pygrin

Release 0.4.4

Aug 06, 2021

Example Usage

A basic collection of routines to ray trace through graded index (GRIN) lenses with a parabolic radial profile.

CHAPTER 1

Usage

Example Light paths in a 0.25 pitch GRIN lens from an ancient Melles Griot Catalog:

```
import pygrin
n = 1.608
gradient = 0.339
length = 5.37
diameter = 1.8

pitch = pygrin.period(gradient, length)
ffl = pygrin.FFL(n,pitch,length)
efl = pygrin.EFL(n,pitch,length)
na = pygrin.NA(n,pitch,length,diameter)

angle = pygrin.max_angle(n,pitch,length,diameter)
print('expected pitch = 0.29,            calculated %.2f' % pitch)
print('expected FFL = 0.46 mm,           calculated %.2f' % ffl)
print('expected NA = 0.46,               calculated %.2f' % na)
print('expected full accept angle = 55°, calculated %.0f°' % (2*angle*180/np.pi))
print('working distance = %.2f mm'%(efl-ffl))
```

Produces:

```
expected pitch = 0.29,            calculated 0.29
expected FFL = 0.46,              calculated 0.47
expected NA = 0.46,               calculated 0.46
expected full accept angle = 55°, calculated 55°
working distance = 1.43 mm
```

But the real utility of this module is creating plots that show the path of rays through a GRIN lens. For examples, see <https://pygrin.readthedocs.io>

CHAPTER 2

Installation

Source code is available at <https://github.com/scottprahl/pygrin> or the module can be installed using *pip*:

```
pip install --user pygrin
```

CHAPTER 3

License

pygrin is licensed under the terms of the MIT license.

3.1 Gradient Index Lenses

Scott Prahl

May 2020

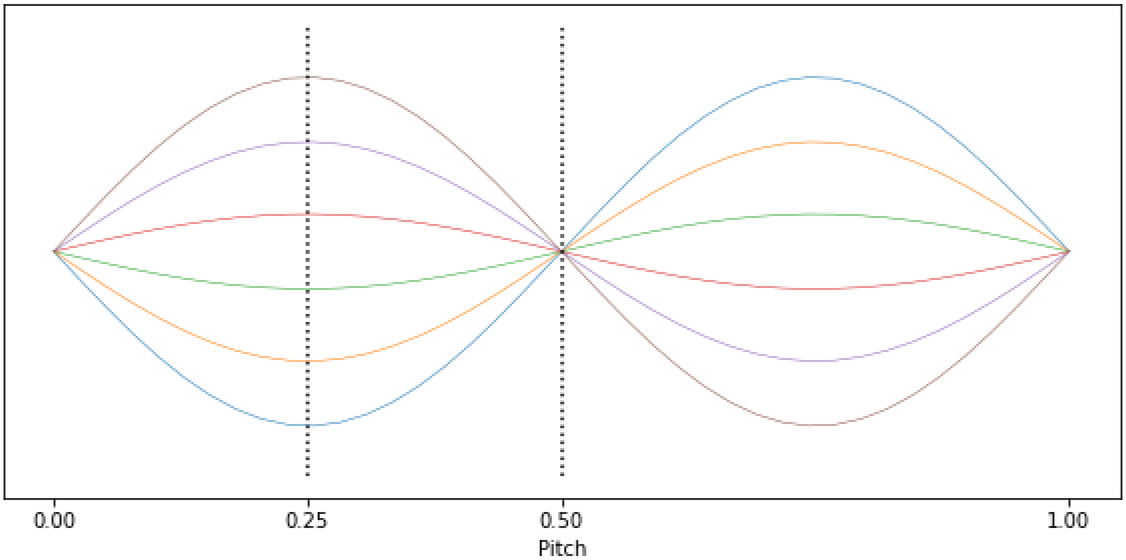
[1]:
```
import sys
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as patches

try:
    import pygrin
except:
    print("***** You need to install the pygrin module first           *****")
    print("***** Execute the following line in a new cell, then retry *****")
    print()
    print("!{sys.executable} -m pip install --user pygrin")
```

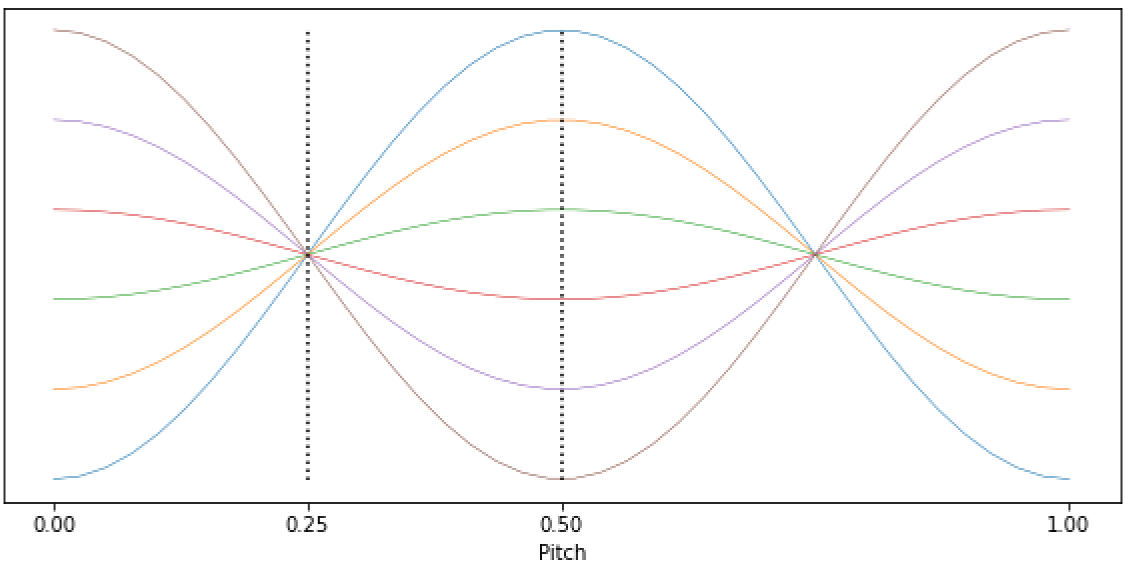
3.1.1 Pitch of a GRIN lens

A gradient index (GRIN) lens is characterized by a sinusoidal ray path along the lens. A lens is typically characterized its **pitch** or the number of sinusoidal periods within the lens.

If the pitch is one (or an integer) then the grin lens acts as a relay and reproduces the light incident on the entrance surface at the exit surface. A half-pitch inverts the pattern. A quarter-pitch lens will collimate a point source or focus a collimated incident beam.

[2]:
```
n0 = 1.608              # centerline index of refraction
length = 1              # mm
diameter = 0.25            # mm
angle = 40*np.pi/180 # radians
xpos = 0
pitch = 1
radius = diameter/2

# max angle in air
max_angle = pygrin.max_angle(n0,pitch,length,diameter)
# range of launch angles
angles = np.linspace(-max_angle,max_angle,6)
# range of launch angles in grin lens
angles = np.arcsin(np.sin(angles/n0))

plt.subplots(2,1,figsize=(10,10))
plt.subplot(2,1,1)
for angle in angles:
    z,r = pygrin.meridional_curve(n0, pitch, length, xpos, angle)
    plt.plot(z,r, lw=0.5)

plt.plot([0.25*length,0.25*length],[-radius,radius],':k')
plt.plot([0.5*length,0.5*length],[-radius,radius],':k')

plt.xticks([0,0.25,0.5,1.0])
plt.yticks([])
plt.xlabel('Pitch')
plt.title('')

plt.subplot(2,1,2)
for xpos in np.linspace(-radius,radius,6):
    z,r = pygrin.meridional_curve(n0, pitch, length, xpos, 0)
    plt.plot(z,r, lw=0.5)

plt.plot([0.25*length,0.25*length],[-radius,radius],':k')
plt.plot([0.5*length,0.5*length],[-radius,radius],':k')

plt.xticks([0,0.25,0.5,1.0])
plt.yticks([])
plt.xlabel('Pitch')
plt.title('')

plt.show()
```





Quarter Pitch

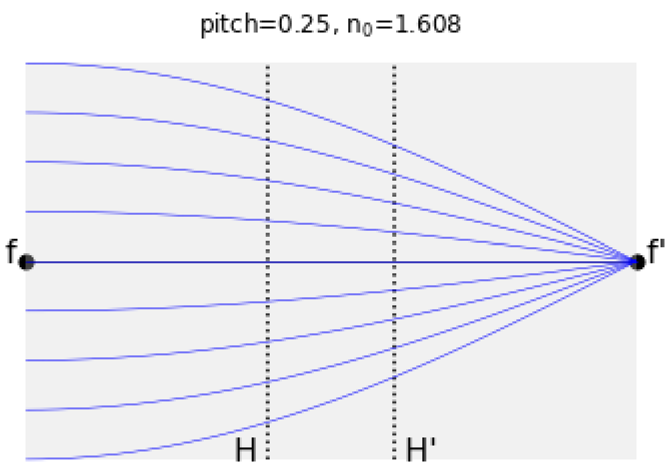
This is the typical example because collimated light is focused to a point, or conversely, a point source is collimated. Here we see an example the former.

[3]:
```
pitch = 0.25
n0 = 1.608              # centerline index of refraction
length = 5              # mm
diameter = 2            # mm
angle = 0*np.pi/180     # radians

pygrin.plot_principal_planes(n0, pitch, length, diameter)
xpos = np.linspace(-diameter/2,diameter/2,9)
for pos in xpos:
```

(continues on next page)

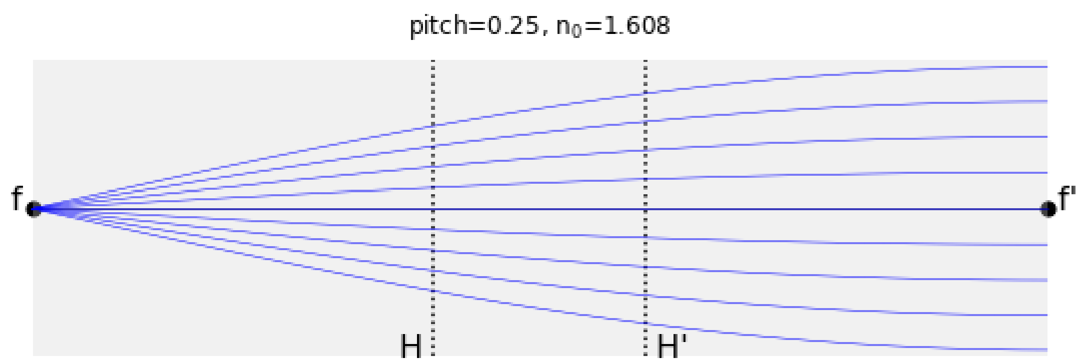
(continued from previous page)

```
    z,r = pygrin.meridional_curve(n0, pitch, length, pos, angle)
    plt.plot(z,r, color='blue',lw=0.5)

plt.rcParams["figure.figsize"] = [10,3]
plt.axis('off')
plt.show()
```



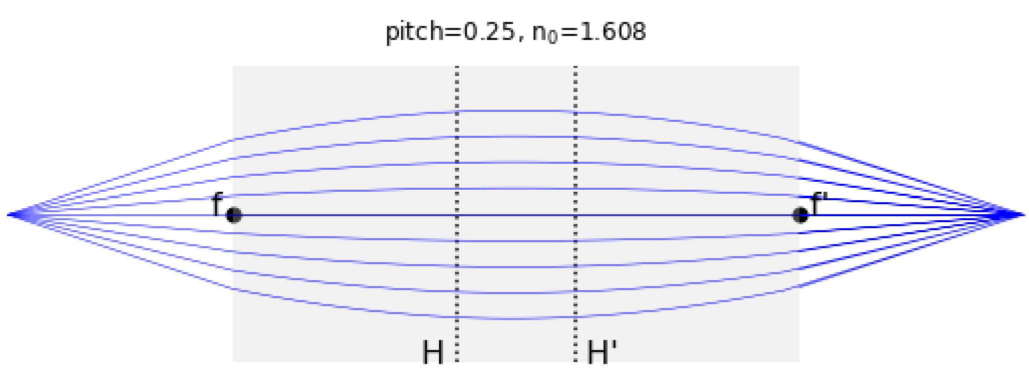
This shows light being collimated from a point source

[4]:
```
pitch = 0.25
n0 = 1.608              # centerline index of refraction
length = 5              # mm
diameter = 2            # mm
xpos = 0                # mm

pygrin.plot_principal_planes(n0, pitch, length, diameter)

# max angle in air
max_angle = pygrin.max_angle(n0,pitch,length,diameter)
# range of launch angles
angles = np.linspace(-max_angle,max_angle,9)
# range of launch angles in grin lens
angles = np.arcsin(np.sin(angles/n0))

for angle in angles:
    z,r = pygrin.meridional_curve(n0, pitch, length, xpos, angle)
    plt.plot(z,r, color='blue',lw=0.5)

plt.rcParams["figure.figsize"] = [10,3]
plt.axis('off')
plt.show()
```



Here is a 4f system. Here the source on the left is on focal distance from the front face and the

[5]:
```
# 4f system
pitch = 0.25
n0 = 1.608              # centerline index of refraction
length = 5              # mm
diameter = 2            # mm
zobj = pygrin.EFL(n0,pitch,length)
xpos = np.linspace(-diameter/4,diameter/4,9)

pygrin.plot_principal_planes(n0, pitch, length, diameter)

for pos in xpos:
    z,r = pygrin.full_meridional_curve(n0, pitch, length, -zobj, 0.0, pos)
    plt.plot(z,r, color='blue',lw=0.5)

plt.rcParams["figure.figsize"] = [10,3]
plt.axis('off')
plt.show()
```



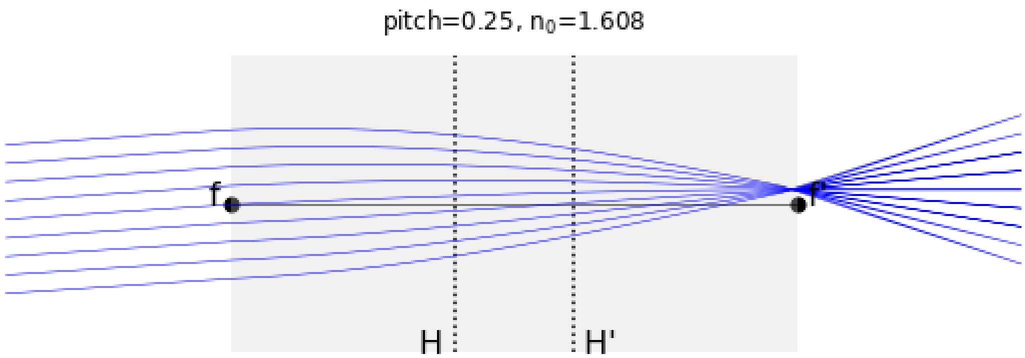
Finally, this shows that collimated light incident at an angle on the lens will be imaged to a point off-axis.

[6]:
```
# off axis launch
pitch = 0.25
n0 = 1.608              # centerline index of refraction
length = 5           # mm
diameter = 2          # mm
zobj = pygrin.EFL(n0,pitch,length)
xpos = np.linspace(-diameter/4,diameter/4,9)
```

(continues on next page)

(continued from previous page)

```
pygrin.plot_principal_planes(n0, pitch, length, diameter)

for pos in xpos:
    z,r = pygrin.full_meridional_curve(n0, pitch, length, -zobj, pos-0.1, pos)
    plt.plot(z,r, color='blue',lw=0.5)

plt.rcParams["figure.figsize"] = [10,3]
plt.axis('off')
plt.show()
```

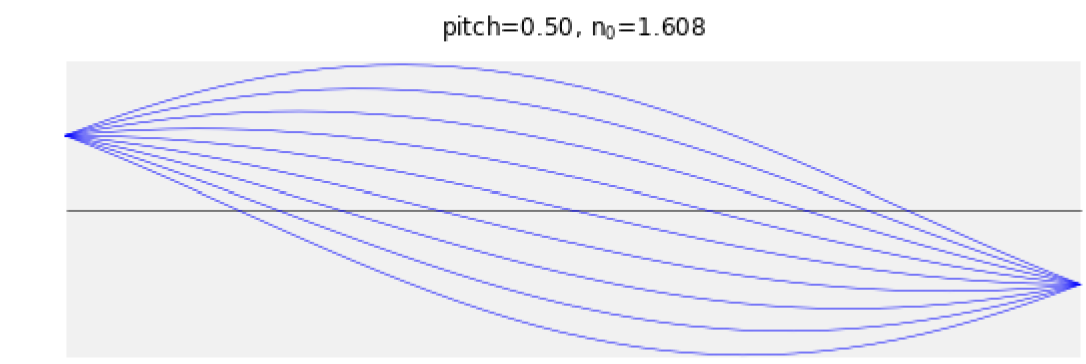


Half Pitch

Here we show that all the rays hitting the front surface at a point are imaged to the same point on the exit surface — but inverted.

The prinicipal planes are not drawn because the effective focal length is infinite.

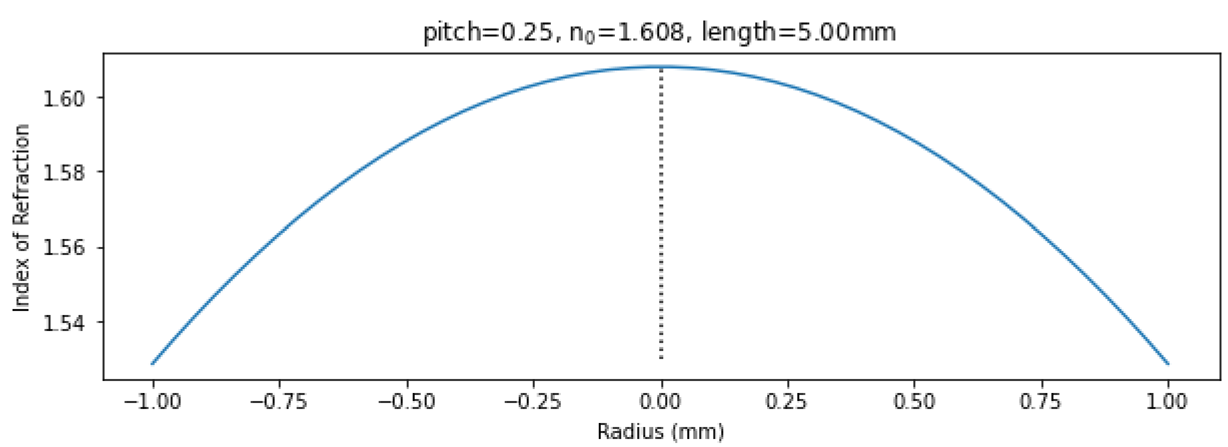
[7]:
```
pitch = 0.5
n0 = 1.608              # centerline index of refraction
length = 5              # mm
diameter = 2            # mm
xpos = 0.5                # mm

pygrin.plot_principal_planes(n0, pitch, length, diameter)

# max angle in air
max_angle = pygrin.max_angle(n0,pitch,length,diameter)
# range of launch angles
angles = np.linspace(-max_angle,max_angle,9)
# range of launch angles in grin lens
angles = np.arcsin(np.sin(angles/n0))

for angle in angles:
    z,r = pygrin.meridional_curve(n0, pitch, length, xpos, angle)
    plt.plot(z,r, color='blue',lw=0.5)

plt.rcParams["figure.figsize"] = [10,3]
plt.axis('off')
plt.show()
```



3.2 Index of Refraction Across Lens

3.2.1 Parabolic Profile

[8]:
```
n0 = 1.608
pitch = 0.25
length = 5    # mm
diameter = 2
r = np.linspace(-diameter/2,diameter/2,50)

plt.plot(r,pygrin.parabolic_profile_index(n0,pitch,length,r))
plt.plot([0,0],[1.53,n0],':k')
plt.xlabel('Radius (mm)')
plt.ylabel('Index of Refraction')
plt.title(r'pitch=%.2f, n$_0$=%.3f, length=%.2fmm'%(pitch,n0,length))
plt.rcParams["figure.figsize"] = [6,4]

plt.show()
```

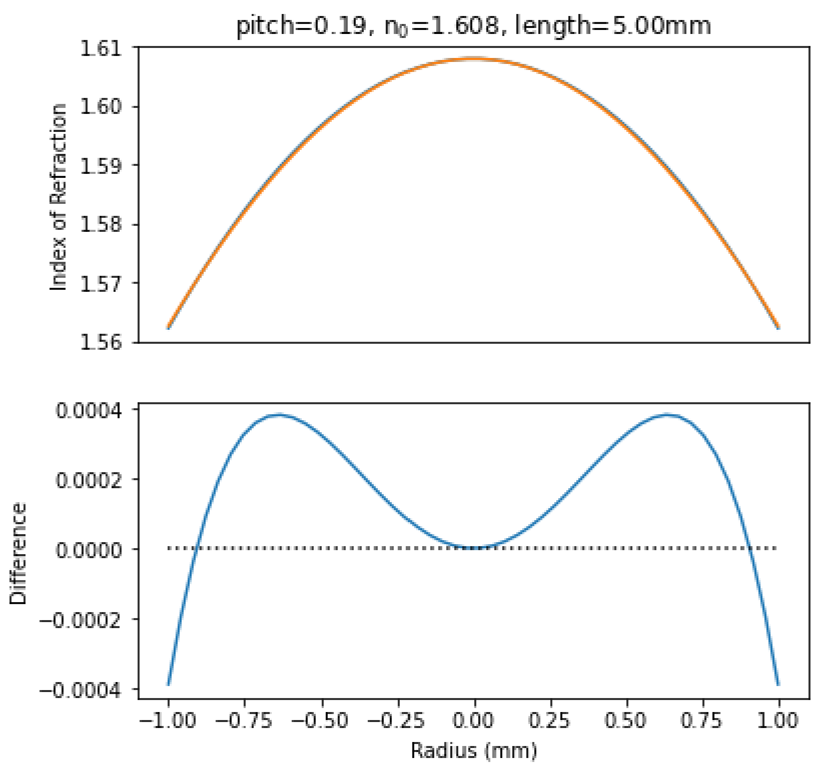


3.2.2 Hyperbolic Secant Profile

This has a few advantages over the parabolic profile: the propagation for a HS grin lens can be solved exactly and there aren't any aberrations for meridional rays.

As you can see below, the HS profile can be quite close to the parabolic profile.

[9]:
```
n0 = 1.608
pitch = 0.19
length = 5    # mm
diameter = 2
gradient = pygrin.gradient(pitch,length)
alpha = 1.303*gradient

r = np.linspace(-diameter/2,diameter/2,50)

plt.subplots(2,1, sharex=True,figsize=(6,6))
plt.subplot(2,1,1)
n_p = pygrin.parabolic_profile_index(n0,pitch,length,r)
n_s = pygrin.hyperbolic_secant_profile_index(n0,alpha,r)
plt.plot(r,n_p)
plt.plot(r,n_s)
plt.xticks([])
plt.ylabel('Index of Refraction')
plt.title(r'pitch=%.2f, n$_0$=%.3f, length=%.2fmm'%(pitch,n0,length))
plt.rcParams["figure.figsize"] = [6,4]

plt.subplot(2,1,2)
plt.plot(r,n_p-n_s)

plt.xlabel('Radius (mm)')
plt.ylabel('Difference')
plt.plot([-1,1],[0,0],':k')

plt.show()
```



3.3 Catalog Examples

3.3.1 Grin Lens from ancient Melles Griot Catalog, 4.67 line 2

Part number LGS-0.25-1.0-2.58-633

[10]:
```
n = 1.608
gradient = 0.608
length = 2.58
diameter = 1

pitch = pygrin.period(gradient, length)
print('expected pitch = 0.25,           calculated %.2f'%pitch)

efl = pygrin.EFL(n,pitch,length)
na = pygrin.NA(n,pitch,length,diameter)
print('expected NA = 0.46,              calculated %.2f'%na)

angle = pygrin.max_angle(n,pitch,length,diameter)
print('expected full accept angle = 55°, calculated %.0f°'%(2*angle*180/np.pi))
```

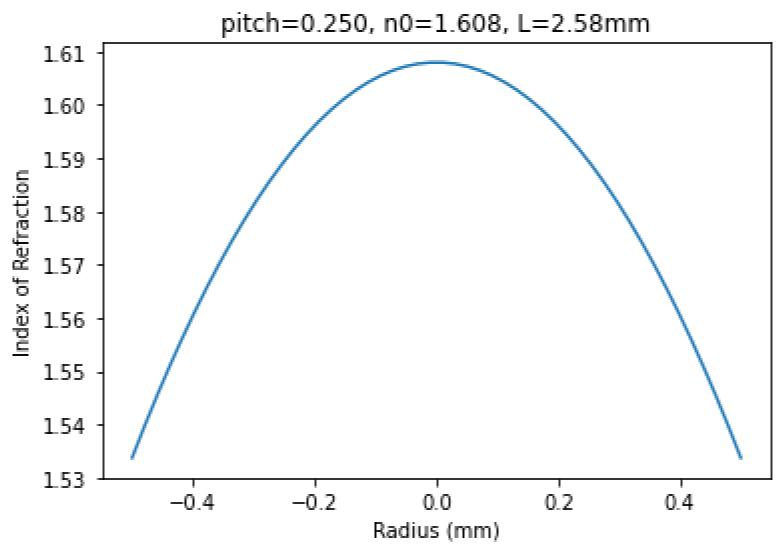
```
expected pitch = 0.25,           calculated 0.25
expected NA = 0.46,              calculated 0.46
expected full accept angle = 55°, calculated 54°
```

[11]:
```
r = np.linspace(-0.5,0.5,50)
plt.plot(r,pygrin.parabolic_profile_index(n,pitch,length,r))
```

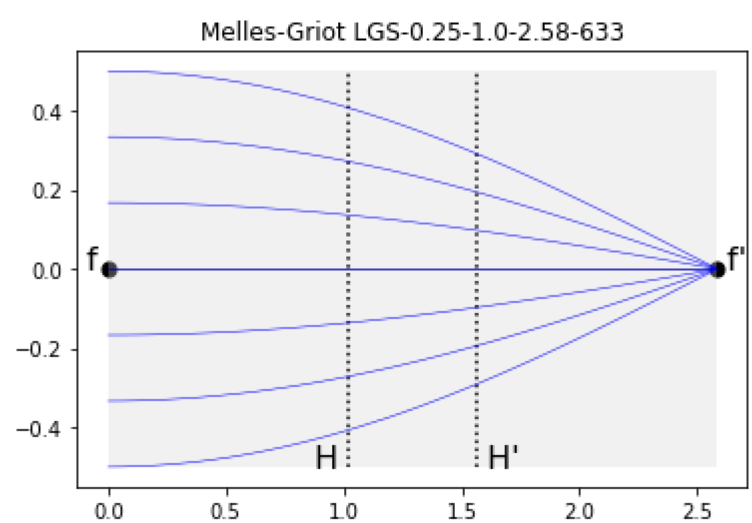
(continues on next page)

(continued from previous page)

```
plt.xlabel('Radius (mm)')
plt.ylabel('Index of Refraction')
plt.title('pitch=%.3f, n0=%.3f, L=%.2fmm'%(pitch,n,length))
plt.show()
```



[12]:
```
pygrin.plot_principal_planes(n, pitch, length, diameter)
xpos = np.linspace(-diameter/2,diameter/2,7)
for pos in xpos:
    z,r = pygrin.meridional_curve(n, pitch, length, pos, 0*np.pi/180)
    plt.plot(z,r, color='blue',lw=0.5)

plt.title('Melles-Griot LGS-0.25-1.0-2.58-633')
plt.show()
```



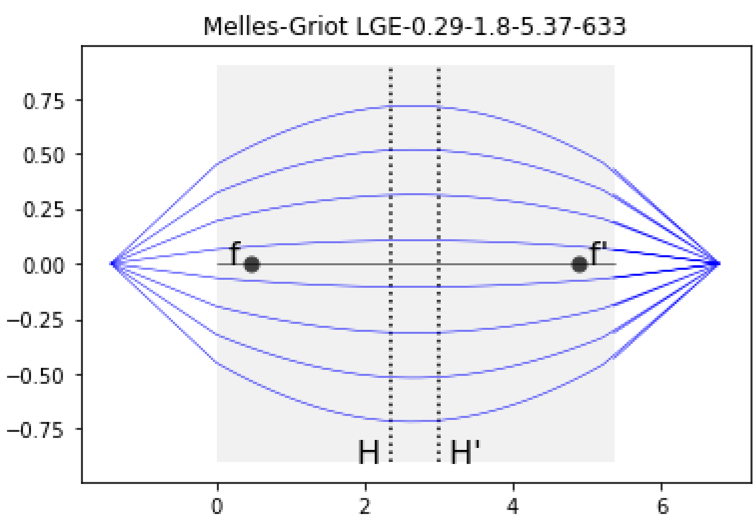
3.3.2 Grin Lens from ancient Melles Griot Catalog, 4.67 line 5

Part number LGE-0.29-1.8-5.37-633

[13]:
```
n = 1.608
gradient = 0.339
length = 5.37
diameter = 1.8

pitch = pygrin.period(gradient, length)
print('expected pitch = 0.29,            calculated %.2f'%pitch)

ffl = pygrin.FFL(n,pitch,length)
efl = pygrin.EFL(n,pitch,length)
print('expected FFL = 0.46,              calculated %.2f'%ffl)

na = pygrin.NA(n,pitch,length,diameter)
print('expected NA = 0.46,               calculated %.2f'%na)

angle = pygrin.max_angle(n,pitch,length,diameter)
print('expected full accept angle = 55°, calculated %.0f°'%(2*angle*180/np.pi))

print('working distance = %.2f mm'%(efl-ffl))
```

```
expected pitch = 0.29,           calculated 0.29
expected FFL = 0.46,             calculated 0.47
expected NA = 0.46,              calculated 0.46
expected full accept angle = 55°, calculated 55°
working distance = 1.43 mm
```

[14]:
```
pygrin.plot_principal_planes(n, pitch, length, diameter)
xpos = np.linspace(-diameter/4, diameter/4,8)
for pos in xpos:
    z,r = pygrin.full_meridional_curve(n, pitch, length, ffl-efl, 0, pos)
    plt.plot(z,r, color='blue', lw=0.5)

plt.title('Melles-Griot LGE-0.29-1.8-5.37-633')

plt.show()
```

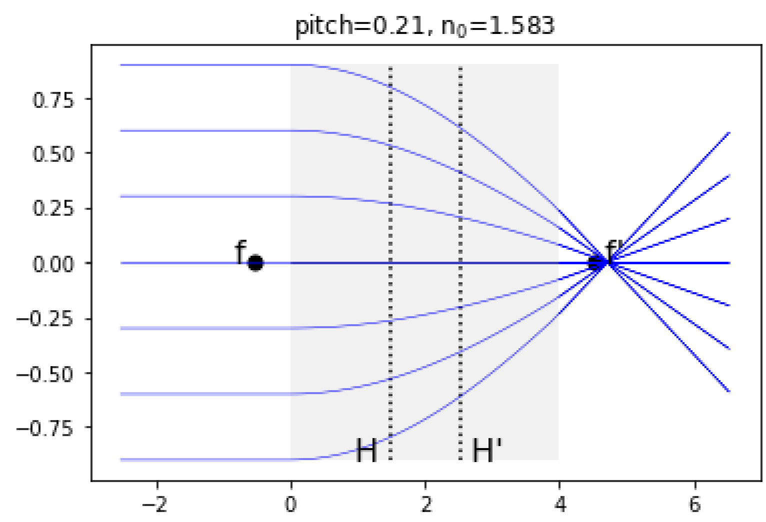


3.3.3 Riedl, page 96

[15]:
```
n = 1.5834
gradient = np.sqrt(0.1067)
length = 4
diameter = 1.8

pitch = pygrin.period(gradient, length)
print('expected pitch = 0.207,          calculated %.3f'%pitch)

efl = pygrin.EFL(n,pitch,length)
print('expected EFL = 2.00,             calculated %.2f'%efl)

ffl = pygrin.FFL(n,pitch,length)
print('expected FFL = -0.52,            calculated %.2f'%ffl)

bfl = pygrin.BFL(n,pitch,length)
print('expected BFL = 4.52,             calculated %.2f'%bfl)

na = pygrin.NA(n,pitch,length,diameter)
print('expected NA = 0.46,              calculated %.2f'%na)

angle = pygrin.max_angle(n,pitch,length,diameter)
print('expected full accept angle = 55°, calculated %.0f°'%(2*angle*180/np.pi))

print('working distance = %.2f mm'%(-ffl+efl))
```

```
expected pitch = 0.207,          calculated 0.208
expected EFL = 2.00,             calculated 2.00
expected FFL = -0.52,            calculated -0.52
expected BFL = 4.52,             calculated 4.52
expected NA = 0.46,              calculated 0.44
expected full accept angle = 55°, calculated 52°
working distance = 2.53 mm
```

[16]:
```
pygrin.plot_principal_planes(n, pitch, length, diameter)
xpos = np.linspace(-diameter/2, diameter/2,7)
for pos in xpos:
    z,r = pygrin.full_meridional_curve(n, pitch, length, ffl-efl, pos, pos)
    plt.plot(z,r, color='blue', lw=0.5)

plt.show()
```



3.4 Grin Lenses at Oregon Tech

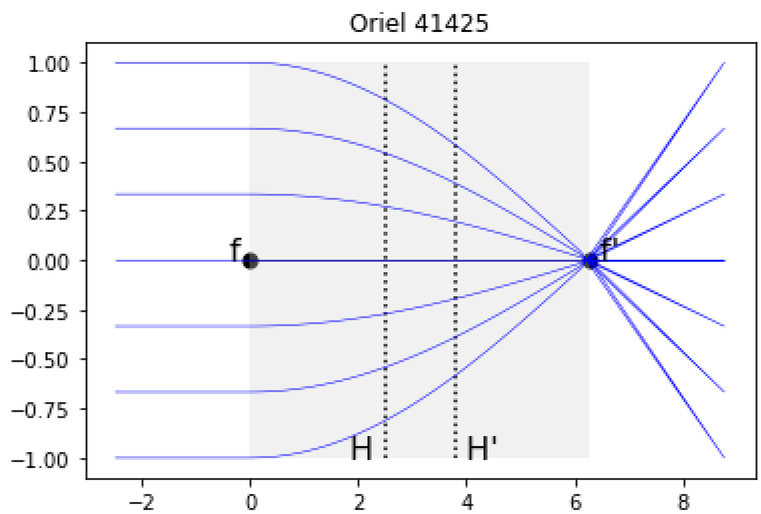
3.4.1 Oriel 41425

Designed for 632.8nm

[17]:
```
n = 1.608
pitch = 0.25
length = 6.28
diameter = 2

gradient = pygrin.gradient(pitch, length)
print('expected gradient = 0.250,        calculated %.3f'%pitch)

efl = pygrin.EFL(n,pitch,length)
#print('expected EFL = 2.00,              calculated %.2f'%efl)

ffl = pygrin.FFL(n,pitch,length)
#print('expected FFL = -0.52,             calculated %.2f'%ffl)

bfl = pygrin.BFL(n,pitch,length)
#print('expected BFL = 4.52,              calculated %.2f'%bfl)

na = pygrin.NA(n,pitch,length,diameter)
print('expected NA = 0.38,              calculated %.2f'%na)

angle = pygrin.max_angle(n,pitch,length,diameter)
print('expected full accept angle = 45°, calculated %.0f°'%(2*angle*180/np.pi))

print('working distance = %.2f mm'%(efl-ffl))
```

```
expected gradient = 0.250,         calculated 0.250
expected NA = 0.38,                calculated 0.38
expected full accept angle = 45°, calculated 45°
working distance = 2.49 mm
```

[18]:
```
pygrin.plot_principal_planes(n, pitch, length, diameter)
xpos = np.linspace(-diameter/2, diameter/2,7)
for pos in xpos:
    z,r = pygrin.full_meridional_curve(n, pitch, length, ffl-efl, pos, pos)
    plt.plot(z,r, color='blue', lw=0.5)

plt.title('Oriel 41425')

plt.show()
```



3.4.2 Oriel 41440

Designed for 632.8nm

[19]:
```
n = 1.608
pitch = 0.29
length = 5.35
diameter = 2

gradient = pygrin.gradient(pitch, length)
print('expected gradient = 0.290,        calculated %.3f'%pitch)

efl = pygrin.EFL(n,pitch,length)
#print('expected EFL = 2.00,              calculated %.2f'%efl)

ffl = pygrin.FFL(n,pitch,length)
#print('expected FFL = -0.52,             calculated %.2f'%ffl)

bfl = pygrin.BFL(n,pitch,length)
#print('expected BFL = 4.52,              calculated %.2f'%bfl)

na = pygrin.NA(n,pitch,length,diameter)
print('expected NA = 0.38,              calculated %.2f'%na)

angle = pygrin.max_angle(n,pitch,length,diameter)
print('expected full accept angle = 60°, calculated %.0f°'%(2*angle*180/np.pi))
```

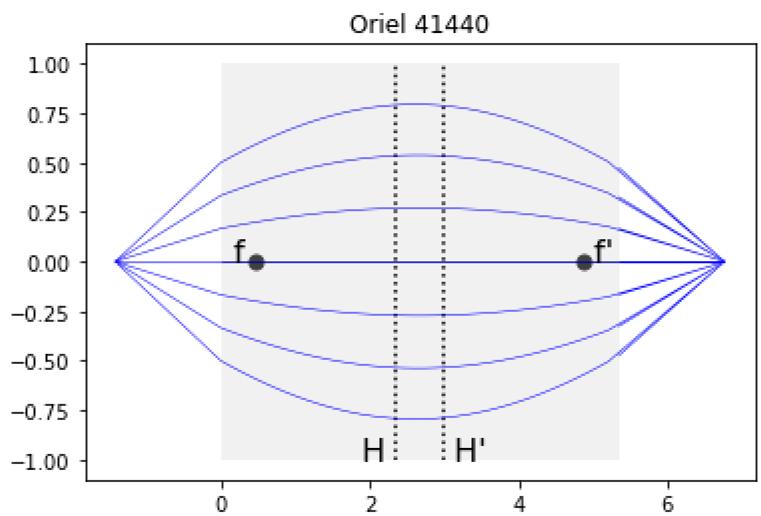
(continues on next page)

(continued from previous page)

```
print('working distance = %.2f mm'%(efl-ffl))
```

```
expected gradient = 0.290,        calculated 0.290
expected NA = 0.38,               calculated 0.50
expected full accept angle = 60°, calculated 60°
working distance = 1.42 mm
```

[20]:
```
pygrin.plot_principal_planes(n, pitch, length, diameter)
xpos = np.linspace(-diameter/4, diameter/4,7)
for pos in xpos:
    z,r = pygrin.full_meridional_curve(n, pitch, length, ffl-efl, 0, pos)
    plt.plot(z,r, color='blue', lw=0.5)

plt.title('Oriel 41440')
plt.show()
```



3.4.3 Newport FK - GR29

Designed for 850nm

[21]:
```
n = 1.608
pitch = 0.29
length = 5.49
diameter = 1.8

gradient = pygrin.gradient(pitch, length)
print('expected gradient = 0.290,        calculated %.3f'%pitch)

efl = pygrin.EFL(n,pitch,length)
print('expected EFL = 1.95,              calculated %.2f'%efl)

ffl = pygrin.FFL(n,pitch,length)
#print('expected FFL = -0.52,             calculated %.2f'%ffl)

bfl = pygrin.BFL(n,pitch,length)
#print('expected BFL = 4.52,              calculated %.2f'%bfl)
```

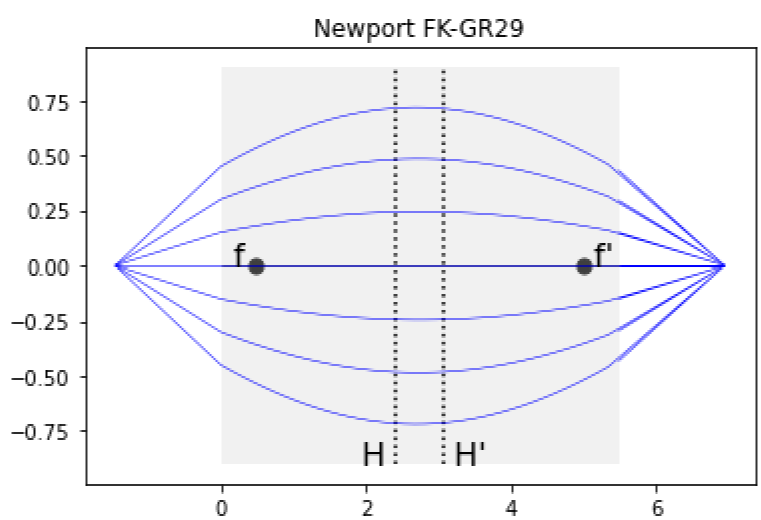
(continues on next page)

(continued from previous page)

```
na = pygrin.NA(n,pitch,length,diameter)
print('expected NA = 0.46,                calculated %.2f'%na)

angle = pygrin.max_angle(n,pitch,length,diameter)
print('expected full accept angle = 60°, calculated %.0f°'%(2*angle*180/np.pi))

print('working distance = %.2f mm'%(efl-ffl))
```

```
expected gradient = 0.290,        calculated 0.290
expected EFL = 1.95,              calculated 1.93
expected NA = 0.46,               calculated 0.45
expected full accept angle = 60°, calculated 53°
working distance = 1.45 mm
```

[22]:
```
pygrin.plot_principal_planes(n, pitch, length, diameter)
xpos = np.linspace(-diameter/4, diameter/4,7)
for pos in xpos:
    z,r = pygrin.full_meridional_curve(n, pitch, length, ffl-efl, 0, pos)
    plt.plot(z,r, color='blue', lw=0.5)

plt.title('Newport FK-GR29')
plt.show()
```



3.5 API for *pygrin* package

3.5.1 Submodules

3.5.2 pygrin.pygrin module

Gradient Index lens calculations and plots.

More documentation at <https://pygrin.readthedocs.io>

Typical usage:

```
import pygin

length = 7                 # mm
diameter = 2               # mm
r = np.linspace(-1,1,11)   # mm
n_0 = 1.48                 # refractive index at r=0
theta_i = 0                # launch angle
pitch = 0.25               # quarter pitch lens

pygrin.plot_principal_planes(n_0, pitch, length, diameter)
for r_i in r:
    z,r = pygrin.meridional_curve(n_0, pitch, length, r_i, theta_i)
    plt.plot(z,r,color='blue')
plt.show()
```

Functions to locate focal points and cardinal points:

```
BFL(n_0, pitch, length)
EFL(n_0, pitch, length)
FFL(n_0, pitch, length)
NA(n_0, pitch, length, diameter)
cardinal_points(n_0, pitch, length)
```

Functions to find properties of GRIN lens:

```
gradient(pitch, length)
period(grad, length)
max_angle(n_0, pitch, length, diameter)
ABCD(n_0, pitch, length, z)
image_distance(n_0, pitch, length, s)
image_mag(n_0, pitch, length, s)
```

Functions to determine refractive index profile:

```
hyperbolic_secant_profile_index(n_0, alpha, r)
parabolic_profile_index(n_0, pitch, length, r)
```

Functions to help raytrace through GRIN lens:

```
full_meridional_curve(n_0, pitch, length, z_obj, r_obj, r_lens)
meridional_curve(n_0, pitch, length, r_i, theta_i)
plot_principal_planes(n_0, pitch, length, diameter)
```

`pygrin.pygrin.`**`ABCD`**(*n_0*, *pitch*, *length*, *z*)

ABCD matrix for meridonal ray propagation.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **z** – distance within lens from front surface [mm]

Returns *float* – the ABCD matrix for meridonal ray propagation [radians]

`pygrin.pygrin.`**`BFL`**(*n_0*, *pitch*, *length*)

Back focal length of a grin lens.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]

Returns *float* – the back focal length of the grin lens [mm]

`pygrin.pygrin.`**`EFL`**`(`*n_0*, *pitch*, *length*`)`

Effective focal length of a grin lens.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]

Returns *float* – the effective focal length of the grin lens [mm]

`pygrin.pygrin.`**`FFL`**`(`*n_0*, *pitch*, *length*`)`

Front focal length of a grin lens.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]

Returns *float* – the front focal length of the grin lens [mm]

`pygrin.pygrin.`**`NA`**`(`*n_0*, *pitch*, *length*, *diameter*`)`

Numerical aperture of a grin lens in air.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **diameter** – diameter of the lens [mm]

Returns *float* – the numerical aperture of the grin lens in air [unitless]

`pygrin.pygrin.`**`cardinal_points`**`(`*n_0*, *pitch*, *length*, *offset=0*`)`

Cardinal points of a grin lens relative to first surface.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **offset** – float (optional) origin relative to first lens surface

Returns *float* – location of the front focal point [mm] float: location of the first lens surface [mm] float: location of the first principal plane [mm] float: location of the second principal plane [mm] float: location of the second lens surface [mm] float: location of the back focal point [mm]

pygrin.pygrin.**full_meridional_curve**(*n_0*, *pitch*, *length*, *z_obj*, *r_obj*, *r_lens*, *npoints=40*)
Points on a path from an object to image through a GRIN lens.

The light ray starts at (z_obj,r_obj) and hits the front surface of the front face of the GRIN lens at (0,r_lens).

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **z_obj** – axial position of the object [mm]
- **r_obj** – radius at which the ray leaves the object [mm]
- **r_lens** – radius at which the ray hits the lens [mm]
- **npoints** – integer (optional) number of points in the returned curve

Returns

z –

array axial points along path from object to image [mm]

r: array radial points along path from object to image [mm]

pygrin.pygrin.**gradient**(*pitch*, *length*)
Gradient of a grin lens based on its pitch and length.

Parameters

- **pitch** – pitch or period of the lens [unitless]
- **length** – length of grin lens [mm]

Returns *float* – the gradient characterizing the index of refraction profile [1/mm]

pygrin.pygrin.**hyperbolic_secant_profile_index**(*n_0*, *alpha*, *r*)
Index of a hyperbolic secant grin lens at a particular radius.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **alpha** – parameter (like gradient for parabolic lens) [1/mm]
- **r** – distance from center of lens [mm]

Returns *float* – the index of a parabolic grin lens at r [unitless]

pygrin.pygrin.**image_distance**(*n_0*, *pitch*, *length*, *s*)
Image distance for an object.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **s** – distance from front of lens to object [mm]

Returns *float* – the image distance from the back of the lens [mm]

pygrin.pygrin.**image_mag**(*n_0*, *pitch*, *length*, *s*)
Transverse magnification of an object located at s.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **s** – distance from front of lens to object [mm]

Returns *float* – the transvers magnification [unitless]

`pygrin.pygrin.`**`max_angle`**(*n_0*, *pitch*, *length*, *diameter*)
Maximum acceptance angle of a grin lens in air.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **diameter** – diameter of the lens [mm]

Returns *float* – the maximum acceptance angle of the lens in air [radians]

`pygrin.pygrin.`**`meridional_curve`**(*n_0*, *pitch*, *length*, *r_i*, *theta_i*, *npoints=40*)
Points on path of a ray passing through a grin lens.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **r_i** – radial distance that ray hits grin lens [mm]
- **theta_i** – angle of incidence [radians]
- **npoints** – integer number of points in the returned curve

Returns

z –

array axial points along the curve inside the grin lens [mm]

r: array radial points along the curve inside the grin lens [mm]

`pygrin.pygrin.`**`parabolic_profile_index`**(*n_0*, *pitch*, *length*, *r*)
Index of a parabolic grin lens at a particular radius.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **r** – distance from center of lens [mm]

Returns *float* – the index of a parabolic grin lens at r [unitless]

`pygrin.pygrin.`**`period`**(*grad*, *length*)
Period or pitch of a grin lens based on its gradient and length.

Parameters

- **grad** – geometric gradient of the lens [1/mm]
- **length** – length of grin lens [mm]

Returns *float* – the pitch or period of the grin lens [unitless]

`pygrin.pygrin.`**`plot_principal_planes`**(*n_0*, *pitch*, *length*, *diameter*)

Create a plot for a grin lens showing the cardinal points.

Parameters

- **n_0** – index of refraction at center of grin lens [unitless]
- **pitch** – pitch or period of the lens [unitless]
- **length** – axial length of the lens [mm]
- **diameter** – diameter of the lens [mm]

3.6 Changelog

3.6.1 v0.4.4

- create pure python packages
- include .whl file
- package as python3 only

3.6.2 v0.4.3

- improve help(pygrin)
- improve html at https://pygrin.readthedocs.io
- use google docstrings again

3.6.3 v0.4.1

- update function parameters for sphinx automodapi

3.6.4 v0.4.0

- sphinx documentation
- *plot_principal_plt* -> *plot_principal_planes*

3.6.5 v0.3.0

- renamed *pitch()* to *period()* to avoid stepping on the name.
- improve readme
- fix *pylint* warnings

- fix *pep257* warnings
- fix *pyroma* warnings

3.6.6 v0.2.0

- improve description
- add *hyperbolic_secant_profile_index*
- fix importing, update for new names

3.6.7 v0.2.0

- fix typos
- change status to alpha

Python Module Index

p

Index