Run time optimizations for the Wave Propagation Method

Matthias Fertig
Konstanz University of Applied Sciences, Alfred-Wachtel Strae 8, 78469 Konstanz, Germany
matthias.fertig@htwg-konstanz.de

Abstract: This paper proposes three extensions of the Wave Propagation Method (WPM) to optimize run times. The optimizations apply to homogeneous symmetric and inhomogeneous symmetric systems. They are neither based on approximations nor increase the memory consumption of the basic algorithm. Such run time optimizations are important to shorten product development cycles where a high number of parameter variations need to be investigated to find optimal solutions. Today, massively parallel implementations are first choice to reduce run times while single-thread algorithms still show potential to be optimized. This paper focuses on the optimization of the single-thread implementation of the scalar unidirectional WPM. The proposed optimizations achieve their run time reductions from symmetries in the spatial frequency domain and spatial index domain. Many problems provide such spatial symmetries or sections of free-space propagation through homogeneous medium so that the optimizations have practical relevance. The paper shows that the optimizations can significantly reduce run times of the single-thread implementation for such types of problems. To show the general potential of the optimizations and not limit the applicability to selected examples, run times and accuracy are analysed for simulations of random homogeneous symmetric and inhomogeneous symmetric systems that are traversed by random Gaussian beams. The benchmarking is performed for a variation of the grid size and performance is compared to the single-thread and a massively parallel implementation of the standard algorithm. The results show that the optimizations reduce run times by a factor of 10 up to over 1000 dependent on the characteristics of the system. The proposed optimizations are on-demand and apply on-the-fly, show no loss of accuracy, execute in-place and do not consume additional memory. They do not depend on the number of different refractive index and are applicable to systems that change their characteristics during simulation.

1. Introduction

Run time and memory consumption are two key indicators for the performance of algorithms. The performance of algorithms is important for the development of photonic devices because thousands of simulations on variations in device parameters are required to find the optimal solution. Such simulations usually contain homogeneities and symmetries that can be used to optimize the simulator performance. Such symmetries typically exist for waveguides, tapers, splitters, couplers, gratings, lenses or free-space propagation between lenses. It is therefore highly desirable to reduce run times and reduce or limit memory consumption during simulations wherever possible. As hardware speeds increase through parallelization and hardware costs reduce due to mass production and general availability, the optimization of single-thread performance tends to be neglected. But even well known and deeply investigated algorithms often contain the potential to improve single-thread performance. This paper focuses on the single-thread performance of the three-dimensional WPM. It provides three types of optimizations to improve run times without any cost in terms of memory consumption or accuracy.

The contribution of this paper is an optimized single-thread implementation of the WPM for homogeneous symmetric and inhomogeneous symmetric systems. The optimizations achieve run time improvements by a factor 10 to 500, which is to say that at least up to 10 times as many simulations can be performed at the same time if the system shows spatial symmetries. The optimizations retain the accuracy and extend the original algorithm, perform an efficient grid analysis and apply on-the-fly and do not consume additional memory. The optimizations are introduced for three-dimensional systems and uni-directional propagation to show the basic concepts. But the proposed principles of optimization affect the propagation scheme of the WPM so that they are also applicable to bidirectional propagation or vector waves. The paper is also a comfortable starting point for engineers who are

new in the field of optical Fourier simulators because it provides the basic theory and implementation examples to start quickly and adapt the code to individual needs.

Beam propagation methods have been primarily used for simulation of light propagation in waveguides. By application of a propagation operator, derived from the slowly varying envelope approximation of the Helmholtz equation, propagation methods are typically restricted to forward propagation and the paraxial regime. In the original BPM scheme by Feit and Fleck [1], the propagation operator was split into two operators, a homogenous medium propagation in the averaged index and a thin element transmission through the index variation. Due to this operator splitting an additional restriction to small index variations was introduced. With the application of propagation methods to more general optical components, such as gradient index media, aspheric lenses and gratings, there is much interest in removing these restrictions. The BPM was improved and extended in various directions. A wide-angular BPM has been introduced by Sharma and Agrawal in [2] or Hadley in [3] to enable the BPM for non-paraxial propagation. Changbao presented a three-dimensional wide-angular BPM in [4] for optical waveguide structures. A semivectorial wide angular BPM was presented by Lee in [5]. The BPM was extended to very-wide angles in [6] but the error that depends on the separation of the propagation operator is still persistent in those methods. Several vector extensions of the BPM have been presented by Yamauchi [7], Liu [8] and [9]. Yamauchi introduced a modified semivectorial beam propagation method retaining the longitudinal field component, Liu published the analysis of polarized modes of rib waveguides with a semivectorial BPM and Perez presented a fully-vectorial three-dimensional extension of the BPM. A third class of extensions is based on the finite element (FE) approach as published by Tsui [10], Stern [11], Pinheiro [12] and Obayya [13]. The FE-based extensions of the BPM are optimized for scalar, semivectorial and full-vectorial fields as well as numerically efficient methods in the order of citation. The fourth and last class of extensions in this brief overview of BPM-based methods is the multigrid approach that is used to reduce computational effort at regions in a system that allow a reduced degree of accuracy. Sewell has introduced a multigrid method for electromagnetic computation in [14]. The BPM was subject to various optimizations in the field of innovative optical simulation methods, but except for the Padé approximation in [3] none of the BPM-based approaches overcomes the limitation that originate from the separation of the operators.

The wave propagation method (WPM) has been introduced by Brenner and Singer in 1993 [15] in order to overcome the two major limitations of the BPM, the restrictions to paraxial propagation and the limitation to small index vatiations. Instead of splitting the propagation operator into two parts, the WPM decomposes a field distribution into its plane wave components and performs a non-paraxial plane-wave propagation in an inhomogenous medium for each plane wave component. The field in the next step is then calculated as a sum over all plane wave components. Since this sum cannot be performed in an FFT-operation, the calculation time of the three-dimensional WPM grows with $\mathcal{O}(N^4)$ whereas the three-dimensional BPM calculation time utilizing FFT and inverse FFT is proportional to $\mathcal{O}(N^2)$, taking N as the number of spatial samples. In [15], the accuracy of the scalar WPM was validated for propagation angles up to 85 degrees and also for index steps greater than 1. With the WPM, the application range could be significantly extended. It has been used for light propagation in gradient index lenses and in commercial software [19] for the calculation of aspheric lenses. Since the original WPM also extended the angular range significantly, the WPM has been extended in 2009 to vector waves with the VWPM by Fertig and Brenner [16] and in 2011 to three-dimensional bidirectional propagation of vector waves by Fertig [17]. In 2017, Brenner proposed a high-speed version of the WPM for systems with a small number of different refractive indice [18] and homogeneous regions in the aperture. With this version a speedup of up to 40000 was achieved for selected system configurations and the complexity reduced from $\mathcal{O}(n^4)$ to $\mathcal{O}(n^2)$.

The paper is organized in five sections. Section 2 explains the original scalar WPM algorithm and introduces the nomenclature used in this paper. The run time optimizations are introduced in section 3. There, the standard algorithm is extended by four individual steps: 'GridAnalysis' (3.1), 'HomogOpt' (3.2), 'FreqOpt' (3.3) and 'SpatOpt' (3.4). The optimizations are incrementally benchmarked in section 4. They are analysed with the original unidirectional scalar algorithm [15] because the proposed optimizations rely on spatial frequency symmetries and symmetric index distributions. The vector [16] or bidirectional [17] versions would siginficantly increase the introduction without being an advantage for the verification. Section 4.3 investigates the improvements for homogeneous symmetric and section 4.4 for inhomogeneous symmetric systems. All benchmarking is performed with random Gaussian beams and random homogeneous symmetric and random inhomogeneous symmetric systems to show the general applicability of the approach. The benchmarking performs on variations of the grid size and provides run times as well as cycle counts to allow performance comparisons with systems running at different clock speeds. CPU single thread timing breakdown and GPU kernel timing breakdowns are shown in sections 4.3 and 4.4.

2. Wave Propagation Method

A discretized three-dimensional system is defined by a spatial refractive index distribution $n(i,j,l)$ where $x = i \cdot \triangle x$, $y = j \cdot \triangle x$ and $z = l \cdot \triangle z$ is the position defined by the index $-n_x/2 \leq i \leq n_x/2$, $-n/y/2 \leq j \leq n_y/2$ and $0 \leq l \leq n_z - 2$. The system has an aperture $X \cdot Y = n_x \cdot \triangle x \cdot n_y \cdot \triangle y$ and a length $Z = n_z \cdot \triangle x$. A layer in the xy-plane of such a system is called homogeneous if the refractive index distribution $n(i,j) = const \; \forall \; i,j$ and symmetric in two dimensions if $n(n_x - 1 - i, j) = n(i, n_y - 1 - j) = n(n_x - 1 - i, n_x - 1 - j) = n(i,j)$. The entire system is called homogeneous if $n = const \; \forall \; i,j,l$.

The WPM algorithm computes the electromagnetic field distribution of a complex harmonic wave propagating along an axis of propagation, i.e. the z-axis in this paper. The x-axis and y-axis are lateral axes and, together with the z-axis, span an orthogonal cartesian coordinate system. The system is split into n_z layers parallel to the xy-plane. Each xy-plane parallel is defined by $n_x \cdot n_y$ samples. The $n_x \cdot n_y \cdot n_z$ samples span the simulation grid of size $(n_x \cdot \triangle x) \cdot (n_y \cdot \triangle y) \cdot (n_z \cdot \triangle z) = X \cdot Y \cdot Z$. $\triangle x$ and $\triangle y$ are obtained from X/n_x and Y/n_y where $X \cdot Y$ is the aperture area. The WPM iterates the z-axis for n_z steps and computes the field distribution in layer $l+1$ from the field distribution in layer l. The electric field in a layer l is E_l and the refractive index distribution is n_l.

2.1. Plane wave decomposition ('FFT')

The WPM is a Fourier method as it derives the spatial field distribution at E_{iz+1} from the plane wave decomposition or plane wave spectrum e_l of layer E_l. The spectrum is obtained from a Fourier Transformation $\mathscr{F}\{\;\}$

$$e_l(\mathbf{k}_\perp) = \mathscr{F}\{E_l(\mathbf{r}_\perp)\} \tag{1}$$

where $\mathscr{F}\{\;\}$ is the two-dimensional Fourier transformation and $\mathbf{r}_\perp = (x \; y)^T$ is the lateral vector in space. $\mathbf{k}_\perp = (k_x \; k_y)^T$ is the lateral spatial frequency vector with $-0.5/\triangle x \leq k_x < 0.5/\triangle x$ and $-0.5/\triangle y \leq k_y < 0.5/\triangle y$.

2.1.1. Spatial frequency normalization and sign modulation

Due to the symmetries of the Fourier transformation, each plane wave amplitude $e_l(\mathbf{k}_\perp)$ has to be multiplied by a factor $(-1)^{p+q}$ [15]. p and q are integer numbers in the range $-n_x/2 \leq p < nx/2$ and $-n_y/2 \leq q < ny/2$ with $k_x = p/X$ and $k_y = q/Y$. If the Fourier transformation does not normalize by $(n_x \cdot n_y)$ the factor has to be expanded to $(-1)^{p+q}/(n_x \cdot n_y)$.

2.2. Amplitude transformation and phase shift ('WPM-Loops')

While propagating a distance $\triangle z = Z/n_z$ through layer $l+1$, each plane wave component $e_{l+1}(\mathbf{k}_\perp)$ experiences a phase shift $\phi_{l+1}(\mathbf{r}_\perp, \mathbf{k}_\perp) = k_{z,l+1}(\mathbf{r}_\perp, \mathbf{k}_\perp) \cdot \triangle z$, where $k_{z,l+1} = sqrt\{k^2_{l+1}(\mathbf{r}_\perp, \mathbf{k}_\perp) - \mathbf{k}^2_\perp\}$ is the z-component of the propagation vector $\mathbf{k} = (k_x \; k_y \; k_z)^T$ in layer $l+1$. $k_{l+1}(\mathbf{r}_\perp) = n_{l+1}(\mathbf{r}_\perp) \cdot k_0$ is the wave number in layer $l+1$. In contrast to the Split-Step-Propagation scheme [1] the wave numer is not derived from an average refractive index.

$$k_{z,l+1}(\mathbf{r}_\perp, \mathbf{k}_\perp) \quad = \quad \sqrt{(n_{l+1}(\mathbf{r}_\perp) \cdot k_0)^2 - \mathbf{k}^2_\perp} \tag{2}$$

where $k_0 = 2\pi/\lambda_0$ and λ_0 is the wavelength of the incident wave at $(l \cdot \triangle z) = z = 0$ is $E(x,y,0) = 1 \cdot exp\{\mathbf{j} \cdot \mathbf{r}_\perp \cdot \mathbf{k}_\perp\}$. Here, a unit amplitude is assumed for simplicity but without prejudice to the generality. In case of a constant refractive index the absolute of the amplitude remains unchanged.

2.2.1. Amplitude transformation

The electromagnetic wave crosses a boundary if the average refractive index in layers l and $l+1$ are different, i.e. $n_l(\mathbf{r}_\perp) \neq n_{l+1}(\mathbf{r}_\perp)$. In this case the z-components of the propagation vector experiences a change, i.e. $k_{z,l} \neq k_{z,l+1}$ and the amplitudes $e_l(\mathbf{k}_\perp)$ transform to $f_{l+1}(\mathbf{k}_\perp)$ according to the Fresnel coefficients of amplitude

$$t_{te}(\mathbf{r}_\perp, \mathbf{k}_\perp) \quad = \quad \frac{2 \cdot k_{z,l}(\mathbf{r}_\perp, \mathbf{k}_\perp)}{k_{z,l}(\mathbf{r}_\perp, \mathbf{k}_\perp) + k_{z,l+1}(\mathbf{r}_\perp, \mathbf{k}_\perp)} \tag{3}$$

$$f_{l+1}(\mathbf{r}_\perp, \mathbf{k}_\perp) \quad = \quad t_{te}(\mathbf{r}_\perp, \mathbf{k}_\perp) \cdot e_l(\mathbf{k}_\perp) \tag{4}$$

for TE-polarized waves. The analysis in this paper is focused on TE-polarized waves without effect on the proposed optimizations. $t_{te} = 1$ if $k_{z,l} = k_{z,l+1}$ or $n_l(\mathbf{r}_\perp) = n_{l+1}(\mathbf{r}_\perp)$. In the worst case, $(n_x \cdot n_y)$ different Fresnel numbers need to be calculated for one layer iteration l to $l+1$.

2.2.2. Space-dependent phase shift

The transmitted amplitude of each plane wave component f_{l+1} propagates a distance $\triangle z$ through layer $l+1$ and experiences a phase shift $\phi_{z,l+1}(\mathbf{r}_\perp,\mathbf{k}_\perp) = k_{z,l+1}(\mathbf{r}_\perp,\mathbf{k}_\perp)\cdot\triangle z$. The plane wave component is then

$$e_{l+1}(\mathbf{r}_\perp,\mathbf{k}_\perp) \quad = \quad f_{l+1}\cdot e^{\mathbf{j}\phi_{z,l+1}}\cdot e^{\mathbf{j}\cdot(\mathbf{k}_\perp\cdot\mathbf{r}_\perp)} = f_{l+1}\cdot e^{\mathbf{j}\phi_{z,l+1}}\cdot e^{\mathbf{j}\cdot\phi_\perp} = f_{l+1}\cdot e^{\mathbf{j}\phi_{l+1}} \tag{5}$$

where $\phi_{z,l+1}$ is the longitudinal, $\phi_\perp = \mathbf{r}_\perp\cdot\mathbf{k}_\perp$ is the lateral and $\phi_{k+1} = \phi_{z,k+1}+\phi_\perp$ is the total phase shift in layer $l+1$. 'WPM-Loops' is finished after $(n_x^2\cdot n_y^2)$ iterations when all plane wave components e_k experienced a space- and frequency-dependent transformation. An implementation of the unidirectional scalar WPM can be found in Appendix.

2.2.3. Evanescent modes

Evanescent modes are obtained for $k^2 \le k_\perp^2$. They are not further discussed in this paper as the proposed optimizations are applicable to those modes without modification. Evanescent modes are automatically considered by a complex propagation vector as shown in [17].

2.3. Superpostition of plane wave components

The WPM cannot benefit from an inverse FFT as for example the BPM [1] because f_{l+1} is space- and frequency-dependent. The field distribution in the space-domain is obtained from a superposition of space- and spatial frequency-dependent components

$$E_{l+1}(\mathbf{r}_\perp) = \int_{\mathbf{r}_\perp}\int_{\mathbf{k}_\perp} \underbrace{t_{te}(\mathbf{r}_\perp,\mathbf{k}_\perp)\cdot \overbrace{e_k(\mathbf{k}_\perp)}^{\mathscr{F}(E_l(\mathbf{r}_\perp))}}_{f_{l+1}(\mathbf{r}_\perp,\mathbf{k}_\perp)} \cdot \underbrace{e^{\mathbf{j}\cdot k_{z,l+1}\cdot\triangle z}\cdot e^{\mathbf{j}\mathbf{k}_\perp\cdot\mathbf{r}_\perp}}_{e^{\mathbf{j}(\phi_z+\phi_\perp)}}\ d\mathbf{k}_\perp\, d\mathbf{r}_\perp = \int_{\mathbf{r}_\perp}\int_{\mathbf{k}_\perp} f_{l+1}(\mathbf{r}_\perp,\mathbf{k}_\perp)\cdot e^{\mathbf{j}\phi}\ d\mathbf{k}_\perp\, d\mathbf{r}_\perp \tag{6}$$

This expression is the foundation for the 'HomogOpt' and 'SpatOpt' optimizations.

2.4. Spatial phase error and run time complexity

Due to a superposition of space- and frequency-dependent plane wave components the WPM provides exact optical path differences $\triangle n\cdot\triangle r$ for all plane wave components. There is no phase error as with the Split-Step-Propagation scheme and so no phase correction is required. Due to this difference in the propagation scheme, the complexity of the Split-Step-Propagation scheme is in $\mathscr{O}(n_x\cdot n_y)+\mathscr{O}(n_x\cdot n_y) = 2\cdot\mathscr{O}(n_x\cdot n_y)\in\mathscr{O}(n^2)$ while the complexity of the Wave-Propagation scheme is in $\mathscr{O}((n_x\cdot n_y)^2) = \mathscr{O}(n^4)$. This makes run time optimizations highly desirable.

3. Run time optimizations

The run time optimizations in this paper utilize symmetries in the spatial frequency vector ('FreqOpt'), symmetries in the spatial index distribution ('SpatOpt') and homogeneities ('HomogOpt'). In this context homogeneous layers are interpreted a special form of symmetry in the spatial refractive index distribution, where the index is constant.

3.1. Symmetry and homogeneity analysis ('Grid Analysis')

The optimizations require a layer analysis step, called 'GridAnalysis' in this paper, to investigate for symmetric and homogeneous layers. Listing 2 shows an efficient way to perform a two-dimensional grid analysis in two half-range loops. With the WPM there is no need to calculate an average refractive index and so 'GridAnalysis' is an additional step of calculation. An implementation of 'GridAnalysis' can be found in Appendix.

3.2. Optimization from homogeneous layers ('HomogOpt')

In case of homogeneous layers, $t_{te}(\mathbf{r}_\perp,\mathbf{k}_\perp)$ becomes $t_{te}(\mathbf{k}_\perp)$, $k_z(\mathbf{r}_\perp,\mathbf{k}_\perp)$ becomes a space-independent $k_z(\mathbf{k}_\perp)$ and equation 6 reduces to an inverse Fourier Transformation

$$E(\mathbf{r}_\perp) = \int_{\mathbf{r}_\perp}\int_{\mathbf{k}_\perp} \underbrace{t_{te}(\mathbf{k}'_\perp)\cdot e_{iz}(\mathbf{k}'_\perp)}_{f_{l+1}(\mathbf{k}'_\perp)}\cdot e^{\mathbf{j}\phi}\ d\mathbf{k}'_\perp\, d\mathbf{r}'_\perp = n_x\cdot n_y\cdot\int_{\mathbf{k}_\perp} f_{l+1}(\mathbf{k}'_\perp)\cdot e^{\mathbf{j}\phi_z}\cdot e^{\mathbf{j}\mathbf{k}'_\perp\cdot\mathbf{r}_\perp}\ d\mathbf{k}'_\perp = \mathscr{F}^{-1}\left\{f_{l+1}(\mathbf{k}_\perp)\cdot e^{\mathbf{j}\phi_z}\right\} \tag{7}$$

because all space-dependent components now depend on a constant refractive index and thereby become independent of space. ϕ_z becomes independent from space and the WPM reduces to the Plane-Wave-Spectrum (PWS). The complexity of the PWS is in $\mathscr{O}(n^2)$, where n is the number of samples in the aperture. An implementation of 'HomogOpt' can be found in Appendix.

3.3. *Optimization from symmetries in the spatial frequency vector ('FreqOpt')*

The spatial frequency vector $\mathbf{k}$ is symmetric around the zero frequency $\mathbf{k}_\perp = (0\ 0)^T$, i.e. perpendicular propagation as shown in table 1. The three-dimensional propagation vector $\mathbf{k} = (k_x\ k_y\ k_z)^T$ derives from its transveral components k_x and k_y and the wave number k_{l+1}, where

$$k_x(p) = \frac{p}{n_x \cdot \triangle x} = \frac{p}{X} \quad , \quad -\frac{n_x}{2} \leq p < \frac{n_x}{2} \tag{8}$$

$$k_y(q) = \frac{q}{n_y \cdot \triangle y} = \frac{q}{Y} \quad , \quad -\frac{n_y}{2} \leq q < \frac{n_y}{2} \tag{9}$$

Table 1 shows the symmetries in k_x and k_y for $n_x = n_y = 8$. There's a two-dimensional symmetry for $1 \leq p \leq n_x/2-1$ and $1 \leq q \leq n_y/2-1$ (called '2D' in this paper) and one-dimensional symmetries for $p = 0$ or $p = n_x/2$ and $1 \leq q \leq n_y/2-1$ (called '1DX') and $q = 0$ or $q = n_x/2$ and $1 \leq p \leq n_x/2-1$ (called '1DY') due to the situation that zero and minimum frequencies have no counterpart with opposite sign. The four edge frequencies occur only once and have no symmetries (called 'NO'). 'FreqOpt' uses a special scheme to address all these situations with a low number of if-clauses.

	$k_x[0]$	$k_x[1]$	...	$k_x[3]$	$k_x[4]$	$k_x[5]$	...	$k_x[7]$
$k_y[0]$	(0,0)	(1/X,0/Y)	...	(3/X,0/Y)	(-4/X,0/Y)	($-k_x[3]$,0/Y)	...	(-1/X,0/Y)
$k_y[1]$	(0/X,1/Y)	(1/X,1/Y)	...	(3/X,1/Y)	(-4/X,1/Y)	($-k_x[3]$,1/Y)	...	($-k_x[1]$,1/Y)
$k_y[2]$	(0/X,2/Y)	(1/X,2/Y)	...	(3/X,2/Y)	(-4/X,2/Y)	($-k_x[3]$,2/Y)	...	($-k_x[1]$,2/Y)
$k_y[3]$	(0/X,3/Y)	(1/X,3/Y)	...	(3/X,3/Y)	(-4/X,3/Y)	($-k_x[3]$,3/Y)	...	($-k_x[1]$,3/Y)
$k_y[4]$	(0,-4/Y)	(1/X,-4/Y)	...	(3/X,-4/Y)	(-4/X,-4/Y)	($-k_x[3]$,-4/Y)	...	($-k_x[1]$,-4/Y)
$k_y[5]$	(0/X,$-k_y[3]$)	(1/X,$-k_y[3]$)	...	(3/X,$-k_y[3]$)	(-4/X,$-k_y[3]$)	($-k_x[3]$,$-k_y[3]$)	...	($-k_x[1]$,$-k_y[3]$)
$k_y[5]$	(0/X,$-k_y[2]$)	(1/X,$-k_y[2]$)	...	(3/X,$-k_y[2]$)	(-4/X,$-k_y[2]$)	($-k_x[3]$,$-k_y[2]$)	...	($-k_x[1]$,$-k_y[2]$)
$k_y[7]$	(0/X,$-k_y[1]$)	(1/X,$-k_y[1]$)	...	(3/X,$-k_y[1]$)	(-4/X,$-k_y[1]$)	($-k_x[3]$,$-k_y[1]$)	...	($-k_x[1]$,$-k_y[1]$)

Table 1. Symmetries in the two-dimensional spatial frequency vector for $n_x = n_y = 8$.

Expression 6 then transforms to

$$E(\mathbf{r}_\perp) = \int_{\mathbf{r}_\perp}\int_{\mathbf{k}_\perp} f_{l+1}(\mathbf{r}'_\perp, \mathbf{k}'_\perp) \cdot e^{\mathbf{j}\phi}\ d\mathbf{k}'_\perp\ d\mathbf{r}'_\perp = \int_{\mathbf{r}_\perp}\int_{\mathbf{0}_\perp}^{\mathbf{k}^+_\perp} f_{l+1}(\mathbf{r}'_\perp, \pm\mathbf{k}'_\perp) \cdot e^{\mathbf{j}\phi}\ d\mathbf{k}'_\perp\ d\mathbf{r}'_\perp \tag{10}$$

where $\mathbf{k}^+_\perp$ are the positive lateral spatial frequencies and the zero frequency is $\mathbf{k}_\perp = (x\ y)^T = (0\ 0)^T = \mathbf{0}_\perp$. $\pm\mathbf{k}'_\perp$ is in the limits and derived from $\mathbf{k}^+_\perp$ as described in the following sections. The reduction rate for the number of iterations depends on the grid size as shown in figure 1. A maximum improvement factor up to four can be expected for 'FreqOpt'.

3.3.1. '2D' symmetric frequencies (4-fold symmetry)

are obtained for $1 \leq i \leq n_x/2-1$ and $1 \leq j \leq n_y/2-1$ and provide two-dimensional symmetries

$$k^2_\perp(n_x - p, n_y - q) = k^2_\perp(n_x - p, q) = k^2_\perp(p, n_y - q) = k^2_\perp(p, q) \tag{11}$$

so that the z-component of the propagation vector k^2_z is $k^2 - k^2_\perp(p,q)$ with $k^2_\perp(p,q) = k^2_x(p) + k^2_y(q)$ and $k = n_{l+1}(\mathbf{r}_\perp) \cdot k_0$. The number of frequencies in this bucket is and determined by the number of samples in the aperture, i.e. $n_{2D} = 4 \cdot (n_x/2-1) \cdot (n_y/2-1)$. The number of iterations reduce by a factor four for these frequencies.

3.3.2. '1DX' symmetric frequencies (2-fold symmetry)

are obtained for $1 \leq p \leq n_x/2-1$ and $q = 0$ or $q = n_y/2$ and provide one-dimensional symmetries

$$k^2_\perp(n_x - p, q) = k^2_\perp(p, q) \tag{12}$$

so that the z-component of the propagation vector k^2_z is $k^2 - k^2_\perp(p,0)$ or $k^2 - k^2_\perp(p, n_y/2)$, where $k = n_{l+1}(\mathbf{r}_\perp) \cdot k_0$. The number of frequencies in this bucket is determined by the number of samples in the aperture, i.e. $n_{1DX} = 2 \cdot (2 \cdot (n_x/2-1))$. The number of iterations reduce by a factor two for these frequencies.

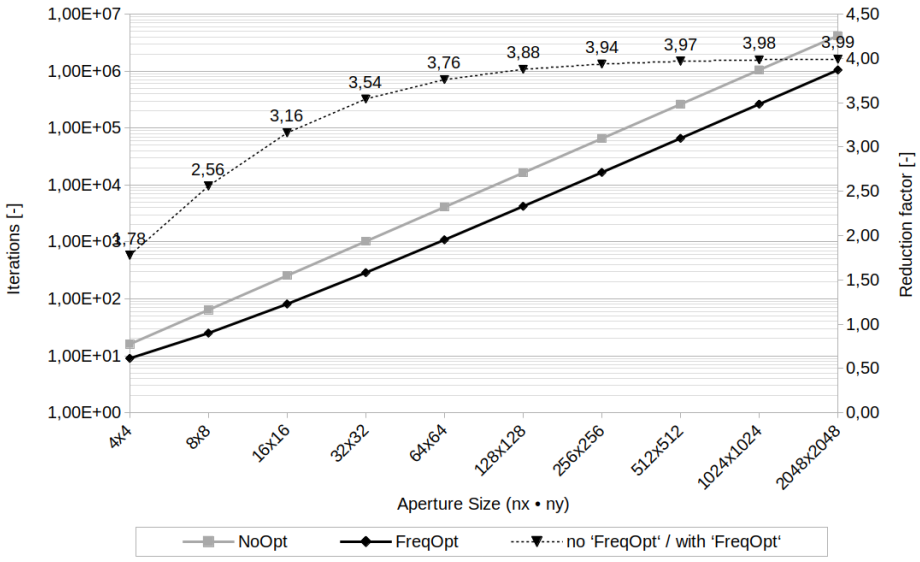

Fig. 1. Number of spatial frequency iterations with 'FreqOpt' n_f (left axis) and reduction factor $n_f/(n_x \cdot n_y)$ (right axis) over aperture size from theory.

3.3.3. '1DY' symmetric frequencies (2-fold symmetry)

are obtained for $1 \leq q \leq n_x/2-1$ and $p = 0$ or $p = n_x/2$ and provide a one-dimensional symmetries

$$k_\perp^2(p, n_y - q) = k_\perp^2(p, q) \tag{13}$$

so that the z-component of the propagation vector k_z^2 is $k^2 - k_\perp^2(0,q)$ or $k^2 - k_\perp^2(n_x/2,q)$, where $k = n_{l+1}(\mathbf{r}_\perp) \cdot k_0$. The number of frequencies in this bucket is determined by the number of samples in the aperture, i.e. $n_{1DY} = 2 \cdot (2 \cdot (n_y/2-1))$. The number of iterations reduce by a factor two for these frequencies.

3.3.4. 'NO' symmetric frequencies

are obtained for all combinations of $p \in \{0, n_x/2\}$ and $q \in \{0, n_y/2\}$ and provide no symmetries so that four individual z-components of the propagation vector are required. The z-component of the propagation vector k_z^2 is $k^2 - k_\perp^2(0,0)$, $k^2 - k_\perp^2(0,n_y/2)$, $k^2 - k_\perp^2(n_x/2,0)$ or $k^2 - k_\perp^2(n_x/2,n_y/2)$, where $k = n_{l+1}(\mathbf{r}_\perp) \cdot k_0$. This number of frequencies is constant $n_{NO} = 4$ and independent from the number of samples in the aperture.

3.3.5. Total number of spatial frequencies with 'FreqOpt'

The total number of frequencies in the four buckets with 'FreqOpt' is

$$n_f = n_{2D} + n_{1DX} + n_{1DY} + n_{NO} \tag{14}$$

This is the number of iterations on the two-dimensonal spatial frequency vector with 'FreqOpt'. Figure 1 shows n_f over the number of samples in the aperture $(n_x \cdot n_y)$ for 'NoOpt' and 'FreqOpt' and the reduction factor compared to 'NoOpt' $n_f/(n_x \cdot n_y)$. The figure shows that the improvement saturates at a factor four for larger grids and that the improvement for small grids is lower. But even for small grids of size 4x4 the reduction of iterations is a factor of ≈ 1.8. For the maximum aperture size in this paper of 256x256 the reduction factor for 'FreqOpt' is expected to be 3.94.

3.4. Optimization from symmetries in the spatial refractive index distribution ('SpatOpt')

Symmetries in the spatial index distribution do not always exist and need to be detected layer by layer. This is performed by 'GridAnalysis' in two nested a half-range loops. If symmetries in the spatial refractive index distribution exist, equation 5 becomes

$$\phi_{z,l+1}(\pm\mathbf{r}_\perp, \mathbf{k}_\perp) = \phi_{z,l+1}(\pm x, \pm y, k_x, k_y) = \triangle z \cdot \sqrt{n_{l+1}^2(\pm x, \pm y) \cdot k_0^2 - (k_x^2 + k_y^2)} \tag{15}$$

where $\pm x$ is indexed by i and $(nx-1-i)$ and $\pm y$ is indexed by j and $(ny-1-j)$ if the point of symmetry is the center of the aperture. The number of iterations thereby reduce by a factor four from $(n_x \cdot n_y)$ to $(n_x/2) \cdot (n_y/2)$,

which is the expected run time improvemenent for 'SpatOpt'. An implementation of 'SpatOpt' can be found in Appendix. The half-cycle loops are only executed in case of spatial symmetries in the x- and y-axis but a one-dimensional half-cycle execution is possible if symmetries in just one of the lateral axes occurs thereby reducing the run time by a factor two. This paper focuses on two-dimensional symmetries because the highest run time improvement can be expected. In combination with 'FreqOpt' equation 15 changes to

$$\phi_{z,l+1}(\pm\mathbf{r}_\perp,\pm\mathbf{k}_\perp)=\phi_{z,l+1}(\pm x,\pm y,\pm k_x,\pm k_y)=\sqrt{n_{l+1}^2(\pm x,\pm y)\cdot k_0^2-\left(\pm k_x^2+\pm k_y^2\right)}\cdot\triangle z \tag{16}$$

where $\pm k_x$ and $\pm k_y$ are symmetric spatial frequencies as shown in sections 3.3.1 to 3.3.3. The spatial frequencies in section 3.3.4 are treated as in the standard algorithm. In combination with 'FreqOpt' expression 10 transforms to

$$E(\mathbf{r}_\perp)=\int_{\mathbf{r}_\perp}\int_{\mathbf{0}_\perp}^{\mathbf{k}_\perp^+} f_{l+1}(\mathbf{r}'_\perp,\pm\mathbf{k}'_\perp)\cdot e^{\mathbf{j}\phi}\;\;d\mathbf{k}'_\perp\;d\mathbf{r}'_\perp=\int_{\mathbf{0}_\perp}^{\mathbf{r}_\perp^+}\int_{\mathbf{0}_\perp}^{\mathbf{k}_\perp^+} f_{l+1}(\pm\mathbf{r}'_\perp,\pm\mathbf{k}'_\perp)\cdot e^{\mathbf{j}\phi}\;\;d\mathbf{k}'_\perp\;d\mathbf{r}'_\perp \tag{17}$$

where $\mathbf{r}_\perp^+$ are the positive spatial locations in the aperture $-X/2\le y<X/2$ and $-Y/2\le x<Y/2$. The center position in the aperture is obtained for $\mathbf{r}_\perp=(x\;y)^T=(0\;0)^T=\mathbf{0}_\perp$. A maximum improvement factor of four can be expected for 'SpatOpt'.

4. Benchmarking

The optimizations are analysed with two types of constrained random performance benchmarks. Each type of benchmark is constrained so that random homogeneous symmetric and random inhomogeneous symmetric systems are traversed by Gaussian beams $exp(\mathbf{r}^2/\sigma^2+\mathbf{j}\cdot\mathbf{k}\cdot\mathbf{r})$ with unity amplitude and wavelengths $500nm\le\lambda\le 1500nm$. The Gaussian beam has a waist $\sigma=\lambda$, vertical incidence and is positioned in the center of the aperture, i.e. $\mathbf{r}_0=1/2\cdot(X\;Y\;0)^T$. The aperture size is $X\cdot Y=(4\lambda)^2$ and $Z=n_z\cdot\triangle z$ with $\triangle z=\triangle x=\triangle y$ and $n_z=4$. n_z is assigned a small number because it is relevant for the improvements and to keep memory consumption and overall benchmarking run times small.

Each benchmark simulates on grid sizes from 8x8x4 to 256x256x4 samples. The aperture grid size is doubled in each axis per grid iteration and six different grid sizes are analysed. 120 runs are performed per level of optimization and 720 runs in total. The levels of optimization are 'NoOpt', which is the original WPM algorithm ([15], code:1), 'HomogOpt' (sec:3.2, code:3), 'HomogOpt+FreqOpt' (sec:3.3, code:4) and 'FreqOpt+SpatOpt' (sec:3.4, code:5). The benchmarks report on run times and speedup factors over levels of optimization and grid sizes as well as accuracy.

4.1. Implementation

The optimized codes are shown in the Appendix. The program code contains all levels of optimization and decides for the appropriate optimization on layer granularity based on 'GridAnalysis' during simulation. 'HomogOpt' and 'SpatOpt' are mutually exclusive optimizations that benefit both from 'FreqOpt'. All optimizations thereby extend but not replace the original algorithm. The code can simulate all types of systems and is not limited to homogeneous symmetric and inhomogeneous asymmetric systems. 'GridAnalysis' is applied for every iteration $0\le l<n_z-2$ to detect homogeneous or inhomogeneous symmetric layers and trigger the appropriate level of optimization. Iterations on l are limited to n_z-2 because every iteration calculates the field in layer $l+1$ from layer l and the last memory index is n_z-1. This yields a total number of n_z-1 iterations. 'FreqOpt' is applicable to all types of systems because symmetries in the spatial frequencies are independent. The benchmarks run on a CPU with 2.1 GHz and on a GPU with 1.13 GHz clock frequency. The program code is single-threaded to focus on the optimizations. To average-out occasional run time glitches from task switches or other interruptions of the operating system, all run times are average values, obtained from 10 runs per grid size and level of optimization.

4.2. Accuracy and memory consumption

All optimizations are benchmarked against the standard program code shown in the Appendix. The maximum tolerance in the relative error of the field distribition is defined to 0.001 percent per sample, i.e. $-50dB$. This accuracy is achieved for all runs. All optimizations perform in-place and require no additional memory. Individual simulations utilize memory for the complex-valued electric field and refractive index, i.e. $4\cdot(n_x\cdot n_y\cdot n_z)\cdot 4$ Bytes for IEEE single precision and $4\cdot(n_x\cdot n_y\cdot n_z)\cdot 8$ Bytes for IEEE double precision floating point numbers.

4.3. Homogeneous symmetric systems

Figure 2 (left) shows run times and speedup factors from benchmarking homogeneous symmetric systems. The x-axis shows the number of samples on a linear scale and the left y-axis the run time in milliseconds on a logarithmic

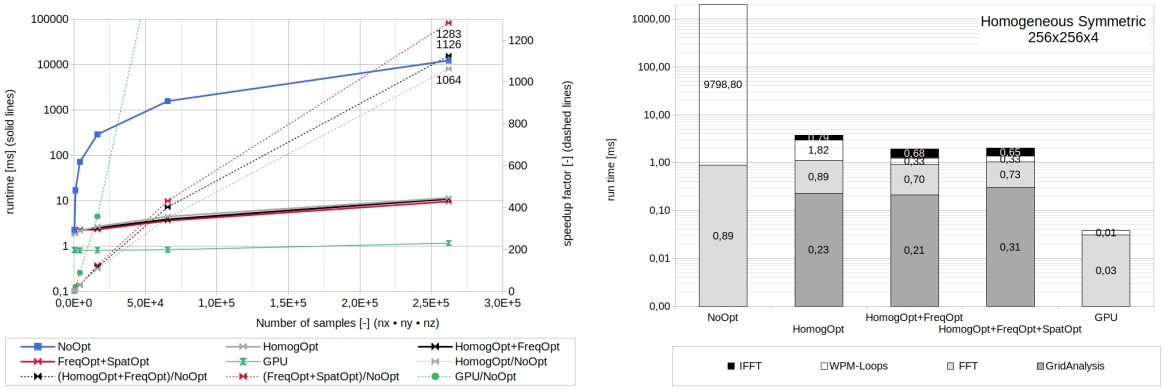

Fig. 2. Run time and speedup factor over system grid size for homogeneous symmetric systems.

scale (base 10). The right y-axis shows the speedup factor on a linear scale. Solid lines indicate run times and are read against the left y-axis while dashed lines indicate speedup factors and are read against the right y-axis. Results are shown for all levels of optimizations and the GPU runs. The GPU runs do not apply any optimization. The run times are given in milliseconds and the speedup factors of a level of optimization are derived from the run time for the level of optimization over the run time for 'NoOpt'. Figure 2 (right) shows the thread and kernel timing breakdown for one iteration of a level of optimization. The latencies are plotted in milliseconds on a logscale axis.

Figure 2 (left) shows that the run times for homogeneous symmetric systems reduce significantly with 'HomogOpt'. This is expected because expression 6 reduces to expression 7 so that four nested loops over $d\mathbf{k}'_\perp\, d\mathbf{r}'_\perp$ reduce to two nested loops over $d\mathbf{k}'_\perp$ and the complexity changes from $\mathcal{O}(n^4)$ to $\mathcal{O}(n^2)$. The graphs for 'NoOpt' and 'HomogOpt' or 'HomogOpt+FreqOpt' show a three order of magnitude difference for large grids. A corresponding speedup factor of up to 1288 is achieved with the single-thread code. All speedup factors for the three levels of optimization are in a similar range and show a similar trend while 'HomogOpt+FreqOpt' shows the best single-thread performance as expected. 'HomogOpt' is the main contributor to this improvement.

Figure 2 (right) depicts the timing breakdown of an iteration for the simulation of an homogeneous symmetric system of size 256x256x4. It shows that 'GridAnalysis' requires six percent of the latency, 'FFT' takes 24 percent, 'WPM-Loops' takes 49 percent for 'HomogOpt', 17 percent for 'HomogOpt+FreqOpt' and 16 percent for 'HomogOpt+FreqOpt+SpatOpt'. The drop from 17 to 16 percent is caused by the higher latency for 'GridAnalysis' and a constant latency for 'WPM-Loops'. 'IFFT' takes 21 percent. In summary forward and inverse Fourier transformation consume 45 percent and 'WPM-Loops' up to 50 percent of the latency. Approximately five percent is required for 'GridAnalysis'.

The massively parallel implementation of the WPM shows remarkable performance and a speedup up to four orders of magnitude is achieved. The difference in run times between single-thread and massively parallel implementation is one order of magnitude or below for the grid sizes under investigation. The difference is smaller for small grids.

4.4. Inomogeneous symmetric systems

Figure 3 shows the run times and speedup factors from benchmarking inhomogeneous symmetric systems (left) as well as thread and kernel timing breakdowns (right). The figure has the same structure as figure 2.

Figure 3 (left) shows that 'HomogOpt' cannot improven run times in case of inhomogeneous symmetric layers as expected. The run times for 'HomogOpt' are slightly higher as for 'NoOpt' because 'GridAnalysis' is applied and the speedup are below one. Run times reduce with 'FreqOpt+HomogOpt' and a speedup up to 3.6 is achieved. This corresponds well with the analysis in figure 1. Run times further reduce with 'FreqOpt+SpatOpt' and a total single-thread speedup up to 10 is achieved. The theoretic limit of four for 'FreqOpt' times four for 'SpatOpt' is not achieved due to 'GridAnalysis' and an expensive memory index arithmetic as well as several if-else clauses needed to retain the full applicability of the implementation to all types of systems.

Figure 3 (right) depicts the timing breakdown of an iteration for the simulation of an inhomogeneous symmetric system of size 256x256x4. It shows that 'GridAnalysis' requires 0.03 percent of the latency, 'FFT' takes 0.08 percent and 'WPM-Loops' take 99.8 percent for 'HomogOpt+FreqOpt+SpatOpt'. The latencies for 'GridAnalysis' and 'FFT' are negligible. Run time for 'WPM-Loops' increases by 13 percent for 'HomogOpt', reduces by

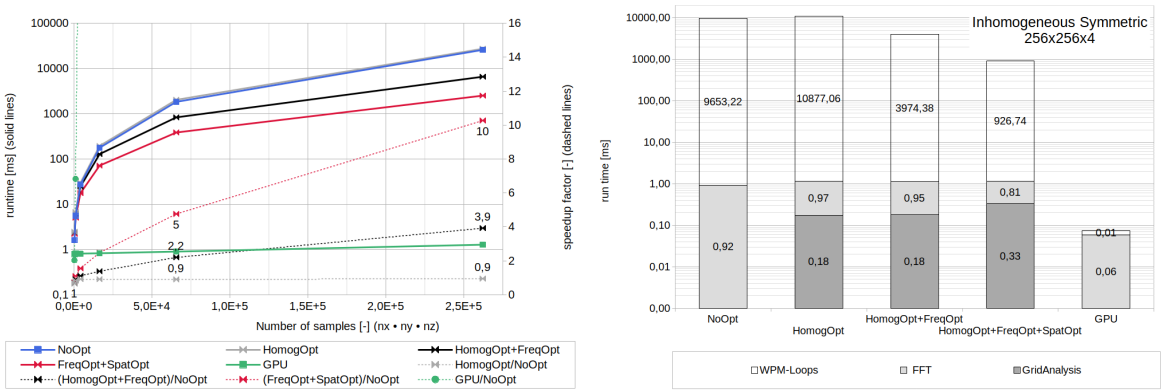

Fig. 3. Run time and speedup factor over system grid size (left) and CPU thread timing breakdown over level of optimization plus GPU kernel timing breakdown (right) for inhomogeneous symmetric systems.

59 percent for 'HomogOpt+FreqOpt' and by 90 percent for 'HomogOpt+FreqOpt+SpatOpt' when compared to 'NoOpt'.

The massively parallel implementation is again significantly faster than the single-thread implementation as expected because the GPU implementation allows to assign a compute core to every sample on the grid.

5. Conclusions

The paper demonstrates that the WPM can be optimized in-place and without loss of accuracy. The proposed optimizations apply to homogeneous and inhomogeneous symmetric systems dynamically on layer basis. They are suitable for apertures that change characteristics during simulation, e.g. from electro-optic, thermic or mechanic effects.

The single-thread optimization for homogeneous symmetric (sub-)systems (3.2) reduce the complexity of the original three-dimensional WPM algorithm from $\mathcal{O}(n^4)$ to $\mathcal{O}(n^2)$.

The single-thread optimization for symmetries in the spatial frequency vector (3.3) reduces run times up to a factor four for all types of apertures.

The single-thread optimization for symmetries in the refractive index distribution (3.4) reduce run times by a factor four for inhomogeneous symmetric apertures.

The speedup factors of the single-thread optimizations scale exponentially with the aperture size and achieve a maximum speedup of over 1000 for homogeneous, 10 for inhomogeneous symmetric and up to 4 for all other types of apertures.

The latency for a two-dimensional on-the-fly 'GridAnalysis' consumes up to six percent of the performance when combined with the proposed optimizations for symmetric homogeneous system and is negligible when combined with the proposed optimizations for inhomogeneous symmetric apertures.

The maximum speed-up factor from theory for 'FreqOpt+SpatOpt' is not reached due to intensive memory indexing arithmetics as shown in code listings 4 and 5 for 'FreqOpt' and 'SpatOpt' and several if-clauses needed to retain the full applicability to all types of apertures.

The WPM algorithm shows excellent performance on a GPU. A speedup of four orders of magnitude are easily achieved when compared to the original single-thread algorithm and three orders of magnitude when compared to the proposed single-thread optimizations for homogeneous symmetric and of one order of magnitude for inhomogeneous symmetric systems.

The proposed optimizations are independent from a number of individual refractive indice or the size of homogeneous regions in the aperture and thereby cannot outperform the high-speed WPM presented in [18]. The proposed optimizations apply to all types of homogeneous symmetric or inhomogeneous symmetric systems.

Disclosures The author declares no conflicts of interest.

References

1. M. Feit, J. Fleck, *Light propagation in graded index fibers*, Appl. Opt., vol. 24, pp. 3390-3998, 1978.
2. Sharma, Anurag & Agrawal, Arti, *Non-paraxial Split-step Finite-difference Method for Beam Propagation*, Optical and Quantum Electronics, 38. 19-34. 10.1007/s11082-006-0019-4, Feb 2006.
3. G.R.Hadley, *Wide-Angle beam propagation method using Padé approximant operators*, Opt Lett, Vol 17, No. 20, pp 1426-1428, 1992.
4. Changbao Ma and Edward Van Keuren, *A three-dimensional wide-angle BPM for optical waveguide structures* in OPTICS EXPRESS, (22 January 2007), Vol. 15, No2.
5. P. Lee, E. Vogoes, *Three-dimensional semi vectorial wide-angle beam propagation method*, J. Lightwave Tech., vol. 12, no. 2, pp. 215-225, Feb. 1994.
6. T. Anada, T. Hokazono, T. Hiraoka, J. Hsu, T. Benson, P. Sewell, *Very-wide-angle beam propagation methods for integraded optical circuits.*, IEICE Trans Electron, Vol E82-C, No. 7, pp. 1154-1158, 1999.
7. Junji Yamauchi, Yuta Nito and Hisamatsu Nakano, *A modified semivectorial beam propagation method retaining the longitudinal field component* in Integrated Photonics and Nanophotonics Research and Applications, (Optical Society of America, 2008), paper IWB5.
8. P. Liu and B.J. Li, *Semivectorial beam-propagation method for analyzing polarized modes of rib waveguide.*, IEEE J. Euantum Electron, Vol. 28, pp 778-782, April 1992.
9. J. Wanguemert-Perez, I. Molina-Fernandez, *A novel Fourier based 3D full-vectorial beam propagation method*, Optical Quantum Electronics, vol. 36, pp. 285-301, Kluwer Academic Publishers 2004, Netherlands.
10. Y. Tsuji, M. Koshiba, N. Takimoto, *Finite element beam propagation method for anisotropic optical waveguides.*, Journal Lightwave Technology, Vol. 17, No. 4, pp.723-828, Apr. 1999.
11. M. Stern, *Semivectorial polarized finite difference method for optical waveguides with arbitrary index profiles*, IEEE Proceedings, Vol. 135, Pt.J, No 1, Feb 1988.
12. H. Pinheiro, A. Barbero, H. Hernandez-Figueroa, *Full-vectorial FE-BPM approach for the analysis of anisotropic medium with off-diagonal permittivity terms*, Mic. Opt. Tech. Letters, V.25, No 1, pp. 12-14, April 2000.
13. S.Obayya, B. Rahman, *New vectorial numerically efficient propagation algorithm based on the finite element method.*, IEEE Journal Lightwave Technology, Vol. 18, No. 3, pp. 409-415, Mar 2000.
14. P. Sewell, J. Wykes, A. Vukovic, D. W. P. Thomas, T.M.Benson, C. Christopoulos *Multi-grid interface in computational electromagnetics*, Elec. Lett. Vol.40 No.3, pp 162-163, 2004.
15. K.-H. Brenner and W. Singer, *Light propagation through microlens: a new simulation method* in Applied Optics 32, (1993), 4984-4988
16. M. Fertig and K.-H. Brenner, *Vector wave propagation method*, Journal of the Optical Society of America (JOSA) A, vol. 27, pp. 709-717, Apr 2010
17. Fertig, M.W. (2011) *Vector Wave Propagation Method, Ein Beitrag Zum Elektromagnetischen Optikrechnen.*, Dissertation, Mannheim, Universitt Mannheim. https://madoc.bib.uni-mannheim.de/29233/.
18. K.-H. Brenner, *A high-speed version of the wave propagation method applied to micro-optics*, 16th Workshop on Information Optics (WIO), IEEE Xplore Digital Libary, Interlaken, Schweiz, (2017)
19. http://www.lighttrans.com

APPENDIX

Implemementation of the scalar 'Wave Propagation Method'

```
for(int q=0; q<ny; q++) {                                   //spatial frequency (y)
  for(int p=0; p<nx; p++) {                                 //spatial frequency (x)
    for(int j=0; j<ny; j++) {                               //space frequency (y)
      for(int i=0; i<nx; i++) {                             //space frequency (x)
        kxy_sq = pow(KX[p],2)+pow(KY[q],2);                 //frequency-dependent
        nik0_sq = pow(N[i,j,k]*k0,2);                       //space-dependent
        ntk0_sq = pow(N[i,j,k+1]*k0,2);                     //space-dependent
        if((kxy_sq>=nik0_sq)||(kxy_sq>=ntk0_sq)) continue;  //skip evanescent modes
        kzt=one*sqrt(ntk0_sq-kxy_sq);                       //space- & frequency-dependent
        if( ni_ave!=nt_ave ){
          kzi=one*sqrt(nik0_sq-kx_sq);
          E[i,j,k+1] = 2*kzi*kzt/(kzi+kzt) * e[i,j,k] * exp(ione*kzt*dz);
        } else {
          E[i,j,k+1] = e[i,j,k] * exp(ione*kzt*dz);
} } } }
```

Listing 1. Base algorithm of the three-dimensional scalar Wave Propagation Method.

Implementation of 'gridAnalysis'

```
const int nxhalf = int(ceil(double(nx/2))), nyhalf = int(ceil(double(ny/2)));
int k_offset = (k+1)*nxy;
double ni_ave = N[k_offset-nxy].real(), nt_ave = N[k_offset].real();
is_homog_i = is_homog_t; is_homog_t = true;
is_symmetric_i = is_symmetric_t; is_symmetric_t = true;
for( int iy=0; iy<nyhalf; iy++){
  j_offset01 = k_offset+j*nx;
  j_offset23 = k_offset+(ny-1-j)*nx;
  for( int ix=0; i<nxhalf; ix++){
    ij_idx0 = j_offset01+i; ij_idx1 = j_offset01+nx-1-i;
    ij_idx2 = j_offset23+i; ij_idx3 = j_offset23+nx-1-i;
    if( N[ij_idx0].real()!=N[ij_idx1].real() ||
        N[ij_idx0].real()!=N[ij_idx2].real() ||
        N[ij_idx0].real()!=N[ij_idx3].real() ){
        is_symmetric_t = false; is_homog_t = false; break;
  } else
    if( nt_ave!=N[ij_idx0].real() ) is_homog_t = false;
    if(0==k)
    if( N[ij_idx0-nxy].real()!=N[ij_idx1-nxy].real() ||
        N[ij_idx0-nxy].real()!=N[ij_idx2-nxy].real() ||
        N[ij_idx0-nxy].real()!=N[ij_idx3-nxy].real() ){
        is_symmetric_i = false; is_homog_i = false; break;
    } else
      if( ni_ave!=N[ij_idx0-nxy].real() ) is_homog_i = false;
  } // ix
    if( (0!=k) && (false == is_symmetric_t) ) break;
} // iy
```

Listing 2. 'gridAnalysis' for a two-dimensional aperture.

Implementation of 'HomogOpt'

```
if( is_homog_i && is_homog_t ){
  nt_ave_k0 = nt_ave*k0;
  nt_ave_k0_sq = pow(nt_ave_k0,2);
  if( ni_ave==nt_ave ){ ni_ave_k0 = nt_ave_k0;
                        ni_ave_k0_sq = nt_ave_k0_sq; }
  else { ni_ave_k0 = ni_ave*k0;
         ni_ave_k0_sq = pow(ni_ave_k0,2); }
  for(int q=0; q<ny; q++) {
    q_offset = q*nx;
    for(int p=0; p<nx; p++) {
      pq_idx = q_offset+p;
      if((0==p) && (0==q)) kzt = nt_ave_k0;
      else { kxy_sq = pow(KX[p].real(),2) + pow(KY[q].real(),2);
             tmp = nt_ave_k0_sq-kxy_sq;
             is_evanescent_t = tmp<=0;
             if(do_evanescent_modes || !is_evanescent_t)
               kzt = ( is_evanescent_t ) ? ione*sqrt( -tmp ) : one*sqrt( tmp );
             else { e[pq_idx] = zero; continue; } }
      if( ni_ave==nt_ave ) e[pq_idx] = e[pq_idx] * exp(idz*kzt);
      else { tmp = ni_ave_k0_sq-kxy_sq;
             is_evanescent_i = tmp<=0;
             if(do_evanescent_modes || !is_evanescent_i){
               kzi = ( is_evanescent_i ) ? ione*sqrt( -tmp ) : one*sqrt( tmp );
               fresnel = (is_TE) ? fresnelTTE(kzi, kzt) : fresnelTTM(ni_ave,nt_ave,kzi,kzt);
               e[pq_idx] = ( fresnel * e[pq_idx] ) * exp(idz*kzt);
             } else { e[pq_idx] = zero; continue; } }
} }
```

Listing 3. 'HomogOpt' optimization for a two-dimensional aperture.

Implementation of 'FreqOpt'

```
for(int q=0; q<=nyhalf; q++) {
  q_offset01 = q*nx; q_offset23 = (ny-q)*nx;
  for(int p=0; p<=nxhalf; p++) {
    pq_idx0 = q_offset01+p; pq_idx1 = q_offset01+(nx-p);
    pq_idx2 = q_offset23+p; pq_idx3 = q_offset23+(nx-p);
    is_center = ( (0<q) && (nyhalf>q) && (0<p) && (nxhalf>p) );
    is_bound_x = (0!=p && nxhalf!=p);
    is_bound_y = (0!=q && nyhalf!=q);
```

```
    if((0==p) && (0==q)) kzt = nt_ave_k0;
    else { kxy_sq = pow(KX[p].real(),2) + pow(KY[q].real(),2);
           tmp = nt_ave_k0_sq-kxy_sq;
           is_evanescent_t = tmp<=0;
           if(do_evanescent_modes || !is_evanescent_t)
             kzt = ( is_evanescent_t ) ? ione*sqrt( -tmp ) : one*sqrt( tmp );
           else { if( is_center ){ e[pq_idx0] = zero; e[pq_idx1] = zero;
                                   e[pq_idx2] = zero; e[pq_idx3] = zero; }
                  else { e[pq_idx0] = zero;
                         if( is_outer_p ) e[pq_idx1] = zero;
                         if( is_outer_q ) e[pq_idx2] = zero; }
                  continue; } }
    if( ni_ave==nt_ave ) ctmp = exp(idz*kzt);
    else { tmp = ni_ave_k0_sq-kxy_sq;
           is_evanescent_i = (tmp<=0);
           if(do_evanescent_modes || !is_evanescent_i){
             kzi = ( is_evanescent_i ) ? ione*sqrt( -tmp ) : one*sqrt( tmp );
             ctmp = ( is_TE ) ? fresnelTTE(kzi, kzt) *  exp(idz*kzt) :
                                fresnelTTM(ni_ave, nt_ave, kzi, kzt) *  exp(idz*kzt); }
           else { if( is_center ){ e[pq_idx0] = zero; e[pq_idx1] = zero;
                                   e[pq_idx2] = zero; e[pq_idx3] = zero; }
                  else { e[pq_idx0] = zero;
                         if( is_bound_x ) e[pq_idx1] = zero;
                         if( is_bound_y ) e[pq_idx2] = zero; }
                continue; } }
    if( is_inner_pq ){ e[pq_idx0] = e[pq_idx0] * ctmp; e[pq_idx1] = e[pq_idx1] * ctmp;
                       e[pq_idx2] = e[pq_idx2] * ctmp; e[pq_idx3] = e[pq_idx3] * ctmp; }
    else { e[pq_idx0] = e[pq_idx0] * ctmp;
           if( is_bound_x ) e[pq_idx1] = e[pq_idx1] * ctmp;
           if( is_bound_y ) e[pq_idx2] = e[pq_idx2] * ctmp; }
} } // p,q (freq)
```

Listing 4. Implementation of 'FreqOpt' for a two-dimensional spatial frequency vector.

Implementation of 'SpatOpt'

```
if( is_symmetric_i && is_symmetric_t ) {
 for(int q=0; q<=nyhalf; q++){
   for(int p=0; p<=nxhalf; p++){
     // define pq_idx0..3 same as in 'FreqOpt' listing
     sign = ((p+q)%2) ? -1:1;
     for(int j=0; j<nyhalf; j++) {
       j_offset01 = k_offset+j*nx; j_offset23 = k_offset+(ny-1-j)*nx;
       for(int i=0; i<nxhalf; i++) {
         ij_idx0 = j_offset01+i; ij_idx1 = j_offset01+nx-1-i;
         ij_idx2 = j_offset23+i; ij_idx3 = j_offset23+nx-1-i;
         ntk0 = N[ij_idx0]*k0; nik0 = N[ij_idx0-nxy]*k0;
         if((0==p) && (0==q)) { kzi = nik0; kzt = ntk0;
                                is_evanescent_i = false; is_evanescent_t = false; }
         else { kxy_sq = pow(KX[p],2) + pow(KY[q],2);
                nik0_sq = pow(nik0,2);
                is_evanescent_i = (kxy_sq >= nik0_sq) ? true : false;
                if( is_evanescent_i && !do_evanescent_modes ) continue;
                ntk0_sq = pow(ntk0,2);
                is_evanescent_t = (kxy_sq >= ntk0_sq) ? true : false;
                if( is_evanescent_t && !do_evanescent_modes ) continue;
                kzi = ( is_evanescent_i ) ? ione*sqrt( kxy_sq - nik0_sq ) :
                                            one*sqrt( nik0_sq - kxy_sq );
                kzt = ( is_evanescent_t ) ? ione*sqrt( kxy_sq - ntk0_sq ) :
                                            one*sqrt( ntk0_sq - kxy_sq ); }
       if( nik0==ntk0 ) ctmp = sign/double(nxy)*exp(ione*(kzt*dz-ntk0.imag()*dz));
       else ctmp = fresnelTTE(kzi,kzt)*sign/double(nxy)*exp(ione*(kzt*dz-ntk0.imag()*dz));
   if( is_inner_pq ){
     E[ij_idx0] = E[ij_idx0] + ctmp * (
          e[pq_idx0] * exp(ione*( KX[p]   *X[i] + KY[q]   *Y[j] )) +
          e[pq_idx1] * exp(ione*( KX[nx-p]*X[i] + KY[q]   *Y[j] )) +
          e[pq_idx2] * exp(ione*( KX[p]   *X[i] + KY[ny-q]*Y[j] )) +
          e[pq_idx3] * exp(ione*( KX[nx-p]*X[i] + KY[ny-q]*Y[j] )) );
        E[ij_idx1] = E[ij_idx1] + ctmp * (
          e[pq_idx0] * exp(ione*( KX[p]   *X[nx-1-i] + KY[q]   *Y[j] )) +
          e[pq_idx1] * exp(ione*( KX[nx-p]*X[nx-1-i] + KY[q]   *Y[j] )) +
          e[pq_idx2] * exp(ione*( KX[p]   *X[nx-1-i] + KY[ny-q]*Y[j] )) +
          e[pq_idx3] * exp(ione*( KX[nx-p]*X[nx-1-i] + KY[ny-q]*Y[j] )) );
```

```
37          E[ij_idx2] = E[ij_idx2] + ctmp * (
38            e[pq_idx0] * exp(ione*( KX[p]   *X[i] + KY[q]   *Y[ny-1-j] )) +
39            e[pq_idx1] * exp(ione*( KX[nx-p]*X[i] + KY[q]   *Y[ny-1-j] )) +
40            e[pq_idx2] * exp(ione*( KX[p]   *X[i] + KY[ny-q]*Y[ny-1-j] )) +
41            e[pq_idx3] * exp(ione*( KX[nx-p]*X[i] + KY[ny-q]*Y[ny-1-j] )) );
42          E[ij_idx3] = E[ij_idx3] + ctmp * (
43            e[pq_idx0] * exp(ione*( KX[p]   *X[nx-1-i] + KY[q]   *Y[ny-1-j] )) +
44            e[pq_idx1] * exp(ione*( KX[nx-p]*X[nx-1-i] + KY[q]   *Y[ny-1-j] )) +
45            e[pq_idx2] * exp(ione*( KX[p]   *X[nx-1-i] + KY[ny-q]*Y[ny-1-j] )) +
46            e[pq_idx3] * exp(ione*( KX[nx-p]*X[nx-1-i] + KY[ny-q]*Y[ny-1-j] )) ); }
47      else {
48        if( is_equal_pq ){
49          E[ij_idx0] = E[ij_idx0] + ctmp * (
50            e[pq_idx0] * exp(ione*( KX[p] * X[i] + KY[q]*Y[j] )) );
51            E[ij_idx1] = E[ij_idx1] + ctmp * (
52              e[pq_idx0] * exp(ione*( KX[p] * X[nx-1-i] + KY[q]*Y[j] )) );
53            E[ij_idx2] = E[ij_idx2] + ctmp * (
54              e[pq_idx0] * exp(ione*( KX[p] * X[i] + KY[q]*Y[ny-1-j] )) );
55            E[ij_idx3] = E[ij_idx3] + ctmp * (
56              e[pq_idx0] * exp(ione*( KX[p] * X[nx-1-i] + KY[q]*Y[ny-1-j] )) ); }
57          else {
58            if( is_outer_p ){
59              E[ij_idx0] = E[ij_idx0] + ctmp * (
60                e[pq_idx0] * exp(ione*( KX[p]   *X[i] + KY[q]   *Y[j] )) +
61                e[pq_idx2] * exp(ione*( KX[p]   *X[i] + KY[ny-q]*Y[j] )) );
62              E[ij_idx1] = E[ij_idx1] + ctmp * (
63                e[pq_idx0] * exp(ione*( KX[p]   *X[nx-1-i] + KY[q]   *Y[j] )) +
64                e[pq_idx2] * exp(ione*( KX[p]   *X[nx-1-i] + KY[ny-q]*Y[j] )) );
65              E[ij_idx2] = E[ij_idx2] + ctmp * (
66                e[pq_idx0] * exp(ione*( KX[p]   *X[i] + KY[q]   *Y[ny-1-j] )) +
67                e[pq_idx2] * exp(ione*( KX[p]   *X[i] + KY[ny-q]*Y[ny-1-j] )) );
68              E[ij_idx3] = E[ij_idx3] + ctmp * (
69                e[pq_idx0] * exp(ione*( KX[p]   *X[nx-1-i] + KY[q]   *Y[ny-1-j] )) +
70                e[pq_idx2] * exp(ione*( KX[p]   *X[nx-1-i] + KY[ny-q]*Y[ny-1-j] )) ); }
71            if( is_outer_q ){
72              E[ij_idx0] = E[ij_idx0] + ctmp * (
73                e[pq_idx0] * exp(ione*( KX[p]   *X[i] + KY[q]*Y[j] )) +
74                e[pq_idx1] * exp(ione*( KX[nx-p]*X[i] + KY[q]*Y[j] )) );
75              E[ij_idx1] = E[ij_idx1] + ctmp * (
76                e[pq_idx0] * exp(ione*( KX[p]   *X[nx-1-i] + KY[q]*Y[j] )) +
77                e[pq_idx1] * exp(ione*( KX[nx-p]*X[nx-1-i] + KY[q]*Y[j] )) );
78              E[ij_idx2] = E[ij_idx2] + ctmp * (
79                e[pq_idx0] * exp(ione*( KX[p]   *X[i] + KY[q]*Y[ny-1-j] )) +
80                e[pq_idx1] * exp(ione*( KX[nx-p]*X[i] + KY[q]*Y[ny-1-j] )) );
81              E[ij_idx3] = E[ij_idx3] + ctmp * (
82                e[pq_idx0] * exp(ione*( KX[p]   *X[nx-1-i] + KY[q]*Y[ny-1-j] )) +
83                e[pq_idx1] * exp(ione*( KX[nx-p]*X[nx-1-i] + KY[q]*Y[ny-1-j] )) ); } } }
```

Listing 5. Implementation of 'SpatOpt' for two-dimensional spatial refractive index symmetries.