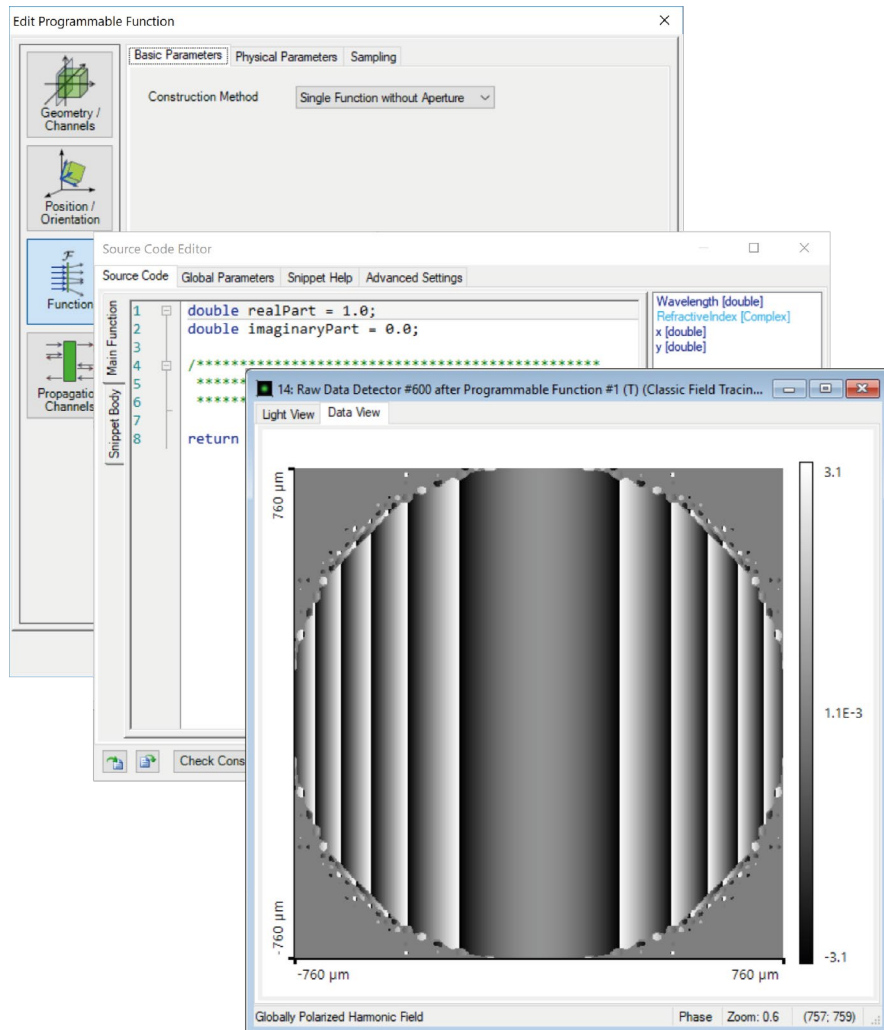


How to Work with the Programmable Function & Example (Cylindrical Lens)

Abstract



Providing maximum versatility for your optical simulations is one of our most fundamental objectives. In this tutorial we explain how to work with the Programmable Function, which can also be thought of as an idealised component acting in a single plane: the workflow entails defining a position-dependent, complex-valued function on the x, y plane, which is then multiplied onto the incoming field. We use the example of an ideal cylindrical lens to go through the whole process in detail.

Where to Find the Programmable Function: Catalog

The image illustrates the steps to find and edit a programmable function in a software application. The steps are numbered 1 through 7:

1. Click on the **Catalogs** tab in the top menu bar.
2. Click on the **Boundary Responses** icon in the left sidebar.
3. In the **Boundary Responses Catalog**, click on the **Definition Type** dropdown menu, which is currently set to **Templates**.
4. In the list of functions, click on **Programmable Function**.
5. Click on the **Tools** icon at the bottom left of the catalog window.
6. In the **Edit Programmable Function** dialog box, click on the **Physical Parameters** tab.
7. Click on the **Edit** button in the **Definition** section of the dialog box.

The **Source Code Editor** window shows the following code:

```
1 double realPart = 1.0;
2 double imaginaryPart = 0.0;
3
4 /****** INSERT YOUR CODE HERE *****/
5
6
7
8 return new Complex(realPart, imaginaryPart);
```

The variable declaration list on the right side of the editor shows:

- Wavelength [double]
- RefractiveIndex [Complex]
- x [double]
- y [double]

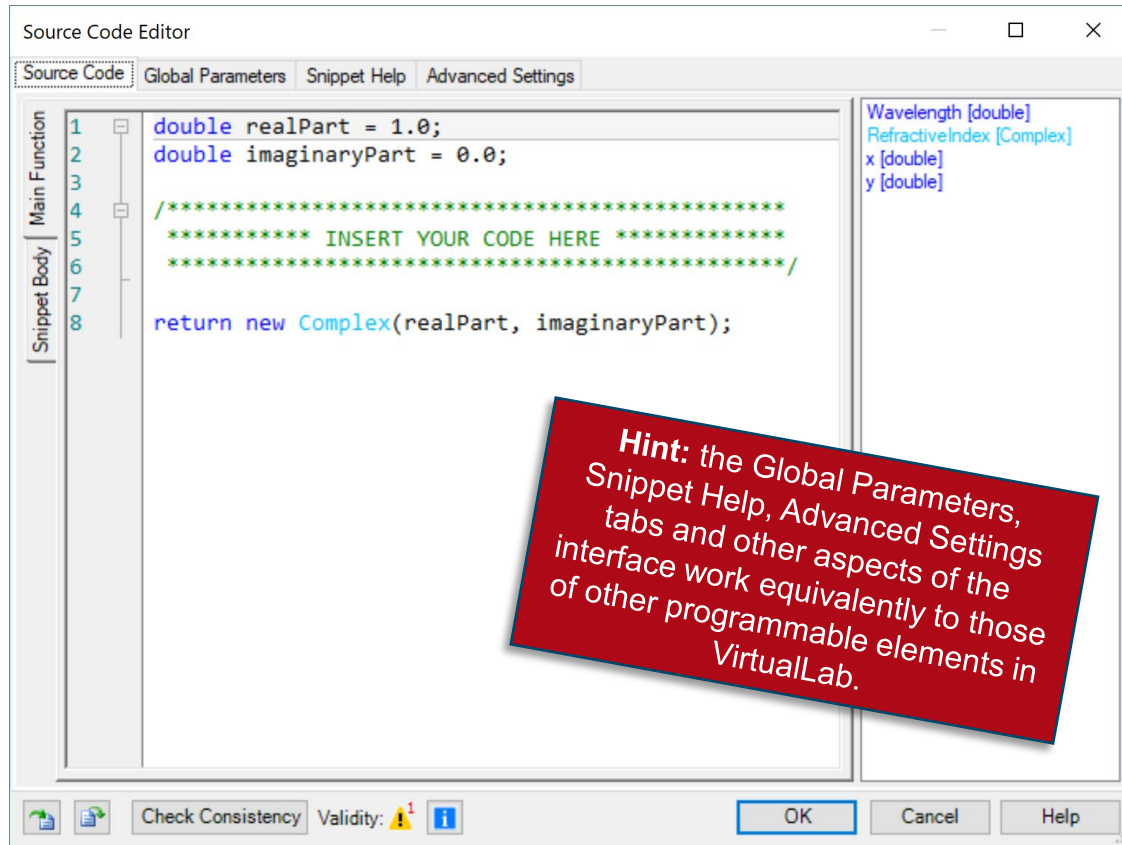
Where to Find the Programmable Function: Optical Setup

The image shows a software interface for an optical setup. On the left, a tree view under 'Ideal Components' has 'Programmable Function' highlighted with a red hand icon labeled '1'. A red arrow points from this icon to a 'Programmable Function' component on a central workspace, which is also highlighted with a red hand icon labeled '2'. Another red arrow points from the component to the 'Edit Programmable Function' dialog box. In this dialog, the 'Physical Parameters' tab is selected with a red hand icon labeled '3', and the 'Edit' button is highlighted with a red hand icon labeled '4'. A red arrow points from the 'Edit' button to a 'Source Code Editor' window. The editor shows a C# code snippet for a main function that returns a complex number based on real and imaginary parts. The code is as follows:

```
1 double realPart = 1.0;
2 double imaginaryPart = 0.0;
3
4 /***** INSERT YOUR CODE HERE *****/
5
6
7
8 return new Complex(realPart, imaginaryPart);
```

On the right side of the main workspace, there is a 'Ray Tracing System Analyzer' component with a value of '800'. The 'Edit Programmable Function' dialog also has tabs for 'Basic Parameters' and 'Sampling', and a 'Validity' indicator showing a green checkmark.

Writing the Code



- The panel on the right shows a list of available independent parameters.
- **Wavelength** is a default independent parameter that permits the user to implement a dispersive ideal component (function).
- **RefractiveIndex** is another default independent parameter that reads the complex-valued refractive index of the embedding medium.
- Finally, **x** and **y** are the last two default independent parameters. They span the plane on which the ideal component (function) is defined.

Writing the Code

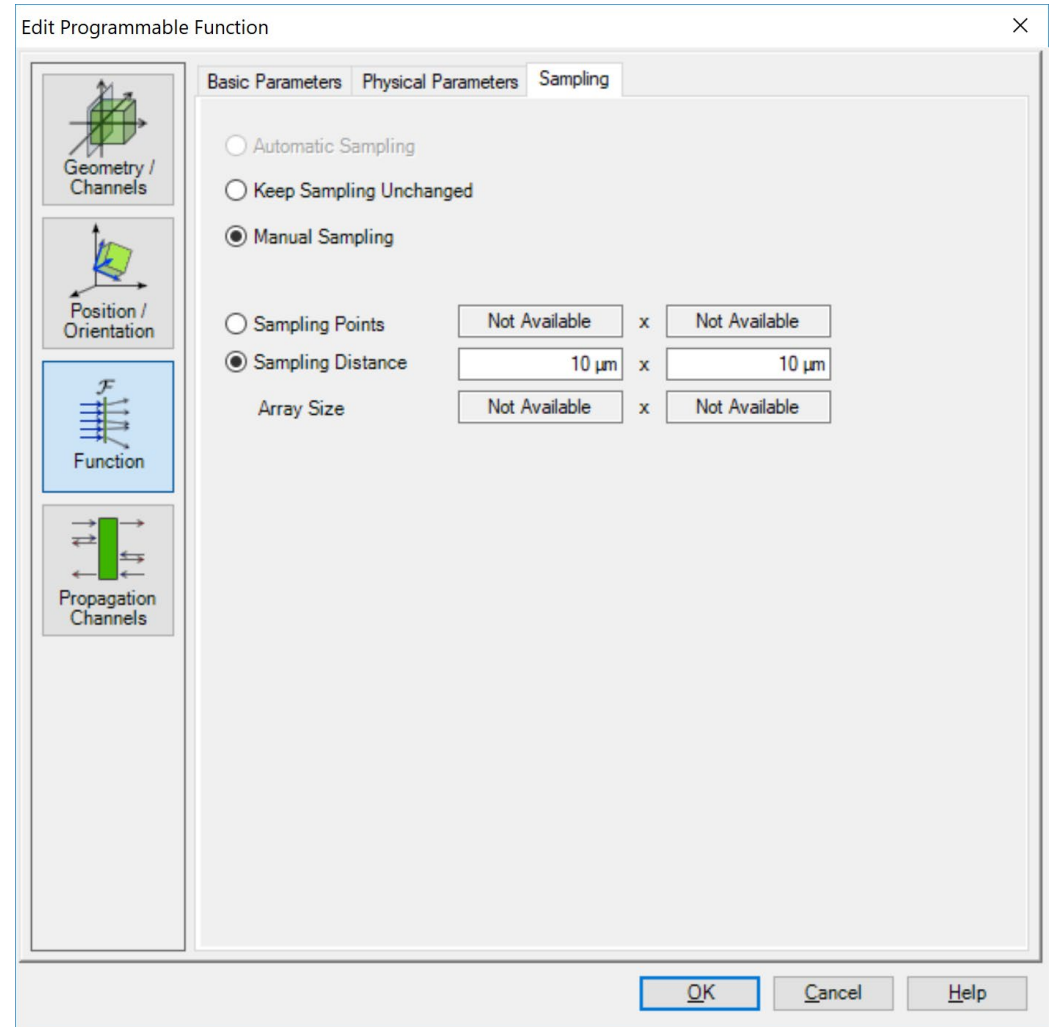
```
Source Code Editor
Source Code Global Parameters Snippet Help Advanced Settings
Main Function
1 double realPart = 1.0;
2 double imaginaryPart = 0.0;
3
4 /*****
5 ***** INSERT YOUR CODE HERE *****
6 *****/
7
8 return new Complex(realPart, imaginaryPart);
Snippet Body
Wavelength [double]
RefractiveIndex [Complex]
x [double]
y [double]
Check Consistency Validity: 1
OK Cancel Help
```

Hint: the Global Parameters, Snippet Help, Advanced Settings tabs and other aspects of the interface work equivalently to those of other programmable elements in VirtualLab.

- The Main Function must return a **Complex** value per **x**, **y** (possibly also **Wavelength**) which will then be multiplied onto the incoming field.
- Use the Snippet Body to group parts of the code in support functions.
- Note that it is possible to use an imported reference field and/or stack, and their associated parameters, in the code of the Programmable Function. The reference field and stack can be defined in the Global Parameters tab.

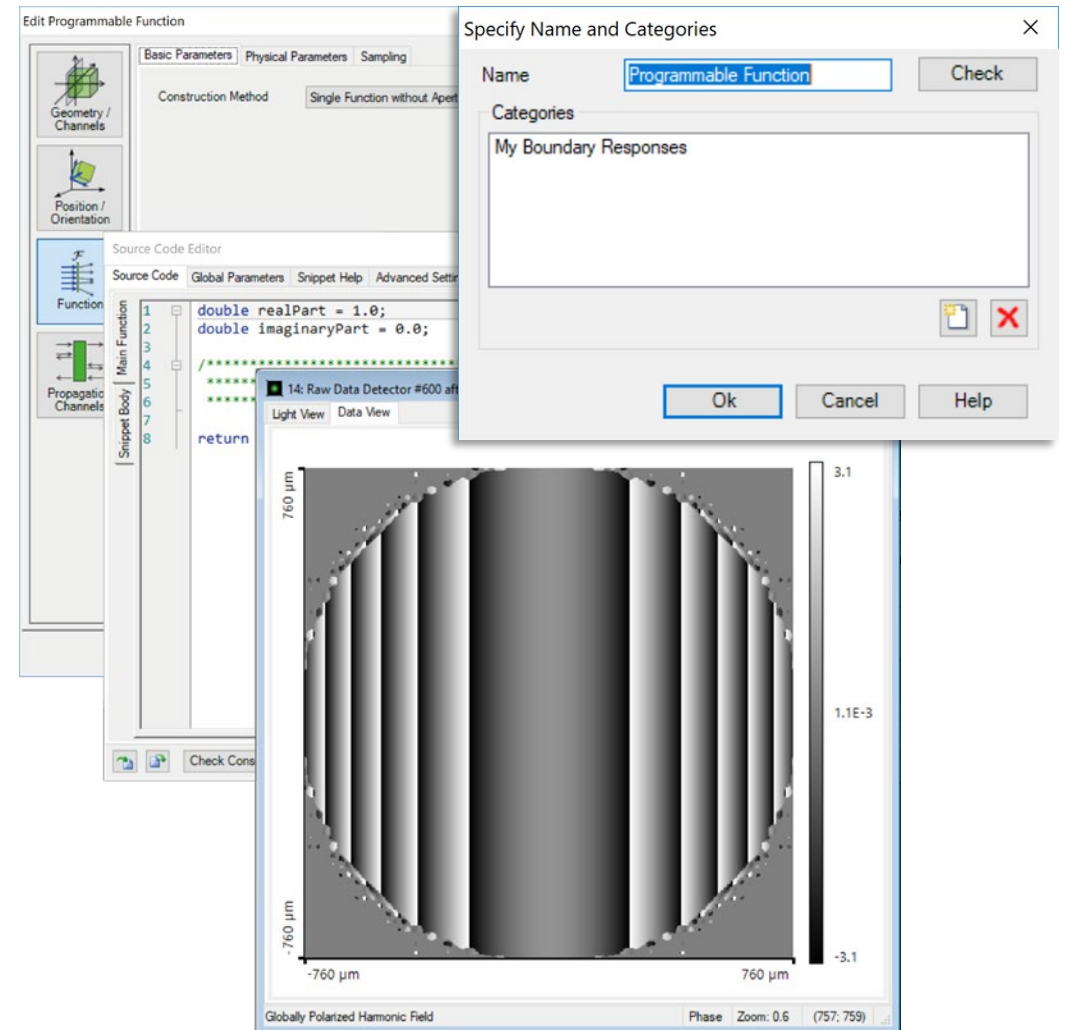
Sampling

- The user must ensure that the sampling (of the field behind the component) is fine enough to resolve the frequencies introduced by the programmable function.
- Use the Sampling tab for this purpose.
- Please note that the sampling may depend on the actual values of the defined global parameters.



Output

- The Programmable Function yields a complex-valued function per wavelength defined on a plane, spanned by x , y .
- In an Optical Setup, it is multiplied onto the incoming field.
- Hint: a snippet which has been programmed for a function can be employed also in the Programmable Source, and vice versa.
- The function can be saved in the Boundary Responses catalog for later use.



Programming a Cylindrical Lens Function

The Cylindrical Lens

A function that performs as a cylindrical lens is a phase-only modulation of the form:

$$\psi^{\text{cyl}}(x, y) = \text{sign}(f) k \sqrt{(x \cos \alpha + y \sin \alpha)^2 + f^2}$$

$f \rightarrow$ Focal length

$k \rightarrow$ Wavenumber

$\alpha \rightarrow$ Angle formed by the optical axis and the focusing direction of the lens

(1)

Where to Find the Programmable Function: Catalog

The image illustrates the steps to find and edit a programmable function in a software application. The steps are numbered 1 through 7:

1. Click on the **Catalogs** menu in the top navigation bar.
2. Click on the **Boundary Responses** icon in the left sidebar.
3. In the **Boundary Responses Catalog**, click on **Programmable Function** in the list.
4. Click the **Edit** button next to **Programmable Function** in the catalog.
5. Click the magnifying glass icon at the bottom of the catalog window.
6. In the **Edit Programmable Function** dialog, click on the **Physical Parameters** tab.
7. Click the **Edit** button in the **Definition** section of the dialog.

The **Source Code Editor** window displays the following code:

```
1 double realPart = 1.0;
2 double imaginaryPart = 0.0;
3
4 /****** INSERT YOUR CODE HERE *****/
5
6
7
8 return new Complex(realPart, imaginaryPart);
```

Additional parameters shown in the editor include: Wavelength [double], RefractiveIndex [Complex], x [double], and y [double].

Where to Find the Programmable Function: Optical Setup

The image illustrates the workflow for editing a programmable function in an optical simulation software. It is divided into three main sections:

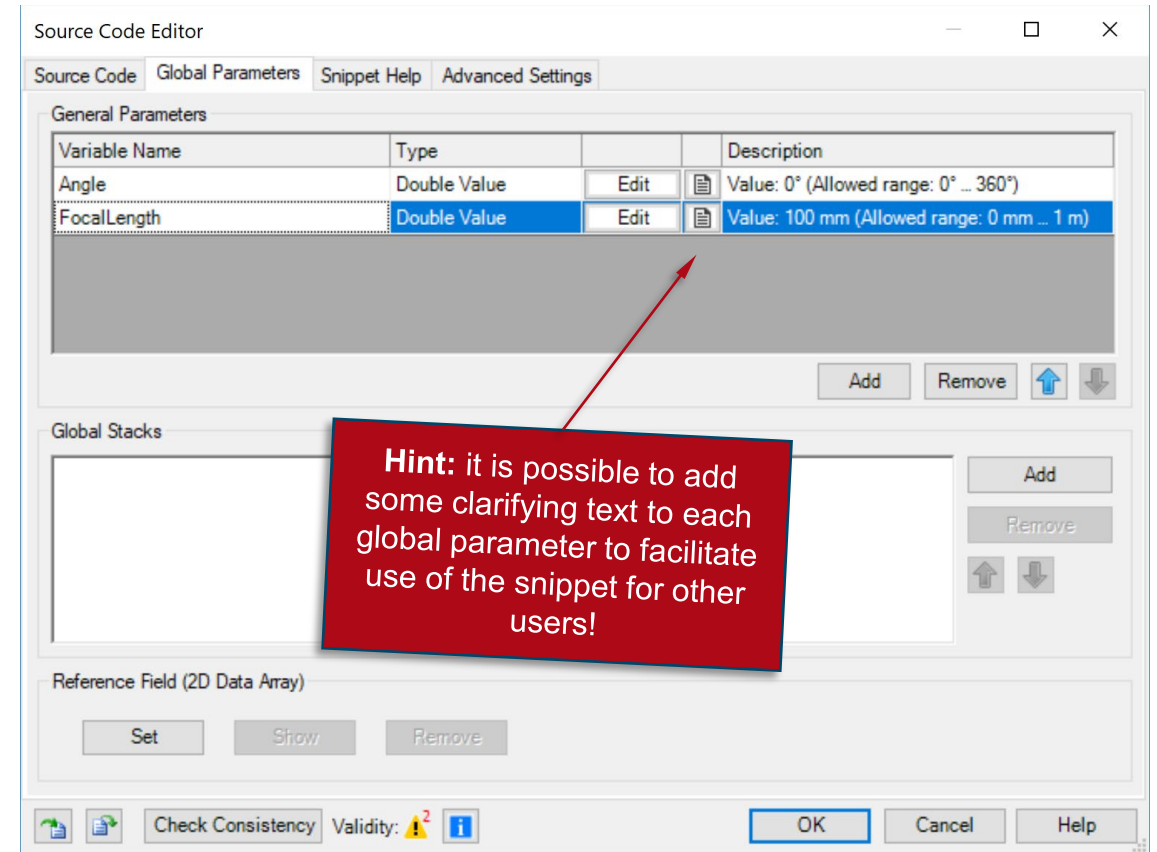
- Component Library:** On the left, a tree view shows various optical components. The 'Programmable Function' component is highlighted with a red hand icon and the number 1.
- Workspace:** The central workspace contains a 'Programmable Function' component (a blue circle) and a 'Ray Tracing System Analyzer' (a hexagon labeled '800'). A red hand icon and the number 2 point to the 'Programmable Function' component, with a red arrow indicating the next step.
- Edit Programmable Function Window:** This window is open on the right. It has tabs for 'Basic Parameters', 'Physical Parameters', and 'Sampling'. The 'Physical Parameters' tab is active, and a red hand icon and the number 3 point to it. Below the tabs, there is a 'Definition' section with an 'Edit' button. A red hand icon and the number 4 point to the 'Edit' button, with a red arrow pointing to the 'Source Code Editor' window.
- Source Code Editor:** This window is open in the foreground, showing a code editor with the following code:

```
1 double realPart = 1.0;
2 double imaginaryPart = 0.0;
3
4 /*****
5 ***** INSERT YOUR CODE HERE *****/
6 *****/
7
8 return new Complex(realPart, imaginaryPart);
```

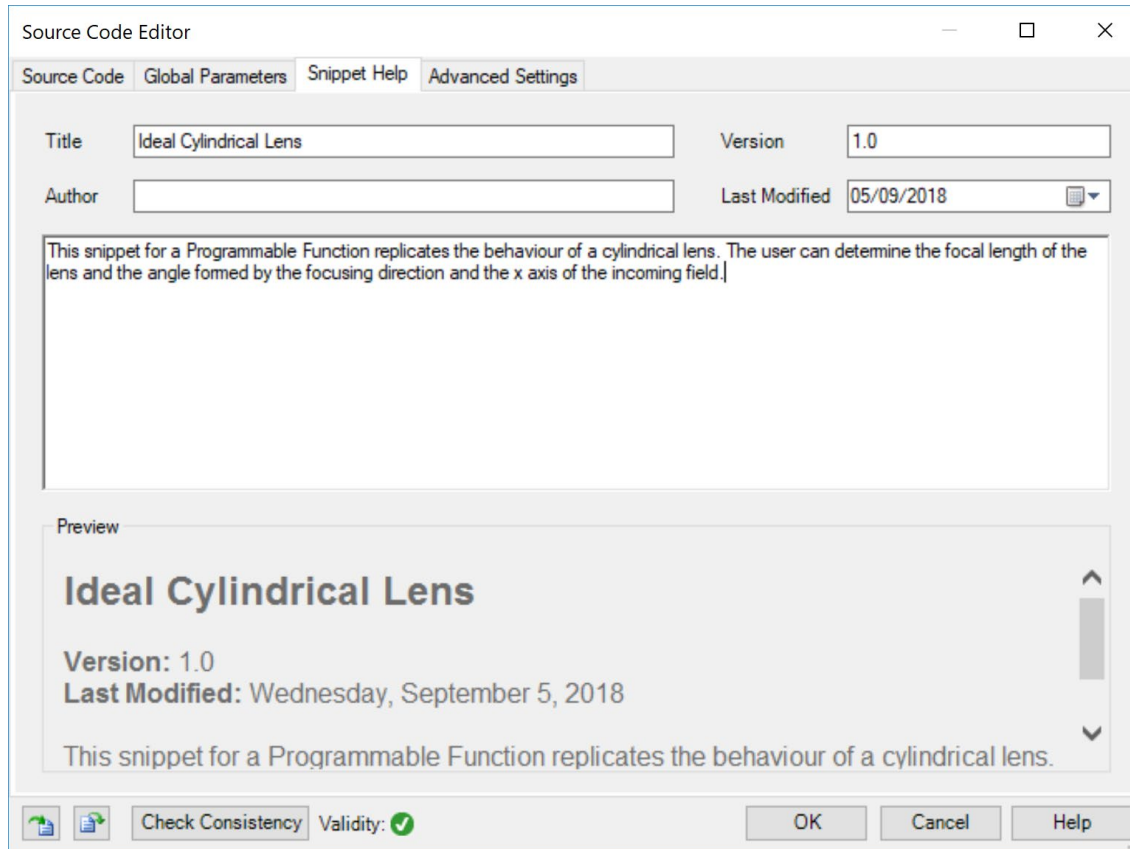
The editor also shows variable declarations on the right: 'Wavelength [double]', 'RefractiveIndex [Complex]', 'x [double]', and 'y [double]'. At the bottom, there are buttons for 'Check Consistency', 'Validity' (with a warning icon), 'OK', 'Cancel', and 'Help'.

Programmable Interface: Global Parameters

- Once you have triggered open the Edit dialogue, go to the Global Parameters tab.
- There, Add and Edit two global parameters:
 - **double** Angle = 0 deg (0 deg, 360 deg): represents the angle formed by the optical axis and the focusing direction.
 - **double** FocalLength = 100 mm (0 m, 1 m): represents the focal length of the lens.
- Use the button with the small “notes” icon to add some explanation to your custom global parameters.



Programmable Interface: Snippet Help



- **Optional:** you can use the Snippet Help tab to write instructions, clarifications, and some metadata associated to your snippet.
- This option is very helpful to keep track of your progress with a programmable element.
- It is especially useful when the programmable element is later disseminated to be handled by other users!

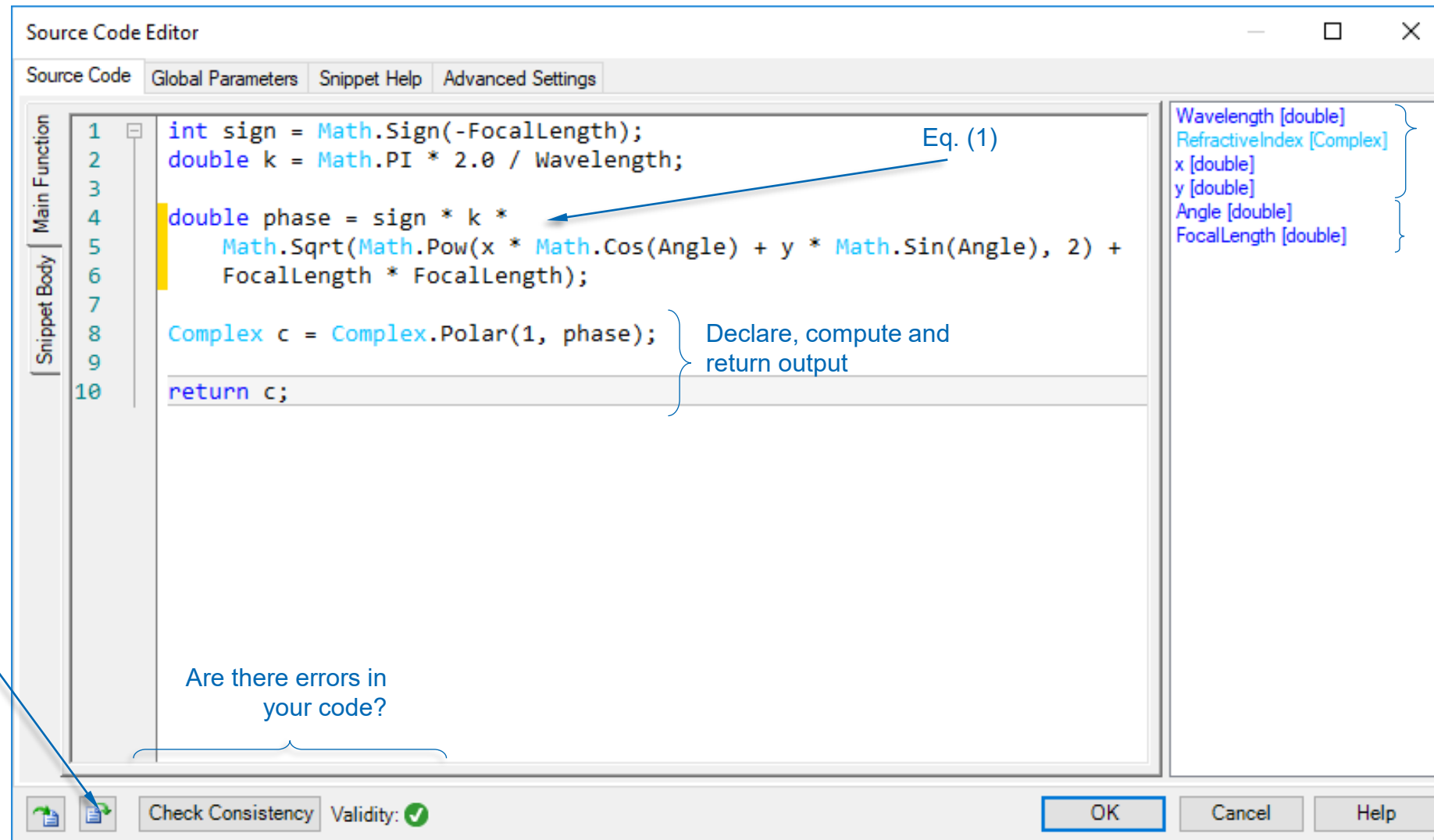
Programmable Interface: Snippet Help

The image shows a software interface with three main windows:

- Source Code Editor:** Contains metadata for the snippet: Title: Ideal Cylindrical Lens, Version: 1.0, Author: (empty), Last Modified: 05/09/2018. It also includes a text description: "This snippet for a Programmable Function replicates the behaviour of a cylindrical lens. The user can determine the focal length of the lens and the angle formed by the optical axis and the focusing direction."
- Edit Programmable Function:** A dialog box with tabs for Basic Parameters, Physical Parameters, and Sampling. It features a left sidebar with icons for Geometry / Channels, Position / Orientation, Function, and Propagation Channels. The main area shows a Definition section with an 'Edit' button and a 'Validity: ✓' indicator. The Parameters section includes 'Angle' (0°) and 'FocalLength' (100 mm). A 'Help' button is located at the bottom right of the dialog, with a red arrow pointing to it.
- Snippet Help:** A separate window titled 'Snippet Help' that displays the following information:
 - Ideal Cylindrical Lens**
 - Version:** 1.0
 - Last Modified:** Wednesday, September 5, 2018
 - Description:** This snippet for a Programmable Function replicates the behaviour of a cylindrical lens. The user can determine the focal length of the lens and the angle formed by the focusing direction and the x axis of the incoming field.
 - Parameter Table:**

PARAMETER	DESCRIPTION
Angle	Represents the angle formed by the focusing direction and the x axis of the incoming field.
FocalLength	The focal length of the lens.

Programmable Interface: Writing the Code

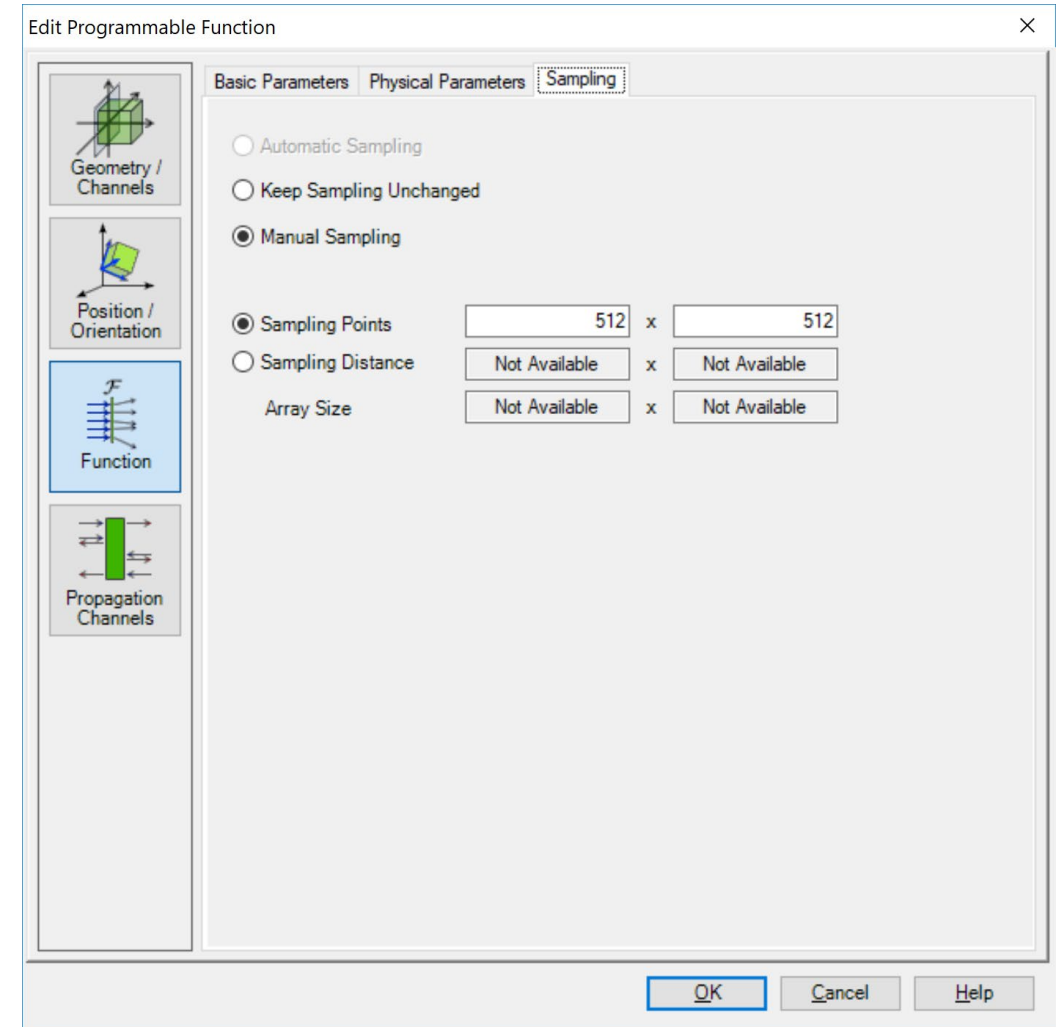


Default global parameters/variables

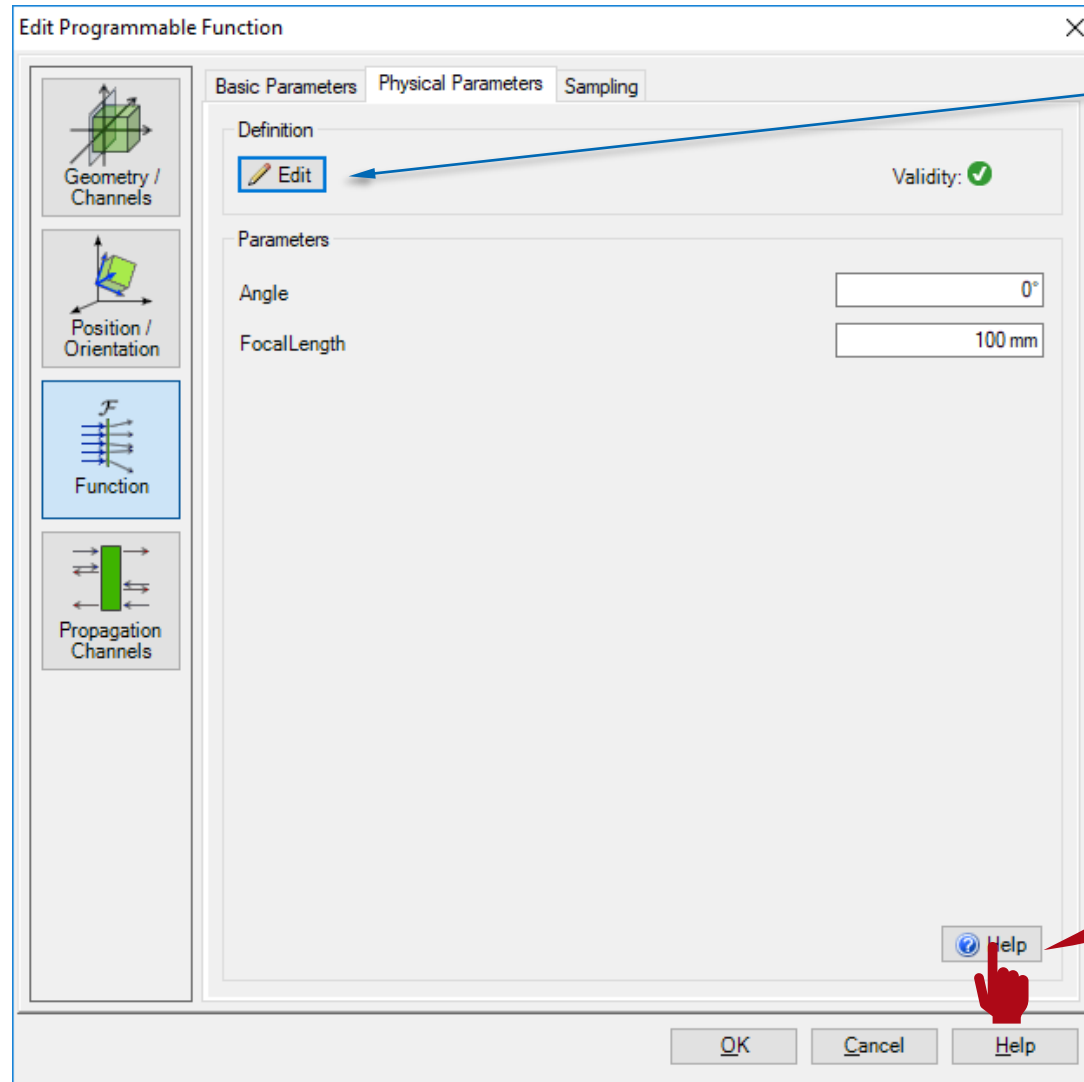
Global parameter defined by user in Global Parameters tab

Sampling

- Depending on the properties of the incoming field and the custom function, the user must determine the appropriate sampling in the Sampling tab.
- For instance, in the case of our cylindrical lens, and for an on-axis collimated incident beam, the sampling must be finer (higher number of sampling points) for a smaller focal length of the lens.

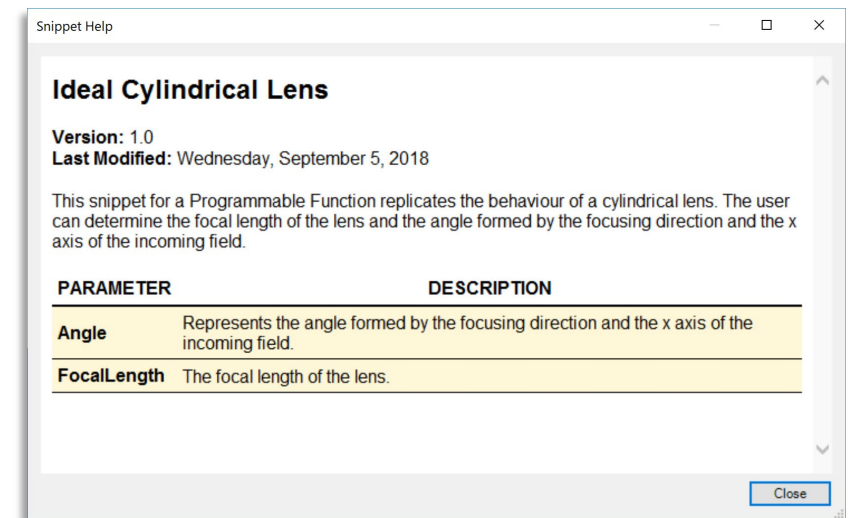


Programmable Interface: Using Your Snippet

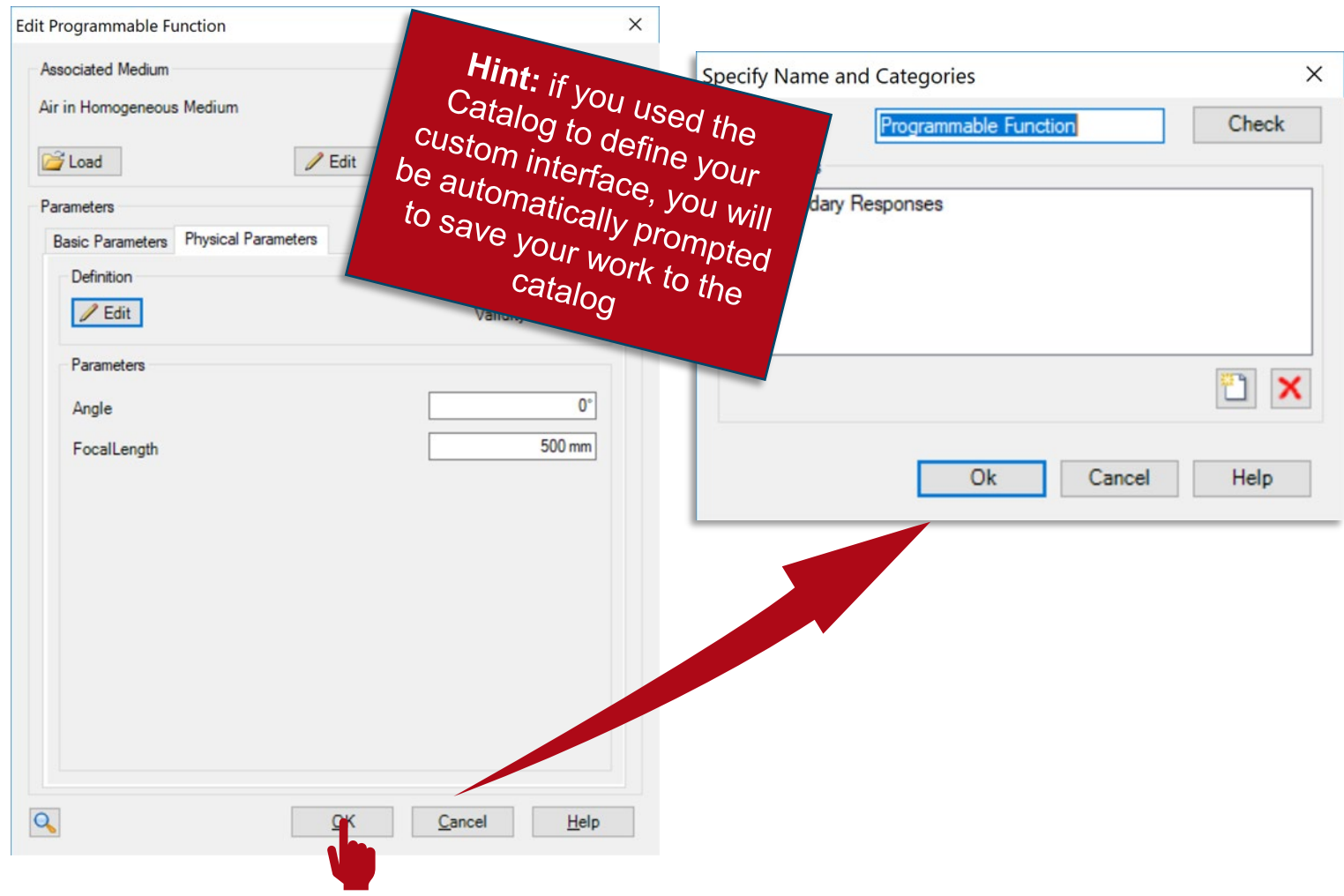


Modify your snippet by clicking on Edit

You can modify the value of the global parameters you defined here



Saving the Custom Function to the Catalog



Output of Programmable Function

Associated Medium
Air in Homogeneous Medium

Load Edit View

Parameters

Basic Parameters Physical Parameters

Definition
Edit

Parameters

Angle
FocalLength 100 mm

OK Cancel Help

1

Preview for Programmable Function

x y
View Range 1 mm 1 mm Show Complete Aperture

Wavelength 530 nm

Accuracy Factor 1

Re Im \mathcal{A} ϕ \mathcal{A}^2

2

Table Value at (x,y)

Phase of Boundary Response [rad]

Y [mm] X [mm]

3.1
0.014
-3.1

-0.4 -0.2 0 0.2 0.4

Close Help

The function is defined completely analytically by the code—full accuracy (up to double precision)

Visualization tool only available in Catalog definition mode!

Test the Code!

Main Function

```
int sign = Math.Sign(-FocalLength); // The sign of the focal length
// (convergent or divergent lens).
double k = Math.PI * 2.0 / Wavelength; // The wavenumber.

double phase = sign * k *
    Math.Sqrt(Math.Pow(x * Math.Cos(Angle) + y * Math.Sin(Angle), 2) +
        FocalLength * FocalLength); // Eq. 1

Complex c = Complex.Polar(1, phase); // Generate the complex-valued function,
// with phase-only modulation.

return c;
```

Document Information

title	How to Work with the Programmable Function in VirtualLab Fusion + Example: Cylindrical Lens
document code	CZT.0099
version	1.0
toolbox(es)	Starter Toolbox
VL version used for simulations	7.4.0.49
category	Feature Use Case
further reading	<ul style="list-style-type: none">- Customizable Help for Programmable Elements- Programming an Axicon Transmission Function