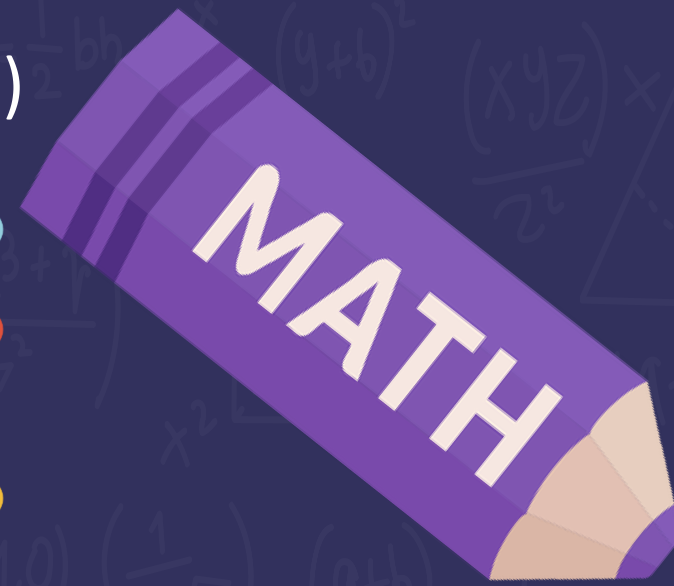
Modélisation mathématique des épidémies

Cas(Covid-19)



Présenté par : Zineb BOUTALEB

Encadré par : Mr. Mohammed ISSOUAL

Plan

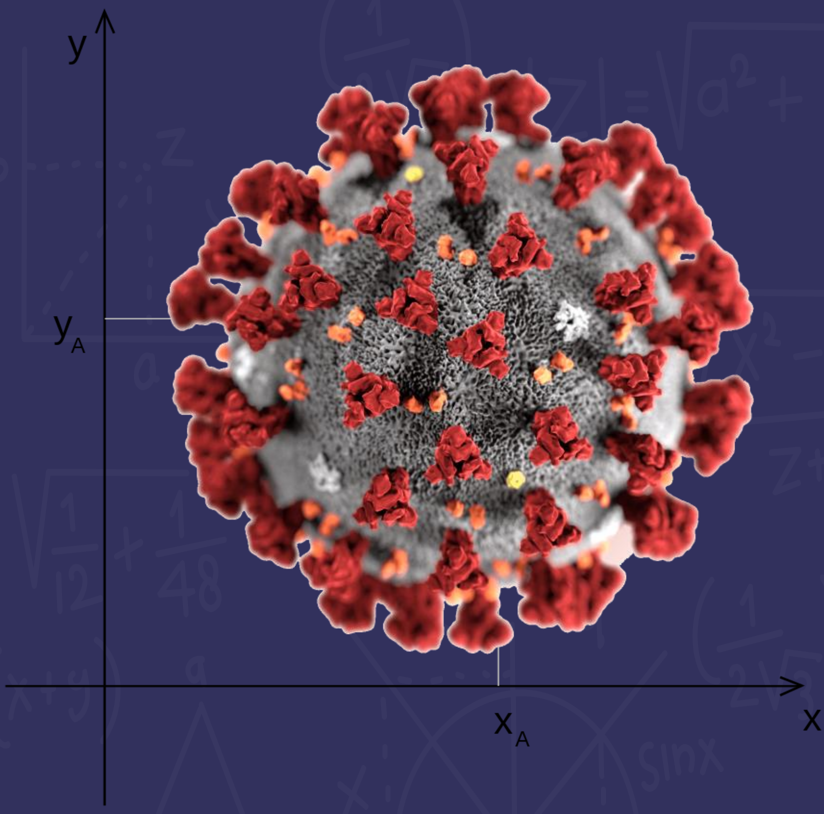
Introduction

- la propagation de plusieurs maladies épidémiques à travers l’histoire
- Accélération de propagation des pandémies due à l’évolution technologique (Transport)
- Les conséquences sanitaires et économiques
- La compréhension de l’évolution épidémique est un élément crucial
- La modélisation mathématique des pandémies

Les mathématiques et la gestion des épidémies

- Acquérir une nouvelle compréhension d'un système
- Organiser et donner un sens aux données biologiques
- Obtenir le comportement de réponse du système
- Rechercher des performances optimales et des stratégies d'intervention
- Faire des prédictions sur le système

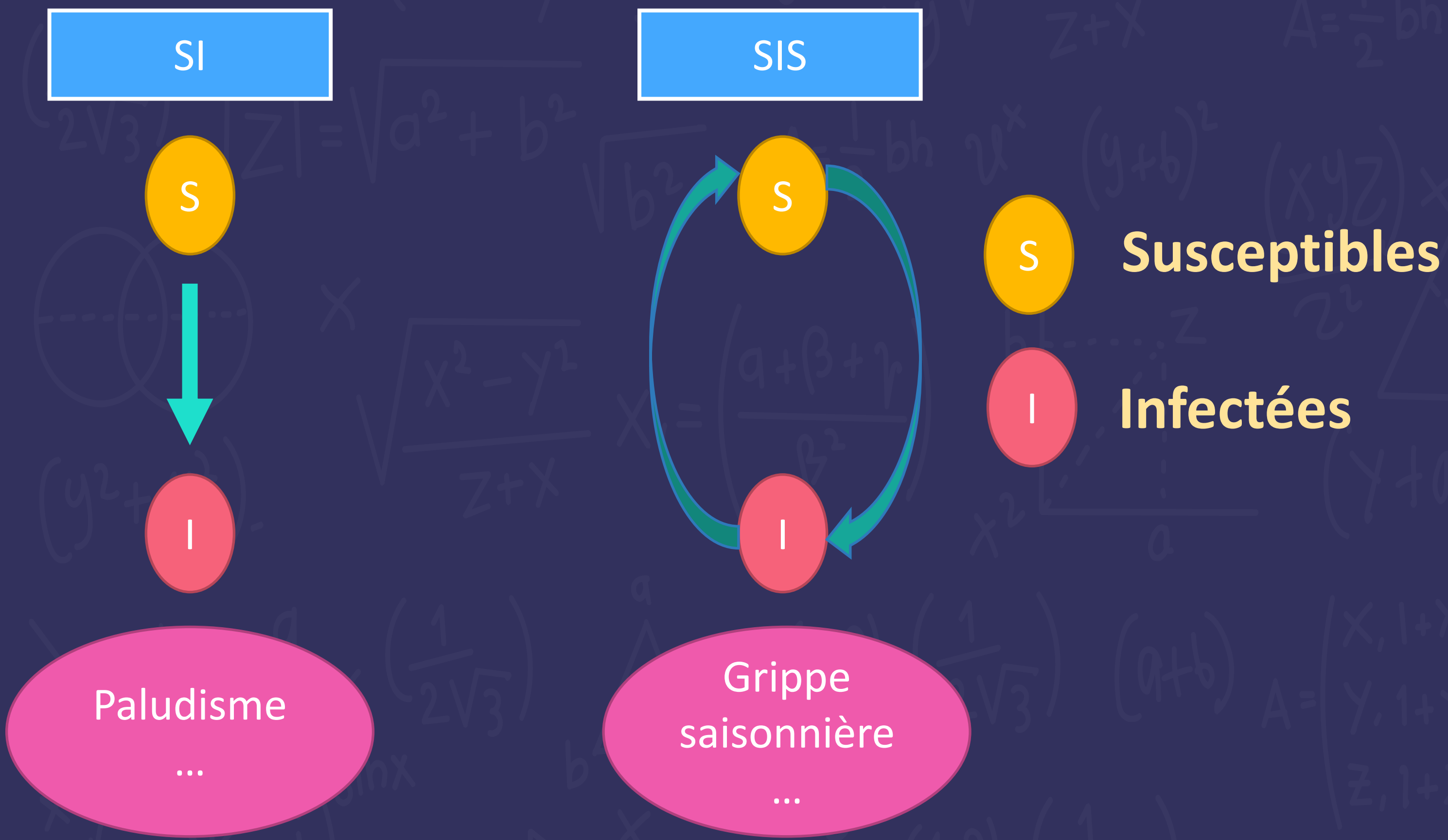
Des modèles mathématiques en épidémiologie



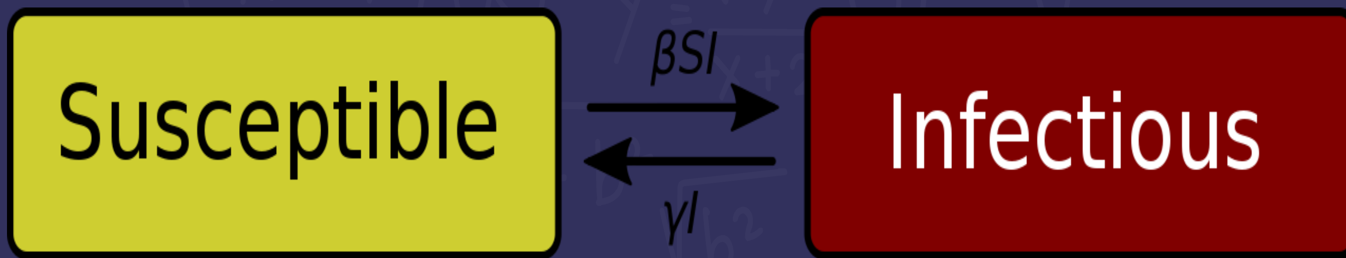
Modèles compartimentaux

- Modèle dynamique
- Modèle déterministe
- Modèle stochastique

Les modèles compartimentaux déterministes (Modèle SIS)



Les modèles compartimentaux déterministes (Modèle SIS)



$$\frac{dS(t)}{dt} = -\beta \frac{SI(t)}{N(t)} + \gamma I(t)$$

$$\frac{dI(t)}{dt} = \beta \frac{SI(t)}{N(t)} - \gamma I(t)$$

S : susceptibles
I : infectées
N : population totale avec N=S+I
B: taux de transmission/infection
ϒ:taux de guérison

Les modèles compartimentaux déterministes (Modèle SIS « R0 »)

Nombre de reproduction de base R_0

le nombre moyen d'individus qu'une personne infectieuse infecte tant qu'elle est contagieuse

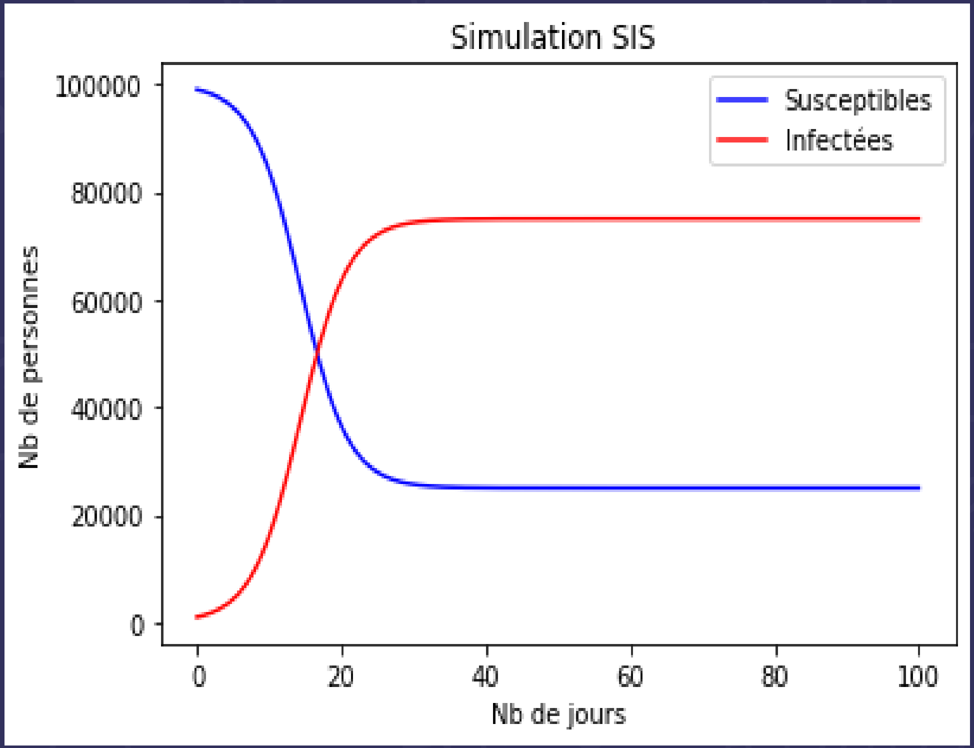
$R_0>1$:la maladie se propage dans la population et devient épidémique

$R_0<1$:l'individu infecté contamine moins d'un autre individu en moyenne, ce qui signifie que la maladie disparaît de la population.

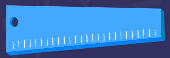
Les modèles compartimentaux déterministes (Modèle SIS)

Paramètres d'entrée (β>ϒ) :

Population : 1000000
Infectés : 1000
Taux d'infection : 0,4
Taux de rétablissement : 0,1

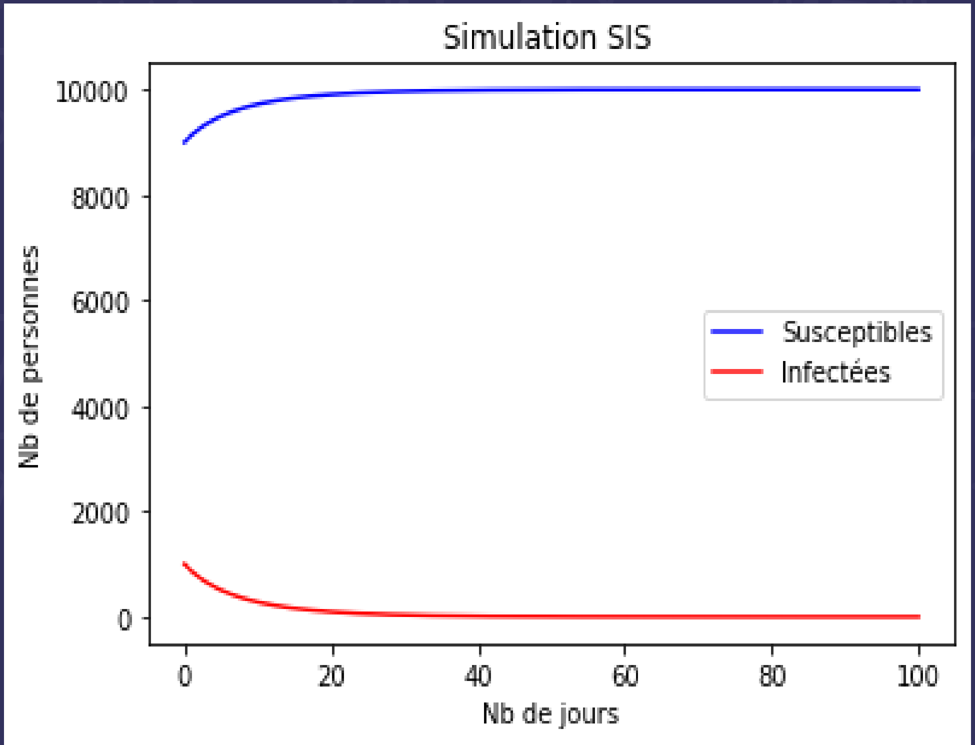


Les modèles compartimentaux déterministes (Modèle SIS)

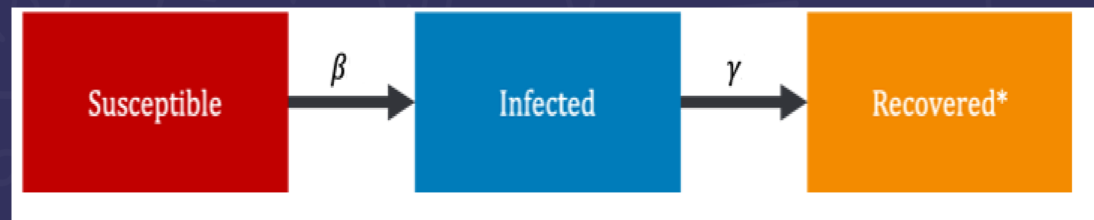


Paramètres d'entrée (β<ϒ) :

Population : 10000
Infectés : 1000
Taux d'infection : 0,5
Taux de rétablissement : 0,6



Les modèles compartimentaux déterministes (Modèle SIR)



$$\frac{dS(t)}{dt} = -\beta \frac{SI(t)}{N(t)}$$

$$\frac{dI(t)}{dt} = \beta \frac{SI(t)}{N(t)} - \gamma I(t)$$

$$\frac{dR(t)}{dt} = \gamma I(t)$$

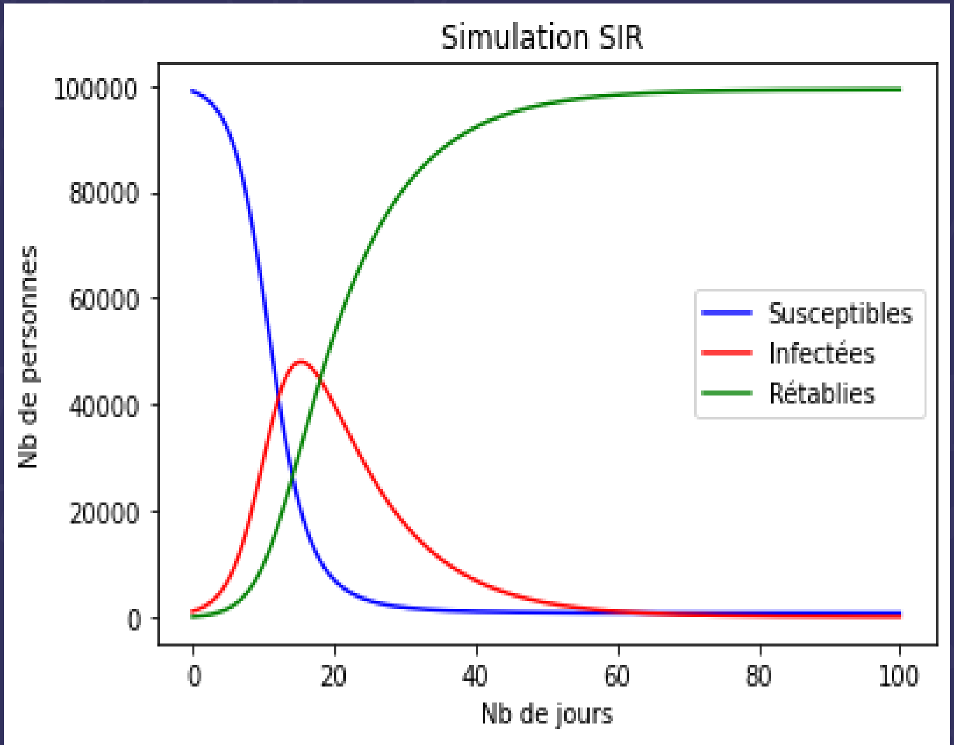
N(t)=S(t)+I(t)+R(t)

N : population totale
B: taux de transmission/infection
ϒ:taux de guérison

Les modèles compartimentaux déterministes (Modèle SIR)

Paramètres d'entrée (β>ϒ) :

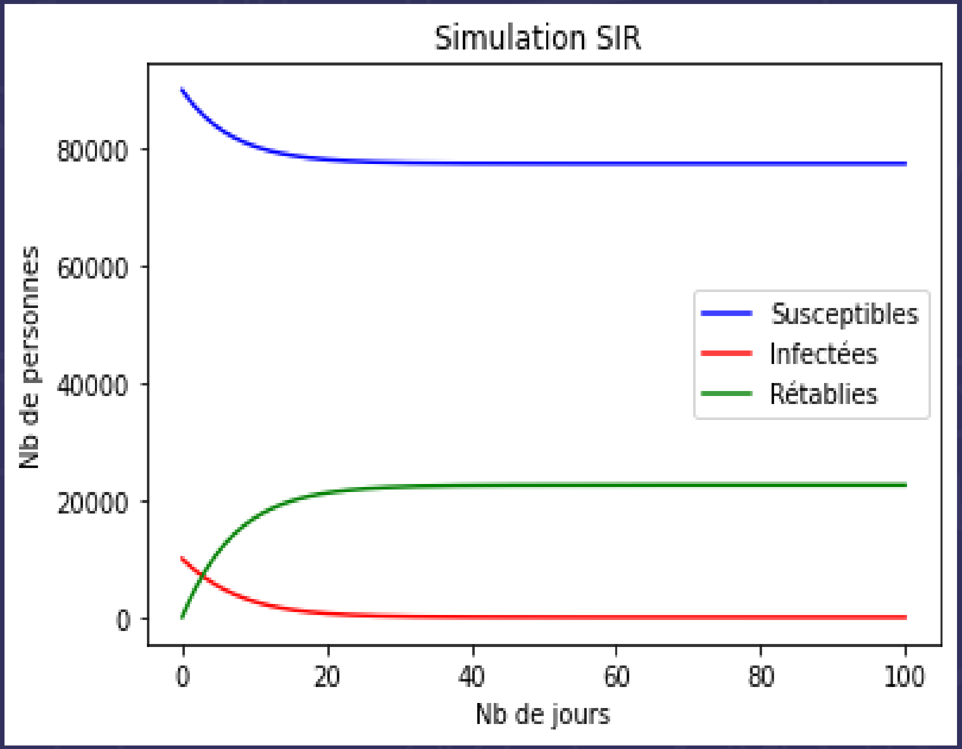
N = 100000
I = 1000
R = 0
β= 0,5
ϒ= 0,1



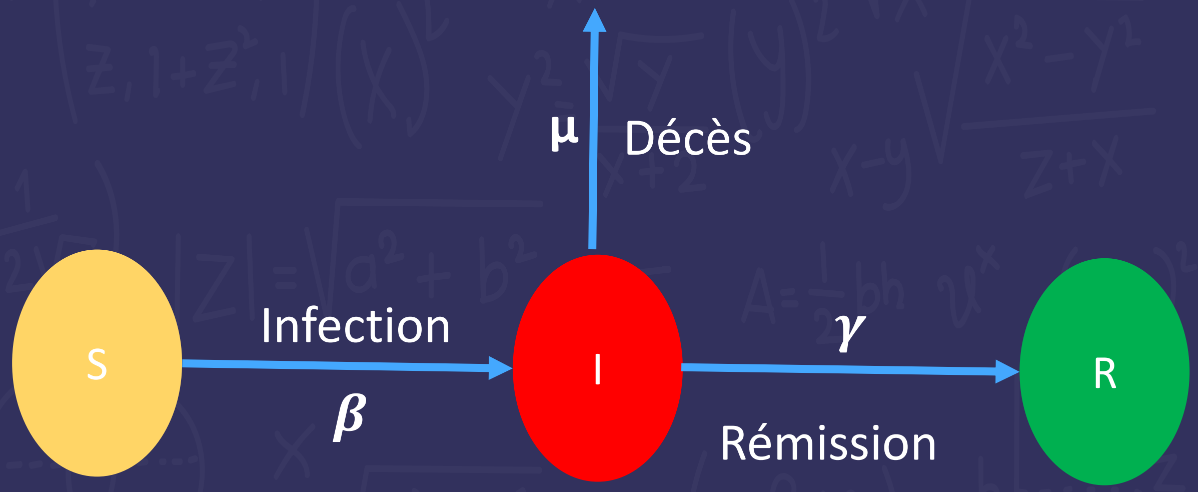
Les modèles compartimentaux déterministes (Modèle SIR)

Paramètres d’entrée (β<ϒ) :

N = 100000
I = 10000
R = 0
β= 0,2
ϒ= 0,3



Application du modèle SIR à Covid-19



$$\frac{dS(t)}{dt} = -\beta \frac{SI(t)}{N(t)}$$

$$\frac{dI(t)}{dt} = \beta \frac{SI(t)}{N(t)} - \gamma I(t) - \mu I(t)$$

$$\frac{dR(t)}{dt} = \gamma I(t)$$

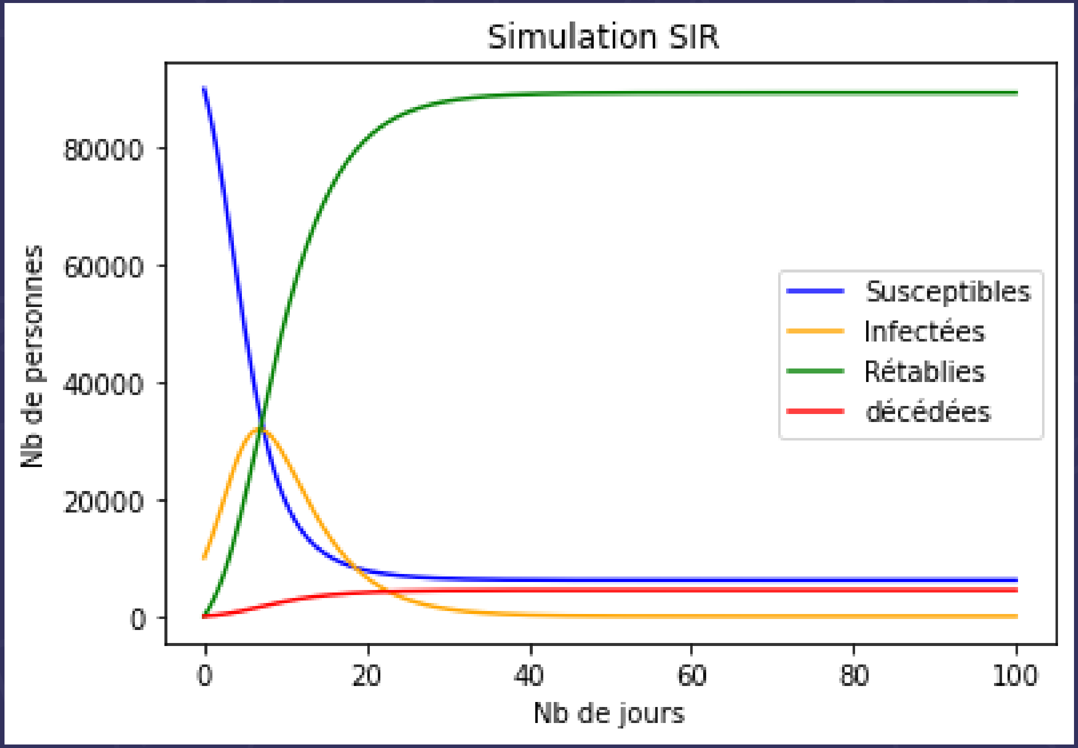
$$\frac{dD(t)}{dt} = \mu I(t)$$

N(t)=S(t)+I(t)+R(t)

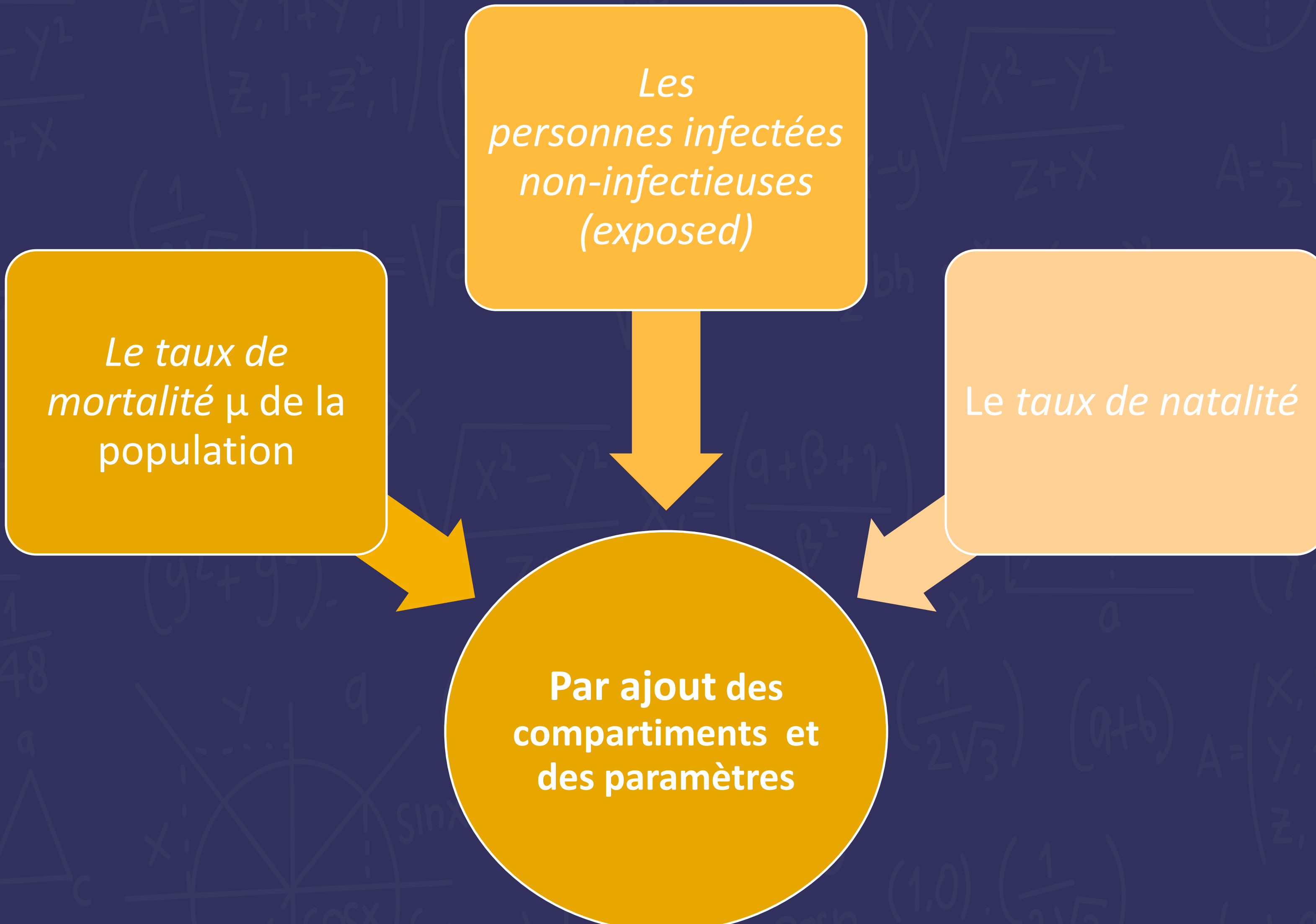
Application du modèle SIR avec des données Covid-19

Paramètres d'entrée :

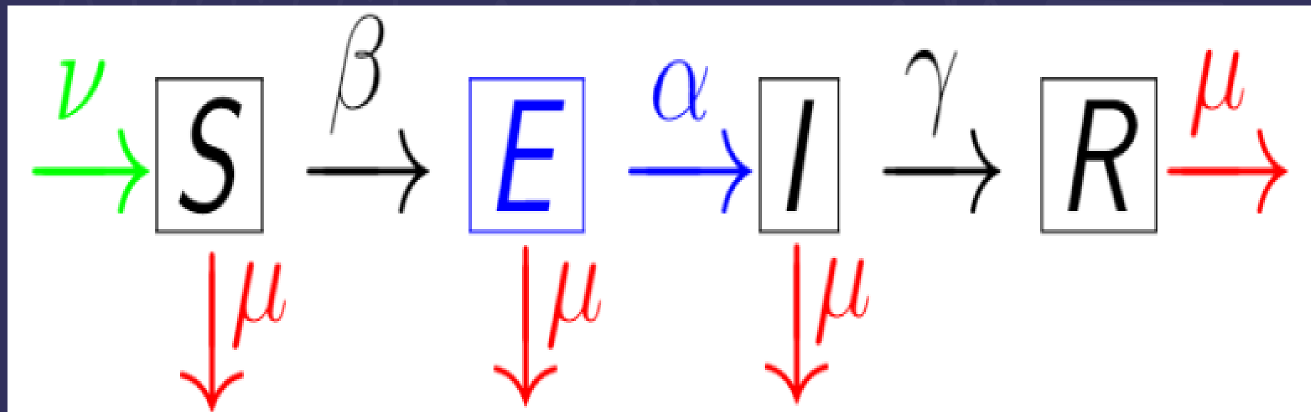
N = 100000
I = 1000
D = 0
R = 0
β= 0.6
ϒ= 0,2
µ=0.01



Amélioration du modèle SIR



Amélioration du modèle SIR



$$\frac{dS(t)}{dt} = -\beta \frac{SI(t)}{N} + \nu N(t) - \mu S(t)$$

$$\frac{dE(t)}{dt} = \beta \frac{SI(t)}{N(t)} - \alpha E(t) - \mu E(t)$$

$$\frac{dI(t)}{dt} = \alpha E(t) - \gamma I(t) - \mu I(t)$$

$$\frac{dR(t)}{dt} = \gamma I(t) - \mu R(t)$$

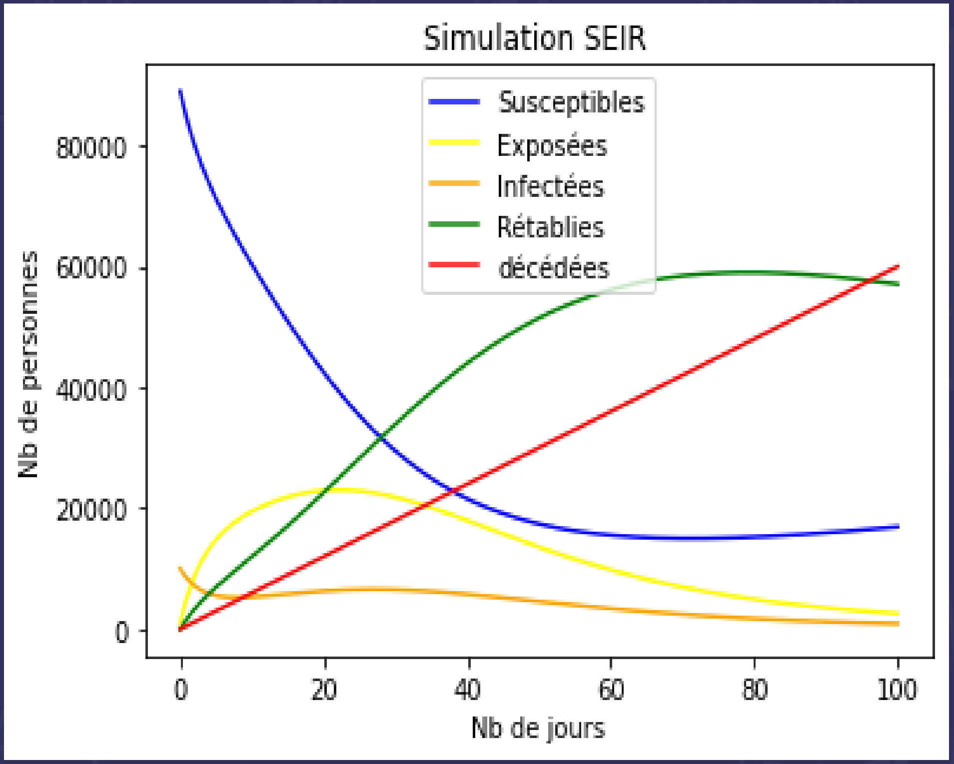
$$\frac{dD(t)}{dt} = \mu N(t)$$

$N(t)=S(t)+E(t)+I(t)+R(t)$

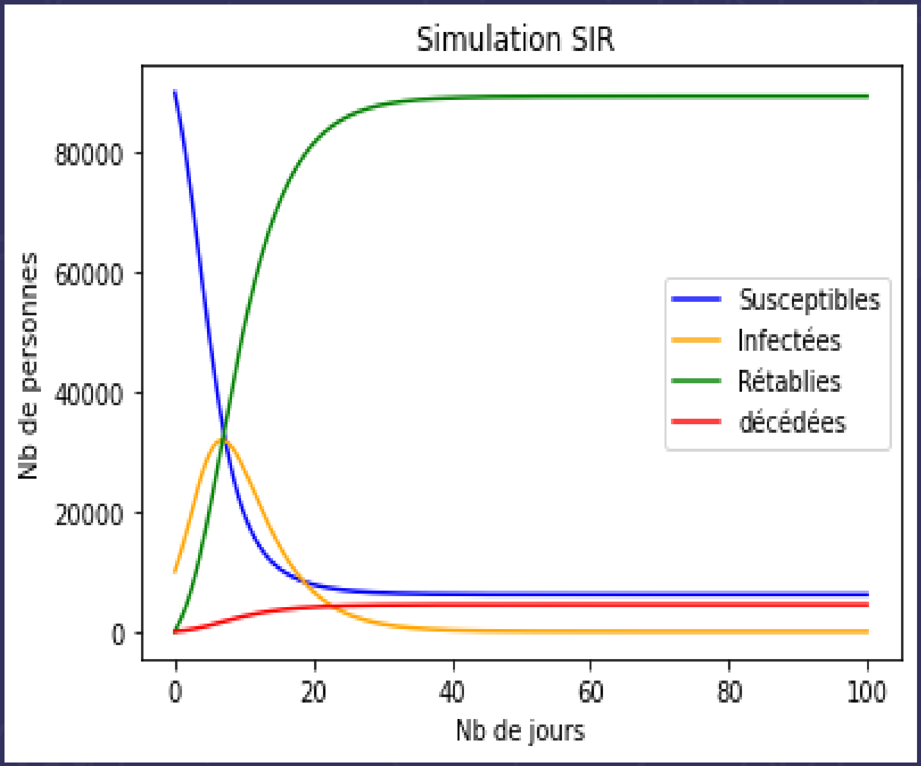
Amélioration du modèle SIR (SEIR)

Paramètres d'entrée (β>ϒ) :

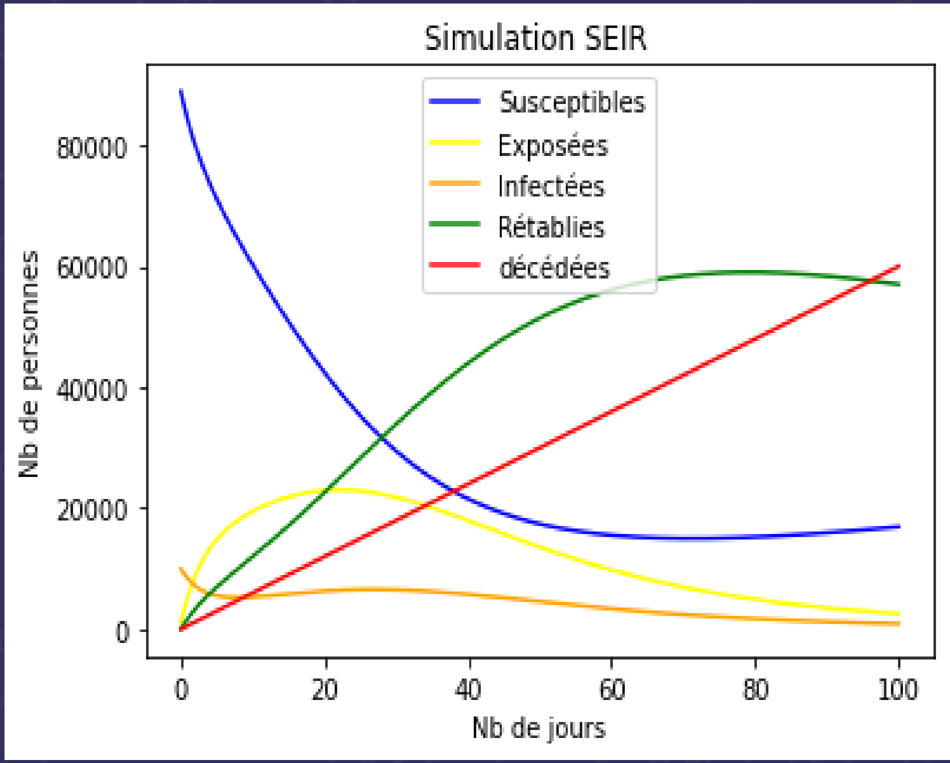
N = 100000
I = 10000
E=1000
D = 0
R = 0
β= 0.6
ϒ= 0,2
Taux d'incubation=0.06
Taux de natalité =0.003
Taux de mortalité=0.006



SIR Vs SEIR



Simulation SIR
80000
60000
40000
20000
0
0
20
40
60
80
100
Nb de jours
Nb de personnes
Susceptibles
Infectées
Rétablies
décédées



Simulation SEIR
80000
60000
40000
20000
0
0
20
40
60
80
100
Nb de jours
Nb de personnes
Susceptibles
Exposées
Infectées
Rétablies
décédées

Amélioration du modèle SIR

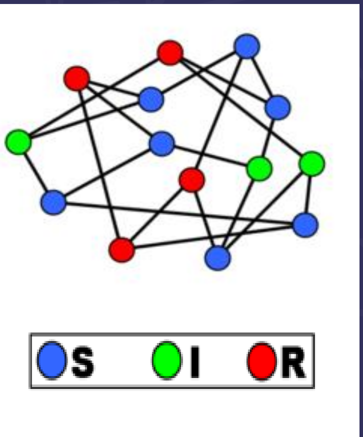
Effet du réseau social

Croissance de la propagation de l'épidémie

La modélisation probabiliste au lieu de la modélisation déterministe.

les nœuds individuels du réseau : réseau de contacts, des villes, de travail…

Par considération des réseaux



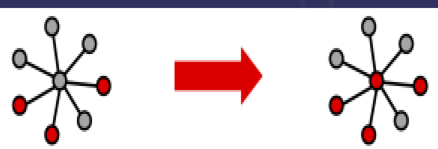
Modèle probabiliste du modèle SIR

infection des nœuds

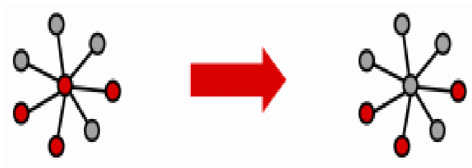
- le nœud central i entouré par plusieurs nœuds infectés

Rétablissement des nœuds

- Ce processus ne dépend pas de ses voisins le nœud rétabli tout seul

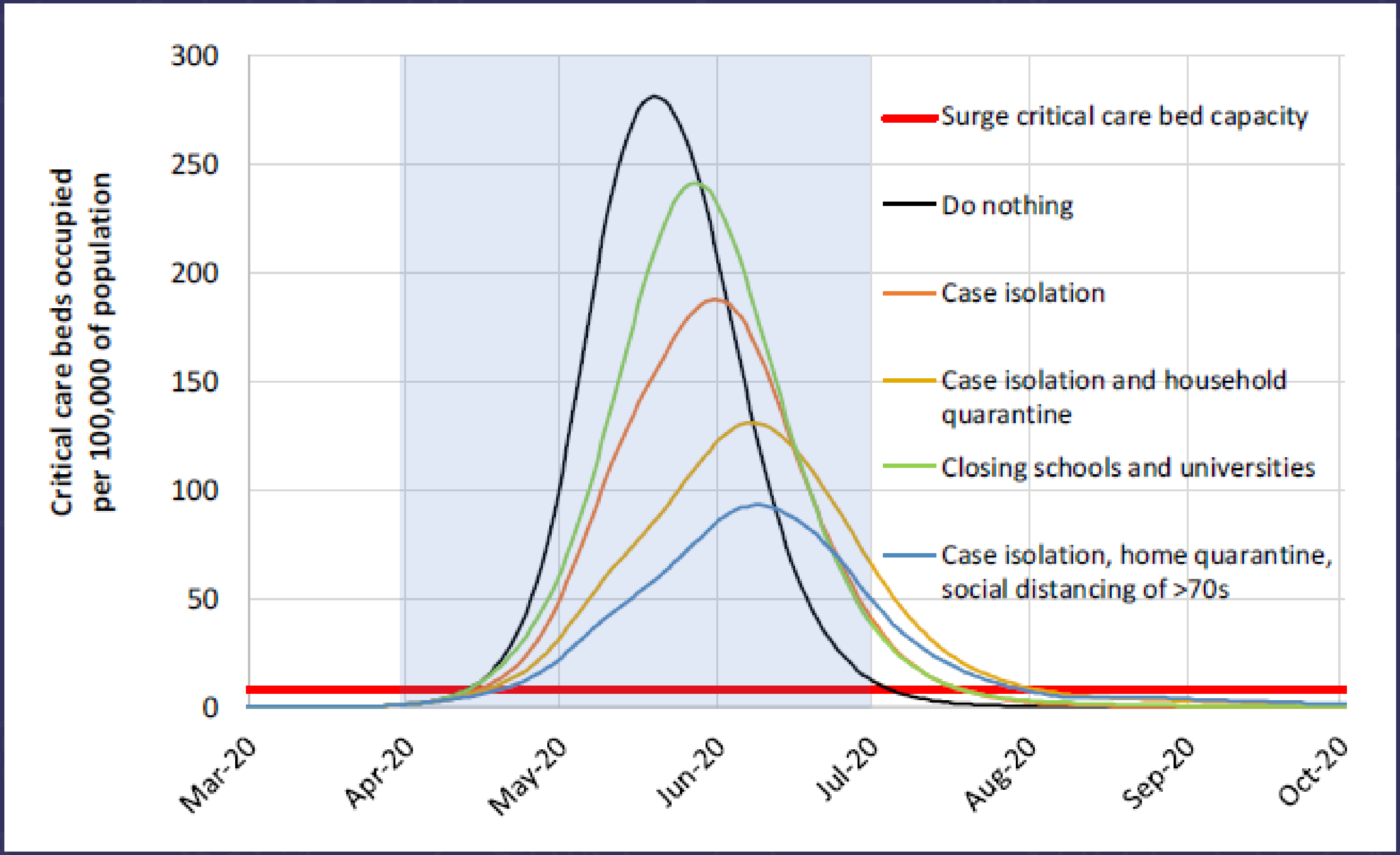


$$P_{inf} = s_i(t)\left(1 - \prod_{j\in\mathcal{N}(i)} \left(1-\beta x_j(t)\delta t\right)\right) \approx \beta s_i(t) \sum_{j\in\mathcal{N}(i)} x_j(t)\delta t$$



$$P_{rec} = \gamma x_i(t)\delta t$$

Le modèle SIR : Outil d'aide à la décision



Source : http://images.math.cnrs.fr/Modelisation-d-une-epidemie-partie-1.html

Conclusion

Merci pour votre attention

Annexes

Le modèle SIS

```
@author: Zineb
"""

import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
N = int(input('population: '))
I0=int(input('personnes infectées initiales: '))
S0 = N - I0
beta=float(input('Taux d infection: '))
gamma=float(input('Taux de rétablissement: '))
t = np.linspace(0, 100, 200)

def eqdiff(y, t, N, beta, gamma):
    S, I = y
    dSdt = -beta * S * I / N + gamma * I
    dIdt = beta * S * I / N - gamma * I

    return dSdt, dIdt

y0 = S0, I0
sol = odeint(eqdiff, y0, t, args=(N, beta, gamma))
S, I = sol.T
plt.title('Simulation SIS ')
plt.plot(t, S, color='blue', label='Susceptibles')
plt.plot(t, I, color='red', label='Infectées')
plt.xlabel('Nb de jours')
plt.ylabel('Nb de personnes')
leg = plt.legend();
plt.show()
```

Le modèle SIR

```python
@author: Zineb
"""
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
N = int(input('population: '))
I0=int(input('personnes infectées initiales: '))
R0=int(input('personnes rétablies initiales: '))
beta=float(input('Taux d infection: '))
gamma=float(input('Taux de rétablissement: '))
S0 = N - I0 - R0
t = np.linspace(0, 100,200)  #en jours
def eqdiff(y, t, N, beta, gamma):
    S, I, R = y
    dSdt = -beta * S * I / N
    dIdt = beta * S * I / N - gamma * I
    dRdt = gamma * I
    return dSdt, dIdt, dRdt
y0 = S0, I0, R0
sol = odeint(eqdiff, y0, t, args=(N, beta, gamma))
S, I, R = sol.T
plt.title('Simulation SIR ')
plt.plot(t, S, color='blue', label='Susceptibles')
plt.plot(t, I, color='red', label='Infectées')
plt.plot(t, R, color='green', label='Rétablies')
plt.xlabel('Nb de jours')
plt.ylabel('Nb de personnes')
leg = plt.legend();
plt.show()
```

Le modèle SIR Covid-19

```python
@author: Zineb
"""
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
N = int(input('population: '))
I0=int(input('personnes infectées initiales: '))
R0=int(input('personnes rétablies initiales: '))
D0=int(input('personnes décédées initiales: '))
beta=float(input('Taux d infection: '))
gamma=float(input('Taux de rétablissement: '))
mu=float(input('Taux de mortalité: '))
S0 = N - I0 - R0
t = np.linspace(0, 100,200)  #en jours
def eqdiff(y, t, N, beta, gamma):
    S, I, R, D = y
    dSdt = -beta * S * I / N
    dIdt = beta * S * I / N - gamma * I - mu*I
    dRdt = gamma * I
    dDdt = mu*I
    return dSdt, dIdt, dRdt, dDdt
y0 = S0, I0, R0, D0
sol = odeint(eqdiff, y0, t, args=(N, beta, gamma))
S, I, R, D = sol.T
plt.title('Simulation SIR ')
plt.plot(t, S, color='blue', label='Susceptibles')
plt.plot(t, I, color='orange', label='Infectées')
plt.plot(t, R, color='green', label='Rétablies')
plt.plot(t, D, color='red', label='décédées')
plt.xlabel('Nb de jours')
plt.ylabel('Nb de personnes')
leg = plt.legend();
plt.show()
```

Le modèle SEIR

```python
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
N = int(input('population: '))
I0=int(input('personnes infectées initiales: '))
E0=int(input('personnes exposées initiales: '))
R0=int(input('personnes rétablies initiales: '))
D0=int(input('personnes décédées initiales: '))
S0 = N - E0- I0 - R0
nu=float(input('Taux de natalité: '))
beta=float(input('Taux d infection: '))
alpha=float(input('Taux d incubation: '))
gamma=float(input('Taux de rétablissement: '))
mu=float(input('Taux de mortalité: '))
t = np.linspace(0, 100,200)
def eqdiff(y, t, N,  nu, beta, alpha, gamma, mu):
    S, E, I, R, D = y
    dSdt =-beta * S * I / N + nu*N - mu*S
    dEdt = beta * S * I / N - alpha*E - mu*E
    dIdt = alpha*E - gamma*I - mu*I
    dRdt = gamma*I - mu*R
    dDdt = mu*N
    return dSdt, dEdt, dIdt, dRdt, dDdt
y0 = S0, E0, I0, R0, D0
sol = odeint(eqdiff, y0, t, args=(N, nu, beta, alpha, gamma, mu))
S, E, I, R, D = sol.T
plt.title('Simulation SEIR ')
plt.plot(t, S, color='blue', label='Susceptibles')
plt.plot(t, E, color='yellow', label='Exposées')
plt.plot(t, I, color='orange', label='Infectées')
plt.plot(t, R, color='green', label='Rétablies')
plt.plot(t, D, color='red', label='décédées')
plt.xlabel('Nb de jours')
plt.ylabel('Nb de personnes')
leg = plt.legend();
plt.show()
```