

# Modélisation et Contrôle de maladies en ville

Modèle des réseaux complexes stochastiques de contact - Contrôle de la propagation de maladies infectieuses en ville

Younes-Jihad Boumoussou

Juillet 2023

# Introduction



## 1 Définition

- Transmission d'une maladie infectieuse

## 2 Modélisation mathématique par réseaux complexes

- Réseaux complexes de contact
- Modèle stochastique et variables aléatoires
- Simulation numérique des résultats

## 3 Contrôle de la propagation de l'épidémie

- Modèle théorique de contrôle d'épidémie
- Application à notre modélisation (Modèle SRMI)
- Simulation numérique des résultats

## 4 Conclusion

## 5 Annexe

# Définition

## Transmission d'une maladie infectieuse



# Modélisation mathématique par réseaux complexes

## 1 Définition

- Transmission d'une maladie infectieuse

## 2 Modélisation mathématique par réseaux complexes

- Réseaux complexes de contact
- Modèle stochastique et variables aléatoires
- Simulation numérique des résultats

## 3 Contrôle de la propagation de l'épidémie

- Modèle théorique de contrôle d'épidémie
- Application à notre modélisation (Modèle SRMI)
- Simulation numérique des résultats

## 4 Conclusion

## 5 Annexe

# Réseaux complexes de contact

## Exemple de réseau complexe

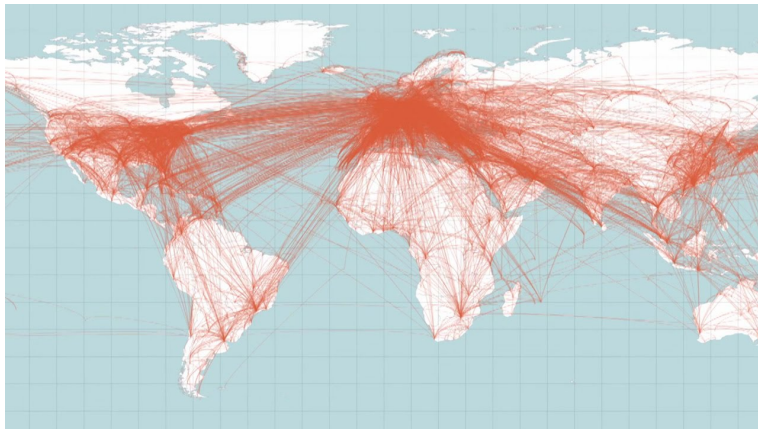


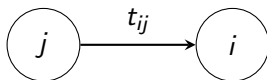
Figure: Réseau complexe mondial

# Modèle stochastique et variables aléatoires

## Probabilité de transmission

Etant donné une population  $\Omega = \{1, \dots, n\}$ ,

Soit une paire  $(i, j) \in \Omega^2$ ,  $j$  étant infecté,  $i$  étant sain et soit  $t_{ij}\delta t$  la probabilité que  $j$  infecte  $i$  pendant une durée  $\delta t$ .



Soit  $T_{ij}(t)$  la probabilité que  $i$  soit infecté par  $j$  à l'instant  $t$ .

$$1 - T_{ij}(t) = \lim_{\delta t \rightarrow 0} (1 - t_{ij}\delta t)^{t/\delta t} = e^{-t_{ij}t}$$

$$\implies T_{ij}(t) = 1 - e^{-t_{ij}t}$$

On définit alors :

$$T(t) = \begin{pmatrix} T_{11}(t) & T_{12}(t) & \cdots & T_{1n}(t) \\ T_{21}(t) & T_{22}(t) & \cdots & T_{2n}(t) \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1}(t) & T_{n2}(t) & \cdots & T_{nn}(t) \end{pmatrix}$$

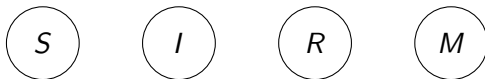
Que l'on nomme la matrice de transmission à l'instant  $t$ .



# Modèle stochastique et variables aléatoires

## Hypothèses du modèle

On définit 4 états pour les individus de la population :



- Infectés à un instant donné de la propagation, on note leur ensemble  $I$
- Sains, on note leur ensemble  $S$
- Rétablis, on note leur ensemble  $R$
- Décédés, on note leur ensemble  $M$

# Modèle stochastique et variables aléatoires

## Variables aléatoires

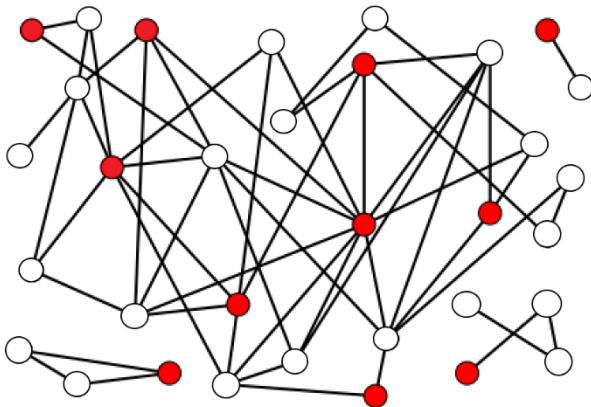
- Soit  $I_k(t)$ ,  $M_k(t)$ ,  $R_k(t)$  les variables aléatoires respectives qui reçoivent 1 si l'individu  $k$  est infecté à un instant donné de la propagation, décédé ou rétabli à l'instant  $t$  et 0 sinon.
- Soit  $I(t)$ ,  $M(t)$ ,  $R(t)$  les variables aléatoires respectives qui reçoivent le nombre total d'individus infectés à un instant donné de la propagation, décédés et rétablis sur l'ensemble de la population à l'instant  $t$ .

Et on note, pour simplifier :

$$p_k(t) = \mathbb{P}(I_k(t) = 1), q_k(t) = \mathbb{P}(M_k(t) = 1), r_k(t) = \mathbb{P}(R_k(t) = 1)$$

# Modèle stochastique et variables aléatoires

## Probabilité d'infection

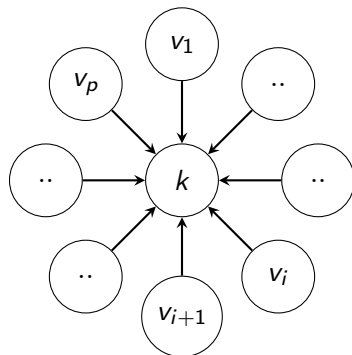


**Figure:** Un réseau complexe de contact contenant des sains et des infectés

# Modèle stochastique et variables aléatoires

## Probabilité d'infection

En notant  $voisins(k) = \{i \in \Omega, \exists t \geq 0, T_{ki}(t) > 0\} = \{v_i, 1 \leq i \leq p\}$  les voisins de  $k$ .



Alors  $\forall k \in \Omega$ ,

$$p_k(t + dt) = \sum_{i \in \text{voisins}(k)} p_i(t) T_{ki}(t)$$

D'où :

$$\begin{pmatrix} p_1(t+dt) \\ p_2(t+dt) \\ \vdots \\ p_n(t+dt) \end{pmatrix} = \begin{pmatrix} T_{11}(t) & T_{12}(t) & \cdots & T_{1n}(t) \\ T_{21}(t) & T_{22}(t) & \cdots & T_{2n}(t) \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1}(t) & T_{n2}(t) & \cdots & T_{nn}(t) \end{pmatrix} \begin{pmatrix} p_1(t) \\ p_2(t) \\ \vdots \\ p_n(t) \end{pmatrix}$$
$$\implies P(t+dt) = T(t)P(t)$$

# Modèle stochastique et variables aléatoires

## Evolution de l'infection dans la population

On discrétise le temps :  $t = n \in \mathbb{N}$

$\forall n \in \mathbb{N}$ ,

$$P_{n+1} = T_n P_n$$

Et on a alors  $\forall n \in \mathbb{N}^*$ ,

$$P_n = \left( \prod_{k=0}^{n-1} T_k \right) P_0$$

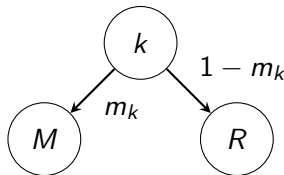
Où les individus initialement infectés fournissent  $P_0$ .

# Modèle stochastique et variables aléatoires

## Probabilité de décès et de guérison

Soient :

- $\tau_k$  La durée caractéristique d'infection de l'individu  $k$ .
- $m_k$  La probabilité de décès de l'individu  $k$  à un instant donné de l'infection, durant  $\tau_k$ .



Soit  $M_k$  la probabilité que  $k$  soit décédé durant  $\tau_k$  de la maladie.

$$1 - M_k = \lim_{\delta t \rightarrow 0} (1 - m_k \delta t)^{\tau_k / \delta t} = e^{-m_k \tau_k}$$

$$\implies M_k = 1 - e^{-m_k \tau_k}$$

# Modèle stochastique et variables aléatoires

## Lois de probabilité

$\forall k \in \Omega,$

$$q_k(t) = \begin{cases} p_k(t - \tau_k)M_k & \text{si } t > \tau_k \\ 0 & \text{sinon} \end{cases}$$

$$r_k(t) = \begin{cases} p_k(t - \tau_k)(1 - M_k) & \text{si } t > \tau_k \\ 0 & \text{sinon} \end{cases}$$

D'où,

$$I_k(t) \sim B(p_k(t)), M_k(t) \sim B(q_k(t)), R_k(t) \sim B(r_k(t))$$



# Modèle stochastique et variables aléatoires

## Espérance du nombre d'infectés

Soit  $k \in \{1, ..n\}$ ,

On note  $(C_n^k(p))_{1 \leq p \leq C_n^k}$  la suite des combinaisons de k éléments parmi n.

$$\mathbb{P}(I(t) = k) = \sum_{p=1}^{C_n^k} \prod_{i \in C_n^k(p)} p_i(t) \prod_{i \in \Omega / C_n^k(p)} (1 - p_i(t))$$

$$\mathbb{E}(I(t)) = \sum_{k=1}^n k \mathbb{P}(I(t) = k)$$

# Modèle stochastique et variables aléatoires

Espérances des nombres d'infectés, de décédés et de guéris

$$\mathbb{E}(I(t)) = \sum_{k=1}^n \sum_{p=1}^{C_n^k} k \prod_{i \in C_n^k(p)} p_i(t) \prod_{i \in \Omega / C_n^k(p)} (1 - p_i(t))$$

$$\mathbb{E}(M(t)) = \sum_{k=1}^n \sum_{p=1}^{C_n^k} k \prod_{i \in C_n^k(p)} q_i(t) \prod_{i \in \Omega / C_n^k(p)} (1 - q_i(t))$$

$$\mathbb{E}(R(t)) = \sum_{k=1}^n \sum_{p=1}^{C_n^k} k \prod_{i \in C_n^k(p)} r_i(t) \prod_{i \in \Omega / C_n^k(p)} (1 - r_i(t))$$

# Modèle stochastique et variables aléatoires

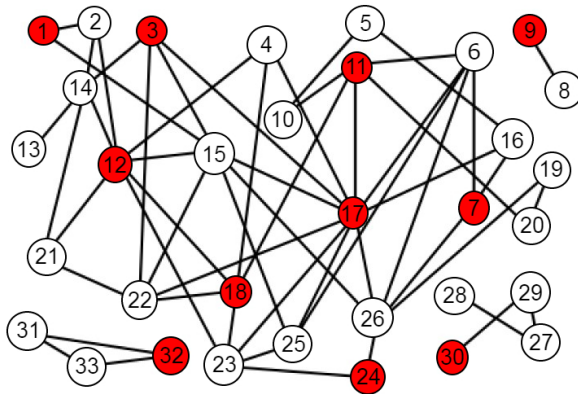
Espérance du nombre de porteurs dans la population

En considérant  $P(t)$  la variable aléatoire du nombre des porteurs, on a :

$$\mathbb{E}(P(t)) = \mathbb{E}(I(t)) - \mathbb{E}(R(t)) - \mathbb{E}(M(t))$$

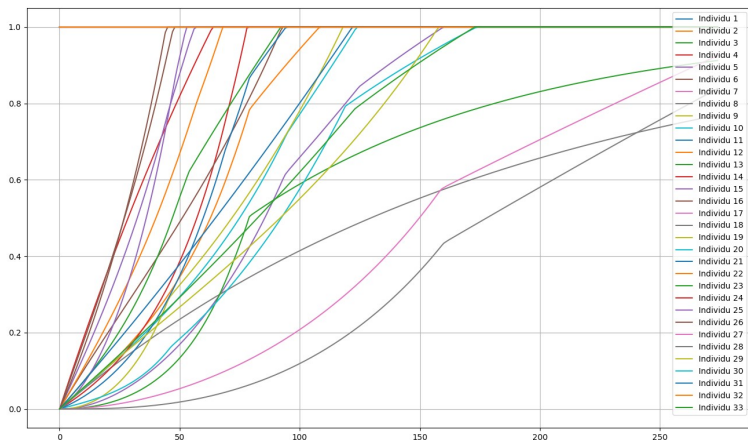
# Simulation numérique des résultats

Graphe de la population considérée



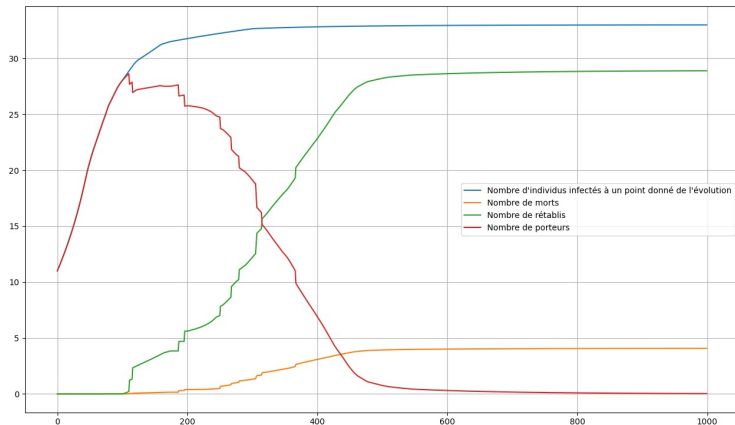
# Simulation numérique des résultats

## Evolution des probabilités d'infection



# Simulation numérique des résultats

Evolution du nombre de porteurs, d'infectés, de décédés et de rétablis



# Contrôle de la propagation de l'épidémie

- 1 Définition
  - Transmission d'une maladie infectieuse
- 2 Modélisation mathématique par réseaux complexes
  - Réseaux complexes de contact
  - Modèle stochastique et variables aléatoires
  - Simulation numérique des résultats
- 3 Contrôle de la propagation de l'épidémie
  - Modèle théorique de contrôle d'épidémie
  - Application à notre modélisation (Modèle SRMI)
  - Simulation numérique des résultats
- 4 Conclusion
- 5 Annexe

# Modèle théorique de contrôle d'épidémie

## Définition du nombre de reproduction de base $R_0$

### Définition : Nombre de reproduction de base $R_0$

En épidémiologie, le nombre de reproduction de base d'une maladie est défini comme le nombre moyen de cas attendus directement générés par un cas dans une population sensible à l'infection.



# Modèle théorique de contrôle d'épidémie

Intérêt du contrôle de  $R_0$

## Théorème de Stabilité d'un système épidémiologique

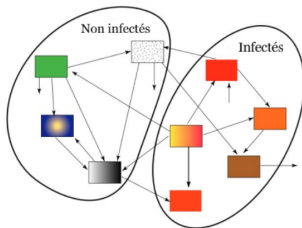
Si  $R_0 < 1$ , alors la maladie disparaît de la population au bout d'un certain temps.

Si  $R_0 \geq 1$ , alors la maladie reste endémique dans la population.

# Modèle théorique de contrôle d'épidémie

## Détermination de $R_0$ : Approche de la matrice de prochaine génération

Soit  $n \in \mathbb{N}^*$ , on définit  $n$  compartiments distincts pour les individus de la population,



$$x = \left( \underbrace{x_1, \dots, x_p}_{\text{non infectés}}, \underbrace{x_{p+1}, \dots, x_n}_{\text{infectés}} \right) \in \mathbb{R}^n$$

Avec  $x_i \geq 0$  correspond au nombre d'individus dans chaque compartiment.

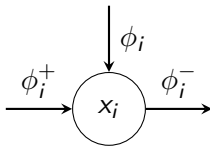
# Modèle théorique de contrôle d'épidémie

## Dynamique de la population

On décrit la propagation de la maladie par  $f : \Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ .

$$\dot{x}(t) = f(x(t))$$

En décomposant les flux :



On pourra alors décomposer  $f$  de la forme suivante :

$$f = \phi + \phi^+ + \phi^-$$

# Modèle théorique de contrôle d'épidémie

## Equilibre de la population

### Théorème de La matrice de prochaine génération (Fondamental)

Soit un point d'équilibre sans maladie,  $x^* = (x_1^*, \dots, x_p^*, 0, \dots, 0)$ ,

Soient les matrices,

$$\mathbb{F} = \left( \frac{\partial \phi_i}{\partial x_j}(x^*) \right)_{p+1 \leq i, j \leq n}, \mathbb{V} = \left( \frac{\partial \phi_i^+ - \phi_i^-}{\partial x_j}(x^*) \right)_{p+1 \leq i, j \leq n}$$

Le nombre de reproduction de base lié au point d'équilibre  $x^*$  est :

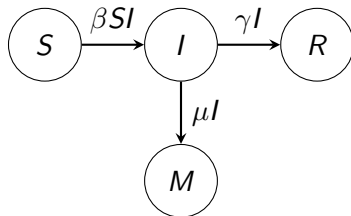
$$R_0 = \rho(-\mathbb{F}\mathbb{V}^{-1})$$

Avec :  $\rho$  désigne le rayon spectral.

# Application à notre modélisation

## Modélisation choisi (Modèle SRMI)

On dispose de 4 compartiments  $x = (S, R, M, I)$  pour la population:



Avec :

- $\mu$  le taux de mortalité de la maladie.
- $\beta$  le taux d'infection par contact.
- $\gamma$  le taux de guérison de la maladie.

# Application à notre modélisation

## Paramètres de contrôle

Soient  $u_1, u_2, u_3$  les paramètres de contrôle de la propagation définis dans  $[0,1]$  comme suit :

- $u_1$  : Degré de confinement et mesures de sécurité (Réduction des contacts).
- $u_2$  : Rétablissement (Prise en charge des infectés et traitement).
- $u_3$  : Prévention des Décès (Intervention dans les cas critiques).

En intégrant ces paramètres, on obtient le système d'ED suivant :

$$\begin{cases} \dot{S} = -(1 - u_1)\beta SI \\ \dot{I} = (1 - u_1)\beta SI - (1 - u_3)\mu I - (1 + u_2)\gamma I \\ \dot{R} = (1 + u_2)\gamma I \\ \dot{M} = (1 - u_3)\mu I \end{cases}$$

# Application à notre modélisation

## Calcul de $R_0$

En appliquant le théorème de la matrice de prochaine génération :

Pour un état d'équilibre sans malades  $x^* = (S^*, R^*, M^*, 0)$ , on retrouve pour le modèle considéré :

$$R_0 = \frac{(1-u_1)\beta}{(1+u_2)\gamma + (1-u_3)\mu} S^*$$

Notre but dorénavant est d'ajuster les paramètres de contrôle de sorte à avoir  $R_0 < 1$  pour un équilibre où l'on a  $S^*$  sains.

### Hypothèses :

- On se donne une population de 110.000 individus dont 10.000 sont infectés.
- Dans un premier temps, nous simulons la maladie avec des paramètres de contrôle nuls, puis nous simulons leur effet sur la propagation de la maladie.

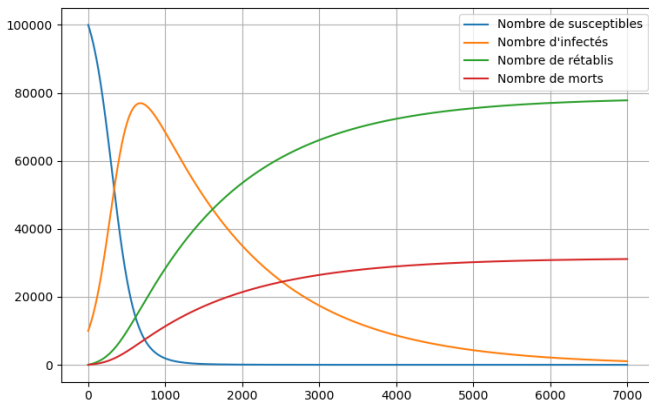


# Simulation numérique des résultats

Evolution de la maladie au sein de la population

**Cas 1 :**  $R_0 = 10$

$u_1 = u_2 = u_3 = 0$  et  $\beta = 7 \cdot 10^{-8}$ ,  $\gamma = 5 \cdot 10^{-4}$ ,  $\mu = 2 \cdot 10^{-4}$

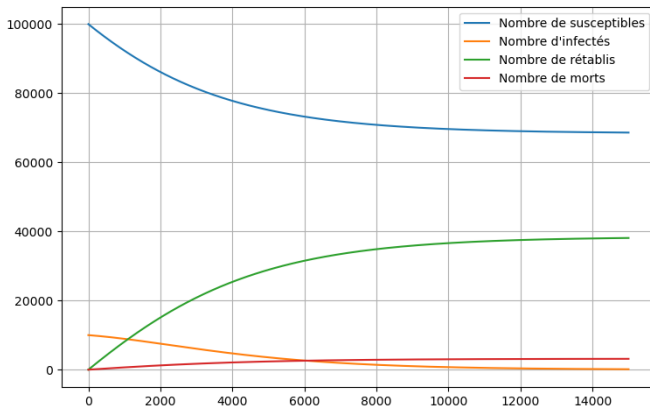


# Simulation numérique des résultats

Evolution de la maladie au sein de la population

**Cas 2 :**  $R_0 = 0.91$

$u_1 = 0.88$  ,  $u_2 = 0.7$  ,  $u_3 = 0.65$  et  $\beta = 7.10^{-8}$ ,  $\gamma = 5.10^{-4}$ ,  $\mu = 2.10^{-4}$



# Conclusion



### Développement du calcul de $R_0$

D'après le système différentiel, pour  $x = (S, R, M, I)$  :

$$\phi(x) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ (1 - u_1)\beta SI \end{pmatrix}, \phi^+(x) = \begin{pmatrix} 0 \\ (1 + u_2)\gamma I \\ (1 - u_3)\mu I \\ 0 \end{pmatrix}, \phi^-(x) = \begin{pmatrix} (1 - u_1)\beta SI \\ 0 \\ 0 \\ (1 + u_2)\gamma I + (1 - u_3)\mu I \end{pmatrix}$$

Pour un point d'équilibre sans maladies  $x^* = (S^*, R^*, M^*, 0)$

En appliquant le théorème :

$$\mathbb{F} = (1 - u_1)\beta S^* \text{ et } \mathbb{V} = -((1 + u_2)\gamma + (1 - u_3)\mu)$$

Ainsi :

$$R_0 = \frac{(1 - u_1)\beta}{(1 + u_2)\gamma + (1 - u_3)\mu} S^*$$

### Commentaires :

- Il en découle ainsi que pour un équilibre avec plus d'individus sains, il est nécessaire d'adopter des paramètres de contrôle plus grands pour assurer cet équilibre.

### 1 Théorème 1 : Matrices de Metzler

#### 1.1 Définitions :

Soit  $A = (a_{i,j})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq m}} \in M_{n,m}(\mathbb{R})$ ,

- A est une matrice non négative si  $\forall i \in \{1, \dots, n\}$ ,  $\forall j \in \{1, \dots, m\}$  on a  $a_{i,j} \geq 0$ , c'est à dire que toute ses entrées non négatives. Nous notons une telle matrice  $A \geq 0$ .
- A est une matrice positive si A est non-négative et  $\exists i \in \{1, \dots, n\}$ ,  $\exists j \in \{1, \dots, m\}$  tel que  $a_{k,l} > 0$ , c'est à dire ayant au moins un coefficient strictement positif.
- A est une matrice strictement positive si  $\forall i \in \{1, \dots, n\}$ ,  $\forall j \in \{1, \dots, m\}$  on a  $a_{i,j} > 0$ , c'est à dire tous ses coefficients sont strictement positifs. Nous noterons une telle matrice  $A >> 0$ .
- On dit que A est une matrice de Metzler si  $\forall (i, j) \in \{1, \dots, n\}^2$  tels que  $i \neq j$  on a  $a_{i,j} \geq 0$ .
- On dit qu'une matrice M est asymptotiquement stable si  $\rho(M) < 1$  avec  $\rho$  le rayon spectral.

#### 1.2 Théorème de Metzler :

Si A est une matrice de Metzler, les conditions suivantes sont équivalentes :

- La matrice de Metzler A est asymptotiquement stable.
- La matrice de Metzler A est inversible et  $-A^{-1} \geq 0$ .
- Si b est un vecteur tel que  $b >> 0$  alors il existe  $x >> 0$  tel que  $Ax + b = 0$ .
- Il existe  $c > 0$  tel que  $Ac << 0$ .
- Il existe  $c >> 0$  tel que  $Ac << 0$ .

### 2 Théorème 2 : Théorème de Perron-Frobenius

Le rayon spectral  $\rho$  d'une matrice positive  $A \geq 0$  est une valeur propre à laquelle est associé un unique vecteur  $x$  de norme 1 à coordonnées strictement positives tel que  $Ax = \rho x$ .

Ce vecteur est unique, à un coefficient multiplicatif près car  $\rho$  est une valeur propre simple c'est à dire d'espace propre de dimension 1.

### 3 Théorème 3 : Critère de Stabilité de Lyapunov

#### 3.1 Définitions :

Soit  $\Omega \subset \mathbb{R}^n$  ouvert, connexe et  $f : \Omega \rightarrow \mathbb{R}^n$  fonction de classe  $C^1$  et  $x : t \in \mathbb{R}^+ \rightarrow \mathbb{R}^n$

Soit un système différentiel autonome défini par  $\dot{x}(t) = f(x(t))$ .

- Nous appelons  $x^*$  un point d'équilibre ou fixe si  $f(x^*) = 0$ .
- Soit  $t_0$  l'origine des temps, on dit que  $x^*$  est stable si,

$$\forall \epsilon > 0, \exists \delta > 0, \forall t \geq t_0, \|x(t_0) - x^*\| < \delta \implies \|x(t) - x^*\| < \epsilon$$

- On dit que  $x^*$  est asymptotiquement stable si,

$$x^* \text{ est stable et } \exists \delta > 0, \|x(t_0) - x^*\| < \delta \implies x(t) \xrightarrow{t \rightarrow +\infty} x^*$$

Remarquer que les points stables ne sont pas forcément asymptotiquement stables.

#### 3.2 Théorème de Lyapunov :

Soit  $x^*$  un point d'équilibre ie  $f(x^*) = 0$ . Si  $f \in C^1(\Omega, \mathbb{R}^n)$  tel que  $Df(x^*) \in M_n(\mathbb{R})$  a toutes ses valeurs propres complexes de partie réelle strictement négative, alors  $x^*$  est un point d'équilibre attractif.

C'est à dire que,  $\exists r > 0$  tel que pour  $x_0 \in \mathbb{R}^n$  proche de  $x^*$  tel que  $\|x_0 - x^*\| \leq r$ , la solution de :

$$\begin{cases} u'(t) = f(u(t)) \\ u(t_0) = x_0 \end{cases}$$

vérifie  $u(t) \xrightarrow{t \rightarrow +\infty} x^*$ ,  $x^*$  est donc asymptotiquement stable. De plus, cette convergence se fait à une vitesse exponentielle.

### 4 Théorème 4 : Théorème de Varga

Soit une matrice de Metzler  $A$ . Alors pour toute décomposition de  $A$  de la forme  $A = F + V$ , où  $F \geq 0$  et  $V$  une matrice de Metzler asymptotiquement stable, les deux propositions suivantes sont équivalentes :

- $A$  est asymptotiquement stable.
- $\rho(-FV^{-1}) < 1$ .



### Démonstration du théorème de Stabilité d'un système épidémiologique

Notons pour  $A \in M_n(\mathbb{R})$ ,

$$\alpha(A) = \max\{Re(\lambda), \forall \lambda \in Sp_{\mathbb{C}}(A)\}$$

D'après le théorème 1 de Metzler,  $-\mathbb{F}\mathbb{V}^{-1}$  est positive, alors par le théorème 2 de Perron-Frobenius,  $R_0 = \rho(-\mathbb{F}\mathbb{V}^{-1})$  est une valeur propre de  $-\mathbb{F}\mathbb{V}^{-1}$ . En utilisant le théorème 3 de Lyapunov, alors soit un point d'équilibre sans maladies  $x^*$ , c'est un point asymptotiquement stable si  $\alpha(Df(x^*)) < 0$  et asymptotiquement instable si  $Df(x^*) \geq 0$ . Sachant  $f = \phi + \nu$ ,  $\alpha(\mathbb{V}) < 0$  et  $\alpha(J_1) < 0$

Alors par le théorème 4 de Varga :

$$\alpha(\mathbb{F} + \mathbb{V}) < 0 \iff R_0 < 1 \implies \alpha(Df(x^*)) < 0 \iff R_0 < 1$$

Ainsi la stabilité équivaut  $R_0 < 1$ .

### Démonstration du théorème de la matrice de prochaine génération

- Sachant que  $R_0$  est le nombre d'infections secondaires produites par une seule infection typique dans une population sensible, alors comment définir  $R_0$  lorsqu'il existe plusieurs types d'individus infectés. Le concept clé est que nous devons maintenant faire la moyenne du nombre prévu de nouvelles infections sur tous les types d'infections possibles.
- Supposons que nous ayons un système dans lequel plusieurs individus sont infectés, la matrice de la prochaine génération est définie comme la matrice carrée  $K$  dans laquelle l'élément  $k_{i,j}$ , est le nombre attendu d'infections secondaires de type  $i$  causées par un seul individu infecté par le type  $j$ .
- Ayant la matrice  $K$ , il est naturel de penser que  $R_0$  représentera la plus grande entrée de  $K$ . le nombre de base de la reproduction est donné par le rayon spectral de  $K$ , la valeur propre dominante de  $K$ .

Soit  $x^*$  un point d'équilibre sans maladies, en posant  $\nu = \phi^+ - \phi^-$ ,

$$Df(x^*) = D\phi(x^*) + D\nu(x^*)$$

$$D\phi(x^*) = \begin{pmatrix} 0 & 0 \\ 0 & \mathbb{F} \end{pmatrix}, D\nu(x^*) = \begin{pmatrix} J_1 & J_2 \\ 0 & \mathbb{V} \end{pmatrix}$$

Avec :

$$\mathbb{F} = \left( \frac{\partial \phi_i}{\partial x_j}(x^*) \right)_{\substack{p+1 \leq i, j \leq n}}, \mathbb{V} = \left( \frac{\partial \nu_i}{\partial x_j}(x^*) \right)_{\substack{p+1 \leq i, j \leq n}}, J_1 = \left( \frac{\partial \nu_i}{\partial x_j}(x^*) \right)_{\substack{1 \leq i, j \leq p}}, J_2 = \left( \frac{\partial \nu_i}{\partial x_j}(x^*) \right)_{\substack{1 \leq i \leq p \\ p+1 \leq j \leq n}}$$

Remarquons que  $\mathbb{F} \geq 0$  et comme  $x^*$  point d'équilibre,  $\mathbb{V}$  est une matrice de Metzler.

### Approximation au voisinage de l'équilibre :

De plus  $\phi = 0$ , car  $x^*$  est sans maladies, nous sommes près de l'équilibre le comportement du système est approximé par le système linéarisé, on se place ainsi dans le cadre de cette approximation.

$$\dot{x} = D\nu(x^*)(x - x^*)$$

$$(\dot{x}_1, \dots, \dot{x}_n)^T = \begin{pmatrix} J_1 & J_2 \\ 0 & \mathbb{V} \end{pmatrix} (x_1 - x_1^*, \dots, x_m - x_m^*, x_{m+1}, \dots, x_n)^T$$

En posant  $x_{infectés}(t) = (x_{m+1}, \dots, x_n)^T$ ,

$$\dot{x}_{infectés}(t) = \mathbb{V} x_{infectés}(t)$$

$$x_{infectés}(t) = \exp(\mathbb{V}t) x_{infectés}(0)$$

Les individus infectés de l'état initial jusqu'à l'équilibre est :

$$\int_0^{+\infty} x_{infectés}(t) dt = \int_0^{+\infty} \exp(\mathbb{V}t) dt x_{infectés}(0) = -\mathbb{V}^{-1} x_{infectés}(0)$$

Ces individus vont générer d'autres cas selon la matrice d'infection  $\mathbb{F}$ , le nombre total d'infections produites est :

$$\int_0^{+\infty} \mathbb{F} x_{infectés}(t) dt = -\mathbb{F}\mathbb{V}^{-1} x_{infectés}(0)$$

D'où :

$$K = -\mathbb{F}\mathbb{V}^{-1} \implies R_0 = \rho(-\mathbb{F}\mathbb{V}^{-1})$$

# Annexe

## Code Python : 1<sup>ère</sup> Simulation

```
1 import numpy as np
2 from matplotlib import pyplot as plt
3 import random
4
5 ##Simulation des résultats théoriques##
6
7 #On se propose d'implémenter la structure de graphes à l'aide d'une liste d'adjacence#
8
9 #Ainsi un graphe G=[s_l,[voisin_l,...,voisin_n],[r_l1,...,r_ln],tau_l,m_l],..., [s_n,[voisin_l,...,voisin_m],
10 [r_n1,...,r_nm],tau_n,m_n]]#
11
12 #s_i re présente un individu, la liste [voisin_l,...,voisin_n] ses voisins(individus avec lesquels il est en contact),
13 [r_l1,...,r_ln] la probabilité d'infection par les autres individus, tau_i sa durée caractéristique d'infection et m_i la
14 probabilité de décès à un instant donné de l'infection#
15
16 def graphe_vide(n:int)->list:
17     '''Cette fonction permet de créer un graphe vide avec n noeuds et 0 liens entre eux'''
18     res=[]
19     for i in range(n):
20         res+=[[i+1,[],[]]]
21     return res
22
23 def voisins(G:list,i:int)->list:
24     '''Cette fonction renvoie la liste des voisins d'un noeud i dans un graphe G'''
25     res=G[i-1][1]
26     return res
27
28 def creer_lien(G,i,j):
29     '''Cette procédure permet de créer un lien entre i et j dans un graphe G'''
30     G[i-1][1].append(j)
31     G[j-1][1].append(i)
32     G[i-1][1].sort()
33     G[j-1][1].sort()
34
35 def random_matrix(n,inf,sup):
36     '''Cette fonction renvoie une matrice avec des coefficients aléatoirement choisis dans une plage bien définie'''
37     res=[]
38     for i in range(n):
39         line=[]
40         for j in range(n):
41             if i==j:
42                 line+=[0]
43             else:
44                 line+=[random.uniform(inf,sup)]
45         res+=[line]
46     return res
```

# Annexe

## Code Python : 1<sup>ère</sup> Simulation

```
45 def random_column(n,type,inf,sup):
46     '''Cette fonction renvoie une matrice colonne avec des coefficients aléatoirement choisis selon le type demandé dans
une plage bien définie'''
47     res=[]
48     if type=='reel':
49         for i in range(n):
50             res+= [random.uniform(inf,sup)]
51     if type=='entier':
52         for i in range(n):
53             res+= [random.randint(inf,sup)]
54     return res
55
56 def show_matrix(T):
57     '''Cette fonction permet de visualiser une matrice'''
58     for line in T:
59         print(line)
60
61 def random_coeff_transmission(G,inf,sup,inf1,sup1,inf2,sup2):
62     '''Cette fonction permet d'affecter au graphe G des probabilités aléatoires de transmission dans un intervalle dt entre
les individus du graphe, les durées caractéristiques de l'infection et les probabilités de décès, ces coefficients sont
choisies dans un intervalle donné'''
63     R=random_matrix(len(G),inf,sup)
64     T=random_column(len(G),'entier',inf1,sup1)
65     M=random_column(len(G),'reel',inf2,sup2)
66     for i in range(len(G)):
67         vois=voisins(G,G[i][0])
68         for j in vois:
69             G[i][2]+=[R[i][j-1]]
70             G[i]+=[T[i],M[i]]
71     return G
72
73 def transmission_probability(G,t,i,j):
74     '''Cette fonction renvoie la probabilité de transmission entre les individus i et j d'un graphe G à l'instant t'''
75     vois=voisins(G,i)
76     if j not in vois:
77         return 0
78     else:
79         return 1-np.exp(-(G[i-1][2][G[i-1][1].index(j)])*t)
```

# Annexe

## Code Python : 1<sup>ère</sup> Simulation

```
81 def transmission_matrix(G,t):
82     '''Cette fonction renvoie la matrice de transmission du graphe G à l'instant t'''
83     res=[]
84     for i in range(len(G)):
85         ligne=[]
86         for j in range(len(G)):
87             if j+1 not in G[i][1]:
88                 ligne+=[]
89             else:
90                 ligne+=[transmission_probability(G,t,i+1,j+1)]
91         res+=[ligne]
92     return res
93
94 def matrix_x_column(A,x):
95     '''Cette fonction renvoie une liste contenant les listes des probabilités d'infection des individus de l'instant
96     initial à l'instant t pour la population G à partir d'un état initial p0'''
97     n=len(A)
98     res=[]
99     for i in range(n):
100         sum=0
101         for j in range(n):
102             sum+=A[i][j]*x[j]
103         res+=[sum]
104     return res
105
106 def infection_probabilities(G,t,p0):
107     '''Cette fonction renvoie une liste contenant les listes des probabilités d'infection des individus de l'instant
108     initial à l'instant t pour la population G à partir d'un état initial p0'''
109     res=[p0]
110     infected=[]
111     for j in range(len(res[0])):
112         if res[len(res)-1][j]==1:
113             infected+=[]
114
115     i=1
116     while i<t:
117         T=transmission_matrix(G,i)
118         a=matrix_x_column(T,res[len(res)-1])
119         for j in range(len(a)):
120             if a[j]>1 or j in infected:
121                 a[j]=1
122         res+=[a]
123         i+=1
124     return res
```

# Annexe

## Graphe de la population considérée : 1<sup>ère</sup> Simulation

```
[1, [2, 15], []]
[2, [1, 12, 14], []]
[3, [14, 15, 17, 22], []]
[4, [12, 17, 18], []]
[5, [10, 16], []]
[6, [7, 11, 17, 25, 26], []]
[7, [6, 16, 26], []]
[8, [9], []]
[9, [8], []]
[10, [5, 11], []]
[11, [6, 10, 17, 18, 20], []]
[12, [2, 4, 14, 15, 18, 21, 23], []]
[13, [14], []]
[14, [2, 3, 12, 13, 21], []]
[15, [1, 3, 12, 17, 22, 25, 26], []]
[16, [5, 7, 17], []]
[17, [3, 4, 6, 11, 15, 16, 22, 23, 25, 26], []]
[18, [4, 11, 12, 22, 23], []]
[19, [20, 26], []]
[20, [11, 19], []]
[21, [12, 14, 22], []]
[22, [3, 15, 17, 18, 21], []]
[23, [12, 17, 18, 24, 25], []]
[24, [23, 26], []]
[25, [6, 15, 17, 23], []]
[26, [6, 7, 15, 17, 19, 24], []]
[27, [28, 29], []]
[28, [27], []]
[29, [27, 30], []]
[30, [29], []]
[31, [32, 33], []]
[32, [31, 33], []]
[33, [31, 32], []]
```

# Annexe

## Matrice aléatoire de la population considérée : 1<sup>ère</sup> Simulation

[1, [2, 15], [0.0032763815471536883, 0.0042655887093318925], 116, 0.00011968838200846785]  
[2, [1, 12, 14], [0.003976940484351248, 0.0018966674036354506, 0.007140472666827515], 378, 0.00019995927430859183]  
[3, [14, 15, 17, 22], [0.0027323683235355, 0.009066156079414744, 0.0036916781007778157, 0.002744353376940498], 111, 0.00023120541873833405]  
[4, [12, 17, 18], [0.008257652771116626, 0.004682273756500029, 0.006355573534223507], 100, 0.0004699928259739991]  
[5, [10, 16], [0.0027644794193266327, 0.0064406654617619715], 243, 0.0009404668277020307]  
[6, [11, 17, 25, 26], [0.008204939000534792, 0.0015914259950801659, 0.007949924604537365, 0.006537521271534095, 0.0016223237711327555], 255, 0.0006190895377669641]  
[7, [6, 16, 26], [0.004429055875680989, 0.004622692072341597, 0.008440875070694652], 268, 0.0004089873416542379]  
[8, [9], [0.005351121386645782], 355, 0.0007955673867984787]  
[9, [8], [0.004930549293873429], 306, 0.0005037619343484045]  
[10, [5, 11], [0.007541087310819459, 0.005559738706637991, 253, 0.0003019835060047906]  
[11, [6, 10, 17, 18, 20], [0.009476814181456768, 0.0011756084166070833, 0.009330636044797357, 0.006595582995271426, 0.0069883525515195945], 307, 0.00041108511254529985]  
[12, [2, 4, 14, 15, 18, 21, 23], [0.003559439823935663, 0.002377808198991901, 0.001235800570399022, 0.00800452773280254, 0.007514548548132448, 0.0017019101974703173, 0.005683467446378362], 251, 0.0008576317215331715]  
[13, [14], [0.00888816501892252], 288, 0.00018387278187533277]  
[14, [2, 3, 12, 13, 21], [0.005115171663798195, 0.00235386810406758, 0.0016363526617590985, 0.0038375034705428516, 0.006958903459154055], 349, 0.00027240977626992154]  
[15, [1, 3, 12, 17, 22, 25, 26], [0.0013981874392566783, 0.0013014573612684633, 0.0013652123107169693, 0.009318682164134204, 0.002351446952736666, 0.002247300589376015, 0.030523645008625473], 289, 0.0002564804554816647]  
[16, [5, 7, 17], [0.007699213318544966, 0.008858830514271141, 0.0016571733526110526], 237, 0.00017277408813322226]  
[17, [3, 4, 11, 15, 16, 22, 23, 25, 26], [0.004377441678360239, 0.006209825262222289, 0.005549086630562283, 0.004690538982115096, 0.009436511555775781, 0.004350300590543201, 0.0036977505486024105, 0.004356981214523371, 0.008215801001002485, 0.0035149793397061223], 315, 0.0006673495268805579]  
[18, [4, 11, 12, 22, 23], [0.00925048445457246, 0.009459857936625123, 0.004436341282218397, 0.004790029895414177, 0.008832932225242868], 367, 0.0005167982699429601]  
[19, [20, 26], [0.008519243457478083, 0.0060698380001968974], 344, 0.00834224710624865777]  
[20, [11, 19], [0.0016396759277380807, 0.0003262277193811131, 328, 0.000556757378561663]  
[21, [2, 14, 22], [0.0026381647246975536, 0.00614957874743175, 0.004272979310412487], 291, 0.0008618923918268285]  
[22, [3, 15, 17, 18, 21], [0.0018607234548556999, 0.002928435796325076, 0.007569208141819231, 0.0011571624513687484, 0.006388335663277425], 106, 0.0007031422313059147]  
[23, [12, 17, 18, 24, 25], [0.00207365387713158, 0.001368424157781074, 0.0019272742277894773, 0.0016569209127067200, 0.005605561143745725], 364, 0.0003846468961844034]  
[24, [23, 26], [0.009911363689031592, 0.0017923115817388728], 187, 0.0008494163936740655]  
[25, [6, 15, 17, 23], [0.006142256781466305, 0.008618298416245031, 0.00430167972436005, 0.007284450874223072], 223, 0.0008927477020606137]  
[26, [6, 7, 15, 17, 19, 24], [0.006291246737536206, 0.0059656523440478014, 0.005498340300654059, 0.002291392432607046, 0.004791207861975831, 0.006327756482415834], 200, 0.001604171611403746]  
[27, [28, 29], [0.0011167044876294243, 0.004454062953500836], 245, 0.0007902821432313956]  
[28, [27], [0.000767874017005628], 175, 0.0006265053230213759]  
[29, [27, 30], [0.008167437077362952, 0.005744931608637256], 300, 0.0008563575613351633]  
[30, [29], [0.0030574466350339464], 200, 0.00050927073237675]  
[31, [32, 33], [0.0071355619736178525, 0.006435384151283527], 354, 0.0007599170910484455]  
[32, [31, 33], [0.004637376767915197, 0.000596203440421317], 196, 0.0004404241536436099]  
[33, [31, 32], [0.0027580807972997544, 0.0050690638789691345], 338, 0.00018049189271204683]



# Annexe

## Code python des Courbes : 1<sup>ère</sup> Simulation

```
218 ##Evolution du nombre d'infectés et de susceptibles##
219 Nombre_infecté=[]
220 for instant in liste_proba_infection:
221     S=0
222     for j in instant:
223         S+=j
224         S1=33-S
225     Nombre_infecté+=[S]
226     Nombre_susc+=[S1]
227
228 plt.plot(t,Nombre_infecté,label="Nombre d'infectés")
229 plt.plot(t,Nombre_susc,label="Nombre de susceptibles")
230 plt.legend()
231 plt.grid()
232 plt.show()
233 ##Evolution du nombre de morts et de rétablis##
234 Nombre_morts=[]
235 Nombre_rétablis=[]
236 for instant in range(len(evolution_morts[0])):
237     S1=0
238     for individu in evolution_morts:
239         S1+=individu[instant]
240     Nombre_morts+=[S1]
241 for instant in range(len(evolution_retab[0])):
242     S2=0
243     for individu in evolution_retab:
244         S2+=individu[instant]
245     Nombre_rétablis+=[S2]
246
247 plt.plot(t,Nombre_morts,label="Nombre de morts")
248 plt.plot(t,Nombre_rétablis,label="Nombre de rétablis")
249 plt.legend()
250 plt.grid()
251 plt.show()
252 ##Evolution du nombre de porteurs##
253 Nombre_porteurs=[]
254 for i in range(len(Nombre_infecté)):
255     res=Nombre_infecté[i]-Nombre_morts[i]-Nombre_rétablis[i]
256     Nombre_porteurs+=[res]
257
258 plt.plot(t,Nombre_porteurs,label="Nombre de porteurs")
259 plt.legend()
260 plt.grid()
261 plt.show()
```

# Annexe

## Code Python et Courbes : 2<sup>ème</sup> Simulation

```
1 import numpy as np
2 from matplotlib import pyplot as plt
3 from scipy.integrate import odeint
4
5 ##Valeurs de la simulation R0>1##
6
7 u1=0
8 u2=0
9 u3=0
10
11 ##Valeurs de la simulation R0<1##
12
13 u1=0.88
14 u2=0.7
15 u3=0.65
16
17 ##
18
19 def Evolution(compartiments,t,beta,gamma,mu):
20     S=compartiments[0]
21     I=compartiments[1]
22     R=compartiments[2]
23     M=compartiments[3]
24     dSdt=-(1-u1)*beta*S*I
25     dIdt=(1-u1)*beta*S*I-(1-u3)*mu*I-(1+u2)*gamma*I
26     dRdt=(1+u2)*gamma*I
27     dMdt=(1-u3)*mu*I
28     return [dSdt,dIdt,dRdt,dMdt]
29
30 t=np.linspace(0,15000,100000)
31 compartiments=[100000,10000,0,0]
32
33 s=odeint(Evolution,compartiments,t,(7e-08,5e-04,2e-04))
34
35 plt.plot(t,s[:,0],label="Nombre de susceptibles")
36 plt.plot(t,s[:,1],label="Nombre d'infectés")
37 plt.plot(t,s[:,2],label="Nombre de rétablis")
38 plt.plot(t,s[:,3],label="Nombre de morts")
39 plt.legend()
40 plt.grid()
41 plt.show()
```