KALLISTOS

An automatic optimisation tool

CrystalWave Tutorial

34 Leopold St
Oxford, OX4 1TW
United Kingdom
Phone + 44 1865 324998
Fax + 44 1865 324991
info@lambda-tek.com
www.lambda-tek.com

Distributed By
Photon Design
34 Leopold St
Oxford, OX4 1TW
United Kingdom

Phone +44 1865 324 990
Fax +44 1865 324 991
Email: info@photond.com

Chapter 1 Introduction

1.1 What is Kallistos?

KALLISTOS is a novel optimisation tool capable of automatically improving existing designs of photonic devices with minimum intervention from the user. Using KALLISTOS, design cycle times of a new device can be dramatically reduced.

Considerable work has gone into all aspects of the product, resulting in a tool incorporating state of the art optimisation algorithms, combined with a powerful graphical user interface making it easy for the user to set up and run a design optimisation calculation, and a comfortable visual interface which allows the user to fully explore the solutions found.

Under the hood, KALLISTOS is equipped with several robust optimisers for the global and local optimisation of continuous functions. Moreover much work has gone into KALLISTOS to make it as efficient as possible. For example, in calculating the sensitivities used by the local optimiser, KALLISTOS takes full advantage of the special structure of the wave equations describing light propagation through photonic devices.

1.2 How it works

In the first iteration of a product design, the user typically starts with an initial guess of a design, such as a linear bend (see the examples). Selected *parameters* and *constraints* defining the structure (e.g. governing the bend shape) are then declared within KALLISTOS, as well as the quantity that needs to be optimised (the *objective function*) e.g. Power Transmission. Finally the calculation is launched with one of the optimisation algorithms available in KALLISTOS.

1.3 The function parser

Although KALLISTOS comes with a variety of predefined *objective functions*, KALLISTOS also comes with a powerful built-in function parser making it possible to define arbitrarily complex parameter constraints and *objective functions.*

1.4 The optimisation algorithms

Finding solutions to complex optimisation problems can be a difficult task. In general there is no optimisation algorithm that works in all cases. KALLISTOS comes with efficient algorithms based on recent advances in mathematical optimisation, each with complementing strengths:

- An efficient local descent routine, ideal for large, computationally intensive structures. An optimum can be found with relatively few iterations.

- A Deterministic global optimisation techniques. This converges more slowly than the above, but is more likely, if not guaranteed, to find a global optimum.

1.5 The monitoring interface

Far from adopting a black box approach, the software comes with a powerful graphical monitoring interface for following the progress of multidimensional optimisation calculations. These prove to be particularly useful for detecting potentially optimal designs when using a global optimiser. For example, the user can quickly locate a point of interest in the parameter space and "home in" on it with a local optimiser.

1.6 Highly speed optimised

Optimisation calculations can be very lengthy for complex, 3D structures. We have exploited to the full, the mathematical structure of the wave equations, for example, by incorporating an analytical procedure for calculation of the sensitivities, essential to the optimisation process. This and other improvements make KALLISTOS an extremely fast optimisation tool for the design of photonic devices.

Chapter

2

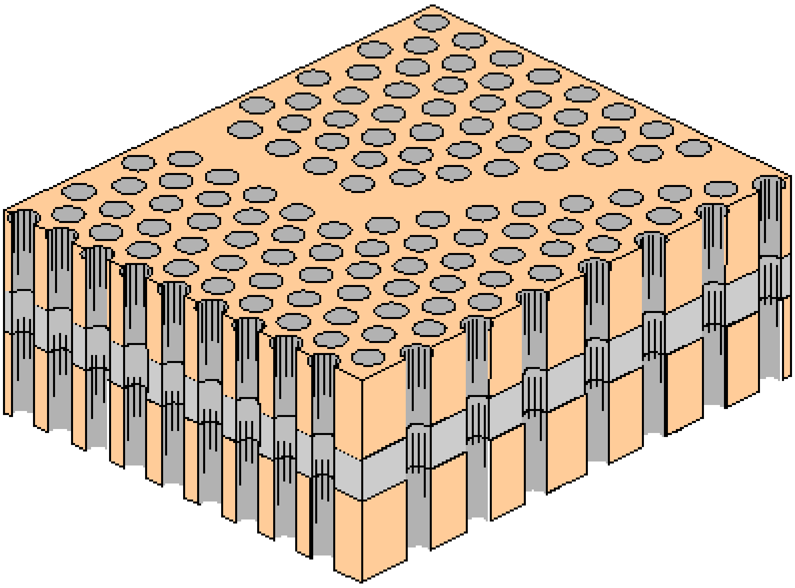
Using Kallistos to optimise the transmission around a 2D photonic crystal bend

This tutorial assumes that you are familiar with CRYSTALWAVE. In the example that follows we will use the project "`Learn_Kallistos_CW`.prj", which contains a ready-made CRYSTALWAVE Device to be optimised. For the completed examples refer to the file "`Complete_Learn_Kallistos_CW`.prj"

2.1 Introducing the problem

We will now use KALLISTOS to minimise the reflection around a 60-degree bend in a 2D photonic crystal slab.

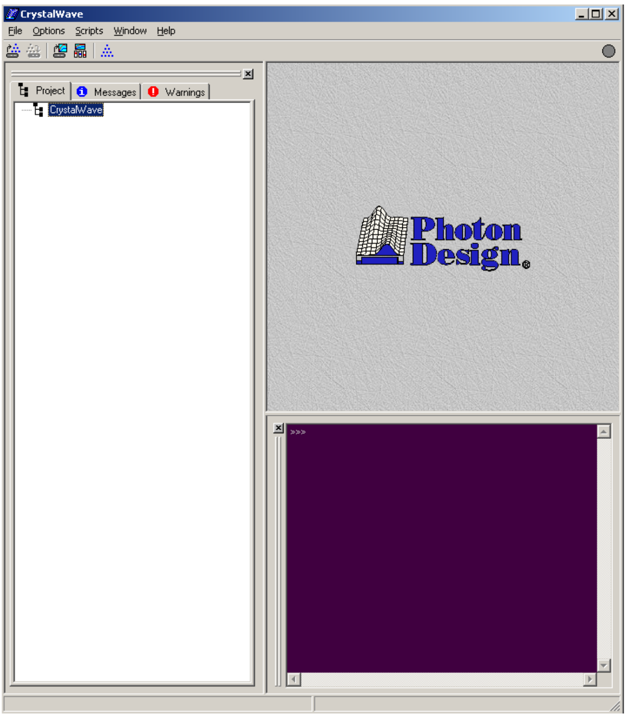
Suppose we have an initial 60-degree bend….



A 60 degree bend in a 3D photonic crystal slab

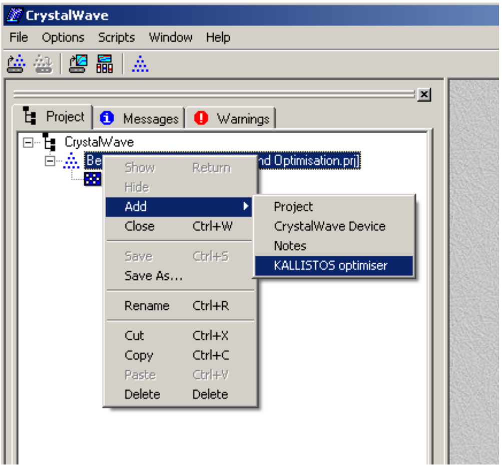
We wish to optimise this bend in order to minimise reflection.

- First open CRYSTALWAVE.



CrystalWave

- ➢ Within CRYSTALWAVE click on the icon and open the project containing the photonic crystal device. (“`Learn_Kallistos_CW.`prj”)
- ➢ Now we need to create an optimiser. Once the project has been opened right click on the icon in the *project tree*:

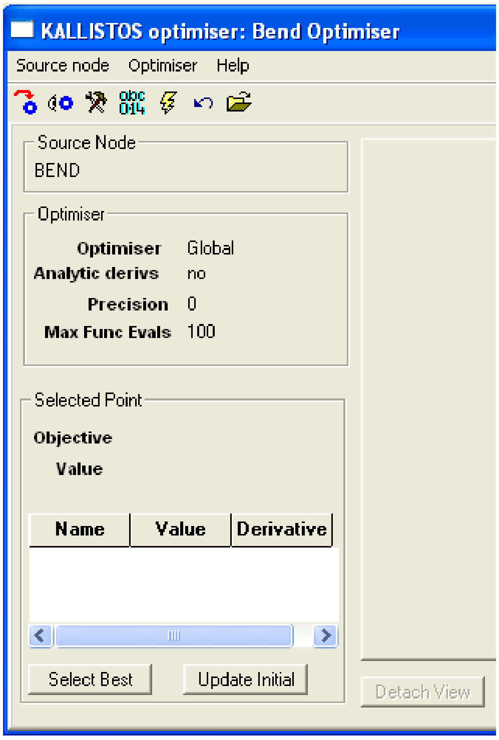


Adding a Kallistos Optimiser to the project

- ➢ In the popup menu that appears, select /**Add/KALLISTOS optimiser**
- ➢ Alternatively you can click on the icon on the toolbar to add an optimiser
- ➢ In the New Node dialog box that appears type the name “Bend Optimiser” as the name of your optimiser.

An optimiser node will be added to your project. It will appear as the icon in the *project tree*.

- ➢ Double click on the optimiser node to open the optimiser. You should see the following window



Source Node: BEND

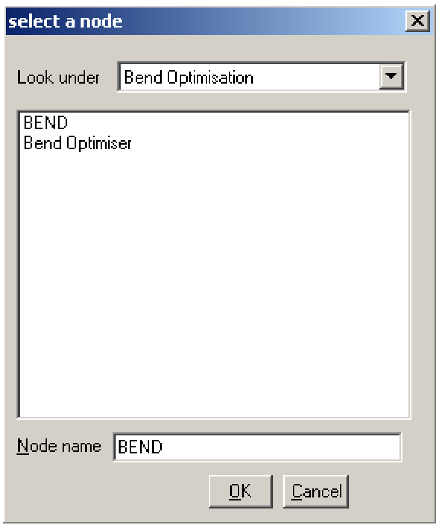
Optimiser: Global
Analytic derivs: no
Precision: 0
Max Func Evals: 100

Optimiser Main Window

We want to specify the node "BEND" as the source node to be optimised.

- Clicking on the toolbar icon will allow you to change the source node within the project that you wish to optimise. Alternatively you can select **source node/set source node** from the top menu

You should see the following dialog window:



Look under: Bend Optimisation

BEND
Bend Optimiser

Node name: BEND

Setting the source node

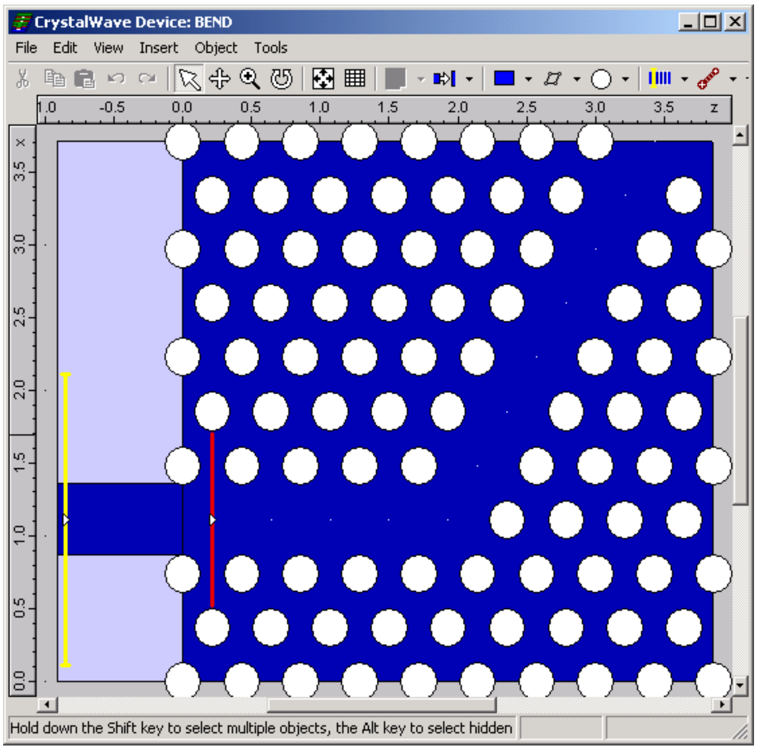
You can set the source node by clicking on an item in the list. The drop down box can be used to select the main node. The items with […] have sub nodes, double clicking on these will allow you select subnode. However in our case we do not need to do this.

- Set the source node to be "BEND" in the text box and press the **OK** button

We can view the node that we have chosen to optimise. This will allow us to check that we have the correct node.

- Click on the view source node tool on the toolbar in the Main Window. This will show the node to be optimised. Alternatively you can select **/source node/view source node** from the top menu

You should see the initial device prior to the optimisation process as shown below


CrystalWave Device: BEND
File Edit View Insert Object Tools
1.0 -0.5 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 z
x 3.5 3.0 2.5 2.0 1.5 1.0 0.5 0.0

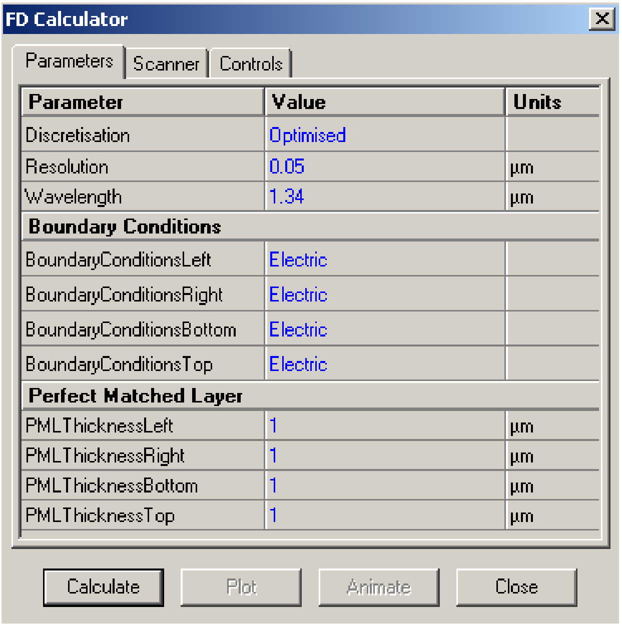
Hold down the Shift key to select multiple objects, the Alt key to select hidden

The initial bend design in CRYSTALWAVE

This photonic crystal has a bulk refractive index of 2.5, (the holes are air). This is currently set up for a 2D simulation

- Select the **Tools/Calculator/FD Calculator…** menu item (or, if the only engine option you have is the FD option, **Tools/FD Calculator…**).

The FD Calculator will appear:



Parameter	Value	Units
Discretisation	Optimised	
Resolution	0.05	μm
Wavelength	1.34	μm
Boundary Conditions		
BoundaryConditionsLeft	Electric	
BoundaryConditionsRight	Electric	
BoundaryConditionsBottom	Electric	
BoundaryConditionsTop	Electric	
Perfect Matched Layer		
PMLThicknessLeft	1	μm
PMLThicknessRight	1	μm
PMLThicknessBottom	1	μm
PMLThicknessTop	1	μm

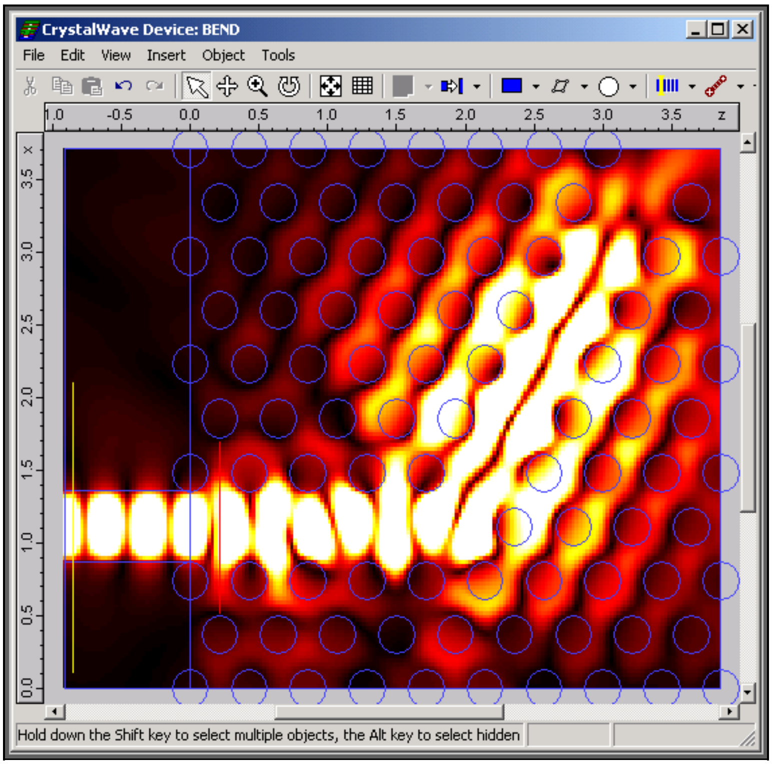
The FD Calculator

Unlike the FDTD calculator this is a Frequency Domain tool and as such, a working wavelength needs to be specified. Here it is set to be 1.34um. Which has been determined previously to be in the band gap for this crystal structure.

- Press **Calculate**

After only a few seconds the FD Calculator has solved the system. Notice that the FD Calculator is extremely fast. (This is why it can be very powerful when used in conjunction with the KALLISTOS optimiser. Thousands of iterations can be performed in an hour!)

- On the Controls panel, make sure **Hy** is selected from the drop-down choice at the top, the radio button to the left of the drop-down choice is selected, and the **amplitude** radio button is selected.

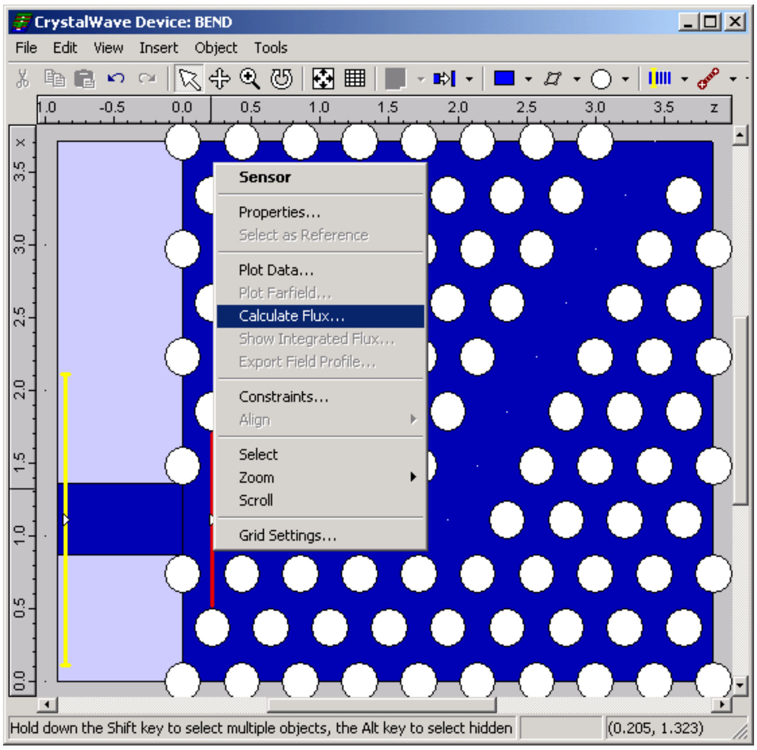

CrystalWave Device: BEND
File Edit View Insert Object Tools
1.0 -0.5 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 z
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 x

Hold down the Shift key to select multiple objects, the Alt key to select hidden

Hy field plot of the non optimsed bend

It is obvious from the field plot that much of the light is reflected at the junction of the bend.

- Close the FD Calculator
- To find out how much is reflected, right-click on the sensor and select **/Calculate Flux...**



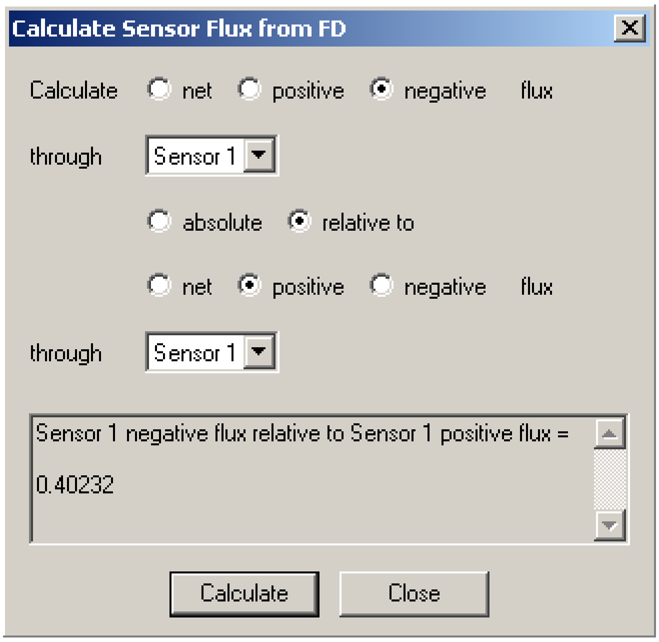
Hold down the Shift key to select multiple objects, the Alt key to select hidden

(0.205, 1.323)

Selecting Calculate Flux…

- ➢ In the **Calculate Sensor Flux from FD** dialog that appears, press **Calculate**

The sensor flux will be calculated. Note that the negative flux (i.e. the reflection from the bend passing back through the sensor) is around 41%.


Calculate Sensor Flux from FD

Calculate ○ net ○ positive ● negative flux

through Sensor 1

○ absolute ● relative to

○ net ● positive ○ negative flux

through Sensor 1

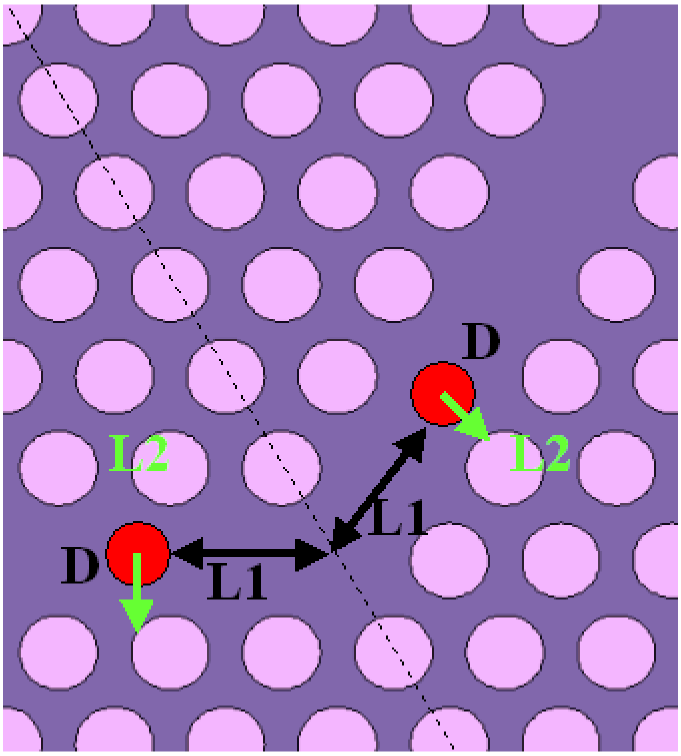
```
Sensor 1 negative flux relative to Sensor 1 positive flux =
0.40232
```

Calculating Sensor Flux from FD

It is this parameter that we plan to minimise by using the KALLISTOS Optimiser.

2.2 Parameterising the device

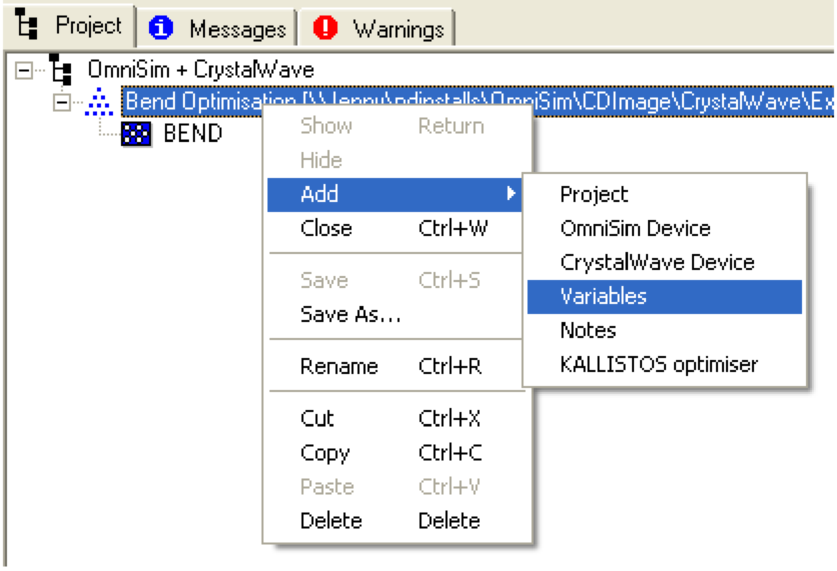
In order to optimise our bend, we will insert two defect atoms of variable size. The optimiser will vary their diameter and their position. These are the variables that the optimiser will vary. Note that we must make sure to keep the symmetry of the junction.



The three variables in the optimisation

Variable Name	Function
L1	The distance along the waveguide axis from the bend
L2	The distance from the waveguide axis towards the outer edge of the bend
D	The diameter of the defect atoms

- ➢ Right-click on the project icon to add a new *Variables* node:

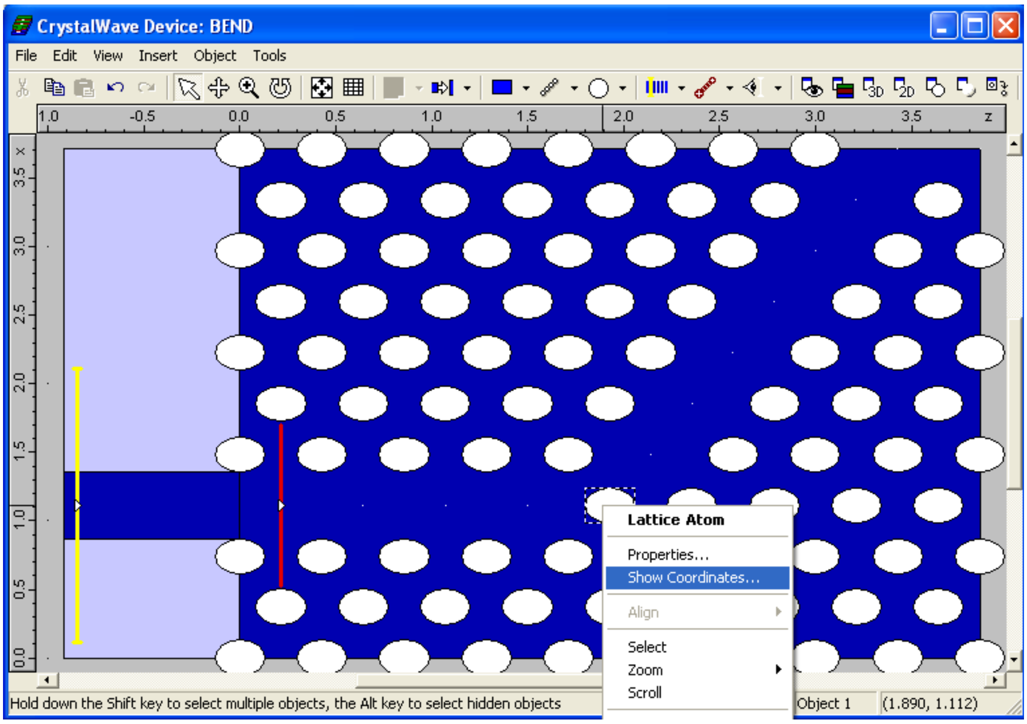


- ➢ Type “Bend Variables” for the *Variables* node name.

- Open the “Bend Variables” *Variables* node from the *Project Tree* by double clicking on it.
- Add 7 new *Variables*, named *L1, L2* and *D* for the parameters of the optimiser, and *Xa, Za, Xb* and *Zb* for the coordinates of the defect atoms which we will express as a function of *L1* and *L2*.

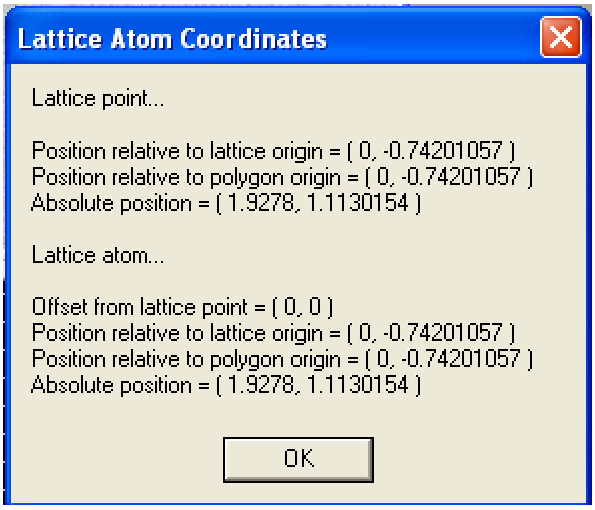
The next step is to adapt the device.

- Open the CrystalWave Device “bend” by double clicking on it at the Project Tree.
- Click on the icon to select the lattice atom tool, then click on the empty lattice point in the middle of the bend to add a lattice atom. Then right click on this atom and select **Show Coordinates…**



Hold down the Shift key to select multiple objects, the Alt key to select hidden objects

You should see the following window:


Lattice Atom Coordinates

Lattice point...

Position relative to lattice origin = (0, -0.74201057)
Position relative to polygon origin = (0, -0.74201057)
Absolute position = (1.9278, 1.1130154)

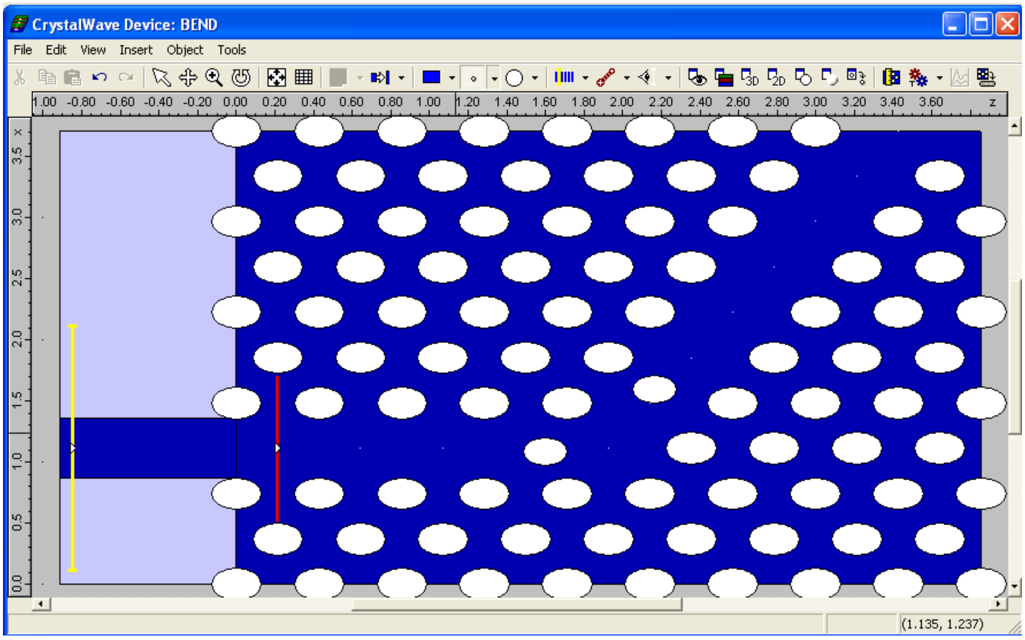
Lattice atom...

Offset from lattice point = (0, 0)
Position relative to lattice origin = (0, -0.74201057)
Position relative to polygon origin = (0, -0.74201057)
Absolute position = (1.9278, 1.1130154)

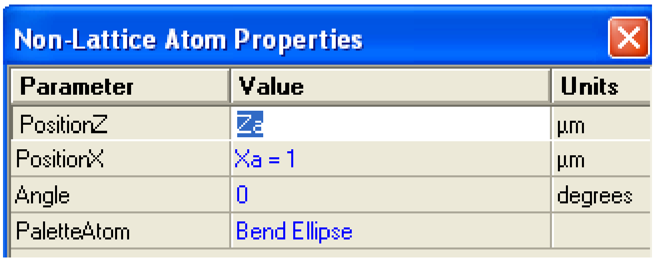
OK

- As you can see on this window, the position of the central atom relative to the lattice origin is defined by z=0, x=-0.74201057. Knowing that the lattice is hexagonal, we can deduce the position of the centres of the two defect atoms as a function of L1 and L2.

- Now, with the lattice atom tool selected, click on the central atom to delete it.
- Click on ○ ○ to select Bend Ellipse, the kind of atom whose diameter we will vary. Click on ∘ to select the non-lattice atom tool, and click on the waveguide twice to create the defects atoms, as shown below.



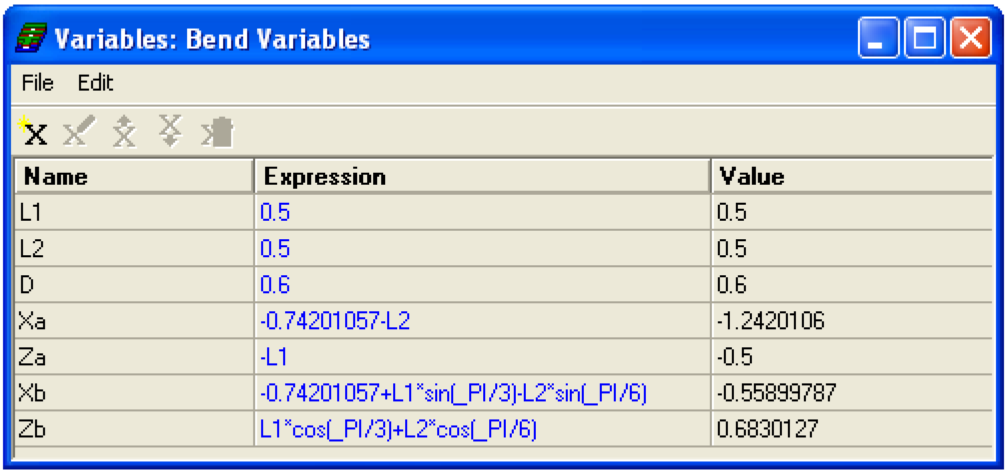
- Right click on the first non-lattice atoms and select Properties… Set *PositionZ* to *Za* and *PositionX* to *Xa*, as shown below, then double click on *Bend Ellipse*.



Non-Lattice Atom Properties

Parameter	Value	Units
PositionZ	Za	µm
PositionX	Xa = 1	µm
Angle	0	degrees
PaletteAtom	Bend Ellipse	

- The 2D Atom Properties appear, set both *SizeX* and *SizeZ* to *D*.
- Repeat the operation for the second atom, this time with *Zb* and *Xb*. This time there is no need to change the 2D Atom Properties.
- Back to the Bend Variables node, set their values according to the following window.



Name	Expression	Value
L1	0.5	0.5
L2	0.5	0.5
D	0.6	0.6
Xa	-0.74201057-L2	-1.2420106
Za	-L1	-0.5
Xb	-0.74201057+L1*sin(_PI/3)-L2*sin(_PI/6)	-0.55899787
Zb	L1*cos(_PI/3)+L2*cos(_PI/6)	0.6830127

The structure is now fully parameterised, and we can start setting up the optimiser.

2.3 Setting up the optimiser

In order to run our optimisation we will need to set the appropriate settings for our problem.

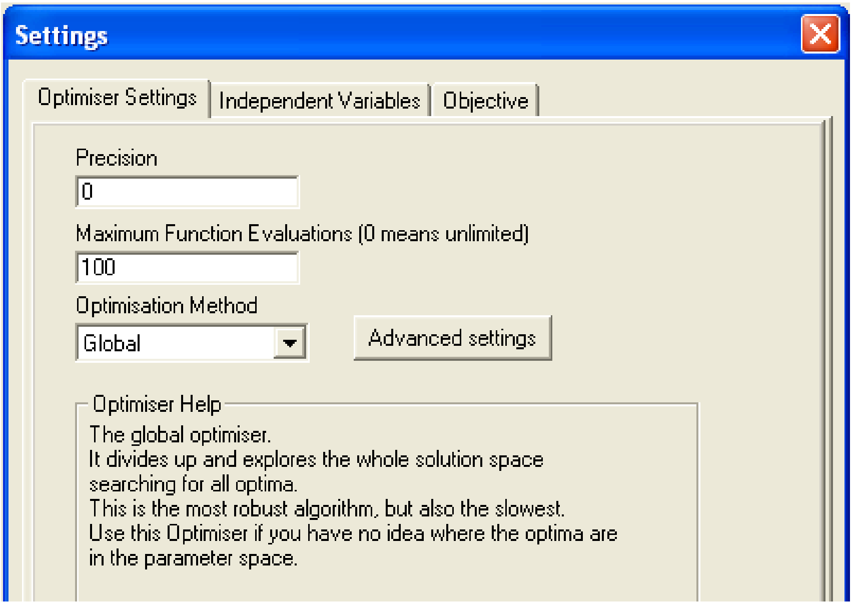
- Open the node of the optimiser.
- Click on the settings 🛠 tool on the toolbar in the main window. Alternatively you can select **optimiser /settings** from the top menu

A dialog should appear with three tabbed panels.

2.3.1 Choosing the optimiser

- In the *Optimiser Settings* tab, set the **Optimisation Method** to *Global*. We do not know beforehand what the optimum configuration might be, therefore we want to search the whole of the solution space so this method is ideal. (The Help box tells you a little information about the optimiser that you have selected to use.)
- Set the **Precision** to *0*.
- We want to set the **Maximum Function Evaluations** to 100 so that the run continues until this number of iterations is completed.

You should now have the following:



Precision: 0

Maximum Function Evaluations (0 means unlimited): 100

Optimisation Method: Global

Optimiser Help

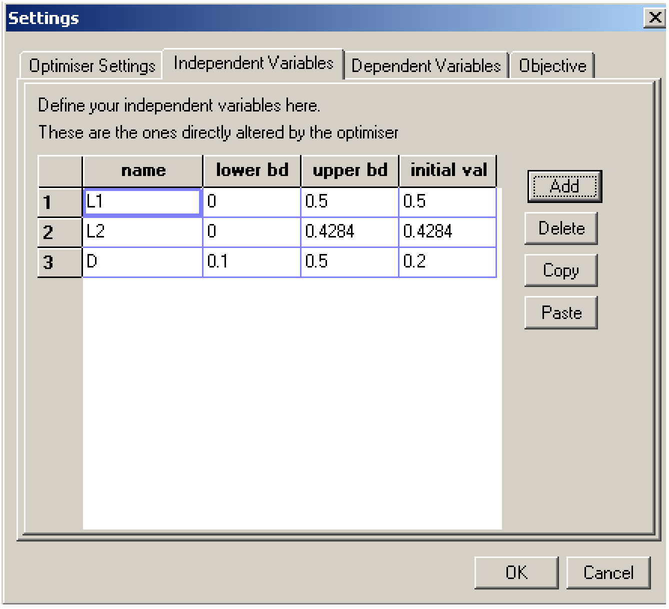
The global optimiser.
It divides up and explores the whole solution space
searching for all optima.
This is the most robust algorithm, but also the slowest.
Use this Optimiser if you have no idea where the optima are
in the parameter space.

The Settings Panel for the bend optimisation

2.3.2 Setting up the Independent Variables

Next, we set up the *Independent Variables* for our problem. These are the variables that the optimiser will vary.

- In the Independent Variables tab, insert these parameters into the table using the **Add** button, giving them the lower, upper bounds and initial values as shown below.



Define your independent variables here.
These are the ones directly altered by the optimiser

	name	lower bd	upper bd	initial val
1	L1	0	0.5	0.5
2	L2	0	0.4284	0.4284
3	D	0.1	0.5	0.2

Setting up the independent variables for the bend optimisaton

Notes

1) The upper bound for L2 is set to the period of the lattice (above this the defect atoms would lie fully in the lattice).

2) The lower bound for D, the diameter, of the defect holes is 0.1. This has been determined by the **Resolution** that is used for the calculations (0.05um in this

example). Shapes smaller than this **Resolution** will not be considered in the calculations.

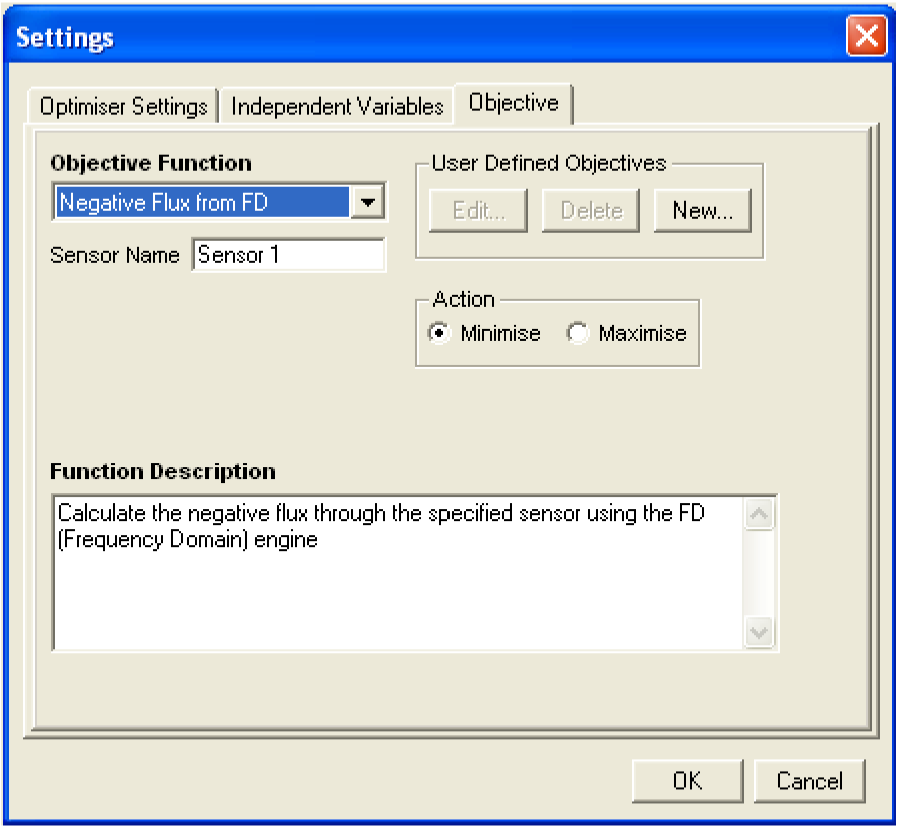
- ➢ Close the KALLISTOS Settings Panel.

2.3.3 Choosing the objective function

Once we have finished setting up the *Dependent Variables* we need to set the objective function that we wish to optimise.

- ➢ Click on the *Objective* tab.
- ➢ We want the minimise the flux reflected back from the junction, so set **Objective Function** to *"Negative Flux From FD"*
- ➢ Type "Sensor 1" into the Sensor Name box.
- ➢ We want to minimise the reflection, so set **Action** to *minimise.*

You should now have the following:



Objective Function

Negative Flux from FD

Sensor Name: Sensor 1

Action: ● Minimise ○ Maximise

Function Description

Calculate the negative flux through the specified sensor using the FD (Frequency Domain) engine

Setting the objective function for the bend optimisation

We have now set all the settings that we need for the optimisation.

- ➢ Click the **OK** button on the settings dialog to accept all these settings.

We are now ready to run the optimiser.

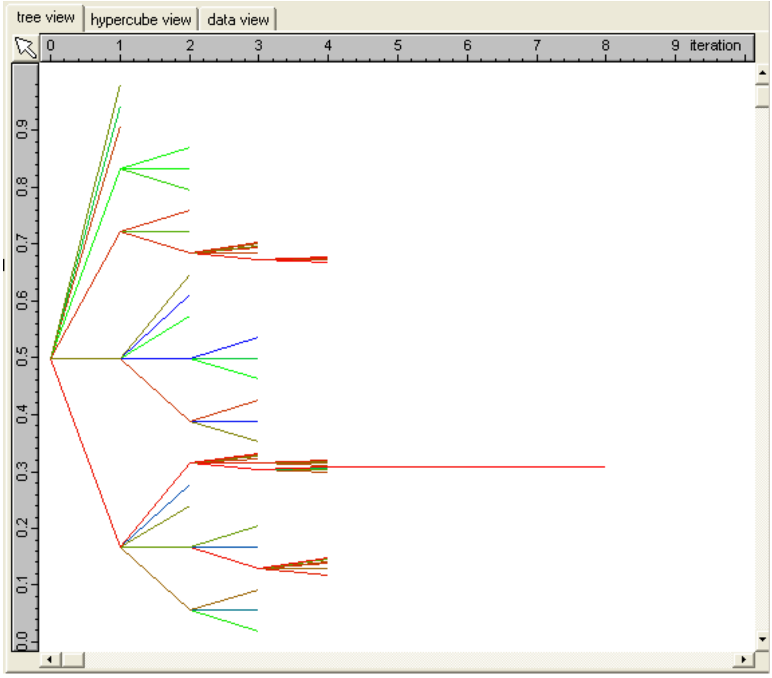
2.4 Running the optimiser.

- ➢ Click on the Run tool ⚡ on the toolbar in the main window (or select **/Optimiser/Run** from the top menu).

The optimiser will start to run. The green dial in CRYSTALWAVE will turn. As the optimiser runs it will update the graphical views, so you should be able to gauge its progress as it runs.

- You can monitor the progress of the optimisation by looking at the various tabbed views. If click on the *data view* you can follow the numerical quantities themselves.
- Now click on the *Tree view* tab.

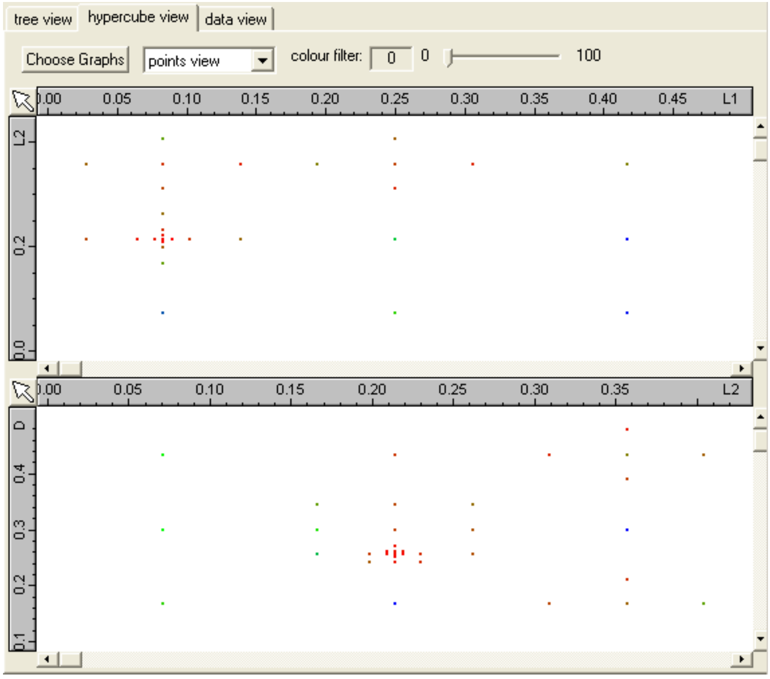
The *Tree view* display is a graphical representation of how the global optimiser systematically divides up and explores the solution space. Each branch represents a different region in the parameter space. The longer the branches, the more the optimiser tried to explore there, so these tend to correspond to local minima. The line colour indicates how good the function value is. It goes from blue for the worst function value, through green, to red for the best one. So the red shoots represent the better local optima. The longest red shoot is the global optimum found so far. For example, on completion of the optimisation you should see the following view:



The Tree View for the bend optimisation

- Click on the *hypercube view* to see the optimisation evolving through the parameter space.

On the *hypercube view* we can see the local optima. Again the colours vary from blue for the worst function values, through green, to red for the best ones. Notice the clustering of points. These correspond to local minima that the optimiser explored in more detail. Indeed this gives an indication of how the optimiser works: it explores the entire parameter space, but preferentially around the “better” points found so far. As a consequence the clusters will correspond to the “interesting regions” in the parameter space. Again, you can click on these to view the local solution.



The Hypercube View Panel

- You can look at the various *views* simultaneously by clicking on the **Detach View** button. This will put the currently selected *view* into a separate window. Click on the **x** button on the top right to put the *view* back in a tabbed window.
- Clicking on any data point in any *view* will highlight the corresponding data point in the **Selected Point** box and all other *views*.
- You can also inspect the best point found so far by pressing **Select Best**. This will also highlight the point in all the *views*.

Note that you can stop the optimisation at any time by clicking the running symbol in the CRYSTALWAVE main window. Clicking the run tool again will continue the run from where it left off.

The FD Engine is extremely fast, so each iteration takes only a few seconds to evaluate. Thus the whole optimisation – 100 points in three-parameter space – should take around 15-20 minutes to complete.

As it runs you should notice some changes to the graphical display on the right panel of the main window.

2.5 Analysing the results

Looking at the final result

Once the optimisation has finished, we can now analyse the optimisation process and look at the optimum solution that the optimiser has found. KALLISTOS provides graphical views in order to assist the user in analysing the optimisation process.

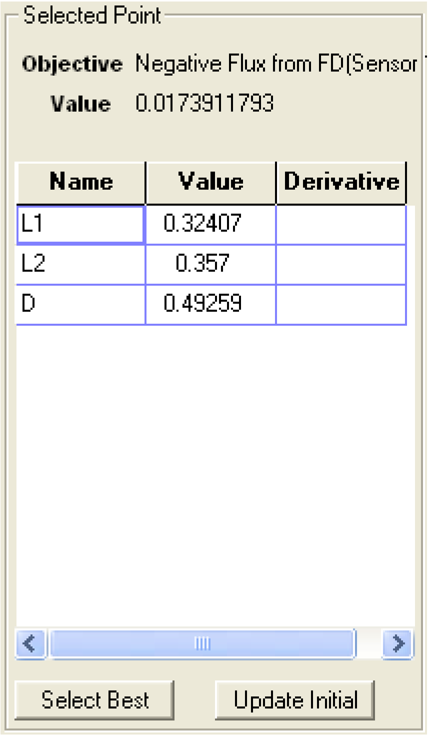
- Click on the **Select Best** button of the main window. The point with best function value will be highlighted in all views.

On the *tree view* the selected point is highlighted with a large black dot.

On the *hypercube view* the selected point is highlighted in each graph with a large black dot.

On the *data view* the selected row corresponding to the selected point is highlighted in blue.

The information concerning this point will be displayed in the **Selected Point** table on the left-hand panel of the main window as shown below



Selected Point

Objective Negative Flux from FD(Sensor

Value 0.0173911793

Name	Value	Derivative
L1	0.32407	
L2	0.357	
D	0.49259	

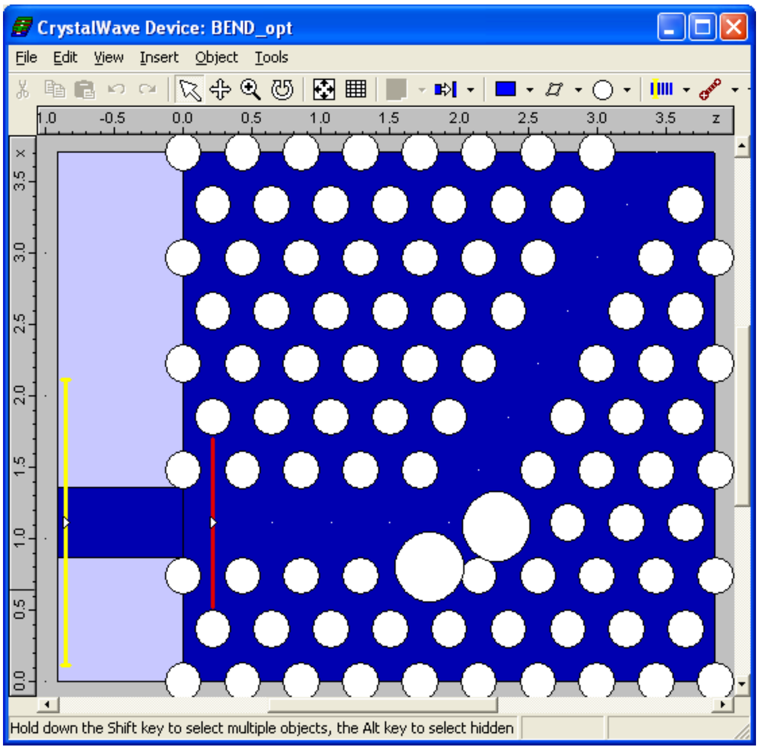
Selecting the best point

In the selected point information we can see that KALLISTOS has found a solution which has small reflection. If we had run the optimisation for longer, e.g. for 500 iterations, KALLISTOS would have found a solution with even smaller reflection.

The corresponding values for each of the parameters are also listed for this solution in the table.

To get a clearer idea of what this solution looks like we can view it in CRYSTALWAVE as we did for our starting solution.

- Click on the **Update Initial** button of the main window. The node is updated to match the optimal value.
- Double click on the node (if not already displayed).

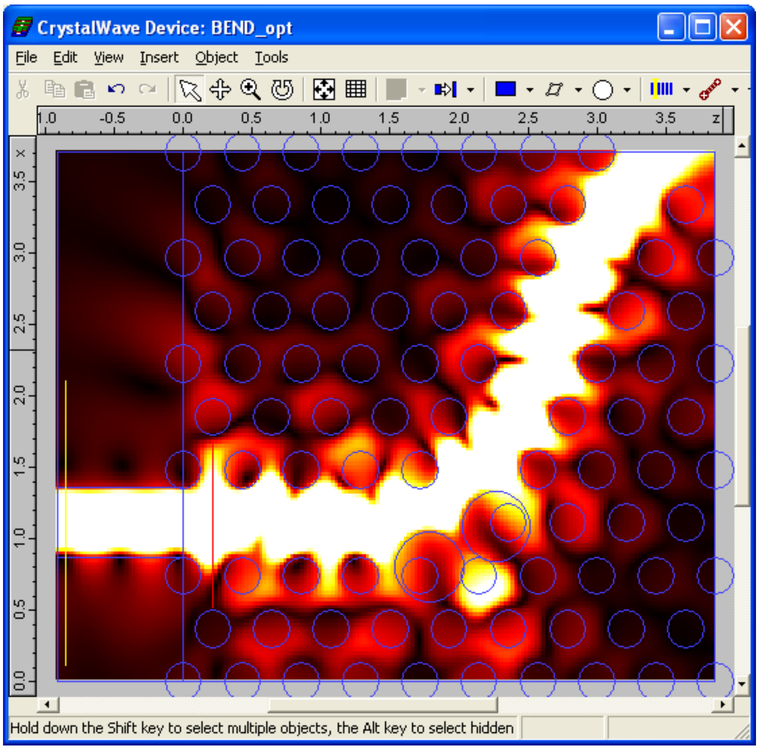

CrystalWave Device: BEND_opt
File Edit View Insert Object Tools
1.0 -0.5 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 z
x 3.5 3.0 2.5 2.0 1.5 1.0 0.5 0.0

Hold down the Shift key to select multiple objects, the Alt key to select hidden

Viewing the optimal device

- ➢ Select **Tools/Calculator/FD Calculator…** from the menu of the node.
- ➢ On the FD Calculator dialog that appears, press **Calculate**
- ➢ On the Controls panel, make sure **Hy** is selected from the drop-down choice at the top, the radio button to the left of the drop-down choice is selected, and the **amplitude** radio button is selected.

You should get the following:


CrystalWave Device: BEND_opt
File Edit View Insert Object Tools
1.0 -0.5 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 z
x 3.5 3.0 2.5 2.0 1.5 1.0 0.5 0.0

Hold down the Shift key to select multiple objects, the Alt key to select hidden

The field intensity for the optimal device

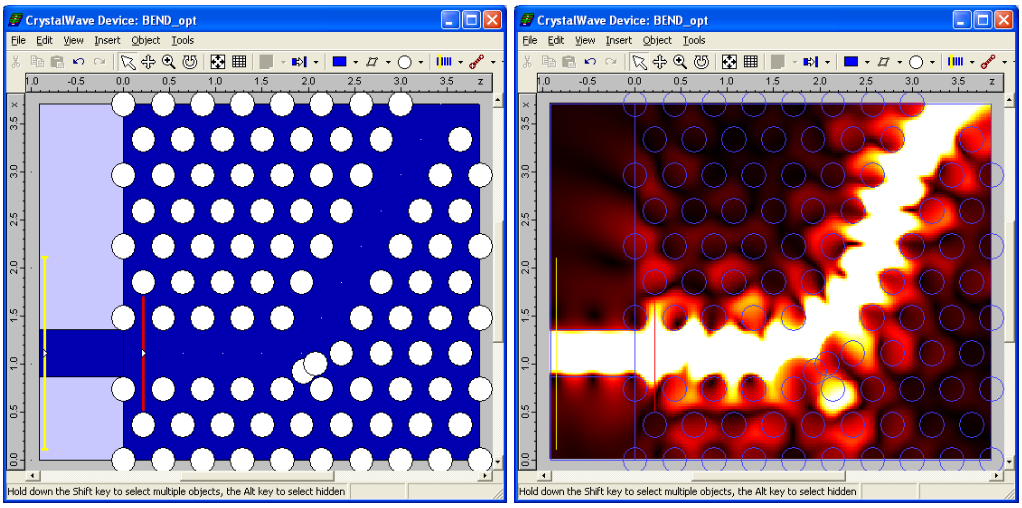
The optimiser has therefore managed to find an more efficient bend in a very short time. Given longer, e.g. 500 iterations, KALLISTOS would have found a highly optimised bend.

Looking at the intermediate results

Here we have used a deterministic algorithm, meaning that if you run it long enough, it is **eventually** guaranteed to find an optimum. This is in contrast with stochastic type algorithms (e.g. genetic, evolutionary, simulated annealing strategies) which may at times prove quicker, but are by no means guaranteed to find the global optimum. This algorithm follows a systematic splitting strategy: it divides all the areas of the parameter space, but sub divisions happen more frequently in those areas deemed more likely to have an optimum. Eventually all points in the parameter space are covered, which is why it is deterministic.

One significant advantage that this method has over genetic or evolutionary strategies is that all history is stored. Thus allowing us to look at all the local minima that have been found in the solution space.

- ➢ Again click on the tab headed *Data view.*
- ➢ Click on the Fvalue tab at the top of the table – this will re-order the iterations from best to worst.
- ➢ Click on the list and use the arrow keys to scroll down the list while making sure that you can see the node at the same time. You will see that there are many different "close-to-optimal" configurations that you may wish to consider in regard to some other criteria (E.g. easier to fabricate). One other configuration with high transmission is shown below.


CrystalWave Device: BEND_opt
File Edit View Insert Object Tools
CrystalWave Device: BEND_opt
File Edit View Insert Object Tools

Hold down the Shift key to select multiple objects, the Alt key to select hidden

Hold down the Shift key to select multiple objects, the Alt key to select hidden

The field intensity for another high transmission device

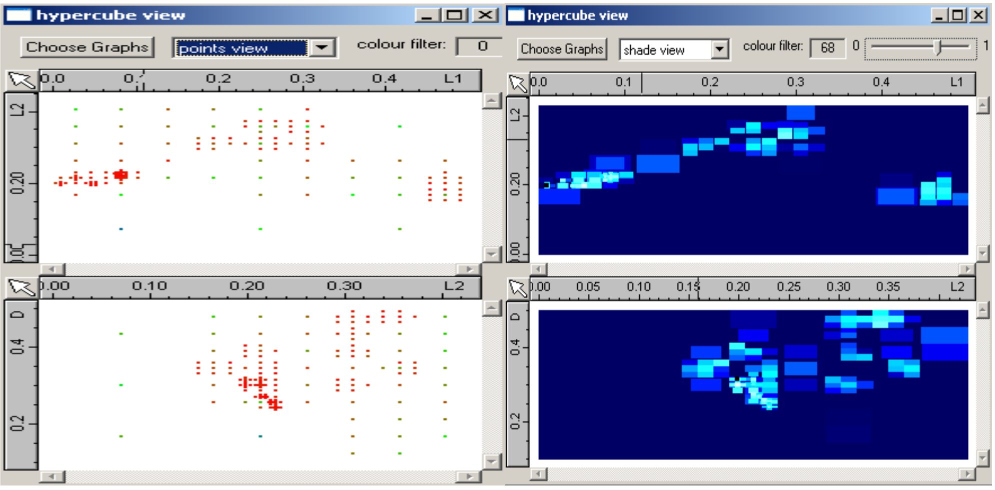
- ➢ Again click on the tab headed *Hypercube view*
- ➢ Select **Shade View**

This view displays the calculated points as regions "filled out" to cover missing gaps, and therefore indicate more clearly the regions with better function values.

You can use the **colour filter** to highlight regions with the better function values. Increasing the colour filter progressively will ignore any regions with non-optimal function values, and will eventually highlight only the best point.

- ➢ Increase the colour filter to around 70%

This allows you an immediate grasp of the tolerance of the various "optimal" solutions.



points view

colour filter: 0

shade view

colour filter: 68

Using the colour filter for the Hypercube view