1
Arithmétique

I
Les classiques à connaître

Algorithme d'Euclide

1. **Utilisations**
1.1 **RMS 133**
982, 986

2. L'**algorithme d'Euclide** permet de calculer le pgcd de deux entiers.
2.1 Étant donnés deux entiers naturels a et b, le pgcd de a et b est égal à a si b est nul et au pgcd de b et de r, le reste de la division euclidienne de a par b, sinon.
2.2 On peut coder de façon récursive ou itérative, comme on préfère.
2.3

```
def pgcd(a,b):
    if b==0:
        return a
    else:
        return pgcd(b, a%b)
```

2.4

```
def pgcd_iter(a,b):
    while b>0:
        a, b = b, a%b
    return a
```

Crible d'Eratosthène

3. **Utilisations**
3.1 **RMS 133**
983

4. Le **crible d'Eratosthène** est un moyen simple de calculer les nombres premiers jusqu'à un certain rang.

5. La première version du crible consiste à renvoyer une liste $L = [a_0, \ldots, a_n]$ où $a_k = 1$ si k est premier et $a_k = 0$ si k n'est pas premier.
5.1 Une fonction auxiliaire détecte tous les multiples de l'entier p dans la liste L.

```
def purger_multiples(L, p):
    N, m = len(L), 2*p
    while m<N:
        L[m] = 0
        m += p
```

5.2

```
def premiers(n):
    L = [1]*(n+1)
    L[0], L[1] = 0, 0
    p, N = 2, len(L)
    while p<N:
        if L[p]==1:
            purger_multiples(L, p)
        p+=1
    return L
```

6. La deuxième version du crible consiste à renvoyer la liste des n premiers nombres premiers.

6.1 Une fonction auxiliaire permet de savoir si un entier m est divisible par l'un des éléments de la liste L.

```
def nouveau_premier(m, L):
    non_divisible = True
    for p in L:
        non_divisible = non_divisible and (m%p!=0)
    return non_divisible
```

6.2 On peut alors écrire une fonction qui renvoie la liste des n premiers entiers premiers pour tout $n \geqslant 2$.

```
def liste_premiers(n):
    m, L = 5, [2,3]
    while len(L)<n:
        if nouveau_premier(m, L):
            L.append(m)
        else:
            m+=2
    return L
```

Diviseurs d'un nombre entier

7. Utilisations

7.1 RMS135

45

8. Pour trouver les diviseurs d'un entier $n \in \mathbb{N}^*$, on passe en revue les entiers $d \in [\![1, n]\!]$ et on cherche si d divise n.

```
def diviseurs(n):
    L = []
    for i in range(1, n+1):
        if n%i==0:
            L.append(i)
    return L
```

9. On peut regretter que, d'une certaine manière, l'algorithme précédent fasse deux fois la même chose. Nous allons le rendre plus efficace, mais aussi plus compliqué — on n'a rien sans rien.

9.1 Si d divise n, alors il existe un entier q tel que $n = dq$ et :

— ou bien $d < q$ et dans ce cas, on a trouvé *deux* diviseurs de n : d et q ;

— ou bien $d = q$ et dans ce cas, on a trouvé *un seul* diviseur de n : d (l'entier n est un carré parfait) ;

— ou bien $d > q$ et dans ce cas, on a déjà trouvé le couple (q, d).

9.2 Pour que la liste des diviseurs soit triée par ordre croissant, il suffit de constituer deux listes : d'une part, la liste des diviseurs d (naturellement ordonnée en croissant) et d'autre part, la liste des diviseurs q (naturellement ordonnée en décroissant).

9.3

```
def diviseurs_2(n):
    d, q = 1, n
    L_inf, L_sup = [d], [q]
    while d<q:
        d += 1
        q = n//d
        if n%d==0:
            if d<q:
                L_inf.append(d)
                L_sup.append(q)
            elif d==q:
                L_sup.append(q)
                return L_inf+L_sup[::-1]
    return L_inf+L_sup[::-1]
```

Coefficients binomiaux

10. **Utilisations**

10.1 **RMS 133**

1032

11. Par convention, si $k < 0$ ou si $k > n$, alors le coefficient binomial $\binom{n}{k}$ est nul.

11.1 Propriété de symétrie :

$$\forall\, 0 \leqslant k \leqslant n, \quad \binom{n}{k} = \binom{n}{n-k}.$$

11.2 Pour tout entier $1 \leqslant k \leqslant {}^{n}/_{2}$,

$$\binom{n}{k} = \frac{n}{k} \cdot \binom{n-1}{k-1} = \frac{n-0}{k} \cdot \frac{n-1}{k-1} \cdots \frac{n-(k-1)}{1} \cdot \binom{n-k}{0} = \prod_{\ell=0}^{k-1} \frac{n-\ell}{\ell+1}.$$

12. La fonction suivante calcule un seul coefficient binomial à la fois.

```
def binom(n, k):
    if (k<0) or (k>n):
        return 0
    elif (2*k>n):
        return binom(n, n-k)
    else:
        cb, num = 1, n-k
        for ell in range(k):
            num += 1
            cb = (cb*num)//(ell+1)
        return cb
```

13. L'écriture récursive n'est pas plus efficace (on fait exactement les mêmes calculs) mais plus lisible.

```
def binom_rec(n, k):
    if (k<0) or (k>n):
        return 0
    elif (2*k>n):
        return binom_rec(n, n-k)
    elif k==0:
        return 1
    else:
        return (binom_rec(n-1, k-1)*n)//k
```

14.1 On a parfois besoin de connaître tous les coefficients binomiaux d'une ligne du triangle de Pascal, c'est-à-dire les coefficients $\binom{n}{k}$ pour $k \in [\![0, n]\!]$.

```
def liste_coeffs_binom(n):
    L = [1]
    for k in range(n):
        cb = (L[-1]*(n-k))//(k+1)
        L.append(cb)
    return L
```

14.2 Le module `numpy` est très commode pour exploiter la formule du triangle de Pascal.

```
def ligne_binome(n):
    L = np.zeros(n+1, dtype=int)
    L[0] = 1
    for k in range(n):
        L[1:] = L[1:]+L[:-1]
    return L
```

Si la ligne `L` contient les coefficients

$$\binom{k}{0}, \binom{k}{1}, \dots, \binom{k}{k}, \dots, \binom{k}{n}$$

alors les tranches `L[1:]` et `L[:-1]` contiennent respectivement

$$\binom{k}{1}, \binom{k}{2}, \dots, \binom{k}{k}, \binom{k}{k+1}, \dots, \binom{k}{n} \quad \text{et} \quad \binom{k}{0}, \binom{k}{1}, \dots, \binom{k}{k-1}, \binom{k}{k}, \dots, \binom{k}{n-1}$$

et, d'après la formule du triangle de Pascal, la somme de ces deux tranches contient les coefficients

$$\binom{k+1}{1}, \binom{k+1}{2}, \dots, \binom{k+1}{k}, \binom{k+1}{k+1}, \dots, \binom{k+1}{n}.$$

14.3 Une légère modification suffit pour calculer les premières lignes du triangle de Pascal.

```
def triangle_Pascal(n):
    L, T = np.zeros(n+1), np.zeros((n+1, n+1), dtype=int)
    L[0] = 1
    T[0] = L
    for k in range(n):
        L[1:] = L[1:]+L[:-1]
        T[k+1] = L
    return T
```

II

Quelques applications

133-983

[1.] *Écrire une fonction calculant le pgcd de deux entiers a et b.*

[2.] *Soit $n \geqslant 2$, un nombre entier. Écrire une fonction qui détermine le nombre de générateurs de $(\mathbb{Z}/n\mathbb{Z})^\times$.*

133-986

[1.] *Pour tout entier $n \in \mathbb{N}^*$, on pose*

$$H_n = \sum_{k=1}^{n} \frac{1}{k} = \frac{p_n}{q_n}$$

où p_n et q_n sont des entiers naturels premiers entre eux.
Écrire une fonction `determiner(n)` *qui renvoie le couple (p_n, q_n).*

[2.] *Pour tout nombre premier n au plus égal à 100, afficher le reste de la division euclidienne de p_{n-1} par n^2. Conjecturer un résultat pour n quelconque.*

133-1272

On cherche à calculer les valeurs de l'indicatrice d'Euler φ.

[1.] *Écrire une fonction* `PremierDiviseur(n)` *qui retourne le plus petit diviseur premier de l'entier $n \geqslant 2$.*

[2.] *Soient $n \geqslant 2$ et p, un diviseur premier de n. Démontrer que : si p^2 divise n, alors*

$$\varphi(n) = p\varphi(n/p)$$

et que, dans le cas contraire,

$$\varphi(n) = (p-1)\varphi(n/p).$$

[3.] *En déduire une fonction récursive* `phi(n)` *qui calcule $\varphi(n)$.*

135-py-36 - Nombre de surjections

Pour tout entier $m \in \mathbb{N}^*$, on pose $S_{0,m} = 0$ et, pour tout entier $n \in \mathbb{N}^*$, on note $S_{n,m}$, le nombre de surjections de $[\![1,n]\!]$ dans $[\![1,m]\!]$. On pose également

$$S_m(z) = \sum_{n=0}^{+\infty} S_{n,m} \frac{z^n}{n!}.$$

[1.] *Calculer $S_{n,m}$ pour $m = 1$, pour $n = m$ et pour $n < m$.*

[2.] *En majorant simplement $S_{n,m}$, démontrer que la somme S_m est définie pour tout $z \in \mathbb{C}$.*

[3.a] *Démontrer que*

$$\forall\, m \in \mathbb{N}^*,\ \forall\, n \in \mathbb{N}, \quad S_{n,m+1} = \sum_{k=1}^{n} \binom{n}{k} S_{n-k,m}.$$

[3.b] *En déduire que*

$$\forall\, m \in \mathbb{N}^*,\ \forall\, z \in \mathbb{C}, \qquad S_m(z) = (e^z - 1)^m.$$

[4.] *Calculer les valeurs de $S_{10,m}$ pour $1 \leqslant m \leqslant 10$.*

[5.] *Démontrer que*

$$\forall\, m \in \mathbb{N}^*,\ \forall\, n \in \mathbb{N}, \quad S_{n,m} = \sum_{k=0}^{m} \binom{m}{k} k^n (-1)^{m-k}.$$

135-py-45

Pour tout entier $n \in \mathbb{N}^*$, on note d_n, le nombre de diviseurs de n (en comptant 1 et n) et on pose

$$D_n = \sum_{k=1}^{n} d_k \qquad \text{et} \qquad H_n = \sum_{k=1}^{n} \frac{1}{k}.$$

[1.] *Écrire des fonctions prenant un entier $n \geqslant 1$ pour argument et renvoyant respectivement D_n et H_n.*

[2.a] *Dénombrer les couples d'entiers naturels* non nuls *(k, ℓ) se situant sous la courbe d'équation $xy = n$.*

[2.b] *En déduire que $D_n = nH_n + \mathcal{O}(n)$, puis un équivalent de D_n.*

[3.] *Pour $x \in]-1,1[$, calculer les sommes suivantes.*

$$\sum_{n=1}^{+\infty} \frac{x^n}{n} \qquad \sum_{n=1}^{+\infty} H_n x^n \qquad \sum_{n=1}^{+\infty} n H_n x^n$$

[4.] *En déduire un équivalent de la somme*

$$\sum d_n x^n$$

lorsque x tend vers 1.

[5.] *Démontrer que la série de fonctions*

$$\sum \frac{x^n}{1-x^n}$$

converge simplement sur $]-1,1[$ et que sa somme est une fonction continue sur cet intervalle.

Solutions

133-983

[1.] Cf l'algorithme d'Euclide [**2.3**] ou [**2.4**].

[2.] Pour tout entier $n \geqslant 2$, l'ensemble $G_n = (\mathbb{Z}/n\mathbb{Z})^\times$ des éléments inversibles de l'anneau $\mathbb{Z}/n\mathbb{Z}$ est un groupe pour la multiplication. L'ordre du groupe G_n est, par définition, l'indicatrice d'Euler $\varphi(n)$.
L'entier a est inversible modulo n si, et seulement si, a et n sont premiers entre eux. On peut donc facilement déterminer la liste des éléments inversibles de $\mathbb{Z}/n\mathbb{Z}$, dont la longueur nous donne l'indicatrice d'Euler.

```
def inversibles(n):
  gpe_inv = []
  for a in range(n):
    if pgcd(a, n)==1:
      gpe_inv.append(a)
  return gpe_inv
```

Un élément a inversible de l'anneau $\mathbb{Z}/n\mathbb{Z}$ engendre le groupe G_n des éléments inversibles si, et seulement si, l'ordre de a est égal à $\varphi(n)$, c'est-à-dire si le plus petit entier $k \geqslant 2$ tel que $a^k \equiv 1 \pmod{n}$ est égal à $\varphi(n)$.

```
def est_generateur(a, n, phi_n):
  x, k = a, 1
  while x!=1:
    k += 1
    x = (a*x)%n
  return (k==phi_n)
```

Il reste alors à parcourir le groupe G_n pour trouver l'ensemble de ses générateurs.

```
def generateurs(n):
  gpe_inv = inversibles(n)
  phi_n = len(gpe_inv)
  liste_gene = []
  for a in gpe_inv:
    if est_generateur(a, n, phi_n):
      liste_gene.append(a)
  return liste_gene
```

133-986

[1.] On se sert de la fonction `pgcd` pour calculer l'expression réduite de la somme de deux rationnels.

```
def somme(u, v):
  num, den = u[0]*v[1]+u[1]*v[0], u[1]*v[1]
  d = pgcd(num, den)
  return (num//d, den//d)
```

On peut ensuite coder la fonction `determiner(n)`.

```
def determiner(n):
  (p,q) = (0,1)
  for k in range(n):
    (p,q) = somme((p,q),(1,k+1))
  return (p,q)
```

[2.] L'algorithme d'Erastosthène [5] nous donne les nombres premiers inférieurs à n. Pour chacun d'eux, on calcule p_n avec la fonction précédente et on affiche le résultat.

```
def conjecture():
  nb_premiers = premiers(100)
  for n in range(len(nb_premiers)):
    if nb_premiers[n]:
      p, q = determiner(n-1)
      reste = p%(n**2)
      print(reste)
```

Voir `https://fr.wikipedia.org/wiki/Nombre_harmonique`, qui renvoie au Théorème de Wolstenholme.

133-1272

[1.] Si l'entier n n'est pas pair, son plus petit diviseur premier est un entier impair. Après avoir examiné la parité de n, on passe en revue tous les entiers impairs, on finira bien par en trouver un qui divise n (au plus tard, ce sera n lui-même).

```
def PremierDiviseur(n):
  if n%2==0:
    return 2
  d = 3
  while n%d!=0:
    d +=2
  return d
```

[2.] On sait que l'indicatrice d'Euler est une **fonction multiplicative** : si a et b sont premiers entre eux, alors $\varphi(ab) = \varphi(a)\varphi(b)$.
Par conséquent, si $n = p^\alpha.q$ où p et q sont premiers entre eux, alors

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

L'entier p^2 divise n si, et seulement si, $\alpha \geqslant 2$. Dans ce cas, $\alpha - 1 \geqslant 1$ et

$$\begin{aligned}\varphi(n) &= p.(p-1)p^{\alpha-2}.\varphi(q) = p.\varphi(p^{\alpha-1}).\varphi(q) = p\varphi(p^{\alpha-1}q)\\ &= p\varphi(n/p).\end{aligned}$$

Dans le cas contraire, $\alpha = 1$ et

$$\varphi(n) = (p-1).\varphi(q) = (p-1).\varphi(n/p).$$

[3.] Si $n = 1$, on sait que $\varphi(n) = 1$. Si $n \geqslant 2$, alors n possède un plus petit diviseur premier p et ce qui précède permet d'exprimer $\varphi(n)$ en fonction de $\varphi(n/p)$.
Comme p divise n et que $q = n/p$, si p divise q, alors p^2 divise le produit pq.

```
def phi(n):
  if n==1:
    return 1
  else:
    p = PremierDiviseur(n)
    q = n//p
    if q%p==0:
      return p*phi(q)
    else:
      return (p-1)*phi(q)
```

135-py-36 - Nombre de surjections

[1.] Si $m = 1$, une seule application de $[\![1, n]\!]$ dans $[\![1, m]\!]$ existe, c'est une application constante et elle est surjective. Donc $S_{n,1} = 1$ pour tout entier $n \geqslant 1$.

❧ Pour $n = m$, l'ensemble de départ et l'ensemble d'arrivée sont des ensembles *finis*, de *même cardinal*. Donc une application est surjective si, et seulement si, elle est bijective et on cherche à dénombrer les permutations d'un ensemble de n éléments. Donc $S_{n,n} = n!$ pour tout $n \geqslant 1$.

❧ Si $n < m$, alors le cardinale de l'ensemble de départ est strictement inférieur au cardinal de l'ensemble d'arrivée. Par conséquent, l'image de toute fonction est une partie stricte de l'ensemble d'arrivée et aucune fonction n'est surjective. Donc $S_{n,m} = 0$ pour tous $n < m$.

[2.] Le nombre d'applications de $[\![1, n]\!]$ dans $[\![1, m]\!]$ est égal à m^n, donc $0 \leqslant S_{n,m} \leqslant m^n$, quels que soient m et n. Quel que soit le réel positif t,

$$\forall\, n \in \mathbb{N}^*, \qquad 0 \leqslant S_{n,m} \cdot \frac{t^n}{n!} \leqslant \frac{(mt)^n}{n!}.$$

Par croissances comparées, le majorant tend vers 0 lorsque n tend vers $+\infty$, donc la suite de terme général ${S_{n,m}}^{t^n}/_{n!}$ est bornée pour tout $t \in \mathbb{R}_+$. Dans ces conditions, le rayon de convergence de la série entière est infini et la somme est définie sur $\mathbb{C}$.

[3.a] L'entier $S_{n,m+1}$ est le cardinal de l'ensemble $\Sigma_{n,m+1}$ des surjections de $[\![1, n]\!]$ dans $[\![1, m+1]\!]$.
Pour toute application $f \in \Sigma_{n,m+1}$, l'élément $(m+1)$ admet au moins un antécédent (puisque f est surjective) et au plus n antécédents (puisqu'il y a n éléments dans l'ensemble de départ).

✍ *On pourrait être plus précis : puisque chacun des m autres éléments de l'ensemble d'arrivée admet au moins un antécédent, l'élément $(m+1)$ admet au plus $(n-m)$ antécédents. C'est sans importance pour la suite du raisonnement.*

On peut définir une partition de l'ensemble $\Sigma_{n,m+1}$ en discutant sur le cardinal de l'image réciproque de $\{m+1\}$: nous noterons $\Sigma^k_{n,m+1}$, l'ensemble des applications surjectives de $[\![1, n]\!]$ dans $[\![1, m+1]\!]$ pour lesquelles $(m+1)$ admet exactement k antécédents. Puisqu'on a défini une partition,

$$\#(\Sigma_{n,m+1}) = \sum_{k=1}^{n} \#(\Sigma^k_{n,m+1}).$$

❧ Il reste à compter le nombre d'éléments de $\Sigma^k_{n,m+1}$. Pour déterminer une application $f \in \Sigma^k_{n,m+1}$, il faut d'abord connaître les k antécédents de l'élément $(m+1)$, ce qui nous fait $\binom{n}{k}$ possibilités.
L'image réciproque de $(m+1)$ étant connue, il faut ensuite déterminer la restriction de f à $[\![1, n]\!] \setminus f^*(\{m+1\})$. Cette restriction est une application surjective d'un ensemble de cardinal $(n-k)$ à valeurs dans $[\![1, m]\!]$, donc il y a (par définition !) $S_{n-k,m}$ restrictions possibles.
Par principe multiplicatif,

$$\#(\Sigma^k_{n,m+1}) = \binom{n}{k} \cdot S_{n-k,m}$$

et finalement

$$S_{n,m+1} = \sum_{k=1}^{n} \binom{n}{k} \cdot S_{n-k,m}. \qquad (\star)$$

✍ *On a remarqué plus haut que $1 \leqslant k \leqslant n-m$. Si $k > n-m$, alors $n-k < m$ et, d'après la première question, $S_{n-k,m}$ est alors nul.*

[3.b] Pour $m = 1$, on sait que $S_{0,m} = 0$ et que $S_{n,1} = 1$ pour tout $n \in \mathbb{N}^*$, donc

$$\forall\, z \in \mathbb{C}, \qquad \sum_{n=0}^{+\infty} S_{n,1} \frac{z^n}{n!} = \sum_{n=1}^{+\infty} \frac{z^n}{n!} = e^z - 1.$$

❧ Supposons que, pour un certain entier $m \in \mathbb{N}^*$, on ait démontré que

$$\forall\, z \in \mathbb{C}, \qquad \sum_{n=0}^{+\infty} S_{n,m} \frac{z^n}{n!} = (e^z - 1)^m.$$

Alors le calcul précédent nous assure que

$$\begin{aligned}(e^z-1)^{m+1} = (e^z-1)^m(e^z-1) &= \Big(\sum_{n=0}^{+\infty} S_{n,m}\frac{z^n}{n!}\Big)\Big(\sum_{n=0}^{+\infty} S_{n,1}\frac{z^n}{n!}\Big)\\ &= \sum_{n=0}^{+\infty}\Big(\sum_{k=0}^{n}\frac{S_{n-k,m}}{(n-k)!}\cdot\frac{S_{k,1}}{k!}\Big)z^n && \text{(produit de Cauchy de deux séries entières)}\\ &= \sum_{n=0}^{+\infty}\Big[\sum_{k=1}^{n} S_{n-k,m}\binom{n}{k}\Big]\frac{z^n}{n!} && \text{(la somme commence avec } k=1 \text{ car } S_{0,1}=0)\\ &= \sum_{n=0}^{+\infty} S_{n,m+1}\frac{z^n}{n!}\end{aligned}$$

d'après la relation de récurrence établie ci-dessus. Le résultat voulu est ainsi établi par récurrence.

✍ *Si les rayons de convergence des séries entières $\sum a_n z^n$ et $\sum b_n z^n$ sont infinis, alors le rayon de convergence du produit de Cauchy $\sum c_n z^n$ défini par*

$$\forall\, n\in\mathbb{N},\qquad c_n = \sum_{k=0}^{n} a_k b_{n-k} = \sum_{k=0} a_{n-k}b_k$$

est infini lui aussi.

[4.] Il est étrange de demander à calculer les valeurs de $S_{10,m}$ après avoir établi une relation de récurrence qui laisse deviner un calcul long, alors qu'une formule sensiblement plus simple est démontrée ensuite.

❧ Quoi qu'il en soit, on retrouve ici le calcul des coefficients binomiaux **[14]**. Le plus économique consiste à calculer dès le début tous les coefficients binomiaux dont on aura besoin.

```
T = triangle_Pascal(10)
```

❧ La relation de récurrence $(\star)$ nous fait calculer les $S_{\ell,m}$ pour $1\leqslant\ell\leqslant 10$ et $1\leqslant m\leqslant 10$, ainsi que les coefficients binomiaux $\binom{n}{k}$ pour $1\leqslant k\leqslant n$ et $1\leqslant n\leqslant 10$. Il reste à récupérer les valeurs $S_{10,m}$ au terme des calculs.

```
S = np.zeros((11, 11), dtype=int)
S[1:,1] = 1
for m in range(1, 10):
    for n in range(1, 11):
        s = 0
        for k in range(1, n+1):
            s += T[n,k]*S[n-k,m]
        S[n,m+1] = s
print(S[10,1:])
```

❧ Le code est plus simple en utilisant l'expression obtenue à la question suivante.

```
S = []
for m in range(1, 11):
    s = 0
    for k in range(m+1):
        s += T[m,k]*k**10*(-1)**(m+k)
    S.append(s)
```

[5.] Comme le rayon de convergence de la série entière est strictement positif (il est infini !), la formule de Taylor nous dit que

$$\forall\, m\in\mathbb{N}^*,\ \forall\, n\in\mathbb{N},\qquad S_{n,m} = S_m^{(n)}(0)$$

et la formule du binôme permet de développer l'expression de la somme :

$$\forall\, m\in\mathbb{N}^*,\ \forall\, z\in\mathbb{C},\qquad S_m(z) = \sum_{k=0}^{m}\binom{m}{k}(e^z)^k(-1)^{m-k} = \sum_{k=0}^{m}\binom{m}{k}e^{kz}(-1)^{m-k}.$$

Par conséquent,

$$\forall\, m \in \mathbb{N}^*,\ \forall\, n \in \mathbb{N}, \qquad S_m^{(n)}(0) = \sum_{k=0}^{m} \binom{m}{k} k^n e^{k\times 0}(-1)^{m-k},$$

ce qui donne la relation attendue.

135-py-45

[1.] Pour écrire la fonction qui calcule D_n, on peut utiliser [8] (qui est simple et raisonnablement efficace, même si elle n'est pas optimale).

```
def D(n):
    s = 0
    for n in range(1, n+1):
```

❧ Le calcul de H_n ne pose aucune difficulté apparente.

```
def H(n):
    s = 0
    for k in range(1, n+1):
        s += 1/k
    return s
```

✍ *En fait, le calcul des nombres harmoniques pose un problème réellement difficile !*
On sait que la suite $(H_n)_{n\geqslant 1}$ tend vers $+\infty$ mais si on travaille en virgule flottante, le code précédent affirme que la suite $(H_n)_{n\geqslant 1}$ est stationnaire !
Ce comportement paradoxal apparaît quelle que soit la précision retenue pour mener les calculs. Si on refait les calculs avec une précision plus élevée, la "limite apparente" de la suite sera plus élevée, mais la suite paraîtra toujours stationnaire.

[2.a] On peut effectuer ce dénombrement de deux manières.
❧ Un couple (x, x') d'entiers non nuls est situé sous la courbe $xy = n$ si, et seulement si, le produit xx' est un entier $k \in [\![1, n]\!]$ et $xx' = k$ si, et seulement si, x est un diviseur de k. Par conséquent, le nombre de couples (x, x') est égal à

$$\sum_{k=1}^{n} d_k = D_n.$$

❧ L'autre méthode consiste à discuter sur l'abscisse x et non plus sur le produit xy.
Pour chaque $x \in [\![1, n]\!]$, le couple (x, x') est situé sous la courbe $xy = n$ si, et seulement si, le produit xx' est inférieur à n. Or x' est un entier non nul, donc on cherche le nombre de multiples (entiers !) de x qui sont inférieurs à n. Par définition, ce nombre est égal à $\lfloor n/x \rfloor$ et donc le nombre de couples cherché est égal à

$$\sum_{x=1}^{n} \left\lfloor \frac{n}{x} \right\rfloor.$$

✍ *On devine (et on peut le constater expérimentalement...) que le code suivant est beaucoup plus efficace pour calculer D_n que le code précédent.*

```
def D_var(n):
    s = 0
    for x in range(1, n+1):
        s += n//x
    return s
```

[2.b] Par définition de la partie entière, $\lfloor x \rfloor \leqslant x \leqslant \lfloor x \rfloor + 1$, donc, d'après la question précédente,

$$D_n = \sum_{k=1}^{n} \left\lfloor \frac{n}{k} \right\rfloor \leqslant \sum_{k=1}^{n} \frac{n}{k} = nH_n \leqslant \sum_{k=1}^{n} \left(\left\lfloor \frac{n}{k} \right\rfloor + 1 \right) = D_n + n.$$

Autrement dit,

$$\forall\, n \in \mathbb{N}^*, \qquad 0 \leqslant nH_n - D_n \leqslant n \tag{$\star$}$$

ce qu'on peut résumer sous la forme $nH_n = D_n + \mathcal{O}(n)$.

❧ Il est bien connu que $H_n \sim \ell\mathrm{n}\, n$ (comparaison somme/intégrale pour une fonction monotone), donc

$$D_n \underset{n\to+\infty}{=} \big[n\,\ell\mathrm{n}\, n + o(n\,\ell\mathrm{n}\, n) \big] + \mathcal{O}(n) = n\,\ell\mathrm{n}\, n + o(n\,\ell\mathrm{n}\, n)$$

et donc $D_n \sim n\,\ell\mathrm{n}\, n$.

✍ *Il s'agit ici d'une simple factorisation : comme $\frac{1}{\ell\mathrm{n}\, n}$ tend vers 0 lorsque n tend vers $+\infty$,*

$$n = \frac{1}{\ell\mathrm{n}\, n} \cdot (n\,\ell\mathrm{n}\, n) \underset{n\to+\infty}{=} o(n\,\ell\mathrm{n}\, n)$$

et on a bien $\mathcal{O}(n) = o(n\,\ell\mathrm{n}\, n)$.

[3.] D'après le cours,

$$\forall\, x \in]-1,1[, \qquad \sum_{n=1}^{+\infty} \frac{x^n}{n} = -\,\ell\mathrm{n}(1-x).$$

❧ Les rayons de convergence des séries $\sum x^n$ et $\sum {}^{x^n}\!/_n$ sont tous les deux égaux à 1, donc le rayon de convergence de leur produit de Cauchy est au moins égal à 1.
La formule générale du produit de Cauchy nous donne

$$\forall\, n \in \mathbb{N}, \quad c_n = \sum_{k=0}^{n} a_k b_{n-k}.$$

Dans notre cas, on a $a_0 = 0$ et $a_k = {}^1\!/_k$ pour tout $k \geqslant 1$ et $b_\ell = 1$ pour tout $\ell \in \mathbb{N}$, donc

$$c_0 = a_0 b_0 = 0 \quad \text{et} \quad \forall\, n \in \mathbb{N}^*, \quad c_n = 0 \times 1 + \sum_{k=1}^{n} \frac{1}{k} \times 1 = H_n.$$

D'après le cours sur le produit de Cauchy,

$$\forall\, x \in]-1,1[, \qquad \sum_{n=1}^{+\infty} H_n x^n = \left(\sum_{n=0}^{+\infty} x^n \right) \left(\sum_{n=1}^{+\infty} \frac{x^n}{n} \right) = \frac{-\,\ell\mathrm{n}(1-x)}{1-x}.$$

❧ Comme le rayon de convergence de la série entière précédente est strictement positif, on peut dériver terme à terme sur l'intervalle ouvert de convergence, ce qui nous donne

$$\forall\, x \in]-1,1[, \quad \sum_{n=1}^{+\infty} nH_n x^{n-1} = \frac{1 - \ell\mathrm{n}(1-x)}{(1-x)^2}$$

et finalement, par une simple multiplication,

$$\forall\, x \in]-1,1[, \quad \sum_{n=1}^{+\infty} nH_n x^n = x \frac{1 - \ell\mathrm{n}(1-x)}{(1-x)^2}.$$

[4.] Pour tout entier $n \geqslant 1$, le nombre d_n de diviseurs de n est évidemment compris entre 2 et n. Par conséquent, la suite de terme général $d_n x^n$ est bornée pour tout $0 \leqslant x < 1$ et elle tend vers $+\infty$ pour tout $x > 1$. Le rayon de convergence de la série entière $\sum d_n x^n$ est donc égal à 1.
La fonction S définie par

$$S(x) = \sum_{n=1}^{+\infty} d_n x^n$$

est donc de classe $\mathscr{C}^\infty$ sur $]-1,1[$.

❧ Comme précédemment, la série $\sum D_n x^n$ peut s'interpréter comme un produit de Cauchy, ce qui nous donne

$$\forall\, x \in]-1,1[, \quad \sum_{n=1}^{+\infty} D_n x^n = \Big(\sum_{n=1}^{+\infty} d_n x^n\Big)\Big(\sum_{n=0}^{+\infty} x^n\Big) = \frac{S(x)}{1-x}.$$

On a encadré plus haut D_n à l'aide de H_n $(\star)$, ce qui nous donne

$$\forall\, 0 \leqslant x < 1, \qquad \sum_{n=1}^{+\infty} D_n x^n \leqslant \sum_{n=1}^{+\infty} n H_n x^n \leqslant \sum_{n=1}^{+\infty} D_n x^n + \sum_{n=1}^{+\infty} n x^n.$$

✍ *Attention, cet encadrement n'a aucune raison d'être vrai pour $-1 < x < 0$!*

Le rayon de convergence de la série entière $\sum x^n$ est strictement positif, donc on peut dériver terme à terme :

$$\sum_{n=1}^{+\infty} n x^n = x \sum_{n=1}^{+\infty} n x^{n-1} = x \cdot \frac{\mathrm{d}}{\mathrm{d}x}\Big(\frac{1}{1-x}\Big) = \frac{x}{(1-x)^2}.$$

❧ On a donc démontré que

$$\forall\, 0 \leqslant x < 1, \quad 0 \leqslant x\frac{1-\ell\mathrm{n}(1-x)}{(1-x)^2} - \frac{S(x)}{1-x} \leqslant \frac{x}{(1-x)^2}.$$

On en déduit que

$$S(x) \underset{x\to 1}{=} x\frac{1-\ell\mathrm{n}(1-x)}{1-x} + \mathcal{O}\Big(\frac{1}{1-x}\Big)$$

et plus simplement que

$$S(x) = \sum_{n=1}^{+\infty} d_n x^n \underset{x\to 1}{\sim} \frac{-\ell\mathrm{n}(1-x)}{1-x}.$$

✍ *Comme la série $\sum d_n$ est grossièrement divergente, il est logique de trouver un équivalent infiniment grand. Comme d_n oscille indéfiniment entre 2 (cas où n est premier) et de grandes valeurs (pour les entiers n qui ont "beaucoup" de diviseurs), il est logique de trouver un équivalent qui est intermédiaire entre la somme de la série entière $\sum x^n$ et celle de la série entière $\sum n x^n$.*

[5.] Pour tout entier $n \in \mathbb{N}$, la fonction u_n définie par $u_n(x) = x^n/(1-x^n)$ est clairement continue sur l'intervalle ouvert $]-1,1[$.
Pour tout entier $n \in \mathbb{N}$ et pour tout réel $0 < a < 1$,

$$\forall\, x \in [-a,a], \qquad \big|u_n(x)\big| \leqslant \frac{|x|^n}{1-a^n} \leqslant \frac{a^n}{1-a^n}.$$

✍ *Si $|x| \leqslant a$, alors $x^n \leqslant |x|^n \leqslant a^n < 1$ et par conséquent*

$$1 - x^n \leqslant 1 - |x|^n \leqslant 1 - a^n > 0.$$

Il faut toujours chercher un encadrement simple avant de se lancer dans l'étude des variations d'une fonction !

Comme $0 < a < 1$, le majorant est équivalent à a^n lorsque n tend vers $+\infty$. On a donc trouvé un majorant indépendant de $x \in [-a,a]$ et qui est le terme général d'une série convergente. On a donc prouvé que la série de fonctions $\sum u_n$ convergeait normalement sur $[-a,a]$ pour tout $0 < a < 1$ et comme les fonctions u_n sont continues, on en déduit que la somme de cette série de fonctions est continue sur l'intervalle ouvert $]-1,1[$ (qui est l'union des segments $[-a,a]$ lorsque a varie entre 0 et 1).

✍ *On peut démontrer (Théorème de Fubini) que*

$$\forall\, x \in]-1,1[, \qquad \sum_{n=1}^{+\infty} \frac{x^n}{1-x^n} = \sum_{k=1}^{+\infty} d_k x^k.$$

Il y a donc bien un rapport entre les deux dernières questions !

2
Calcul matriciel

1. Le plus difficile est de se retrouver dans l'aide-mémoire sans perdre de temps : il faut s'entraîner suffisamment pour savoir quelles sont les fonctions qui viennent du module `numpy` et celles qui viennent du module `numpy.linalg`.

I
Applications

135-py-50 - Approximation de l'exponentielle par la méthode d'Euler

[1.] *Pour $x \in \mathbb{R}$, déterminer*

$$\lim_{n\to+\infty}\left(1+\frac{x}{n}\right)^n.$$

[2.] *Pour $n \in \mathbb{N}^*$, on pose*

$$A_n = \begin{pmatrix} 1 & 1/n & 1/n \\ 1/n & 1 & 1/n \\ 1/n & 1/n & 1 \end{pmatrix}.$$

[2.a] *Écrire une fonction Python d'argument $n \in \mathbb{N}^*$ qui renvoie la matrice A_n.*
[2.b] *Écrire un code Python renvoyant les valeurs propres de A_n^n pour $n \in [\![5, 10]\!]$.*
[2.c] *Diagonaliser A_n et déterminer la limite de A_n^n lorsque n tend vers $+\infty$. Vérifier le résultat avec Python.*
[3.] *On note $A_{n,m} \in \mathfrak{M}_m(\mathbb{R})$ la matrice dont les coefficients diagonaux sont égaux à 1 et les autres coefficients sont égaux à $1/n$.*
[3.a] *Exprimer la matrice $A_{n,m}$ en fonction de $n \in \mathbb{N}^*$, de I_n et de la matrice $J_m \in \mathfrak{M}_m(\mathbb{R})$ dont tous les coefficients sont égaux à 1.*
[3.b] *Déterminer la limite de $A_{n,m}^n$ lorsque n tend vers $+\infty$ comme une combinaison linéaire des matrices I_m et J_m.*

135-py-51 - Fonctions d'une matrice diagonalisable

[1.] *Écrire une fonction appliquer$(f,\ M)$ dont le premier argument est une application $f : \mathbb{C} \to \mathbb{C}$ et le second argument une matrice diagonalisable $M \in \mathfrak{M}_n(\mathbb{C})$. Si la matrice M est égale à*

$$P\,\mathrm{Diag}(a_1, \dots, a_n)P^{-1},$$

alors la fonction doit renvoyer la matrice

$$P\,\mathrm{Diag}\big(f(a_1), \dots, f(a_n)\big)P^{-1}.$$

[2.] *Dans cette question, on suppose que*

$$f = \left[x \mapsto x^2\right] \quad \text{et que} \quad M = \frac{1}{10}\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.$$

Comparer le résultat de appliquer$(f,\ M)$ et la matrice M^2.
[3.a] *Écrire une fonction S(N, M) dont le premier argument est un entier $N \in \mathbb{N}$ et le second une matrice carrée M, qui renvoie la matrice*

$$S_N(M) = \sum_{k=0}^{N} \frac{(-1)^k}{(2k+1)!} M^{2k+1}.$$

[3.b] *Tester avec la matrice M définie à la question précédente pour $N = 9$.*
[3.c] *Comparer avec le résultat de appliquer$(f,\ M)$ pour une fonction f bien choisie.*
[4.] *Soit $M \in \mathfrak{M}_n(\mathbb{C})$, une matrice diagonalisable.*
Soient φ et u, les endomorphismes canoniquement associés aux matrices

$$M = P\,\mathrm{Diag}(a_1, \dots, a_n)P^{-1} \quad \text{et} \quad P\,\mathrm{Diag}\big(f(a_1), \dots, f(a_n)\big)P^{-1}.$$

[4.a] *Démontrer que, pour toute valeur propre λ de M,*

$$u_{|E_\lambda(M)} = f(\lambda)\,\mathrm{Id}_{E_\lambda(M)}$$

où $E_\lambda(M)$ désigne le sous-espace propre de φ associé à la valeur propre λ.
[4.b] *On suppose que*

$$M = Q\,\mathrm{Diag}(b_1,\dots,b_n)Q^{-1}.$$

Démontrer que

$$P\,\mathrm{Diag}\big(f(a_1),\dots,f(a_n)\big)P^{-1} = Q\,\mathrm{Diag}\big(f(b_1),\dots,f(b_n)\big)Q^{-1}.$$

[5.a] *Démontrer que les matrices*

$$M_k = \begin{pmatrix} 1/k & 1 \\ 0 & 0 \end{pmatrix} \qquad et \qquad N_k = \begin{pmatrix} -1/k & 1 \\ 0 & 0 \end{pmatrix}$$

dont diagonalisables.
[5.b] *Soit $f = [x \mapsto |x|]$. Comparer $f(M_k)$ et $f(N_k)$.*

135-py-65 Matrices tridiagonales de Jacobi

On considère les matrices

$$A = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix} \in \mathfrak{M}_n(\mathbb{R}) \qquad et \qquad \forall\, 1 \leqslant \ell \leqslant n, \quad X_\ell = \begin{pmatrix} \sin\frac{\ell\pi}{n+1} \\ \vdots \\ \sin\frac{\ell n\pi}{n+1} \end{pmatrix} \in \mathfrak{M}_{n,1}(\mathbb{R}).$$

On note $P \in \mathfrak{M}_n(\mathbb{R})$, la matrice dont les colonnes sont $X_1,\dots,X_n$. Enfin, pour p et q dans $\mathbb{Z}$, on pose

$$T_q = \sum_{k=1}^{n} \cos\frac{qk\pi}{n+1} \qquad et \qquad S_{p,q} = \sum_{k=1}^{n} \sin\frac{pk\pi}{n+1}\sin\frac{qk\pi}{n+1}.$$

[1.] *Calculer T_1.*
[2.] *La matrice A est-elle diagonalisable ? Que dire de ses sous-espaces propres ?*
[3.] *Écrire une fonction* `A(n)` *qui renvoie la matrice A de taille n.*
[4.] *La fonction suivante renvoie la matrice P de taille n.*

```
def P(n):
    M = np.pi*np.ones((n, n), dtype=int)
    for i in range(n):
        M[i,:] *= (i+1)
    for j in range(n):
        M[:,j] *= (j+1)
    return np.sin(M/(n+1))
```

Donner un script qui calcule $P^{-1}AP$.
[5.] *Proposer une conjecture sur le cardinal du spectre de A ainsi qu'une conjecture sur la matrice P.*
[6.] *Calculer la somme T_p pour tout $p \in \mathbb{Z}$.*
[7.] *Calculer la somme $S_{p,q}$ pour tout $1 \leqslant p,q \leqslant n$.*
[8.] *Démontrer les conjectures faites sur le spectre de A et sur la matrice P.*

135-py-66 - Matrice de Vandermonde pour la Transformation de Fourier discrète

Pour $n \in \mathbb{N}^*$, on pose

$$\omega_n = \exp\frac{2i\pi}{n}, \qquad W_n = (\omega_n^{k\ell})_{0\leqslant k,\ell\leqslant n} \in \mathfrak{M}_n(\mathbb{C}) \qquad et \qquad V_n = \frac{1}{\sqrt{n}}W_n.$$

[1.] *Calculer V_n^2, $V_n\overline{V_n}$ et V_n^4.*
[2.] *Calculer $\det W_n$ puis $\det V_n$.*
[3.] *Écrire une fonction* `V(n)` *qui renvoie la matrice V_n. Tester pour $n = 4$.*
[4.] *Écrire une fonction* `Spectre(n)` *qui renvoie une valeur approchée du spectre de V_n. Conjecturer le nombre de valeurs propres de V_n.*
[5.] *Démontrer cette conjecture.*

Solutions

135-py-50 - Approximation de l'exponentielle par la méthode d'Euler

[1.] Lorsque n est assez grand pour que $|x/n| < 1$, on peut composer par la fonction ℓn.

$$\left(1+\frac{x}{n}\right)^n = \exp\left[n\,\ell\mathrm{n}\left(1+\frac{x}{n}\right)\right] = \exp[x+o(1/n)] \xrightarrow[n\to+\infty]{} e^x.$$

[2.a] Il suffit de ne pas calculer six fois le même quotient pour que le code soit élégant.

```
def A(n):
    q = 1/n
    return np.array([[1,q,q],[q,1,q],[q,q,1]])
```

[2.b] Il faut savoir calculer les puissances d'une matrice (fonction `matrix_power` du module `numpy.linalg`), ainsi que son spectre (la fonction `eigvals` du même module donne une valeur approchée des valeurs propres).

```
for n in range(5, 11):
    M = alg.matrix_power(A(n), n)
    print(alg.eigvals(M))
```

Il semblerait qu'une valeur propre double soit proche de 0,35 et qu'une valeur propre simple soit proche de 6,19.

[2.c] Comme $J_3^2 = 3J_3$, la matrice $\Pi = (1/3)\cdot J_3$ est une matrice de projection. La matrice $I_3 - \Pi$ est donc la projection associée à Π et la décomposition

$$J_3 = 3\cdot\Pi + 0\cdot(I_3-\Pi)$$

nous donne la forme diagonalisée de chaque polynôme en J_3. En particulier, comme

$$A_n = \frac{1}{n}\cdot J_3 + \left(1-\frac{1}{n}\right)\cdot I_3 = \frac{3}{n}\cdot\Pi + \left(1-\frac{1}{n}\right)\cdot(\Pi + I_3 - \Pi) = \left(1+\frac{2}{n}\right)\cdot\Pi + \left(1-\frac{1}{n}\right)\cdot(I_3-\Pi)$$

alors

$$\forall\, k\in\mathbb{N},\qquad A_n^k = \left(1+\frac{2}{n}\right)^k\cdot\Pi + \left(1-\frac{1}{n}\right)^k\cdot(I_3-\Pi).$$

En particulier,

$$A_n^n = \left(1+\frac{2}{n}\right)^n\cdot\Pi + \left(1-\frac{1}{n}\right)^n\cdot(I_3-\Pi) \xrightarrow[n\to+\infty]{} e^2\cdot\Pi + e^{-1}\cdot(I_3-\Pi).$$

❧ **Variante**

Sachant que $J_3^2 = 3J_3$, on sait que les seules valeurs propres possibles pour J_3 sont 0 et 3. En étudiant les noyaux de J_3 et de $J_3 - 3I_3$, on forme la matrice

$$P = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \end{pmatrix}$$

qui est inversible et qui vérifie

$$P^{-1}J_3P = \mathrm{Diag}(3,0,0).$$

On en déduit que

$$P^{-1}A_nP = \frac{1}{n}\cdot P^{-1}J_3P + \left(1-\frac{1}{n}\right)\cdot I_3$$

et donc que

$$P^{-1}A_n^nP = (P^{-1}A_nP)^n = \mathrm{Diag}\big((1+2/n)^n,(1-1/n)^n,(1-1/n)^n\big) \xrightarrow[n\to+\infty]{} \mathrm{Diag}(e^2,e^{-1},e^{-1}).$$

❧ On peut confirmer numériquement cette étude si on sait calculer l'inverse d'une matrice (fonction `inv` du module `numpy.linalg`) ainsi que le produit de trois matrices (fonction `dot`).

```
P = np.array([[1,1,1],[1,-1,0],[1,0,-1]])

def forme_diag(n):
    D = alg.inv(P).dot(A(n)).dot(P)
    return alg.matrix_power(D, n)
```

✍ *Bien évidemment, les coefficients de l'ordre de 10^{-14} ou de 10^{-29} doivent être tenus pour nuls !*
Les coefficients diagonaux de la matrice calculée pour $n = 100$ sont respectivement proches de $0,724$ (une fois) et de $0,366$ (deux fois) et on calcule par ailleurs $e^2 \approx 7,389056$ et $e^{-1} \approx 0,367879$.

[3.a] Comme plus haut, on remarque facilement que $J_m^2 = mJ_m$ et on en déduit que

$$A_{n,m} = \frac{1}{n}\cdot J_m + \left(1-\frac{1}{n}\right)\cdot I_m = \left(1+\frac{m-1}{n}\right)\cdot \Pi_m + \left(1-\frac{1}{n}\right)\cdot (I_m - \Pi_m)$$

où $\Pi_m = (1/m)\cdot J_m$ est une projection.
[3.b] Comme plus haut, Π_m et $(I_m - \Pi_m)$ sont deux projections qui vérifient

$$\Pi_m(I_m - \Pi_m) = (I_m - \Pi_m)\Pi_m = O_m.$$

On en déduit que

$$\forall\, \alpha,\beta \in \mathbb{C},\ \forall\, k \in \mathbb{N}^*,\quad (\alpha\cdot\Pi_m + \beta\cdot[I_m - \Pi_m])^k = \alpha^k\cdot\Pi_m + \beta^k\cdot(I_m - \Pi_m)$$

et en particulier que

$$\forall\, n \in \mathbb{N}^*,\quad A_{n,m}^n = \left(1+\frac{m-1}{n}\right)^n\cdot\Pi_m + \left(1-\frac{1}{n}\right)^n\cdot(I_m - \Pi_m)$$

d'où finalement

$$\begin{aligned}\lim_{n\to+\infty} A_{n,m}^n &= e^{m-1}\cdot\Pi_m + e^{-1}\cdot(I_m - \Pi_m) = \frac{e^{m-1}}{m}\cdot J_m + e^{-1}\cdot\left(I_m - \frac{1}{m}\cdot J_m\right)\\ &= e^{-1}\cdot I_m + \frac{e^{m-1}-e^{-1}}{m}\cdot J_m\end{aligned}$$

❧ On peut vérifier numériquement.

```
def Am(m, n):
    q = 1/n
    return (1-q)*np.eye(m)+q*np.ones((m,m))

m = 4
for k in range(1, 5):
    n = 10**k
    M = Am(m,n)
    print(alg.matrix_power(M, n))
```

135-py-51 - Fonctions d'une matrice diagonalisable

[1.] La fonction `eig` du module `numpy.linalg` nous donne d'une part la liste des valeurs propres et d'autre part une matrice de vecteurs propres si bien que, si la matrice M est diagonalisable, on peut la diagonaliser sans peine.

```
M = 0.1*np.array([[1,2,3],[4,5,6],[7,8,9]])
spectre, P = alg.eig(M)
diag = np.diag(spectre)
```

❧ Pour vérifier ce calcul, il ne faut pas être naïf ! L'exécution de la commande

```
print(P.dot(diag).dot(alg.inv(P))==M)
```

renvoie une matrice dont tous les coefficients sont égaux à `False` !
Les erreurs d'arrondi sont en cause, bien sûr. On contourne le problème en calculant la norme de la différence. On rappelle que la norme euclidienne canonique (ou **norme de Frobenius**) pour une matrice à coefficients réels est définie par $\sqrt{\operatorname{tr} M^\top . M}$.

```
erreur = P.dot(diag).dot(alg.inv(P)) - M
print(np.sqrt(np.trace(erreur.T.dot(erreur))))
```

Il n'est pas mauvais de savoir que `alg.norm(erreur)` donne le même résultat avec moins d'efforts.

✍ *Pour une matrice à coefficients complexes, il vaut mieux calculer la norme produit avec* `abs(M).max()`.

❧ En supposant que la fonction f prise comme second argument s'applique à un tableau `numpy`, on peut donner un code très simple.

```
def appliquer(f, M):
    spectre, P = alg.eig(M)
    diag = np.diag(f(spectre))
    return P.dot(diag).dot(alg.inv(P))
```

[2.]

```
fM = appliquer(lambda x: x**2, M)
M2 = M.dot(M) # plus rapide que alg.matrix_power(M, 2)
```

Comme on s'y attend, la matrice obtenue est égale à la matrice M^2 aux erreurs d'arrondi près (la norme de Frobenius de la différence est de l'ordre de 10^{-15}).
[3.a] Pour limiter au minimum les calculs matriciels, il faut remarquer que le premier terme de la somme ($k = 0$) est égal à M et que

$$\forall\, k \in \mathbb{N}, \quad \frac{(-1)^{k+1}}{(2[k+1]+1)!}\, M^{2(k+1)+1} = \frac{-1}{(2k+2)(2k+3)}\, M^2 \cdot \frac{(-1)^k}{(2k+1)!}\, M^{2k+1}.$$

Il s'agit donc de calculer la somme

$$\sum_{k=0}^{N} U_k = U_0 + \sum_{k=0}^{N-1} U_{k+1} \quad \text{où} \quad U_{k+1} = \frac{-1}{(2k+2)(2k+3)}\, M^2 \cdot U_k.$$

❧ Le code Python s'en déduit facilement, même s'il faut faire attention à bien détacher la variable locale U de l'argument M.
Le produit matriciel $I_n \cdot M$ crée une nouvelle matrice, on aurait pu aussi bien procéder à une copie avec l'instruction `U = copy.deepcopy(M)`.

```
def S(N, M):
    U, M2 = np.eye(len(M)).dot(M), M.dot(M) # copy.deepcopy(M), M.dot(M)
    somme = U
    for k in range(N):
        U = -U.dot(M2)/((2*k+2)*(2*k+3))
        somme += U
    return somme
```

[3.b] On obtient une matrice... Il est intéressant de constater qu'on obtient la même matrice (aux erreurs d'arrondi près) pour $N = 10$, $N = 11$, $N = 12$...

[3.c] Normalement, on a reconnu le développement limité de la fonction sin.
En "appliquant" la fonction sin à la matrice M, on trouve en effet une matrice très proche de la matrice calculée à la question précédente.

```
A = S(9, M)
B = appliquer(np.sin, M)
print(alg.norm(A-B))
```

La norme calculée est de l'ordre de 10^{-16}, on peut donc considérer que les deux matrices sont bien égales.
[4.a] Par définition, un vecteur x appartient au sous-espace propre $E_\lambda(M)$ si, et seulement si, $\varphi(x) = \lambda x$. Il s'agit de vérifier que, pour un tel vecteur x, on a $u(x) = f(\lambda) \cdot x$.

✍ *Si F est un sous-espace stable par un endomorphisme $u \in \mathrm{L}(E)$, la notation u_F désigne l'endomorphisme de F induit par restriction de u, c'est-à-dire l'application linéaire de F dans F définie par*

$$\forall\, x \in F, \qquad u_F(x) = u(x).$$

Si le sous-espace F n'est pas stable par u (si par exemple, comme ici, on n'a pas encore cherché à vérifier si F était stable par u), alors la notation $u_{|F}$ désigne l'application linéaire de F dans E définie par

$$\forall\, x \in F, \qquad u_{|F}(x) = u(x).$$

La seule différence entre ces deux applications est donc l'espace d'arrivée ; la première application linéaire est un endomorphisme : $u_F \in \mathrm{L}(F)$ (endomorphisme de F) mais pas l'autre : $u_{|F} \in \mathrm{L}(E, F)$.

❧ Comme la matrice P est inversible, c'est une matrice de passage : on notera $\mathscr{B} = (\varepsilon_1, \dots, \varepsilon_n)$ la base de E telle que P soit la matrice de passage de la base canonique $\mathscr{B}_{\text{can}}$ à la base $\mathscr{B}$.

$$P = \mathfrak{Mat}(\mathscr{B}_{\text{can}} \to \mathscr{B}) \qquad P^{-1} = \mathfrak{Mat}(\mathscr{B} \to \mathscr{B}_{\text{can}})$$

Par hypothèse, la matrice M représente l'endomorphisme φ dans la base canonique de $\mathbb{R}^n$ et, d'après la formule du changement de base, la matrice $P^{-1}MP$ représente l'endomorphisme φ dans la base $\mathscr{B}$.

$$M = \mathfrak{Mat}_{\mathscr{B}_{\text{can}}}(\varphi) \qquad P^{-1}MP = \mathfrak{Mat}_{\mathscr{B}}(\varphi)$$

❧ Par hypothèse, la matrice $P^{-1}MP = \mathrm{Diag}(a_1, \dots, a_n)$ est diagonale, donc les vecteurs ε_1, …, ε_n de la base $\mathscr{B}$ (= les vecteurs représentés dans la base canonique par les colonnes de la matrice de passage P) sont des vecteurs propres de φ et, plus précisément,

$$\forall\, 1 \leqslant k \leqslant n, \qquad \varphi(\varepsilon_k) = a_k \cdot \varepsilon_k.$$

Le sous-espace propre $E_\lambda(M)$ est donc engendré par les vecteurs ε_k tels que $a_k = \lambda$: si $x \in E_\lambda(M)$, alors il existe des scalaires $(x_k)_{1 \leqslant k \leqslant n}$ tels que

$$x = \sum_{k=1}^{n} x_k \cdot \varepsilon_k = \sum_{\substack{1 \leqslant k \leqslant n : \\ a_k = \lambda}} x_k \cdot \varepsilon_k.$$

❧ Par définition de u, la base $\mathscr{B}$ est aussi une base de vecteurs propres pour u :

$$\forall\, 1 \leqslant k \leqslant n, \qquad u(\varepsilon_k) = f(a_k) \cdot \varepsilon_k$$

et par conséquent

$$u(x) = \sum_{\substack{1 \leqslant k \leqslant n : \\ a_k = \lambda}} x_k \cdot u(\varepsilon_k) = \sum_{\substack{1 \leqslant k \leqslant n : \\ a_k = \lambda}} x_k f(a_k) \cdot \varepsilon_k = f(\lambda) \sum_{\substack{1 \leqslant k \leqslant n : \\ a_k = \lambda}} x_k \cdot \varepsilon_k = f(\lambda) \cdot x.$$

[4.b] On suppose cette fois qu'il existe une autre base $\mathscr{C} = (\eta_k)_{1 \leqslant k \leqslant n}$ constituée de vecteurs propres pour φ :

$$Q = \mathfrak{Mat}(\mathscr{B}_{\text{can}} \to \mathscr{C}) \qquad \mathrm{Diag}(b_1, \dots, b_n) = Q^{-1}MQ = \mathfrak{Mat}_{\mathscr{C}}(\varphi).$$

On a donc

$$\mathrm{Diag}(b_1, \dots, b_n) = Q^{-1}\big(P \times \mathfrak{Mat}_{\mathscr{B}}(\varphi) \times P^{-1}\big)Q = (P^{-1}Q)^{-1}\,\mathrm{Diag}(a_1, \dots, a_n)(P^{-1}Q). \qquad (\star)$$

❧ Les deux familles $(a_k)_{1\leqslant k\leqslant n}$ et $(b_k)_{1\leqslant k\leqslant n}$ représentent les valeurs propres de φ comptées avec multiplicité, elles se déduisent donc l'une de l'autre par une permutation : il existe une permutation $\sigma \in \mathfrak{S}_n$ telle que

$$\forall\, 1 \leqslant k \leqslant n, \qquad b_k = a_{\sigma(k)}.$$

❧ Comme le spectre de φ est un ensemble fini de cardinal inférieur à n, il existe un polynôme $L \in \mathbb{C}[X]$ tel que

$$\forall\, 1 \leqslant k \leqslant n, \qquad L(a_k) = f(a_k). \tag{†}$$

✍ *En général, la théorie des polynômes interpolateurs de Lagrange impose de choisir des abscisses deux à deux distinctes. Ici, on sait que : si $a_k = a_\ell$, alors $f(a_k) = f(a_\ell)$ et cette contrainte fait qu'on n'a pas besoin de supposer que les a_k soient deux à deux distincts.*

Par conséquent,

$$\forall\, 1 \leqslant k \leqslant n, \quad f(b_k) = f(a_{\sigma(k)}) = L(a_{\sigma(k)}) = L(b_k). \tag{‡}$$

On déduit alors de $(\star)$ que

$$L\big(\mathrm{Diag}(b_1,\dots,b_n)\big) = (P^{-1}Q)^{-1} \times L\big(\mathrm{Diag}(a_1,\dots,a_n)\big) \times (P^{-1}Q).$$

✍ *La formule de changement de base $\big[A \mapsto R^{-1}AR\big]$ est un morphisme d'algèbres, donc $L(R^{-1}AR) = R^{-1}L(A)R$, quel que soit le polynôme L.*

❧ Comme L est un polynôme, d'après (†),

$$L\big(\mathrm{Diag}(a_1,\dots,a_n)\big) = \mathrm{Diag}\big(L(a_1),\dots,L(a_n)\big) = \mathrm{Diag}\big(f(a_1),\dots,f(a_n)\big)$$

et, de même, d'après (‡),

$$L\big(\mathrm{Diag}(b_1,\dots,b_n)\big) = \mathrm{Diag}\big(f(b_1),\dots,f(b_n)\big).$$

On a donc démontré que

$$Q\,\mathrm{Diag}\big(f(b_1),\dots,f(b_n)\big)Q^{-1} = P\,\mathrm{Diag}\big(f(a_1),\dots,f(a_n)\big)P^{-1}.$$

❧ **Variante** (plus courte, pas nécessairement plus simple)
Notons v, l'endomorphisme canoniquement associé à la matrice $Q\,\mathrm{Diag}\big(f(b_1),\dots,f(b_n)\big)Q^{-1}$. D'après la question précédente,

$$\forall\, \lambda \in \mathrm{Sp}(M), \qquad v_{|E_\lambda(M)} = f(\lambda)\,\mathrm{Id}_{E_\lambda(M)} = u_{|E_\lambda(M)}.$$

Comme M est supposée diagonalisable,

$$E = \bigoplus_{\lambda\in\mathrm{Sp}(M)} E_\lambda(M)$$

et on vient de constater que les endomorphismes u et v coïncident sur chacun des sous-espaces propres. Par conséquent, $u = v$ et

$$Q\,\mathrm{Diag}\big(f(b_1),\dots,f(b_n)\big)Q^{-1} = \mathfrak{Mat}_{\mathscr{B}_{\mathrm{can}}}(v) = \mathfrak{Mat}_{\mathscr{B}_{\mathrm{can}}}(u) = P\,\mathrm{Diag}\big(f(a_1),\dots,f(a_n)\big)P^{-1}.$$

✍ *Cette propriété rend légitime la notation $f(M)$ pour désigner la matrice $P\,\mathrm{Diag}\big(f(a_1),\dots,f(a_n)\big)P^{-1}$ puisqu'on obtient la même matrice, quelle que soit la manière de diagonaliser la matrice M.*

✍ *Le cours se borne à étudier les matrices $f(M)$ quand f est une application polynomiale. On n'est donc pas obligé de se restreindre aux applications polynomiales.*
Cela dit, comme le spectre d'une matrice est un ensemble fini et que toute application d'une partie finie de $\mathbb{C}$ à valeurs dans $\mathbb{C}$ est en fait une application polynomiale (Théorie de l'interpolation de Lagrange), ce n'est pas une si grande surprise.

[5.a] Les matrices M_k et N_k sont triangulaires, donc leurs valeurs propres sont leurs coefficients diagonaux. Ce sont donc des matrices de $\mathfrak{M}_2(\mathbb{R})$ qui ont deux valeurs propres distinctes, elles sont diagonalisables.
[5.b] Si $P^{-1}M_kP$ et $P^{-1}N_kP$ sont diagonales (les matrices M_k et N_k sont alors des **matrices co-diagonalisables**), alors elles commutent et on en déduit facilement que M_k et N_k commutent.

✍ *La réciproque est vraie, mais moins évidente.*

Mais

$$\forall\, k \in \mathbb{N}^*, \qquad M_k N_k = \frac{1}{k^2}\begin{pmatrix} -1 & k \\ 0 & 0 \end{pmatrix} \neq \frac{1}{k^2}\begin{pmatrix} -1 & -k \\ 0 & 0 \end{pmatrix} = N_k M_k$$

donc il n'existe pas de matrice inversible P telle que $P^{-1}M_kP$ et $P^{-1}N_kP$ soient toutes les deux diagonales.

❧ Les sous-espaces propres de M_k et de N_k sont à peu près aussi évidents que leurs spectres. Avec

$$P = \begin{pmatrix} 1 & -k \\ 0 & 1 \end{pmatrix} \quad \text{et} \quad Q = \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}, \qquad \text{on a} \qquad P^{-1}M_kP = \operatorname{Diag}(1/k, 0) \quad \text{et} \quad Q^{-1}N_kQ = \operatorname{Diag}(-1/k, 0).$$

✍ *Ce n'est pas prudent d'omettre l'indice k sur les matrices de passage, mais c'est bien pratique.*

❧ Par conséquent,

$$P^{-1}f(M_k)P = \operatorname{Diag}(1/k, 0) = P^{-1}M_kP \qquad \text{et} \qquad Q^{-1}f(N_k)Q = \operatorname{Diag}(1/k, 0) = -Q^{-1}N_kQ.$$

On en déduit que

$$\forall\, k \geqslant 2, \qquad f(M_k) = M_k \qquad \text{et que} \qquad f(N_k) = -N_k \neq M_k.$$

✍ *On voit sur cet exemple que deux matrices diagonalisables peuvent avoir des images par f différentes alors que leurs spectres ont même image par f.*

✍ *On peut également vérifier que : si M est semblable à $\operatorname{Diag}(a, b)$ avec $a < 0 < b$, alors $f(M) \neq M$ et $f(M) \neq -M$.*

135-py-65 Matrices tridiagonales de Jacobi

[1.] Voir plus loin le calcul de T_p.

[2.] La matrice A est symétrique réelle, donc elle est diagonalisable et ses sous-espaces propres sont deux à deux orthogonaux.

[3.] Code sans difficulté.

```
def A(n):
    M = np.zeros((n, n), dtype=int)
    for i in range(1, n):
        M[i,i-1] = 1
        M[i-1,i] = 1
    return M
```

❧ Les pythoniens fanatiques auront vu que la matrice A est la somme de deux matrices qui ressemblent à l'identité (`np.eye(n)` – oui, c'est un calembour).

```
def A_var(n):
    J = np.eye(n,k=1,dtype=int)+np.eye(n,k=-1,dtype=int)
    return J
```

[4.] Le produit matriciel s'appelle `dot` et c'est aussi bien une fonction du module `numpy` qu'une méthode de l'objet matrice.

```
  np.dot(A, B)
  A.dot(B)
```

En revanche, le calcul de l'inverse s'effectue grâce à une fonction du sous-module `numpy.linalg`.

```
def conjuguer(n):
    Q = P(n)
    return np.round(alg.inv(Q).dot(A(n).dot(Q)), 2)
```

❧ On a choisi d'arrondir le résultat au centième pour qu'il soit plus lisible.

[5.] Après quelques essais, on constate que $P^{-1}AP$ est diagonale, que les valeurs propres de A varient entre -2 et 2 et que A possède n valeurs propres distinctes.

❧ On a constaté sur les exemples que les colonnes de P étaient des vecteurs propres de A. S'il y a effectivement n valeurs propres deux à deux distinctes, alors les sous-espaces propres sont des droites vectorielles et ces droites sont deux à deux orthogonales. Les colonnes de P sont donc deux à deux orthogonales.
❧ Il est alors tentant de vérifier si la matrice P est orthogonale en calculant $P.P^\top$. Après quelques essais, on devine plutôt que

$$\forall\, n \in \mathbb{N}^*, \qquad P_n.P_n^\top = \frac{n+1}{2}\cdot I_n.$$

[6.] Pour alléger l'écriture des calculs, nous allons poser

$$\theta = \frac{\pi}{2(n+1)} \qquad \text{et} \qquad \omega = \exp\frac{i\pi}{n+1} = \exp(2i\theta)$$

si bien que $\omega^{2(n+1)} = 1$.

✍ *Puisqu'il s'agit d'alléger les calculs (et que l'entier n est fixé), on s'autorise à écrire θ et ω au lieu de θ_n et ω_n.*

❧ Pour tout $p \in \mathbb{Z}$,

$$T_p = \sum_{k=1}^{n} \cos\frac{kp\pi}{n+1} = \mathfrak{Re}\sum_{k=1}^{n}(\omega^p)^k.$$

On reconnaît une somme géométrique de raison ω^p.
❧ Si p est un multiple de $2(n+1)$, alors $\omega^p = 1$ et $T_p = n$.
❧ Si p n'est pas un multiple de $2(n+1)$, alors $\omega^p \neq 1$ et

$$\begin{aligned}\sum_{k=1}^{n}(\omega^p)^k = \omega^p\frac{\omega^{pn}-1}{\omega^p-1} &= e^{2ip\theta}\frac{e^{inp\theta}}{e^{ip\theta}}\frac{2i\sin np\theta}{2i\sin p\theta} = e^{i(n+1)p\theta}\frac{\sin np\theta}{\sin p\theta}\\ &= i^p\frac{\sin np\theta}{\sin p\theta}\end{aligned}$$

puisque $(n+1)\theta = \pi/2$.
Si p est impair, alors la somme est imaginaire pure, donc $T_p = 0$.
Si $p = 2k$ est pair, alors

$$T_{2k} = (-1)^k\frac{\sin 2nk\theta}{\sin 2k\theta}.$$

Mais $2nk\theta + 2k\theta = 2(n+1)k\theta = k\pi$, donc

$$\sin 2nk\theta = \sin(k\pi - 2k\theta) = (-1)^{k+1}\sin 2k\theta$$

et finalement $T_{2k} = -1$.
❧ On distingue donc trois cas :
— si p est impair, alors $T_p = 0$;
— si p est un multiple de $2(n+1)$, alors $T_p = n$;
— si p est pair sans être un multiple de $2(n+1)$, alors $T_p = -1$.

[7.] Quels que soient p et q dans $\mathbb{Z}$,

$$\begin{aligned}S_{p,q} = \sum_{k=1}^{n}\sin 2kp\theta\sin 2kq\theta &= \frac{1}{2}\sum_{k=1}^{n}(\cos 2k(p-q)\theta - \cos 2k(p+q)\theta)\\ &= \frac{T_{p-q}-T_{p+q}}{2}.\end{aligned}$$

❧ Comme $1 \leqslant p, q \leqslant n$, alors $2 \leqslant p+q \leqslant 2n$ et $-n < p-q < n$. Par conséquent, l'entier $p+q$ n'est jamais un multiple de $2(n+1)$ et l'entier $p-q$ est un multiple de $2(n+1)$ si, et seulement si, $p = q$.

✍ *Les entiers $p-q$ et $p+q$ ont même parité.*

❧ Pour $p \neq q$, alors $S_{p,q} = (0-0)/2 = 0$ si $(p-q)$ et $(p+q)$ sont impairs et $S_{p,q} = (-1+1)/2 = 0$ si $(p-q)$ et $(p+q)$ sont pairs.
❧ Pour $p = q$, alors $p+q$ est pair et $S_{p,p} = [n-(-1)]/2 = (n+1)/2$.

❧ Finalement,

$$\forall\, 1 \leqslant p \leqslant n, \quad S_{p,p} = \frac{n+1}{2} \qquad \text{et} \qquad \forall\, 1 \leqslant p < q \leqslant n, \quad S_{p,q} = S_{q,p} = 0.$$

[8.] Les égalités précédentes signifient que, pour le produit scalaire canonique sur $\mathfrak{M}_{n,1}(\mathbb{R})$, les colonnes X_p de la matrice P sont deux à deux orthogonales :

$$\forall\, 1 \leqslant p < q \leqslant n, \qquad \langle\, X_p \,|\, X_q \,\rangle = X_p^\top . X_q = S_{p,q} = 0$$

et que leurs normes sont toutes égales :

$$\forall\, 1 \leqslant p \leqslant n, \qquad \|X_p\| = \sqrt{X_p^\top . X_p} = \sqrt{\frac{n+1}{2}}.$$

Par conséquent,

$$P^\top . P = \frac{n+1}{2} \quad \text{ou, si on préfère,} \quad \sqrt{\frac{2}{n+1}}\, P \in O_n(\mathbb{R}).$$

❧ En linéarisant à nouveau, pour tout $1 \leqslant \ell \leqslant n$,

$$AX_\ell = \begin{pmatrix} \sin 4\ell\theta \\ \vdots \\ \sin 2(k-1)\ell\theta + \sin 2(k+1)\ell\theta \\ \vdots \\ \sin 2n\ell\theta = \sin 2(n-1)\ell\theta + \sin 2(n+1)\ell\theta \end{pmatrix} = 2\cos 2\ell\theta \cdot X_\ell$$

car $\sin 2(n+1)\ell\theta = 0$ pour tout entier ℓ.
On a déjà démontré que $X_\ell \neq 0$ (puisque sa norme n'est pas nulle), donc on a démontré que la colonne X_ℓ est bien un vecteur propre de A associé à la valeur propre $2\cos 2\ell\theta$.

❧ Ainsi, on a trouvé une base orthogonale $(X_\ell)_{1\leqslant\ell\leqslant n}$ de vecteurs propres et les valeurs propres associées

$$2\cos\frac{\pi\ell}{n+1} \qquad \text{pour } 1 \leqslant \ell \leqslant n$$

sont deux à deux distinctes car $0 < 2\ell\theta < \pi$ et cos est injective sur $[0, \pi]$.
Pour tout entier $n \geqslant 1$, la matrice A est donc diagonalisable et

$$P^{-1}AP = \operatorname{Diag}\Big(2\cos\frac{\pi}{n+1},\, 2\cos\frac{2\pi}{n+1},\, \ldots,\, 2\cos\frac{n\pi}{n+1}\Big).$$

✍ *Ici, les vecteurs propres sont parachutés par l'énoncé, mais on peut les trouver avec méthode (et de la culture).*
❧ *Tout d'abord, le lemme d'Hadamard sur les matrices à diagonale fortement dominante montre que $A - \lambda I_n$ est inversible pour tout nombre complexe λ tel que $|\lambda| > 2$.*
❧ *Ensuite, comme la matrice A est symétrique réelle, on sait que ses valeurs propres appartiennent au segment $[-2, 2]$, donc chaque valeur propre de A peut s'écrire sous la forme $\lambda = 2\cos\alpha$ avec $0 \leqslant \alpha \leqslant \pi$.*
❧ *Enfin, on peut remarquer que l'égalité vectoriel $AX = \lambda X$ avec $X = (x_1, \ldots, x_n) \neq 0$ revient à trouver une famille*

$$Y = (y_0, y_1, \ldots, y_n, y_{n+1}) = (0, x_1, \ldots, x_n, 0) \in \mathbb{R}^{n+2} \setminus \{0\}$$

telle que

$$\forall\, 1 \leqslant\ k \leqslant n, \qquad y_{k-1} + y_{k+1} = \lambda y_k = 2\cos\alpha\, y_k.$$

On sait exprimer les familles qui vérifient une telle relation de récurrence linéaire d'ordre deux et on en déduit l'expression des vecteurs propres de A.

135-py-66 - Matrice de Vandermonde pour la Transformation de Fourier discrète

[1.] Notons $c_{k,\ell}$, les coefficients de W_n^2 (pour $0 \leqslant k, \ell < n$). D'après la formule du produit matriciel,

$$c_{k,\ell} = \sum_{0 \leqslant m < n} \omega_n^{km} \omega_n^{m\ell} = \sum_{m=0}^{n-1} (\omega_n^{k+\ell})^m.$$

Si $k+\ell$ est un multiple de n, alors $\omega_n^{k+\ell} = 1$ et $c_{k,\ell} = n$. Mais si $k+\ell$ n'est pas un multiple de n, alors $\omega_n^{k+\ell}$ est une racine n-ième de l'unité distincte de 1 et $c_{k,\ell} = 0$.

❧ Pour $0 \leqslant k, \ell < n$, on a $0 \leqslant k+\ell < 2n$, donc la somme $k+\ell$ est un multiple de n si, et seulement si, elle est nulle (pour $k = \ell = 0$ seulement) ou si elle est égale à n (pour $1 \leqslant k < n$ et $\ell = n-k$). On en déduit que

$$W_n^2 = n \begin{pmatrix} 1 & & \\ & & 1 \\ & 1 & \end{pmatrix} \qquad \text{et donc que} \qquad W_n^4 = n^2 \begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix} = n^2 I_n.$$

❧ Notons $d_{k,\ell}$, les coefficients de $W_n \overline{W_n}$ (pour $0 \leqslant k, \ell < n$). De même,

$$d_{k,\ell} = \sum_{m=0}^{n-1} (\omega_n^{k-\ell})^m$$

donc $d_{k,\ell} = n$ si $k-\ell$ est un multiple de n et $d_{k,\ell} = 0$ sinon.
Pour $0 \leqslant k, \ell < n$, on a $-n < k-\ell < n$, donc $k-\ell$ est un multiple de n si, et seulement si, $k = \ell$. On en déduit que $W_n \overline{W_n} = nI_n$.

❧ Finalement,

$$V_n^2 = \begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix}, \qquad V_n \overline{V_n} = V_n^4 = I_n.$$

[2.] Comme $\det \overline{W_n} = \overline{\det W_n}$ et que $W_n \overline{W_n} = nI_n$, alors $|\det W_n|^2 = n^n$ et donc $|\det W_n| = n^{n/2}$.

❧ La matrice W_n est aussi la matrice de Vandermonde associée aux nombres complexes a^{2k} pour $0 \leqslant k < n$ où $a = e^{i\pi/n}$. Par conséquent,

$$\det W_n = \prod_{0 \leqslant k < \ell < n} (a^{2\ell} - a^{2k}) = \prod_{0 \leqslant k < \ell < n} a^{k+\ell}(a^{(\ell-k)} - a^{-(\ell-k)}) = \prod_{0 \leqslant k < \ell < n} 2ia^{k+\ell} \sin \frac{(\ell-k)\pi}{n}.$$

Pour $0 \leqslant k < \ell < n$, on a $0 < \ell - k < n$, donc chaque sinus est strictement positif et le produit compte $\binom{n}{2}$ facteurs. Par conséquent,

$$\det W_n = (2i)^{\binom{n}{2}} \prod_{0 \leqslant k < \ell < n} \sin \frac{(\ell-k)\pi}{n} \times \prod_{\ell=1}^{n-1} \prod_{k=0}^{\ell-1} a^{k+\ell}$$

avec

$$\prod_{\ell=1}^{n-1} \prod_{k=0}^{\ell-1} a^{k+\ell} = \prod_{\ell=1}^{n-1} a^{\ell^2} \cdot a^{(\ell-1)\ell/2} = a^N \qquad \text{où} \qquad N = \sum_{\ell=1}^{n-1} \frac{3\ell^2 - \ell}{2} = \frac{n(n-1)^2}{2}.$$

✍ *On doit savoir que*

$$\sum_{\ell=1}^{n-1} \ell = \frac{(n-1)n}{2} \qquad \textit{et que} \qquad \sum_{\ell=1}^{n-1} \ell^2 = \frac{(n-1)(2n-1)n}{6}.$$

❧ Finalement, comme $a = e^{i\pi/n}$,

$$\det W_n = 2^{\binom{n}{2}} \prod_{0 \leqslant k < \ell < n} \sin \frac{(\ell-k)\pi}{n} \times i^{\binom{n}{2}} \times i^{(n-1)^2} = |\det W_n| \cdot i^{(n-1)(3n-2)/2}.$$

✍ *En dépit des dénominateurs, tous les exposants qui précèdent sont des entiers — heureusement !*

Et comme $|\det W_n|$ est connu, on en déduit que

$$\det W_n = n^{n/2} \cdot i^{(n-1)(3n-2)/2} \quad \text{puis que} \quad \det V_n = \frac{\det W_n}{(\sqrt{n})^n} = i^{(n-1)(3n-2)/2}.$$

[3.] On peut définir la matrice W_n en fatiguant plus la machine qu'on ne se fatigue les neurones.

```
def W(n):
    w = np.exp(2j*np.pi/n)
    M = np.zeros((n, n), dtype= complex)
    for k in range(n):
        for ell in range(n):
            M[k, ell] = w**(k*ell)
    return M
```

❧ On peut préférer une version plus économique, au prix d'un code un peu plus élaboré.

```
def W(n):
    w = np.exp(2j*np.pi/n)
    M = np.zeros((n, n), dtype= complex)
    wk = 1
    for k in range(n):
        wkl = 1
        for ell in range(n):
            M[k, ell] = wkl
            wkl *= wk
        wk *= w
    return M
```

❧ La matrice W_n étant connue, il est facile d'en déduire V_n.

```
def det_V(n):
    return np.round(alg.det(W(n)/np.sqrt(n)), 2)
```

On reconnaît alors

$$W_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \qquad W_3 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & j & j^2 \\ 1 & j^2 & j \end{pmatrix}, \qquad W_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}.$$

[4.] Pour obtenir une valeur approchée des valeurs propres de V_n, il suffit de connaître la bonne fonction Python.

```
def Spectre(n):
    V = W(n)/np.sqrt(n)
    return np.round(alg.eigvals(V), 2)
```

❧ On sait que $V_n^4 = I_n$, donc les valeurs propres de V_n sont des racines du polynôme annulateur $X^4 - 1$.
Les essais numériques montrent que, pour $n \geqslant 5$, les quatre nombres complexes ± 1 et $\pm i$ sont bien des valeurs propres de V_n.

✍ *En revanche, pour $n \leqslant 3$, la matrice V_n possède au plus n valeurs propres distinctes !*

[5.] Notons e_k, $0 \leqslant k < n$, les vecteurs de la base canonique de $\mathbb{C}^n$. On voit sur la matrice W_n qu'aucun vecteur de la base canonique n'est un vecteur propre de W_n. Par conséquent, chaque couple $(e_k, W_n e_k)$ est une famille libre.

❧ On a démontré que

$$W_n^2 e_0 = n \cdot e_0 \qquad \text{et que} \qquad \forall\, 1 \leqslant k < n, \quad W_n^2 e_k = n \cdot e_{n-k} \quad \text{d'où} \quad W_n^2 e_{n-k} = n \cdot e_k. \tag{†}$$

On en déduit que

$$\forall\, 1 \leqslant k < n \qquad W_n^2(e_k + e_{n-k}) = n \cdot (e_k + e_{n-k}) \qquad \text{et que} \qquad W_n^2(e_k - e_{n-k}) = -n \cdot (e_k - e_{n-k}). \tag{⋆}$$

❧ Pour tout entier $0 \leqslant k < n$, on pose $X_k = W_n e_k$ (= la k-ième colonne de la matrice W_n) et remarquons que la colonne $X_{n-k} = W_n e_{n-k}$ est la conjuguée de la colonne X_k. On a alors

$$W_n(e_k + e_{n-k}) = X_k + \overline{X_k} = 2\begin{pmatrix} 1 \\ \cos\frac{2k\pi}{n} \\ \vdots \\ \cos\frac{2(n-1)k\pi}{n} \end{pmatrix} \quad \text{et} \quad W_n(e_k - e_{n-k}) = X_k - \overline{X_k} = 2i\begin{pmatrix} 0 \\ \sin\frac{2k\pi}{n} \\ \vdots \\ \sin\frac{2(n-1)k\pi}{n} \end{pmatrix}. \tag{‡}$$

❧ Les relations (⋆) montrent que les deux sous-espaces vectoriels

$$P_k = \mathrm{Vect}(e_k + e_{n-k}, X_k + X_{n-k}) \qquad \text{et} \qquad Q_k = \mathrm{Vect}(e_k - e_{n-k}, X_k - X_{n-k})$$

sont stables par W_n, quel que soit l'entier $1 \leqslant k < {}^n\!/_2$.

✍ *C'est parce que $e_k + e_{n-k} = e_{n-k} + e_{n-(n-k)}$ qu'on peut (ou qu'on doit) se limiter aux indices k tels que $k < n - k$, c'est-à-dire tels que $k < {}^n\!/_2$.*

De même, d'après (†), le plan

$$P_0 = \mathrm{Vect}(e_0, X_0)$$

est stable par W_n et, si la dimension $n = 2q$ est paire, alors $e_{n-q} = e_q$ et le plan

$$P_q = \mathrm{Vect}(e_q, X_q)$$

est stable par W_n.

❧ Soit φ, l'endomorphisme de $\mathbb{C}^n$ canoniquement associé à W_n.

Nous allons maintenant représenter matriciellement l'endomorphisme induit par restriction de φ aux différents plans stables que nous venons de rencontrer.

❧ D'après (†), la matrice relative à (e_0, X_0) de l'endomorphisme induit par restriction à P_0 et, si $n = 2p$ est pair, la matrice relative à (e_p, X_p) de l'endomorphisme induit par restriction à Q_0 sont toutes les deux égales à la matrice

$$\begin{pmatrix} 0 & n \\ 1 & 0 \end{pmatrix}.$$

Les valeurs propres de cette matrice sont $\pm\sqrt{n}$, donc les deux réels $\pm\sqrt{n}$ sont des valeurs propres de W_n.

✍ *La matrice W_2 admet $\pm\sqrt{2}$ pour valeurs propres.*

❧ *La matrice W_3 admet $\pm\sqrt{3}$ pour valeurs propres et comme $\mathrm{tr}\, W_3 = 1 + 2j = j - j^2 = i\sqrt{3}$, elle admet $i\sqrt{3}$ comme troisième et dernière valeur propre.*

❧ Si les vecteurs $e_k - e_{n-k}$ et $X_k - X_{n-k}$ ne sont pas colinéaires, alors le sous-espace stable Q_k est un plan et le couple $(e_k - e_{n-k}, X_k - X_{n-k})$ est une base de ce plan.

Dans cette base, l'endomorphisme induit par restriction de φ au plan Q_k est représenté par la matrice

$$\begin{pmatrix} 0 & -n \\ 1 & 0 \end{pmatrix}$$

d'après (⋆). Les valeurs propres de cette matrice : $\pm i\sqrt{n}$ sont alors des valeurs propres de W_n.

✍ *Pour $n = 2$, le vecteur $e_k - e_{n-k} = e_1 - e_1$ est nul et ce cas est sans objet.*

❧ *Pour $n = 3$, les vecteurs $e_k - e_{n-k} = e_1 - e_2 = (0, 1, -1)$ et $X_k - X_{n-k} = (j - j^2) \cdot (0, 1, -1)$ sont proportionnels et le sous-espace stable Q_1 est une droite, ce qui prouve que le vecteur $e_1 - e_2$ qui dirige cette droite est un vecteur propre. Le calcul qui précède montre que la valeur propre associée à ce vecteur est égale à $(j - j^2) = i\sqrt{3}$, ainsi qu'on l'a vu plus haut.*

❧ Pour $n = 4 = 2 \cdot 2$, on a explicité quatre sous-espaces stables :

$$P_0 = \text{Vect}\big((1,0,0,0),(1,1,1,1)\big), \qquad P_1 = \text{Vect}\big((0,1,0,1),(2,0,-2,0)\big)$$
$$P_2 = \text{Vect}\big((0,0,1,0),(1,-1,1,-1)\big), \qquad Q_1 = \text{Vect}\big((0,1,0,-1),(0,2i,0,-2i)\big) = \text{Vect}\big((0,1,0,-1)\big).$$

Le vecteur $(0,1,0,-1)$ qui dirige Q_1 est un vecteur propre de W_4 associé à la valeur propre $2i$. Dans les bases "naturelles" des trois plans P_0, P_1 et P_2, la matrice de l'endomorphisme induit par restriction de φ est la matrice

$$\begin{pmatrix} 0 & 4 \\ 1 & 0 \end{pmatrix},$$

dont les valeurs propres sont ± 2.
Si W_4 admettait une quatrième valeur propre, ce serait nécessairement $-2i$ et les quatre valeurs propres seraient des valeurs propres simples. Dans ces conditions, la trace de W_4 serait nulle, ce qui est faux.
La matrice W_4 n'a donc que trois valeurs propres : $2i$ et ± 2. Comme $\operatorname{tr} W_4 = 2 + 2i$, on en déduit que 2 est une valeur propre double et que $2i$ et -2 sont des valeurs propres simples de W_4.

❧ Considérons maintenant un entier $n \geqslant 5$.
D'après (‡), les vecteurs $e_1 + e_{n-1}$ et $X_1 + X_{n-1}$ ne sont pas proportionnels (comparer les coordonnées d'indice 0) et dans cette base "naturelle" du plan stable P_1, la matrice de l'endomorphisme induit par restriction de φ est

$$\begin{pmatrix} 0 & n \\ 1 & 0 \end{pmatrix}$$

donc $\pm\sqrt{n}$ sont des valeurs propres de W_n.
De même, les vecteurs

$$e_1 - e_{n-1} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix} \qquad \text{et} \qquad X_1 - X_{n-1} = 2i \begin{pmatrix} 0 \\ \sin\frac{2\pi}{n} \\ \sin\frac{4\pi}{n} \\ \vdots \\ -\sin\frac{4\pi}{n} \\ -\sin\frac{2\pi}{n} \end{pmatrix}$$

ne sont pas proportionnels ($\sin {}^{4\pi}/_{n} > 0$ car $n \geqslant 5$). Donc le sous-espace stable Q_1 est un plan et, d'après ($\star$), la matrice de l'endomorphisme induit par restriction de φ au plan Q_1 relative à la base "naturelle" de ce plan est

$$\begin{pmatrix} 0 & -n \\ 1 & 0 \end{pmatrix}$$

donc $\pm i\sqrt{n}$ sont des valeurs propres de W_n.

❧ On a ainsi démontré que, pour tout entier $n \geqslant 5$, la matrice V_n admettait exactement quatre valeurs propres : ± 1 et $\pm i$.

3
Variables aléatoires

I

Applications

135-py-40 - Loi des évènements rares

Toutes les variables aléatoires sont définies sur un même espace probabilisé $(\Omega, \mathcal{A}, \mathbf{P})$.
Soient $\lambda > 0$ *et* Y*, une variable aléatoire suivant la loi de Poisson* $\mathscr{P}(\lambda)$.
Pour tout entier $n > \lambda$*, on considère une variable aléatoire* S_n *suivant la loi binomiale* $\mathscr{B}(n, \lambda/n)$.

[1.] *Étant donné une fonction bornée* f *et les deux réels* $\varepsilon > 0$ *et* $\lambda > 0$ *comme arguments, écrire une fonction qui renvoie une valeur approchée de* $\mathbf{E}[f(Y)]$ *à* $\varepsilon \cdot \|f\|_\infty$ *près.*

[2.] *Étant donné une fonction* f*, le réel* $\lambda > 0$ *et l'entier* $n > \lambda$ *comme arguments, écrire une fonction qui renvoie la valeur de* $\mathbf{E}[f(S_n)]$.

[3.a] *On choisit ici* $\lambda = 2$ *et* $f(x) = e^{-x}$*. Afficher les valeurs de* $\mathbf{E}[f(S_n)]$ *pour* $n \in [\![3, 100]\!]$*. Comparer avec* $\mathbf{E}[f(Y)]$ *à* 10^{-4} *près.*

[3.b] *Mêmes questions avec* $f(x) = \frac{1}{1+x^2}$*. Établir une conjecture.*

[3.c] *Démontrer cette conjecture.*

[4.] *Soient* X *et* Y*, deux variables aléatoires à valeurs dans* $\mathbb{N}$*. Démontrer que*

$$\forall\, A \subset \mathbb{N}, \qquad \big|\mathbf{P}(X \in A) - \mathbf{P}(Y \in A)\big| \leqslant \mathbf{P}(X \neq Y).$$

Solutions

135-py-40 - Loi des évènements rares

[1.] Comme la fonction f est bornée, la variable aléatoire $f(Y)$ est bornée, donc c'est une variable aléatoire d'espérance finie et, d'après la formule de transfert,

$$\mathbf{E}\big[f(Y)\big] = \sum_{k=0}^{+\infty} f(k)\,\mathbf{P}(Y=k) = \sum_{k=0}^{+\infty} e^{-\lambda} f(k)\frac{\lambda^k}{k!}.$$

✍ *Si on sait que la variable aléatoire $f(Y)$ est d'espérance finie, alors la famille $\big(f(k)\,\mathbf{P}(Y=k)\big)_{k\in\mathbb{N}}$ est sommable et il n'est donc pas nécessaire de justifier cette propriété.*

❧ D'après la formule de la série exponentielle et l'inégalité de Taylor-Lagrange,

$$\forall\,\lambda>0,\ \forall\,N\in\mathbb{N},\quad 0\leqslant \sum_{n=N}^{+\infty}\frac{\lambda^n}{n!} = e^\lambda - \sum_{n=0}^{N-1}\frac{\lambda^n}{n!} \leqslant \frac{\lambda^N}{N!}e^\lambda$$

et donc

$$\forall\,\lambda>0,\ \forall\,N\in\mathbb{N},\quad 0\leqslant \sum_{n=N}^{+\infty} e^{-\lambda}\frac{\lambda^n}{n!} \leqslant \frac{\lambda^N}{N!}.$$

✍ *Avec $f(t)=e^t$, on a $f^{(N)}(t)=e^t$ et par conséquent $\big|f^{(N)}(t)\big|\leqslant e^\lambda$ pour tout $t\in[0,\lambda]$.*

❧ Comme la fonction f est supposée bornée,

$$\forall\,k\in\mathbb{N},\quad \left|f(k)e^{-\lambda}\frac{\lambda^k}{k!}\right| \leqslant \|f\|_\infty\, e^{-\lambda}\frac{\lambda^k}{k!}$$

et comme le majorant est le terme général d'une série convergente, on en déduit que

$$\forall\,N\in\mathbb{N},\quad \left|\sum_{k=N}^{+\infty} f(k)e^{-\lambda}\frac{\lambda^k}{k!}\right| \leqslant \|f\|_\infty\Big(\sum_{k=N}^{+\infty} e^{-\lambda}\frac{\lambda^k}{k!}\Big) \leqslant \|f\|_\infty\,\frac{\lambda^N}{N!}$$

par inégalité triangulaire.

❧ Cette estimation nous permet de contrôler la précision lors du calcul approché de l'espérance de $f(Y)$:

$$\mathbf{E}\big(f(Y)\big) = \sum_{k=0}^{+\infty} f(k)e^{-\lambda}\frac{\lambda^k}{k!} \approx \sum_{k=0}^{N-1} f(k)e^{-\lambda}\frac{\lambda^k}{k!}.$$

❧ On commence par une fonction auxiliaire qui détermine la valeur de N en fonction du paramètre λ et de la tolérance ε choisie.

```
def nb_iterations(lbd, eps):
    k, erreur = 1, lbd
    while erreur>eps:
        k += 1
        erreur *= lbd/k
    return k
```

On calcule ${}^{\lambda^N}\!/_{N!}$ par récurrence en remarquant que $\frac{\lambda^k}{k!} = \frac{\lambda^{k-1}}{(k-1)!}\times\frac{\lambda}{k}$.

❧ Connaissant le nombre d'itérations à effectuer pour assure la précision voulue, on calcule une somme partielle de la série.

```
def esp_Poisson(f, lbd, eps):
    N = nb_iterations(lbd, eps)
    s, pk = 0, 1
    for k in range(N):
        s += f(k)*pk
        pk *= lbd/(k+1)
    return s*np.exp(-lbd)
```

Ici encore, on calcule $\mathbf{P}(Y=k)$ par récurrence et on attend la dernière minute pour faire apparaître le facteur $e^{-\lambda}$.

[2.] Toujours d'après la formule de transfert,

$$\mathbf{E}\big[f(S_n)\big] = \sum_{k=0}^{n} f(k)\binom{n}{k}\Big(\frac{\lambda}{n}\Big)^k\Big(1-\frac{\lambda}{n}\Big)^{n-k} = \sum_{k=0}^{n} f(k)\cdot \underbrace{\frac{\lambda^k}{k!}\cdot\Big[\frac{n!}{(n-k)!}\cdot\frac{(n-\lambda)^{n-k}}{n^n}\Big]}_{p_k}.$$

✍ *Il s'agit ici de calculer une somme finie, donc on peut calculer la valeur exacte de l'espérance — exacte aux erreurs d'arrondi près, bien entendu !*

❧ Pour $k=0$, le cofacteur p_k de $f(k)$ est égal à $(1-{}^{\lambda}/_{n})^n$ et pour $0\leqslant k<n$,

$$p_{k+1} = p_k\cdot\frac{\lambda}{k+1}\cdot\frac{n-k}{n-\lambda}.$$

Il reste alors à coder le calcul de

$$\mathbf{E}\big[f(S_n)\big] = f(0)\cdot p_0 + \sum_{0\leqslant k<n} f(k+1)\cdot p_{k+1}.$$

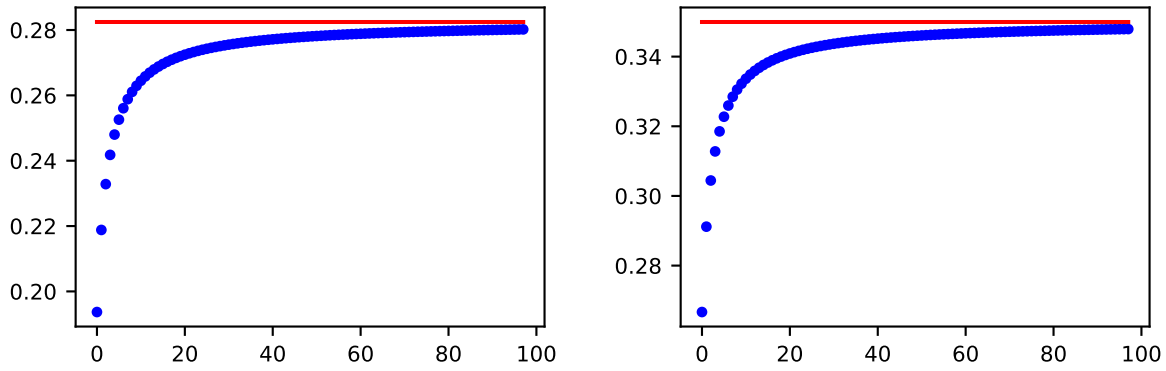
```
def esp_binomiale(f, n, lbd):
    pk = (1-lbd/n)**n
    s = pk*f(0)
    for k in range(n):
        pk *= lbd*(n-k)/((k+1)*(n-lbd))
        s += f(k+1)*pk
    return s
```

[3.a] Le code est sans aucun mystère et son exécution suggère que $\mathbf{E}[f(S_n)]$ se rapproche de $\mathbf{E}[f(Y)]$ (voir plus bas, figure de gauche).

```
lbd = 2
def f(x):
    return np.exp(-x)
E_bin = [esp_binomiale(f, n, lbd) for n in range(3, 101)]
E_P   = [esp_Poisson(f, lbd, 1e-2)]*len(E_bin)
plt.plot(E_bin, 'b.')
plt.plot(E_P, 'r')
```

[3.b] Le code est analogue, on introduit une autre manière de définir une fonction.
Cette fois encore, l'espérance $\mathbf{E}[f(S_n)]$ se rapproche de $\mathbf{E}[f(Y)]$ lorsque n augmente (figure de droite ci-dessous).

```
f = lambda x:1/(1+x**2)
E_bin = [esp_binomiale(f, n, lbd) for n in range(3, 101)]
E_P   = [esp_Poisson(f, lbd, 1e-2)]*len(E_bin)
plt.plot(E_bin, 'b.')
plt.plot(E_P, 'r')
```



[3.c] Nous allons démontrer que $\mathbf{E}[f(S_n)]$ converge vers $\mathbf{E}[f(Y)]$ lorsque n tend vers $+\infty$, quelle que soit la fonction f continue et bornée sur $\mathbb{R}_+$.

✍ *Cette question ne figurait pas dans l'énoncé original.*

❧ D'après le cours ("loi des évènements rares"),

$$\forall\, k \in \mathbb{N}, \quad \lim_{n\to+\infty} \mathbf{P}(S_n = k) = \mathbf{P}(Y = k)$$

et donc

$$\forall\, N \in \mathbb{N}, \quad \lim_{n\to+\infty} \sum_{k=0}^{N-1} \big|\, \mathbf{P}(S_n = k) - \mathbf{P}(Y = k)\big| = 0. \tag{$\star$}$$

❧ Fixons un réel $\varepsilon > 0$. Les restes d'une série convergente tendent vers 0, donc il existe un entier $N \in \mathbb{N}$ tel que

$$0 \leqslant \sum_{k=N}^{+\infty} \mathbf{P}(Y = k) \leqslant \varepsilon. \tag{$\dagger$}$$

Cet entier N étant fixé, on déduit de $(\star)$ qu'il existe un entier $n_0 \in \mathbb{N}$ tel que

$$\forall\, n \geqslant n_0, \qquad \sum_{k=0}^{N-1} \big|\, \mathbf{P}(S_n = k) - \mathbf{P}(Y = k)\big| \leqslant \varepsilon. \tag{$\ddagger$}$$

❧ Comme f est bornée, on déduit de la formule de transfert et de l'inégalité triangulaire que

$$\begin{aligned}\big|\, \mathbf{E}\big(f(Y)\big) - \mathbf{E}\big(f(S_n)\big)\big| &\leqslant \sum_{k=0}^{N-1} \big|f(k)\big| \cdot \big|\, \mathbf{P}(Y = k) - \mathbf{P}(S_n = k)\big| + \sum_{k=N}^{+\infty} \big|f(k)\big| \cdot \big(\mathbf{P}(Y = k) + \mathbf{P}(S_n = k)\big) \\ &\leqslant \|f\|_\infty \cdot \Big(\varepsilon + \varepsilon + \sum_{k=N}^{+\infty} \mathbf{P}(S_n = k)\Big)\end{aligned}$$

d'après $(\dagger)$ et $(\ddagger)$.

❧ Également de (†) et de (‡), on déduit que

$$\sum_{k=N}^{+\infty} \mathbf{P}(S_n = k) = 1 - \sum_{k=0}^{N-1} \mathbf{P}(S_n = k) = \Big(1 - \sum_{k=0}^{N-1} \mathbf{P}(Y = k)\Big) + \sum_{k=0}^{N-1} \big[\mathbf{P}(Y = k) - \mathbf{P}(S_n = k)\big]$$
$$= \sum_{k=N}^{+\infty} \mathbf{P}(Y = k) + \sum_{k=0}^{N-1} \big[\mathbf{P}(Y = k) - \mathbf{P}(S_n = k)\big]$$
$$\leqslant \sum_{k=N}^{+\infty} \mathbf{P}(Y = k) + \sum_{k=0}^{N-1} \big|\mathbf{P}(Y = k) - \mathbf{P}(S_n = k)\big| \leqslant 2\varepsilon.$$

On a ainsi prouvé qu'il existait un entier n_0 tel que

$$\forall\, n \geqslant n_0, \qquad \big|\mathbf{E}[f(Y)] - \mathbf{E}[f(S_n)]\big| \leqslant 4\|f\|_\infty\, \varepsilon$$

et donc que $\mathbf{E}[f(S_n)]$ converge effectivement vers $\mathbf{E}[f(Y)]$ quand n tend vers $+\infty$.
[4.]

✍ *Si on admet que $[X = Y] \in \mathcal{A}$ pour toutes les variables aléatoires X et Y définies sur l'espace probabilisé $(\Omega, \mathcal{A}, \mathbf{P})$, alors $[X \neq Y] = [X = Y]^c \in \mathcal{A}$ et la démonstration suivante vaut pour* toutes *les variables aléatoires et non pas seulement pour les variables aléatoires discrètes.*

Il est clair que

$$[Y \in A] = [Y \in A, X = Y] \sqcup [Y \in A, X \neq Y]$$

donc

$$\mathbf{P}(Y \in A) \leqslant \mathbf{P}(Y \in A, X = Y) + \mathbf{P}(Y \in A, X \neq Y)$$

mais il est aussi clair que

$$[Y \in A] \cap [X = Y] \subset [X \in A] \quad \text{et que} \quad [Y \in A, X \neq Y] \subset [X \neq Y]$$

donc

$$\mathbf{P}(Y \in A) \leqslant \mathbf{P}(X \in A) + \mathbf{P}(X \neq Y).$$

❧ On a ainsi démontré que

$$\mathbf{P}(Y \in A) \leqslant \mathbf{P}(X \in A) + \mathbf{P}(X \neq Y).$$

Par symétrie, on en déduit que

$$\mathbf{P}(X \in A) \leqslant \mathbf{P}(Y \in A) + \mathbf{P}(X \neq Y)$$

et finalement, on a démontré que

$$\big|\mathbf{P}(Y \in A) - \mathbf{P}(X \in A)\big| \leqslant \mathbf{P}(X \neq Y).$$

✍ *Cette question, totalement hors sujet, figurait dans l'énoncé original.*

4
Analyse numérique

Équations différentielles

1. La fonction `odeint` du module `scipy.integrate` permet de calculer de manière approchée la solution d'un problème de Cauchy du premier ordre :

$$X(a) = X_0 \qquad \forall\, t \in [a,b], \quad X'(t) = f\big(X(t), t\big).$$

Toute autre équation différentielle doit donc être ramenée à une équation du premier ordre.

1.1 Un système différentiel du premier ordre

$$\forall\, t \in I, \qquad \begin{cases} x'(t) = a(t)x(t) + b(t)y(t) \\ y'(t) = c(t)x(t) + d(t)y(t) \end{cases}$$

devient une équation du premier ordre en considérant une inconnue à valeurs vectorielles :

$$\forall\, t \in I, \quad X'(t) = A(t)X(t)$$

avec

$$A(t) = \begin{pmatrix} a(t) & b(t) \\ c(t) & d(t) \end{pmatrix}, \quad X(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}, \quad X'(t) = \begin{pmatrix} x'(t) \\ y'(t) \end{pmatrix}.$$

1.2 De manière analogue, une équation du second ordre

$$\forall\, t \in I, \qquad a(t)x''(t) + b(t)x'(t) + c(t)x(t) = g(t)$$

devient une équation du premier ordre en considérant une inconnue à valeurs vectorielles :

$$\forall\, t \in I, \quad X'(t) = A(t)X(t) + B(t)$$

avec

$$A(t) = \frac{1}{a(t)} \begin{pmatrix} 0 & a(t) \\ -c(t) & -b(t) \end{pmatrix}, \quad B(t) = \frac{1}{a(t)} \begin{pmatrix} 0 \\ g(t) \end{pmatrix}, \quad X(t) = \begin{pmatrix} x(t) \\ x'(t) \end{pmatrix}, \quad X'(t) = \begin{pmatrix} x'(t) \\ x''(t) \end{pmatrix}$$

à condition, bien entendu que la fonction $a(t)$ ne s'annule pas sur l'intervalle I.

2.1 Pour une résolution numérique (approchée), il est nécessaire de choisir une **condition initiale**, c'est-à-dire un *instant initial* t_0, choisi comme origine des temps, et une *position initiale* (réelle ou vectorielle) qui est la valeur imposée à la solution $X(t)$ à l'instant $t = t_0$.

2.2 De même, une résolution numérique n'a de sens que sur un intervalle de temps borné.

2.3 Pour une condition initiale (t_0, X_0), la fonction `odeint` impose de fournir la position initiale X_0 (réel ou vectoriel) en deuxième argument et un intervalle de temps discrétisé sous la forme d'une liste *monotone* d'instants :

$$t_0 \leqslant t_1 \leqslant t_2 \leqslant \cdots \leqslant t_f \qquad \text{ou} \qquad t_0 \geqslant t_1 \geqslant t_2 \geqslant \cdots \geqslant t_f.$$

Dans le cas où les instants sont croissants, on résout sur l'intervalle $[t_0, t_f]$ et dans le cas où les instants sont décroissants, on résout sur l'intervalle $[t_f, t_0]$.

2.4 Pour tracer l'allure de la solution sur l'intervalle $[-1, 1]$ en imposant une condition initiale $(t_0, X_0) = (0, X_0)$, il faut donc résoudre en deux étapes : sur $[0, 1]$ (avec des instants croissants) et sur $[-1, 0]$ (avec des instants décroissants).

I

Applications

135-py-32 - Résolution d'une équation différentielle linéaire

On considère l'équation différentielle

$$\forall\, x \in \mathbb{R}, \qquad (1+x^2)y''(x) + xy'(x) - \frac{y(x)}{4} = 0. \tag{E}$$

[1.] *On commence par une résolution purement numérique.*
[1.a] *Démontrer qu'il existe une, et une seule, solution de (E) telle que* $y(0) = \sqrt{3}$ *et* $y'(0) = 0$.
[1.b] *Compléter le code suivant pour tracer l'allure du graphe de cette solution pour* $t \in [-1, 1]$.

```
def f(x, t):
  return np.array(...)

T = np.arange(0, 1, 0.01)
X = integr.odeint(...)
plt.plot(T, ...)
T = np.arange(0, -1, -0.01)
X = integr.odeint(...)
plt.plot(T, ...)
```

[2.] *Dans cette question, on cherche les solutions qui sont développables en série entière.*
[2.a] *Déterminer les solutions de (E) qui sont développables en série entière. (On ne demande pas d'expliciter leur somme.)*
[2.b] *Démontrer que toute solution de (E) est développable en série entière au voisinage de l'origine.*
[2.c] *En déduire une autre façon de tracer la courbe tracée à la question précédente.*
[3.] *Dans cette dernière question, on résout littéralement l'équation (E) au moyen d'un changement de variable bien choisi.*
[3.a] *Soit h, une solution de (E). Déterminer une équation différentielle vérifiée par* $g = h \circ \mathrm{sh}$.
[3.b] *En déduire les solutions de (E) sur* $\mathbb{R}$.

135-py-34 - Convergence uniforme

Pour $x \in [0,1]$ *et* $n \in \mathbb{N}^*$, *on pose*

$$P_n(x) = x\prod_{k=1}^{n}\Big(1 - \frac{x}{k}\Big).$$

[1.a] *Écrire une fonction qui prend n et x comme arguments et qui renvoie la valeur de* $P_n(x)$.
[1.b] *Tracer le graphe de* P_n *sur* $[0,1]$ *pour* $n \in [\![1, 10]\!]$ *sur une même figure. En déduire une conjecture sur la suite* $(P_n)_{n\geqslant 1}$.
[2.a] *Démontrer que* P_n *atteint un maximum sur* $[0,1]$.
[2.b] *Écrire une fonction Max(n, eps) qui renvoie une valeur approchée à* ε *près d'un réel* x_n *tel que*

$$P_n(x_n) = \max_{x\in[0,1]} P_n(x).$$

[2.c] *Calculer une liste avec les valeurs de* x_n $\ell\mathrm{n}\, n$ *pour* $n = 10^i$ *avec* $i \in [\![1, 5]\!]$.
[3.a] *Démontrer que*

$$\forall\, n \geqslant 2, \qquad \sum_{k=1}^{n}\frac{1}{k} \leqslant \frac{1}{x_n} \leqslant \frac{1}{1-x_n} + \sum_{k=1}^{n-1}\frac{1}{k}.$$

[3.b] *En déduire un équivalent de* x_n.
[3.c] *Que peut-on en déduire ?*

135-py-35 - Produit eulérien du sinus

Pour tout entier $n \geqslant 1$, *on pose*

$$u_n = \prod_{k=1}^{n}\Big(1 - \frac{1}{4k^2}\Big).$$

[1.] *Déterminer une valeur approchée de* u_n *pour* $n \in [\![1, 10]\!]$ *et de* u_{10^k} *pour* $k \in [\![1, 4]\!]$.
Proposer une conjecture sur la suite $(u_n)_{n\geqslant 1}$.

[2.] *Pour tout réel* $0 < t < \pi$, *on pose*

$$f(t) = \sum_{n=1}^{+\infty} \frac{2t}{t^2 - n^2\pi^2} \quad et \quad g(t) = \frac{\cos t}{\sin t} - \frac{1}{t}.$$

[2.a] *Tracer les graphes de* f *et de* g. *Quelle conjecture peut-on en déduire ?* ***On admettra cette conjecture dans la suite.***
[2.b] *Déterminer la limite de* g *au voisinage de* 0.
[2.c] *Démontrer que les fonctions* f *et* g *peuvent être prolongées en des fonctions continues sur* $[0, \pi[$.
[3.] *En calculant l'intégrale de* $g(t)$ *sur* $[0, x]$ *de deux manières, déterminer la limite de*

$$\sum_{k=1}^{n} \ell n\Big(1 - \frac{x^2}{k^2\pi^2}\Big)$$

lorsque n *tend vers* $+\infty$.
[4.] *La fonction* f *est-elle intégrable sur* $]0, \pi[$ *?*

135-py-38 - Développement eulérien
Pour tout $x \in \mathbb{R} \setminus \mathbb{Z}$, *on pose*

$$f(x) = \sum_{n\in\mathbb{Z}} \frac{1}{(n-x)^2} \quad et \quad g(x) = f(x) - \frac{\pi^2}{\sin^2(\pi x)}.$$

[1.a] *Démontrer que* f *est bien définie sur* $\mathbb{R} \setminus \mathbb{Z}$.
[1.b] *Démontrer que* f *est* 1*-périodique.*
[1.c] *Démontrer que* f *est continue sur* $\mathbb{R} \setminus \mathbb{Z}$.
[1.d] *Tracer l'allure du graphe de* f *sur l'intervalle* $[10,1; 10,9]$.
[2.] *Démontrer que la fonction* g *admet un prolongement continu sur* $\mathbb{R}$.
[3.] *Tracer le graphe de* g. *Quelle conjecture peut-on faire sur* f *?*

135-py-43 - Inégalité de Gronwall
Soit $q \in \mathscr{C}^0(\mathbb{R}_+, \mathbb{R})$, *une fonction intégrable. On considère une fonction* f, *solution du problème de Cauchy*

$$\forall\, x \in \mathbb{R}, \quad y''(x) + [1 + q(x)]y(x) = 0, \quad y(0) = a, \quad y'(0) = b. \tag{E}$$

[1.] *Écrire une fonction trace* $(a,\ b,\ u,\ v,\ q)$ *qui trace l'allure du graphe d'une solution du problème de Cauchy (E) sur le segment* $[u, v]$. *Quelle conjecture peut-on faire ?*
[2.] *Pour tout* $x \in \mathbb{R}_+$, *on pose*

$$z(x) = f(x) + \int_0^x \sin(x - t)q(t)f(t)\,\mathrm{d}t.$$

[2.a] *Démontrer que* z *est de classe* $\mathscr{C}^2$ *et simplifier l'expression* $z''(x) + z(x)$.
[2.b] *Démontrer que*

$$\forall\, x \in \mathbb{R}_+, \quad \big|f(x)\big| \leqslant |a| + |b| + \int_0^x \big|q(t)f(t)\big|\,\mathrm{d}t.$$

[2.c] *En déduire que la fonction* f *est bornée.*

135-py-44 - Intégrale fonction d'un paramètre
Soit E, *l'espace vectoriel des fonctions continues et bornées sur* $\mathbb{R}$. *Pour* $f \in E$, *on pose*

$$\forall\, x \in \mathbb{R}, \quad \Phi(f)(x) = \int_0^{+\infty} \mathrm{Arctan}(tx)\frac{f(t)}{1+t^2}\,\mathrm{d}t.$$

[1.] *Démontrer que* Φ *est un endomorphisme de* E.
[2.] *Soit* $f \in E$. *Démontrer que* $\Phi(f)$ *est de classe* $\mathscr{C}^1$ *sur* $]0, +\infty[$.
[3.] *Soit* f_0, *la fonction constante, égale à* 1. *On pose* $g = \Phi(f_0)$.
[3.a] *Tracer l'allure du graphe de* g *sur* $[0, 5]$. *Conjecturer la limite de* g *au voisinage de* $+\infty$.
[3.b] *Déterminer la limite de* g *au voisinage de* $+\infty$.
[3.c] *Simplifier l'expression* $g(x) + g(1/x)$. *(On pourra s'aider d'un graphe tracé à l'ordinateur.)*

Solutions

135-py-32 - Résolution d'une équation différentielle linéaire

[1.a] L'équation différentielle (E) est une équation du second ordre linéaire et homogène. Ses coefficients sont clairement des fonctions continues et le coefficient de $y''(x)$ ne s'annule pas. Par conséquent, on peut réécrire l'équation (E) sous forme résoluble d'un système linéaire du premier ordre :

$$\forall\, x \in \mathbb{R}, \qquad \begin{pmatrix} y'(x) \\ y''(x) \end{pmatrix} = \frac{1}{4(1+x^2)} \begin{pmatrix} 0 & 4 \\ 1 & 4x \end{pmatrix} \begin{pmatrix} y(x) \\ y'(x) \end{pmatrix}. \tag{S}$$

D'après le Théorème de Cauchy-Lipschitz, quel que soit $x_0 \in \mathbb{R}$ et $(y_0, v_0) \in \mathbb{R}^2$, le système (S) admet une unique solution $Y(x)$ telle que $Y(x_0) = (y_0, v_0)$. Autrement dit : l'équation (E) admet une unique solution $y(x)$ telle que $y(x_0) = y_0$ et $y'(x_0) = v_0$.

[1.b] Dans l'équation (E), le facteur de la dérivée seconde ne s'annule pas, donc on n'a aucune difficulté à écrire l'équation sous forme résoluble.

$$\forall\, t \in \mathbb{R}, \quad \begin{pmatrix} x'(t) \\ x''(t) \end{pmatrix} = \frac{1}{1+t^2} \begin{pmatrix} x'(t) \\ 1/4\, x(t) - t x'(t) \end{pmatrix}$$

Sous cette forme, l'inconnue est le couple $X = (x, x')$, donc à chaque instant, la position $x(t)$ est représentée par `X[0]` et la vitesse $x'(t)$ par `X[1]`.
On en déduit l'expression de la fonction f.

```
def f(X, t):
    return  np.array([X[1], (X[0]/4-t*X[1])/(1+t**2)])
```

❧ On résout dans un premier temps sur l'intervalle $[0,1]$, en précisant la condition initiale : $t = 0$, $x(0) = \sqrt{3}$ et $x'(0) = 0$.
La solution `X` est un tableau `numpy` de la forme

$$X = \big((x_k, v_k)\big)_{0 \leqslant k < N}.$$

On récupère les positions successives en fonction du temps $(x_k)_{0 \leqslant k < N}$ avec `X[:,0]`. On pourrait récupérer de même les vitesses successives en fonction du temps $(v_k)_{0 \leqslant k < N}$ avec `X[:,1]`.

```
T = np.arange(0, 1, 0.01)
Xp = integr.odeint(f, np.array([np.sqrt(3), 0]), T)
plt.plot(T, Xp[:,0], 'r')
```

❧ On résout ensuite sur l'intervalle $[-1,0]$ en conservant la même condition initiale, donc en considérant une discrétisation *décroissante* de l'intervalle $[-1,0]$.

```
T = np.arange(0, -1, -0.01)
Xm = integr.odeint(f, np.array([np.sqrt(3), 0]), T)
plt.plot(T, Xm[:,0], 'r')
```

✍ *On a imposé la même couleur (rouge) sur les deux tracés pour obtenir une courbe d'une seule couleur. (Par défaut, deux courbes consécutives sont tracées dans des couleurs différentes).*

[2.a] Si y est une fonction développable en série entière, alors il existe un réel $r > 0$ et une suite $(a_n)_{n\in\mathbb{N}}$ tels que

$$y(x) = \sum_{n=0}^{+\infty} a_n x^n, \quad xy'(x) = \sum_{n=0}^{+\infty} n a_n x^n, \quad x^2 y''(x) = \sum_{n=0}^{+\infty} n(n-1) a_n x^n, \quad y''(x) = \sum_{n=0}^{+\infty} (n+2)(n+1) a_{n+2} x^n$$

pour tout $x \in \,]-r, r[$ car la fonction y est alors de classe $\mathscr{C}^\infty$ et ses dérivées successives peuvent être calculées en dérivant terme à terme.

❧ On en déduit (après quelques simplifications) que

$$\forall\, x \in]-r,r[, \quad (1+x^2)y''(x) + xy'(x) - \frac{1}{4}y(x) = \sum_{n=0}^{+\infty}\Big[(n+2)(n+1)a_{n+2} + \frac{(2n+1)(2n-1)}{4}\,a_n\Big]x^n$$

et comme le rayon de convergence est strictement, on en déduit que y est solution de (E) sur $]-r,r[$ si, et seulement si,

$$\forall\, n \in \mathbb{N}, \qquad (n+2)(n+1)a_{n+2} + \frac{(2n+1)(2n-1)}{4}\,a_n$$

(Unicité du développement en série entière).

❧ Ainsi, la somme de la série entière $\sum a_n x^n$ est solution de (E) si, et seulement si, le rayon de convergence de la série entière est strictement positif et si ses coefficients vérifient

$$\forall\, n \in \mathbb{N}, \qquad a_{n+2} = -\frac{(2n+1)(2n-1)}{4(n+2)(n+1)}a_n.$$

❧ Une telle relation de récurrence nous conduit à discuter sur la parité des indices.

❧ Les coefficients de rang pair sont liés par la relation

$$\forall\, k \in \mathbb{N}, \qquad a_{2k+2} = -\frac{(4k+1)(4k-1)}{4(2k+2)(2k+1)}a_{2k} \tag{$\star$}$$

et entièrement déterminés par la donnée de a_0 : la suite $(\alpha_{2k})_{k\in\mathbb{N}}$ définie par $\alpha_0 = 1$ et

$$\forall\, k \in \mathbb{N}^*, \qquad \alpha_{2k} = (-1)^k \prod_{\ell=1}^{k} \frac{(4k+1)(4k-1)}{(4k+4)(4k+2)}$$

nous donne $a_{2k} = \alpha_{2k}\cdot a_0$ pour tout $k \in \mathbb{N}$.
Les coefficients de rang impair sont liés par la relation

$$\forall\, k \in \mathbb{N}, \qquad a_{2k+3} = -\frac{(4k+3)(4k+1)}{4(2k+3)(2k+2)}a_{2k+1} \tag{$\star\star$}$$

et entièrement déterminés par la donnée de a_1 : la suite $(\alpha_{2k+1})_{k\in\mathbb{N}}$ définie par $\alpha_1 = 1$ et

$$\forall\, k \in \mathbb{N}^*, \qquad \alpha_{2k+1} = (-1)^k \prod_{\ell=1}^{k} \frac{(4\ell+3)(4\ell+1)}{(4\ell+6)(4\ell+4)}$$

nous donne $a_{2k+1} = \alpha_{2k+1}\cdot a_1$ pour tout $k \in \mathbb{N}$.

❧ Si $a_0 \neq 0$, alors les termes de rang pair sont tous différents de 0 et

$$\lim_{n\to+\infty} \Big|\frac{a_{2k+2}x^{2k+2}}{a_{2k}x^{2k}}\Big| = x^2.$$

Par conséquent, la série $\sum a_{2k}x^{2k}$ converge absolument pour $|x|<1$ et diverge grossièrement pour $|x|>1$ (règle de d'Alembert).

❧ Si $a_1 \neq 0$, alors les termes de rang impair sont tous différents de 0 et

$$\lim_{n\to+\infty} \Big|\frac{a_{2k+3}x^{2k+3}}{a_{2k+1}x^{2k+1}}\Big| = x^2.$$

Par conséquent, la série $\sum a_{2k+1}x^{2k+1}$ converge absolument pour $|x|<1$ et diverge grossièrement pour $|x|>1$.

❧ Finalement, les solutions de (E) qui sont développables en série entière sont les fonctions de la forme

$$y = a_0 S_0 + a_1 S_1 \qquad \text{où} \qquad S_0(x) = \sum_{k=0}^{+\infty}\alpha_{2k}x^{2k} \quad \text{et} \quad S_1(x) = \sum_{k=0}^{+\infty}\alpha_{2k+1}x^{2k+1}.$$

[2.b] La théorie de Cauchy-Lipschitz nous assure que l'ensemble des solutions de (E) est un plan vectoriel.
Les calculs précédents nous ont montré que l'ensemble des solutions de (E) qui sont développables en série entière est le sous-espace engendré par les fonctions S_0 et S_1. Comme ces deux fonctions ne sont pas identiquement nulles ($\alpha_0 = \alpha_1 = 1$), que S_0 est paire et que S_1 est impaire, ces deux fonctions ne sont pas proportionnelles. L'ensemble des solutions développables en série entière est donc un plan.
Par conséquent, toute solution de (E) est développable en série entière au voisinage de 0 (avec un rayon de convergence égale à 1 — sauf pour la solution nulle, bien entendu).
[2.c] Puisque S_0 est paire et que S_1 est impaire, on a

$$S_0(0) = 1, \qquad S_0'(0) = 0, \qquad S_1(0) = 0, \qquad S_1'(0) = 1$$

et l'unique solution de l'équation (E) associée au problème de Cauchy $(x_0, y_0, v_0) = (0, \sqrt{3}, 0)$ s'exprime donc

$$\forall\, x \in]-1,1[, \qquad y(x) = y_0 \cdot S_0(x) + v_0 \cdot S_1(x) = \sqrt{3} \cdot \sum_{k=0}^{+\infty} \alpha_{2k} x^{2k}.$$

❧ Pour tracer le graphe de cette solution, il suffit de savoir calculer une valeur approchée assez précise de la somme de cette série entière.
Les relations $(\star)$ et $(\star\star)$ montrent que les séries entières $\sum \alpha_{2k} x^{2k}$ et $\sum \alpha_{2k+1} x^{2k+1}$ vérifient les hypothèses du Critère spécial des séries alternées pour tout $x \in]-1,1[$.

✍ *On peut démontrer que la suite $(\alpha_n)_{n\in\mathbb{N}}$ tend vers 0 (ce n'est pas spécialement compliqué, mais ça n'a rien d'évident). On en déduit que le Critère spécial des séries alternées s'applique aussi pour $x = \pm 1$ et par conséquent les solutions de (E) sont développables en série entière sur le* segment $[-1,1]$.

❧ L'estimation du reste donnée par le Critère spécial des séries alternées nous montre qu'on aura un tracé satisfaisant en ne considérant que les 20 premiers termes de la série.

```
def DSE(T, n):
    s, puiss = np.ones_like(T), np.ones_like(T)
    coeff = 1
    for k in range(n):
        coeff *= -(4*k+1)*(4*k-1)/(4*(2*k+2)*(2*k+1))
        puiss *= T**2
        s += coeff*puiss
    return s

T = np.arange(-1, 1, 0.01)
Z = np.sqrt(3)*DSE(T, 20)
plt.plot(T, Z)
```

[3.a] Quelle que soit la fonction h de classe $\mathscr{C}^2$ sur $\mathbb{R}$, la fonction $g = h \circ \mathrm{sh}$ vérifie

$$\forall\, t \in \mathbb{R}, \quad g'(t) = \mathrm{ch}\, t \cdot h'(\mathrm{sh}\, t) \quad \text{et} \quad g''(t) = (1 + \mathrm{sh}^2 t) h''(\mathrm{sh}\, t) + \mathrm{sh}\, t h'(\mathrm{sh}\, t).$$

Comme sh réalise une bijection de $\mathbb{R}$ sur $\mathbb{R}$, on en déduit que

$$\begin{aligned} &\forall\, x \in \mathbb{R}, && (1+x^2)h''(x) + xh'(x) - \frac{h(x)}{4} = 0 \\ \iff &\forall\, t \in \mathbb{R}, && g''(t) - \frac{1}{4} g(t) = 0 \end{aligned}$$

✍ *En résolvant une équation du second degré, on trouve l'expression de la bijection réciproque :*

$$\forall\, x \in \mathbb{R}, \qquad \mathrm{Arg\,sh}\, x = \ell\mathrm{n}(x + \sqrt{x^2+1}).$$

Comme sh *est une fonction impaire, sa réciproque est également une fonction impaire — on ne peut pas dire que ça saute aux yeux.*

[3.b] Les solutions de cette équation (*à coefficients constants*) sont bien connues : il existe deux constantes réelles a_0 et a_1 telles que

$$\forall\, t \in \mathbb{R}, \qquad g(t) = a_0 \operatorname{ch} \frac{t}{2} + a_1 \operatorname{sh} \frac{t}{2}$$

et par conséquent la solution générale de (E) est de la forme

$$\forall\, x \in \mathbb{R}, \quad h(x) = a_0 \operatorname{ch} \frac{\ell n(x + \sqrt{x^2+1})}{2} + a_1 \operatorname{sh} \frac{\ell n(x + \sqrt{x^2+1})}{2}.$$

✍ *Il est bon de conserver la décomposition de la solution générale comme somme d'une fonction paire et d'une fonction impaire, ça aide à voir plus clair.*

✍ *Il est facile de vérifier qu'on n'a pas fait de faute de calcul en superposant le graphe obtenu pour $a_0 = \sqrt{3}$ et $a_1 = 0$ aux graphes précédents.*

```
Y = np.sqrt(3)*np.cosh(.5*np.log(T+np.sqrt(1+T**2)))
plt.plot(T, Y, 'b')
```

135-py-34 - Convergence uniforme

[1.a] On peut modifier légèrement la définition

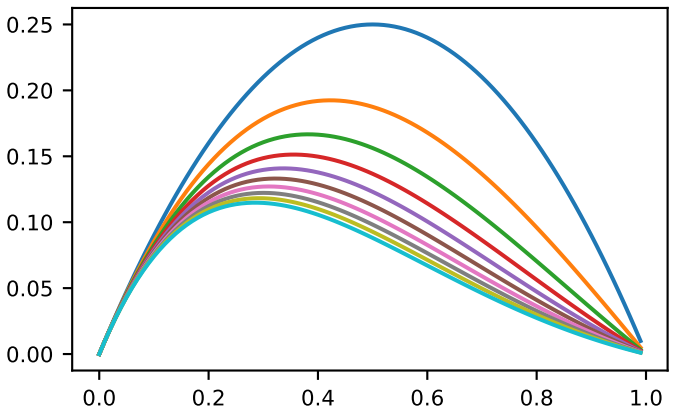
$$P_0(x) = x \qquad \text{et} \qquad \forall\, 0 \leqslant k < n, \quad P_{k+1}(x) = P_k(x) \times \left(1 - \frac{x}{k+1}\right)$$

pour que le code devienne immédiat.

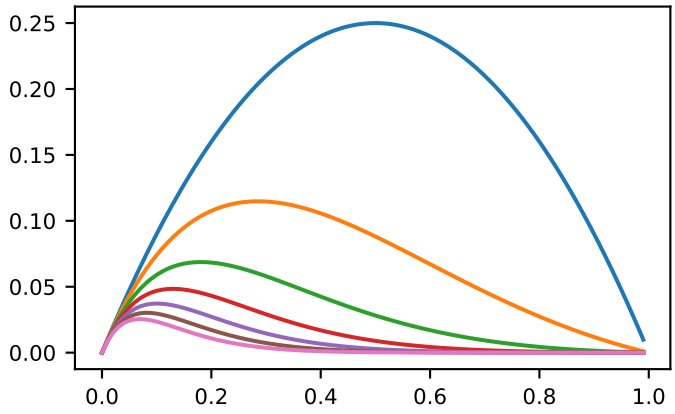
```
def P(n, x):
    y = x
    for k in range(n):
        y = y*(1-x/(k+1))
    return y
```

[1.b] Code sans difficulté, mais la figure produite n'est pas simple à interpréter.

```
X = np.arange(0, 1, 0.01)
for n in range(1, 11):
    Y = P(n, X)
    plt.plot(X, Y)
```



❧ La figure suivante montre les graphes de P_{10^k} pour $0 \leqslant k \leqslant 6$ et laisse mieux deviner que la suite $(P_n)_{n\geqslant 1}$ converge uniformément sur $[0,1]$ vers la fonction nulle.



[2.a] La fonction P_n est continue (polynomiale) sur le segment $[0,1]$, donc elle atteint un maximum.

✍ *Comme $P_n(x) \geqslant 0$ (produit de facteurs positifs), ce maximum est en fait $\|P_n\|_\infty$.*

[2.b] Le comportement des fonctions P_n étant très régulier, on peut se contenter de considérer une liste d'abscisses $(x_k)_{0\leqslant k\leqslant N}$ en progression arithmétique de raison ε et de chercher pour quel indice k la valeur de $P_n(x_k)$ est maximale.

```
def Max(n, eps):
    X = np.arange(0, 1, eps/2)
    Y = P(n, X)
    x0, y0 = X[0], Y[0]
    for (x,y) in zip(X, Y):
        if y>y0:
            x0, y0 = x, y
    return x0
```

[2.c] On choisit $\varepsilon = 5.10^{-3}$ pour avoir une précision satisfaisante.

```
eps = 0.005
N = [10**i for i in range(1, 6)]
L = [Max(n, eps)*np.log(n) for n in N]
```

Le code précédent nous donne : `[0.656, 0.829, 0.898, 0.921, 0.950]` ce qui suggère que $x_n \ln n$ tend vers 1 lorsque n tend vers $+\infty$.

[3.a] La fonction P_n est nulle en $x = 0$ et en $x = 1$ (quel que soit $n \geqslant 2$). Sur l'intervalle ouvert $]0,1[$, la fonction P_n est strictement positive (produit de facteurs strictement positifs), donc la fonction P_n atteint son maximum exactement au point où la fonction $\ln P_n$ atteint son maximum.

✍ *La fonction $\ln$ est strictement croissante sur $]0,+\infty[$.*

❧ Calculons la dérivée logarithmique de P_n :

$$\forall\, 0<x<1, \qquad \frac{P_n'(x)}{P_n(x)} = \frac{1}{x} - \sum_{k=1}^{n} \frac{1}{k}\cdot\frac{1}{1-x/k} = \frac{1}{x} - \sum_{k=1}^{n}\frac{1}{k-x}.$$

Comme $P_n(0) = P_n(1) = 0$, alors P_n est maximal en $x_n \in]0,1[$ et comme P_n est de classe $\mathscr{C}^1$, il faut que $P_n'(x_n) = 0$. On en déduit que

$$\frac{1}{x_n} = \sum_{k=1}^{n}\frac{1}{k-x_n} \quad \text{c'est-à-dire} \quad \frac{1}{x_n} = \frac{1}{1-x_n} + \sum_{k=2}^{n}\frac{1}{k-x_n}.$$

❧ Comme $0 < x_n < 1$, on a $k-1 < k-x_n < k$ pour tout $k \geqslant 1$, donc

$$\frac{1}{x_n} > \sum_{k=1}^{n} \frac{1}{k} \quad \text{et aussi} \quad \frac{1}{x_n} < \frac{1}{1-x_n} + \sum_{k=2}^{n} \frac{1}{k-1} = \frac{1}{1-x_n} + \sum_{k=1}^{n-1} \frac{1}{k}.$$

Ainsi, en notant comme d'habitude H_n la n-ième somme partielle de la série harmonique,

$$\forall\, n \geqslant 2, \qquad 0 \leqslant \frac{1}{x_n} - H_n \leqslant \frac{x_n}{1-x_n}.$$

[3.b] On sait que H_n tend vers $+\infty$, donc la minoration montre que x_n tend vers 0 lorsque n tend vers $+\infty$. Par conséquent, le majorant tend vers 0 et l'encadrement précédent montre que

$$\frac{1}{x_n} - H_n \underset{n\to+\infty}{=} o(1)$$

et on en déduit que

$$\frac{1}{x_n} \underset{n\to+\infty}{=} H_n + o(1) \underset{n\to+\infty}{\sim} \ln n.$$

[3.c] Pour tout entier $n \geqslant 2$ et tout $x \in [0,1]$,

$$0 \leqslant P_n(x) = x(1-x)\prod_{k=2}^{n}\left(1-\frac{x}{k}\right) \leqslant x(1-x)$$

(puisque les facteurs sont compris entre 0 et 1 pour tout $k \geqslant 2$). Par conséquent,

$$\forall\, n \geqslant 2, \qquad 0 \leqslant \|P_n\|_\infty = P_n(x_n) \leqslant x_n(1-x_n)$$

et comme x_n tend vers 0, on en déduit que $\|P_n\|_\infty$ tend vers 0.
On a ainsi démontré que la suite $(P_n)_{n\geqslant 1}$ convergeait uniformément sur $[0,1]$ vers la fonction nulle.

135-py-35 - Produit eulérien du sinus

[1.] Par définition, $u_1 = 3/4$ et

$$\forall\, n \geqslant 2, \qquad u_n = u_{n-1} \times \left(1 - \frac{1}{4n^2}\right).$$

Un code récursif s'en déduit immédiatement et suffit pour calculer les premiers termes de la suite.

```
def u(n):
    if n==1:
        return 3/4
    else:
        return (1-1/(4*n**2))*u(n-1)

for n in range(1, 11):
    print(u(n))
```

❧ La récursivité est très limitée en Python et il n'est pas possible d'utiliser un code récursif pour calculer les valeurs de u_n avec $n > 10^3$.

```
def u_iter(n):
    un = 3/4
    for k in range(2, n+1):
        un *= (1-1/(4*k**2))
    return un

for k in range(1, 5):
    print(u_iter(10**k))
```

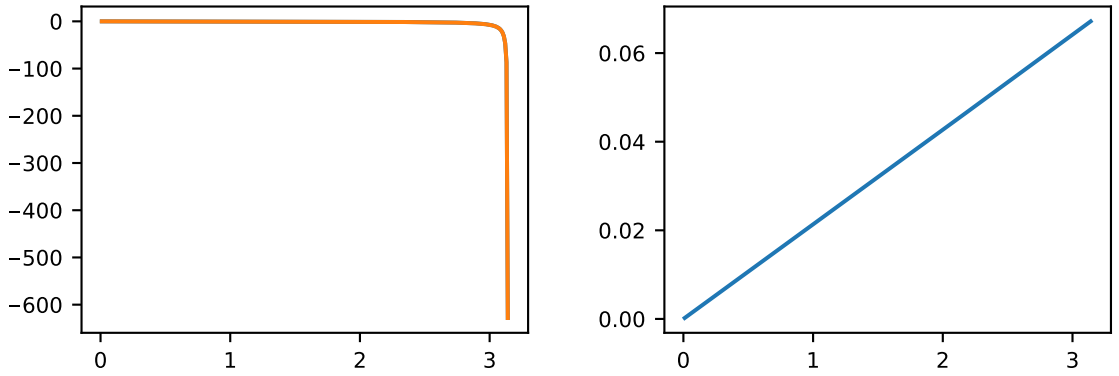
Il semble que la suite $(u_n)_{n\in\mathbb{N}}$ converge (mais lentement).

[2.a] Le code est sans problème.

```
T = np.arange(0.01, np.pi, 0.01)
plt.plot(T, f(10, T))
plt.plot(T, g(T))
```

Mais on n'y voit pas grand'chose (figure de gauche) ! Il vaudrait mieux tracer la différence entre les deux fonctions (figure de droite).

```
plt.figure(figsize=(3., 2.25))
plt.plot(T, f(10, T)-g(T))
```



On peut conjecturer que la différence entre f et g est nulle.

✍ *On n'a pas calculé les valeurs de f, mais seulement des sommes partielles et sans aucune subtilité. Il est donc tout à fait naturel que la différence entre f et g augmente légèrement avec t : c'est l'erreur de calcul sur f qui se manifeste ainsi.*

[2.b] Par définition,

$$g(t) = \frac{t\cos t - \sin t}{t\sin t} \underset{t\to 0^+}{\sim} \frac{t\cos t - \sin t}{t^2}.$$

Lorsque t tend vers 0,

$$t\cos t - \sin t \underset{t\to 0}{=} \left(t - \frac{t^3}{2} + o(t^3)\right) - \sin t = \left(\frac{1}{6} - \frac{1}{2}\right)t^3 + o(t^3)$$

donc $g(t) \sim -t/3$ au voisinage de 0. La fonction g tend donc vers 0 au voisinage de 0.

[2.c] Il est clair que g est continue sur l'intervalle ouvert $]0,\pi[$ et on vient de voir que g tendait vers 0 au voisinage de l'origine. En posant $g(0) = 0$, on définit donc un prolongement de g qui est continu sur $[0,\pi[$.

❧ Pour tout entier $n \geqslant 1$, on pose

$$\forall\, t \in [0,\pi[\,, \quad \varphi_n(t) = \frac{2t}{t^2 - n^2\pi^2}.$$

Il est clair que φ_1 est continue sur $[0,\pi[$ et que φ_n est prolongeable en une fonction continue sur le segment $[0,\pi]$ pour tout $n \geqslant 2$.

❧ Pour tout entier $n \geqslant 2$ et tout réel $t \in [0,\pi]$,

$$|\varphi_n(t)| = \frac{2t}{n^2\pi^2 - t^2} \leqslant \frac{2\pi}{(n^2-1)\pi^2}.$$

On a trouvé un majorant indépendant de t et et ce majorant est $\mathcal{O}(1/n^2)$ lorsque n tend vers $+\infty$, donc la série $\sum_{n\geqslant 2}\varphi_n$ converge normalement sur $[0,\pi]$.
La somme de la série $\sum_{n\geqslant 1}\varphi_n$ est donc continue sur $[0,\pi[$ et elle coïncide avec f sur l'intervalle ouvert $]0,\pi[$. On a ainsi démontré que f admettait un prolongement continu sur $[0,\pi[$.

✍ *On a admis que $f(t) = g(t)$ pour $0 < t < \pi$. Ayant prolongé les deux fonctions par continuité en 0, on en déduit que $f(t) = g(t)$ pour tout $0 \leqslant t < \pi$.*

[3.] Comme g est continue sur $[0,\pi[$, elle admet des primitives sur cet intervalle et la fonction G définie par

$$\forall\, x \in [0,\pi[\,, \qquad G(x) = \int_0^x g(t)\,\mathrm{d}t$$

est la primitive qui est nulle en $x = 0$.

❧ Il est clair que $\ell\mathrm{n}\sin x - \ell\mathrm{n}\, x = \ell\mathrm{n}\,\frac{\sin x}{x}$ est une primitive de g sur l'intervalle ouvert $]0,\pi[$. Comme cette expression tend vers 0 au voisinage de 0, on en déduit que

$$\forall\, x \in [0,\pi[\,, \qquad G(x) = \ell\mathrm{n}\,\frac{\sin x}{x}.$$

✍ *Une primitive sur $[0,\pi[$ est en particulier une fonction continue en 0. Par conséquent, l'unique primitive qui est nulle en 0 est aussi l'unique primitive qui tend vers 0 en 0.*

❧ En **admettant** que $f = g$ sur $[0,\pi[$,

$$\ell\mathrm{n}\,\frac{\sin x}{x} = \int_0^x f(t)\,\mathrm{d}t = \int_0^x \frac{2t}{t^2 - x^2}\,\mathrm{d}t + \int_0^x \sum_{n=2}^{+\infty} \varphi_n(t)\,\mathrm{d}t.$$

On a démontré que la série de fonctions continues $\sum_{n\geqslant 2}\varphi_n$ convergeait normalement sur $[0,\pi]$. Par conséquent,

$$\forall\, x \in [0,\pi], \quad \int_0^x \sum_{n=2}^{+\infty} \varphi_n(t)\,\mathrm{d}t = \sum_{n=2}^{+\infty} \int_0^x \varphi_n(t)\,\mathrm{d}t = \sum_{n=2}^{+\infty} \ell\mathrm{n}\,\frac{x^2 - n^2\pi^2}{-n^2\pi^2}$$

✍ *L'égalité*

$$\int_0^x \varphi_1(t)\,\mathrm{d}t = \ell\mathrm{n}\,\frac{x^2-\pi^2}{-\pi^2} = \ell\mathrm{n}\Big(1 - \frac{x^2}{\pi^2}\Big)$$

n'est pour sa part vraie que pour $0 \leqslant x < \pi$.

❧ On a donc démontré que

$$\forall\, 0 < x < \pi, \qquad \ell\mathrm{n}\,\frac{\sin x}{x} = \sum_{n=1}^{+\infty} \ell\mathrm{n}\Big(1 - \frac{x^2}{n^2\pi^2}\Big)$$

ou, si on préfère,

$$\forall\, 0 < x < \pi, \qquad \frac{\sin x}{x} = \prod_{n=1}^{+\infty} \Big(1 - \frac{x^2}{n^2\pi^2}\Big).$$

✍ *Par continuité de la fonction* exp*, on en déduit que*

$$\lim_{n\to+\infty} u_n = \exp\Big[\sum_{n=1}^{+\infty} \ell\mathrm{n}\Big(\Big(1 - \frac{(\pi/2)^2}{n^2\pi^2}\Big)\Big] = \frac{\sin(\pi/2)}{\pi/2} = \frac{2}{\pi}.$$

[4.] La fonction f est continue sur $[0,\pi[$ (depuis qu'on l'a prolongée par continuité en 0) et elle est de signe constant (négative, en tant que somme de fonctions négatives), donc elle est intégrable sur $]0,\pi[$ si, et seulement si, l'intégrale

$$\int_0^x f(t)\,\mathrm{d}t = \ell\mathrm{n}\,\frac{\sin x}{x}$$

tend vers une limite finie lorsque x tend vers π. Ce n'est pas le cas, donc f n'est pas intégrable sur $]0,\pi[$.

✍ ***Variante.***
On a décomposé f en somme de deux fonctions :

$$f(t) = \frac{2t}{(t-\pi)(t+\pi} + S(t) \qquad \text{où} \qquad S(t) = \sum_{n=2}^{+\infty} \varphi_n(t).$$

La fonction S est continue sur le segment $[0,\pi]$ car, pour tout $n \geqslant 2$, les fonctions φ_n sont continues sur $[0,\pi]$ et la série $\sum_{n\geqslant 2} \varphi_n$ converge normalement sur $[0,\pi]$.
Ainsi, la fonction f est intégrable sur $]0,\pi[$ si, et seulement si, la fonction φ_1 est intégrable sur $]0,\pi[$. Or

$$\varphi_1(t) \underset{t\to\pi}{\sim} \frac{1}{t-\pi}$$

donc φ_1 n'est pas intégrable au voisinage de π et f ne l'est pas non plus.

135-py-38 - Développement eulérien

[1.a] Pour tout $n \in \mathbb{Z}$ et tout $x \in \mathbb{R}\setminus\mathbb{Z}$, on pose

$$u_n(x) = \frac{1}{(n-x)^2}.$$

On remarque tout de suite que

$$\forall\, n \in \mathbb{Z},\ \forall\, x \in \mathbb{R}\setminus\mathbb{Z}, \qquad u_{-n}(x) = u_n(-x) \tag{†}$$
$$\text{et que} \quad u_n(x+1) = u_{n-1}(x). \tag{‡}$$

❧ Si $x \in \mathbb{R}\setminus\mathbb{Z}$, alors $(n-x)^2 > 0$ pour tout $n \in \mathbb{Z}$, donc tous les $u_n(x)$ sont bien définis (pas de division par zéro).
❧ De plus, la série de Riemann $\sum 1/n^2$ est absolument convergente et

$$\forall\, x \in \mathbb{R}\setminus\mathbb{Z}, \qquad u_n(x) \underset{n\to+\infty}{\sim} \frac{1}{n^2}$$

donc la série $\sum_{n\geqslant 1} u_n(x)$ est absolument convergente.
❧ Avec (†) et le fait que $-x \in \mathbb{R}\setminus\mathbb{Z}$, on en déduit que la série $\sum_{n\leqslant 0} u_n(x)$ est aussi absolument convergente et donc que $f(x)$ est bien définie (en tant que somme des sommes de deux séries absolument convergentes).

✍ *La définition présente $f(x)$ comme la somme de la famille $\big((n-x)^2\big)_{n\in\mathbb{Z}}$, il s'agissait donc ici de justifier que cette famille était* ***sommable****.*
On s'est appuyé ici sur le résultat suivant : si $I = I_1 \cup I_2$, alors la famille $(u_i)_{i\in I}$ est sommable si, et seulement si, les deux sous-familles $(u_i)_{i\in I_1}$ et $(u_i)_{i\in I_2}$ sont sommables (sommation par paquets, *version élémentaire du Théorème de Fubini).*
❧ *Le même raisonnement avec (†) et le fait que $[n \mapsto -n]$ soit une bijection de $\mathbb{Z}$ sur $\mathbb{Z}$ montre que f est une fonction paire.*

[1.b] Si $x+1 \in \mathbb{Z}$, alors $x = (x+1)-1 \in \mathbb{Z}$. Par contraposée : si $x \in \mathbb{R}\setminus\mathbb{Z}$, alors $x+1 \in \mathbb{R}\setminus\mathbb{Z}$.
❧ Sachant que la famille $\big(u_n(x)\big)_{n\in\mathbb{Z}}$ est sommable et que l'application $[n \mapsto n-1]$ est une bijection de $\mathbb{Z}$ sur $\mathbb{Z}$, on conclut que la famille $\big(u_{n-1}(x)\big)_{n\in\mathbb{Z}}$ est sommable et que

$$f(x) = \sum_{n\in\mathbb{Z}} u_n(x) = \sum_{n\in\mathbb{Z}} u_{n-1}(x) \overset{(‡)}{=} \sum_{n\in\mathbb{Z}} u_n(x+1) = f(x+1)$$

[1.c] Par périodicité, il suffit de démontrer que f est continue sur l'intervalle ouvert $]0,1[$.
❧ Comme la continuité est une propriété locale et que

$$]0,1[= \bigcup_{0<a<1/2} [a, 1-a],$$

il suffit de démontrer que f est continue sur tout segment $[a,1-a]$ pour tout $0 < a < 1/2$.

❧ Soit $0 < a < 1/2$. Pour tout $x \in [a, 1-a]$, on a

$$a \leqslant 1-x \leqslant 1-a \qquad \text{et} \qquad \forall\, n \in \mathbb{Z}, \quad n+a-1 \leqslant n-x \leqslant n-a.$$

Si $n \geqslant 1$, alors $n+a-1 = (n-1)+a > 0$ et si $n \leqslant 0$, alors $n-a \leqslant -a < 0$, donc

$$\forall\, n \geqslant 1, \quad 0 \leqslant u_n(x) \leqslant \frac{1}{(n+a-1)^2} \qquad \text{et} \qquad \forall\, n \leqslant 0, \quad 0 \leqslant u_n(x) \leqslant \frac{1}{(n-a)^2}.$$

Tous les majorants trouvés sont indépendants de x et les deux familles

$$\left(\frac{1}{(n+a-1)^2}\right)_{n\geqslant 1} \qquad \left(\frac{1}{(n-a)^2}\right)_{n\leqslant 0}$$

sont sommables, donc on a prouvé que la convergence était normale sur $[a, 1-a]$.

❧ En tant que somme des sommes de deux séries de fonctions continues qui convergent normalement sur $[a, 1-a]$, la fonction f est continue sur $[a, 1-a]$, quel que soit $0 < a < 1/2$, et donc continue sur $\mathbb{R} \setminus \mathbb{Z}$.

[1.d] On sait que le reste d'ordre n de la série de Riemann $\sum 1/k^2$ est équivalent à $1/n$. L'encadrement obtenu à la question précédente nous assure donc que l'approximation

$$f(x) \approx \sum_{n=-N}^{N} u_n(x)$$

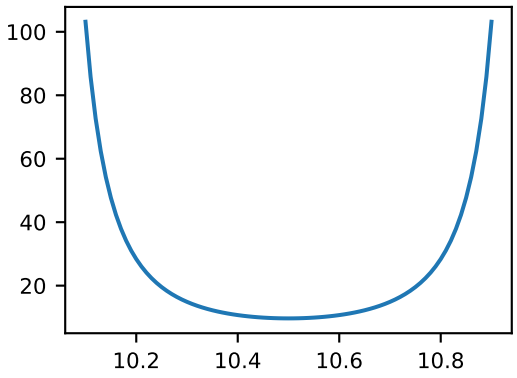
conduit à une erreur sur $f(x)$ qui est de l'ordre de $1/N$.
En choisissant $N = 10$, on a une précision largement suffisante pour tracer une courbe à l'écran.

❧ La périodicité de f nous assure que

$$\forall\, x \in \mathbb{R} \setminus \mathbb{Z}, \qquad f(x) = f(x - \lfloor x \rfloor).$$

En évaluant toujours f sur l'intervalle $]0, 1[$, on n'a pas à faire varier en fonction de x les indices n retenus pour calculer la somme partielle.

```
def f(x, N):
    u = x - np.floor(x)
    y = np.zeros_like(u)
    for n in range(-N, N+1):
        y += (u-n)**(-2)
    return y

X = np.arange(10.1, 10.9, 0.01)
Y = f(X, 10)
plt.plot(X, Y)
```



[2.] La fonction g est, comme la fonction f, définie sur $\mathbb{R} \setminus \mathbb{Z}$, continue sur son ensemble de définition, paire et périodique, de période 1. Par conséquent, il suffit de vérifier que g tend vers une limite finie en 0 pour en déduire que g est prolongeable en une fonction continue sur $\mathbb{R}$.

❧ Pour x non nul proche de 0,

$$\frac{\pi^2}{\sin^2 \pi x} = \pi^2\Big(\pi x - \frac{\pi^3 x^3}{6} + o(x^3)\Big)^{-2} = \frac{\pi^2}{\pi^2 x^2}\Big(1 - \frac{\pi^3 x^2}{6} + o(x^2)\Big)^{-2} = \frac{1}{x^2} + \frac{\pi^2}{3} + o(1).$$

❧ Reprenons l'étude de la continuité de f. En remarquant que

$$\forall\, x \in \mathbb{R}\setminus\mathbb{Z}, \qquad f(x) = \frac{1}{x^2} + \sum_{n=1}^{+\infty}\Big(\frac{1}{(n-x)^2} + \frac{1}{(-n-x)^2}\Big) = \frac{1}{x^2} + \sum_{n=1}^{+\infty}\Big(\frac{1}{(n-x)^2} + \frac{1}{(n+x)^2}\Big),$$

on est conduit à considérer la série de fonctions $\sum v_n$ où

$$\forall\, n \in \mathbb{N}^*,\ \forall\, x \in \mathbb{R}\setminus\mathbb{Z}, \qquad v_n(x) = \frac{1}{(n-x)^2} + \frac{1}{(n+x)^2}.$$

Il est clair que

$$\forall\, n \in \mathbb{N}^*,\ \forall\, 0 < a < 1,\ \forall\, x \in [-a,a], \quad 0 \leqslant v_n(x) \leqslant \frac{1}{n^2} + \frac{1}{(n-a)^2}.$$

Le majorant est indépendant de x et c'est le terme général d'une série convergente, donc la série de fonctions $\sum v_n$ converge normalement sur $[-a,a]$ et comme les fonctions v_n sont toutes continues sur $[-a,a]$, on en déduit que la somme de cette série est une fonction continue sur $[-a,a]$.

❧ En particulier, cela prouve que

$$f(x) \underset{x\to 0}{=} \frac{1}{x^2} + \sum_{n=1}^{+\infty} v_n(0) + o(1)$$

et donc que

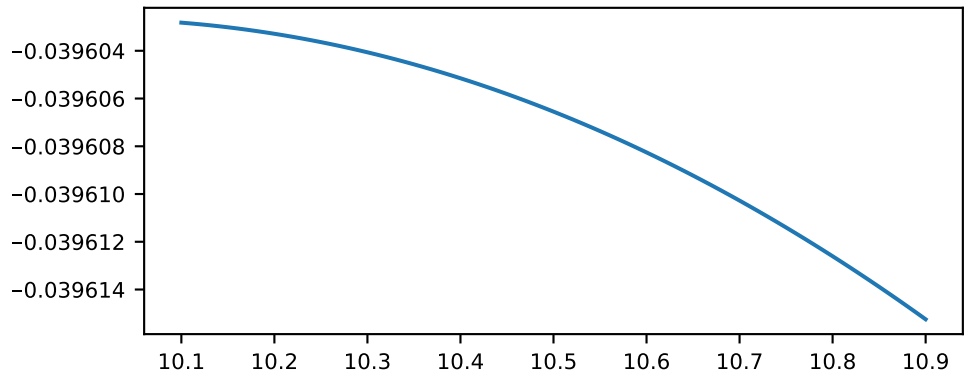
$$\lim_{x\to 0} g(x) = \sum_{n=1}^{+\infty} v_n(0) - \frac{\pi^2}{3}.$$

✍ *Si on sait que $\zeta(2) = \pi^2/6$, alors on peut en déduire que g tend vers 0 au voisinage de l'origine.*
Mais si on sait démontrer (cf question suivante) que g est prolongeable en une fonction identiquement nulle sur $\mathbb{R}$, alors on a démontré aussi que $\zeta(2) = \pi^2/6$.

[3.] Le tracé de g ne pose aucune difficulté.

```
def g(x, n):
    return f(x, n)-np.pi**2/np.sin(np.pi*x)**2
plt.plot(X, g(X, 50))
```

❧ Il faut ensuite prendre soin de comparer les légendes du graphe ainsi produit avec le graphe de f obtenu précédemment.



Les valeurs prises par g sont extrèmement faibles par rapport aux valeurs prises par f. On peut donc conjecturer que g est en fait identiquement nulle et donc que

$$\forall\, x \in \mathbb{R} \setminus \mathbb{Z}, \quad f(x) = \frac{\pi^2}{\sin^2 \pi x}.$$

135-py-43 - Inégalité de Gronwall

[1.] L'équation différentielle considérée est une équation linéaire et homogène du second ordre. Le coefficient de y'' est constant et la fonction q est continue sur $\mathbb{R}_+$, donc le Théorème de Cauchy-Lipschitz s'applique sur l'intervalle $\mathbb{R}_+$: quelle que soit la condition initiale $(x_0, y_0, v_0) \in \mathbb{R}_+ \times \mathbb{R} \times \mathbb{R}$, cette équation admet une, et une seule, solution $f \in \mathscr{C}^2(\mathbb{R}_+, \mathbb{R})$ telle que

$$\big(f(x_0), f'(x_0)\big) = (y_0, v_0).$$

Pour (E), $x_0 = 0$, $y_0 = a$ et $v_0 = b$.

❧ Il faut résoudre sur l'intervalle $[0, v]$ (puisque la condition initiale prend $x_0 = 0$ comme "instant initial" et tronquer le résultat pour ne tracer que sur le sous-intervalle $[u, v]$.

```
from scipy.integrate import odeint
import matplotlib.pyplot as plt

def trace(a, b, u, v, q):
    if u>v:
        u, v = v, u
    def f(x, t):
        return np.array([x[1], -(1+q(t))*x[0]])
    T = np.arange(0, v, 0.1)
    X = odeint(f, np.array([a, b]), T)
    n_debut = sum(T<u)
    plt.plot(T[n_debut:], X[n_debut:,0])
```

✍ *Pour bien faire, il faudrait tester si $u < v$.*

❧ Après quelques essais, on constate qu'une sorte de régime permanent s'établit. Il semble que la solution de ce problème de Cauchy soit bornée.

[2.a] Remarquons tout d'abord que

$$\forall\, x \in \mathbb{R}_+, \quad z(x) = f(x) + \sin x \int_0^x \cos tq(t)f(t)\,\mathrm{d}t - \cos x \int_0^x \sin tq(t)f(t)\,\mathrm{d}t.$$

La fonction q est continue et la fonction f est de classe $\mathscr{C}^2$, donc les

$$\int_0^x \cos tq(t)f(t)\,\mathrm{d}t \qquad \text{et} \qquad \int_0^x \sin tq(t)f(t)\,\mathrm{d}t$$

sont de classe $\mathscr{C}^1$ (Théorème fondamental), donc z est de classe $\mathscr{C}^1$ et

$$\forall\, x \in \mathbb{R}_+, \quad z'(x) = f'(x) + \cos x \int_0^x \cos tq(t)f(t)\,\mathrm{d}t + \sin x \int_0^x \sin tq(t)f(t)\,\mathrm{d}t.$$

Il apparaît sur cette expression que z' est de classe $\mathscr{C}^1$ et que

$$\forall\, x \in \mathbb{R}_+, \quad z''(x) = f''(x) - \sin x \int_0^x \cos tq(t)f(t)\,\mathrm{d}t + \cos x \int_0^x \sin tq(t)f(t)\,\mathrm{d}t + q(x)f(x)$$

donc

$$\forall\, x \in \mathbb{R}_+, \quad z''(x) + z(x) = f''(x) + f(x) - q(x)f(x) = 0$$

d'après l'équation différentielle vérifiée par f.
[2.b] D'après les calculs qui précèdent, $z(0) = f(0) = a$ et $z'(0) = f'(0) = b$ et il existe deux réels A et B tels que

$$\forall\, x \in \mathbb{R}_+, \qquad z(x) = A\cos x + B\sin x.$$

On en déduit facilement que $A = a$ et que $B = b$, donc

$$\forall\, x \in \mathbb{R}_+, \qquad f(x) = a\cos x + b\sin x - \int_0^x \sin(x-t)q(t)f(t)\,\mathrm{d}t.$$

Comme $x \geqslant 0$, on intègre bornes croissantes et on peut déduire de l'inégalité triangulaire pour les intégrales que

$$\begin{aligned}\forall\, x \in \mathbb{R}_+, \quad \big|f(x)\big| &\leqslant |a| \times 1 + |b| \times 1 + \int_0^x \big|sin(x-t)q(t)f(t)\big|\,\mathrm{d}t \\ &\leqslant |a| + |b| + \int_0^x \big|q(t)\big|\,\big|f(t)\big|\,\mathrm{d}t.\end{aligned}$$

[2.c] Simplifions l'encadrement précédent : il existe deux fonctions continues et positives $g(t) = \big|f(t)\big|$ et $\varphi(t) = \big|q(t)\big|$ ainsi qu'une constante positive $c \in \mathbb{R}_+$ telles que

$$\forall\, x \in \mathbb{R}_+, \quad g(x) \leqslant c + \int_0^x \varphi(t)g(t)\,\mathrm{d}t.$$

Comme le produit φg est une fonction continue sur $\mathbb{R}_+$, le second membre est une fonction de classe $\mathscr{C}^1$ sur $\mathbb{R}_+$:

$$h(x) = c + \int_0^x \varphi(t)g(t)\,\mathrm{d}t$$

et

$$\forall\, x \in \mathbb{R}_+, \quad h'(x) = \varphi(x)g(x) \leqslant \varphi(x)h(x).$$

❧ Comme la fonction φ est continue sur $\mathbb{R}_+$, elle admet une primitive Φ de classe $\mathscr{C}^1$ sur $\mathbb{R}_+$ et

$$\forall\, x \in \mathbb{R}_+, \qquad h'(x) - \varphi(x)h(x) = h'(x) - \Phi'(x)h(x) \leqslant 0.$$

On en déduit que

$$\forall\, x \in \mathbb{R}_+, \qquad \big[h'(x) - \Phi'(x)h(x)\big]\exp\big(-\Phi(x)\big) \leqslant 0$$

et donc que l'expression

$$h(x)\exp\big(-\Phi(x)\big)$$

est une fonction décroissante (et positive) de x.

❧ On sait que $h(0) = c$ et on peut choisir $\Phi(0) = 0$ (une primitive est définie à une constante additive près). Dans ces conditions, on a démontré que

$$\forall\, x \in \mathbb{R}_+, \quad \big|f(x)\big| \leqslant h(x) \leqslant h(0)\exp\big(\Phi(x)\big) = c\exp\int_0^x \big|q(t)\big|\,\mathrm{d}t.$$

Comme q est intégrable sur $\mathbb{R}_+$, le second membre est majoré par

$$c\exp\int_0^{+\infty} \big|q(t)\big|\,\mathrm{d}t,$$

ce qui prouve que la fonction $|f|$ est bien bornée sur $\mathbb{R}_+$.

135-py-44 - Intégrale fonction d'un paramètre

[1.] Pour $f \in E$ fixée, posons

$$\forall\,(x,t) \in \mathbb{R}\times\mathbb{R}_+, \quad \varphi(x,t) = \mathrm{Arctan}(tx)\frac{f(t)}{1+t^2}.$$

❧ **Intégrabilité** — Pour tout $x \in \mathbb{R}$, il est clair que la fonction $[t \mapsto \varphi(x,t)]$ est continue sur l'intervalle $I = [0,+\infty[$ et comme la fonction f est bornée (par définition de E),

$$\forall\,(x,t) \in \mathbb{R}\times I, \qquad \big|\varphi(x,t)\big| \leqslant \frac{\pi\|f\|_\infty}{2(1+t^2)}. \tag{$\star$}$$

Le majorant est une fonction de t intégrable au voisinage de $+\infty$, ce qui prouve que $[t \mapsto \varphi(x,t)]$ est intégrable sur I.
La fonction $\Phi(f)$ est donc bien définie sur $\mathbb{R}$.

❧ **Régularité** — Pour tout $t \in \mathbb{R}$, il est clair que la fonction $[x \mapsto \varphi(x,t)]$ est continue sur $\Omega = \mathbb{R}$.

❧ **Domination** — L'encadrement $(\star)$ est vraie pour tout $x \in \mathbb{R}$ et tout $t \in I$; le majorant est indépendant du paramètre $x \in \mathbb{R}$ et il est intégrable sur I en tant que fonction de t.
Par conséquent, la fonction $\Phi(f)$ est continue sur $\mathbb{R}$.

❧ On déduit également de $(\star)$ que $\Phi(f)$ est bornée :

$$\forall\, x \in \mathbb{R}, \quad \big|\Phi(x)\big| \leqslant \int_0^{+\infty}\big|\varphi(x,t)\big|\,\mathrm{d}t \leqslant \frac{\pi\|f\|_\infty}{2}\cdot\int_0^{+\infty}\frac{\mathrm{d}t}{1+t^2} = \frac{\pi^2\|f\|_\infty}{4}.$$

Par conséquent, $\Phi(f) \in E$ pour toute fonction $f \in E$.

❧ On a prouvé l'intégrabilité de $[t \mapsto \varphi(x,t)]$ pour toute fonction $f \in E$ et tout paramètre $x \in \mathbb{R}$, la linéarité de Φ découle alors directement de la linéarité de l'intégrale.
Ainsi Φ est bien un endomorphisme de E.

[2.] (On reprend les notations précédentes.)

❧ **Régularité** — Il est clair que, pour tout $t \in I$, la fonction $[x \mapsto \varphi(x,t)]$ est de classe $\mathscr{C}^1$ sur $\mathbb{R}$ et

$$\forall\,(x,t) \in \mathbb{R}\times I, \quad \frac{\partial\varphi}{\partial x}(x,t) = \frac{tf(t)}{[1+(xt)^2](1+t^2)}.$$

❧ **Intégrabilité** — Comme f est bornée,

$$\forall\,(x,t) \in \mathbb{R}\times I, \quad \left|\frac{\partial\varphi}{\partial x}(x,t)\right| \leqslant \frac{t\|f\|_\infty}{[1+(xt)^2](1+t^2)}.$$

La majorant est une fonction continue de t sur I et si $x \neq 0$, cette fonction est $\mathcal{O}(1/t^3)$ au voisinage de $+\infty$, donc c'est une fonction intégrable sur I. Par comparaison, pour tout $x \in \mathbb{R}^*$, la fonction

$$\left[t \mapsto \frac{\partial\varphi}{\partial x}(x,t)\right]$$

est intégrable sur I.

✍ *Pour $x = 0$, le majorant est seulement $\mathcal{O}(1/t)$ et l'intégrabilité n'est pas garantie au voisinage de $+\infty$. (La conclusion dépend alors de la fonction f choisie.)*

❧ **Domination** — Soit $a > 0$. Par monotonie en fonction de x (pour $t \in I$ fixé),

$$\forall\, (x,t) \in [a,+\infty[\times I, \quad \left|\frac{\partial \varphi}{\partial x}(x,t)\right| \leqslant \left|\frac{\partial \varphi}{\partial x}(a,t)\right|$$

et ce majorant est intégrable sur I d'après l'encadrement précédent. Comme ce majorant est indépendant du paramètre $x \in [a,+\infty[$, on a établi la propriété de domination.

❧ D'après le Théorème de dérivation sous le signe $\int$, la fonction $\Phi(f)$ est de classe $\mathscr{C}^1$ sur l'intervalle $[a,+\infty[$ pour tout $a > 0$, donc elle est de classe $\mathscr{C}^1$ sur

$$]0,+\infty[= \bigcup_{a>0} [a,+\infty[$$

et de plus

$$\forall\, x > 0, \quad [\Phi(f)]'(x) = \int_0^{+\infty} \frac{t f(t)}{[1+(xt)^2](1+t^2)}\, \mathrm{d}t.$$

✍ *Il est clair que la fonction $\Phi(f)$ est impaire, donc elle est de classe $\mathscr{C}^1$ sur $\mathbb{R}^*$ (et l'expression de sa dérivée est encore vraie pour $x < 0$).*

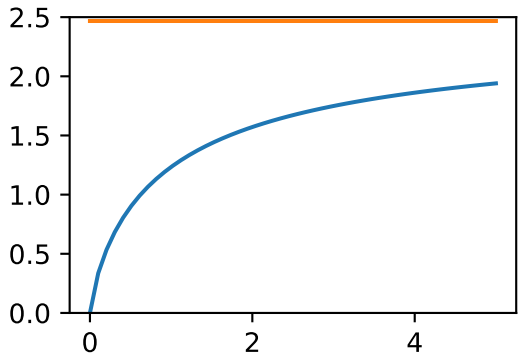
[3.a] On commence par définir la fonction g. (Il suffit de suivre les indications données dans le Vademecum.)

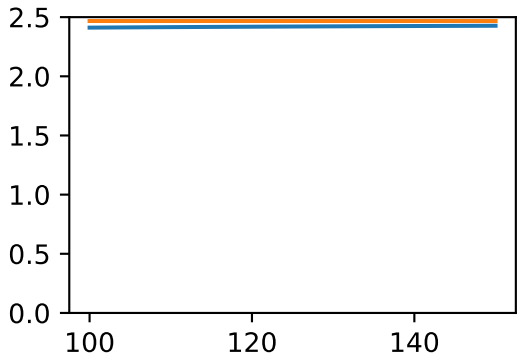
```
import scipy.integrate as integr

def g(x):
    def f(t):
        return np.arctan(t*x)/(1+t**2)
    return integr.quad(f, 0, np.inf)[0]
```

Le tracé de la courbe est ensuite classique (figure de gauche).

```
X = np.linspace(0, 5)
Y = [g(x) for x in X]
plt.plot(X, Y)
asptt = (np.pi/2)**2
plt.plot([X[0],X[-1]], [asptt,asptt])
plt.ylim(0, 2.5)
```

Le résultat apparaît plus clairement sur l'intervalle $[100, 150]$ (figure de droite).





[3.b] Il est clair que

$$\forall\, t > 0, \quad \lim_{x\to+\infty} \frac{\mathrm{Arctan}(tx)}{1+t^2} = \frac{\pi}{2(1+t^2)}.$$

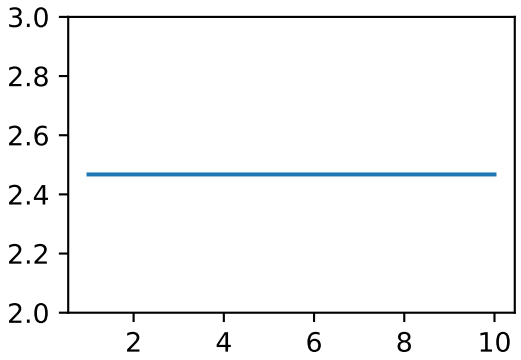
✍ *Attention, si $t = 0$, cette limite est nulle — mais c'est sans importance pour la suite.*

La domination $(\star)$ est établie au voisinage de $+\infty$ pour x (puisque $\mathbb{R}$ est en particulier un voisinage de $+\infty$), ce qui permet de passer à la limite sous le signe $\int$. Ainsi, la fonction g tend vers une limite finie au voisinage de $+\infty$ et

$$\lim_{x\to+\infty} g(x) = \int_0^{+\infty} \lim_{x\to+\infty} \frac{\operatorname{Arctan}(tx)}{1+t^2}\,\mathrm{d}t = \left(\frac{\pi}{2}\right)^2.$$

[3.c] Commencer par tracer l'allure du graphe de $G = [x \mapsto g(x) + g(1/x)]$.

```
X = np.linspace(1, 10)
Z = [g(x)+g(1/x) for x in X]
plt.plot(X, Z)
plt.ylim(2, 3)
```



Cela ressemble beaucoup à une fonction constante !

❧ On a démontré plus haut que la fonction g était de classe $\mathscr{C}^1$ sur $\mathbb{R}^*$ et que

$$\forall\, x > 0, \quad g'(x) = \int_0^{+\infty} \frac{t\,\mathrm{d}t}{[1+(xt)^2](1+t^2)}.$$

Par conséquent, la fonction G est de classe $\mathscr{C}^1$ sur $\mathbb{R}^*$ et

$$\forall\, x > 0, \quad G'(x) = g'(x) - \frac{1}{x^2} g'(1/x).$$

Avec le changement de variable usuel $u = 1/t$,

$$\begin{aligned}\frac{1}{x^2} g'(1/x) &= \int_0^{+\infty} \frac{t\,\mathrm{d}t}{x^2[1+(t/x)^2](1+t^2)} \\ &= \int_0^{+\infty} \frac{(u/u^2)\cdot(\mathrm{d}u/u^2)}{x^2[1+1/(xu)^2][1+(1/u)^2]} \\ &= \int_0^{+\infty} \frac{u\,\mathrm{d}u}{[1+(xu)^2](1+u^2)} = g'(x).\end{aligned}$$

Comme sa dérivée est nulle sur l'intervalle $]0,+\infty[$, la fonction G est constante sur $]0,+\infty[$.

✍ *La fonction g est impaire, donc la fonction G est impaire elle aussi, elle n'est donc pas constante sur $\mathbb{R}^*$:*

$$\forall\, x \in \mathbb{R}^*, \qquad G(x) = -G(-x).$$

Comme la fonction g est continue sur $\mathbb{R}$ et qu'elle tend vers $(\pi/2)^2$ au voisinage de $+\infty$, on en déduit que

$$\forall\, x > 0, \quad G(x) = \lim_{x\to+\infty} g(x) + g(1/x) = \lim_{x\to+\infty} g(x) + g(0) = \left(\frac{\pi}{2}\right)^2.$$

✍ *On pourrait aussi remarquer que*

$$\forall\, A \in \mathbb{R}_+, \quad \int_0^A \frac{\operatorname{Arctan} t}{1+t^2}\,\mathrm{d}t = \left[\frac{(\operatorname{Arctan} t)^2}{2}\right]_0^A = \frac{1}{2}(\operatorname{Arctan} A)^2$$

et en déduire la valeur de $G(x)$ en faisant tendre A vers $+\infty$.