An Algorithm for Creation of an Optimized Adaptive Grid for Improved Explicit Finite Difference Scheme

RAKA JOVANOVIC
Institute of Physics
Belgrade
Pregrevica 118, Zemun
SERBIA
rakabog@yahoo.com

MILAN TUBA
Faculty of Mathematics
University of Belgrade
Studentski trg 16
SERBIA
tubamilan@ptt.rs

DANA SIMIAN
Department of Computer Science
Lucian Blaga University of Sibiu
5-7 dr. I. Ratiu str.
ROMANIA
d_simian@yahoo.com

Abstract: - This paper deals with the two main shortcomings of explicit finite difference schemes: the use of a discretization grid with the same resolution over the entire problem space, and low level of precision and stability. We present a combination of two improvements. Their application is illustrated with the numerical simulation of the propagation of a light beam in a photonic lattice. The discretization problem is avoided by using a multi-resolution grid. An algorithm for the grid creation is developed and that algorithm is optimized for software implementation and parallelization. The efficiency of the algorithm is increased by further improving the precision of the explicit method by use of a multidimensional generalization of the Runge-Kutta scheme. Due to the multidimensionality and nonlinearity of the considered problem, our improved explicit finite difference gave better results than Crank-Nicholson scheme.

Key-Words: - Adaptive grid algorithm, Multi-resolution, Finite differences, Simulation, Numerical optimization

1 Introduction

Partial differential equations (PDE) are used a variety of different fields of research like physics [1,2], engineering [3], image processing [4], etc. There are methods for solving PDE analytically like Bäcklund transformation, characteristics, Green's function, integral transform, Lax pair, separation of variables [5], but in general, partial differential equations are difficult to solve analytically and in some cases even impossible. Due to the widespread of use of PDEs, a significant number of numerical methods has been developed for solving them. Most standard used methods belong to one of the following groups: finite difference (FD) [6, 7, 8], finite element (FEM) [9,10], using Fast Fourier Transform (FFT) [11], Monte Carlo methods [11], Lagrangian methods, and Wavelets [12]. The choice of which method will be used is very important because not all methods are adequate for some problems. Computers are an essential part of using these methods and because of this, their implementation as computer programs should be viewed as a separate problem.

FD method for the solution of partial differential equations is commonly used for a wide range of problems in physics and engineering, due to the simplicity of its implementation and parallelization on multiprocessor machines [13, 14]. One of the biggest drawbacks of the standard FD scheme is that it uses a discretization grid with the same resolution over the entire problem space. Usually, a high resolution is required only in a small fraction of the problem space, and the use of an uniform grid unnecessarily inflates the demand for computer resources and increases the time needed for calculations. One of the possible ways to solve this problem is to combine the finite difference with the finite element methods [15, 16]. However, in such case one loses the main advantage of the FD, which is its simplicity of implementation.

This paper presents a simple algorithm for creation of a multi-resolution grid, and its use with an explicit FD method. It is based on the improvement of the usual explicit FD, by using the analog of an one dimensional (1-D) Runge-Kutta procedure which greatly increases the precision. The second improvement is the creation of a multi-resolution grid, which separates the calculation of the optimum grid resolution for the numerical calculations, from the grid corrections for the use in a particular program. As an example, the application of our algorithm is presented for the numerical simulation of the propagation of a light beam in a photonic lattice. This problem was usually treated by the use of the FFT [17, 18]. However, a significant drawback of the FFT is that it requires a grid of the $2^k \times 2^k$ size [19], which needs to be sufficiently large to hold all possible 'interesting'

areas. The method represented in this article does not have such a strict rule for the grid size, which can be of any dimension $2^i m \times 2^i n$. While the use of FFT is more efficient in the typical cases, for certain particular cases an FD with adaptive grid can give better results.

The idea of using FD with an adaptive grid is not new [20]. However, in earlier works, the creation of a multi-resolution grid was mostly dedicated to the numerical part of the problem, and not to the implementation as it is done in this paper. The proposed grid correction is aimed for the implementation, and it makes the program implementation and parallelization greatly simpler.

The paper is organized as follows. In Section 2 the basic physical problem is described, and the corresponding analytical equations are presented, which belong to the class of nonlinear Schrödinger (NLS) equations. In Section 3 we apply the FD method to this equation, with a forward difference to get an explicit scheme. In the second part of Section 3, an improvement similar to the one dimensional Runge-Kutta is presented. Section 4 shows the multi-resolution grid creation and its optimization for the use in computer programs. In the second part of Section 4 we present the analysis of some problems, with the use of this kind of grid, and introduce the necessary conversions of the basic equations from Section 2. In the first part of Section 5 we analyze and compare the results of simulations implemented by the FD method, Crank-Nicholson method, FD with a simple predictor method improvement and FD with Runge-Kutta improvement. In the second part of this section we observe the affects of using the adaptive grid combined with FD with Runge-Kutta improvement.

2 Physical Model

It is well known that in linear optical media, the light beams have a tendency to spread as they propagate, due to the diffraction and the dispersion of incoherent light. Conversely, in a carefully designed nonlinear media and under certain conditions, the light waves may propagate without spreading or scattering. Instead, they keep their shape and intensity constant. These dynamically and structurally stable objects are called optical solitons [14, 21]. The stability of these objects results from the interplay of the dispersion and diffraction with the nonlinear effects, which tend to localize the wave. Stable objects, also known as solitons, emerge when effects of diffraction are completely compensated by nonlinear effects. Solitons can be viewed as waves that are restricted to the specific interval of time and region of space.

Photorefractive media are those in which the photorefractive effect takes place, i.e. those whose refractive index is altered in the transverse region that is occupied by the light beam. The interaction of a laser beam with a photorefractive crystal can be described by the paraxial wave equation. We can inspect the beams with copropagating (CO) geometries and with different input beam shapes (Gaussian, dipole, quadruple and vortex).

Optically induced photonic lattices are the realization of the photonic crystal concept [22]. Photonic crystals are the materials that posses a periodic structure in space, which enables them to control the propagation of light. Photonic crystals can be viewed as an optical analogue of semiconductors, in the sense that they modify the propagation characteristics of light just as an atomic lattice modifies the properties of electrons through a bandgap structure. If, for a certain frequency range, a photonic crystal reflects light of any polarization, incident at any angle, we say that the crystal has a complete photonic band gap (PBG). This can be regarded as the analogue of the bandgap structure in semiconductors.

2.1 Mathematical Model

The behavior of CO beams in photonic lattices is described by a time-independent model for the formation of self-trapped CO optical beams, based on the theory of photorefractive (PR) effect. The mathematical model consists of one wave equation in the paraxial approximation for the propagation of CO beams. The model equation has the standard form of a NLS equation with a nonlinearity that is a rational function of the beam intensity, and in the computational space it has the form [23]:

$$i\partial_z F = -\Delta F - \Gamma F \frac{|F|^2 + I_g}{1 + |F|^2 + I_g} \qquad (1)$$

where F is the forward propagating beam envelope, Δ is the transverse Laplacian, and Γ is the dimensionless coupling constant. The quantity $|F|$ is the laser light intensity, and it is constant over all iterations. The above dimensionless propagation equation is written under the scaling $x/x_0 \longleftarrow\!\!\!\shortmid x, y/x_0 \longleftarrow\!\!\!\shortmid y, z/L_D \longleftarrow\!\!\!\shortmid z$, where x_0 is the typical Full width at half maximum (FWHM) beam waist and L_D is the diffraction length. I_g is the transverse intensity distribution of the optically

induced lattice array, formed by positioning Gaussian beams at the sites of the lattice, which is a known function in Equation 1. Different geometries of the lattice can be considered, such as hexagonal, cylindrical and square.

3 Application of Finite Differences

For an explicit finite difference method, applied to the 3D function *F*, defined on a grid $x_i = x_0 + i*h$, $y_i = y_0 + i*h$, and $z_i = z_0 + i*dz$, we use the following approximations:

$$F_{ijk} = F(x_i, y_j, z_k)$$

$$\partial_x F(x_i, y_j, z_k) = \frac{F_{i+1jk} - F_{ijk}}{h} = F^x_{ijk}, O(h)$$

$$\partial_y F(x_i, y_j, z_k) = \frac{F_{ij+1k} - F_{ijk}}{h} = F^y_{ijk}, O(h)$$

$$\begin{aligned}\partial^2_{x^2} F(x_i, y_j, z_k) &= \frac{F^x_{i+1jk} - F^x_{ijk}}{h} \\ &= \frac{F_{i+1jk} - 2F_{ijk} + F_{i-1jk}}{h^2} = F^{xx}_{ijk}, O(h^2)\end{aligned} \quad (2)$$

$$\begin{aligned}\partial^2_{y^2} F(x_i, y_j, z_k) &= \frac{F^y_{ij+1k} - F^y_{ijk}}{h} \\ &= \frac{F_{ij+1k} - 2F_{ijk} + F_{ij-1k}}{h^2} = F^{yy}_{ijk}, O(h^2)\end{aligned}$$

$$\begin{aligned}\Delta F(x_i, y_j, z_k) &= \partial^2_{x^2} F(x_i, y_j, z_k) + \partial^2_{y^2} F(x_i, y_j, z_k) \\ &= F^{xx}_{ijk} + F^{yy}_{ijk}, O(h^2)\end{aligned} \quad 3.1$$

$$F(z_0, x, y) = F_0 \quad (3)$$

In an explicit difference method, we proceed by using the approximations defined by Equation 2, which after the substitution in Equation 1 yield

$$\begin{aligned} &i = 1, nxy \\ &j = 1, nxy \end{aligned} \quad (4)$$

$$\begin{aligned}F_{n+1}((x_i, y_j) &= F_n(x_i, y_j) + \\ &idz\left(\frac{F_{i+1j} + F_{ij+1} - 4F_{ij} + F_{ij-1} + F_{i-1j}}{h^2}\right. \\ &\left. + \frac{|F_n(x_i, y_j)|^2 + I_{gij}}{1 + |F_n(x_i, y_j)|^2 + I_{gij}}\right)\end{aligned} \quad (5)$$

We use this approximated system for iterating our system over the variable *z*. The use of Equations 4 and 5 gave poor results due to the lack of precision, the accuracy over *z* is proportional to dz. The second problem was low stability of the method that mainly appears because of the use of an explicit FD scheme.

The first approach to solving this problem was using an implicit FD scheme like Crank-Nicholson.

$$\begin{aligned}\partial_z F(x, y, z) &= -i\left(-\Delta F - \Gamma F \frac{|F|^2 + I_g}{1 + |F|^2 + I_g}\right) \\ &= G(x, y, z)\end{aligned} \quad (6)$$

$$\frac{F_{ijk+1} - F_{ijk}}{dz} = \frac{1}{2}\left(G_{ijk+1} + G_{ijk+1}\right) \quad (7)$$

The usual way of using Crank-Nicholson is creating the a set of algebraic equations and solving that system. When solving one dimensional partial differential equations the matrix of the system is three-diagonal and it is simple to solve. In a two dimensional case we get a five-diagonal system. Solving this system is much more time consuming, and with lower precision. This problem is avoided by using the alternative direction (ADI) method, that solves both of these problems. In our case this did not give good results , due to the existence of the nonlinear term $|F|^2$. The method was stable but the precision much less than the expected second level.

4 Finite Differences Improvements

Using the Euler integration for eq. (1), with the step *dz*, and calculating the nonlinear term $|F|^2$ at z_n, we obtain

$$\begin{aligned}F_{n+1} &= F(z_{n+1}, x, y) \\ &= F_n + idz\left(\Delta F_n + \frac{|F_n|^2 + I_g}{1 + |F_n|^2 + I_g}\right).\end{aligned} \quad (8)$$

These are the same equations that one would obtain by using the forward difference over *z* eq.(5). When we perform the Euler integration over *z*, the *z* component of the error is proportional to *dz*. In a one dimensional case this part of the error can be reduced by using the a simple Predictor method to dz^2. If we use an analogue to this method we get the following equations:

$$Diff(F,K) = K \times i \times dz \left(\frac{F_{i+1j} + F_{ij+1} - 4F_{ij} + F_{ij-1} + F_{i-1j}}{h^2} + \frac{|F_n(x_i,y_j)|^2 + I_{gij}}{1+|F_n(x_i,y_j)|^2 + I_{gij}}\right) \quad (9)$$

$$S_1 = Diff(F_n, 1) \quad (10)$$

$$S_2 = Diff(F_n + \frac{S_1}{2}, \frac{1}{2}) \quad (11)$$

$$F_{n+1} = F_n + \frac{1}{2}(S_1 + S_2) \quad (12)$$

This part of the error can be reduced to dz^4 if we use the following analogue of the Runge-Kutta method, viz.

$$S_1 = Diff(F_n, 1) \quad (13)$$

$$S_2 = Diff(F_n + \frac{S_1}{2}, \frac{1}{2}) \quad (14)$$

$$S_3 = Diff(F_n + \frac{S_2}{2}, \frac{1}{2}) \quad (15)$$

$$S_4 = Diff(F_n + S_3, 1) \quad (16)$$

$$F_{n+1} = F_n + \frac{1}{6}(S_1 + 2S_2 + 2S_3 + S_4) \quad (17)$$

The integration of the function *F* over *z* can be done either in the implicit or in the explicit way. We adopt an explicit procedure, because implicit methods, such as the alternative-direction, give a low level of accuracy in the case of multidimensional problems, due to the existence of the nonlinear term $|F|^2$ [24]. The problem of stability of explicit methods is not easy to solve, but the use of an adaptive step for the Runge-Kutta procedure usually gives satisfactory results.

4 Adaptive Grid Implementation

As we have mentioned before, one of the biggest problems with the FD method is that the grid has the same resolution over the entire problem space. However, in the simulation of CO beam propagation, the most of the beam intensity is usually located in a small part of the problem space, as it can be seen in Fig. 1.

We would like to calculate the function *F* more precisely in the high intensity areas, where the changes between neighboring grid points are much bigger than in the low intensity areas. In the latter case the changes are often so small that they can even be neglected. In the application of an adaptive grid to the problem defined by Equation 1, there are two main problems. First, it is necessary to create a grid that has a high resolution at the correct places, and which allows the easy translation of indices from the high to low resolution areas and vice versa, that is required in the program implementation. Second, one needs to implement the necessary corrections to Equation 9 for the grid blocks that have neighbors with different resolutions.

4.1 Grid Creation

When creating a multi-resolution grid we first have to define criteria for calculating block sizes at different positions in the grid. To achieve this we must define a function that will give an estimate of the expected resolution of the grid at each point. This is best done with an integer value that corresponds to some scale. To do this, let us define the function *Mass, Scale*:

$$Scale: \mathbf{R} \longrightarrow \{1,..,ScaleMax\}$$
$$Mass: \{1,n\} \times \{1,m\} \longrightarrow Scale \quad (18)$$

$$Mass_{ij} = Mass(i,j) = Scale(|F_{ij} - F_{i+1j}| + |F_{ij} - F_{i-1j}| + |F_{ij} - F_{ij+1}|) + |F_{ij} - F_{ij-1}|) \quad (19)$$

Here *Mass*, represented as a matrix generated for some function *F*, has the form

$$Mass = \begin{pmatrix} 3&3&3&3&2&3&3&1 \\ 3&3&3&3&2&2&3&3 \\ 3&3&3&3&3&2&2&3 \\ 3&3&3&3&3&2&2&3 \\ 3&3&3&3&1&2&2&3 \\ 3&3&3&3&2&2&1&3 \\ 3&3&3&3&2&2&1&3 \\ 3&3&3&3&3&3&3&3 \end{pmatrix} \rightarrow \begin{pmatrix} 3&3&3&3&2&2&1&1 \\ 3&3&3&3&2&2&1&1 \\ 3&3&3&3&2&2&2&2 \\ 3&3&3&3&2&2&2&2 \\ 2&2&2&2&1&1&1&1 \\ 2&2&2&2&1&1&1&1 \\ 2&2&2&2&2&2&1&1 \\ 2&2&2&2&2&2&1&1 \end{pmatrix} = \tilde{Mass} \quad (20)$$

In this representation of *Mass*, larger values represent the points in which larger grid blocks could be used, and smaller are used for the opposite. With some corrections *Mass*, could be used to define a grid with different resolutions that can be efficiently used in computer programs. For this to be possible, the corrected $\tilde{Mass}$ needs to have the following characteristics:

- grid blocks need to be squares of the size $2^k \times 2^k$,
- for all the blocks of the level k, the values of the indexes i, j corresponding to the top left corner of the block in the highest resolution are such that $i, j = 0 (mod) 2^k$,
- for every i, j, $Mass_{ij} >= Mass_{ij}$
- the level difference between neighboring blocks cannot be bigger than 1.

We wish to point out that $Mass$ could be used for multi-resolution grid definition even if not all mentioned criteria where followed. In that case, implementation would be significantly more difficult. Equation 20 is the example of a matrix that has been corrected by using these rules. The creation of $Mass$ can be explained the best by the following pseudo code

```
for Level = 1,ScaleMax
  begin
    Step = 2**Level
     for ix = 0,n :Step
       for iy = 0,m :Step
        CalcMassBlock(ix,iy, Level)
   end
```

We pass through each level, starting from the lowest, and for each block of the size $2^{CurrentLevel} \times 2^{CurrentLevel}$ we call the function *CalcMassBlock*. *CalcMassBlock (GridStartX, GridStartY ,CurrentLevet)* is a function that corrects a matrix block of the size $2^{CurrentLevel} \times 2^{CurrentLevel}$ and the neighboring blocks. It has the following pseudo code

```
function CalcMassBlock
         (GridStartX,GridStartY,
          CurrentLevel)

ActiveBlock = block that starts at
GridStartX, GridStartY of size
2**CurrentLevel X 2**CurrentLevel

if((Mass(A) = CurrentLevel))
 begin

       Fill all grid points in
       ActiveBlock with

       min(CurrentLevel,
           CurrentGridPointValue)

       Fill all grid points in
       blocks neighboring
       ActiveBlock of
            level(CurrentLevel+1)
       with
       min(Level+1,
       ,CurrentGridPointValue)

 end
```

The creation of this grid is illustrated below by creating a grid that was used for simulation of light propagation defined by Equation 1. We show a case where the input light ray was of the quadruple type. The input ray is displayed in Fig. 1

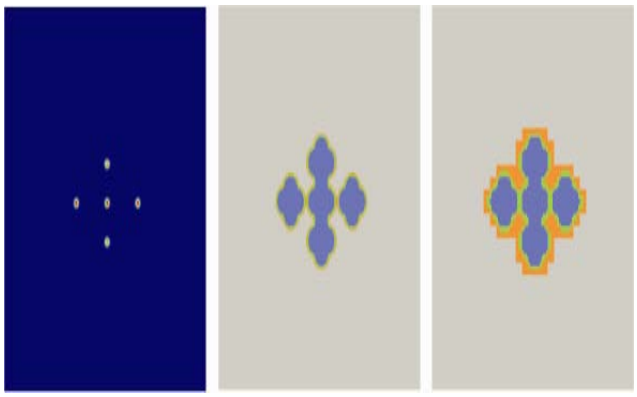

Fig. 1. Grid creation Example

The quadruple with central beam , we used was created position 5 Gaussians at appropriate positions. Each of them was calculated using Eq.21

$$F(x, y) = Ae^{-\frac{x^2+y^2}{2\sigma^2}} \quad (21)$$

The grid was defined by XYMax = 100, NXY = 256. The left image represents intensity distribution. The center image is the initial mass distribution generated using the scale defined by Table 19. The right image is the corrected mass distribution. The color scale used for mass distribution is blue for the value 1 (corresponding to the block size 1X1), green for 2(corresponding to block size 2X2), orange for 3(corresponding to block size 4X4), and grey for 4 (corresponding to block size 8X8).

4.2 FDTD Equations for Adaptive Grid

With an adaptive grid the same analogue of the Runge-Kutta method will be used to integrate over the variable z. To do this, we must modify Equation 9 in such a way that it can be used with the new grid. There are several parameters that need to be

analyzed for each block of level k. Let h be the grid spacing of the highest resolution. Then h^2 changes to $(2^{k-1}h)^2$. I_g is a precalculated value that represents the transverse intensity distribution of the optically induced lattice array. This parameter should be precalculated with blocks of sizes $(2^{k-1}h)$, in all necessary resolutions and for all possible levels. Let us denote each of these grids with Ig^k. F_{ij} will stay the same, but instead of F_{i-1j}, F_{i+1j}, F_{ij-1} and F_{ij+1}, we will use the representatives for right, left, upper or lower grid block of the size 2^{k-1}. In this problem, an arithmetic average of block members is used as the representative of a block, but for different problems solved by the same method, one may adopt a different procedure to calculate the representatives. Now Equation 9 takes the form

$$DiffR(F,K) = K \times i \times dz \left(\frac{F_{right} + F_{down} - 4F_{ij} + F_{up} + F_{left}}{(2^{Level-1}h)^2} \frac{|F_n(x_i,y_j)|^2 + I_g^{Level}}{1+|F_n(x_i,y_j)|^2 + I_g^{Level}}\right) \quad (21)$$

The original matrix that represents F is not equal at all grid points within a block of size $2^k \times 2^k$ defined in $Mass$, which is required for the use of Equation 21. A simple approach in creating $\bar{F}$ would be to use the arithmetic average of the corresponding values in F. However, this is not correct, because

$$|F|^2 = \int_{x,y} |F(x,y)|^2 \, dxdy \neq \sum_i \sum_j |F_{ij}|^2 \quad (22)$$

is constant in Equation 1. Because of this, we calculate the new value for a block of level k, B in $\bar{F}$ as

$$\bar{F}_B = \sqrt{\frac{\sum_{(i,j)\in B} |F_{ij}|^2}{2^{k-1}}} \quad (23)$$

4.3 Software Implementation

When creating a software implementation of simulation of the propagation of light beams in photonic lattices, the problem defined by Equation 1, the choice of input parameters is important. We have chosen to use as physical input parameters Γ; *xymax*, and *nxy* from which h is calculated; *nz* and *zmax* from which we calculate *dz*, type of lattice from which I_g is precalculated in the required resolutions, and the type of the input ray from which we calculate initial F. A different type of input parameter is *Scale* that defines the creation of *Mass*, which is a table that represents the mass value depending on the sum of differences

Scale	$5*10^{-3}$	10^{-4}	10^{-6}	10^{-7}
Mass	1	2	3	4

(24)

During the simulation tests, the use of Equation 19 for the calculation of *Mass* gave poor results, and it was replaced by

$$DistNorm(i,j,d) = |F_{ij} - F_{i+dj}| + |F_{ij} - F_{i-dj}| + |F_{ij} - F_{ij+d}|) + |F_{ij} - F_{ij-d}| \quad (25)$$

$$Mass_{ij} = Mass(i,j) = Scale(DistNorm(i,j,1) + \frac{1}{2}DistNorm(i,j,2) + \frac{1}{4}DistNorm(i,j,3) + \frac{1}{8}DistNorm(i,j,4)) \quad (26)$$

Parameter *Scale* proved to be of great importance for the simulation calculations, and to obtain the best results with respect to the relation between the precision and the performance, it was necessary to perform a series of tests, for each type of input parameters. When using the adaptive grid, *Mass* and $\bar{F}$ should be calculated once in the beginning of the simulation, and again every N (depends of the problem) iterations to reflect the changes on F. In each iteration calculations are done just for the top left point of each block using the Runge-Kutta analog with difference defined by Equation 21, and the *Mass* grid is used to jump to the next point (representing a block) that needs calculation.

The software implementation for the iterative process defined for the problem modeled by Equation 1 handling of the border of the problem space must also be solved in detail. The physical problem that is modeled, light beam propagation, has the property that at low intensity points

nonlinear effects are very weak. In the experiments we conducted of beam propagation in photonic lattices the beam mostly stays localized, because of this light intensity near the problem border is very low. When there are not any nonlinear effects light tends to spread, this gives justification to model the behavior at the border of problem space by extending the first derivative. In practical we wish that the model at the border has the same behavior as near the border. To attain this goal, and also keep the simplicity of implementation, the following adaptation must be applied to the basic algorithm. First grid points at the border cannot be calculated by Equation 17 due to non existing points. These points will be approximated in a way that preserves the first derivative. For a non adaptive grid this is achieved by Equations 27, 28, 29, 30 and for an adaptive grid changes similar to ones presented in Section 4.2. are needed.

$$F_{0j} = F_{1j} + (F_{1j} - F_{2j}) = 2F_{1j} - F_{2j} \quad (27)$$

$$F_{(nxy)j} = F_{(nxy-1)j} + (F_{(nxy-1)j} - F_{(nxy-2)j}) = 2F_{(nxy-1)j} - F_{(nxy-2)j} \quad (28)$$

$$F_{i0} = F_{i1} + (F_{i1} - F_{i2}) = 2F_{i1} - F_{i2} \quad (29)$$

$$F_{i(nxy)} = F_{i(nxy-1)} + (F_{i(nxy-1)} - F_{i(nxy-2)}) = 2F_{i(nxy-1)} - F_{i(nxy-2)} \quad (30)$$

5 Simulation results

In the first part of this section we analyze and compare the speed, precision and stability of simple FD scheme, the Crank-Nicholson method with ADI, Predictor improvement of FD, and Runge-Kutta improvement of FD. In the second part we compare results of Runge-Kutta improvement of FD with the same improvement with the use of an adaptive grid created by the method explained in Section 4. This will be done for grids of different sizes.

All tests have been performed on an Inter(R) Core(TM)2 6400 at 2.13 GHz with 4GB of memory. Simulations of beam propagation have been done for different propagation lengths *zmax*, and for different numbers of iterations *zn*. We used a photonic lattice grid defined by the following equation

$$GF = \left(\cos\frac{\pi(x+y)}{d}\right)^2 \left(\cos\frac{\pi(x+y)}{d}\right)^2 \quad (31)$$

$$I_g(x,y) = I_o GF$$

In our comparison of methods represented, we adopt as criteria of precision of given solution after N iterations ($\hat{F}$) the preservation of the invariant of the system represented by Equation 22 as an absolute value. The final evaluation of the quality of the calculated solution is comparing it to a solution F created by the FFT method . We do this in two ways. First, we observe the general behavior through successive iterations and we use the following norm

$$|F - \hat{F}| = \iint |F(x,y) - \hat{F}(x,y)|\, dxdy \quad (32)$$

to compare the FD approximated function and a FFT approximation at the final iteration. We also use as an aid for explaining some effects of these methods the behavior of these values between iterations and after all iterations have been completed.

In the simulations tested in Tables 1-4 we used a Gaussian type input ray and with the invariant $|F|$ = 14.12534533.

Method	Max Stable Iterations	Invariant at MSI	Precision between iterations	time
SFD	11	17.84628	0.9928	3.59
C-N	240	173.9651	0.9880	3.76
PIFD	9	15.7	0.9994	4.78
RKIFD	37	17.71	0.9973	13.84

Table 1. XYMAX = 30, NXY= 256, ZMax=5, NZ =240

From Table 1 we see that the simple FD and FD with the predictor method improvement are very unstable after a low number of iterations. The Crank-Nicholson method is absolutely stable for the number of observed iterations. The behavior of the approximation with this method is very similar to the one of the correct solution, but the preservation of the invariant of the Equation 1 is not adequate because this value has been increased 12 times. FD with the Runge-Kutta improvement stays stable for a much larger number of iterations that the two other FD methods, but it does not reach the needed number of iterations. During the stable period it has correct behavior and precision that is in needed boundaries.

Method	Max Stable Iterations	Invariant at MSI	Precision between iterations	time
SFD	32	15.1290284	0.9992	11.87
C-N	720	27.0903	0.9990	10.83
PIFD	103	14.8712844	0.9989	12.17
RKIFD	720	14.1276899	0.9999	23.75

Table 2. XYMAX = 30, NXY= 256, ZMax=5, NZ =720

Table 2 represents the same simulation with a greater number of iterations, or in other words, a smaller step *dz*. In this case FD with the predictor method improvement gives us a better result, both in precision and in stability than simple FD, but the stability still lasts only for a few iterations. In this case the Runge-Kutta improvement is stable in the whole area of interest, and gives best results of all methods.

It should be noted that the predictor improvement now gives better results that Crank-Nicholson. These results are better in both cases, if viewed only for successive iterations and at the end of the stable period. This is interesting because both approximations have an error of dz^2 in a linear equation. This points out the problems of using Crank-Nicholson in a (2+1) dimensional problem with a non-linearity. When implementing Crank-Nicholson for a more then (1+1) dimensional problem we use the ADI approach for getting a three diagonal matrix at each iteration for getting the solution. Combining it with a non linearity gives us very poor precision. One of the problems that appeared in the use of this method was that the error that appears between two successive iterations is similar to the one of the predictor improvement, but it accumulates in one direction and becomes great after a large number of iterations. This could probably be avoided by some correction to Crank-Nicholson implementation, but that is not the subject of this paper.

In Table 3 we use the same *dz* but the grid size is changed from 256 to 512. The larger grid greatly increases the instability of the FD methods. Due to this instability in some cases the time needed for calculations has been greatly increased since a large number of iterations has been done with numbers out of usual computer bounds.

Method	Max Stable Iterations	Invariant at MSI	Precision between iterations	time
SFD	14	17.84628	0.9854	257.62
C-N	720	27.072	0.99890	93.65
PIFD	10	14.163749	0.9989	845.06
RKIFD	11	14.1633052	0.9983	1853.68

Table 3. XYMAX = 30, NXY= 512, ZMax=5, NZ =720

In Table 4 we have taken a smaller value for *dz* to get a sufficient level of stability for Runge-Kutta improvement. The precision is higher for a grid with higher resolution as expected. In a number of test we conducted it was shown that the stability of the method depends on the value of *dz/h*, in these case the value was 0.04.

Method	Max Stable Iterations	Invariant at MSI	Precision between iterations	time
SFD	35	14.61782	0.99990	889.42
C-N	2000	17.519283	0.99987	257.57
PIFD	57	14.541818	0.99270	292.87
RKIFD	2000	14.125345	1	501.48

Table 4. XYMAX = 30, NXY=512, ZMax=5, NZ =2000

We use Tables 5 and 6 to analyze the effects of using an adaptive grid combined with the Runge-Kutta improvement of FD. Table 5 represents a simulation where approximately 10% of the whole problem space is in high resolution , and Table 6 has approximately 20% in high resolution. In these two tables we compare the results for different resolutions. We use the same of number of iterations (*N*) and value for *dz* in all simulations. In the simulations we analyze one loop cycle presented in section 4.3. We can observe that the increase in speed is around 50% in the lower resolution and up to almost 70% in the high resolution case with little loss in precision. We can also notice that the precision in the higher resolution is worse than in

the lower one. The reason for this is that we used the same *Scale* table in both cases. We did this to illustrate the importance of this parameter, when using a different *Scale* the higher resolution gives better results as expected.

Method	**Resolut.**	**Invariant**	**time**
RK	256	14.1253449	4.21
RKAG	256	14.12534413	2.07 + 0.02
RK	512	14.12534413	28.53
RKAG	512	14.1250873	10.60 + 0.14

Table 5. Gaussian input ray, invariant value 14.125345293. Approximately 10% of the adaptive grid is in the highest resolution.
XYMAX = 50, NXY= 256, ZMax=1, NZ =120

Method	**Resolut.**	**Invariant**	**time**
RK	256	71.8911501	5.87
RKAG	256	71.8911938	2.45 +0.03
RK	512	71.8911912	33.85
RKAG	512	71.8908379	12.89 + 0.19

Table 6. Quadruple input ray, invariant value 71.8911965. Approximately 20% of the adaptive grid is in the highest resolution**.**
XYMAX = 50, NXY= 256, ZMax=1, NZ =120

It should be understood that for solving Equation 1 and similar equations, a higher level of precision and stability could be achieved by a greater level of analytical analysis, and using that information for adding some additional terms or even using a different iterative method like the Petviashvili's method [25, 26]. We did not do this because we wanted to show the advantages of the Runge-Kutta improvement of FD, over Crank-Nicholson method in direct application to equations.

6 Conclusion

In this paper an algorithm is shown for the creation and implementation of an adaptive grid, as well as the implementation of the FD method with such algorithm. The basic steps of its implementation to (2+1) dimensional problems are illustrated by the application of FD with an adaptive grid to the simulation of the propagation of light beams in photonic lattices. The first step is the conversion of the starting equation to the approximation of derivatives with finite differences. It is shown that the integration over z can be performed much more efficiently by using an analogue of Runge-Kutta, instead of directly using the finite differences. In the conducted numerical experiments, we have shown that in the case of our simulation this improvement gives better results than the standard Crank-Nicholson scheme. The necessary calculation time for the FD can be considerably reduced by using an adaptive grid. In our simulations, this time was reduced more than 50%, depending on the input parameters. An algorithm for the creation of multi-resolution grids has been presented. It is divided in two parts, i.e. the creation of a mass grid that represents the level of detail that is needed at different grid points, and the conversion of the mass grid to one that could be used in computer programs. This approach is very convenient, because for the creation of grids for different problems, only the first part of the algorithm needs to be modified according to the characteristics of the problem. The use in a computer program of a grid created this way is simple and allows an easy parallelization on multiprocessor machines. A similar approach could be used also for (3+1) dimensional problems, with an even greater level of reduction of the calculation time, on which further research will be done.

References:

[1] Nikolaevich Andreĭ, Tikhonov, Aleksandr, Andreevich Samarskiĭ, *Partial differential equations of mathematical physics*, Holden-Day, 1964

[2] Raimonds Vilums, Andris Buikis, Conservative Averaging and Finite Difference Methods for Transient Heat Conduction in 3D Fuse, *WSEAS Transactions on Heat and Mass Transfer*, Vol 3, No. 1, 2008

[3] Kenneth S. Miller, *Partial Differential Equations in Engineering Problems*, Pearson Education Canada, 2000

[4] Xue-Cheng Tai, Knut-Andreas Lie, Tony F. Chan, Stanley Osher, *Image Processing Based on Partial Differential Equations*, Springer, 2007

[5] S. V. Meleshko, *Methods for Constructing Exact Solutions of Partial Differential Equations*, Springer, 2005

[6] Trefethen L. N., *Finite Difference and Spectral Methods for Ordinary and Partial Differential Equations*, Cornell University, 1996.

[7] Damelys Zabala, Aura L. Lopez De Ramos, Effect of the Finite Difference Solution Scheme in a Free Boundary Convective Mass Transfer Model, *WSEAS Transactions on Mathematics*, Vol. 6, No. 6, 2007, pp. 693-701

[8] Mastorakis N E., An Extended Crank-Nicholson Method and its Applications in the Solution of Partial Differential Equations: 1-D and 3-D Conduction Equations, *WSEAS Transactions on Mathematics*, Vol. 6, No. 1, 2007, pp 215-225

[9] Darrell W. Pepper, Juan C. Heinrich, *The Finite Element Method: Basic Concepts and Applications*, Taylor & Francis, 1992

[10] Nikos E. Mastorakis, Numerical Solution of Non-Linear Ordinary Differential Equations via Collocation Method (Finite Elements) and Genetic Algorithm", *WSEAS Transactions on Information Science and Applications*, Vol. 2, No. 5, 2005, pp. 467-473

[11] Robert Vichnevetsky, Robert S. Stepleman, *Advances in Computer Methods for Partial Differential Equations*, IMACS, 1981

[12] Robert John Benedetto, Michael Frazier, *Wavelets: Mathematics and Applications*, CRC Press, 1993

[13] Zhao J., Davison M., Corless R. M., Compact finite difference method for American option pricing, *Journal of Computational and Applied Mathematics*, Vol. 206 , No. 1, 2007, pp. 306-321

[14] Sun X.H, Joslin R.D., Massively Parallel Algorithm for Compact Finite Difference Schemes, *Parallel Processing*, Vol 3, No 1, 1994, pp. 282 – 289

[15] Wang B. L, Tian Z. H, Application of finite element–finite difference method to the determination of transient temperature field in functionally graded materials, *Finite elements in analysis and design*, Vol. 41, No 4, 2005, pp. 335-349

[16] Xing J. T., Price W. G., Chen Y. G., A Mixed Finite-Element Finite-Difference Method for Nonlinear Fluid-Structure Interaction Dynamics, *Fluid-Rigid Structure Interaction, Proceedings: Mathematical, Physical and Engineering Sciences*, Vol. 459, No. 2038, 2003, pp. 2399-2430.

[17] Jović D., Jovanović R., Prvanović S., Petrović M., and Belić M., Counterpropagating beams in rotationally symmetric photonic lattices, *Optical Materials* , Vol. 30, 2008, pp.1173–1176

[18] Jović D. M., Prvanović S., Jovanović R. D., and Petrović M. S., Gaussian induced rotation in periodic photonic lattices, *Optic Letters*, Vol. 32, No 13, 2007, pp. 1857-1859

[19] Press William H., Teukolsky Saul A., Vetterling William T., Flannery Brian P., *Numerical Recepies in C: The Art of Scientific computing*, Cambridge University Press, 1992.

[20] Chang B.J.; Kim H.-S.; Lee H.Y.; Braunstein J., An Algorithm for Generation of Non-uniform Meshes for Finite Difference Time Domain Simulations, *Electromagnetic Field Computation, 2006 12th Biennial IEEE Conference*, Vol. No 1, 2006, pp. 53 – 53

[21] Kivshar Y. S. and Agrawal G. P., *Optical Solitons*, Academic Press, San Diego, 2003

[22] Joannopoulos J. D., Meade R. D., and Winn J. N., *Photonic Crystals: Molding the flow of light*, Princeton University Press, 1995.

[23] Belić M., Petrović M., Jović D., Strinić A., Arsenović D., Motzek K., Kaiser F., Ph. Jander, Denz C., Tlidi., and Mandel., Transverse modulational instabilities of counterpropagating solitons in photorefractive crystals, *Optics Express*, Vol. 12, No. 4, 2004, pp. 708-716.

[24] Houffman D. J., *Numerical methods for scientists and engineers*, McGraw-Hill, 1992.

[25] V. I. Petviashvili and O. A. Pokhotelov, *Solitary Waves in Plasmas and in the Atmosphere* , Gordon and Breach, Philadelphia, 1992.

[26] J. Yang, I. Makasyuk, A. Bezryadina, and Z. Chen, Dipole and quadruple solitons in optically-induced two-dimensional photonic lattices: theory and experiment, *Studies in Applied Mathematics* Vol.113, 2004, pp. 389-412