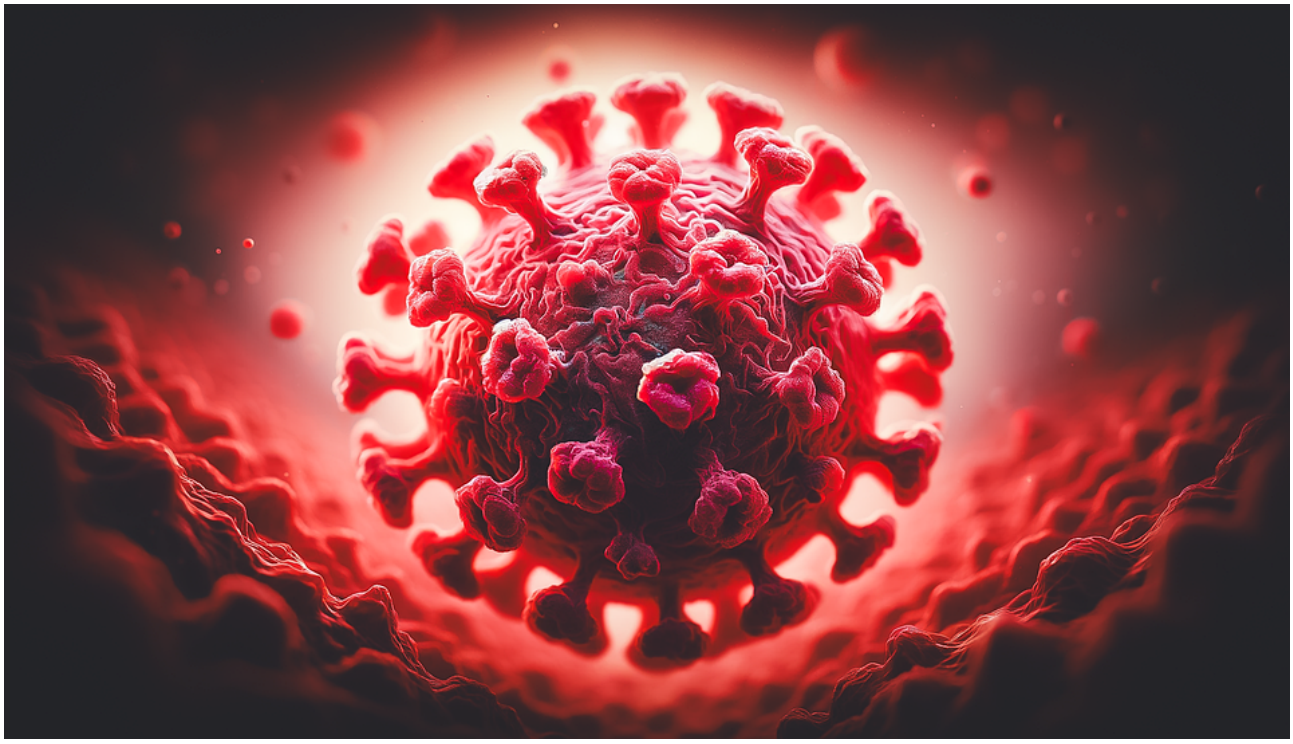
,

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

Projet de Physique P6
STPI/P6/2024 -29

Simulation d'épidémies



Étudiants :
Foucauld Bourdon
Alexandre Clergeaud-Martinez
Adrien His
Anastasia Laudet
Elisa Picou
Amr Zaki Salih

Enseignant-responsable du projet :
Julien Saunier

Date de remise du rapport : 17/06/2024

Référence du projet : STPI/P6/2024 – 29

Intitulé du projet : Simulation d'épidémies

Type de projet : Simulation informatique

Objectifs du projet :

Les objectifs du projet sont :

— la modélisation de la façon la plus réaliste possible une épidémie
— la compréhension de la propagation et les facteurs qui jouent sur l'influence d'une maladie dans une population donnée
— l'évaluation de l'efficacité des différentes stratégies et ainsi de tester des hypothèses sur les comportements des individus
— la découverte et l'obtention des équations mathématiques qui régissent le modèle
— manipulation du langage Python

Mots-clés du projet :

— Modélisation mathématique et informatique
— Propagation
— Approximation
— Fiabilité des modèles

Institut National des Sciences Appliquées de Rouen
Département Sciences et Techniques Pour l'Ingénieur
685 Avenue de l'Université BP 08 - 76801 Saint-Étienne-du-Rouvray
Tél : +33 2 32 95 66 21 - Fax : +33 2 32 95 66 31

Table des matières

Introduction

L'étude et la simulation des épidémies sont des domaines cruciaux pour la santé publique, la médecine et la gestion des crises sanitaires. Avec l'augmentation de la densité de population, la globalisation accrue des échanges et le changement climatique, le monde est de plus en plus vulnérable à l'émergence de nouvelles maladies infectieuses. Ces facteurs contribuent à la propagation rapide des agents pathogènes, rendant impérative la capacité à modéliser et prédire les dynamiques épidémiques. Cette capacité permet non seulement de comprendre comment les maladies se diffusent, mais aussi de planifier des interventions efficaces pour les contenir.

La modélisation des épidémies est une discipline qui mobilise des concepts issus des mathématiques, des statistiques, de l'informatique et des sciences sociales. Elle s'appuie sur des modèles théoriques et des simulations informatiques pour étudier la propagation des maladies infectieuses et évaluer les impacts de différentes stratégies de contrôle. En combinant ces diverses approches, les chercheurs peuvent anticiper les évolutions possibles d'une épidémie et proposer des mesures préventives et curatives adaptées.

Au cœur de l'étude des épidémies se trouve une question centrale : comment les maladies infectieuses se propagent-elles dans une population donnée ? Peut on vraiment prédire l'évolution d'une épidémie ?

Dans ce projet, nous explorerons deux approches fondamentales pour comprendre et analyser les dynamiques des épidémies. Tout d'abord, nous exposerons en détail le modèle SIR en soulignant ses principes fondamentaux, ses hypothèses et ses limites. Cette présentation sera illustrée par des graphiques que nous avons créés. Ensuite, nous nous tournerons vers les simulations probabilistes, en expliquant le fonctionnement de notre code Python.

À l'aide de graphiques et de données simulées, nous examinerons comment ces deux approches – le modèle SIR et les simulations probabilistes – nous aident à mieux comprendre le comportement des épidémies et à évaluer l'efficacité des différentes stratégies de contrôle. Enfin, nous essayerons d'analyser nos résultats. Nous aborderons également les défis à relever pour améliorer les modèles existants et développer de nouvelles méthodes pour la surveillance et la gestion des épidémies à l'échelle mondiale.

Ce projet vise donc à fournir une compréhension approfondie des outils et des techniques utilisés pour modéliser et simuler les épidémies, afin de mieux préparer les professionnels de la santé et les décideurs à faire face aux crises sanitaires futures.

Organisation du travail

0.1 Organisation adoptée pour le déroulement du travail

Nous avons choisi de coder le projet en langage Python pour plusieurs raisons. Sa richesse en bibliothèques spécialisées offre une vaste gamme de fonctionnalités pour la manipulation des données, les calculs scientifiques et la visualisation des résultats. En combinant ce langage avec des techniques de modélisation probabiliste, nous sommes en mesure de créer des simulations flexibles et personnalisables, ce qui les rend claires et lisibles.

Nous avons au début du projet effectué un cahier des charges dans lequel nous avons spécifié les dates des différentes versions comme suit :

1. 23/02/24 une version de base comprenant le modèle de base SIR(avec les personnes décédées et sans) et probabiliste avec un affichage graphique.
2. 18/03/24 une amélioration du programme (régler problèmes rencontrés, ajouter les déplacements et réaliser des recherches sur ceux ci, sliders etc).
3. 15/04/24 de faire plusieurs boîtes correspondant aux différents pays et d'ajouter des mesures de confinement et de mises en quarantaine (cas contact)
4. 13/05/24 d'incorporer vaccination hygiène (travail de recherche)
5. 27/05/24 de faire les recherches, les dernières améliorations possibles et de commencer à rédiger le rapport avec les interprétations des graphiques obtenus. On réalisera des tests en continu en faisant varier les différents paramètres
6. 03/06/24 d'avoir à ce jour une version finale du code et du rapport.

0.2 Organigramme des tâches réalisées et des étudiants concernés

Nous nous sommes répartis les tâches comme suit :

Vous trouverez la réparition des tâches sous forme d'organigramme à la fin du rapport.

Foucauld :
- Travail sur la simulation probabiliste
- Base du code ModelTest
- Menu
- Gestion du Temps
- Gestion de l'emploi du temps (génération, attribution et lecture)
- Confinement
- Amélioration de l'affichage, simulation et données.
- Rédaction du rapport (Partie Probabiliste)

Alexandre :
- Travail sur la simulation théorique
- Assistance débogage
- Correction du code

Adrien :
- Travail sur la simulation probabiliste
- Affichage
- Implémentation mesures de confinement
- Amélioration du programme

Anastasia :

- Travail sur la simulation théorique
- Modélisation SIR et code associé
- Rédaction Rapport LateX

Elisa :

- Travail sur la simulation probabiliste
- Modification de ModelTest (ajout adresse et amélioration de la performance du code)
- Simulation de la première version du code GestionTemps

Amr :

- Travail sur la simulation probabiliste
- Gestion des lieux
- Gestion répartition

Chapitre 1

Présentation du Modèle Théorique et du Code associé

1.1 Le modèle SIR

Ce modèle a été inventé par 3 chercheurs : Soper, Kermack et McKendrick en 1924.

Leur modèle est toujours applicable aujourd'hui et est considéré comme valide. A l'époque, il a été trouvé par des chercheurs qui essayaient de comprendre pourquoi la grande pandémie de grippe espagnole de 1918 n'avait pas infecté toute la population.

Le modèle SIR est l'un des modèles mathématiques les plus fondamentaux pour représenter une épidémie. Ce modèle divise la population en 3 catégories : les personnes susceptibles d'être infectées (S), les personnes infectées et contagieuses (I), et celles qui se sont rétablies de la maladie et qui sont immunisées (R). Annexe 1

Les équations qui régissent le modèle sont :

$$\begin{cases} \frac{dR}{dt} = \gamma I \\ \frac{dI}{dt} = \frac{\beta IS}{N} - \gamma I \\ \frac{dS}{dt} = \frac{-\beta IS}{N} \end{cases}$$

Nous obtenons alors la courbe suivante : Annexe 2

On peut de plus compléter ce modèle en ajoutant la proportion de personnes décédées. Celles-ci passent forcément par la catégorie «Infectées» car elles ne transmettent plus la maladie. Les équations différentielles sont légèrement modifiées comme suit :

$$\begin{cases} \frac{dR}{dt} = \gamma I \\ \frac{dI}{dt} = \frac{\beta IS}{N} - \gamma I - mI \\ \frac{dS}{dt} = \frac{-\beta IS}{N} \\ \frac{dD}{dt} = mI \end{cases}$$

1.2 Les hypothèses de ce modèle

Les hypothèses de ce modèle sont les suivantes :

- Les personnes passent de S à I en fonction du nombre de personnes infectées et de la proportion de la population qui est sensible.
- Un pourcentage constant de personnes passent de I à R chaque jour .
- Un modèle utilisant des valeurs moyennes permet de faire des prédictions significatives à l'échelle d'un pays.

Ces hypothèses sont manifestement erronées dans le cas de l'épidémie de Covid-19. En effet, certaines personnes sont plus isolées et contaminent moins de monde, alors que certaines professions sont en contact avec beaucoup d'autres personnes et propagent davantage la maladie.

1.3 Présentation du code Python associé

Les courbes se trouvent ici : Annexe 2

Tout d'abord on commence par importer les librairies nécessaires au fonctionnement du code. Puis on procède à l'initialisation des différents paramètres ainsi que les tableaux dans lesquels on va stocker les différentes valeurs. On écrit après cela une fonction « simu » dans laquelle on ajoute aux tableaux les valeurs des paramètres S,I,R,D mis à jour à l'aide des équations différentielles. Ensuite, on crée une fonction « plot » d'affichage pour notre graphique.

Puis nous avons voulu améliorer le code et ajouter des « sliders » afin de pouvoir modifier les paramètres m, β et γ en même temps que le graphique, afin de voir directement sur les courbes l'influence de ces paramètres. Pour réaliser cela nous avons utilisé la bibliothèque tkinter et nous avons ajouté, en plus des fonctions déjà existantes, une fonction « update_plot » et un programme principal « main ». La première fonction sert à mettre à jour les sliders et la deuxième à les créer.

1.4 Les limites de ce modèle

Ce modèle est intéressant à étudier pour une première réponse à la problématique posée. Cependant il présente de nombreuses limites.

1. La simplicité des catégories de population : bien que le modèle SIRD ajoute une catégorie supplémentaire pour les décès, il reste une simplification de la complexité réelle des interactions humaines et des dynamiques épidémiologiques. Par exemple, il ne prend pas en compte les variations individuelles dans la susceptibilité, l'immunité acquise ou les taux de transmission.
2. Les hypothèses statiques : celui ci repose sur des hypothèses statiques telles que des taux de transmission et de récupération constants, ce qui peut ne pas refléter la réalité des épidémies dynamiques où ces paramètres varient au fil du temps en fonction de divers facteurs, tels que par exemple les interventions sanitaires et les comportements humains.
3. L'hétérogénéité de la population : ce modèle considère souvent la population comme homogène et interconnectée, ce qui néglige la structure complexe des réseaux sociaux réels. En réalité, les interactions entre individus se produisent de manière non aléatoire, ce qui peut influencer la propagation de la maladie de manière significative.
4. L'absence de facteurs géographiques : il ne tient généralement pas compte des variations géographiques dans la propagation de la maladie, ce qui peut être crucial pour comprendre les épidémies régionales ou locales où les conditions environnementales et démographiques diffèrent.

C'est pour ces différentes raisons que nous avons décidé d'étudier une deuxième approche probabiliste pour faire face à ces problèmes.

Chapitre 2

Présentation du Modèle Probabiliste

2.1 Introduction au modèle

La simulation probabiliste diffère des modèles mathématiques traditionnels, comme ceux présentés précédemment, même si elle repose sur le même principe de « compartiments » de population. En ce sens, elle se base sur des probabilités et des règles stochastiques pour simuler l'évolution de l'épidémie au lieu de s'appuyer sur des équations différentielles déterministes. Le but du modèle probabiliste est de se rapprocher des contaminations qui se passent tous les jours lors d'une épidémie.

Les simulations probabilistes offrent une approche plus flexible et réaliste en tenant compte de la variabilité individuelle, des interactions sociales et des caractéristiques spécifiques de la maladie. Cependant, il est important de rappeler que c'est un modèle « jouet » avec une petite population, où les phénomènes de la réalité sont grandement simplifiés . Nous essayons de décrypter les mécanismes fondamentaux qui font l'épidémie, bien loin de la complexité des personnes réelles et des populations plus larges.

2.2 Une première version du modèle

Nous avons choisi le langage Python en raison de sa vaste gamme de bibliothèques modernes, de sa popularité qui garantit une abondance de ressources pédagogiques, ainsi que pour sa portabilité et sa clarté. Il nous fallait, par la suite, choisir les bibliothèques adaptées à nos besoin. Après plusieurs recherches et concertations, nous avons opter pour les modules suivants :

-  Pygame : *gérer l'affichage et l'intéraction de différents élements de notre simulation.*
-  NumPy : *la gestion du « hasard », les calculs numériques.*
-  Matplotlib : *l'affichage des données sous forme de graphiques.*

Le principe de base cette simulation est simple :

- On initialise chaque individu avec une position, une vitesse aléatoire, un état (S ou I ou R ou D).
- On délimite un rectangle dans lequel va se passer la simulation, les collisions sur les bords du domaine sont élastiques.
- À chaque tic (itération) de simulation : on regarde les collisions entre les S et les I. Chaque S qui rentre en collision a une probabilité $p_{infecté}$ de devenir I.
- Pour les nouveaux infectés, on initialise un compteur qui correspond à la durée de la maladie. À la fin du compteur, la personnes a une certaine probabilité : $p_{Recovered}$ d'être R et donc $p_{Mort} = 1 - p_{Recovered}$ d'être D.

Pendant la durée de la simulation, définie par l'utilisateur, on affiche le tout à l'écran, avec un nombre d'images par seconde réglable, dans une fenêtre, associant une couleur pour chaque état. À la fin, des graphiques sont affichés qui représente comme leur titre l'indique, la dynamique de la population au cours du temps. Il y a donc, en abscisse : le temps et en ordonnée : le nombre d'individus pour chaque état. Deux sont générés : un "classique", nuage de points Annexe 6, et un à bandes empilées (stackplot) **??**. On visualise donc à chaque instant, t, la répartition de la population.

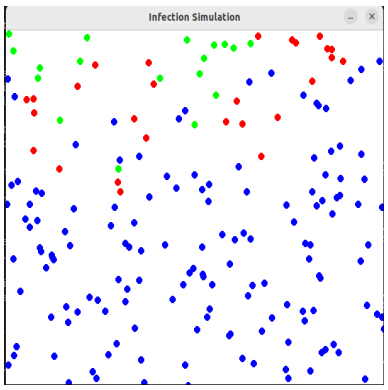


FIGURE 2.1 – La fenêtre de visualisation de la première version

Penchons nous désormais sur le code . Notre programme se présente sous la forme d'un seul fichier.py. Nous allons vous présenter ici les principaux éléments structurels et les fonctionnalités :

- **Classe *Dot*** : Représente un individu de la population. Chaque individu est un point mobile à l'écran, d'où la dénomination "dot". Avec comme expliqué précédement : une position, une vitesse et une couleur selon leur état.
- **Classe *Simulation*** : Gère la simulation dans son ensemble. Elle initialise la population avec un certain nombre d'individus susceptibles et infectés. Elle contrôle également le déroulement de la simulation au fil du temps, y compris la propagation de l'infection, les rétablissements et les décès.
- **Propagation de l'infection** : Les individus infectés peuvent transmettre [avec une probabilité de transmission] l'infection aux individus susceptibles lorsqu'ils entrent en collision.
- **Rétablissement et mortalité** : Les individus infectés ont une chance de guérir et de devenir immunisés ou de décéder après un certain nombre de cycles (déterminé par cycles_to_fate [durée de la maladie] et mortality_rate [taux de mortalité]). Les individus récupérés sont représentés en vert, tandis que les individus décédés sont retirés de la simulation.
- **Visualisation des données** : Le programme génère les deux graphiques mentionnés précédement. On a récupéré pour cela le nombre de personnes dans chaque catégorie à chaque instant.

2.3 Version comprenant différents lieux, et catégories d'individus

Nous avons décidé de mettre en place une version plus élaborée du code. On pourrait qualifier cette version de « coeur » de la version finale. La population est dorénavant répartie dans différentes « classes », aussi appelées occupations. Elles représentent différentes tranches d'âges et catégories socio-professionnelles. Cette version vit le jour, après avoir mené diverses recherches sur les répartitions démographiques et les statistiques qui en découlent. Majoritairement sur le site de l'INSEE car, nous nous basons sur des statistqiues françaises pour batir notre modèle. [4, 5, 6]

Après concertation et synthèse des différents opinions et différentes recherches de chacun : dans une optique de simplicité, il est ressorti quatre "classes" : **enfant**, **travailleur**, **sans-emploi**, **retraité**. Représentant la diversité de notre population « modèle ». Il existe aussi maintenant, des lieux, symbolisant de manière très simplifié (encore une fois), les localisation du quotidien. Nous nous sommes basés sur la routine journalière d'un citadin moyen. Par un processus similaire à celui présenté précédement, nous indentifions quatre types de lieux différents : maison, école, travail, commerce, loisir. En outre, chaque individu possède un emploi du temps personnel, attribué en fonction de sa « classe » et géneré aléatoirement à partir d'un modèle pour la « classe ».

L'emploi du temps moyen pour chaque classe (en h par semaine) :

Enfant : ÉCOLE : 36 *h*
LOISIRS : 6 *h* (Week-end)

Travailleur : TRAVAIL : 40 *h*
LOISIRS : 6 *h* (Week-end)

Sans-emploi : LOISIRS : 14 *h*

Retraité : LOISIRS : 7 *h*

Toutes les classes : COMMERCES : 3 *h*
MAISON : *le reste du temps*

Nous miniaturisons donc, en quelque sorte, une ville (sachant que pour des raison de perfromances nous ne dépassons jamais le seuil des 1000 individus). À l'initialisation, nous infectons un seul individu d'une« classe »(définie par l'utilisateur).

2.3.1 Présentation du code Python associé

Le code a été séparé en quatre sous programmes, dans quatre fichiers distincts pour améliorer la lisibilité, la modularité et utiliser l'encapsulation. Nous allons lister ces différents fichiers, lister les éléments importants qu'ils contiennent et détailler les fonctionnalités ajoutées par rapport à la version précédente.

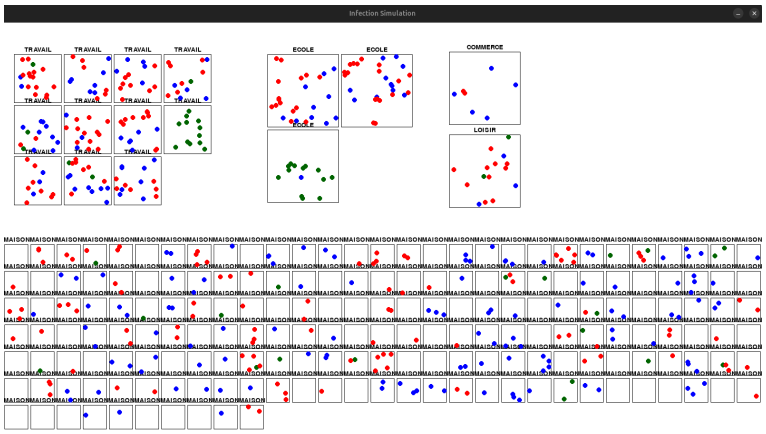


FIGURE 2.2 – La fenêtre de visualisation de la deuxième version

2.3.1.1 activites.py

— **Classe *Activity*** : représente une activité avec les attributs : *name* : le nom de l'activité, *start_time* : l'heure de début de l'activité, *end_time* : l'heure de fin de l'activité

— **Fonction *generate_ schedule*** : génère un emploi du temps en fonction de l'occupation spécifiée (enfant, travailleur, sans emploi, retraité). L'emploi du temps se présente sous la forme d'un dictionnaire. En Python, un dictionnaire est une structure de données qui associe des clés à des valeurs. Dans ce cas, les jours de la semaine (clés) sont associés à des listes qui contiennent les activités de la journée. Les activités sont triées selon leurs horaires de début dans chaque liste jour. Les horaires de début et fin d'activité sont choisis au hasard selon une loi normale centrée réduite $\mathcal{N}(0,1)$ autour d'un horaire choisi. Cet horaire peut être :

— fixe pour tous les individus d'une classe (*e.g.* début TRAVAIL : $8h$)
— choisi selon une loi uniforme entre deux horaires cibles $\mathcal{U}(h_1, h_2)$ (*e.g.* début LOISIRS entre $8h$ et $20h$)

L'emploi du temps est donc unique pour chaque individu.

2.3.1.2 lieu.py

— **Classe *Lieu*** : représente un espace dans la simulation, comme une maison, une école, un lieu de travail, un magasin ou un lieu de loisirs. Elle possède les attributs : *nom* : le nom du lieu, x et y : les coordonnées du coin supérieur gauche du lieu, *largeur* et *hauteur* : la hauteur et la largeur du lieu, *container*, un conteneur à l'intérieur duquel se trouve les dots présent dans le lieu. Une méthode, *dessiner* : permet de représenter le lieu dans la fenêtre de simulation.
— **Fonction *Créationlieu*** . créer une liste de lieux (utile quand on a besoin de construire plusieurs lieux de la même sorte (*i.e* pour les maisons, les écoles et les lieux de travail). Tout cela en fonction du nombre d'individus total et du nombre d'individus par lieu unique. Certains autres paramètres sont relatifs à l'affichage sur l'écran et ils renseignent sur les positions minimales et maximales entre lequelles pourront être créé ces lieux, évitant le chevauchement.

2.3.1.3 ModelTest.py

— **Classe *Dot*** : elle contient désormais, en plus : les attributs *occupation*, *schedule* et *lieu_actuel*.[1] Des méthodes ont également été rajoutées pour lire l'emploi du temps et changer de lieu en conséquence.
— **Classe *Simulation*** en plus, des éléménts de la version précédente, cette classe doit maintenant créer les individus des différentes classes selon les proportions spécifiées par l'utilisateur et les répartir dans les maisons, (+ leur associer un numéro de travail ou d'école pour les enfants et les travailleurs). Les lieux sont dessinés à l'écran. Le but de cette classe est aussi d'organiser les Dot dans les différentes maisons afin d'avoir une proportion définie. Le nombre de maisons crées est défini grâce à cette proportion et différentes fonctions permettent de répartir les individus intelligement. Il faut noter qu'aucun enfant ne peut se retrouver sans un parent. La boucle de la simulation est comprise dans la méthode *start*.
— ***Section principale*** : créer une instance de la classe Simulation et lance la simulation. Les paramètres de la simulation, tels que le nombre initial de personnes, les probabilités d'infection et de mortalité, sont définies ici. On appelle la méthode start de ***Simulation***. Après la fin de la simulation, comme pour la version précédente, deux graphiques sont générés pour représenter la dynamique de la population au fil du temps.

2.3.1.4 temps.py

— **Classe *Gerer_Temps*** : c'est l'unique élément que contient ce fichier. Le but de cette partie est de définir simplement un modèle de temps pour notre simulation. Ces attributs sont ainsi : *minute*, *heure*, *jour*, *semaine*. Chaque itération de notre boucle correspondera donc à une minute « fictive ». Des méthodes sont ainsi disponibles pour obtenir les attributs de la classe sous différents formats :
  — get_time_array : renvoi le temps sous forme d'un tableau [heure,minute]
  — get_time_minutes : renvoi le temps converti en minutes

2.4 Une version comprenant un affichage et une infection revue, un confinement, des informations statistiques supplémentaires

L'affichage a été revu pour permettre à l'utilisateur de mieux se repérer. La date est en haut au centre de la fenêtre. Les lieux groupés (maisons, écoles, travails) sont délimités par un rectangle épais. Le nom de la catégorie des lieux est affiché au dessus. En haut à droite se trouve à présent un graphique en direct, présentant la proportion de chaque classe en fonction du temps. Enfin, au dessus encore, il y a un bouton pour activer / desactiver le confinement.

1. D'autres attributs ont été ajoutés mais ne sont pas nécéssaires à la compréhension globale du code.

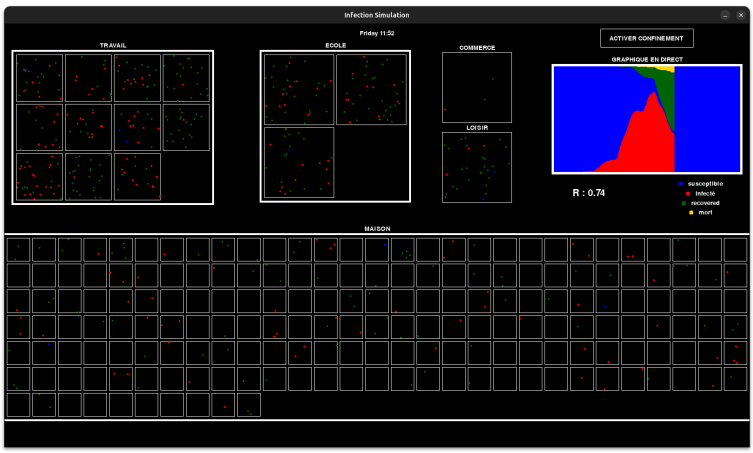


FIGURE 2.3 – La fenêtre de visualisation de la deuxième version

2.4.1 menu.py

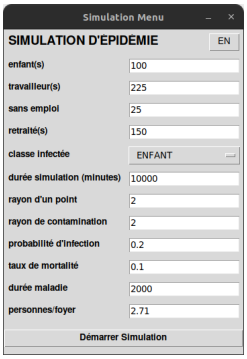


FIGURE 2.4 – Le menu en français

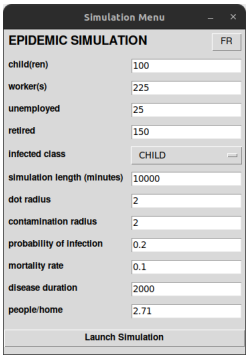
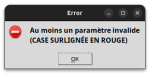


FIGURE 2.5 – Le menu en anglais

Toujours dans l'optique de rendre le programme plus accessible à l'utilisateur, il fut proposé la création d'un menu. Après avoir employé une méthode d'essais erreurs avec différentes bibliotèques Python, nous avons choisi Tkinter. Disponible en français et en anglais, il permet de choisir les différents paramètres de la simulation sans rentrer dans le code. De plus, il est verifié que le paramètre entré n'a pas une valeur abberante.

Auquel cas, un message d'erreur s'affichera : 

2.4.2 Nouveau processus d'infection

Le processus d'infection utilisait la méthode *pygame.sprite.groupcollide*, efficace en terme de complexité de calcul. Cependant, cette méthode détecte les collisions entre deux *surfaces pygame* (chacune portant un point), mais ces *surfaces* sont nécessairement rectangulaires. Ainsi, au lieu de vérifier un rayon précis autour de chaque individu, on vérifie une zone rectangulaire plus étendue. En outre, cette méthode obligeait à augmenter la taille des *surfaces* pour augmenter la zone de transmission de la maladie, ce qui n'était pas pratique.

Nous avons ainsi créé une nouvelle méthode infection_of_dots_within_radius qui prend en compte les individus dans un cercle de rayon défini partant du centre de l'individu infecté. Les individus se trouvant dans cette zone ont une certaine porbabilité de contracter la maladie. Le principe d'infection est ensuite le même que expliqué précedement.

2.4.3 Nombre effectif de reproduction : R

Effective reproduction number, en français « Nombre de reproduction effectif », dérive du concept de « nombre de reproduction de base », $\mathbf{R_0}$ qui est une notion vraiment commune en épidémiologie. Le nombre de reproduction de base $\mathbf{R_0}$ est le nombre de cas secondaires qu'un cas produirait dans une population totalement

susceptible à l'infection.[9]Ce nombre dépend de la durée de la période infectieuse, de la probabilité d'infecter un individu susceptible lors d'un contact et du nombre de nouveaux individus infectés par unité de temps. Il nous sert à quantifier la propagation d'une infection.

Le nombre R0 peut par exemple servir à calculer le seuil d'immunité collective : Ce seuil peut être défini comme : « *la proportion minimale au sein d'une population (P) qui doit être immunisée par l'infection naturelle ou par vaccination (si disponible) pour empêcher le déclenchement ou la persistance d'une épidémie* »[10]

$$P = 1 - \frac{1}{R_0}$$

On remarque clairement que plus $\mathbf{R_0}$ est grand, plus la part de la population immunisée doit être importante.

Le nombre **R** lui est dit effectif car il correspond à $\mathbf{R_0}$ mais dans une population pas totalement susceptible, qui a donc été partiellement immunisée. Il va nous servir à quantifier la propagation durant l'épidémie. Si :

— $\mathbf{R} > 1$, l'infection s'accroit de manière exponentielle : c'est une **épidémie**.
— $\mathbf{R} = 1$, l'infection est transmise de 1 à 1 : c'est **endémique**.
— $\mathbf{R} < 1$, l'infection décroit (pas de terme établi).

La façon dont R est estimé dans notre simulation est la suivante : on regarde chaque individu actuellement, on compte le nombre de personnes qu'il a infectées jusqu'à présent, et on estime le nombre total de personnes qu'il va infecter en fonction de la durée de la maladie, puis on fait ensuite la moyenne sur la population d'infectés.[11] On affiche la moyenne des derniers **R** calculés *(on prend les 20 derniers, car afficher le dernier* **R** *calculé n'aurait aucun sens puisqu'il représente la simulation sur un tic et non une tendance générale de propagation)*. On estime en fait $\mathbf{R_0}$ jusqu'au moment à partir duquel il y a des personnes immunisées à l'infection. À compter de cet instant : $\mathbf{R_0}$ devient **R**. Une méthode *Calcul_de_R* été rajoutée dans la classe *Simulation*.

En bref, cela permet à l'utilisateur d'avoir un indicateur global sur la propagation de l'épidémie.

2.4.4 Le confinement et l'affichage en direct de l'avancée de l'épidémie

Dans l'idée, l'ajout d'un confinement parrait simple. Il faut générer un nouvel emploi du temps contenant uniquement l'activité MAISON et l'activité COMMERCES et que chaque individu l'utilise en cas de confinement Dans les faits ce fut un peu plus compliqué, le processus d'initialisation de la lecture des emplois du temps n'étant pas compatible avec le fait de commencer à un autre instant que l'instant $t = 0$.

Fonction *generate_schedule_confinement* : génère un emploi du temps selon la logique précédente mais, il ne contient que des activités MAISON et LOISIRS et ne dépend pas de l'occupation.

Le fichier lieux.py contient maintenant la gestion d'un bouton confinement que l'utilisateur peut presser à n'importe quel moment de la simulation.

La classe *Dot* possède de nouvelles méthodes permettant de changer d'un emploi du temps à un autre.

Avec le développement de cette nouvelle fonctionnalité, nous nous sommes rendu compte qu'il fallait que l'utilisateur puisse voir les conséquencse de ses actions en direct. Il était nécéssaire d'implémenter un graphique en direct. Nous avons incorporé cela à la classe simulation. Il est en effet possible avec pygame de le faire, cependant il est moins précis dans la lecture que ceux générés à la fin. Il offre tout de même la possibilité d'observer les tendances.

2.5 Les difficultés rencontrées et améliorations possibles

Tout d'abord, nous pensons qu'il est impéartif de préciser que nous débutons Python. Certains d'entre nous n'avaient jamais fait de programmes dans ce langage. Il etait parfois difficile de nous organiser, car la vitesse de progression de chacun peut être drastiquement différente.

Une étape difficile fut de passer de la première version établie du modèle à la deuxième, avec les classes et les emplois du temps.

Enfin, comme souvent dans les projets, de nombreux imprévus reportent la version finale. Notamment, dans notre cas : la répartition dans les maisons.

Un exemple d'amélioration de la simulation pourrait se baser sur la compilation de recherches suivante :

Répartitions des personnes dans les foyers en France (adaptée au modèle de notre simulation)[8] :

- Personnes seules (pas un "ENFANT") : 30,8%
- Un couple (composé de "TRAVAILLEUR" et/ou "SANS_EMPLOIS") : 56,1%
  - sans enfant : 24,5%
  - avec enfant(s) : 31,6%
    - : avec enfant(s) mineur(s) (donc "ENFANT") : 79%
    - : avec enfant(s) majeur(s) (nous considérons qu'ils travaillent donc "TRAVAILLEUR") : 21%
    - + Ces couples avec des enfants mineurs ou majeurs ont soit :
      - 1 enfant : 45,2%
      - 2 enfants : 38,3%
      - 3 enfants : 12,8%
      - 4 enfants ou plus : 3,7%
- Personnes seules avec un enfant (composées de "TRAVAILLEUR" et d'un "ENFANT") : 7,6%
- Le reste est complexe : ce qui ne correspond pas à ces catégories : 5,5%

Une implémentation possible en code aurait été de transformer les lieux comme étant également des listes contenant les individus.

Nous demanderions à l'utilisateur, non pas de spécifier un nombre de personnes dans chaque classes, mais un nombre total d'individus permettant par la suite de créer les maisons nécessaires et ainsi connaître le nombre de personnes par classe.

2.6 Les limites de ce modèle

1. Notre simulation est basée certes sur des recherches en amont. Mais, les caractéritiques de celle-ci : échelle de temps, vitesse des individus, choix et taille des différents lieux, occupations choisies, emploi du temps, ... Tout cela reste des choix arbitraires.
2. Difficultés à modéliser l'hétérogénéité : nous avons fait le choix des déplacements des invividus vers les lieux suivants : foyer, travail, école, transports, commerces, loisirs. Cependant, nous ne modélisons pas avec précisions tous les flux d'individus.

Ce modèle doit être vu plus comme une exploration de l'outil informatique pour tenter de modéliser des comportements épidemiques et pas comme un outil reconnu qui modélise avec précision les comportements observés dans le monde réel.

Chapitre 3

Interprétation des résultats obtenus

3.1 Interprétation des graphiques

3.1.1 Les courbes

(S) : Au début de l'épidémie, lorsque la maladie est introduite dans la population, cette courbe est élevée car la majorité des individus sont susceptibles d'être infectés. Au fur et à mesure que l'épidémie progresse et que les personnes sont infectées, le nombre de personnes susceptibles diminue.

(I) : Initialement, cette courbe est faible car peu d'individus sont initialement infectés. Ensuite, elle augmente rapidement à mesure que la maladie se propage dans la population. Une fois que la maladie atteint son pic, le nombre d'infections commence à diminuer à mesure que les personnes récupèrent ou décèdent.

(R) : D'abord, cette courbe est proche de zéro car peu de personnes se sont rétablies. Cependant, au fur et à mesure que l'épidémie progresse, le nombre de récupérations augmente, entraînant une augmentation de la courbe. À la fin de l'épidémie, la courbe atteint un plateau, indiquant que la plupart des personnes ont soit récupéré de la maladie, soit succombé à celle-ci.

(D) : Au début de l'épidémie, le nombre de décès est généralement faible car peu de personnes sont infectées. Cependant, à mesure que l'épidémie progresse et que le nombre d'infections augmente, le nombre de décès augmente également, car une proportion de personnes infectées peut succomber à la maladie. La courbe des décès suit souvent de près la courbe des infections, mais avec un certain retard, car les décès surviennent généralement après une période de maladie. Cette courbe est importante pour évaluer l'impact de l'épidémie sur la mortalité de la population et pour informer les décisions de santé publique et les stratégies d'intervention.

3.1.2 Les paramètres modifiables

β représente le taux d'infection. Il influence la vitesse à laquelle la maladie se propage dans la population. Un β élevé signifie que la maladie se propage rapidement, ce qui se traduit par une augmentation rapide du nombre d'infections (courbe I) et une diminution rapide du nombre de personnes susceptibles (courbe S). Lorsque β diminue, la propagation de la maladie ralentit, ce qui peut prolonger la durée de l'épidémie et aplatir la courbe I. Une valeur plus faible de β peut également réduire le pic de l'épidémie.

γ est le taux de guérison. Ce paramètre montre la rapidité avec laquelle les individus infectés récupèrent de la maladie. Un γ élevé signifie que les individus guérissent rapidement et sont retirés de la population infectée, ce qui entraîne une augmentation rapide du nombre de récupérations (courbe R) et une diminution rapide du nombre d'infections (courbe I). Une diminution de γ peut prolonger la durée de l'épidémie, car les individus restent infectés pendant une période plus longue, ce qui peut également entraîner un pic d'infections plus élevé et une courbe R moins abrupte.

μ ou m comme dans la simulation représente le taux de mortalité. Il détermine le taux de mortalité parmi les individus infectés. Un m élevé signifie qu'un plus grand nombre d'individus décèdent des suites de la maladie. Cela se voit sur la courbe par une augmentation plus rapide du nombre de décès (courbe D). Une diminution de m peut réduire le nombre de décès en proportion du nombre d'infections.

3.1.3 Quelques détails supplémentaires

— Courbe des Susceptibles (S) :

β élevé : La courbe des susceptibles diminue plus rapidement, car un taux de transmission élevé implique que plus de personnes susceptibles sont infectées plus rapidement.

γ et m : Ces paramètres n'ont pas d'effet direct sur la courbe des susceptibles, mais influencent indirectement la durée pendant laquelle les infectés peuvent transmettre l'infection.

— Courbe des Infectées (I) :

β élevé : Augmente le nombre de nouvelles infections, ce qui rend la courbe des infectés plus raide et plus haute. L'infection se propage plus rapidement.

γ élevé : Accélère la récupération des infectés, ce qui réduit le pic et la durée de la courbe des infectés.

m élevé : Augmente le taux auquel les infectés décèdent, ce qui peut réduire le pic des infectés mais aussi augmenter le nombre de décès.

— Courbe des Infectées (R) :

γ élevé : Augmente le taux auquel les infectés se rétablissent, ce qui fait croître plus rapidement la courbe des récupérés.

β et m : Indirectement, un β élevé entraîne plus d'infectés et donc potentiellement plus de soignés, tandis qu'un m élevé réduit le nombre d'infectés qui peuvent se rétablir.

— Courbe des Infectées (D) :

m élevé : Augmente le taux de décès parmi les infectés, ce qui rend la courbe des décès plus raide et plus haute

β élevé : Augmente le nombre d'infectés, ce qui peut indirectement augmenter le nombre de décès.

γ élevé : Réduit le nombre d'infectés susceptibles de mourir, ce qui peut aplanir la courbe des décès.

3.1.4 Dynamique globale

— Un taux de transmission élevé (β) et un faible taux de guérison (γ) et de mortalité (m) peuvent conduire à une épidémie prolongée avec un grand nombre de personnes infectées en même temps

— Un taux de guérison élevé (γ) et/ou un taux de mortalité élevé (m) réduisent la durée pendant laquelle les personnes restent infectées, ce qui peut aplatir la courbe des infectés.

3.1.5 Des illustrations concrètes

Se réferer à l'annexe : Annexe 4

3.2 Essais sur le modèle probabiliste

Nous avons fondé nos essais sur une population de 500 individus afin de rendre la simulation moins exigeante en termes de ressources de calcul. Pour la répartition de cette population dans les différentes classes, nous avons suivi une approche similaire : nous avons mené nos recherches[7]. Dans un deuxième temps nous avons discuté et assemblé les données obtenues.

Nous nous trouvons donc avec :

- 100 enfants
- 225 travailleurs
- 25 sans emploi
- 150 retraités

On va partir d'une base puis changer certains paramètres, un à un pour observer leurs effets sur notre simulation. Nous comparerons les graphiques en sortie.

Pour effectuer nos tests on créer une fonction dédiée ***test_simulation*** qui génère une instance de simulation avec des paramètres spécifiés. Cette fonction passe donc outre le menu. Cela nous permet de programmer plusieurs simualtions à la suite afin de réduire le temps de test. Changer les paramètres et relancer les simulations « à la main » peut être laborieux.

— **Paramètres qui resterons invariants dans nos tests :**

- répartition des classes (*comme précisé précédement*)
- temps de simulation : **10000 tics** (*nécéssaire de rester sur une même base pour pouvoir comparer*)
- rayon du point : **2** (*a peu d'intérêt ici*) [1]
- probabilité d'infection : **0.2**
- probabilité de mort : **0.05**(*qu'il soit mort ou recovered l'individu n'agit plus dans la simulation*)

— **Paramètres qui variront pour les tests :**

- classe avec l'infecté de départ : **enfant** [2]
- rayon de contamination : **5**
- probabilité d'infection : **0.05**
- durée de la maladie : **2000 tics**

3.2.1 Base de test

La base de test a été choisie de telle sorte à ce que les courbes se stabilisent en fin de simulation. Pour les graphiques de tous les tests se référer à : Annexe 7 et Annexe 8.

3.2.2 Probabilité d'infection à 0.2

On remarque que le pic d'infection est un peu décalé (plus d'infections plus tôt) et plus abrupte, (plus d'infections sur une période plus courte) mais ce qui est surtout remarquable, c'est la taille de ce pic : il est, en effet, plus important. Comme on pouvait s'y attendre cela a un impact sur la vitesse de propagation de la maladie. Le nombre de gens total ayant contracté la maladie est plus élévé (addition hauteur des courbes jaune, verte et rouge [ou nombre de personnes dans la simulation au départ - hauteur courbe bleu à la fin]), ce qui également cohérent car influencé par la vitesse de propagation.

3.2.3 Rayon de contamination $2\times$ plus grand : 10

On remarque que le pic d'infection est décalé ; plus abrute et plus haut. Ces caractéristiques sont plus marqués que pour l'augmentation de la probabilité d'infection à échelle égale. En effet, on a multiplié par 4 la probabilité d'infections tandis que le rayon lui est multiplié par 2, mais la surface d'infection est un disque de rayon r, $Aire_{disque} = \pi \cdot r^2$ donc la surface est multipliée par $2^2 = 4$. Comme on pouvait s'y attendre également, cela a un impact sur la vitesse de propagation de la maladie. Cependant comme expliqué, l'augmentation de ce paramètre influera d'autant plus la vitesse de propagation et donc in fine le nombre de gens atteint par la maladie.

3.2.4 Durée de maladie $2\times$ plus petite : 1000 tics

On ne remarque pas de pic marqué d'infection. Le nombre d'infecté atteint au maximum environ 50 et oscille autour de 30. L'infection se propage lentement, mais elle se propage quand même, on peut preque parler d'endémie et pas d'épidemie. Voir : 2.4.3 Les malades son infectés moins longtemps et peuvent donc transmettre leur maladie à moins d'indivdus en moyenne. Il suffirait d'une petite perturbation, type confienement, pour enrayer la propagation.

1. Conditionne uniquement les collisions avec les limites du domaine et l'affichage.
2. Nous aurions pu prendre travailleur, cela aurait causé des différence mineures dûes à la répartition dans les maisons ainsi qu'au nombre de personnes par travail.

3.2.5 Durée de maladie 2× plus grande : 4000 tics

La pente initiale de la courbe représentant les personnes infectées est similaire à la base de test. Cependant, étant donné que la maladie dure plus longtemps, de nombreuses personnes sont atteintes en même temps et elles peuvent ainsi infecter une part plus importante de la population. On se retrouve donc au bout de 10000 tics avec moins de susceptibles que pour la simulation de base. Les individus n'ont également pas le temps de mourir ou d'être recovered, les courbes ne se sont pas stabilisés en fin de simulation.

3.2.6 Classe infectée : sans emploi

Dans notre simulation la classe sans emploi passe en moyenne moins de temps par jour en dehors de chez elle, (elle n'est pas au travail, ni à l'école). On remarque un pic d'infection plus tardif et plus étalé que pour la base, mais le nombre de personnes ayant contracté la maladie à t = 10000 tics est proche de celui de la base de test. Ainsi, la propagation bien que retardée, est epidémique une fois installée. L'analyse serait similaire pour un infecté de classe : retraité, bien que le pic serait encore un plus retardé, dû à son emploi du temps. Voir : 2.3

3.2.7 Et le confinement ?

Dans ce test, on met en place un confinement à environ t = 4000 tics que l'on enlève à environ t = 6500 tics. Nous pouvons clairement observer deux pics d'infection. Pour le premier pic, nous constatons l'efficacité du confinement qui réduit le nombre d'infections et donc d'infectés. Ce confinement endigue l'épidémie certes, mais dès que les personnes retournent à leur vie normale l'épidemie reprend de plus belle et on se retrouve avec un un nombre de personnes suceptibles à la fin assez similaire qu'avec aucun confinement.

3.2.8 Liens entre les modèles et commentaires sur le modèle et les tests

On peut assimiler le β du modèle théorique à notre paramètre rayon d'infection et probabilité d'infection. Ce sont en effet ces deux paramètres qui représentent le taux d'infection, ils influent sur la facilité pour un individu infecté à transmettre la maladie.

Le γ, lui, peut être assimilé à la durée de la maladie et au taux de mortalité. Plus γ est élévé plus la durée de la maladie est courte et la proportion de recovered est élévée. On peut en effet établir des similarités entre l'allure de nos graphiques (attention code couleur différent) pour un taux d'infection élevé et pour un taux de guérison élévé.

Mais comme on l'a vu : l'avantage avec notre modèle probabiliste c'est que l'on peut faire varier bon nombre de paramètres sur les caractéristiques de notre population et de l'infection.

Il aurait été possible de commenter sur la valeur de R pendant nos tests. Aussi, nous aurions pu réaliser des simulations sur des échelles de temps plus longues, sur l'équivalent dans la simulation de plusieurs semaines et ainsi pouvoir ajuster notre échelle de temps avec une probabilité d'infection plus faible. En outre, regarder l'effet de changement de répartition de population aurait pu être intéressant. En somme, nous aurions pu réaliser un nombre conséquent de tests suplémentaires, mais ces derniers prennent un temps considérable à mettre en place (choix de caractéristiques intéressantes à observer). Enfin, l'objectif premier était de réaliser un modèle qui nous parassait un minimum cohérent ; ce que l'on a en partie démontré avec les tests.

Conclusion et perspectives

3.3 Le travail réalisé

Dans le cadre de ce projet nous avons exploré deux approches distinctes : la simulation théorique et la simulation probabiliste, afin de modéliser de la manière la plus réaliste possible les épidémies. Nous avons intégré des graphiques, illustrations et modélisations sous forme de « boîtes » et même une simulation avec un confinement grâce à notre code python. Ces outils visent à représenter fidèlement les dynamiques de propagation d'une épidémie, en tenant compte des variables et des interactions complexes qui influencent sa propagation.

3.4 L'apport personnel de cet E.C. projet

Ce projet de simulation d'épidémies a été une expérience enrichissante, nous apportant à la fois des connaissances techniques et des compétences humaines.

Travailler en groupe nous a permis de mettre en place la collaboration et de l'entraide, qui ont été essentielles pour surmonter les défis techniques rencontrés en cours de route, renforçant ainsi l'esprit d'équipe. Ce projet nous a permis d'approfondir notre compréhension des modèles épidémiologiques et des mécanismes de propagation des maladies et également de nous améliorer en codage informatique.

Nous avons pu développer un projet final dont nous sommes fiers.

Cependant, cela n'a pas été sans difficultés. La coordination des tâches et la programmation en équipe ont parfois été laborieuses, nécessitant une communication constante et une répartition efficace des responsabilités.

Malgré ces obstacles, le projet a été une expérience formatrice, nous enseignant l'importance de la persévérance, de la patience et de la collaboration dans la réalisation de projets complexes. Nous en ressortons non seulement avec un ensemble de compétences techniques améliorées, mais également avec une appréciation plus profonde du travail d'équipe.

3.5 Perspectives pour la suite de ce projet

Les perspectives pour la suite de ce projet sont prometteuses, notamment en faisant le lien avec les cours de M8 et I4 : c'est à dire en intégrant de l'analyse statistique. En utilisant des données réelles et des modèles de Machine Learning par exemple, nous pouvons affiner nos simulations pour mieux modéliser la propagation d'épidémies. L'exploitation de statistiques permettra d'améliorer la précision de nos modèles et ainsi les prédire. Par exemple, en analysant les taux d'incidence et de prévalence nous pouvons mieux comprendre la dynamique de propagation de la maladie dans différentes populations. De plus, l'utilisation de statistiques démographiques peut aider à identifier les groupes les plus vulnérables et à adapter les stratégies de prévention en conséquence.

3.5.1 Quelques propositions de sujet supplémentaires

Nous avons décidé de vous proposer quelques sujets en lien avec les statistiques :

1. Utilisation de modèles de régression pour prédire la propagation des épidémies dans différentes populations. (M8)
2. Méthodes de Monte-Carlo pour estimer l'incertitude dans les modèles de simulation d'épidémies. (I4)
3. Analyse de clusters spatiaux-temporels pour identifier les zones à haut risque de propagation des maladies. (I4)

Bibliographie

[1] Photo page de garde, https://cdn.pixabay.com/photo/2024/02/29/08/22/ai-generated-8603651_960_720.png
(Valide à la date du 09/06/2024)

[2] Schéma du SIR et équations différentielles , http://mpechaud.fr/scripts/SIR/SIR.html
(Valide à la date du 09/06/2024)

[3] Explication du modèle SIR, https://fr.wikipedia.org/wiki/Mod%C3%A8les_compartimentaux_en_%C3%A9pid%C3%A9miologie
(Valide à la date du 09/06/2024)

[4] INSEE, temps de travail, https://www.insee.fr/fr/statistiques/7456897?sommaire=7456956
(Valide à la date du 09/06/2024)

[5] INSEE, emploi du temps, https://www.insee.fr/fr/statistiques/1281050
(Valide à la date du 09/06/2024)

[6] INSEE, dossier complet Paris, https://www.insee.fr/fr/statistiques/2011101?geo=DEP-75
(Valide à la date du 09/06/2024)

[7] INSEE, démographie pour choix taille des classes,

https://www.insee.fr/fr/statistiques/2415121

https://www.insee.fr/fr/statistiques/3676623?sommaire=3696937

https://www.insee.fr/fr/outil-interactif/5367857/tableau/20_DEM/21_POP#
(Valides à la date du 09/06/2024)

[8] INSEE, répartition dans les foyers, https://www.insee.fr/fr/statistiques/4277630?sommaire=4318291
(Valide à la date du 09/06/2024)

[9] Dietz, Klaus *The estimation of the basic reproduction number for infectious diseases. Statistical Methods in Medical Research.* , PubMed, 1993.

[10] Page Wikipedia R_0, https://fr.wikipedia.org/wiki/Nombre_de_reproduction_de_base (Valide à la date du 08/06/2024)

[11] Video modèle simulation 3B1B, https://www.youtube.com/watch?v=gxAaO2rsdIs&t=429s (Valide à la date du 08/06/2024)

[12] Video modèle simulation SIR , https://www.youtube.com/watch?v=2nzNDyUt1vg&list=WL&index=72&t=5183s (Valide à la date du 10/06/2024)

Annexes

Annexe SIR

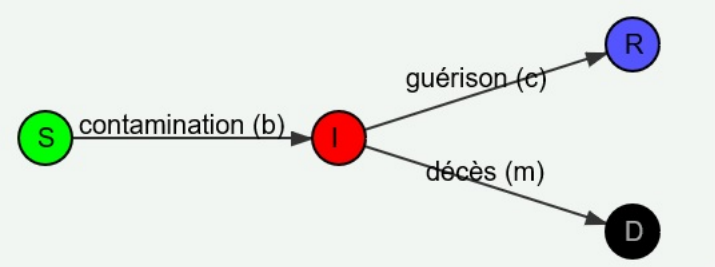


FIGURE ANNEXE 1 – **Représentation schématique du modèle SIR**

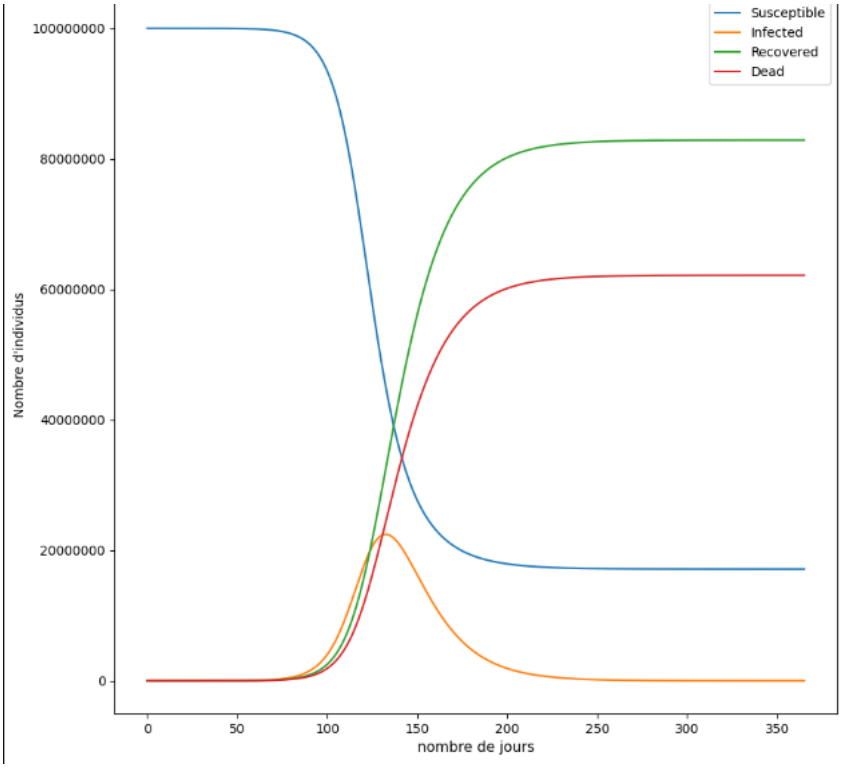


FIGURE ANNEXE 2 – **Graphique du modèle SIR**

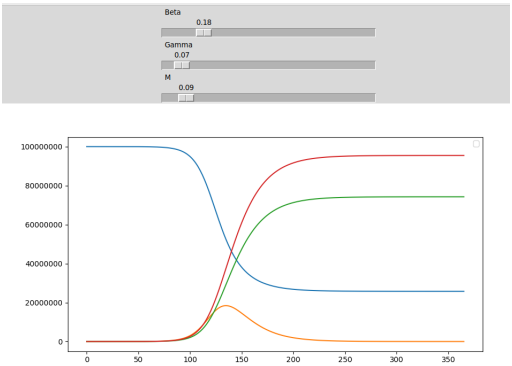


FIGURE ANNEXE 3 – **Graphique du modèle SIR (sliders)**

FIGURE ANNEXE 4 – **Visualisations SIR**

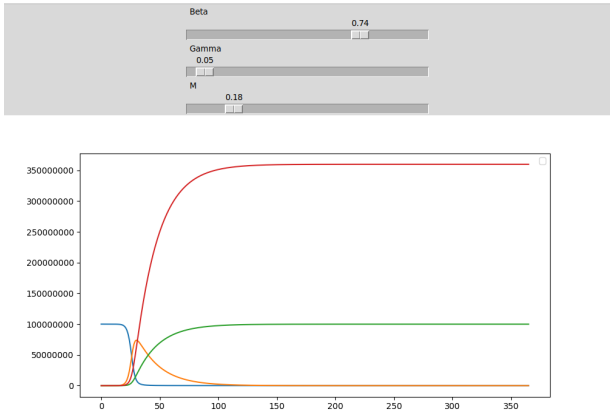


FIGURE ANNEXE 4 – Graphique avec Beta elevé

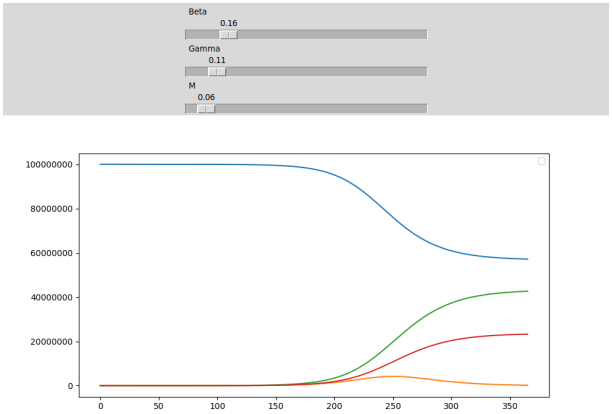


FIGURE ANNEXE 4 – Graphique avec Gamma elevé

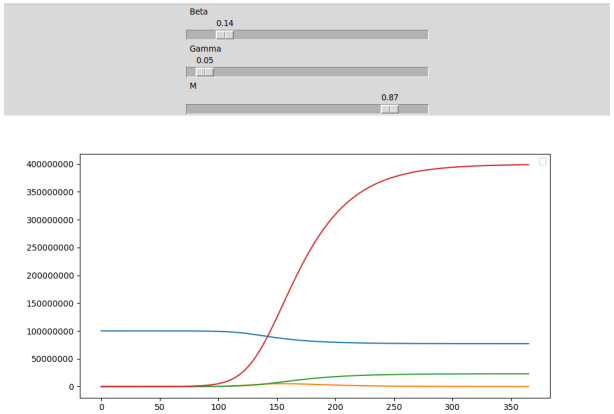


FIGURE ANNEXE 4 – Graphique avec M elevé

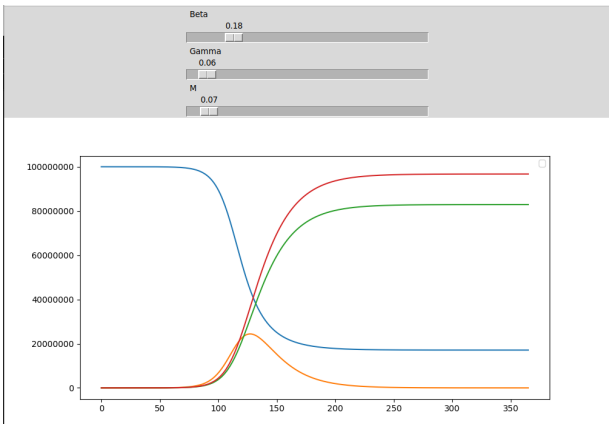


FIGURE ANNEXE 4 – Graphique avec les paramètres normaux

Annexe simualtion probabiliste

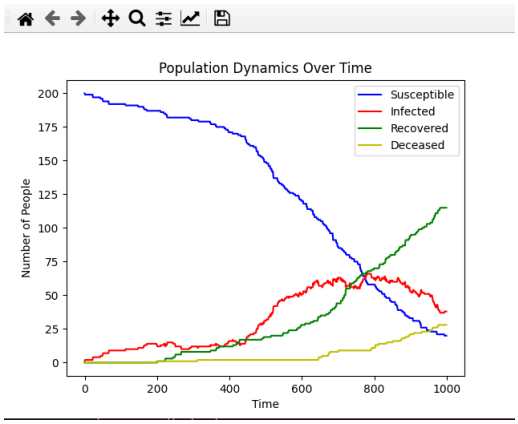


FIGURE ANNEXE 5 – Graphique nuages de points de la première version du modèle

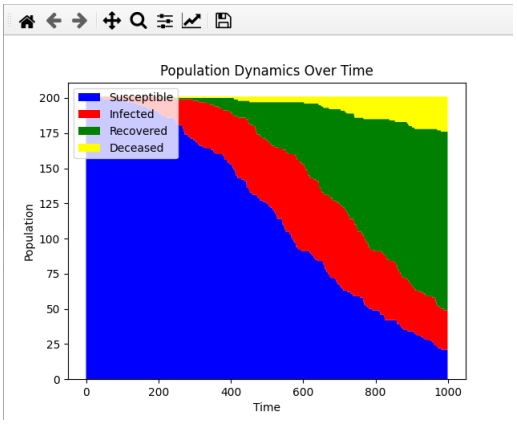


FIGURE ANNEXE 6 – Graphique stack de la première version du modèle

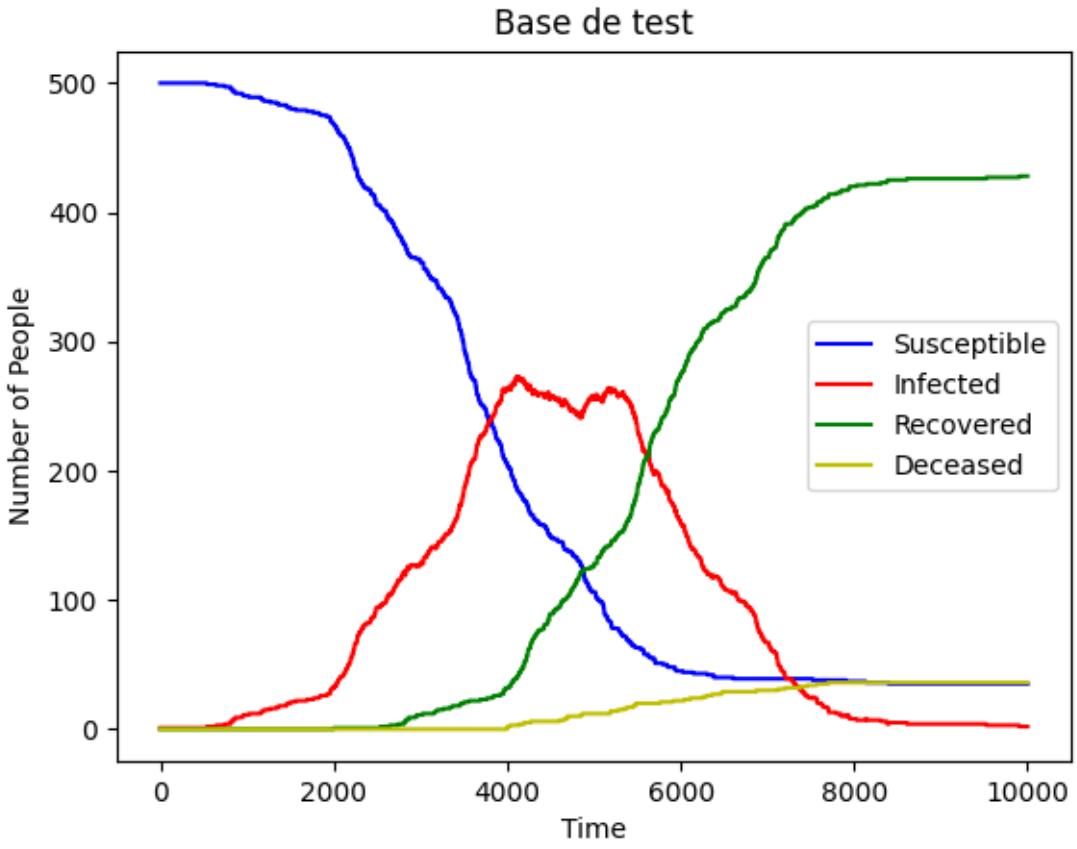
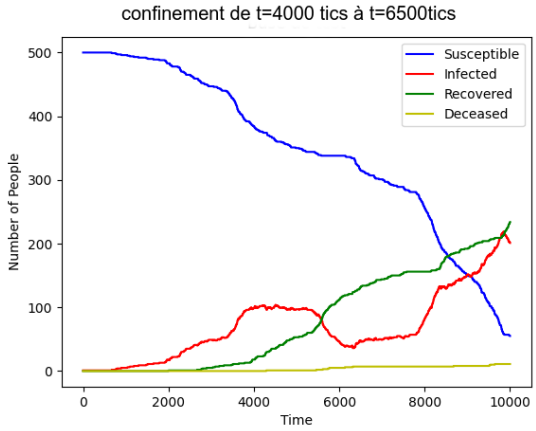
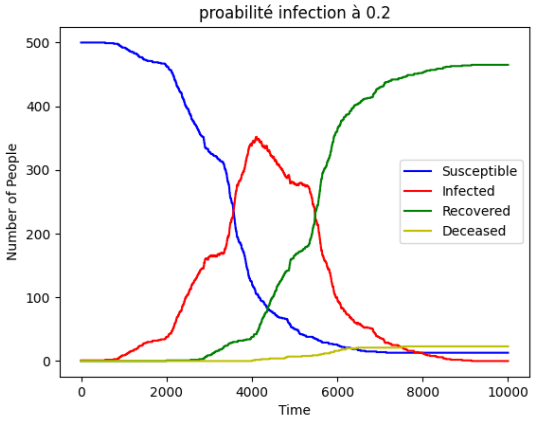
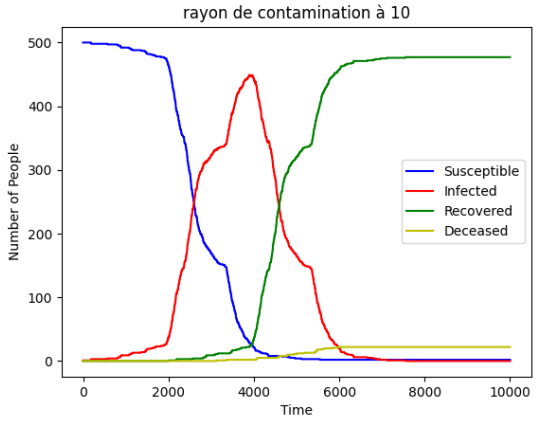


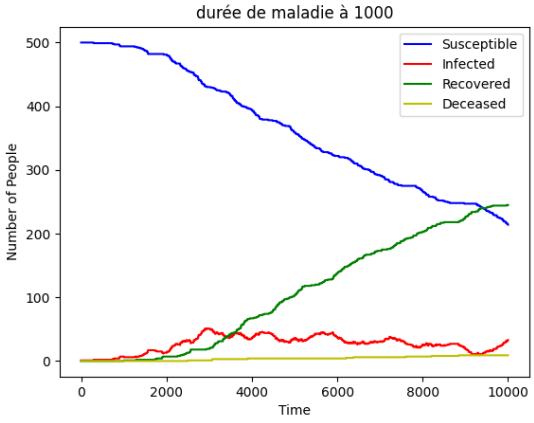
FIGURE ANNEXE 7 – **Base de test du modèle**

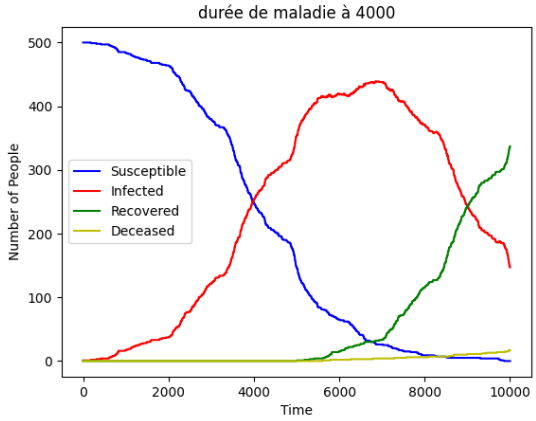
FIGURE ANNEXE 8 – **Les graphiques type nuage de points des différents tests**

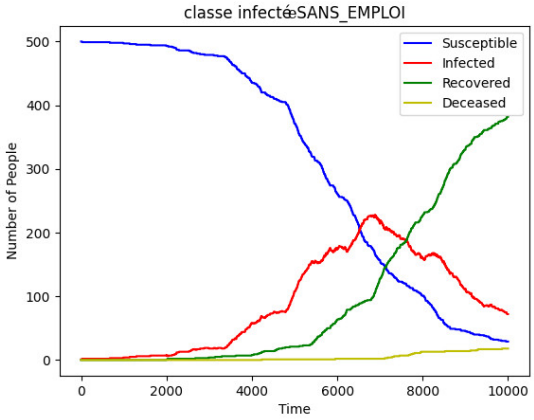












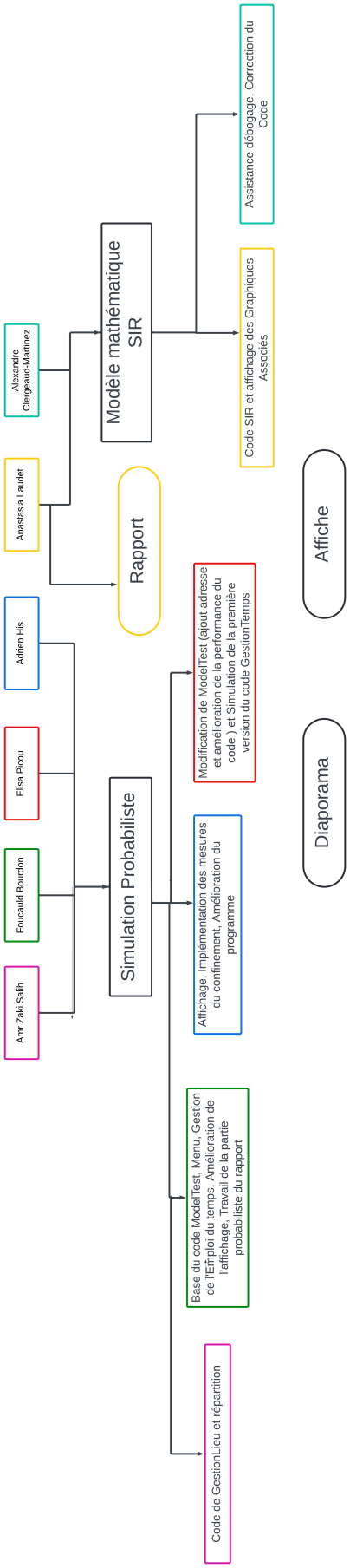


FIGURE ANNEXE 9 – **Répartition tâches**