GaussOpt Documentation

Release 0.1.dev1

John Garrett

Aug 13, 2018

Contents:

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)

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 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.

1.1 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
```

1.2 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
```

1.3 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
```

1.4 Build Optical System

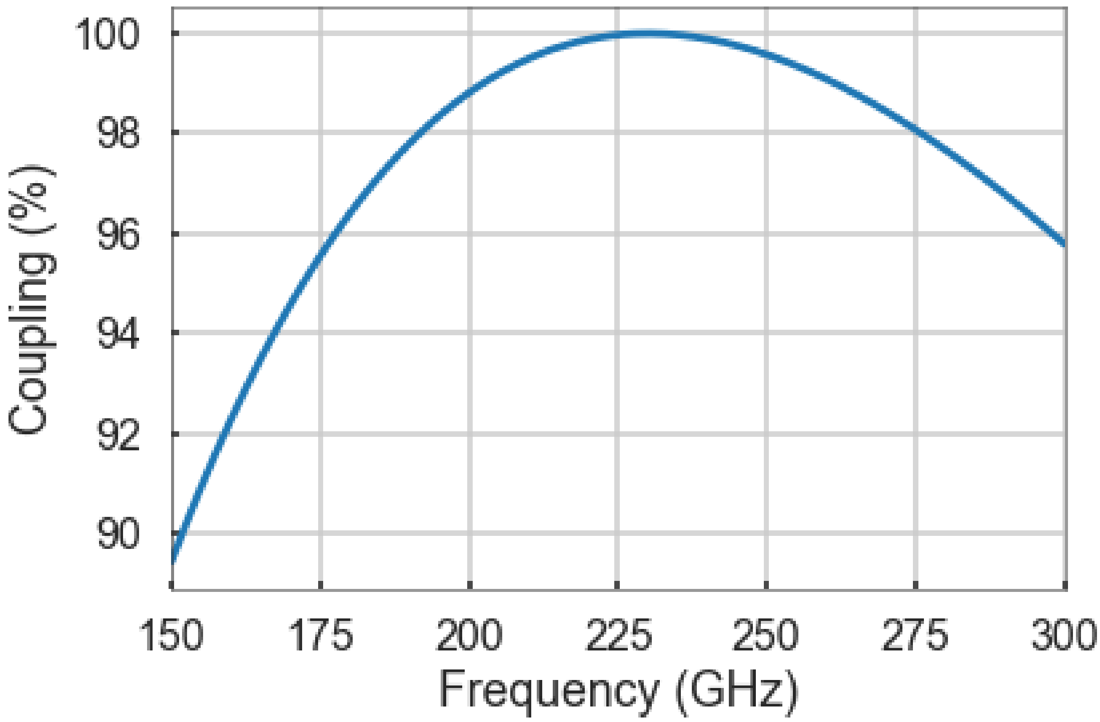
```
component_list = (d_red, m1, d, d, m2, d_red)

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

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

1.5 Plot Coupling

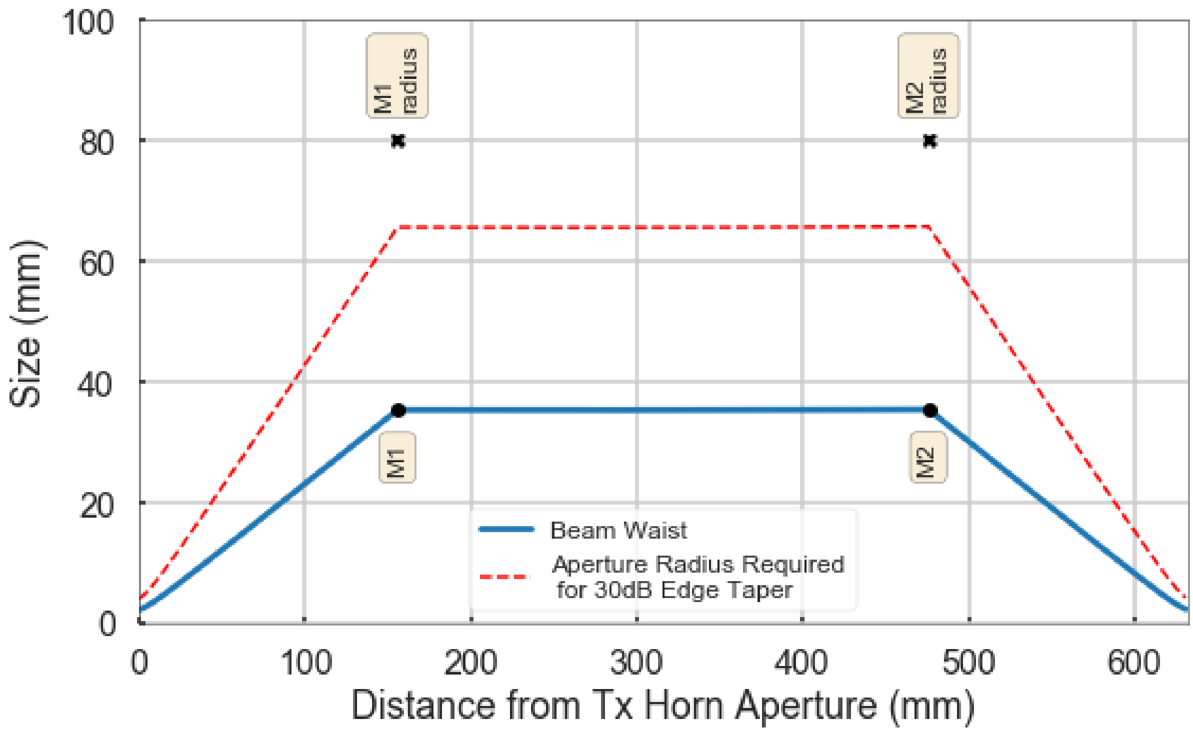
```
system.plot_coupling()
system.print_best_coupling()
```




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

1.6 Plot Beam Propagation

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



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

CHAPTER 2

gaussopt package

2.1 Submodules

2.2 gaussopt.component module

Classes for optical components.

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'

2.3 gaussopt.frequency module

Class for frequency sweep.

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

Bases: `object`

Class for 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

2.4 gaussopt.system module

Build optical system and analyze.

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

2.5 gaussopt.util module

General utilities.

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*`)`

2.6 Module contents

Analyze quasi-optical systems using Gaussian beam analysis.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

Index

Q

R

S

T

W

Z