Solving the Nonlinear Schrödinger equation

Arash Mafi
Assistant Professor
Department of Electrical Engineering and Computer Science
University of Wisconsin-Milwaukee
3/11/2009

The following code is a simple solver for the nonlinear Schrödinger equation, according to Eq. 2.4.4 of the "Nonlinear Fiber Optics" by G. P. Agrawal.

We are going to do quite a bit of plotting. In order to simplify our life, we use the following functions to plot the absolute value, square of the absolute value, real part, and imaginary part of the array function.
The array is in the x-y format. The inputs are the array, color of the plot, and the number of points we would like to cut from both sides of the array for better visibility.

```
PlotAbsSignal[Efield_, color_, cut_] := Module[{},
   Clear[signal1];
   signal1 = Drop[Drop[Efield, cut], -cut];
   ListPlot[Transpose[{Transpose[signal1][[1]], Abs[Transpose[signal1][[2]]]}],
    PlotRange → All, Joined → True, PlotStyle → color]
  ];
PlotAbs2Signal[Efield_, color_, cut_] := Module[{},
   Clear[signal1];
   signal1 = Drop[Drop[Efield, cut], -cut];
   ListPlot[Transpose[{Transpose[signal1][[1]], Abs[Transpose[signal1][[2]]]^2}],
    PlotRange → All, Joined → True, PlotStyle → color]
  ];
PlotReSignal[Efield_, color_, cut_] := Module[{},
   Clear[signal1];
   signal1 = Drop[Drop[Efield, cut], -cut];
   ListPlot[Transpose[{Transpose[signal1][[1]], Re[Transpose[signal1][[2]]]}],
    PlotRange → All, Joined → True, PlotStyle → color]
  ];
PlotImSignal[Efield_, color_, cut_] := Module[{},
   Clear[signal1];
   signal1 = Drop[Drop[Efield, cut], -cut];
   ListPlot[Transpose[{Transpose[signal1][[1]], Im[Transpose[signal1][[2]]]}],
    PlotRange → All, Joined → True, PlotStyle → color]
  ];
```

The next two modules are used to calculate the discrete Fourier transform of a signal of time. It is often confusing how terms are ordered and given a time series, what the assigned frequencies are.
The input signal is in the form of a 2D array of {time,signal}.
FTsignal generates the Fourier transform where the ordering starts from zero frequency and goes to maximum positive frequency. Then starts from maximum negative frequency and goes back down to the nearest negative frequency to zero.
FTsignalNormal does the same, expect ordering is from the most negative to most positive frequencies. FTsignalNormal is great form making plots.

```
FTsignal[Efield_] := Module[{fsignal},
   Clear[signalin, dimsignal, totaltime, freqs, fsignal];
   signalin = (Transpose[Efield])[[2]];
   dimsignal = Dimensions[signalin][[1]];
   totaltime = (dimsignal) (Efield[[2, 1]] - Efield[[1, 1]]);
   freqs = 1.0/totaltime Join[Table[i, {i, 0, dimsignal/2 - 1}], Table[i, {i, -dimsignal/2, -1}]];
   fsignal = Fourier[signalin, FourierParameters → {1, -1}];
   fsignal = Transpose[{freqs, fsignal}];

   fsignal
  ];

FTsignalNormal[Efield_] := Module[{fsignal},
   Clear[signalin, dimsignal, totaltime, freqs, fsignal];
   signalin = (Transpose[Efield])[[2]];
   dimsignal = Dimensions[signalin][[1]];
   totaltime = (dimsignal) (Efield[[2, 1]] - Efield[[1, 1]]);
   freqs = 1.0/totaltime Join[Table[i, {i, 0, dimsignal/2 - 1}], Table[i, {i, -dimsignal/2, -1}]];
   fsignal = Fourier[signalin, FourierParameters → {1, -1}];
   fsignal = RotateRight[Transpose[{freqs, fsignal}], dimsignal/2];

   fsignal
  ];
```

How about generating moments of a time pulse according to Eq. 3.2.27 of the "Nonlinear Fiber Optics" by G. P. Agrawal. PulseMoments uses a time signal and outputs the moments up to the order m.

The convention in Eq. 3.2.27 generates a second moment $\sqrt{< T^2 >}$ that is different by a factor of $\sqrt{2}$ from T0 if applied to the Gaussian pulse of Eq. 3.2.7.
This is corrected in Pulse T0 where it outputs the proper second moment where it also accounts for the shift in time, so it produces T0 as the wisth, even if the pulse is shifted in time.

```
PulseMoments[Efield_, m_] := Module[{moments},
   dimsignal = Dimensions[Efield][[1]];
   moments = Table[(Sum[Efield[[j, 1]]^i Abs[Efield[[j, 2]]]^2, {j, 1, dimsignal}] / Sum[Abs[Efield[[j, 2]]]^2, {j, 1, dimsignal}])^(1.0/i), {i, 1, m}];
   Chop[moments]
  ];

PulseT0[Efield_] := Module[{moments},
   moments = PulseMoments[Efield, 2];
   Sqrt[2.0] Sqrt[Chop[moments[[2]]^2 - moments[[1]]^2]]
  ];
```

Here is a little module that generates the attenuation as used in NLSE from dB/km attenuation measurements.

```
AttenuationFromdBperKm[alphaIndBperkm_] := Module[{alphaOut},
  (* alpha is actually for power. There
   is factor of 2 difference between this and Demir's*)
  alphaOut = alphaIndBperkm / (10. Log[10, E]);
  alphaOut
 ];
```

Last but not least is the NLSE solver. Efield is the array of the electric field sampled at equal time intervals. It is in the format of {{t1,E1},{t2,E2},...}

```
NLSpropagate[Efield_, beta2_, beta3_, gamma_, loss_, lengthF_, numsteps_] := Module[{signal},

  Dhat[w_] = I beta2/2 w^2 - I beta3/6 w^3 - loss/2;
  Nhat[u_] = I Abs[u]^2 gamma;

  dz = lengthF/numsteps;


  signal = (Transpose[Efield])[[2]];
  dimsignal = Dimensions[signal][[1]];
  totaltime = (dimsignal) (Efield[[2, 1]] - Efield[[1, 1]]);
  omegas = 2.0 π/totaltime Join[Table[i, {i, 0, dimsignal/2 - 1}], Table[i, {i, -dimsignal/2, -1}]];


  Fsignal = Fourier[signal, FourierParameters → {1, -1}];
  Fsignal = Exp[Dhat[omegas] dz/2.0] * Fsignal;

  Do[{
    signal = InverseFourier[Fsignal, FourierParameters → {1, -1}];
    Nhatlist = Exp[Nhat[signal] dz];
    signal = Nhatlist * signal;
    Fsignal = Fourier[signal, FourierParameters → {1, -1}];
    Fsignal = Exp[Dhat[omegas] dz] * Fsignal;
   }, {run, 1, numsteps}];

  Fsignal = Exp[-Dhat[omegas] dz/2.0] * Fsignal;
  signal = InverseFourier[Fsignal, FourierParameters → {1, -1}];

  signal = Transpose[{Transpose[Efield][[1]], signal}];

  signal
 ];
```

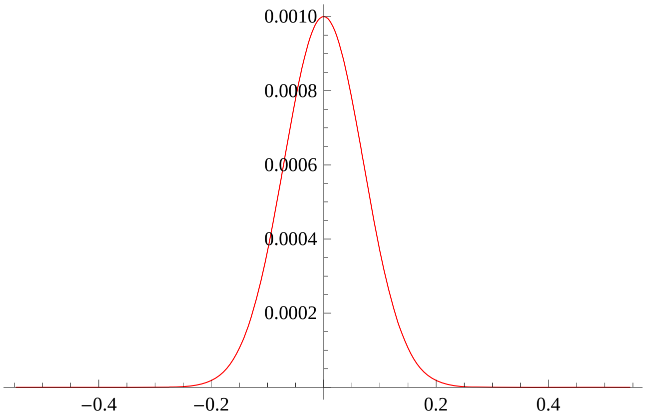
Here are some parameters that we use in our simulations.

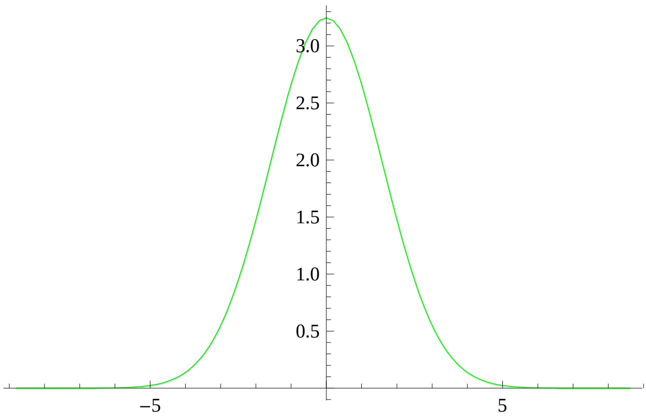
```
β2 = -20.0 * 10^-6;  (* 2nd order dispersion, -20.0 ps^2/km or -20.0*10^-6 ns^2/km*)
β3 = 0;                  (* 3rd order dispersion *)
γ = 2.0; (* nonlinearlity parameter, 2.0/W/km *)
length = 1000.0;          (* propagation length in km *)
α = AttenuationFromdBperKm[0]; (* This is loss parameter *)
power = 10^-3;(* This is optical power in Watts *)
```

The following is just a sample pulse based on Eq. 3.2.24. A bunch of other interesting parameters such as the nonlinear length, dispersion length, effective length, and maximum total nonlinear phase are calculated in here.

```
Clear[T, m, T0, C1, P0];
f[T_, m_, T0_, C1_, P0_] = Sqrt[P0] Exp[-(1 + I C1)/2.0 (T/T0)^(2 m)];

nt = 2048;
Tmax = 5.0;
t0 = 0.1;
order = 1;
chirp = 0;

dtau = Tmax/nt;
tau = Table[i * dtau, {i, -nt/2, nt/2 - 1}];

signal = Transpose[{tau, f[tau, order, t0, chirp, power]}];

cutf = 800;
PlotAbs2Signal[signal, Red, cutf]
cutf = 980;
PlotAbsSignal[FTsignalNormal[signal], Green, cutf]
LD = PulseT0[signal]^2/Abs[β2];
LNL = 1.0/(γ Max[Abs[Transpose[signal][[2]]]^2]);
Clear[x];
Leff = Limit[(1 - Exp[-x length])/x, x -> α];

Phimax = Leff/LNL;

MatrixForm[
 {{"length", length}, {"LD", LD}, {"LNL", LNL}, {"Leff", Leff}, {"Phimax", (Phimax / Pi) "π"}}]
```

$$f[T\_, m\_, T0\_, C1\_, P0\_] = \sqrt{P0}\,\mathrm{Exp}\left[-\frac{1 + i\,C1}{2.0}\left(\frac{T}{T0}\right)^{2m}\right];$$





length	1000.
LD	500.
LNL	500.
Leff	1000.
Phimax	0.63662 π

We also generate the signal moments and calculate T0 to show that it equals the input T0

```
PulseMoments[signal, 4]
PulseT0[signal]
```

{0, 0.0707107, 1.72032×10^{-7}, 0.0930605}

0.1

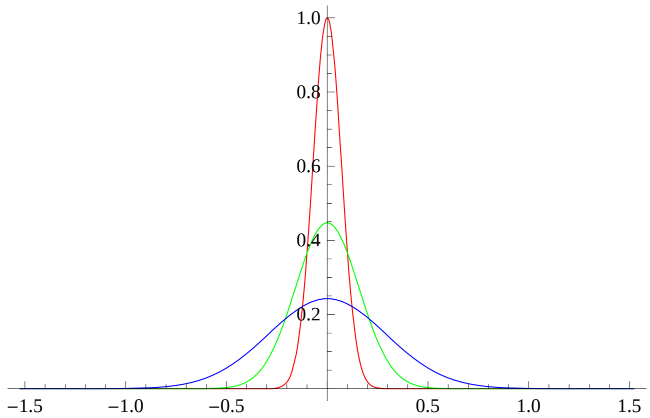
And now, some examples from chapter 3 of the "Nonlinear Fiber Optics" by G. P. Agrawal.

Figure 3.1:

```
β2 = -20.0 * 10^-6;  (* 2nd order dispersion, -20.0 ps^2/km or -20.0*10^-6 ns^2/km*)
β3 = 0;                  (* 3rd order dispersion *)
γ = 10^-18; (* nonlinearlity parameter, 0.0/W/km *)
α = AttenuationFromdBperKm[0]; (* This is loss parameter *)
power = 1.0;(* This is optical power in Watts *)

nt = 2048;
Tmax = 5.0;
t0 = 0.1;
order = 1;
chirp = 0;

dtau = Tmax/nt;
tau = Table[i * dtau, {i, -nt/2, nt/2 - 1}];

signal = Transpose[{tau, f[tau, order, t0, chirp, power]}];

LD = PulseT0[signal]^2/Abs[β2];
LNL = 1.0/(γ Max[Abs[Transpose[signal][[2]]]^2]);
Clear[x];
Leff = Limit[(1 - Exp[-x length])/x, x -> α];

Phimax = Chop[Leff/LNL];
Clear[length];

MatrixForm[
 {{"length", length}, {"LD", LD}, {"LNL", LNL}, {"Leff", Leff}, {"Phimax", (Phimax / Pi) "π"}}]
```

length	length
LD	500.
LNL	$1. \times 10^{18}$
Leff	length
Phimax	0

```
signalA = NLSpropagate[signal, β2, β3, γ, α, 0 * LD, 10];
signalB = NLSpropagate[signal, β2, β3, γ, α, 2 * LD, 10];
signalC = NLSpropagate[signal, β2, β3, γ, α, 4 * LD, 10];
cutf = 400;
Show[PlotAbs2Signal[signalA, Red, cutf],
 PlotAbs2Signal[signalB, Green, cutf], PlotAbs2Signal[signalC, Blue, cutf]]

Clear[phiB, dwB, phiC, dwC, t];
phiB = Interpolation[Drop[Drop[Transpose[
      {Transpose[signalB][[1]], -I * Log[Transpose[signalB][[2]] / Abs[Transpose[signalB][[2]]]]}], cutf], -cutf]];
dwB[t_] = -t0 D[phiB[t], t];
phiC = Interpolation[Drop[Drop[Transpose[
      {Transpose[signalC][[1]], -I * Log[Transpose[signalC][[2]] / Abs[Transpose[signalC][[2]]]]}], cutf], -cutf]];
dwC[t_] = -t0 D[phiC[t], t];
Show[Plot[dwB[t t0], {t, -Tmax/(4.95 t0), Tmax/(5 t0)}, PlotRange → All, PlotStyle → Green],
 Plot[dwC[t t0], {t, -Tmax/(5 t0), Tmax/(5 t0)}, PlotRange → All, PlotStyle → Blue]]
```



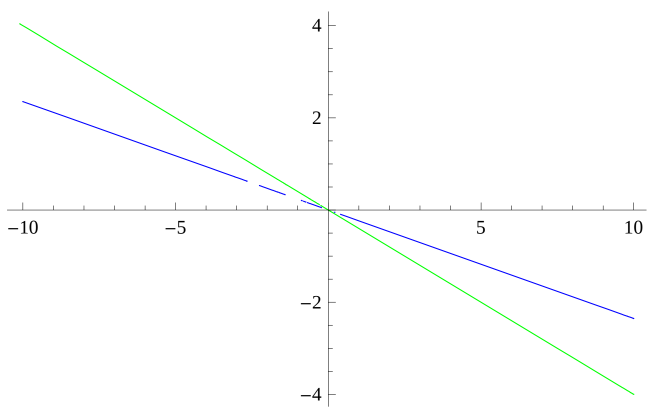


Figure 3.2:

```
β2 = -20.0 * 10^-6; (* 2nd order dispersion, -20.0 ps^2/km or -20.0*10^-6 ns^2/km*)
β3 = 0;                  (* 3rd order dispersion *)
γ = 10^-18; (* nonlinearlity parameter, 0.0/W/km *)
α = AttenuationFromdBperKm[0]; (* This is loss parameter *)
power = 1.0;(* This is optical power in Watts *)

nt = 2048;
Tmax = 5.0;
t0 = 0.1;
order = 1;
chirp = 2.0;

dtau = Tmax/nt;
tau = Table[i * dtau, {i, -nt/2, nt/2 - 1}];

signal = Transpose[{tau, f[tau, order, t0, chirp, power]}];

LD = PulseT0[signal]^2/Abs[β2];
LNL = 1.0/(γ Max[Abs[Transpose[signal][[2]]]^2]);
Clear[x];
Leff = Limit[(1 - Exp[-x length])/x, x -> α];

Phimax = Chop[Leff/LNL];
Clear[length];

MatrixForm[
 {{"length", length}, {"LD", LD}, {"LNL", LNL}, {"Leff", Leff}, {"Phimax", (Phimax / Pi) "π"}}]
```

length	length
LD	500.
LNL	$1.\times 10^{18}$
Leff	length
Phimax	0

```
signalA = NLSpropagate[signal, β2, β3, γ, α, 0 * LD, 10];
signalB = NLSpropagate[signal, β2, β3, γ, α, 0.5 * LD, 10];
signalC = NLSpropagate[signal, β2, β3, γ, α, 2 * LD, 10];
cutf = 800;
Show[PlotAbs2Signal[signalA, Red, cutf],
 PlotAbs2Signal[signalB, Green, cutf], PlotAbs2Signal[signalC, Blue, cutf]]
```

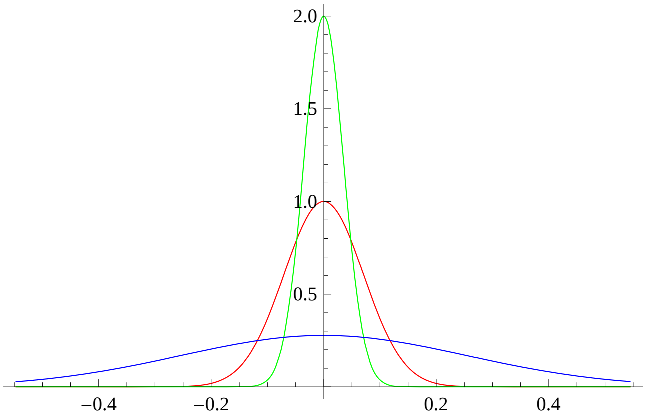


Figure 4.1:

```
β2 = 1.0 * 10^-18; (* 2nd order dispersion, 0.0 ps^2/km or 0.0*10^-6 ns^2/km*)
β3 = 0;                    (* 3rd order dispersion *)
γ = 2.0; (* nonlinearlity parameter, 2.0/W/km *)
α = AttenuationFromdBperKm[0]; (* This is loss parameter *)
power = 1.0;(* This is optical power in Watts *)

nt = 2048;
Tmax = 5.0;
t0 = 0.1;
order = 1;
chirp = 0;

dtau = Tmax/nt;
tau = Table[i * dtau, {i, -nt/2, nt/2 - 1}];

signal = Transpose[{tau, f[tau, order, t0, chirp, power]}];

LD = PulseT0[signal]^2/Abs[β2];
LNL = 1.0/(γ Max[Abs[Transpose[signal][[2]]]^2]);
Clear[x];
Leff = Limit[(1 - Exp[-x length])/x, x -> α];

Phimax = Chop[Leff/LNL];
Clear[length];

MatrixForm[
 {{"length", length}, {"LD", LD}, {"LNL", LNL}, {"Leff", Leff}, {"Phimax", (Phimax / Pi) "π"}}]

signalA = NLSpropagate[signal, β2, β3, γ, α, LNL, 100];

order = 3;
signal = Transpose[{tau, f[tau, order, t0, chirp, power]}];

signalB = NLSpropagate[signal, β2, β3, γ, α, LNL, 100];
```

length	length
LD	$1. \times 10^{16}$
LNL	0.5
Leff	length
Phimax	$0.63662\,\pi$ length

```
cutf = 900;
Show[PlotAbs2Signal[signalA, Red, cutf], PlotAbs2Signal[signalB, Green, cutf]]

Clear[phiA, dwA, phiB, dwB, t];
phiA = Interpolation[Drop[Drop[Transpose[
    {Transpose[signalA][[1]], -I * Log[Transpose[signalA][[2]] / Abs[Transpose[signalA][[2]]]]}], cutf], -cutf]];
dwA[t_] = -t0 D[phiA[t], t];

phiB = Interpolation[Drop[Drop[Transpose[
    {Transpose[signalB][[1]], -I * Log[Transpose[signalB][[2]] / Abs[Transpose[signalB][[2]]]]}], cutf], -cutf]];
dwB[t_] = -t0 D[phiB[t], t];
Show[Plot[dwA[t t0], {t, -Tmax/(30 t0), Tmax/(30 t0)}, PlotRange → All, PlotStyle → Red],
 Plot[dwB[t t0], {t, -Tmax/(30 t0), Tmax/(30 t0)}, PlotRange → All, PlotStyle → Green]]
```

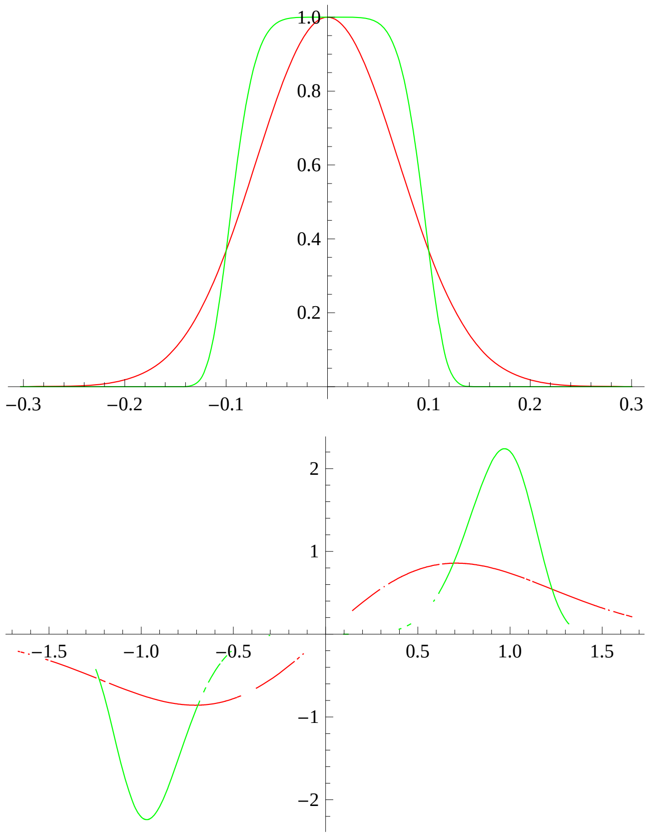


Figure 4.2:

```
β2 = 1.0 * 10^-18;  (* 2nd order dispersion, 0.0 ps^2/km or 0.0*10^-6 ns^2/km*)
β3 = 0;                  (* 3rd order dispersion *)
γ = 2.0; (* nonlinearlity parameter, 2.0/W/km *)
α = AttenuationFromdBperKm[0]; (* This is loss parameter *)
power = 0.7853981633974483;(* This is optical power in Watts *)

nt = 2048;
Tmax = 5.0;
t0 = 0.1;
order = 1;
chirp = 0;
```

$$dtau = \frac{Tmax}{nt};$$

$$tau = Table\left[i * dtau, \left\{i, -\frac{nt}{2}, \frac{nt}{2} - 1\right\}\right];$$

```
signal = Transpose[{tau, f[tau, order, t0, chirp, power]}];
```

$$LD = \frac{PulseT0[signal]^2}{Abs[\beta_2]};$$

$$LNL = \frac{1.0}{\gamma\, Max[Abs[Transpose[signal][[2]]]\text{^}2]};$$

```
Clear[x];
```

$$Leff = Limit\left[\frac{1 - Exp[-x\ length]}{x}, x \to \alpha\right];$$

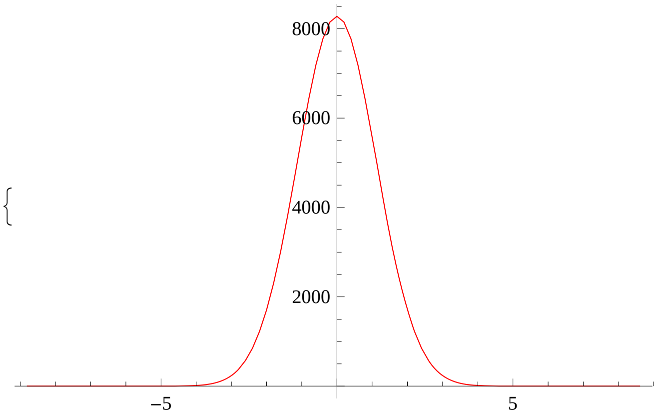
$$Phimax = Chop\left[\frac{Leff}{LNL}\right];$$

```
Clear[length];

MatrixForm[
 {{"length", length}, {"LD", LD}, {"LNL", LNL}, {"Leff", Leff}, {"Phimax", (Phimax / Pi) "π"}}]
```

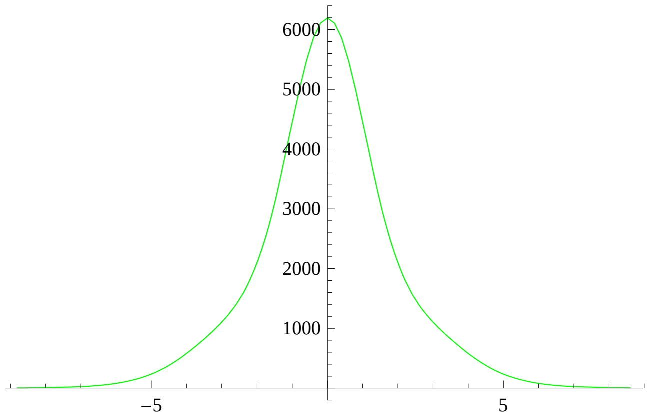
$$\begin{pmatrix} length & length \\ LD & 1.\times 10^{16} \\ LNL & 0.63662 \\ Leff & length \\ Phimax & 0.5\,\pi\ length \end{pmatrix}$$

```
signalA = NLSpropagate[signal, β2, β3, γ, α, 0.0, 100];
signalB = NLSpropagate[signal, β2, β3, γ, α, 1.0, 100];
signalC = NLSpropagate[signal, β2, β3, γ, α, 2.0, 100];
signalD = NLSpropagate[signal, β2, β3, γ, α, 3.0, 100];
signalE = NLSpropagate[signal, β2, β3, γ, α, 5.0, 100];
signalF = NLSpropagate[signal, β2, β3, γ, α, 7.0, 100];

cutf = 980;
{PlotAbs2Signal[FTsignalNormal[signalA], Red, cutf],
 PlotAbs2Signal[FTsignalNormal[signalB], Green, cutf],
 PlotAbs2Signal[FTsignalNormal[signalC], Blue, cutf]}
cutf = 900;
{PlotAbs2Signal[FTsignalNormal[signalD], Red, cutf],
 PlotAbs2Signal[FTsignalNormal[signalE], Green, cutf],
 PlotAbs2Signal[FTsignalNormal[signalF], Blue, cutf]}
```

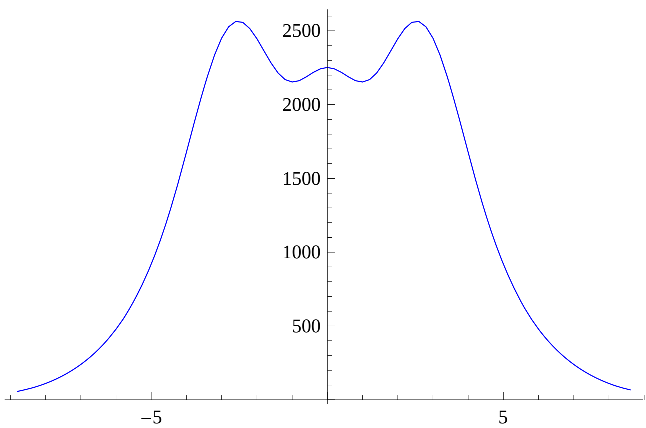
{


8000
6000
4000
2000
−5
5

,


6000
5000
4000
3000
2000
1000
−5
5

,


2500
2000
1500
1000
500
−5
5

}

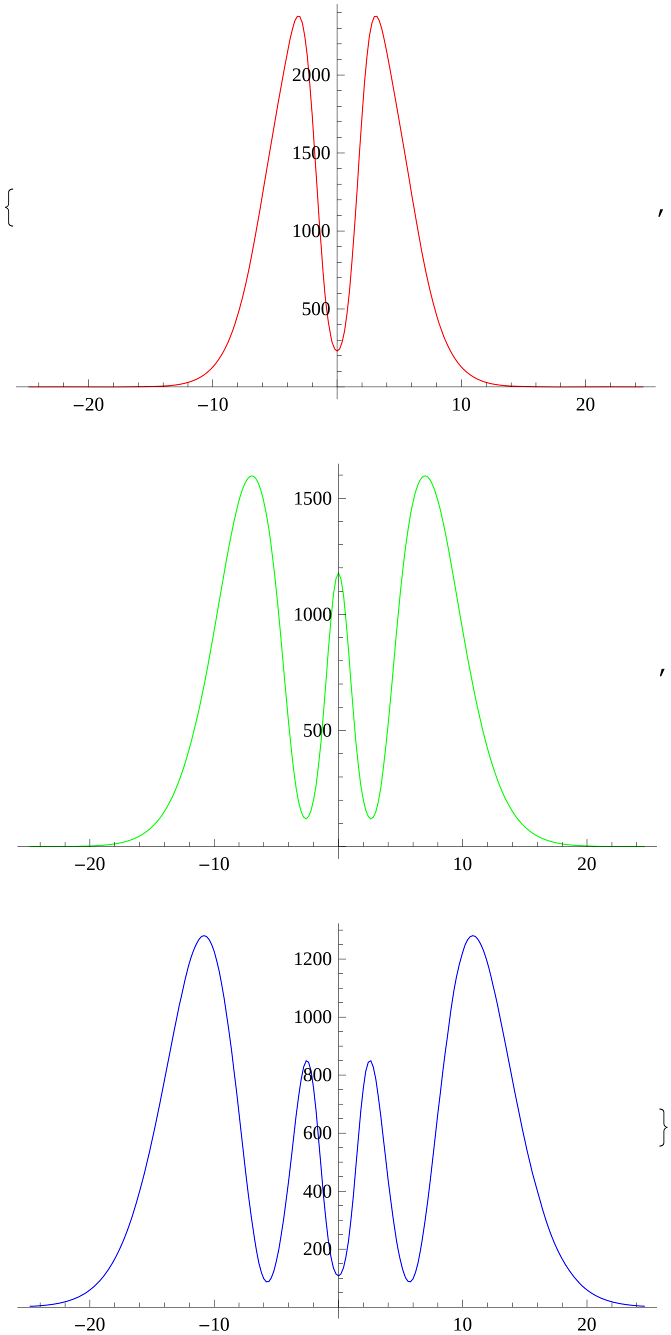

2000
1500
1000
500
−20
−10
10
20
1500
1000
500
−20
−10
10
20
1200
1000
800
600
400
200
−20
−10
10
20

Figure 4.5:

```
β2 = 1.0 * 10^-18;  (* 2nd order dispersion, 0.0 ps^2/km or 0.0*10^-6 ns^2/km*)
β3 = 0;                  (* 3rd order dispersion *)
γ = 2.0; (* nonlinearlity parameter, 2.0/W/km *)
α = AttenuationFromdBperKm[0]; (* This is loss parameter *)
power = 0.7853981633974483;(* This is optical power in Watts *)
length = 9.0; (*length in km to make nonlinear phase equal to 4.5π*)

nt = 2048;
Tmax = 5.0;
t0 = 0.1;
order = 1;
chirp = 0;
```

$$dtau = \frac{Tmax}{nt};$$

$$tau = Table\left[i * dtau, \left\{i, -\frac{nt}{2}, \frac{nt}{2} - 1\right\}\right];$$

$$LD = \frac{PulseT0[signal]^2}{Abs[\beta_2]};$$

$$LNL = \frac{1.0}{\gamma\, Max[Abs[Transpose[signal][[2]]]^2]};$$

```
Clear[x];
```

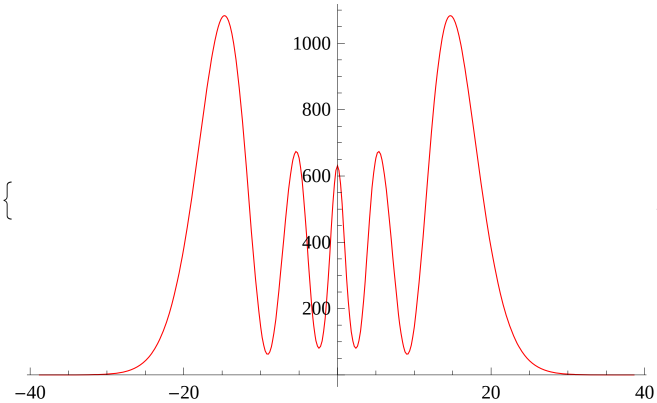
$$Leff = Limit\left[\frac{1 - Exp[-x\, length]}{x}, x \to \alpha\right];$$

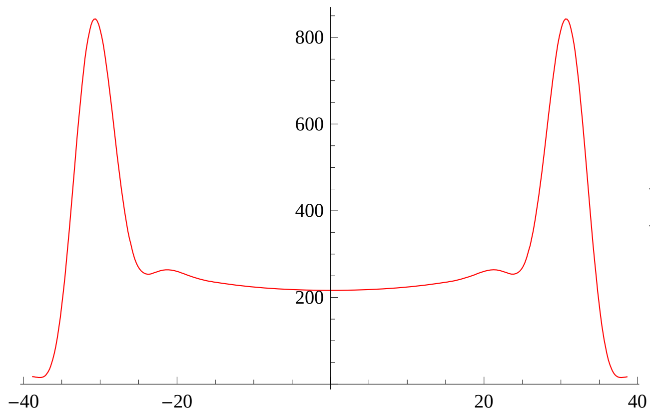
$$Phimax = Chop\left[\frac{Leff}{LNL}\right];$$

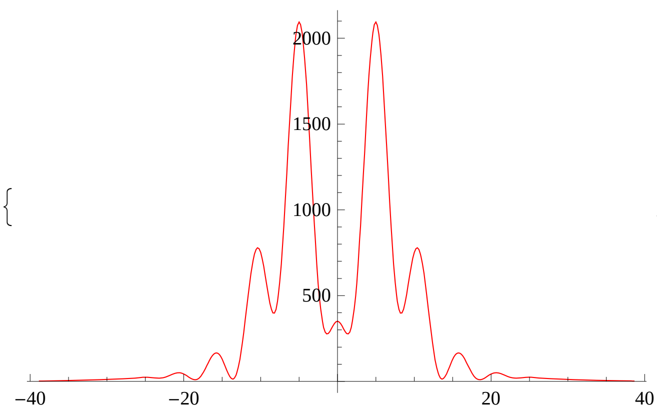
```
MatrixForm[
 {{"length", length}, {"LD", LD}, {"LNL", LNL}, {"Leff", Leff}, {"Phimax", (Phimax / Pi) "π"}}]
```

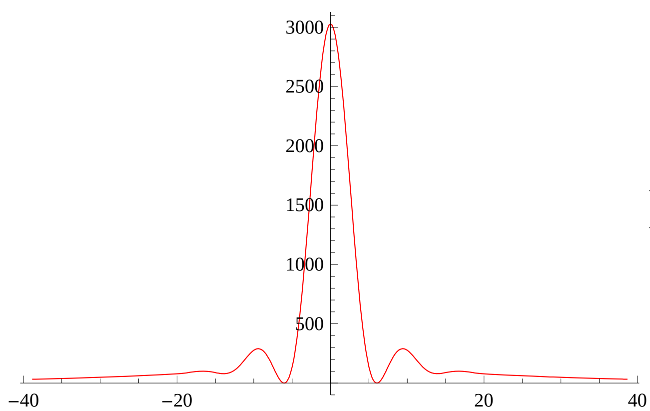
length	9.
LD	$1. \times 10^{16}$
LNL	0.63662
Leff	9.
Phimax	4.5 π

```
signal = Transpose[{tau, f[tau, order, t0, 0, power]}];
signalA = NLSpropagate[signal, β2, β3, γ, α, length, 100];
signal = Transpose[{tau, f[tau, order, t0, 10, power]}];
signalB = NLSpropagate[signal, β2, β3, γ, α, length, 100];
signal = Transpose[{tau, f[tau, order, t0, -10, power]}];
signalC = NLSpropagate[signal, β2, β3, γ, α, length, 100];
signal = Transpose[{tau, f[tau, order, t0, -20, power]}];
signalD = NLSpropagate[signal, β2, β3, γ, α, length, 100];

cutf = 830;
{PlotAbs2Signal[FTsignalNormal[signalA], Red, cutf],
 PlotAbs2Signal[FTsignalNormal[signalB], Red, cutf]}
{PlotAbs2Signal[FTsignalNormal[signalC], Red, cutf],
 PlotAbs2Signal[FTsignalNormal[signalD], Red, cutf]}
```

{ 
1000
800
600
400
200
−40
−20
20
40
,


800
600
400
200
−40
−20
20
40
}

{ 
2000
1500
1000
500
−40
−20
20
40
,


3000
2500
2000
1500
1000
500
−40
−20
20
40
}