

Calcul d'une intégrale par la méthode des rectangles

On s'intéresse à l'aire comprise sous la représentation graphique d'une fonction définie sur l'ensemble des réels par :

$$f(x) = \frac{1}{7\sqrt{2\pi}} e^{-\frac{(x-170)^2}{98}}$$

Cette fonction étant très complexe à primitiver, nous allons programmer une fonction en Python qui nous permettra d'aboutir à une bonne approximation de la valeur recherchée.

Etude de la fonction

1. Calculer les limites de la fonction $f(x)$ aux bornes de l'ensemble de définition.

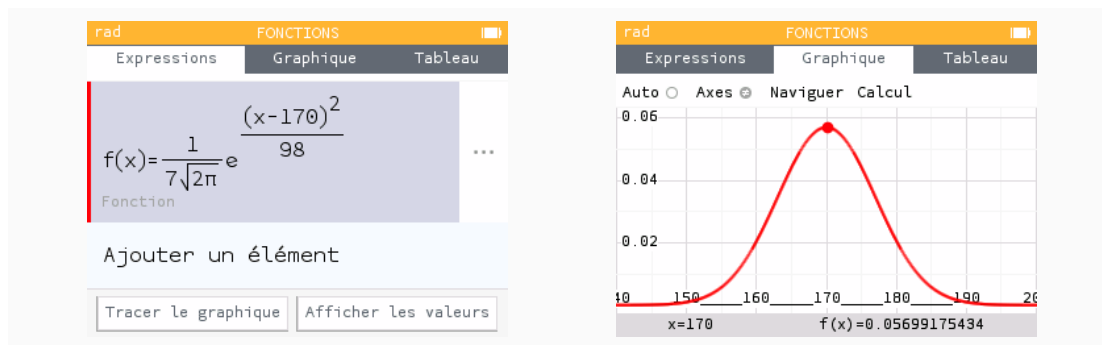
$$\lim_{x \rightarrow -\infty} f(x) = \lim_{x \rightarrow +\infty} f(x) = 0$$

2. Dresser le tableau de variation de la fonction.

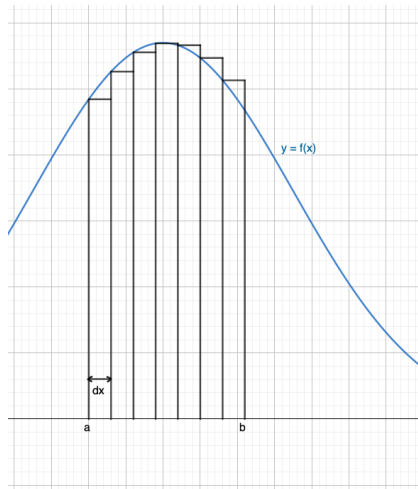
On calcule $f'(x) = \frac{1}{7\sqrt{2\pi}} \times \frac{-2(x-170)}{98} e^{-\frac{(x-170)^2}{98}}$.

La fonction est croissante lorsque $x < 170$ et décroissante pour $x > 170$.

3. Représenter graphiquement la courbe sur l'intervalle $[140; 200]$.



Approximation par la méthode des rectangles



On procède à un découpage de la courbe par rectangles de largeur identique dx , tout comme sur le dessin ci-contre.

L'objectif est d'obtenir une approximation de la surface comprise sous la courbe entre a et b en faisant la somme des aires des rectangles.

1. Que faut-il faire pour que l'approximation obtenue soit la plus proche possible du résultat recherché ?

Pour que le résultat soit le plus précis possible, il faut augmenter le nombre de rectangles et donc diminuer la largeur de chacun d'entre eux.

2. On considère l'aire comprise sous la courbe $f(x)$ entre deux abscisses a et b . Exprimer la largeur dx en fonction de a , b et le nombre n de rectangles découpant cet espace.

$$dx = \frac{b - a}{n}$$

3. Exprimer en fonction de f la hauteur du premier rectangle au point d'abscisse a . Quelle est la hauteur du second rectangle situé au point d'abscisse $a + dx$?

Le premier rectangle à partir du point d'abscisse a admet pour hauteur $f(a)$. Le second rectangle à partir du point d'abscisse $a + dx$ admet pour hauteur $f(a + dx)$.

4. On propose d'écrire une fonction `approx_rect(n, x_inf, x_sup)` permettant de calculer l'intégrale d'une fonction $f(x)$ (que l'on aura préalablement définie dans une autre fonction) entre deux bornes x_{inf} et x_{sup} par un découpage en n rectangles.

On utilisera dans ce programme deux variables : `x`, qui évolue entre les deux bornes, et `somme`, qui stocke la somme des rectangles au cours du calcul. Comment sont initialisées ces deux variables au début du programme ?

Au début du programme, `x=x_inf` et `somme=0`.

5. Compléter l'algorithme suivant afin d'obtenir le résultat souhaité :

```
1 from math import *
2 def f(x):
```

```

3     return (1/(7*sqrt(2*pi)))*exp(-((x-170)**2)/98)
4 def approx_rect(n,x\_inf,x\_sup):
5     somme= ...
6     x= ...
7     dx= ...
8     while(x< ... ):
9         somme=somme+dx*f(x)
10        x+=dx
11    return ...

```

```

1 from math import *
2 def f(x):
3     return (1/(7*sqrt(2*pi)))*exp(-((x-170)**2)/98)
4 def approx_rect(n,x\_inf,x\_sup):
5     somme=0
6     x= x\_inf
7     dx = (x\_sup-x\_inf)/n
8     while(x<x\_sup):
9         somme=somme+dx*f(x)
10        x+=dx
11    return somme

```

Exécution de l'algorithme et loi normale

1. Quelle valeur obtient-on lorsque l'on exécute le programme pour les valeurs $n = 5000$, $x_{inf} = 120$ et $x_{sup} = 230$?

Le programme retourne 1.

La fonction $f(x)$ est ce que l'on appelle une **fonction de densité**, c'est-à-dire qu'il s'agit d'une fonction continue et positive sur un intervalle et telle que son intégrale sur cet intervalle est égal à 1. Dans ce cas particulier, la courbe en cloche que l'on obtient lorsque l'on représente graphiquement cette fonction est appelée **courbe de Gauss**.

Les courbes de Gauss sont utilisées car elle traduisent une réalité statistique souvent observée sur des grandes populations pour des phénomènes aléatoires : les effectifs se répartissent autour d'une valeur centrale en suivant une courbe en cloche dont on peut connaître l'équation. C'est vrai lorsque l'on s'intéresse à la masse exacte d'un grand nombre de boîtes d'une même conserve, ou à la taille des chats adultes en région parisienne. On dit que leur répartition suit une **loi normale**.

2. Dans notre cas, la fonction de densité $f(x)$ permet d'étudier la répartition des tailles des adultes dans la population française, de moyenne $\mu = 170$ (cm) et d'écart-type $\sigma = 7$ (cm).
Comment interpréter le nombre obtenu lorsqu'on a exécuté le programme entre 120 et 230 ?

Cela signifie que l'aire se trouvant sous la courbe est égale à 1. Etant donné qu'il s'agit d'une fonction de densité permettant d'étudier la répartition des tailles dans la population française, on peut supposer que les personnes mesurant entre 120 et 230 cm représentent (approximativement) 100% de la population.

3. Exécuter le programme pour $n = 5000$, $x_{inf} = \mu - \sigma$ et $x_{sup} = \mu + \sigma$. Comment interpréter ce résultat ?

```
rad PYTHON
>>> from approximation_integra
>>> approx_rect(5000, 163, 177)
0.6826894856839875
>>> |
```

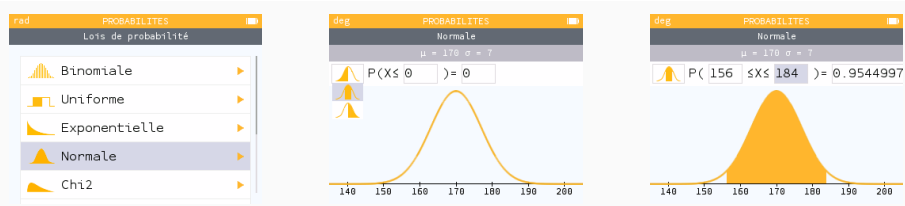
L'aire se trouvant sous la courbe entre 163 et 177 est égale à environ 0,68. On en déduit qu'approximativement 68% des français mesurent entre 163 et 177 cm.

4. Même question pour une exécution entre $x_{inf} = \mu - 2\sigma$ et $x_{sup} = \mu + 2\sigma$.

```
rad PYTHON
>>> from approximation_integra
>>> approx_rect(5000, 156, 184)
0.9545429173605008
>>> |
```

L'aire se trouvant sous la courbe entre 156 et 184 est égale à environ 0,95. On en déduit qu'approximativement 95% des français mesurent entre 156 et 184 cm.

5. Rendez-vous maintenant dans l'application Probabilités de la calculatrice : sélectionner Normale dans le menu puis entrer les paramètres μ et σ définis plus haut. Modifier l'icône en haut à gauche de façon à sélectionner le second dessin, puis encadrer X entre 156 et 184. Et maintenant... admirez !



Un peu plus de Python pour les plus rapides !

Ecrire une fonction `approx_trapeze(n, x_inf, x_sup)` permettant une autre approximation de la surface située sous la courbe par un découpage en trapèzes.

Un script est disponible sur cette page^a.

^a. https://my.numworks.com/python/elodie-gamot/approximation_integrale