beam waist*

It was shown how a spatially varying, circular symmetric field could be expanded as a series of *Laguerre Gaussian* modes. It is favourable to expand the field by as fem modes as possible and this is done by choosing the correct basis for the three interacting waves. The choice of basis should be such that without depletion the coupling from the fundamental mode of the pump should couple to the fundamental mode of the signal and the idler [31]. If the beam waist of the pump a $w_{0,pump}$ and the beam waist of the idler is $w_{0,idler}$ (see Section 4.5.1) it can be shown that the beam waist facilitating coupling between the fundamental modes are the given by

$$\frac{1}{w_{0,signal}} = \frac{1}{w_{0,pump}} + \frac{1}{w_{0,idler}} \tag{115}$$

6.6 PULSES

Parametric Frequency Conversion using *plane wave, monochromatic slowly varying pulses* is discussed in Section 5.5. Simulation using the *Split-step Fourier Method* has been implemented in *Matlab*. The simulation routine is sketched as a flowchart in Figure 15 and the full code is seen in Appendix L.

The Fourier transformation from step 2 to 3, propagation from step 3 to 4 and lastly Fourier transforming back to time domain (see Figure 15) is implemented in Matlab by

```
1    %Linear part
2    Idler.E.f(nn,:)    = fftshift(FourierT(Idler.E.t(nn,:), dt));
3    Idler.E.f(nn,:)    = Idler.E.f(nn,:) .* exp(1i.*w_shift.*dz *
         ...
4        (1/Idler.v-1/Pump.v)); %Linear dispersion
5    Idler.E.f(nn,:)    = fftshift( Idler.E.f(nn,:) );
6    Idler.E.t(nn+1,:)  = IFourierT(Idler.E.f(nn,:), dt);
7
8    Signal.E.f(nn,:)   = fftshift(FourierT(Signal.E.t(nn,:), dt))
        ;
```

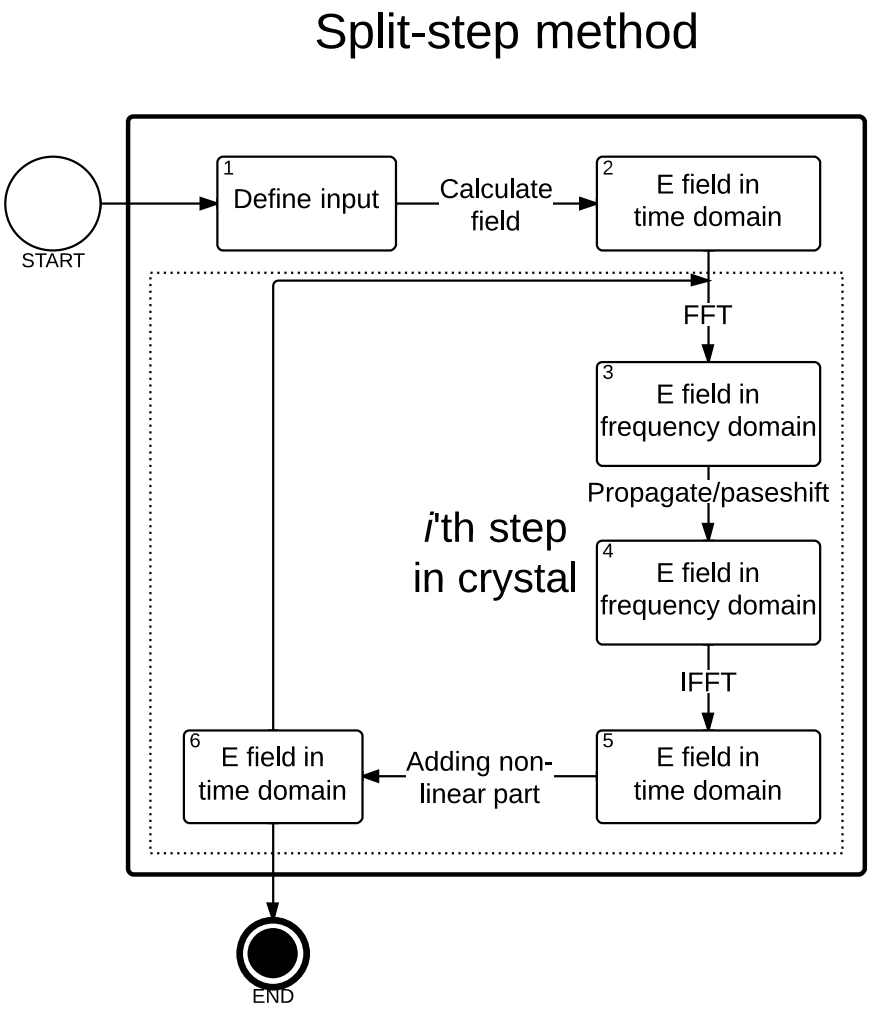


Figure 15: Flowchart illustrating the procedure implemented in the Matlab simulation using *Split-steps Fourier Method* to simulate propagation of pulses.

```
9     Signal.E.f(nn,:)   = Signal.E.f(nn,:) .* exp(1i.*w_shift.*dz
          * ...
10        (1/Signal.v-1/Pump.v)); %Linear dispersion
11    Signal.E.f(nn,:)   = fftshift( Signal.E.f(nn,:) );
12    Signal.E.t(nn+1,:) = IFourierT(Signal.E.f(nn,:), dt);
13
14    Pump.E.f(nn,:)     = fftshift(FourierT(Pump.E.t(nn,:), dt));
15    Pump.E.f(nn,:)     = Pump.E.f(nn,:) .* exp(1i.*w_shift.*dz *
          ...
16        (1/Pump.v-1/Pump.v)); %Linear dispersion
17    Pump.E.f(nn,:)     = fftshift( Pump.E.f(nn,:) );
18    Pump.E.t(nn+1,:)   = IFourierT(Pump.E.f(nn,:), dt);
```

Note that the Fourier transformation and inverse transformation are not native Matlab functions but custom scripts. The script is seen in Appendix G and originates from [33]. The non-linear contribution from step 5 to 6 is implemented in Matlab as in the plane wave simulation by

```
1   %Non linear part
2   Idler.E.t(nn+1,:)  = Idler.E.t(nn+1,:) + dz.*1i.*Idler.Omega
        ...
3       .* const.chi_eff ./ (2*n(1)*const.c) .* conj( Signal.E.t(
            nn,:) )...
4       .* Pump.E.t(nn,:) .* exp(1i * deltak*dz);
5   Signal.E.t(nn+1,:) = Signal.E.t(nn+1,:) + dz.*1i.*Signal.
        Omega ...
6       .* const.chi_eff ./ (2*n(2)*const.c) .* conj( Idler.E.t(
            nn,:) ) ...
7       .* Pump.E.t(nn,:) .* exp(1i * deltak*dz);
8
9   if( setting.Depletion == true )
10      Pump.E.t(nn+1,:)    = Pump.E.t(nn+1,:) + dz.*1i.*Pump.
            Omega ...
11          .* const.chi_eff ./ ( 2*n(3) * const.c) .* Idler.E.t(
                nn,:) ...
12          .* Signal.E.t(nn,:) .* exp(-1i * deltak*dz);
13  end
```

It worth noticing that the Fast Fourier method is significantly faster if the number of time steps, n_t is a power, p, of two, meaning that $n_t = 2^p$ [25, 16]. This is a direct result of the underlying algorithm, *The Fastest Fourier Transform in the West*, developed by Matteo Frigo and Steven G. Johnson at MIT [16].

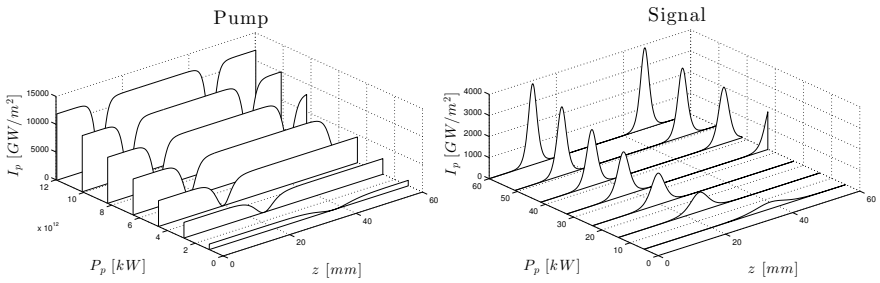
7

PLANE WAVES

Simulations of difference frequency generation (DFG) for monochromatic plane waves have been made as described in Section 6.4. Simulations of the DFG process with a pump wavelength of 1064 nm, idler wavelength of 1570 nm and signal wavelength of 3313 nm is presented in the following. An effective area of a circular beam of radius 40 µm has been used, $A_{eff} = \pi r^2 = \pi 40\,\mu m^2 \approx 5.03 \times 10^{-9}\,m^2$.

The fields were phase matched using a 1st order QPM pooling of period $\Lambda = 30.49\,\mu m$ which, in accordance with Figure 5, is fulfilled at a temperature of 108.8 °C. The QPM structure is implemented by use of the effective QPM structure discussed in Section 6.1.2. This is done by setting $\Delta k = 0$ and reducing the effective susceptibility by a factor of $\frac{2}{\pi}$ giving an effective susceptibility of $\frac{2}{\pi} \cdot \chi^{(2)}_{eff} = \frac{2}{\pi} \cdot 22\,pm\,V^{-1} \approx 14\,pm\,V^{-1}$, in accordance with [13].

The pump power is varied from 5 kW to 60 kW and the result is seen in Figure 16. The crystal length is set to 60 mm and back conversion arising from depletion is seen for all the pump powers. The computation time for the simulation used to create the data for Figure 16 is $t_{cal} \approx 3\,s$. The relatively low computation time is mainly due to two reasons; the first being *Matlab*'s native ODE solver, *Runge-Kutta 45*, the second being the implementation of an effective QPM structure instead of the actual domain inversion.

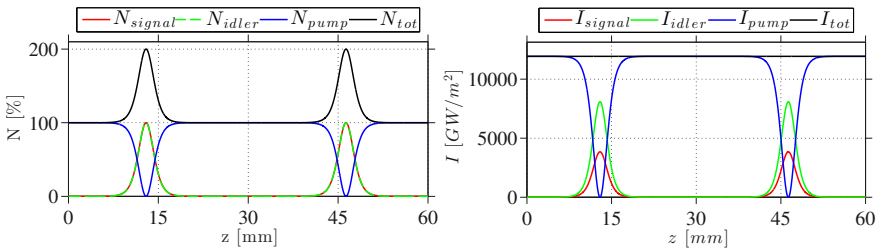


(a) (b)

Figure 16: Plane wave DFG simulation. Propagation through crystal for different pump powers for. (a) pump and (b) signal.

The simulation from Figure 16 with a pump power of 60 kW is seen more clearly in Figure 17 that shows the propagation of total number of photons, Figure 17a, and the irradiance, Figure 17b, for the

signal (red), idler (green) and pump (blue). It is clearly seen that the total power is conserved while the total number of photons relative to the number of photons at the beginning of the crystal increases to 200% since each pump photons splits to an idler photon and a signal photon.



(a) (b)

Figure 17: Plane wave difference frequency generation through crystal. Comparison of. (a) photon number (b) power of signal, idler and pump. The pump power is $P_p = 60\,\mathrm{kW}$

This simulation clearly does not account for losses in the crystal since the total irradiance is conserved (black). Nor is the dipole-dipole interactions taken into account since the first Born approximation is used. If the dipole-dipole interaction were to be included further frequencies would arise if these are phase matched and while the total irradiance would be conserved the total irradiance of the three beams presented here would not.

7.1 REALISTIC INTENSITIES

It should be noted that the values of the pump irradiance used in Figure 16 is extremely high. This was intentionally set to show the concept of multiple depletion and back conversion (depletion cycle) as is seen for a pump power $P_p > 30\,\mathrm{kW}$ meaning pump irradiances of $I_p > 30\,\mathrm{kW}/\pi(40\,\mu\mathrm{m})^2 \approx 6 \times 10^3\,\mathrm{GW}/\,\mathrm{m}^2$. The damage threshold for MgO:PPLN crystal is somewhere between $1 \times 10^3\,\mathrm{GW}/\,\mathrm{m}^2$ and $1.5 \times 10^4\,\mathrm{GW}/\,\mathrm{m}^2$ depending on CW or pulsed operation [13]. This means that a plane wave (approximately plane wave) would not experience multiple depletion cycles but can experience full depletion.

Covesion has shown SHG of a 10 W CW laser at 1064 nm with a pump irradiance was above $5.00\,\mathrm{GW}/\,\mathrm{m}^2$. No damage to the crystal was measured for operating times of more than 2000 hours. Covesion likewise showed no damage to the crystal when generating midin-

REGIME	IRRADIANCE	DAMAGE	NOTES
CW	500 kW/ cm^2	No	1064 nm, 10 W
CW	500 kW/ cm^2	No	1560 nm, 30 W
CW	200 kW/ cm^2	No	532 nm, 2.2 W
ns	100 MW/ cm^2	Yes	1064 nm, 10 – 20 ns
ps	100 MW/ cm^2	No	1060 nm, 20 ps, 24 W
ps	1.5 GW/ cm^2	No	1064 nm, 7 ps
ps	1.8 MW/ cm^2	Yes	530 nm, 20 ps, 500 mW
ps	7.5 MW/ cm^2	Yes	530 nm, 20 ps, 17 W
ps	468 MW/ cm^2	No	1064 nm, 7 ps, 17 W
fs	4 GW/ cm^2	Yes	1550 nm, 200 fs, 200 mW

Table 3: Experiments to determine the damage threshold of 5% MgO doped PPLN crystals using pulsed and CW operation. For more details regarding the experiments see [13].

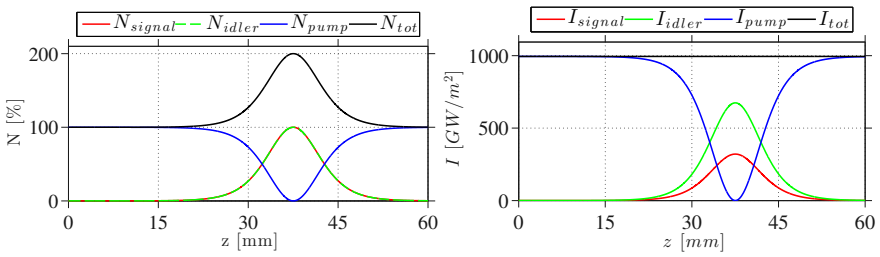
frared light from parametric frequency generation using a 7 ps pulse laser at 1064 nm with a peak pump irradiance of 1.5×10^4 GW/ m^2 [13].

Table 3 shows damage measurements done by *Covesion* and indicates that MgO:PPLN crystal has extremely high damage threshold for pulse operation while it is substantially lower for CW operation. This is to be expected as the time averaged power is much lower for pulses even with high peak irradiances. It is however still of interest to analyse the behaviour of high intensity light propagation since, as will be shown later, some pulses can be modelled to a good approximation by the simpler CW theory by introducing a retarded time frame.

A more realistic CW irradiance is plotted in Figure 18 where the irradiance is so high that depletion occurs but not such that multiple depletion cycles occur.

7.2 DEPLETION AND NONDEPLETION

Plane wave CW simulation has been carried out to determine at what point depletion becomes significant. Figure 19 illustrates the simu-



(a) (b)

Figure 18: Plane wave difference frequency generation through crystal. Comparison of. (a) photon number (b) power of signal, idler and pump. The pump power is $P_p = 5\,kW$

lated efficiency of the DFG process at the end of the end of a 60 mm long MgO:PPLN crystal. The efficiency is defined as

$$\gamma(z) = \frac{I_{signal}(z)}{I_{pump}(0)} \tag{116}$$

The efficiency plotted in Figure 19 is evaluated at the crystal end, $z = L = 60\,mm$. The efficiency is seen to rise from 0 to 0.32 as the pump power increases from 0 kW to 2.32 kW for an effective area of $A_{eff} = \pi(40\,\mu m)^2 \approx 5.0 \times 10^{-9}\,m^2$. The value of 0.32 is set by the energy proportion of the pump and signal as one signal photon is created as one pump photon is annihilated and vice versa. Since $\lambda_{pump} = 1064\,nm$ and $\lambda_{signal} = 3313\,nm$ we get

$$\frac{\omega_{signal}}{\omega_{pump}} = \frac{\lambda_{pump}}{\lambda_{signal}} = \frac{1064\,nm}{3313\,nm} \approx 0.32 \tag{117}$$

The efficiency drop after the peak, which is caused by depletion, and falls to zero due to back conversion from signal to pump (depletion). The second peak illustrated the aforementioned multiple depletion cycles where conversion and back conversion is repeated multiple times.

The same effect is seen in Figure 20 where the z-location of the first optimum point is plotted. The horizontal line at pump powers from 0 kW to 2.32 kW illustrates how the most efficient length of the crystal is the end of the crystal until depletion occurs and hence the optimum location occurs before the end of the crystal. The optimum point is seen to asymptotically approach zero as the pump power is increased.

Figure 19 and 20 illustrates clearly that plane wave theory must account for depletion at least for pump powers above 2.32 kW for an

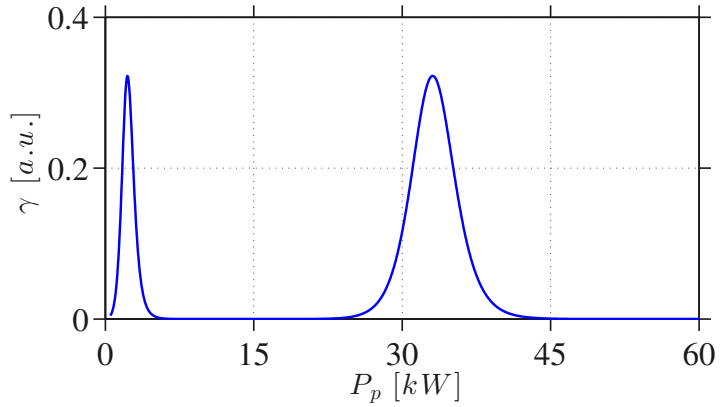


Figure 19: Plane wave difference frequency generation through crystal. Efficiency defined by Equation (116) evaluated at the crystal end as function of pump power.

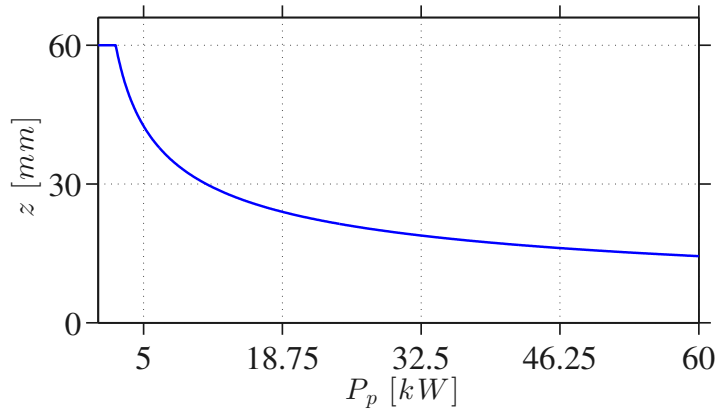


Figure 20: Plane wave difference frequency generation through crystal. The location in the crystal where the efficiency defined by Equation (116) reaches the first maximum.

effective area of $A_{eff} = \pi(40\,\mu m)^2 \approx 5.0 \times 10^{-9}\,m^2$ meaning for a pump irradiance of $460\,GW/\,m^2$. This is where complete depletion occur but the assumption of no depletion is expected to break down even when fraction of the pump power converted is of significance.

The validity of the model not accounting for depletion is seen by comparing the model not account for depletion to the model that does account for depletion. This is seen in Figure 21 that shows hos the efficiency falls off as depletions occurs, for the model accounting for depletion (blue) while the model not accounting for depletion calculates a constantly increasing efficiency (red). It is thus clearly seen that plane wave theory must account for depletion at least for pump

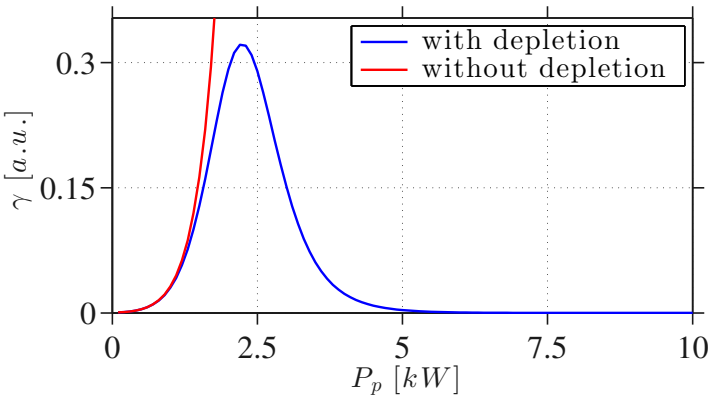


Figure 21: Plane wave difference frequency generation through crystal. Efficiency defined by Equation (116) evaluated at the crystal end as function of pump power with (blue) and without (red) accounting for depletion.

powers above $\approx$ 1 kW for an effective area of $A_{eff} = \pi(40\,\mu m)^2 \approx 5.0 \times 10^{-9}\,m^2$ meaning for a pump irradiance of $\approx 200\,GW/\,m^2$.

8

GAUSSIAN BEAMS

Simulations of difference frequency generation (DFG) for spatially confined fields has been carried out as described in detail in Section 6.5. The results presented in this chapter uses the same parameters as the simulation in Chapter 7 where DFG is simulated for a pump wavelength of 1064 nm, an idler wavelength of 1570 nm and a signal wavelength of 3313 nm.

A basis is chosen such that coupling occurs from the fundamental pump mode to the fundamental signal mode for pump powers in the range where no depletion occurs.

The simulations include varying beams waist sizes and varying pump powers. The results of the simulations are compared to the plane wave simulation results presented in Chapter 7. The computation time of the plane wave simulation is significantly less than that of the Gaussian basis method and is favourable to use whenever possible.

A seed is provided as the generation of signal would not occur if no seed was provided, as discussed in Section 5.6 where the effect of quantum fluctuations was also discussed. The seed is provided as a fundamental mode but should in a quantum description be added statistically to every mode. This thesis does not deal with the modelling of quantum fluctuations as mentioned earlier and the approach of a fundamental signal seed is used.

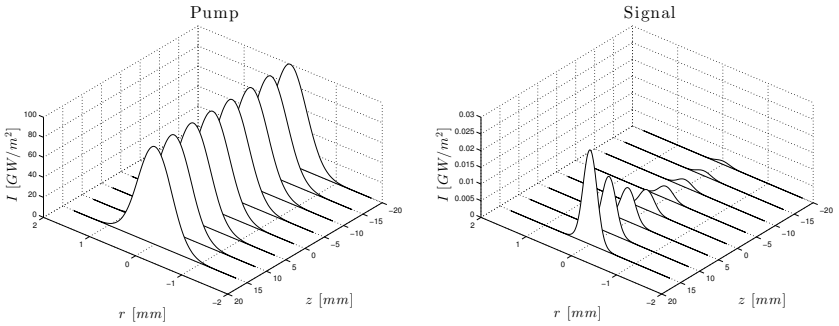
The beam waist of all beams presented in this chapter is located at the center of the crystal and the z axis has origin at the beam waist. The crystal front is thus located at $z = -z_0 = -20\,\text{mm}$ while the crystal end is located at $z = z_0 = 20\,\text{mm}$.

8.1 SLIGHTLY CONFINED MODES

This section presents simulation results of modes having a large beam waist and thus being *slightly confined spatial modes*. These modes experience small divergence and are expected to resemble plane waves. The computation time for the Gaussian simulation presented below is $\approx 100\,\text{s}$ which compared to the plane wave simulation is ≈ 200 times greater (plane wave took $\approx 3\,\text{s}$ to compute 7 simulation).

8.1.1 *Weak coupling*

A low pump power means that the conversion is low and hence we expect a non depleted approach would approximate the situation. Since slightly confined spatial fields is approximately divergence free we expect each spatial point to follow the plane wave theory.



(a) (b)

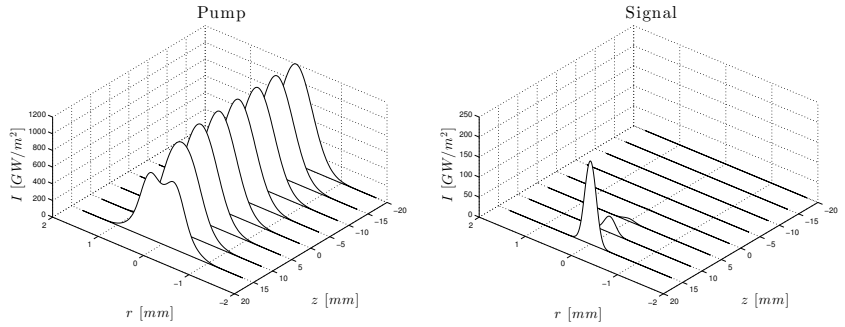
Figure 22: DFG simulation using a pump wavelength of 1064 nm, an idler wavelength of 1570 nm, a signal wavelength of 3313 nm, a beam waist of 1 mm and a pump power of $\approx$ 78 kW. Illustrates the (a) pump and (b) signal as they propagate down the crystal.

Figure 22 illustrates how a slightly confined fundamental pump mode of beam waist $w_{0,\mathrm{pump}} = 1\,\mathrm{mm}$ couples to the fundamental mode of the signal having beam waist $\frac{w_{0,\mathrm{pump}}}{1+\sqrt{2}} \approx 0.414\,\mathrm{mm}$ in the case where the irradiance of the pump is low and thus no depletion occur. The total power of the pump is $\approx 78\,\mathrm{kW}$ and the beam waist is 1 mm giving a peak irradiance of $\approx 100\,\mathrm{GW/\,m^2}$. Such powers is difficult to generate using CW lasers but can easily be generated using pulsed lasers with peak powers high above this. The field should then be considered as a pulse which is not the case for these simulations, but some pulses can be simulated as CW fields, as shall be discussed later. It is clearly seen that the energy transfer from the pump illustrated in Figure 22 is negligible and the spatial mode profile of the pump is unchanged along the crystal.

The signal and the pump can both be described completely using just the fundamental mode of each. This is due to the small divergence (large beam waist) and the weak coupling that causes the power transfer to occur only from the fundamental pump mode to the fundamental signal mode.

8.1.2 *Strong coupling*

Simulations of slightly confined spatial modes has also been preformed in the regime of strong coupling where a total pump power of $\approx 780\,\mathrm{kW}$ and beam waist of $w_{0,\mathrm{pump}} = 1\,\mathrm{mm}$ was used. Figure 23 illustrates how the pump is depleted at the center where power is transferred most efficiently from the high intensity region of the pump to the signal. The signal beam is narrower than the pump which corresponds to the choice of basis.



(a) (b)

Figure 23: DFG simulation using a pump wavelength of 1064 nm, an idler wavelength of 1570 nm, a signal wavelength of 3313 nm, a beam waist of 1 mm and a pump power of $\approx 780\,\mathrm{kW}$. Illustrates the (a) pump and (b) signal as they propagate down the crystal.

One would expect back conversion from the signal to the pump once the pump depletes in the high intensity region near the beam center but the irradiance is not sufficient that complete depletion occurs within the crystal length of 40 mm. If the crystal length is increased or the irradiance is increased one would expect back conversion.

The signal and the pump can no longer be described completely using just the fundamental mode of each. This is due to the significant power transfer from the pump to the signal that depletes the field only at the center and hence excite higher order modes. This is supported by Figure 24 that shows that the pump and signal is no longer described solely by their fundamental mode at the end of the crystal.The figure plots the power at the end of the crystal as function of modes in the expansion, given by

$$P(n) = \sum_{j=1\cdots n} C_j u_{j,0} \tag{118}$$

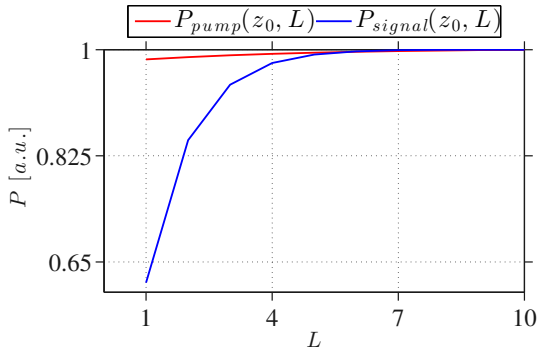
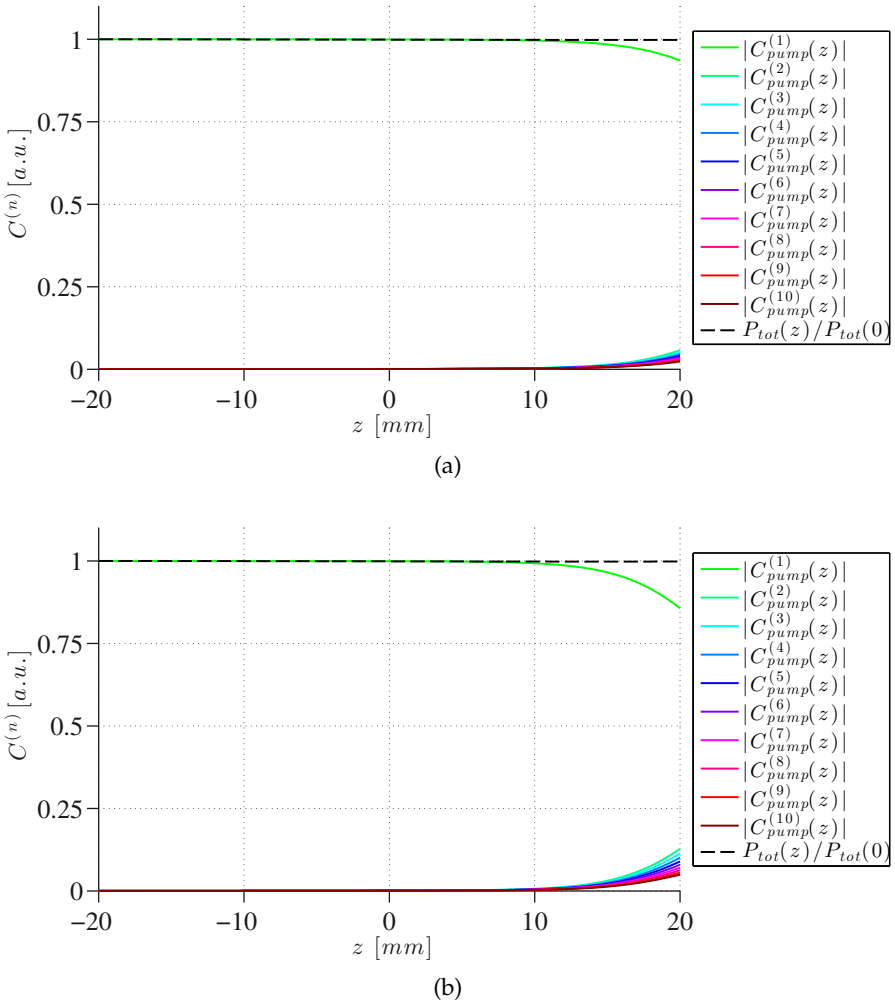


Figure 24: Mode power of slightly confined and weak coupled field, for both pump (red) and signal (blue). It is clear that only the fundamental mode is needed to describe the fields.

where C_j is the expansion coefficients, $u_{j,0}$ is the Laguerre-Gaussian modes and $n \leqslant L_{max}$ where $L_{max} = 10$ is the number of modes used in the simulation. It was postulated by [7] that only six Laguerre-Gaussian modes are needed to describe the field. Figure 24 confirms this for the case of slightly confined spatial modes in the strong coupling regime.

The effect of depletion on the number of excited modes is easiest seen in Figure 25 and 26 where the pump peak irradiance is varied from 25a 1000 GW/ m^2 to 25b 1200 GW/ m^2, 26a 1500 GW/ m^2 and 26b
2300 GW/ m^2. The figure shows the propagation of the expansion coefficients from Equation (118). A higher number of modes is needed to describe the field as depletion becomes more significant and occurs earlier in the crystal as the pump power is increased.

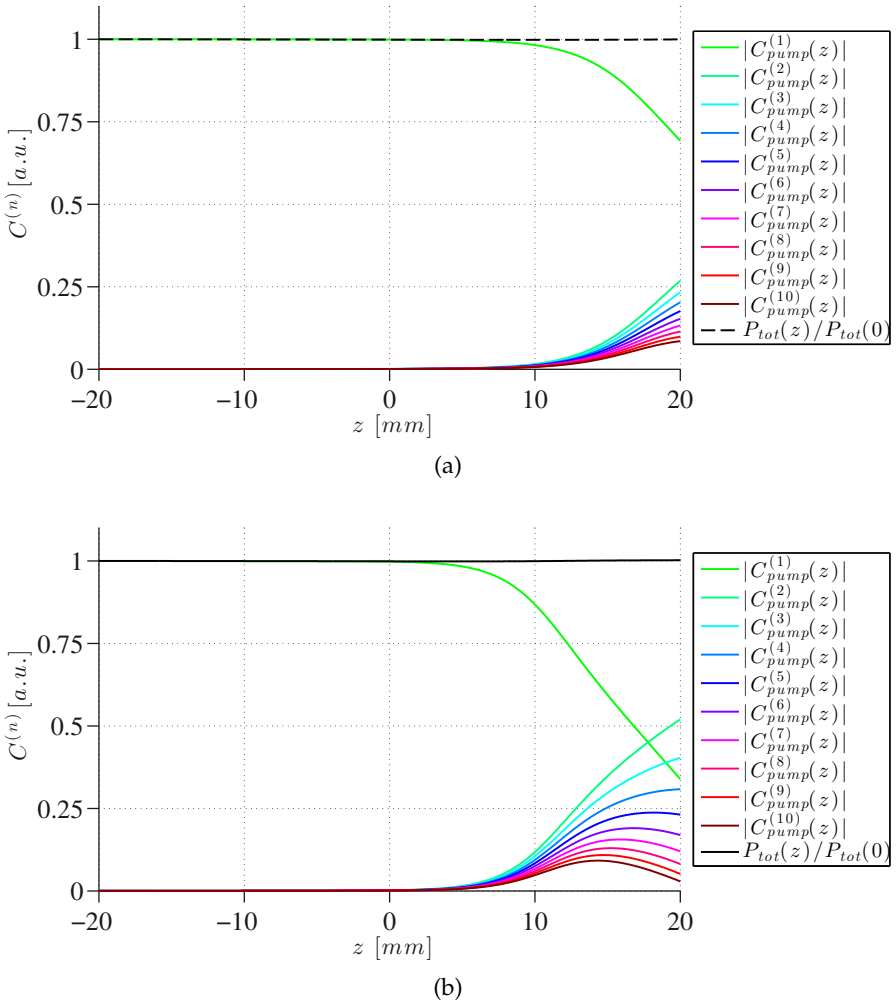


(a)

(b)

Figure 25: Propagation of expansion coefficinets from Equation (118) for slightly confined beam with beam waist of $w_{0,\mathrm{pump}} = 1\,\mathrm{mm}$ for a peak pump irradiance of (a) $1000\,\mathrm{GW/\ m^2}$ and (b) $1200\,\mathrm{GW/\ m^2}$.

.



(a)

(b)

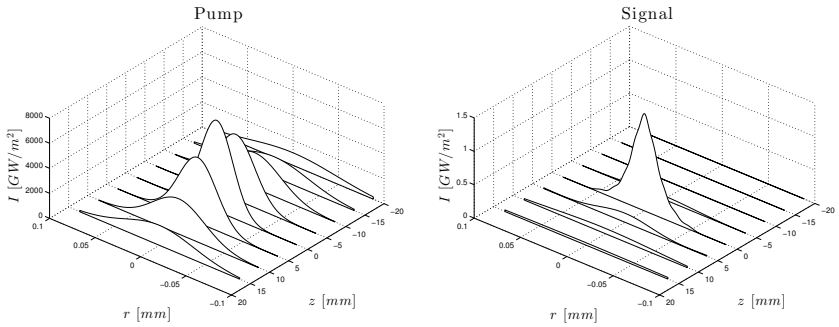
Figure 26: Propagation of expansion coefficinets from Equation (118) for slightly confined beam with beam waist of $w_{0,\mathrm{pump}} = 1\,\mathrm{mm}$ for a peak pump irradiance of (a) $1500\,\mathrm{GW/\ m^2}$ and (b) $2300\,\mathrm{GW/\ m^2}$

.

8.2 HIGHLY CONFINED MODES

The more confined a Gaussian mode is, the more it diverges as it propagates away from the beam waist. The region of highest intensity is therefore on-axis, close to the beam waist and it is thus expected that the conversion in this region is high. Converting from a highly focused pump to an even more tightly confined signal will cause the signal to diverge rapidly as it propagates away from the beam waist.

It is expected that an optimal focusing exists for a given configuration since the overall efficiency depends on the irradiance, the nonlinear coupling and the distance of interaction.



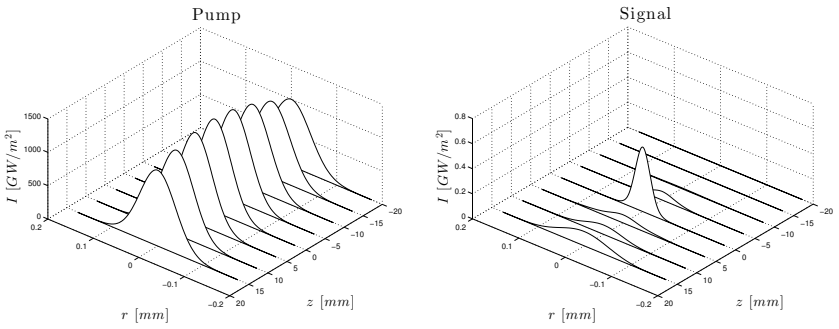
(a) (b)

Figure 27: DFG simulation using a pump wavelength of 1064 nm, an idler wavelength of 1570 nm, a signal wavelength of 3313 nm, a beam waist of 0.05 mm and a pump power of $\approx$ 14 kW. (a) pump and (b) signal as they propagate down the crystal.

Figure 27 shows the simulation of DFG for a pump beam with a small beam waist of 0.05 mm and a pump power of $\approx$ 14 kW. The peak irradiance is thus $\approx$ 1000 GW/ m^2 at the front end of the crystal, this is the same irradiance as the simulations described earlier and shown in Figure 22 and 23. The simulation shows how the signal diverges more rapidly than the pump as expected due to the smaller beam waist.

A simulation like the one just presented in Figure 27 for a larger beam waist of 0.1 mm is seen in Figure 28. It is seen that the divergence of the pump is negligible while the divergence of the more confined signal causes the signal to diverge rapidly. The pump peak irradiance is the same as in Figure 27 giving a pump power of $\approx$ 11 kW

It is seen at the crystal end of Figure 28 that the energy dissipates away from the center of the signal because of the relatively high confinement, although slower than in the highly confined case seen in Figure 27. As the beam diverges, the intensity has reached a level fa-



(a) (b)

Figure 28: DFG simulation using a pump wavelength of 1064 nm, an idler wavelength of 1570 nm, a signal wavelength of 3313 nm, a beam waist of 0.1 mm and a pump power of $\approx$ 11 kW. (a) pump and (b) signal as they propagate down the crystal.

cilitating a larger conversion efficiency meaning that the energy transfer from the pump increases in spite of the divergence. This is seen easily from Figure 29 where the relative on-axis irradiance is plotted. Note that the irradiance is higher for the idler than the signal since the wavelength of the idler is shorter but they follow the same trend.

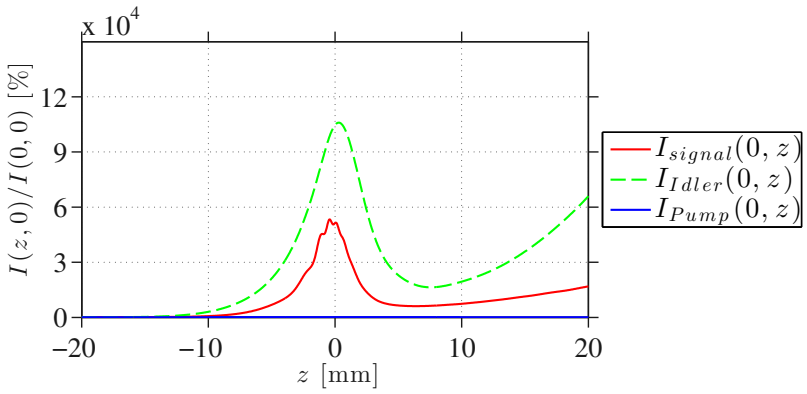


Figure 29: The on-axis irradiance relative to the pump irradiance as function of distance in crystal for DFG. The simulation uses a pump wavelength of 1064 nm, an idler wavelength of 1570 nm, a signal wavelength of 3313 nm, a beam waist of 0.1 mm and a pump power of $\approx$ 11 kW.

8.3 COUPLINGS EFFICIENCY

The overall couplings efficiency is often defined as the ratio of the signal power at the end of the crystal to the pump power at the front of the crystal [5, 32]. This is the same definition introduced in Section 7.2 when dealing with plane wave simulations. Often the signal is modified or measured by a process that utilizes only the fundamental model. This includes coupling to single mode fibers and mode dependent detectors. The coupling efficiency to the fundamental mode is thus of interest since this is the measured value in various applications where only the fundamental mode is utilized.

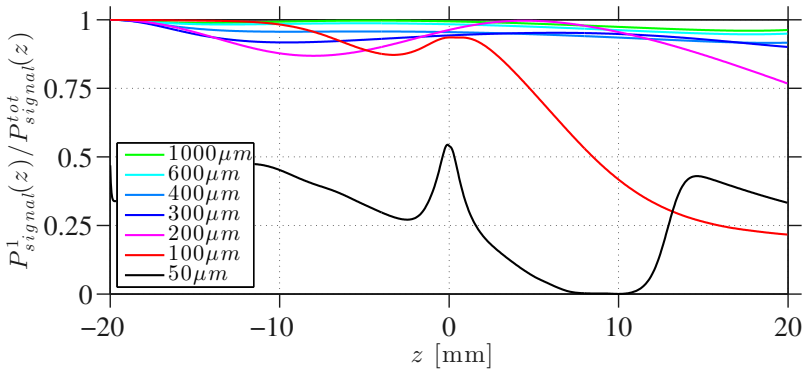


Figure 30: the ratio of power in the fundamental mode and the total power of the signal, as function of propagation distance in the crystal. The beam waist is varied from 1 mm to 50 µm.

Figure 30 shows the signal power in the fundamental mode divided by the overall power in the signal as function of propagation distance in the crystal. The peak irradiance has been kept constant but the beam waist was varied from 50 µm to 1 mm. Is is clearly seen that at beam waist of more than 300 µm facilitates a substantial coupling to the fundamental mode while the coupling significantly drops for beam waists smaller than 200 µm.

The coupling to the fundamental mode is dependent upon parameters like beam waist, pump power and initial mode distribution. Fast numerical simulations can therefore constituent an essential tool in system design since a variety of setup parameters can be changed, simulated and compared with relatively ease once the simulation model has been implemented. This simulation was limited to pumps initially consisting solely of their fundamental modes.

8.4 COMPARING TO PLANE WAVE

Results from the plane wave theory presented in Section 5.2.1 was presented in Chapter 7. The plane wave simulations have the advantageous of easy implementation and very low computation time. If multiple simulations must be processed these can be processed at the same time by multiple processors since the simulations are mutually independent. The Gaussian method simulates a complete transverse profile in one simulation and cannot be processed by separate processors since the points are mutually dependent.

The Gaussian method, in contrast to plane wave simulations, gives information on the mode distribution as mentioned in Section 8.3. The plane wave method can thus be used for cases of weak coupling of beams with large beam waist and where detailed information on the mode distribution is not of essence. The computation time of the Gaussian method is only comparable to that of the plane wave approach if the number of eigenfunctions used to describe the field is significantly lower than the number of discrete points used to describe the transverse profile in a plane wave method. It was shown that this is possible by choosing a basis for each wave such that a fundamental pump and idler couples to a fundamental signal mode. Since the coupling from the fundamental signal and idler is not to the fundamental pump this creates a high number of excited modes once depletions becomes significant.

9 PULSES

This chapter presents simulation results of temporal pulses using plan wave theory as described in Section 5.5.

The simulation method used is the *Split-step Fourier method* described in detail in Section 6.6 and the pulses simulated are all assumed to have a temporal Gaussian form. A natural next step would be to consider non symmetric pulses of arbitrary temporal shape but this has been omitted in the following.

It is expected that slowly varying pulses in the nano second regime will have negligible dispersion and hence be described approximately by plane wave theory. Shorter pulses will experience more significant dispersion and hence the waves will not temporally overlap as they propagates down the crystal.

Depletion is likewise expect to be significant in the strong coupling regime and only for pulses having a sufficiently large temporal overlap.

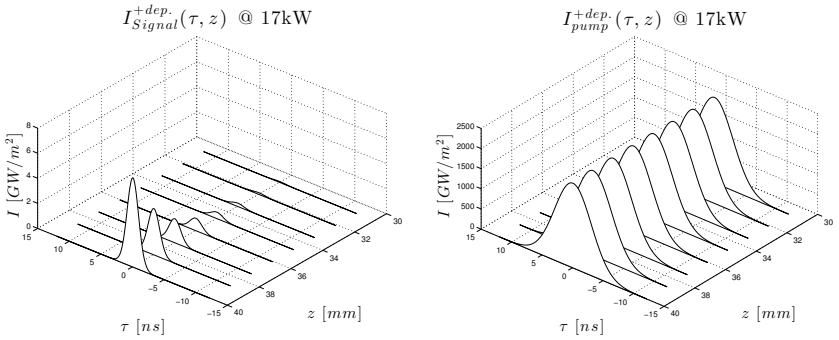
All simulations are of *difference frequency generation* where a pump pulse of carrier wavelength 1064 nm interacts with quantum fluctuations modelled as a small seed. This was described earlier in Section 4.5.1 and used in the Gaussian simulations from Section 6.5 and Chapter 8.

Propagation of pulses of different temporal length is considered in the following using the *Split-step Fourier method*. The *Matlab* scripts is seen in Appendix L.

9.1 NANO SECOND PULSES

Nano second pulses are optically temporally long pulses. The dispersion is small and it is therefore expected that every point in time, with respect to the retarded time frame, follow the plane wave theory.

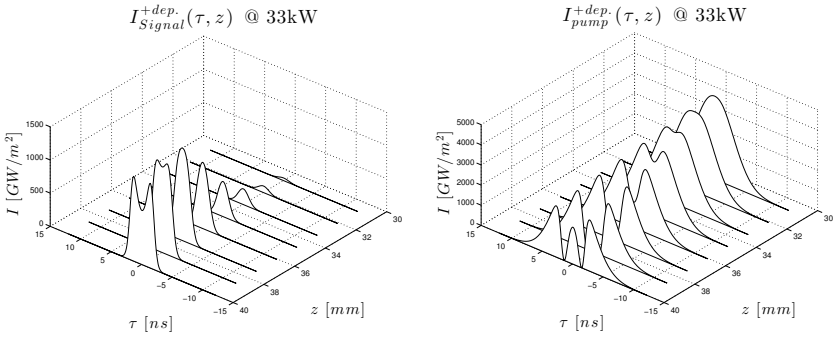
Figure 31 clearly shows how the signal arises as a temporal Gaussian beam with width smaller than that of the pump pulse. There is no significant energy transfer from the pump to the signal in the weak coupling regime which is verified by Figure 31b where the pump is seen to remain approximately unchanged as it propagates. This is in contrast to the *strong coupling regime* where the pump is of higher power and a significant fraction of the power is transferred from the



(a) (b)

Figure 31: DFG Propagation of a 7 ns pulse with depletion implemented for (a) signal and (b) pump in the weak coupling regime.

pump to the signal and back again. This is seen in Figure 32 where the pulse shape is, for both signal and pump, no longer a Gaussian shape. Figure 32 thus shows the effect of temporal depletion. The pulse shape is seen to change significantly from the original Gaussian shape.



(a) (b)

Figure 32: DFG Propagation of a 7 ns pulse with depletion implemented for (a) signal and (b) pump.

9.1.1 *Pump power variation*

The conversion efficiency as function of pump power is seen in Figure 33 for a 7 ns pulse confined to an effective area of $A_{eff} = \pi(50\,\mu m)^2 \approx 7.85 \times 10^{-9}\,m^{-2}$. It is seen that the conversion efficiency increases rapidly as the pulse propagates down the crystal. For high pump power the efficiency increases faster but decreases after depletion

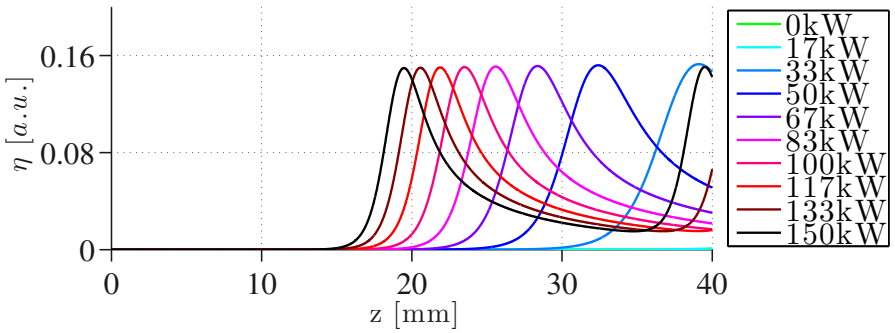
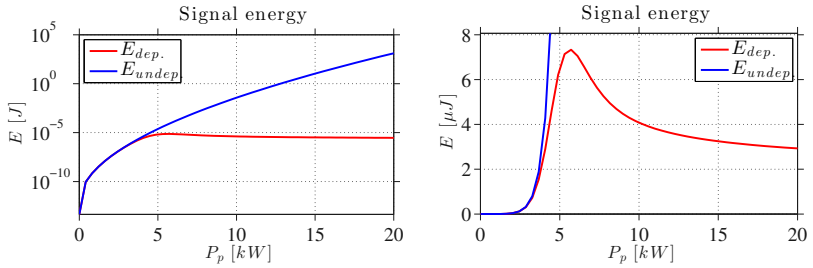


Figure 33: Efficiency as function of propagation distance within the crystal for different pump powers.

has occurred. It is noteworthy that the efficiency reaches the same level regardless of pump power. This has not been shown analytically but should be the subject of further investigation. Recalling that complete depletion where the entire pump pulse is converted to signal and idler pulses would give an efficiency of 0.32 as described in Section 7.2. Almost half of the photons is converted by choosing the correct crystal length and pump power, since the efficiency reaches ≈ 0.15 according to Figure 33.



(a) (b)

Figure 34: Energy of the signal at the crystal end as function of power on a (a) logaritmic and (b) linear scale.

The total signal pulse energy as function of pump power is seen in Figure 34. Figure 34a is plotted on a logarithmic scale while Figure 34b is plotted on a linear scale. The *Split-step Fourier* method with depletion (red) is seen to reach a maximum value after which energy is back converted, while the model without depletion (blue) continuously increases as function of pump power. The model with depletion implemented is thus essential for pump powers above $\approx 4\,\mathrm{kW}$ for an area of $A_{eff} = \pi(50\,\mu\mathrm{m})^2 \approx 7.85 \times 10^{-9}\,\mathrm{m}^{-2}$ giving an irradiance of $\approx 500\,\mathrm{GW/\,m^2}$.

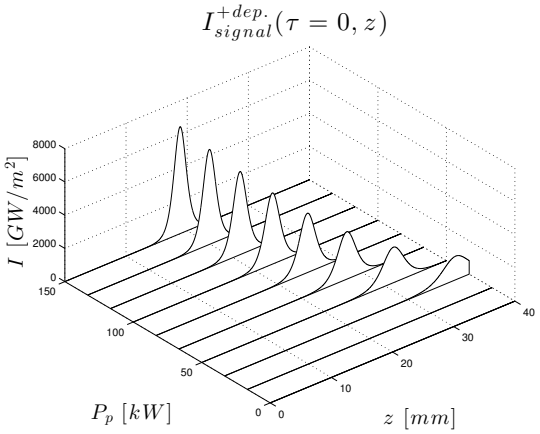
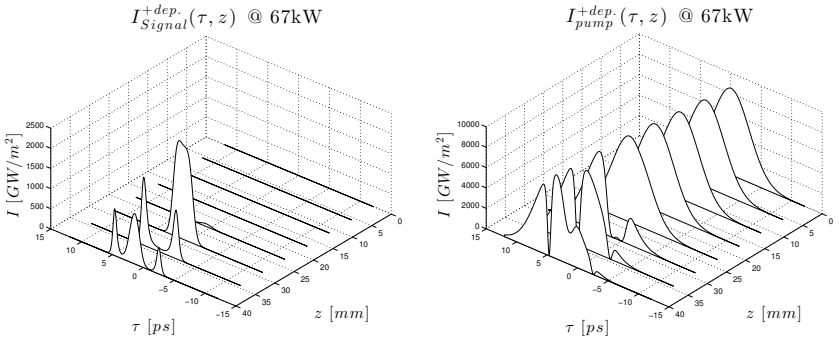


Figure 35: Propagation of 7 ns pulse center as function of pump power.

Figure 35 illustrates how the pulse center of a 7 ns pulse propagates as function of pump power. The propagation is seen to mimic that of the plane wave theory presented in Figure 16b. This supports the statement that nano second pulses have negligible dispersion and hence follow the simple plane wave theory.

9.2 PICO SECOND PULSES

The simulations based on the *Split-step Fourier method* has also been carried out for 7 ps pulses. It should be noted that the pump power cannot be compared directly with that of the nano second pulses as the temporal duration of the pulses is less than that of nano second pulses.



(a) (b)

Figure 36: DFG Propagation of a 7 ps pulse with depletion implemented for (a) signal and (b) pump.

Figure 36 illustrates that the stationary frame is that of the signal and that the pump moves relative to the frame due to dispersion. As a result, the generated pulses is no longer symmetric. The temporally narrow region of depletion indicates that an even higher precision of the simulation could be needed, hence a high number of discrete points must be used to describe the temporal shape adequately.

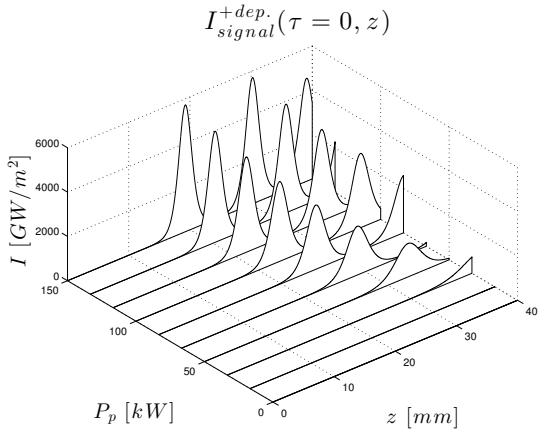


Figure 37: Propagation of 7 ps pulse center as function of pump power.

Propagation of the center of the 7 ps signal pulse is seen in Figure 37. The center is seen to behave significantly different from the case of nano second pulses seen in Figure 35 and plane waves seen in Figure 16b. The main difference between the case of nano second pulses and that of pico second pulses is the back conversion time. Nano second pulses follow the plane wave theory while pico second pulses deplete temporally while the pump and signal moves relative to each other in time. This means that the temporal point of the pump that is depleted at time t_1 is not the same temporal point that converts at later times $t > t_1$. The effect is a more rapid back conversion.

The rapid back conversion seen in Figure 37 is confirmed by comparing the efficiency as function of distance for a 7 ps pulse, Figure 38, to the efficiency of a 7 ns pulse, Figure 33. The efficiency is seen to decrease after depletion, just as the case of nano second pulses and plane waves, but increases again as the temporal overlap between the signal high intensity region and the pump depleted region is diminished as the pulses propagates away from each other due to dispersion.

The plane wave theory is thus not accurate enough that it can adequately describe the propagation and conversion of the 7 ps pulse.

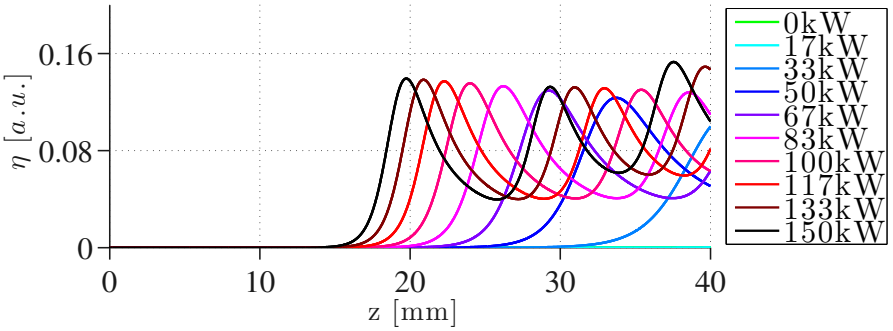


Figure 38: Efficiency as function of propagation distance within the crystal for different pump powers.

9.3 MEASUREMENTS

Lasse Høgstedt has kindly provided measurements that is used to verify the tendency of the simulation and estimate the quantum noise level.

The setup used for the measurements is sketched in Figure 39 and consist of two processes. The first process, *pulse generation*, is a difference frequency generation where a Q-switched 1064 nm laser pump mixes with quantum fluctuations, giving a pulsed signal at $\approx$ 3300 nm. The second process, *detection*, is a sum frequency generation process where the signal is mixed with a CW 1064 nm laser generating visible pulses at $\approx$ 800 nm. A more detail sketch of the setup is provided in Appendix O

The simulations focus on the first part of the setup, the process of mid infrared pulse generation by DFG. The quantum noise is, as mentioned before, of a complicated nature. In the following the quantum noise is thought as a small pulse seed. The simulations is thus of a 1064 nm CW pump and a 7 ns 1570 nm seed pulse that propagate through a MgO:PPLN creating a pulse at the *difference frequency* having wavelength

$$\lambda_3 = \left(\frac{1}{1064\,\text{nm}} - \frac{1}{1570\,\text{nm}} \right)^{-1} \approx 3300\,\text{nm} \tag{119}$$

The measurements on the other hand is of a 1064 nm CW pump that mixes with a broad quantum noise signal. Only the frequencies close to phase match is efficiently coupled and hence the generated signal has a relatively narrow bandwidth. The quasi-monochromatic approach of the simulation is thus expected to approximate the generated signal.

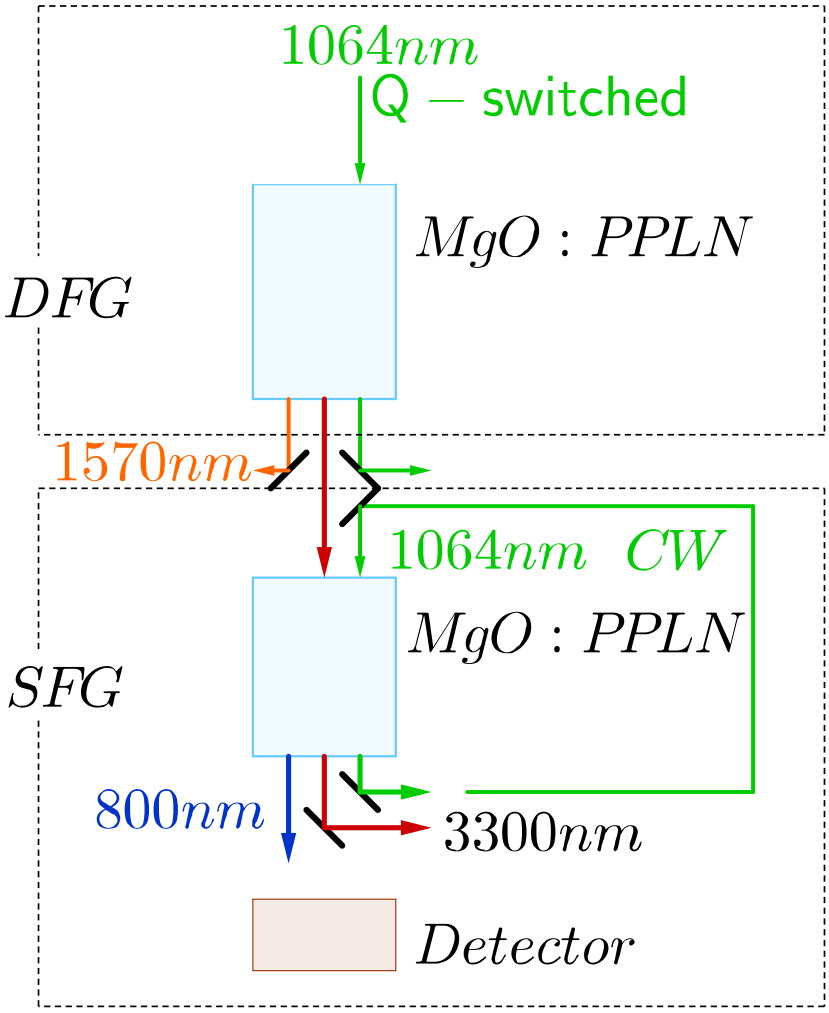


Figure 39: Sketch of measurement setup. Consist of two parts, firstly a DFG process creating mid infrared pulses and secondly a SFG process converting the mid infrared signal to visible at $\approx 800\,\text{nm}$.

The measurements is preformed with a variable damping inserted such that the power of the Q-switched laser can be varied. Both peak power and time average power of the DFG generated pulse is measured for a variety of pump levels, and is seen in Figure 41 .

9.3.1 *Estimation of quantum noise level*

Figure 40 shows the measured traces of the mid infrared signal and the pump, before and after the crystal at different pump powers.

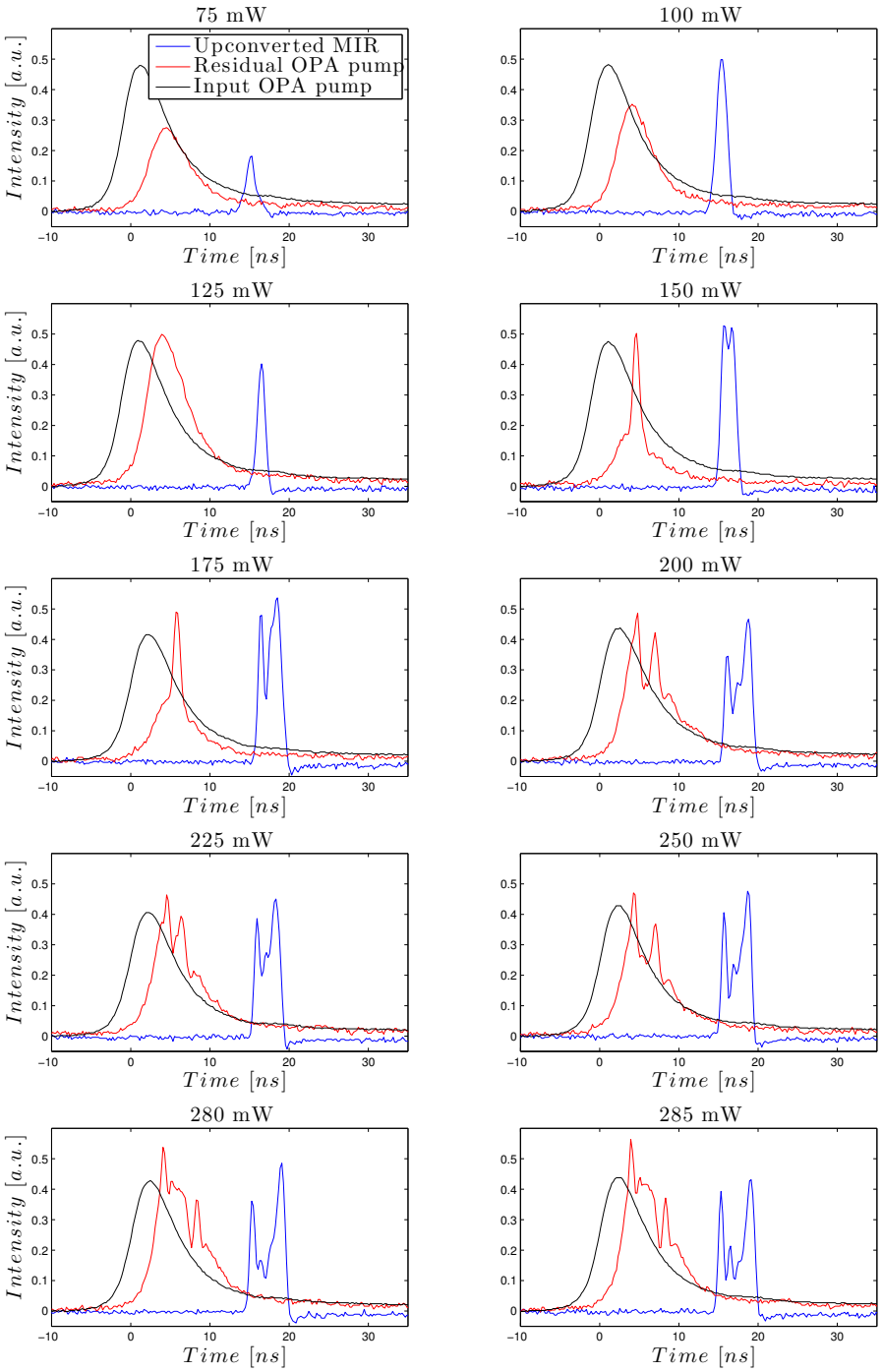


Figure 40: Measurements of upconverted mid infrared signal (blue), input power (black) and the residual pump (red) for different pump levels.

It is seen that depletion occurs for pump powers above $\approx 200\,\mathrm{mW}$. Figure 41 plots the signal power for different pump powers. Using now that a pump power of 175 mW creates a non depleted pulse of power $\approx 8\,\mathrm{mW}$ and inserting this into Equation 111 yields that $I_{signal}(0) = 10\,\mathrm{mW}/\,\mathrm{m}^2$. Since no seed is applied in these measurements the signal must originate from quantum noise.

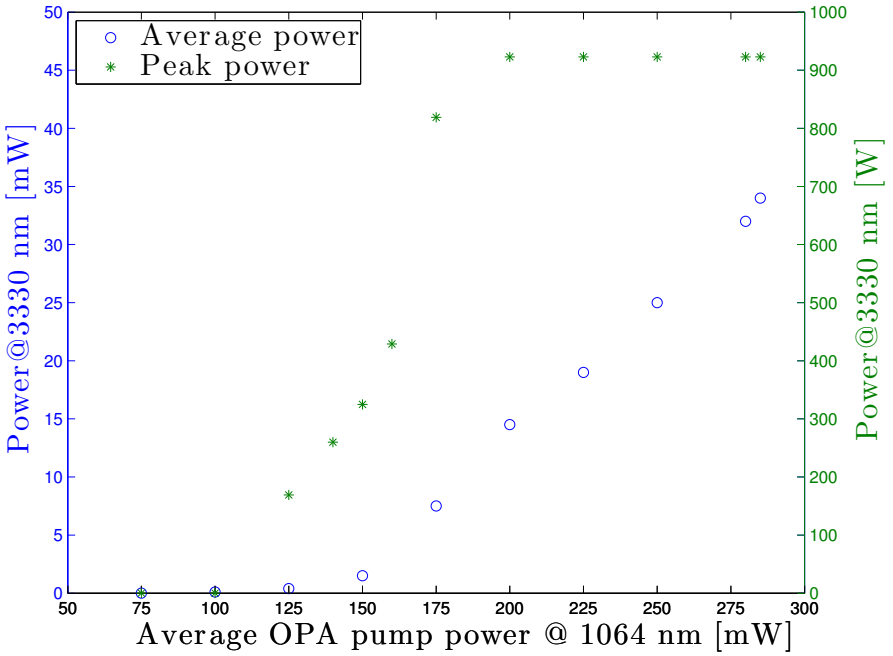


Figure 41: Measurements of the signal power for different pump powers. Depletion occurs as the peak power flattens.

It should be noted that the very loosely estimated quantum level seems extremely high and that the simple estimation should be subject to serious concern. It is, as earlier mentioned, however not the subject of this thesis to deal with quantum noise. The results therefore might be different if no seed is present but only a low intensity, broad band, quantum fluctuation seed. If however a narrow seed of $\approx 10\,\mathrm{mW}/\,\mathrm{m}^2$ is applied then the simulation should approximate the configuration.

9.3.2 *Depletion*

The effect of depletion is clearly seen from the measurement presented in Figure 40 and 41. The measurements is only preliminary and should only serve as an indication that temporal depletion could be a serous effect for nano second pulses. The temporal shape of the

pulse is not Gaussian and hence cannot be directly compared to the simulated results. The detection and generation is not only temporally dependent but also spatially dependent. The simulations indicate that spatial depletion could also occur and the measured depletion could potentially also arise from the effect of spatial confinement.

The measurements thus indicate that temporal depletion occurs for nano second pulses as simulated using the *Split-step Fourier method*, but the validity of the measurements should be subject of further research.

Part III

CONCLUSION

10 CONCLUSION

This thesis has presented detailed theoretical, experimental, analytical and numerical work on second order non-linear effects in MgO doped Lithium Niobate crystals.

A detailed theoretical framework was presented to give a clear, essential and fundamental understanding of the underlying assumptions and models presented. A consistent notation was introduced and the theory was mainly combined from [5, 18, 29, 31, 32, 35].

Simulation was implemented in *Matlab R2013a* and numerous optimizations was accounted for in order to decrease computation time, simplify the complexity and ease the understanding of the code [16, 25, 33, 35].

A real physical electromagnetic field has both a spatial and temporal confinement and should be treated as such. It was argued that both the temporal dependence and the transverse spatial dependence could, in some distinct cases, be neglected. This led to three basis models that was analysed as described below.

10.1 PLANE WAVE

Difference frequency generation (DFG) of monochromatic plane waves was analysed analytically and numerically and compared to results of [5, 31, 32].

It was shown that it was favourable to use the plane wave theory whenever possible since the computation time is extremely small and the model facilitates the use of *adaptive step-sizes* when using the *Runge-kutta 45* method. This enables a more precise and optimized (low computation time) algorithm to solve a system of ordinary differential equations. It was argued the the case of weak coupling (non-depletion) has analytical solutions but is only valid for relatively small pump powers and a numerical model accounting for depletions is therefore needed when dealing with high intensity fields.

It was shown that the analytical non depleted theory did not agree with the numerical simulations at a pump irradiance above $\approx 200\,\mathrm{GW/\,m^2}$. This was for perfect phase match and for the specific crystals and interacting wavelengths that were described in Chapter 7.

10.2 SPATIAL DEPENDENCE

DFG of monochromatic waves having a transverse spatial dependence were analysed numerically and compared to results of [7] which was in good agreement.

Simulations included spatially confined mode in different limit; weakly confined modes in weak and strong coupling regime and highly confined modes in weak and strong coupling regime all of which was compared to the simpler plane wave simulations.

It was firstly shown that monochromatic fields having a transverse spatial dependence while no temporal dependence, except that of the CW carrier wave, could be described by expanding the field on a basis of Gaussian modes. A circular symmetry was assumed and hence a Laguerre-Gaussian basis was chosen for the simulations presented.

Lastly it was shown how the divergence of tightly focused Gaussian beams created a high pump irradiance near the beam waist, causing a significant conversion within that short section of the crystal. This effect led to depletion of the pump signal while the conversion of a less focused pump beam would cause only a small conversion and hence no depletion.

10.3 PULSES

DFG for quasi-monochromatic fields were analysed using a *Split-step* method where the pulses were assumed to have no transverse spatial dependence (plane wave).

Each point on pulses of different temporal length was simulated using a *Split-step Fourier* method and compared to the aforementioned plane wave theory. It was shown that *slow* pulses of temporal duration of 7 ns could adequately be described by CW plane wave theory while 7 ps pulses could not be be described adequately without the implementation of the *Split-step Fourier* method. The simulations were done by introducing a retarded time frame utilizing that dispersion was negligible and hence separating the time dependence from the spatial dependence.

The simulation can thus be carried out as parallel processes where each temporal point is simulated simultaneously and thereby decreasing the simulation time significantly.This means that the Gaussian simulations could be preformed for each temporal point on the pulse separately and simultaneously using parallel processing.

For temporal pulses where dispersion is not negligible the treatment becomes significantly more complicated. The Gaussian simula-

tions could perhaps be implemented as the non-linear propagation between step 5 and step 6 in the Split-step model seen in Figure 15. This is still using the approximation that the pulses are so slow that group velocity dispersion can still be ignored. If group velocity dispersion is to be included then the propagation in time space is not simply a phase change and another approach than the proposed *Split-step Fourier method* must be used.

10.4 MEASUREMENTS

Measurements of DFG for nano second pulses indicated the process of temporal depletion. Simulations of nano second pulses also indicated the same behaviour of temporal depletion. Spatial depletion was however also indicated by the simulations of spatial confined CW fields and the type of depletion measured in the measurements should be subject to further investigation before it can with certainty be stated that the depletion measured is the phenomena of temporal depletion.

11

OUTLOOK

The aim was to develop a relatively fast, numerical tool that could simulate the propagation of electromagnetic fields in MgO:PPLN crystals. This has been done using a variety of different assumptions some more strict than others. The validity condition of these assumptions should be carefully examined and some have been examined in this thesis while others have been omitted.

It has been mentioned that dealing with nonlocal responses and expanding the order of the Born approximation will significantly effect the complexity and computation time of the simulations. It could be possible, without excessive increase of computation time, to include *group velocity dispersion* in the *Split-step* method and drop the *Slowly Varying Envelope Approximation* applied in all the presented models. This has not been tested nor analysed thoroughly but would be an obviously further development of the models.

Combining the models simulating the propagation of spatial continuous waves and those simulation plane wave pulses would be possible. This would also increase the computation time significantly but probably not so dramatically that experiments would be less time consuming. This would therefore constitute an essential tool when planning experiments since a variety of parameters can be varied with a minimum of effort.

Part IV

APPENDIX

A

AMPLITUDE MODULATION

Imagine a *carrier wave* oscillating at frequency ω_0 with amplitude A given by

$$\zeta(t) = A\exp(-i\omega_0 t) + c.c. \tag{120}$$

Imagine then a harmonic function oscillating at a frequency ω_σ much smaller than the carrier frequency, $\omega_\sigma << \omega_0$, given by

$$m(t) = M\exp(-i\omega_\sigma t) + c.c. \tag{121}$$

where the real parameter $|M| < 1$ is the modulation amplitude. A signal given as the carrier wave modulated by the positive harmonic function $1 - m(t)$ gives

$$\begin{aligned}(1+m(t))\,\zeta(t) &= (A\exp(-i\omega_0 t) + c.c)\,(1\\ &\quad + M\exp(-i\omega_\sigma t) + c.c)\\ &= \zeta(t) + AM\exp(-i(\omega_0+\omega_\sigma)t) + A^*M\exp(i(\omega_0-\omega_\sigma)t)\\ &\quad + AM\exp(-i(\omega_0-\omega_\sigma)t) + A^*M\exp(i(\omega_0+\omega_\sigma)t)\\ &= \zeta(t) + [AM\exp(-i(\omega_0+\omega_\sigma)t) + c.c.]\\ &\quad + [AM\exp(-i(\omega_0-\omega_\sigma)t) + c.c.]\,.\end{aligned} \tag{122}$$

This shows that a harmonic modulation at frequency $\omega_\sigma << \omega_0$ will cause the signal to consist of the carrier frequency ω_0 and the slightly shifted frequencies $\omega_+ = \omega_0 + \omega_\sigma$ and $\omega_- = \omega_0 - \omega_\sigma$, both having a power smaller than the carrier frequency.

A general modulation is not harmonic but can be Fourier expanded in harmonic functions. The spectra of the modulated signal will, by the same logic as for the harmonic modulation, be centred around the carrier frequency ω_0 and the bandwidth of the signal will depend on the modulation. A slowly varying modulation consists of small frequencies meaning that the signal bandwidth is narrow while a rapidly varying modulation means that the signal bandwidth is broad.

B

MONOCHROMATIC FIELDS

An electric field consisting of a finite number of monochromatic waves is given by

$$\mathbf{E}(\mathbf{r},t) = \frac{1}{2}\sum_{j\geqslant 0}\left[\mathbf{A}_j(\mathbf{r},t)e^{-i\omega_j t} + \mathbf{A}_j^*(\mathbf{r},t)e^{i\omega_j t}\right] \tag{123}$$

Fouriere transforming (123) gives

$$\begin{aligned}
\mathbf{E}(\mathbf{r},\omega) &= \int_{-\infty}^{\infty}\mathbf{E}(\mathbf{r},t)e^{i\omega t}\,dt \\
&= \frac{1}{2}\sum_{j\geqslant 0}\int_{-\infty}^{\infty}\left[\mathbf{A}_j(\mathbf{r},t)e^{-i\omega_j t} + \mathbf{A}_j^*(\mathbf{r},t)e^{i\omega_j t}\right]e^{i\omega t}\,dt \\
&= \frac{1}{2}\sum_{j\geqslant 0}\left[\mathbf{A}_j(\mathbf{r},t)\int_{-\infty}^{\infty}e^{i(\omega-\omega_j)t}\,dt + \mathbf{A}_j^*(\mathbf{r},t)\int_{-\infty}^{\infty}e^{i(\omega+\omega_j)t}\,dt\right] \\
&= \frac{1}{2}\sum_{j\geqslant 0}\left[\mathbf{A}_j(\mathbf{r},t)\delta(\omega-\omega_j) + \mathbf{A}_j^*(\mathbf{r},t)\delta(\omega+\omega_j)\right] \qquad (124)
\end{aligned}$$

where $\delta(\omega \pm \omega_j)$ is the *Dirac Delta Function* given by

$$\delta(\omega\pm\omega_j) = \int_{-\infty}^{\infty}e^{i(\omega\pm\omega_j)t}\,dt \tag{125}$$

C

THREE WAVE MIXING

Inserting the induced polarization, (74), (75) and (76), into Equation (61) yields

$$2ik_{0,1}\frac{d}{dz}A_1 = \frac{-\omega_{0,1}^2}{\epsilon_0 c^2}\epsilon_0 4\chi_{eff}^{(2)}\frac{1}{2}A_3^*(z)e^{-ik_{0,3}z}\frac{1}{2}A_2(z)e^{ik_{0,2}z}e^{-ik_{0,1}z} \tag{126}$$

$$2ik_{0,2}\frac{d}{dz}A_1 = \frac{-\omega_{0,2}^2}{\epsilon_0 c^2}\epsilon_0 4\chi_{eff}^{(2)}\frac{1}{2}A_3^*(z)e^{-ik_{0,3}z}\frac{1}{2}A_1(z)e^{ik_{0,1}z}e^{-ik_{0,2}z} \tag{127}$$

$$2ik_{0,3}\frac{d}{dz}A_1 = \frac{-\omega_{0,3}^2}{\epsilon_0 c^2}\epsilon_0 4\chi_{eff}^{(2)}\frac{1}{2}A_1(z)e^{ik_{0,1}z}\frac{1}{2}A_2(z)e^{ik_{0,2}z}e^{-ik_{0,3}z} \tag{128}$$

Using now that $k_{0,j} = \frac{n(\omega_{0,j})\omega_{0,j}}{c}$ and $c = \sqrt{1/\epsilon_0\mu_0}$ gives

$$\frac{d}{dz}A_1 = \frac{i\omega_{0,1}}{2n(\omega_{0,1})}\chi_{eff}^{(2)}A_3^*(z)A_2(z)e^{-i(k_{0,3}-k_{0,2}+k_{0,1})z} \tag{129}$$

$$\frac{d}{dz}A_2 = \frac{i\omega_{0,2}}{2n(\omega_{0,2})}\chi_{eff}^{(2)}A_3^*(z)e^{-ik_{0,3}z}A_1(z)e^{ik_{0,1}z}e^{-i(k_{0,3}-k_{0,1}+k_{0,2})z} \tag{130}$$

$$\frac{d}{dz}A_3 = \frac{i\omega_{0,3}}{2n(\omega_{0,2})}\chi_{eff}^{(2)}A_1(z)A_2(z)e^{-i\Delta kz} \tag{131}$$

rewriting $A_j(z)$ as $A_j(z) = \sqrt{2\eta_j\hbar\omega_{0,j}}a_j(z)$ yields

$$\frac{d}{dz}a_1 = \frac{i\omega_{0,1}}{2n(\omega_{0,1})}\chi_{eff}^{(2)}\frac{\sqrt{2\eta_\hbar\omega_{0,3}}\sqrt{2\eta_2\hbar\omega_{0,2}}}{\sqrt{2\eta_1\hbar\omega_{0,1}}} a_3^*(z)a_2(z)e^{-i(k_{0,3}-k_{0,2}+k_{0,1})z} \tag{132}$$

$$\frac{d}{dz}a_2 = \frac{i\omega_{0,2}}{2n(\omega_{0,2})}\chi_{eff}^{(2)}\frac{\sqrt{2\eta_\hbar\omega_{0,3}}\sqrt{2\eta_1\hbar\omega_{0,1}}}{\sqrt{2\eta_2\hbar\omega_{0,2}}} a_3^*(z)a_1(z)e^{-i(k_{0,3}-k_{0,1}+k_{0,2})z} \tag{133}$$

$$\frac{d}{dz}a_3 = \frac{i\omega_{0,3}}{2n(\omega_{0,2})}\chi_{eff}^{(2)}\frac{\sqrt{2\eta_1\hbar\omega_{0,1}}\sqrt{2\eta_2\hbar\omega_{0,2}}}{\sqrt{2\eta_3\hbar\omega_{0,3}}} e^{-i(k_{0,3}-k_{0,1}-k_{0,2})z} \tag{134}$$

Letting now $\eta_j^2 = \frac{\mu_0}{\epsilon_0}\frac{1}{n(\omega_{0,j})^2}$ yields

$$\frac{d}{dz}a_1 = \frac{i\epsilon_0}{2n(\omega_{0,1})}\chi^{(2)}_{eff}\sqrt{2\eta_1\eta_2\eta_3\hbar\omega_{0,1}\omega_{0,2}\omega_{0,3}}\, a_3^*(z)a_2(z)e^{-i(k_{0,3}-k_{0,2}+k_{0,1})z} \tag{135}$$

$$\frac{d}{dz}a_2 = \frac{i\epsilon_0}{2n(\omega_{0,2})}\chi^{(2)}_{eff}\sqrt{2\eta_1\eta_2\eta_3\hbar\omega_{0,1}\omega_{0,2}\omega_{0,3}}\, a_3^*(z)a_1(z)e^{-i(k_{0,3}-k_{0,1}+k_{0,2})z} \tag{136}$$

$$\frac{d}{dz}a_3 = \frac{i\epsilon_0}{2n(\omega_{0,2})}\chi^{(2)}_{eff}\sqrt{2\eta_1\eta_2\eta_3\hbar\omega_{0,1}\omega_{0,2}\omega_{0,3}}\, a_1(z)a_2(z)e^{-i(k_{0,3}-k_{0,1}-k_{0,2})z} \tag{137}$$

Lastly rewriting $\Delta k = k_{0,3} - k_{0,2} - k_{0,1}$, $\eta^3 = \eta_1\eta_2\eta_3$ and $g = \epsilon_0\chi^{(2)}_{eff}\sqrt{\frac{1}{2}\eta^3\hbar\omega_{0,1}\omega_{0,2}\omega_{0,3}}$ we get the final equations

$$\frac{d}{dz}a_1 = i\epsilon_0 g a_3^*(z)a_2(z)e^{-i(k_{0,3}-k_{0,2}+k_{0,1})z} \tag{138}$$

$$\frac{d}{dz}a_2 = i\epsilon_0 g a_3^*(z)a_1(z)e^{-i(k_{0,3}-k_{0,1}+k_{0,2})z} \tag{139}$$

$$\frac{d}{dz}a_3 = i\epsilon_0 g a_1(z)a_2(z)e^{-i(k_{0,3}-k_{0,1}-k_{0,2})z} \tag{140}$$

D

CARRIER WAVE

Matlab script used to create Figure 2.

```
%%
%  Plot of a 20fs pulse
%  consisting of a carrier wave (blue) at 1064nm in
%  air with a gaussian amplitude modulation (red)
%  with a FWHM of 20fs
%
% The form of the amplitude modulation is gaussian:
% A(t) = A0 exp(-t^2 / 2 sigma^2)
% The intensity is proportional to A(t)^2 that is also
% gaussian and has the form
% A(t)^2 = exp(-t^2 / sigma^2)
% where sigma = T0 / sqrt(2)
% sigma is the standart deviation while
% FWHM of the intensity is; FWHM = 2 sigma sqrt( ln(2) )

%% Initialize
clc;
clear all;
close all;

%% Constants
c = 3e8;            %Speed of light in [m/s]

%% Independent Parameters
A0      = 1;         %Amplitude
tsteps  = 10001;     %Number of time steps
FWHM    = 20e-15;    %FWHM of the intensity in [s]
lambda  = 1064e-9;   %Free space wavelength of the carrier wave
    in [m]

%% Dependent parameters
omega = 2*pi*c/lambda;
t = linspace(-3*FWHM,3*FWHM,tsteps);
Amplitude = A0.*gaussmf(t,[FWHM/sqrt(2) 0]);

CarrierWave = 1/2*exp(-1i*omega*t) + 1/2*exp(1i*omega*t);
E = CarrierWave .* Amplitude;

%% Plotting the amplityde squared and the resulting intensity
figure(1)
```

```
40 p21 = plot(t.*1e15,Amplitude,'r','LineWidth',1.5);
41 hold on;
42 p22 = plot(t.*1e15,E ./ E(ceil(tsteps/2)),'b','LineWidth',1);
43 hold off;
44
45 %% Setting axis, labels, legends, plotsize and saving the plot
46 xlim([t(1)*1e15 t(end)*1e15]);
47 title('Amplitude modulated signal','Interpreter','latex','
       FontSize',18);
48 l2 = legend([p21 p22], '$A(t)^2$','$E(t)/E(0)$','location','
       northeast');
49 set(l2,'Interpreter','latex','color','white');
50 PlotSetAxisLabels('$t~[fs]$','$E~[a.u.]$',18);
51 PlotSetFigureDimensions(20,2,20,10,0,0,0,0);
52 PlotSetAxisProperties(5,3,20);
53 set(gca, 'Units', 'normalized','Position', [0.15 0.22 0.77 0.65])
       ;
54 set(gca, 'Color', 'none'); % Sets axes background
55 export_fig EnvelopeFunction -pdf -transparent;
```

E

PROGRESSBAR

Matlab script used to create a processbar that tells the current stage of the simulation and calculates the estimated time left. Originally written by *Steve Hoelzer* and edited by *Quan Quach* and *Anders Bilfeldt*.

```
%% Description
%   progressbar(fractiondone,position) provides an indication of the progress of
% some task using graphics and text. Calling progressbar repeatedly will update
% the figure and automatically estimate the amount of time remaining.
%   This implementation of progressbar is intended to be extremely simple to use
% while providing a high quality user experience.
%
%% Features
%   - Can add progressbar to existing m-files with a single line of code.
%   - The figure closes automatically when the task is complete.
%   - Only one progressbar can exist so old figures don't clutter the desktop.
%   - Remaining time estimate is accurate even if the figure gets closed.
%   - Minimal execution time. Won't slow down code.
%   - Random color and position options. When a programmer gets bored....
%
%% Usage
%   fractiondone specifies what fraction (0.0 - 1.0) of the task is complete.
% Typically, the figure will be updated according to that value. However, if
% fractiondone == 0.0, a new figure is created (an existing figure would be
% closed first). If fractiondone == 1.0, the progressbar figure will close.
%   position determines where the progressbar figure appears on screen. This
% argument only has an effect when a progress bar is first created or is reset
% by calling with fractiondone = 0. The progress bar's position can be specifed
```

```
% as follows:
%       [x, y]  - Position of lower left corner in normalized units (0.0 - 1.0)
%           0   - Centered (Default)
%           1   - Upper right
%           2   - Upper left
%           3   - Lower left
%           4   - Lower right
%           5   - Random [x, y] position
%   The color of the progressbar is choosen randomly when it is created or
% reset. Clicking inside the figure will cause a random color change.
%   For best results, call progressbar(0) (or just progressbar) before starting
% a task. This sets the proper starting time to calculate time remaining.
%
%% Example Function Calls
%   progressbar(fractiondone,position)
%   progressbar               % Initialize/reset
%   progressbar(0)            % Initialize/reset
%   progressbar(0,4)          % Initialize/reset and specify position
%   progressbar(0,[0.2 0.7])  % Initialize/reset and specify position
%   progressbar(0.5)          % Update
%   progressbar(1)            % Close
%
% Demo:
%   n = 1000;
%   progressbar % Create figure and set starting time
%   for i = 1:n
%       pause(0.01) % Do something important
%       progressbar(i/n) % Update figure
%   end
%
%% Author: Steve Hoelzer
%
% Revisions:
% 2002-Feb-27   Created function
% 2002-Mar-19   Updated title text order
% 2002-Apr-11   Use floor instead of round for percentdone
% 2002-Jun-06   Updated for speed using patch (Thanks to waitbar.m)
% 2002-Jun-19   Choose random patch color when a new figure is created
% 2002-Jun-24   Click on bar or axes to choose new random color
```

```
% 2002-Jun-27   Calc time left, reset progress bar when fractiondone == 0
% 2002-Jun-28   Remove extraText var, add position var
% 2002-Jul-18   fractiondone input is optional
% 2002-Jul-19   Allow position to specify screen coordinates
% 2002-Jul-22   Clear vars used in color change callback routine
% 2002-Jul-29   Position input is always specified in pixels
% 2002-Sep-09   Change order of title bar text
% 2003-Jun-13   Change 'min' to 'm' because of built in function 'min'
% 2003-Sep-08   Use callback for changing color instead of string
% 2003-Sep-10   Use persistent vars for speed, modify titlebarstr
% 2003-Sep-25   Correct titlebarstr for 0% case
% 2003-Nov-25   Clear all persistent vars when percentdone = 100
% 2004-Jan-22   Cleaner reset process, don't create figure if percentdone = 100
% 2004-Jan-27   Handle incorrect position input
% 2004-Feb-16   Minimum time interval between updates
% 2004-Apr-01   Cleaner process of enforcing minimum time interval
% 2004-Oct-08   Seperate function for timeleftstr, expand to include days
% 2004-Oct-20   Efficient if-else structure for sec2timestr
% 2017-Dec-12   Modifyed by Quan Quach (quan.quach@gmail.com)
% 2014-Jun-17   Modifyed by Anders Bilfeldt (anders@bilfeldt.dk)
%

function [stopBar] =  progressbar(fractiondone, position)

if(~exist('fractiondone'))
    return
end

stopBar = 0;
persistent progfig progpatch starttime lastupdate firstIteration

% Set defaults for variables not passed in
if nargin < 1
    fractiondone = 0;
end
if nargin < 2
    position = 0;
end

try
    % Access progfig to see if it exists ('try' will fail if it doesn't)
    dummy = get(progfig,'UserData');
```

```
    % If progress bar needs to be reset, close figure and set
        handle to empty
    if fractiondone == 0
        delete(progfig) % Close progress bar
        progfig = []; % Set to empty so a new progress bar is
            created
    end
catch
    progfig = []; % Set to empty so a new progress bar is created
end


percentdone = floor(100*fractiondone);

% Create new progress bar if needed

if (isempty(progfig) && (isempty(firstIteration)))
    firstIteration = 1;
    % Calculate position of progress bar in normalized units
    scrsz = [0 0 1 1];
    width = scrsz(3)/4;
    height = scrsz(4)/50;
    if (length(position) == 1)
        hpad = scrsz(3)/64; % Padding from left or right edge of
            screen
        vpad = scrsz(4)/24; % Padding from top or bottom edge of
            screen
        left   = scrsz(3)/2 - width/2; % Default
        bottom = scrsz(4)/2 - height/2; % Default
        switch position
            case 0 % Center
                % Do nothing (default)
            case 1 % Top-right
                left   = scrsz(3) - width  - hpad;
                bottom = scrsz(4) - height - vpad;
            case 2 % Top-left
                left   = hpad;
                bottom = scrsz(4) - height - vpad;
            case 3 % Bottom-left
                left   = hpad;
                bottom = vpad;
            case 4 % Bottom-right
                left   = scrsz(3) - width  - hpad;
                bottom = vpad;
            case 5 % Random
                left   = rand * (scrsz(3)-width);
                bottom = rand * (scrsz(4)-height);
            otherwise
```

```
149                 warning('position must be (0-5). Reset to 0.')
150             end
151             position = [left bottom];
152         elseif length(position) == 2
153             % Error checking on position
154             if (position(1) < 0) | (scrsz(3)-width < position(1))
155                 position(1) = max(min(position(1),scrsz(3)-width),0);
156                 warning('Horizontal position adjusted to fit on
                        screen.')
157             end
158             if (position(2) < 0) | (scrsz(4)-height < position(2))
159                 position(2) = max(min(position(2),scrsz(4)-height),0)
                        ;
160                 warning('Vertical position adjusted to fit on screen.
                        ')
161             end
162         else
163             error('position is not formatted correctly')
164         end
165
166         % Initialize progress bar
167         progfig = figure(...
168             'Units',            'normalized',...
169             'Position',         [position width height],...
170             'NumberTitle',      'off',...
171             'Resize',           'off',...
172             'MenuBar',          'none',...
173             'BackingStore',     'off' );
174         progaxes = axes(...
175             'Position',         [0.02 0.15 0.96 0.70],...
176             'XLim',             [0 1],...
177             'YLim',             [0 1],...
178             'Box',              'on',...
179             'ytick',            [],...
180             'xtick',            [] );
181         progpatch = patch(...
182             'XData',            [0 0 0 0],...
183             'YData',            [0 0 1 1],...
184             'EraseMode',        'none' );
185
186 % enable this code if you want the bar to change colors when the
187 % user clicks on the progress bar
188 %     set(progfig,  'ButtonDownFcn',{@changecolor,progpatch});
189 %     set(progaxes, 'ButtonDownFcn',{@changecolor,progpatch});
190 %     set(progpatch,'ButtonDownFcn',{@changecolor,progpatch});
191 %     changecolor(0,0,progpatch)
192
193         set(progpatch,'FaceColor',[.1 1 .1]);
```

```
194
195     % Set time of last update to ensure a redraw
196     lastupdate = clock - 1;
197
198     % Task starting time reference
199     if isempty(starttime) | (fractiondone == 0)
200         starttime = clock;
201     end
202
203     set(progfig,'CloseRequestFcn',@closeBar);
204
205 end
206
207 %if the user closes the progress bar during the data processing
208 %then this will erase all the variables are return 1 to the
        output
209 if (isempty(progfig) && ~(fractiondone==0))
210     delete(progfig) % Close progress bar
211
212     % Clear persistent vars
213     clear progfig progpatch starttime lastupdate firstIteration
214     stopBar = 1;
215     return
216
217 end
218
219
220 %Enforce a minimum time interval between updates
221 %but allows for the case when the bar reaches 100% so that the
        user can see
222 %it
223 if (etime(clock,lastupdate) < 0.01 && ~(percentdone == 100))
224     return
225 end
226
227 % Update progress patch
228 set(progpatch,'XData',[0 fractiondone fractiondone 0])
229
230 % Update progress figure title bar
231 if (fractiondone == 0)
232     titlebarstr = ' 0%';
233 else
234     runtime = etime(clock,starttime);
235     timeleft = runtime/fractiondone - runtime;
236     timeleftstr = sec2timestr(timeleft);
237     titlebarstr = sprintf('%2d%%    %s remaining',percentdone,
            timeleftstr);
238 end
```

```
239 set(progfig,'Name',titlebarstr)
240
241 % Force redraw to show changes
242 drawnow
243
244 % If task completed, close figure and clear vars, then exit
245
246 if percentdone == 100 % Task completed
247 delete(progfig) % Close progress bar
248
249 %change the close request function back to normal
250 % Clear persistent vars
251 clear progfig progpatch starttime lastupdate firstIteration
252 return
253 end
254 % Record time of this update
255 lastupdate = clock;
256
257 %% Fucntion to change color of the progressbar
258 function changecolor(h,e,progpatch)
259 Change the color of the progress bar patch
260
261 colorlim = 2.8; % Must be <= 3.0 - This keeps the color from
        being too light
262 thiscolor = rand(1,3);
263 while sum(thiscolor) > colorlim
264     thiscolor = rand(1,3);
265 end
266 set(progpatch,'FaceColor',thiscolor);
267
268
269 %% Function to estimate the resisting calculation time
270 function timestr = sec2timestr(sec)
271 % Convert a time measurement from seconds into a human readable
        string.
272
273 % Convert seconds to other units
274 d = floor(sec/86400); % Days
275 sec = sec - d*86400;
276 h = floor(sec/3600); % Hours
277 sec = sec - h*3600;
278 m = floor(sec/60); % Minutes
279 sec = sec - m*60;
280 s = floor(sec); % Seconds
281
282 % Create time string
283 if d > 0
284     if d > 9
```

```
285         timestr = sprintf('%d day',d);
286     else
287         timestr = sprintf('%d day, %d hr',d,h);
288     end
289 elseif h > 0
290     if h > 9
291         timestr = sprintf('%d hr',h);
292     else
293         timestr = sprintf('%d hr, %d min',h,m);
294     end
295 elseif m > 0
296     if m > 9
297         timestr = sprintf('%d min',m);
298     else
299         timestr = sprintf('%d min, %d sec',m,s);
300     end
301 else
302     timestr = sprintf('%d sec',s);
303 end
304
305 %%
306 function closeBar(src,evnt)
307
308 selection = questdlg('Do you want to stop this process?',...
309                      'Stop process',...
310                      'Yes','No','Yes');
311 switch selection,
312    case 'Yes',
313     delete(gcf)
314    case 'No'
315      return
316 end
```

F

TOLERENCES

Matlab script used to simulate *difference frequency generation* for monochromatic plane waves using Equation (77), (78) and (79). Values used can be seen in Table 4.

This script calls three other scripts, the *progressbar* seen in Appendix E, the *Scalar DFG* script seen in Appendix H and the equations to solve, Appendix I.

F.1 NUMERICAL VALUES

A number of constants is used in the simulations presented. Table 4 shows the constants used.

NAME	MATLAB	VALUE
Signal pump power	*Signal.Po*	20 µW
Idler pump power	*Idler.Po*	20 µW
Pump pump power	*Pump.Po*	16 kW
QPM order	*QPM*	1

Table 4: Values used in the simulation.

F.2 MATLAB CODE

```
%% Simulate the effect of
% A) A QPM domain structure
% B) An effective susceptiblity approach
% by varying the tolerences used in the RungeKutta45 method


%% Initialize
close all;                      % Close all figures
clear all;                      % Clear all variabels
clc;                            % Clear command window

%% Parameters
```

```
13 Tolerance.Abs = logspace(-1,-8,40); %linspace(1e-1,1e-5,4);  %
       Absolute tolerence: logspace or linspace
14 Tolerance.Rel = Tolerance.Abs ./ 10;    % Relative tolerence
15 Signal.P0     = 20e-6;                   % Signal peak power
16 Idler.P0      = Signal.P0;             % Idler peak power
17 Pump.P0       = 16.*1e3;                 % Pump peak power
18 zsteps        = 5000;                    % Number of z-steps
19 zlength       = 40e-3;                   %Length of nonlinear
       crystal in [m]
20 QPM           = 1;                       % Use QPM or effective
       QPM
21 QPMtype       = 'domain';                   % Type of QPM: domain
       og eff.
22
23
24 %% Allocate memory
25 Signal.P    = zeros(size(Tolerance.Abs,2),zsteps);
26 Idler.P     = zeros(size(Tolerance.Abs,2),zsteps);
27 Pump.P      = zeros(size(Tolerance.Abs,2),zsteps);
28 Total.P     = zeros(size(Tolerance.Abs,2),zsteps);
29 Signal.N    = zeros(size(Tolerance.Abs,2),zsteps);
30 Idler.N     = zeros(size(Tolerance.Abs,2),zsteps);
31 Pump.N      = zeros(size(Tolerance.Abs,2),zsteps);
32 Total.N     = zeros(size(Tolerance.Abs,2),zsteps);
33
34 %% Run simulation
35 progressbar; % Initialize processbar
36 for nn=1:size(Tolerance.Abs,2)
37
38     % Run a 'ScalarDFG' simulation
39     [z, output1, output2, output3, output4, output5, output6,
           output7, ...
40         output8] = ScalarDFG(Tolerance.Abs(nn), Tolerance.Rel(nn)
               , ...
41         Signal.P0, Idler.P0, Pump.P0,zsteps,zlength,QPM,QPMtype);
42
43     % Save the calculated powers for signal and pump.
44     Signal.P(nn,:)  = output1;
45     Idler.P(nn,:)   = output2;
46     Pump.P(nn,:)    = output3;
47     Total.P(nn,:)   = output4;
48     Signal.N(nn,:)  = output5;
49     Idler.N(nn,:)   = output6;
50     Pump.N(nn,:)    = output7;
51     Total.N(nn,:)   = output8;
52
53     % Update processbar
54     progressbar( nn/size(Tolerance.Abs,2) );
```

```
55
56 end
```

G

FOURIER TRANSFORM AND ITS INVERSE

Matlab script used to Fourier transform and inverse Fourier transform. Borrowed from [33]

Fourier transforming from time domain to frequency domain.

```
function y = FourierT(x, dt)
    % FourierT(x,dt) computes forward FFT
    % of x with sampling time interval dt
    % FourierT approximates the
    % Fourier transform where the integrand of the
    % transform is x*exp(2*pi*i*f*t)
    % For NDE applications the frequenc
    % y components are normally in MHz,
    % dt in microseconds

    [nr, nc] = size(x);

    if nr == 1
        N = nc;
    else
        N = nr;
    end

    y = N*dt*ifft(x);

end
```

Fourier transforming from frequency domain to time domain.

```
function y = IFourierT(x, dt)
    % IFourierT(x,dt) computes the inverse FFT
    % of x, for a sampling time interval dt
    % IFourierT assumes the integrand of the inverse transform is
        given by
    % x*exp(-2*pi*i*f*t)
    % The first half of the sampled values
    % of x are the spectral components for
    % positive frequencies ranging from 0
    % to the Nyquist frequency 1/(2*dt)
    % The second half of the sampled values
    % are the spectral components for
    % the corresponding negative freque
    % ncies. If these negative frequency
```

```
    % values are set equal to zero then to
    % recover the inverse FFT of x we must
    % replace x(1) by x(1)/2 and then
    % compute 2*real(IFourierT(x,dt))

    [nr,nc] = size(x);

    if nr == 1
        N = nc;
    else
        N = nr;
    end

    y =(1/(N*dt))*fft(x);

end
```

H

SCALAR DFG

Matlab script used to simulate *difference frequency generation* for monochromatic plane waves using Equation (77), (78) and (79).

```
%% Scalar Difference Frequency Generation
% Simulates DFG for monochromatic, CW, plane waves.
%
% Input:
%   TolAbs  Absolute tolerence used in RungeKutta45
%   TolRel  Relative tolerence used in RungeKutta45
%   Pp1     Signal seed power
%   Pp2     Idler seed power
%   Pp3     Pump power
%   zsteps  Number of steps that shall be returned from the
    RungeKutta45
%   zlength Length of crystal
%   QPM     Order of QPM structure
%   QPMtype Type of QPM simulation; 'eff' (effeective) or 'domain
    '
%
% Output:
%   output0     z axis
%   output1     Signal power;
%   output2     Idler power;
%   output3     Pump power
%   output4     Total power
%   output5     Signal photon number
%   output6     Idler photon number
%   output7     Pump photon number
%   output8     Total photon number


function [output0, output1, output2, output3, output4, output5,
    output6,...
                    output7, output8] = ScalarDFG(TolAbs, TolRel,
                        Pp1,...
                    Pp2, Pp3, zsteps,zlength, QPM, QPMtype)

%% Independent Parameters
Idler.Lambda  = 1.570e-6;           %wavelength of idler in [m]
Pump.Lambda   = 1.064e-6;           %wavelength of pump in [m]
Param.Length  = zlength;            %Length of nonlinear crystal
    in [m]
```

```
35 Param.Area    = pi*(40e-6)^2;       %Area of the nonlinear
       crystal in [m^2]
36 Param.Temp    = 88;                 %Temperature in degree
37 Param.QPM     = QPM;                %QPM: 0=NON, 1=1st order, 2=2
       th order,...
38 Param.QPMtype = QPMtype;            %Type of QPM: domain og eff.
39 Param.zsteps  = zsteps;             %Number of z-steps
40
41
42 %% Boundary conditions
43 Signal.PeakPower    = Pp1;         %power in watt - signal
44 Idler.PeakPower     = Pp2;         %power in watt - idler
45 Pump.PeakPower      = Pp3;         %power in watt - pump
46
47
48 %% constants
49 const.c           = 299792458;           %Speed of light in [m/s
       ]
50 const.hbar        = 1.05457173e-34; %const.hbar in [m2*kg/s]
51 const.epsilon0    = 8.854187817e-12;%in [F/m]
52 const.mu0         = pi*4e-7;        %in [V*s/A*m]
53 const.chi         = 22e-12;       %eff. nonlinear suceptibility
       in [m/V]
54
55 %% Dependent Parameters
56 Idler.Omega     = 2*pi*const.c / Idler.Lambda;
57 Pump.Omega      = 2*pi*const.c / Pump.Lambda;
58 Signal.Omega    = Pump.Omega - Idler.Omega;
59 Signal.Lambda   = 2*pi*const.c / Signal.Omega;
60
61 %% Refractive index from Sellmeier equations
62 [n , kVector] = sellmeier([Signal.Lambda Idler.Lambda Pump.Lambda
       ],Param.Temp);
63 % k vector
64 Signal.kVector  = kVector(1);
65 Idler.kVector   = kVector(2);
66 Pump.kVector    = kVector(3);
67 % refractive index
68 Signal.n        = n(1);
69 Idler.n         = n(2);
70 Pump.n          = n(3);
71
72 deltak  = Pump.kVector - Signal.kVector - Idler.kVector;
73 lcoh    = 2 * pi / deltak;
74 if( strcmp(Param.QPMtype , 'eff') )
75    const.chi  = 2/(pi*Param.QPM)*const.chi;
76    lcoh       = 1e10;
77    deltak     = 0;
```

```
78  end
79
80
81  %% Nonlinear parameters
82  eta             = sqrt( const.mu0 / const.epsilon0 ) / nthroot(
        Signal.n*Idler.n*Pump.n,3);
83  g               = const.epsilon0 * const.chi * sqrt(1/2 * const.
        hbar * eta^3 ...
84                  * Signal.Omega * Idler.Omega * Pump.Omega);
85
86  %calculate amplitudes
87  Signal.A    = sqrt(Signal.PeakPower / (const.hbar*Signal.Omega)/
        Param.Area);
88  Idler.A     = sqrt(Idler.PeakPower  / (const.hbar*Idler.Omega)/
        Param.Area);
89  Pump.A      = 1i*sqrt(Pump.PeakPower   / (const.hbar*Pump.Omega)/
        Param.Area);
90
91  %% Solve set of ODEs
92      % Sets the global and local maximum fault tolerance.
93  options = odeset('AbsTol',TolAbs,'RelTol',TolRel);
94      % Solve ODE: ode45(solve for , interval , init.values ,
            options).
95
96  [z, A]  = ode45(@(z,A) ode45equation(z,A,g,deltak,Param.QPM,lcoh)
        , ...
97      linspace(0,Param.Length,Param.zsteps), [Signal.A, Idler.A,
            Pump.A], options);
98
99  %% Find number of photons instead of amplitudes
100 Signal.N    = A(:,1).*conj(A(:,1)) ;
101 Idler.N     = A(:,2).*conj(A(:,2)) ;
102 Pump.N      = A(:,3).*conj(A(:,3)) ;
103 Total.N     = Signal.N + Idler.N + Pump.N;
104
105 %% Find electric fields
106 Signal.E    = A(:,1) * sqrt( 2 * eta * const.hbar * Signal.Omega)
         .* exp(1i .* Signal.kVector .* z);
107 Idler.E     = A(:,1) * sqrt( 2 * eta * const.hbar * Idler.Omega )
         .* exp(1i .* Idler.kVector .* z);
108 Pump.E     = A(:,1) * sqrt( 2 * eta * const.hbar * Pump.Omega )
        .* exp(1i .* Pump.kVector .* z);
109
110 %% Find powers instead of amplitudes
111 Signal.P    = Signal.N .* Param.Area .* const.hbar .* Signal.
        Omega;
112 Idler.P     = Idler.N .* Param.Area .* const.hbar .* Idler.Omega;
113 Pump.P      = Pump.N .* Param.Area .* const.hbar .* Pump.Omega;
```

```
Total.P     = Signal.P + Idler.P + Pump.P;

%% Z axis
Axis.zmili  = z*1e3;

output0 = Axis.zmili;
output1 = Signal.P;
output2 = Idler.P;
output3 = Pump.P;
output4 = Total.P;
output5 = Signal.N;
output6 = Idler.N;
output7 = Pump.N;
output8 = Total.N;

end
```

I

ODE 45 EQUATIONS

Matlab implementation of Equaiton *difference frequency generation* for monochromatic plane waves using Equation (77), (78) and (79).

I.1 MATLAB CODE

```
%% ODE45 equations to be solved
%   The following is considered:
%   QPM


function dA = ode45equation(z,A,g,deltak,QPM,lcoh)
    dA=zeros(3,1);

    %% QPM factor
    %   Effectevely adds a pi phase shift of the induced
    %   polarisation and hente the source term in the
    %   wave-equations every time the wave has traveled
    %   half a coherence length, lcoh
    if QPM == 0
        s=1;
    else
        s = (-1).^floor(2*z/(lcoh*QPM));
    end

    %% Calculates the solution to the coupled differential
        equation
    %   A(1) is the signal
    %   A(2) is the idler
    %   A(3) is the pump
    dA(1)   =   1i*g*A(3)*conj(A(2))    *exp(1i*deltak*z)   * s;
    dA(2)   =   1i*g*A(3)*conj(A(1))    *exp(1i*deltak*z)   * s;
    dA(3)   =   1i*g*A(1)*A(2)          *exp(-1i*deltak*z)  * s;
end
```

J

GAUSSIAN BASIS

Matlab script to create linear transformation matrix, T, than transforms from r− space to Gaussian space as described in Section 6.5.2.

```
%% Create a linear transformation matrix, T
%  that transforms from r-space to Gaussian space.
%
% Input:
%  Lmax     Number of modes in Gaussian space
%  r        The space vector r
%  z        The z-cordinate
%  lambda   Wavelength
%  omega0   Frequency
%  n        Refractive index at frequency omega0
%  deltar   Step size in r-space
%
% Output:
%  T        A (Lmax x size(r,2)) linear transformation matrix

function T = GaussianBasis(Lmax,r,z,lambda,omega0,n,deltar)

  T   = zeros(Lmax,size(r,2));

  for l=1:Lmax
    T(l,:) = conj(r.*2.*pi.*deltar.* GaussianMode(l,r,z,lambda,
        omega0,n) );
  end

end
```

K

SPLIT-STEP METHOD

Matlab script used to simulate *difference frequency generation* for quasi-monochromatic pulses. One subscript, *SplitStepDFG*, is called to execute the *Split-step Fourier method*, this is seen in Appendix L. Two scripts is used to plot the simulations data. The first is named *RUNpowerPlotWaterfall* and is seen in Appendix M while the second is named *RUNpowerPlotPropagation* and is seen in Appendix N.

```
%%
clear all;
close all;
clc;
tic;

%% Simulation paramenters
powersteps            = 10;
power                 = linspace(1,150e3,powersteps);
setting.runSimulation = true;
setting.plot          = true;

%% Indepenent parameters
    ------------------------------------------------
Param.t_FWHM      = 7e-9 * sqrt(2);       %FWHM of (E) pulse in
    time domain
Param.t_steps     = 2^11;                 %Steps of t-axis
Param.t_0         = 11*Param.t_FWHM;      %Span of t; -Param.t_0
    : Param.t_0
Param.z_steps     = 5000;                 %Steps of z-axis
Param.z_0         = 40e-3;                %Span of z; 0 : Param.
    z_0
Param.A_eff       = pi*(50e-6)^2;         %Area of beam
Param.lambda_3    = 1064e-9;              %Wavelength [m] of
    signal
Param.lambda_1    = 1600e-9;              %Wavelength [m] of
    idler
Param.P.peak1     = 20e-6;                %Peak power [W] of
    signal
Param.P.peak2     = 0;                    %Peak power [W] of
    idler
Param.Temp        = 88;                   %Param.Temperature in
    degree
Param.chi_eff     = 22e-12;               %m/V
```

```
26
27  axis.t                 = linspace(-Param.t_0,Param.t_0,Param.
        t_steps);
28  axis.z                 = linspace(0,Param.z_0,Param.z_steps );
29
30  %% Simulation
31  if(setting.runSimulation == true)
32      Idler.I.depleted       = zeros(powersteps,Param.z_steps,Param
            .t_steps);
33      Signal.I.depleted      = zeros(powersteps,Param.z_steps,Param
            .t_steps);
34      Pump.I.depleted        = zeros(powersteps,Param.z_steps,Param
            .t_steps);
35
36      Idler.I.undepleted     = zeros(powersteps,Param.z_steps,Param
            .t_steps);
37      Signal.I.undepleted    = zeros(powersteps,Param.z_steps,Param
            .t_steps);
38      Pump.I.undepleted      = zeros(powersteps,Param.z_steps,Param
            .t_steps);
39
40      Signal.Energy.depleted = zeros(powersteps,Param.z_steps);
41      Idler.Energy.depleted  = zeros(powersteps,Param.z_steps);
42      Pump.Energy.depleted   = zeros(powersteps,Param.z_steps);
43
44      Signal.Energy.undepleted = zeros(powersteps,Param.z_steps);
45      Idler.Energy.undepleted  = zeros(powersteps,Param.z_steps);
46      Pump.Energy.undepleted   = zeros(powersteps,Param.z_steps);
47
48      Axis.t_ns   = zeros(powersteps,Param.t_steps);
49      Axis.z_mm   = zeros(powersteps,Param.z_steps);
50
51      % Initialize progressbar
52      progressbar;
53
54      for nn=1:powersteps
55
56          [AX1, AX2, A, B, C, D, E, F] = SplitStepDFG(power(nn),
                Param.t_steps,Param.t_FWHM,true,false);
57
58          Axis.t_ns(nn,:) = AX1;
59          Axis.z_mm(nn,:) = AX2;
60
61          Signal.I.depleted(nn,:,:)      = A;
62          Idler.I.depleted(nn,:,:)       = B;
63          Pump.I.depleted(nn,:,:)        = C;
64
65          Signal.Energy.depleted(nn,:)   = D;
```

```
66          Idler.Energy.depleted(nn,:)    = E;
67          Pump.Energy.depleted(nn,:)     = F;
68
69          % Update progressbar
70          progressbar(nn/(1*powersteps));
71
72          [AX1, AX2, A, B, C, D, E, F] = SplitStepDFG(power(nn),
                Param.t_steps,Param.t_FWHM,false,false);
73
74          Axis.t_ns(nn,:) = AX1;
75          Axis.z_mm(nn,:) = AX2;
76
77          Signal.I.undepleted(nn,:,:)      = A;
78          Idler.I.undepleted(nn,:,:)       = B;
79          Pump.I.undepleted(nn,:,:)        = C;
80
81          Signal.Energy.undepleted(nn,:)   = D;
82          Idler.Energy.undepleted(nn,:)    = E;
83          Pump.Energy.undepleted(nn,:)     = F;
84
85          % Update progressbar
86          progressbar(nn/powersteps);
87          disp(['I am ' sprintf('%0.1f',nn/powersteps*100) '% done'
                ]);
88
89      end
90
91      disp(['I am done. It took ' sprintf('%0.2f',toc) ' seconds'])
            ;
92
93      %save(['Workspace-',datestr(now, 'yyyy-mm-dd--HH-MM-SS')]);
94  end
95
96  %% Plot
97
98  if(setting.plot == true)
99
100     SplitStepPlotWaterfall;
101     SplitStepPlotPropagation;
102
103 end
```

L

SPLIT-STEP METHOD

Matlab script used to implement the *Split-step Fourier method* in the simulation of *difference frequency generation* for quasi-monochromatic pulses, Appendix K

```
%%
% DFG
%
% Pump.Omega > Signal.Omega > Idler.Omega
% Pump.Omega - Idler.Omega => Signal.Omega
%

% close all;
% clc;
% clear all;
%tic.All = tic;
function [AX1, AX2, output_1, output_2, output_3, output_4,
    output_5, output_6] = SplitStepDFG(peakpower,tsteps,tfwhm,
    depletion,plotting)

%% Check for optional parameters
    ----------------------------------------
if nargin < 5
  plotting = true;
end
if nargin < 4
  depletion = true;
end
if nargin < 3
  tfwhm = 7e-9 * sqrt(2);
end
if nargin < 2
  tsteps = 2^9;
end
if nargin < 1
  peakpower = 10e3;
end

%% Setting
    ------------------------------------------------------------------

setting.Plot_graphs         = plotting;
setting.SaveFigure          = true;
```

```
34 setting.Movie               = false;
35 setting.Depletion            = depletion;%true;
36
37 %% Indepenent parameters
       ------------------------------------------------
38 Param.t_FWHM      = tfwhm;              %FWHM of (E) pulse in
       time domain
39 Param.t_steps     = tsteps;             %Steps of t-axis (2^9)
40 Param.t_0         = 11*Param.t_FWHM;    %Span of t; -Param.t_0 :
       Param.t_0
41 Param.z_steps     = 5000;               %Steps of z-axis
42 Param.z_0         = 40e-3;              %Span of z; 0 : Param.z_0
43 Param.A_eff       = pi*(50e-6)^2;       %Area of beam
44 Param.lambda_3    = 1064e-9;            %Wavelength [m] of signal
45 Param.lambda_1    = 1600e-9;            %Wavelength [m] of idler
46 Param.P.peak1     = 1e-6;               %Peak power [W] of signal
47 Param.P.peak2     = 0;                  %Peak power [W] of idler
48 Param.P.peak3     = peakpower;%20e3;    %Peak power [W] of pump
49 Param.Temp        = 88;                 %Param.Temperature in
       degree
50
51 %% Constants
       ------------------------------------------------------------

52 const.c           = 3e8;                %Speed of light in [m/s]
53 const.hbar        = 1.05457173e-34;     %Hbar in [m2*kg/s]
54 const.epsilon0    = 8.854187817e-12;    %In [F/m]
55 const.mu0         = pi*4e-7;            %In [V*s/A*m]
56 const.chi_eff     = 22e-12;             %Effective nonlinear
       suceptibility in [m/V]
57
58 %% Dependent parameters
       ------------------------------------------------
59 t_sigma = Param.t_FWHM / (2*sqrt(2*log(2)));
60 t       = linspace(-Param.t_0,Param.t_0,Param.t_steps); %Time
       axis
61 dt      = t(2)-t(1);                                   %Time
       steps
62 f       = linspace(0, 1/dt, Param.t_steps);            %
       Frequency axis
63 f_shift = linspace(-1/(2*dt), 1/(2*dt), Param.t_steps); %Centered
        frequency axis
64 w       = f .*2*pi;                                   %Angular
       frequency axis
65 w_shift = f_shift .*2*pi;                             %Angular
       frequency shifted
66 z       = linspace(0,Param.z_0,Param.z_steps);         %Crystal
       axis
```

```
67  dz      = z(2)-z(1);                                    %Crystal
        steps
68
69  % Frequency and wavelength, DFG
70  Idler.Omega     = 2*pi*const.c / Param.lambda_1;
71  Pump.Omega      = 2*pi*const.c / Param.lambda_3;
72  Signal.Omega    = Pump.Omega - Idler.Omega;
73  lambda_2        = 2*pi*const.c / Signal.Omega;
74
75  % Pulse energy
76
77  % Sellmeier equation
78  [n,kVector] = sellmeier([Param.lambda_1 lambda_2 Param.lambda_3],
        Param.Temp);
79  deltak      = kVector(3) - kVector(2) - kVector(1);
80  lcoh        = 2 * pi / deltak;
81
82  Idler.v         = groupvelocity(Param.lambda_1,Param.Temp);
83  Signal.v        = groupvelocity(lambda_2,Param.Temp);
84  Pump.v          = groupvelocity(Param.lambda_3,Param.Temp);
85
86  % Use effective 1st order QPM
87  deltak          = 0;
88  const.chi_eff   = 2/pi * const.chi_eff;
89
90  %% Initial pulse
        ------------------------------------------------------------
91  Idler.E.t       = zeros(Param.z_steps,Param.t_steps);   %
        Allocate time matrix
92  Idler.E.f       = zeros(Param.z_steps,Param.t_steps);   %
        Allocate frequency matrix
93  Idler.E.t(1,:)   = gaussmf(t,[t_sigma t(ceil(end/2))]);  %
        Defining gaussian pulse
94  Idler.E.t(1,:)   = Idler.E.t(1,:) * sqrt( 2 * Param.P.peak1 ...
95      / (const.c * const.epsilon0 * n(1) * Param.A_eff) );
96
97  Signal.E.t       = zeros(Param.z_steps,Param.t_steps);  %
        Allocate time matrix
98  Signal.E.f       = zeros(Param.z_steps,Param.t_steps);  %
        Allocate frequency matrix
99  Signal.E.t(1,:)   = gaussmf(t,[t_sigma t(ceil(end/2))]); %
        Defining gaussian pulse
100 Signal.E.t(1,:)   = Signal.E.t(1,:) * sqrt( 2 * Param.P.peak2 ...
101     / (const.c * const.epsilon0 * n(2) * Param.A_eff) );
102
103 Pump.E.t        = zeros(Param.z_steps,Param.t_steps);   %Allocate
        time matrix
```

```
104 Pump.E.f        = zeros(Param.z_steps,Param.t_steps);   %Allocate
        frequency matrix
105 Pump.E.t(1,:)   = gaussmf(t,[t_sigma t(ceil(end/2))]);  %Defining
        gaussian pulse
106 Pump.E.t(1,:)   = Pump.E.t(1,:) * sqrt( 2 * Param.P.peak3 ...
107     / (const.c * const.epsilon0 * n(3) * Param.A_eff) );
108
109 %% Propagate pulses
        ----------------------------------------------------------
110 for nn=1:Param.z_steps-1
111
112     %Linear part
113     Idler.E.f(nn,:)    = fftshift(FourierT(Idler.E.t(nn,:), dt));
114     Idler.E.f(nn,:)    = Idler.E.f(nn,:) .* exp(1i.*w_shift.*dz *
            ...
115         (1/Idler.v-1/Pump.v)); %Linear dispersion
116     Idler.E.f(nn,:)    = fftshift( Idler.E.f(nn,:) );
117     Idler.E.t(nn+1,:)  = IFourierT(Idler.E.f(nn,:), dt);
118
119
120     Signal.E.f(nn,:)   = fftshift(FourierT(Signal.E.t(nn,:), dt))
            ;
121     Signal.E.f(nn,:)   = Signal.E.f(nn,:) .* exp(1i.*w_shift.*dz
            * ...
122         (1/Signal.v-1/Pump.v)); %Linear dispersion
123     Signal.E.f(nn,:)   = fftshift( Signal.E.f(nn,:) );
124     Signal.E.t(nn+1,:) = IFourierT(Signal.E.f(nn,:), dt);
125
126     Pump.E.f(nn,:)     = fftshift(FourierT(Pump.E.t(nn,:), dt));
127     Pump.E.f(nn,:)     = Pump.E.f(nn,:) .* exp(1i.*w_shift.*dz *
            ...
128         (1/Pump.v-1/Pump.v)); %Linear dispersion
129     Pump.E.f(nn,:)     = fftshift( Pump.E.f(nn,:) );
130     Pump.E.t(nn+1,:)   = IFourierT(Pump.E.f(nn,:), dt);
131
132     %Non linear part
133     Idler.E.t(nn+1,:)  = Idler.E.t(nn+1,:) + dz.*1i.*Idler.Omega
            ...
134         .* const.chi_eff ./ (2*n(1)*const.c) .* conj( Signal.E.t(
                nn,:) )...
135         .* Pump.E.t(nn,:) .* exp(1i * deltak*dz);
136     Signal.E.t(nn+1,:) = Signal.E.t(nn+1,:) + dz.*1i.*Signal.
            Omega ...
137         .* const.chi_eff ./ (2*n(2)*const.c) .* conj( Idler.E.t(
                nn,:) ) ...
138         .* Pump.E.t(nn,:) .* exp(1i * deltak*dz);
139
140     if( setting.Depletion == true )
```

```
141         Pump.E.t(nn+1,:)    = Pump.E.t(nn+1,:) + dz.*1i.*Pump.
                Omega ...
142             .* const.chi_eff ./ ( 2*n(3) * const.c) .* Idler.E.t(
                    nn,:) ...
143             .* Signal.E.t(nn,:) .* exp(-1i * deltak*dz);
144     end
145
146 end
147
148 %% Calculate Power
        ---------------------------------------------------------
149 Idler.I     = (const.c * const.epsilon0 * n(1) /2) * abs(Idler.E.
        t).^2;
150 Signal.I    = (const.c * const.epsilon0 * n(2) /2) * abs(Signal.E
        .t).^2;
151 Pump.I      = (const.c * const.epsilon0 * n(3) /2) * abs(Pump.E.t
        ).^2;
152
153 Idler.P     = Idler.I   .* Param.A_eff; %(c * epsilon0 * n(1) *
        area /2) * sum(abs(Idler.E.t).^2,2);
154 Signal.P    = Signal.I  .* Param.A_eff; %(c * epsilon0 * n(2) *
        area /2) * sum(abs(Signal.E.t).^2,2);
155 Pump.P      = Pump.I    .* Param.A_eff; %(c * epsilon0 * n(3) *
        area /2) * sum(abs(Pump.E.t).^2,2);
156 Total.P     = Idler.P + Signal.P + Pump.P;
157
158 Idler.Energy    = dt.*sum(Idler.P,2);
159 Signal.Energy   = dt.*sum(Signal.P,2);
160 Pump.Energy     = dt.*sum(Pump.P,2);
161 Total.Energy    = dt.*sum(Total.P,2);
162
163 %% Define axis
        ------------------------------------------------------------
164 Axis.t_ns        = t .* 1e9;
165 Axis.f_gh_shift  = f_shift .* 1e-9;
166 Axis.z_mm        = z .* 1e3;
167
168 %% Calculate FWHM
        ---------------------------------------------------------
169 Idler.FWHM  = zeros(Param.z_steps,3);
170 Signal.FWHM = zeros(Param.z_steps,3);
171 Pump.FWHM   = zeros(Param.z_steps,3);
172 for count_n=1:Param.z_steps
173     Idler.FWHM(count_n,:)   = fwhm( Axis.t_ns , Idler.I(count_n
            ,:)    );
174     Signal.FWHM(count_n,:)  = fwhm( Axis.t_ns , Signal.I(count_n
            ,:)   );
```

```
175     Pump.FWHM(count_n,:)    = fwhm( Axis.t_ns , Pump.I(count_n,:)
            );
176 end
177
178 %% plot figures
    ------------------------------------------------------------
179 if(setting.Plot_graphs == true)
180     SplitStepDFGplot;
181 end
182
183 %% Function output
    ------------------------------------------------------------
184
185 AX1      = Axis.t_ns;
186 AX2      = Axis.z_mm;
187
188 output_1 = Signal.I(:,:);
189 output_2 = Idler.I(:,:);
190 output_3 = Pump.I(:,:);
191
192 output_4 = Signal.Energy(:);
193 output_5 = Idler.Energy(:);
194 output_6 = Pump.Energy(:);
195
196 end
```

M

SPLIT-STEP METHOD - PLOT WATERFALL

Matlab script used to plot the simulation data from the *Split-step Fourier method* seen in Appendix K.

```
plotlimit = floor(Param.t_steps/2-Param.t_steps/18):floor(Param.
    t_steps/2+Param.t_steps/18);
powerplot = 1:powersteps;
FontSize  = 20;
if( Param.t_FWHM*1e9 / sqrt(2) > 1)
    plotName  = [int2str(Param.t_FWHM*1e9 / sqrt(2)) 'ns'];
elseif( Param.t_FWHM*1e12 / sqrt(2) > 1)
    plotName  = [int2str(Param.t_FWHM*1e12 / sqrt(2)) 'ps'];
else
    plotName  = [num2str(Param.t_FWHM / sqrt(2),'%10.3e\n') 's'];
end

    figure(1)
    clearvars plot1data;
    plot1data(1:size(powerplot,2),1:Param.z_steps) = ...
        squeeze( Signal.I.depleted(powerplot,:,end/2) );
    w1 = waterfall(axis.z.*1e3,power(powerplot)./1000,plot1data
        .*1e-9);
    title('$I_{signal}^{+dep.}\,\,(\tau=0,z)$','Interpreter','
        latex','FontSize',FontSize+4);
    xlabel('$z~[mm]$','Interpreter','latex','FontSize',FontSize);
    ylabel('$P_p~[kW]$','Interpreter','latex','FontSize',FontSize
        );
    zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
        FontSize);
    set(w1,'edgecolor','k');
    %zlim([0 2e11])
    view([-140 50])
    export_fig(['CenterPropagation-' plotName '-Signal-Depleted'
        ], '-pdf', '-transparent');

    figure(2)
    clearvars plot2data;
    plot2data(1:size(powerplot,2),1:size(plotlimit,2)) = ...
        squeeze( Signal.I.depleted(powerplot,end,plotlimit) );
    w2 = waterfall(axis.t(plotlimit).*1e12,power(powerplot)
        ./1000,plot2data.*1e-9);
    title('$I_{signal}^{+dep.}\,\,(\tau,z=z_0)\,$','Interpreter',
        'latex','FontSize',FontSize+4);
```

```
32      xlabel('$\tau~[ps]$','Interpreter','latex','FontSize',
            FontSize);
33      ylabel('$P_p~[kW]$','Interpreter','latex','FontSize',FontSize
            );
34      zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
            FontSize);
35      set(w2,'edgecolor','k');
36      %zlim([0 2e11])
37      view([-140 50])
38      export_fig(['CrystalEnd_' plotName '_Signal_Depleted'], '-pdf
            ', '-transparent');
39
40      figure(3)
41      clearvars plot3data;
42      plot3data(1:size(powerplot,2),1:size(plotlimit,2)) = ...
43          squeeze( Pump.I.depleted(powerplot,end,plotlimit) );
44      w3 = waterfall(axis.t(plotlimit).*1e12,power(powerplot)
            ./1000,plot3data.*1e-9);
45      title('$I_{pump}^{+dep.}\,\,(\tau,z=z_0)\,$','Interpreter','
            latex','FontSize',FontSize+4);
46      xlabel('$\tau~[ps]$','Interpreter','latex','FontSize',
            FontSize);
47      ylabel('$P_p~[kW]$','Interpreter','latex','FontSize',FontSize
            );
48      zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
            FontSize);
49      set(w3,'edgecolor','k');
50      %zlim([0 2e11])
51      view([-140 50])
52      export_fig(['CrystalEnd_' plotName '_Pump_Depleted'], '-pdf',
             '-transparent');
53
54      figure(4)
55      clearvars plot4data;
56      plot4data(1:size(powerplot,2),1:Param.z_steps) = ...
57          squeeze( Signal.I.undepleted(powerplot,:,end/2) );
58      w4 = waterfall(axis.z.*1e3,power(powerplot)./1000,plot4data
            .*1e-9);
59      title('$I_{signal}^{-dep.}\,\,(\tau=0,z)$','Interpreter','
            latex','FontSize',FontSize+4);
60      xlabel('$z~[mm]$','Interpreter','latex','FontSize',FontSize);
61      ylabel('$P_p~[kW]$','Interpreter','latex','FontSize',FontSize
            );
62      zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
            FontSize);
63      set(w4,'edgecolor','k');
64      %zlim([0 2e11])
65      view([-140 50])
```

```
66      export_fig(['CenterPropagation_' plotName '_Signal_Undepleted
            '], '-pdf', '-transparent');
67
68      figure(5)
69      clearvars plot5data;
70      plot5data(1:size(powerplot,2),1:size(plotlimit,2)) = ...
71          squeeze( Signal.I.undepleted(powerplot,end,plotlimit) );
72      w5 = waterfall(axis.t(plotlimit).*1e12,power(powerplot)
            ./1000,plot5data.*1e-9);
73      title('$I_{signal}^{-dep.}\,\,(\tau,z=z_o)\,$','Interpreter',
            'latex','FontSize',FontSize+4);
74      xlabel('$\tau~[ps]$','Interpreter','latex','FontSize',
            FontSize);
75      ylabel('$P_p~[kW]$','Interpreter','latex','FontSize',FontSize
            );
76      zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
            FontSize);
77      set(w5,'edgecolor','k');
78      %zlim([0 2e11])
79      view([-140 50])
80      export_fig(['CrystalEnd_' plotName '_Signal_Undepleted'], '-
            pdf', '-transparent');
81
82      figure(6)
83      clearvars plot6data;
84      plot6data(1:size(powerplot,2),1:size(plotlimit,2)) = ...
85          squeeze( Pump.I.undepleted(powerplot,end,plotlimit) );
86      w6 = waterfall(axis.t(plotlimit).*1e12,power(powerplot)
            ./1000,plot6data.*1e-9);
87      title('$I_{pump}^{-dep.}\,\,(\tau,z=z_o)\,$','Interpreter','
            latex','FontSize',FontSize+4);
88      xlabel('$\tau~[ps]$','Interpreter','latex','FontSize',
            FontSize);
89      ylabel('$P_p~[kW]$','Interpreter','latex','FontSize',FontSize
            );
90      zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
            FontSize);
91      set(w6,'edgecolor','k');
92      %zlim([0 2e11])
93      view([-140 50])
94      export_fig(['CrystalEnd_' plotName '_Pump_Undepleted'], '-pdf
            ', '-transparent');
```

N

SPLIT-STEP METHOD - PLOT PROPAGATION

Matlab script used to plot the simulation data from the *Split-step Fourier method* seen in Appendix K.

```
%% Propagation of pump in depleted model

    plotPumpPower = powerplot(2);
    plotlimZ = floor(linspace(Param.z_steps/1.3,Param.z_steps
        ,8));
    figure(12);
    w12 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
        z_mm(plotPumpPower,plotlimZ),squeeze(Pump.I.depleted(
        plotPumpPower,plotlimZ,plotlimit)).*1e-9);
    title(['$I_{pump}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
        power(plotPumpPower)./1000) 'kW'],'Interpreter','
        latex','FontSize',FontSize+4);
    xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
        FontSize);
    ylabel('$z~[mm]$','Interpreter','latex','FontSize',
        FontSize);
    zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
        FontSize);
    set(w12,'edgecolor','k');
    %zlim([0 2e11])
    view([-140 50])
    export_fig(['Propagation--' plotName '--Pump-Depleted--
        PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
         '-pdf', '-transparent');

    plotPumpPower = powerplot(3);
    plotlimZ = floor(linspace(Param.z_steps/1.3,Param.z_steps
        ,8));
    figure(13);
    w13 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
        z_mm(plotPumpPower,plotlimZ),squeeze(Pump.I.depleted(
        plotPumpPower,plotlimZ,plotlimit)).*1e-9);
    title(['$I_{pump}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
        power(plotPumpPower)./1000) 'kW'],'Interpreter','
        latex','FontSize',FontSize+4);
    xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
        FontSize);
    ylabel('$z~[mm]$','Interpreter','latex','FontSize',
        FontSize);
```

```
23          zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
24          set(w13,'edgecolor','k');
25          %zlim([0 2e11])
26          view([-140 50])
27          export_fig(['Propagation_' plotName '_Pump_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
28
29          plotPumpPower = powerplot(4);
30          plotlimZ = floor(linspace(Param.z_steps/2,Param.z_steps
                ,8));
31          figure(14);
32          w14 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Pump.I.depleted(
                plotPumpPower,plotlimZ,plotlimit)).*1e-9);
33          title(['$I_{pump}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
34          xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
35          ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
36          zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
37          set(w14,'edgecolor','k');
38          %zlim([0 2e11])
39          view([-140 50])
40          export_fig(['Propagation_' plotName '_Pump_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
41
42          plotPumpPower = powerplot(5);
43          plotlimZ = floor(linspace(1,Param.z_steps,8));
44          figure(15);
45          w15 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Pump.I.depleted(
                plotPumpPower,plotlimZ,plotlimit)).*1e-9);
46          title(['$I_{pump}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
47          xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
48          ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
49          zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
50          set(w15,'edgecolor','k');
```

```
51          %zlim([0 2e11])
52          view([-140 50])
53          export_fig(['Propagation_' plotName '_Pump_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
54
55          plotPumpPower = powerplot(6);
56          plotlimZ = floor(linspace(1,Param.z_steps,8));
57          figure(16);
58          w16 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Pump.I.depleted(
                plotPumpPower,plotlimZ,plotlimit)).*1e-9);
59          title(['$I_{pump}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
60          xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
61          ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
62          zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
63          set(w16,'edgecolor','k');
64          %zlim([0 2e11])
65          view([-140 50])
66          export_fig(['Propagation_' plotName '_Pump_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
67
68          plotPumpPower = powerplot(7);
69          plotlimZ = floor(linspace(1,Param.z_steps,8));
70          figure(17);
71          w17 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Pump.I.depleted(
                plotPumpPower,plotlimZ,plotlimit)).*1e-9);
72          title(['$I_{pump}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
73          xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
74          ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
75          zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
76          set(w17,'edgecolor','k');
77          %zlim([0 2e11])
78          view([-140 50])
```

```
79          export_fig(['Propagation_' plotName '_Pump_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
80
81          plotPumpPower = powerplot(8);
82          plotlimZ = floor(linspace(1,Param.z_steps,8));
83          figure(18);
84          w18 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Pump.I.depleted(
                plotPumpPower,plotlimZ,plotlimit)).*1e-9);
85          title(['$I_{pump}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
86          xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
87          ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
88          zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
89          set(w18,'edgecolor','k');
90          %zlim([0 2e11])
91          view([-140 50])
92          export_fig(['Propagation_' plotName '_Pump_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
93
94      %% Propagation of signal in depleted model
95
96          plotPumpPower = powerplot(2);
97          plotlimZ = floor(linspace(Param.z_steps/1.3,Param.z_steps
                ,8));
98          figure(22);
99          w22 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Signal.I.
                depleted(plotPumpPower,plotlimZ,plotlimit)).*1e-9);
100         title(['$I_{Signal}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
101         xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
102         ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
103         zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
104         set(w22,'edgecolor','k');
105         %zlim([0 2e11])
106         view([-140 50])
```

```
107         export_fig(['Propagation--' plotName '--Signal--Depleted--
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '--pdf', '--transparent');
108
109         plotPumpPower = powerplot(3);
110         plotlimZ = floor(linspace(Param.z_steps/1.3,Param.z_steps
                ,8));
111         figure(23);
112         w23 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Signal.I.
                depleted(plotPumpPower,plotlimZ,plotlimit)).*1e-9);
113         title(['$I_{Signal}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
114         xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
115         ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
116         zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
117         set(w23,'edgecolor','k');
118         %zlim([0 2e11])
119         view([-140 50])
120         export_fig(['Propagation--' plotName '--Signal--Depleted--
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '--pdf', '--transparent');
121
122         plotPumpPower = powerplot(4);
123         plotlimZ = floor(linspace(Param.z_steps/2,Param.z_steps
                ,8));
124         figure(24);
125         w24 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Signal.I.
                depleted(plotPumpPower,plotlimZ,plotlimit)).*1e-9);
126         title(['$I_{Signal}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
127         xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
128         ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
129         zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
130         set(w24,'edgecolor','k');
131         %zlim([0 2e11])
132         view([-140 50])
```

```
133         export_fig(['Propagation_' plotName '_Signal_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
134
135         plotPumpPower = powerplot(5);
136         plotlimZ = floor(linspace(1,Param.z_steps,8));
137         figure(25);
138         w25 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Signal.I.
                depleted(plotPumpPower,plotlimZ,plotlimit)).*1e-9);
139         title(['$I_{Signal}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
140         xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
141         ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
142         zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
143         set(w25,'edgecolor','k');
144         %zlim([0 2e11])
145         view([-140 50])
146         export_fig(['Propagation_' plotName '_Signal_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
147
148         plotPumpPower = powerplot(6);
149         plotlimZ = floor(linspace(1,Param.z_steps,8));
150         figure(26);
151         w26 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Signal.I.
                depleted(plotPumpPower,plotlimZ,plotlimit)).*1e-9);
152         title(['$I_{Signal}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
153         xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
154         ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
155         zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
156         set(w26,'edgecolor','k');
157         %zlim([0 2e11])
158         view([-140 50])
159         export_fig(['Propagation_' plotName '_Signal_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
160
```

```
161         plotPumpPower = powerplot(7);
162         plotlimZ = floor(linspace(1,Param.z_steps,8));
163         figure(27);
164         w27 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Signal.I.
                depleted(plotPumpPower,plotlimZ,plotlimit)).*1e-9);
165         title(['$I_{Signal}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
166         xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
167         ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
168         zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
169         set(w27,'edgecolor','k');
170         %zlim([0 2e11])
171         view([-140 50])
172         export_fig(['Propagation_' plotName '_Signal_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
173
174         plotPumpPower = powerplot(8);
175         plotlimZ = floor(linspace(1,Param.z_steps,8));
176         figure(28);
177         w28 = waterfall(Axis.t_ns(plotPumpPower,plotlimit),Axis.
                z_mm(plotPumpPower,plotlimZ),squeeze(Signal.I.
                depleted(plotPumpPower,plotlimZ,plotlimit)).*1e-9);
178         title(['$I_{Signal}^{+dep.}\,\,(\tau,z)\,$ @ ' int2str(
                power(plotPumpPower)./1000) 'kW'],'Interpreter','
                latex','FontSize',FontSize+4);
179         xlabel('$\tau~[ns]$','Interpreter','latex','FontSize',
                FontSize);
180         ylabel('$z~[mm]$','Interpreter','latex','FontSize',
                FontSize);
181         zlabel('$I~[GW/m^2]$','Interpreter','latex','FontSize',
                FontSize);
182         set(w28,'edgecolor','k');
183         %zlim([0 2e11])
184         view([-140 50])
185         export_fig(['Propagation_' plotName '_Signal_Depleted_
                PumpPower' int2str(power(plotPumpPower)./1000) 'kW'],
                 '-pdf', '-transparent');
186
187
188 %% compare signal energy
189 ColorSet=varycolor(size(powerplot,2));
190
```

```
figure(1003)
hold all
for nn=1:size(powerplot,2)
    plot(Axis.z_mm(nn,:),Signal.Energy.depleted(nn,:)./Pump.
        Energy.depleted(nn,1),'LineWidth',1.5,'Color',ColorSet(nn
        ,:),'DisplayName','kW');
    legendInfo{nn} = [int2str(power(powerplot(nn))./1000) 'kW'];
end
hold off;
    l1003 = legend(legendInfo,...%'First','Second','Third','4th
        ','5th','6th','7th','8th','9th','10th',...
        'Orientation','vertical','Location','EastOutside');
    set(l1003,'Interpreter','latex','color','none');
    PlotSetAxisLabels('z [mm]','$\eta~[a.u.]$',18);
    PlotSetFigureDimensions(1,5,22,10,0,0,0,0);
    PlotSetAxisProperties(5,3,20);
    set(gca, 'Units', 'normalized','Position', [0.15 0.22 0.62
        0.55]);
    set(gca, 'Color', 'none'); % Sets axes background
    export_fig(['Propagation-' plotName '-SignalEnergy-Depleted-
        PumpPower'], '-pdf', '-transparent');

figure(1004)
hold on;
for nn=1:size(powerplot,2)
    plot(Axis.z_mm,Signal.Energy.undepleted(nn,:)./Pump.Energy.
        undepleted(nn,1),'LineWidth',1.5,'Color',ColorSet(nn,:));
end
hold off;
set(gca,'YScale','log');
```

0 MEASUREMENT SETUP

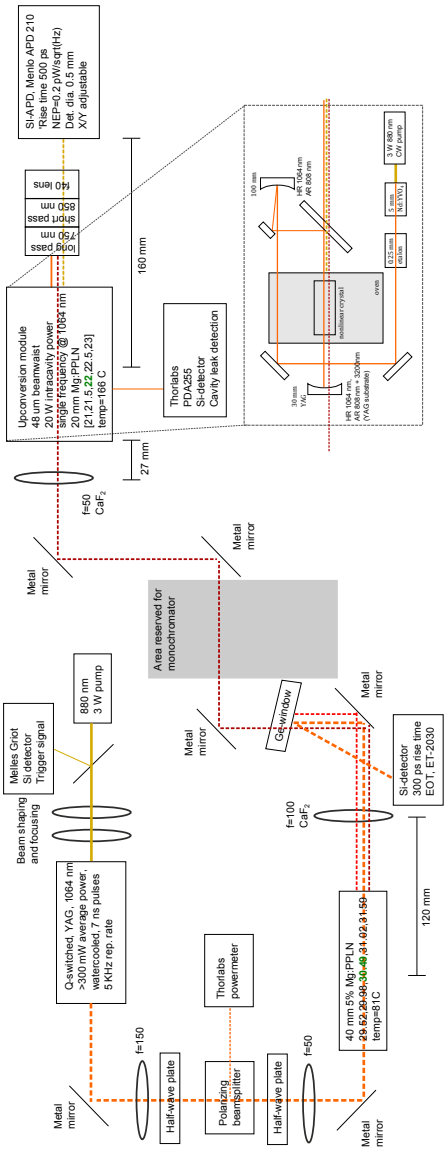


Figure 42: Detailed sketch of measurement setup. Consist of two parts, firstly a DFG process creating mid infrared pulses and secondly a SFG process creating visible pulses at $\approx 800\,\mathrm{nm}$.

BIBLIOGRAPHY

[1] Gunnar Arisholm. Quantum noise initiation and macroscopic fluctuations in optical parametric oscillators. *Journal of the Optical Society of America B*, 16(1):117–127, 1999.

[2] L. Arizmendi. Photonic applications of lithium niobate crystals. *Physica Status Solidi (a)*, 201(2):253–283, January 2004. ISSN 0031-8965.

[3] G. D. Boyd. Parametric Interaction of Focused Gaussian Light Beams. *Journal of Applied Physics*, 39(8):3597, 1968. ISSN 00218979.

[4] R W Boyd and G L Fischer. Nonlinear Optical Materials. In K H Jürgen Buschow, Robert W Cahn, Merton C Flemings, Bernhard Ilschner, Edward J Kramer, Subhash Mahajan, and Patrick Veyssière, editors, *Encyclopedia of Materials: Science and Technology (Second Edition)*, pages 6237–6244. Elsevier, Oxford, second edi edition, 2001. ISBN 978-0-08-043152-9.

[5] Robert W. Boyd. *Nonlinear optics*. 3rd edition, 2008.

[6] D A Bryan, Robert Gerson, and H E Tomaschke. Increased optical damage resistance in lithium niobate Increased optical damage resistance in lithium niobate. *Applied Physics Letters*, 847 (1984):223–226, 2007.

[7] Preben Buchhave and Peter Tidemand-Lichtenberg. Generation of higher order Gauss-Laguerre modes in single-pass 2nd harmonic generation. *Optics express*, 16(22), October 2008. ISSN 1094-4087.

[8] Paul Campagnola. Second Harmonic Generation Imaging Microscopy: Applications to Diseases Diagnostics. *Analytical Chemistry*, 83:3224–3231, 2011.

[9] Lionel Carrion and Claude Bernard Lyon I. Development of a simple model for optical parametric generation. *Journal of the Optical Society of America B*, 17(1):78–83, 2000.

[10] Stefano Cattaneo and Martti Kauranen. Application of second-harmonic generation to retardation measurements. *Journal of the Optical Society of America B*, 20(3):520, 2003. ISSN 0740-3224.

[11] DTU Computing Center. HPC Competence Center. URL http://www.cc.dtu.dk/.

[12] D.S. Chemla. *Nonlinear Optical Properties of Organic Molecules and Crystals, Volume 1*. Elsevier Science, 2012.

[13] Covision. Covesion Guide to PPLN - Second Order Nonlinearity. URL http://www.covesion.com/support/material-properties-of-lithium-niobate.html.

[14] V.G. Dmitriev. *Handbook of nonlinear optical crystals*. Springer, 1999. ISBN 9783540653943.

[15] Okan K. Ersoy. *Diffraction, Fourier Optics and Imaging*. John Wiley & Sons, Inc., 1st edition, 2006.

[16] Matteo Frigo and Steven G Johnson. Matteo Frigo Steven G. Johnson. (November), 2012.

[17] O. Gayer, Z. Sacks, E. Galun, and A. Arie. Temperature and wavelength dependent refractive index equations for MgO-doped congruent and stoichiometric LiNbO3. *Applied Physics B*, 91(2):343–348, April 2008. ISSN 0946-2171. doi: 10.1007/s00340-008-2998-2.

[18] David .J Griffiths. *Introduction to Electrodynamics*. Prentice Hall, 3rd edition, 1999.

[19] ISO 11146-1:2005. Lasers and laser-related equipment. Test methods for laser beam widths, divergence angles and propagation ratios - Part 1: Stigmatic and simple astigmatic beams. Technical report, International Organization for Standardization, 2005.

[20] Peipei Jiang, Tao Chen, Dingzhong Yang, Bo Wu, Shuangshuang Cai, and Yonghang Shen. A fiber laser pumped dual-wavelength mid-infrared optical parametric oscillator based on aperiodically poled magnesium oxide doped lithium niobate. *Laser Physics Letters*, 10(11):115405, November 2013. ISSN 1612-2011.

[21] Helge S. Kragh. *Dirac: A Scientific Biography*, volume 59. Cambridge University Press, 1990.

[22] Mathworks. Techniques for Improving Performance. URL http://www.mathworks.se/help/matlab/matlab_prog/techniques-for-improving-performance.html.

[23] Mathworks. Cell vs. Struct Arrays; Matlab R2014a documentation, 2014. URL http://www.mathworks.se/help/matlab/matlab_prog/cell-vs-struct-arrays.html.

[24] James Clerk Maxwell. *A Treatise on Electricity and Magnetism*. 1873.

[25] Cleve Moler and Steve Eddins. Faster Finite Fourier Transforms MATLAB. URL http://www.mathworks.se/company/newsletters/articles/faster-finite-fourier-transforms-matlab.html.

[26] Kosuke Murate, Yuusuke Taira, Saroj R. Tripathi, Shin'ichiro Hayashi, Koji Nawata, Hiroaki Minamide, and Kodo Kawase. Frequency tunable, high dynamic range THz spectrometer using parametric processes in Lithium Niobate crystal. *Cleo: 2014*, 2014.

[27] L.E. Myers and W.R. Bosenberg. Periodically poled lithium niobate and quasi-phase-matched optical parametric oscillators. *IEEE Journal of Quantum Electronics*, 33(10):1663–1672, 1997. ISSN 00189197.

[28] Rüdiger Paschotta. Phase Matching. URL http://www.rp-photonics.com/phase_matching.html.

[29] Frank L. Pedrotti. *Introduction to optics*. Pearson Prentice Hall, 2007. ISBN 9780131971332.

[30] Jiahui Peng. Developments of mid-infrared optical parametric oscillators for spectroscopic sensing: a review. *Optical Engineering*, 53(6):061613, February 2014. ISSN 0091-3286.

[31] Karsten Rottwitt and Peter Tidemand-Lichtenberg. Nonlinear Optics. 2013.

[32] Bahaa E. A. Saleh and Malvin Carl Teich. *Fundamentals of Photonics*. Wiley, 2nd edition, 2007.

[33] Lester W. Schmerr and Sung-Jin. Song. *Ultrasonic nondestructive evaluation systems : models and measurements*. Springer, 2007.

[34] A V Smith, Russell J Gehr, and Mark S Bowers. Numerical models of broad-bandwidth nanosecond optical parametric oscillators. *Optical Society of America*, 16(4), 1999.

[35] WA Strauss. *Partial differential equations: An introduction*. Wiley, 2nd edition, 2007.

[36] Aasmund Sv Sudbo. Why Are Accurate Computations of Mode Fields in Rectangular Dielectric Waveguides Difficult? *Journal of Lightwave Technology*, 10(4), 1992.

[37] Xuesong Wang, Jianbing Li, Tao Wang, and Shunping Xiao. Validity criterion for the Born approximation convergence in microscopy imaging: comment. *Journal of the Optical Society of America. A, Optics, image science, and vision*, 28(4):662–4; discussion 665–6, April 2011. ISSN 1520-8532.

DECLARATION

This thesis is a presentation of original work by undersigned. Wherever contributions of others are involved, either research results, theoretical framework, illustrations or other aspects, every effort is made to indicate this clearly with due reference to the literature and acknowledgement of collaborative researchers.

The work presented was done under the guidance of Associate Professor Peter Tidemand-Lichtenberg and Ph.D. Student Lasse Høgstedt at the Technical University of Denmark, Department of Photonics Engineering.

Risø, July 2014

Anders C. Bilfeldt

Technical University
of Denmark

Anders C. Bilfeldt: *Numerical modeling of optical parametric frequency conversion,* for temporally confined pulses and spatially confined modes, © July 2014

This thesis and all related material is avaliable for download at
http://www.anders.bilfeldt.dk/thesis/
alternatively use the Q-code above