Solver Guide for the MATLAB solid-core-fiber pulse propagation

Yi-Hao Chen
Applied and Engineering Physics, Cornell University

September 28, 2025


CORNELL UNIVERSITY
FOUNDED A.D. 1865

Contents

Chapter 1

Overview

1.1 Mathematical background

This package aims to solve the multimode unidirectional pulse propagation equation (MM-UPPE) in a solid-core fiber:

$$\begin{aligned}\partial_z A_p(z,\Omega) = {} & i\left[\beta_p(\omega) - \left(\beta_{(0)} + \beta_{(1)}\Omega\right)\right] A_p(z,\Omega) + g_p(z,\Omega)A_p(z,\Omega) \\ & + i\sum_{\ell} Q_{p\ell} A_\ell(z,\Omega) \\ & + \frac{i\omega}{4}\epsilon_0^2 n_{\text{eff}}^2 c n_2 \sum_{\ell mn}\Bigg\{ (1-f_R)\, Q^K_{p\ell mn}\mathfrak{F}\left[A_\ell A_m A_n^* + \tilde{\Gamma}^K_{p\ell mn}\right] \\ & \qquad + f_R\Bigg\{ f_a Q^{R_a}_{p\ell mn}\mathfrak{F}\left[A_\ell\left[h_a * \left(A_m A_n^* + \tilde{\Gamma}^{R_a}_{p\ell mn}\right)\right]\right] + \\ & \qquad\qquad f_b Q^{R_b}_{p\ell mn}\mathfrak{F}\left[A_\ell\left[h_b * \left(A_m A_n^* + \tilde{\Gamma}^{R_b}_{p\ell mn}\right)\right]\right]\Bigg\}\Bigg\}, \quad (1.1)\end{aligned}$$

which includes dispersion, as well as instantaneous electronic and delayed Raman nonlinearities. $A_p(z,t)$ is the electric field ($\sqrt{\text{W}}$) of mode p, whose Fourier transform is $A_p(z,\Omega) = \mathfrak{F}[A_p(z,T)]$. The Fourier transform is applied with respect to angular frequency $\Omega = \omega - \omega_0$, where ω_0 is the center angular frequency of the numerical frequency window required to cover the investigated physical phenomena. β_p is the propagation constant of the mode p. $\beta_{(0)}$ and $\beta_{(1)}$ are to reduce the propagating global-phase increment to facilitate simulations, $\beta_{(1)}$ is the inverse group velocity of the moving frame, which introduces the delayed time $T = t - \beta_{(1)}z$. $g_p(z,\Omega)$ is the gain (or loss), n_2 is the nonlinear refractive index (m^2/W; refractive index change from nonlinearity $\triangle n = n_2 I$ where I is the light intensity), c is the speed of light; f_R is the Raman fraction representing the contribution of the Raman response of all nonlinearities where f_a and f_b are Raman fractions of the total Raman response for isotropic and anisotropic Raman responses, respectively ($f_a + f_b = 1$); h_a and h_b are isotropic and anisotropic Raman response functions. $\tilde{\Gamma}^{R_a}_{p\ell mn}$, $\tilde{\Gamma}^{R_b}_{p\ell mn}$, and $\tilde{\Gamma}^{K}_{p\ell mn}$ are to model the spontaneous Raman and noise-seeded Kerr contributions

[1], following

$$\tilde{\Gamma}^{K}_{p\ell mn}(T) = A^{\text{noise}}_{\ell} A_m A^*_n + A_\ell A^{\text{noise}}_m A^*_n + A_\ell A_m \left(A^{\text{noise}}_n\right)^* \tag{1.2a}$$

$$\tilde{\Gamma}^{R_r}_{p\ell mn}(T) = A_m \left(A^{\text{noise}}_n\right)^* + A^{\text{noise}}_m A^*_n. \tag{1.2b}$$

The Raman term follows

$$h_r(t) * X(t) = \mathfrak{F}^{-1}\left[\frac{1}{C_{\mathfrak{F}}}\mathfrak{F}\left[h_r(t)\right]\mathfrak{F}\left[X(t)\right]\right]. \tag{1.3}$$

p, ℓ, m, and n the eigenmode indices. Q^K, Q^{R_a}, and Q^{R_b} are overlap integrals:

$$Q^K_{p\ell mn} = \frac{2}{3}Q^{R_a}_{p\ell mn} + \frac{1}{3}Q^{k}_{p\ell mn} \qquad , Q^k_{p\ell mn} = \frac{\int \left(\vec{F}^*_p \cdot \vec{F}^*_n\right)\left(\vec{F}_\ell \cdot \vec{F}_m\right) \mathrm{d}x\,\mathrm{d}y}{N_p N_\ell N_m N_n} \tag{1.4a}$$

$$Q^{R_a}_{p\ell mn} = \frac{\int \left(\vec{F}^*_p \cdot \vec{F}_\ell\right)\left(\vec{F}_m \cdot \vec{F}^*_n\right) \mathrm{d}x\,\mathrm{d}y}{N_p N_\ell N_m N_n} \tag{1.4b}$$

$$Q^{R_b}_{p\ell mn} = \frac{1}{2}\left[\frac{\int \left(\vec{F}^*_p \cdot \vec{F}_m\right)\left(\vec{F}_\ell \cdot \vec{F}^*_n\right) \mathrm{d}x\,\mathrm{d}y}{N_p N_\ell N_m N_n} + Q^k_{p\ell mn}\right] = \frac{1}{2}\left(Q^{r_b}_{p\ell mn} + Q^k_{p\ell mn}\right), \tag{1.4c}$$

where $\vec{F}_p$ is the p-th spatial eigenmode, n_{eff} and $n_{i,\text{eff}}$ in each N_i ($i \in \{p, \ell, m, n\}$) is often taken as the refractive index of silica such that

$$\frac{\epsilon_0^2 n_{\text{eff}}^2 c^2}{N_p N_\ell N_m N_n} = 4. \tag{1.5}$$

With Eq. (1.5), Eq. (1.1) is simplified to

$$\begin{aligned}
\partial_z A_p(z,\Omega) = {} & i\left[\beta_p(\omega) - \left(\beta_{(0)} + \beta_{(1)}\Omega\right)\right] A_p(z,\Omega) + g_p(z,\Omega) A_p(z,\Omega) \\
& + i\sum_{\ell} Q_{p\ell} A_\ell(z,\Omega) \\
& + \frac{i\omega n_2}{c}\sum_{\ell mn}\Bigg\{ (1-f_R)\, S^K_{p\ell mn}\mathfrak{F}\left[A_\ell A_m A^*_n + \tilde{\Gamma}^K_{p\ell mn}\right] + \\
& \qquad f_R\Bigg\{ f_a S^{R_a}_{p\ell mn}\mathfrak{F}\left[A_\ell\left[h_a * \left(A_m A^*_n + \tilde{\Gamma}^{R_a}_{p\ell mn}\right)\right]\right] + \\
& \qquad f_b S^{R_b}_{p\ell mn}\mathfrak{F}\left[A_\ell\left[h_b * \left(A_m A^*_n + \tilde{\Gamma}^{R_b}_{p\ell mn}\right)\right]\right]\Bigg\}\Bigg\},
\end{aligned} \tag{1.6}$$

with modified overlap integrals:

$$S^K_{p\ell mn} = \frac{2}{3}S^{R_a}_{p\ell mn} + \frac{1}{3}S^k_{p\ell mn} \qquad , S^k_{p\ell mn} = \int \left(\vec{F}^*_p \cdot \vec{F}^*_n\right)\left(\vec{F}_\ell \cdot \vec{F}_m\right) \mathrm{d}x\,\mathrm{d}y \tag{1.7a}$$

$$S^{R_a}_{p\ell mn} = \int \left(\vec{F}^*_p \cdot \vec{F}_\ell\right)\left(\vec{F}_m \cdot \vec{F}^*_n\right) \mathrm{d}x\,\mathrm{d}y \tag{1.7b}$$

$$S^{R_b}_{p\ell mn} = \frac{1}{2}\left[\int \left(\vec{F}^*_p \cdot \vec{F}_m\right)\left(\vec{F}_\ell \cdot \vec{F}^*_n\right) \mathrm{d}x\,\mathrm{d}y + S^k_{p\ell mn}\right] = \frac{1}{2}\left(S^{r_b}_{p\ell mn} + S^k_{p\ell mn}\right). \tag{1.7c}$$

In silica, it is sometimes overkill to run with UPPE due to mostly narrowband scenarios. In this case, $\beta(\omega)$ is obtained from its Taylor-series coefficients $\beta_{(0)} + \beta_{(1)}\Omega + \frac{\beta_2}{2}\Omega^2 + \frac{\beta_3}{3!}\Omega^3 + \cdots$, which is, in fact, equivalent to a more-commonly-used GMMNLSE.

It is worth noting that the field A_p is defined as

$$\begin{aligned}\vec{\mathbb{E}}(\vec{x},t) &= \frac{1}{2}\left[\vec{\mathcal{E}}(\vec{x},t) + \text{c.c.}\right] \quad , \vec{\mathcal{E}} \text{ is the analytic signal of } \vec{\mathbb{E}} \\ &= \sum_p \int \mathrm{d}\omega \frac{1}{2}\left\{\frac{\vec{F}_p(x,y,\omega)}{N_p(\omega)} A_p(z,\omega)\mathrm{e}^{i\left[\beta_p(\omega)z - \omega t\right]} + \text{c.c.}\right\}.\end{aligned} \tag{1.8}$$

This makes MATLAB "ifft" become the Fourier transform and "fft" become the *inverse* Fourier transform. The convolution theorem also becomes different with different conventions. In our package, we follow this convention [Eq. (1.8)]. For example, to see the spectrum, please use

```
c = 299792.458; % nm/ps
wavelength = c./f; % nm
Nt = size(field,1);
dt = t(2)-t(1); % ps
factor_correct_unit = (Nt*dt)^2/1e3; % to make the spectrum of the correct unit
    "nJ/THz"
                                     % "/1e3" is to make pJ into nJ
spectrum = abs(fftshift(ifft(field),1)).^2*factor_correct_unit; % in frequency
    domain
```

Use "fftshift" to shift the spectrum from small frequency to large frequency. Note that it is not "ifftshift". They differ when the number of points is odd. To understand this, think about what the first data point is in "ifft(field):" it is the zero-frequency component, so we need to use "fftshift" with the frequency defining as

```
f = f0+(-Nt/2:Nt/2-1)'/(Nt*dt); % THz
```

Check the supplements of our femtosecond-LWIR generation [2] and [3] for details. We forgot to add this information in our JOSAB's multimode-gain paper [4].

1.2 High-level understanding of this package

This package is designed for both single-spatial(transverse) mode or multi-spatial modes. Not only scalar but also polarized fields can be simulated, as well as Raman scattering and the gain. The package exhibits an adaptive control of the step size, except for situations with amplified stimulated emission (ASE). In addition to CPU, highly parallelized cuda computation with a Nvidia GPU is implemented, which is strongly recommended for running with multimodes. In

single-mode simulations, for sampling numbers less than approximately 2^{25}, they can still run faster with CPU than with GPU. The package uses "RK4IP" (Runge-Kutta in the interaction picture) for single mode [5, 6] and "MPA" (Massively Parallel Algorithm) for multimode [7].

The fastest way to learn how to use this code is to start with the example codes in the package.

Chapter 2

Before I go deeply into details

2.1 Introduction

This document describes how to use the GMMNLSE_propagate() MATLAB function.

Below is how to call this function in general.

```
prop_output = GMMNLSE_propagate(fiber, ...
                                initial_condition, ...
                                sim[, ...
                                gain_rate_eqn])
```

prop_output

It contains the information of the output field after propagating through the fiber, such as the field amplitudes and the positions of each saved field, etc.

fiber

It contains the information of the fiber, such as β_2, S^R, and the MFD, etc.

initial_condition

It contains the information of the input.
Typically it's the input field amplitude. If you run it not only with the rate-equation gain model but considering ASE, it also contains the forward ASE at the input and backward ASE at the output.

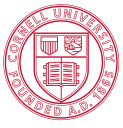
Figure 2.1: Initial conditions

sim

It contains a multitude of information about the simulation, such as the algorithm to use, if running with an adaptive-step method, and the center wavelength, etc.

[**gain_rate_eqn**]

This is required only for the rate-equation gain model.
It contains the information required for the rate-equation gain model, such as the pump power, the doped ion density, etc.



Chapter 3

Input arguments

Below, I use N_t as the number of time/frequency sampling points, N_m as the number of modes, and N_{sm} as the number of spatial modes. If there is no polarized mode, $N_m = N_{sm}$; otherwise, $N_m = 2N_{sm}$.

I recommend to use the information below as a reference guide if you're confused. Start with an example script is always better than reading this first.

Some parameters are required only when you enable some settings. Below I labeled in blue the parameters required all the time.

3.1 fiber

betas

Typically, this is the variable that saves the $\beta_0, \beta_1, \beta_2 \ldots$. It has the unit of ps^n/m.

It's a column vector of

$$\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \end{bmatrix} \tag{3.1}$$

if this is a single-mode simulation.

For multimode, it becomes

$$\begin{bmatrix} \beta_{0|1} & \beta_{0|2} & \cdots \\ \beta_{1|1} & \beta_{1|2} & \cdots \\ \beta_{2|1} & \beta_{2|2} & \cdots \\ \beta_{3|1} & \beta_{3|2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}, \tag{3.2}$$

where $\beta_{i|m}$ is the β_i for the m-th mode.

Besides the narrowband Taylor-series expansion of $\beta(\omega)$, I've also implemented the broadband version whose **betas** are column vectors of $\beta(\omega)$, that is, it becomes

$$\begin{bmatrix} \beta_{\cdot|1} & \beta_{\cdot|2} & \cdots \end{bmatrix}, \tag{3.3}$$

where each $\beta_{\cdot|m}$ is a column vector of the propagation constant of the m-th mode. It's a function of frequency, with an order from small to large. If the simulation is run with N_t time/frequency points, this $\beta_{\cdot|m}$ should have the length of N_t as well.

n2

It's the nonlinear coefficient of the fiber. By default, GMMNLSE_propagate() uses $2.3 \times 10^{-20}\,\mathrm{m}^2/\mathrm{W}$ assuming we use a silica fiber around 1 µm if fiber.n2 is left empty.

SR

It's the overlap integral S^R in scalar GMMNLSE. It's loaded in a dimension of N_{sm}^4.

For scalar GMMNLSE, S^K is S^R if the field is linearly polarized or $\frac{2}{3}S^R$ is it's circularly polarized. The unit is m^{-2}.

For polarized fields, their S^R and S^K can be calculated from the scalar S^R. This will be done by GMMNLSE_propagate() automatically if "sim.scalar=false." For details, check Chap.5.

L0

This is the fiber length. The unit is meter.

material

This specifies the material of the fiber. It's 'silica' by default. It's used only for specifying which Raman model to use. It's either 'silica', 'chalcogenide', or 'ZBLAN'.

If we use the Gaussian-gain model, the following parameters are required.

dB_gain

The small-signal gain amplification of the pulse energy in dB. This is used to calculate the gain_coeff (default to 30).

gain_coeff

This is the small-signal gain coefficient, the g in

$$A(z) = \mathrm{e}^{gz/2}A(0). \tag{3.4}$$

It's a scalar with the unit of m^{-1}.

gain_fwhm

This is the gain bandwidth. A typical number for Yb-doped gain fibers is 40 nm. This parameter has the unit of meter.

gain_doped_diameter

The diameter of the doped core to compute the overlap integral between mode fields and the doped core and accurately find their gain.

saturation_intensity

This is for multimode simulations. It has the unit of J/m^2.

saturation_energy

This is for single-mode simulations. It has the unit of nJ.

3.2 initial_condition

dt

This is the time sampling step $\triangle t$ with a unit of ps.

fields

This is the input field's temporal amplitude ($A(t)$ with the unit $\sqrt{\text{W}}$). Its size is $N_t \times N_m$.

If its size is $N_t \times N_m \times N_z$, only the last N_z is taken as the input field.

If attempting to run only for the ASE powers, users can set this "fields" variable to all zeros. Please see example "ASE_evolutions.m" in "Examples/Gain-rate-equation model" for details.

The two parameters above are required all the time.

If we run the simulation with ASE (and also of course with the rate-equantion gain model), two extra parameters are required. Typically they are both all-zero $N_t \times N_m$ column vectors.

Power.ASE.forward

This is the forward ASE spectral power ($P_{\text{ASE}}(\nu)$ with the unit W/THz) at the input ($z = 0$). Its array size is the same "fields" above.

Power.ASE.backward

This is the backward ASE spectral power ($P_{\text{ASE}}(\nu)$ with the unit W/THz) at the output ($z = L_0$) because backward ASE starts from the output end of the fiber.

3.3 sim

Below are the most basic parameters for a simulation.

betas

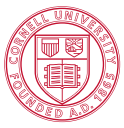
In UPPE, we not only create a moving frame that follows the pulse with the inverse velocity $\beta_{(1)}$ but extract out the reference propagation constant $\beta_{(0)}$. The benefit of extracting $\beta_{(0)}$ is that it reduces the rate of global phase increment such that the simulation can run with a larger step. This is similar to the limitation of multimode simulations that different spatial modes have different propagation constants that generate beating. To resolve the multimode beating, the size of the z-step cannot be too large.

This "betas" is a 2×1 column vector.

$$\begin{bmatrix} \beta_{(0)} \\ \beta_{(1)} \end{bmatrix} \tag{3.5}$$

By default, under the narrowband case where fiber.betas is a column vector of the Taylor series coefficient of $\beta(\omega)$, GMMNLSE_propagate() uses the 1st mode as the reference, that is,

$$\begin{bmatrix} \beta_{(0)} \\ \beta_{(1)} \end{bmatrix} = \begin{bmatrix} \beta_{0\vert1} \\ \beta_{1\vert1} \end{bmatrix}. \tag{3.6}$$



In the broadband case where $\beta_{\cdot|m}$ are functions of frequency [Eq. (3.3)], the code will compute the $\beta_{(0)}$ and $\beta_{(1)}$ at the center frequency of the 1st mode of input signal pulse, and use it as the "betas" here.

f0

The center frequency (THz). It's a scalar.

dz

The z-step size (m). This is required only for non-adaptive-step method. For an adaptive-step method, this parameter varies during computation; setting it here is meaningless.

This may need to be 1–50 µm to account for intermodal beating, even if the nonlinear length is large.

save_period

The length between saved fields (m). If it's zero, it's equaivalent to save_period=fiber.L0 that saves only the input and output fields.

Be aware that this number needs to be a divisor of the fiber length, fiber.L0; otherwise, GMMNLSE_propagate() will throw an error. For an adaptive-step method, I have the maximum step size set as the $\frac{1}{10}$ of the save_period and the position of the saved fields will be chosen as the one that first passes through each saved point.

3.3.1 MPA

Here are the parameters if the simulation uses MPA step method [7]. All parameters are contained within a "sim.MPA" structure.

MPA.M

This is the parallel extent for MPA. 1 is no parallelization. 5–20 is recommended; there are strongly diminishing returns after 5–10. 10 is recommended.

MPA.n_tot_max

The maximum number of iterations for MPA. This doesn't really matter because if the step size is too large, the algorithm will diverge after a few iterations. 20 is a typical number for this.

MPA.n_tot_min

The minimum number of iterations for MPA. 2 is recommended.

MPA.tol

The tolerance of convergence for MPA, which is related to the values of the average NRMSE between consecutive itertaions in MPA at which the step is considered converged. 10^{-6} is recommended.

3.3.2 Random linear mode coupling

Here are the parameters if the simulation uses random mode coupling. All parameters are contained within a "sim.rmc" structure. To run with random mode coupling, random-coupling matrices need to be created beforehand by calling

```
save_points=int32(fiber.L0/sim.dz);
sim.rmc.matrices = create_rmc_matrices(fiber,sim,num_modes,save_points);
```

"sim.dz" is required since there is no adaptive step-size control for computations with random mode coupling.

rmc.model

false (0)	includes random mode coupling
true (1)	don't include random mode coupling

rmc.varn

The variations of refractive index of the fiber. It is used to control the strength of random mode coupling.

rmc.stdQ_polarizedmode

Similar to "rmc.varn", it is used control the strength of random polarization-mode coupling.

rmc.lambda0

It lets the code know the wavelength of the eigenmode fields to load for random mode coupling to compute the coupling strengths.

rmc.downsampling_factor

To compute the couplng strengths among spatial modes, loading mode profiles is required. This downsampling factor determines the downsampled factor after loading to improve the performance of the random-mode-coupling matrices. Since it won't affect the latter nonlinear pulse propagation, I typically just set it to 1.

3.3.3 Polarization modes

Here are the parameters if the simulation includes polarization modes.

scalar

false (0)	includes polarization-mode coupling
true (1)	don't include polarization-mode coupling

If the simulation is solved with "sim.scalar=true," the input field takes only the scalar fields, e.g.,

$$\begin{bmatrix}\text{mode 1} & \text{mode 2} & \text{mode 3} & \dots\end{bmatrix}.$$

Otherwise, the input field of each polarized mode needs to be specified in the order of

$$\begin{bmatrix}\text{mode } 1_+ & \text{mode } 1_- & \text{mode } 2_+ & \text{mode } 2_- & \dots\end{bmatrix},$$

where (+,-) can be (x,y), (right-handed circular, left-handed circular), or any orthogonally polarized modes.

Based on whether to include polarization-mode coupling, S^R and S^K are automatically calculated to its polarized version by GMMNLSE_propagate().

ellipticity

The ellipticity of the polarization modes. Please refer to "Nonlinear Fiber Optics, Eq. (6.1.18) Agrawal" for the equations.

0	linear polarization	(+,-)=(x,y)
1	circular polarization	(+,-)=(right,left)

3.3.4 Adaptive-step method

Here are the parameters if the simulation uses adaptive-step method. All parameters are contained within a "sim.adaptive_dz" structure. The user doesn't need to specify whether to use adaptive-step method or not; the code determines itself. With the adaptive-step method, the initial step size is set to a small 10^{-6} m.

adaptive_dz.threshold
The threshold of the adaptive-step method. It controls the accuracy of the simulation and determines whether to increase or decrease the step size. I typically use 10^{-6}.

adaptive_dz.max_dz
The maximum z-step size (m) of the adaptive-step method. It's 1/10 the save_period by default.

3.3.5 Narrowband transformation (scaled Fourier transform)

Here are the parameters if the simulation applies narrowband transformation, based on the scaled Fourier transform [8]. All parameters are contained within a "sim.cs" structure.

cs.model
The model of narrowband transformation to use. Narrowband transformation is correct only in clean phase or amplitude information, so picking which model to use is important. Physics with strong correlation with both phase and amplitude cannot be fully captured, so users need to decide which dominates the process. Narrowband transformation doesn't apply to physical phenomena that require both to be correct. For example, in CPA, amplification with rare-earth-doped fiber is not correlated phase, so we can apply narrowband transformation. However, for four-wave-mixing processes whose gain is correlated to phase mismatch, we cannot apply narrowband transformation. Raman amplification "far away from Raman gain suppression point" is dominated only only the (amplitude) Raman gain, so it's good for narrowband transformation too.

For details, see Sec. 3D in the supplement of [8].

1 targeting correct nonlinear phase modulation (for nonlinear phase accumulation in CPA, Raman gain suppression, etc.)
2 targeting correct nonlinear amplitude modulation (for Raman gain, etc.)

cs.cs
The scaled factor in scaled Fourier transform/narrowband transformation. It is 1 by default, which means no application of the narrowband transformation.

3.3.6 Algorithm to use

gpu_yes
true (1) use GPU
false (0) don't use GPU

include_Raman

false(0) ignore Raman effect
true(1) Raman model including the anisotropic contribution ("Ch. 2.3, p. 43" and "Ch. 8.5, p. 340," Nonlinear Fiber Optics (5th), Agrawal)

Typically, only isotropic Raman is considered, which is based on a single vibrational Raman mode of molecules (Ch. 2.3, p.42, Nonlinear Fiber Optics (5th), Agrawal). Here, we include the anisotropic part if there is an existing model, such as the one in silica ("Ch. 2.3, p. 43" and "Ch. 8.5, p. 340," Nonlinear Fiber Optics (5th), Agrawal).

For more details about anisotropic Raman, please read "Raman response function for silica fibers," by Q. Lin and Govind P. Agrawal (2006). Besides silica, chalcogenide and ZBLAN are also included.

gain_model

Except for the rate-equation gain model, all the other gain models use a Gaussian gain; thus, the gain_coeff, gain_fwhm, and gain saturation intensity or energy need to be specified in "fiber."

0 no gain
1 Gaussian gain
2 rate-equation gain: see Chap.6 for details

pulse_centering

Because the pulse will evolve in the fiber, it's hard to have the moving frame always move with the same speed as the pulse. As a result, the pulse will go out of the time window and come back from the other side due to the use of periodic assumption of discrete Fourier transform. The shift in time is saved in "prop_output.t_delay" so that you don't lose the information

When enabling pulse_centering, the pulse will be centered to the center of the time window based on the moment of the field intensity ($|A|^2$).

true (1) center the pulse according to the time window
false (0) don't center the pulse

cuda_dir_path

The path to the cuda directory into which ptx files will be compiled and stored. This is "/GMMNLSE/cuda/."

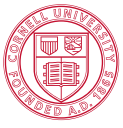
gpuDevice.Index

The GPU to use. It's typically 1 if the computer has only one GPU. MATLAB starts the index with 1. By starting multiple MATLAB sessions, we can run simulations on different GPUs simultaneously if different sessions are set with different GPU indices here.

Here are the parameters for the progress bar used in the simulation. It's useful in general to see how a simulation progresses.

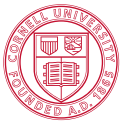
progress_bar

true (1) show progress bar
false (0) don't show progress bar



progress_bar_name

The name of the GMMNLSE shown on the progress bar. If not set (no "sim.progress_bar_name"), it uses a default empty string, ".



Chapter 4

Output arguments

fields
The $N_t \times N_m \times N_z$ output fields.

dt
This is the time sampling step $\triangle t$ with a unit of ps.

z
This is the positions of each saved field.

dz
The z-step size (m).

For an adaptive-step method, this contains the step size at each saved point. You can see how the step size evolves through the propagation with this parameter.

betas
The "sim.betas,"$[\beta_{(0)}; \beta_{(1)}]$, used in this propagation.

t_delay
The time delay of the pulse at each saved point due to pulse centering.

seconds
The time spent for this simulation.

4.1 For rate-equation gain model

4.1.1 Power

Here saves the pump and ASE power. They are saved in the "prop_output.Power" structure.

Power.pump.forward
The forward pump power along the fiber. Its size is $1 \times 1 \times N_z$.

Power.pump.backward
The backward pump power along the fiber. Its size is $1 \times 1 \times N_z$.

If ASE is considered,

Power.ASE.forward
The forward ASE spectral power ($P_{\mathrm{ASE}}(\nu)$ with the unit W/THz) along the fiber. Its size is $N_t \times N_m \times N_z$ if run with multimode and $1 \times 1 \times N_z$ if run with single mode.

Power.ASE.backward
The backward ASE spectral power ($P_{\mathrm{ASE}}(\nu)$ with the unit W/THz) along the fiber. Its size is $N_t \times N_m \times N_z$ if run with multimode and $1 \times 1 \times N_z$ if run with single mode.

4.1.2 Others

population
The doped ion density of the population of various energy levels. Its size is $N_x \times N_x \times N_z \times N_{\ell-1}$ if run with multimode and $1 \times 1 \times N_z \times N_{\ell-1}$ if run with single mode. N_x is the number of cross-sectional spatial sampling, and N_ℓ is the number of energy levels.

doped ion	N_ℓ	**energy levels (low to high)**
Nd	12	${}^4I_{9/2}, {}^4I_{11/2}, {}^4I_{13/2}, {}^4I_{15/2}, {}^4F_{3/2}, {}^4F_{5/2}, {}^4F_{7/2}, {}^4F_{9/2}, {}^4H_{11/2}, {}^4K_{13/2}, {}^4K_{15/2}, {}^4P_{1/2},$
Yb	2	${}^4F_{7/2}, {}^4F_{5/2}$
Er	9	${}^4I_{15/2}, {}^4I_{13/2}, {}^4I_{11/2}, {}^4I_{9/2}, {}^4F_{9/2}, {}^4S_{3/2}, {}^4F_{7/2}, {}^4F_{5/2}, {}^4H_{9/2}$
Tm	8	${}^3H_6, {}^3F_4, {}^3H_5, {}^3H_4, {}^3F_3, {}^1G_4, {}^1D_2, {}^1I_6$
Ho	5	${}^5I_8, {}^5I_7, {}^5I_6, {}^5I_5, {}^5I_4$

The following output occrus only when gain_rate_eqn.reuse_data=true.

saved_data
This is used for an oscillator to converge faster. There's no need for a user to read this. It'll be sent to GMMNSLE_propagate() in the next roundtrip. Please see several examples about oscillators to learn how to use this variable.

Chapter 5

Polarization modes

If the simulation is solved with "sim.scalar=true," the input field takes only the scalar fields, e.g.,

$$\Big[\text{mode 1} \quad \text{mode 2} \quad \text{mode 3} \quad \ldots\Big].$$

Otherwise, the input field of each polarized mode needs to be specified in the order of

$$\Big[\text{mode } 1_+ \quad \text{mode } 1_- \quad \text{mode } 2_+ \quad \text{mode } 2_- \quad \ldots\Big],$$

where (+,-) can be (x,y), (right-handed circular, left-handed circular), or any orthogonally polarized modes.

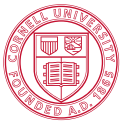
If the input β has a dimension of only the number of spatial modes, N_{sm}, I assume there's no significant influence from birefringence; thus, it's expanded into $2N_{sm}$ dimension with each i and j (polarization) modes being degenerate by GMMNLSE_propagate(). For polarized fields,

$$S^R_{plmn} = \frac{\int \mathrm{d}x\,\mathrm{d}y\Big[\mathbf{F}^*_p \cdot \mathbf{F}_l\Big]\Big[\mathbf{F}^*_m \cdot \mathbf{F}_n\Big]}{\left[\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_p|^2\right)\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_l|^2\right)\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_m|^2\right)\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_n|^2\right)\right]^{1/2}} \tag{5.1}$$

$$S^K_{plmn} = \frac{2}{3}S^R_{plmn} + \frac{1}{3}\frac{\int \mathrm{d}x\,\mathrm{d}y\Big[\mathbf{F}^*_p \cdot \mathbf{F}^*_n\Big]\Big[\mathbf{F}_m \cdot \mathbf{F}_l\Big]}{\left[\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_p|^2\right)\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_l|^2\right)\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_m|^2\right)\left(\int \mathrm{d}x\,\mathrm{d}y|\mathbf{F}_n|^2\right)\right]^{1/2}} \tag{5.2}$$

Therefore, S^R_{plmn} isn't zero as (p,l) and (m,n) both have the same polarization, and we get four possibilities for (p,l,m,n), (0,0,0,0), (0,0,1,1), (1,1,0,0), and (1,1,1,1), with their values directly derived from the scalar S^R_{plmm}. For S^K_{plmn}, in addition to the permutations of S^R_{plmn}, we need to consider those from the fraction above which isn't zero as (p,l,m,n) is (0,0,0,0), (0,1,1,0), (1,0,0,1), and (1,1,1,1). Notice that some of them can add up with S^R_{plmn} while some of them can't, so the value has a prefactor of 1, $\frac{2}{3}$, $\frac{1}{3}$.

The above generalization of the scalar S^R to polarized S^R, S^K needs each $\mathbf{F}_p$ to be either parallel or orthogonal to one another, so (i,j) has to be an orthogonal group in 2D, e.g., (x, y) or (σ_+, σ_-).



Chapter 6

Rate-equation gain model

To run with rate-equation gain model, you need to run "gain_info()" first. This precomputes the required information for this model and saves the computational time. For example, it computes the doped ion density based on the absorption and the cladding area. It also loads the multimode spatial profiles for multimode gain evolution.

Below is the code sequence of how to run the rate-equation gain model:

```
f = ifftshift( (-N/2:N/2-1)/N/dt + sim.f0 ); % in the order of "omegas" in
    GMMNLSE_propagate()
c = 299792.458; % nm/ps
lambda = c./f; % nm

% First call gain_info():
gain_rate_eqn = gain_info( fiber,sim,gain_rate_eqn,lambda );

% And then send it to GMMNLSE_propagate():
output_field = GMMNLSE_propagate( fiber,input_field,sim,gain_rate_eqn );
```

6.1 Input arguments

Most of the important parameters are contained in the gain_rate_eqn structure. Some parameters are used only with multimode simualtions. I labelled in blue those required all the time whether it's single-mode or multimode. I put (SM) if it's only for single-mode simulations and (MM) if it's only for multimode ones.

6.1.1 gain_rate_eqn

Multimode mode-profile folder

MM_folder$^{(MM)}$

A string; where the betas.mat and S_tensor_?modes.mat are. This is used only for multimode simulations which need to load their betas and SR values in the mat files from the mode solver.

Oscillator info

reuse_data

True (1) or false (0).

For a ring or linear cavity, the pulse will enter a steady state eventually. If reusing the pump and ASE data from the previous roundtrip, the convergence can be much faster, especially for counterpumping.

linear_oscillator

True (1) or false (0), about whether the simulation is for a linear oscillator.

For a linear oscillator, there are pulses from both directions simultaneously, which will both contribute to saturating the gain; therefore , the backward-propagating pulses need to be taken into account.

For a linear oscillator, gain_rate_eqn.reuse_data must be "true" to consider the backward-propagating pulses. If gain_rate_eqn.reuse_data is "false", gain_info() will correct it to "true".

How to use it:

```
% previous_rate_gain_saved_data comes from the GMMNLSE_propagate() output
    from the previous roundtrip
gain_rate_eqn.saved_data = previous_rate_gain_saved_data;
prop_output = GMMNLSE_propagate(fiber,...
                                input_field,...
                                sim,...
                                gain_rate_eqn);
(next)previous_rate_gain_saved_data = prop_output.saved_data;
```

Gain-fiber info

core_diameter

For double-clad fibers, this is where the doped ion is and the pulse propagates in (μm).

cladding_diameter

The cladding diameter (μm).

core_NA$^{(SM)}$

The core numerical aperture of the gain fiber. This is used only for single-mode simulations to calculate the MFD and further the overlap factor between the signal pulse and the doped ion. For multimode, the overlap factor is obtained from loaded spatial profiles; therefore, it doesn't need core_NA.

Doped-ion info

absorption_wavelength_to_get_N_total

The wavelength specified by the manufacturer which they use to measure the absorption of the gain fiber (nm).

absorption_to_get_N_total

The absorption measured with the wavelength specified above (dB/m).

If the pump power is weak such that the upper-state population is negligible ($N_2 = 0$, and thus $N_1 = N_{\text{total}}$), pump power follows

$$P_P(z + \triangle z) = \exp\left(-\frac{A_{\text{core}}}{A_{\text{cladding}}}\sigma_a(\nu_P)N_{\text{total}}(z)\triangle z\right)P_P(z). \tag{6.1}$$

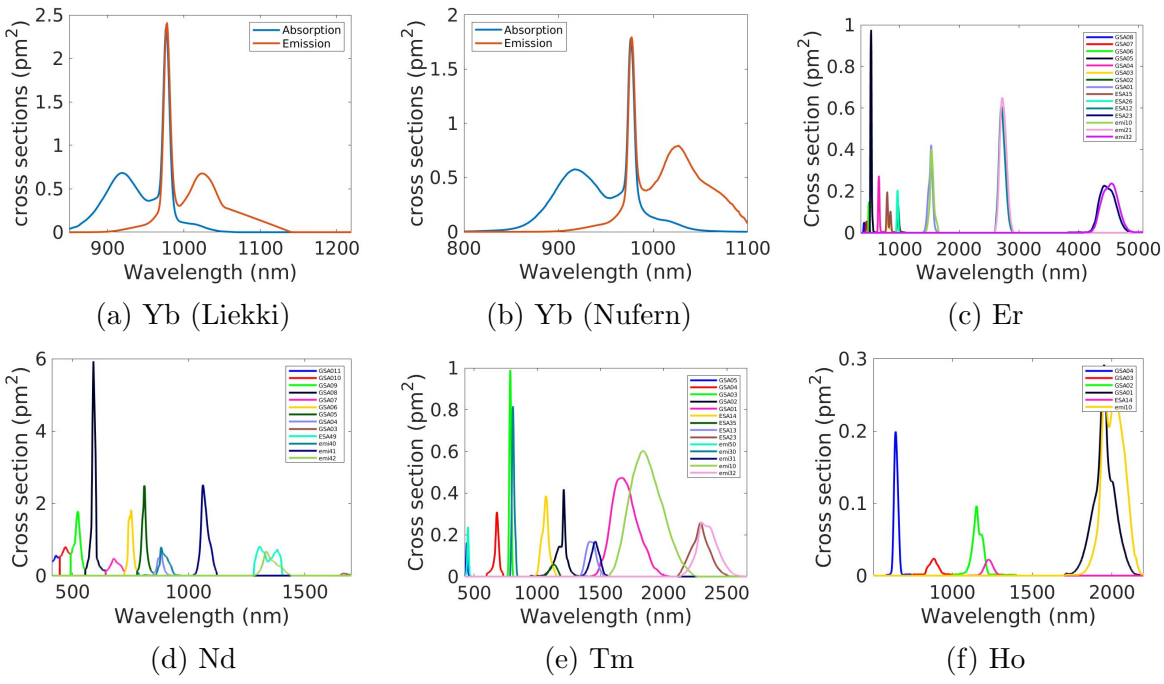
Therefore, the absorption in dB/m is $\alpha_{\text{dB/m}} = 10\log_{10}\left[\exp\left(\frac{A_{\text{core}}}{A_{\text{cladding}}}\sigma_a(\nu_P)N_{\text{total}}(z)\right)\right]$, which leads to the total doped-ion population

$$N_{\text{total}} = \frac{\ln\left(10^{\alpha_{\text{dB/m}}/10}\right)}{\frac{A_{\text{core}}}{A_{\text{cladding}}}\sigma_a(\nu_P)} = \frac{\frac{\alpha_{\text{dB/m}}}{10}\ln(10)}{\frac{A_{\text{core}}}{A_{\text{cladding}}}\sigma_a(\nu_P)}. \tag{6.2}$$

gain_medium

The gain medium. Current options are Nd, Yb, Er, Tm, and Ho.

For Yb, its cross sections include "Liekki Yb_AV_20160530.txt" and "Yb_Gen_VIII_Cross_Section (Nufern).txt" For others, due to their multi-level features, they are more complicated. Check the comments in the functions "read_cross_sections_?()" for details.



(a) Yb (Liekki) (b) Yb (Nufern) (c) Er

(d) Nd (e) Tm (f) Ho

Figure 6.1: Absorption and emission cross sections.

base_medium

The base medium of the fiber. Typical option is "silica". For other possible options for each type of doped ion, please see the function "calc_Judd_Ofelt()". I organized them here:

Cornell University

Doped ion	base_medium
Nd	silicate[9], silica[10], flurophosphate[11], ZBLAN[11]
Yb	silica
Er	ERZSG[12], silica[13, 14], silicate[15], flurophosphate[15], phosphate[15], ZBLAN[15–17], fluoride, ZBLA[15], tellurite[18]
Tm[19]	silica, silicate, fluorophosphate, ZBLAN, fluoride, germanate, tellurite
Ho[19]	silica, silicate, fluorophosphate, chalcohalide, ZBLAN, fluoride, germanate, tellurite

Pump info

pump_wavelength
The pump wavelength (nm).

copump_power
This is the pump sending in from the input end of the fiber (W). It co-propagates with the signal pulse. If it's counterpumping, set this to zero.

counterpump_power
This is the pump sending in from the output end of the fiber (W). It counter-propagates with the signal pulse. If it's copumping, set this to zero.

Mode profiles

If "mode_profiles" is ignored in the input arguments of gain_info(), it will read the mode profiles from gain_rate_eqn.MM_folder. Some extra operations on these mode profiles are done with the following parameters.

downsampling_factor$^{(MM)}$
This is the factor of downsampling that reduces the size of mode profiles for multimode. Typically the loaded mode profiles from the mode solver is 800×800, so downsampling it down to 100×100, or even smaller, speeds up a simulation a lot. Under this circumstances, gain_rate_eqn.downsampling_factor=8.

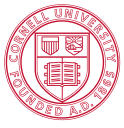
Computational info

t_rep
The roundtrip time (1/repetition rate) of the pulse, which is used to calculate the power of the "signal" pulse (s).

This rate-equation gain model assumes a high repetition rate such that the gain is able to reach a steady state. Therefore, the information of the repetition rate of the pulse is necessary. If the repetition is too low, gain_info() will throw a warning.

ASE info

ignore_ASE
True (1) or false (0).



sponASE_spatial_modes

The number of available spatial modes for ASE.

In principle, the number of spatial modes for ASE should be the same as those for signal fields. However, due to computational simplicity, a smaller number of spatial modes for signal fields might be considered. In such a situation, ASE spatial modes should still be considered to correctly approximate the amount of generated ASE. For example, in large-mode-area fibers, the number of ASE modes can be larger than that of the signal field, where usually, only the fundamental mode is considered for the signal field due to fiber coiling. If this value is left empty like "[ ]", it is "length(sim.midx)," the number of signal spatial modes. I use "sponASE" because ASE grows from spontaneous emission. The number of spatial modes manifests itself as more spontaneous emission generated.

Algorithm info

max_iterations

The maximum number of iterations.

If having iterations is required, you can see that more iterations are needed for a longer fiber to converge. Play around this value to get the result to converge.

tol

The tolerance of this iteration. If the difference of pulse energy or ASE power between the last two results is smaller than this tolerance, it's done.

verbose

Show the information (final pulse energy) during iterations.

memory_limit

The memory limit for the simulation. This can be found by default. It's important only when gain_rate_eqn.reuse_data=true for an oscillator.

With GPU computing, it is "sim.gpuDevice.Device.AvailableMemory/2". Otherwise, it looks for the available RAM for MATLAB and becomes "userview.MemUsedMATLAB/2 $\cdot$ 2^{20}" for windows. This is supported only under windows and linux, not iOS because I have no experience in iOS. The command between linux and iOS shouldn't differ too much, it's possible to implement it.

6.1.2 lambda and mode_profiles

lambda

The wavelengths of the simulation (nm). It's ordered as right after taking "ifft." Check the code of how to call gain_info() at the beginning of this chapter.

mode profiles

If this isn't specified or left empty, gain_info() will load mode profiles from gain_rate_eqn.MM_folder.

mode_profiles

The eigenmode profiles of field amplitudes. It'll be normalized into the unit of 1/μm in gain_info().

mode_profiles_x
The x-position of mode profiles (µm). It's an array of length N_x.

6.2 Output arguments

Precomputing overlap_factor, FmFnN, and GammaN is the main reason of running this function before GMMNLSE_propgate() with the rate-equation gain. For multimode, it saves a huge amount of time.

gain_rate_eqn.cross_sections_pump
The absorption and emission cross sections at the pump wavelength (µm^2).

gain_rate_eqn.cross_sections
The absorption and emission cross sections over the frequency domain (µm^2). This is used for signal pulse and ASE.

gain_rate_eqn.overlap_factor
The overlap factors of both the pump and the signal pulse. It contains overlap_factor.pump and overlap_factor.signal and determines the overlap between each mode and the doped ion. It has no unit for single-mode but has the unit of 1/µm^2 for multimode.

gain_rate_eqn.N_total
The doped ion density (1/µm^3). For single-mode, it's a scalar; while for multimode, its size is $N_x \times N_x$.

gain_rate_eqn.FmFnN
It precomputes $\int_{A_{\text{core}}} F_{m_i} F_{n_i}^* N_T \, \mathrm{d}^2x$, the integral2(overlap_factor*N_total), for the signal and ASE.

gain_rate_eqn.GammaN
It precomputes $\int_{A_{\text{core}}} \frac{N_T}{A_{\text{cladding}}} \, \mathrm{d}^2x$, the integral2(overlap_factor*N_total), for the pump.

Chapter 7

load_default_GMMNLSE_propagate()

Because of the overwhelming parameters of input arguments, I've created a function that loads the default value for each parameter. If a user has specified the value already, the user's value precedes over the default one.

Here is a typical way of calling this function.

```
[fiber,sim] = load_default_GMMNLSE_propagate(input_fiber,...
                                             input_sim[,type_of_mode])
```

input_fiber and input_sim are user-defined parameters. type_of_mode is either 'single-mode' or 'multimode'; if it's ignored, 'single-mode' is assumed by default. Below are some examples.

```
% User-defined parameters
fiber.betas = [0;0;0.02;0];
fiber.L0 = 3; % m

% Incorporate default settings
[fiber,sim] = load_default_GMMNLSE_propagate(fiber,[]); % single_mode

% If there are "sim" settings
sim.gpu_yes = false;
[fiber,sim] =  load_default_GMMNLSE_propagate(fiber,sim); % single_mode

% Use only user-defined "sim", not "fiber"
[fiber,sim] = load_default_GMMNLSE_propagate([],sim); % single_mode

% For multimode, you must add the string 'multimode' as the last argument.
[fiber,sim] = load_default_GMMNLSE_propagate(fiber,sim,'multimode');
```

Besides loading the default values, this function gives a user more options to obtain several parameters. This function transforms them into the allowed parameters of GMMNLSE_propagate(). I list them below. If both equivalence are specified unfortunately, the allowed GMMNLSE_propagate() input has the higher priority.

Description	Allowed GMMNLSE_propagate()'s input	Equivalent input arguments for this function
center frequency/wavelength	sim.f0 (THz)	sim.lambda0 (m)
nonlinear coefficient	fiber.SR (m^{-2})	fiber.MFD (μm)

Several other input arguments are

midx

An array of the mode indices. It helps select only those modes we want to use in the simulation. For example, if I want only mode 2 and mode 4 in simulations,

```
sim.midx = [2,4];
```

This function will read "betas" and "SR" with

```
betas = betas_mat_file(:,midx);
SR = SR_mat_file(midx,midx,midx,midx);
```

To load multimode mode profiles, use the following three parameters.

MM_folder

This specifies the folder where betas and SRSK mat files are stored; only used in multimode.

betas_filename

The filename of the mat file that stores betas.

Note that the input unit of betas in GMMNLSE_propagate() is ps^n/m while the one from the mode solver is fs^n/m. Besides loading the betas data, this function helps transform into the unit GMMNLSE_propagate() needs after loading. If the user provides their own betas, they need to make sure the unit is correct; this function assumes the user's input has the correct unit and won't modify it.

S_tensors_filename

The filename of the mat file that stores S^R tensors.

A few values about the gain are used only in this file to calculate the gain saturation intensity or energy. They are labelled with an asterisk $\star$. If you provide the saturation intensity or energy directly, you don't need to worry about these parameters.

Below is the process flow of this "load_default_GMMNLSE_propagate()" function. Read this if you're not sure whether your input will be used or overwritten. Because user-defined parameters take precedence, overwritten should happen only for (f0,lambda0) and (SR,MFD) mentioned above.

```
% <-- Uncorrelated parameters are loaded directly -->

sim.f0 - depend on input f0 or lambda0
         If no input f0 or lambda0, f0=3e5/1030e-9 (THz)

% If there's a user-defined one, use user's instead for the parameters below.
% Below I list the default values -->
fiber.material = 'silica';
fiber.n2 = 2.3e-20;

sim.dz = 1000e-6;
sim.save_period = 0;
sim.ellipticity = 0; % linear polarization

sim.MPA.M = 10;
sim.MPA.n_tot_max = 20;
sim.MPA.n_tot_min = 2;
sim.MPA.tol = 1e-6;
```

```
19
20 sim.rmc.model = false;
21 sim.rmc.varn = 0;
22 sim.rmc.stdQ_polarizedmode = 0;
23 sim.rmc.lambda0 = default_sim.lambda0;
24 sim.rmc.downsampling_factor = 1;
25
26 sim.scalar = true;
27
28 sim.adaptive_dz.threshold = (1e-6 if RK4IP or 1e-3 if MPA);
29
30 sim.gpu_yes = true;
31 sim.gain_model = 0;
32
33 sim.pulse_centering = true;
34 sim.include_Raman = true;
35 sim.gpuDevice.Index = 1;
36 sim.progress_bar = true;
37 sim.progress_bar_name = '';
38 sim.cuda_dir_path = 'GMMNLSE/cuda';
39
40 % <-- Correlated parameters are loaded based on the input or default -->
41
42 % single-mode -- >
43
44 sim.midx = 1;
45
46 %   Assume 920 nm for positive dispersion if lambda0 < 1000 nm,
47 fiber.betas = [9.8810e6; 4.8872e3; 0.0315; 2.0457e-5; 1.2737e-9];
48 fiber.MFD = 4; % um; 1030nm from IXblue's IXF-2CF-PAS-PM-4-80-0.16-P
49 %   Assume 1030 nm for positive dispersion if 1000 nm < lambda0 < 1300 nm (~ZDW
       for a silica fiber),
50 fiber.betas = [8.8268e6; 4.8821e3; 0.0209; 32.9e-6; -26.7e-9];
51 fiber.MFD = 5.95; % um; from Thorlabs 1060XP
52 %   Assume 1550 nm for negative dispersion if lambda0 > 1300 nm (~ZDW for a
       silica fiber),
53 fiber.betas = [5.8339e6; 4.8775e3; -0.0123; 0.1049e-6; -378.3e-9];
54 fiber.MFD = 8.09; % um; from Thorlabs 1060XP
55
56 (input SR precedes over input MFD)
57 fiber.SR = (1) input SR, if there's input SR
58            (2) 1/Aeff, if (a) there's input MFD
59                           (b) MFD is taken from the default one and there's no
                                 input MFD
60
61 *fiber.gain_Aeff = 1/fiber.SR (taken from above)
62 *fiber.gain_doped_diameter = fiber.MFD;
63
64 % multimode -->
65
66 fiber.MFD = [] (not used)
67
68 sim.midx = (1) input midx
69            (2) 1:num_modes (num_modes is determined by loading "betas.mat")
70
71 fiber.betas = (1) input betas
```

```
72                (2) loaded from betars_filename in fiber.MM_folder (loaded modes
                      are based on the above midx)
73  fiber.SR - (1) input SR
74             (2) loaded from S_tensors_filename in fiber.MM_folder (loaded modes
                   are based on the above midx)
75  *fiber.gain_Aeff = (1) pi*( (input gain_doped_diameter) *1e-6/2)^2;
76                     (2) 1/fiber.SR(1,1,1,1) (taken from above)
77  *fiber.gain_doped_diameter = (1) input gain_doped_diameter
78                               (2) fiber.MFD;
79
80  % For both single-mode and multimode -->
81
82  fiber.L0 = (1) input L0
83             (2) 2 (m)
84
85  fiber.dB_gain = (1) input gain under dB/m
86                  (2) 30 (dB/m)
87  % overlap_factor is obtained from gain_doped_core. It's the overlap between the
       mode fields and the doped core.
88  fiber.gain_coeff = (1) (input gain_coeff)*overlap_factor
89                     (2) fiber.dB_gain*log(10)/(10*fiber.L0)*overlap_factor; % m
                           ^-1, from db/m
90  fiber.gain_fwhm = (1) input gain_fwhm
91                    (2) 40e-9; % m
92
93  *fiber.gain_tau = (1) input gain_tau
94                    (2) 840e-6; % s; 840 us is the lifetime of Yb ions
95  *fiber.t_rep = (1) input t_rep
96                 (2) 1/15e6; % s; assume 15 MHz repetition rate
97  *fiber.gain_cross_section = (1)input gain_cross_section
98                              (2) 6.43e-25 + 4.53e-26; % m^2; the total cross
                                    section of Yb ions at 1030 nm
99
100 fiber.saturation_intensity = (1) input saturation_intensity
101                             (2) calculated according to gain_tau, t_rep, and
                                    gain_cross_section above
102 fiber.saturation_energy = (1) input saturation energy
103                           (2) calculated according to saturation_intensity and
                                  gain_Aeff above
```

Chapter 8

Obtain the multimode information (BuildFiber)

In this package, we also offer the capability of computing the multimode mode profiles, propagation constant of each mode, and the required overlap integrals among modes [Eq. (1.7)]. In particular, the generated files are related to the commands of **MM_folder**, **betas_filename**, and **S_tensors_filename** in the function **load_default_GMMNLSE_propagate()**.

The capability is implemented with those in the **BuildFiber** folder, whose usage follows

1. Run **solve_for_modes** with the desired fiber parameters. This creates the mode profiles and the effective indices of each mode at different wavelengths.

2. Run **calc_fiber_parameters** for the generated fiber data. With the previously-generated effective indices, propagation constants is found. With the mode profiles, overlap integrals is found.

3. These creates all required data in a folder in "Fibers/." Copy this folder elsewhere for load_default_GMMNLSE_propagate() to load for.

8.1 Solve_for_modes

This script computes the mode profiles in parallel (with MATLAB's "parfor"), so MATLAB's **parallelization toolbox** is required. The overlap integrals are always computed at the specified center frequency, while the propagation constants can be the (narrowband) Taylor-series coefficients [Eq. (3.2)] or the (broadband) frequency-dependent column vector for each mode [Eq. (3.3)].

It requires setting the following parameters:

Spatial information

Nx

The number of spatial points of mode profiles. 400 should be good to go.

spatial_window

The size of the spatial window. It is in μm.

Spectral information

Nf

The number of frequency points at which the modes will be calculated. For computing the (narrowband) Taylor-series coefficients, 20 is enough.

wavelength0

The center wavelength of the frequency window to compute the mode information. Its unit is m.

freq_range

The size of the frequency window. If it is 0, only the center wavelength will be considered. It is usually 100 THz. Its unit is THz.

Mode information

num_modes

The number of modes to compute for.

This code will use the largest frequency in the frequency window to determine the maximum mode supported in the fiber if "include_cladding_modes=false" (the command introduced later). It is determined based on seeing whether the energy distribution is confined in the core; if not, it is considered a cladding mode that is not supported by the fiber (core). Therefore, if the user wants to compute all the modes of the fiber without knowing exactly how many modes are supported, they can just put a huge number here.

include_cladding_modes

True (1) or false (0).

Because the computation of modes rely on finite element, the mode needs to satisfy the exterior boundary condition as well. If the mode extends significantly beyond the user-defined core, it will be affected the the exterior boundary condition, creating another set of spatial eigenmodes. This is what I call "cladding modes" here. If "include_cladding_modes=false," the computation will include these modes if possible. Fig. 8.1 illustrates the idea, but its spatial window is made larger than the cladding. If the spatial window is smaller than the cladding, the generated cladding mode will be distorted by the (exterior) boundary condition of the spatial window.

Fiber information

Any fiber parameters can come from user's inputs or from the repository I created in **fiber_collections()**.

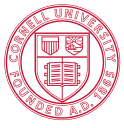
use_fiber_collection

True (1) or false (0).

This controls whether to use the fiber repository in this package or not. If "true," the user needs to specify the fiber name. Please find them in "BuildFiber/helpers/fiber_collection." User is free to add their own fibers in it.

fiber

The fiber name.



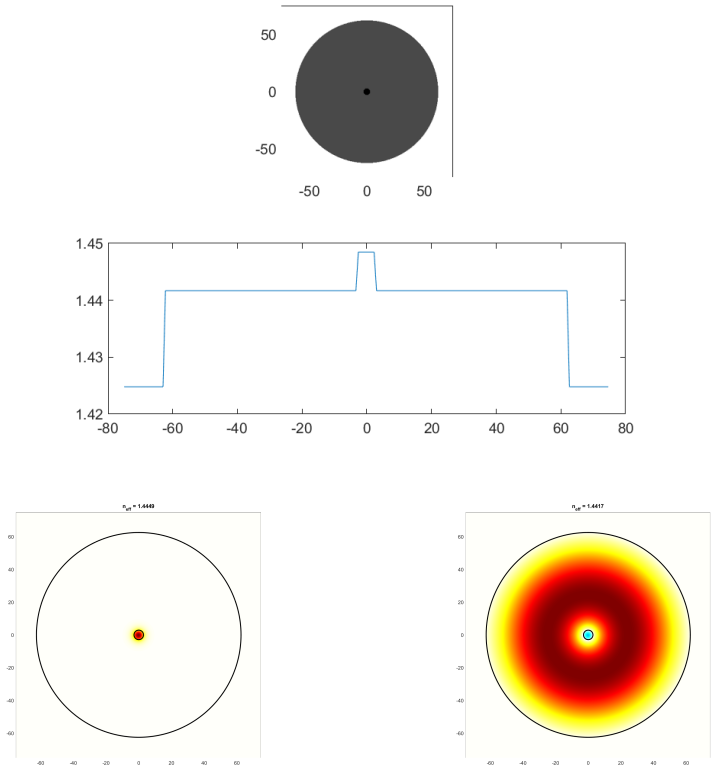


Figure 8.1: Thorlabs 1060XP single-mode fiber for 1060 µm. (Top) Fiber index profile. (Bottom left) The supported mode of this fiber. (Bottom right) The cladding mode that propagates inside the 125-µm cladding. Typically, the spatial window only needs to cover the desired core regime. This figure is for demonstration purpose, so it has a huge spatial window covering the entire cladding.

If "use_fiber_collection=false," then user needs to provide the fiber parameters themselves, which include

alpha
The shape parameter α for graded-index (GRIN) fibers. It controls the index gradient that follows $n(r) = n_0 \left[1 - \left(\frac{r}{a}\right)^{\alpha}\right]$ for $r \leq a$.

fiber_type
The type of fiber index profile. It is a string of either 'step' or 'GRIN.'

core_diameter
Core diameter of the fiber. It is in µm.

clad_diameter
Cladding diameter of the fiber. For almost all situations, this parameter is not important since we consider only the supported modes. It is in µm.

core_NA

The numerical aperture (NA) of the core. Since commercial fibers show only the NA, it is used to find the index contrast and the profile.

clad_NA
The numerical aperture (NA) of the cladding. Since commercial fibers show only the NA, it is used to find the index contrast and the profile.

fname_user_defined
The folder name for generated mode profiles and other data.

8.2 calc_fiber_parameters

After computing the mode profiles, use this script to find the propagation constants, by calling **calc_dispersion()**, and overlap integrals, by calling **calc_SRSK_tensors()**. It calls **delete_modeprofiles()** that removes mode profiles at wavelengths other than the center wavelength to save the disk space.

mode_used
The modes to consider. This is a scalar array. Since **solve_for_modes** cannot selectively compute the modes we want, it always starts with the fundamental mode and consecutively computes up to the maximum mode index we specify. However, in practice, we might only want a few modes in computations, such as only the second the the third modes, LP_{11a} and LP_{11b}. This parameter is thus used to select only the mode we want. For example, to output the data only for LP_{11a} and LP_{11b}, it should be $[2, 3]$.

Nx
The number of spatial points of mode profiles. This should be the same as Nx in **solve_for_modes**.

gpu_yes
True (1) or false (0). Whether to use GPU or not.

folder_name
The folder name of the generated data. This should be the same as fname_user_defined in **solve_for_modes**.

Nf
The number of frequency points. This should be the same as Nf in **solve_for_modes**.

wavelength0
The center wavelength. This should be the same as wavelength0 in **solve_for_modes**.

freq_range
The size of the frequency window. This should be the same as freq_range in **solve_for_modes**.

bandwidth
'narrowband' or 'broadband'.

This controls the type of propagation constants to output. If it is 'narrowband,' it will compute the propagation constants with Taylor-series coefficients at the center wavelength

[Eq. (3.2)]. If it is 'broadband,' it will output the propagation constants as a matrix with column vectors of propagation constant of each mode [Eq. (3.3)].

polynomial_fit_order

The polynomial fitting order of the Taylor series of propagation constants if " 'bandwidth='narrowband'." This is usually 7.

num_disp_orders

The order of propagation constants to output. Although polynomial_fit_order controls the fitting order, not all Taylor-series coefficients are necessary saved. The saved order is control by this parameter, so num_disp_orders cannot be larger than polynomial_fit_order. This is usually 5. Five orders is sufficient for simulating narrowband pulse propagation. From my experience, a narrowband pulse is typically inert to high-order dispersion effects, which only broadband pulse suffers from. However, Taylor-series expansion is only good around a narrowband regime, so 'bandwidth=broadband' option should be used to generate column-vector frequency-dependent propagation constants, rather than increasing the number of Taylor-series orders.

Chapter 9

Diagram of the calling sequence

It's not necessary to know how or when each function is called. I keep it here for documentation or in case someone wants to modify the code.

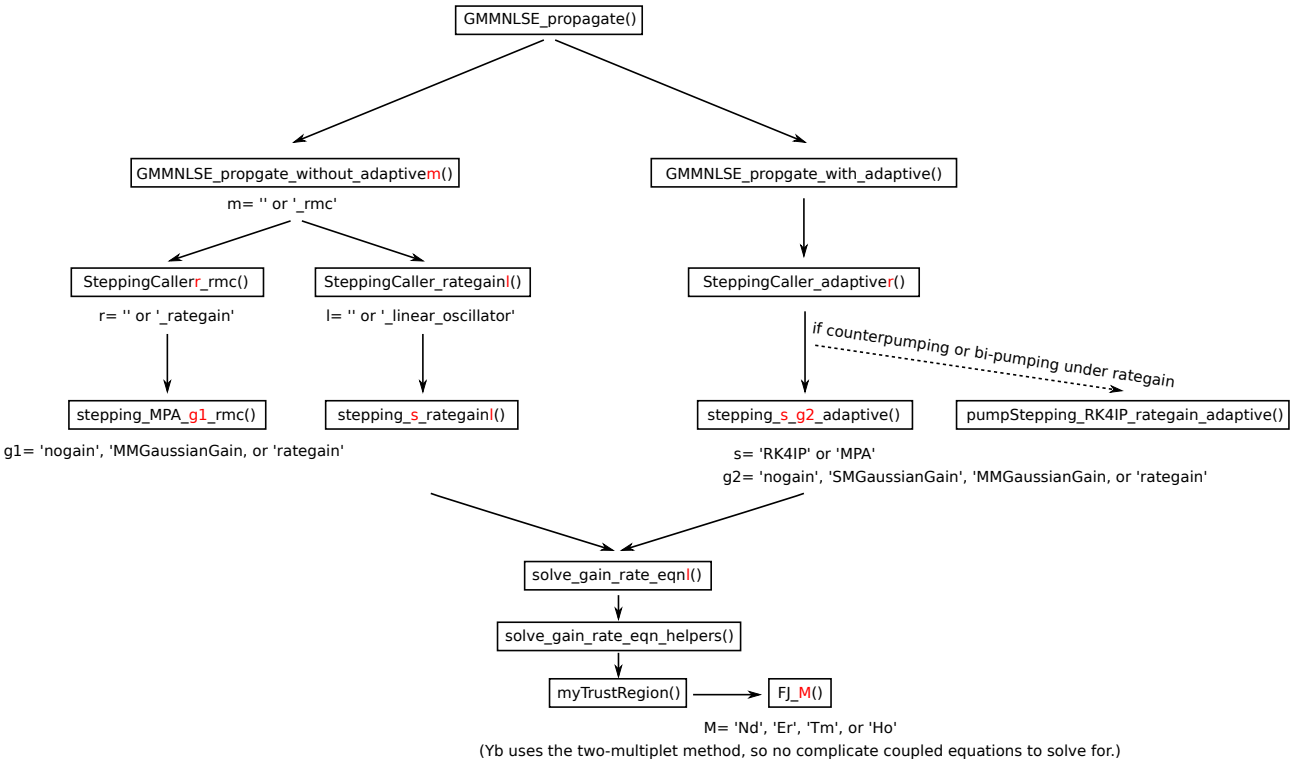


Figure 9.1: Diagram of the calling sequence.

The code uses "RK4IP" (Runge-Kutta in the interaction picture) for single mode [5, 6] and "MPA" (Massively Parallel Algorithm) for multimode [7].

Chapter 10

Dechirper and Stretcher

This chapter gives the phase accumulated after propagating through a grating dechirper or a stretcher. There are four configurations: Treacy [20], prism [21, 22], Offner [23], and Martinez [24, 25]. Although they can be found in papers, the studies in them might deviate people's attention. Here, I focus only on showing the full phase accumulation $\phi(\omega) = k\ell(\omega) + \phi_g(\omega)$, where k is the wave vector, ℓ is the path length, and ϕ_g is the grating phase. It varies with the light angular frequency ω and is implemented numerically to dechirp or stretch a pulse. If readers are interested in their group delay dispersion, $\frac{\mathrm{d}^2\phi}{\mathrm{d}\omega^2}$, or third-order dispersion, $\frac{\mathrm{d}^3\phi}{\mathrm{d}\omega^3}$, please refer to their papers. They are widely used in stretching and dechirping pulses mentioned throughout this thesis. Typically, the grating is designed to be a blazed grating worked under the Littrow configuration whose diffraction order $m = -1$ is only considered.

10.1 Treacy type

In this section, both reflective and transmissive Treacy grating dechirpers/stretchers are introduced. They can add negative chirp (corresponding to anomalous dispersion) to a pulse.

10.1.1 Reflective Treacy type

The single-pass optical path length (Fig. 10.1) is

$$\begin{aligned}\ell = \ell_1 + \ell_2 &= d\sec\theta_{\text{out}}\left[1 + \cos(\theta_{\text{in}} + \theta_{\text{out}})\right] \\ &= d\sec\theta_{\text{out}}\left(1 + \cos\theta_{\text{in}}\cos\theta_{\text{out}} - \sin\theta_{\text{in}}\sin\theta_{\text{out}}\right) \\ &= d\left(\sec\theta_{\text{out}} + \cos\theta_{\text{in}} - \sin\theta_{\text{in}}\tan\theta_{\text{out}}\right), \end{aligned} \tag{10.1}$$

where

$$\Lambda\left(\sin\theta_{\text{out}} - \sin\theta_{\text{in}}\right) = m\lambda, \tag{10.2}$$

Λ is the grating line spacing, and m is the diffraction order.

For the grating, the accumulated phase considers not only the geometric path length but also the grating phase. The first grating does not add any grating phase because all spectral components are diffracted at the same position. However, the second one imposes a grating phase because different spectral components are now diffracted at different positions of the grating. To calculate the grating phase, only the relative position matters.

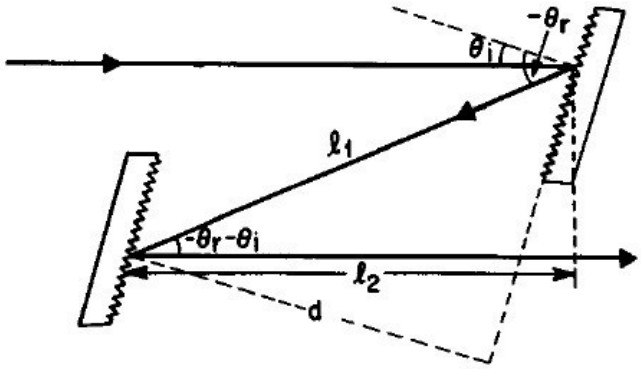


Figure 10.1: Reflective grating dechirper/stretcher [26].

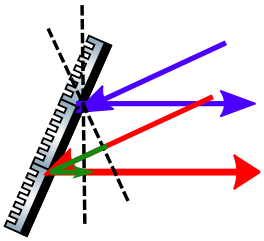


Figure 10.2: The grating phase of a Treacy-type reflective grating dechirper/stretcher.

Because of diffraction after the first grating, the red light propagates farther than the blue light (Fig. 10.2). This extra propagation phase (from the green line in Fig. 10.2) needs to be canceled so that it is directed horizontally after the grating. As a result, the grating phase follows

$$\phi_g(x) = \pi + m\frac{x}{\Lambda}2\pi. \tag{10.3}$$

π is due to the reflection and will be taken out if we use transmissive gratings. x is the relative position on the grating plane, which in the case of a reflective grating dechirper/stretcher, becomes $d\tan(-\theta_{\text{out}})$. Therefore, the single-pass phase $\phi_{\text{single-pass}}$ becomes

$$\begin{aligned}
\phi_{\text{single-pass}} &= k\ell + \phi_g \\
&= k\ell + \left[\pi + m\frac{2\pi d\tan(-\theta_{\text{out}})}{\Lambda}\right] \\
&= kd\left(\sec\theta_{\text{out}} + \cos\theta_{\text{in}} - \sin\theta_{\text{in}}\tan\theta_{\text{out}}\right) + \left(\pi - m\frac{2\pi d\tan\theta_{\text{out}}}{\Lambda}\right) \\
&= kd\left(\sec\theta_{\text{out}} + \cos\theta_{\text{in}}\right) + \pi - d\tan\theta_{\text{out}}\left(k\sin\theta_{\text{in}} + m\frac{2\pi}{\Lambda}\right) \\
&= kd\left(\sec\theta_{\text{out}} + \cos\theta_{\text{in}}\right) + \pi - kd\tan\theta_{\text{out}}\sin\theta_{\text{out}} \\
&= kd\left(\cos\theta_{\text{out}} + \cos\theta_{\text{in}}\right) + \pi.
\end{aligned} \tag{10.4}$$

To avoid spatial chirp, a mirror is introduced after the first pass of the grating pair so that all spectral components propagate back to where they are, eliminating the spatial chirp. Due to this extra reflecting propagation, the total phase is two times larger, that is,

$$\phi_{\text{double-pass}} = 2\left[kd\left(\cos\theta_{\text{out}} + \cos\theta_{\text{in}}\right) + \pi\right]. \tag{10.5}$$

10.1.2 Transmissive Treacy type

The derivation follows the reflective grating dechirper/stretcher discussed previously. The total optical path length (Fig. 10.3) is

$$\ell = \ell_1 + \ell_2 = \ell_1 + \left[M - (M - \ell_2)\right] . \tag{10.6}$$

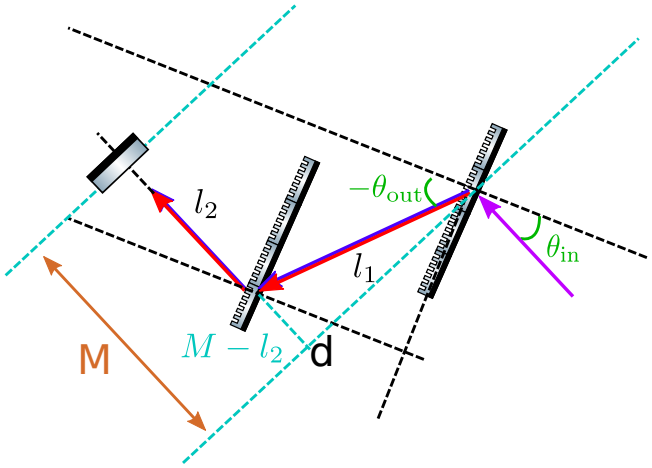


Figure 10.3: Transmissive grating dechirper/stretcher.

Since M is independent of frequency, I keep only the parameters relevant in pulse dechirping/stretching. Hence, ℓ becomes

$$\begin{aligned}
\ell &= \ell_1 + \left[M - (M - \ell_2)\right] \\
&\to \ell_1 - (M - \ell_2) \\
&= \ell_1 - \ell_1 \sin\left(\frac{\pi}{2} - \theta_{\text{in}} + \theta_{\text{out}}\right) \\
&= \ell_1 - \ell_1 \cos\left(\theta_{\text{in}} - \theta_{\text{out}}\right) \\
&= d \sec\theta_{\text{out}} \left[1 - \cos(\theta_{\text{in}} - \theta_{\text{out}})\right] \\
&= d \sec\theta_{\text{out}} \left(1 - \cos\theta_{\text{in}} \cos\theta_{\text{out}} - \sin\theta_{\text{in}} \sin\theta_{\text{out}}\right) \\
&= d \left(\sec\theta_{\text{out}} - \cos\theta_{\text{in}} - \sin\theta_{\text{in}} \tan\theta_{\text{out}}\right) .
\end{aligned} \tag{10.7}$$

Similar to the reflective gratings, the grating phase from the transmissive grating is added to calculate the single-pass total phase.

$$\begin{aligned}
\phi_{\text{single-pass}} &= k\ell + \phi_g = k\ell + m\frac{2\pi d \tan(-\theta_{\text{out}})}{\Lambda} \\
&= kd \left(\cos\theta_{\text{out}} - \cos\theta_{\text{in}}\right) .
\end{aligned} \tag{10.8}$$

To eliminate the spatial chirp, the total phase after introducing a mirror becomes

$$\phi_{\text{double-pass}} = 2\left[kd \left(\cos\theta_{\text{out}} - \cos\theta_{\text{in}}\right)\right] . \tag{10.9}$$

10.1.3 Group delay dispersion of Treacy dechirper/stretcher

Here, group delay dispersion (GDD) of both reflective and transmissive grating dechirpers/stretchers are shown. They are used to calculate an initial guess of the grating separation, d, for the subsequent grating-pair optimization schemes.

$$\frac{\mathrm{d}^2\phi_{\text{double-pass}}}{\mathrm{d}\omega^2} = -\frac{m^2\lambda^3 d}{\pi c^2 \Lambda^2 \cos^3\theta_{\text{out}}} \tag{10.10a}$$

$$\frac{\mathrm{d}^3\phi_{\text{double-pass}}}{\mathrm{d}\omega^3} = \frac{3m^2\lambda^4 d\,(1+\sin\theta_{\text{in}}\sin\theta_{\text{out}})}{2\pi^2 c^3 \Lambda^2 \cos^5\theta_{\text{out}}} \tag{10.10b}$$

where m is the diffraction order and is typically -1, λ is the pulse center wavelength, d is the grating separation, and Λ is the grating line spacing.

10.2 Prism type

As grating pair, prism pair that disperses color can be used as a dechirper or stretcher. It operates as in Fig. 10.4. The light should hit the prism as close to the apex as possible.

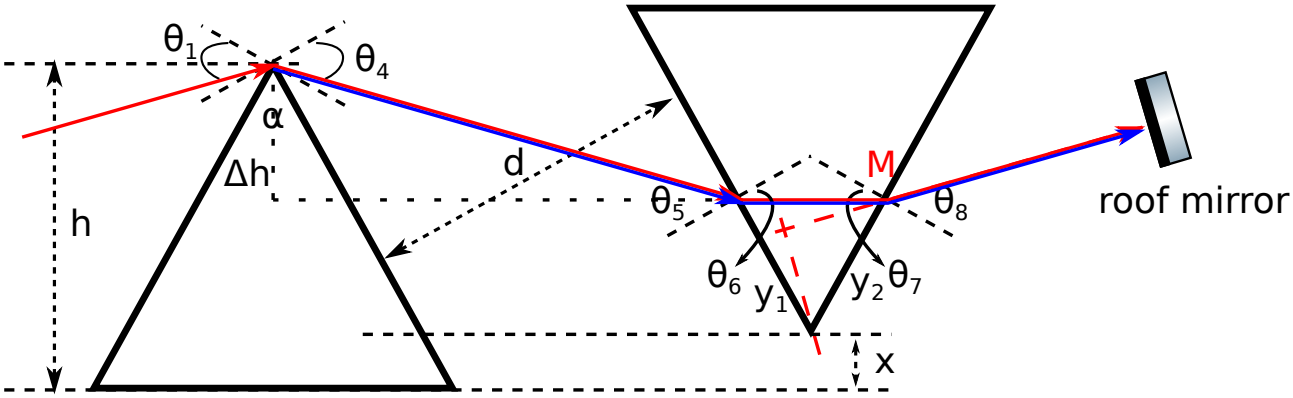


Figure 10.4: Prism dechirper/stretcher.

In principle, the α angle can be varied to tune the dispersion properties of a prism dechirper/stretcher. In practice, however, the geometry is chosen such that the incident and refracted beam have the same angle at the central wavelength of the spectrum to be dechirped/stretched. This configuration is known as the "angle of minimum deviation," and is easier to align than arbitrary angles (Fig. 10.5). In this configuration, the angles of refraction through the prism are symmetric, that is in Fig. 10.5,

$$\theta_1 = \theta_4 \tag{10.11a}$$

$$\theta_2 = \theta_3 = \frac{\alpha}{2}. \tag{10.11b}$$

To calculate the phase added to the pulse, the incident angle needs to be determined so that the beam of its center wavelength λ_0 enters the configuration of "angle of minimum deviation:"

$$\sin\theta_{\text{in}} = n(\lambda_0)\sin\theta_2, \quad \text{with } \theta_2 = \frac{\alpha}{2}$$

$$\Rightarrow \theta_{\text{in}} = \sin^{-1}\left(n(\lambda_0)\sin\frac{\alpha}{2}\right). \tag{10.12}$$

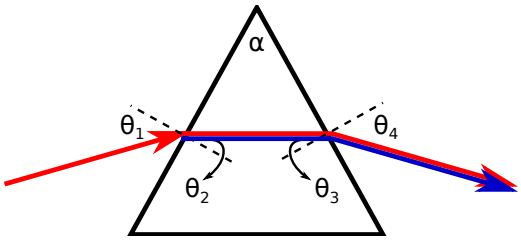


Figure 10.5: Prism operation with a minimum beam deviation.

Next, we calculate each angle in Fig. 10.5:

$$n(\lambda)\sin\theta_2 = \sin\theta_{\text{in}} \tag{10.13a}$$
$$\Rightarrow \theta_3 = \alpha - \theta_2 \tag{10.13b}$$
$$\Rightarrow \sin\theta_4 = n(\lambda)\sin\theta_3. \tag{10.13c}$$

With calculated θ_4 for each wavelength, we subsequently obtain

$$\theta_5 = \theta_4 \tag{10.14a}$$
$$\Rightarrow n(\lambda)\sin\theta_6 = \sin\theta_5 \tag{10.14b}$$
$$\Rightarrow \theta_7 = \alpha - \theta_6 \tag{10.14c}$$
$$\Rightarrow \sin\theta_8 = n(\lambda)\sin\theta_7. \tag{10.14d}$$

These lead to

$$\theta_5 = \theta_4 \tag{10.15a}$$
$$\theta_6 = \theta_3 \tag{10.15b}$$
$$\theta_7 = \theta_2 \tag{10.15c}$$
$$\theta_8 = \theta_1 = \theta_{\text{in}}. \tag{10.15d}$$

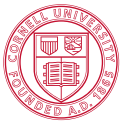
Eq. (10.15d) shows that all the colors exit the second prism with the same angle such that they can all be reflected back to their original paths after the roof mirror.

To find the path lengths after entering the second prism, we need to calculate the change of height of the beam when hitting the second prism. It is $\triangle h = \ell_s \cos\left(\frac{\pi}{2} - \theta_4 + \frac{\alpha}{2}\right) = \ell_s \sin\left(\theta_4 - \frac{\alpha}{2}\right)$, where the path length between two prisms is $\ell_s = d\sec(\theta_4)$, leading to $\triangle h = d\left(\tan\theta_4 \cos\frac{\alpha}{2} - \sin\frac{\alpha}{2}\right)$. The distance between where the beam hitting the prism and the apex of the second prism is $y_1 = (h - \triangle h - x)\sec\frac{\alpha}{2}$. It allows us to find the path length to travel inside the second prism ℓ_p and the length on the hypotenuse y_2 through the following relation:

$$\frac{y_1}{\sin\left(\frac{\pi}{2} - \theta_7\right)} = \frac{\ell_p}{\sin\alpha} = \frac{y_2}{\sin\left(\frac{\pi}{2} - \theta_6\right)}. \tag{10.16}$$

And thus

$$\ell_p = \frac{y_1 \sin\alpha}{\cos\theta_2} \tag{10.17a}$$
$$y_2 = \frac{y_1 \cos\theta_3}{\cos\theta_2} \tag{10.17b}$$



y_2 contributes to the change of path length to the roof mirror by $\ell_M = M - y_2\cos\left(\frac{\pi}{2} - \theta_8\right) = M - y_2\sin\theta_{\text{in}} \sim -y_2\sin\theta_{\text{in}}$.

In conclusion, the path lengths to consider are

$$\ell_s = d\sec\theta_4 \tag{10.18a}$$

$$\ell_p = \frac{y_1\sin\alpha}{\cos\theta_2} = \frac{(h - \triangle h - x)\sec\frac{\alpha}{2}\sin\alpha}{\cos\theta_2} = \frac{2\,(h - \triangle h - x)\sin\frac{\alpha}{2}}{\cos\theta_2} \tag{10.18b}$$

$$\ell_M = -y_2\sin\theta_{\text{in}} = -\frac{y_1\cos\theta_3}{\cos\theta_2}\sin\theta_{\text{in}} = -\frac{(h - \triangle h - x)\cos\theta_3}{\cos\frac{\alpha}{2}\cos\theta_2}\sin\theta_{\text{in}} \tag{10.18c}$$

$$= -\frac{(h - \triangle h - x)\,(\cos\alpha + \sin\alpha\tan\theta_2)}{\cos\frac{\alpha}{2}}\sin\theta_{\text{in}} \tag{10.18d}$$

As a result, the total phase of this prism dechirper/stretcher is

$$\phi_{\text{double-pass}} = 2k\left(\ell_s + n\ell_p + \ell_M\right). \tag{10.19}$$

x is picked so that the dispersed beam hits the apex of the second prism such that $\min_{\lambda\in\text{pulse spectrum}} y_1(\lambda) = 0$.

10.2.1 Group delay dispersion of prism dechirper/stretcher

From [22], the path length that determines the GDD added to the pulse is

$$P = 2\ell\cos\beta, \tag{10.20}$$

where ℓ is the length between apexes of two prisms, and β is the angle of the beam with respect to the line connecting two apexes (Fig. 10.6). The corresponding GDD is

$$\begin{aligned}\frac{\mathrm{d}^2\phi}{\mathrm{d}\omega^2} &= \frac{\lambda^3}{2\pi c^2}\frac{\mathrm{d}^2P(\lambda)}{\mathrm{d}\lambda^2}\\ &= \frac{\lambda^3}{2\pi c^2}4\ell\left\{\left[\frac{\mathrm{d}^2n}{\mathrm{d}\lambda^2} + \left(2n - \frac{1}{n^3}\right)\left(\frac{\mathrm{d}n}{\mathrm{d}\lambda}\right)^2\right]\sin\beta - 2\left(\frac{\mathrm{d}n}{\mathrm{d}\lambda}\right)^2\cos\beta\right\},\end{aligned} \tag{10.21}$$

where n is the refractive index of the prism material.

To find the value of Eq. (10.21), we need to determine ℓ and β that depend on the prism separation d, pulse bluest wavelength λ_b, and its center wavelength λ_0. In prism operations, the blue edge of the light is, in principle, put near the apex of the second prism, which leads to

$$\ell = \ell_s(\lambda_b) = d\sec\theta_4(\lambda_b) \tag{10.22a}$$

$$\beta = \theta_4(\lambda_b) - \theta_4(\lambda_0). \tag{10.22b}$$

Finally, we obtain

$$\begin{aligned}\frac{\mathrm{d}^2\phi}{\mathrm{d}\omega^2}(\lambda_0) &= \frac{\lambda_0^3}{2\pi c^2}\frac{\mathrm{d}^2P(\lambda_0)}{\mathrm{d}\lambda^2}\\ &= \frac{\lambda_0^3}{2\pi c^2}4d\sec\theta_4(\lambda_b)\\ &\quad\times\left\{\left[\frac{\mathrm{d}^2n}{\mathrm{d}\lambda^2}(\lambda_0) + \left(2n(\lambda_0) - \frac{1}{\left(n(\lambda_0)\right)^3}\right)\left(\frac{\mathrm{d}n}{\mathrm{d}\lambda}(\lambda_0)\right)^2\right]\sin\beta - 2\left(\frac{\mathrm{d}n}{\mathrm{d}\lambda}(\lambda_0)\right)^2\cos\beta\right\}.\end{aligned} \tag{10.23}$$

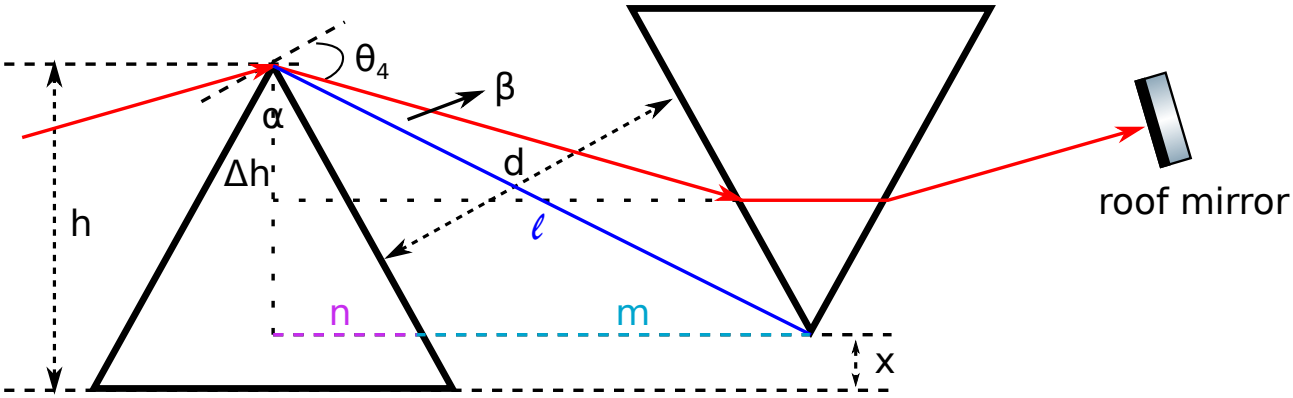


Figure 10.6: Schematic for calculating the prism GDD.

10.3 Grism type

Grating-based stretchers/dechirpers have the opposite sign of GDD and TOD, leading to more TOD after dechirping a fiber-stretched pulse. As a result, it is desirable to have a dechirper that has the same sign of GDD and TOD. Although prism dechirper meets the need, its GDD is too weak for huge dechirping. Therefore, a combination of prism and grating is proposed [27–30].

10.3.1 Configuration 1

Grism can be operated differently. Here I employ one operation that follows Fig. 10.7, where a transmission grating is attached before the right-angle prism.

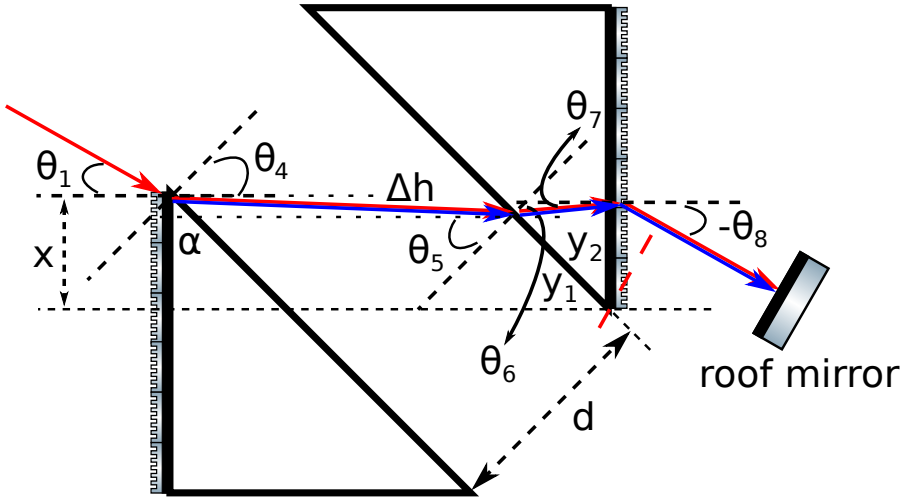


Figure 10.7: Grism dechirper/stretcher.

Similar to prism stretcher/dechirper [Eqs. (10.15)],

$$\theta_5 = \theta_4 \tag{10.24a}$$
$$\theta_6 = \theta_3 \tag{10.24b}$$
$$\theta_7 = -\theta_2 \tag{10.24c}$$
$$-\theta_8 = \theta_1 = \theta_{\text{in}}. \tag{10.24d}$$

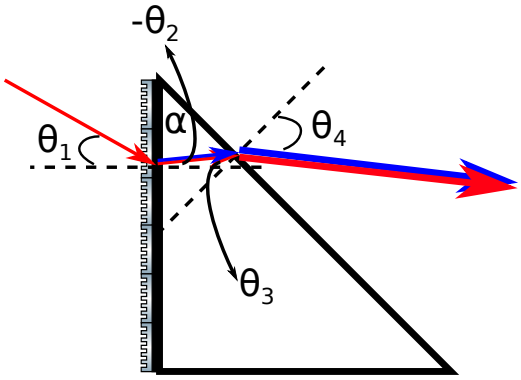


Figure 10.8: One of grism operations with a transmission grating.

Also,

$$\Lambda\left(n(\lambda)\sin\theta_2 - \sin\theta_{\text{in}}\right) = m\lambda$$
$$\Rightarrow \sin\theta_2 = \frac{1}{n(\lambda)}\left(m\frac{\lambda}{\Lambda} + \sin\theta_{\text{in}}\right) \tag{10.25a}$$

$$\theta_3 = \alpha + \theta_2 \tag{10.25b}$$

$$\sin\theta_4 = n(\lambda)\sin\theta_3 \tag{10.25c}$$

The path length between two prisms is $\ell_s = d\sec\theta_4$. The vertical translation of the beam is $\triangle h = \ell_s\sin(\theta_4 - \alpha) = d(\tan\theta_4\cos\alpha - \sin\alpha)$.

To find the path length inside the prism, we need the following relation:

$$\frac{\ell_p}{\sin\alpha} = \frac{y_1}{\sin\left(\frac{\pi}{2} - \theta_7\right)} = \frac{y_2}{\sin\left(\frac{\pi}{2} - \theta_6\right)}. \tag{10.26}$$

With $y_1 = (x - \triangle h)\sec\alpha$, we obtain

$$\ell_p = \frac{y_1\sin\alpha}{\cos\theta_7} = \frac{x - \triangle h}{\cos\theta_7}\tan\alpha \tag{10.27a}$$

$$y_2 = \frac{y_1\cos\theta_6}{\cos\theta_7} = \frac{(x - \triangle h)\cos\theta_6}{\cos\theta_7}\sec\alpha. \tag{10.27b}$$

In addition, the grating phase $\phi_g = m\frac{y_2}{\Lambda}2\pi$ needs to be considered.

In conclusion, the path lengths to consider are

$$\ell_s = d\sec\theta_4 \tag{10.28a}$$

$$\ell_p = \frac{y_1\sin\alpha}{\cos\theta_7} = \frac{x - \triangle h}{\cos\theta_2}\tan\alpha \tag{10.28b}$$

$$\ell_M = y_2\cos\left(\frac{\pi}{2} + \theta_8\right) = \frac{(x - \triangle h)\cos\theta_3}{\cos\theta_2}\sec\alpha\sin\theta_{\text{in}}$$
$$= (x - \triangle h)(\cos\alpha - \sin\alpha\tan\theta_2)\sec\alpha\sin\theta_{\text{in}} \tag{10.28c}$$

As a result, the total phase of this prism dechirper/stretcher is

$$\phi_{\text{double-pass}} = 2\left[k\left(\ell_s + n\ell_p + \ell_M\right) + \phi_g\right]. \tag{10.29}$$

Similar to the prism dechirper/stretcher, x is picked so that the dispersed beam hits the apex of the second prism such that $\min_{\lambda \in \text{pulse spectrum}} y_1(\lambda) = 0$.

10.3.2 Configuration 2

In this section, I employ another operation that follows Fig. 10.9. In contrast to Fig. 10.7, prisms are operated with minimum deviation for the pulse center wavelength as a prism dechirper/stretcher. Therefore, gratings cannot be attached to prisms.

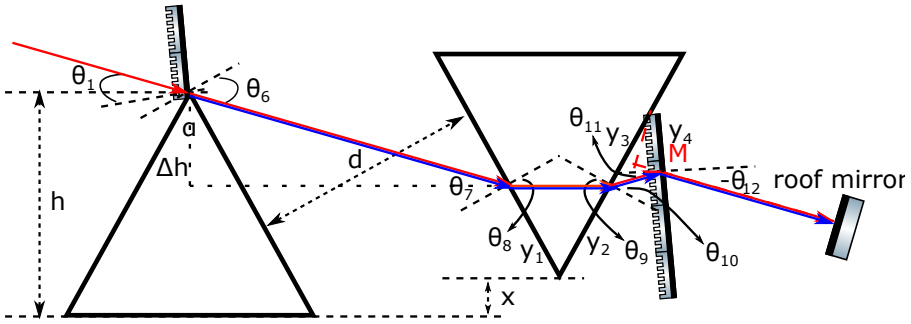


Figure 10.9: Grism dechirper/stretcher.

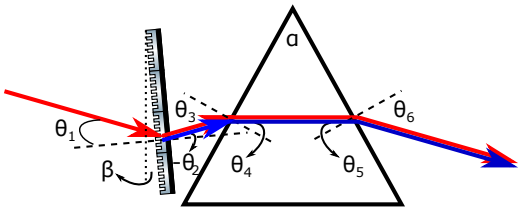


Figure 10.10: One of grism operations with a transmission grating.

Similar to prism stretcher/dechirper [Eqs. (10.15)],

$$\theta_7 = \theta_6 \tag{10.30a}$$
$$\theta_8 = \theta_5 \tag{10.30b}$$
$$\theta_9 = \theta_4 \tag{10.30c}$$
$$\theta_{10} = \theta_3 \tag{10.30d}$$
$$\theta_{11} = -\theta_2 \tag{10.30e}$$
$$-\theta_{12} = \theta_1 = \theta_{\text{in}}. \tag{10.30f}$$

Also,

$$\Lambda\left(\sin\theta_2 - \sin\theta_{\text{in}}\right) = m\lambda$$
$$\Rightarrow \sin\theta_2 = m\frac{\lambda}{\Lambda} + \sin\theta_{\text{in}} \tag{10.31a}$$

$$\theta_3 = \frac{\alpha}{2} + \beta - \theta_2 \tag{10.31b}$$

$$n(\lambda)\sin\theta_4 = \sin\theta_3 \tag{10.31c}$$

$$\theta_5 = \alpha - \theta_4 \tag{10.31d}$$
$$\sin\theta_6 = n(\lambda)\sin\theta_5 \tag{10.31e}$$

The path length between two prisms is $\ell_s = d\sec\theta_6$. The vertical translation of the beam is $\triangle h = \ell_s\cos\left(\frac{\pi}{2} - \theta_6 + \frac{\alpha}{2}\right) = \ell_s\sin\left(\theta_6 - \frac{\alpha}{2}\right) = d\left(\tan\theta_6\cos\frac{\alpha}{2} - \sin\frac{\alpha}{2}\right)$.

To find the path length inside the prism, we need the following relation:

$$\frac{\ell_p}{\sin\alpha} = \frac{y_1}{\sin\left(\frac{\pi}{2} - \theta_9\right)} = \frac{y_2}{\sin\left(\frac{\pi}{2} - \theta_8\right)}. \tag{10.32}$$

With $y_1 = (h - x - \triangle h)\sec\frac{\alpha}{2}$, we obtain

$$\ell_p = \frac{y_1\sin\alpha}{\cos\theta_9} = \frac{h - x - \triangle h}{\cos\theta_9}\sin\alpha\sec\frac{\alpha}{2} = 2\frac{h - x - \triangle h}{\cos\theta_9}\sin\frac{\alpha}{2} \tag{10.33a}$$
$$y_2 = \frac{y_1\cos\theta_8}{\cos\theta_9} = \frac{(h - x - \triangle h)\cos\theta_8}{\cos\theta_9}\sec\frac{\alpha}{2}. \tag{10.33b}$$

To find the path length between the prism and the grating, we use:

$$\frac{\ell_{pg}}{\sin\left(\frac{\alpha}{2} + \beta\right)} = \frac{y_3 - y_2}{\sin\left(\frac{\pi}{2} + \theta_{11}\right)} = \frac{y_4}{\sin\left(\frac{\pi}{2} - \theta_{10}\right)}, \tag{10.34}$$

which gives us

$$\ell_{pg} = \frac{y_3 - y_2}{\sin\left(\frac{\pi}{2} + \theta_{11}\right)}\sin\left(\frac{\alpha}{2} + \beta\right) = \frac{y_3 - y_2}{\cos\theta_{11}}\sin\left(\frac{\alpha}{2} + \beta\right) \tag{10.35a}$$
$$y_4 = \frac{y_3 - y_2}{\sin\left(\frac{\pi}{2} + \theta_{11}\right)}\sin\left(\frac{\pi}{2} - \theta_{10}\right) = \frac{y_3 - y_2}{\cos\theta_{11}}\cos\theta_{10} \tag{10.35b}$$

In addition, the grating phase $\phi_g = m\frac{-y_4}{\Lambda}2\pi$ needs to be considered.

In conclusion, the path lengths to consider are

$$\ell_s = d\sec\theta_6 \tag{10.36a}$$
$$\ell_p = \frac{y_1\sin\alpha}{\cos\theta_4} \tag{10.36b}$$
$$\ell_{pg} = \frac{y_3 - y_2}{\cos\theta_2}\sin\left(\frac{\alpha}{2} + \beta\right) \tag{10.36c}$$
$$\ell_M = -y_4\cos\left(\frac{\pi}{2} + \theta_{12}\right) = -y_4\sin\theta_{\text{in}} \tag{10.36d}$$

As a result, the total phase of this prism dechirper/stretcher is

$$\phi_{\text{double-pass}} = 2\left[k\left(\ell_s + n\ell_p + \ell_{pg} + \ell_M\right) + \phi_g\right]. \tag{10.37}$$

Similar to the prism dechirper/stretcher or the previous configuration 1 of grism, x is picked so that the dispersed beam hits the apex of the second prism such that $\min_{\lambda\in\text{pulse spectrum}} y_1(\lambda) = 0$. Besides, the second grating is placed to minimize ℓ_{pg} which is dominated by grating dechirper/stretcher introducing unwanted positive TOD, so $\min_{\lambda\in\text{pulse spectrum}}\left[y_3(\lambda) - y_2(\lambda)\right] = 0$.

Due to the requirement of minimum deviation, $\theta_4(\lambda_0) = \frac{\alpha}{2}$, leading to $\theta_3(\lambda_0) = \sin^{-1}(n(\lambda_0)\sin\frac{\alpha}{2})$. Because $\sin\theta_2(\lambda_0) = m\frac{\lambda}{\Lambda} + \sin\theta_{\text{in}}$ and Eq. (10.31b), the tilt angle of gratins β can be determined so that the diffracted beam satisfies the operation of minimum deviation for prisms.

10.4 Martinez type

Unlike the Treacy and prism types that add only negative chirp, Martinez type can add an arbitrary sign of chirp based on the relationship between the focal length f, and the distance ℓ, between the grating and the lens (Fig. 10.11). It adds positive chirp (corresponding to normal dispersion) when $\ell < f$ and negative chirp (corresponding to anomalous dispersion) when $\ell > f$; nothing happens when $\ell = f$ which is simply a 4f-telescope. It is often used as a pulse stretcher for chirped pulse amplification while a Treacy type is for latter pulse dechirper to cancel the chirp added by the Martinez stretcher.

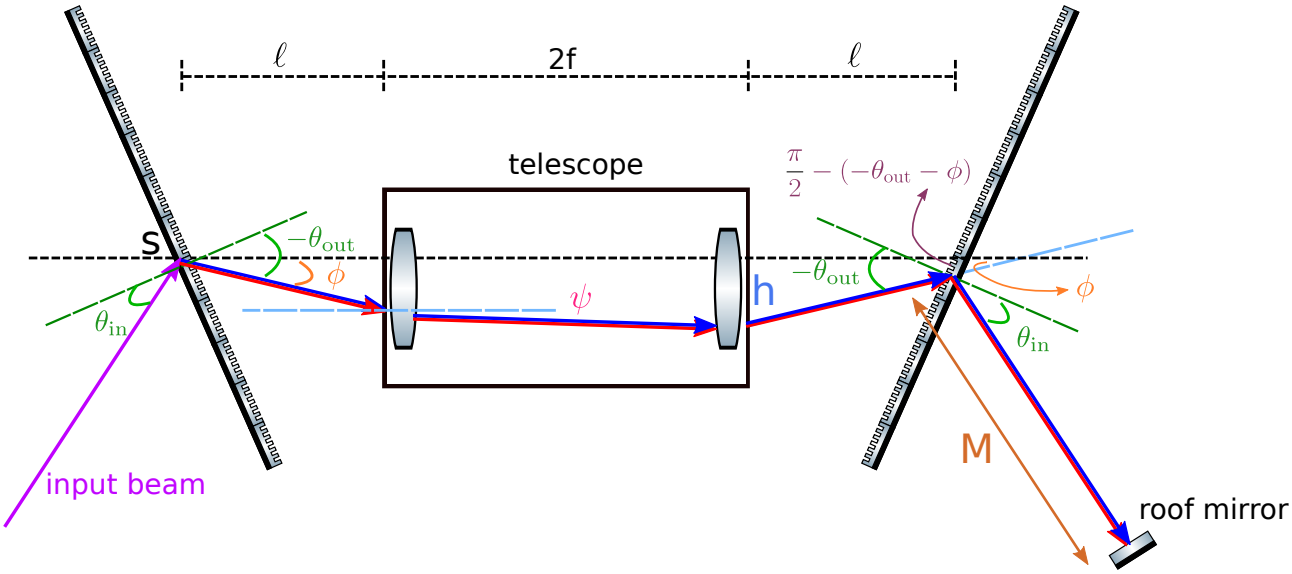


Figure 10.11: Martinez dechirper/stretcher.

There are four path lengths required in the phase calculation: ℓ_1, the first grating to the first lens; ℓ_2, the distance between two lenses; ℓ_3, the second lens to the second grating; ℓ_4, the second grating to the roof mirror for a second pass of the entire system to cancel the introduced spatial chirp after a single pass.

To calculate the first path length $\ell_1 = \ell \sec\phi$, we need to calculate the relative diffraction angle ϕ. Because the alignment of the telescope is determined by the center wavelength of the input signal, we need to first determine the "center" diffraction angle to calculate ϕ. From here, we can see that the actual value of ϕ can be quite arbitrary.

To calculate the second path length $\ell_2 = 2f \sec\psi$, we calculate ψ with ABCD matrices.

$$T_{\text{lens}} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & \ell \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & \ell \\ -\frac{1}{f} & -\frac{\ell}{f}+1 \end{bmatrix} \tag{10.38a}$$

$$T_{\text{lens}} \begin{bmatrix} 0 \\ \phi \end{bmatrix} = \begin{bmatrix} \ell\phi \\ \left(1-\frac{\ell}{f}\right)\phi \end{bmatrix} = \begin{bmatrix} \ell\tan\phi \\ \left[1-\tan^{-1}\left(\frac{\ell}{f}\right)\right]\phi \end{bmatrix}, \tag{10.38b}$$

which leads to

$$\psi = \left(1-\frac{\ell}{f}\right)\phi = \left[1-\tan^{-1}\left(\frac{\ell}{f}\right)\right]\phi, \ \tan^{-1} \text{ is more accurate} \tag{10.39}$$

Before we calculate the third path length ℓ_3, we are interested in the light passing through the telescope.

$$T = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & 2f \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & \ell \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 2f-\ell \\ 0 & -1 \end{bmatrix} \tag{10.40a}$$

$$\Rightarrow \quad T \begin{bmatrix} 0 \\ \phi \end{bmatrix} = \begin{bmatrix} (2f-\ell)\phi \\ -\phi \end{bmatrix} \tag{10.40b}$$

This shows that the light maintains the same output angle as the input angle but with a vertical offset $h = (2f-\ell)\phi = (2f-\ell)\tan\phi$.

To calculate the third path length $\ell_3 = h\csc\phi - x$, we need to calculate x (Fig. 10.12).

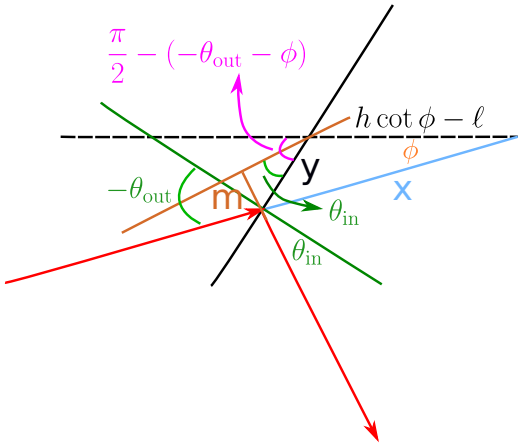


Figure 10.12: Diagram of a Martinez dechirper/stretcher to calculate the propagation length.

$$\frac{h\cot\phi - \ell}{\sin\left(\frac{\pi}{2} - (-\theta_{\text{out}} - \phi) - \phi\right)} = \frac{x}{\sin\left(\frac{\pi}{2} - (-\theta_{\text{out}} - \phi)\right)} = \frac{y}{\sin\phi}$$

$$\Rightarrow \quad \frac{h\cot\phi - \ell}{\cos\theta_{\text{out}}} = \frac{x}{\cos\left(\theta_{\text{out}} + \phi\right)} = \frac{y}{\sin\phi} \tag{10.41}$$

However, $h\cot\phi$ term can create NaN ("not a number" in MATLAB) during computation. When $\phi = 0$, $h = 0$; this term becomes $0 \times \infty$. To avoid the ambiguity, it is preferable to use $h\cot\phi = 2f - \ell$. Thus, the relation above becomes

$$\frac{2(f-\ell)}{\cos\theta_{\text{out}}} = \frac{x}{\cos\left(\theta_{\text{out}} + \phi\right)} = \frac{y}{\sin\phi}, \tag{10.42}$$

which gives

$$x = \frac{2(f-\ell)}{\cos\theta_{\text{out}}} \cos\left(\theta_{\text{out}} + \phi\right) \tag{10.43a}$$

$$y = \frac{2(f-\ell)}{\cos\theta_{\text{out}}} \sin\phi. \tag{10.43b}$$

The last path length is $\ell_4 = M - m$, where $m = y\sin\theta_{\text{in}}$.

Therefore, the single-pass path length is

$$\begin{aligned}\ell_{\text{single-pass}} &= \ell\sec\phi + 2f\sec\psi + h\csc\phi - x + M - y\sin\theta_{\text{in}} \\ &\to \ell\sec\phi + 2f\sec\psi + h\csc\phi - x - y\sin\theta_{\text{in}}.\end{aligned} \tag{10.44}$$

Similarly, there is a grating phase (Fig. 10.13)

$$\phi_g = -m\frac{y}{\Lambda}2\pi. \tag{10.45}$$

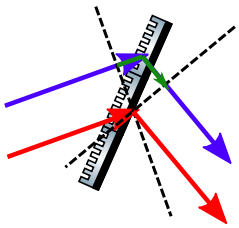


Figure 10.13: The grating phase of an Martinez-type transmissive grating dechirper/stertcher.

Besides the above obvious optical path length and grating phase, there is another phase required to be taken into account: the lens phase, or the lens-thickness optical path length. This explains why the light direction changes after it passes through a lens with Huygens principle. It adds a phase to the light incident on different positions of a lens and thus bends the light, similar to a grating. Since the light from the focus travels in collimation (having a flat phase front) after a lens, the lens path length can be calculated as

$$\text{The 1st lens: } \ell_{\text{lens 1}} = -\sqrt{h_{\text{lens 1}}^2 + f^2} = -\sqrt{(\ell\tan\phi)^2 + f^2} \tag{10.46a}$$

$$\text{The 2nd lens: } \ell_{\text{lens 2}} = -\sqrt{h_{\text{lens 2}}^2 + f^2} = -\sqrt{h^2 + f^2}. \tag{10.46b}$$

$$\ell_{\text{lens}} = \ell_{\text{lens 1}} + \ell_{\text{lens 2}} = -\sqrt{(\ell\tan\phi)^2 + f^2} - \sqrt{h^2 + f^2}. \tag{10.47}$$

Thus, the single-pass total phase is

$$\phi_{\text{single-pass}} = k\ell_{\text{single-pass}} + k\ell_{\text{lens}} + \phi_g. \tag{10.48}$$

To eliminate spatial chirp, two passes are required. Finally, the double-pass total phase is

$$\phi_{\text{double-pass}} = 2\left(k\ell_{\text{single-pass}} + k\ell_{\text{lens}} + \phi_g\right). \tag{10.49}$$

10.5 Offner type

A typical Martinez stretcher relies on a telescope (Fig. 10.11) but these refractive components can introduce aberration for broadband pulses [31]; therefore, Offner stretcher is preferred due to the use of all reflective optical components (Fig. 10.14).

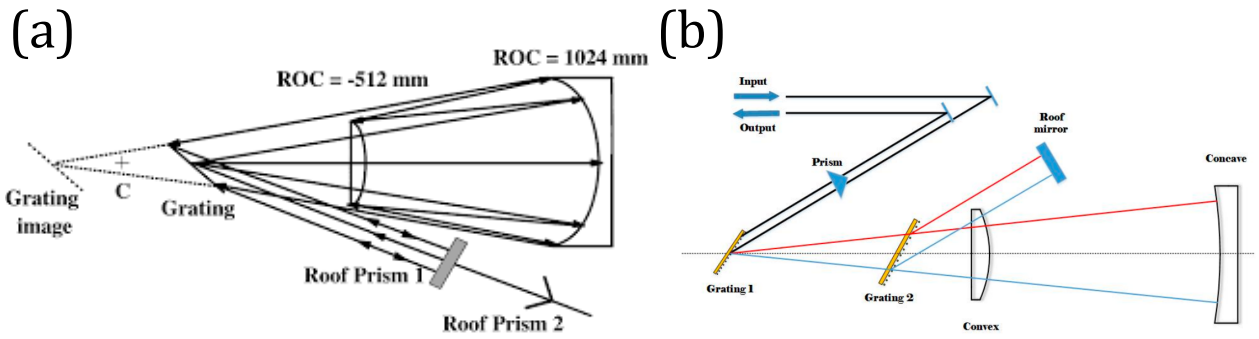


Figure 10.14: (a) Single-grating [31] and (b) double-grating reflective Offner dechirpers/stretchers [32].

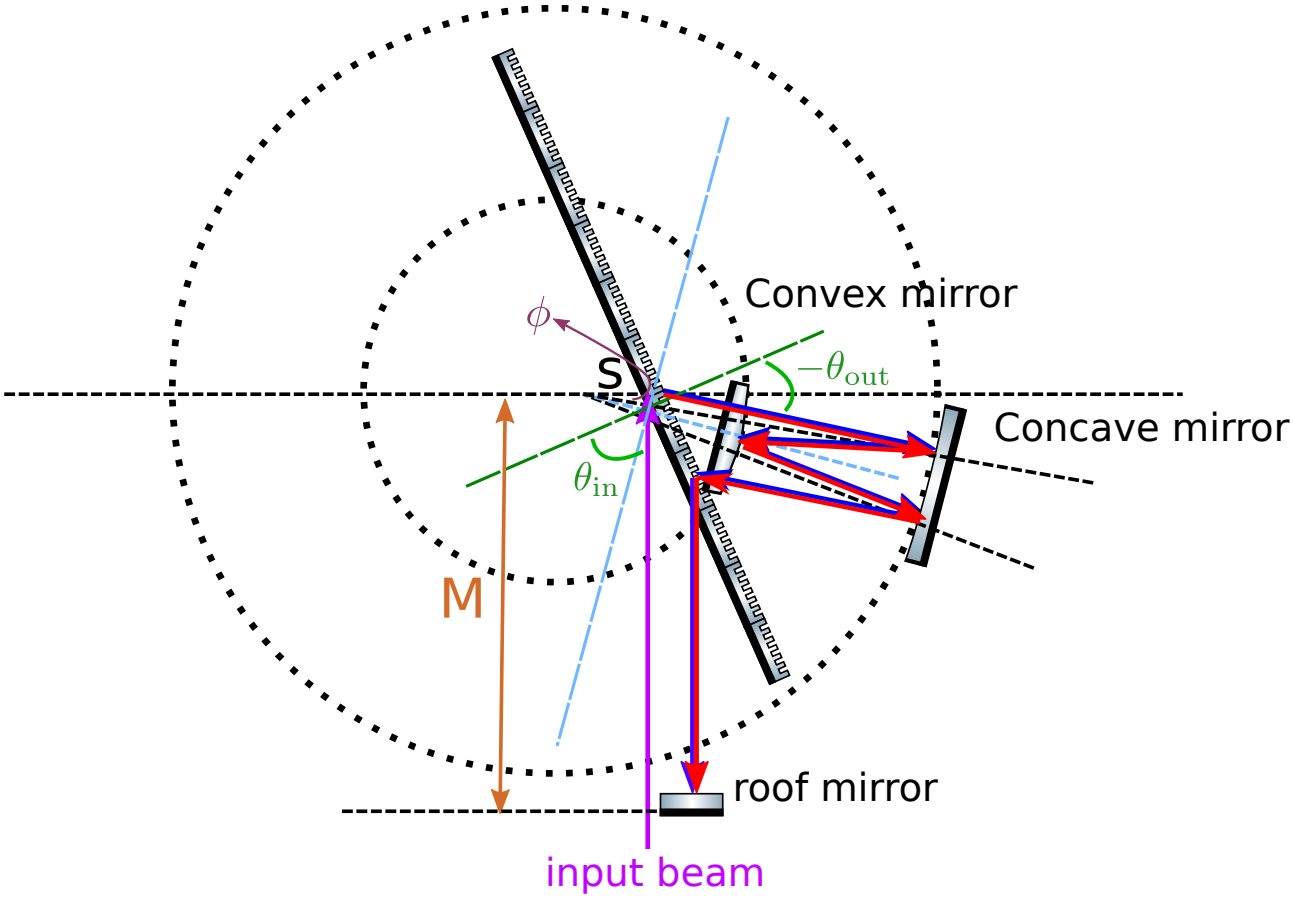


Figure 10.15: Diagram of a single-grating transmissive Offner dechirper/stretcher.

10.5.1 Transmissive single-grating Offner type

I first calculate the accumulated phase of an Offner stretcher with a single transmissive grating (Fig. 10.15). There are two concentric convex and concave mirrors, and the transmissive grating is slightly deviated from the center of circles, which introduces abberation.

Suppose the deviation of the transmissive grating from the spherical center is s, and the convex and concave radius of curvature are R and $2R$. From Fig. 10.15 and 10.16, we have

$$\Lambda\left(\sin\theta_{\text{out}} - \sin\theta_{\text{in}}\right) = m\lambda \tag{10.50a}$$

$$\frac{2R}{\sin\left(\theta_{\text{in}} - \theta_{\text{out}} - \frac{\pi}{2}\right)} = \frac{s}{\sin\theta} = \frac{\ell_1}{\sin\phi} \tag{10.50b}$$

$$\frac{R}{\sin\theta} = \frac{2R}{\sin\psi} = \frac{\ell_2}{\sin\left(\psi-\theta\right)} \tag{10.50c}$$

$$\theta + \phi = \theta_{\text{in}} - \theta_{\text{out}} - \frac{\pi}{2}. \tag{10.50d}$$

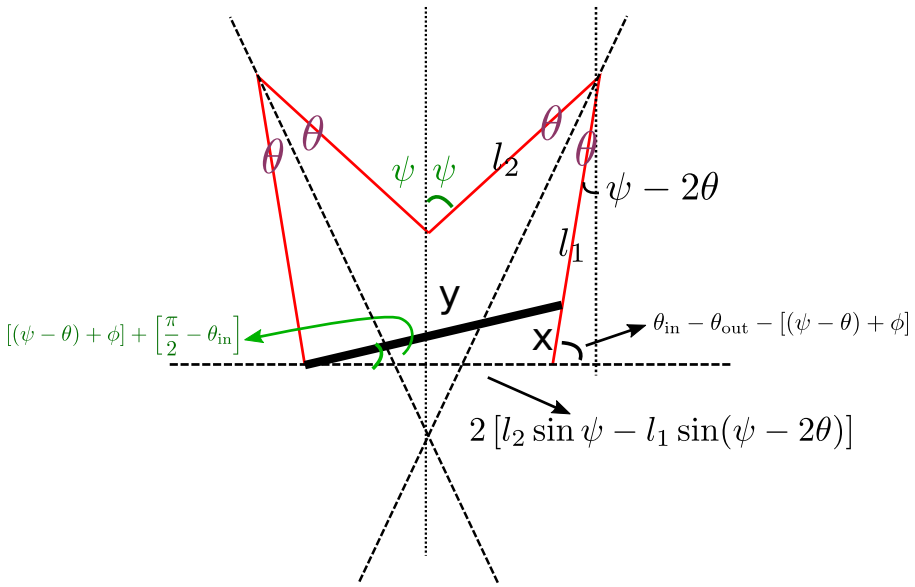


Figure 10.16: Diagram of an Offner dechirper/stretcher to calculate the propagation length.

Note that if $\theta_{\text{in}} - \theta_{\text{out}} - \frac{\pi}{2} < 0$, the diffracted beam goes to the upper-half plane, instead of going downward to the lower-half plane as in Fig. 10.15. Some angles may thus become negative.

To calculate x (Fig. 10.16), we need the following relation,

$$\frac{2\left[\ell_2\sin\psi - \ell_1\sin(\psi-2\theta)\right]}{\sin\Big(2\theta_{\text{in}} - \theta_{\text{out}} - \frac{\pi}{2} - 2\left[(\psi-\theta)+\phi\right]\Big)} = \frac{x}{\sin\Big(\left[(\psi-\theta)+\phi\right] + (\frac{\pi}{2} - \theta_{\text{in}})\Big)} \tag{10.51a}$$

$$= \frac{y}{\sin\Big(\theta_{\text{in}} - \theta_{\text{out}} - \left[(\psi-\theta)+\phi\right]\Big)}. \tag{10.51b}$$

If the deviation of the transmissive grating from the spherical center is small, different colors that go toward the roof mirror are almost parallel to the input beam. The total optical path

length thus becomes

$$\begin{aligned}\ell &= 2\left(\ell_1+\ell_2\right) - x + M - y\cos\left(\frac{\pi}{2}-\theta_{\text{in}}\right)\\ &= 2\left(\ell_1+\ell_2\right) - x + M - y\sin\theta_{\text{in}}\\ &\to 2\left(\ell_1+\ell_2\right) - x - y\sin\theta_{\text{in}}, \quad \text{after ignoring } M.\end{aligned} \tag{10.52}$$

Here, we go through some algebra. Eq. (10.50b) leads to

$$\frac{2R}{-\cos(\theta_{\text{in}}-\theta_{\text{out}})} = \frac{s}{\sin\theta} = \frac{\ell_1}{\sin\phi} \quad \Rightarrow \quad \sin\theta = -\frac{s}{2R}\cos(\theta_{\text{in}}-\theta_{\text{out}}) \tag{10.53}$$

Eq. (10.50d) leads to

$$\begin{aligned}\sin\phi = \sin\left(\theta_{\text{in}}-\theta_{\text{out}}-\frac{\pi}{2}-\theta\right) &= -\cos(\theta_{\text{in}}-\theta_{\text{out}}-\theta)\\ &= -\cos(\theta_{\text{in}}-\theta_{\text{out}})\cos\theta - \sin(\theta_{\text{in}}-\theta_{\text{out}})\sin\theta.\end{aligned} \tag{10.54}$$

With Eq. (10.54), Eq. (10.50b) leads to

$$\begin{aligned}\ell_1 &= \frac{2R}{-\cos(\theta_{\text{in}}-\theta_{\text{out}})}\sin\phi\\ &= 2R\left[\cos\theta + \tan(\theta_{\text{in}}-\theta_{\text{out}})\sin\theta\right].\end{aligned} \tag{10.55}$$

With Eq. (10.53), Eq. (10.50c) leads to

$$\sin\psi = 2\sin\theta = -\frac{s}{R}\cos(\theta_{\text{in}}-\theta_{\text{out}}). \tag{10.56}$$

With Eq. (10.56), Eq. (10.50c) leads to

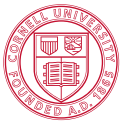
$$\begin{aligned}\ell_2 &= \frac{R}{\sin\theta}\sin(\psi-\theta)\\ &= \frac{R}{\sin\theta}\left(\sin\psi\cos\theta - \cos\psi\sin\theta\right)\\ &= \frac{R}{\sin\theta}\left(2\sin\theta\cos\theta - \cos\psi\sin\theta\right)\\ &= R\left(2\cos\theta - \cos\psi\right).\end{aligned} \tag{10.57}$$

To calculate x and y, we use Eq. (10.53) to find $(\psi-\theta)+\phi$.

$$(\psi-\theta)+\phi = (\psi-2\theta) + \left(\theta_{\text{in}}-\theta_{\text{out}}-\frac{\pi}{2}\right). \tag{10.58}$$

It is then put into Eq. (10.51).

$$\begin{aligned}&\frac{2\left[\ell_2\sin\psi - \ell_1\sin(\psi-2\theta)\right]}{-\cos\Big(2\theta_{\text{in}}-\theta_{\text{out}}-2\left[(\psi-\theta)+\phi\right]\Big)} = \frac{x}{\sin\big((\psi-2\theta)-\theta_{\text{out}}\big)} = \frac{y}{\sin\big(\frac{\pi}{2}-(\psi-2\theta)\big)}\\ \Rightarrow\quad &\frac{2\left[\ell_2\sin\psi - \ell_1\sin(\psi-2\theta)\right]}{\cos\big(\theta_{\text{out}}-2\left(\psi-2\theta\right)\big)} = \frac{x}{\sin\big((\psi-2\theta)-\theta_{\text{out}}\big)} = \frac{y}{\cos(\psi-2\theta)}.\end{aligned} \tag{10.59}$$



Finally, this leads to

$$\begin{aligned} x &= \frac{2\left[\ell_2 \sin\psi - \ell_1 \sin(\psi - 2\theta)\right]}{\cos\big(\theta_{\text{out}} - 2\left(\psi - 2\theta\right)\big)} \sin\big((\psi - 2\theta) - \theta_{\text{out}}\big) \\ &= \frac{2\left[\ell_2 \sin\psi - \ell_1 \sin(\psi - 2\theta)\right]}{\cos(\psi - 2\theta)\cot\big((\psi - 2\theta) - \theta_{\text{out}}\big) - \sin(\psi - 2\theta)} \end{aligned} \tag{10.60a}$$

$$\begin{aligned} y &= \frac{2\left[\ell_2 \sin\psi - \ell_1 \sin(\psi - 2\theta)\right]}{\cos\big(\theta_{\text{out}} - 2\left(\psi - 2\theta\right)\big)} \cos(\psi - 2\theta) \\ &= \frac{2\left[\ell_2 \sin\psi - \ell_1 \sin(\psi - 2\theta)\right]}{\cos\big((\psi - 2\theta) - \theta_{\text{out}}\big) - \tan(\psi - 2\theta)\sin\big((\psi - 2\theta) - \theta_{\text{out}}\big)} \end{aligned} . \tag{10.60b}$$

With ℓ_1, ℓ_2, x, and y, we can calculate the optical path length ℓ [Eq. (10.52)].

Recall that the grating phase needs to be considered for a total accumulated phase. Unlike Treacy type, the blue light propagates farther than the red light (Fig. 10.17). The larger the relative position y, the more grating phase needs to be added to redirect the light vertically. Thus, the single-pass total phase is

$$\phi_{\text{single-pass}} = k\ell - m\frac{y}{\Lambda}2\pi. \tag{10.61}$$

The double-pass total phase is

$$\phi_{\text{double-pass}} = 2\phi_{\text{single-pass}} = 2k\ell - 4m\pi\frac{y}{\Lambda}. \tag{10.62}$$

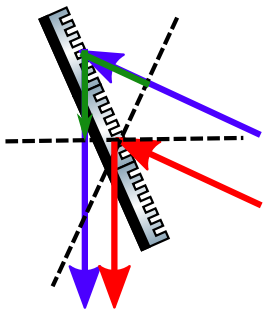


Figure 10.17: The grating phase of an Offner-type transmissive grating dechirper/stertcher.

10.5.2 Reflective single-grating Offner type

Its optical path length is similar to the transmissive one except a reflective π phase. The double-pass total phase is

$$\phi_{\text{double-pass}} = 2k\ell + 2\left(\pi - m\frac{y}{\Lambda}2\pi\right). \tag{10.63}$$

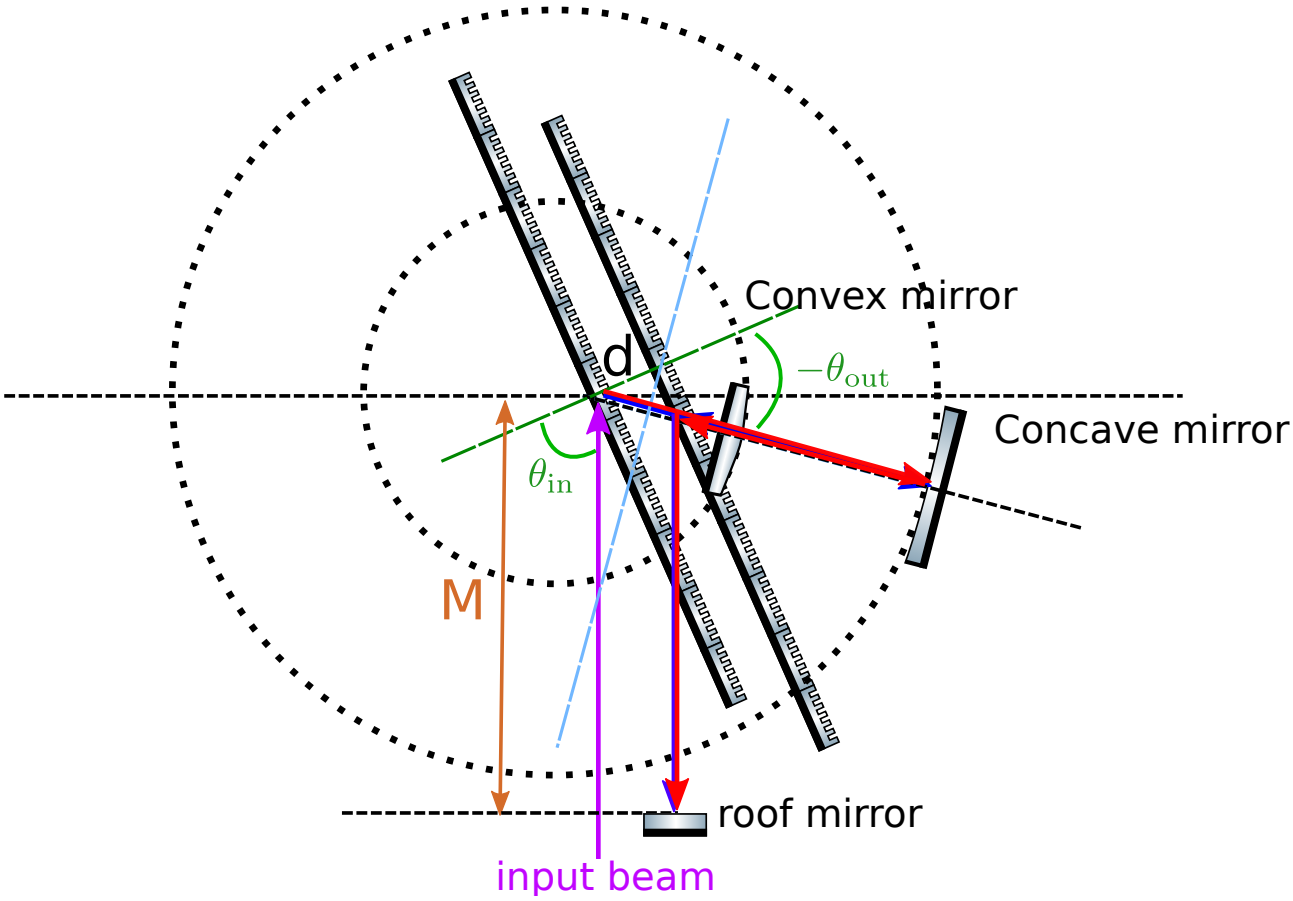


Figure 10.18: Aberration-free double-grating transmissive Offner dechirper/stretcher.

10.5.3 Aberration-free transmissive Offner type

The Offner dechirper/stretcher discussed previously (Fig. 10.15) introduces an off-center distance to mitigate the difficulty of aligning two parallel gratings; however, this introduces aberration. Here, for broadband pulses, an aberration-free design (Fig. 10.18) is preferred to avoid distortion during the dechirping process.

Assume the offset of two gratings is d, the path lengths to travel are, in order,

1. $\ell_1 = 2R$
2. $x = d\sec(-\theta_{\text{out}}) \quad \Rightarrow \quad \ell_2 = 2R - x$
3. $\phi = -\theta_{\text{out}} - \left(\frac{\pi}{2} - \theta_{\text{in}}\right) \quad \Rightarrow \quad \ell_3 = 2\left(M - x\sin\phi\right)$
4. ℓ_2
5. ℓ_1

The single-pass optical path length ℓ is

$$\begin{aligned} \ell &= 2\left(\ell_1 + \ell_2\right) + \ell_3 \\ &= 8R - 2x + 2M - 2x\sin\phi \\ &\to -2x\left[1 + \sin\phi\right] \end{aligned} \tag{10.64}$$

Thus, the single-pass phase, including the grating phase, is

$$\phi_{\text{single-pass}} = k\ell - m\frac{d\tan(-\theta_{\text{out}})}{\Lambda}2\pi \tag{10.65}$$

The double-pass total phase is

$$\phi_{\text{double-pass}} = 2\phi_{\text{single-pass}} = 2k\ell - 4m\pi\frac{d\tan(-\theta_{\text{out}})}{\Lambda}. \tag{10.66}$$

Fig. 10.19 shows how a real Offner dechirper/stretcher is aligned.

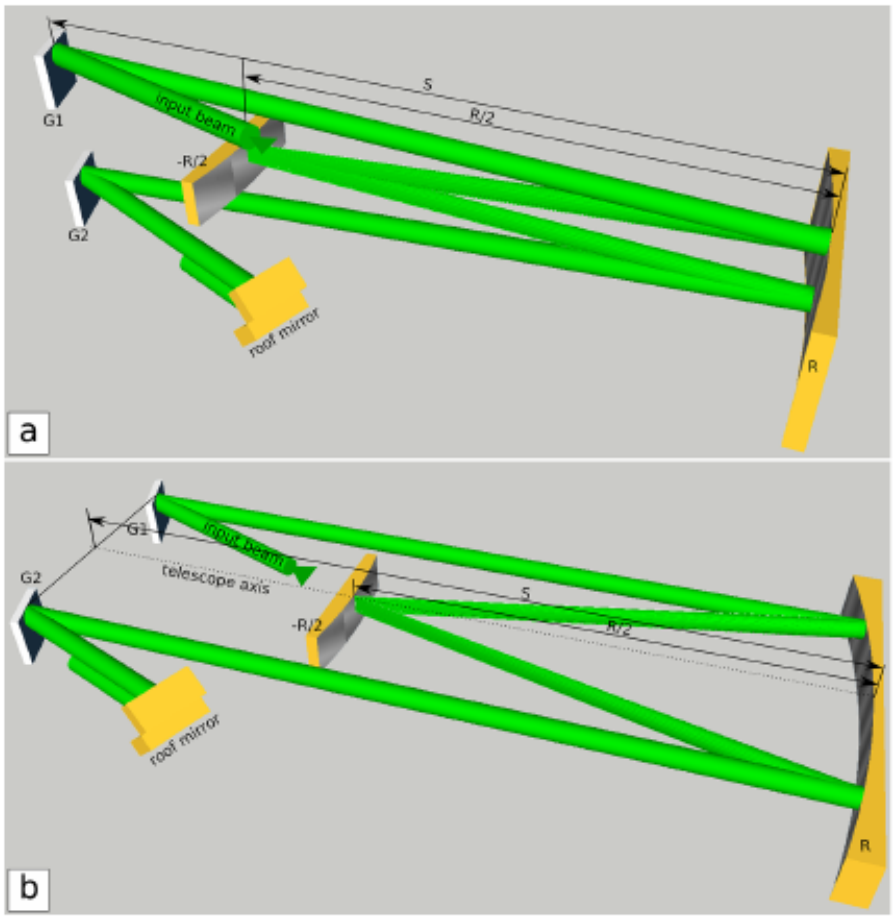


Figure 10.19: Two real configurations of an aberration-free double-grating Offner dechirper/stretcher [33].

Chapter 11

Derivation

In this chapter, I show the derivation of 1D-UPPE [Eq. (1.1)]. Although it has been shown in the paper [3], I realized that there are a few mistakes in equations. Filing an errata might be a future option, but it's good for me to have one document that I can easily edit and update.

In fiber optics, light propagates in an isotropic dielectric medium, or gas if it is a hollow-core fiber, with its waveguide boundary conditions. Since there are no charge and current sources, such electromagnetic fields are governed by the following Maxwell's equations:

$$\nabla \times \vec{\mathbb{E}}(\vec{x},t) = -\mu_0 \partial_t \vec{\mathbb{H}}(\vec{x},t) \tag{11.1a}$$

$$\nabla \times \vec{\mathbb{H}}(\vec{x},t) = \epsilon_0 \partial_t \left[\epsilon_r(\vec{x},t) * \vec{\mathbb{E}}(\vec{x},t)\right] + \partial_t \vec{\mathbb{P}}(\vec{x},t), \tag{11.1b}$$

where $\vec{\mathbb{E}}$ and $\vec{\mathbb{H}}$ are electric and magnetic fields, respectively; $\vec{\mathbb{P}}$ is the induced nonlinear polarization; ϵ_r is the relative dielectric constant of the medium; ϵ_0 and μ_0 are permittivity and permeability in vacuum. In "linear" light propagation when the nonlinear polarization $\vec{\mathbb{P}}$ is negligible, Maxwell's equations [Eqs. (11.1)] with the waveguide boundary conditions lead to the discrete fiber eigenmodes with propagation constants $\beta_n(\omega)$, along with their corresponding electric field $\vec{F}_n(x,y,\omega)\mathrm{e}^{i\beta_n z}$ and magnetic field $\vec{G}_n(x,y,\omega)\mathrm{e}^{i\beta_n z}$ with an orthogonality relation [34]:

$$\frac{1}{4}\int \mathrm{d}x\,\mathrm{d}y \left(\vec{F}_m^* \times \vec{G}_n + \vec{F}_n \times \vec{G}_m^*\right)\cdot \hat{z} = \delta_{mn} N_n^2(\omega). \tag{11.2}$$

In situations where the nonlinearity only acts as a perturbation to the electromagnetic fields, eigenmode expansion can be applied to the electric and magnetic fields:

$$\begin{aligned}
\vec{\mathbb{E}}(\vec{x},t) &= \frac{1}{2}\left[\vec{\mathcal{E}}(\vec{x},t) + \text{c.c.}\right] \quad , \vec{\mathcal{E}} \text{ is the analytic signal of } \vec{\mathbb{E}} \\
&= \sum_p \int \mathrm{d}\omega \frac{1}{2}\left\{\frac{\vec{F}_p(x,y,\omega)}{N_p(\omega)} A_p(z,\omega)\mathrm{e}^{i\left[\beta_p(\omega)z-\omega t\right]} + \text{c.c.}\right\} \qquad (11.3a) \\
\vec{\mathbb{H}}(\vec{x},t) &= \frac{1}{2}\left[\vec{\mathcal{H}}(\vec{x},t) + \text{c.c.}\right] \quad , \vec{\mathcal{H}} \text{ is the analytic signal of } \vec{\mathbb{H}} \\
&= \sum_p \int \mathrm{d}\omega \frac{1}{2}\left\{\frac{\vec{G}_p(x,y,\omega)}{N_p(\omega)} A_p(z,\omega)\mathrm{e}^{i\left[\beta_p(\omega)z-\omega t\right]} + \text{c.c.}\right\}; \qquad (11.3b)
\end{aligned}$$

Assume $\vec{F}_p(x,y,\omega) = \vec{F}_p(x,y)$ and $\vec{G}_p(x,y,\omega) = \vec{G}_p(x,y)$ are independent of frequency,

$$\begin{aligned}
\vec{\mathbb{E}}(\vec{x},t) &= \sum_p \frac{1}{2}\left\{\frac{\vec{F}_p(x,y)}{N_p}\left[A_p(z,t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\right] + \text{c.c.}\right\} \\
&= \sum_p \frac{1}{2}\left\{\frac{\vec{F}_p(x,y)}{\sqrt{\frac{\epsilon_0 n_{p,\text{eff}} c}{2}}}\left[A_p(z,t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\right] + \text{c.c.}\right\}, \qquad (11.4a) \\
\vec{\mathbb{H}}(\vec{x},t) &= \sum_p \frac{1}{2}\left\{\frac{\vec{G}_p(x,y)}{N_p}\left[A_p(z,t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\right] + \text{c.c.}\right\} \\
&= \sum_p \frac{1}{2}\left\{\frac{\vec{G}_p(x,y)}{\sqrt{\frac{\epsilon_0 n_{p,\text{eff}} c}{2}}}\left[A_p(z,t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\right] + \text{c.c.}\right\}, \qquad (11.4b)
\end{aligned}$$

where $\vec{\mathbb{E}}(\vec{x},t)$ and $\vec{\mathbb{H}}(\vec{x},t)$ are the real-valued electric and magnetic fields; "c.c." stands for complex conjugate, p is the eigenmode index ($p \in \mathbb{N}$), $\vec{F}_p(x,y)$ is the normalized spatial mode profile (with the unit of 1/m) of the eigenmode with the normalization condition, $\int \left|\vec{F}_p\right|^2 \mathrm{d}^2x = 1$, and is assumed to be independent of frequency. $\vec{\mathbb{E}}(\vec{x},t)$ has the unit of V/m. $A_p(z,t)$ represents the envelope of the electric field and is normalized to have the unit of $\sqrt{\text{W}}$ with the normalization constant $N_p = \sqrt{\frac{\epsilon_0 n_{p,\text{eff}} c}{2}}$ (see the details in Section 11.0.1). $\beta_p(\omega) = n_{p,\text{eff}}(\omega)k_0$ is the propagation constant of the pth eigenmode. $\beta_{(0)}$ and ω_0 are two free parameters, usually chosen as the propagation constant and the center angular frequency of a pulse. A_p has the following time-frequency relation,

$$A_p(z,t) = \int \mathrm{d}\omega A_p(z,\omega)\mathrm{e}^{i\left[\left(\beta_p(\omega)-\beta_{(0)}\right)z-(\omega-\omega_0)t\right]}. \qquad (11.5)$$

Because analytic signal contains only the positive-frequency part and its complex conjugate contains only the negative-frequency part [35], analytic signal is sufficient to solve for the Maxwell's equations [Eqs. (11.1)]. By assuming that $\vec{\mathbb{H}}$, $\vec{\mathbb{J}}$, and $\vec{\mathbb{P}}$ have similar forms of analytic-signal expansion, with Fourier transform, Eqs. (11.1) become

$$\begin{aligned}
\mu_0 i\omega\vec{\mathcal{H}} &= \nabla\times\vec{\mathcal{E}} \qquad (11.6a) \\
-i\omega\vec{\mathcal{P}} - i\omega\epsilon_0\epsilon_r\vec{\mathcal{E}} &= \nabla\times\vec{\mathcal{H}}, \qquad (11.6b)
\end{aligned}$$

To solve Eqs. (11.6), we need to find the representations of $\vec{\mathcal{E}}$ and $\vec{\mathcal{H}}$ with respect to A_p. Because

$$\begin{aligned}
\vec{\mathcal{E}}(\vec{x},t) &= \sum_p \int \mathrm{d}\omega\ \frac{\vec{F}_p}{N_p}A_p\mathrm{e}^{i(\beta_p z-\omega t)} \\
&= C_{\mathfrak{I}\mathfrak{F}}\int \mathrm{d}\omega\ \vec{\mathcal{E}}(\vec{x},\omega)\mathrm{e}^{-i\omega t}\text{: inverse Fourier transform}, \qquad (11.7)
\end{aligned}$$

$\vec{\mathcal{E}}(\vec{x},\omega)$ has the following eigenmode-expansion relation, which is the same for $\vec{\mathcal{H}}(\vec{x},\omega)$:

$$\vec{\mathcal{E}}(\vec{x},\omega) = \frac{1}{C_{\mathfrak{IF}}}\sum_p \frac{\vec{F}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} \tag{11.8a}$$

$$\vec{\mathcal{H}}(\vec{x},\omega) = \frac{1}{C_{\mathfrak{IF}}}\sum_p \frac{\vec{G}_p}{N_p} A_p \mathrm{e}^{i\beta_p z}. \tag{11.8b}$$

With Eqs. (11.8), we can reduce Maxwell's equations [Eq. (11.6)] to the one with A_p so that we can solve for the nonlinear evolution of A_p. First, we expand Eqs. (11.6) with Eqs. (11.8):

$$\begin{aligned}
&\begin{cases}
\frac{1}{C_{\mathfrak{IF}}} i\mu_0\omega \sum_p \frac{\vec{G}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} &= \frac{1}{C_{\mathfrak{IF}}} \nabla \times \sum_p \frac{\vec{F}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} \\
-i\omega\vec{\mathcal{P}} - \frac{1}{C_{\mathfrak{IF}}} i\omega\epsilon_0\epsilon_r \sum_p \frac{\vec{F}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} &= \frac{1}{C_{\mathfrak{IF}}} \nabla \times \sum_p \frac{\vec{G}_p}{N_p} A_p \mathrm{e}^{i\beta_p z},
\end{cases} \\
\Rightarrow &\begin{cases}
\frac{1}{C_{\mathfrak{IF}}} i\mu_0\omega \sum_p \frac{\vec{G}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} &= \frac{1}{C_{\mathfrak{IF}}} \sum_p \left[A_p \left(\nabla \times \frac{\vec{F}_p \mathrm{e}^{i\beta_p z}}{N_p} \right) + \left(\nabla A_p\right) \times \frac{\vec{F}_p \mathrm{e}^{i\beta_p z}}{N_p} \right] \\
-i\omega\vec{\mathcal{P}} - \frac{1}{C_{\mathfrak{IF}}} i\omega\epsilon_0\epsilon_r \sum_p \frac{\vec{F}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} &= \frac{1}{C_{\mathfrak{IF}}} \sum_p \left[A_p \left(\nabla \times \frac{\vec{G}_p \mathrm{e}^{i\beta_p z}}{N_p} \right) + \left(\nabla A_p\right) \times \frac{\vec{G}_p \mathrm{e}^{i\beta_p z}}{N_p} \right]
\end{cases} \\
\Rightarrow &\begin{cases}
\frac{1}{C_{\mathfrak{IF}}} i\mu_0\omega \sum_p \frac{\vec{G}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} &= \frac{1}{C_{\mathfrak{IF}}} \sum_p \left[A_p \left(\nabla \times \frac{\vec{F}_p \mathrm{e}^{i\beta_p z}}{N_p} \right) + \left(\partial_z A_p\right) \hat{z} \times \frac{\vec{F}_p \mathrm{e}^{i\beta_p z}}{N_p} \right] \\
-i\omega\vec{\mathcal{P}} - \frac{1}{C_{\mathfrak{IF}}} i\omega\epsilon_0\epsilon_r \sum_p \frac{\vec{F}_p}{N_p} A_p \mathrm{e}^{i\beta_p z} &= \frac{1}{C_{\mathfrak{IF}}} \sum_p \left[A_p \left(\nabla \times \frac{\vec{G}_p \mathrm{e}^{i\beta_p z}}{N_p} \right) + \left(\partial_z A_p\right) \hat{z} \times \frac{\vec{G}_p \mathrm{e}^{i\beta_p z}}{N_p} \right]
\end{cases}.
\end{aligned} \tag{11.9}$$

Since eigenmode fields satisfy the linear Maxwell's equations [$\vec{\mathcal{P}} = 0$ in Eq. (11.6)]:

$$\mu_0 i\omega \left(\vec{G}_p \mathrm{e}^{i\beta_p z} \right) = \nabla \times \left(\vec{F}_p \mathrm{e}^{i\beta_p z} \right) \tag{11.10a}$$

$$-i\omega\epsilon_0\epsilon_r \left(\vec{F}_p \mathrm{e}^{i\beta_p z} \right) = \nabla \times \left(\vec{G}_p \mathrm{e}^{i\beta_p z} \right), \tag{11.10b}$$

Eqs. (11.9) can be simplified to

$$0 = \frac{1}{C_{\mathfrak{IF}}} \sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \hat{z} \times \frac{\vec{F}_p}{N_p} \tag{11.11a}$$

$$-i\omega\vec{\mathcal{P}} = \frac{1}{C_{\mathfrak{IF}}} \sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \hat{z} \times \frac{\vec{G}_p}{N_p}. \tag{11.11b}$$

Multiplying the top and bottom equations in Eqs. (11.11) with $\vec{G}_m^*$ and $\vec{F}_m^*$, respectively, gives

$$\begin{cases} 0 & = \dfrac{1}{C_{\mathfrak{IF}}} \displaystyle\sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \vec{G}_m^* \cdot \left(\hat{z} \times \dfrac{\vec{F}_p}{N_p}\right) \\ -i\omega \vec{F}_m^* \cdot \vec{\mathcal{P}} & = \dfrac{1}{C_{\mathfrak{IF}}} \displaystyle\sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \vec{F}_m^* \cdot \left(\hat{z} \times \dfrac{\vec{G}_p}{N_p}\right) \end{cases}$$

$$\Rightarrow \begin{cases} 0 & = \dfrac{1}{C_{\mathfrak{IF}}} \displaystyle\sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \hat{z} \cdot \left(\dfrac{\vec{F}_p}{N_p} \times \vec{G}_m^*\right) \\ -i\omega \vec{F}_m^* \cdot \vec{\mathcal{P}} & = \dfrac{1}{C_{\mathfrak{IF}}} \displaystyle\sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \hat{z} \cdot \left(\dfrac{\vec{G}_p}{N_p} \times \vec{F}_m^*\right) \end{cases}$$

$$\begin{aligned} \Rightarrow -i\omega \vec{F}_m^* \cdot \vec{\mathcal{P}} &= \frac{1}{C_{\mathfrak{IF}}} \sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \hat{z} \cdot \left(\frac{\vec{F}_p}{N_p} \times \vec{G}_m^* + \frac{\vec{G}_p}{N_p} \times \vec{F}_m^*\right) \\ &= -\frac{1}{C_{\mathfrak{IF}}} \sum_p \mathrm{e}^{i\beta_p z} \left(\partial_z A_p\right) \frac{1}{N_p} \hat{z} \cdot \left(\vec{G}_m^* \times \vec{F}_p + \vec{F}_m^* \times \vec{G}_p\right). \end{aligned} \tag{11.12}$$

Next, apply the normalization condition [Eq. (11.2)].

$$\partial_z A_p(z,\omega) = C_{\mathfrak{IF}} \frac{i\omega}{4N_p(\omega)} \mathrm{e}^{-i\beta_p(\omega) z} \int \vec{F}_p^* \cdot \vec{\mathcal{P}}(\vec{x},\omega)\, \mathrm{d}x\, \mathrm{d}y. \tag{11.13}$$

It is preferable to solve Eq. (11.13) with the envelope of the polarization, $\vec{P}(\vec{x},t)$.

$$\begin{aligned} \vec{\mathbb{P}}(\vec{x},t) &= \frac{1}{2}\left[\vec{\mathcal{P}}(\vec{x},t) + \text{c.c.}\right] \\ &= \frac{1}{2}\left[\vec{P}(\vec{x},t)\mathrm{e}^{i\left(\beta_{(0)} z - \omega_0 t\right)} + \text{c.c.}\right] \\ &= C_{\mathfrak{IF}} \int \mathrm{d}\omega \frac{1}{2}\left[\vec{\mathcal{P}}(\vec{x},\omega)\mathrm{e}^{-i\omega t} + \text{c.c.}\right] : \text{inverse Fourier transform.} \end{aligned} \tag{11.14}$$

By using the relations $2\pi\delta(\omega - \omega') = \int \mathrm{d}\omega \mathrm{e}^{-i\left(\omega-\omega'\right)t}$ and $C_{\mathfrak{F}} = 1/\left(2\pi C_{\mathfrak{IF}}\right)$,

$$\vec{\mathcal{P}}(\vec{x},\omega) = C_{\mathfrak{F}} \int \mathrm{d}t \vec{P}(\vec{x},t) \mathrm{e}^{-i\left[-\beta_{(0)} z - (\omega-\omega_0)t\right]}. \tag{11.15}$$

Next, applying Eq. (11.15) to Eq. (11.13) gives us

$$\partial_z A_p(z,\omega) = \frac{1}{2\pi} \frac{i\omega}{4N_p(\omega)} \int \mathrm{d}x\, \mathrm{d}y \vec{F}_p^* \cdot \int \mathrm{d}t \vec{P}(\vec{x},t) \mathrm{e}^{-i\left[\left(\beta_p(\omega) - \beta_{(0)}\right) z - (\omega-\omega_0)t\right]}. \tag{11.16}$$

Temporal evolution

To solve for $A_p(z,t)$ in the time domain, rather than $A_p(z,\omega)$, take the derivative, with respect to z, of Eq. (11.5):

$$\partial_z A_p(z,t) = \int \mathrm{d}\omega \left[\partial_z A_p(z,\omega) + i\left(\beta_p(\omega) - \beta_{(0)}\right) A_p(z,\omega)\right] \mathrm{e}^{i\left[\left(\beta_p(\omega) - \beta_{(0)}\right) z - (\omega-\omega_0)t\right]}. \tag{11.17}$$

Dispersion

The second term on the right-hand side of Eq. (11.17) is the dispersion by noting that

$$\begin{aligned}
&\int i\beta_p(\omega)A_p(z,\omega)\mathrm{e}^{i\left[\left(\beta_p(\omega)-\beta_{(0)}\right)z-(\omega-\omega_0)t\right]}\,\mathrm{d}\omega \\
&= \int i\left[\beta_p^{(0)}(\omega_0)+(\omega-\omega_0)\,\beta_p^{(1)}(\omega_0)+\cdots\right]A_p(z,\omega)\mathrm{e}^{i\left[\left(\beta_p(\omega)-\beta_{(0)}\right)z-(\omega-\omega_0)t\right]}\,\mathrm{d}\omega \\
&= i\left[\beta_p^{(0)}(\omega_0)+\beta_p^{(1)}(\omega_0)\,(i\partial_t)+\frac{\beta_p^{(2)}(\omega_0)}{2}\,(i\partial_t)^2+\cdots\right]A_p(z,t) \\
&= i\sum_m \frac{(i\partial_t)^m}{m!}\beta_p^{(m)}(\omega_0)A_p(z,t). \qquad (11.18)
\end{aligned}$$

Kerr and Raman nonlinearities

For the first term on the right-hand side of Eq. (11.17),

$$\begin{aligned}
&\int \mathrm{d}\omega\;\partial_z A_p(z,\omega)\mathrm{e}^{i\left[\left(\beta_p(\omega)-\beta_{(0)}\right)z-(\omega-\omega_0)t\right]} \\
&\qquad = \frac{1}{2\pi}\int \mathrm{d}\omega\frac{i\omega}{4N_p(\omega)}\int \mathrm{d}x\,\mathrm{d}y\vec{F}_p^*\cdot\int \mathrm{d}t'\,\vec{P}(\vec{x},t')\mathrm{e}^{-i(\omega-\omega_0)\left(t-t'\right)}. \qquad (11.19)
\end{aligned}$$

Note that, in general, the nonlinear polarization has the following form [36]:

$$\vec{\mathbb{P}}(t)=\int_{-\infty}^{\infty}\chi^{(3)}(t_1,t_2,t_3)\vdots\vec{\mathbb{E}}(t-t_1)\vec{\mathbb{E}}(t-t_2)\vec{\mathbb{E}}(t-t_3)\,\mathrm{d}t_1\,\mathrm{d}t_2\,\mathrm{d}t_3, \qquad (11.20)$$

where

$$\chi^{(3)}(t_1,t_2,t_3)=\delta(t_1)\delta(t_2-t_3)\hat{R}^{ijk\ell}(t_3) \qquad (11.21a)$$

$$\hat{R}^{ijk\ell}(t)=\epsilon_0\chi^{(3)}_{\text{electronic}}\frac{\delta^{ij}\delta^{k\ell}+\delta^{ik}\delta^{j\ell}+\delta^{i\ell}\delta^{jk}}{3}\delta(t)+\mathtt{R}_a(t)\delta^{ij}\delta^{k\ell}+\mathtt{R}_b(t)\frac{\delta^{ik}\delta^{j\ell}+\delta^{i\ell}\delta^{jk}}{2} \qquad (11.21b)$$

includes the instantaneous Kerr nonlinearity and the delayed isotropic ($\mathtt{R}_a$) and anisotropic ($\mathtt{R}_b$) Raman nonlinearities; therefore,

$$\mathbb{P}^i(t)=\sum_{jk\ell}\mathbb{E}^j(t)\int_{-\infty}^{\infty}\hat{R}^{ijk\ell}(\tau)\mathbb{E}^k(t-\tau)\mathbb{E}^\ell(t-\tau)\,\mathrm{d}\tau, \qquad \text{by applying } (t_3\to\tau). \qquad (11.22)$$

We introduce the envelopes, $\vec{E}$ and $\vec{P}$, to solve for the nonlinear term.

$$\vec{\mathbb{E}}(t)=\frac{1}{2}\left[\vec{E}(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right] \qquad (11.23a)$$

$$\vec{\mathbb{P}}(t)=\frac{1}{2}\left[\vec{P}(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right]. \qquad (11.23b)$$

First, we solve for the Kerr nonlinearity:

$$\begin{aligned}\mathbb{P}^i(t) &= \sum_{jkl} \epsilon_0 \frac{1}{2}\left[E^j(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right]\chi^{(3)}_{\text{electronic}}\frac{\delta^{ij}\delta^{k\ell}+\delta^{ik}\delta^{j\ell}+\delta^{i\ell}\delta^{jk}}{3} \\ &\qquad \times \frac{1}{2}\left[E^k(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right]\frac{1}{2}\left[E^\ell(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right] \\ &= \frac{\epsilon_0\chi^{(3)}_{\text{electronic}}}{8}\sum_k \Bigg\{ \left(E^i(E^k)^2\mathrm{e}^{3i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right) \\ &\qquad + \left[\left((E^i)^*(E^k)^2+2E^i\left|E^k\right|^2\right)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right]\Bigg\}. \end{aligned} \tag{11.24}$$

Next, we solve for the isotropic Raman term:

$$\begin{aligned}\mathbb{P}^i(t) &= \frac{1}{2}\left[E^i(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right]\int \mathtt{R}_a(\tau)\sum_k\left\{\frac{1}{2}\left[E^k(t-\tau)\mathrm{e}^{i\left[\beta_{(0)}z-\omega_0(t-\tau)\right]}+\text{c.c.}\right]\right\}^2 \\ &= \frac{1}{8}\sum_k\Bigg\{\left[E^i(t)\mathrm{e}^{3i\left(\beta_{(0)}z-\omega_0 t\right)}\int \mathtt{R}_a(\tau)\left(E^k(t-\tau)\right)^2\mathrm{e}^{2i\omega_0\tau}\,\mathrm{d}\tau+\text{c.c.}\right] \\ &\qquad + \left[2E^i(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\int \mathtt{R}_a(\tau)\left|E^k(t-\tau)\right|^2\mathrm{d}\tau+\text{c.c.}\right] \\ &\qquad + \left[\left(E^i(t)\right)^*\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\int \mathtt{R}_a(\tau)\left(E^k(t-\tau)\right)^2\mathrm{e}^{2i\omega_0\tau}\,\mathrm{d}\tau+\text{c.c.}\right]\Bigg\},\end{aligned} \tag{11.25}$$

and the anisotropic term:

$$\begin{aligned}\mathbb{P}^i(t) &= \sum_j \mathbb{E}^j(t)\int \mathtt{R}_b(\tau)\mathbb{E}^i(t-\tau)\mathbb{E}^j(t-\tau)\,\mathrm{d}\tau \\ &= \sum_j \frac{1}{2}\left[E^j(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}+\text{c.c.}\right] \\ &\qquad \int \mathtt{R}_b(\tau)\frac{1}{2}\left\{E^i(t-\tau)\mathrm{e}^{i\left[\beta_{(0)}z-\omega_0(t-\tau)\right]}+\text{c.c.}\right\}\frac{1}{2}\left\{E^j(t-\tau)\mathrm{e}^{i\left[\beta_{(0)}z-\omega_0(t-\tau)\right]}+\text{c.c.}\right\}\mathrm{d}\tau \\ &= \frac{1}{8}\sum_j\Bigg\{\left[E^j(t)\mathrm{e}^{3i\left(\beta_{(0)}z-\omega_0 t\right)}\int \mathtt{R}_b(\tau)E^i(t-\tau)E^j(t-\tau)\mathrm{e}^{2i\omega_0\tau}+\text{c.c.}\right] \\ &\qquad + \left[E^j(t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\int \mathtt{R}_b(\tau)\left(E^i(E^j)^*+(E^i)^*E^j\right)(t-\tau)\,\mathrm{d}\tau+\text{c.c.}\right] \\ &\qquad + \left[(E^j)^*\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\int \mathtt{R}_b(\tau)E^i(t-\tau)E^j(t-\tau)\mathrm{e}^{2i\omega_0\tau}\,\mathrm{d}\tau+\text{c.c.}\right]\Bigg\}.\end{aligned} \tag{11.26}$$

By ignoring the 3rd harmonic terms, which are terms with $\mathrm{e}^{3i\left(\beta_{(0)}z-\omega_0 t\right)}$, the polarization can be

expressed, in terms of its envelope, as

$$
\begin{aligned}
P^i = \frac{1}{4}\sum_j \Bigg\{ & \epsilon_0 \chi^{(3)}_{\text{electronic}} \left[(E^i)^* (E^j)^2 + 2E^i \left| E^j \right|^2 \right] \\
& + 2E^i(t) \int \mathtt{R}_a(\tau) \left| E^j(t-\tau) \right|^2 \mathrm{d}\tau + (E^i)^*(t) \int \mathtt{R}_a(\tau) (E^j)^2 (t-\tau) \mathrm{e}^{2i\omega_0\tau} \, \mathrm{d}\tau \\
& + E^j(t) \int \mathtt{R}_b(\tau) \left(E^i (E^j)^* + (E^i)^* E^j \right) (t-\tau) \, \mathrm{d}\tau \\
& \qquad + (E^j(t))^* \int \mathtt{R}_b(\tau) \left(E^i E^j \right) (t-\tau) \mathrm{e}^{2i\omega_0\tau} \, \mathrm{d}\tau \Bigg\} \quad (11.27)
\end{aligned}
$$

With the polarization evolution in terms of the envelopes, $\vec{E}$ and $\vec{P}$, Eq. (11.19) can be further simplified. By expanding the total field $\vec{E}(\vec{x},t)$ into mode fields $A_p(z,t)$ [Eq. (11.4a)], Eq. (11.19), with the help of Eq. (11.27), becomes

$$
\begin{aligned}
& \int \mathrm{d}\omega \; \partial_z A_p(z,\omega) \mathrm{e}^{i\left[\left(\beta_p(\omega) - \beta_{(0)}\right) z - (\omega-\omega_0)t\right]} \\
& = \frac{1}{32\pi} \int \mathrm{d}\omega \frac{i\omega}{N_p(\omega) N_\ell(\omega) N_m(\omega) N_n(\omega)} \int \mathrm{d}x \, \mathrm{d}y \, \mathrm{d}t' \mathrm{e}^{-i(\omega-\omega_0)(t-t')} \\
& \quad \sum_{ij} \sum_{\ell mn} \Bigg\{ (F_p^i)^* F_\ell^i F_m^j (F_n^j)^* \cdot 2A_\ell \int \left[\epsilon_0 \chi^{(3)}_{\text{electronic}} \delta(\tau) + \mathtt{R}_a(\tau) \right] (A_m A_n^*) (t-\tau) \, \mathrm{d}\tau \\
& \qquad + (F_p^i)^* (F_\ell^i)^* F_m^j F_n^j A_\ell^* \int \left[\epsilon_0 \chi^{(3)}_{\text{electronic}} \delta(\tau) + \mathtt{R}_a(\tau) \mathrm{e}^{2i\omega_0\tau} \right] (A_m A_n) (t-\tau) \, \mathrm{d}\tau \\
& \qquad + (F_p^i)^* F_\ell^j F_m^i (F_n^j)^* A_\ell \int \mathtt{R}_b(\tau) A_m A_n^* \, \mathrm{d}\tau \\
& \qquad + (F_p^i)^* F_\ell^j (F_m^i)^* F_n^j A_\ell \int \mathtt{R}_b(\tau) A_m^* A_n \, \mathrm{d}\tau \\
& \qquad + (F_p^i)^* (F_\ell^j)^* F_m^i F_n^j A_\ell^* \int \mathtt{R}_b(\tau) A_m A_n \mathrm{e}^{2i\omega_0\tau} \, \mathrm{d}\tau \Bigg\} . \quad (11.28)
\end{aligned}
$$

To solve this, we need to digress for a while and derive that

$$
\begin{aligned}
& \int \omega Q(\omega) Y(z,t') \mathrm{e}^{-i(\omega-\omega_0)(t-t')} \, \mathrm{d}t' \, \mathrm{d}\omega \\
& = \int \left[\omega_0 Q(\omega_0) + (\omega-\omega_0) \, \partial_\omega \left(\omega Q \right)|_{\omega_0} \right] Y(z,t') \mathrm{e}^{-i(\omega-\omega_0)(t-t')} \, \mathrm{d}\omega \, \mathrm{d}t' \\
& = \int \left\{ \omega_0 Q(\omega_0) + (\omega-\omega_0) \left[Q(\omega_0) + \omega_0 Q'(\omega_0) \right] \right\} Y(z,t') \mathrm{e}^{-i(\omega-\omega_0)(t-t')} \, \mathrm{d}\omega \, \mathrm{d}t' \\
& = \int \left[\omega Q(\omega_0) + (\omega-\omega_0) \, \omega_0 Q'(\omega_0) \right] Y(z,t') \mathrm{e}^{-i(\omega-\omega_0)(t-t')} \, \mathrm{d}\omega \, \mathrm{d}t' \\
& = \int \left[\frac{\omega}{\omega_0} + (\omega-\omega_0) \, \partial_\omega \left(\ln Q \right)|_{\omega_0} \right] \omega_0 Q(\omega_0) Y(z,t') \mathrm{e}^{-i(\omega-\omega_0)(t-t')} \, \mathrm{d}\omega \, \mathrm{d}t' \\
& = \omega_0 Q(\omega_0) \int \left[\frac{1}{\omega_0} \int \omega \mathrm{e}^{-i(\omega-\omega_0)(t-t')} \, \mathrm{d}\omega + \right.
\end{aligned}
$$

$$
\begin{aligned}
&\partial_\omega \left(\ln Q\right)|_{\omega_0} \int \left(\omega-\omega_0\right) \mathrm{e}^{-i(\omega-\omega_0)\left(t-t'\right)} \,\mathrm{d}\omega \Bigg] Y(z,t')\,\mathrm{d}t' \\
&= \omega_0 Q(\omega_0) \int \left[\frac{1}{\omega_0}\left(i\partial_t+\omega_0\right)+\partial_\omega\left(\ln Q\right)|_{\omega_0}\left(i\partial_t\right)\right] 2\pi\delta(t-t')Y(z,t')\,\mathrm{d}t' \\
&= 2\pi\omega_0 Q(\omega_0)\left\{1+\left[\frac{1}{\omega_0}+\partial_\omega\left(\ln Q\right)|_{\omega_0}\right]\left(i\partial_t\right)\right\} Y(z,t). \qquad (11.29)
\end{aligned}
$$

Let

$$
\tau_{plmn} = \frac{1}{\omega_0}+\partial_\omega\left(\ln Q_{plmn}\right)\Big|_{\omega_0}, \qquad (11.30)
$$

Eq. (11.28) leads to

$$
\begin{aligned}
&\int \mathrm{d}\omega\ \partial_z A_p(z,\omega) \mathrm{e}^{i\left[\left(\beta_p(\omega)-\beta_{(0)}\right)z-(\omega-\omega_0)t\right]} \\
&= \frac{i\omega_0}{16}\left[1+\tau_{p\ell mn}\left(i\partial_t\right)\right]\cdot \\
&\quad \sum_{\ell mn}\Bigg\{ Q^{\mathtt{R}_a}_{p\ell mn}(\omega_0) 2A_\ell \int \left[\epsilon_0\chi^{(3)}_{\text{electronic}}\delta(\tau)+\mathtt{R}_a(\tau)\right]\left(A_m A_n^*\right)(t-\tau)\,\mathrm{d}\tau \\
&\qquad + Q^{k}_{p\ell mn}(\omega_0) A_n^* \int \left[\epsilon_0\chi^{(3)}_{\text{electronic}}\delta(\tau)+\mathtt{R}_a(\tau)\mathrm{e}^{2i\omega_0\tau}\right]\left(A_m A_\ell\right)(t-\tau)\,\mathrm{d}\tau\Bigg\} \\
&\qquad + Q^{r_b}_{p\ell mn}(\omega_0) A_\ell \int \mathtt{R}_b(\tau) A_m A_n^*\,\mathrm{d}\tau \\
&\qquad + Q^{k}_{p\ell mn}(\omega_0) A_\ell \int \mathtt{R}_b(\tau) A_n^* A_m\,\mathrm{d}\tau \\
&\qquad + \frac{(F_p^i)^*(F_\ell^j)^* F_m^i F_n^j}{N_p N_\ell N_m N_n} A_m^* \int \mathtt{R}_b(\tau) A_\ell A_n \mathrm{e}^{2i\omega_0\tau}\,\mathrm{d}\tau\Bigg\}. \qquad (11.31)
\end{aligned}
$$

Conclude the temporal evolution equation

Now that we have solved the dispersion [Eq. (11.18)] and the polarization [Eq. (11.31)] terms. The temporal evolution of $A_p(z,t)$ [Eq. (11.17)], after neglecting highly-oscillating integrals with $\mathrm{e}^{2i\omega_0\tau}$, becomes

$$
\begin{aligned}
\partial_z A_p(z,t) = {}& i\left[\sum_m \frac{\left(i\partial_t\right)^m}{m!}\beta_p^{(m)}(\omega_0)-\beta_{(0)}\right] A_p(z,t) \\
&+ \frac{i\omega_0}{16}\left[1+\tau_{p\ell mn}\left(i\partial_t\right)\right]\sum_{\ell mn}\Bigg\{\epsilon_0\chi^{(3)}_{\text{electronic}} 3Q^K_{p\ell mn}(\omega_0)\left[A_\ell A_m A_n^*\right] \\
&+ 2\Bigg\{ Q^{\mathtt{R}_a}_{p\ell mn}(\omega_0)\left[A_\ell\left[\mathtt{R}_a * \left(A_m A_n^*\right)\right]\right] + Q^{\mathtt{R}_b}_{p\ell mn}(\omega_0)\left[A_\ell\left[\mathtt{R}_b * \left(A_m A_n^*\right)\right]\right]\Bigg\}\Bigg\}, \qquad (11.32)
\end{aligned}
$$

where

$$
Q^K_{p\ell mn} = \frac{1}{3}\left(2Q^{\mathtt{R}_a}_{p\ell mn}+Q^k_{p\ell mn}\right) \qquad Q^k_{p\ell mn} = \frac{\int\left(\vec{F}_p^*\cdot\vec{F}_n^*\right)\left(\vec{F}_\ell\cdot\vec{F}_m\right)\mathrm{d}^2x}{N_p N_\ell N_m N_n} \qquad (11.33a)
$$

$$Q^{\mathtt{R}_a}_{p\ell mn} = \frac{\int \left(\vec{F}^*_p \cdot \vec{F}_\ell\right)\left(\vec{F}_m \cdot \vec{F}^*_n\right) \mathrm{d}^2x}{N_p N_\ell N_m N_n} \tag{11.33b}$$

$$Q^{\mathtt{R}_b}_{p\ell mn} = \frac{1}{2}\left(Q^{r_b}_{p\ell mn} + Q^{k}_{p\ell mn}\right) \qquad Q^{r_b}_{p\ell mn} = \frac{\int \left(\vec{F}^*_p \cdot \vec{F}_m\right)\left(\vec{F}_\ell \cdot \vec{F}^*_n\right) \mathrm{d}^2x}{N_p N_\ell N_m N_n} \tag{11.33c}$$

are the overlap integrals of eigenmode fields $\vec{F}_j(\vec{r}_\perp)$ (in 1/m), where $\vec{r}_\perp = (x, y)$. $\mathrm{d}^2x = \mathrm{d}x\,\mathrm{d}y$ represents the integral over the spatial (waveguide tranverse) domain. For simplicity, $N_pN_\ell N_mN_n \approx \epsilon_0^2 n_{\mathrm{eff}}^2 c^2/4$, where $n_{\mathrm{eff}} = \beta_1(\omega)/k_0$ is taken from the effective index of mode 1.

Consider the moving frame:

$$T = t - \beta_{(1)} z \quad \Rightarrow \quad \partial_z A_p(z,t) = \partial_z A_p(z,T) - \beta_{(1)}\partial_T A_p(z,T), \tag{11.34}$$

where the ∂_z on the left-hand side is actually a total derivative, then

$$\begin{aligned}\partial_z A_p(z,T) = &\left\{ i\left[\beta_p^{(0)}(\omega_0) - \beta_{(0)}\right] - \left[\beta_p^{(1)}(\omega_0) - \beta_{(1)}\right]\partial_T \right\} A_p(z,T) \\ &+ i\sum_{m\geq 2} \frac{(i\partial_T)^m}{m!}\beta_p^{(m)}(\omega_0) A_p(z,T) \\ &+ \frac{i\omega_0}{4}\left[1 + \tau_{p\ell mn}\,(i\partial_t)\right]\sum_{\ell mn}\left\{ \epsilon_0 \chi^{(3)}_{\mathrm{electronic}} \frac{3}{4} Q^K_{p\ell mn}(\omega_0)\left[A_\ell A_m A_n^*\right] \right. \\ &\left. + \frac{1}{2}\left\{ Q^{\mathtt{R}_a}_{p\ell mn}(\omega_0)\left[A_\ell\left[\mathtt{R}_a * (A_m A_n^*)\right]\right] + Q^{\mathtt{R}_b}_{p\ell mn}(\omega_0)\left[A_\ell\left[\mathtt{R}_b * (A_m A_n^*)\right]\right] \right\} \right\}.\end{aligned} \tag{11.35}$$

Transform into the frequency domain

To account for the frequency dependence of all orders, we would like to work in the frequency domain so that the approximation of the Taylor series expansion can be avoided. Therefore, we now return to the frequency domain by taking the Fourier transform of Eq. (11.35) with respect to the frequency variable $\Omega = \omega - \omega_0$. It becomes

$$\begin{aligned}\partial_z A_p(z,\Omega) = &\, i\left[\beta_p(\omega) - \left(\beta_{(0)} + \beta_{(1)}\Omega\right)\right] A_p(z,\Omega) \\ &+ \mathfrak{F}\left[\frac{i\omega_0}{4}\left[1 + \tau_{p\ell mn}\,(i\partial_t)\right]\sum_{\ell mn}\left\{ \epsilon_0 \chi^{(3)}_{\mathrm{electronic}} \frac{3}{4} Q^K_{p\ell mn}(\omega_0)\left[A_\ell A_m A_n^*\right] \right.\right. \\ &\left.\left. + \frac{1}{2}\left\{ Q^{\mathtt{R}_a}_{p\ell mn}(\omega_0)\left[A_\ell\left[\mathtt{R}_a * (A_m A_n^*)\right]\right] + Q^{\mathtt{R}_b}_{p\ell mn}(\omega_0)\left[A_\ell\left[\mathtt{R}_b * (A_m A_n^*)\right]\right] \right\} \right\}\right].\end{aligned} \tag{11.36}$$

To solve this, we first derive another relation:

$$\begin{aligned}\int \omega Q(\omega) Y(z,t') \mathrm{e}^{-i(\omega-\omega_0)(t-t')}\,\mathrm{d}t'\,\mathrm{d}\omega &= \frac{1}{C_{\mathfrak{F}}}\mathfrak{F}^{-1}\left[\int \omega Q(\omega) Y(z,t') \mathrm{e}^{i\Omega t'}\,\mathrm{d}t'\right] \\ &= \frac{1}{C_{\mathfrak{F}} C_{\mathfrak{I}\mathfrak{F}}}\mathfrak{F}^{-1}\left[\omega Q(\omega) Y(z,\Omega)\right] \\ &= 2\pi\mathfrak{F}^{-1}\left[\omega Q(\omega) Y(z,\Omega)\right].\end{aligned} \tag{11.37}$$

With it, the complicated terms in nonlinearity, by working backward in Eq. (11.29), are

$$\begin{aligned}\omega_0 Q(\omega_0)\left[1+\tau\left(i\partial_t\right)\right]Y(z,t) &= \frac{1}{2\pi}\int \omega Q(\omega) Y(z,t') \mathrm{e}^{-i(\omega-\omega_0)\left(t-t'\right)}\,\mathrm{d}t'\,\mathrm{d}\omega \\ &= \mathfrak{F}^{-1}\left[\omega Q(\omega) Y(z,\Omega)\right]. \end{aligned} \tag{11.38}$$

Eventually, we obtain the final UPPE from Eq. (11.36):

$$\begin{aligned}\partial_z A_p(z,\Omega) = & \, i\left[\beta_p(\omega)-\left(\beta_{(0)}+\beta_{(1)}\Omega\right)\right]A_p(z,\Omega) \\ & + \frac{i\omega}{4}\sum_{\ell mn}\Bigg\{\left(\frac{3}{4}\epsilon_0\chi^{(3)}_{\text{electronic}}\right)Q^K_{p\ell mn}\mathfrak{F}[A_\ell A_m A_n^*] \\ & \quad + \Bigg\{Q^{\mathtt{R}_a}_{p\ell mn}\mathfrak{F}\left[A_\ell\left[\left(\frac{1}{2}\mathtt{R}_a\right)*\left(A_m A_n^*\right)\right]\right] \\ & \quad + Q^{\mathtt{R}_b}_{p\ell mn}\mathfrak{F}\left[A_\ell\left[\left(\frac{1}{2}\mathtt{R}_b\right)*\left(A_m A_n^*\right)\right]\right]\Bigg\}\Bigg\}. \end{aligned} \tag{11.39}$$

If we apply the model commonly used in solid-core silica fiber, where

$$\epsilon_0\chi^{(3)}_{\text{electronic}} = \left(1-f_R\right)\epsilon_0\chi^{(3)}_{xxxx} = \left(1-f_R\right)\frac{4\epsilon_0^2 n_{\text{eff}}^2 c}{3}n_2 \tag{11.40a}$$

$$\mathtt{R}_a = f_R f_a \epsilon_0\chi^{(3)}_{xxxx}\frac{3}{2}h_a = 2 f_R f_a \epsilon_0^2 n_{\text{eff}}^2 c n_2 h_a(t) \tag{11.40b}$$

$$\mathtt{R}_b = f_R f_b \epsilon_0\chi^{(3)}_{xxxx}\frac{3}{2}h_b = 2 f_R f_b \epsilon_0^2 n_{\text{eff}}^2 c n_2 h_b(t) \tag{11.40c}$$

and include the artificial gain term, we reach the UPPE of solid-core fiber [Eq. (1.1)] [4].

11.0.1 Normalization relation

Since it is not straightforward to compute N_n from Eq. (11.2), an approximate form of N_n is introduced and used more widely.

Let

$$\vec{\mathcal{F}}_m(x,y,\omega) = \vec{F}_m(x,y,\omega)\mathrm{e}^{i\beta_m z} \tag{11.41a}$$

$$\vec{\mathcal{G}}_m(x,y,\omega) = \vec{G}_m(x,y,\omega)\mathrm{e}^{i\beta_m z}, \tag{11.41b}$$

the normalization relation [Eq. (11.2)], with Eq. (11.10b), gives

$$\begin{aligned}
\delta_{nm}N_n^2(\omega) &= \frac{1}{4}\int \mathrm{d}x\,\mathrm{d}y\left(\vec{F}_m^* \times \vec{G}_n + \vec{F}_n \times \vec{G}_m^*\right)\cdot\hat{z} \\
&= \frac{1}{4}\int \mathrm{d}x\,\mathrm{d}y\left[\vec{\mathcal{F}}_m^* \times \vec{\mathcal{G}}_n + \vec{\mathcal{F}}_n \times \vec{\mathcal{G}}_m^*\right]\cdot\hat{z} \\
&= \frac{1}{4}\int \mathrm{d}x\,\mathrm{d}y\left\{\vec{\mathcal{F}}_m^* \times \left[\frac{\nabla\times\vec{\mathcal{F}}_n}{\mu_0 i\omega}\right] + \vec{\mathcal{F}}_n \times \left[\frac{\nabla\times\vec{\mathcal{F}}_m}{\mu_0 i\omega}\right]^*\right\}\cdot\hat{z} \\
&= \frac{1}{4\mu_0 i\omega}\int \mathrm{d}x\,\mathrm{d}y\left\{\vec{\mathcal{F}}_m^* \times \left[\nabla\times\vec{\mathcal{F}}_n\right] - \vec{\mathcal{F}}_n \times \left[\nabla\times\vec{\mathcal{F}}_m\right]^*\right\} \\
&= \frac{1}{4\mu_0 i\omega}\int \mathrm{d}x\,\mathrm{d}y\left\{(\mathcal{F}_m^x)^*\left[\partial_z\mathcal{F}_n^x - \partial_x\mathcal{F}_n^z\right] - (\mathcal{F}_m^y)^*\left[\partial_y\mathcal{F}_n^z - \partial_z\mathcal{F}_n^y\right]\right. \\
&\qquad \left. -\mathcal{F}_n^x\left[\partial_z\mathcal{F}_m^x - \partial_x\mathcal{F}_m^z\right]^* + \mathcal{F}_n^y\left[\partial_y\mathcal{F}_m^z - \partial_z\mathcal{F}_m^y\right]^*\right\} \\
&= \frac{1}{4\mu_0 i\omega}\int \mathrm{d}x\,\mathrm{d}y\left\{i\beta_n\left[(F_m^x)^* F_n^x + (F_m^y)^* F_n^y\right] + i\beta_m\left[F_n^x (F_m^x)^* + F_n^y (F_m^y)^*\right]\right. \\
&\qquad \left. -(F_m^x)^*(\partial_x F_n^z) - (F_m^y)^*\left(\partial_y F_n^z\right) + F_n^x(\partial_x F_m^z)^* + F_n^y\left(\partial_y F_m^z\right)^*\right\}
\end{aligned} \tag{11.42}$$

Because $F_n^z = 0$, the normalization relation with $\vec{F}_m$ and $\vec{G}_m$ is reduced to the one with only $\vec{F}_m$:

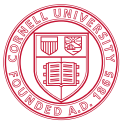
$$\begin{aligned}
\delta_{nm}N_n^2(\omega) &= \frac{1}{4}\int \mathrm{d}x\,\mathrm{d}y\left(\vec{F}_m^* \times \vec{G}_n + \vec{F}_n \times \vec{G}_m^*\right)\cdot\hat{z} \\
&= \delta_{nm}\frac{\beta_n}{2\mu_0\omega}\int \mathrm{d}x\,\mathrm{d}y\left|\vec{F}_n\right|^2 \\
&= \delta_{nm}\frac{n_n}{2\mu_0 c}\int \mathrm{d}x\,\mathrm{d}y\left|\vec{F}_n\right|^2 \\
&= \delta_{nm}\frac{\epsilon_0 n_n c}{2}\int \mathrm{d}x\,\mathrm{d}y\left|\vec{F}_n\right|^2
\end{aligned} \tag{11.43}$$

Generally, when E has a unit of V/m, a transformation into the unit $\sqrt{\mathrm{W/m^2}}$ is introduced as follows.

$$|E_W|^2 = \frac{\epsilon_0 nc}{2}|E_V|^2. \tag{11.44}$$

$$\begin{aligned}
\vec{\mathbb{E}}(\vec{x},t) &= \frac{1}{2}\left\{\sum_n \frac{\vec{F}_n(x,y)}{N_n}\left[A_n(z,t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\right] + \text{c.c.}\right\} \\
&= \frac{1}{2}\left\{\sum_n \frac{\vec{F}_n(x,y)}{\sqrt{\frac{\epsilon_0 n_n c}{2}\left(\int\left|\vec{F}_n\right|^2 \mathrm{d}^2x\right)}}\left[A_n(z,t)\mathrm{e}^{i\left(\beta_{(0)}z-\omega_0 t\right)}\right] + \text{c.c.}\right\}
\end{aligned} \tag{11.45}$$

In Eq. (11.45), $\vec{\mathbb{E}}(\vec{x},t)$ is further represented by $A_n(z,t)$ that has the unit $\sqrt{\mathrm{W}}$.



Cornell University

Bibliography

[1] Y.-H. Chen and F. Wise, "A simple accurate way to model noise-seeded ultrafast nonlinear processes", arXiv preprint arXiv: 2410.20567 (2024).

[2] Y.-H. Chen, J. Moses, and F. Wise, "Femtosecond long-wave-infrared generation in hydrogen-filled hollow-core fiber", J. Opt. Soc. Am. B **40**, 796–806 (2023).

[3] Y.-H. Chen and F. Wise, "Unified and vector theory of Raman scattering in gas-filled hollow-core fiber across temporal regimes", APL Photonics **9**, 030902 (2024).

[4] Y.-H. Chen, H. Haig, Y. Wu, Z. Ziegler, and F. Wise, "Accurate modeling of ultrafast nonlinear pulse propagation in multimode gain fiber", J. Opt. Soc. Am. B **40**, 2633–2642 (2023).

[5] A. M. Heidt, "Efficient Adaptive Step Size Method for the Simulation of Supercontinuum Generation in Optical Fibers", J. Light. Technol. **27**, 3984–3991 (2009).

[6] S. Balac and F. Mahé, "Embedded Runge-Kutta scheme for step-size control in the interaction picture method", Comput. Phys. Commun. **184**, 1211–1219 (2013).

[7] L. G. Wright, Z. M. Ziegler, P. M. Lushnikov, Z. Zhu, M. A. Eftekhar, D. N. Christodoulides, and F. W. Wise, "Multimode Nonlinear Fiber Optics: Massively Parallel Numerical Solver, Tutorial, and Outlook", IEEE J. Sel. Top. Quantum Electron. **24**, 1–16 (2018).

[8] Y.-H. Chen and F. Wise, "Field-based treatment of transient gain in short-pulse optical amplifiers", Optica **12**, 879–889 (2025).

[9] W. Krupke, "Induced-emission cross sections in neodymium laser glasses", IEEE J. Quantum Electron. **10**, 450–457 (1974).

[10] Q. Yanbo, D. Ning, P. Mingying, Y. Lüyun, C. Danping, Q. Jianrong, Z. Congshan, and T. Akai, "Spectroscopic Properties of Nd^{3+}-Doped High Silica Glass Prepared by Sintering Porous Glass", J Rare Earth **24**, 765–770 (2006).

[11] M. Naftaly and A. Jha, "Nd^{3+}-doped fluoroaluminate glasses for a 1.3 μm amplifier", J. Appl. Phys. **87**, 2098–2104 (2000).

[12] Y. Yamasaki, R. Azuma, Y. Kagebayashi, K. Fujioka, and Y. Fujimoto, "Optical properties of Er^{3+} heavily doped silica glass fabricated by zeolite method", J. Non-Cryst. Solids **543**, 120149 (2020).

[13] P. Babu, H. J. Seo, K. H. Jang, R. Balakrishnaiah, C. K. Jayasankar, K.-S. Lim, and V. Lavín, "Optical spectroscopy, 1.5 μm emission, and upconversion properties of Er^{3+}-doped metaphosphate laser glasses", J. Opt. Soc. Am. B **24**, 2218–2228 (2007).

[14] D. Ning, Q. Yanbo, Y. Lüyun, P. Mingying, W. Chen, Z. Qingling, Z. Chongjun, Q. Jianrong, Z. Congshan, C. Danping, and A. Tomoko, "High Quantum Efficiency and High Concentration Erbium-Doped Silica Glasses Fabricated by Sintering Nanoporous Glasses", J Rare Earth **24**, 761–764 (2006).

[15] R. S. Quimby, W. J. Miniscalco, and B. A. Thompson, "Excited-state absorption at 980 nm in erbium-doped glass", in Fiber laser sources and amplifiers iii, Vol. 1581, edited by M. J. F. Digonnet and E. Snitzer (International Society for Optics and Photonics, 1992), pp. 72–79.

[16] S. Tanabe, T. Ohyagi, N. Soga, and T. Hanada, "Compositional dependence of Judd-Ofelt parameters of Er^{3+} ions in alkali-metal borate glasses", Phys. Rev. B **46**, 3305–3310 (1992).

[17] G. S. Ofelt, "Intensities of Crystal Spectra of Rare-Earth Ions", J. Chem. Phys. **37**, 511–520 (1962).

[18] L. Gomes, M. Oermann, H. Ebendorff-Heidepriem, D. Ottaway, T. Monro, A. Felipe Henriques Librantz, and S. D. Jackson, "Energy level decay and excited state absorption processes in erbium-doped tellurite glass", J. Appl. Phys. **110**, 083111 (2011).

[19] W. C. Wang, W. J. Zhang, L. X. Li, Y. Liu, D. D. Chen, Q. Qian, and Q. Y. Zhang, "Spectroscopic and structural characterization of barium tellurite glass fibers for mid-infrared ultra-broad tunable fiber lasers", Opt. Mater. Express **6**, 2095–2107 (2016).

[20] E. Treacy, "Optical pulse compression with diffraction gratings", IEEE J. Quantum Electron. **5**, 454–458 (1969).

[21] W. Dietel, J. J. Fontaine, and J.-C. Diels, "Intracavity pulse compression with glass: a new method of generating pulses shorter than 60 fsec", Opt. Lett. **8**, 4–6 (1983).

[22] R. L. Fork, O. E. Martinez, and J. P. Gordon, "Negative dispersion using pairs of prisms", Opt. Lett. **9**, 150–152 (1984).

[23] A. Offner, "Unit power imaging catoptric anastigmat", U.S. pat. 3748015 (June 1971).

[24] O. E. Martinez, R. L. Fork, and J. P. Gordon, "Theory of passively mode-locked lasers including self-phase modulation and group-velocity dispersion", Opt. Lett. **9**, 156–158 (1984).

[25] O. E. Martinez, R. L. Fork, and J. P. Gordon, "Theory of passively mode-locked lasers for the case of a nonlinear complex-propagation coefficient", J. Opt. Soc. Am. B **2**, 753–760 (1985).

[26] G. P. Agrawal, "Chapter 6 - pulse compression", in *Applications of nonlinear fiber optics (second edition)* (Academic Press, Burlington, 2008), pp. 245–300.

[27] P. Tournois, "New diffraction grating pair with very linear dispersion for laser pulse compression", English, Electron. Lett. **29**, 1414–1415 (1993).

[28] S. Kane and J. Squier, "Grism-pair stretcher–compressor system for simultaneous second- and third-order dispersion compensation in chirped-pulse amplification", J. Opt. Soc. Am. B **14**, 661–665 (1997).

[29] T. H. Dou, R. Tautz, X. Gu, G. Marcus, T. Feurer, F. Krausz, and L. Veisz, "Dispersion control with reflection grisms of an ultra-broadband spectrum approaching a full octave", Opt. Express **18**, 27900–27909 (2010).

[30] N. Forget, V. Crozatier, and P. Tournois, "Transmission Bragg-grating grisms for pulse compression", Appl. Phys. B **109**, 121–125 (2012).

[31] G. Cheriaux, P. Rousseau, F. Salin, J. P. Chambaret, B. Walker, and L. F. Dimauro, "Aberration-free stretcher design for ultrashort-pulse amplification", Opt. Lett. **21**, 414–416 (1996).

[32] X. Liu, C. Wang, X. Wang, X. Lu, P. Bai, Y. Liu, Y. Li, K. Liu, L. Yu, Y. Leng, and R. Li, "Dispersion Management in 10-PW Laser Front End", Optics **1**, 191–201 (2020).

[33] D. Shvydkoy and V. Trunov, "Negatively chirped pulse compressor with internal telescope for 1.4 µm range", Appl. Phys. B **126**, 116 (2020).

[34] A. R. Mickelson, *Guided Wave Optics* (Springer New York, NY, 1993).

[35] M. Conforti, A. Marini, D. Faccio, and F. Biancalana, "Negative frequencies get real: a missing puzzle piece in nonlinear optics", arXiv preprint arXiv:1305.5264, `10.48550/ARXIV.1305.5264` (2013).

[36] Q. Lin and G. P. Agrawal, "Raman response function for silica fibers", Opt. Lett. **31**, 3086–3088 (2006).