TP – Focométrie

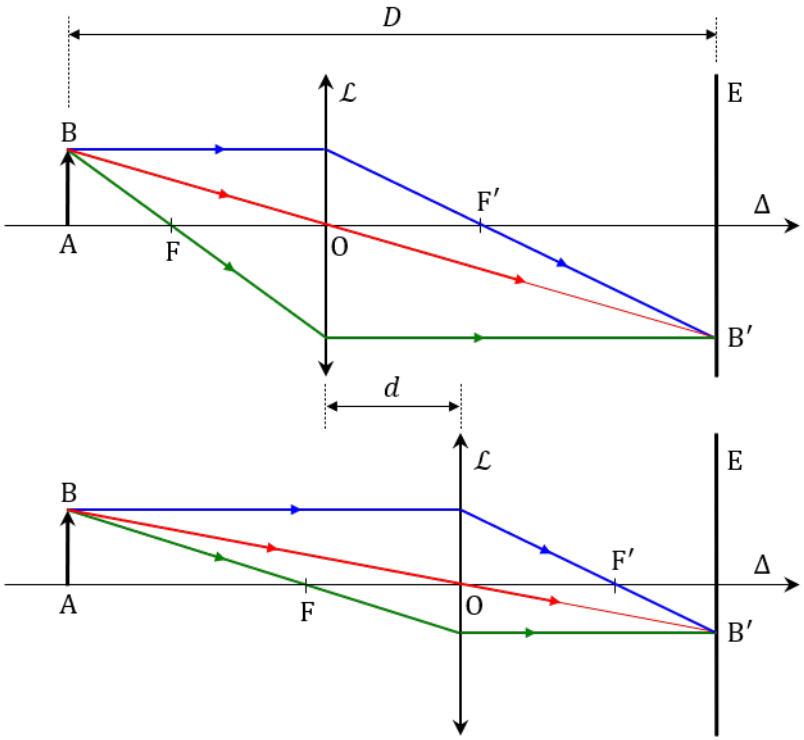
NOM :

Dans ce TP, nous allons apprendre à mesurer la distance focale de lentilles convergentes et divergentes par différentes méthodes, puis à estimer et comparer la précision de ces méthodes.

I) Méthode de Bessel

1) Rappels théoriques

La **méthode de Bessel** permet une mesure précise de la distance focale d'une lentille mince convergente f'.



Soit un objet AB et un écran E espacés d'une distance D. Si $D > 4f'$, alors il existe deux positions (cf. schéma ci-contre) de la lentille permettant d'observer sur l'écran l'image $A'B'$. La distance entre ces positions est notée d. On peut montrer que :

$$f' = \frac{D^2 - d^2}{4D}$$

2) Mesures

Réaliser le protocole décrit ci-dessous.

(1) Fixer la position de l'objet. On note x_A sa position lue sur le banc d'optique.

(2) Fixer la position de l'écran, tout en respectant le critère $D > 4f'$ (à vérifier qualitativement, puisque l'on ne connaît pas encore la valeur de f'). On note $x_{A'}$ sa position lue sur le banc d'optique.

(3) Mesurer les positions x_1 et x_2 de la lentille permettant d'observer l'image de l'objet sur l'écran.

(4) Recommencer les points (2) et (3) pour 7 valeurs de D différentes.

x_A (cm)	$x_{A'}$ (cm)	x_1 (cm)	x_2 (cm)

Estimer (en justifiant) le demi-intervalle de confiance $\Delta(x)$, puis l'incertitude-type $u(x)$ pour une mesure. Par simplicité, on prendra la même incertitude pour toutes les mesures.

3) Analyse d'une mesure unique – incertitudes de type B

Dans cette partie, nous allons déterminer f' à l'aide de la première mesure uniquement (et non pas de l'ensemble des 7 mesures).

- Compléter le code Python pour déterminer la valeur de f'.
- *Cette question est à traiter en fin de TP en cas de temps restant.* À l'aide d'une simulation de Monte-Carlo, déterminer $u(f')$. Écrire f' et son incertitude-type avec le bon nombre de chiffres significatifs.

4) Analyse statistique – incertitudes de type A

Dans cette partie, nous allons faire une analyse statistique des 7 mesures de $\{f'_n\}$. La valeur de f' ainsi obtenue devrait être plus précise que dans la partie précédente.

- Calculer la valeur moyenne de la série de mesures et son incertitude-type. Exprimer le résultat avec un nombre adéquat de chiffres significatifs.

5) Analyse par régression linéaire

Dans cette dernière partie, nous allons obtenir la valeur de f' par régression linéaire. On remarque que :

$$f' = \frac{D^2 - d^2}{4D} \quad \Rightarrow \quad \underbrace{D^2 - d^2}_{y} = \underbrace{f'}_{a} \times \underbrace{4D}_{x}$$

- Utiliser la fonction `polyfit` de la bibliothèque `numpy` pour effectuer une régression affine. En déduire la valeur de f'. On ne cherchera pas à estimer l'incertitude-type.

II) Mesure de la focale d'une lentille divergente

1) Auto-collimation à deux lentilles

Il est possible de montrer qu'un système de deux lentilles minces $\mathcal{L}_1$ et $\mathcal{L}_2$ accolées (i.e. de même centre optique), de vergence respective V_1 et V_2, est équivalent à une lentille unique $\mathcal{L}$ de vergence :

$$V = V_1 + V_2 \quad \Rightarrow \quad \frac{1}{f'} = \frac{1}{f'_1} + \frac{1}{f'_2} \quad \Rightarrow \quad \boxed{f' = \frac{f'_1 \times f'_2}{f'_1 + f'_2}}$$

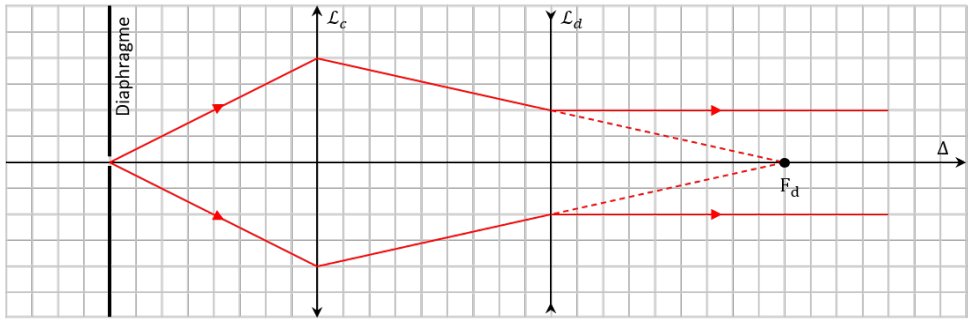
On souhaite accoler une lentille convergente $f'_c > 0$ et une lentille divergente $f'_d < 0$ afin que le doublet se comporte comme une lentille convergente.

- Quelle relation doivent vérifier f'_c et f'_d pour que cette propriété soit vérifiée ?
- Déterminer f'_d par auto-collimation.

2) Méthode du faisceau de rayons parallèles

On considère le montage ci-dessous, comprenant une source lumineuse, un diaphragme servant de petit objet circulaire, une lentille convergente et une lentille divergente.

- Réaliser le montage sans la lentille divergente. Repérer la position de l'image du diaphragme.
- Placer la lentille divergente afin que le faisceau émerge parallèle, i.e. que la tâche lumineuse sur l'écran ne change pas de diamètre, quelle que soit la position de l'écran. En déduire la distance focale de la lentille divergente.



Aides pour Python

`import numpy as np` permet d'importer l'ensemble des fonctions du module numpy.

`len(u)` renvoie le nombre d'éléments contenus dans `u`.

`np.array(u)` crée un tableau numpy contenant les éléments de `u`.

`np.arange(a, b, p)` crée un tableau numpy contenant les valeurs comprises entre `a` (inclus) et `b` (exclu), régulièrement espacées du pas `p`.

`np.linspace(a, b, N)` crée un tableau numpy contenant `N` valeurs régulièrement espacées et comprises entre `a` (inclus) et `b` (inclus).

`a, b = np.polyfit(x, y, 1)` stocke dans les variables `a` et `b` le résultat de la régression affine $y = ax + b$.

`np.random.normal(moy, std, N)` crée un tableau numpy contenant `N` valeurs tirées aléatoirement selon une loi normale de valeur moyenne `moy` et d'écart-type `std`.

`np.mean(u)` et `np.average(u)` calculent la valeur moyenne du tableau numpy `u`.

`np.std(u, ddof=1)` calcule l'écart-type du tableau numpy `u`.

`np.sqrt(x)` calcule la racine carrée du nombre `x`.