Light-Ultracold Atomic Python Codes

Grant W. Henderson

SUPA & Department of Physics, University of Strathclyde, Glasgow, Scotland, G4 0NG, United Kingdom

Biomathematics and Statistics Scotland, James Clerk Maxwell Building, The King's Buildings, Peter Guthrie Tait Road, Edinburgh, EH9 3FD, United Kingdom (from Feb. 2025)

January 29, 2025

This is a 'handbook' to accompany the various codes prepared for current and future FOAM group members at Strathclyde. Each principal model employs a split-step Fourier method to propagate or evolve various combinations of (mostly) two-dimensional optical and atomic fields in a variety of physical interaction configurations.

Contents

1 Example System

1.1 Theoretical Model

The various codes have been written to evolve a combination of partial differential equations (PDEs) that describe the evolution of two-dimensional (2D) fields through the interplay of various linear and nonlinear forces. Take, for example, a theoretical model which considers the evolution of a single PDE of the form:

$$\partial_\alpha \mathcal{A} = \underbrace{i\beta_\mathrm{L}\nabla_\perp^2\mathcal{A}}_{\text{Linear}} + \underbrace{i\beta_\mathrm{NL}|\mathcal{A}|^2\mathcal{A}}_{\text{Nonlinear}}. \tag{1}$$

Eq. (1) may be used to describe the evolution of a field, $\mathcal{A}$, in an evolution dimension, α, subject to the interplay of two forces. The labelled linear force is used to represent the repulsive spreading or attractive focusing of the field in the perpendicular ($\perp$) dimensions, and its strength and nature is scaled by the parameter β_L. The labelled nonlinear force, which may represent a number of physical phenomena, again potentially repulsive or attractive, has its strength and nature scaled by the parameter β_NL.

1.1.1 Scalings

In realising efficient integration of models akin to Eq. (1), dimensionless units will generally be used, and as such everything that appears in Eq. (1) will have one (or more) scaling(s) associated with it. Most are model dependent and as such outlined at the appropriate time, but the scaling performed upon the perpendicular $\perp = (\xi, \eta)$ domains are the same for all models:

$$(\xi, \eta) = \frac{\sqrt{2}(x, y)}{w_\mathrm{s}}, \tag{2}$$

where w_s represents a scaling beam waist, typically $\mathcal{O}\left(10^{-5}\mathrm{m}\right)$ for the fields considered in these models.

1.2 Numerical Integration: "Example_PDE.py"

So, how do you integrate Eq. (1)? The file "Example_PDE.py" provides the solution. It applies the split-step Fourier method (SSFM) [1] to take the field's initial state ($\alpha = 0$) in the scaled transverse dimensions $\perp$ and evolves it, subject to Eq. (1), in α. The code itself is relatively short, but relies on several packages contained in the accompanying "FOAMilyCAC.py" library. In the next few pages, various important sections of both the main code and associated function library are outlined.

1.2.1 Parameter Setup

Lines 6-37 play an important role: this is where parameters and fields are initially defined, which will determine the subsequent dynamics.

```
10 final_alpha = 0.5 # alpha-value to evolve to
```

This line allows you to set the final value of α to evolve to.

```
12 w_L = 10 # w_0, scaling beam waist, microns
13 w_F = w_L # Optional control to change the initial w_F without w_L, microns.
14 l_mode = 0 # \ell-index of field
15 l_mode_prof = l_mode # Optional control to alter OAM without profile change
16 OAM = 1 # Do you want OAM on = 1? or off = 0?
17 p_mode = 0 # p-index of field
18 F_amp = 1.0 # Initial amplitude of field
```

These lines define the parameters associated with the initial profile of the field $\mathcal{A}$. Line 12 defines the scaling beam waist w_s of Eq. (2), before line 13 allows the setting of the initial profile's beam waist different to w_s if desired. The code is setup to input Laugerre-Gaussian modes, defined in e.g. [2, 3], and so lines 14-18 define the mode's key parameters. Line 15 allows you to change the OAM applied to the mode without changing its amplitude profile, whilst line 16 allows you to totally 'turn off' the mode's OAM if desired.

```
20 noise_amplitude = 10**-2 # Relative to field amplitude
```

Line 20 defines the computational noise strength applied to the field - set relative to the field $\mathcal{A}$'s maximum amplitude.

```
22 beta_L = 1 # Linear term prefactor
23 beta_NL = 1 # Nonlinear term prefactor
```

This is where the parameters β_{L} and β_{NL} present in Eq. (1) are defined.

```
25 N_w = 10 # Number of characteristic beam waists in domain size, multiples of w_L
26 Nx = Ny = 2**8 # Number of grid points, square grid s.t. Nx = Ny
27 boundary_extent = 0.85 # proportion of grid covered by boundaries, 0 -> 1
28 boundary_grad = 1 # Gradient of absorbing boundaries, fairly arbitrary
```

These lines define parameters relating to the 2D numerical grid. The grid size is defined relative to the scaling size in line 25, such that the combination of lines 25 and 10 here will realise a grid that goes from $-50\mu m$ to 50μm. The next line stipulates the number of grid points, meaning that this $-50 \rightarrow 50\mu$m space will be split into 256 evenly-spaced segments. The greater this number then the more precise, yet the slower, your numerical model will be.

The final two lines construct an absorbing boundary, which is applied to the atomic field to prohibit 'wrapping' around boundaries at the domain edges. It is constructed as a 'top hat' - in the central regions it's equal to one so doesn't impact the dynamics, but at the edges of the grid it's equal to zero such that the field falls off to zero, prohibiting wrapping. The extent of the boundary is set relative to the grid's extent in the penultimate line, and the gradient, defined in the final line, sets the fall-off rate.

```
30 data_fac = 0.05 # alpha-interval to store data at
31 plot_fac = 0.05 # alpha-interval to plot fields at
32 track_to_max = 1 # Do you want your field plots all normalised to run max? Yes = 1, No = 0
33
34 output_directory = 'SimulationOutput' # Name of directory you wish to store output files
35 overwrite = 1 # Do you want to overwrite results if 'output_directory' exists? Yes = 1, No
   ↪  = 0 (job will fail)
```

These lines control what you get out of the simulation. Line 30 sets the rate at which you get data files of the current field output in terms of α, line 31 does the same but for outputting a plot of the field, and the following line determines whether you want to normalise all plots to the tracked maximum intensity of $\mathcal{A}$ (if so, set this equal to 1) or to each step's maximum intensity (if so, do not set this equal to 1). Finally, line 34 determines where your run's outputs will be stored, relative to where the Python script is, and line 35 determines whether, if this directory already exists, you are happy to overwrite what's there or not.

1.2.2 Grid Setup and Scalings

Lines 45-76 construct the 2D grid in (scaled) real space, along with k-space, for subsequent use, as specified by the selections made in Section 1.2.1. The following lines may be particularly useful for converting back to unscaled domains (as defined by Eq. (2)) for things like plotting / analysis:

```
61 x_physical = (x * xy_scaling) # microns
62 y_physical = (y * xy_scaling) # microns
```

The evolution step size, $d\alpha$, is set in line 74:

```
74 da = dx**2 / 4 # step size along alpha-axis, ensures anti-aliasing satisfaction
```

This is *not* left as a free parameter but is instead set, relative to the scaled dx, such that anti-aliasing conditions associated with fast Fourier transforms are satisfied.

1.2.3 Field Creation

Lines 76-89 create the initial field $\mathcal{A}[\alpha = 0]$ and absorbing boundaries:

```
80 # Field - LG mode
81
82 A = CACFuncs.LGMode(X, Y, N_opt, l_mode, l_mode_prof, p_mode, OAM, F_amp, noise_amplitude)
83 max_A = numpy.max(numpy.abs(A))
84
85 # Absorbing boundary creation
86
87 boundaries = CACFuncs.Boundaries(x, boundary_extent, boundary_grad, N_w, X, Y)
```

As mentioned, $\mathcal{A}[\alpha = 0]$ is given by a Laguerre-Gaussian mode [2, 3] in the scaled transverse domains. Its creation is performed using a function within the "FOAMilyCAC.py" library (lines 22-41), which was imported in line 42 of "Example_PDE.py" as 'CACFuncs'. Thanks to the parameter selections of lines 12-20, $\mathcal{A}[\alpha = 0]$ corresponds to a Gaussian profile of beam waist w_s.

Line 87 calls another function from 'CACFuncs' (lines 65-70 of "FOAMilyCAC.py") to construct the absorbing boundary.

1.2.4 Pre-Loop Configuration

Lines 89-132 configure some final pre-evolution-loop essentials within the code. They begin with lines 93-97, which simply introduce some key tracking variables which will be used throughout the upcoming evolution of the field $\mathcal{A}$.

Assuming that data and / or figures will be output, lines 99-122 then configure the directory structures as required, using a directory setup function from 'CACFuncs'. They then make initial outputs as appropriate:

- Data is output as '.csv' files where, for each output step, two data outputs will be created per field: the real part of the field ('FieldXRe_Y.csv') and the imaginary part of the field ('FieldXRe_Y.csv'). Here, X represents a field integer (always 1 in this case, as there is only one field being evolved so only one field to output at any given α), and Y represents the Y^{th} data output. As $\alpha = 0$, $Y = 0$ initially. They are output in this way to be of a format readable both back into Python and into Mathematica.
- Plots are output as a single '.png' file for each step: 'Fields_Y.png' representing the Y^{th} plot output. As $\alpha = 0$, $Y = 0$ initially. For each field output, two panels exist: the upper is an amplitude distribution (potentially scaled to the field's maximum along all α if setup in line 32), and the lower the corresponding phase distribution (scaled between $0 \rightarrow 2\pi$ using the cyclical 'hsv' colour scheme). The output corresponding to the parameters of this handbook at $\alpha = 0$ is shown in Fig. 1.

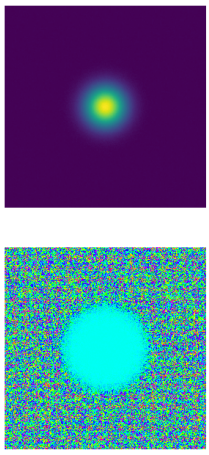

Figure 1: $\alpha = 0$ output figure for the outlined parameters used in "Example_PDE.py". Top row corresponds to the amplitude of $\mathcal{A}$, bottom row corresponds to the phase of $\mathcal{A}$.

Note that, if neither data or plot outputs are configured, the simulation will assume a configuration error and terminate before integration with a message indicating this.

Lines 124-130 output a file 'Info.txt' in the location defined in line 34 that contains the essential input information relating to the run for future reference.

1.2.5 Field Evolution

The while loop within lines 132-186 is where the important stuff happens: this is the loop that evolves the field $\mathcal{A}$ in α according to Eq. (1). It applies the split-step Fourier method [1], treating the linear term of Eq. (1) in k-space and the nonlinear term in real-space using a second-order Runge-Kutta (RK) method [4]. The essential procedure is:

- The field enters the loop in k-space through a fast Fourier transform:

```
137 A_fft = numpy.fft.fftshift(numpy.fft.fft2(A))/(Nx*Ny)
```

- The linear term of Eq. (1) is evolved for a $\frac{d\alpha}{2}$ step in k-space:

```
142 A_fft = numpy.exp(-1j * beta_L * ((Ex**2 + Ey**2) * da / 2)) * A_fft
```

- The atomic field is returned to real-space through a fast Fourier transform:

```
145 A = numpy.fft.ifft2(numpy.fft.fftshift(A_fft))*(Nx*Ny)
```

- A second-order RK method evolves the nonlinear term of Eq. (1) for a full $d\alpha$ step:

```
148 A_h = A
149 dAdt = (1j * (beta_NL*numpy.abs(A_h)**2)) * A_h
150 A_t = A_h + da*dAdt/2
151 dAdt = (1j * (beta_NL*numpy.abs(A_t)**2)) * A_t
152 A = A_h + da*dAdt
```

- The absorbing boundaries are applied to the field whilst in real space:

```
155 A = A * boundaries
```

- The field returns to k-space through a further fast Fourier transform:

```
158 A_fft = numpy.fft.fftshift(numpy.fft.fft2(A))/(Nx*Ny)
```

- The linear term of Eq. (1) is evolved for a $\frac{d\alpha}{2}$ step in k-space:

```
161 A_fft = numpy.exp(-1j * beta_L * ((Ex**2 + Ey**2) * da / 2)) * A_fft
```

At the end of this procedure, a full $d\alpha$ evolution has occurred for both linear and nonlinear terms, and so the loop may go round again. However, before that, some other useful checks are carried out: lines 168-170 test if a numerical breakdown has occurred (due to, e.g., strong focusing nonlinearities): if so, it'll kill the evolution. Lines 172-177 (179-184) check whether you have reached the next α value for a data (plot) output: if so, it outputs the data (plot) as requested.

1.2.6 Post-Loop Configuration

After the final α value defined in line 10 has been met or exceeded, or if a numerical breakdown has occurred, the simulation exits the loop and comes to lines 186-205. Assuming that data (figures) were output throughout the prior evolution, lines 190-195 (197-202) save (plot) the final field, setting the Y indicator to 'Fin' to indicate that this now relates to the final field of the run. Finally, line 205 updates the 'Info.txt' file with essential post-run α and runtime information.

This example is entirely customisable, so feel free to treat it as a 'toy model' and explore how changing various parameters impacts the output. The next sections explore particular light / atom / light-atom models for different interaction configurations: each code is built on the principles outlined here.

Part I
Propagation Systems

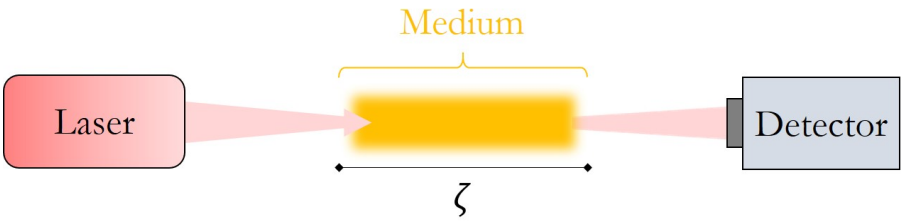

Figure 2: Schematic of a generic propagation system, of the kind considered in Part I. A laser produces a coherent beam that encounters some sort of propagation medium, of length ζ. Having propagated through the medium, the beam subsequently meets a detector.

2 Light Propagating in Free Space

2.1 Theoretical Model

In this case, we are considering the evolution along the longitudinal propagation dimension ζ of an optical field A, where the propagation medium is free space. This is modelled by the paraxial wave equation [5]:

$$\partial_\zeta A = i\nabla_\perp^2 A, \tag{3}$$

where the term in $\nabla_\perp^2$ represents transverse diffraction of the optical field.

2.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), the longitudinal domain, ζ, is also scaled by

$$\zeta = \frac{z}{k_L w_s^2}, \tag{4}$$

where k_L represents the optical wavevector, and w_s represents a scaling beam waist. It follows that propagation to $\zeta = 0.5$ is equivalent to one Rayleigh length of the optical field used for scaling.

2.2 Numerical Integration: "Propagation_PWE.py"

The Python code used to evolve a field according to the model of Eq. (3) is given in "Propagation_PWE.py". It takes a very similar form to the example of Section 1.2, but contains no nonlinear terms.

3 Light Propagating in a Kerr Medium

3.1 Theoretical Model

In this case, we are considering the evolution along the longitudinal propagation dimension ζ of an optical field A, where the propagation medium is a generic Kerr medium. This is modelled by the nonlinear Schrödinger equation [6]:

$$\partial_\zeta A = i\nabla_\perp^2 A + i\frac{\beta_\mathrm{K}|A|^2}{(1+\sigma_\mathrm{sat}|A|^2)}A, \tag{5}$$

where the term in $\nabla_\perp^2$ represents transverse diffraction of the optical field, the term in β_K represents self-focusing or self-defocusing, depending on the sign of β_K, due to a Kerr nonlinearity of strength $|\beta_\mathrm{K}|$, and the term in σ_sat represents optical saturation.

3.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), the longitudinal domain, ζ, is also scaled by

$$\zeta = \frac{z}{k_L w_s^2}, \tag{6}$$

where k_L represents the optical wavevector and w_s represents a scaling beam waist. It follows that propagation to $\zeta = 0.5$ is equivalent to one Rayleigh length of the optical field used for scaling. In addition, β_K and σ_sat are both scaled parameters relating to coefficients of the Kerr medium: see [5, 6] for further details.

3.2 Numerical Integration: "Propagation_NLSE.py"

The Python code used to evolve a field according to the model of Eq. (5) is given in "Propagation_NLSE.py". It takes a very similar form to the example of Section 1.2, with only minor adjustments reflecting the differences between the model and parameters of Eq. (5) compared to those of Eq. (1).

4 Evolution of an Ultracold Atomic Field

4.1 Theoretical Model

In this case, we are considering the evolution along the space/time variable ζ of an ultracold atomic field ψ, i.e. effectively 'turning off' the optical components of Fig. 2, and modelling the free evolution of the medium only. This is modelled by the Gross-Pitaevskii equation [7, 8]:

$$\partial_\zeta \psi = i\nabla_\perp^2 \psi - i\left(\beta_\mathrm{col}|\psi|^2 - iL_3|\psi|^4\right)\psi, \tag{7}$$

where the term in $\nabla_\perp^2$ represents kinetic energy, the term in β_col represents interatomic collisions, and the term in L_3 represents three-body atomic loss at high field strengths.

4.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), β_col and L_3 are scaled parameters relating to the strength and nature of interatomic scattering, and the characteristic features of the atomic species comprising the BEC: see [7, 8] for further details.

4.2 Numerical Integration: "Evolution_GPE.py"

The Python code used to evolve a field according to the model of Eq. (7) is given in "Evolution_GPE.py". It takes a similar form to the example of Section 1.2, with some adjustments reflecting the differences between the model and parameters of Eq. (7) compared to those of Eq. (1).

A summary of alterations to the code compared with the procedures described in Section 1.2 are:

- **Parameter Setup:** Minor alterations for updated field and model parameters.
- **Grid Setup and Scalings:** No conversions pertaining to optical fields for scaling.
- **Field Creation:** A new function, 'CACFuncs.TFDist', is called to construct the initial atomic field as a Thomas-Fermi distribution of defined parameters.
- **Pre-Loop Configuration:** The output steps now provide data and plots relating to the model's single atomic field. The information file output is also tweaked accordingly.
- **Field Evolution:** The atomic field is now evolved subject to Eq. (7), so there are slight variations in the RK application. The simulation outputs are adjusted as described in the pre-loop step.
- **Post-Loop Configuration:** No major changes - the simulation outputs are adjusted as described in the pre-loop step.

5 Evolution of an Ultracold Atomic Mix

5.1 Theoretical Model

In this case, we are considering the evolution along the space/time variable ζ of coupled ultracold atomic fields ψ_1 and ψ_2 representing the two atomic species of the mix, i.e. effectively 'turning off' the optical components of Fig. 2, and modelling the free evolution of the medium only. This is modelled by the coupled Gross-Pitaevskii equations [9]:

$$\partial_\zeta \psi_1 = i\nabla_\perp^2 \psi_1 - i\left(\beta_{\mathrm{col},11}|\psi_1|^2 + \beta_{\mathrm{col},12}|\psi_2|^2 - iL_{3,1}|\psi_1|^4\right)\psi, \tag{8}$$

$$\partial_\zeta \psi_2 = i\nabla_\perp^2 \psi_2 - i\left(\beta_{\mathrm{col},22}|\psi_2|^2 + \beta_{\mathrm{col},21}|\psi_1|^2 - iL_{3,2}|\psi_2|^4\right)\psi, \tag{9}$$

where the terms in $\nabla_\perp^2$ represents kinetic energy, the terms in β_{col} encapsulate both interatomic and cross-species (if applicable) collisions, and the terms in L_3 represent three-body atomic loss at high field strengths.

5.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), β_{col} and L_3 are scaled parameters relating to the strength and nature of interatomic and cross-species scattering, and the characteristic features of the atomic species comprising the mix component: see [7, 8] for further details.

5.2 Numerical Integration: "Evolution_GPE_GPE.py"

The Python code used to evolve a field according to the model of Eq. (7) is given in "Evolution_GPE_GPE.py". It takes a similar form to the example of Section 4, with some adjustments reflecting the differences between the model and parameters of Eqs. (8)-(9) compared to those of Eq. (7).

A summary of alterations to the code compared with the procedures described in Section 4 are:

- **Parameter Setup:** All atomic parameters now have two components reflecting the two atomic fields, with the definition of β_{col} (line 19) having four pertaining to interatomic and cross-species scattering.
- **Grid Setup and Scalings:** No conversions pertaining to optical fields for scaling.
- **Field Creation:** The function 'CACFuncs.TFDist' is now called twice to construct the two initial atomic fields as Thomas-Fermi distributions of defined parameters. The maximum value of the atomic field (line 85) is defined as the sum of both components.
- **Pre-Loop Configuration:** The output steps now provide data and plots relating to the model's dual atomic fields. The information file output is also tweaked accordingly.
- **Field Evolution:** The two atomic fields are now evolved subject to Eqs. (8)-(9), so there are slight variations in the RK application. The simulation outputs are adjusted as described in the pre-loop step.
- **Post-Loop Configuration:** No major changes - the simulation outputs are adjusted as described in the pre-loop step.

6 Light Co-Propagating with Ultracold Atoms

6.1 Theoretical Model

In this case, we are considering the evolution along the longitudinal propagation dimension ζ of an optical field A, where the propagation medium is a co-propagating ultracold atomic field ψ. If the fields are assumed to propagate with the same wavevectors, i.e. $k_a \approx k_L$, then this is modelled by coupled nonlinear Schrödinger and Gross-Pitaevskii equations [8]:

$$\partial_\zeta A = i\nabla_\perp^2 A + i\left(\frac{-s|\psi|^2 + \beta_{\mathrm{dd}}|\psi|^4}{1+\sigma_{\mathrm{sat}}|A|^2}\right)A, \tag{10}$$

$$\partial_\zeta \psi = i\nabla_\perp^2 \psi - i\left(s|A|^2 - 2\beta_{\mathrm{dd}}|A|^2|\psi|^2 + \beta_{\mathrm{col}}|\psi|^2 - iL_3|\psi|^4\right)\psi, \tag{11}$$

where in Eq. (10) the term in $\nabla_\perp^2$ represents transverse diffraction of the optical field, the term in s represents dipole focusing or defocusing from the atomic field dependent on s, the sign of the atom-light detuning, the term in $\beta_{\rm dd}$ represents higher-order corrective dipole-dipole focusing forces, and the term in $\sigma_{\rm sat}$ represents optical saturation. In Eq. (11), the term in $\nabla_\perp^2$ represents atomic kinetic energy, the term in s represents dipole focusing or defocusing from the optical field again dependent on s, the term in $\beta_{\rm dd}$ again represents higher-order dipole-dipole focusing, the term in $\beta_{\rm col}$ represents interatomic collisions, and the term in L_3 represents three-body atomic loss at high field strengths.

6.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), the longitudinal domain, ζ, is also scaled by

$$\zeta = \frac{z}{k_L w_s^2}, \tag{12}$$

where k_L represents the optical wavevector and $w_{\rm s}$ represents a scaling beam waist. It follows that propagation to $\zeta = 0.5$ is equivalent to one Rayleigh length of the optical field used for scaling.

The two field amplitudes, A and ψ, are related to their unscaled equivalents, A' and $\Phi_{\rm g}$, through

$$A = \frac{w_s \mu}{2\hbar}\sqrt{\frac{m_{\rm a}}{\hbar|\Delta|}}A', \tag{13}$$

$$\psi = \frac{k_{\rm L} w_s \mu}{2}\sqrt{\frac{1}{\epsilon_0 \hbar|\Delta|}}\Phi_{\rm g}, \tag{14}$$

where μ represents the atomic dipole moment, $m_{\rm a}$ the mass of the atomic species, $\hbar$ the reduced Planck constant, Δ the atom-light detuning, and ϵ_0 the free space permittivity.

The dimensionless parameters of Eqs. (10)-(11) may also be related to physical counterparts through

$$\beta_{\rm dd} = \frac{2}{3k_{\rm L}^2 w_s^2}, \tag{15}$$

$$\beta_{\rm col} = \frac{16\pi\epsilon_0 \hbar a_{\rm gg}|\Delta|}{k_{\rm L}^2 \mu^2}, \tag{16}$$

where $a_{\rm gg}$ represents the ground-state atomic scattering length.

6.2 Numerical Integration: "Propagation_NLSE_GPE.py"

The Python code used to evolve a field according to the model of Eqs. (10)-(11) is given in "Propagation_NLSE_GPE.py". It takes a similar form to the example of Section 1.2, but now considers the coupled evolution of two fields through two PDEs, unlike the single field and PDE of Eq. (1).

A summary of alterations to the code compared with the procedures described in Section 1.2 are:

- **Parameter Setup:** Minor alterations for updated field and model parameters.
- **Grid Setup and Scalings:** Inclusion of $\beta_{\rm dd}$ calculation at line 59 from scaling parameters.
- **Field Creation:** A new function, 'CACFuncs.TFDist', is called to construct the initial atomic field as a Thomas-Fermi distribution of defined parameters.
- **Pre-Loop Configuration:** The output steps now reflect the co-propagation of two fields, with data output for each field, and plots now displayed in a 2×2 grid (atoms left column, light right column). The information file output is also tweaked to capture the additional details of this model.
- **Field Evolution:** Two fields are now co-propagated according to Eqs. (10)-(11), so for each time step double the number of fast Fourier transforms occur, double the number of RK steps, etc.. The simulation outputs are adjusted as described in the pre-loop step.
- **Post-Loop Configuration:** No major changes - the simulation outputs are adjusted as described in the pre-loop step.

7 1D Reduction

It is possible to reduce the model given by Eqs. (10)-(11) to an effective 1D system, which the code "Propagation_NLSE_GPE_1D.py" realises. The theoretical model itself remains unaltered, whilst the numerical implementation is largely as described in Section 6. Alterations are made to the **Grid Setup and Scalings** that reflect the move from a 2D to 1D system, and the **Pre-Loop Configuration** is altered, with plots and data output reflecting the reduced dimensionality. Finally, the **Field Evolution** is tweaked to use 1D fast Fourier transforms, and the k-space steps are accordingly altered. Such a model can, in select configurations, be used as a rapid means of predicting 2D dynamics, but some important features (such as orbital angular momentum, for example) are difficult to capture in 1D.

8 Light Co-Propagating with an Ultracold Atomic Mix

8.1 Theoretical Model

In this case, we are considering the evolution along the longitudinal propagation dimension ζ of an optical field A, where the propagation medium is a co-propagating ultracold atomic mix of coupled states ψ_1 and ψ_2. If the fields are assumed to propagate with the same wavevectors, i.e. $k_a \approx k_L$ (for both atomic states), then this is modelled by coupled nonlinear Schrödinger and Gross-Pitaevskii equations [8–10]:

$$\partial_\zeta A = i\nabla_\perp^2 A + i\left(\frac{-s_1\alpha_{\mathrm{d}1}|\psi_1|^2 - s_2\alpha_{\mathrm{d}2}|\psi_2|^2 + \alpha_{\mathrm{dd}1}|\psi_1|^4 + \alpha_{\mathrm{dd}2}|\psi_2|^4}{1+\sigma_{\mathrm{sat}}|A|^2}\right)A, \tag{17}$$

$$\partial_\zeta \psi_1 = i\nabla_\perp^2\psi_1 - i\left(s_1\beta_{\mathrm{d}1}|A|^2 - \beta_{\mathrm{dd}}|A|^2|\psi_1|^2 - s_1s_2\beta_{\times 12}|A|^2|\psi_2|^2 + \beta_{\mathrm{col}11}|\psi_1|^2 + \beta_{\mathrm{col}12}|\psi_2|^2 - iL_{31}|\psi|^4\right)\psi_1, \tag{18}$$

$$\partial_\zeta \psi_2 = i\nabla_\perp^2\psi_2 - i\left(s_2\beta_{\mathrm{d}2}|A|^2 - \beta_{\mathrm{dd}}|A|^2|\psi_2|^2 - s_2s_1\beta_{\times 21}|A|^2|\psi_1|^2 + \beta_{\mathrm{col}22}|\psi_2|^2 + \beta_{\mathrm{col}21}|\psi_1|^2 - iL_{32}|\psi|^4\right)\psi_2, \tag{19}$$

where in Eq. (17) the term in $\nabla_\perp^2$ represents transverse diffraction of the optical field, the terms in $\alpha_{\mathrm{d}n}$ represent dipole focusing or defocusing from the n^{th} atomic field dependent on s_n, the sign of the atom-light detuning, the terms in $\alpha_{\mathrm{dd}n}$ represent higher-order corrective dipole-dipole focusing forces arising from the n^{th} atomic field, and the term in σ_{sat} represents optical saturation. In Eqs. (18) & (19), the term in $\nabla_\perp^2$ represents atomic kinetic energy, the term in $\beta_{\mathrm{d}n}$ represents dipole focusing or defocusing from the optical field for the n^{th} atomic field again dependent on s_n, the term in β_{dd} represents higher-order dipole-dipole focusing, the term in $\beta_{\times nm}$ represents cross-state dipole-dipole focusing dependent on s_ns_m, the product of the signs of the two atom-light detuning selections, the terms in $\beta_{\mathrm{col}nm}$ represent inter- (when $n = m$) and cross- (when $n \neq m$) atomic collisions, and the term in L_{3n} represents three-body atomic loss at high field strengths in the n^{th} atomic field.

8.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), the longitudinal domain, ζ, is also scaled by

$$\zeta = \frac{z}{k_L w_s^2}, \tag{20}$$

where k_L represents the optical wavevector and w_{s} represents a scaling beam waist. It follows that propagation to $\zeta = 0.5$ is equivalent to one Rayleigh length of the optical field used for scaling.

The field amplitudes, A and ψ_n, are related to their unscaled equivalents, A' and $\Phi_{\mathrm{g}n}$, through

$$A = \frac{k_{\mathrm{L}}}{2\sqrt{2\omega\sqrt{|\Delta_1||\Delta_2|}}}A', \tag{21}$$

$$\psi_n = \frac{m_n v_{\mathrm{a}n} w_s^2 \mu_n^2}{\sqrt{2\epsilon_0\hbar^5|\Delta_n|}}\sqrt[4]{\frac{|\Delta_m|}{|\Delta_n|}}\Phi_{\mathrm{g}n}, \tag{22}$$

where $\hbar\omega = \left(m_n v_{an}^2\right)/2$, $\hbar$ represents the reduced Planck constant, m_n represents the mass of the n^{th} atomic species, v_{an} represents the atomic group velocity of the n^{th} atomic species, Δ_n represents the atom-light detuning for the n^{th} atomic state, μ_n represents the atomic dipole moment of the n^{th} atomic species, and ϵ_0 the free space permittivity.

The dimensionless parameters of Eqs. (17)-(19) may also be related to physical counterparts through

$$\alpha_{\mathrm{d}n} = \frac{\hbar^4 k_{\mathrm{L}}^2}{2m_n^2 v_{\mathrm{a}n}^2 w_s^2 \mu_n^2} \sqrt{\frac{|\Delta_n|}{|\Delta_m|}}, \tag{23}$$

$$\alpha_{\mathrm{dd}n} = \frac{\hbar^8 k_{\mathrm{L}}^2}{6m_n^4 v_{\mathrm{a}n}^4 w_s^6 \mu_n^4} \frac{|\Delta_n|}{|\Delta_m|}, \tag{24}$$

$$\beta_{\mathrm{d}n} = \frac{m_n^2 v_{\mathrm{a}n}^2 w_s^2 \mu_n^2}{\hbar^4 k_{\mathrm{L}}^2} \sqrt{\frac{|\Delta_m|}{|\Delta_n|}}, \tag{25}$$

$$\beta_{\mathrm{dd}} = \frac{2}{3k_{\mathrm{L}}^2 w_s^2}, \tag{26}$$

$$\beta_{\times nm} = \frac{m_n v_{\mathrm{a}n} \mu_n^2}{3k_{\mathrm{L}}^2 w_s^2 m_m v_{\mathrm{a}m} \mu_m^2} \sqrt{\frac{m_n}{m_m}} \frac{|\Delta_m|}{|\Delta_n|}, \tag{27}$$

$$\beta_{\mathrm{col}nm} = \frac{4\pi \left(m_n + m_m\right) \epsilon_0 \hbar^5 a_{nm} |\Delta_m|}{m_m^3 v_{\mathrm{a}m}^2 w_s^2 \mu_m^4} \sqrt{\frac{|\Delta_m|}{|\Delta_n|}}, \tag{28}$$

where a_{nm} represents the ground- (when $n = m$) and cross- (when $n \neq m$) state atomic scattering length.

8.2 Numerical Integration: "Propagation_NLSE_GPE_GPE.py"

The Python code used to evolve a field according to the model of Eqs. (17)-(19) is given in "Propagation_NLSE_GPE_GPE.py". It takes a similar form to the example of Section 6, but now considers the coupled evolution of three fields through three PDEs, unlike the two fields and PDEs of Eq. (10)-(11).

A summary of alterations to the code compared with the procedures described in Section 1.2 are:

- **Parameter Setup:** Minor alterations for updated field and model parameters.
- **Grid Setup and Scalings:** Inclusion of β_{dd} calculation at line 64 from scaling parameters.
- **Field Creation:** 'CACFuncs.TFDist' is now called twice to construct both initial atomic fields as Thomas-Fermi distributions of defined parameters.
- **Pre-Loop Configuration:** The output steps now reflect the co-propagation of three fields, with data output for each field, and plots now displayed in a 3×2 grid (the atomic fields in the left and central columns, with the light in the right column). The information file output is also tweaked to capture the additional details of this model.
- **Field Evolution:** Three fields are now co-propagated according to Eqs. (17)-(19), so for each time step 1.5× the number of fast Fourier transforms occur, 1.5× the number of RK steps, etc.. The simulation outputs are adjusted as described in the pre-loop step.
- **Post-Loop Configuration:** No major changes - the simulation outputs are adjusted as described in the pre-loop step.

Part II
Driven Cavity Systems

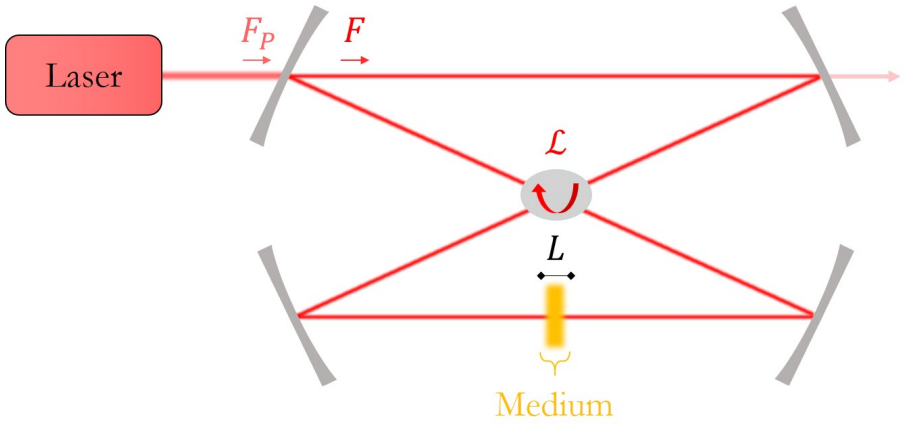

Figure 3: Schematic of a generic driven cavity system, of the kind considered in Part II. A laser produces a coherent beam that acts as a pump, F_P, upon a four-mirror ring / bow-tie style cavity, of round-trip length $\mathcal{L}$. The intra-cavity field, F, encounters a medium once per cavity round trip, of length L. The high-reflectivity mirrors have a small amount of loss per round trip, indicatively marked on the right of the schematic.

9 Light in a Driven Optical Cavity

9.1 Theoretical Model

In this case, we are considering the evolution in scaled time τ of an optical intra-cavity field F that encounters a medium once per cavity round trip, all within a cavity driven by a pump F_P. This system is modelled by the Lugiato-Lefever equation [11]:

$$\partial_\tau F = F_\mathrm{P} - (1 + i\theta)\, F + i\alpha\nabla_\perp^2 F + i\frac{\beta_\mathrm{K}|F|^2}{(1 + \sigma_\mathrm{sat}|F|^2)}F, \tag{29}$$

where the term in θ represents the cavity detuning from resonance, α is a numerical prefactor that scales the term in $\nabla_\perp^2$ representing transverse diffraction of the optical field, the term in β_K represents self-focusing or self-defocusing, depending on the sign of β_K, due to a Kerr nonlinearity of strength $|\beta_\mathrm{K}|$, and the term in σ_sat represents optical saturation.

9.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), the temporal domain, τ, is also scaled by

$$\tau = \frac{cT}{2\mathcal{L}}\left(t + \left[\frac{\mathcal{L} - L}{c}\right]\frac{z}{L}\right), \tag{30}$$

where c represents the speed of light, T the mirror transmittivity within the cavity, $\mathcal{L}$ the cavity length, and L the length of the medium. In addition, α, β_K, and θ are all scaled parameters relating to coefficients of the system: see [11] for further details.

9.2 Numerical Integration: "Cavity_LLE.py"

The Python code used to evolve the intra-cavity field according to the model of Eq. (29) is given in "Cavity_LLE.py". It takes a similar form to the example of Section 1.2, but with notable differences given the different physical configuration being modelled.

A summary of alterations to the code compared with the procedures described in Section 1.2 are:

- **Parameter Setup:** Minor alterations for updated field and model parameters.
- **Field Creation:** The new function 'CACFuncs.Noisy' is called to construct the initial intra-cavity field as noise (line 92).
- **Pre-Loop Configuration:** The output steps now reflect the updated physical configuration, with plots now displayed in a 2×2 grid (intra-cavity left column, pump right column), and data output initially for both fields also. The information file output is also tweaked to capture the additional details of this model.
- **Field Evolution:** The intra-cavity field is now evolved according to Eq. (29). The simulation outputs are adjusted, as described in the pre-loop step for plotting, and output data files for only the intra-cavity field.
- **Post-Loop Configuration:** No major changes - the simulation outputs are adjusted as described in the field evolution step.

10 Light and Ultracold Atoms in a Driven Optical Cavity

10.1 Theoretical Model

In this case, we are considering the evolution in scaled time τ of an optical intra-cavity field F coupled to an ultracold atomic field ψ, both within a cavity driven by a pump F_{P}. This system is modelled by coupled Lugiato-Lefever and Gross-Pitaevskii equations [12, 13]:

$$\partial_\tau F = F_{\mathrm{P}} - (1 + i\theta)\, F + i\alpha_F \nabla_\perp^2 F - i\frac{\beta_F(s|\psi|^2 - \beta_{\mathrm{dd}}|\psi|^4)}{(1 + \sigma_{\mathrm{sat}}|F|^2)} F, \tag{31}$$

$$\partial_\tau \psi = \frac{\alpha_\psi}{\kappa}\left[i\nabla_\perp^2 \psi - i\Big(s|F|^2 - 2\beta_{\mathrm{dd}}|F|^2|\psi|^2 + \beta_{\mathrm{col}}|\psi|^2 - iL_3|\psi|^4 \Big)\psi \right], \tag{32}$$

where in Eq. (31) the term in θ represents the cavity detuning from resonance, α_{F} is a numerical prefactor that scales the term in $\nabla_\perp^2$ representing transverse diffraction of the optical field, β_{F} is a numerical prefactor that scales the nonlinear terms: namely the term in s, which represents dipole focusing or defocusing from the atomic field dependent on s, the sign of the atom-light detuning, the term in β_{dd}, which represents higher-order corrective dipole-dipole focusing forces, and the term in σ_{sat} represents optical saturation. In Eq. (32), the numerical prefactor α_ψ/κ, where $\kappa = (cT)/(2\mathcal{L})$, scales all terms: the term in $\nabla_\perp^2$ represents atomic kinetic energy, the term in s represents dipole focusing or defocusing from the optical field again dependent on s, the term in β_{dd} again represents higher-order dipole-dipole focusing, the term in β_{col} represents interatomic collisions, and the term in L_3 represents three-body atomic loss at high field strengths.

10.1.1 Scalings

In addition to the transverse domain scaling (Eq. (2)), the temporal domain, τ, is also scaled by

$$\tau = \frac{cT}{2\mathcal{L}}\left(t + \left[\frac{\mathcal{L} - L}{c}\right]\frac{z}{L}\right), \tag{33}$$

where c represents the speed of light, T the mirror transmittivity within the cavity, $\mathcal{L}$ the cavity length, and L the length of the atomic medium.

The field amplitudes F, F_{P}, and ψ, are related to their unscaled equivalents, A', F'_{P}, and Φ_{g}, through

$$F = \frac{w_{\mathrm{s}}\mu}{2\hbar}\sqrt{\frac{m_{\mathrm{a}}}{\hbar|\Delta|}}A', \tag{34}$$

$$F_P = 2w_{\mathrm{s}}\sqrt{\frac{m_{\mathrm{a}}}{\hbar T}}F'_P, \tag{35}$$

$$\psi = \frac{w_{\mathrm{s}}k_{\mathrm{L}}\mu}{2n}\sqrt{\frac{1}{\epsilon_0\hbar|\Delta|}}\Phi_{\mathrm{g}}, \tag{36}$$

where μ represents the atomic dipole moment, $m_{\rm a}$ the mass of the atomic species, $\hbar$ the reduced Planck constant, Δ the atom-light detuning, k_L the optical wavevector, n the refractive index of the atomic medium, and ϵ_0 the free space permittivity.

The dimensionless parameters of Eqs. (31)-(32) may also be related to physical counterparts through

$$\theta = \frac{2\left(\omega_{\rm c} - \omega_{\rm P}\right)\mathcal{L}}{cT}, \tag{37}$$

$$\alpha_F = \frac{2\mathcal{L}'}{k_{\rm L} w_{\rm s}^2 T}, \tag{38}$$

$$\beta_F = \frac{2L}{k_{\rm L} w_{\rm s}^2 T}, \tag{39}$$

$$\beta_{\rm dd} = \frac{2}{3k_{\rm L}^2 w_s^2}, \tag{40}$$

$$\alpha_\psi = \frac{\hbar}{m_{\rm a} w_{\rm s}^2}, \tag{41}$$

$$\beta_{\rm col} = \frac{16\pi\epsilon_0 \hbar a_{\rm gg}|\Delta|}{k_{\rm L}^2 \mu^2}, \tag{42}$$

where $\omega_{\rm c}$ represents the longitudinal cavity mode closest to the input frequency $\omega_{\rm P}$, $\mathcal{L}'$ offers the potential to re-set the optical diffraction length using an intra-cavity telescope, and $a_{\rm gg}$ represents the ground-state atomic scattering length.

10.2 Numerical Integration: "Cavity_LLE_GPE.py"

The Python code used to evolve a field according to the model of Eqs. (31)-(32) is given in "Cavity_LLE_GPE.py". It takes a similar form to the example of Section 1.2, but now considers the coupled evolution of two fields through two PDEs, unlike the single field and PDE of Eq. (1).

A summary of alterations to the code compared with the procedures described in Section 1.2 are:

- **Parameter Setup:** Minor alterations for updated field and model parameters.
- **Grid Setup and Scalings:** Inclusion of $\beta_{\rm dd}$ calculation at line 67 from scaling parameters.
- **Field Creation:** Two new functions are used: 'CACFuncs.TFDist' is called to construct the initial atomic field as a Thomas-Fermi distribution of defined parameters, and 'CACFuncs.Noisy' is called to construct the initial intra-cavity field as noise.
- **Pre-Loop Configuration:** The parameters used in Eq. 32 are scaled by α_ψ/κ in lines 124-130: these updated parameters are used at the appropriate RK steps. The output steps now reflect the co-evolution of two fields, dependent on a third, with plots now displayed in a 3×2 grid (atoms left column, light intra-cavity centre column, optical pump right column), and data output initially for all three fields also. The information file output is also tweaked to capture the additional details of this model.
- **Field Evolution:** Two fields are now co-evolved according to Eqs. (31)-(32), so for each time step double the number of fast Fourier transforms occur, double the number of RK steps, etc.. The simulation outputs are adjusted, as described in the pre-loop step for plotting, and output data files for both the atomic and intra-cavity field.
- **Post-Loop Configuration:** No major changes - the simulation outputs are adjusted as described in the field evolution step.

11 1D Reduction

It is possible to reduce the model given by Eqs. (31)-(32) to an effective 1D system, which the code "Cavity_LLE_GPE_1D.py" realises. The theoretical model itself remains unaltered, whilst the numerical implementation is largely as described in Section 10. Alterations are made to the **Grid Setup and Scalings** that reflect the move from a 2D to 1D system, and the **Pre-Loop Configuration** is altered, with plots and data output reflecting the reduced dimensionality. Finally, the **Field Evolution** is tweaked

to use 1D fast Fourier transforms, and the k-space steps are accordingly altered. Such a model can, in select configurations, be used as a rapid means of predicting 2D dynamics, but some important features (such as orbital angular momentum, for example) are difficult to capture in 1D.

12 Eliminated Intra-Cavity Dynamics

In this section, we recognise that typical parameters associated with the model of Section 10 lead to two effective evolutionary time scales: a fast optical time scale of order τ, and a slow atomic time scale of order α_ψ/κ. These two time scales make the model of Eqs. (31)-(32) numerically inefficient, and we thus consider here two approaches to bypassing this inefficiency.

12.1 Finite Grid Method

By setting $\partial_\tau F = 0$ and $\sigma_{\rm sat} = 0$ in Eq. (31), a steady-state intra-cavity field F_s, for a given static ψ', is given by

$$F_{\rm P} = (1 + i\theta)\, F_{\rm s} - i\alpha_F \nabla_\perp^2 F_{\rm s} + i\beta_F \left(s|\psi'|^2 - \beta_{\rm dd}|\psi'|^4\right) F_{\rm s}, \tag{43}$$

or, equivalently, that

$$\boldsymbol{F_P} = \boldsymbol{\gamma F_{\rm s}}, \tag{44}$$

where $\boldsymbol{\gamma}$ represents the coefficients of all interactions.

12.1.1 2D Theoretical Model

Writing the Laplacian as a discrete operator on a grid, given by

$$\partial_{x,y}^2 F = \frac{F_{x-1,y} + F_{x,y-1} - 4F_{x,y} + F_{x+1,y} + F_{x,y+1}}{(dx)^2}, \tag{45}$$

where the grid is assumed to be identical in x and y, of grid spacing $dx \equiv dy$, Eq. (43) becomes a series of coupled linear equations in the sole unknown $F_{\rm s}$. For this configuration, it follows that we may write the non-zero elements of the coefficient tensor $\boldsymbol{\gamma}$ for any given $F_{\rm s}(x, y)$ value as

$$\boldsymbol{\gamma_{x-1,y,x,y}} = -i\frac{\alpha_F}{(dx)^2}, \tag{46}$$

$$\boldsymbol{\gamma_{x,y-1,x,y}} = -i\frac{\alpha_F}{(dx)^2}, \tag{47}$$

$$\boldsymbol{\gamma_{x,y,x,y}} = (1 + i\theta) + i\frac{4\alpha_F}{(dx)^2} + i\beta_F \left(s|\psi'_{x,y}|^2 - \beta_{\rm dd}|\psi'_{x,y}|^4\right), \tag{48}$$

$$\boldsymbol{\gamma_{x,y+1,x,y}} = -i\frac{\alpha_F}{(dx)^2}, \tag{49}$$

$$\boldsymbol{\gamma_{x+1,y,x,y}} = -i\frac{\alpha_F}{(dx)^2}. \tag{50}$$

After its evaluation, the solution for $F_{\rm s}$ is used to evolve the atomic field at the re-scaled atomic slow time $\tau_{\rm s}$ through a re-writing of Eq. (32) to

$$\partial_{\tau_{\rm s}}\psi = i\nabla_\perp^2\psi - i\Big(s|F_{\rm s}|^2 - 2\beta_{\rm dd}|F_{\rm s}|^2|\psi|^2 + \beta_{\rm col}|\psi|^2 - iL_3|\psi|^4\Big)\psi. \tag{51}$$

12.1.2 2D Numerical Integration: "Cavity_Elim_FiniteGrid_LLE_GPE.py"

The Python code used to evolve a field according to the eliminated model of Eqs. (45)-(51) is given in "Cavity_Elim_FiniteGrid_LLE_GPE.py". A summary of alterations to the code compared with the procedures described in Section 10 are:

- **Parameter Setup:** The definition of the final run time (line 11) is now in terms of the slow atomic time, $\tau_{\rm s}$.
- **Pre-Loop Configuration:** The atomic parameters are no longer scaled by α_ψ/κ. The information file output is also tweaked according to the updated numerical method.
- **Field Evolution:** Only the atomic field now undergoes fast Fourier transforms, an RK procedure, etc.. A new function, 'CACFuncs.Elim_FiniteGrid' is called to setup and solve the tensor system described above for each previous real-space RK optical step (lines 174 and 177).

12.1.3 1D Theoretical Model

Building on the reduced 1D model of Section 11, it is also possible to perform a finite grid-based approximation on this system. Again writing the Laplacian as a discrete operator:

$$\partial_x^2 F = \frac{F_{x-1} - 2F_x + F_{x+1}}{(dx)^2}, \tag{52}$$

Eq. (43) again becomes a series of coupled linear equations in the sole unknown F_s. In this 1D case, the non-zero elements of the coefficient matrix $\boldsymbol{\gamma}$ for any given $F_\mathrm{s}(x)$ value are given by

$$\boldsymbol{\gamma_{x-1,x}} = -i\frac{\alpha_F}{(dx)^2}, \tag{53}$$

$$\boldsymbol{\gamma_{x,x}} = (1+i\theta) + i\frac{2\alpha_F}{(dx)^2} + i\beta_F\left(s|\psi'_x|^2 - \beta_\mathrm{dd}|\psi'_x|^4\right), \tag{54}$$

$$\boldsymbol{\gamma_{x+1,x}} = -i\frac{\alpha_F}{(dx)^2}. \tag{55}$$

After its evaluation, the solution for F_s is used to evolve the atomic field at the re-scaled atomic slow time τ_s through the same re-writing of the atomic evolution of Eq. (51).

12.1.4 1D Numerical Integration: "Cavity_Elim_FiniteGrid_LLE_GPE_1D.py"

The Python code used to evolve a field according to the eliminated model of Eqs. (52)-(55) & (51) is given in "Cavity_Elim_FiniteGrid_LLE_GPE_1D.py". A summary of alterations to the code compared with the procedures described in Section 11 are:

- **Parameter Setup:** The definition of the final run time (line 11) is now in terms of the slow atomic time, τ_s.
- **Pre-Loop Configuration:** The atomic parameters are no longer scaled by α_ψ/κ. The information file output is also tweaked according to the updated numerical method.
- **Field Evolution:** Only the atomic field now undergoes a fast Fourier transforms, RK procedure, etc.. A new function, 'CACFuncs.Elim_FiniteGrid' is called to setup and solve the tensor system described above for each previous real-space RK optical step (lines 169 and 172).

12.2 Relaxation Method

In this approach to de-coupling the two time scales, we consider the independent evolution of Eq. (31) to steady state in order to obtain a solution for the intra-cavity field at each evolution step, which is then applied to Eq. (32) to progress the atoms at their slow time scale.

12.2.1 Theoretical Method

Splitting of the two system time scales means that Eqs. (31)-(32) may effectively be written as

$$\partial_\tau F = F_\mathrm{P} - (1+i\theta)F + i\alpha_F\nabla_\perp^2 F - i\frac{\beta_F\left(s|\psi'|^2 - \beta_\mathrm{dd}|\psi'|^4\right)}{(1+\sigma_\mathrm{sat}|F|^2)}F, \tag{56}$$

$$\partial_{\tau_\mathrm{s}}\psi = i\nabla_\perp^2\psi - i\Big(s|F'|^2 - 2\beta_\mathrm{dd}|F'|^2|\psi|^2 + \beta_\mathrm{col}|\psi|^2 - iL_3|\psi|^4\Big)\psi, \tag{57}$$

where the factor α_ψ/κ has been incorporated into a newly re-scaled slow time τ_s introduced in Eq. (57), and F' represents the steady-state intra-cavity result of Eq. (31) for any $d\tau$ step's fixed ψ'.

12.2.2 Numerical Integration: "Cavity_Elim_Relax_LLE_GPE.py"

The Python code used to evolve a field according to the eliminated model of Eqs. (56)-(57) is given in "Cavity_Elim_Relax_LLE_GPE.py". A summary of alterations to the code compared with the procedures described in Section 10 are:

- **Parameter Setup:** The definition of the final run time (line 11) is now in terms of the slow atomic time, τ_s.

- **Pre-Loop Configuration:** The atomic parameters are no longer scaled by α_ψ/κ. The information file output is also tweaked according to the updated numerical method.
- **Field Evolution:** Only the atomic field now undergoes a fast Fourier transforms, RK procedure, etc. within the main file. A new function, 'CACFuncs.Elim_Relax' is called to evolve the fast intra-cavity field to steady state (defined as differences in the field's amplitude remaining beneath 10^{-5} over ten $d\tau$ steps) for each previous real-space RK optical step (lines 174 and 177).

Part III
Miscellaneous Codes and Functions

13 Scaling Calculators

Two codes are provided to help convert 'physical' parameter selections to 'numerical' values used within the propagation models described:

- "Propagation_NLSE_GPE_Parameters.py" is tailored to the co-propagating single optical and single atomic field model described in Section 6, with all calculated parameters therefore matching the descriptions given by Eqs. (10)-(16).
- "Propagation_NLSE_GPE_GPE_Parameters.py" is tailored to the co-propagating single optical and dual species atomic mix model described in Section 8, with all calculated parameters therefore matching the descriptions given by Eqs. (17)-(28).

A further code is provided to help convert 'physical' parameter selections to 'numerical' values used within a cavity model described:

- "Cavity_LLE_GPE_Parameters.py" is tailored to the light and ultracold atoms in a driven optical cavity model described in Section 10, with all calculated parameters therefore matching the descriptions given by Eqs. (31)-(42).

Note that in each case the current input parameters are random numbers at the indicative orders of magnitude expected of realistic experimental selections, so the outputs will vary on each calculation.

14 Dynamic (Space)Time Step

One '∼issue' with the codes above that has always troubled me is the requirement of the operator to actively define a value for $d\zeta$ / $d\tau$ / $d\tau_{\rm s}$ (as appropriate). In my opinion, best practice should have the code, after parameter input, run entirely 'hands-off' with *guaranteed* numerical convergence: an actively defined step size is clearly at odds with this. As you may note in the codes above, this is eliminated to an extent: step sizes are defined in factors of $(dx)^2/n$, where $n > \pi$. However, it is still perfectly conceivable that codes will experience numerical breakdowns: with fields that can have amplitudes significantly greater than 1, squared (or higher order terms) can quickly act to overwhelm the selected step size.

I have therefore spent some time recently implementing an adaptive Runge-Kutta-Fehlberg 45 (RKF45) method [14], which incorporates a dynamically calculated (space)time step, into the numerical methods described above. Though typically, but not always slower than the methods described above, there are significant benefits of using these methods: (i) the simulations apply a higher-order Runge-Kutta method than those previously described; (ii) in conditions where a larger step size is permissible, it is implemented automatically; and (iii) the maximum permitted truncation error in the simulations (defined $\epsilon_{T\,\rm max}$ is quantified and user-defined, i.e. greater certainty in numerical convergence is attained from every run.

I therefore include here three outline codes that implement a dynamic step size into the most commonly used propagation (Section 6), fully dynamical cavity (Section 10), and eliminated intra-cavity (Section 12.2) codes. The principal difference from their equivalents above comes in the evaluation of the real space nonlinearities: rather than the previous RK2 application, an RKF45 method is now implemented. This method follows a Butcher Tableau of [14]

0						
$\frac{1}{4}$	$\frac{1}{4}$					
$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$				
$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$			
1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$		
$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$	
RKF4:	$\frac{25}{216}$	0	$\frac{1408}{2565}$	$\frac{2197}{4104}$	$-\frac{1}{5}$	0
RKF5:	$\frac{16}{135}$	0	$\frac{6656}{12825}$	$\frac{28561}{56430}$	$-\frac{9}{50}$	$\frac{2}{55}$

where the penultimate and ultimate lines represent the coefficients of the RKF4 and RKF5 methods, respectively. An estimation of the truncation error in the method, ϵ_{T}, may then be obtained by calculating the absolute value of the difference between the RKF4 and RKF5 methods, and summing over all orders one to six (i.e. the result of the final two rows of the table). An updated time step, dt_{new} may then be calculated as

$$dt_{\mathrm{new}} = 0.9\ dt\ \left(\frac{\epsilon_{\mathrm{T}\,\mathrm{max}}}{\epsilon_{\mathrm{T}}}\right)^{\frac{1}{5}} \tag{58}$$

where dt was the step size used to attain the RKF4, RKF5, and truncation error ϵ_{T} values, and $\epsilon_{\mathrm{T}\,\mathrm{max}}$ is the user-defined maximum permitted truncation error. Following this procedure, the outcome can go two ways: if $\epsilon_{\mathrm{T}} \leq \epsilon_{\mathrm{T}\,\mathrm{max}}$, the (more accurate) RKF5 solution is accepted and the loop continues before, at the end of the entire current step (i.e. after another half step in Fourier space), the step size is updated to the more optimal dt_{new} through Eqn. (58). Instead, if $\epsilon_{\mathrm{T}} > \epsilon_{\mathrm{T}\,\mathrm{max}}$, the attained solution is rejected, and the current iteration of the loop continues no further. The step size is instead reduced to dt_{new} according Eqn. (58), and the current loop iteration starts again with the new, smaller step size.

The various codes implementing these methods are:

- "Propagation_NLSE_GPE_DTS.py" implements this dynamic step size method to the co-propagating light and ultracold atomic field model described in Section 6. The maximum permitted truncation error is set in line 88, before the majority of changes occur between lines 162-222: the internal loop ensuring truncation error satisfaction is setup between lines 162-165, the individual order evaluations for the RKF45 algorithm occur between lines 175-201, the RKF4 and RKF5 solutions are evaluated between lines 203-207, the truncation error is estimated between lines 209-211, and $\partial\zeta$ is re-calculated between lines 213-218. If the solution is accepted, then the principal SSFM loop continues with the RKF5 solutions accepted between lines 220-222, before the updated $\partial\zeta_{\mathrm{new}}$ is applied in line 237.

- "Cavity_LLE_GPE_DTS.py" implements this dynamic step size method to the to the light and ultracold atoms in a driven optical cavity model described in Section 10. The maximum permitted truncation error is set in line 92, the internal loop ensuring truncation error satisfaction is setup between lines 178-181, the individual order evaluations occur between lines 191-223, the RKF4 and RKF5 solutions are evaluated between lines 225-229, the truncation error is estimated between lines 231-233, and $\partial\tau$ is re-calculated between lines 235-240. If the solution is accepted, then the RKF5 solutions are accepted between lines 242-244, before $\partial\tau_{\mathrm{new}}$ is applied in line 259.

- "Cavity_Elim_Relax_LLE_GPE_DTS.py" implements this dynamic step size method to the to the light and ultracold atoms in a driven optical cavity model, with eliminated intra-cavity dynamics realised through a relaxation method, described in Section 12.2. In doing so, it uses the new function 'CACFuncs.Elim_Relax45' to implement the RKF45 method whilst relaxing the intra-cavity field. For both fields, the maximum permitted truncation error is set in line 92, before for the atoms the internal loop ensuring truncation error satisfaction is setup between lines 169-171, the individual order evaluations occur between lines 179-199, the RKF4 and RKF5 solutions are evaluated between lines 201-203, the truncation error is estimated between lines 205-206, and $\partial\tau_{\mathrm{s}}$ is re-calculated between lines 208-213. If the solution is accepted, then the RKF5 solution is accepted between lines 215-217, before $\partial\tau_{\mathrm{s}_{\mathrm{new}}}$ is applied in line 230.

Depending on the field amplitude and nonlinearity selections, therefore, one may find that a dynamic step simulation is simultaneously faster, and is certainly more assured of numerical convergence (to truncation error $\epsilon_{\mathrm{T}\max}$), than a fixed step size equivalent.

15 Alternative Field Profiles

A few additional functions are provided, in addition to those introduced within the earlier sections of this document, to create useful initial field profiles. They, as standard, are contained within 'FOAMily-CAC.py'.

- "Homogeneous(X, Y, amp, noise_amplitude)" creates a homogeneous (planar) field on an (X, Y) grid. The field is scaled to have a maximum amplitude of 'amp', and then has complex noise applied, of amplitude 'noise_amplitude'.
- "TopHat(X, Y, amp, x, w_F, N_w, w_L, l_mode, grad, noise_amplitude)" creates a homogeneous 'top hat'-like profile, of waist size w_{F} and scaled gradient 'grad', on an (X, Y) grid. It optionally has OAM applied, of order 'ℓ_mode' - with this term set to zero, it will have homogeneous phase. The field is scaled to have a maximum amplitude of 'amp', and has complex noise applied, of amplitude 'noise_amplitude'.
- "BGMode(X, Y, w_F, w_L, l_mode, OAM, gaus_width, kappa, amp, noise_amplitude)" creates a Bessel-Gaussian mode, of Bessel waist size w_{F}, width controlled by κ, and Gaussian width controlled by 'gaus_width' (relative to w_F), on an (X, Y) grid. It optionally has OAM applied (if 'OAM' is set to 1), of order 'ℓ_mode'. The field is scaled to have a maximum amplitude of 'amp', and has complex noise applied, of amplitude 'noise_amplitude'.

Note that, if desired, these functions also work for generating 1D fields: just follow the implementation examples of Sections 7 & 11 if required in this configuration.

References

[1] P. Suarez, "An introduction to the Split Step Fourier Method using MATLAB", 10.13140/RG.2.1.1394.0965 (2015).

[2] A. M. Yao and M. J. Padgett, "Orbital angular momentum: origins, behavior and applications", Adv. Opt. Photon. **3**, 161–204 (2011).

[3] S. M. Barnett and R. Zambrini, "Orbital Angular Momentum of Light", in *Quantum imaging*, edited by M. I. Kolobov (Springer, New York, 2007), pp. 277–311.

[4] For further information, see e.g.: O. Henrich, "Computational Physics: An Introduction to Numerical Algorithms with Applications to Physics", Undergraduate Lecture Notes, University of Strathclyde (2024).

[5] R. W. Boyd, *Nonlinear Optics* (Academic press, 2020).

[6] W. J. Firth and D. V. Skryabin, "Optical Solitons Carrying Orbital Angular Momentum", Phys. Rev. Lett. **79**, 2450–2453 (1997).

[7] C. Foot, *Atomic physics* (Oxford University Press, UK, 2005).

[8] G. W. Henderson, G. R. M. Robb, G.-L. Oppo, and A. M. Yao, "Control of Light-Atom Solitons and Atomic Transport by Optical Vortex Beams Propagating through a Bose-Einstein Condensate", Phys. Rev. Lett. **129**, 073902 (2022).

[9] K. E. Wilson, A. Guttridge, I.-K. Liu, J. Segal, T. P. Billam, N. G. Parker, N. P. Proukakis, and S. L. Cornish, "Dynamics of a degenerate cs-yb mixture with attractive interspecies interactions", Phys. Rev. Res. **3**, 033096 (2021).

[10] G. W. Henderson, *A Model To Describe a Co-Propagating Optical Field and Ultracold Atomic Mix*, 2025.

[11] L. A. Lugiato and R. Lefever, "Spatial Dissipative Structures in Passive Optical Systems", Phys. Rev. Lett. **58**, 2209–2211 (1987).

[12] G. W. Henderson, G. R. M. Robb, G.-L. Oppo, and A. M. Yao, "Self-Organization of Ultracold Atomic-Light Lattices in a Driven Optical Cavity", in preparation, 2025.

[13] G. W. Henderson, G. R. M. Robb, G.-L. Oppo, and A. M. Yao, "Ultracold Atomic Currents from Structured Light in a Driven Optical Cavity", in preparation, 2025.

[14] E. Hairer, S. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff problems*, Second (Springer, Berlin, 2000).