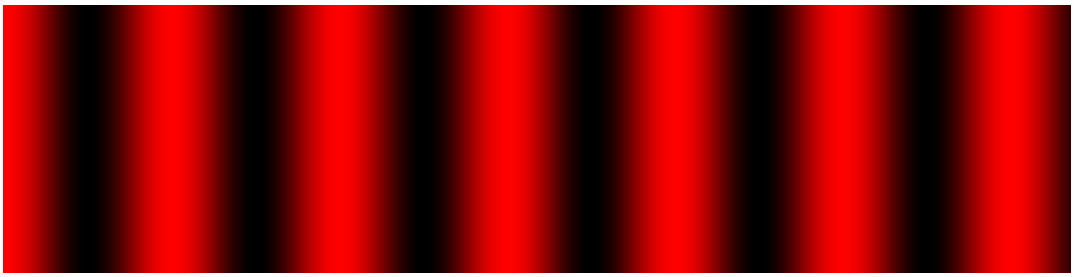
Interférences et images

Qqs fonctions rapidement pour montrer ce qu'on peut faire avec PIL dans le chapitre interférences

In [34]:

```
from PIL import Image
from math import cos
def monochromatique():
    img = Image.new("RGB",(400, 100))
    for j in range(100):
        for i in range(400):
            img.putpixel((i, j), (int(255*cos(i/20)**2), 0,0))
    return img

monochromatique()
```

Out [34]:



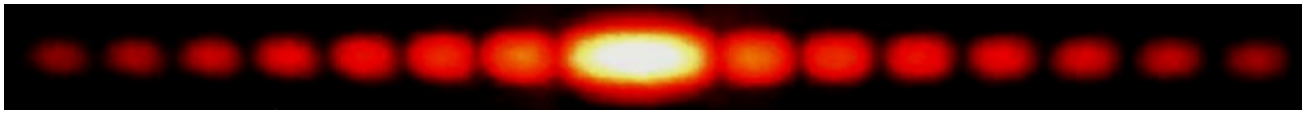
In [35]:

```
import matplotlib.pyplot as plt
import numpy as np

NBVALS = 200

def get_values(image, nb):
    h = image.height
    w = image.width
    out = list()
    for i in range(nb):
        pixel = image.getpixel((
            int(i*w/nb),
            int(h/2)
        ))
        out.append(pixel[0]+pixel[1]+pixel[2])
    return out
```

In [36]:

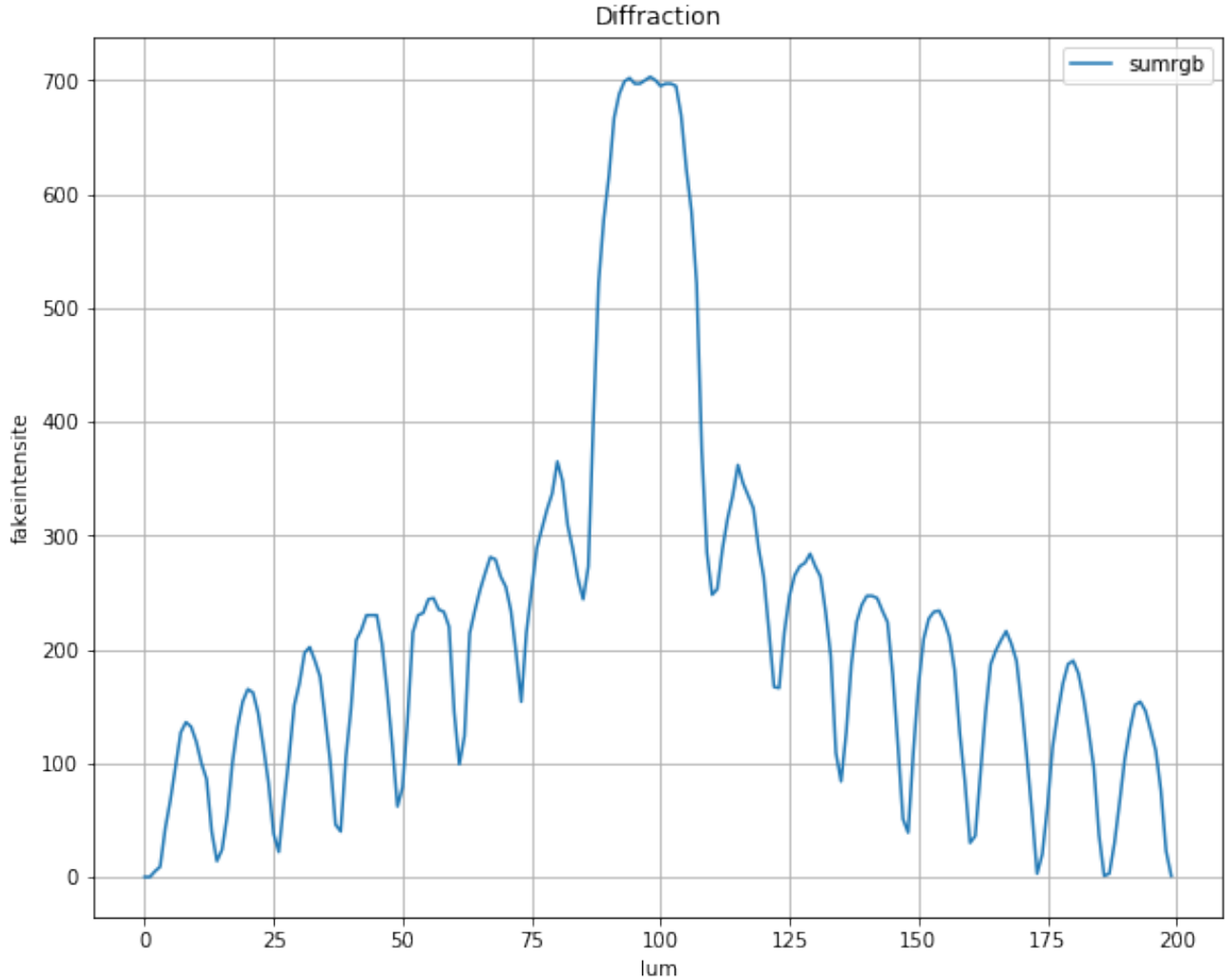
```
# Image pourrie chopées d'un jpeg en ligne, à reprendre avec une vraie image de CCD
img = Image.open("./images/diffraction.png")
img
```

Out [36]:



In [37]:

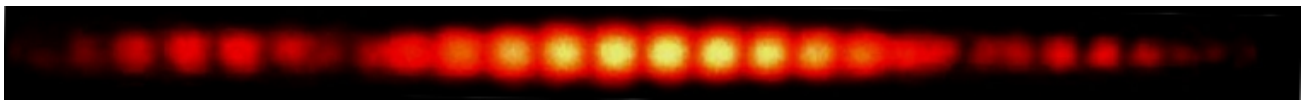
```
values = get_values(img, NBVALS)
plt.figure (figsize = (10,8))
plt.plot(range(NBVALS), values,label="sumrgb")
plt.xlabel("lum")
plt.ylabel("fakeintensite")
plt.legend()
plt.grid()
plt.title ("Diffraction")
plt.show()
```



In [38]:

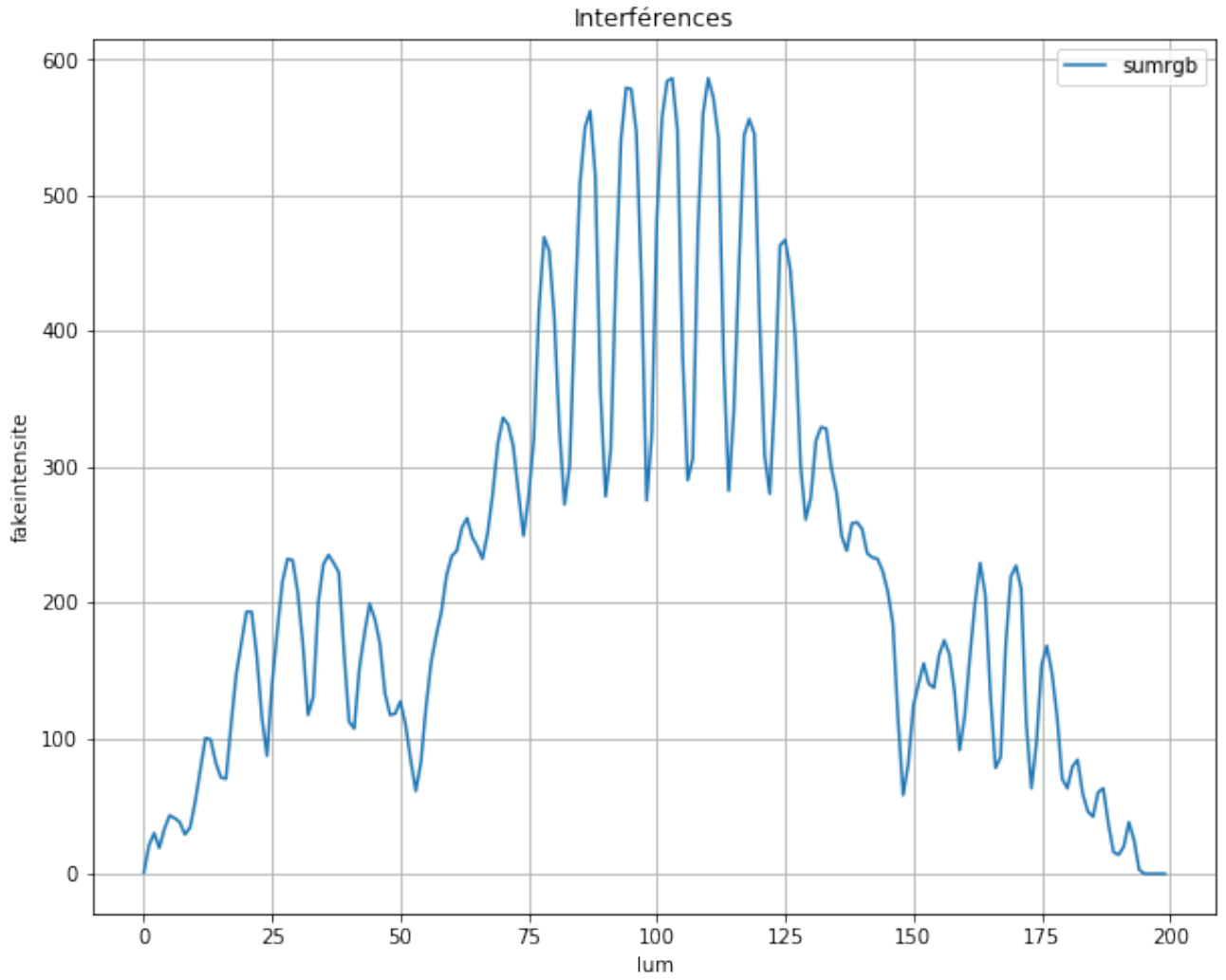
```
# Idem, image pourrie chopées d'un jpeg en ligne, à reprendre avec une vraie image de CCD
img = Image.open("./images/interferences.png")
img
```

Out [38]:



In [39]:

```
values = get_values(img, NBVALS)
plt.figure (figsize = (10,8))
plt.plot(range(NBVALS), values,label="sumrgb")
plt.xlabel("lum")
plt.ylabel("fakeintensite")
plt.legend()
plt.grid()
plt.title ("Interférences")
plt.show()
```



In [40]:

```
# A RETRAVAILLER :)
```

In [ ]:

```

```