TP 2 : Lois de Descartes

Code Capytale : 4381-7013801

Entrée[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import math
```

Résultats expérimentaux

Définir une liste contenant les valeurs de i1 en radians

Entrée[2]:
```
i1=[0,10,20,30,40,50,60,70,80]
i1rad=[math.radians(angle) for angle in i1]
```

Définir une liste contenant les valeurs de i2 en radians

Entrée[3]:
```
i2=[0,6.5,13,18.5,25,32.5,35,38.5,40]
i2rad=[math.radians(angle) for angle in i2]
```

Vérification de la loi de réfraction

Entrée[4]:
```
plt.clf() #pour réinitialiser le tracé à chaque nouvelle exécution
plt.plot(np.sin(i2rad),np.sin(i1rad),'o')
plt.xlabel('sin(i2)')
plt.ylabel('sin(i1)')
plt.grid()
plt.axhline()
plt.axvline()
plt.show()
```

Détermination de l'indice de réfraction du plexiglas

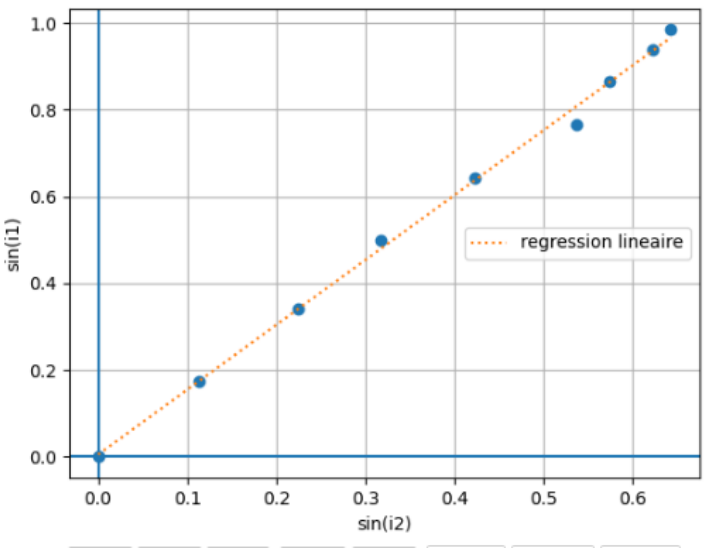
Graphiquement

Entrée[5]:
```
x=np.sin(i2rad)
y=np.sin(i1rad)
[a,b] = np.polyfit (x,y,1)
print("n = {:.5f}".format(a))
```

```
n = 1.49697
```

Entrée[6]:
```
ymodel=a*x+b
plt.plot(x,ymodel,':',label='regression lineaire')
plt.legend()
plt.show()
```

Figure 1



Etude statistique

Entrée[7]:
```
n=[]
for k in range(1,len(i1)):
    n.append(np.sin(i1rad[k])/np.sin(i2rad[k]))
nmoy=np.mean(n)
print(nmoy)
np.std(n,ddof=1)
```

```
1.5160385540364976
```

Sortie[7]: 0.04213297678578562

Réflexion totale

Entrée des données

Entrée[8]:
```
il=math.radians(42.5)
u=math.radians(1)/np.sqrt(6) # incertitude de double lecture sur le rapporteur gradué
delta_il=u*np.sqrt(3) # calcul de la précision
```

Simulation répétée 10000 fois

Entrée[9]:
```
il_sim=np.random.uniform(il-delta_il,il+delta_il,10000) # tirage au sort aléatoire de 10000 valeurs de il dans l'intervalle [il-
n=1/np.sin(il_sim) #liste des valeurs de n correspondantes
u_n=np.std(n,ddof=1)
print("u(n) = {:.5f}".format(u_n))
```

```
u(n) = 0.01149
```

Résultat

Entrée[10]:
```
n=1/np.sin(il)
print("n = {:.3f}".format(n),"+- {:.3f}".format(u_n))
```

```
n = 1.480 +- 0.011
```