Fiche d'exercices : suites auxiliaires

Première spécialité mathématiques

Principe. Une suite (u_n) est difficile à étudier directement ? On construit une suite *auxiliaire* (v_n) à partir de (u_n), dont le comportement est plus simple (constante, positive, monotone, explicite…), puis on revient à (u_n).
Consigne. Dans cette fiche, aucune suite auxiliaire n'est arithmétique ou géométrique (hors cas dégénéré d'une suite constante). Les résultats qui se démontrent par récurrence sont admis. Chaque exercice est accompagné d'un algorithme Python à exécuter, à comprendre et à commenter. Niveau : ⋆ accessible, ⋆⋆ intermédiaire, ⋆⋆⋆ plus délicat.

Exercice 1 — Une suite qui converge très vite ⋆⋆

Soit (u_n) définie par $u_0 = \dfrac{3}{2}$ et, pour tout $n \in \mathbb{N}$, $u_{n+1} = u_n^2 - 2u_n + 2$. On pose $v_n = u_n - 1$.

1. Calculer u_1 et u_2.
2. Montrer que, pour tout n, $v_{n+1} = v_n^2$.
3. On admet (par récurrence) que $0 < v_n \leq \dfrac{1}{2}$ pour tout n.
4. En étudiant le signe de $v_{n+1} - v_n$, en déduire que (v_n) est décroissante, puis que (u_n) est décroissante et minorée par 1.
5. Calculer v_0, v_1, v_2 et conjecturer une expression de v_n en fonction de n. On admet (par récurrence) que cette expression est valable pour tout n.
6. Exécuter l'algorithme ci-dessous. Que renvoie-t-il ? Le comparer avec la question 5.

Algorithme Python

```python
def u(n):
    x = 1.5
    for _ in range(n):
        x = x*x - 2*x + 2
    return x

n = 0
while u(n) - 1 >= 1e-10:   # seuil sur la suite auxiliaire v_n = u_n - 1
    n += 1
print(n, u(n))
```

Exercice 2 — Suite et télescopage ⋆

Soit (u_n) définie par $u_0 = 1$ et $u_{n+1} = u_n + \dfrac{1}{(n+1)(n+2)}$. On pose $v_n = u_n + \dfrac{1}{n+1}$.

1. Calculer u_1 et u_2.
2. Vérifier que $\dfrac{1}{(n+1)(n+2)} = \dfrac{1}{n+1} - \dfrac{1}{n+2}$.
3. Montrer que $v_{n+1} = v_n$ pour tout n. Quelle est la valeur de v_n ?
4. En déduire l'expression de u_n en fonction de n.
5. Montrer que (u_n) est croissante et que $u_n < 2$ pour tout n.
6. Que calcule l'algorithme ci-dessous ? Quelle valeur de n affiche-t-il ? Le justifier par le calcul.

Algorithme Python

```
u = 1
n = 0
while u <= 1.97:
    u = u + 1/((n+1)*(n+2))
    n += 1
print(n, u)
```

Exercice 3 — Suites adjacentes ⋆⋆

Pour $n \geq 1$, on pose $a_n = \sum_{k=1}^{n} \frac{1}{k^2}$ et $b_n = a_n + \frac{1}{n}$.

1. Montrer que (a_n) est croissante.
2. Montrer que $b_{n+1} - b_n = -\frac{1}{n(n+1)^2}$. En déduire que (b_n) est décroissante.
3. On pose $d_n = b_n - a_n$. Donner l'expression de d_n. Que dire de son signe et de son comportement quand n devient grand ?
4. En déduire que, pour tout $n \geq 1$, $a_n \leq b_n \leq 2$.
5. Exécuter l'algorithme ci-dessous. Interpréter les valeurs affichées, puis vérifier que $\frac{\pi^2}{6} \approx 1{,}6449$ est bien situé entre a_n et b_n.

Algorithme Python

```
from math import pi

a = 0.0
n = 0
while True:
    n += 1
    a += 1/n**2
    b = a + 1/n
    if 1/n < 0.01:          # la suite auxiliaire d_n = 1/n est < 0,01
        break
print(n, a, b, a <= pi**2/6 <= b)
```

Exercice 4 — Approcher $\sqrt{2}$ (méthode de Héron) ⋆⋆⋆

Soit (u_n) définie par $u_0 = 2$ et $u_{n+1} = \frac{u_n}{2} + \frac{1}{u_n}$. On pose $v_n = u_n - \sqrt{2}$.

1. On admet (par récurrence) que $u_n > 0$ pour tout n.
2. Montrer que $v_{n+1} = \frac{v_n^2}{2u_n}$. En déduire que $v_n \geq 0$ pour tout n, c'est-à-dire $u_n \geq \sqrt{2}$.
3. Montrer que $u_{n+1} - u_n = \frac{2 - u_n^2}{2u_n}$, puis que (u_n) est décroissante.
4. Que dire de (u_n) ? Pourquoi u_n est-il une valeur approchée par excès de $\sqrt{2}$?
5. Exécuter l'algorithme ci-dessous. Combien d'itérations sont nécessaires pour que $|u_n^2 - 2| \leq 10^{-12}$?

Algorithme Python

```
u = 2.0
n = 0
while abs(u*u - 2) > 1e-12:
    u = u/2 + 1/u
    n += 1
print(n, u)
```

Exercice 5 — Une suite de racines emboîtées ⋆⋆

Soit (u_n) définie par $u_0 = 0$ et $u_{n+1} = \sqrt{2 + u_n}$. On pose $v_n = 2 - u_n$.

1. Calculer u_1 et u_2.
2. On admet (par récurrence) que $0 \leq u_n < 2$ pour tout n. En déduire que $v_n > 0$.
3. Montrer que $v_{n+1} = \dfrac{v_n}{2 + u_{n+1}}$ (on pourra multiplier par la quantité conjuguée). En déduire que $0 < v_{n+1} \leq \dfrac{v_n}{2}$.
4. Déduire de la question 3 que (v_n) est décroissante, puis que (u_n) est croissante.
5. On admet (par récurrence) que $v_n \leq \dfrac{2}{2^n}$ pour tout n.
6. Exécuter l'algorithme ci-dessous. Quelle précision garantit la question 5 pour la valeur de n obtenue ?

Algorithme Python

```
from math import sqrt

u = 0.0
n = 0
while 2 - u >= 1e-6:
    u = sqrt(2 + u)
    n += 1
print(n, u)
```

Exercice 6 — Divergence par minoration ⋆⋆

Soit (u_n) définie par $u_0 = 1$ et $u_{n+1} = u_n + 1 + \dfrac{1}{u_n}$. On pose $v_n = u_n - n$.

1. On admet (par récurrence) que $u_n > 0$ pour tout n.
2. Montrer que $v_{n+1} - v_n = \dfrac{1}{u_n}$. En déduire que (v_n) est croissante.
3. En déduire que $u_n \geq n + 1$ pour tout n.
4. Justifier que, pour tout réel A, il existe un rang à partir duquel $u_n > A$.
5. Exécuter l'algorithme ci-dessous. La valeur affichée est-elle cohérente avec la question 3 ?

Algorithme Python

```
u = 1.0
n = 0
while u <= 100:
    u = u + 1 + 1/u
    n += 1
print(n, u)
```

Exercice 7 — Une somme avec une suite auxiliaire cubique ⋆⋆

Pour $n \in \mathbb{N}$, on pose $S_n = \sum_{k=1}^{n} k(k+1)$ (avec $S_0 = 0$) et $v_n = \dfrac{n(n+1)(n+2)}{3}$.

1. Exprimer S_{n+1} en fonction de S_n et n.
2. Montrer que $v_{n+1} - v_n = (n+1)(n+2)$.
3. On pose $w_n = S_n - v_n$. Montrer que $w_{n+1} = w_n$, puis calculer w_0. En déduire une expression de S_n.
4. Calculer S_{20}.
5. Exécuter l'algorithme ci-dessous : quel est le but du test final ?

Algorithme Python

```python
def S(n):
    s = 0
    for k in range(1, n+1):
        s += k*(k+1)
    return s

def v(n):
    return n*(n+1)*(n+2)//3

print(S(100), v(100))
print(all(S(n) == v(n) for n in range(0, 201)))
```

Exercice 8 — Quotients de termes consécutifs de la suite de Fibonacci ⋆⋆⋆

Soit (F_n) définie par $F_0 = 1$, $F_1 = 1$ et $F_{n+2} = F_{n+1} + F_n$. On pose $r_n = \dfrac{F_{n+1}}{F_n}$.

1. Calculer F_2, F_3, F_4, F_5 puis r_0, r_1, r_2, r_3, r_4.
2. On admet que $F_n > 0$ pour tout n. Montrer que $r_{n+1} = 1 + \dfrac{1}{r_n}$.
3. On admet (par récurrence) que $1 \leq r_n \leq 2$ pour tout n.
4. On admet que (r_n) se rapproche d'un réel $\varphi > 0$ tel que $\varphi = 1 + \dfrac{1}{\varphi}$. Montrer que $\varphi = \dfrac{1+\sqrt{5}}{2}$.
5. Exécuter l'algorithme ci-dessous. À partir de quel rang r_n est-il une valeur approchée de φ à 10^{-6} près ?

Algorithme Python

```python
from math import sqrt

phi = (1 + sqrt(5)) / 2
a, b = 1, 1                    # a = F_n, b = F_(n+1)
n = 0
while abs(b/a - phi) >= 1e-6:  # r_n = b/a
    a, b = b, a + b
    n += 1
print(n, b/a)
```

Exercice 9 — Une suite périodique ⋆⋆

Soit (u_n) définie par $u_0 = 2$ et $u_{n+1} = \dfrac{1}{1-u_n}$.

1. Calculer u_1, u_2, u_3, u_4. Que remarque-t-on ?
2. Montrer que, si $u_n \notin \{0, 1\}$, alors $u_{n+2} = \dfrac{u_n - 1}{u_n}$, puis que $u_{n+3} = u_n$.
3. On pose $w_n = u_n + u_{n+1} + u_{n+2}$. Montrer que $w_{n+1} = w_n$ et calculer w_0.
4. En déduire $\displaystyle\sum_{k=0}^{2024} u_k$ puis la valeur de u_{2025}.
5. Exécuter l'algorithme ci-dessous (calcul en fractions exactes) et comparer avec la question 4.

Algorithme Python

```python
from fractions import Fraction

u = Fraction(2)
s = Fraction(0)
for n in range(2025):
    s += u              # s = u_0 + ... + u_n
    u = 1 / (1 - u)     # u devient u_(n+1)
print(s, u)             # somme de u_0 a u_2024, puis u_2025
```

Exercice 10 — Régions du plan ⋆⋆

On trace n droites du plan, deux à deux sécantes et sans que trois d'entre elles soient concourantes. On note R_n le nombre de régions du plan ainsi délimitées. On a $R_0 = 1$ et, en ajoutant la $(n+1)$-ième droite, celle-ci traverse $n+1$ régions et les coupe chacune en deux :

$$R_{n+1} = R_n + n + 1.$$

On pose $v_n = R_n - \dfrac{n(n+1)}{2}$.

1. Calculer R_1, R_2, R_3 et vérifier ces valeurs par un schéma.
2. Montrer que (v_n) est constante. Quelle est sa valeur ?
3. En déduire l'expression de R_n en fonction de n.
4. Combien de régions obtient-on avec 10 droites ?
5. Exécuter l'algorithme ci-dessous. Retrouver sa réponse par le calcul (résolution d'une inéquation).

Algorithme Python

```python
R = 1
n = 0
while R <= 1000:
    n += 1
    R = R + n       # R_n = R_(n-1) + n
print(n, R)         # plus petit nombre de droites donnant plus de 1000 regions
```

Éléments de réponse

Exercice 1. $u_1 = \frac{5}{4}$, $u_2 = \frac{17}{16}$; $v_n = \left(\frac{1}{2}\right)^{2^n}$; l'algorithme affiche $n = 6$ (car $2^{-32} \approx 2{,}3 \times 10^{-10} \geq 10^{-10}$ mais $2^{-64} < 10^{-10}$).

Exercice 2. $u_1 = \frac{3}{2}$, $u_2 = \frac{5}{3}$; $v_n = 2$ donc $u_n = 2 - \frac{1}{n+1}$; $u_n > 1{,}97 \iff n + 1 > \frac{100}{3}$, donc $n = 33$.

Exercice 3. $d_n = \frac{1}{n}$; l'algorithme s'arrête à $n = 101$ avec $a_{101} \approx 1{,}6351$ et $b_{101} \approx 1{,}6450$; l'écart est inférieur à 0,01 et $\frac{\pi^2}{6} \in [a_n ; b_n]$.

Exercice 4. $u_1 = \frac{3}{2}$, $u_2 = \frac{17}{12}$; (u_n) décroissante et minorée par $\sqrt{2}$; l'algorithme affiche $n = 5$ et $u_5 \approx 1{,}414213562373095$.

Exercice 5. $u_1 = \sqrt{2}$, $u_2 = \sqrt{2 + \sqrt{2}}$; l'algorithme affiche $n = 11$; la majoration $v_{11} \leq \frac{2}{2^{11}} \approx 10^{-3}$ est moins fine que le résultat réel (la suite converge en fait plus vite, avec un rapport proche de $\frac{1}{4}$).

Exercice 6. $u_n \geq n + 1$ donc $u_n > 100$ dès $n = 100$; l'algorithme affiche $n = 95$, ce qui est cohérent (la minoration est seulement une borne : en fait $u_n \approx n + \ln n + \dots$).

Exercice 7. $S_{n+1} = S_n + (n + 1)(n + 2)$; $w_n = 0$ donc $S_n = \frac{n(n+1)(n+2)}{3}$; $S_{20} = 3080$, $S_{100} = 343\,400$.

Exercice 8. $F_2, \dots, F_5 = 2, 3, 5, 8$; $r_0, \dots, r_4 = 1, 2, \frac{3}{2}, \frac{5}{3}, \frac{8}{5}$; $\varphi^2 - \varphi - 1 = 0$ donne $\varphi = \frac{1+\sqrt{5}}{2}$; l'algorithme affiche $n = 15$.

Exercice 9. $u_1 = -1$, $u_2 = \frac{1}{2}$, $u_3 = 2$ (période 3) ; $w_n = \frac{3}{2}$; $\sum_{k=0}^{2024} u_k = 675 \times \frac{3}{2} = \frac{2025}{2}$; $u_{2025} = u_0 = 2$.

Exercice 10. $R_1 = 2$, $R_2 = 4$, $R_3 = 7$; $v_n = 1$ donc $R_n = 1 + \frac{n(n+1)}{2}$; $R_{10} = 56$; $R_n > 1000 \iff n(n + 1) > 1998 \iff n \geq 45$ ($R_{45} = 1036$).