DEVOIR SURVEILLÉ 7 - RÉDUCTION ET COUPLES DE VARIABLES DISCRÈTES

Durée : 3h

La présentation, la lisibilité, la qualité de la rédaction et la précision des raisonnements entreront pour une part importante dans l'appréciation des copies.
Les consignes suivantes sont à respecter sous peine d'absence de correction ou de notation :
- *Les résultats doivent être encadrés,*
- *Les pages doivent être numérotées,*
- *Chaque nouvel exercice ou nouveau problème commencera sur une nouvelle copie, ou à défaut, sur une nouvelle feuille. (Les exercices peuvent être traités dans l'ordre souhaité.)*
- *Tout résultat doit être dûment justifié.*
- *Les programmes Python doivent être expliqués.*

Les téléphones portables et les calculatrices sont interdits et doivent être rangés dans les sacs.
Si, au cours de l'épreuve, un candidat repère ce qui lui semble être une erreur d'énoncé, il la signalera sur sa copie et poursuivra en expliquant les raisons des initiatives qu'il sera amené à prendre.

Chaque étudiant portera en tête de son devoir le tableau suivant :

Rédiger	*Rais.*	*Mod.*	*Calc.*	*Rech.*	*Représ.*	*Comm.*	*Cours*

EXERCICE 1 : À l'approche des fêtes de Pâques, un chocolatier a décidé de proposer quatre types de chocolats à la dégustation pour chaque client entrant dans le magasin. Le nombre X de clients par heure suit une loi de Poisson de moyenne N. Chacun d'entre eux choisit au hasard un des chocolats parmi les quatre variétés C_1, C_2, C_3 et C_4. On note alors X_1, X_2, X_3, X_4 respectivement le nombre de chocolats choisis de type C_1, C_2, C_3, C_4 pendant une heure.

1. **a)** Pour tout $k \in \{1,2,3,4\}$ et $x \in \mathbb{N}$, reconnaître la loi de $X_{k/X=x}$.
b) En déduire la loi conjointe du couple (X_k, X), où $k \in [\![1,4]\!]$.
c) Déterminer la loi de X_1. En déduire la loi de X_2, X_3, X_4.

2. On définit pour tout $k \in [\![1,4]\!]$, la variable aléatoire Y_k par :

$$Y_k = \frac{X_k}{X}.$$

Calculer les espérances des variables aléatoires Y_k.

PROBLÈME 1 :

Nous allons nous intéresser ici à la résolution d'une équation différentielle homogène du troisième ordre à coefficients constants.
Pour une fonction $y : \mathbb{R} \to \mathbb{R}$ de classe $\mathcal{C}^3$ sur $\mathbb{R}$, $y^{(3)}$ désigne la dérivée troisième de y.
Considérons l'équation différentielle :

$$y^{(3)} - 2y'' - y' + 2y = 0 \tag{1}$$

1. Soit y une solution de (1) sur $\mathbb{R}$, et x un nombre réel.
Notons : $Y = \begin{pmatrix} y''(x) \\ y'(x) \\ y(x) \end{pmatrix}$, $Y' = \begin{pmatrix} y^{(3)}(x) \\ y''(x) \\ y'(x) \end{pmatrix}$, $A = \begin{pmatrix} 2 & 1 & -2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.
Déterminer une relation entre Y', A et Y.

2. Nous allons étudier les éléments propres de la matrice A.
a) Déterminer les valeurs propres de A et en déduire que A est diagonalisable.
b) Déterminer les sous-espaces propres de A. *Les vecteurs des bases des sous-espaces propres seront choisis avec une troisième composante égale à* 1.
Déterminer alors une matrice P réelle carrée d'ordre 3 et inversible telle que :

$$D = P^{-1}AP = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

3. Soit y une solution de (1) sur $\mathbb{R}$. Montrer que : $P^{-1}Y' = DP^{-1}Y$.

4. On pose $Z = P^{-1}Y$. On note $Z = \begin{pmatrix} z_1(x) \\ z_2(x) \\ z_3(x) \end{pmatrix}$.
a) Justifier que les fonctions z_i sont $\mathcal{C}^1$ pour $i = 1,2,3$. On pose alors $Z' = \begin{pmatrix} z_1'(x) \\ z_2'(x) \\ z_3'(x) \end{pmatrix}$. Montrer que $Z' = P^{-1}Y'$.
b) En déduire une relation entre z_i et z_i' pour $i = 1,2,3$ puis expliciter les fonctions z_i.

5. **a)** Déterminer alors l'ensemble S des solutions de (1) sur $\mathbb{R}$.
b) Montrer que S est un $\mathbb{R}$-espace vectoriel de dimension 3.

Problème 2 :

On souhaite dans ce problème écrire un programme permettant de calculer, sous certaines conditions, un vecteur propre d'une matrice. Les différents programmes demandés devront être écrits en Python. Quelques fonctions utiles sont rappelées en annexe à la fin du sujet.

Notons $\mathcal{M}_{m,n}(\mathbb{R})$ l'ensemble des matrices de taille $m \times n$ à coefficients réels et $\mathcal{M}_p(\mathbb{R}) = \mathcal{M}_{p,p}(\mathbb{R})$ l'ensemble des matrices carrées. À chaque matrice $M = (m_{i,j}) \in \mathcal{M}_{m,n}(\mathbb{R})$, on associe un nombre réels positif $||M||$, appelé *norme de* M, défini par

$$||M|| = \max_{i,j} |m_{i,j}|.$$

Remarquons que $||M||$ est toujours strictement positif, sauf lorsque la matrice M est nulle.

1. Écrire une fonction `norme(M)` qui, étant donnée une matrice `M` de taille quelconque, calcule et renvoie le nombre $||M||$. *On interdit le recours à une fonction* `max` *prédéfinie pour cette question.*

2. Écrire une fonction `Normalise(v)` qui, étant donné une matrice colonne $v \in \mathcal{M}_{p,1}(\mathbb{R})$ non nulle renvoie une nouvelle matrice colonne $\tilde{v}$, de même forme, égale à

$$\tilde{v} = \frac{v}{||v||}$$

On se donne à présent une matrice carrée $A \in \mathcal{M}_p(\mathbb{R})$. Soit v_0 un élément quelconque de $M_{p,1}(\mathbb{R})$. En supposant qu'aucun des termes n'est dans le noyau de A, on peut former la suite $(v_n)_{n\geq 0}$ d'éléments de $M_{p,1}(\mathbb{R})$ définie par la relation de récurrence

$$v_{n+1} = \frac{Av_n}{||Av_n||}.$$

3. Écrire une fonction `PuissanceIteree(A,n)` qui, étant donnée une matrice carrée A et un entier naturel n, détermine la taille p de A, choisit aléatoirement une matrice colonne $v_0 \in M_{p,1}(\mathbb{R})$, puis calcule et renvoie, ensupposant que tous les termes de la suite sont bien définis, la matrice colonne v_n.

On peut montrer que si $A \in M_p(\mathbb{R})$ est diagonalisable et possède les valeurs propres $\lambda_1, \ldots, \lambda_p$ avec

$$|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_p|,$$

alors la suite (v_n) ci-dessus existe et ses composantes convergent vers les composantes d'un vecteur propre de A associé à la valeur propre λ_1, sauf pour quelques choix de v_0. La probabilité, en choisissant aléatoirement v_0, de tomber sur l'une de ces exceptions, est nulle.

On se propose maintenant d'écrire une fonction `VecteurPropre(A,e)` qui, étant donné une matrice carrée A satisfaisant les hypothèses décrites ci-dessus et un nombre `e`>0, calcule les termes de la suite (v_n) jusqu'à ce que deux termes successifs vérifient $||v_n - v_{n+1}|| < e$, et renvoie alors la matrice v_{n+1}. Voici trois propositions de programme.

Programme 1

```
def ValeurPropre(A,e):
    d=A.shape
    v=matrix(random.rand(d[0],1))
    v=Normalise(v)
    w=Normalise(A*v)
    while Norme(v-w)>=e:
        v=w
        w=Normalise(A*v)
    return(w)
```

Programme 2

```
def ValeurPropre(A,e):
    d=A.shape
    v=matrix(random.rand(d[0],1))
    v=Normalise(v)
    w=Normalise(A*v)
    ecart=Norme(v-w)
    while ecart>=e:
        v=w
        w=Normalise(A*v)
    return(w)
```

Programme 3

```
def ValeurPropre(A,e):
    d=A.shape
    v=matrix(random.rand(d[0],1))
    v=Normalise(v)
    while Norme(v-Normalise(A*v))>=e:
        v=Normalise(A*v)
    return(Normalise(A*v))
```

4. Parmi ces trois programmes, indiquer lequel est (ou lesquels sont) correct(s). Pour chaque programme incorrect, on indiquera succinctement ce qui ne va pas.

ANNEXE : rappel de quelques commandes utiles

Interprétation	*Commande Python*
Construction d'une nouvelle matrice de taille $m \times n$, ne contenant que des zéros	`matrix(zeros([m,n]))`
Matrice identité de taille p	`matrix eye(p))`
Construction d'une nouvelle matrice de taille $m \times n$, remplie avec des coefficients choisis aléatoirement dans $[0, 1[$.	`matrix(random.rand(m,n))`
Copie de la matrice A dans une nouvelle matrice B	`B=matrix(A)`

Dimensions de la matrice A
- nombre de lignes
- nombre de colonnes
	`d=A.shape`
	`d[0]`
	`d[1]`

Opérations matricielles (pour des matrices A et B de taille compatibles)	`A+B, A-B, A*B`
Coefficient d'indices (i, j) dans la matrice M	`M[i,j]`

DS 7 - Réduction et couples de variables discrètes — Solutions

Exercice 1

1. 1.**a)** Si le nombre de clients est x, alors le nombre X_k de chocolats correspond au nombre de succès "choix du chocolat C_k" de probabilité $\frac{1}{4}$ dans une succession d'épreuves de Bernoulli indépendantes. Ainsi,

$$\boxed{X_{k/X=x} \hookrightarrow \mathcal{B}\left(x, \frac{1}{4}\right)}$$

1.**b)** La variable X_k est à valeur $\mathbb{N}$, donc $Supp((X_k, X)) \subset \mathbb{N}^2$

Soit $(i,j) \in \mathbb{N}$.

On commence par remarquer que

$$P\left((X_k = i) \cap (X = j)\right) = 0 \qquad \text{si } i > j.$$

Pour $i \leq j$, on a :

$$P((X_k = i) \cap (X = j)) = P_{(X=j)}(X_k = i)P(X = j) = e^{-N}\frac{N^j}{j!}P_{(X=j)}(X_k = i)$$

Or la loi de X_k sachant $X = j$ est la loi binomiale $\mathcal{B}(j, 1/4)$. D'où :

$$P_{(X=j)}(X_k = i) = \binom{j}{i}\left(\frac{1}{4}\right)^i\left(\frac{3}{4}\right)^{j-i}.$$

D'où :

$$\boxed{P((X_k = i) \cap (X = j)) = \begin{cases} 0 & \text{si } i > j \\ e^{-N}\dfrac{N^j}{j!}\binom{j}{i}\dfrac{3^{j-i}}{4^j} & \text{si } 0 \leq i \leq j \end{cases}}$$

1.**c)** On a $Supp(X_1) = \mathbb{N}$. Soit $i \in \mathbb{N}$.

Comme $\{(X = j)\}_{j\in\mathbb{N}}$ est un système complet, on a :

$$\begin{aligned} P(X_1 = i) &= \sum_{j=0}^{+\infty} P(X_1 = i, X = j) \\ &= \sum_{j=i}^{+\infty} e^{-N}\frac{N^j}{j!}\binom{j}{i}\frac{3^{j-i}}{4^j} \\ &= e^{-N}\frac{N^i}{4^i i!}\sum_{j=i}^{+\infty}\frac{N^{j-i}3^{j-i}}{(j-i)!4^{j-i}} \\ &= e^{-N}\frac{N^i}{4^i i!}\sum_{k=0}^{+\infty}\frac{\left(\frac{3N}{4}\right)^k}{k!} \\ &= e^{-N}\frac{\left(\frac{N}{4}\right)^i}{i!}e^{\frac{3N}{4}} \\ &= e^{\frac{-N}{4}}\frac{\left(\frac{N}{4}\right)^i}{i!} \end{aligned}$$

Ainsi,

$$\boxed{X_1 \hookrightarrow \mathcal{P}\left(\frac{N}{4}\right)}$$

Par symétrie du problème, on a

$$\boxed{X_1, X_2, X_3, X_4 \hookrightarrow \mathcal{P}\left(\frac{N}{4}\right)}$$

2. La variable Y_k est finie. Son espérance existe donc pour tout $k \in [\![1, 4]\!]$.

On observe que

$$Y_1 + Y_2 + Y_3 + Y_4 = 1.$$

Par symétrie, on a de plus :

$$\mathbb{E}(Y_1) = \mathbb{E}(Y_2) = \mathbb{E}(Y_3) = \mathbb{E}(Y_4),$$

d'où

$$\boxed{E(Y_k) = 1/4 \qquad \forall k = 1, 2, 3, 4}$$

Problème 1 :

1. On a

$$\begin{pmatrix} y^{(3)}(x) \\ y''(x) \\ y'(x) \end{pmatrix} = \begin{pmatrix} 2y''(x) + y'(x) - 2y(x) \\ y''(x) \\ y'(x) \end{pmatrix} = AY$$

On voit alors que si y est solution de (1), alors

$$\boxed{Y' = AY}$$

2. 2.**a)** Soit $\lambda \in \mathbb{C}$ et I la matrice identité de taille 3. On a

$$(A-\lambda I)\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \Leftrightarrow \begin{cases} (2-\lambda)x & +y & -2z & = & 0 \\ x & -\lambda y & & = & 0 \\ & y & -\lambda z & = & 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} (2-\lambda)x & +y & -2z & = & 0 \\ x & -\lambda y & & = & 0 \\ -\lambda(2-\lambda)x & +(2-\lambda)y & & = & 0 \quad L_3 \leftarrow 2L_3 - \lambda L_1 \end{cases}$$

$$\Leftrightarrow \begin{cases} (2-\lambda)x & +y & -2z & = & 0 \\ x & -\lambda y & & = & 0 \\ & (2-\lambda)(1-\lambda^2)y & & = & 0 \quad L_3 \leftarrow L_3 + \lambda(2-\lambda)L_2 \end{cases}$$

$$\Leftrightarrow \begin{cases} \boxed{-2}z & +(2-\lambda)x & & +y & = & 0 \\ & \boxed{1}x & & -\lambda y & = & 0 \\ & & \boxed{(2-\lambda)(1-\lambda)(1+\lambda)}y & & = & 0 \end{cases}$$

Ce dernier système étant échelonné, on voit que le système considéré est de rang 3 si et seulement si $(2-\lambda)(1-\lambda)(1+\lambda) \neq 0$, ce qui assure que les valeurs propres de A sont $-1, 1$ et 2.

$$\boxed{\text{Les valeurs propres de } A \text{ sont } -1, 1 \text{ et } 2.}$$

On voit ainsi que la matrice (d'ordre 3) A admet trois valeurs propres distinctes, ce qui assure qu'elle est diagonalisable.

$$\boxed{A \text{ est diagonalisable.}}$$

2.**b)** • Pour $\lambda = 1$, le système précédent équivaut à

$$\begin{cases} -2z & +x & +y & = & 0 \\ & x & -y & = & 0 \end{cases}$$

soit encore à

$$\begin{cases} z = x \\ x = y \end{cases}$$

On voit alors que le sous-espace propre associé à 1 est la droite vectorielle engendrée par $(1, 1, 1)$.

$$\boxed{\text{Le sous-espace propre associé à 1 est la droite vectorielle engendrée par } u = (1, 1, 1).}$$

• Pour $\lambda = -1$, le système précédent équivaut à

$$\begin{cases} -2z & +3x & +y & = & 0 \\ & x & +y & = & 0 \end{cases}$$

soit encore à

$$\begin{cases} z = x \\ y = -x \end{cases}$$

On voit alors que le sous-espace propre associé à -1 est la droite vectorielle engendrée par $v = (1, -1, 1)$.

$$\boxed{\text{Le sous-espace propre associé à } -1 \text{ est la droite vectorielle engendrée par } v = (1, -1, 1).}$$

• Pour $\lambda = 2$, le système précédent équivaut à

$$\begin{cases} -2z & +y & = & 0 \\ x & -2y & = & 0 \end{cases}$$

soit encore à

$$\begin{cases} z = \dfrac{1}{2}y \\ x = 2y \end{cases}$$

On voit alors que le sous-espace propre associé à 2 est la droite vectorielle engendrée par $(4, 2, 1)$.

$$\boxed{\text{Le sous-espace propre associé à 2 est la droite vectorielle engendrée par } w = (4, 2, 1).}$$

La famille (u, v, w) étant formée de vecteurs propres de A associés aux valeurs propres $1, -1, 2$, on en déduit qu'il s'agit d'une base de $\mathbb{R}^3$ (puisque ces valeurs propres sont deux à deux distinctes) dans laquelle la matrice de A est

$$D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

Si P désigne alors la matrice de passage de la base canonique à (u,v,w), soit $P=\begin{pmatrix}1&1&4\\1&-1&2\\1&1&1\end{pmatrix}$, on a alors :

$$A=PDP^{-1}.$$

D'où le résultat.

$$\boxed{P=\begin{pmatrix}1&1&4\\1&-1&2\\1&1&1\end{pmatrix} \text{ convient.}}$$

3. On a

$$Y'=AY=PDP^{-1}Y.$$

On obtient alors directement le résultat en multipliant cette égalité par P^{-1} à gauche.

$$\boxed{P^{-1}Y'=DP^{-1}Y}$$

4. 4.**a)** On a $Z=P^{-1}Y$. Comme P^{-1} est une matrice constante par-rapport à x, Z est donc un vecteur colonne constitué de combinaisons linéaires des fonctions y, y' et y'', qui sont toutes $\mathcal{C}^1$. $\boxed{\text{Les coordonnées de } Z \text{ sont donc également } \mathcal{C}^1.}$

De plus, si on dérive toutes ces combinaisons linéaires, la fonction y devient y', y' devient y'' et y'' devient $y^{(3)}$. Les constantes ne bougent pas. De plus, les combinaisons linéaires issues de $P^{-1}Y$ sont donc exactement les mêmes que celles issues de $P^{-1}Y'$. Ainsi, on obtient

$$\boxed{Z'=P^{-1}Y'.}$$

4.**b)** D'après la question précédente, comme $Z'=P^{-1}Y'$, $Z=P^{-1}Y$ et $P^{-1}Y'=DP^{-1}Y$, on obtient donc bien

$$Z'=DZ$$

dont on déduit directement

$$\boxed{\begin{cases}z_1'=z_1\\z_2'=-z_2\\z_3'=2z_3\end{cases}}$$

puis, par propriété de cours, il existe $\lambda_1,\lambda_2,\lambda_3\in\mathbb{R}$ tels que, pour tout $x\in\mathbb{R}$,

$$\boxed{\begin{cases}z_1(x)=\lambda_1e^{x}\\z_2(x)=\lambda_2e^{-x}\\z_3(x)=\lambda_3e^{2x}\end{cases}}$$

5. 5.**a)** On vient de déterminer l'ensemble des vecteurs Z possibles si y est solution de l'équation différentielle. Or, on sait également par définition, que

$$Z=P^{-1}Y$$

ce qui signifie également que

$$Y=PZ.$$

Comme on connait Z et que la fonction y est donnée par la troisième composante du vecteur Y, il suffit d'observer la troisième ligne du produit PZ pour obtenir y. On obtient qu'il existe $\lambda_1,\lambda_2,\lambda_3\in\mathbb{R}$ tels que

$$y(x)=\lambda_1e^{x}+\lambda_2e^{-x}+\lambda_3e^{2x}$$

On vient de prouver que

$$S\subset Vect(x\mapsto e^{x},x\mapsto e^{-x},x\mapsto e^{2x}).$$

Réciproquement, il reste à vérifier toute fonction de la forme $y(x)=\lambda_1e^{x}+\lambda_2e^{-x}+\lambda_3e^{2x}$ est bien une solution de (ε_3'), ce qui se fait trivialement : On a

$$\begin{cases}y'(x)&=\lambda_1e^{x}-\lambda_2e^{-x}+2\lambda_3e^{2x}\\y''(x)&=\lambda_1e^{x}+\lambda_2e^{-x}+4\lambda_3e^{2x}\\y^{(3)}(x)&=\lambda_1e^{x}-\lambda_2e^{-x}+8\lambda_3e^{2x}\end{cases}$$

D'où

$$\begin{aligned}y^{(3)}(x)-2y''(x)-y'(x)+2y(x)&=\left(\lambda_1e^{x}-\lambda_2e^{-x}+8\lambda_3e^{2x}\right)\\&\quad-2\left(\lambda_1e^{x}+\lambda_2e^{-x}+4\lambda_3e^{2x}\right)\\&\quad-\left(\lambda_1e^{x}-\lambda_2e^{-x}+2\lambda_3e^{2x}\right)\\&\quad+2\left(\lambda_1e^{x}+\lambda_2e^{-x}+\lambda_3e^{2x}x\right)\\&=0.\end{aligned}$$

D'où le résultat.

$$\boxed{S=Vect(x\mapsto e^{-x},x\mapsto e^{2x},x\mapsto e^{x})}$$

5.**b)** On déduit du résultat ci-dessus que S est le sous-espace vectoriel de $C^1(\mathbb{R},\mathbb{R})$ engendré par $x\mapsto e^{-x},x\mapsto e^{2x},x\mapsto e^{x}$. Pour démontrer que celui-ci est de dimension 3, il suffit de prouver que la famille $\left(x\mapsto e^{-x},x\mapsto e^{2x},x\mapsto e^{x}\right)$ est libre.

On pose $\alpha, \beta, \gamma \in \mathbb{R}$ tels que, pour tout $x \in \mathbb{R}$, on a

$$\alpha e^{-x} + \beta e^{2x} + \gamma e^{x} = 0 \qquad (*)$$

Quand x tend vers $-\infty$, le membre de gauche doit admettre une limite, ce qui n'est la cas que si

$$\alpha = 0$$

En dérivant ensuite $(*)$, on obtient

$$2\beta e^{2x} + \gamma e^{x} = 0 \qquad (*_2)$$

En combinant $(*)$ et $(*_2)$ en $x = 0$, on obtient

$$\begin{cases} \beta + \gamma = 0 \\ 2\beta + \gamma = 0 \end{cases}$$

de solution unique $\beta = \gamma = 0$ Conclusion : La famille est libre et

$$\boxed{\dim S = 3}$$

Problème 2 :

1. Pour calculer la norme de M, on utilise le principe de calcul du maximum : on initialise une variable `maxi` à n'importe quelle valeur de M (en valeur absolue) ; au plus simple : $|M[0,0]|$.
Ensuite, on parcours M grâce à une double boucle. La valeur `maxi` est donc au final la borne maximale de toutes les valeurs absolues des coefficients de M. On propose donc :

```
1  def Norme(M):
2      (n,m)=shape(M)  # récupération de la taille de
           M
3      maxi=abs(M[0,0])    # initialisation du maximum
4      for i in range(n):  # on parcourt les colonnes
           de M
5          for j in range(m):  # on parcourt les
               lignes de M
6              if abs(M[i,j])>maxi:    # on modifie le
                   max si on en trouve un plus grand
7                  maxi=abs(M[i,j])
8      # on a pourcouru toute la matrice. Ainsi, maxi
           est bien le maximum cherché.
9      return(maxi)
```

2. Il s'agit tout simplement de diviser les valeurs de v par la norme :

```
1  def Normalise(v):
2      return(v/Norme(v))
```

3. On propose :

```
1  def PuissanceIteree(A,n):
2      (p,m)=shape(A)  # on récupère la taille de A. p
           : nb de lignes et m : nb de colonnes
3      if p!=m:            # si A n'est pas une
           matrice carrée
4          print("la matrice A doit être carrée")
5      else:               # si A est carrée
6          v=matrix(random.rand(p,1))  # on créer un
               vecteur v=v_0 aléatoire
7          for i in range(n):      # on calcule v_{i
               +1} à partir de v_i pour i de 0 à n-1
8              v=Normalise(A*v)
9          # à la fin de la boucle, on obtient v_n
10     return(v)
```

4. En observant les programmes, on constate que

$$\boxed{\text{Les programmes A et C sont corrects (et similaires).}}$$

$$\boxed{\text{Le programme B est incorrect.}}$$

En effet, dans le programme B, *ecart* est calculé avant la boucle *while* mais n'est pas recalculée ensuite. Par conséquent, si la boucle démarre, elle sera infinie !