# Chapitre 6
# Suites

## I Quelques notions sur l'étude des suites

### I.1 Croissance

Soit une suite $(u_n)_{n\in\mathbb{N}}$. La suite est dite :
- **Croissante** si :
$$\forall n \in \mathbb{N}, \quad u_n \leqslant u_{n+1} \tag{1}$$
- **Décroissante** si :
$$\forall n \in \mathbb{N}, \quad u_{n+1} \leqslant u_n \tag{2}$$
- **Strictement croissante**, **strictement décroissante** si : on peut remplacer les inégalités larges par des inégalités strictes ci-dessus.
- **Monotone** si : elle est croissante ou décroissante.
- **Strictement monotone** si : elle est strictement croissante ou strictement décroissante.
- **Constante** si :
$$\forall n \in \mathbb{N}, \quad u_{n+1} = u_n \tag{3}$$
Dans ce cas alors $\forall n \in \mathbb{N}$, $u_n = u_0$.

*Remarque* 1. Se sortir de la tête qu'une suite est soit croissante soit décroissante.
Exemple de base : $u_n = (-1)^n$ : $+1, -1, +1, -1, \dots$.
De plus « strictement croissante » ne signifie **pas** « croissante mais pas constante ».
Exemple de base : $0, 0, 1, 1, 2, 2, \dots$

### I.2 Majorations

Soit une suite $(u_n)_{n\in\mathbb{N}}$. La suite est dite :
- **Majorée** si :
$$\exists M \in \mathbb{R}, \forall n \in \mathbb{N}, \quad u_n \leqslant M \tag{4}$$
- **Minorée** si :
$$\exists m \in \mathbb{R}, \forall n \in \mathbb{N}, \quad m \leqslant u_n \tag{5}$$
- **Bornée** si : elle est majorée et minorée.

*Remarque* 2. 1. Cela est équivalent à dire que $\{u_n \mid n \in \mathbb{N}\}$ est majoré (resp. minoré, borné).
2. On peut aussi remplacer les inégalités larges ci-dessus par des inégalités strictes.

On définit aussi :
- $(u_n)_{n\in\mathbb{N}}$ admet un **maximum** si : $\{u_n \mid n \in \mathbb{N}\}$ admet un maximum. C'est à dire :
$$\exists n_0 \in \mathbb{N}, \forall n \in \mathbb{N}, \quad u_n \leqslant u_{n_0} \tag{6}$$
- $(u_n)_{n\in\mathbb{N}}$ admet un **minimum** si :
$$\exists n_0 \in \mathbb{N}, \forall n \in \mathbb{N}, \quad u_{n_0} \leqslant u_n \tag{7}$$
- $(u_n)_{n\in\mathbb{N}}$ admet une **borne sup** (resp. **borne inf**) si : $\{u_n \mid n \in \mathbb{N}\}$ admet une borne sup (resp. borne inf).

Quelques propriétés déjà vues en découlent :

**Proposition 1.** *1. Toute suite majorée admet une borne sup.*
*2. Toute suite minorée admet une borne inf.*
*3. Une suite $(u_n)_{n\in\mathbb{N}}$ est bornée si et seulement si :*
$$\exists R \in \mathbb{R}, \forall n \in \mathbb{N}, \quad |u_n| \leqslant R \tag{8}$$

On peut aussi rajouter :

**Proposition 2.** *1. Si $(u_n)_{n\in\mathbb{N}}$ et $(v_n)_{n\in\mathbb{N}}$ sont deux suites bornées alors les suites $(u_n + v_n)_{n\in\mathbb{N}}$ et $(u_n \cdot v_n)_{n\in\mathbb{N}}$ sont encore bornées.*
*2. Si $(u_n)_{n\in\mathbb{N}}$ est bornée et si $\lambda \in \mathbb{R}$ alors la suite des $(\lambda u_n)_{n\in\mathbb{N}}$ est encore bornée.*
*3. La suite $(u_n)_{n\in\mathbb{N}}$ est bornée si et seulement si elle est bornée à partir d'un certain rang :*
$$\exists n_0 \in \mathbb{N}, \exists S \in \mathbb{R}, \forall n \geqslant n_0, \quad |u_n| \leqslant S \tag{9}$$

Ces propriétés sont aussi vraies dans $\mathbb{C}$, où on prend pour définition de « borné » la propriété (8).

## Annexe : quelques morceaux de programmes à connaitre

... et à savoir reproduire et adapter au cas par cas !

Soit une suite $(u_n)_{n\in\mathbb{N}}$ de terme initial $u_0$ donné et définie par une relation de récurrence du type $u_{n+1} = f(u_n)$ :

Calculer et afficher les $n$ premiers termes (de $u_0$ à $u_{n-1}$) :

```
# u représente u_(i)
u = u_0
for i in range(n):
    print("i = ", i, "u = ", u)
    u = f(u)
```

Emboiter tout cela dans une fonction qui calcule directement $u_n$ :

```
# calcule u_(n)
def suite(n):
    u = u_0
    for i in range(n):
        u = f(u)
    return u
```

Remplir une liste `L` des $n$ premiers termes de la suite ($u_0$ à $u_{n-1}$) :

```
# L[i] représente u_(i)
L = [0] * n
L[0] = u_0
for i in range(1, n):
    L[i] = f(L[i-1])
```

Autre version en ajoutant les termes uns par uns avec `append()` :

```
# u représente u_(i)
L = []
u = u_0
for i in range(n):
    L.append(u)
    u = f(u)
```

Idem avec une suite d'ordre 2 satisfaisant une relation du type $u_{n+2} = f(u_n, u_{n+1})$ et des premiers termes $u_0$, $u_1$ donnés :

Calculer en même temps le couple $(u_n, u_{n+1})$ :

```
# x représente u_(i) et y représente u_(i+1)
(x, y) = (u_0, u_1)
for i in range(n):
    print("i = ", i, "u = ", x)
    (x, y) = (y, f(x, y))
```

Ou dans une liste des $n$ premiers termes :

```
# L[i] représente u_(i)
L = [0] * n
L[0] = u_0
L[1] = u_1
for i in range(2, n):
    L[i] = f(L[i-2], L[i-1])
```