Fonctions_Arithmetiques

January 22, 2019

1 Quelques fonctions arithmétiques

Marc Lorenzi
21 janvier 2019

```
In [1]: import matplotlib.pyplot as plt
        import math

In [2]: plt.rcParams['figure.figsize'] = (10, 6)
```

1.1 1 Préliminaires

1.1.1 1.1 De quoi allons-nous parler ?

Une fonction arithmétique est une fonction définie sur l'ensemble $\mathbb{N}^*$ des entiers naturels non nuls, à valeurs dans $\mathbb{R}$ ou $\mathbb{C}$. Évidemment, vu comme ça, vous me direz : bon, une fonction arithmétique c'est une suite ?

Oui, c'est une suite. Mais nous allons examiner ces fonctions différemment. Certaines fonctions arithmétiques jouent un rôle essentiel en théorie des nombres. Leur étude approfondie permet de démontrer des théorèmes célèbres, comme par exemple le

Théorème des nombres premiers : Pour tout entier $n \geq 1$, soit $\pi(n)$ le nombre de nombres premiers inférieurs à n. Lorsque n tend vers l'infini, on a

$$\pi(n) \sim \frac{n}{\ln n}$$

Nous allons nous concentrer dans ce qui suit sur quelques fonctions arithmétiques. Nous allons également définir une opération sur les fonctions arithmétiques, le **produit de Dirichlet**, qui permet d'obtenir des relations "magiques" entre les différentes fonctions que nous introduirons.

Avant tout cela, il nous sera nécessaire de pouvoir factoriser des entiers, rechercher tous leurs diviseurs, calculer des valuations p-adiques. Nous allons mettre en place quelques fonctions, inefficaces pour de grands entiers mais suffisantes pour nos petits besoins.

NOTE : Le sujet des fonctions arithmétiques est immense. Il est l'un des fondements de la **théorie analytique des nombres**. Chacune des fonctions dont nous allons parler mériterait un notebook à elle seule. J'ai donc fait des choix très arbitraires. Concernant les aspects mathématiques, certaines propriétés seront montrées. D'autres seront admises, soit pour des raisons de longueur, soit à cause de leur difficulté.

1.1.2 1.2 Plus petit diviseur d'un entier

Soit $n \geq 2$. Supposons que n est composé. Il existe donc $a, b \in [2, n[$ tels que $n = ab$. Supposons que $a^2 > n$. On a alors $b^2 = \left(\frac{n}{a}\right)^2 = \frac{n^2}{a^2} < \frac{n^2}{n} = n$. Dit autrement, $a^2 \leq n$ ou $b^2 < n$. Ainsi, n a un diviseur inférieur ou égal à $\sqrt{n}$.

Soit p le plus petit diviseur de n différent de 1 et n. D'après la remarque précédente, $p^2 \leq n$. La fonction `plus_petit_diviseur` ci-dessous exploite ce résultat. Elle effectue une boucle `while`. Si un tel diviseur n'est pas trouvé, c'est que n est premier et la fonction renvoie n. La complexité de `plus_petit_diviseur` est en $O(\sqrt{n})$, en nombre d'opérations sur des entiers. Inutile donc d'en attendre des miracles mais elle suffira pour des entiers de taille raisonnable, disons inférieurs à un milliard.

Exercice : Soit $n \geq 2$. Montrer que le plus petit diviseur de n différent de 1 est premier.

```
In [3]: def plus_petit_diviseur(n):
            p = 2
            while p * p <= n and n % p != 0: p += 1
            if p * p > n: return n
            else: return p

In [4]: plus_petit_diviseur(12345678910111213)

Out[4]: 113
```

Il est maintenant évident d'écrire une fonction `est_premier`. Cette fonction a aussi une complexité en $O(\sqrt{n})$.

```
In [5]: def est_premier(n):
            if n <= 1: return False
            else: return plus_petit_diviseur(n) == n

In [6]: print([n for n in range(1000) if est_premier(n)])

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97,
```

1.1.3 1.3 Factorisation

Définition : Soit $n \geq 1$. Soit p un nombre premier. La valuation p-adique de n, $v_p(n)$, est le plus grand entier naturel a tel que p^a divise n.

La fonction `valuation`prend deux entiers n et p en paramètres et renvoie $v_p(n)$.

```
In [7]: def valuation(n, p):
            a = 0
            while n % p == 0:
                n = n // p
                a = a + 1
            else: return a

In [8]: print(valuation(100000, 2))
```

5

La fonction `facteurs`, enfin, renvoie une factorisation de n en produit de nombres premiers. Si $n = 1$ elle renvoie la liste vide. Si $n \geq 2$ s'écrit $\prod_{i=1}^{k} p_i^{a_i}$ où les p_i sont premiers et distincts, et les a_i sont des entiers non nuls, la fonction renvoie la liste des couples (p_i, a_i) avec les p_i rangés dans l'ordre croissant.

```
In [9]: def facteurs(n):
            s = []
            while n > 1:
                p = plus_petit_diviseur(n)
                a = valuation(n, p)
                s.append((p, a))
                n = n // (p ** a)
            return s

In [10]: facteurs(1234567891011112132)

Out[10]: [(2, 2), (3, 1), (1223, 1), (2357, 1), (3569009401, 1)]

In [11]: facteurs(2019)

Out[11]: [(3, 1), (673, 1)]
```

Quelle est la complexité C_n de la fonction de factorisation ? Soit $n = \prod_{i=1}^{k} p_i^{\alpha_i}$ la décomposition de n où $p_1 < p_2 < \ldots < p_k$ et les $\alpha_i \geq 1$.

Proposition : $C_n = O(\sqrt{n} \log n)$.

Démonstration : Considérons un certain nombre de cas.

1. n est premier. La fonction `plus petit_diviseur` renvoie le bon résultat en $O(\sqrt{n})$ opérations.
2. $n = p^\alpha$, où $\alpha \geq 2$ est une puissance d'un nombre premier. Le résultat est renvoyé en $O(\sqrt{n} + \alpha)$ opérations. Que dire de α ? On a $p \geq 2$ donc $n = p^\alpha \geq 2^\alpha$. Ainsi, $\alpha \leq \lg n$. Donc $C_n = O(\sqrt{n})$.
3. Il y a au moins deux nombres premiers distincts qui divisent n. Combien au plus ? Soit k le nombre de facteurs premiers de n. On a $n \geq \prod_{i=1}^{k} p_i \geq 2^k$ donc $k \leq \lg n$. Chacun des p_i est trouvé en au plus (majoration très grossière) $\sqrt{n}$ opérations, puis l'exposant de p_i dans la décomposition de n est trouvé en au plus $\lg n$ opérations, par le même raisonnement que celui fait dans le cas 2. Au total, le nombre d'opérations est $O((\sqrt{n} + \lg n) \lg n) = O(\sqrt{n} \lg n)$ opérations.

Ce n'est pas très bon, il faut l'avouer. Remarquons tout de même que si les facteurs premiers de n sont "petits" (disons inférieurs à un certain entier N, alors $C_n = O(N \lg n)$ et la factorisation de n sera faite de façon "instantanée". Enfin bon, à condition que n soit de taille "raisonnable". Évitons les nombres à un million de chiffres :-).

1.1.4 1.4 Diviseurs d'un entier

Soit $n \geq 2$. Comment obtenir tous les diviseurs de n ?

Soit p un diviseur premier de n. Soit a la valuation p-adique de n. Soit $n' = \frac{n}{p^a}$. Les diviseurs de n sont :

- Les diviseurs de n'
- p fois les diviseurs de n'
- p^2 fois les diviseurs de n'
- ...
- p^a fois les diviseurs de n'

Une fonction récursive s'impose. La fonction `diviseurs` fait le travail.

```
In [12]: def diviseurs(n):
             if n == 1: return [1]
             else:
                 p = plus_petit_diviseur(n)
                 a = valuation(n, p)
                 ds = diviseurs(n // p ** a)
                 ds1 = [p ** k * d for d in ds for k in range(a + 1)]
                 ds1.sort()
             return ds1
```

Quels sont les diviseurs de 100 ?

```
In [13]: print(diviseurs(100))
```

```
[1, 2, 4, 5, 10, 20, 25, 50, 100]
```

Quels sont les diviseurs de 10! ?

```
In [14]: print(diviseurs(3628800))
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 15, 16, 18, 20, 21, 24, 25, 27, 28, 30, 32, 35, 36, 40,
```

1.2 2 Les fonctions d'Euler et de Möbius

1.2.1 2.1 Fonctions multiplicatives

Définition : Soit f une fonction arithmétique.

- On dit que f est **multiplicative** lorsque pour tous a, b tels que $a \wedge b = 1$ on a $f(ab) = f(a)f(b)$.
- Lorsque la propriété est vraie pour tous a, b sans aucune condition de pgcd, on dit que f est **complètement multiplicative**.

Les fonctions arithmétiques que nous allons étudier sont presque toutes multiplicatives. Mais commençons par donner quelques exemples, essentiels pour la suite, de fonctions complètement multiplicatives.

Proposition : Les fonctions $\mathbb{1}, \delta, id, N_\alpha$ définies ci-dessous sont complètement multiplicatives. - Étant donné $\alpha \in \mathbb{R}$, pour tout $n \geq 1$, $N_\alpha(n) = n^\alpha$. - Pour tout $n \geq 1$, $\mathbb{1}(n) = N_0(n) = 1$. - Pour tout $n \geq 1$, $id(n) = N_1(n) = n$. - $\delta(1) = 1$ et pour tout $n \geq 2$, $\delta(n) = 0$.

Preuve : Facile. Notez les définitions de ces fonctions sur un bout de papier parce qu'elles réapparaîtront de temps en temps, sans prévenir. Ne me demandez pas dans une heure : "C'est qui, $\mathbb{1}$" :-) ?

```
In [15]: def un(n): return 1

         def delta(n):
             if n == 1: return 1
             else: return 0

         def id(n): return n
```

1.2.2 2.2 La fonction μ de Möbius

Définition : On pose $\mu(1) = 1$. Pour tout $n \geq 2$, on pose $\mu(n) = (-1)^k$ si n est le produit de k nombres premiers distincts, et $\mu(n) = 0$ sinon.

```
In [16]: def mu(n):
             fs = facteurs(n)
             m = 1
             for (p, a) in fs:
                 if a > 1: return 0
                 else: m = -m
             return m

In [17]: [mu(n) for n in range(1, 20)]

Out[17]: [1, -1, -1, 0, -1, 1, -1, 0, 0, 1, -1, 0, -1, 1, 1, 0, -1, 0, -1]
```

La fonction μ a vraiment l'air de rien, mais elle est **la clé**.

Proposition : La fonction μ est multiplicative.

Démonstration : Soient $m, n \geq 1$. Supposons $m \wedge n = 1$. Si $m = 1$ ou $n = 1$ il est clair que $\mu(mn) = \mu(m)\mu(n)$. Supposons donc $m, n \geq 2$.

- Si $\mu(m) = 0$, il existe un nombre premier p et un entier $a \geq 2$ tel que $p^a | m$. Mais alors $p^a | mn$, donc $\mu(mn) = 0 = \mu(m)\mu(n)$. Idem si $\mu(n) = 0$.
- Pour terminer, si $m = p_1 \dots p_k$ et $n = q_1 \dots q_l$ où les p_i, q_i sont premiers et distincts, alors $mn = p_1 \dots p_k q_1 \dots q_l$ donc $\mu(mn) = (-1)^{k+l} = (-1)^k(-1)^l = \mu(m)\mu(n)$.

Exercice : Où a-t-on utilisé que $m \wedge n = 1$?

Proposition : Soit $n \geq 2$. On a $\sum_{d|n} \mu(d) = 0$.

Démonstration : Écrivons $n = \prod_{i=1}^k p_i^{a_i}$ où les p_i sont premiers et distincts et les $a_i \geq 1$. Les diviseurs de n sont les entiers $d = \prod_{i=1}^k p_i^{b_i}$ où $0 \leq b_i \leq a_i$. Si l'un des b_i est supérieur ou égal à 2,

alors $\mu(d) = 0$. Ainsi, $\sum_{d|n} \mu(d) = \sum'_{d|n} \mu(d)$ où le "prime" signifie que l'on somme sur les d qui sont des produits de nombres premiers distincts.

Regroupons les termes de cette somme par "nombre de facteurs" :

$$\sum_{d|n} \mu(d) = \sum_{j=0}^{k} \sum_{d|n,(j)} \mu(d) = \sum_{j=0}^{k} (-1)^j \sum_{d|n,(j)} 1$$

où le (j) signifie que l'on somme sur les nombre qui ont exactement j facteurs premiers distincts. Combien y en a-t-il ? Il s'agit en fait de choisir j nombres premiers parmi les k nombres $p_1, \dots, p_k$. Il y en a donc $\binom{k}{j}$. Ainsi,

$$\sum_{d|n} \mu(d) = \sum_{j=0}^{k} (-1)^j \binom{k}{j} = (1-1)^k = 0$$

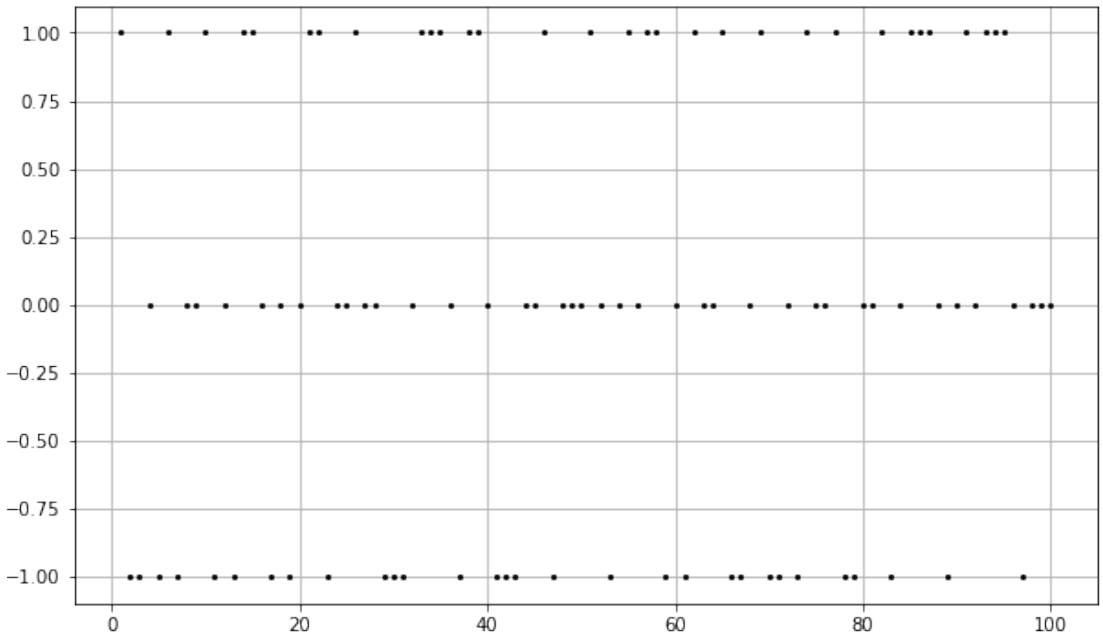
Remarque : La fonction arithmétique $n \mapsto \sum_{d|n} \mu(d)$ est donc nulle pour tout $n \geq 2$. De plus, elle vaut clairement 1 pour $n = 1$. Cela ne vous rappelle rien ? Avez-vous noté sur un bout de papier comme je vous l'ai demandé ? C'est la fonction δ.

Proposition : Pour tout $n \geq 1$, $\sum_{d|n} \mu(d) = \delta(n)$.

Dessinons μ. Comme on va dessiner beaucoup d'autres fonctions, écrivons d'abord une fonction `plot_fun` qui trace la fonction arithmétique f entre les entiers a et b, avec des paramètres `opt` et `opt2` pour la fonction `plt.plot`.

```
In [18]: def plot_fun(f, a, b, *opt, **opt2):
             xs = range(a, b + 1)
             ys = [f(x) for x in xs]
             plt.plot(xs, ys, *opt, **opt2)
             plt.grid()
             plt.savefig('fig.png')
             plt.show()

In [19]: plot_fun(mu, 1, 100, 'ok', markersize=2)
```



Oui, bon, bof, c'est plein de 0, de 1 et de -1 ... Bientôt, nous ferons des dessins plus intéressants.

1.2.3 2.3 La fonction indicatrice d'Euler

Définition : Pour tout $n \geq 1$, $\varphi(n)$ est le nombre d'entiers compris entre 1 et n (1 et n inclus) et premiers avec n. La fonction φ est appelée la **fonction indicatrice d'Euler**.

2.3.1 Euler pour les nuls La définition nous donne évidemment une (mauvaise) méthode pour calculer $\varphi(n)$:

```
In [20]: def phi(n):
             compte = 0
             for k in range(1, n + 1):
                 if pgcd(n, k) == 1: compte += 1
             return compte
```

où pgcd est la fonction ci-dessous, qui utilise l'algorithme d'Euclide.

```
In [21]: def pgcd(a, b):
             while b != 0:
                 a, b = b, a % b
             return a
```

Calculons $\varphi(n)$ pour les entiers n entre 1 et 20 :

```
In [22]: for n in range(1, 21):
             print('%3d%3d' % (n, phi(n)))
```

```
 1  1
 2  1
 3  2
 4  2
 5  4
 6  2
 7  6
 8  4
 9  6
10  4
11 10
12  4
13 12
14  6
15  8
16  8
17 16
18  6
19 18
20  8
```

Notre fonction a une complexité épouvantable, en gros $O(n \log n)$. Nous avons n calculs de `pgcd` à faire, et pour $1 \leq k \leq n$, le calcul de $k \wedge n$ se fait en $O(\log k)$ opérations. Or, $\sum_{k=1}^{n} \log k = O(n \log n)$. Bref, inacceptable. Si vous n'êtes pas convaincu, évaluez la cellule ci-dessous ... et attendez plusieurs secondes.

```
In [23]: phi(5000000)
```

```
Out[23]: 2000000
```

2.3.2 Faisons mieux **Proposition** : La fonction φ est multiplicative.

Démonstration : Admettons-le pour l'instant. Ce sera évident d'ici la fin du notebook.

Proposition : Pour tout nombre premier p et tout entier $\alpha \geq 1$, on a

$$\varphi(p^\alpha) = p^\alpha - p^{\alpha-1}$$

Démonstration : Quels sont les entiers entre 1 et p^α qui ne sont pas premiers avec p^α ? Ce sont les multiples de p : $p, 2p, 3p, \ldots, p^\alpha = p^{\alpha-1}p$. Il y en a $p^{\alpha-1}$. D'où le résultat.

Nous y sommes. Nous voilà en position d'une formule pour calculer $\varphi(n)$.

Proposition : Soit $n \geq 2$. Posons $n = \prod_{i=1}^{k} p_i^{\alpha_i}$. On a

$$\varphi(n) = \prod_{i=1}^{k} \left(p_i^{\alpha_i} - p_i^{\alpha_i - 1} \right)$$

Démonstration : Les $p_i^{\alpha_i}$ sont premiers entre-eux deux à deux et la fonction φ est multiplicative.

Remarque : Il existe plusieurs façons de regrouper les facteurs dans l'expression ci-dessus. Par exemple, on trouve fréquemment dans la littérature

$$\varphi(n) = n \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right)$$

Nous voici en possession d'un nouvel algorithme pour calculer $\varphi(n)$.

1. Décomposer n en produit de nombres premiers.
2. Appliquer la formule ci-dessus.

La complexité de notre nouvelle fonction est $O(\sqrt{n} \log n)$, ce qui est clairement mieux que le $O(n \log n)$ de notre premier jet. En plus, le $O(\sqrt{n} \log n)$ est très pessimiste pour beaucoup d'entiers n. Par exemple, si n n'a que de petits facteurs premiers, le calcul de $\varphi(n)$ est en $O(N \log n)$ où N est un majorant des facteurs premiers de n.

```
In [24]: def phi(n):
             fs = facteurs(n)
             m = 1
             for (p, a) in fs:
                 m = m * p ** (a - 1) * (p - 1)
             return m

In [25]: for n in range(1, 21):
             print('%3d%3d' % (n, phi(n)))

  1  1
  2  1
  3  2
  4  2
  5  4
  6  2
  7  6
  8  4
  9  6
 10  4
 11 10
 12  4
 13 12
 14  6
 15  8
 16  8
 17 16
 18  6
 19 18
 20  8


In [26]: phi(5000000)

Out[26]: 2000000
```

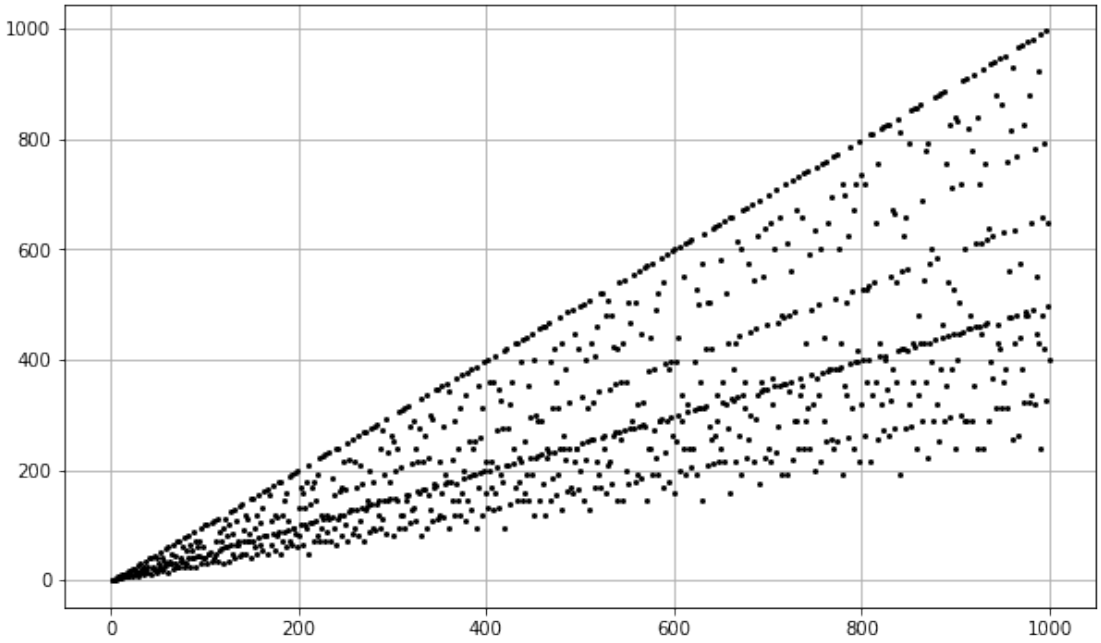
```
In [27]: print(est_premier(100000007))
         phi(100000007)
```

```
True
```

```
Out[27]: 100000006
```

Le graphe de φ ?

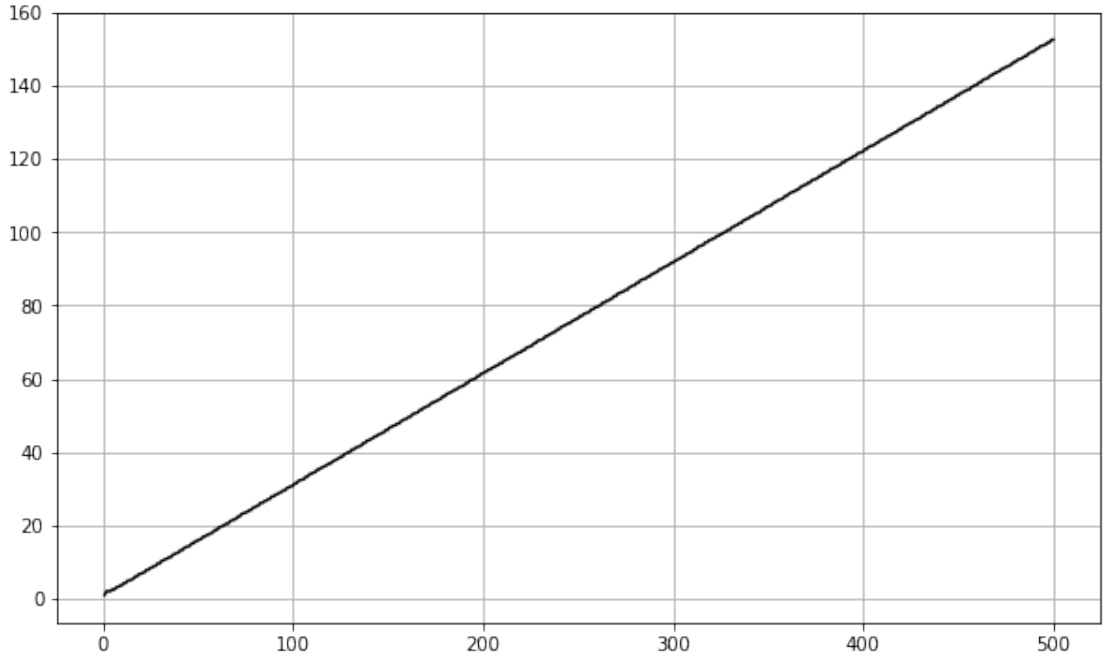
```
In [28]: plot_fun(phi, 1, 1000, 'ok', markersize=2)
```



Le graphe de la fonction φ est terrible. Mais derrière ce chaos apparent se cachent des propriétés remarquables. Un exemple ? Au lieu de tracer φ, traçons la **moyenne** de φ sur $[1, n]$, c'est à dire la fonction $\overline{\varphi} : n \mapsto \frac{1}{n}\sum_{k=1}^{n} \varphi(k)$.

```
In [29]: def plot_moyennes(f, n, g=None):
             s = [f(k) for k in range(1, n + 1)]
             for k in range(1, n):
                 s[k] = s[k] + s[k - 1]
             for k in range(1, n):
                 s[k] = s[k] / k
             xs = list(range(1, n + 1))
             if g != None:
                 ys = [g(x) for x in xs]
                 plt.plot(xs, ys, 'r')
             plt.plot(xs, s, 'k')
             plt.grid()
             plt.show()
```

```
In [30]: plot_moyennes(phi, 500)
```

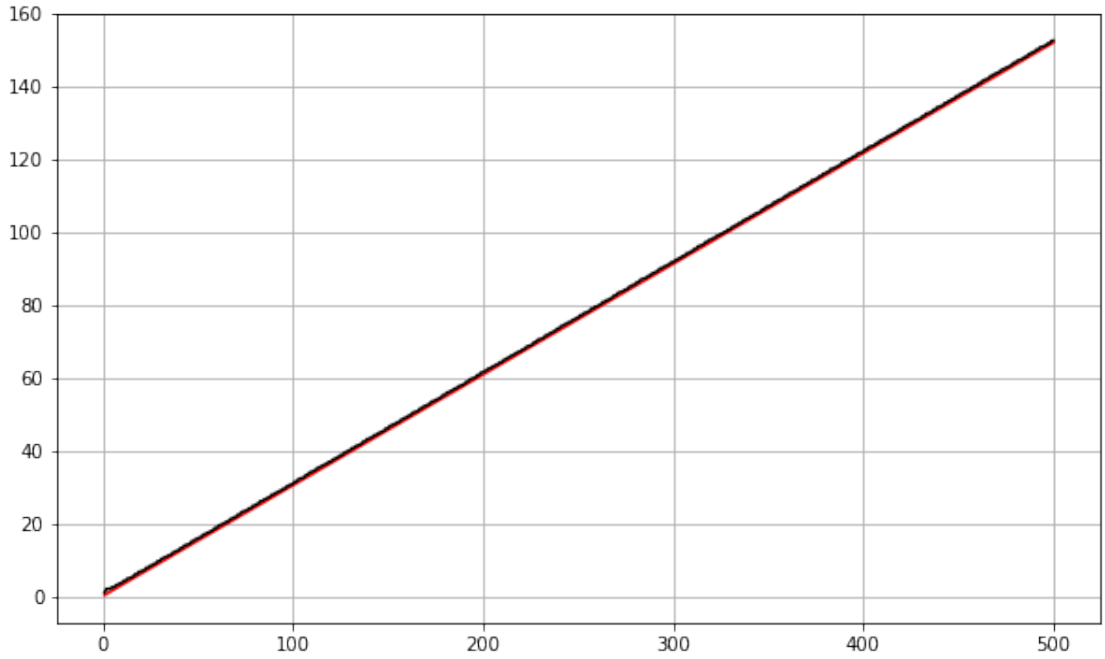


On dirait une droite . . .

Proposition : $\overline{\varphi}(n) \sim \frac{3}{\pi^2} n$.

Démonstration : La marge de ce notebook est trop étroite pour contenir la démonstration. Plus sérieusement, ce n'est pas *difficile*, c'est juste un peu long.

```
In [31]: plot_moyennes(phi, 500, lambda x:3 * x / math.pi ** 2)
```

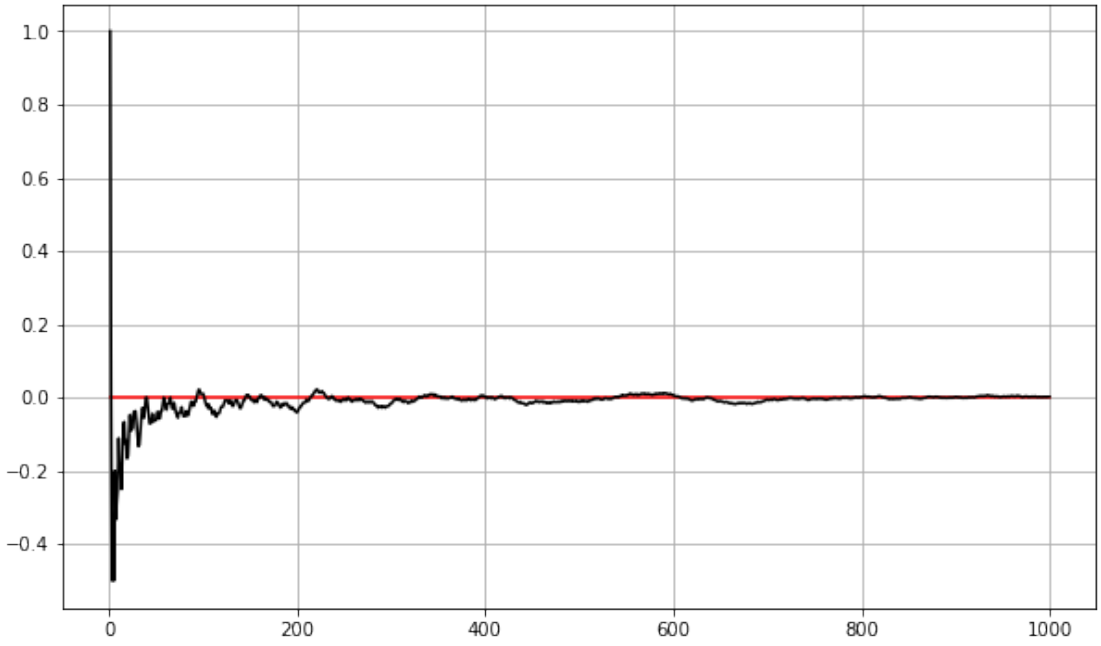


Les courbes noire et rouge sont quasi-superposées.

1.2.4 2.4 La valeur moyenne de μ

Une question nous vient soudain à l'esprit : quelle est la valeur moyenne de μ ? Faisons une petite expérience.

```
In [32]: plot_moyennes(mu, 1000, lambda x:0)
```



Bon, on voit. Étant donné que μ prend des valeurs $0, \pm 1$ "un peu n'importe comment" (selon nous), il ne doit pas être bien difficile de montrer que $\overline{\mu}(n) \to 0$ lorsque n tend vers l'infini. **Détrompez-vous** : on vient d'ouvrir la boîte de Pandore.

Proposition : $\overline{\mu}(n) \to 0$ lorsque n tend vers l'infini.

Proposition : La proposition précédente est équivalente au théorème des nombres premiers.

Considérons les deux propriétés (P_1) $\overline{\mu}(n) \to 0$ et (P_2) $\pi(n) \sim \frac{n}{\ln n}$. Dire que ces propriétés sont équivalentes veut dire qu'elles sont toutes les deux vraies ... ou toutes les deux fausses. Si ce n'était que cela, ce ne serait pas très intéressant. En fait nous avons une équivalence **constructive** entre P_1 et P_2 : Si l'on connaît une démonstration de l'une des deux, alors on connaît une démonstration de l'autre.

Fait subjectif : Montrer que $\overline{\mu} \to 0$ est aussi difficile que démontrer le théorème des nombres premiers .

1.3 3 Une opération sur les fonctions arithmétiques

Notons $\mathcal{A} = \mathbb{R}^{\mathbb{N}^*}$ l'ensemble des fonctions arithmétiques. Nous allons définir sur $\mathcal{A}$ une loi de composition interne particulièrement intéresssante.

Exercice : Il y a une faute d'orthographe dans la phrase ci-dessus. Trouvez-la.

1.3.1 3.1 Le produit de Dirichlet

Pour $f, g \in \mathcal{A}$, on note $f \star g$ la fonction de $\mathcal{A}$ définie pour tout $n \geq 1$ par

$$(f \star g)(n) = \sum_{d|n} f(d) g(\frac{n}{d})$$

Ce genre de somme est déjà apparu plus haut : on somme sur tous les diviseurs d de l'entier n.

```
In [33]: def dirichlet_aux(f, g, n):
             ds = diviseurs(n)
             s = 0
             for d in ds:
                 s = s + f(d) * g(n // d)
             return s

         def dirichlet(f, g):
             return lambda n: dirichlet_aux(f, g, n)
```

Calculons quelques produits pas du tout au hasard ...
Commençons par $\phi * \mathbb{1}$.

```
In [34]: print([dirichlet(phi, un)(n) for n in range(1, 100)])
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
```

Proposition : $\varphi \star \mathbb{1} = id$.
Démonstration : Cette propriété exprime que pour tout entier $n \geq 1$,

$$n = \sum_{d|n} \varphi(d)$$

Écrivons

$$n = \sum_{0<k\leq n} 1 = \sum_{d|n} \sum_{0<k\leq n, k\wedge n=d} 1$$

On a $k \wedge n = d$ si et seulement si $\frac{k}{d} \wedge \frac{n}{d} = 1$. Ainsi,

$$\sum_{0<k\leq n, k\wedge n=d} 1 = \sum_{0<q\leq n/d, q\wedge n/d=1} 1$$

où on a posé $q = \frac{k}{d}$. Mais $\sum_{0<q\leq n/d, q\wedge n/d=1} 1 = \varphi(\frac{n}{d})$. Donc

$$n = \sum_{d|n} \varphi(\frac{n}{d})$$

Il ne rete plus qu'à poser $d' = \frac{n}{d}$ dans la somme ci-dessus et à remarquer que lorsque d décrit l'ensemble des diviseurs de n, d' fait de même.

Autre tentative ?

```
In [35]: print([dirichlet(mu, id)(n) for n in range(1, 100)])
```

```
[1, 1, 2, 2, 4, 2, 6, 4, 6, 4, 10, 4, 12, 6, 8, 8, 16, 6, 18, 8, 12, 10, 22, 8, 20, 12, 18, 12,
```

Ces nombres ne nous rappellent-ils rien ?

Proposition : $\mu \star id = \varphi$.

Ceci exprime sous forme ultracompacte, que pour tout $n \geq 1$, $\sum_{d|n} \mu(d)\frac{n}{d} = \varphi(n)$, ou encore

$$\varphi(n) = n\sum_{d|n}\frac{\mu(d)}{d}$$

Il existe donc un lien profond entre μ et φ. D'ici la fin du notebook, cette propriété sera une conséquence immédiate de la précédente.

Encore un essai.

```
In [36]: print([dirichlet(mu, un)(n) for n in range(1, 100)])
```

```
[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

Proposition : $\mu \star \mathbb{1} = \delta$.

Démonstration : En fait nous avons déjà fait la démonstration. Nous avons vu que pour tout entier $n \geq 2$, $\sum_{d|n} \mu(d) = 0$. Dit autrement, $(\mu \star \mathbb{1})(n) = 0$. Et comme, bien évidemment, $(\mu \star \mathbb{1})(1) = \mu(1)\mathbb{1}(1) = 1 = \delta(1)$, on a le résultat.

Une petite dernière ... Pour tout $n \geq 1$, nous avons $(id \star id)(n) = \sum_{d|n} d\frac{n}{d} = n\sum_{d|n} 1 = n\sigma_0(n)$ où $\sigma_0(n)$ est le nombre de diviseurs de n (nous y reviendrons plus loin). Évaluons donc $\frac{id\star id}{id}$.

```
In [37]: print([dirichlet(id, id)(n) // id(n) for n in range(1, 100)])
```

```
[1, 2, 2, 3, 2, 4, 2, 4, 3, 4, 2, 6, 2, 4, 4, 5, 2, 6, 2, 6, 4, 4, 2, 8, 3, 4, 4, 6, 2, 8, 2, 6,
```

Nous aurons bientôt une formule explicite pour calculer $\sigma_0(n)$, nous pourrons alors comparer.

1.3.2 3.2 Commutativité et associativité du produit de Dirichlet

Proposition : Le produit de Dirichlet est commutatif.

Démonstration : Soient f, g deux fonctions arithmétiques. Soit $n \geq 1$. On a

$$(f \star g)(n) = \sum_{a|n} f(a)g(\frac{n}{a})$$

Posons $n = ab$. il vient alors

$$(f \star g)(n) = \sum_{ab=n} f(a)g(b)$$

où la somme s'effectue sur tous les couples (a, b) tels que $ab = n$. Sous cette forme, il est clair que $f \star g = g \star f$.

Proposition : Le produit de Dirichlet est associatif.

Démonstration : Soient f, g, h trois fonctions arithmétiques. Soit $n \geq 1$. On a

$$((f \star g) \star h)(n) = \sum_{kc=n} (f \star g)(k)h(c) = \sum_{kc=n} \sum_{ab=k} f(a)g(b)h(c) = \sum_{abc=n} f(a)g(b)h(c)$$

Sous cette dernière forme, l'associativité est claire.

1.3.3 3.3 Élément neutre

Proposition : Pour toute fonction arithmétique f, on a

$$f \star \delta = \delta \star f = f$$

Démonstration : Exercice. C'est très facile.

Si nous nous résumons, nous obtenons la

Proposition : Soit $\mathcal{A}$ l'ensemble des fonctions arithmétiques. Alors $(\mathcal{A}, \star)$ est un monoïde commutatif.

L'ensemble $\mathcal{A}$ est-il un groupe ? Nous verrons que non, mais un sous-ensemble important de $\mathcal{A}$ en est un. Patience ...

1.3.4 3.4 La formule d'inversion de Möbius

Vous avez peut-être remarqué que des sommes du type "$g(n) = \sum_{d|n} f(d)$" apparaissaient très souvent. Ces sommes sont omniprésentes en théorie analytique des nombres. Très souvent, la fonction g est connue et c'est la fonction f qui nous intéresse. Comment "inverser" la formule ?

Voici la solution à tous nos problèmes, il s'agit de la **formule d'inversion de Möbius**. Et c'est là que la fonction μ prend tout son intérêt.

Proposition : Soient f et g deux fonctions arithmétiques. On suppose que pour tout $n \geq 1$,

$$g(n) = \sum_{d|n} f(d)$$

Alors, pour tout $n \geq 1$,

$$f(n) = \sum_{d|n} \mu(d) g(\frac{n}{d})$$

La réciproque est également vraie.

Démonstration : C'est très très facile (si, si !) : il s'agit de montrer que

$$g = \mathbb{1} \star f \Leftrightarrow f = \mu \star g$$

- ($\Rightarrow$) Supposons $g = \mathbb{1} \star f$. Multiplions par μ :

$$\mu \star g = \mu \star (\mathbb{1} \star f) = (\mu \star \mathbb{1}) \star f = \delta \star f = f$$

- ($\Leftarrow$) Supposons $f = \mu \star g$. Multiplions par $\mathbb{1}$:

$$u \star f = \mathbb{1} \star (\mu \star g) = (\mathbb{1} \star \mu) \star g = \delta \star g = g$$

Exemple : Nous avons admis plus haut que $id = \varphi \star \mathbb{1}$, c'est à dire que pour tout $n \geq 1$, $\sum_{d|n} \varphi(d) = n$. La formule d'inversion de Möbius nous dit ici que $\varphi = \mu \star id$, c'est à dire que pour tout $n \geq 1$,

$$\varphi(n) = \sum_{d|n} \mu(d) \frac{n}{d} = n \sum_{d|n} \frac{\mu(d)}{d}$$

1.3.5 3.4 Fonctions inversibles pour le produit de Dirichlet

L'ensemble $\mathcal{A}$, muni du produit de Dirichlet $\star$, est un monoïde commutatif. Quels sont ses éléments inversibles ? Dit autrement, quelles sont les fonctions $f \in \mathcal{A}$ telles qu'il existe $g \in \mathcal{A}$ pour laquelle $f \star g = \delta$? Avec en récompense, la réponse à cette question : connaissant deux fonctions f et h, peut-on trouver g telle que $f \star g = h$?

Remarque : Si une telle fonction g existe, elle est nécessairement unique par l'associativité et la commutativité de l'opération $\star$.

Proposition : Soit $f \in \mathcal{A}$. Si $f(1) = 0$, f n'est pas inversible pour $\star$.

Démonstration : En effet, pour toute fonction $g \in \mathcal{A}$, $(f \star g)(1) = f(1)g(1) = 0 \neq 1 = \delta(1)$. Donc $f \star g \neq \delta$.

Proposition : Soit $f \in \mathcal{A}$. Si $f(1) \neq 0$, alors f est inversible pour $\star$: il existe $g \in \mathcal{A}$ telle que $f \star g = \delta$. De plus, $g(1) \neq 0$.

Démonstration : Posons pour tout $n \geq 1$, $a_n = f(n)$. On a donc $a_0 \neq 0$. Soit $g \in \mathcal{A}$. Posons pour tout $n \geq 1$, $b_n = g(n)$. On a $f \star g = \delta$ si et seulement si $a_1 b_1 = 1$, c'est à dire $b_1 = \frac{1}{a_1}$, et pour tout $n \geq 2$

$$\sum_{d|n} b_d a_{n/d} = 0$$

c'est à dire $a_1 b_n = -\sum_{d|n, d \neq n} b_d a_{n/d}$.

La fonction g est donc définie de façon unique par récurrence forte par :

- $g(1) = \frac{1}{f(1)}$.
- Pour tout $n \geq 2$, $g(n) = -\frac{1}{f(1)} \sum_{d|n, d \neq n} g(d) f(\frac{n}{d})$.

Corollaire : L'ensemble $\mathbb{G}$ des fonctions arithmétiques qui ne s'annulent pas en 1 est un groupe pour l'opération $\star$.

Corollaire : L'ensemble $\mathbb{H}$ des fonctions arithmétiques **à valeurs entières** qui prennent la valeur 1 en 1 est un sous-groupe de $\mathbb{G}$.

Remarque : Les fonctions $\delta, \mathbb{1}, id, N_\alpha, \varphi, \mu$ appartiennent toutes à $\mathbb{H}$.

La fonction `inverse` ci-dessous prend en paramètres une fonction arithmétique f telle que $f(1) \neq 0$ et un entier n. Elle renvoie la liste $[f^{-1}(1), \ldots, f^{-1}(n)]$.

```
In [38]: def inverse(f, n):
             s = (n + 1) * [0]
             s[1] = 1 / f(1)
             for k in range(2, n + 1):
                 ds = diviseurs(k)
```

```
            b = 0
            for d in ds:
                if d != k:
                    b = b - s[d] * f(k // d) / f(1)
            s[k] = b
        return s[1:]
```

```
In [39]: print(inverse(mu, 20))
```

```
[1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
```

La fonction `inverse1` ci-dessous est un cas particulier important de la précédente. Elle prend en paramètres une fonction arithmétique f à valeurs entières telle que $f(1) = 1$ et un entier n. Elle renvoie la liste $[f^{-1}(1), \ldots, f^{-1}(n)]$.

Pourquoi $f(1) = 1$? Parce que dans ce cas, si la fonction f est à valeurs entières, son inverse de Dirichlet aussi. Et nos fonctions préférées, $\delta, id, \mathbb{1}, \mu, \varphi$ vérifient toutes cette propriété.

```
In [40]: def inverse1(f, n):
            s = (n + 1) * [0]
            s[1] = 1
            for k in range(2, n + 1):
                ds = diviseurs(k)
                b = 0
                for d in ds:
                    if d != k:
                        b = b - s[d] * f(k // d)
                s[k] = b
            return s[1:]
```

```
In [41]: print(inverse1(mu, 20))
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

Proposition : $\mu^{-1} = \mathbb{1}$.
Démonstration : Résultat connu depuis un certain temps : $\mu \star \mathbb{1} = \delta$.

1.3.6 3.5 L'inverse de φ

Quel est l'inverse de l'indicatrice d'Euler ??? Tentons une petite expérience.

```
In [42]: s = inverse1(phi, 30)
         for k in range(30):
             print(k + 1, s[k])
```

```
1 1
2 -1
3 -2
```

```
4 -1
5 -4
6 2
7 -6
8 -1
9 -2
10 4
11 -10
12 2
13 -12
14 6
15 8
16 -1
17 -16
18 2
19 -18
20 4
21 12
22 10
23 -22
24 2
25 -4
26 12
27 -2
28 6
29 -28
30 -8
```

Avant de donner une formule pour $\varphi^{-1}(n)$, un résultat théorique :
Proposition : Soient $f, g \in \mathcal{A}$.

- Si f et g sont multiplicatives alors $f \star g$ est multiplicative.
- Si f et $f \star g$ sont multiplicatives alors g est multiplicative.

Démonstration : Ce n'est pas très difficile mais j'admettrai ce résultat par manque de place.
Corollaire : φ est multiplicative.
Démonstration : En effet, $\varphi \star \mathbb{1} = id$, et $\mathbb{1}$ et id sont multiplicatives.
Corollaire : Soit $f \in \mathbb{G}$. Si f est multiplicative, alors f^{-1} est multiplicative.
Démonstration : f et $f \star f^{-1} = I$ sont multiplicatives, donc f^{-1} l'est aussi.
Corollaire : φ^{-1} est multiplicative.
Démonstration : Parce que φ l'est.
Notons $\psi = \varphi^{-1}$. La fonction ψ est donc multiplicative par le résultat précédent. Il nous suffit donc de calculer $\psi(p^\alpha)$ pour p premier et $\alpha \geq 1$.

- Commençons par $\psi(p)$: on a $(\varphi \star \psi)(p) = I(p) = 0$. Ou encore $\varphi(1)\psi(p) + \varphi(p)\psi(1) = 0$, c'est à dire $\psi(p) + (p-1) = 0$. Ainsi, $\psi(p) = -(p-1)$.
- Que vaut $\psi(p^2)$? Même technique. $\psi(p^2)\varphi(1) + \psi(p)\varphi(p) + \psi(1)\varphi(p^2) = 0$, ou encore $\psi(p^2) + (p-1)^2 + p^2 - p = 0$. D'où $\psi(p^2) = -(p-1)$.

- Que vaut $\psi(p^3)$? $\psi(p^3)\varphi(1) + \psi(p^2)\varphi(p) + \psi(p)\varphi(p^2) + \psi(1)\varphi(p^3) = 0$, ou encore $\psi(p^3) - (p-1)^2 - (p-1)(p^2-p) + (p^3-p^2) = 0$. Arrangeons tout cela :

$$\psi(p^3) - (p-1)^2 - p(p-1)^2 + p^2(p-1) = 0$$

$$\psi(p^3) = (p-1)(p-1+p(p-1)-p^2) = 1-p$$

Exercice : Montrer par récurrence forte sur α que pour tout $\alpha \geq 1$ et tout $p \in \mathcal{P}$, $\psi(p^\alpha) = 1-p$.
On en déduit la
Proposition : Pour tout $n = \prod_{i=1}^{k} p_i^{\alpha_i}$ où les p_i sont premiers distincts et les $\alpha_i \geq 1$, on a

$$\psi(n) = \prod_{i=1}^{k}(1-p_i)$$

```
In [43]: def psi(n):
             fs = facteurs(n)
             x = 1
             for (p, a) in fs:
                 x = (1 - p) * x
             return x
```

Faisons une petite vérification ...

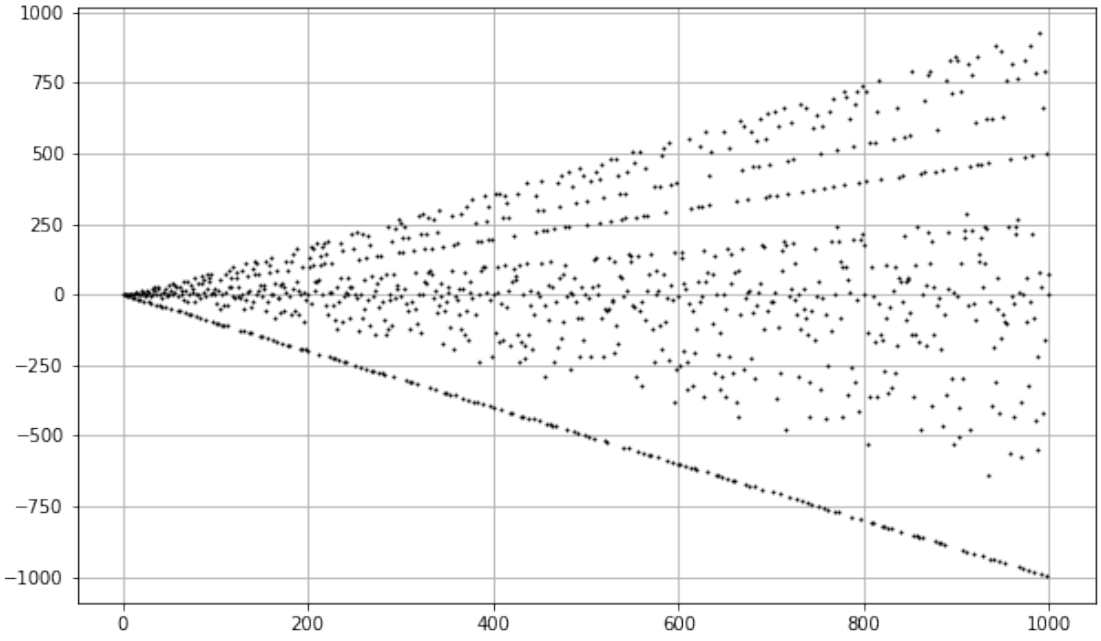
```
In [44]: for k in range(1, 31):
             print('%3d%5d%5d' % (k, psi(k), s[k - 1]))
```

```
  1    1    1
  2   -1   -1
  3   -2   -2
  4   -1   -1
  5   -4   -4
  6    2    2
  7   -6   -6
  8   -1   -1
  9   -2   -2
 10    4    4
 11  -10  -10
 12    2    2
 13  -12  -12
 14    6    6
 15    8    8
 16   -1   -1
 17  -16  -16
 18    2    2
 19  -18  -18
 20    4    4
 21   12   12
 22   10   10
```

```
23  -22  -22
24    2    2
25   -4   -4
26   12   12
27   -2   -2
28    6    6
29  -28  -28
30   -8   -8
```

C'est fou, la théorie donne un résultat exact en pratique :-).
Allez, à quoi ressemble le graphe de φ^{-1} ?.

```
In [45]: plot_fun(psi, 1, 1000, '.k', markersize=2)
```



La formule que nous avons obtenue pour ψ est assez cool, mais peut-on creuser ? Creusons.
Proposition = $\varphi^{-1} = id \times (\mu \star N_{-1})$.
Remarquons que $(id \times (\mu \star N_{-1}))(n) = n\sum_{d|n} \mu(d) N_{-1}(\frac{n}{d}) = n\sum_{d|n} \mu(d)\frac{d}{n} = \sum_{d|n} d\mu(d)$. Ce que dit notre proposition, c'est que pour tout $n \geq 1$,

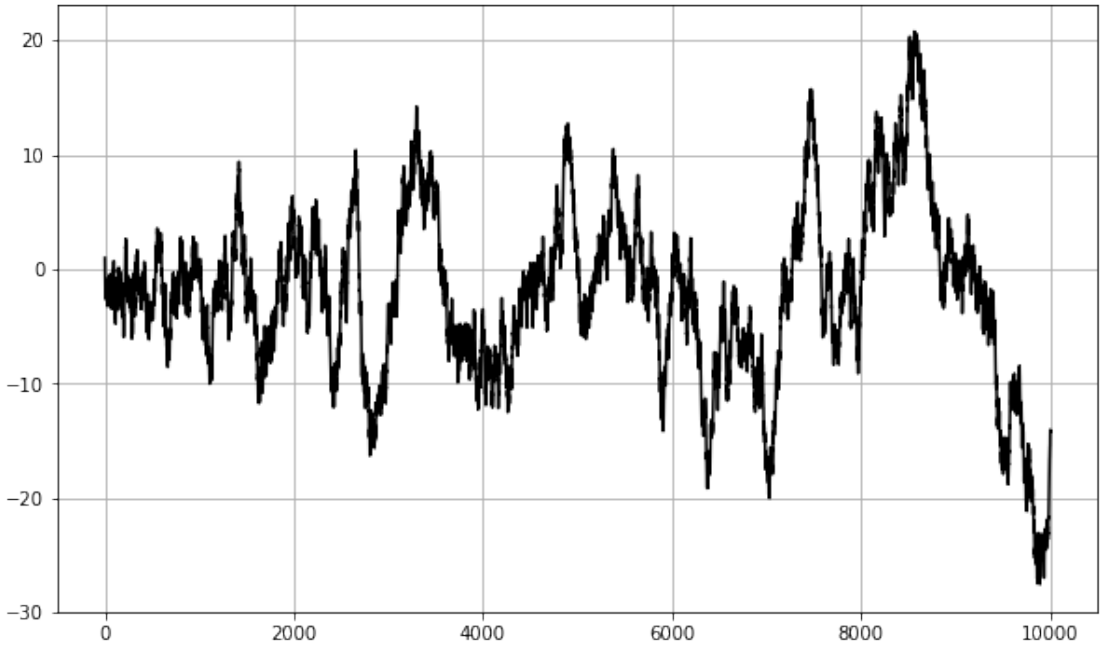
$$\varphi^{-1}(n) = \sum_{d|n} d\mu(d)$$

Démonstration : La fonction ψ et la fonction $n \mapsto n(\mu \star N_{-1})(n)$ sont multiplicatives. Il suffit donc de vérifier qu'elle coincident pour tout entier $n = p^\alpha$ où p est premier et $\alpha \geq 1$. Prenons donc un tel entier n. Les seuls diviseurs de $n = p^\alpha$ en lesquels μ est non nulle sont 1 et p. Ainsi,

$$\sum_{d|n} d\mu(d) = 1 - p = \psi(n)$$

D'où l'égalité cherchée par multiplicativité.
Et la moyenne de φ^{-1}, elle ressemble à quoi ?

```
In [46]: plot_moyennes(psi, 10000)
```



Gloups ... Ça monte, ça descend. Mais ça n'a pas l'air de monter très haut ni de descendre très bas. Sur le graphique ci-dessus, n va de 1 à 10000, alors que $\varphi^{-1}(n)$ oscille entre -30 et 20 (euh un peu plus). Il y a sûrement encore beaucoup à dire.

1.3.7 3.6 L'inverse d'une fonction complètement multiplicative

Exercice : Soit f une fonction complètement multiplicative. Montrer que si $f(1) \neq 1$ alors f est identiquement nulle.

Soit f une fonction **complètement multiplicative** et pas identiquement nulle (comme par exemple u, I, N_α). Quel est son inverse de Dirichlet ? Calculons $f \star (\mu f)$. Soit $n \geq 1$. On a

$$(f \star (\mu f))(n) = \sum_{d|n} f(d)\mu(d)f(\frac{n}{d}) = \sum_{d|n} f(d\frac{n}{d})\mu(d) = f(n)\sum_{d|n}\mu(d) = f(n)I(n)$$

Ainsi, $f \star (\mu f) = fI$. Comme f n'est pas identiquement nulle, $f(1) = 1$) et donc $fI = I$.
Proposition : Pour toute fonction complètement multiplicative f, on a $f^{-1} = \mu f$.
Petit exemple avec la fonction N_2 ?

```
In [47]: n = 20
         s = inverse1(lambda n:n ** 2, n)
         print((n * '%5d') % tuple(range(1, n + 1)))
         print((n * '%5d') % tuple(s))
```

```
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
  1  -4  -9   0 -25  36 -49   0   0 100 -121   0 -169 196 225   0 -289   0 -361
```

Si vous connaissez vos carrés vous apprécierez.

1.4 4 Encore des fonctions arithmétiques

1.4.1 4.1 Nombre de diviseurs, somme des diviseurs, etc.

Pour $\alpha \in \mathbb{R}$ et $n \geq 1$, on pose

$$\sigma_\alpha(n) = \sum_{d|n} d^\alpha$$

Deux cas particuliers importants sont :

- $\alpha = 0$: $d_0(n) = d(n)$ est le nombre de diviseurs de n.
- $\alpha = 1$: $d_1(n)$ est la somme des diviseurs de n.

Nous admettrons ici la
Proposition : La fonction σ_α est multiplicative.
Démonstration : Vous l'aviez sûrement remarqué :

$$\sigma_\alpha = \mathbb{1} \star N_\alpha$$

Or, $\mathbb{1}$ et N_α sont multiplicatives, cqfd.
Conséquence : Pour calculer $\sigma_\alpha(n)$, il suffit donc de savoir calculer $\sigma_\alpha(p^a)$ lorsque p est un nombre premier et $a \in \mathbb{N}^*$. Mais ceci est facile. En effet, les diviseurs de p^a sont : $1, p, p^2, \ldots, p^a$. Ainsi,

- Si $\alpha = 0$, $d_0(p^a) = a + 1$.
- Si $\alpha \neq 0$, $d_\alpha(p^a) = \sum_{i=0}^{a} (p^i)^\alpha = \sum_{i=0}^{a} (p^\alpha)^i = \frac{p^{\alpha(a+1)}-1}{p^\alpha - 1}$.

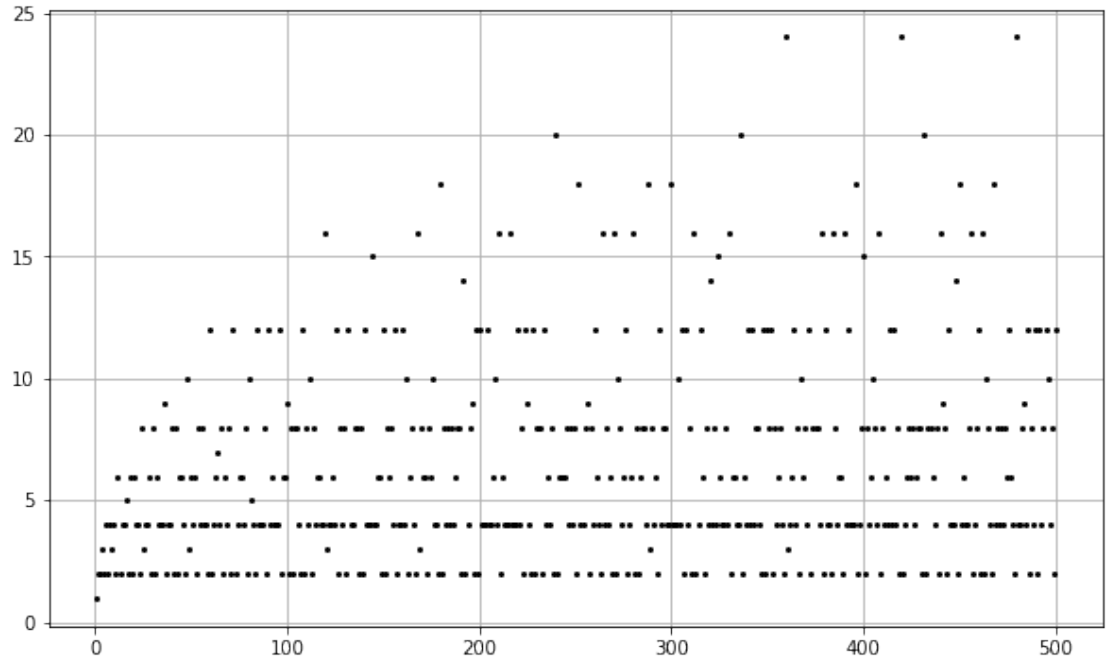
Voici donc la fonction `sigma`. Cette fonction distingue le cas où α est un entier naturel non nul, afin de renvoyer dans ce cas une valeur entière.

```
In [48]: def sigma(n, alpha):
             fs = facteurs(n)
             m = 1
             for (p, a) in fs:
                 if alpha == 0:
                     m = m * (a + 1)
                 elif type(alpha) == type(int) and alpha > 0:
                     m = m * (p ** (alpha * ( a + 1)) - 1) // (p ** alpha - 1)
                 else:
                     m = m * (p ** (alpha * ( a + 1)) - 1) / (p ** alpha - 1)
             return m

In [49]: for n in range(1, 31):
             print('%3d%3d%3d %5f' % (n, sigma(n, 0), sigma(n, 1), sigma(n, -1)))
```

```
 1  1  1 1.000000
 2  2  3 1.500000
 3  2  4 1.333333
 4  3  7 1.750000
 5  2  6 1.200000
 6  4 12 2.000000
 7  2  8 1.142857
 8  4 15 1.875000
 9  3 13 1.444444
10  4 18 1.800000
11  2 12 1.090909
12  6 28 2.333333
13  2 14 1.076923
14  4 24 1.714286
15  4 24 1.600000
16  5 31 1.937500
17  2 18 1.058824
18  6 39 2.166667
19  2 20 1.052632
20  6 42 2.100000
21  4 32 1.523810
22  4 36 1.636364
23  2 24 1.043478
24  8 60 2.500000
25  3 31 1.240000
26  4 42 1.615385
27  4 40 1.481481
28  6 56 2.000000
29  2 30 1.034483
30  8 72 2.400000
```

```
In [50]: plot_fun(lambda n:sigma(n, 0), 1, 500, 'ok', markersize=2)
```

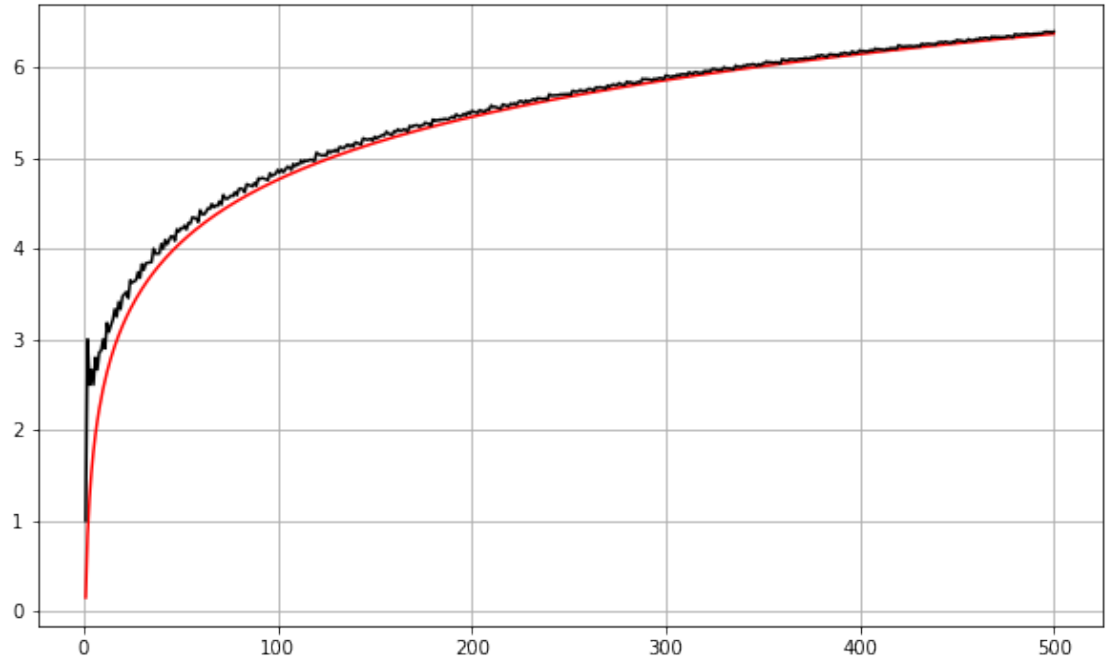


Et si on refaisait le coup des moyennes ?

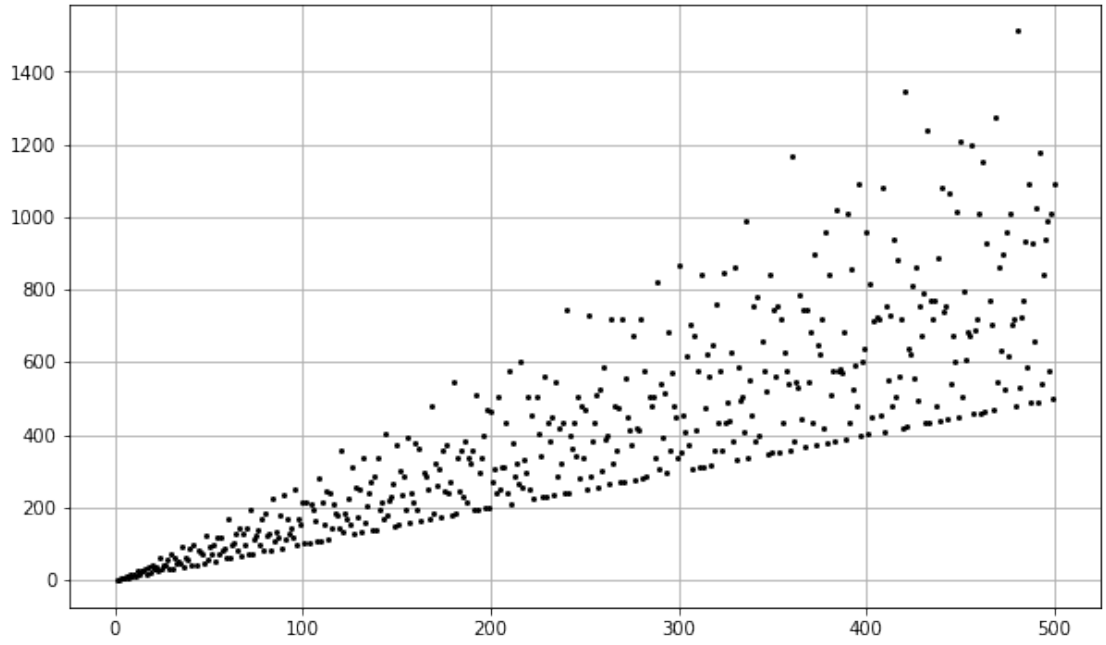
Proposition : $\overline{\sigma}_0(n) = \ln n + (2C - 1) + O(\frac{1}{\sqrt{n}})$ où $C \simeq 0.577216$ est la constante d'Euler.

Démonstration : pas la place (mauvaise foi flagrante). La preuve n'est pas très difficile, cela dit.

```
In [51]: C = 0.577215664
         plot_moyennes(lambda n: sigma(n, 0), 500, lambda n: math.log(n) + (2 * C - 1))
```

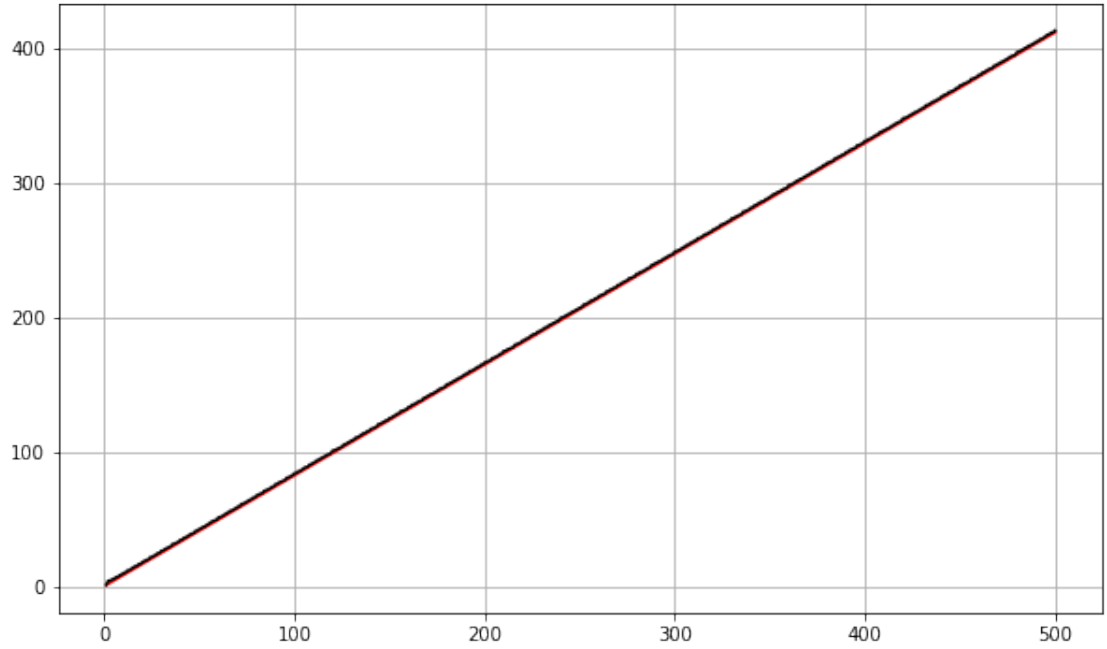


```
In [52]: plot_fun(lambda n:sigma(n, 1), 1, 500, 'ok', markersize=2)
```

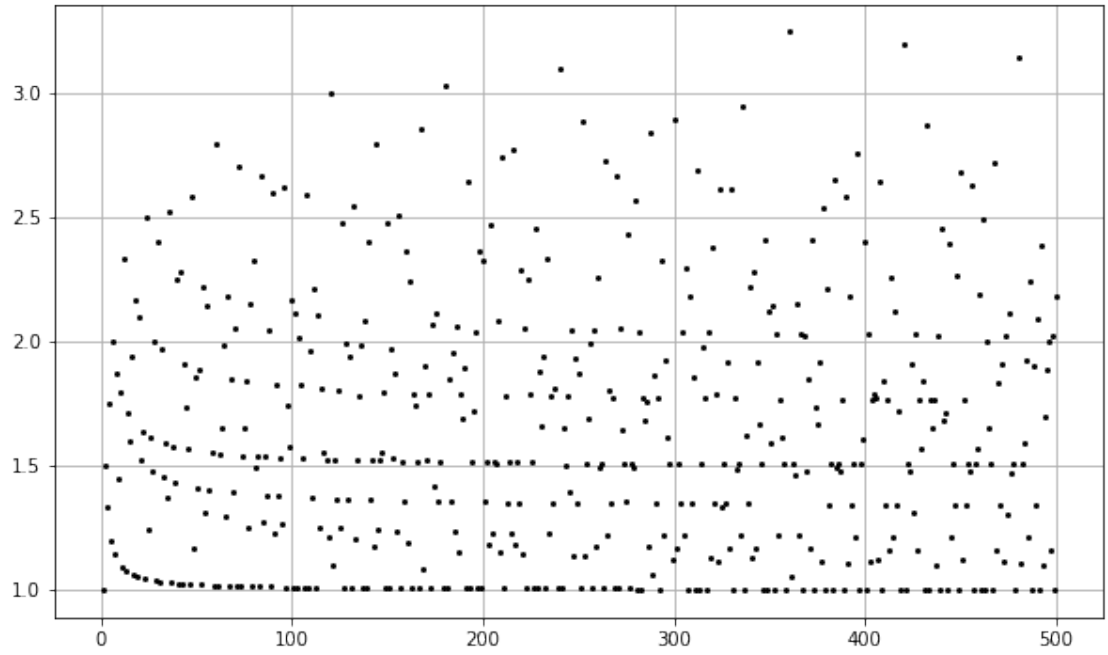


Proposition : $\overline{\sigma}_1(n) \sim \frac{\pi^2 n}{12}$.

```
In [53]: plot_moyennes(lambda n: sigma(n, 1), 500, lambda n: math.pi ** 2 / 12 * n)
```

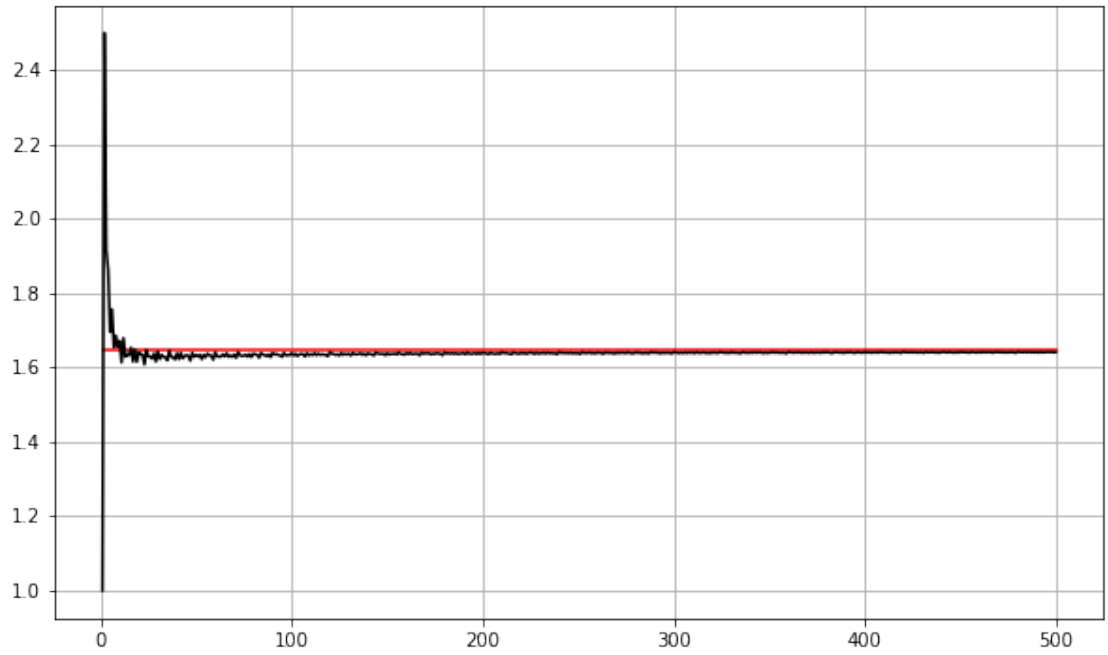


```
In [54]: plot_fun(lambda n:sigma(n, -1), 1, 500, 'ok', markersize=2)
```



Proposition : $\overline{\sigma}_{-1}(n) \to \frac{\pi^2}{6}$ lorsque n tend vers $+\infty$.

```
In [55]: plot_moyennes(lambda n: sigma(n, -1), 500, lambda n: math.pi ** 2 / 6)
```



1.4.2 4.2 La fonction Λ de von Mangoldt

Pour ne pas donner l'impression que toutes les fonctions arithmétiques importantes sont multiplicatives, en voici une qui ne l'est pas. La fonction Λ de von Mangoldt joue un rôle plus que très extrêmement important en théorie analytique des nombres.

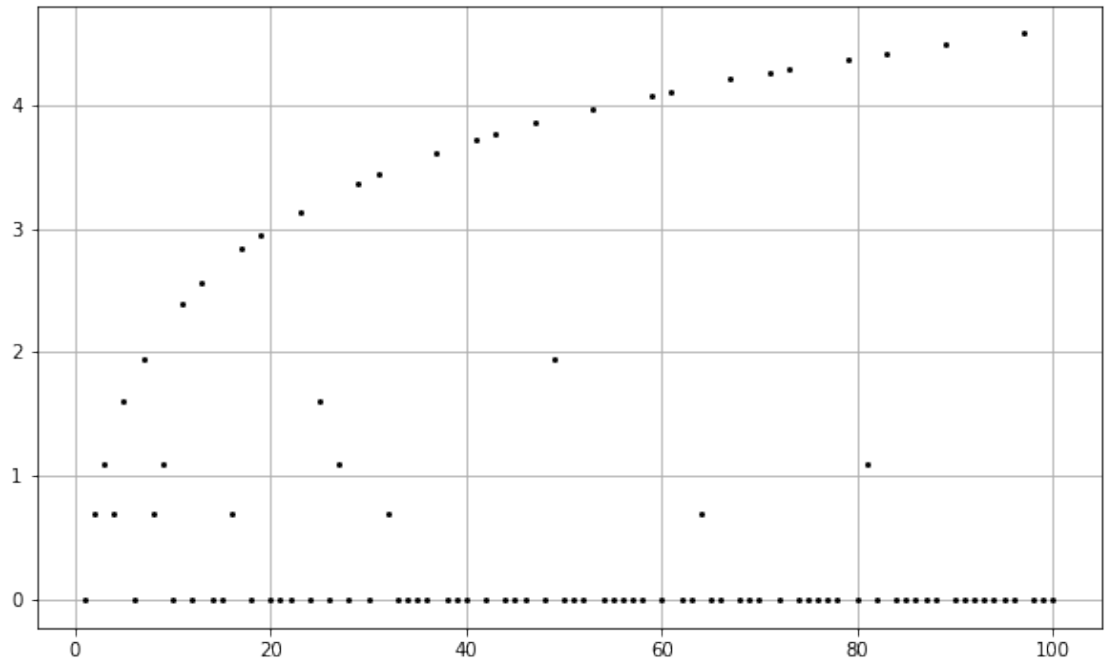
Définition : Soit $n \geq 1$. - Si $n = p^\alpha$ est une puissance d'un nombre premier, $\Lambda(n) = \ln n$. - Sinon, $\Lambda(n) = 0$.

```
In [56]: def mangoldt(n):
             fs = facteurs(n)
             if len(fs) != 1: return 0
             else:
                 p = fs[0][0]
                 return math.log(p)
```

```
In [57]: print([mangoldt(n) for n in range(1, 20)])
```

```
[0, 0.6931471805599453, 1.0986122886681098, 0.6931471805599453, 1.6094379124341003, 0, 1.9459101
```

```
In [58]: plot_fun(mangoldt,1, 100, 'ok', markersize=2)
```




```
In [59]: print([dirichlet(mangoldt, lambda n:1)(n) - math.log(n) for n in range(1, 100)])
```

```
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -4.440892098500626e-16, 0.0, 0.0, 0.0, 0.0, 0.0, 0
```

Comme quoi Λ ce n'est pas n'importe quoi. En fait, c'est peut-être la fonction la plus essentielle de ce notebook. Elle est à la base de la démonstration du théorème des nombres premiers.

Proposition : En désignant par ln la restriction du logarithme à $\mathbb{N}^*$,

$$\Lambda \star \mathbb{1} = \ln$$

et donc, par inversion :

$$\ln \star \mu = \Lambda$$

Démonstration : La fonction Λ est non nulle uniquement en les puissances des nombres premiers. Ainsi,

$$(\Lambda \star \mathbb{1})(n) = \sum_{p^\alpha | n} \Lambda(p^\alpha) = \sum_{p^\alpha | n} \ln p$$

la somme étant entendue pour tous les nombres premiers p et les entiers $\alpha \geq 1$ tels que p^α divise n.

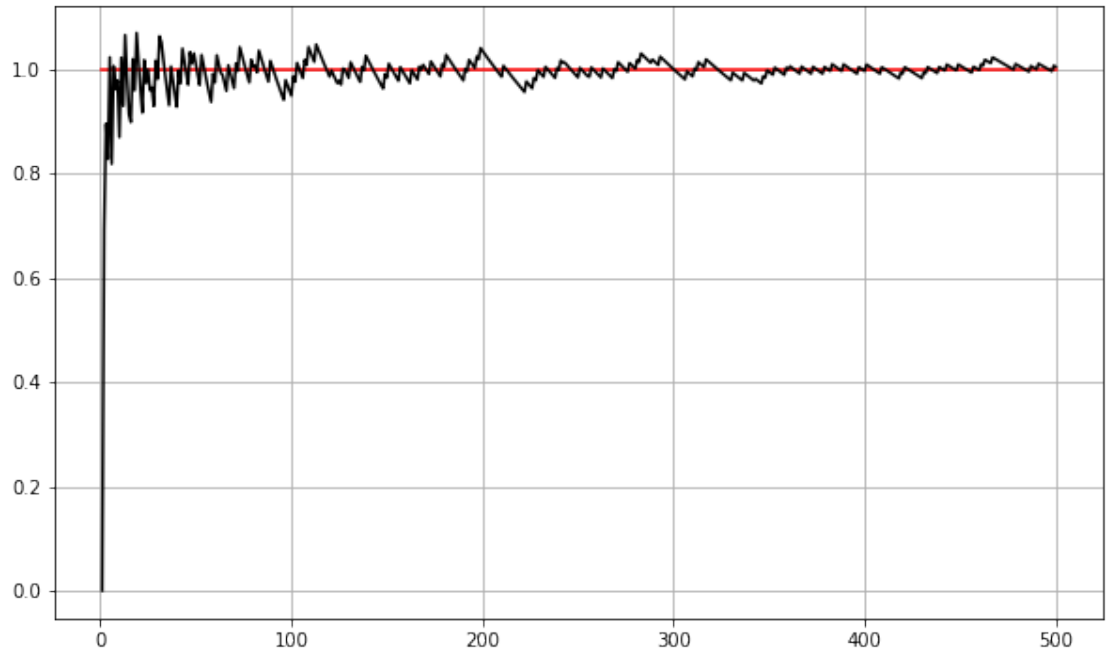
Pour chaque diviseur premier p de n, il y a exactement $\nu_p(n)$ valeurs possibles pour α. Ainsi,

$$(\Lambda \star \mathbb{1})(n) = \sum_{p|n} \nu_p(n) \ln p = \ln \prod_{p|n} p^{\nu_p(n)} = \ln n$$

1.4.3 4.3 Les moyennes de Λ

Eh oui, on ne peut pas ne pas se poser la question !

```
In [60]: plot_moyennes(mangoldt, 500, lambda x:1)
```



Limite égale à 1 ? Facile à montrer ? Non, c'est exactement comme pour la fonction μ.

Proposition : $\overline{\Lambda}(n) \to 1$ lorsque n tend vers l'infini. Ou encore :

$$\frac{1}{n}\sum_{k=1}^{n}\Lambda(k) \to_{n\to\infty} 1$$

Proposition : La propriété précédente est équivalente au théorème des nombres premiers.

Inutile, donc, de nous acharner à essayer de la prouver, à moins que nous ne nous appellions Hadamard ou de la Vallée Poussin :-).

1.5 5 Dérivation

Allons, un dernier effort. Pour finir je voudrais éclairer les égalités précédentes concernant Λ sous un jour nouveau. Plus de Python dans ce qui va suivre ...

1.5.1 5.1 C'est quoi la dérivée ?

Tout le monde sait ce que c'est, n'est-ce pas ? Eh bien non, parce que nous allons définir la dérivée d'une fonction arithmétique de façon très atypique.

Définition : Soit $f \in \mathcal{A}$. La dérivée de f est la fonction $f' \in \mathcal{A}$ définie pour tout $n \geq 1$ par

$$f'(n) = f(n)\ln n$$

Dit autrement,

$$f' = f \times \ln$$

Exemples

- $\delta' = 0$.
- La dérivée de $\mathbb{1}$ est $\mathbb{1}' = \ln$. Du coup, les formules que nous avons vues à la fin du paragraphe précédent deviennent

$$\Lambda \star \mathbb{1} = \mathbb{1}'$$

$$\mathbb{1}' \star \mu = \Lambda$$

1.5.2 5.2 Propriétés

On peut évidemment définir tout et n'importe quoi. Qu'est-ce qui fait la différence entre une bonne et une mauvaise définition ? Ce sont les propriétés que nous pouvons en déduire. Et ici, notre définition de la dérivée est une **très** bonne définition. Notre dérivée se comporte comme la dérivée "normale", sauf que l'on remplace le produit classique $\times$ par l'opération $\star$.

Propriété : Soient $f, g \in \mathcal{A}$. On a :

- $(f+g)' = f' + g'$.

- $(f \star g)' = f' \star g + f \star g'$.
- Si $f(1) \neq 0$, $(f^{-1})' = -f' \star (f \star f)^{-1}$.

Démonstration :

- La somme est laissée en exercice.
- Pour le produit, la clé est $\ln d + \ln \frac{n}{d} = \ln n$. En effet,

$$(f \star g)'(n) = \sum_{d|n} f(d) g(\frac{n}{d}) \ln n = \sum_{d|n} f(d) g(\frac{n}{d}) \ln d + \sum_{d|n} f(d) g(\frac{n}{d}) \ln \frac{n}{d}$$

d'où le résultat.

- On a $I' = 0 = (f \star f^{-1})' = f' \star f^{-1} + f \star (f^{-1})'$. Ainsi,

$$f \star (f^{-1})' = -f' \star f^{-1}$$

Multiplions par f^{-1}. On obtient $(f^{-1})' = -f^{-1} \star (f' \star f^{-1}) = -f' \star f^{-1} \star f^{-1} = -f' \star (f \star f)^{-1}$.

1.5.3 5.3 L'identité de Selberg

Nous terminerons ce notebook par une identité qui est à la base d'une démonstration "élémentaire" du théorème des nombres premiers : l'identité de Selberg.

Proposition : Pour tout $n \geq 1$, on a

$$\Lambda(n) \ln n + \sum_{d|n} \Lambda(d) \Lambda(\frac{n}{d}) = \sum_{d|n} \mu(d) \ln^2 \frac{n}{d}$$

Démonstration magique : On part de $\Lambda \star \mathbb{1} = \mathbb{1}'$ et on dérive :

$$\Lambda' \star \mathbb{1} + \Lambda \star \mathbb{1}' = \mathbb{1}''$$

Mais $\mathbb{1}' = \Lambda \star \mathbb{1}$. Donc

$$\Lambda' \star \mathbb{1} + \Lambda \star \Lambda \star \mathbb{1} = \mathbb{1}''$$

Multiplions les deux côtés par $\mu = \mathbb{1}^{-1}$. Il vient

$$\Lambda' + \Lambda \star \Lambda = \mathbb{1}'' \star \mu$$

Exercice : Vérifiez qu'on a fini :-).

Exercice : Vous pensez avoir tout compris. En admettant $\varphi \star \mathbb{1} = id$, prouvez que φ est multiplicative. C'est immédiat si l'on sait où chercher.

1.6 6 Bibliographie

La partie I (tests naïfs de primalité, factorisation) fait partie de la mémoire de l'humanité.

Pour tout le reste, j'ai puisé sans limites dans les chapitres 2 et 3 du fantastique livre de Tom M. Apostol *Introduction to Analytic Number Theory*, Springer, 1976.