

The not-not so short Introduction to **LaOndaGA**
-The Beta Manual-

A. Håkansson

October 3, 2007

Chapter 1

Introduction

LaOndaGA is a design program for photonic device design. The program is based on an inverse design approach including a fusion of a direct EM solver, The Multiple Scattering Theory (MST) [1], and a stochastic optimization algorithm, The Genetic Algorithm (GA) [2, 3, 4]. The program uses the MST to solve a direct two dimensional (2D) EM scattering problem and the GA to find an optimal distribution of scatterers for a chosen functionality. This program has been the base of a number of publications in both photonics [5, 6, 7, 8, 9, 10, 11], acoustics [12, 13, 14] and ultrasound [15]. A more detailed description of the theory as well as a full list of references can be found in my thesis which can be downloaded from here: <http://www.box.net/shared/10bxufytz2>

The program is coded in Fortran 90 and is run from the command line in windows. I am sorry but so far there is no GUI. I was working on it for a while in C# but I never had time to finish it. The output and input is handled through text based ASCII files. To map the resulting scattered EM field from the device I've been using Origin, however any other similar program should be OK.

Chapter 2

Files

Input files;

- coord.dat
- parameters.dat
- gap.dat
- scatterer.dat

Output files (the ones you need);

- gamax.dat
- bestind01.dat
- cristal.dat
- campo.dat

2.1 Input-files descriptions

2.1.1 coord.dat

This file contains the position of the fixed lattice sites (LS) used in the optimization process.

Example: coord.dat						
ln	col1	col2	col3	col4	col5	col6
1	0	5.25	0.3	1	0	0
2	0	2.1	0.3	1	0	0
3	0	1.05	0.3	1	0	0
4	0	3.15	0.3	1	0	0
5	0	4.2	0.3	1	0	0
6	0.90933	0.525	0.3	0	0	0
7	0.90933	1.575	0.3	1	0	0
8	0.90933	4.725	0.3	1	0	0
9	0.90933	2.625	0.3	1	0	0
10	0.90933	3.675	0.3	1	0	0
11	0.90933	5.775	0.3	1	0	0
12	1.81866	0	0.3	0	0	0
13	1.81866	4.2	0.3	1	0	0
14	1.81866	5.25	0.3	1	0	0
15	1.81866	2.1	0.3	1	0	0
16	1.81866	1.05	0.3	1	0	0
17	1.81866	3.15	0.3	1	0	0

There are two ways of introduce these; using 3 or 6 parameters for each site. which one of the two input methods is used is set by a binary parameter in the file "paramters.dat", ln20 "use advance coordinates".

The 3 parameters input are real values and correspond to the x, the y and the radius of the scatterer, i.e. x—y—radius. These values should be placed on one line separated with a white space.

The 6 parameter input is first the three expressed in the previous section, i.e., x—y—radius and then three extra; sym—imx—imy. The first is an integer and the last are represented by two real values. The most important of the three is the integer which is used for symmetric cluster simulations. The last two real parameters, imx and imy, are to place the scatterer in the complex plain in order to implement PML-layers (see [16]).

sym=0 This is to indicate that the symmetry does not apply for this LS. This is correct if the LS i placed on the symmetry axis, i.e. $y = 0$, or if we deal with a non-symmetric system.

sym=1 sym=1: This marks the LS as a symmetric-LS; The scatterer is automatically mirrored in the symmetry plane (with the coordinates $x, -y$) in the calculation. It is important to only define a symmetry pair with one LS. If both coordinates where specified, i.e. x, y and $x, -y$ in the file coord.dat the program gives an erroneus result!

About Symmetric Clusters

To speed up the simulation considerable (approx. 4 times), it is a good idea to use the implemented symmetry condition (this is of coarse if you have a symmetric system... otherwise it might not be such a good idea). The symmetry axis for the system is set with

respect to $y = 0$. Each lattice site in the file `coord.dat`, which is positioned off this axis, must be given a “sym” value equal 1. Please notice that the incident wave also has to be symmetric with respect to this axis.

2.1.2 parameters.dat

This file contains all the parameters related to the direct solver, including incident beam, materials, convergence parameters, wavelength, ...

Example: parameters.dat

ln	value	description
1	F	***[T] Write to file?
2	0	***[0] (GA) fitness
3	1.0000 12.1	*** eps1(bg), eps2(scatt)
4	3	***[3] 1=TM, 2=TE, 3=NO POL
5		
6	T	***[T] Inc. incident wave
7	0	***[0] RandLevel; Alpha; Rep
8	0 0.000	***[0 0.0] SourceType
9		
10	2	***[3] qlim
11	1	***[1] Tmatrix
12	0.1	*** max radio
13	0.5500	*** lattice parameter
14	0.0100	***[0.010] delta-x (intensity)
15	0.005	***[0.005] h (gaussian beam)
16		
17	0	***[0] Units
18	0	***[0] (GA) crystalcode
19	T	***[F] adv. coordinates
20	1	***[1] grid index
21	20.000 20.000 0.100	***xmin, xmax, xstep
22	0.0000 0.0000 0.100	***ymin, ymax, ystep
23	3	***[1] wave index
24	1.5500 1.5500 1.000	*** fmin, fmax, fstep

ln1 Write to file? (bool) This parameter is set if you want the program to save the complete output to files. This parameter should be set to `false` when you do the optimization process, and to `true` when you want to visualize the result.

ln2 fitness (int) This value sets the fitness-function. There are many ways to calculate the fitness, and they can get quite complicated when including many frequencies and field points. However the basic Fitness indexes are;

0 When you want to maximize any of the specified field points. This is normally used for designing lens devices.

- 1** This maximize the sum of all specified field points.
- ln3 eps** (**real real**) These parameters set the dielectric constant for the background and the scatterers, respectively.
- ln4 polarization** (**int**) Here you specify if you're interested in only a specific polarization, or both;
- 1** TM-polarization, in plane magnetic field
 - 2** TE-polarization, in plane electric field
 - 3** Unpolarized, calculates both polarizations
- ln6 Include inc. wave?** (**bool**) Specifies whether the final field calculation should include the incident wave. This parameter should almost always be set to "T".
- ln7 randlevel** (**int real int**) This parameter is used for robustness calculations by including some random errors in the position and shape of the scatterers.
- 0 - -** No random error! Here there is no need to specify the second and third value since these values are used to control the size of the error.
- ln8 sourcetype** (**int real real**) The first integer value set the type of the source, the following real parameters is related to that specific source;
- (0 0.0 -)** This is a plane wave (int value = 0), propagating in the positive x -direction with a 0° tilt (real parameter = 0.0). The plane wave only has one parameter hence the second real value is obsolete.
 - (1 0.0 0.0)** This is a point source (int value = 1) placed at the origin of the coordinate system; ($x = 0.0, y = 0.0$) (real parameters = 0.0, 0.0).
 - (2 10.0 -)** This specifies a Gaussian shaped beam (int value = 2) with the width $10\mu\text{m}$ set by the following real parameter. The beam is always propagating in the positive x -direction. The second real parameter is obsolete.
- ln10 qlim** (**int**) The **qlim** parameter specifies the highest order of Bessel-function used to represent the scattered field in the background media. The convergence limit is highly dependent on the ratio between the radius of the scatterer each scatterer and the wavelength. Though, if the wavelength is about twice the length of the width of the scatterer (4 times the radius) a value of 3 is sufficient, even 2 might be enough for convergence. The CPU time varies a lot with this number, so it is a good thing to try to keep it as low as possible during the design process. Of course when analyzing the designed device this value can be increased to check the validity of the process.
- ln11 T-matrix** (**int**) Due to symmetry conditions, there are various ways of calculating the transition matrix of a scatterer. If the scatterer has a circular cross-section the boundary condition can be expressed with a

very simple equation. for more complex systems Waterman derived an integral equation [17] to calculate the different elements of the matrix.

1 This indicates that the simplified equation can be used since the scatterers have circular cross-section.

2 If the scatterers are not circular the value has to be set to 2.

IMPORTANT In addition the file “scatterer.dat” must be completed to specify the actual shape of the atoms.

ln12 max-radio This value marks the minimum radius that includes the largest possible scatterer in the device. This value is only used for checking whether there is a possible overlap between scatterers. This value is not used in the calculation.

ln13 lattice-parameter Self explaining, i.e. the lattice parameter used in the lattice. This value is used for checking whether there is a possible overlap between scatterers. This value is only used in the calculation for estimating the relative frequency when defined in arbitrary units of $a \lambda$.

ln14 delta-x Used to calculate numerically the derivative for the Poynting vector.

ln15 h Used for the numerical calculation of the incident beam Gaussian beam distribution.

ln17 units (int) Whether the field points should be expressed using the EM-field or the Poynting vector;

1 Electromagnetic Field

2 Poynting Vector

ln18 crystal-code (int) This parameter is used in the GA optimization for including special lattice sites symmetries.

ln19 adv-coords (bool) If using advanced coordinates (6 parameters) or simple (3 parameters) for defining the lattice sites. For details see 2.1.1.

ln20 grid-idx (int) There are three different ways of defining the grid field points for which the EM-field should be calculated

1 Cartesian coordinate system - (x, y)

2 Cylindrical coordinate system - (θ, r)

3 Read grid from file (grid.dat) ..I never used this option so I don't really know whether it's working.

ln21 x θ -min-max-step (real real real) These three parameters defines the min, max, and step value of the first parameter of the grid (x or θ depending whether grid-idx is 1 or 2). Normally I do a map with a resolution around 100×100 grid points. For example (-5.0 5.01 0.1) will calculate the EM-field from -5 to +5 with a resolution of 0.1. It is a good idee to set the max variable just a little bit bigger than the last value. This is to be sure that this value will be included in the calculation. **The step parameter can not be equal zero!**

ln22 yr-min-max-step (real real real) These three parameters defines the min, max, and step value of the second parameter of the grid (y or r depending whether grid-indx is 1 or 2). **The step parameter can not be equal zero!**

ln23 wave-indx (int) The wave index defines how the wavelength of the incident wave should be expressed

- 1 Frequency in absolute units
- 2 Frequency in arbitrary units, a/λ
- 3 The free-space wavelength in absolute units

ln24 f_r -min-max-step (real real real) The Maxwell's equations are solved in k-space, hence a normal run of the program will include only one frequency. In this case the first two variables must be identical and will define the wavelength in whatever units was set in "wave-indx". Once again it is important that the step variable is set not-equal to zero. It is however also possible to do multi-frequency optimization. For these calculations, similar to the grid points definition, the parameters defines, min, max and the frequency step, respectively.

2.1.3 gap.dat

This file includes all parameter related to the optimization process.

ln	value	description
1	1	***[1] calcval 1=GA; 2=GAview, ...
2	T	***[F] continue old run?
3	1	***[1] seed (uint for Win32; real for Linux)
4	0	***[0] number of fixed scatterers
5	81	*** number of bits
6	1	***[1] bits per gene
7	100	*** number of individuals (>=2)
8	1	***[1] size of mult struct opt. (random)
9	0.02000	*** mutation op
10	6	***[1] crossType
11	0.80	***[0.50] crossover op
12	1	***[1] selectionType: 1-Tournament, 2-Rank
13	2	***[2] selection par
14	T	***[T] elitism (only option implemented)
15	F	***[F] niche
16	1	***[1] sigma (iff niche==T)
17	1000	***[1000] number of max generations
18	1	***[1] number of children

ln1 calcval (int) This first value set the operation state of the program; optimizing or visualizing

- 1 Run an inverse design using the GA-optimization code
 - 2 Calculate the scattered field from the structure. This mode is set after running the optimization in order to calculate the scattered EM-field of the device.
 - 3 ???
 - 4 This value calculates all possible configurations of the bit parameters. This option is only valid for a low number of bit parameters. The number of possible configurations scale with 2^N , where N is the number of bits.
 - 5 Random-walk search
 - 6 Run an inverse design using a Hill-Climbing optimization algorithm. This algorithm takes a one bit step for each generation controlled by the highest increase in fitness value.
- ln2 continue** (bool) After each generation of the GA-optimization the present state of the search is saved to 3 files (`bestind01.dat`, `restart.dat`, `gamax.dat`). If the search is stopped before converging this parameter can be used with these three files to restart the search from its present state.
- ln3 seed** (real) This value is the seed value for the random generate algorithm. Different values generate different number sequences.
- ln4 nr-of-fixed-scatt** (int) Here you have to define any fixed scatterer which will not be included in the design process. Normally this value is set to 0. These coordinates need to be placed last in the `coord.dat`.
- ln5 no-of-bits** (int) This integer parameter sets the size of the optimization problem, i.e. the number of binary parameters included in the inverse design process. In the normal approach, each lattice-site defined in the file `coord.dat` corresponds to one bit parameter.
- ln6 bits-per-gene** (int) In the normal design process each gene is only one bit coding the presence or absence of a scatterer in a fixed lattice site. However using different `crystal-code` parameters in `parameters.dat`, it is possible to increase the design freedom and, for example, include different sized scatterers.
- ln7 population** (int) The convergence as well as the result of the GA optimization is strongly dependent on the size of the GA population. A bigger population will result in a slow convergence but a better solution will be found with higher probability. As a general rule try to keep the population as big as possible with respect to the design time, however it should be at least as big as the `no-of-bits` parameter. **This value has to be equal and greater than 2.**
- ln8** ???
- ln9 mutation** (real) One of the three GA operators. The uniform Mutation operator is basically a random walk which is included to maintain a diversity of the population. This value should be considerably small. A

good indication is: $1/\text{no-of-bits}$. This value together with the **selection** operator will include the standard deviation (s.d.) of the converged population. A good s.d. value should be around 3-8. If the population does not converge this value can be decreased in order to increase the pressure.

ln10 crossover-type (int) The second GA operator is the Crossover. This operator is here implemented in different ways. This parameter set which type of Crossover is used in the GA optimization.

- 1 uniform
- 2 one-point
- 3 two-point
- 4 Z3, 2D-crossover with 9 fields
- 5 Z3, 2D-crossover with 16 fields
- 6 geographical crossover with $k=4$

ln11 crossover-op (real) This gives the probability of the use of the crossover operator.

ln12 selection-type (int) The third GA operator is the Selection. This parameter set which type of Selection is used in the GA optimization. Tournament selection is strongly recommended.

- 1 tournament selection
- 2 rank selection

ln13 selection-par (int) This parameter is used with **tournament selection** to set the number of individuals competing in the selection process. This value can be used to increase the pressure of convergence. The higher the value the faster the population will converge.

ln14 elitism (bool) Setting this value to T will guarantee that the best solution will always be copied into the next generation and not get lost using crossover.

ln15 niche (bool) This parameters is set when you want to include niches in the optimization population. This parameter is almost always set to F.

ln16 sigma (int) This value is only used when **niche** is set to true.

ln17 number-of-max-gen (int) This value marks the maximum number of generation that will be used. After this number of iteration the search will automatically stop. However, the search can always be stopped using Ctrl-C.

ln18 number-of-children (int) The number of children produced by the Crossover can vary. Normally this value is always set equal to 1.

2.1.4 scatterer.dat

This file defined the geometry for scatterers with non-circular cross-section. This file is only read if the T-matrix value in `parameters.dat` is set equal to 2.

This is an example for square shaped cross-section measuring 0.4×0.4 .

Example: scatterer.dat		
ln	value	description
1	4	***index
2	0.2000	***constant x
3	0.2000	***constant y

ln1 index (int) The index mark one of the four geometries for the cross-section.

- 1 Cylinders; This parameter is never used since there is no need for surface integral computation for calculating the elements of the transition matrix.
- 2 Ellipses
- 3 Capsule; The capsule form is to half-circles connected with a rectangle.
- 4 Rectangle

ln2 constant-x (real) This is the constant related to the x-direction of the geometry. In the case of the ellipse geometry this value equals the radius along the x-axis, and in for the rectangle geometry the half of the side along the x-axis.

ln3 constant-y (real) This is an identical parameter as `constant-x`, but related to the y-axis instead.

2.2 Output-files descriptions

2.2.1 gamax.dat

`gamax.dat` is a record of the GA optimization process. After each generation one line is added at the end of the file writing out: (**Generation**) which generation; (**Calls**) the total number of configurations calculated, i.e. the generation times the population-size; (**Maxvalue**) the fitness value of the highest quality device; (**Meanvalue**) the mean fitness-value of the population; (**Stdv**) the standard deviation of the fitness-value of the population; (**Fitjump**) the number of bits that was changed when a better solution was found, with respect to the best solution in the previous generation.

Example: gamax.dat					
col1	col1	col2	col3	col4	col5
Generation	Calls	Maxvalue	Meanvalue	Stdv	FitJump
1	100	19.33317	4.84266	41.17835	0.
2	200	19.83303	5.16557	40.37714	15.
3	300	19.83303	4.98518	39.43835	0.
4	400	19.83303	5.28041	39.88494	0.
5	500	19.83303	5.52839	39.83666	0.
6	600	20.34283	5.97563	37.38160	38.
7	700	20.81038	5.99299	38.66510	47.
8	800	20.81038	5.31041	37.77699	0.
...					

This is an example of a run using a Population size of 100 and 81 bit-parameters. The 81 bits can be confirmed looking at the Stdv of Generation 1, since the initial population is randomized generated this value should equal the half size of the total number of bits, here equal 40.5, i.e. if comparing two bits from two solutions there is a 50% chance that they are different, $0.5 * 81 = 40.5$. If you didn't get this, no worries... it's probably my fault anyway.

2.2.2 bestind01.dat

The parameters of the best solution is stored in bestind01.dat after each generation.

Example: bestind.dat	
81	
0.0000000E+00	
0 1 1 1 1 1 1 0 0 0 1 0 1 1 1 1 0 0 1 1 1 ...	

These lines correspond to;

ln1 no-of-bits, see. 2.1.3

ln2 ???, don't really know what this is...

ln3 The string of parameters of the design with highest fitness value. This string should include no-of-bits bool values. Here it has been cut due to lack of space. Anyway, you get what I mean, right?

2.2.3 cristal.dat

This file shows the optimized design. The file is identical to coord.dat in the sense that it includes all 6 parameters defining the coordinates and size of each scatter.

Example: cristal.dat					
col1	col2	col3	col4	col5	col6
X	Y	r	Index	imX	imY
-1.819	-9.450	0.300	1	0.000	0.000
-1.819	2.100	0.300	1	0.000	0.000
-1.819	-2.100	0.300	1	0.000	0.000
-1.819	7.350	0.300	1	0.000	0.000
-1.819	-7.350	0.300	1	0.000	0.000
-1.819	0.000	0.300	0	0.000	0.000
-1.819	1.050	0.300	1	0.000	0.000
-1.819	-1.050	0.300	1	0.000	0.000
-0.909	4.725	0.300	1	0.000	0.000
-0.909	-4.725	0.300	1	0.000	0.000
-0.909	5.775	0.300	1	0.000	0.000
-0.909	-5.775	0.300	1	0.000	0.000
-0.909	8.925	0.300	1	0.000	0.000
-0.909	-8.925	0.300	1	0.000	0.000
-0.909	3.675	0.300	1	0.000	0.000
-0.909	-3.675	0.300	1	0.000	0.000

2.2.4 campo.dat

From this file you will import the different field point in the area you've specified in `parameters.dat`. There are 9 columns corresponding to; (**x**) The x-coordinate; (**y**) The y-coordinate; (**absTE**) The absolute value of the TE-mode component, i.e. the time average value of the magnetic field; (**absTM**) The absolute value of the TM-mode; (**Mod**) The time average of the total field including both polarizations; (**reTE**) The real value of the TE-mode, i.e. the magnetic field at t_0 ; (**imTE**) The imaginary value of the TE-mode, i.e. the magnetic field at $t_0 + T/4$; (**reTM**) The real value of the TM-mode; (**imTM**) The imaginary value of the TM-mode.

Example: campo.dat

col1	col2	col3	col4	col5	col6	col7	col8	col9
x	y	absTE	absTM	Mod	reTE	imTE	reTM	imTM
20.0	-6.00	0.0	0.16	0.16	0.0	0.0	-0.16	-0.02
20.0	-5.75	0.0	0.11	0.11	0.0	0.0	-0.11	-0.04
20.0	-5.50	0.0	0.09	0.09	0.0	0.0	-0.06	-0.06
20.0	-5.25	0.0	0.09	0.09	0.0	0.0	-0.03	-0.08
20.0	-5.00	0.0	0.09	0.09	0.0	0.0	-0.02	-0.08
20.0	-4.75	0.0	0.07	0.07	0.0	0.0	-0.01	-0.07
20.0	-4.50	0.0	0.03	0.03	0.0	0.0	-0.00	-0.03
20.0	-4.25	0.0	0.03	0.03	0.0	0.0	0.015	0.025
20.0	-4.00	0.0	0.01	0.10	0.0	0.0	0.051	0.092
20.0	-3.75	0.0	0.18	0.18	0.0	0.0	0.105	0.154
20.0	-3.50	0.0	0.26	0.26	0.0	0.0	0.169	0.197

Please notice: The accuracy has been reduced in the example in order to fit the width to the page. In the real output file each value is given with four relevant digits.

Chapter 3

Execution

The program is compiled to be executed in Windows under the command line. You can simply execute it from Explorer after properly filling in the input files. When running the design process, i.e. mode 1 in `gap.dat ln1`, the info about the present state of the optimization process is printed out to the screen after each iteration. The program can simply be stopped using Ctrl-C, no worries the present state of the process is saved after each generation (see. 2.2).

The program can easily be ported to the Linux platform using wine (<http://www.winehq.org/>)!

Bibliography

- [1] Akira Ishimaru. *Electromagnetic Wave Propagation, Radiation, and Scattering*. Prentice Hall, New Jersey, 1991.
- [2] J.H. Holland. *Adaptation in natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, 1975.
- [3] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Learning*. Addison Wesley, Reading , MA, 1989.
- [4] D.E. Goldberg. *The design of Innovation*. Kluwer Academic Publishers, 2002.
- [5] L. Sanchis, A. Håkansson, D. Lopez-Zanón, J. Bravo-Abad, and J. Sánchez-Dehesa. Integrated optical devices design by genetic algorithm. *Appl. Phys. Lett.*, 84:4460–4462, 2004.
- [6] A. Håkansson, J. Sánchez-Dehesa, and L. Sanchis. Inverse design of photonic crystal devices. *IEEE J. Sel. Area Comm.*, 23:1365–1371, 2005.
- [7] A. Håkansson and J. Sánchez-Dehesa. Optimal design of microscaled scattering optical elements. *Appl. Phys. Lett.*, 87:193506–193508, 2005.
- [8] A. Håkansson and J. Sánchez-Dehesa. Inverse designed photonic crystal de-multiplex waveguide coupler. *Opt. Exp.*, 13:5440–5449, 2005.
- [9] A. Håkansson and J. Sánchez-Dehesa. Comment on 'optimization of photonic crystal structures', *j. opt. soc. am. a* 21, 2223-2232 (2004). *arXiv.org*, 2005. cond-mat/0504581.
- [10] A. Håkansson, H. T. Miyazaki, and J. Sánchez-Dehesa. Inverse design for full control of spontaneous emission using light emitting scattering optical elements. *Phys. Rev. Lett.*, 96(153902), 2006.
- [11] A. Håkansson. Cloaking of objects from electromagnetic fields by inverse design of scattering optical elements. *Opt. Exp.*, 15:4328–4334, 2007.
- [12] A. Håkansson, L. Sanchis, and J. Sánchez-Dehesa. Acoustic lens design by genetic algorithms. *Phys. Rev. B*, 70(214302), 2004.
- [13] A. Håkansson, F. Cervera, and J. Sánchez-Dehesa. Sound focusing by flat acoustic lenses without negative refraction. *Appl. Phys. Lett.*, 86:054102–054104, 2005.

- [14] A. Håkansson, J. Sánchez-Dehesa, , and F. Cervera. Experimental realization of sonic demultiplexing devices based on inverse-designed scattering acoustic elements. *Appl. Phys. Lett.*, 88:163506–163508, 2006.
- [15] A. Håkansson, Daniel Torrent, Francisco Cervera, and J. Sánchez-Dehesa. Directional acoustic source by scattering acoustical elements. *Appl. Phys. Lett.*, pages 224107–224109, 2007.
- [16] Davy Pisssoort and Frank Olyslager. Termination of periodic waveguides by pmls in time-harmonic integral equation-like techniques. *IEEE Antennas and Wireless Propagation Letters*, 2:281–284, 2003.
- [17] P.C. Waterman. Symmetry, unitarity, and geometry in electromagnetic scattering. *Phys. Rev. D*, 3(4), 1979.