Introduction aux suites numériques, cours, première S

F.Gaudon

22 avril 2011

Table des matières

1 Notion de suite

1.1 Définitions

Définition :

On appelle *suite* toute application u de $\mathbb{N}$ dans $\mathbb{R}$.

Exemple :

$$\begin{aligned} u : \mathbb{N} &\longrightarrow \mathbb{R} \\ n &\longmapsto 3n^2 + 4 \end{aligned}$$

On a $u_0 = 3 \times 0^2 + 4 = 4$, $u_1 = 3 \times 1^2 + 4 = 7$, $u_5 = 3 \times 5^2 + 4 = 79$.

Définition :

- L'image de n par la suite u est notée u_n ou $u(n)$.
- u_n est appelé *terme* de la suite
- La suite u est notée $(u_n)_n \in \mathbb{N}$ ou $(u_n)_n$.

Remarque :

- Si u_0 est le premier *terme* de la suite, u_n est le $n+1^{\text{e}}$ terme.
- Si u_1 est le premier *terme* de la suite, u_n est le n^{e} terme.

1.2 Méthodes de construction des suites

1.2.1 Définition explicite

Définition :

Soit f une fonction de $\mathbb{R}^+$ dans $\mathbb{R}$, on définit une suite $(u_n)_n$ en posant pour tout $n \in \mathbb{N}$, $u_n = f(n)$.

Exemple :

$$\begin{aligned} u : \mathbb{N} &\longrightarrow \mathbb{R} \\ n &\longmapsto u_n = 3n \end{aligned}$$

On a $u_0 = 3 \times 0 - 2 = -2$ et $u_1 = 3 \times 1 - 2 = 1$ et $u_6 = 3 \times 6 - 2 = 16$.

1.2.2 Définition par récurrence

Définition :

> Soit f une fonction de $\mathbb{R}$ dans $\mathbb{R}$. Une suite définie par *récurrence* est une suite définie par la donnée de son premier terme u_p où p est un entier naturel et par la relation pour tout n entier naturel tel que $n \geq p$, $u_{n+1} = f(u_n)$.

Exemple :

$$u_n = \begin{cases} u_0 = 4 \\ u_{n+1} = 3u_n - 2 \end{cases}$$

On a $u_1 = 3 \times 4 - 2 = 10$, $u_2 = 3 \times 10 - 2 = 28$ et $u_3 = 3 \times 28 - 2 = 82$.

Premier algorithme :
Algorithme d'obtention du terme de rang n d'une suite (u_n) définie à partir d'un rang p et définie par $u_{n+1} = f(u_n)$ pour tout $n \geq p$.

```
Données : p, n, u_p
Début traitement
    u prend la valeur u_p ;
    pour k allant de p+1 à n faire
        u prend la valeur f(u) ;
    fin
    Afficher u
Fin
```

Exemple :
Soit (u_n) définie par $u_{n+1} = 3u_n + 2$ pour tout entier naturel n non nul et par $u_1 = 2$ n. p désigne le premier rang de la suite (1 ici), n désigne le rang dont on cherche à calculer le terme et u désigne les différents termes de la suite (sa première valeur, 2 ici, est demandée par le programme).

TI :	**Casio :**	**XCas :**

Prompt P,U,N
While $P < N$
$P + 1 \triangleright P$
$3 * U + 2 \triangleright U$
End
Disp "U=",U
	"P" :?→ P
	"U" :?→ U
	"N" :?→ N
	While $P < N$
	$P + 1 \to P$
	$3 * U + 2 \to U$
	WhileEnd
	"U" :U◢
		saisir("Premier rang p :",p);
		saisir("Premier terme u_p :",u);
		saisir("Rang du terme :",n);
		pour k de p+1 jusque n faire
		u:=3*u+2;
		fpour;
		afficher("Terme de rang n"+u);

Deuxième algorithme :
Algorithme d'obtention de la liste des termes jusqu'à un rang n donné pour une suite (u_n) définie par récurrence par $u_{n+1} = f(u)$ et son premier terme u_p.

Données : p, n, u_p
Début traitement
  u prend la valeur u_p ;
  s est une liste de vide ;
  pour k ***allant de*** $p+1$ ***à*** n **faire**
    Ajouter u à la liste s ;
    u prend la valeur $f(u)$;
  fin
  Afficher s
Fin

Exemple :
Soit (u_n) définie par $u_{n+1} = 3u_n + 2$ pour tout entier naturel n non nul et par $u_1 = 2$ n. p désigne le premier rang de la suite (1 ici), n désigne le rang dont on cherche à calculer le terme et u désigne les différents termes de la suite (sa première valeur, 2 ici, est demandée par le programme).

XCas :

```
saisir("Premier rang p :",p);
saisir("Premier terme up :",u);
saisir("Rang du terme :",n);
s:=[];
s[0]:=u;
pour k de p+1 jusque n faire
   u:=3*u+2;
   s[k-p]:=u;
   fpour;
afficher("Liste des termes : "+s);
```

Python :

```
p=int(raw_input("Premier rang p : "))
u=float(raw_input("Premier terme up : "))
n=int(raw_input("Rang n du dernier terme : "))
s=[]
s.append(u)
for k in range(p+1,n+1):
  u=3*u+2
  s.append(u)
print ("Liste des termes :",s)
```