Chapitre 7
Sommes et produits

I Le symbole somme

I.1 Définition

Soit $n \in \mathbb{N}$, soit une liste de $n+1$ nombres $(a_0, a_1, \dots, a_n)$. On définit

$$\sum_{i=0}^{n} a_i = a_0 + a_1 + \cdots + a_n \tag{1}$$

Cette définition se fait en fait naturellement par récurrence :

$$\sum_{i=0}^{n=0} a_i = a_0 \qquad \sum_{i=0}^{n+1} a_i = \left(\sum_{i=0}^{n} a_i\right) + a_{n+1} \tag{2}$$

La variable n est une variable **libre** alors que la variable i est dite **muette**.
La somme a alors les propriétés suivantes :

— Pour tout $k \in \mathbb{N}$ tel que $0 \leqslant k < n$:

$$\sum_{i=0}^{k} a_i + \sum_{i=k+1}^{n} a_i = \sum_{i=0}^{n} a_i \tag{3}$$

— Pour tout $\lambda \in \mathbb{R}$:

$$\sum_{i=0}^{n} (\lambda a_i) = \lambda \left(\sum_{i=0}^{n} a_i\right) \tag{4}$$

— Pour toute autre liste $(b_0, \dots, b_n)$:

$$\sum_{i=0}^{n} (a_i + b_i) = \left(\sum_{i=0}^{n} a_i\right) + \left(\sum_{i=0}^{n} b_i\right) \tag{5}$$

— Si on considère des sommes de nombres complexes alors on a aussi

$$\overline{\left(\sum_{k=0}^{n} a_k\right)} = \sum_{k=0}^{n} \bar{a}_k \tag{6}$$

et donc

$$\mathrm{Re}\left(\sum_{k=0}^{n} a_k\right) = \sum_{k=0}^{n} \mathrm{Re}(a_k) \qquad \mathrm{Im}\left(\sum_{k=0}^{n} a_k\right) = \sum_{k=0}^{n} \mathrm{Im}(a_k) \tag{7}$$

I.2 Quelques sommes à connaitre

… et à savoir démontrer par récurrence !

— Soit $n \in \mathbb{N}^*$:

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2} \qquad \sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6} \tag{8}$$

— Soit $q \in \mathbb{R} \setminus \{1\}$ (ou même $q \in \mathbb{C} \setminus \{1\}$), soit $n \in \mathbb{N}$:

$$\sum_{k=0}^{n} q^k = \frac{1 - q^{n+1}}{1 - q} \tag{9}$$

Exercice 1.
1. Calculer $\sum_{k=1}^{n} 2k$ et $\sum_{k=1}^{n} (2k+1)$.
2. Calculer $\sum_{k=1}^{n} \frac{1}{2^k}$.
3. Soit $\theta \in \mathbb{R}$. Calculer $\sum_{k=0}^{n} \cos(k\theta)$ et $\sum_{k=0}^{n} \sin(k\theta)$

I.3 Notion de famille indexée par un ensemble

Soit un ensemble fini I quelconque.

Définition 1. Une *famille de nombres réels indexée par I* est la donnée, pour chaque élément $i \in I$, d'un nombre réel a_i. On note la famille $(a_i)_{i \in I}$.

On note pour $a, b \in \mathbb{Z}$ l'*intervalle entier*

$$[\![a, b]\!] = \{i \in \mathbb{Z} \mid a \leqslant i \leqslant b\} \tag{10}$$

alors $(a_0, \dots, a_n) = (a_i)_{i \in [\![0,n]\!]}$.
Soit une famille de réels $(a_i)_{i \in I}$ indexée par I. Alors on peut définir une somme

$$\sum_{i \in I} a_i \tag{11}$$

et on a notammnent les relations suivantes : si $I = \emptyset$ alors

$$\sum_{i\in\emptyset} a_i = 0 \tag{12}$$

et si $I = I_1 \cup I_2$ avec $I_1 \cap I_2 = \emptyset$ alors

$$\sum_{i\in I} a_i = \left(\sum_{i\in I_1} a_i\right) + \left(\sum_{i\in I_2} a_i\right) \tag{13}$$

Exercice 2. Calculer les sommes

$$\sum_{\substack{k\in[\![1,n]\!]\\ k \text{ pair}}} k^2 \qquad \sum_{\substack{k\in[\![1,n]\!]\\ k \text{ impair}}} k^2 \qquad \sum_{k\in[\![1,2n]\!]} (-1)^{k+1}k \tag{14}$$

I.4 Sommes doubles

Une « famille double » $(a_{i,j})$ où i varie entre 1 et $n \in \mathbb{N}$ et où j varie entre 1 et $m \in \mathbb{N}$, est en fait une famille indexée par l'ensemble fini $[\![1, n]\!] \times [\![1, m]\!]$ et on a notamment l'égalité

$$\sum_{i=1}^{n}\sum_{j=1}^{m} a_{i,j} = \sum_{j=1}^{m}\sum_{i=1}^{n} a_{i,j} = \sum_{(i,j)\in[\![1,n]\!]\times[\![1,m]\!]} a_{i,j} \tag{15}$$

Exercice 3. Calculer les sommes

$$\sum_{1\leqslant i,j\leqslant n} (i+j) \qquad \sum_{1\leqslant i,j\leqslant n} ij \qquad \sum_{1\leqslant i\leqslant j\leqslant n} i \tag{16}$$

II Le symbole produit

II.1 Définition

Soit $n \in \mathbb{N}$, soit une famille de nombres réels (ou même complexes) $(a_0, \dots, a_n)$. On définit

$$\prod_{i=0}^{n} a_i = a_0 \times a_1 \times \cdots \times a_n \tag{17}$$

Les propriétés sont similaires à celles des sommes, insistons notamment sur :

— Pour tout $k \in \mathbb{N}$ tel que $0 \leqslant k < n$:

$$\left(\prod_{i=0}^{k} a_i\right) \times \left(\prod_{i=k+1}^{n} a_i\right) = \prod_{i=0}^{n} a_i \tag{18}$$

— Pour tout $\lambda \in \mathbb{R}$:

$$\prod_{i=0}^{n} (\lambda a_i) = \lambda^{n+1} \times \prod_{i=0}^{n} a_i \tag{19}$$

— Pour un ensemble fini I et pour toute famille $(a_i)_{i\in I}$ indexée par I on peut définir $\prod_{i\in I} a_i$ et

$$\prod_{i\in\emptyset} a_i = 1 \tag{20}$$

II.2 Factorielles et combinatoire

Définition 2. Pour $n \in \mathbb{N}$, la factorielle de n est le nombre $n!$ défini par $0! = 1$ et si $n > 0$

$$n! = \prod_{k=1}^{n} k \tag{21}$$

ainsi $n! = n \times (n-1)!$.

Proposition 3. *$n!$ est le nombre de permutations de n objets distincts.*

Les valeurs de $n!$ augmentent très vite :

n	0	1	2	3	4	5	6
$n!$	1	1	2	6	24	120	720

(22)

Exercice 4. Soit $n \in \mathbb{N}^*$, calculer

$$\prod_{\substack{k\in[\![1,2n]\!]\\ k \text{ pair}}} k \qquad \prod_{\substack{k\in[\![1,2n]\!]\\ k \text{ impair}}} k \tag{23}$$

Définition 4. Pour tout $n \in \mathbb{N}$ et pour tout entier k avec $0 \leqslant k \leqslant n$, on définit le *coefficient binomial* (lire « k parmi n »)

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} = \frac{n \times \cdots \times (n-k+1)}{k \times \cdots \times 1} \tag{24}$$

Proposition 5. *$\binom{n}{k}$ est le nombre de façon de choisir k objets parmi n objets donnés.*

On convient que $\binom{n}{k} = 0$ si $k > n$ ou si $k < 0$.

Proposition 6. *On a les formules de base suivantes, pour tous n, k :*

— *Valeurs remarquables :*

$$\binom{n}{0} = 1 \qquad \binom{n}{n} = 1 \qquad \binom{n}{1} = n \qquad \binom{n}{n-1} = n \tag{25}$$

— *Symétrie :*

$$\binom{n}{k} = \binom{n}{n-k} \tag{26}$$

— *Formule de Pascal :*

$$\binom{n}{k} + \binom{n}{k+1} = \binom{n+1}{k+1} \tag{27}$$

Cette dernière permet de calculer rapidement les $\binom{n}{k}$ en les rangeant dans le **triangle de Pascal** : sur la n-ième ligne on lit à la suite les coefficients $\binom{n}{k}$, et chaque coefficient est obtenu à partir de la somme des deux coefficients au-dessus de lui.

n					
0	1				
1	1	1			
2	1	2	1		
3	1	3	3	1	
4	1	4	6	4	1

(28)

II.3 La formule du binôme de Newton

Proposition 7. *Pour tous nombres $a, b \in \mathbb{R}$ (et même $a, b \in \mathbb{C}$), pour tout $n \in \mathbb{N}$:*

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k \tag{29}$$

En conséquence on a aussi :

$$(a-b)^n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} a^{n-k} b^k \tag{30}$$

Exercice 5. Calculer les sommes suivantes :

$$\sum_{k=0}^{n} \binom{n}{k} \qquad \sum_{k=0}^{n} (-1)^k \binom{n}{k} \qquad \sum_{k=0}^{n} k \binom{n}{k} \qquad \sum_{k=0}^{n} \binom{n}{k} \cos(k\theta) \quad (\theta \in \mathbb{R}) \tag{31}$$

II.4 Une autre formule

Proposition 8. *Pour tous nombres $a, b \in \mathbb{R}$ (et même $a, b \in \mathbb{C}$), pour tout $n \in \mathbb{N}$:*

$$a^n - b^n = (a-b) \times \sum_{k=0}^{n-1} a^{n-k-1} b^k \tag{32}$$

Annexe : programmes à connaitre

But : écrire un programme qui calcule la somme de la famille $(a_i)_{i \in [\![0,n]\!]}$.
Premier cas : la famille est donnée par une liste de $n+1$ éléments `A`. Bien qu'il existe une fonction Python `sum()` qui calcule la somme de tous les termes d'une liste, il faut savoir la re-programer soi-même sans hésiter. Ce morceau calcule sa somme de *toute* la liste :

```
s = 0
for x in A:
    s = s + x
```

ou bien, calculer exactement la somme de `A[0]` à `A[n]` :

```
s = 0
for i in range(n+1):
    s = s + A[i]
```

Deuxième cas : la famille est donnée par une fonction `a(i)`.

```
s = 0
for i in range(n+1):
    s = s + a(i)
```

Troisième cas : la famille est en fait donnée par une relation de récurrence du type $a_{i+1} = f(a_i)$, avec une fonction écrite en Python `f(x)`, et un premier terme a_0 donné par une variable Python `a_0` : on calcule *en même temps* la suite $(a_i)_{i \in \mathbb{N}}$ et la somme.

```
s = 0
a = a_0
for i in range(n+1):
    # ici a = a_(i)
    # et s = somme de 0 à i-1
    s = s + a
    # maintenant s = somme de 0 à i
    # puis a devient a_(i+1) :
    a = f(a)
```

Remarque 1. 1. Dans tous ces programmes il faut faire très attention aux bornes et aux indices ; écrire des commentaires comme ci-dessus est une bonne pratique pour vérifier que les indices concordent bien, et qu'en fin de boucle la variable `s` est bien la somme de $i = 0$ à $i = n$. On peut aussi ajouter des `print(i, a, s)` pour voir évoluer la suite et la somme en même temps.

2. La variable `s` s'appelle un **accumulateur** car elle accumule au fur et à mesure la quantité cherchée.
3. On peut aussi utiliser ces mêmes schémas de programme directement avec des chaines de caractères (pour concaténer), des listes, ou remplacer les sommes par des produites.