SPIP: a computer program for the simulation of light-wave propagation in optical fibre

Stéphane Balac
UMR FOTON, CNRS, Université de Rennes 1
ENSSAT, 22305 Lannion, France
stephane.balac@univ-rennes1.fr

Arnaud Fernandez
Laboratoire d'Analyse et d'Architecture des Systèmes LAAS, CNRS
Université de Toulouse, BP 54200, 31031 Toulouse, France
afernand@laas.fr

Release 1.1 - September 2015

Abstract

SPIP is a portable command-line driven utility written in C language for Linux and MS-Windows aimed at solving the Generalized Non-Linear Schrödinger Equation (GNLSE) as well as the Non-Linear Schrödinger Equation (NLSE) involved in optics in the modelling of light-wave propagation in an optical fibre. In the SPIP program is implemented the Interaction Picture method, a new efficient alternative method to the Symmetric Split-Step method together with a dedicated costless adaptive step-size control based on the use of a 4^{th} order embedded Runge-Kutta method.

www.franquin.com

Contents

1 Introduction

In optics, the non-linear Schrödinger equation occurs for modeling light-wave propagation into an optical fibre. This particular form of the Schrödinger equation is obtained from the general set of Maxwell equations taking advantage of a certain number of assumptions made possible from the very specific characteristics of (quasi-)monochromatic wave propagation in a medium such as a fibre [1, 6]. One of the major assumption, referred as the *slowly varying envelope approximation*, concerns the expression of the electric field in the optical fibre. It assumes that the electric field $\mathbf{E}$ is linearly polarized along a direction $\mathbf{e_x}$ transverse to the direction of propagation $\mathbf{e_z}$ defined by the fibre and can be represented as a function of time τ and position $\mathbf{r} = (x, y, z)$ in a reference frame $(O, \mathbf{e_x}, \mathbf{e_y}, \mathbf{e_z})$ as

$$\mathbf{E}(\mathbf{r}, \tau) = A(z, t)\, F(x, y)\, \mathrm{e}^{-\mathrm{i}(\omega_0 \tau - kz)}\ \mathbf{e_x} \tag{1}$$

where the complex valued function A represents the slowly varying electric pulse envelope, F is the electric wave transverse representation, k is the wave number, ω_0 is the wave frequency, and t denotes a local time in a moving frame travelling along with the pulse at the group velocity $v_g = c/n_g$ where n_g is the fibre group index and c the speed of light in vacuum. The relationship between the absolute time τ and the local time t is: $t = \tau - z/v_g$. The expression of the electric wave transverse representation F can most of the time be computed explicitly using the classical method of separation of variables for partial differential equations (PDE). For instance, for circular constant transverse section fibres, it is expressed in terms of Bessel functions [1, 6].

In the simplest cases of light-wave propagation in an optical fibre, the evolution of the slowly varying pulse envelope A is governed by the following PDE referred in the literature as the Non-Linear Schrödinger Equation (NLSE) [1, 6]

$$\frac{\partial}{\partial z} A(z, t) = -\frac{\alpha}{2} A(z, t) - \frac{\mathrm{i}}{2} \beta_2 \frac{\partial^2}{\partial t^2} A(z, t) + \mathrm{i}\gamma A(z, t) |A(z, t)|^2. \tag{2}$$

Equation (2) describes wave propagation in a single mode fibre taking into account phenomena such as the optical Kerr effect through the non-linear coefficient γ, linear attenuation through the linear attenuation coefficient α and linear dispersion through the chromatic dispersion coefficient β_2.

In a more accurate model of light-wave propagation in an optical fibre, the evolution of the slowly varying pulse envelope A obeys the so-called Generalized Non-Linear Schrödinger Equation (GNLSE) [1, 6]

$$\begin{aligned} \frac{\partial}{\partial z} A(z, t) = &-\frac{\alpha}{2} A(z, t) + \left(\sum_{n=2}^{n_{\max}} \mathrm{i}^{n+1} \frac{\beta_n}{n!} \frac{\partial^n}{\partial t^n} A(z, t) \right) \\ &+ \mathrm{i}\gamma \left(1 + \frac{\mathrm{i}}{\omega_0} \frac{\partial}{\partial t} \right) \Big[A(z, t) \Big((1 - f_R) \, |A(z, t)|^2 + f_R \int_{-\infty}^{+\infty} h_R(s) \, |A(z, t - s)|^2 \, \mathrm{d}s \Big) \Big]. \end{aligned} \tag{3}$$

In equation (3) linear dispersion is now taken into account through the dispersion coefficients β_n, $n = 2, \ldots, n_{\max}$ whereas non linear dispersion is taken into account through

the simplified optical shock parameter $\tau_{shock} = 1/\omega_0$. Instantaneous Kerr effect manifests itself through the term $(1 - f_R)\,|A|^2$. The delayed Raman contribution in the time domain is taken into account through the convolution product between the instantaneous power $|A|^2$ and the Raman time response function h_R. The constant f_R represents the fractional contribution of the delayed Raman response to non-linear polarization.

Both evolution type PDE (2) and (3) have to be considered together with an initial condition at $z = 0$ in the form

$$\forall t \in \mathbb{R} \qquad A(0,t) = a_0(t) \tag{4}$$

where a_0 is a given function referred as the *incident* slowly varying electric pulse envelope in the following. Both equations are solved for all $t \in \mathbb{R}$ and all $z \in [0, L]$ where L denotes the length of fibre. The aim of the SPIP program is to solve the GNLSE (3) and the NLSE (2).

The SPIP program solves the PDE problem corresponding to the GNLSE (3) or to the NLSE (2) by means of the Interaction Picture (IP) method. The main idea of the IP method is a change of unknown in order to transform the GNLSE for the unknown A into a new equation where only remains an explicit reference to the partial derivation with respect to the space variable z and where the time variable t appears as a parameter, see [3]. This new equation can be solved numerically using the usual methods for ordinary differential equations (ODE) such as the fourth order Runge-Kutta (RK4) method. Then, by using the inverse transform we obtain approximate values for the unknown A at the grid points of a subdivision of the fibre length interval $[0, L]$. This numerical approach is referred to as the RK4-IP method. Recently an efficient embedded RK method based on Dormand and Prince RK4(3)-T formula and specifically designed for the IP method has been proposed in [4] to provide a costless adaptive step-size control in the RK4-IP method. It is this method, termed the ERK4(3)-IP method, that is implemented in the SPIP program. We refer to [2] for details on the ERK4(3)-IP method for solving the GNLSE.

The SPIP program is written in C language. It has been tested under Linux and MS-Windows. The acronym SPIP stands for Simulation of light-wave Propagation by the Interaction Picture method. Spip is also the name of one of the central characters of the long-running Franco-Belgian comic series Spirou and Fantasio. He is a courageous and sharp, grouchy pet squirrel.

2 Copyright

software under the terms of the CeCILL license as circulated by CEA, CNRS and INRIA at the following URL: `http://www.cecill.info`.

As a counterpart to the access to the source code and rights to copy, modify and redistribute granted by the license, users are provided only with a limited warranty and the software's author, the holder of the economic rights, and the successive licensors have only limited liability.

In this respect, the user's attention is drawn to the risks associated with loading, using, modifying and/or developing or reproducing the software by the user in light of its specific status of free software, that may mean that it is complicated to manipulate, and that also therefore means that it is reserved for developers and experienced professionals having in-depth computer knowledge. Users are therefore encouraged to load and test the software's suitability as regards their requirements in conditions enabling the security of their systems and/or data to be ensured and, more generally, to use and operate it in the same conditions as regards security.

3 Installation

3.1 External librairies

SPIP requires 2 external librairies:

- FFTW a C subroutine library for computing the discrete Fourier transform (DFT) developed at MIT by Matteo Frigo and Steven G. Johnson, see `http://www.fftw.org` for details.
- Gnuplot a portable command-line driven graphing utility, see `http://www.gnuplot.info` for details.

3.1.1 Installation of FFTW

Under Linux

First download the current stable version of FFTW form the FFTW home page: `http://www.fftw.org/download.html` Then, untar the FFTW gzipped archive `fftw-3.3.4.tar.gz` in the usual location for such installations (e.g. `/usr/local/`).

```
$ tar -xzf  fftw-3.3.4.tar.gz
```

In short, installation can be as simple as typing in a shell window the following Unix command as root:

```
$ cd fftw-3.3.4
$ ./configure
$ make
$ make install
```

This will build the uniprocessor complex and real transform libraries along with the test programs. The "make install" command installs the FFTW libraries in standard places, and typically requires root privileges. You can also type "make check" to put the FFTW test programs through their paces. Read chapter 10 "Installation and Customization" of the manual `fftw3.pdf` located in directory `fftw-3.3.4/doc` for additional information on the installation procedure and option setting.

Under MS-Windows

There exists precompiled FFTW 3.3.4 MS-Windows DLLs that can be download from FFTW website: `http://www.fftw.org/install/windows.html`. They are sufficient for compiling the SPIP program, so that you need not worry about compiling FFTW yourself.

- 32-bit version: fftw-3.3.3-dll32.zip (2.4MB) available from `ftp://ftp.fftw.org/pub/fftw/fftw-3.3.3-dll32.zip`
- 64-bit version: fftw-3.3.3-dll64.zip (2.8MB) available from `ftp://ftp.fftw.org/pub/fftw/fftw-3.3.3-dll64.zip`

For convenience, the FFTW DLLs are included in the SPIP archive so that you don't have to worry about downloading them.

Alternatively, it is of course possible to compile FFTW on MS-Windows. See the Installation on non-Unix Systems section of the FFTW 3 manual or MS-Windows Installation Notes at the following URL: `http://www.fftw.org/install/windows.html`.

3.1.2 Installation of Gnuplot

Under Linux

Check to see if you already have Gnuplot in your Linux distribution, e.g. by typing 'which gnuplot' in a shell window.

If you need to install Gnuplot, you can download the current stable version of Gnuplot from the Gnuplot project website: `http://www.gnuplot.info/`. Once you have download the gzipped tar file (e.g. `gnuplot-4.6.6.tar.gz`) with Gnuplot sources, untar the archive in the usual location for such installations (e.g. `/usr/local/`).

```
$ tar -xzf gnuplot-4.6.6.tar.gz
```

Installation can be made as simple as typing the following Unix commands as root in a shell window (see the installation instructions available in the source archive itself for more detailed information on the installation procedure).

```
$ cd gnuplot-4.6.6
$ ./configure
$ make
$ make install
```

Alternatively RPMs for Gnuplot are available from `rpmfind.net`. If you are using Debian Linux, Gnuplot will be downloaded and installed if you issue the following command as root: 'apt-get install gnuplot'.

Display requests to a Gnuplot session from SPIP is achieved thanks to the `gnuplot_i` module developed by Nicolas Devillard, see `http://ndevilla.free.fr/gnuplot/`. The 2 files `gnuplot_i.c` and `gnuplot_i.h` useful for the SPIP program are provided in the SPIP archive so that there is no need to download the `gnuplot_i` module.

Under MS-Windows

Self-installer package of Gnuplot MS-Windows binaries is available from the following URL: `http://sourceforge.net/projects/gnuplot/files/gnuplot/4.6.6/` where it is referred as `gp466-win32-setup.exe` or `gp466-win64-setup.exe`.

The `gnuplot_i` module designed to display requests to a Gnuplot session from the SPIP program has been extended to support MS-Windows by Robert Bradley, see `http://robert-bradley.co.uk/gnuplot/`. Unfortunately a bug in the display systematically occurs. Therefore the possibility to interactively display the slowly electric pulse envelope at the fibre entrance and at the fibre end under MS-Windows had been disabled. The user can view the draws of the slowly electric pulse envelope after exiting the SPIP program using Matlab/Octave drawing facilities thanks to the scripts provided in the SPIP archive, see Section 4.2.

3.2 Installing SPIP

3.3 Under Linux

Unzip the SPIP archive `spip-1.1.zip` in a suitable location of the home directory by typing in a shell window:

```
$ unzip spip-1.1.zip
```

Go to the Linux directory:

```
$ cd  spip-1.1/linux
```

Compile the C program thanks to the Makefile available in the directory:

```
$ make
```

The executable is named `spipxx`. Optionally, you can delete object files by typing:

```
$ make clean
```

3.4 Under MS-Windows

If you have installed the Linux-like environment Cygwin for Windows (`https://www.cygwin.com/`) you can install the SPIP program proceed pretty much as the for a Linux installation.

Otherwise, untar the SPIP gzipped archive `spip-1.1.tar.gz` in a suitable location of the home directory with your favourite MS-Windows utility for manipulating archives. A compiled version of the SPIP program is available in the SPIP archive in directory `windows` as file `spipxx_windows.exe`. You can launch the SPIP program by double-clicking on the `spipxx_windows.exe`.

Alternatively, you can use an C language IDE (e.g. Code::Blocks, `http://www.codeblocks.org/`) to compile the sources. A Code::Blocks project `spip_windows.cbp` is available in the archive under the directory `windows` for convenience. Compilation of the sources will be required e.g. if you want to add an other shape type for the incident slowly varying electric pulse envelope, see Section 4.3.

Note that display requests to a Gnuplot session from SPIP is not available under Windows. It is however possible to display the slowly varying electric pulse envelope using the Matlab/Octave scripts provided in the archive, see section 4.2 for details.

4 Running the SPIP program

4.1 An annotated execution example

The SPIP program is launched under Linux by typing the following command in a shell window:

```
$ ./spipxx
```

Under MS-Windows, you can launch the SPIP program by double-clicking on the `spipxx_windows.exe` located in the `windows` directory of the SPIP archive.

In the SPIP command line user interface, the user is first asked to provide the name and path of the directory where the result files will be stored.

```
[?] Directory where result files will be stored (must exist): [. by default]: res
```

Typing a carriage return will set this directory to the current directory by default. If the folder doesn't exist, the program will end to let the user create it. For instance, one gets:

```
[?] Directory where result files will be stored (must exist): [. by default]: Res
 :-( *** Error : cannot access directory [Res]. Create it!
```

Then the user is asked to provide a keyword for the simulation. The result files generated by the SPIP program will automatically bear this keyword in their name with an additional unique key corresponding to the elapsed time in second from Epoch.

```
[?]  Keyword for the present simulation: sol3
```

At the end of the run, the result files directory (here `res`) will contain the following execution files, all of them ending with the same label made from the keyword provided by the user (here: sol3) and the unique key generated by the program (here: 1421248167):

```
$ ls res
data_sol3_1421248167.txt      out_time_sol3_1421248167.png
in_spec_sol3_1421248167.png   pulse_in_sol3_1421248167.txt
in_time_sol3_1421248167.png   pulse_out_sol3_1421248167.txt
out_spec_sol3_1421248167.png  zstep_sol3_1421248167.txt
```

Then the type of equation to solve must be specified: Non-Linear Schrödinger Equation or Generalized Non-Linear Schrödinger Equation.

```
[?] Equation type [0 for the NLS or 1 for the GNLS] =  0
```

Four pre-defined incident shapes for the slowly varying electric pulse envelope are available: Soliton, Gaussian and Super-Gaussian, Hyperbolic Secant. The mathematical expressions of these incident pulses are given in Section 4.3. Additionally, it is possible to provide any other incident pulse shape in the program. To do so, one must append the expression of the incident pulse in file `inpulse_shape.c` in the space provided and one must re-compile the program.

```
[?] Shape of the incident slowly varying pulse envelope [1: Soliton, 2: Gaussian,
  3: Hyperbolic Secant, 4: Super-Gaussian, 5: user-defined] = 1
```

The next stage consists in providing the incident pulse envelope features. The items may vary depending on the selected shape. For the Soliton they are : the Soliton order, the wavelength of the pulse and the half-width at 1/e-intensity point of the pulse.

```
[?] Soliton order = 3

[?] Central wavelength of the pulse [nm] = 1550

[?] Width parameter of the incident pulse [ps] : T0 = 2.8365
```

It is followed by the features of the optical fibre, i.e. the values of the parameters involved in equation (2) or (3). For the NLSE they are: fibre length L, linear loss/gain coefficient α, number $n_{\max}$ of non-zero dispersion coefficients and values of these coefficients $\beta_2, \ldots, \beta_{n_{\max}}$, non-linear coefficient γ.

```
[?] Fiber length [km] = 0.637

[?] Linear loss/gain coefficient alpha [km^-1] = 0

[?] Number of non-zeros beta_k (k>=2) coefficients = 1

[?] Value of beta_2 [ps^2 km^-1]= -19.83

[?] non-linear coefficient gamma [W^-1 km^-1] = 4.3
```

In the case of the GNLSE (3), the fractional contribution of the delayed Raman response f_R must also be supplied. Note that the expression of the Raman time response function h_R provided in [1] is used in the SPIP program but this function can be easily modified in file `hraman.c` (the program must then be re-compiled).

In a last stage, the user must provide the values of the tuning parameters of the ERK4(3)-IP method: size of the time window and number of sampling point for the FFT as well as the initial step-size and tolerance for the adaptive step-size method.

```
[?] Size of the time window; winT [ps] = 50

[?] Number of time sampling points 2^p with p = 14

[?] Initial step-size length [m] = 1

[?] Tolerance value = 1e-6
```

Before computations start, the variation with respect to time of the modulus, real and imaginary parts of the slowly varying pulse envelope $A(L, t)$ at the fibre entrance are depicted in a Gnuplot graphics window (this feature is not available under MS-Windows). In a second window are depicted the Fourier Transform of these quantities, see Fig.

1. The 2 figures are automatically recorded in the results directory (in the current simulation: `res`) into 2 files in PNG format named respectively `in_spec_xxx.png` and `in_time_xxx.png` respectively, where `xxx` stands for the key label of the simulation.

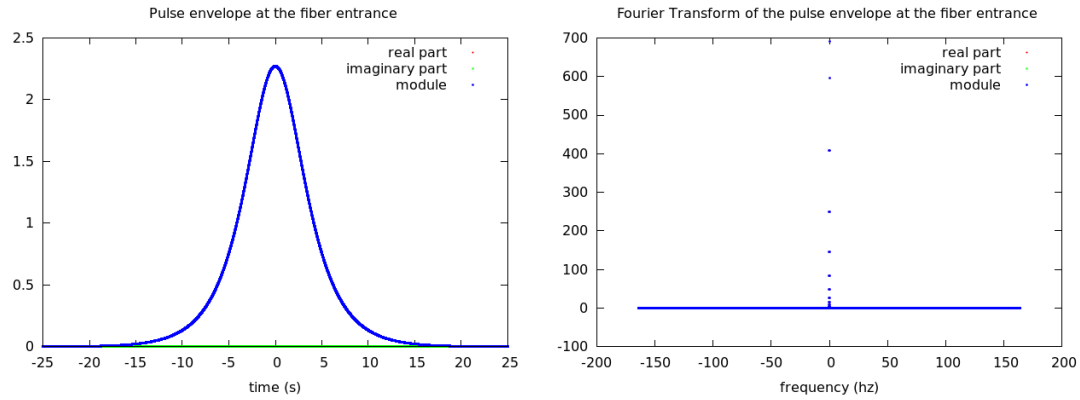

Figure 1: Screen-shot of the Gnuplot graphics windows where are depicted the time variation of the modulus, real and imaginary parts of the slowly varying pulse envelope $A(L,t)$ at the fibre end (on the left) and the Fourier Transform of these quantity (on the right).

During the computations, the current step number, the current step-size and the progress of the calculations are printed in the console window.

Once the computations are achieved, the time variation of the modulus, real and imaginary parts of the slowly varying pulse envelope $A(L,t)$ at the fibre end are depicted in a Gnuplot graphics window (this feature is not available under MS-Windows). In a second graphics window are depicted the Fourier Transform of the modulus, real and imaginary parts of the slowly varying pulse envelope $A(L,t)$ at the fibre end, see Fig. 5. The 2 figures are also automatically recorded in the results directory into 2 files in PNG format named respectively `out_spec_xxx.png` and `out_time_xxx.png` respectively, where `xxx` stands for the label key of the simulation. Additionally, the L^2 norm, the L^1 norm and the $\mathbb{L}^\infty$ norm of the solution are printed. They are respectively defined by:

$$\|A(L)\|_2 = \left(\int_{\mathbb{R}} |A(L,t)|^2 \, \mathrm{d}t \right)^{\frac{1}{2}}, \quad \|A(L)\|_1 = \int_{\mathbb{R}} |A(L,t)| \, \mathrm{d}t, \quad \|A(L)\|_\infty = \sup_{t \in \mathbb{R}} |A(L,t)|.$$

```
Dispersion length = 4.057353e-01 km
Non-linear length = 4.508170e-02 km
Number of spatial steps = 439

L 2-norm of the solution at fibre end = 5.409691e+00
```

```
L 1-norm                                = 2.023695e+01
L infinity-norm                         = 2.271319e+00
```

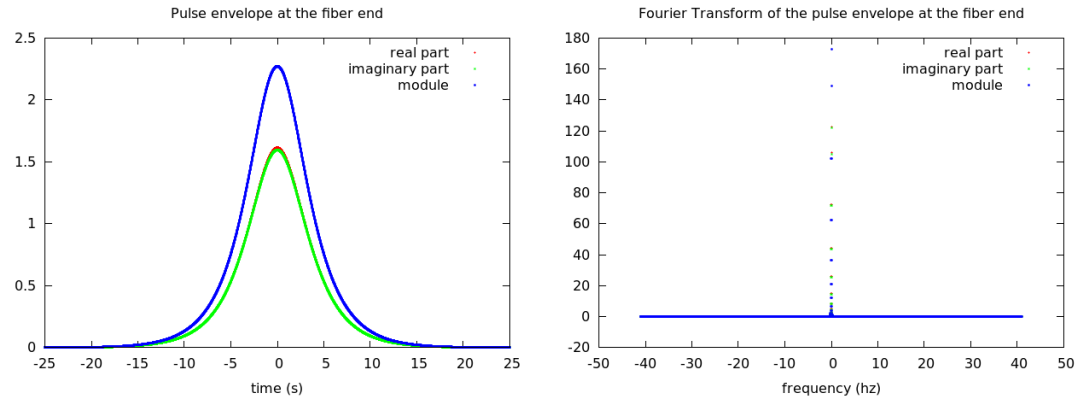

Figure 2: Screen-shot of the Gnuplot graphics windows where are depicted the time variation of the modulus, real and imaginary parts of the slowly varying pulse envelope $A(L,t)$ at the fibre end (on the left) and the Fourier Transform of these quantity (on the right).

When required, one can record the solution to the Schrödinger equation at every computational step. In the header file `spip.h` one must first set the macro `REC` to the value 1 and recompile the program. During the next executions of the SPIP program, the computed solution to the Schrödinger equation will be recorded at every computational step in a binary file name `pulse_all_xxx.spip`. This functionality is not set by default in the SPIP program since it generates very large files on the disc depending on the problem to be solved or on the simulation parameters. To circumvent this issue, it is also possible to record the solution at periodical steps. The period must be chosen depending on the problem to be solve. The period value is specify in the header file `spip.h` and corresponds to the macro `NREC`. The program must be recompiled to take into account any modification in the value. In the header file `spip.h` the two macros `REC` and `NREC` are set by default as follows.

```
#define REC 0 // When REC = 1 all the intermediate results are stored in a file
              // To be used with care due to the huge data file generated !!!!
#define NREC 1 // When REC = 1, it prompts the record the solution every NREC steps
               // (use this option to avoid too large recorded files)
```

At the end of the simulation, the directory specified to store the results files contains the following files (`xxx` stands for the key label of the current execution):

- `data_xxx.txt`: this ASCII file contains the values of the physical and numerical parameters used for the current simulation.

- `pulse_in_xxx.txt`: this ASCII file contains the incident slowly varying electric pulse envelope sampled over the time window.
- `pulse_out_xxx.txt`: this ASCII file contains the slowly varying electric pulse envelope at the fibre end sampled over the time window.
- `zstep_xxx.txt`: this ASCII file contains the step-size values along the propagation direction computed by the adaptive step-size method.
- `in_time_xxx.png`: image of the time variation of the incident slowly varying electric pulse envelope in PNG format as generated by Gnuplot, see Fig. 1.
- `in_spec_xxx.png`: image of the frequency variation of the Fourier Transform of the incident slowly varying electric pulse envelope in PNG format as generated by Gnuplot, see Fig. 1.
- `out_time_xxx.png`: image of the time variation of the slowly varying electric pulse envelope at fiber end in PNG format as generated by Gnuplot, see Fig. 5.
- `out_spec_xxx.png`: image of the frequency variation of the Fourier Transform of the slowly varying electric pulse envelope at fiber end in PNG format as generated by Gnuplot, see Fig. 5.

For instance, for the current simulation we have the following files in the directory `res`:

```
$ ls res
data_sol3_1421248167.txt      out_time_sol3_1421248167.png
in_spec_sol3_1421248167.png   pulse_in_sol3_1421248167.txt
in_time_sol3_1421248167.png   pulse_out_sol3_1421248167.txt
out_spec_sol3_1421248167.png  zstep_sol3_1421248167.txt
```

The last 4 files in PNG format are not generated by the SPIP program under MS-Windows. However the pictures can be obtained from the 2 files `pulse_in_xxx.txt` and `pulse_out_xxx.txt` thanks to the Matlab/Octave script `plotsol.m` provided in the SPIP archive, see Section 4.2 for details.

The file `data_sol3_1421248167.txt` recording the simulation data contains the following information:

```
$ more data_sol3_1421248167.txt
Simulation date : Thu Jan 15 14:56:31 2015

Solving the NLSE by the ERK4(3)-IP method
CPU time for the simulation : 1.762619 s.
Incident slowly varying pulse envelope [1: Soliton, 2: Gaussian,
3: Hyperbolic Secant, 4: Super-Gaussian, 5: user-defined] = 1
Wavelength = 1550 nm
Fiber length = 0.637210 km
Half-width of the incident pulse: T0 = 2.836500 ps
```

```
PeakPower = 5.158592 W
Linear attenuation coefficient : alpha = 0.000000 km^(-1)
linear dispersion coefficients :
beta_2 = -19.830000 ps^2/km
non-linear parameter : gamma = 4.300000 W^(-1) km^(-1)
Linear fiber length = 0.405735
non-linear fiber length = 0.045082
Time windows size = 50.000000
Time step-size = 0.003052
Number of FFT points = 16384
Tolerance set for the adaptive space step-size = 1.000000e-06
Initial step-size for the spatial discretization = 0.001000 Km
Number of iterations of the ERK4(3)-IP method = 185
Number of FFT achieved = 2961
L^2 norm of the solution at fiber end = 5.4096852398216813e+00
L^1 norm of the solution at fiber end = 2.0235598036077338e+01
L^infinity norm of the solution at fiber end = 2.2712537077671460e+00
```

4.2 Matlab/Octave tools

Together with the SPIP program are provided under directory `matlab` in the SPIP archive several Matlab[1] /Octave[2] scripts to handle SPIP results files. These scripts can be copied in the user result files directory or a symbolic link to these scripts from the user result files directory can be created.

4.2.1 Solution plotting tool

File `plotsol.m` contains a script to draw under Matlab/Octave the slowly varying electric pulse envelope A at fibre entrance or end, respectively from files `pulse_in_xxx.txt` and `pulse_out_xxx.txt` generated by the SPIP program. It also plots the Fourier Transform of the slowly varying electric pulse envelope A. The script allows to plot again the solution when the Gnuplot graphical windows of the SPIP program have been closed.

For instance, under Matlab/Octave, typing

```
>> plotsol
Filename = pulse_out_sol3_1421248167.txt
Size of the time windows [ps] = 50
```

produces the plots depicted in Fig. 3.

Moreover graphics tools available under Matlab/Octave allow image manipulations such as to zoom in to enlarge the draw of the Fourier Transform of the pulse or to modify the figure setting.

[1]Matlab is a numerical computing environment developed by the MathWorks company, see `http://www.mathworks.com/`.

[2]GNU Octave is a high-level interpreted language, primarily intended for numerical computations, distributed under the terms of the GNU General Public License. The Octave language is quite similar to Matlab so that most programs are easily portable. See `https://www.gnu.org/software/octave/`

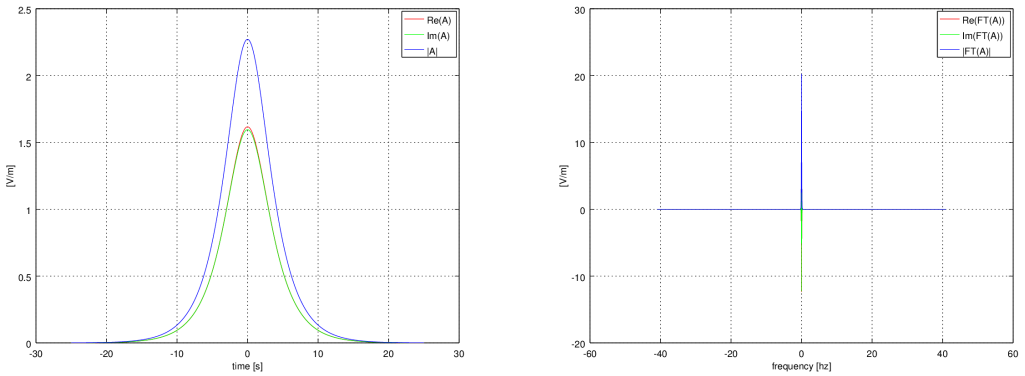

Figure 3: Screen-shot of the Octave graphics windows where is depicted the time variation of the modulus, real and imaginary parts of the slowly varying pulse envelope $A(L,t)$ at the fibre end (on the left) and the Fourier Transform of these quantity (on the right).

As we point out in the last section, when required, the solution to the Schrödinger equation at every computational step (or according to a period) can be recorded and these recorded data can be used to plot the solution at various positions along the fibre length. The Matlab/Octave script entitled `plotspip.m` has been written to draw the variations of the solution in the time domain as well as in the wavelength (frequency) domain at various positions along the fibre length.

For instance, under Matlab/Octave, typing

```
octave:1> plotspip
Filename = pulse_all_sol3_1421248167.spip
Size of the time windows [ps] = 50
Central wavelength of the pulse [nm] = 1550
Time window to be plotted [t_min,t_max] = [-25 25]
```

produces a plot of the power of the slowly varying pulse envelope power $|A|^2$ as a surface plot as well as as a contour plot, see Fig. 4.

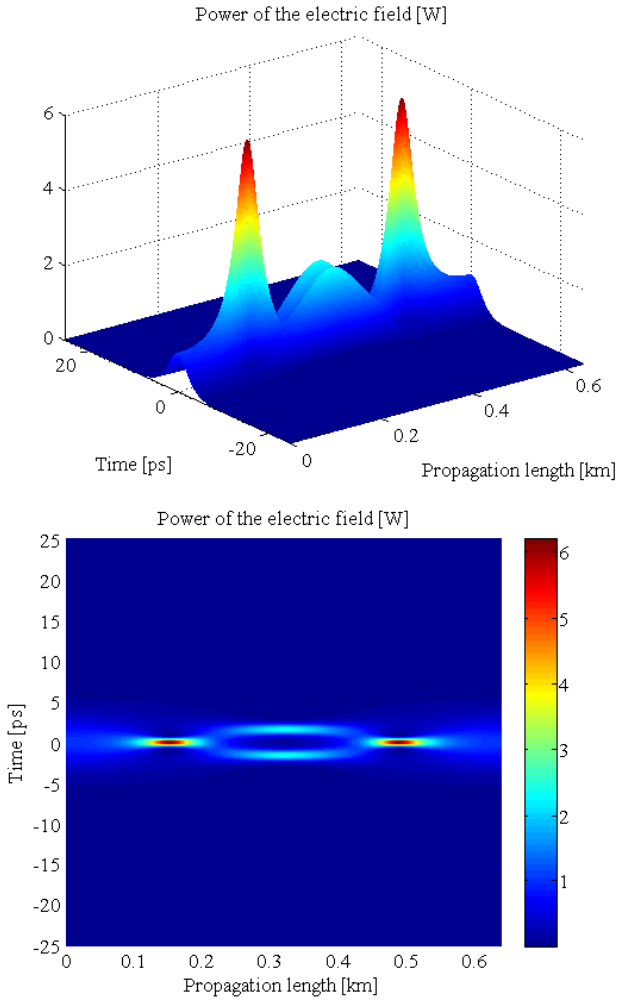

Figure 4: Time variation of the power of the slowly varying pulse envelope power $|A|^2$ along the fibre for the 3^{rd} order Soliton.

It also produces a plot of the power spectral density in dB as a function of the wavelength and position along the fibre, see Fig. 5.

4.2.2 Step-size plotting tool

The script `plotstep.m` can be used under Matlab/Octave to draw the variation of the step-size along the fibre length resulting from the adaptive step-size strategy used in the SPIP program. Since the step-size is adapted so that the local error matches the given tolerance at each computational step, small step-sizes indicate an area in the fibre where the slowly varying pulse envelope varies a lot. On the contrary, larger step-sizes indicate areas where the solution varies in a very smooth way.

For instance, under Matlab/Octave, typing

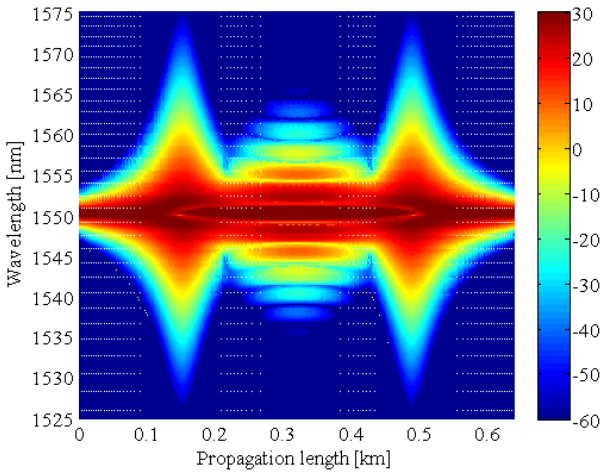

Figure 5: Power spectral density in dB as a function of wavelength and position along the fibre for the propagation of a 3rd order Soliton.

```
>> plotstep
Filename =  zstep_sol3_1421248167.txt
```

produces the plots depicted in Fig. 6.

4.3 Predefined incident pulse shapes

In the SPIP program are implemented the usual shapes for the incident field as described in [1]. They are defined in file `inpulse_shape.c` and commented below.

4.3.1 Soliton

The incident slowly varying electric pulse envelope A corresponding to an optical Soliton is of the form

$$a_0(t) = A(z=0,t) = \frac{N_s}{\sqrt{\gamma L_D}\cosh(t/T_0)} \tag{5}$$

where N_s is the Soliton order, T_0 is the pulse half-width and $L_D = -T_0^2/\beta_2$ is the dispersion length. These quantity are related to the peak power P_0 by the relation $P_0 = \frac{N_s^2}{\gamma L_D}$.

4.3.2 Gaussian pulse

Pulses emitted from many lasers can be approximated by a Gaussian shape. In the case of a Gaussian pulse the incident slowly varying electric pulse envelope A is of the form

$$a_0(t) = A(z=0,t) = \sqrt{P_0}\ \exp\left(-\frac{1}{2}\frac{t^2}{T_0^2}\right) \tag{6}$$

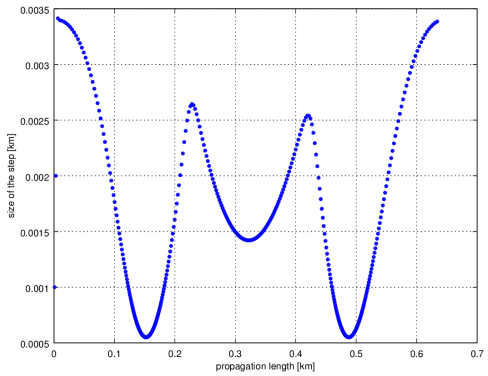

Figure 6: Screen-shot of the Octave graphics window where is depicted the variation of the step-size resulting from the adaptive step-size strategy used in the SPIP program for the 3^{rd} order Soliton.

where T_0 is the half-width at 1/e-intensity point of the pulse and P_0 is the peak-power of the pulse. In practice, it is customary to use the full width at half maximum (FWHM) in place of T_0. For a Gaussian pulse, the two are related as: $T_{FWHM} = 2\sqrt{\ln 2}\, T_0 \approx 1.665\, T_0$.

We also consider the case of linearly chirped Gaussian pulses for which the slowly varying electric pulse envelope A can be written as

$$a_0(t) = A(z=0,t) = \sqrt{P_0}\ \exp\left(-\frac{1}{2}\,(1+\mathrm{i}C)\,\frac{t^2}{T_0^2}\right) \tag{7}$$

where C is the chirp parameter (a positive or negative real number).

4.3.3 Hyperbolic secant pulse

The hyperbolic secant pulse shape occurs naturally in the context of optical Solitons and pulses emitted from some mode-locked lasers. The incident slowly varying electric pulse envelope A takes the form

$$a_0(t) = A(z=0,t) = \sqrt{P_0}\ \frac{\exp(-\frac{1}{2}\,\mathrm{i}C\,t^2/T_0^2)}{\cosh(t/T_0)} \tag{8}$$

where T_0 is the half-width at 1/e-intensity point of the pulse and C is chirp parameter (a positive or negative real number).

4.3.4 Super-Gaussian pulse

For a super-Gaussian pulse, relation (7) is generalized to take the form

$$A(z=0,t) = \sqrt{P_0}\ \exp\left(-\frac{1}{2}\,(1+\mathrm{i}C)\,\frac{t^m}{T_0^m}\right) \tag{9}$$

where the parameter m controls the degree of edge sharpness. For $m = 1$ we recover the case of chirped Gaussian pulses. For larger value of m, the pulse becomes square shaped with sharper leading and trailing edges. In general, a pulse with steeper leading and trailing edges broadens more rapidly with propagation simply because such a pulse has a wider spectrum to start with.

4.3.5 User defined pulse

It is also possible for the user to define its own shape for the the incident slowly varying electric pulse envelope. It suffices to append the expression of the incident pulse in file `inpulse_shape.c` in the space provided and to re-compile the program.

5 Running examples

In this section, we present examples where the SPIP program is used to simulated light-wave propagation in an optical fibre under various experimental conditions. All the simulation were achieved under Linux Ubuntu 14.04 LTS on a desktop computer equipped with an Intel Core i5-4200M processor and 8 GO RAM.

5.1 Solving the GNLSE (I)

We consider the case of the GNLSE (3) with the following set of physical parameters: $\omega_0 = 1770\,\text{Thz}$, $\gamma = 4.3\,\text{W}^{-1}\text{km}^{-1}$, $\beta_2 = 19.83\,\text{ps}^2\text{km}^{-1}$, $\beta_3 = 0.031\,\text{ps}^3\text{km}^{-1}$ and $\beta_n = 0$ for $n \geq 4$, $\alpha = 0.046\,\text{km}^{-1}$, $L = 96,77\,\text{m}$, $f_R = 0.245$. The Gaussian pulse at the fibre entrance ($z = 0$) is expressed as

$$\forall t \in \mathbb{R} \quad a_0(t) = \sqrt{P_0}\,\text{e}^{-\frac{1}{2}(t/T_0)^2} \tag{10}$$

where $T_0 = 2.8365\,\text{ps}$ is the pulse half-width and $P_0 = 100\,\text{W}$ is the pulse peak power. The number of sampling points for the FFT computations was set to 2^{14}. The tolerance for the step-size control was set to 10^{-6} and the initial step-size was 1 m. The number of discretisation steps along the fibre was found to be 300 and the computation time was 4.52 s. We have depicted in Fig. 7 the modulus, real and imaginary parts of the slowly varying pulse envelope A at the fibre entrance and at fibre end.

The detail of the execution of the SPIP program is given below.

```
 $./spipxx

[?] Directory where result files will be stored (must exist): [. by default]: res
[?] Keyword for the present simulation: green
[?] Equation type [0 for the NLS or 1 for the GNLS] =  1

 ****************************************
 **** Incident pulse envelope features ****
```

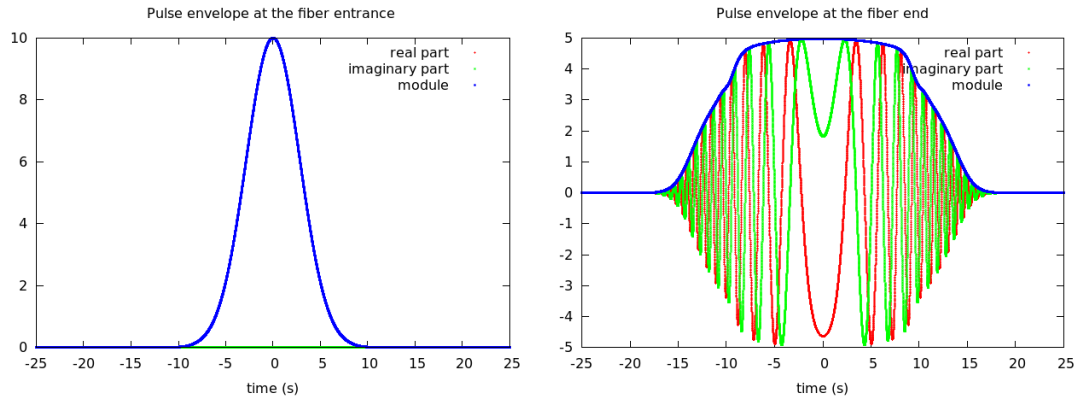

Figure 7: Modulus, real and imaginary parts of the slowly varying pulse envelope A at the fibre entrance (on the left) and at fibre end (on the right).

```
 ****************************************

[?] Shape of the incident slowly varying pulse envelope [1: Soliton, 2: Gaussian,
3: Hyperbolic Secant, 4: Super-Gaussian, 5: user-defined] = 2
[?] Central wavelength of the pulse [nm] = 1064
[?] Peakpower [W] = 100
[?] Chirp constant = 0
[?] Half-width of the incident pulse [ps] : T0 = 2.8365

 ****************************************
 ****      Optical fibre features      ****
 ****************************************

[?] Fibre length [km] = 96.77e-3
[?] Linear loss/gain coefficient alpha [km^-1] = 0.046
[?] Number of non-zeros beta_k (k>=2) coefficients = 2
[?] Value of beta_2 [ps^2 km^-1]= 19.83
[?] Value of beta_3 [ps^3 km^-1]= 0.031
[?] non-linear coefficient gamma [W^-1 km^-1] = 4.3
[?] Fractional contribution of the delayed Raman response: fr = 0.245

 ****************************************
 ****   Numerical parameters setting   ****
 ****************************************

[?] Size of the time window; winT [ps] = 50
[?] Number of time sampling points 2^p with p = 14
[?] Initial step-size length [m] = 0.1
[?] Tolerance value = 1e-6
```

```
------------------------------------------------
Computations start on: Fri Jan 16 15:13:40 2015
Computations end on: Fri Jan 16 15:13:45 2015
Simulation lasted (s) : 4.517599
------------------------------------------------

****************************************
**** Results (see also files in dir. [./res/])
****************************************
Dispersion length = 4.057354e-01 km
Non-linear length = 2.325581e-03 km
Number of spatial steps = 300

L^2-norm of the solution at fibre end = 2.237221e+01
L^1-norm                              = 1.161340e+02
L^infinity-norm                       = 4.979086e+00

Press [Enter] to close figures and exit program ...
```

5.2 Solving the GNLSE (II)

We use the SPIP program to solve the GNLSE on a test example chosen to match with a typical case of high speed data propagation through a $L = 20\,\mathrm{km}$ single mode fibre in optical telecommunication with a data's carrier frequency located in the C band of the infra-red spectrum ($f_0 = 193\,\mathrm{Thz}$). The following set of fibre's parameters were used for the simulation: $\alpha = 0.046\,\mathrm{km}^{-1}$, $\gamma = 4.3\,\mathrm{W}^{-1}\mathrm{km}^{-1}$, $f_R = 0.245$, $\beta_2 = -19.83\,\mathrm{ps}^2\mathrm{km}^{-1}$, $\beta_3 = 0.031\,\mathrm{ps}^3\mathrm{km}^{-1}$ and $\beta_n = 0$ for $n \geq 4$. The source term $a_0 = A(z = 0)$ was represented as a first order Gaussian pulse:

$$a_0 : t \mapsto \sqrt{P_0}\,\mathrm{e}^{-\frac{1}{2}(t/T_0)^2}$$

where T_0 is the pulse half-width at 1/e intensity point and P_0 is the pulse peak power. Simulations were carried out for a pulse-width $T_0 = 6.8\,\mathrm{ps}$ and for a peak power value of $25\,\mathrm{mW}$.

The number of sampling points for the FFT computations was set to 2^{14}. The tolerance for the step-size control was set to 10^{-6} and the initial step-size was 1 m. The number of discretisation steps along the fibre was found to be 26 and the computation time was 1.16 s. We have depicted in Fig. 9 the modulus, real and imaginary parts of the slowly varying pulse envelope A at the fibre entrance and at fibre end.

The detail of the execution of the SPIP program is given below.

```
$./spipxx
```

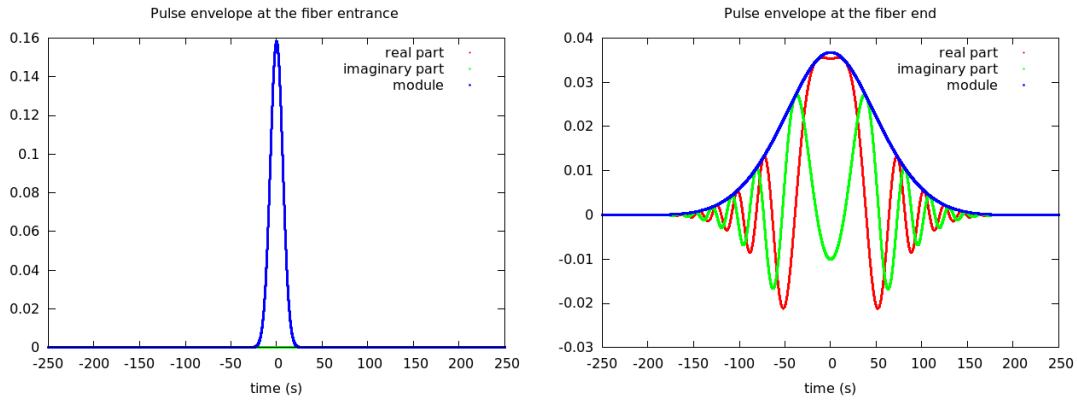

Figure 8: Modulus, real and imaginary parts of the slowly varying pulse envelope A at the fibre entrance (on the left) and at fibre end (on the right).

```
[?] Directory where result files will be stored (must exist): [. by default]: res
[?] Keyword for the present simulation: telcom
[?] Equation type [0 for the NLS or 1 for the GNLS] =  1

 ******************************************
 **** Incident pulse envelope features ****
 ******************************************

[?] Shape of the incident slowly varying pulse envelope [1: Soliton, 2: Gaussian,
 3: Hyperbolic Secant, 4: Super-Gaussian, 5: user-defined] = 2
[?] Central wavelength of the pulse [nm] = 1550
[?] Peak power [W] = 25e-3
[?] Chirp constant = 0
[?] Half-width of the incident pulse [ps] : T0 = 6.8

 ******************************************
 ****      Optical fibre features      ****
 ******************************************

[?] Fibre length [km] = 20
[?] Linear loss/gain coefficient alpha [km^-1] = 0.046
[?] Number of non-zeros beta_k (k>=2) coefficients = 2
[?] Value of beta_2 [ps^2 km^-1]= -19.83
[?] Value of beta_3 [ps^3 km^-1]= 0.031
[?] non-linear coefficient gamma [W^-1 km^-1] = 4.3
[?] Fractional contribution of the delayed Raman response: fr = 0.245

 ******************************************
 ****   Numerical parameters setting   ****
```

```
 ****************************************

[?] Size of the time window; winT [ps] = 500
[?] Number of time sampling points 2^p with p = 14
[?] Initial step-size length [m] = 1
[?] Tolerance value = 1e-6

 -------------------------------------------------
 Computations start on: Fri Jan 16 16:15:40 2015
 Computations end on: Fri Jan 16 16:15:41 2015
 Simulation lasted (s) : 1.160005
 -------------------------------------------------

 ****************************************
 **** Results (see also files in dir. [./res/])
 ****************************************
 Dispersion length = 2.331820e+00 km
 Non-linear length = 9.302326e+00 km
 Number of spatial steps = 26

 L^2-norm of the solution at fibre end = 3.465264e-01
 L^1-norm                              = 4.717671e+00
 L^infinity-norm                       = 3.675944e-02

 Press [Enter] to close figures and exit program ...
```

We have depicted in Fig. 9 the time variation along the fibre of the slowly varying pulse envelope power $|A|^2$ (top figure) and the power spectral density expressed in dB as a function of the wavelength along the fibre (bottom figure). The figures were obtained using the `plotspip.m` Matlab script.

5.3 Soliton collisions

We now present numerical simulation results for the collision of 2 first order Solitons [1]. It is known that when two neighbouring Solitons are launched with the same phase, they are initially attracted towards each other and then the two pulses periodically coalesce to form one pulse and separate [5]. The source term was

$$a_0 : t \in \mathbb{R} \mapsto \frac{1}{\sqrt{\gamma L_D}} \left(\frac{1}{\cosh((t-T_1)/T_0)} + \frac{R\mathrm{e}^{\mathrm{i}\phi}}{\cosh(R(t+T_1)/T_0)} \right)$$

where T_0 is the pulse half-width, $L_D = -T_0^2/\beta_2$ is the dispersion length, R accounts for the relative amplitude, ϕ for the relative phase shift and T_1 for the initial separation time.

In order to take into account this new incident pulse shape, file `inpulse_shape.c` has been appended as follows:

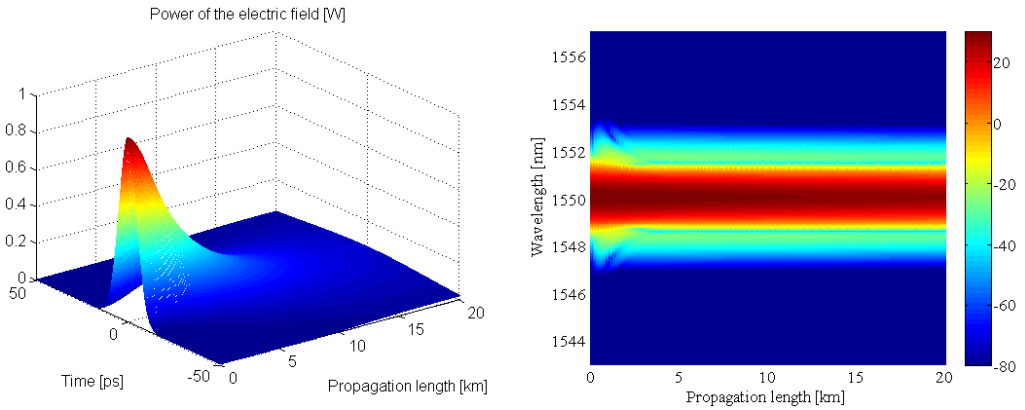

Figure 9: Slowly varying pulse envelope power $|A|^2$ in W (left) as a function of time and space and power spectral density in dB (right) as a function of wavelength and space for the propagation of a Gaussian pulse along a 20 km long fibre.

```
case 5:             // source for the 2 solitons collision
        q0=3.5;
        r = 1;  // relative amplitude
        theta=0;// relative pulses shift
        LD=pow(T0,2.0)/fabs(betaw[2]); // Dispersion length [km]
       *PeakPower=1/(gamma*LD);   // Peak power [W]
        for(k=0; k<nt; k++)
       {
           u0[k]=sqrt(*PeakPower)*(1/cosh((t[k]-q0*T0)/T0)+
                 r*cexp(I*theta)/cosh(r*(t[k]+q0*T0)/T0));
       }
   break;
```

and the SPIP program recompiled.

The following physical parameters were taken for the numerical experiment: $L = 5000\,\text{km}$, $\lambda = 1550\,\text{nm}$, $\gamma = 2.2\,\text{W}^{-1}\,\text{km}^{-1}$, $\beta_2 = -0.1\,\text{ps}^2\,\text{km}^{-1}$, $T_0 = 4\,\text{ps}$, $T_1 = 100\,\text{ps}$, $R = 1$ and $\phi = 0$. For the simulation, the time windows was 400 ps and the number of FFT nodes was 2^{14}. The initial step-size was set to 1 km and the tolerance to 10^{-6}.

We have depicted in Fig. 13 the modulus, real and imaginary parts of the slowly varying pulse envelope A at the fibre entrance and at fibre end.

With the values considered in the simulation, the collision of the 2 Solitons is predicted to happen at a distance of 4161 km [1]. This is confirmed by the plot of the variation of the step-size resulting from the adaptive step-size strategy, see Fig. 11. When the 2 Solitons catch up, the numerical solution varies a lot over a small distance and therefore the step-size decreases so that the local error meets the prescribed tolerance. Using the following Matlab/Octave commands gives a value of $4165.3 \pm 2.9\,\text{km}$ for the Solitons collision.

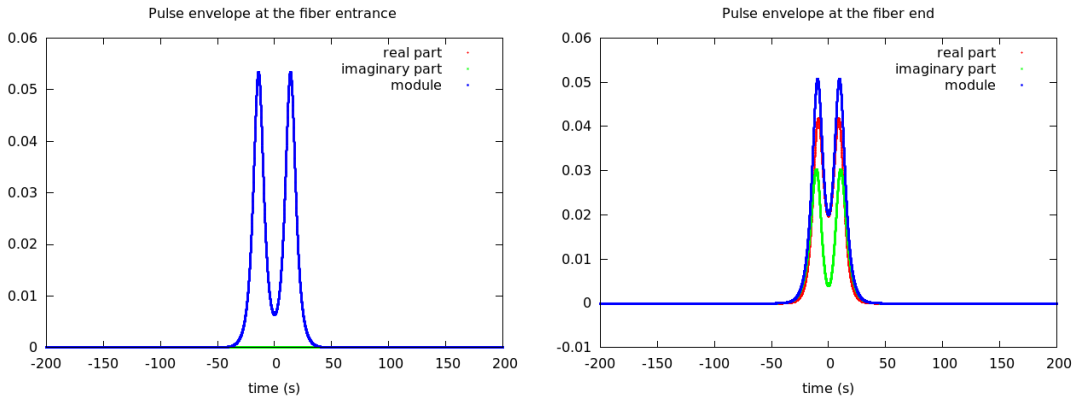

Figure 10: Modulus, real and imaginary parts of the slowly varying pulse envelope A at the fibre entrance (on the left) and at fibre end (on the right).

```
octave:1> plotstep
 Filename: zstep_solcol_1421424189.txt
octave:2> k0=20;
octave:3> dzz=dz(k0:length(dz));
octave:4> [vmin,k]=min(dzz)
vmin =  2.8933
k =  363
octave:5> Lz(k+k0-1)
ans =  4165.3
```

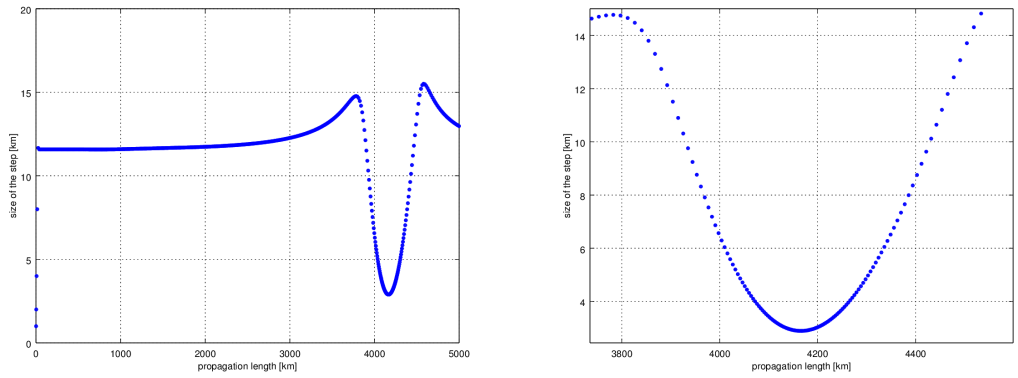

Figure 11: Variation of the step-size resulting from the adaptive step-size strategy (left) and zoom in the area of collision (right).

We have depicted in Fig. 12 the modulus, real and imaginary parts of the slowly

varying pulse envelope A at a distance of 4161 km where the 2 Solitons collapse.

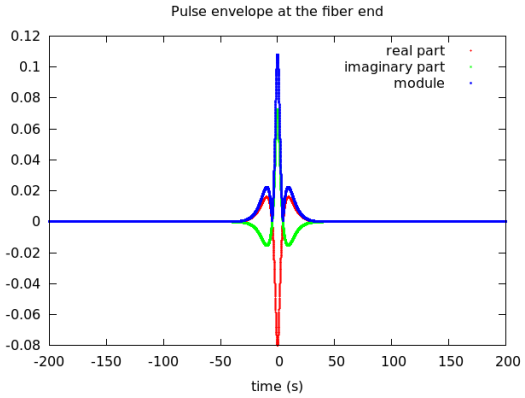

Figure 12: Modulus, real and imaginary parts of the slowly varying pulse envelope A at a distance of 4161 km where the 2 Solitons collapse.

The detail of the execution of the SPIP program is given below.

```
$./spipxx

[?] Directory where result files will be stored (must exist): [. by default]: res
[?] Keyword for the present simulation: solcol
[?] Equation type [0 for the NLS or 1 for the GNLS] =  0

 ****************************************
 **** Incident pulse envelope features ****
 ****************************************

[?] Shape of the incident slowly varying pulse envelope [1: Soliton, 2: Gaussian,
 3: Hyperbolic Secant, 4: Super-Gaussian, 5: user-defined] = 50
[?] Central wavelength of the pulse [nm] = 1550
[?] Peak power [W] = 0
[?] Chirp constant = 0
[?] Half-width of the incident pulse [ps] : T0 = 4

 ****************************************
 ****      Optical fibre features      ****
 ****************************************

[?] Fibre length [km] = 5000
[?] Linear loss/gain coefficient alpha [km^-1] = 0
[?] Number of non-zeros beta_k (k>=2) coefficients = 1
[?] Value of beta_2 [ps^2 km^-1]= -0.1
[?] non-linear coefficient gamma [W^-1 km^-1] = 2.2
```

```
 ****************************************
 ****   Numerical parameters setting   ****
 ****************************************

[?] Size of the time window; winT [ps] = 400
[?] Number of time sampling points 2^p with p = 14
[?] Initial step-size length [m] = 1e3
[?] Tolerance value = 1e-6

 ------------------------------------------------
 Computations start on: Fri Jan 16 17:05:53 2015
 Computations end on: Fri Jan 16 17:05:57 2015
 Simulation lasted (s) : 3.851490
 ------------------------------------------------

 ****************************************
 **** Results (see also files in dir. [./res/])
 ****************************************
 Dispersion length = 1.600000e+02 km
 Non-linear length = 1.600000e+02 km
 Number of spatial steps = 481

 L^2-norm of the solution at fibre end = 2.145575e-01
 L^1-norm                              = 1.336797e+00
 L^infinity-norm                       = 5.070566e-02

 Press [Enter] to close figures and exit program ...
```

We have depicted in Fig. 13 the evolution of the lowly varying pulse envelope power $|A|^2$ as a function of time and space and the evolution of the power spectral density in dB as a function of wavelength and space. The figures were obtained thanks to the `plotspip.m` Matlab/Octave script. We can easily identify in the two figures the position of the Solitons collision. With the values considered for this simulation, the collision of the two Solitons is predicted to happen at a distance of 4161 km [1].

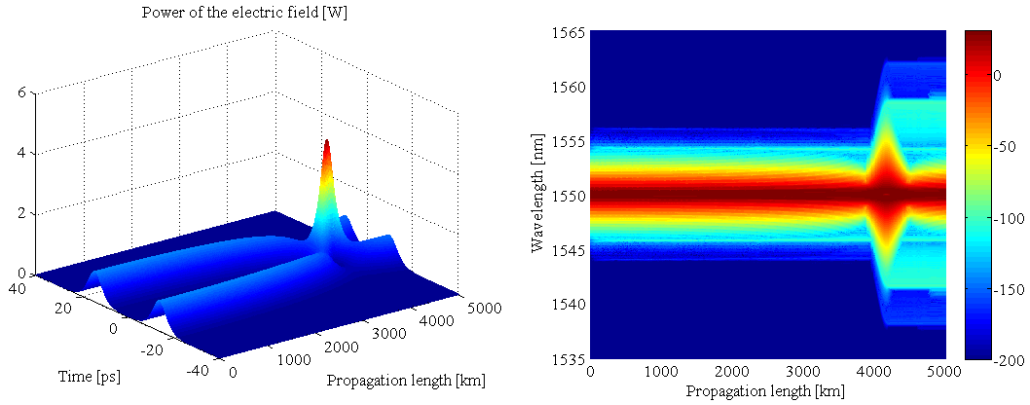

Figure 13: Slowly varying pulse envelope power $|A|^2$ in W (left) as a function of time and space and power spectral density in dB (right) as a function of wavelength and space for the propagation of two neighbouring Solitons in a 5000 km long fibre.

References

[1] G. Agrawal. *Nonlinear fiber optics*. Academic Press, 4th edition, 2013.

[2] S. Balac and A. Fernandez. A computer program implementing the Interaction Picture method for simulation of light-wave propagation in optical fibre. *Submitted to Comput. Phys. Commun.*, 2015.

[3] S. Balac, A. Fernandez, F. Mahé, F. Méhats, and R. Texier-Picard. The Interaction Picture method for solving the nonlinear Schrödinger equation in optics. *ESAIM:M2AN*, 2015.

[4] 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.

[5] C. Desem and P.L. Chu. Reducing soliton interaction in single-mode optical fibres. *Optoelectronics, IEE Proceedings J.*, 134(3):145–151, 1987.

[6] K. Okamoto. *Fundamentals of Optical Waveguides*. Optics and Photonics. Elsevier, 2006.