Method of lines solution to the transient SBS equations for nanosecond Stokes pulses

F. S. Gökhan
Department of Electrical and Electronics Engineering, Havaalanı Yolu üzeri 8. km, Hasan Kalyoncu University, Sahinbey, Gaziantep, Turkey

G. W. Griffiths
School of Engineering and Mathematical Sciences, City University, London, UK

W. E. Schiesser
Engineering and Mathematics, Lehigh University, Bethlehem, PA, USA

The spectral and temporal evolution of distributed sensing based on stimulated Brillouin scattering (SBS) in optical fibers for several-nanosecond Stokes pulses is demonstrated by using the method of lines (MOL) solution of the transient SBS equations. A superbee flux limiter is utilized to avoid numerical damping and dispersion that would otherwise be brought on by the approximation of spatial derivatives associated with the partial differential equations (PDEs). In order to increase computational efficiency, an approach is adopted whereby the sparse PDE Jacobian matrix integrator option of the ODE solver(s) is employed. Simulation examples of SBS-based sensing for fibers containing sections with different Brillouin frequencies are presented. To the best of our knowledge, this MOL solution is proposed for the first time for modeling of the transient SBS equations for nanosecond Stokes pulses with different waveforms in a SBS based fiber optic sensor. [DOI: http://dx.doi.org/10.2971/jeos.2013.13049]

Keywords: Brillouin scattering, method of lines, finite differences, simulation, flux limiter, sparse matrix integrator

1 INTRODUCTION

Stimulated Brillouin scattering (SBS) in optical fibers permits us to measure temperature and/or strain on a truly distributed basis, over kilometric ranges with high resolution [1]–[3]. It consists in the coupling between two counter-propagating optical waves, the pump and the Stokes wave, and an acoustic wave. In this three-wave mixing process, power is transferred from the pump light wave to the Stokes light wave (that is, to the light wave having a lower frequency) and also to the acoustic wave. The interaction strength depends on the frequency offset between the two light waves and attains its maximum at the so-called Brillouin frequency \( \nu_B \). As the Brillouin frequency shift changes linearly with temperature and strain, a distributed temperature-strain sensor can be realized using SBS.

The theoretical model used to simulate the sensing in single-mode fibers is based on the wave equations for the pump and Stokes optical waves, the pump and the Stokes wave, and also to the acoustic wave. Therefore, the evolution of acoustic wave equation for the field \( \rho(z,t) \) with second-order time derivative follows [5]:

\[
\frac{\partial^2 \rho}{\partial t^2} + (\Gamma_B - i2\omega_B) \frac{\partial \rho}{\partial t} - i\Gamma_B \omega_B \rho = \frac{\gamma e \omega_0^2}{4\pi} A_p A_s^* \quad (1c)
\]

Simulation has proven to be a valuable tool in understanding SBS based distributed sensor systems. Accurate simulation models allow for the rapid design and optimization of both sensor infrastructure and signal processing techniques. Visualizaton of the spatiotemporal behavior of all three fields involved in SBS, at all points through the fiber, and is especially desirable in investigating phenomena whose origins may not be clear from the ‘observable’ data in an experiment [6]–[8].

If long pulses are assumed, the time dependence in Eq. (1) can be neglected, resulting in a simple and accurate solution to the problem. However, in the short pulse regime, the temporal derivatives cannot be neglected and the solution becomes more complex [9]. For this solution, high speed and accuracy with the low complexity is highly desirable. In general, for the solution of Eq. (1), conventional first order finite difference time domain (FDTD) solution schemes are preferred since they offer simplicity with respect to the form of the solution, although they suffer from several drawbacks, e.g. they require more spatial nodes to achieve the same accuracy as...
higher order schemes, and this often results in longer computer run-times.

To solve Eq. (1), a numerical method based on the Simpson’s rule to approximate temporal integrals was introduced by [10]. However, they used an implicit method and employed non-linearization by assuming that \( |E_n^{m+1}| \approx |E_n^m| \) and this replacement weakens the coupling of the two laser fields and hence the results deviate from the exact solutions. In Ref. [6] the authors proposed a modified solution which is valid only when the time step size is equal to spatial step size. Moreover, their solution differs considerably from the real solution at locations where abrupt change takes place. Recently, we introduced the MOL solution of the SBS equations with appropriate flux limiter and sparse matrix integrator [11]. In this paper, we extent the results in Ref. [11] while computing a system of three second-order hyperbolic PDEs (Eqs. (1)).

In this paper we introduce, for the first time to the best of our knowledge, the solution of transient SBS equations for nanosecond Stokes pulses using Matlab ODE solvers [12]. By using this solution, the spectral and temporal evolution of distributed sensing based on stimulated Brillouin scattering (SBS) in optical fibers for several-nanosecond Stokes pulses is demonstrated. We introduce the solution technique and have adapted the software to provide a fast and stable solution by employing sparse matrix techniques and by using an appropriate flux limiter. Briefly, our solution technique combined with the Matlab software abilities provides a powerful method of simulating this SBS problem.

In arriving at a solution, the problem domain is first discretized over a finite grid and organized in a way that provides solutions for the optical and acoustic fields. The resolution of the solution can thus be varied to suit computational requirements. We have also employed an appropriate flux limiter to combat numerical damping/dispersion which avoids spurious oscillations in the solution. We have used software which identifies the sparsity pattern of the solution which is then supplied as an input to the solver. Additionally, we have determined the appropriate Matlab ODE solver for this particular application. By use of an appropriate sparse matrix ode solver along with an effective flux limiter, for nanosecond pulse widths, it is shown that the computation can be successfully performed in a reasonable computation time without the damping factor exceeding 3%. Examples are presented, showing the utility of this efficient simulation technique.

2 PROBLEM FORMULATION

2.1 The Dynamics of Stimulated Brillouin Scattering (SBS)

In this paper we consider a system whereby continuous wave (CW) light, at frequency, \( \nu_p \), is injected into the fiber at position \( z = L \). Pulses of light, known as Stokes pulses, are injected into the fiber at \( z = 0 \), at frequency, \( \nu_s \). Due to the SBS, there will be coupling of a counter propagating CW pump wave and a Stokes pulse wave via an induced acoustic wave. An enhanced interaction between the two beams occurs when the frequency difference of the lasers matches the frequency of the longitudinal acoustic phonons of the optical fiber. There is then a transfer of energy from the high frequency beam (pump) to the low frequency beam (Stokes). The amount of loss of the pump is recorded at \( z = 0 \) as a function of the frequency difference in the form of the Brillouin spectrum. Thus the power of the CW field intensity over time at \( z = 0 \) corresponds to the available time domain information in a real sensing experiment.

A distributed Brillouin sensor, based on a Brillouin Optical Time Domain Analysis (BOTDA), is a device where Stokes and pump beams counter-propagate and where frequency and time dependent variations of the Stokes (or pump) intensity is detected. These sensors have been shown to provide the best performances in terms of sensing length, spatial resolution, temperature, and, strain accuracies [2, 13]. The interaction is a maximum when the optical frequency difference between pump (\( \nu_p \)) and Stokes (\( \nu_s \)) corresponds to the Brillouin frequency (\( \nu_B \)). At the Brillouin frequency, the Stokes beam is amplified at the expense of the pump. \( \nu_B \) is the longitudinal acoustic phonon frequency and is a local material signature. An increase (decrease) in temperature or strain induces a proportional increase (decrease) of \( \nu_B \). The sensor detects the Brillouin frequency variation as a function of position. The distributed nature of the sensor is achieved by field modulation of the probe beam. The topology of this sensor is shown in Figure 1 [14].

2.2 Theoretical Model

By making suitable transformation, Eqs. (1) can be written as in Eqs. (3) which is a system of three second-order hyperbolic
After propagating through the fiber, it is assumed that the pump and Stokes pulses exit the fiber without reflection. The pump pulse \( T \) determines the spatial resolution [15].

PDEs,

\[
\begin{align*}
\partial E_p/\partial t - \nu \partial^2 E_p/\partial z^2 &= -E_a \cdot E_s \quad (3a) \\
\partial E_a/\partial t + \nu \partial^2 E_a/\partial z^2 &= E_s^* E_p \quad (3b) \\
\partial^2 E_a/\partial t^2 + 2 (\omega - j\Omega) \cdot \partial E_a/\partial t + \cdots &= \left( \Omega_B^2 - \Omega^2 - 2j\Gamma\Omega \right) E_a = jg E_s E_s^* \quad (3c)
\end{align*}
\]

where \( E_p, E_a, \) and \( E_s \) are the pump, Stokes, and acoustic fields, respectively, and \( j \) represents the imaginary number \( \sqrt{-1}. \) \( \Gamma = 1/\tau_{ph} \) is the relaxation rate \( \omega = \omega_p - \omega_v, \) \( \omega_p, \omega_v \) are the acoustic field, pump, and Stokes frequencies, respectively, \( \Omega_B \) is the Brillouin frequency, \( g = i\Gamma\Omega_B \) and \( g_B \) is the SBS gain factor. Here, \( \tau_{ph} \) is the phonon lifetime and it is \( 10^{11}s \) for silica fibers.

2.3 Boundary and Initial Conditions

The CW light from a laser source representing the pump field, \( E_p(z = L, t) \) is injected into the fiber at \( z = L \). Therefore, the boundary condition of the pump wave is:

\[
E_p(z = L, t) = E_p(z = L) \quad (4)
\]

where \( E_p(z = L) \) is the power of the CW laser. At the opposite end, the Stokes field, \( E_s \), is introduced via a pulsed source. We assume a three-section pump pulse of the form (Fig. 1a):

\[
\begin{align*}
E_s(z, t) &= E_s^0 [a \cdot u(t - z/v) + \cdots] \\
&= (\beta - \alpha) \cdot u(t - t_0 - z/v) + \cdots \quad (5a) \\
E_s(z = 0, t) &= E_s^0 [a \cdot u(t) + \cdots] \\
&= (\beta - \alpha) \cdot u(t - t_0) + \cdots \quad (5b)
\end{align*}
\]

where \( a, \beta, \gamma \) are amplitudes of the corresponding time intervals. Here \( u(\cdot) \) represent the Heaviside unit step function. The first section of the pulse of height \( a A_0^0 \) and width \( t_0 \) enters the fiber \( (z = 0) \) at \( t = 0 \), followed by a \( \beta A_0^0 \)-high, T-wide middle section, ending with an infinitely long section of height \( \gamma A_0^0 \).

In Matlab \( u(\cdot) \) can be implemented via the function \texttt{heaviside} function. For the Bright pulse with a CW component waveform the base level of the Stokes pulse is characterized by the extinction ratio \( ER = 10 \log(\beta/\alpha) \).

After propagating through the fiber, it is assumed that the pump and Stokes pulses exit the fiber without reflection. The acoustic field is assumed to be zero everywhere at \( t = 0 \), and the CW light and Stokes leakage intensities turn on everywhere at \( t = 0 \). Therefore the initial conditions are [6],

\[
E_p(z, t = 0) = E_p(z = L) \quad (6a)
\]

\[
E_s(z, t = 0) = \beta \text{ for intensity pulse and } \alpha \text{ for the bright pulse with a CW component, dark pulse waveform and } \pi \text{-phase pulse, respectively.} \quad (6b)
\]

2.4 AC Detection of the pump output in the case of a finite CW component (base) of the Stokes pulse

Distributed sensing based on stimulated Brillouin scattering (SBS) is obtained by temporally resolved Brillouin spectra, i.e. the depleted output pump as a function of the detuning from the Brillouin frequency. The Brillouin frequency gives the strain and/or temperature along the fiber. The time-dependent output pump power or the time-dependent deviation from the stationary output power, induced by the Stokes pulse, is measured in the experiment. There are two possibilities of pump loss definition as DC or AC detection, respectively. However, AC detection is most likely to be used in the experiment to measure the small relative variations of the output pump signal in the case of several-nanosecond Stokes pulses. The pump loss defined as AC detection, \( \alpha_{AC} = E_p(z = 0, t = 0) - E_p(z = 0, t), \) each as a function of the time and frequency detuning \( \Delta\Omega = \Omega - \Omega_B, \) represent the Brillouin spectrum. Here \( E_p(z = 0, t) \) is the time-dependent output pump power at \( z = 0, E_p(z = 0, t = 0) \) is the output pump power at the initial time moment, defined by the stationary pump-base SBS before the Stokes pulse arrives in the fiber [7].

3 NUMERICAL SOLUTION METHOD

3.1 Finite Difference Time Domain (FDTD) Solution

To obtain a solution to Eqs. (3a)-(3c), we follow the authors in Ref. [9] and substitute the following:

\[
\begin{align*}
E_p &= A_p e^{j\phi_p} \quad (7a) \\
E_s &= A_s e^{j\phi_s} \quad (7b) \\
E_a &= A_a e^{j\phi_a} \quad (7c)
\end{align*}
\]

\[
\begin{align*}
\partial E_s/\partial t &= e^{j\phi_s} \left( \partial^2 A_s/\partial t^2 - A_s \left( \partial^2 \phi_a/\partial t^2 \right) \right) + \cdots \quad (7b)
\end{align*}
\]
into Eqs. (3a)-(3c) and with making transformation \( z' = \frac{z}{17r} \cdot \gamma_a \) to remove \( v \) yields the equations,

\[
\begin{align*}
\frac{\partial A_s}{\partial t} + \frac{\partial A_d}{\partial z} &= A_p A_s \cdot \cos(\Phi) \\
\frac{\partial A_d}{\partial t} - \frac{\partial A_d}{\partial z} &= -A_s A_d \cdot \cos(\Phi) \\
\frac{\partial^2 A_d}{\partial t^2} &= -g A_p A_d \cos \Phi - \cdots
\end{align*}
\]

(8a) for the first derivatives in \( z \) so that \( E_p \approx (E_p - E_{p-1})/\Delta z \), where \( i \) is an index designating a position along a grid in \( z \) and \( \Delta z \) is the spacing in \( z \) along the grid (\( i - 1 \) designates the upward direction). For the MOL solution of Eq. (1b), \( E_p(z,t) \) can also be approximated by upwind FD approximations, i.e., \( E_s \approx (E_{s,j+1} - E_{s,j})/\Delta z (i + 1 \) designates the upward direction).

For the discretization of Eqs. (8a)-(8f), variables are defined as \( y_1 = A_s, y_2 = A_p, y_3 = A_d, y_4 = dy_3/dt, y_5 = \Phi, y_6 = dy_5/dt \). The boundary conditions are \( y_1(z = 0,t) = E_s(z = 0,t), y_2(z = L,t) = E_p(z = L) \). The initial conditions are \( y_1(z,t = 0) = \beta \) for intensity pulse and \( y_2(z,t = 0) = \alpha \) for the bright pulse with a CW component, dark pulse waveform and \( \pi \)-phase pulse, respectively. \( y_3(z,t = 0) = 1 \cdot 10^{-11}, y_4(z,t = 0) = 0, y_5(z,t = 0) = 0, y_6(z,t = 0) = 0 \). By discretizing Eqs. (10a)-(10f) in the space variable \( z \), the resulting system of ODEs is as follows:

\[
\begin{align*}
\frac{dy_1}{dt} &= -(y_1 - y_{i-1})/(x_i - x_{i-1}) - \cdots \\
y_2 \cdot y_3 \cdot \cos(\Phi) &< 2 \leq i \leq N \\
y_1 &= E_s(z = 0,t) \\
\frac{dy_2}{dt} &= -(y_2 - y_{i+1})/(x_i - x_{i+1}) - \cdots \leq 2 \leq i \leq N \\
y_1 \cdot y_3 \cdot \cos(\Phi) &2 \leq i \leq N \\
y_2 N &= E_p(z = L) \\
\frac{dy_3}{dt} &= y_4, \\
\frac{dy_4}{dt} &= -g \cdot y_1 \cdot y_2 \cdot \sin(\Phi) - \cdots \\
(2 \cdot y_4 + 2 \cdot y_3 \cdot \Omega \cdot y_6) &\leq 1 \leq N \leq \text{Solve the system of ODEs} \\
y_3 \cdot (y_6)^2 &\leq y_4 \cdot \Omega \cdot y_5 \cdot y_3 &1 \leq i \leq N \\
\frac{dy_5}{dt} &= y_6, 1 \leq i \leq N \\
\frac{dy_6}{dt} &= -(y_2 \cdot y_2 \cdot \sin(\Phi) - \cdots \\
2 \cdot y_4 \cdot \Omega \cdot (y_4 + \Gamma \cdot y_3) &\leq 1 \leq N \\
2 \cdot y_6 \cdot (y_4 + \Gamma \cdot y_3) &1 \leq i \leq N \\
\end{align*}
\]

The handling of Eqs. (10a)-(10f) is discussed in the appendix. The solver integrates the PDEs via the Matlab statement, \([t,u]=ode23t(@(pde,tout,u0,options);\) and options, which include tolerances and Jacobian sparsity information, are supplied to the solver.

### 3.2 Application of a Sparse Matrix Integrator

For a large system of ODEs, it is typical that only a few components of \( y \) appear in each equation. If component \( j \) of \( y \) does not appear in component \( i \) of \( f(t,y) \), then the partial derivative \( f_i/y_j \) is zero. If most of the entries of a matrix are zero, the matrix is said to be sparse. By storing only the nonzero entries of a sparse Jacobian, storage is reduced from the square of the number of equations \( d \) to a modest multiple of \( d \). If the Jacobian is sparse then so is the iteration matrix. As with storage, the cost of solving linear systems by elimination can be

\[
Z_{\text{transformed}} = \begin{cases} 
0 \leq \frac{z}{v} \gamma_a \leq \text{fibers length}, \\
0 \leq t_{\text{real}} \cdot \gamma_a \leq \text{fibers length} \end{cases}
\]

(9)
reduced dramatically by paying attention to zero entries in the matrix. By taking into account the known value of zero for most of the entries in the Jacobian, it is typically possible to approximate all the nonzero entries of several columns at a time. An important special case of a sparse Jacobian is one that has all its nonzero entries located in a band of diagonals [17, 18].

If the system of PDEs could be represented purely by banded Jacobians, then a banded integrator would be even more efficient than the sparse integrator since the banded integrator would know in advance where the nonzero elements occur (they all are in the band). It would therefore not have to search for the nonzero elements, and follow the resulting logic to use these nonzero elements, all of which add complexity to a sparse integrator. However, these nonzero elements are located/displayed with the Matlab command spy( ). For the PDE systems of Eqs. (10a)-(10f), some of the nonzero elements are located along the main diagonal as well as the others located outside the band. Therefore, for this case, a sparse integrator is particularly effective since it also operates on out-of-band elements, the so-called outliers, in performing the numerical integration. Thus, this combination of elements in a band along the main diagonal, plus outliers, leads to the efficiency of sparse matrix integration [11].

Since this particular application has a main diagonal, plus outliers, we have used the routine jpattern num() to produce a Jacobian map for the sparse matrix option of the ode solver. In this routine, the elements of the Jacobian matrix are computed by finite differences (FDs). This requires a base point, or base points, around which the numerical derivatives are computed. A base point can be any value within the range of the variation of the associated dependent variable. A precise value is not required (but if the ode solver fails, some experimentation with this value may be required). In our application we set the base points ybase(i), equal to the initial condition for each dependent variable which is sufficiently accurate to provide a good solution.

In order to calculate the partial derivatives in the Jacobian matrix by FD approximations, the derivatives at the base point, dyi/dt, are also required. Note that the routine pde() (see Appendix), that calculates the derivatives dyi/dt in Eqs. (10a)-(10f) is called by the Matlab command:

\[ \text{ybase} = \text{pde}(\text{tbase}, \text{ybase}) \]

The elements of the Jacobian matrix are evaluated numerically by the Matlab command:

\[ [\text{Jac}, \text{fac}] = \text{numjac} (@\text{pde}, \text{tbase}, \text{ybase}, \text{ybase}, \text{thresh}, \text{fac}, \text{vectorized}) \]

where the following argument values are used: tbase=0, thresh=1.10^-16 and fac=[]. The input variable thresh represents the threshold of significance for Y(i), such that Y(i) < thresh is not important; the input variable fac represents a working storage column vector and should not be changed between calls; and vectorized = 'off' or 'on' tells numjac() whether single or multiple values of Y respectively can be obtained with a single function evaluation.

Once the Jacobian matrix has been defined by this call to numjac(), a map of the Jacobian is plotted by the code that follows the call to numjac(). With a call to the routine spones(), each non-zero element is replaced by a “1” so as to create a “0-1” map of the Jacobian matrix [11, 16].

\[ S = \text{spones} (\text{sparse} (\text{Jac})) \]

In Figure 2, the sparsity pattern of Eqs. (10a)-(10f) is plotted by a subsequent call to the routine spy(S). Code for jpattern num() is available as a library routine which can be downloaded from Ref. [19].

Note that only 0.185% of the 6x300 x 6x300 elements of the Jacobian map of Figure 2 are nonzero. This is not uncommon; that is, most of the elements are zero. This condition is the reason for the use of sparse matrix integrators since they will in general process only the nonzero elements and will not expend computer time processing the many zero elements. The processing time with and without jpattern num() is illustrated in Table 1 for the implicit (stiff) ODE solvers (ode15s, ode23s, ode23t, and ode23tb). The computation is performed on a i5 Core, 4 GB RAM, Laptop computer running the Windows 7 operating system. From Table 1, it can be seen that the sav-

![Jacobian sparsity pattern – nonzeros 5981 (0.185%)](image)

**FIG. 2 Sparsity Pattern of the Eqs. (10a)-(10f).** The solution is obtained with \( n = 300 \).

<table>
<thead>
<tr>
<th>Run Time (sec)</th>
<th>Abs Tol and Rel Tol= 1 \times 10^{-5}, Pulse width=5 ns. Without Flux-limiter</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=300</td>
<td>Without jpattern num</td>
</tr>
<tr>
<td>ode15s</td>
<td>204</td>
</tr>
<tr>
<td>ode23s</td>
<td>242</td>
</tr>
<tr>
<td>ode23t</td>
<td>292</td>
</tr>
<tr>
<td>ode23tb</td>
<td>Computation failed</td>
</tr>
</tbody>
</table>

Table 1: Run-time comparison of implicit ODE solvers with/without use of jpattern num()
3.3 Use of a Flux Limiter

Numerical dissipation is generally observed when solving strongly convective or strongly hyperbolic PDEs. The main cause of the dissipation is low order approximation of the discretization. It should be noted that numerical dissipation produced by first order approximations is to be expected due to truncation of the Taylor series that is the basis for the approximations. In recent years, high resolution schemes have been developed to obviate this effect with a high degree of accuracy, albeit at the expense of algorithmic and computational complexity. Examples of particularly effective schemes are based upon flux/slope limiters [20] and WENO methods [21].

Flux limiters are used in numerical schemes to solve problems in science and engineering that are described by partial differential equations (PDEs). Their main purpose is to avoid the spurious oscillations (wiggles) that would otherwise occur. Use of flux limiters, together with an appropriate high-resolution scheme, makes the solutions total variation diminishing. Flux limiters are also referred to as slope limiters because they both have the same mathematical form and both have the effect of limiting the solution gradient near shocks or discontinuities. In general, the term flux limiter is used when the limiter acts on system fluxes, and slope limiter is used when the limiter acts on system states. The main idea behind the construction of flux limiter schemes is to limit the spatial derivatives to realistic values; this usually means physically realizable values. They are used in high-resolution schemes for solving problems described by PDEs and only come into operation when sharp wave fronts are present [22].

Considering the semidiscrete scheme below,

$$\frac{du_i}{dt} + \frac{1}{\Delta x_j} \left[ F(u_{i+\frac{1}{2}}) - F(u_{i-\frac{1}{2}}) \right] = 0 \quad (11a)$$

where, for a finite difference scheme, $F(u_{i+1/2})$ and $F(u_{i-1/2})$ represent flux values on the grid at point $x = x_i+1/2$ and point $x = x_i-1/2$. If these fluxes can be represented by low- and high resolution schemes, then a flux limiter can switch between these schemes depending upon the gradients close to the particular cell as follows:

$$F(u_{i+\frac{1}{2}}) = f_{i+\frac{1}{2}}^{\text{low}} - \phi(r_i)(f_{i+\frac{1}{2}}^{\text{low}} - f_{i+\frac{1}{2}}^{\text{high}}) \quad (11b)$$

$$F(u_{i-\frac{1}{2}}) = f_{i-\frac{1}{2}}^{\text{low}} - \phi(r_{i-1})(f_{i-\frac{1}{2}}^{\text{low}} - f_{i-\frac{1}{2}}^{\text{high}}) \quad (11c)$$

where, $f_{i}^{\text{low}}$ low-resolution flux, $f_{i}^{\text{high}}$ high-resolution flux, $\phi(r) = \text{flux limiter function,}$ and $r$ represents the ratio of successive gradients on the solution mesh, i.e.,

$$r_i = (u_{i+1} - u_{i-1}) / (u_{i+1} - u_{i}) \quad (11d)$$

The limiter function is constrained to be greater than or equal to zero, i.e., $r \geq 0$. Therefore, when the limiter is equal to zero (sharp gradient, opposite slopes, or zero gradient), the flux is represented by a low-resolution scheme. Similarly, when the limiter is equal to 1 (smooth solution), it is represented by a high-resolution scheme. The various limiters listed below have differing switching characteristics and are selected to suit the particular problem and numerical solution scheme. No particular limiter has been found to work well for all problems, and a particular choice is usually made on a trial-and-error basis [11, 22].

van Leer: $\phi_{\text{vl}} = \frac{r + |r|}{1 + r}; \lim_{r \to \infty} \phi_{\text{vl}}(r) = 2 \quad (12a)$

smart: $\phi_{\text{sm}}(r) = \max \{0, \min(2r, (0.25 + 0.75r), 4)\}; \lim_{r \to \infty} \phi_{\text{sm}}(r) = 4 \quad (12b)$

superbee: $\phi_{\text{sb}}(r) = \max \{0, \min(2r, 1, \min(r, 2))\}; \lim_{r \to \infty} \phi_{\text{sb}}(r) = 2 \quad (12c)$

The parameters of the flux limiter, which available as a library routine which can be downloaded from [23], are as follows,

$y_{1x} = \text{flux function}(xl,xu,n,y_{1},1)$; $y_{2x} = \text{flux function}(xl,xu,n,y_{2},1)$.

The fifth parameter of direction is the flow for the linear advection equation $u_t + cu_x = 0$. This argument is required, since the limiter requires the direction of flow or wave propagation. It takes the values $+1$ or $-1$ depending on the direction of the upwinding of the FD. The fifth parameter value of $-1$ for $y_{2x}$ indicates that the pulse moves right to left in $z$ at velocity $c = -1$ (i.e., counter propagating in $z$). The spatial domain in $z$ is defined as $xl \leq z \leq xu$ with $n$ points. The interval $xu - xl$ is determined by the fiber length.

It should be noted that when applying flux limiter solutions it is necessary to have sufficient grid points (say 10 or more) between the occurrence of adjacent steep gradients. If insufficient grid points are used, then smearing of the solution may occur or, in extreme cases, the solution could blow up.

In order to establish the best flux limiter function to suit our particular application we performed numerical experiments to compute the degree of smearing. In Figure 3, The effect of van Leer, Smart and Superbee limiters in terms of smearing can be seen.

In Figure 3(a), it is clearly seen that the Stokes magnitude is damped due to the numerical dissipation. In Figure 3(b)-(e), the effect of different flux limiters on the Stokes pulse is demonstrated. Damping which can be considered as the amplitude ratio at $z = 0$ and at $z = L$ is calculated as, (b) 62.2 % for van Leer, (c) 55.44 % for smart (d) 47.74% for superbee flux limiter, respectively (n=100). In Figure 3(e), the damping effect is decreased to only 2.8 % with superbee flux limiter for 300 discretization points. In Figure 3(b)-(e), it is clearly seen that, for this particular application, the superbee limiter is more capable of avoiding numerical damping compared to the performance of van Leer and Smart flux limiters. Since it has superior performance for this application, the superbee flux limiter is preferred.

4 EXAMPLE

The simulation models an optical fiber, 15 m in length, and the parameter values of the simulation are inferred from the pa-
The Brillouin frequency, $\delta\nu_{\text{Res}}(z)$, is 12.800 GHz along the fiber, except for a specified section in the center where its value is different from the main fibers’ Brillouin frequency, representing a strained section. The moment in time, when the Stokes pulse enters the fiber is assumed to be equal to $t_0$ with respect to the pulse leading edge. Bright pulse with a CW component is used as a Stokes waveform. The other relevant parameters are: CW power = 5 mW; Stokes power = 10 mW; Stokes pulse rise time = 100 ps. The simulation models a pulse on-resonance, with the strained section ($\nu_p - \nu_s$ is varied). The other Brillouin parameters typical for the single-mode silica fibers at the wavelength 1.3 $\mu$m: $\tau_{\text{ph}} = 10$ ns, $A_{\text{eff}} = 50$ $\mu$m$^2$, $g_B = 5 \times 10^{-11}$ m/W, $\nu = 0.2$ m/ns.

The expected result of this simulation is that the Stokes pulse will propagate down the section of fiber with no loss or dispersion. Since the fiber’s natural $\delta\nu_{\text{Res}}(z)$ is much different from $\nu_p - \nu_s$, there should be little interaction in the ‘unstrained’ portions of the fiber. Upon entering the ‘strained’ section, the pulse will interact with the CW field, causing a rise in the acoustic field intensity. The CW field will be depleted, and the pulse will be enhanced. The range of the depletion depends on the extinction ratio and the interaction is stronger for a low extinction ratio. After the Stokes pulse passes through the strained section, the acoustic field will decay away and the CW field will return to its steady state value. The effect of the interaction will take the form of a depleted region of CW light propagating to the end of the fiber [6].

Figure 4 shows the Stokes, CW, and acoustic fields, simulated with and without the superbee flux limiter. As can be seen in Figure 4(a), dissipation and damping is clearly observed. Over time, the Stokes pulse broadens, as well as decreases in intensity. However, the expectation from this simulation is that Stokes pulse will propagate down the section of fiber with no loss or dispersion. It must be noted that, in the ‘unstrained’ portions of the fiber $\Delta \neq 0$ i.e., $\nu_p - \nu_s \neq \delta\nu_{\text{Res}}(z)$, in the “strained” section of the fiber $\Delta = 0$, i.e., $\nu_p - \nu_s = \delta\nu_{\text{Res}}(z)$.

Plots (c) and (d) show the fields simulated with the superbee flux limiter. Especially in plot (c), with flux limiter, it can be clearly seen that damping is suppressed and the ratio of the amplitudes at $z = 0$ and at $z = L$ becomes 0.972 which means
that Stokes wave lost only 2.8% of its magnitude. Another expectation of the simulation is that, when the pulse enters the strained section there should be a rise in the acoustic field intensity. However, the superbee flux limiter prevents this rise during stabilization.

For the case of a 4-ns Stokes pulse with ER = 15 dB and 50 dB, in Figure 5, AC detection is plotted as a function of the time and frequency detuning $\Delta \Omega = \Omega - \Omega_B$. Here $t_0$ and fiber length is assumed to be 5 ns and 15m, respectively. As can be seen in Figure 5, AC detection shows different characteristics at the time $t < t_0$, before the probe pulse arrives in the fiber, and at $t > 2L/\nu + t_0 + t_s$ (159 ns), after it has left the fiber and the related pump depletion has propagated from $z = 0$ to the output $z = L$. The AC-detected Brillouin spectrum shows a Lorentzian line shape with the Brillouin natural width $\Delta \nu_B = \Gamma / \pi \approx 32$ MHz resulting from the pump-base SBS and this corresponds to no signal before pulse arrival ($t < t_0$) and transient relaxation again to zero signal through complicated Brillouin loss signal oscillations with even negative values when out-of-resonance ($t > 2L/\nu + t_0 + t_s$).

Now we address the SBS-based sensing for longitudinally in-homogeneous distribution of the Brillouin frequency along the fiber. First we consider AC-detected Brillouin spectra of the fiber consisting of two 10-m long sections with Brillouin frequencies 12.800 and 12.840 GHz for the case of a 4-ns Stokes pulse (Figure 6). We verify how the sections and especially the boundary between them are identified AC detection for two different base levels of 15 and 50 dB. The thick dashed curve is drawn at the time $t = 157$ ns, when the pulse peak center is passing through the middle of the fiber, and this point should be identified by sensing as a boundary between the sections. Additionally, at the moment in time corresponding to the boundary, the spectrum transition from one line to another is clearly visible. However, due to broad pulse spectrum this transition is complicated by the prolonged relaxation of the acoustic field at the frequency related to the previous section, as the pulse has already progressed to the next section.

Figure 7 illustrates the SBS-based sensing of a 20-cm section in the middle of a 10-m fiber, where the Brillouin frequency of the section is shifted by 80 MHz relative to that of the rest of the fiber. In Figure 7, we have used a 2-ns Stokes pulse with ER = 15 dB (Figure 7(a)) and 50 dB (Figure 7(b)), respectively. The other parameters are taken from Figure 5. Thick dashed curves are drawn for the output pump at times when the pulse peak center passes through the section’s center, and at the frequency shift corresponding to the Brillouin frequency of the section. They denote the time and the frequency, which should be revealed from the detection as a section’s spatial and spectral “location”.

However, it was observed that, for pulses substantially shorter than the acoustic lifetime in silica, the measured gain profiles abandon their expected spectral broadening and the linewidth was shown to gradually return to the natural value, as determined by the acoustic lifetime [15]. This was explained by the fact that in a real experiment the Stokes pulses have a small cw component (base) due to unavoidable leakage of any optical amplitude modulator used to generate nanosecond pulses. In this case the linewidth could still be close to the natural linewidth, provided the relative contribution of the transient regime on the output pump depletion is weak [7]. In our numerical experiment the same trend is observed in Figure 8.

5 CONCLUSIONS

In this paper we introduce, for the first time to the best of our knowledge, the solution of transient SBS equations for nanosecond Stokes pulses with different waveforms using Matlab ODE solvers (MATLAB, 2012). Bu using this solution the spectral and temporal evolution of the AC detection is
simulated. The MOL scheme provided the basis for a solution which was adapted to a suitable mesh size allowing rapid calculation of the fields involved in the SBS interaction. We introduce the solution technique and have adapted the software to provide a fast and stable solution by employing sparse matrix techniques and by using an appropriate flux limiter. The simulation method presented here can be used to foster a rapid understanding of the internal dynamics of the distributed sensing based on SBS, lending insight for both experimental design and interpretation of results.

6 ACKNOWLEDGEMENTS

The University of Hasan Kalyoncu Optical Fibre Systems Laboratory is supported in part by grants from the TUBITAK.

References


FIG. 8 FWHM versus Stokes pulse duration for AC detection at $t = 157$ ns, different extinction ratio and the same pump and Stokes pulse parameters as in Figure 5. Dashed line shows the natural Brillouin linewidth.


A APPENDIX

function ut=pde(t,u)
%
% Problem parameters

global xl xu n ncall gamma_k
global coeff1 coeff2 coeff3 coeff4 TT1 TT2 omega1 omega2
global dx1 dx2 g1 g2

i = 1:n; % spatial index
i1 = i(dx1<i & i<=dx2); % 'inner' region
i2 = i(i<dx1 | dx2<i); % 'outer' region
coeff = zeros(n,1);
coeff(i1) = coeff2;
coeff(i2) = coeff1;

coeffsin = zeros(n,1);
coeffsin(i1) = coeff4;
coeffsin(i2) = coeff3;

omega = zeros(n,1);
omega(i1) = omega2;
omega(i2) = omega1;

TT = zeros(n,1);
TT(i1) = TT2;
TT(i2) = TT1;

g = zeros(n,1);
g(i1) = g2;
g(i2) = g1;

% u to y, columnn vectors

u = reshape(u,n,6);
y1 = u(:,1); %As
y2 = u(:,2); %Ap
y3 = u(:,3); %Aa
y4 = u(:,4); %dAa/dt
y5 = u(:,5); %Phi_a
y6 = u(:,6); %dPhi_a/dt

% BCs
y1(1)=pulse(t);
y2(n)=5e-3;
% Derivatives in t
% d(y1)/dt
y1t = y1x - y2.*y3.*coeff;
y1t(1) = 0; % boundary condition

% d(y2)/dt
y2t = y2x + y1.*y3.*coeff;
y2t(n) = 0; % boundary condition

y3t(i)=y4; % y3t= dAa/dt
y5t(i)=y6; % y5t= dPhi_a/dt  y6t= d^2Phi_a/dt^2

y4t= (-g*y1*y2*coeff*sin-(2*gamma_k*y4+2*y3*omega*y6 )+... y3*y6*y6 -TT*y3 );
y6t= (-g*y1*y2*coeff*sin+2*omega*( y4+ gamma˙k*y3 )-... 2*y6*(y4+y3*gamma_k))/y3 ;

% yt to ut, column vector
ut = [y1t;y2t;y3t;y4t;y5t;y6t];

% Increment calls to pde'
call=ncall+1;