CHAP 03 Suites numériques

Compétences

- Calculer des termes d'une suite (explicite, récurrence, algorithme)
- Déterminer une relation explicite ou de récurrence
- Représenter graphiquement une suite (nuage de points, « toile »)
- Étudier le sens de variation d'une suite
- Conjecturer la limite éventuelle d'une suite, y compris l'absence de limite
- Utiliser python pour calculer un terme, un rang (seuil) ou une somme

I. Généralités sur les suites

I.1. Vocabulaire et notations

Définition 1.1

Une **suite numérique** est une fonction définie sur $\mathbb{N}$ (ou à partir d'un certain rang) et à valeurs dans $\mathbb{R}$: à chaque entier n, elle associe un unique réel noté $u(n)$ ou, le plus souvent, u_n.
Le nombre u_n s'appelle le **terme d'indice** n (ou **de rang** n). L'ensemble de tous ces termes, qui constitue la suite, se note u ou plus souvent (u_n).

Remarques 1.1

- Il ne faut pas confondre u_n, qui est un **nombre** (le terme d'indice n), et (u_n), qui désigne la **suite** tout entière.
- Il ne faut pas confondre u_{n+1} (le terme *suivant* u_n) et $u_n + 1$ (le nombre u_n augmenté de 1).
- Comme $\mathbb{N}$ a pour plus petit élément 0, on note très souvent u_0 le premier terme, mais certaines suites ne sont définies qu'à partir d'un certain rang : la suite de terme général $u_n = \dfrac{1}{n}$ n'est définie que pour $n \geqslant 1$ (son premier terme est u_1).

I.2. Modes de génération d'une suite

Définitions 1.2

Une suite peut être engendrée de plusieurs façons :

- de façon **explicite** : $u_n = f(n)$, où f est une fonction. Les termes ne dépendent que de leur rang n : on peut calculer n'importe quel terme *directement* ;
- par une relation de **récurrence** : on donne le premier terme et une relation $u_{n+1} = f(u_n)$ qui permet de passer d'un terme au suivant (la relation peut aussi faire intervenir n, par exemple $u_{n+1} = u_n + 2n + 1$). Les termes sont définis *de proche en proche* : pour calculer un terme, il faut *a priori* calculer tous les précédents ;
- par un **algorithme** qui calcule les termes de proche en proche ;
- par un **motif géométrique** : la suite donne le nombre d'éléments d'une configuration qui dépend de l'entier n.

Méthode 1.1

Pour bien comprendre et étudier une suite, il faut avant tout savoir si elle est définie de façon **explicite** ou par **récurrence** : les techniques d'étude ne sont pas les mêmes dans les deux cas.

Exemple 1.1

- La suite (v_n) définie par $v_n = 2^n - 5$ pour tout $n \in \mathbb{N}$ est **explicite**. Pour calculer un terme, seul son rang est nécessaire : $v_0 = 2^0 - 5 = 1 - 5 = -4$ et $v_1 = 2^1 - 5 = 2 - 5 = -3$ et $v_7 = 2^7 - 5 = 128 - 5 = 123$.
- La suite (u_n) définie par $\begin{cases} u_0 = 5 \\ u_{n+1} = 2u_n - 3 \quad \text{pour tout } n \in \mathbb{N} \end{cases}$ est **récurrente**. Pour passer d'un terme au suivant, on le multiplie par 2 puis on soustrait 3 :
  $u_0 = 5$ d'après l'énoncé, puis $u_1 = 2 \times 5 - 3 = 7$, puis $u_2 = 2 \times 7 - 3 = 11$, puis $u_3 = 2 \times 11 - 3 = 19$, etc.

Licence Ouverte Etalab 2.0

▸ Exemple 1.2

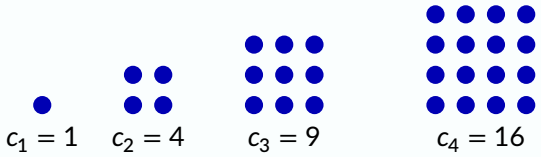
Une ville compte 20 000 habitants en 2026. Chaque année, 2 % des habitants quittent la ville et 300 nouveaux habitants s'installent. On note u_n le nombre d'habitants en $2026 + n$.
On a $u_0 = 20\,000$ et, pour passer d'une année à la suivante, on multiplie par $1 - 0{,}02 = 0{,}98$ puis on ajoute 300 :

$$u_{n+1} = 0{,}98\,u_n + 300.$$

La suite (u_n) est donc définie par récurrence, et on a par exemple : $\begin{cases} u_1 = 0{,}98 \times 20\,000 + 300 = 19\,900 \\ u_2 = 0{,}98 \times 19\,900 + 300 = 19\,802 \end{cases}$.

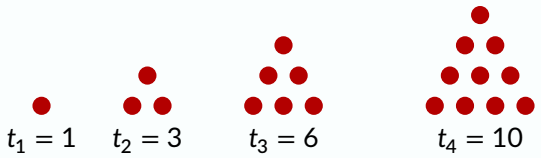
▸ Exemple 1.3

On dispose des points en carrés : c_n désigne le nombre de points du n-ième motif.



On reconnaît le carré d'un entier, donc la formule **explicite** $c_n = n^2$. En passant du motif n au motif $n+1$, on ajoute une « équerre » de $n + 1 + n$ points, donc la relation de **récurrence** $c_{n+1} = c_n + 2n + 1$ avec $c_1 = 1$.

Avec des points en triangles, t_n désigne le nombre de points du n-ième motif :



Pour passer du motif n au motif $n+1$, on ajoute une rangée de $n+1$ points : $t_{n+1} = t_n + (n+1)$ avec $t_1 = 1$. (Il existe aussi une formule explicite, que l'on démontrera plus tard.)

▸ Exercice 1.1

1. Soit (a_n) la suite définie pour tout $n \in \mathbb{N}$ par $a_n = n^2 - 3$. Calculer a_0, a_5 et a_{10}.
2. Soit (w_n) la suite définie par $w_0 = 1$ et $w_{n+1} = 3w_n - 2$ pour tout $n \in \mathbb{N}$. Calculer w_1, w_2 et w_3. Peut-on calculer w_{10} sans calculer les termes précédents ?
3. À l'aide de la relation de récurrence trouvée pour les motifs triangulaires, calculer t_5 et t_6.

I.3. Calculer des termes avec des outils numériques

▸ Calculatrice 1.1

À la calculatrice, la touche `Ans` permet de calculer les termes consécutifs d'une suite récurrente : on saisit le premier terme, puis la formule de récurrence en remplaçant u_n par `Ans` ; chaque validation donne le terme suivant. Les calculatrices disposent aussi d'un module `Suites` (tableau de valeurs, nuage de points).

▸ Logiciel 1.1

Dans un tableur, on utilise les références de cellules puis la poignée de recopie :

— suite **explicite** : on écrit la formule en fonction du rang (cellule A2), puis on recopie vers le bas ;

— suite **récurrente** : on saisit le terme initial, puis la formule en fonction de la cellule du dessus, que l'on recopie vers le bas.

	A	B
1	n	$v(n)$
2	0	`=2^A2-5`
3	1	…

	A	B
1	n	$u(n)$
2	0	5
3	1	`=2*B2-3`

▸ Algorithme 1.1

On peut aussi utiliser un algorithme, par exemple en Python, pour calculer le terme de rang n d'une suite récurrente. Ici, pour la suite définie par $u_0 = 5$ et $u_{n+1} = 2u_n - 3$:

</> Code Python

```
def terme_un(n):
    u = 5                       # premier terme de la suite : u_0
    for i in range(1, n+1):     # i va de 1 à n
        u = 2*u - 3             # formule de récurrence
    return u
```

L'appel `terme\protect \TU\textunderscore un(3)` renvoie `19`. (Les boucles sont détaillées dans la dernière partie du chapitre.)

II. Représentation graphique

II.1. Cas général : nuage de points

▸ Méthode 2.1

Pour représenter graphiquement une suite (u_n), on dessine le **nuage de points** de coordonnées $(n\,;\,u_n)$ pour n entier naturel : en **abscisse** les **rangs** n (entiers positifs) et en **ordonnée** les **termes** u_n.

▸ Remarque 2.1

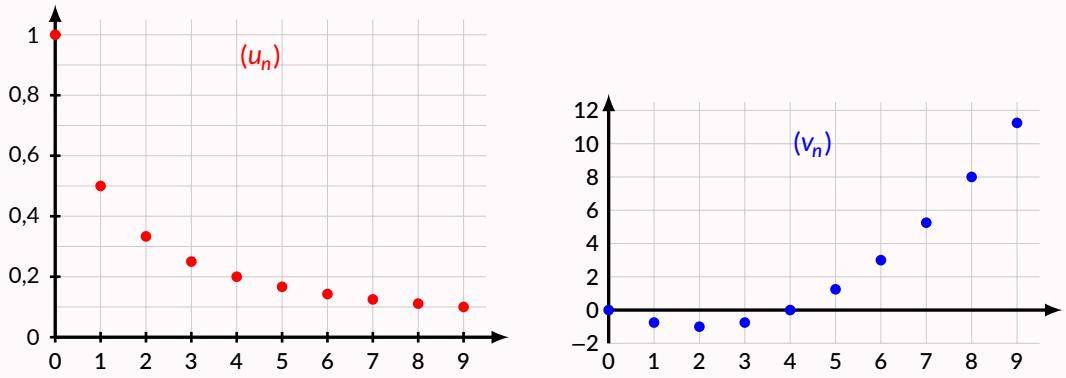
Attention : ces points ne doivent **pas être reliés**, contrairement à ce que l'on fait pour une fonction. Une fonction est définie sur un intervalle (entre deux valeurs, on peut toujours en trouver une troisième) alors qu'entre deux entiers consécutifs, il n'y a rien. Si $u_n = f(n)$, on peut tracer la courbe de f puis y matérialiser les points d'abscisse entière.

▸ Exercice 2.1

On considère les suites (u_n) et (v_n) définies sur $\mathbb{N}$ par $u_n = \dfrac{1}{n+1}$ et $v_n = 0{,}25n^2 - n$.

À l'aide de la calculatrice, et notamment du module `Suites`, représenter graphiquement (nuage de points) les 10 premiers termes de ces deux suites.

▸ Illustration 2.1



▸ Remarque 2.2

Le nuage de points permet(tra) de *conjecturer* le sens de variations et le comportement à l'infini de la suite

II.2. Cas d'une suite récurrente : la « toile »

▸ Méthode 2.2

Pour une suite définie par une relation $u_{n+1} = f(u_n)$, on peut la représenter sur l'axe des abscisses *sans calculer les termes* :

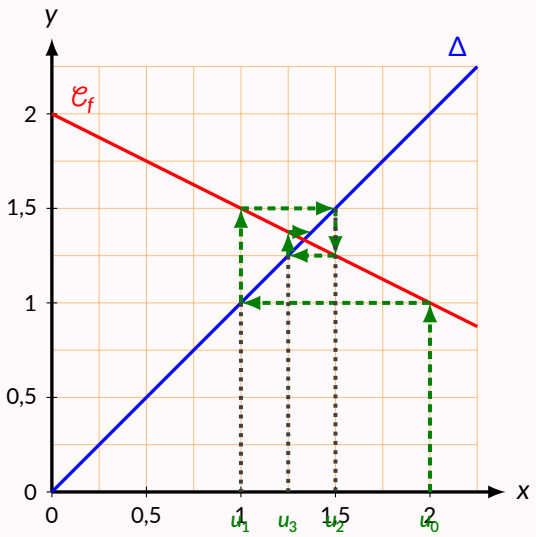
- on trace la courbe de la fonction f associée et la droite Δ d'équation $y = x$;
- on place le premier terme u_0 sur l'axe des abscisses ;
- on lit son image $f(u_0) = u_1$ sur l'axe des ordonnées grâce à la courbe ;
- on reporte u_1 sur l'axe des abscisses grâce à la droite Δ ;
- et on recommence… On obtient une « toile » (en « escalier » ou en « escargot ») qui permet de *conjecturer* le sens de variation et l'éventuelle limite de la suite.

▸ Exemple 2.1

Représenter graphiquement la suite définie par $\begin{cases} u_0 = 2 \\ u_{n+1} = -0{,}5u_n + 2 \quad \text{pour tout } n \in \mathbb{N} \end{cases}$.

La fonction associée est $f : x \mapsto -0{,}5x + 2$. On place $u_0 = 2$ puis on construit la toile : on lit $u_1 = 1$, $u_2 = 1{,}5$, $u_3 = 1{,}25$… Les termes se rapprochent d'une valeur proche de 1,3 en oscillant de part et d'autre.

▸ Illustration 2.2



▸ Exercice 2.2

On considère la suite définie par $u_0 = 1$ et $u_{n+1} = 1 + \dfrac{1}{u_n}$ pour tout entier n.

1. Calculer les 10 premiers termes de (u_n) et représenter son nuage de points.
2. En utilisant la technique de la « toile », représenter les premiers termes de (u_n) (on pourra tracer la courbe de $f : x \mapsto 1 + \frac{1}{x}$ sur $[0{,}8 ; 2{,}5]$ et la droite Δ).

Licence Ouverte Etalab 2.0

III. Sens de variation

III.1. Définitions

▸ Définition 3.1

Soit (u_n) une suite et n_0 un entier naturel. On dit que (u_n) est :

- **croissante** (à partir du rang n_0) si, pour tout entier $n \geqslant n_0$, $u_{n+1} \geqslant u_n$;
- **décroissante** (à partir du rang n_0) si, pour tout entier $n \geqslant n_0$, $u_{n+1} \leqslant u_n$;
- **constante** (à partir du rang n_0) si, pour tout entier $n \geqslant n_0$, $u_{n+1} = u_n$;
- **monotone** si elle est croissante ou décroissante.

▸ Remarques 3.1

- On définit de la même façon une suite **strictement croissante** ou **strictement décroissante** en remplaçant l'inégalité large par une inégalité stricte.
- Quand rien n'est précisé, $n_0 = 0$ (ou le rang du premier terme). Une suite peut n'être monotone qu'à partir d'un certain rang.
- Une suite n'est pas toujours monotone : la suite $u_n = (-1)^n$ vaut alternativement 1 et -1, elle n'est ni croissante, ni décroissante. C'est aussi le cas de la suite de la « toile » précédente, dont les termes sont alternativement au-dessus et en dessous de la limite.

III.2. Méthodes d'étude

▸ Méthode 3.1

Pour étudier le sens de variation d'une suite (u_n) :

- **par la différence** : on calcule $u_{n+1} - u_n$ et on étudie son signe. Si $u_{n+1} - u_n \geqslant 0$ pour tout n, la suite est croissante ; si $u_{n+1} - u_n \leqslant 0$ pour tout n, elle est décroissante ;
- **par le quotient**, si tous les termes sont **strictement positifs** : on calcule $\dfrac{u_{n+1}}{u_n}$ et on le compare à 1. Si $\dfrac{u_{n+1}}{u_n} \geqslant 1$ pour tout n, la suite est croissante ; si $\dfrac{u_{n+1}}{u_n} \leqslant 1$ pour tout n, elle est décroissante ;
- **par la fonction**, si la suite est explicite $u_n = f(n)$: si f est croissante (resp. décroissante) sur $[0 ; +\infty[$, alors (u_n) est croissante (resp. décroissante).

▸ Démonstration 3.1

Si tous les termes sont strictement positifs, multiplier l'inégalité $\frac{u_{n+1}}{u_n} \geqslant 1$ par $u_n > 0$ donne $u_{n+1} \geqslant u_n$. Si les termes étaient strictement négatifs, l'inégalité changerait de sens : c'est pourquoi on exige des termes strictement positifs.

▸ Remarques 3.2

- La méthode par la fonction ne fonctionne **que pour les suites explicites**, et sa réciproque est fausse : la suite $u_n = n^2 - n$ est croissante $(0 ; 0 ; 2 ; 6 \ldots)$ alors que $f : x \mapsto x^2 - x$ est décroissante sur $[0 ; 0{,}5]$.
- Pour une suite récurrente $u_{n+1} = f(u_n)$, les variations de f ne donnent **pas** celles de (u_n) (la toile précédente a une fonction f décroissante, mais la suite n'est pas monotone) : on utilise la différence $u_{n+1} - u_n$.

▸ Exemple 3.1

- **Différence.** Soit (u_n) définie pour tout $n \in \mathbb{N}$ par $u_n = n^2 + 3n$. Pour tout $n \in \mathbb{N}$:

$$\begin{aligned} u_{n+1} - u_n &= (n+1)^2 + 3(n+1) - (n^2 + 3n) \\ &= n^2 + 2n + 1 + 3n + 3 - n^2 - 3n \\ &= 2n + 4 = 2(n+2). \end{aligned}$$

  Or $n \geqslant 0$ donc $2(n+2) \geqslant 4 > 0$: la suite (u_n) est strictement croissante.
- **Différence (suite récurrente).** Soit (w_n) définie par $w_0 = 2$ et $w_{n+1} = w_n - n^2$.
  Alors $w_{n+1} - w_n = -n^2 \leqslant 0$ pour tout n : la suite (w_n) est décroissante.

— **Quotient.** Soit (b_n) définie par $b_n = \dfrac{5}{2^n}$. Tous les termes sont strictement positifs et

$$\frac{b_{n+1}}{b_n} = \frac{5}{2^{n+1}} \times \frac{2^n}{5} = \frac{2^n}{2^{n+1}} = \frac{1}{2} < 1,$$

donc la suite (b_n) est strictement décroissante.

▸ **Exercice 3.1**

Étudier le sens de variation des suites suivantes, définies pour tout $n \in \mathbb{N}$ (ou $n \geqslant 1$ pour la dernière) :

1. $a_n = n^2 - 5n$ (que peut-on dire à partir d'un certain rang ?) ;
2. $c_n = 4 \times 3^n$;
3. $d_0 = 10$ et $d_{n+1} = d_n - 3n - 1$;
4. $e_n = \dfrac{2^n}{n}$ pour $n \geqslant 1$.

IV. Notion intuitive de limite

S'intéresser à la limite d'une suite (u_n), c'est étudier le comportement des termes u_n lorsque le rang n prend des valeurs entières de plus en plus grandes, c'est-à-dire lorsque n « tend vers $+\infty$ ». Cette année, il ne s'agit que d'en avoir une notion *intuitive*, et de *conjecturer* une limite à l'aide d'un tableau de valeurs, d'un graphique ou d'un programme.

IV.1. Limite finie : suite convergente

▸ **Définition 4.1**

On dit que la suite (u_n) **converge** vers un réel ℓ lorsque, quand n devient très grand, les termes u_n se rapprochent de ℓ autant qu'on le souhaite. On note alors :

$$\lim_{n\to+\infty} u_n = \ell.$$

▸ **Exemple 4.1**

— La suite (u_n) définie pour $n \geqslant 1$ par $u_n = \dfrac{1}{n}$:

n	10	100	1 000	100 000
$u_n = \dfrac{1}{n}$	0,1	0,01	0,001	0,00001

Les termes se rapprochent de 0 : $\lim\limits_{n\to+\infty} \dfrac{1}{n} = 0$.

— La suite (v_n) définie par $v_n = \dfrac{4n-5}{2n+3}$:

n	100	1 000	10 000	100 000
v_n	1,9458	1,9945	1,9995	1,9999

Les termes se rapprochent de 2 : $\lim\limits_{n\to+\infty} \dfrac{4n-5}{2n+3} = 2$.

(Valeurs arrondies à 10^{-4}.)

IV.2. Limite infinie

▸ **Définition 4.2**

— On dit que (u_n) **tend vers** $+\infty$ si les termes u_n deviennent aussi grands que l'on veut, à condition de choisir n assez grand. On note $\lim\limits_{n\to+\infty} u_n = +\infty$.

— On dit que (u_n) **tend vers** $-\infty$ si les termes u_n sont négatifs et leur valeur absolue devient aussi grande que l'on veut,

à condition de choisir n assez grand. On note $\lim\limits_{n\to+\infty} u_n = -\infty$.

▸ **Exemple 4.2**

Pour $u_n = n^2$, les termes 100, 10 000, 1 000 000… (pour $n = 10$, 100, 1 000…) deviennent aussi grands que l'on veut : $\lim\limits_{n\to+\infty} n^2 = +\infty$.

IV.3. Suite sans limite

▸ **Définition 4.3**

Certaines suites n'admettent ni limite finie, ni limite infinie : on dit qu'elles **n'ont pas de limite**. Une suite qui ne converge pas (qu'elle tende vers $\pm\infty$ ou qu'elle n'ait pas de limite) est dite **divergente**.

▸ **Exemple 4.3**

La suite (u_n) définie par $u_n = (-1)^n$ vaut alternativement 1 (pour n pair) et -1 (pour n impair). Les termes ne se rapprochent d'aucune valeur précise : la suite diverge et n'a pas de limite.

▸ **Remarque 4.1**

Monotonie et limite sont deux notions différentes : une suite croissante ne tend pas forcément vers $+\infty$. Par exemple, $u_n = \dfrac{n}{n+1} = 1 - \dfrac{1}{n+1}$ est croissante mais converge vers 1. Inversement, une suite qui converge n'est pas forcément monotone (voir la « toile » de la partie 2).

▸ **Exercice 4.1**

1. À l'aide d'un tableur ou d'un programme, calculer les termes de la suite de la population de la ville (avec comme données $u_0 = 20\,000$ et $u_{n+1} = 0{,}98u_n + 300$) pour $n = 10$, $n = 100$ puis $n = 400$. Conjecturer la limite de (u_n) et interpréter dans le contexte.
2. Même question pour la suite $u_0 = 1$ et $u_{n+1} = 1 + \dfrac{1}{u_n}$.
3. La suite $u_n = \sin(n)$ semble-t-elle avoir une limite ? (On pourra utiliser le nuage de points.)

V. Suites et algorithmes

▸ **Définition 5.1**

Un passage dans une boucle s'appelle une **itération**. On distingue :

— la boucle `POUR` (`for`), pour répéter des instructions un **nombre de fois connu à l'avance** :

Code Python

```
for i in range(<première valeur>, <dernière valeur + 1>):
    <instructions>
```

Dans `range`, on écrit la première valeur de `i` utilisée et la première valeur de `i` *non* utilisée ;

— la boucle `TANT QUE` (`while`), pour répéter des instructions **jusqu'à ce qu'une condition ne soit plus vérifiée** :

Code Python

```
while <condition>:
    <instructions>
```

La condition doit être vérifiée au départ pour entrer dans la boucle.

▸ **Algorithme 5.1**

Pour calculer un terme donné d'une suite récurrente, on connaît le nombre d'itérations : on utilise une boucle `POUR`. Voici un programme pour la suite (u_n) définie par $u_0 = 300$ et $u_{n+1} = 1{,}02 \times u_n$ (un capital de 300 € placé à 2 % par an) :

Code Python

```python
def terme_un(n):
    u = 300                     # terme initial
    for i in range(1, n+1):     # i va de 1 à n
        u = 1.02 * u            # formule de récurrence
    return u
```

L'appel `terme_un(7)` renvoie environ `344.61`.

Algorithme 5.2

Si on cherche le rang à partir duquel les termes vérifient une condition, on ne connaît pas le nombre d'itérations : on utilise une boucle `TANT QUE`. Tant que la condition n'est pas atteinte on continue à calculer ; dès qu'elle l'est, on sort de la boucle.
Par exemple, pour savoir au bout de combien d'années le capital de 300 € placé à 2 % atteint au moins S euros :

Code Python

```python
def seuil(S):
    n = 0                       # rang initial
    u = 300                     # 300 € disponibles
    while u < S:                # tant qu'on a moins de S euros
        u = 1.02 * u            # on ajoute 2 %
        n = n + 1               # on incrémente le rang
    return n, u                 # on renvoie le rang et la somme
```

L'appel `seuil(350)` renvoie `(8, 351.4978...)` : il faut attendre 8 ans, et on dispose alors de $u_8 \approx 351{,}50$ €. (On avait bien $u_7 \approx 344{,}61 < 350$.)

Algorithme 5.3

Pour additionner les premiers termes d'une suite, on calcule la somme *au fur et à mesure*, car les termes sont écrasés d'une itération à l'autre. Par exemple, pour la somme $1^2 + 2^2 + \cdots + n^2$:

Code Python

```python
def somme_carres(n):
    s = 1                       # le premier carré vaut 1, la somme aussi
    for i in range(2, n+1):     # i va de 2 à n
        s = s + i**2            # on ajoute le carré i²
    return s
```

On obtient $1^2 + 2^2 + 3^2 = 14$ avec `somme_carres(3)` et $1^2 + 2^2 + \cdots + 100^2 = 338\,350$ avec `somme_carres(100)`.

Exercice 5.1

Reprendre la suite définie par $u_0 = 5$ et $u_{n+1} = 2u_n - 3$. Écrire une fonction `seuil_un(S)` qui renvoie le plus petit rang n tel que $u_n \geqslant S$, puis déterminer le plus petit rang pour lequel $u_n \geqslant 10^6$.

Point histoire

Les suites numériques sont liées à la mathématique de la mesure (mesures d'un phénomène prises à intervalles de temps réguliers) et à l'analyse (une suite numérique est l'équivalent discret d'une fonction numérique).
La notion de suite est présente dès qu'apparaissent des *procédés illimités* de calcul. On en trouve, par exemple, chez Archimède, spécialiste des procédés illimités d'approximation (séries géométriques de raison $\frac{1}{4}$) pour des calculs d'aires et de volumes, ou en Égypte vers 1700 av. J.-C. et plus récemment au I^{er} siècle apr. J.-C. dans le procédé d'extraction d'une racine carrée par la méthode de Héron d'Alexandrie.
On retrouve ensuite cette préoccupation plusieurs siècles plus tard (à partir du XV^{e} siècle) avec la méthode des indivisibles (Cavalieri, Torricelli, Pascal, Roberval). C'est ainsi que l'on voit Bernoulli, Newton, Moivre, Stirling et Wallis s'intéresser aux suites pour approcher des valeurs numériques. C'est à Lagrange que l'on doit, semble-t-il, la notation indicielle.
L'étude des suites ouvre la porte à celle des séries entières dont le but est d'approcher, non plus des nombres, mais des fonctions. Dans la seconde moitié du XX^{e} siècle, le développement des calculateurs et des ordinateurs donne un second souffle à l'étude des suites en analyse numérique grâce à la méthode des éléments finis. On en retrouve l'usage aussi dans les mathématiques financières.

Source : *Wikipédia*

Licence Ouverte Etalab 2.0