GaussOpt Documentation

Release 0.1.dev1

John Garrett

Oct 12, 2017

Contents:

1 gaussopt package **3**
- 1.1 Submodules . . . 3
- 1.2 gaussopt.component module . . . 3
- 1.3 gaussopt.frequency module . . . 6
- 1.4 gaussopt.system module . . . 7
- 1.5 gaussopt.util module . . . 8
- 1.6 Module contents . . . 9

2 Gaussopt Example **11**
- 2.1 Define frequency sweep . . . 11
- 2.2 Define horns . . . 12
- 2.3 Define optical components . . . 12
- 2.4 Build Optical System . . . 13
- 2.5 Plot Coupling . . . 13
- 2.6 Plot Beam Propagation . . . 13

3 Indices and tables **15**

Python Module Index **17**

Index **19**

Analyze quasi-optical systems using Gaussian beam analysis

Quasi-optical analysis is important whenever the wavelength is comparable to the size of the optical components. Gaussian beam analysis of quasi-optical systems assumes that the transverse amplitude profile (the E- or H-field) of the beam is similar to a Gaussian function. This is roughly true for beams originating from waveguide horn antennas.

Author: John Garrett (garrettj403@gmail.com)

License: MIT

Language: Python 2.7

Version: 0.1

Release: 0.1.dev1

On PyPi: https://pypi.python.org/pypi/GaussOpt

On GitHub: https://github.com/garrettj403/GaussOpt/

CHAPTER 1

gaussopt package

1.1 Submodules

1.2 gaussopt.component module

class gaussopt.component.**Component**(*matrix=None*, ***kwargs*)
Bases: object

Base-class for a generic component in a Gaussian optical system.

To get initialization information, see __init__().

matrix
ndarray – beam transformation matrix, 2x2

Component constructor.

Parameters

- **matrix** (*ndarray*) – beam transformation matrix, 2x2
- ****kwargs** – key word arguments, such as 'comment', 'units', 'radius' and 'verbose'

class gaussopt.component.**Dielectric**(*thickness*, *n=1.0*, ***kwargs*)
Bases: *gaussopt.component.Component*

Propagation through a dielectric slab.

To get initialization information, see __init__().

matrix
ndarray – beam transformation matrix, 2x2

Dielectric constructor.

Parameters

- **thickness** (*float*) – thickness of dielectric slab
- **n** (*float, optional*) – index of refraction
- ****kwargs** – key word arguments, such as 'comment', 'units', 'radius' and 'verbose'

class gaussopt.component.**Freespace**(*distance*, ***kwargs*)
Bases: *gaussopt.component.Component*

Freespace propagation.

To get initialization information, see __init__().

matrix
ndarray – beam transformation matrix, 2x2

Freespace constructor.

Parameters

- **distance** (*float*) – freespace propagation distance
- ****kwargs** – key word arguments, such as 'comment', 'units', 'radius' and 'verbose'

class gaussopt.component.**Horn**(*freq*, *slen*, *arad*, *hf=0.59*, ***kwargs*)
Bases: object

Waveguide horn antenna.

To get initialization information, see __init__().

f
ndarray/float – frequency

w
ndarray/float – beam waist at aperture

z
ndarray/float – z-offset

q
ndarray/complex – beam parameter at aperture

Horn constructor.

Parameters

- **freq** (*class 'gaussopt.frequency.Frequency'*) – frequency class
- **slen** (*float*) – slant length
- **arad** (*float*) – aperture radius

- **hf** (*float, optional*) – horn factor
- ****kwargs** – key word arguments, such as 'comment', 'units', and 'verbose'

copy(***kwargs*)
Copy horn to new instance.

Parameters ****kwargs** – keyword arguments to pass to new instance

Returns new instance of the Horn class

Return type class

plot_properties()
Plots beam waist and z-offset over frequency range.

waist(*units='mm'*)
Beam waist at aperture.

Parameters **units** (*str, optional*) – units to use for returned values

Returns waist at aperture

Return type float

z_offset(*units='mm'*)
Get distance between horn aperture and beam waist (a.k.a. z-offset).

Parameters **units** (*str, optional*) – units to use for returned value

Returns z offset

Return type float

class gaussopt.component.**Mirror**(*focal_length*, ***kwargs*)
Bases: *gaussopt.component.Component*

Reflection off of a parabolic mirror.

To get initialization information, see __init__().

matrix
ndarray – beam transformation matrix, 2x2

Mirror constructor.

Parameters

- **focal_length** (*float*) – mirror focal length
- ****kwargs** – key word arguments, such as 'comment', 'units', 'radius' and 'verbose'

class gaussopt.component.**Window**(***kwargs*)
Bases: *gaussopt.component.Component*

A window.

To get initialization information, see `__init__()`.

matrix
ndarray – beam transformation matrix, 2x2

Window constructor.

Parameters ****kwargs** – key word arguments, such as 'comment', 'units', 'radius' and 'verbose'

1.3 gaussopt.frequency module

class `gaussopt.frequency.`**Frequency**(*start=None*, *stop=None*, *npts=301*, *units='GHz'*, ***kwargs*)

Bases: `object`

Frequency sweep.

f
float/ndarray – frequency array (in Hz)

w
float/ndarray – wavelength array (in m)

idx_center
int – index of center value

Frequency constructor.

Parameters

- **start** (*float, optional*) – start frequency
- **stop** (*float, optional*) – stop frequency
- **npts** (*int, optional*) – number of points
- **units** (*str, optional*) – frequency units (e.g., 'GHz', 'MHz')
- ****kwargs** – keyword arguments center/span can be used to specify frequency single can be used to use a single frequency verbose and comment

idx(*freq*, *units='GHz'*)
Get index of value closest to specified frequency.

Parameters

- **freq** (*float*) – target frequency
- **units** (*str*) – units for the frequency

Returns index of value closest to given frequency

Return type int

wavelength()
Get wavelength.

Returns wavelength in m

Return type float/ndarray

1.4 gaussopt.system module

class gaussopt.system.**System**(*horn_tx*, *component_list*, *horn_rx=None*, ***kwargs*)
Bases: object

Entire optical system.

matrix
ndarray – cascaded beam transformation matrix

f
ndarray/float – frequency

wout
ndarray/float – output beam waist

qout
ndarray/complex – output beam parameter

rout
ndarray/float – output beam radius

System constructor.

Parameters

- **horn_tx** (*class*) – transmitting horn
- **component_list** (*tuple/list*) – list of optical components
- **horn_rx** (*class*) – receiving horn
- ****kwargs** – keyword arguments (comment, verbose)

best_coupling()
Get best coupling.

Returns best coupling

Return type float

best_coupling_frequency()
Get best coupling frequency.

Returns frequency where best coupling is found

Return type float

coupling()
Get coupling between the horns.

Returns coupling between the antennas

Return type ndarray

plot_aperture_30db(*ax=None*)

plot_coupling(*ax=None*)
Plot coupling (in percentage) versus frequency.

plot_coupling_db(*ax=None*)
Plot coupling (in dB) versus frequency.

plot_coupling_mag(*ax=None*)
Plot coupling versus frequency.

plot_edge_taper_db(*ax=None*)

plot_system(*freq=None*, *ax=None*, *figname=None*)

plot_waists(*ax=None*)

print_best_coupling(*units='GHz'*)
Print best coupling and frequency where it is found.

Parameters **units** (*str, optional*) – units for frequency

gaussopt.system.**transform_beam**(*sys_matrix*, *q_in*)
Transform a beam using the beam transformation matrix.

Parameters

- **sys_matrix** (*ndarray*) – beam transformation matrix
- **q_in** (*complex/ndarray*) – input beam parameter (q)

Returns output beam parameter (q)

Return type complex/ndarray

1.5 gaussopt.util module

gaussopt.util.**set_d_units**(*units*)
Read distance units.

Parameters **units** (*str*) – distance units (e.g., 'mm', 'um', 'cm')

Returns multiplier

Return type float

gaussopt.util.**set_f_units**(*units*)
Read frequency units.

Parameters **units** (*str*) – frequency units (e.g., 'THz', 'GHz', 'MHz')

Returns multiplier

Return type float

gaussopt.util.**set_verbosity**(*verbosity*)

1.6 Module contents

Analyze quasi-optical systems using Gaussian beam analysis.

CHAPTER 2

Gaussopt Example

- This example will walk through the basics of setting up a Gaussian telescope.
- Note: All distances are in mm and all frequencies are in GHz unless specified otherwise.

```
# Import the gaussopt package
from gaussopt import *

# Import modules for this notebook
import matplotlib.pyplot as plt
%matplotlib inline
from IPython.display import Image
```

- Gaussian telescopes use two mirrors to couple energy between two horn antennas. If the mirrors have focal lengths f, then the mirrors should be separated by $2f$ and the distance between each horn's beam waist and it's respective mirror should be f.

2.1 Define frequency sweep

- The standard way to initialize this class is to define the start and end frequency.
- This class assumes GHz unless a unit is provided.

```
freq = Frequency(150, 300, comment='rf sweep')
```

```
Frequency sweep: rf sweep
    f = 150.0 to 300.0 GHz, 301 pts
```

2.2 Define horns

```
slen = 22.64  # slant length (in mm)
arad = 3.6    # aperture radius (in mm)
hfac = 0.59   # horn factor
horn_tx = Horn(freq, slen, arad, hfac, comment='Trasmitting')
horn_rx = horn_tx.copy(comment='Receiving')
```

```
Horn: Trasmitting
    slen = 22.64 mm
    arad =  3.60 mm
    hf   =  0.59

Horn: Receiving
    slen = 22.64 mm
    arad =  3.60 mm
    hf   =  0.59
```

2.3 Define optical components

- These classes will assume mm unless a unit is provided.

```
d = Freespace(160)
m1 = Mirror(16, units='cm', radius=8, comment='M1')
m2 = Mirror(16, units='cm', radius=8, comment='M2')
```

```
Freespace:
    d = 160.0 mm

Mirror: M1
    f = 16.0 cm

Mirror: M2
    f = 16.0 cm
```

- Note that the distance between the horn and the mirror needs to be reduced because the actual beam waist will be behind the horn aperture.

```
z_offset = horn_tx.z_offset(units='mm')[freq.idx(230)]
d_red = Freespace(160 - z_offset, comment='reduced')
```

```
Freespace: reduced
    d = 155.8 mm
```

2.4 Build Optical System

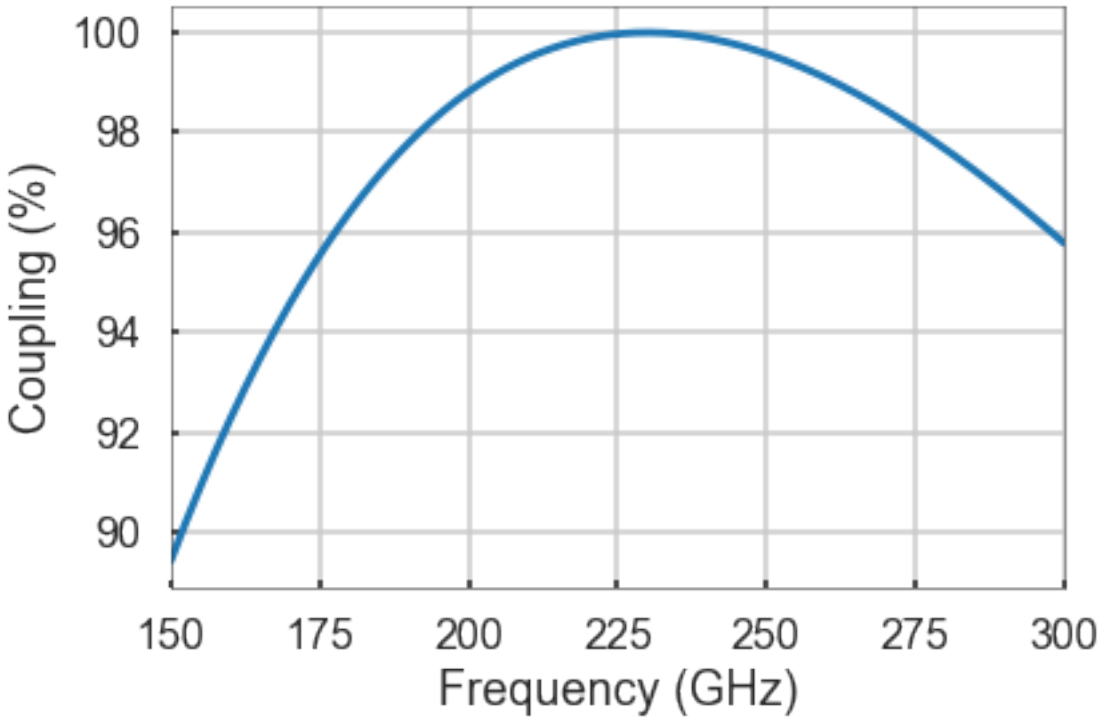
```
component_list = (d_red, m1, d, d, m2, d_red)

system = System(horn_tx, component_list, horn_rx)
```

```
System:
[[-1.          0.00848684]
 [ 0.         -1.        ]]
```

2.5 Plot Coupling

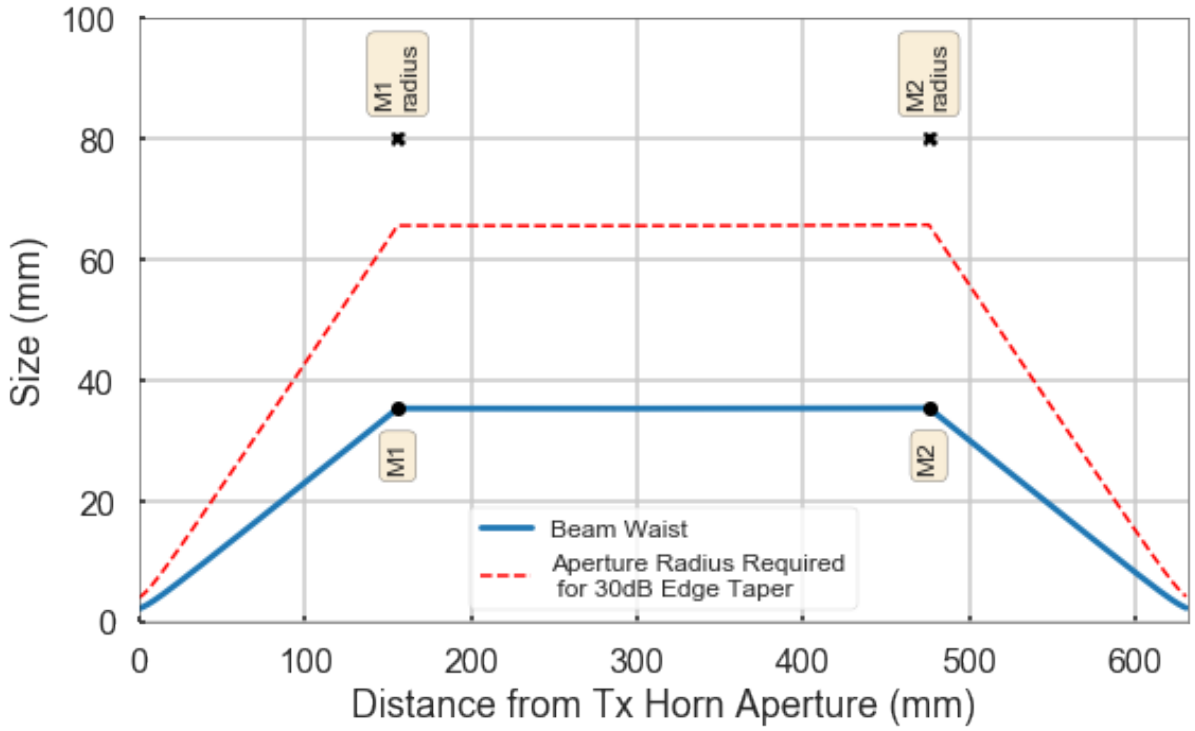
```
system.plot_coupling()
system.print_best_coupling()
```




```
Best coupling: 100.0 % at 230.0 GHz
```

2.6 Plot Beam Propagation

```
fig, ax = plt.subplots(figsize=(8,5))
system.plot_system(ax=ax)
```


M1 radius
M2 radius
M1
M2
Beam Waist
Aperture Radius Required for 30dB Edge Taper
Size (mm)
Distance from Tx Horn Aperture (mm)
0
20
40
60
80
100
0
100
200
300
400
500
600

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

gaussopt.component, 3
gaussopt.frequency, 6
gaussopt.system, 7
gaussopt.util, 8

Index

B

best_coupling() (gaussopt.system.System method), 7
best_coupling_frequency() (gaussopt.system.System method), 7

C

Component (class in gaussopt.component), 3
copy() (gaussopt.component.Horn method), 5
coupling() (gaussopt.system.System method), 7

D

Dielectric (class in gaussopt.component), 3

F

f (gaussopt.component.Horn attribute), 4
f (gaussopt.frequency.Frequency attribute), 6
f (gaussopt.system.System attribute), 7
Freespace (class in gaussopt.component), 4
Frequency (class in gaussopt.frequency), 6

G

gaussopt (module), 9
gaussopt.component (module), 3
gaussopt.frequency (module), 6
gaussopt.system (module), 7
gaussopt.util (module), 8

H

Horn (class in gaussopt.component), 4

I

idx() (gaussopt.frequency.Frequency method), 6
idx_center (gaussopt.frequency.Frequency attribute), 6

M

matrix (gaussopt.component.Component attribute), 3
matrix (gaussopt.component.Dielectric attribute), 3
matrix (gaussopt.component.Freespace attribute), 4
matrix (gaussopt.component.Mirror attribute), 5
matrix (gaussopt.component.Window attribute), 6
matrix (gaussopt.system.System attribute), 7
Mirror (class in gaussopt.component), 5

P

plot_aperture_30db() (gaussopt.system.System method), 8
plot_coupling() (gaussopt.system.System method), 8
plot_coupling_db() (gaussopt.system.System method), 8
plot_coupling_mag() (gaussopt.system.System method), 8
plot_edge_taper_db() (gaussopt.system.System method), 8
plot_properties() (gaussopt.component.Horn method), 5
plot_system() (gaussopt.system.System method), 8
plot_waists() (gaussopt.system.System method), 8
print_best_coupling() (gaussopt.system.System method), 8

Q

q (gaussopt.component.Horn attribute), 4
qout (gaussopt.system.System attribute), 7

R

rout (gaussopt.system.System attribute), 7

S

set_d_units() (in module gaussopt.util), 8
set_f_units() (in module gaussopt.util), 8
set_verbosity() (in module gaussopt.util), 9
System (class in gaussopt.system), 7

T

transform_beam() (in module gaussopt.system), 8

W

w (gaussopt.component.Horn attribute), 4
w (gaussopt.frequency.Frequency attribute), 6
waist() (gaussopt.component.Horn method), 5
wavelength() (gaussopt.frequency.Frequency method), 6
Window (class in gaussopt.component), 5
wout (gaussopt.system.System attribute), 7

Z

z (gaussopt.component.Horn attribute), 4
z_offset() (gaussopt.component.Horn method), 5