



TP5 : Itérations et paramétrage (2/2)

1 Equation logistique sous Excel

L'équation logistique modélise l'évolution temporelle d'une population et présente une forte sensibilité aux conditions initiales. Mathématiquement, celle-ci s'écrit :

$$P_{t+1} = P_t + (F_croiss * P_t * (1 - P_t))$$

P_t correspondant au terme « *population ancienne* », ou plus justement, population au temps t .

P_{t+1} correspondant à ce qui a été appelé « *population nouvelle* », que l'on redéfini comme population au temps $t+1$.

Nous considérerons que P_t est exprimé en fréquence et varie dans l'intervalle $[0,1]$.

Deux paramètres doivent être prédéfinis :

- la « *population initiale* » (au temps $t=0$) : P_{t0} , et
- le facteur de croissance (ou taux de croissance, appelé F_croiss).

Connaissant ces deux constantes, il est possible de calculer la population à $t = 1$, puis par itérations successives, $t = 2, 3, 4, \dots$ puis de représenter l'évolution graphique de la population au cours du temps.

L'évolution d'une population (animale, cellulaire ...) selon le modèle de l'équation logistique dépend principalement de ces deux facteurs.

Nous vous proposons d'étudier le comportement de l'équation logistique en fonctions des valeurs données à ces deux facteurs, on parlera de « conditions initiales », en utilisant un tableur comme Excel.

L'objectif général du travail présenté dans ce TP est de se donner les moyens de comparer les courbes d'évolution en fonction de différentes conditions initiales. L'idée est de calculer automatiquement 6 courbes d'évolution (une sur chaque page d'un fichier Excel) et de les représenter graphiquement.

Pour cela, nous allons

1. créer des macros simples pour calculer les données nous permettant de tracer une courbe,
2. paramétrer une macro permettant de dessiner la courbe,
3. généraliser les macros précédemment créées pour construire automatiquement les 6 courbes.

N'oubliez pas d'enregistrer régulièrement votre travail.

1.1 Création de macros simples

L'objectif de cette partie (Exercices 1.1 à 1.5) est de créer le jeu de données suivant sur une feuille de calcul vierge.

	A	B	C	D	E
1				Taux de croissance	0,5
2					
3	Temps	Population			
4	0	0,01			
5	1	0,01495			
6	2	0,02231325			
...
103	99	1			
104	100	1			

Exercice 1.1 : Création d'un compteur : écriture de la macro « *compteur* »

Créer une macro compteur, qui initialise les cellules de la colonne 1 entre les lignes 4 et 104 comme sur le tableau précédent.

Enregistrer votre travail :

dans votre répertoire *INF112/EqLogistique* sous le nom *Logistique.xls*

Exercice 1.2 : Création d'une macro « *CalculPopulation* » pour le calcul de la population

Le calcul de la population peut-être réalisé de façon similaire au calcul des valeurs successives du compteur de temps. On crée une action, dans laquelle on initialise la cellule E1 (*taux de croissance*) et la cellule B4 (*population initiale*), puis on calcule la valeur des cellules B5 à B104 avec une itération.

On choisira: 0,01 pour la « *population initiale* » et 0,5 pour le « *taux de croissance* »

A) Complétez l'algorithme ci-dessous :

```

Action « CalculPopulation »
Début
  {initialisation E1 et B4}
    Cellule(      ) ←
    Cellule(      ) ←
  {Calcul des cellules B5 à B104}
    Pour i = 5 jusqu'à 104 faire
      Cellule(      ) ←
    Fin pour
Fin

```

B) implantez la macro en VBA puis exécutez là. Vérifiez le résultat obtenu.

Exercice 1.3 : Création d'une macro « *Legendes* »

Créez une macro « *Legendes* » afin d'initialiser les cellules D1, A3 et B3 avec les textes adéquats correspondants aux légendes du tableau. Contrôlez votre macro en l'exécutant.

Exercice 1.4 : Réalisation d'une macro « *Data* »

A) Transformez la macro « *CalculPopulation()* » pour que la population initiale et le taux de croissance soient des paramètres. Pour cela, on remplacera les valeurs 0.01 et 0.5 respectivement par les deux noms de variables de type DOUBLE et nommées respectivement *pop0* et *taux0*.

B) On souhaite réaliser une macro *Data()* (sans paramètre), qui appelle successivement les macros « *Legendes()* », « *Compteur()* » et « *CalculPopulation()* ».

- Complétez l'algorithme ci-dessous et réalisez une macro,
- Implémentez en VBA, vérifiez le résultat.

```

Action « Data »
Début
  {déclaration de variables}
  Taux0 : Double
  pop0 : Double

  {initialisation des variables}
  taux0 ← 0,5
  pop0 ← 0,01
  Legendes()
  Compteur()
  CalculPop( _____ , _____ )
Fin

```

Exercice 1.5 : Observation du code de la macro « *Graphe* »

- Sur le BV, récupérez le document macroGraphe.doc.
- Copiez le contenu de ce document dans l'éditeur de macro.
- Exécutez la macro « *Graphe* » et observez le résultat. Que fait la macro ?
- Observez le code de la macro « *Graphe* » et l'algorithme correspondant donné ci-dessous.

<p>Algo « Graphe » Début</p> <p>Sélectionner la zone B4..B104 Menu Insertion graphique Courbes (Affiche une tendance dans le temps) vérification plage de données</p> <p>nommer "Tc:0,5" la série Insérer le graphique en tant qu'objet dans la feuille 1</p> <p>Avec le graphique : Le graphique a un titre Titre de graphique(Evolution démographique) L'axe des abscisses à un nom Nomme les abscisses (Temps) L'axe des ordonnées à un nom Nomme les ordonnées (Fréquence) Quitte Avec.</p> <p>Avec l'abscisse du graphique suppression du quadrillage majeur suivant x suppression du quadrillage mineur suivant x Quitte Avec.</p> <p>Avec les axes du graphique suppression du quadrillage majeur suivant y suppression du quadrillage mineur suivant y Quitte Avec</p> <p>le graphique a une légende Sélectionne la légende Positionne la légende en bas du graphique</p> <p>Fin</p>	<p>Sub Graphe()</p> <p>Range("B4:B104").Select Charts.Add ActiveChart.ChartType = xlLine ActiveChart.SetSourceData Source:=Sheets("Feuil1") .Range("B4:B104"), PlotBy :=xlColumns ActiveChart.SeriesCollection(1).Name = ""Tc:0,5"" ActiveChart.Location.Where:=xlLocationAsObject.Name:="Feuil1"</p> <p>With ActiveChart .HasTitle = True .ChartTitle.Characters.Text = "Evolution démographique" .Axes(xlCategory, xlPrimary).HasTitle = True .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Temps" .Axes(xlValue, xlPrimary).HasTitle = True .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Fréquence" End With</p> <p>With ActiveChart.Axes(xlCategory) .HasMajorGridlines = False .HasMinorGridlines = False End With</p> <p>With ActiveChart.Axes(xlValue) .HasMajorGridlines = False .HasMinorGridlines = False End With</p> <p>ActiveChart.HasLegend = True ActiveChart.Legend.Select Selection.Position = xlBottom</p> <p>End Sub</p>
--	---

1.2 Modélisation de différentes évolutions démographiques

Exercice 1.6 : Simulation pour un taux de croissance de 2

- Modifiez dans votre macro « Data » la valeur du taux de croissance : remplacez 0,5 par 2.
- Enregistrez votre modification. Exécutez les macros « Data » puis « Graphe ».
- Que remarquez vous ?

On peut ainsi tester toutes les conditions que vous désirez.

Mais : Pour tester une nouvelle condition, il faut modifier le code. De plus, afin que tous les graphes ne se superposent pas et que les légendes soient réactualisées en fonction des nouvelles conditions. Il faudrait :

- Introduire une itération pour exécuter plusieurs fois ces mêmes actions,
- Afficher chaque résultat dans une feuille différente.

Dans la suite, on choisit de faire les calculs pour 6 taux différents. **Il faut donc 6 feuilles de calcul.**

Exercice 1.7 : Créer et nommer de nouvelles feuilles de calcul.

A) Créez les feuilles de calcul nécessaires. Dans le menu « Insertion », et choisir « feuille ».

Les noms des 6 feuilles doivent avoir la même forme du type « Texte*i* » (i sera utilisé pour la construction de l'itération). On choisira de nommer les 6 feuilles : « Feuil1 », « Feuil2 », « Feuil3 » ... « Feuil6 »

B) Vérifiez le nom des feuilles. Les renommer si nécessaire : cliquez avec le bouton droit de la souris sur l'onglet correspondant portant le nom de la feuille et renommez.

Exercice 1.8 : Mise en place d'une itération sur la macro « Data »

A chaque itération, on changera de feuille et on mettra à jour le taux de croissance selon l'exemple suivant.

```
Feuil = "Feuil" & i      ' nom donné à la feuille pour la condition testée
Sheets(Feuil).Select   ' choix de la feuille correspondant à la condition testée
taux = taux * i        ' calcul du taux pour la nouvelle condition testée
```

A) Compléter l'algorithme suivant, selon les propositions ci-dessus énoncées.

```
Action « Data »
Début
  {déclaration de variables}
  taux : Double
  pop : Double

  {initialisation des variables}
  taux ← 0,5
  pop ← 0,01

  Legendes()
  Compteur()
  CalculPop( _____ , _____ )

Fin
```

B) Effectuez les modifications dans le code. Cela comprend

- déclaration des variables,
- initialisation des constantes,
- positionnement de la boucle for,
- ajout des étapes de calcul.

C) Vérifiez et corrigez si nécessaire.

Exercice 1.9 : Prise en compte de la macro « Graphe »

A) Dans la macro « Data », ajoutez dans l'itération l'appel à la méthode « Graphe ». Exécutez « Data ».

Que se passe-t-il ?

B) On souhaite dessiner chaque graphe dans la page correspondante. De plus, on souhaite que les graphes soient correctement légendés. Pour cela, il faut paramétrer la macro « Graphe » avec le **nom de la série** et la **feuille** sur laquelle se dessine le graphe. Les modifications sont présentées ci-dessous sous forme algorithmique et VBA.

<p>Algo Graphe (<i>Serie : Chaine, Feuille : Chaine</i>) Début</p> <p><i>Sélectionner la zone B4..B104</i> <i>Menu Insertion graphique</i> <i>Courbes (Affiche une tendance dans le temps)</i> <i>vérification plage de données</i></p> <p><i>nommer la série Serie</i> <i>Insérer le graphique en tant qu'objet dans la feuille Feuille</i></p> <p><u>Avec le graphique</u> : <i>Le graphique a un titre</i> <i>Titre de graphique (Evolution démographique)</i> <i>L'axe des abscisses à un nom</i> <i>Nomme les abscisses (Temps)</i> <i>L'axe des ordonnées à un nom</i> <i>Nomme les ordonnées (Fréquence)</i> <i>Quitte Avec.</i></p> <p><u>Avec l'abscisse du graphique</u> <i>suppression du quadrillage majeur suivant x</i> <i>suppression du quadrillage mineur suivant y</i> <i>Quitte Avec.</i></p> <p><u>Avec les axes du graphique</u> <i>suppression du quadrillage majeur suivant y</i> <i>suppression du quadrillage mineur suivant x</i> <i>Quitte Avec</i></p> <p><i>le graphique a une légende</i> <i>Sélectionne la légende</i> <i>Positionne la légende en bas du graphique</i></p> <p>Fin</p>	<p>Sub Graphe (<i>Serie As String, Feuille As String</i>)</p> <p>Range("B4:B104").Select Charts.Add ActiveChart.ChartType = xlLine ActiveChart.SetSourceData Source:=Sheets(<i>Feuille</i>) .Range("B4:B104"), PlotBy :=xlColumns ActiveChart.SeriesCollection(1).Name = "<i>Serie</i>" ActiveChart.Location.Where:=xlLocationAsObject.Name:=<i>Feuille</i></p> <p>With ActiveChart .HasTitle = True .ChartTitle.Characters.Text = "Evolution démographique" .Axes(xlCategory, xlPrimary).HasTitle = True .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Temps" .Axes(xlValue, xlPrimary).HasTitle = True .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Fréquence" End With</p> <p>With ActiveChart.Axes(xlCategory) .HasMajorGridlines = False .HasMinorGridlines = False End With</p> <p>With ActiveChart.Axes(xlValue) .HasMajorGridlines = False .HasMinorGridlines = False End With</p> <p>ActiveChart.HasLegend = True ActiveChart.Legend.Select Selection.Position = xlBottom</p> <p>End Sub</p>
--	---

C) Compléter l'algorithme ci-dessous, pour prendre en compte les nouveaux paramètres de « Graphe »
Comme nous avons fait dans l'exercice précédent, il faut déclarer les variables utiles pour l'appel de la macro.

```

Action « Data »
Début
  {déclaration de variables}
  i : entier
  taux : Double
  pop : Double
  uneFeuille : _____
  uneSerie : _____

  {initialisation des variables}
  taux ← 0,5
  pop ← 0,01

  Pour i = 1 à _____
    uneFeuille = _____
    uneSerie = "Tc : " & _____
    Selectionner(uneFeuille)

    Legendes()
    Compteur()
    CalculPop( _____ , _____ )
    Graphe( _____ , _____ )

  Fin Pour

Fin

```

Exercice 1.10 : Autres tests

Modifiez le code pour que les calculs soient effectués pour un taux de croissance variant entre 2.6 et 2.7 avec un pas de 0.02 (2.60, 2.62, 2.64, ..., 3.0). Observez les modifications entre les différents graphiques.

2 Analyse d'une séquence d'ADN sous Excel

Nous allons maintenant concevoir un algorithme et une macro afin d'analyser une séquence d'ADN disposée dans un tableau Excel. Il s'agit de compter:

- le nombre total de bases,
- le nombre de A – exprimer la proportion de A en pourcentage,
- le nombre de T – exprimer la proportion de T en pourcentage,
- le nombre de C – exprimer la proportion de C en pourcentage,
- le nombre de G – exprimer la proportion de G en pourcentage.

Le travail sera réalisé sous Excel à partir du fichier **ADN.xls** (disponible sur le BV et à sauvegarder sur Z)

On pourra utiliser des variables compteurs qui permettront de compter le nombre total de bases, le nombre de bases A, T, C et G (que l'on peut nommer *cpttotal*, *cptA*, *cptT*, *cptC*, *cptG*). N'oubliez pas qu'un compteur de dénombrement doit être toujours initialisé à zéro.

L'algorithme se déroulera selon un schéma de deux itérations imbriquées qui assureront le balayage de votre tableau :

```

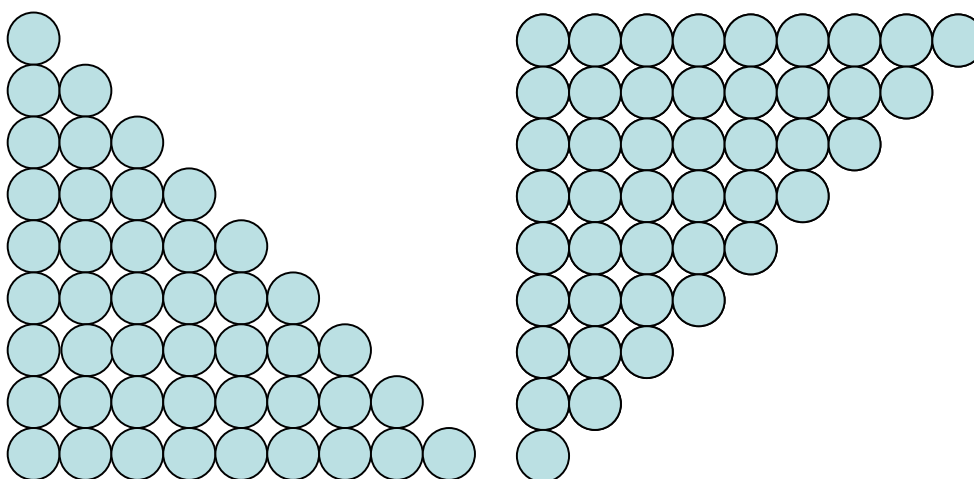
Pour i de 1 à NbLignes
  pour j de 1 à NbColonnes
    traitement
  fin pour j
fin pour i

```

Le **traitement** consistera à lire le contenu de la cellule active et à examiner ce contenu. Si le contenu est un « A » alors on augmente de 1 le compteur de « A » (cptA), si c'est un « T » on augmente de 1 le compteur de « T » (cptT)... etc. On terminera par l'affichage des résultats des compteurs dans différentes cellules, puis le pourcentage de chacune des bases. On n'oubliera pas de légender le tableau ainsi produit.

- A) Proposez un algorithme
- B) Implantez sous Excel
- C) Exécutez et vérifiez la cohérence des résultats.

3 Dessins sous PowerPoint



Exercice 3.1 : Premier escalier

- A) Proposez un algorithme pour réaliser le premier escalier (représentation de gauche ci-dessus).
- Remarque : sur la $i^{\text{ème}}$ ligne, il y a i éléments.
- B) Implantez l'algorithme en VBA

Exercice 3.2 : Second escalier

- A) Proposez un algorithme pour réaliser le second escalier (représentation de droite ci-dessus).
- Remarque : c'est un escalier qui comporte L lignes. Sur la $i^{\text{ème}}$ ligne, il y a $L+1-i$ éléments.
- B) Implantez l'algorithme en VBA

4 Fonctions du second degré.

Une fonction du second degré peut s'écrire sous la forme générale $y=f(x) = ax^2 + bx+c$

Le but de l'exercice est de construire un algorithme (dont l'implantation donnera lieu à une macro sous Excel) pour calculer des valeurs pour une ou plusieurs fonctions du second degré.

Le principe est le suivant. Dans la colonne 1, on inscrit les valeurs de x , entre 0 et 10 de 0,1 en 0,1. Dans la colonne 2, on calcule les valeurs de $f(x)$ correspondantes.

Ci-contre,

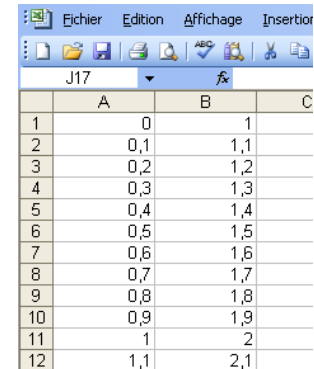
on donne un exemple du résultat attendu pour $f(x) = 0*x^2 + 1*x + 1$

On propose de faire une action qui, dans un premier temps, calcule les valeurs pour la première colonne, puis dans un second temps, calcule les valeurs de la seconde colonne pour la fonction $f(x) = 3*x^2 - 2*x + 1$.

Exercice 4.1 : CalculerValeursFonction

A) Complétez l'algorithme ci-dessous.

B) Implémentez en VBA, testez et vérifiez le résultat.



	A	B	C
1	0	1	
2	0,1	1,1	
3	0,2	1,2	
4	0,3	1,3	
5	0,4	1,4	
6	0,5	1,5	
7	0,6	1,6	
8	0,7	1,7	
9	0,8	1,8	
10	0,9	1,9	
11	1	2	
12	1,1	2,1	

Action CalculerValeursFonction

1. Début
2. i : entier {compteur}
3. a, b, c : entiers {coefficients}
4. {initialisations}
5. $a \leftarrow 3$
6. $b \leftarrow 2$
7. $c \leftarrow 1$
8. _____
9. {itération pour la première colonne}
10. Pour i _____ jusqu'à _____ faire
11. Cellule(i , 1) \leftarrow _____
12. Fin Pour
13. _____
14. {itération pour la deuxième colonne}
15. Pour i _____ jusqu'à _____ faire
16. Cellule(i , 2) \leftarrow _____
17. Fin Pour
18. Fin action

Exercice 4.2 : Paramétrage

A) Transformez l'algorithme *CalculerValeursFonction* en une action *CalculerValeursFonctionParam* pour que les variables **a**, **b**, et **c** (coefficients de la fonction calculée) deviennent des paramètres.

Réaliser l'action qui permet de calculer $f(x) = 3x^2 - 2x + 1$ en utilisant *CalculerValeursFonctionParam*

B) Implémenter en VBA, tester et vérifier le résultat.

5 Suite de Lucas

Reprendre le TD5.

Implantez une macro pour calculer les éléments de la suite de Lucas, à l'aide d'une macro paramétrée.