Computer Exercises for Lecture "Computational Photonics" **(Date: 2022-05-31)**

Exercise 04 - The nonlinear Schrödinger equation

Part II

Before you start working on the assignments and before you hand in a solution, read the instructions below.

- You can work on these assignments in groups of up to 3 students. Note, however, that everybody has to hand in his/her own solution.
- Please personalize your completed script `main_ex04_part2_ID.py` by replacing the handle "`ID`" by your student-id. For redundancy, make sure to also include your name and student-id in the completed scripts that you hand in. We therefore provided three metadata variables

```
__STUDENT_NAME__   = "YOUR_NAME_HERE"
__STUDENT_ID__     = "YOUR_STUDENT_ID_HERE"
__GROUP_MEMBERS__ = "YOUR_TEAM_HERE"
```

  at the beginning of the script.
- To complete the code, look out for TODO statements, they will provide further hints!
- Make sure to hand in the completed script into the assignment folder `Exercise04_part2/Solutions`.

Problem 2: Numerical solution of the higher-order NSE (7 + 3 = 10 points)

In the second part of exercise 4 we will be concerned by a variant of the NSE that includes higher orders of dispersion, referred to as the higher-order nonlinear Schrödinger equation (HONSE). The respective initial-boundary value problem associated to the HONSE takes the form

$$\begin{cases} \partial_z A(z,t) = i\sum_{n=2}^{N} \frac{i^n}{n!}\beta_n \partial_t^n A(z,t) + i\gamma|A(z,t)|^2 A(z,t), & z \geq 0, \ -t_{\max} \leq t \leq t_{\max}, \\ A(z,-t_{\max}) = A(z,t_{\max}), & z \geq 0, \\ A(z,t)|_{z=0} = A_0(t), & -t_{\max} \leq t \leq t_{\max}. \end{cases} \tag{1}$$

As in the previous case, the field $A(z,t)$ describes the complex valued slowly varying envelope of an optical pulse, z signifies the position along the fiber, and t is the time in the local frame of reference moving with the pulse. In terms of "complexity", the HONSE resides in between the NSE and the gernalized NSE (GNSE), discussed in lecture 6. Similar to the NSE its nonlinear part takes into account the Kerr-effect through the nonlinear coefficient γ. Similar to the GNSE its linear part includes higher orders of dispersion through the dispersion coefficients β_n, $n = 2, \ldots, N$. Again, we will consider initial conditions of the form

$$A_0(t) = \sqrt{P_0}\ \mathrm{sech}\left(\frac{t}{t_0}\right), \tag{2}$$

with pulse peak power P_0 and a hyperbolic secant shape of duration t_0. Download the file `main_ex04_part2_ID.py` from folder `Exercise04_part2` on StudIp. Wade through the code, and consider the problems below.

(a) Consider the function `main_a()`. It specifies simulation parameters for the numerical solution of the HONSE, including linear dispersion terms up to order $N = 4$. Also, in its current state, it initializes a fundamental soliton as initial condition for the propagation routine `SSFM_HONSE_symmetric()` imported from the module `split_step_solver.py`. Note that this function does not exist, yet. Hence, the script will not run as is. Instead, it will terminate with an error message of the form

```
ImportError: cannot import name 'SSFM_HONSE_symmetric' from 'split_step_solver'
```

Follow the steps below to complete your code and to amend it so that it solves the HONSE.

Step 1: Work in the module `split_step_solver.py`. Follow the steps below to prepare the module for your implementation of the HONSE propagation routine.

1. Make a copy of the function `SSFM_NSE_symmetric()`, i.e. the function you amended during part I of the exercise, and rename it to `SSFM_HONSE_symmetric()`.
2. Have a look at the function call in the script `main_ex04_part2_ID.py`, reading

   ```
   z, Azt = SSFM_HONSE_symmetric(_z, t, A0, beta2, beta3, beta4, gamma, nSkip)
   ```

   As the function is implemented using positional arguments, it is important to note in which order the parameters are passed to the function. In particular, it is important for you to note where the parameters `beta3` and `beta4` enter. Amend the function definition in the file `split_step_solver.py` so that you can work with these values in the body of the function `SSFM_HONSE_symmetric()`.

Step 2: Work out the numerical method that details how to use the values of `beta3` and `beta4` in order to implement the field update $A(z,t) \to A(z+\Delta z, t)$ for the HONSE. Before you implement anything, take a sheet of paper and figure it out in written form. Follow the steps below:

1. First, it is important to note how $\mathcal{F}$ and $\mathcal{F}^{-1}$ are to be interpreted. Here, in order to be consistent with lecture 6 and the literature, it is important to note that

$$A_\omega(z) = \mathcal{F}[A(z,t)] \equiv \int A(z,t)\; e^{i\omega t}\; \mathrm{dt}, \tag{3}$$

$$A(z,t) = \mathcal{F}^{-1}[A_\omega(z)] \equiv \frac{1}{2\pi}\int A_\omega(z)\; e^{-i\omega t}\; \mathrm{d}\omega. \tag{4}$$

   This implies that the operation we expect $\mathcal{F}$ to perform is, in terms of a DFT, provided by `numpy`'s function `numpy.fft.ifft`. Likewise, a proper technical basis to realize $\mathcal{F}^{-1}$ is `numpy.fft.fft`. Be aware that this is different than in exercise 3. In exercise 3 we simply studied the DFT *on its own*. Here, we instead *use* the DFT as *technical basis* to realize a particular form of the Fourier-transform which is specified as part of the model! This has consequences for the spectral derivative. As evident from Eq. (4), t-derivatives transform to multiplications in ω-space according to the rule $\partial_t^n e^{-i\omega t} = (-i\omega)^n e^{-i\omega t}$. Consequently, spectral derivatives of the field $A(z,t)$ are now given by

$$\partial_t^n A(z,t) = \mathcal{F}^{-1}\left[(-i\omega)^n \mathcal{F}\left[A(z,t)\right]\right]. \tag{5}$$

2. Now, consider the linear operator on the right-hand-side (rhs) of the HONSE restricted to $N=4$ (cf. slides 13 and 28 of lecture 6), given by

$$\hat{D}(\partial_t) = i\sum_{n=2}^{4}\frac{i^n}{n!}\beta_n\partial_t^n = -\frac{i}{2}\beta_2\partial_t^2 + \frac{1}{6}\beta_3\partial_t^3 + \frac{i}{24}\beta_4\partial_t^4, \tag{6}$$

   From part I of this exercise you already know (even if you did not realize it at that time), that the action of the first term on the rhs of Eq. (6) on the field $A(z,t)$ reads

$$-\frac{i}{2}\beta_2\partial_t^2 A(z,t) = \mathcal{F}^{-1}\left[\frac{i}{2}\beta_2\omega^2\mathcal{F}\left[A(z,t)\right]\right]. \tag{7}$$

   Carry out the calculations that tell how the second and third term on the rhs of Eq. (6) act on $A(z,t)$. Therefore, make use of the spectral derivative Eq. (5).

Step 3: Review slides 13 and 28 of lecture 6 to see how to amend the function `SSFM_HONSE_symmetric()` to account for the higher orders of dispersion included in Eq. (6).

Step 4: As initial condition, use a NSE soliton of order $N_s = 1.4$. This will allow you to reproduce Fig. 1.

Step 5: Upload your completed module `split_step_solver.py` to the folder `Exercise04_solutions` (or discuss your results with one of the tutors).

Hints:

– As technical basis one could also use a different Fourier-transform pair than the one defined by Eqs. (3-4). In principle this is possible, see e.g. this article, where a different transform pair was used. However, note that if you go to dispersion orders $N \geq 3$ this then complicates an interpretation of the resulting spectra.

– In order to support you in assessing the correctness of your implementation, Fig. 1 shows the z-propagation evolution of $A(z,t)$ for the pre-set parameters in `main_a()`.

(b) Perform a series of numerical experiments using the HONSE solver. The preset parameters in `main_a()` describe realistic fiber parameters for a fused silica fiber and realistic pulse parameters. This parameter set is frequently used in the literature to demonstrate the performance of propagation algorithms, see, e.g., here. Thus, these parameters allow for a realistic verification test.

Now, modify these parameters to explore the effect of higher-order dispersion on the z-evolution of the spectrum. Therefore, in addition to β_2 it is actually sufficient to consider a nonzero value of β_3, only. Hence, set `beta4=0` for the subsequent simulation runs.

1. Perform two simulation runs. For the first simulation run use the pre-set value of `beta3`. For the second simulation run change the sign of `beta3`. Compare the resulting spectra for both cases. What do you observe? Include a short answer as string variable `ANSWER_b1`.
2. Change the initial condition to represent a NSE soliton of order $N_s = 2$. What do you observe? Is the resulting z-evolution of $A(z,t)$ different from the evolution under the NSE (i.e. for the case `beta3=0`, `beta4=0`)? Include a short answer as string variable `ANSWER_b2`.

Hints:

– Note that the propagation routine does imply particular units of the simulation parameters. Instead, the numerical values of the parameters set the units. In our pre-implemented example the dispersion parameters β_n have the units ps^n/m, i.e. times are measured in ps and lengths in m.

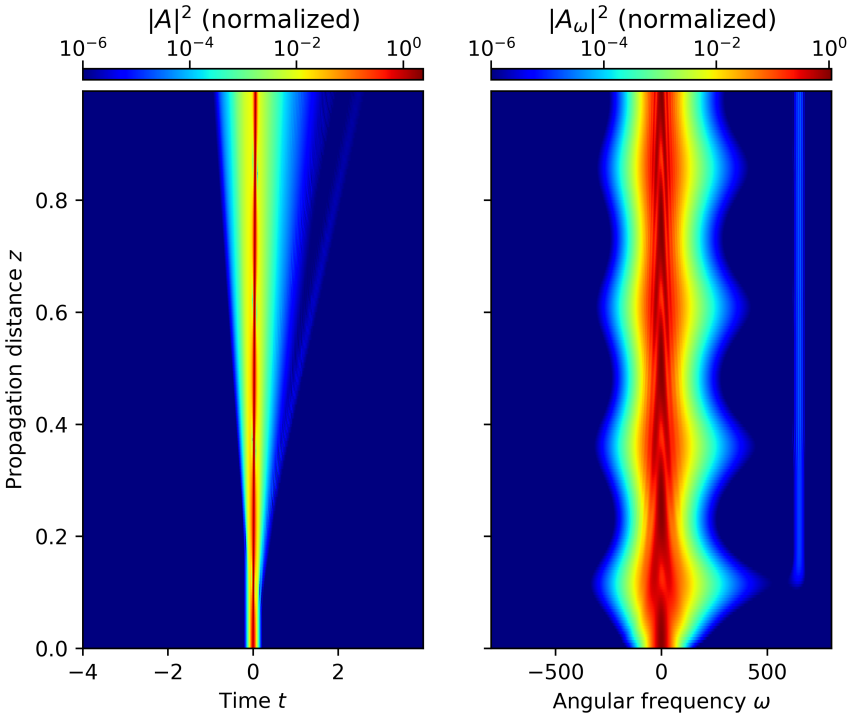

Figure 1: Evolution of the intensity (left) and spectral intensity (right) for the control parameters pre-set in `main_(a)`. As initial condition, a NSE soliton of order $N_s = 1.4$ is used.