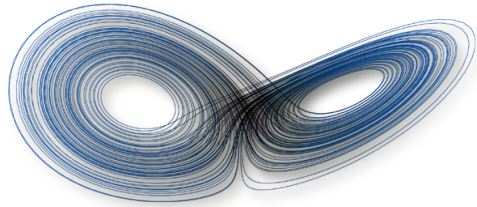


Introduction au calcul scientifique

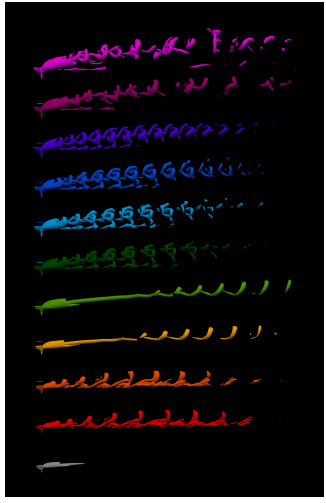
Jean-Christophe Loiseau

jean-christophe.loiseau@ensam.eu
Laboratoire DynFluid
Arts et Métiers, France.

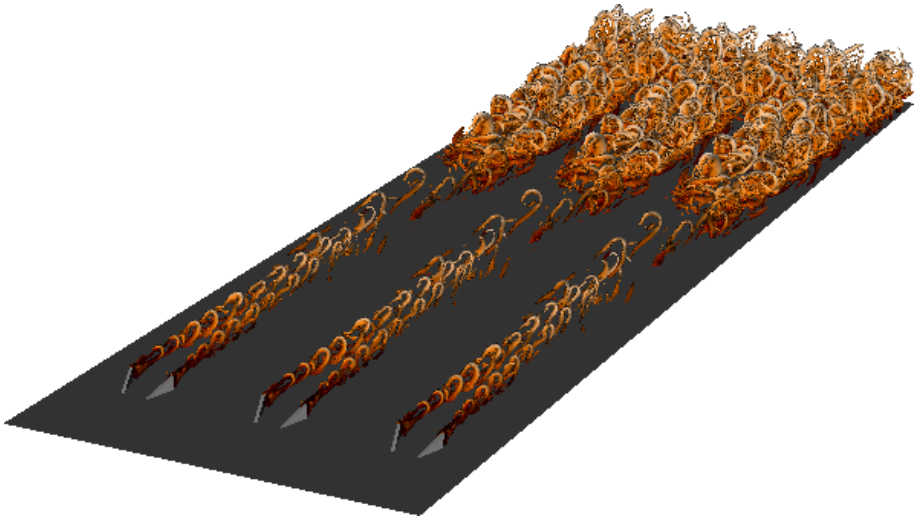
Simulation numérique

Avec l'accroissement de nos capacités de calcul, la **simulation numérique** est devenue aujourd'hui l'un des outils les plus utilisés en ingénierie.

Etre capable de simuler efficacement, précisément et rapidement les **équations différentielles** décrivant l'évolution de systèmes potentiellement très complexes est donc d'une importance capitale.



Simulation numérique



Simulation numérique

Il existe deux grandes familles d'équations différentielles :

- Les **équations différentielles ordinaires** permettent de modéliser des systèmes dont l'évolution ne dépend que du temps. Elles sont en générales de la forme

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u})$$

avec $\boldsymbol{x} \in \mathbb{R}^n$ le vecteur d'état et $\boldsymbol{u}$ une action externe (e.g. loi de contrôle).

- Les **équations aux dérivées partielles** permettent quant à elles de simuler des systèmes dont l'évolution dépend du temps et de l'espace. L'exemple le plus connu sont les équations de Navier-Stokes

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) = -\nabla p + \frac{1}{Re}\nabla^2 \boldsymbol{u}$$

$$\nabla \cdot \boldsymbol{u} = 0$$

où le champ de vitesse $\boldsymbol{u}$ dépend du temps et de l'espace.

Equations différentielles ordinaires

Linear-time invariant dynamical systems

Intéressons-nous tout d'abord à des systèmes linéaires du type

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$$
$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u}.$$

Ici, $\boldsymbol{x} \in \mathbb{R}^n$ est le vecteur d'état, $\boldsymbol{u} \in \mathbb{R}^p$ les inputs et $\boldsymbol{y} \in \mathbb{R}^q$ les mesures faites au cours du temps.

Equations différentielles ordinaires

Linear-time invariant dynamical systems

Si $\boldsymbol{u}(t)$ est connu à l'avance, alors la solution du système s'écrit

$$\boldsymbol{x}(t) = e^{t\boldsymbol{A}}\boldsymbol{x}_0 + \int_0^t e^{(t-\tau)\boldsymbol{A}}\boldsymbol{B}\boldsymbol{u}(\tau)\,\mathrm{d}\tau.$$

Le premier terme correspond à la **réponse naturelle** (ou homogène) tandis que le second décrit la **réponse forcée** du système.

Equations différentielles ordinaires

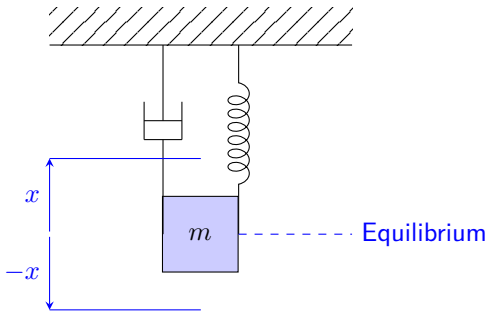
Linear-time invariant dynamical systems

En partant des principes de Newton, les équations du mouvement sont données par

$$\ddot{x} = -2\lambda\dot{x} - x + u(t).$$

En introduisant $x_1 = x$ et $x_2 = \dot{x}$, on peut réécrire le système comme

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -2\lambda \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$



Equations différentielles ordinaires

Linear-time invariant dynamical systems

L'exponentielle d'une matrice est définie comme

$$e^{\boldsymbol{A}} = \boldsymbol{V} e^{\boldsymbol{\Lambda}} \boldsymbol{V}^{-1}$$

où $\boldsymbol{V}$ est la matrice des vecteurs propres et $\boldsymbol{\Lambda}$ est une matrice diagonale dont les entrées sont les valeurs propres.

Pour des problèmes de grande taille, calculer ces valeurs propres et vecteurs propres peut néanmoins s'avérer extrêmement compliqué.

$$\begin{aligned} e^{\boldsymbol{A}} &= \boldsymbol{I} + \boldsymbol{A} + \frac{\boldsymbol{A}^2}{2} + \frac{\boldsymbol{A}^3}{3!} + \cdots \\ &= \boldsymbol{V}\boldsymbol{V}^{-1} + \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1} + \frac{\boldsymbol{V}\boldsymbol{\Lambda}^2\boldsymbol{V}}{2} + \cdots \\ &= \boldsymbol{V}\left(\boldsymbol{I} + \boldsymbol{\Lambda} + \frac{\boldsymbol{\Lambda}^2}{2} + \cdots\right)\boldsymbol{V}^{-1} \\ &= \boldsymbol{V} e^{\boldsymbol{\Lambda}} \boldsymbol{V}^{-1} \end{aligned}$$

Equations différentielles ordinaires

Linear-time invariant dynamical systems

Comment faire alors pour simuler $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$? Pour cela, on va avoir besoin de **discrétiser** l'équation, i.e. considérer la solution à des intervalles de temps réguliers $t_k = k\Delta t$ avec Δt le **pas de temps**.

Equations différentielles ordinaires

Linear-time invariant dynamical systems

En absence de forçage externe, la solution au temps $t_{k+1} = (k+1)\Delta t$ est donnée par

$$\boldsymbol{x}_{k+1} = e^{\Delta t \boldsymbol{A}} \boldsymbol{x}_k.$$

Si Δt est suffisament petit, on peut alors approximer l'exponentielle par son développement de Taylor

$$e^{\Delta t \boldsymbol{A}} = \boldsymbol{I} + \Delta t \boldsymbol{A} + \mathcal{O}(\Delta t^2)$$

ce qui nous donne alors

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta t \boldsymbol{A} \boldsymbol{x}_k + \mathcal{O}(\Delta t^2).$$

Equations différentielles ordinaires

Linear-time invariant dynamical systems

L'approximation $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta t \boldsymbol{A}\boldsymbol{x}_k$ est ce qu'on appelle le **schéma d'Euler explicite du premier ordre**. Il correspond à l'approximation suivante du système

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} \rightarrow \frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{\Delta t} = \boldsymbol{A}\boldsymbol{x}_k.$$

Il en existe de nombreuses variantes plus ou moins précises et plus ou moins couteuses en temps de calcul.

Equations différentielles ordinaires

Linear-time invariant dynamical systems

Schéma d'Euler implicite du premier ordre : La solution au temps t_k peut également s'écrire

$$e^{-\Delta t \boldsymbol{A}} \boldsymbol{x}_{k+1} = \boldsymbol{x}_k.$$

De la même façon, un développement limité au premier ordre donne

$$\begin{aligned} (\boldsymbol{I} - \Delta t \boldsymbol{A})\, \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k \\ \boldsymbol{x}_{k+1} &= (\boldsymbol{I} - \Delta t \boldsymbol{A})^{-1}\, \boldsymbol{x}_k \end{aligned}$$

Il correspond à l'approximation suivante du système

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} \rightarrow \frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{\Delta t} = \boldsymbol{A}\boldsymbol{x}_{k+1}.$$

Equations différentielles ordinaires

Linear-time invariant dynamical systems

Schéma de Crank-Nicholson du deuxième ordre : La solution au temps t_k peut également s'écrire

$$e^{-\Delta t/2 \boldsymbol{A}} \boldsymbol{x}_{k+1} = e^{\Delta t/2 \boldsymbol{A}} \boldsymbol{x}_k$$

Un développement limité de l'exponentielle à droite et à gauche du signe égal donne

$$\left(\boldsymbol{I} - \frac{\Delta t}{2}\boldsymbol{A}\right) \boldsymbol{x}_{k+1} = \left(\boldsymbol{I} + \frac{\Delta t}{2}\boldsymbol{A}\right) \boldsymbol{x}_k$$

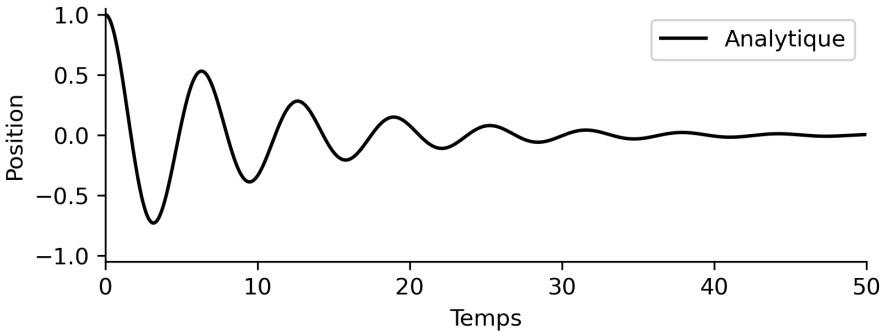
$$\boldsymbol{x}_{k+1} = \left(\boldsymbol{I} - \frac{\Delta t}{2}\boldsymbol{A}\right)^{-1} \left(\boldsymbol{I} + \frac{\Delta t}{2}\boldsymbol{A}\right) \boldsymbol{x}_k.$$

Il correspond à l'approximation suivante du système

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} \rightarrow \frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{\Delta t} = \frac{1}{2}\left(\boldsymbol{A}\boldsymbol{x}_{k+1} + \boldsymbol{A}\boldsymbol{x}_k\right).$$

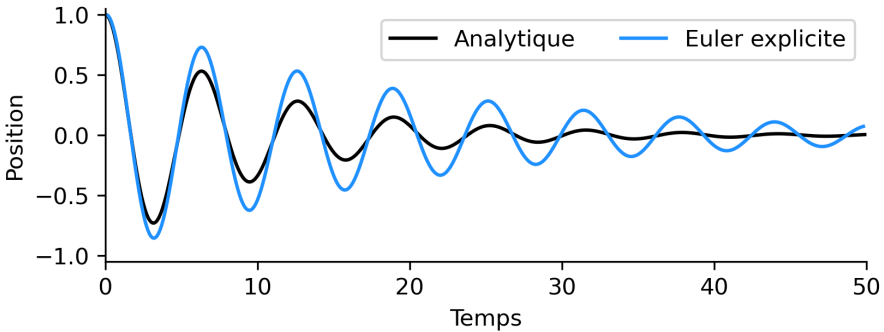
Equations différentielles ordinaires

Linear-time invariant dynamical systems



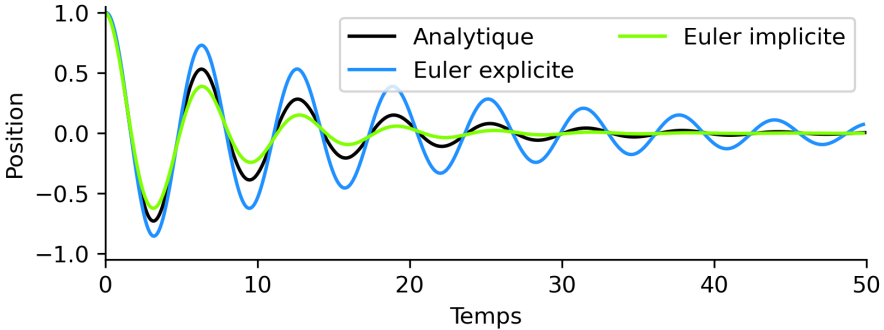
Equations différentielles ordinaires

Linear-time invariant dynamical systems



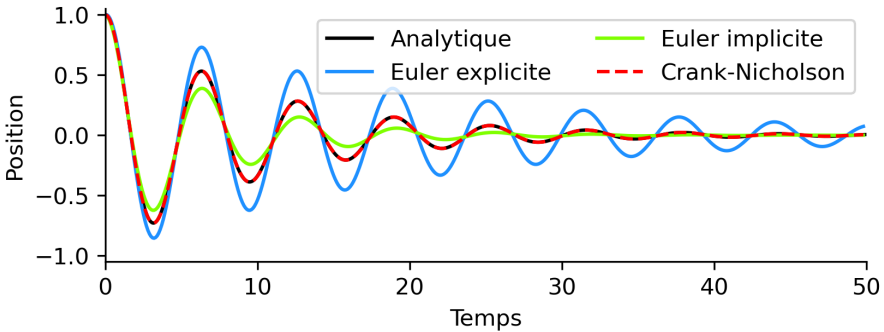
Equations différentielles ordinaires

Linear-time invariant dynamical systems



Equations différentielles ordinaires

Linear-time invariant dynamical systems



Equations différentielles ordinaires

Linear-time invariant dynamical systems

Méthode	Implémentation	Coût de calcul	Robustesse	Choix de Δt
Explicite	Facile	Peu cher	Moyenne	Petit Δt
Implicite	Plus compliqué	Plus cher	Excellente	Grand Δt

Equations différentielles ordinaires

Systèmes non-linéaires

La plupart des systèmes d'intérêt en pratique sont non-linéaires

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

avec $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ une fonction non-linéaire qui ne respecte donc pas le principe de superposition, i.e.

$$\boldsymbol{f}(\alpha \boldsymbol{x}) \neq \alpha \boldsymbol{f}(\boldsymbol{x}) \quad \text{et} \quad \boldsymbol{f}(\boldsymbol{x} + \boldsymbol{y}) \neq \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{f}(\boldsymbol{y}).$$

Leur simulation repose néanmoins sur les mêmes principes.

Equations différentielles ordinaires

Systèmes non-linéaires

- **Schéma d'Euler explicite du premier ordre**

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta t \boldsymbol{f}(\boldsymbol{x}_k)$$

- **Schéma d'Euler implicite du premier ordre**

$$\boldsymbol{x}_{k+1} - \Delta t \boldsymbol{f}(\boldsymbol{x}_{k+1}) = \boldsymbol{x}_k$$

- **Schéma de Crank-Nicholson du deuxième ordre**

$$\boldsymbol{x}_{k+1} - \frac{\Delta t}{2} \boldsymbol{f}(\boldsymbol{x}_{k+1}) = \boldsymbol{x}_k + \frac{\Delta t}{2} \boldsymbol{f}(\boldsymbol{x}_k)$$

Equations différentielles ordinaires

Python

En python, la simulation d'équations différentielles ordinaires se fait à l'aide de la fonction `solve_ivp` du package `scipy.integrate` donnant accès à de nombreuses méthodes d'intégration différentes.

Equations différentielles ordinaires

Python

```
1  from scipy.integrate import solve_ivp
2
3  # --> dx/dt = a x.
4  def dynsys(t, x, a):
5      return a*x
6
7  # --> Parametres pour l'integration.
8  tspan = (0.0, 10.0)
9  teval = np.linspace(tspan[0], tspan[1], 1024)
10 a = -0.1
11 x0 = 1.0
12
13 # --> Simulation.
14 sol = solve_ivp(
15     lambda t, x: dynsys(t, x, a),
16     tspan,
17     x0,
18     teval=teval
19     )
20
```

Equations différentielles ordinaires

Exemple : le système de Lorenz

Le système de Lorenz est un modèle très simplifié du phénomène de convection atmosphérique. Il est donné par

$$\begin{aligned}\dot{x} &= \sigma(y - x)\\ \dot{y} &= x(\rho - z) - y\\ \dot{z} &= xy - \beta z.\end{aligned}$$

C'est l'un des systèmes historiquement les plus importants en théorie des systèmes dynamiques et notamment en théorie du chaos.

Equations différentielles ordinaires

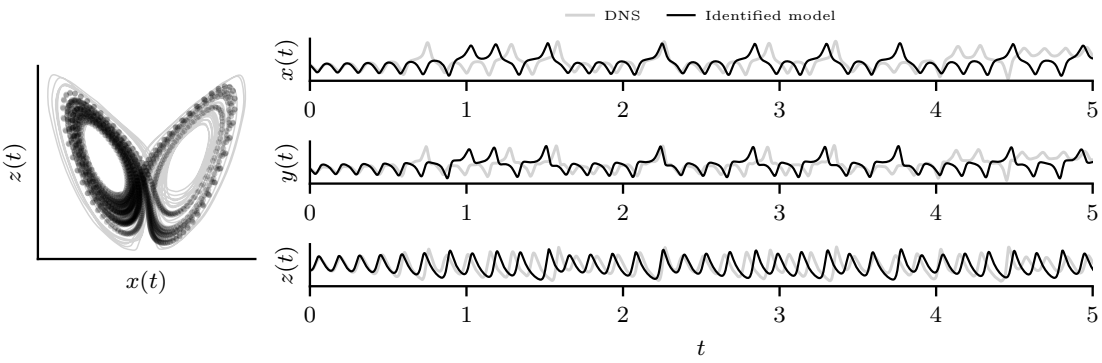
Exemple : le système de Lorenz

```
def dynsys(t, u, p):
    # Unpack parameters.
    sigma, rho, beta = p

    # Unpack variables.
    x, y, z = u

    # Lorenz system
    dx = sigma * ( y - x)
    dy = x * (rho - z) - y
    dz = x * y - beta *z

    return dx, dy, dz

```

```
# Parametres du systeme.
sigma, rho, beta = 10, 28, 8/3
p = [sigma, rho, beta]

# Parametres pour l'integration.
tspan = (0.0, 20.0)
u0 = np.array([1.0, 0.0, 0.0])

# Simulation.
sol = solve_ivp(
lambda t, u : dynsys(t, u, p),
tspan,
x0)


```

Equations différentielles ordinaires

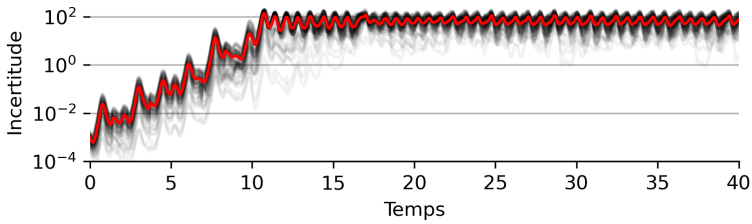
Exemple : le système de Lorenz

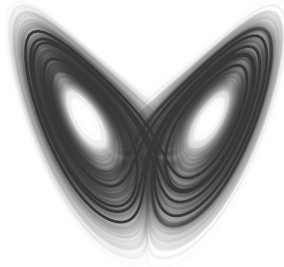


Equations différentielles ordinaires

Exemple : le système de Lorenz

Le système de Lorenz présente la propriété de **sensibilité aux conditions initiales** rendant toute prédiction précise inutile au delà d'une certaine échelle de temps.





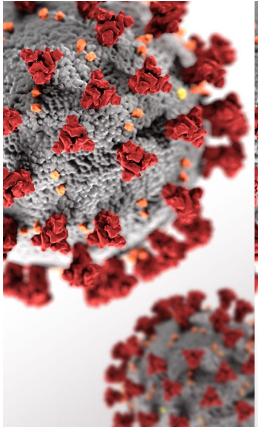
Equations différentielles ordinaires

Exemple : le modèle SIR

Le modèle SIR est un modèle utilisé en épidémiologie pour modéliser l'évolution d'une épidémie. Il est donné par

$$\frac{dS}{dt} = -\beta SI$$
$$\frac{dI}{dt} = \beta SI - \gamma I$$
$$\frac{dR}{dt} = \gamma I$$

où S, I et R représentent respectivement la fraction de la population saine, actuellement infectée et s'étant remise de l'infection. Les paramètres β et γ caractérisent certaines propriétés du virus.



Equations différentielles ordinaires

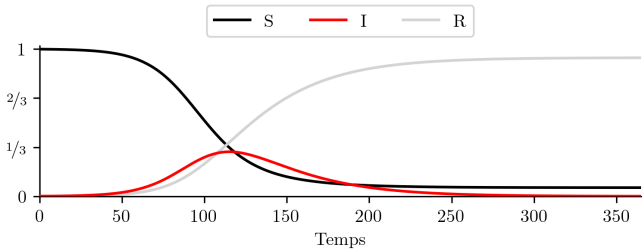
Exemple : le modèle SIR

```python
def sir(t, u, p):
    # Unpack parameters.
    beta, gamma = p

    # Unpack variables.
    s, i, r = u

    # Equations.
    ds = - beta * s * i
    di = beta * s * i - gamma * i
    dr = gamma *i

    return ds, di, dr

```

```python
# Parametres de l'epidemie.
beta, gamma = 1.0, 1.0
p = [beta, gamma]

# Parametres pour l'integration.
tspan = (0.0, 20.0)
u0 = np.array([0.99, 0.01, 0.0])

# Simulation.
sol = solve_ivp(
    lambda t, u : sir(t, u, p),
    tspan,
    u0)

```

Equations différentielles ordinaires

Exemple : le modèle SIR



R_0	:	3
Pop. tot. infectée	:	94%
Pic de l'épidémie	:	120^{e} jour