modesolverpy Documentation

Release 0.2.2

Jean-Luc Tambasco

Contents

1	Intro	oduction	•
2	Exar	mples	4
	2.1	Example 1	4
	2.2	Example 2	(
3	API	documentation	7
	3.1	Mode Solvers	7
	3.2	Pre-defined Structures	12
	3.3	Structure Creation	27
	3.4	Design Tools	3
	3.5	Coupling Efficiency	38
Pv	thon]	Module Index	4 !

Contents:

Contents 1

2 Contents

		- 4
CHA	PTF	R I

Introduction

This documentation demonstrates the potential of modesolverpy.

CHAPTER 2

Examples

Two example scripts.

2.1 Example 1

```
import modesolverpy.mode_solver as ms
import modesolverpy.structure as st
import numpy as np
# All units are relative. [um] were chosen in this case.
x_step = 0.02
y_step = 0.02
wg_height = 0.4
wg\_width = 0.5
sub\_height = 0.5
sub\_width = 2.
clad_height = 0.5
n_sub = 1.4
n_wg = 3.
n_{clad} = 1.
film_thickness = 0.5
wavelength = 1.55
angle = 75.
structure = st.RidgeWaveguide(wavelength,
                               x_step,
                               y_step,
                               wg_height,
                               wg_width,
                               sub_height,
                               sub_width,
                               clad_height,
                               n_sub,
```

(continues on next page)

(continued from previous page)

2.2 Example 2

```
import modesolverpy.mode_solver as ms
import modesolverpy.structure as st
import opticalmaterialspy as mat
import numpy as np
wl = 1.55
x_step = 0.06
y_step = 0.06
wg_height = 0.8
wg_width = 1.8
sub_height = 1.0
sub_width = 4.
clad_height = 1.0
film_thickness = 1.2
angle = 60.
def struct_func(n_sub, n_wg, n_clad):
    return st.RidgeWaveguide(wl, x_step, y_step, wg_height, wg_width,
                             sub_height, sub_width, clad_height,
                             n_sub, n_wg, angle, n_clad, film_thickness)
n_sub = mat.Si02().n(wl)
n_wg_xx = mat.Ktp('x').n(wl)
n_wg_yy = mat.Ktp('y').n(wl)
n_wg_zz = mat.Ktp('z').n(wl)
n_{clad} = mat.Air().n()
struct_xx = struct_func(n_sub, n_wq_xx, n_clad)
struct_yy = struct_func(n_sub, n_wg_yy, n_clad)
struct_zz = struct_func(n_sub, n_wg_zz, n_clad)
struct_ani = st.StructureAni(struct_xx, struct_yy, struct_zz)
struct_ani.write_to_file()
solver = ms.ModeSolverFullyVectorial(8)
solver.solve(struct_ani)
solver.write_modes_to_file()
solver.solve_ng(struct_ani, 0.01)
solver.solve_sweep_wavelength(struct_ani, np.linspace(1.501, 1.60, 21))
```

CHAPTER 3

API documentation

3.1 Mode Solvers

3.1.1 Classes

ModeSolverFullyVectorial(n_eigs[, tol,])	A fully-vectorial mode solver object used to setup and run a mode solving simulation.
ModeSolverSemiVectorial(n_eigs[, tol,])	A semi-vectorial mode solver object used to setup and
	run a mode solving simulation.

ModeSolverFullyVectorial

A fully-vectorial mode solver object used to setup and run a mode solving simulation.

- **n_eigs** (*int*) The number of eigen-values to solve for.
- tol (float) The precision of the eigen-value/eigen-vector solver. Default is 0.001.
- **boundary** (str) The boundary conditions to use. This is a string that identifies the type of boundary conditions applied. The following options are available: 'A' Hx is antisymmetric, Hy is symmetric, 'S' Hx is symmetric and, Hy is antisymmetric, and '0' Hx and Hy are zero immediately outside of the boundary. The string identifies all four boundary conditions, in the order: North, south, east, west. For example, boundary='000A'. Default is '0000'.
- initial_mode_guess (list) An initial mode guess for the modesolver.
- initial_n_eff_guess (list) An initial effective index guess for the modesolver.

Methods Summary

solve(structure)	Find the modes of a given structure.
<pre>solve_ng(structure[, wavelength_step, filename])</pre>	Solve for the group index, n_g , of a structure at a par-
	ticular wavelength.
solve_sweep_structure(structures,[,	Find the modes of many structures.
])	
solve_sweep_wavelength(structure, wave-	Solve for the effective indices of a fixed structure at
lengths)	different wavelengths.
write_modes_to_file([filename, plot,])	Writes the mode fields to a file and optionally plots
	them.

Methods Documentation

solve (structure)

Find the modes of a given structure.

Parameters structure (Structure) – The target structure to solve for modes.

Returns The 'n_effs' key gives the effective indices of the modes. The 'modes' key exists of mode profiles were solved for; in this case, it will return arrays of the mode profiles.

Return type dict

solve_ng (structure, wavelength_step=0.01, filename='ng.dat')

Solve for the group index, n_q , of a structure at a particular wavelength.

Parameters

- **structure** (Structure) The target structure to solve for modes.
- wavelength_step (float) The step to take below and above the nominal wavelength. This is used for approximating the gradient of $n_{\rm eff}$ at the nominal wavelength. Default is 0.01.
- **filename** (str) The nominal filename to use when saving the effective indices. Defaults to 'wavelength_n_effs.dat'.

Returns A list of the group indices found for each mode.

Return type list

Find the modes of many structures.

Parameters

- structures (list) A list of Structures to find the modes of.
- **sweep_param_list** (*list*) A list of the parameter-sweep sweep that was used. This is for plotting purposes only.
- **filename** (str) The nominal filename to use when saving the effective indices. Defaults to 'structure_n_effs.dat'.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Returns A list of the effective indices found for each structure.

Return type list

solve_sweep_wavelength (*structure*, *wavelengths*, *filename='wavelength_n_effs.dat'*, *plot=True*) Solve for the effective indices of a fixed structure at different wavelengths.

Parameters

- **structure** (Slabs) The target structure to solve for modes.
- wavelengths (list) A list of wavelengths to sweep over.
- **filename** (str) The nominal filename to use when saving the effective indices. Defaults to 'wavelength n effs.dat'.
- **plot** (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Returns A list of the effective indices found for each wavelength.

Return type list

Writes the mode fields to a file and optionally plots them.

Parameters

- **filename** (str) The nominal filename to use for the saved data. The suffix will be automatically be changed to identify each field and mode number. Default is 'mode.dat'
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.
- **fields_to_write** (tuple) A tuple of strings where the strings can be 'Ex', 'Ey', 'Ez', 'Hx', 'Hy' and 'Hz' defining what part of the mode should be saved and plotted. By default, all six components are written and plotted.

Returns A dictionary containing the effective indices and mode field profiles (if solved for).

Return type dict

ModeSolverSemiVectorial

```
class ModeSolverSemiVectorial (n\_eigs, tol=0.001, boundary='0000', mode\_profiles=True, initial\_mode\_guess=None, semi\_vectorial\_method='Ex')

Bases: modesolverpy.mode_solver._ModeSolver
```

A semi-vectorial mode solver object used to setup and run a mode solving simulation.

Parameters

- n_eigs (int) The number of eigen-values to solve for.
- tol (float) The precision of the eigen-value/eigen-vector solver. Default is 0.001.
- **boundary** (str) The boundary conditions to use. This is a string that identifies the type of boundary conditions applied. The following options are available: 'A' Hx is antisymmetric, Hy is symmetric, 'S' Hx is symmetric and, Hy is antisymmetric, and '0' Hx and Hy are zero immediately outside of the boundary. The string identifies all four boundary conditions, in the order: North, south, east, west. For example, boundary='000A'. Default is '0000'.
- mode_profiles (bool) True if the the mode-profiles should be found, 'False if only the effective indices should be found.
- initial_mode_guess (list) An initial mode guess for the modesolver.

3.1. Mode Solvers 9

• **semi_vectorial_method** (*str*) – Either 'Ex' or 'Ey'. If 'Ex', the mode solver will only find TE modes (horizontally polarised to the simulation window), if 'Ey', the mode solver will find TM modes (vertically polarised to the simulation window).

Methods Summary

solve(structure)	Find the modes of a given structure.
solve_ng(structure[, wavelength_step, filename])	Solve for the group index, n_g , of a structure at a par-
	ticular wavelength.
solve_sweep_structure(structures,[,	Find the modes of many structures.
])	
solve_sweep_wavelength(structure, wave-	Solve for the effective indices of a fixed structure at
lengths)	different wavelengths.
write_modes_to_file([filename, plot, anal-	Writes the mode fields to a file and optionally plots
yse])	them.

Methods Documentation

solve (structure)

Find the modes of a given structure.

Parameters structure (Structure) - The target structure to solve for modes.

Returns The 'n_effs' key gives the effective indices of the modes. The 'modes' key exists of mode profiles were solved for; in this case, it will return arrays of the mode profiles.

Return type dict

solve_ng (*structure*, *wavelength_step=0.01*, *filename='ng.dat'*)
Solve for the group index, n_a , of a structure at a particular wavelength.

Parameters

- **structure** (Structure) The target structure to solve for modes.
- wavelength_step (float) The step to take below and above the nominal wavelength. This is used for approximating the gradient of $n_{\rm eff}$ at the nominal wavelength. Default is 0.01.
- **filename** (str) The nominal filename to use when saving the effective indices. Defaults to 'wavelength_n_effs.dat'.

Returns A list of the group indices found for each mode.

Return type list

Find the modes of many structures.

- **structures** (list) A list of *Structures* to find the modes of.
- **sweep_param_list** (*list*) A list of the parameter-sweep sweep that was used. This is for plotting purposes only.
- **filename** (str) The nominal filename to use when saving the effective indices. Defaults to 'structure_n_effs.dat'.

• plot (bool) - True if plots should be generates, otherwise *False*. Default is *True*.

Returns A list of the effective indices found for each structure.

Return type list

solve_sweep_wavelength (*structure*, *wavelengths*, *filename='wavelength_n_effs.dat'*, *plot=True*) Solve for the effective indices of a fixed structure at different wavelengths.

Parameters

- **structure** (Slabs) The target structure to solve for modes.
- wavelengths (list) A list of wavelengths to sweep over.
- **filename** (str) The nominal filename to use when saving the effective indices. Defaults to 'wavelength_n_effs.dat'.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Returns A list of the effective indices found for each wavelength.

Return type list

write_modes_to_file (filename='mode.dat', plot=True, analyse=True)
Writes the mode fields to a file and optionally plots them.

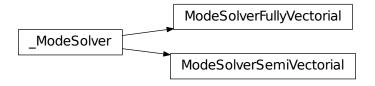
Parameters

- **filename** (*str*) The nominal filename to use for the saved data. The suffix will be automatically be changed to identify each mode number. Default is 'mode.dat'
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.
- analyse (bool) *True* if an analysis on the fundamental mode should be performed. The analysis adds to the plot of the fundamental mode the power mode-field diameter (MFD) and marks it on the output, and it marks with a cross the maximum E-field value. Default is *True*.

Returns A dictionary containing the effective indices and mode field profiles (if solved for).

Return type dict

3.1.2 Class Inheritance Diagram



3.1. Mode Solvers

3.2 Pre-defined Structures

3.2.1 Classes

RidgeWaveguide(wavelength, x_step, y_step,)	A general ridge waveguide structure.
Slab(name, x_step, y_step, x_max, y_max,)	A Slab represents a horizontal slice of the refractive
	index profile.
Slabs(wavelength, y_step, x_step, x_max[, x_min])	Class to implement device refractive index profile cross-
	section designs.
Structure(x_step, y_step, x_max, y_max[,])	
StructureAni(structure_xx, structure_yy,)	Anisottropic structure object.
$WgArray$ (wavelength, x_step, y_step,[,])	

RidgeWaveguide

A general ridge waveguide structure.

- wavelength (float) Wavelength the structure should operate at.
- $x_step(float)$ The grid step in x that the structure is created on.
- **y_step** (float) The grid step in y that the structure is created on.
- wg_height (float) The height of the ridge.
- wg_width (float) The width of the ridge.
- **sub_height** (*float*) The thickness of the substrate.
- **sub_width** (*float*) The width of the substrate.
- **clad_height** (*float*) The thickness of the cladding.
- n_sub(float, function) Refractive index of the substrate. Either a constant (float), or a function that accepts one parameters, the wavelength, and returns a float of the refractive index. This is useful when doing wavelength sweeps and solving for the group velocity. The function provided could be a Sellmeier equation.
- n_wg (float, function) Refractive index of the waveguide. Either a constant (float), or a function that accepts one parameters, the wavelength, and returns a float of the refractive index. This is useful when doing wavelength sweeps and solving for the group velocity. The function provided could be a Sellmeier equation.
- angle (float) The angle of the sidewall [degrees] of the waveguide. Default is 0 degrees (vertical sidewalls).
- n_clad (float, function) Refractive index of the cladding. Either a constant (float), or a function that accepts one parameters, the wavelength, and returns a float of the refractive index. This is useful when doing wavelength sweeps and solving for the group velocity. The function provided could be a Sellmeier equation. Default is air.
- **film_thickness** (*float*, *str*) The thickness of the film the waveguide is on. If the waveguide is a true ridge (fully etched), then the film thickness can be set to 'wg_height',

otherwise the waveguide is a rib waveguide, and a float should be given specifying the thickness of the film.

Attributes Summary

eps	np.array - A grid of permittivies representing the
	permittivity profile of the structure.
eps_func	function – a function that when passed a x and y val-
	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	np.array – The refractive index profile matrix of the
	current slab.
n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
X	<i>np.array</i> – The grid points in x.
x_ctr	float – The centre distance in x.
x_ctr x_pts	<i>int</i> – The number of grid points in x.
XC	np.array – The centre points of the x points.
xc_max	<i>float</i> – The maximum value of xc.
xc_min	<i>float</i> – The minimum value of xc.
xc_pts	int – The number of points in xc .
Y	<i>np.array</i> – The grid points in y.
y_ctr	float – The centre distance in y
y_ctr y_pts	<i>int</i> – The number of grid points in y.
ус	<i>np.array</i> – The centre points of the y points.
yc_max	float – The maximum value of yc.
yc_min	float – The minimum value of yc .
yc_pts	int – The number of points in yc .

Methods Summary

add_slab(height[, n_background])	Creates and adds a Slab object.
change_wavelength(wavelength)	Changes the wavelength of the structure.
write_to_file([filename, plot])	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps_func

function – a function that when passed a x and y values, returns the permittivity profile of the structure, interpolating if necessary.

n

np.array – The refractive index profile matrix of the current slab.

n func

function – a function that when passed a x and y values, returns the refractive index profile of the structure, interpolating if necessary.

```
x
     np.array – The grid points in x.
x_ctr
    float – The centre distance in x.
x pts
     int – The number of grid points in x.
хc
     np.array – The centre points of the x points.
xc max
    float – The maximum value of xc.
xc min
    float – The minimum value of xc.
xc_pts
     int – The number of points in xc.
     np.array – The grid points in y.
y_ctr
    float – The centre distance in y
y pts
     int – The number of grid points in y.
уc
     np.array – The centre points of the y points.
yc_max
    float – The maximum value of yc.
yc_min
    float – The minimum value of yc.
yc_pts
     int – The number of points in yc.
Methods Documentation
add_slab (height, n_background=1.0)
     Creates and adds a Slab object.
         Parameters
              • height (float) – Height of the slab.
              • n_background (float) - The nominal refractive index of the slab. Default is 1 (air).
         Returns The name of the slab.
         Return type str
change_wavelength (wavelength)
     Changes the wavelength of the structure.
     This will affect the mode solver and potentially the refractive indices used (provided functions were pro-
```

vided as refractive indices).

Parameters wavelength (float) – The new wavelength.

write_to_file (filename='material_index.dat', plot=True)

Write the refractive index profile to file.

Parameters

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Slab

class Slab (name, x_step , y_step , x_max , y_max , x_min , y_min , $n_background$, wavelength)

Bases: $modesolverpy.structure_base.Structure$

A Slab represents a horizontal slice of the refractive index profile.

A Slabs object composes many Slab objects. The more Slab are added, the more horizontal slices are added. A Slab has a chosen fixed height, and a background (nominal) refractive index. A slab can then be customised to include a desired design.

Parameters

- name (str) The name of the slab.
- **x_step** (*float*) The step in x.
- y_step (float) The step in y.
- **x_max** (float) The maximum x-value.
- y_max (float) The maximum y-value.
- **x_min** (float) The minimum x-value.
- **y_min** (float) The minimum x-value.
- n_background (float) The nominal refractive index.
- wavelength (float) The wavelength the structure operates at.

name

str – The name of the Slab object.

position

int – A unique identifier for the

:class:`Slab` object.

Attributes Summary

eps	np.array - A grid of permittivies representing the
	permittivity profile of the structure.
eps_func	function – a function that when passed a x and y val-
	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	np.array – A grid of refractive indices representing
	the refractive index profile of the structure.
	0 1 1

Continued on next page

Table 7 – continued from previous page

n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
position	
X	np.array – The grid points in x.
x_ctr	<i>float</i> – The centre distance in x.
x_pts	int – The number of grid points in x.
XC	<i>np.array</i> – The centre points of the x points.
xc_max	float – The maximum value of xc.
xc_min	float – The minimum value of xc.
xc_pts	int – The number of points in xc .
Y	np.array – The grid points in y.
y_ctr	float – The centre distance in y
y_pts	<i>int</i> – The number of grid points in y.
ус	np.array – The centre points of the y points.
yc_max	float – The maximum value of yc.
yc_min	float – The minimum value of yc.
yc_pts	int – The number of points in yc .

Methods Summary

<pre>add_material(x_min, x_max, n[, angle])</pre>	Add a refractive index between two x-points.
write_to_file([filename, plot])	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps func

function – a function that when passed a x and y values, returns the permittivity profile of the structure, interpolating if necessary.

n

np.array – A grid of refractive indices representing the refractive index profile of the structure.

n func

function – a function that when passed a x and y values, returns the refractive index profile of the structure, interpolating if necessary.

position = 0

X

np.array – The grid points in x.

x_ctr

float – The centre distance in x.

x_pts

int – The number of grid points in x.

ХC

np.array – The centre points of the x points.

```
xc max
     float – The maximum value of xc.
xc min
     float – The minimum value of xc.
xc pts
     int – The number of points in xc.
У
     np.array – The grid points in y.
y_ctr
     float – The centre distance in y
y_pts
     int – The number of grid points in y.
уc
     np.array – The centre points of the y points.
yc max
     float – The maximum value of yc.
yc_min
     float – The minimum value of yc.
yc pts
     int – The number of points in yc.
```

Methods Documentation

```
add_material (x_min, x_max, n, angle=0)
Add a refractive index between two x-points.
```

Parameters

- **x_min** (float) The start x-point.
- **x_max** (float) The stop x-point.
- **n** (*float*), *function*) Refractive index between *x_min* and *x_max*. Either a constant (*float*), or a function that accepts one parameters, the wavelength, and returns a float of the refractive index. This is useful when doing wavelength sweeps and solving for the group velocity. The function provided could be a Sellmeier equation.
- angle (float) Angle in degrees of the slope of the sidewalls at x_min and x_max . This is useful for defining a ridge with angled sidewalls.

```
write_to_file (filename='material_index.dat', plot=True)
Write the refractive index profile to file.
```

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Slabs

class Slabs (wavelength, y_step, x_step, x_max, x_min=0.0)

Bases: modesolverpy.structure_base._AbstractStructure

Class to implement device refractive index profile cross-section designs.

Slabs is a collection of Slab objects. Each slab has a fixed height (usually less than the maximum height of the desired simulation window), and is as wide as the simulation window.

Slabs objects can be index using [name] to return the various Slab objects. The bottom slab is returned first and so on up to the top slab.

Parameters

- wavelength (float) The wavelength the structure operates at.
- **y_step** (float) The step in y.
- **x_step** (float) The step in x.
- **x_max** (float) The maximum x-value.
- **x_min** (float) The minimum x-value. Default is 0.

slabs

dict – The key is the name of the slab, and the value is the Slab object.

slab count

int – The number of Slab objects added so far.

Attributes Summary

eps	np.array – A grid of permittivies representing the
	permittivity profile of the structure.
eps func	function – a function that when passed a x and y val-
1 —	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	<i>np.array</i> – The refractive index profile matrix of the
11	current slab.
n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
X	<i>np.array</i> – The grid points in x.
x_ctr	<i>float</i> – The centre distance in x.
x_pts	int – The number of grid points in x.
XC	<i>np.array</i> – The centre points of the x points.
xc_max	<i>float</i> – The maximum value of xc.
xc_min	<i>float</i> – The minimum value of <i>xc</i> .
xc_pts	int – The number of points in xc .
Y	np.array – The grid points in y.
y_ctr	float – The centre distance in y
y_pts	<i>int</i> – The number of grid points in y.
y_ctr y_pts yc	<i>np.array</i> – The centre points of the y points.
yc_max	<i>float</i> – The maximum value of <i>yc</i> .
yc_min	<i>float</i> – The minimum value of <i>yc</i> .

Continued on next page

Table 9 – continued from previous page

	· · · · · · · · · · · · · · · · · · ·
yc_pts	int – The number of points in yc .

Methods Summary

add_slab(height[, n_background])	Creates and adds a Slab object.
change_wavelength(wavelength)	Changes the wavelength of the structure.
write_to_file([filename, plot])	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps_func

function – a function that when passed a x and y values, returns the permittivity profile of the structure, interpolating if necessary.

n

np.array – The refractive index profile matrix of the current slab.

n_func

function – a function that when passed a x and y values, returns the refractive index profile of the structure, interpolating if necessary.

X

np.array – The grid points in x.

x_ctr

float – The centre distance in x.

x_pts

int – The number of grid points in x.

хc

np.array – The centre points of the x points.

xc_max

float – The maximum value of *xc*.

xc_min

float – The minimum value of *xc*.

xc_pts

int – The number of points in xc.

У

np.array – The grid points in y.

y_ctr

float – The centre distance in y

y_pts

int – The number of grid points in y.

уc

np.array – The centre points of the y points.

yc_max

float – The maximum value of yc.

yc_min

float – The minimum value of yc.

yc_pts

int – The number of points in yc.

Methods Documentation

add_slab (height, n_background=1.0)

Creates and adds a Slab object.

Parameters

- height (float) Height of the slab.
- n_background (float) The nominal refractive index of the slab. Default is 1 (air).

Returns The name of the slab.

Return type str

change_wavelength (wavelength)

Changes the wavelength of the structure.

This will affect the mode solver and potentially the refractive indices used (provided functions were provided as refractive indices).

Parameters wavelength (float) – The new wavelength.

write_to_file (filename='material_index.dat', plot=True)

Write the refractive index profile to file.

Parameters

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Structure

class Structure (*x_step*, *y_step*, *x_max*, *y_max*, *x_min=0.0*, *y_min=0.0*, *n_background=1.0*)

Bases: modesolverpy.structure_base._AbstractStructure

Attributes Summary

eps	np.array - A grid of permittivies representing the
	permittivity profile of the structure.
eps_func	function – a function that when passed a x and y val-
	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	np.array – A grid of refractive indices representing
	the refractive index profile of the structure.

Continued on next page

Table 11 – continued from previous page

n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
X	np.array – The grid points in x.
x_ctr	float – The centre distance in x.
x_pts	<i>int</i> – The number of grid points in x.
XC	<i>np.array</i> – The centre points of the x points.
xc_max	<i>float</i> – The maximum value of xc.
xc_min	<i>float</i> – The minimum value of <i>xc</i> .
xc_pts	int – The number of points in xc .
	<i>np.array</i> – The grid points in y.
y_ctr	float – The centre distance in y
y_pts	<i>int</i> – The number of grid points in y.
уC	np.array – The centre points of the y points.
yc_max	float – The maximum value of yc.
yc_min	<i>float</i> – The minimum value of <i>yc</i> .
yc_pts	int – The number of points in yc .

Methods Summary

<pre>write_to_file([filename, plot])</pre>	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps func

function – a function that when passed a x and y values, returns the permittivity profile of the structure, interpolating if necessary.

n

np.array – A grid of refractive indices representing the refractive index profile of the structure.

n_func

function – a function that when passed a x and y values, returns the refractive index profile of the structure, interpolating if necessary.

x

np.array – The grid points in x.

x_ctr

float – The centre distance in x.

x_pts

int – The number of grid points in x.

ХC

np.array – The centre points of the x points.

xc_max

float – The maximum value of xc.

xc min

float – The minimum value of xc.

xc_pts

int – The number of points in xc.

У

np.array – The grid points in y.

y_ctr

float – The centre distance in y

y_pts

int – The number of grid points in y.

уc

np.array – The centre points of the y points.

yc_max

float – The maximum value of *yc*.

yc_min

float – The minimum value of yc.

yc pts

int – The number of points in yc.

Methods Documentation

write_to_file (filename='material_index.dat', plot=True)

Write the refractive index profile to file.

Parameters

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

StructureAni

class StructureAni (structure_xx, structure_yy, structure_zz, structure_xy=None, structure_yx=None)
Bases: object

Anisottropic structure object.

This is used with the fully-vectorial simulation when an anisotropic material is being used.

The form of the refractive index is

$$n = \begin{bmatrix} n_{xx} & n_{xy} & 0 \\ n_{yx} & n_{yy} & 0 \\ 0 & 0 & n_{zz} \end{bmatrix}.$$

- **structure_xx** (Structure) The structure with refractive index, n_{xx} .
- **structure_yy** (Structure) The structure with refractive index, n_{yy} . Presumably the same structure as $structure_xx$, but with different refractive index parameters.
- **structure_zz** (Structure) The structure with refractive index, n_{zz} . Presumably the same structure as *structure_xx*, but with different refractive index parameters.

- **structure_xy** (*None*, Structure) The structure with refractive index, n_{yx} . Presumably the same structure as *structure_xx*, but with different refractive index parameters. Default is *None*.
- **structure_yx** (*None*, Structure) The structure with refractive index, n_{yx} . Presumably the same structure as *structure_xx*, but with different refractive index parameters. Default is *None*.

Attributes Summary

eps	
eps_func	
n	
n_func	
X	
x_ctr	
x_pts	
x_step	
XC	
xc_max	
xc_min	
xc_pts	
y	
y_ctr	
y_pts	
y_step	
yc	
yc_max	
yc_min	
yc_pts	

Methods Summary

change_wavelength(wavelength)	Changes the wavelength of the structure.
<pre>write_to_file([filename, plot])</pre>	Write the refractive index profile to file.

Attributes Documentation

eps

eps_func

n

n_func

x

x_ctr

x_pts

x_step

хc

xc_max

xc_min

xc_pts

У

y_ctr

y_pts

y_step

yс

yc_max

yc_min

yc_pts

Methods Documentation

change_wavelength (wavelength)

Changes the wavelength of the structure.

This will affect the mode solver and potentially the refractive indices used (provided functions were provided as refractive indices).

Parameters wavelength (float) – The new wavelength.

write_to_file (filename='material_index.dat', plot=True)

Write the refractive index profile to file.

Parameters

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

WgArray

class WgArray (wavelength, x_step, y_step, wg_height, wg_widths, wg_gaps, sub_height, sub_width, clad_height, n_sub, n_wg, angle=0, n_clad=1.0)

Bases: modesolverpy.structure_base.Slabs

Attributes Summary

eps	np.array – A grid of permittivies representing the
	permittivity profile of the structure.
eps_func	function – a function that when passed a x and y val-
	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	np.array – The refractive index profile matrix of the
	current slab.
	0 1' 1

Continued on next page

Table 15 – continued from previous page

	mana na aantimaaa mann promoto panga
n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
X	<i>np.array</i> – The grid points in x.
x_ctr	float – The centre distance in x.
x_pts	<i>int</i> – The number of grid points in x.
XC	<i>np.array</i> – The centre points of the x points.
xc_max	<i>float</i> – The maximum value of xc.
xc_min	<i>float</i> – The minimum value of <i>xc</i> .
xc_pts	int – The number of points in xc .
Y	<i>np.array</i> – The grid points in y.
y_ctr	float – The centre distance in y
y_pts	<i>int</i> – The number of grid points in y.
УC	<i>np.array</i> – The centre points of the y points.
yc_max	float – The maximum value of yc.
yc_min	<i>float</i> – The minimum value of <i>yc</i> .
yc_pts	int – The number of points in yc .

Methods Summary

add_slab(height[, n_background])	Creates and adds a Slab object.
change_wavelength(wavelength)	Changes the wavelength of the structure.
write_to_file([filename, plot])	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps func

function – a function that when passed a x and y values, returns the permittivity profile of the structure, interpolating if necessary.

n

np.array – The refractive index profile matrix of the current slab.

n func

function – a function that when passed a x and y values, returns the refractive index profile of the structure, interpolating if necessary.

x

np.array – The grid points in x.

x_ctr

float – The centre distance in x.

x_pts

int – The number of grid points in x.

хc

np.array – The centre points of the x points.

xc max

float – The maximum value of *xc*.

```
xc min
    float – The minimum value of xc.
xc_pts
     int – The number of points in xc.
У
     np.array – The grid points in y.
y_ctr
    float – The centre distance in y
y_pts
     int – The number of grid points in y.
уc
     np.array – The centre points of the y points.
yc_max
    float – The maximum value of yc.
yc min
     float – The minimum value of yc.
yc_pts
     int – The number of points in yc.
Methods Documentation
add_slab (height, n_background=1.0)
     Creates and adds a Slab object.
         Parameters
             • height (float) - Height of the slab.
             • n_background (float) - The nominal refractive index of the slab. Default is 1 (air).
         Returns The name of the slab.
         Return type str
change_wavelength (wavelength)
```

Changes the wavelength of the structure.

This will affect the mode solver and potentially the refractive indices used (provided functions were provided as refractive indices).

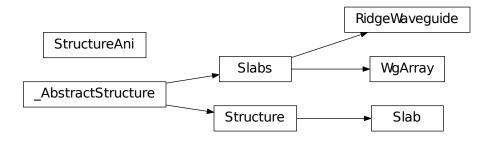
Parameters wavelength (float) – The new wavelength.

```
write_to_file (filename='material_index.dat', plot=True)
```

Write the refractive index profile to file.

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

3.2.2 Class Inheritance Diagram



3.3 Structure Creation

3.3.1 Classes

Slab(name, x_step, y_step, x_max, y_max,)	A Slab represents a horizontal slice of the refractive index profile.
Slabs(wavelength, y_step, x_step, x_max[, x_min])	Class to implement device refractive index profile cross- section designs.
Structure(x_step, y_step, x_max, y_max[,])	
StructureAni(structure_xx, structure_yy,)	Anisottropic structure object.

Slab

 $\textbf{class Slab} \ (name, x_step, y_step, x_max, y_max, x_min, y_min, n_background, wavelength)$

Bases: modesolverpy.structure_base.Structure

A Slab represents a horizontal slice of the refractive index profile.

A Slabs object composes many Slab objects. The more Slab are added, the more horizontal slices are added. A Slab has a chosen fixed height, and a background (nominal) refractive index. A slab can then be customised to include a desired design.

Parameters

- name (str) The name of the slab.
- **x_step** (float) The step in x.
- **y_step** (*float*) The step in y.
- **x_max** (float) The maximum x-value.
- y_max (float) The maximum y-value.
- **x_min** (*float*) The minimum x-value.
- **y_min** (float) The minimum x-value.

3.3. Structure Creation 27

- n_background (float) The nominal refractive index.
- wavelength (float) The wavelength the structure operates at.

name

str – The name of the *Slab* object.

position

int – A unique identifier for the

:class:`Slab` object.

Attributes Summary

eps	np.array – A grid of permittivies representing the
1	permittivity profile of the structure.
eps_func	function – a function that when passed a x and y val-
. –	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	np.array – A grid of refractive indices representing
	the refractive index profile of the structure.
n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
position	
X	np.array – The grid points in x.
x_ctr	float – The centre distance in x.
x_pts	<i>int</i> – The number of grid points in x.
XC	<i>np.array</i> – The centre points of the x points.
xc_max	<i>float</i> – The maximum value of <i>xc</i> .
xc_min	float – The minimum value of xc .
xc_pts	int – The number of points in xc .
Y	<i>np.array</i> – The grid points in y.
y_ctr	float – The centre distance in y
y_pts	<i>int</i> – The number of grid points in y.
уC	np.array – The centre points of the y points.
yc_max	float – The maximum value of yc.
yc_min	<i>float</i> – The minimum value of <i>yc</i> .
yc_pts	int – The number of points in yc .

Methods Summary

<pre>add_material(x_min, x_max, n[, angle])</pre>	Add a refractive index between two x-points.
<pre>write_to_file([filename, plot])</pre>	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps_func

function – a function that when passed a x and y values, returns the permittivity profile of the structure,

```
interpolating if necessary.
n
     np.array – A grid of refractive indices representing the refractive index profile of the structure.
n func
     function – a function that when passed a x and y values, returns the refractive index profile of the structure,
     interpolating if necessary.
position = 0
     np.array – The grid points in x.
x ctr
     float – The centre distance in x.
x_pts
     int – The number of grid points in x.
хc
     np.array – The centre points of the x points.
xc max
     float – The maximum value of xc.
xc min
     float – The minimum value of xc.
xc_pts
     int – The number of points in xc.
У
     np.array – The grid points in y.
y_ctr
     float - The centre distance in y
y_pts
     int – The number of grid points in y.
уc
     np.array – The centre points of the y points.
yc_max
     float – The maximum value of yc.
yc_min
     float – The minimum value of yc.
yc_pts
     int – The number of points in yc.
Methods Documentation
add_material(x_min, x_max, n, angle=0)
     Add a refractive index between two x-points.
         Parameters
              • x_min (float) - The start x-point.
```

3.3. Structure Creation 29

• **x_max** (float) – The stop x-point.

- n (float, function) Refractive index between x_min and x_max. Either a constant (float), or a function that accepts one parameters, the wavelength, and returns a float of the refractive index. This is useful when doing wavelength sweeps and solving for the group velocity. The function provided could be a Sellmeier equation.
- **angle** (float) Angle in degrees of the slope of the sidewalls at x_min and x_max . This is useful for defining a ridge with angled sidewalls.

 ${\tt write_to_file}~(filename='material_index.dat', plot=True)$

Write the refractive index profile to file.

Parameters

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Slabs

class Slabs (wavelength, y_step, x_step, x_max, x_min=0.0)

Bases: modesolverpy.structure_base._AbstractStructure

Class to implement device refractive index profile cross-section designs.

Slabs is a collection of Slab objects. Each slab has a fixed height (usually less than the maximum height of the desired simulation window), and is as wide as the simulation window.

Slabs objects can be index using [name] to return the various Slab objects. The bottom slab is returned first and so on up to the top slab.

Parameters

- wavelength (float) The wavelength the structure operates at.
- y_step (float) The step in y.
- **x_step** (float) The step in x.
- **x**_max (float) The maximum x-value.
- **x_min** (*float*) The minimum x-value. Default is 0.

slabs

dict – The key is the name of the slab, and the value is the Slab object.

slab count

int – The number of Slab objects added so far.

Attributes Summary

eps	np.array - A grid of permittivies representing the
	permittivity profile of the structure.
eps_func	function – a function that when passed a x and y val-
	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	np.array – The refractive index profile matrix of the
	current slab.

Continued on next page

Table 20 – continued from previous page

n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
X	np.array – The grid points in x.
x_ctr	float – The centre distance in x.
x_pts	<i>int</i> – The number of grid points in x.
XC	<i>np.array</i> – The centre points of the x points.
xc_max	<i>float</i> – The maximum value of xc.
xc_min	<i>float</i> – The minimum value of <i>xc</i> .
xc_pts	int – The number of points in xc .
	<i>np.array</i> – The grid points in y.
y_ctr	float – The centre distance in y
y_pts	<i>int</i> – The number of grid points in y.
уC	np.array – The centre points of the y points.
yc_max	float – The maximum value of yc.
yc_min	float – The minimum value of yc.
yc_pts	<i>int</i> – The number of points in <i>yc</i> .

Methods Summary

add_slab(height[, n_background])	Creates and adds a Slab object.
change_wavelength(wavelength)	Changes the wavelength of the structure.
write_to_file([filename, plot])	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps func

function – a function that when passed a x and y values, returns the permittivity profile of the structure, interpolating if necessary.

n

np.array – The refractive index profile matrix of the current slab.

n func

function – a function that when passed a x and y values, returns the refractive index profile of the structure, interpolating if necessary.

x

np.array – The grid points in x.

x_ctr

float – The centre distance in x.

x_pts

int – The number of grid points in x.

хc

np.array – The centre points of the x points.

xc max

float – The maximum value of *xc*.

3.3. Structure Creation 31

```
xc min
    float – The minimum value of xc.
xc_pts
     int – The number of points in xc.
У
     np.array – The grid points in y.
y_ctr
    float – The centre distance in y
y_pts
     int – The number of grid points in y.
уc
     np.array – The centre points of the y points.
yc_max
    float – The maximum value of yc.
yc min
     float – The minimum value of yc.
yc_pts
     int – The number of points in yc.
Methods Documentation
add_slab (height, n_background=1.0)
     Creates and adds a Slab object.
```

- height (float) Height of the slab.
- n_background (float) The nominal refractive index of the slab. Default is 1 (air).

Returns The name of the slab.

Return type str

Parameters

change_wavelength (wavelength)

Changes the wavelength of the structure.

This will affect the mode solver and potentially the refractive indices used (provided functions were provided as refractive indices).

Parameters wavelength (float) – The new wavelength.

```
write_to_file (filename='material_index.dat', plot=True)
```

Write the refractive index profile to file.

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

Structure

class Structure (*x_step*, *y_step*, *x_max*, *y_max*, *x_min=0.0*, *y_min=0.0*, *n_background=1.0*)

Bases: modesolverpy.structure_base._AbstractStructure

Attributes Summary

eps	np.array – A grid of permittivies representing the
	permittivity profile of the structure.
eps_func	function – a function that when passed a x and y val-
	ues, returns the permittivity profile of the structure,
	interpolating if necessary.
n	np.array – A grid of refractive indices representing
	the refractive index profile of the structure.
n_func	function – a function that when passed a x and y val-
	ues, returns the refractive index profile of the struc-
	ture, interpolating if necessary.
X	<i>np.array</i> – The grid points in x.
x_ctr	float – The centre distance in x.
x_pts	<i>int</i> – The number of grid points in x.
XC	<i>np.array</i> – The centre points of the x points.
xc_max	<i>float</i> – The maximum value of xc.
xc_min	<i>float</i> – The minimum value of xc.
xc_pts	int – The number of points in xc .
Y	<i>np.array</i> – The grid points in y.
y_ctr	float – The centre distance in y
y_ctr y_pts	<i>int</i> – The number of grid points in y.
ус	<i>np.array</i> – The centre points of the y points.
yc_max	float – The maximum value of yc.
yc_min	float – The minimum value of yc .
yc_pts	<i>int</i> – The number of points in <i>yc</i> .

Methods Summary

write_to_file([filename, plot])	Write the refractive index profile to file.

Attributes Documentation

eps

np.array – A grid of permittivies representing the permittivity profile of the structure.

eps_func

function – a function that when passed a x and y values, returns the permittivity profile of the structure, interpolating if necessary.

n

np.array – A grid of refractive indices representing the refractive index profile of the structure.

n func

function – a function that when passed a x and y values, returns the refractive index profile of the structure, interpolating if necessary.

3.3. Structure Creation 33

```
x
           np.array – The grid points in x.
     x_ctr
          float – The centre distance in x.
     x pts
           int – The number of grid points in x.
     хc
           np.array – The centre points of the x points.
     xc max
          float – The maximum value of xc.
     xc min
          float – The minimum value of xc.
     xc_pts
           int – The number of points in xc.
           np.array – The grid points in y.
     y_ctr
          float – The centre distance in y
     y pts
           int – The number of grid points in y.
     уc
           np.array – The centre points of the y points.
     yc_max
          float – The maximum value of yc.
     yc_min
          float – The minimum value of yc.
     yc_pts
           int – The number of points in yc.
     Methods Documentation
     write_to_file (filename='material_index.dat', plot=True)
           Write the refractive index profile to file.
               Parameters
                   • filename (str) – The nominal filename the refractive index data should be saved to.
                   • plot (bool) – True if plots should be generates, otherwise False. Default is True.
StructureAni
class StructureAni (structure_xx, structure_yy, structure_zz, structure_xy=None, structure_yx=None)
     Bases: object
```

This is used with the fully-vectorial simulation when an anisotropic material is being used.

Anisottropic structure object.

The form of the refractive index is

$$n = \begin{bmatrix} n_{xx} & n_{xy} & 0 \\ n_{yx} & n_{yy} & 0 \\ 0 & 0 & n_{zz} \end{bmatrix}.$$

Parameters

- **structure_xx** (Structure) The structure with refractive index, n_{xx} .
- **structure_yy** (Structure) The structure with refractive index, n_{yy} . Presumably the same structure as $structure_xx$, but with different refractive index parameters.
- **structure_zz** (Structure) The structure with refractive index, n_{zz} . Presumably the same structure as *structure_xx*, but with different refractive index parameters.
- **structure_xy** (*None*, Structure) The structure with refractive index, n_{yx} . Presumably the same structure as *structure_xx*, but with different refractive index parameters. Default is *None*.
- **structure_yx** (*None*, Structure) The structure with refractive index, n_{yx} . Presumably the same structure as *structure_xx*, but with different refractive index parameters. Default is *None*.

Attributes Summary

ps	
ps_func	
_func	
<u>_ctr</u>	
<u>_pts</u>	
<u>_step</u>	
TC .	
c_max	
c_min	
cc_pts	
7	
_ctr _pts	
<u>_pts</u>	
z_step	
7C	
rc_max	
rc_min	
rc_pts	

Methods Summary

change_wavelength(wavelength)	Changes the wavelength of the structure.
write_to_file([filename, plot])	Write the refractive index profile to file.

3.3. Structure Creation 35

Attributes Documentation

```
eps
eps_func
n_func
x
x_ctr
x_pts
x_step
ХC
xc_max
xc_min
xc_pts
У
y_ctr
y_pts
y_step
уc
yc_max
yc_min
yc_pts
```

Methods Documentation

change_wavelength (wavelength)

Changes the wavelength of the structure.

This will affect the mode solver and potentially the refractive indices used (provided functions were provided as refractive indices).

Parameters wavelength (float) – The new wavelength.

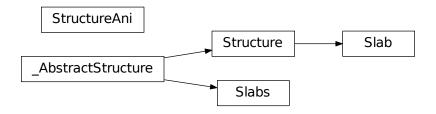
```
write_to_file (filename='material_index.dat', plot=True)
```

Write the refractive index profile to file.

Parameters

- **filename** (str) The nominal filename the refractive index data should be saved to.
- plot (bool) *True* if plots should be generates, otherwise *False*. Default is *True*.

3.3.2 Class Inheritance Diagram



3.4 Design Tools

3.4.1 Functions

directional_coupler_lc(wavelength_nm,)		Calculates the coherence length (100% power transfer)
		of a directional coupler.
grating_coupler_period(wavelength,	n_eff,	Calculate the period needed for a grating coupler.
)		

directional coupler lc

directional_coupler_lc (wavelength_nm, n_eff_1, n_eff_2)

Calculates the coherence length (100% power transfer) of a directional coupler.

Parameters

- wavelength_nm (float) The wavelength in [nm] the directional coupler should operate at.
- n_eff_1 (float) n_eff of the fundamental (even) supermode of the directional coupler.
- n_eff_2 (float) n_eff of the first-order (odd) supermode of the directional coupler.

Returns The length [um] the directional coupler needs to be to achieve 100% power transfer.

Return type float

grating coupler period

grating_coupler_period (wavelength, n_eff, n_clad, incidence_angle_deg, diffration_order=1)
Calculate the period needed for a grating coupler.

Parameters

- wavelength (float) The target wavelength for the grating coupler.
- n_eff (float) The effective index of the mode of a waveguide with the width of the grating coupler.

3.4. Design Tools 37

- **n_clad** (*float*) The refractive index of the cladding.
- **incidence_angle_deg** (*float*) The incidence angle the grating coupler should operate at [degrees].
- **diffration_order** (*int*) The grating order the coupler should work at. Default is 1st order (1).

Returns The period needed for the grating coupler in the same units as the wavelength was given at.

Return type float

3.5 Coupling Efficiency

3.5.1 Functions

<pre>coupling_efficiency(mode_solver, fibre_mfd)</pre>	Finds the coupling efficiency between a solved fundamental mode and a fibre of given MFD.
reflection(n1, n2)	Calculate the power reflection at the interface of two refractive index materials.
transmission(n1, n2)	Calculate the power transmission at the interface of two refractive index materials.

coupling_efficiency

coupling_efficiency (*mode_solver*, *fibre_mfd*, *fibre_offset_x=0*, *fibre_offset_y=0*, *n_eff_fibre=1.441*) Finds the coupling efficiency between a solved fundamental mode and a fibre of given MFD.

Parameters

- mode_solver (_ModeSolver) Mode solver that has found a fundamental mode.
- **fibre_mfd** (float) The mode-field diameter (MFD) of the fibre.
- **fibre_offset_x** (float) Offset the fibre from the centre position of the window in x. Default is 0 (no offset).
- **fibre_offset_y** (float) Offset the fibre from the centre position of the window in y. Default is 0 (no offset).
- n_eff_fibre (float) The effective index of the fibre mode. Default is 1.441.

Returns The power coupling efficiency.

Return type float

reflection

reflection(n1, n2)

Calculate the power reflection at the interface of two refractive index materials.

Parameters

- n1 (float) Refractive index of material 1.
- **n2** (float) Refractive index of material 2.

Returns The percentage of reflected power.

Return type float

transmission

transmission(n1, n2)

Calculate the power transmission at the interface of two refractive index materials.

Parameters

- **n1** (float) Refractive index of material 1.
- **n2** (*float*) Refractive index of material 2.

Returns The percentage of transmitted power.

Return type float

Python Module Index

m

modesolverpy.coupling_efficiency, 38
modesolverpy.design, 37
modesolverpy.mode_solver, 7
modesolverpy.structure, 12
modesolverpy.structure_base, 27

42 Python Module Index

Index

A add_material() (Slab method), 17, 29 add_slab() (RidgeWaveguide method), 14 add_slab() (Slabs method), 20, 32 add_slab() (WgArray method), 26 C	modesolverpy.coupling_efficiency (module), 38 modesolverpy.design (module), 37 modesolverpy.mode_solver (module), 7 modesolverpy.structure (module), 12 modesolverpy.structure_base (module), 27 ModeSolverSemiVectorial (class in modesolverpy.mode_solver), 9	
change_wavelength() (RidgeWaveguide method), 14 change_wavelength() (Slabs method), 20, 32 change_wavelength() (StructureAni method), 24, 36 change_wavelength() (WgArray method), 26 coupling_efficiency() (in module modes- olverpy.coupling_efficiency), 38	N n (RidgeWaveguide attribute), 13 n (Slab attribute), 16, 29 n (Slabs attribute), 19, 31 n (Structure attribute), 21, 33 n (StructureAni attribute), 23, 36	
D directional_coupler_lc() (in module modes- olverpy.design), 37 E eps (RidgeWaveguide attribute), 13 eps (Slab attribute), 16, 28 eps (Slabs attribute), 19, 31	n (WgArray attribute), 25 n_func (RidgeWaveguide attribute), 13 n_func (Slab attribute), 16, 29 n_func (Slabs attribute), 19, 31 n_func (Structure attribute), 21, 33 n_func (StructureAni attribute), 23, 36 n_func (WgArray attribute), 25 name (Slab attribute), 15, 28	
eps (Structure attribute), 21, 33 eps (StructureAni attribute), 23, 36 eps (WgArray attribute), 25 eps_func (RidgeWaveguide attribute), 13 eps_func (Slab attribute), 16, 28 eps_func (Slabs attribute), 19, 31 eps_func (Structure attribute), 21, 33 eps_func (StructureAni attribute), 23, 36	P position (Slab attribute), 15, 16, 28, 29 R reflection() (in module modesolverpy.coupling_efficiency), 38 RidgeWaveguide (class in modesolverpy.structure), 12	
eps_func (WgArray attribute), 25 G grating_coupler_period() (in module modes- olverpy.design), 37 M	Slab (class in modesolverpy.structure), 15 Slab (class in modesolverpy.structure_base), 27 slab_count (Slabs attribute), 18, 30 Slabs (class in modesolverpy.structure), 18 Slabs (class in modesolverpy.structure_base), 30 slabs (Slabs attribute), 18, 30	
ModeSolverFullyVectorial (class in modes- olverpy.mode_solver), 7	solve() (ModeSolverFullyVectorial method), 8	

solve() (ModeSolverSemiVectorial method), 10	xc (Slab attribute), 16, 29
solve_ng() (ModeSolverFullyVectorial method), 8	xc (Slabs attribute), 19, 31
solve_ng() (ModeSolverSemiVectorial method), 10	xc (Structure attribute), 21, 34
solve_sweep_structure() (ModeSolverFullyVectorial	xc (StructureAni attribute), 23, 36
method), 8	xc (WgArray attribute), 25
solve_sweep_structure() (ModeSolverSemiVectorial	xc_max (RidgeWaveguide attribute), 14
method), 10	xc_max (Slab attribute), 16, 29
solve_sweep_wavelength() (ModeSolverFullyVectorial	xc_max (Slabs attribute), 19, 31
method), 8	xc_max (Structure attribute), 21, 34
solve_sweep_wavelength() (ModeSolverSemiVectorial	xc_max (StructureAni attribute), 24, 36
method), 11	xc_max (WgArray attribute), 25
Structure (class in modesolverpy.structure), 20	xc_min (RidgeWaveguide attribute), 14
Structure (class in modesolverpy.structure_base), 33	xc_min (Slab attribute), 17, 29
StructureAni (class in modesolverpy.structure), 22	xc_min (Slabs attribute), 19, 31
StructureAni (class in modesolverpy.structure_base), 34	xc_min (Structure attribute), 21, 34
Т	xc_min (StructureAni attribute), 24, 36
	xc_min (WgArray attribute), 25
transmission() (in module modes-	xc_pts (RidgeWaveguide attribute), 14
olverpy.coupling_efficiency), 39	xc_pts (Slab attribute), 17, 29
W	xc_pts (Slabs attribute), 19, 32
VV	xc_pts (Structure attribute), 21, 34
WgArray (class in modesolverpy.structure), 24	xc_pts (StructureAni attribute), 24, 36
write_modes_to_file() (ModeSolverFullyVectorial	xc_pts (WgArray attribute), 26
method), 9	Υ
write_modes_to_file() (ModeSolverSemiVectorial	
method), 11	y (RidgeWaveguide attribute), 14
write_to_file() (RidgeWaveguide method), 15	y (Slab attribute), 17, 29
write_to_file() (Slab method), 17, 30	y (Slabs attribute), 19, 32
write_to_file() (Slabs method), 20, 32	y (Structure attribute), 22, 34
write_to_file() (Structure method), 22, 34	y (StructureAni attribute), 24, 36
write_to_file() (StructureAni method), 24, 36	y (WgArray attribute), 26
write_to_file() (WgArray method), 26	y_ctr (RidgeWaveguide attribute), 14
V	y_ctr (Slab attribute), 17, 29
X	y_ctr (Slabs attribute), 19, 32
x (RidgeWaveguide attribute), 13	y_ctr (Structure attribute), 22, 34
x (Slab attribute), 16, 29	y_ctr (StructureAni attribute), 24, 36
x (Slabs attribute), 19, 31	y_ctr (WgArray attribute), 26
x (Structure attribute), 21, 33	y_pts (RidgeWaveguide attribute), 14
x (StructureAni attribute), 23, 36	y_pts (Slab attribute), 17, 29
x (WgArray attribute), 25	y_pts (Slabs attribute), 19, 32
x_ctr (RidgeWaveguide attribute), 14	y_pts (Structure attribute), 22, 34
x_ctr (Slab attribute), 16, 29	y_pts (StructureAni attribute), 24, 36
x_ctr (Slabs attribute), 19, 31	y_pts (WgArray attribute), 26
x_ctr (Structure attribute), 21, 34	y_step (StructureAni attribute), 24, 36
x_ctr (StructureAni attribute), 23, 36	yc (RidgeWaveguide attribute), 14
x_ctr (WgArray attribute), 25	yc (Slab attribute), 17, 29
x_pts (RidgeWaveguide attribute), 14	yc (Slabs attribute), 19, 32
x_pts (Slab attribute), 16, 29	yc (Structure attribute), 22, 34
x_pts (Slabs attribute), 19, 31	yc (StructureAni attribute), 24, 36
x_pts (Structure attribute), 21, 34	yc (WgArray attribute), 26
x_pts (StructureAni attribute), 23, 36	yc_max (RidgeWaveguide attribute), 14
x_pts (WgArray attribute), 25	yc_max (Slab attribute), 17, 29
x_step (StructureAni attribute), 23, 36	yc_max (Slabs attribute), 19, 32
vc (Ridge Wayaguide attribute) 14	vc max (Structure attribute), 22, 34

44 Index

yc_max (StructureAni attribute), 24, 36 yc_max (WgArray attribute), 26 yc_min (RidgeWaveguide attribute), 14 yc_min (Slab attribute), 17, 29 yc_min (Slabs attribute), 20, 32 yc_min (Structure attribute), 22, 34 yc_min (StructureAni attribute), 24, 36 yc_min (WgArray attribute), 26 yc_pts (RidgeWaveguide attribute), 14 yc_pts (Slab attribute), 17, 29 yc_pts (Slabs attribute), 20, 32 yc_pts (Structure attribute), 22, 34 yc_pts (Structure attribute), 22, 34 yc_pts (StructureAni attribute), 24, 36 yc_pts (WgArray attribute), 26

Index 45