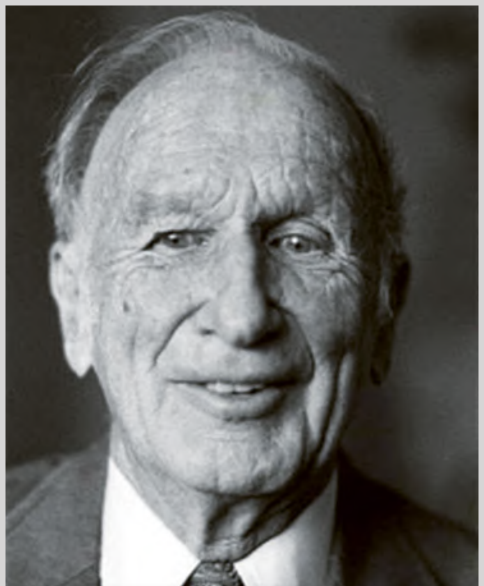
Étude de systèmes chaotiques :
La suite logistique et le pendule à oscillations forcées



Edward Lorenz

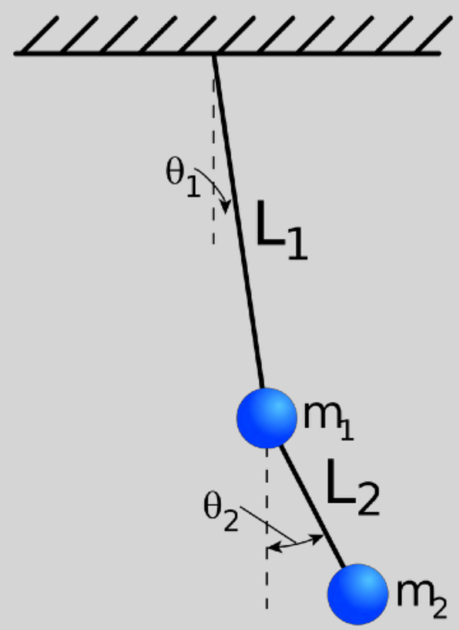
$$x' = \sigma(y - x)$$

$$y' = x(\rho - z) - y$$

$$z' = xy - \beta z$$

Système dynamique de Lorenz

Henri Poincaré



Double pendule

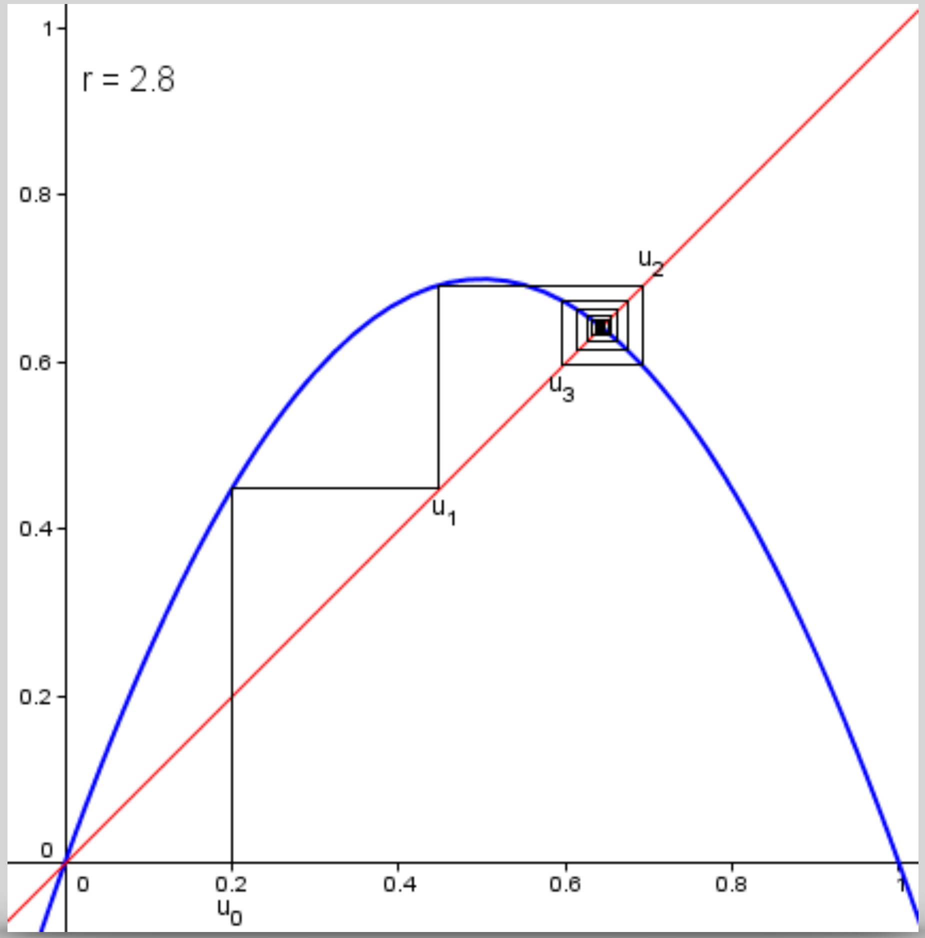
TIPE 2018 - 2019
Victor KHAMESI
Candidat 20405

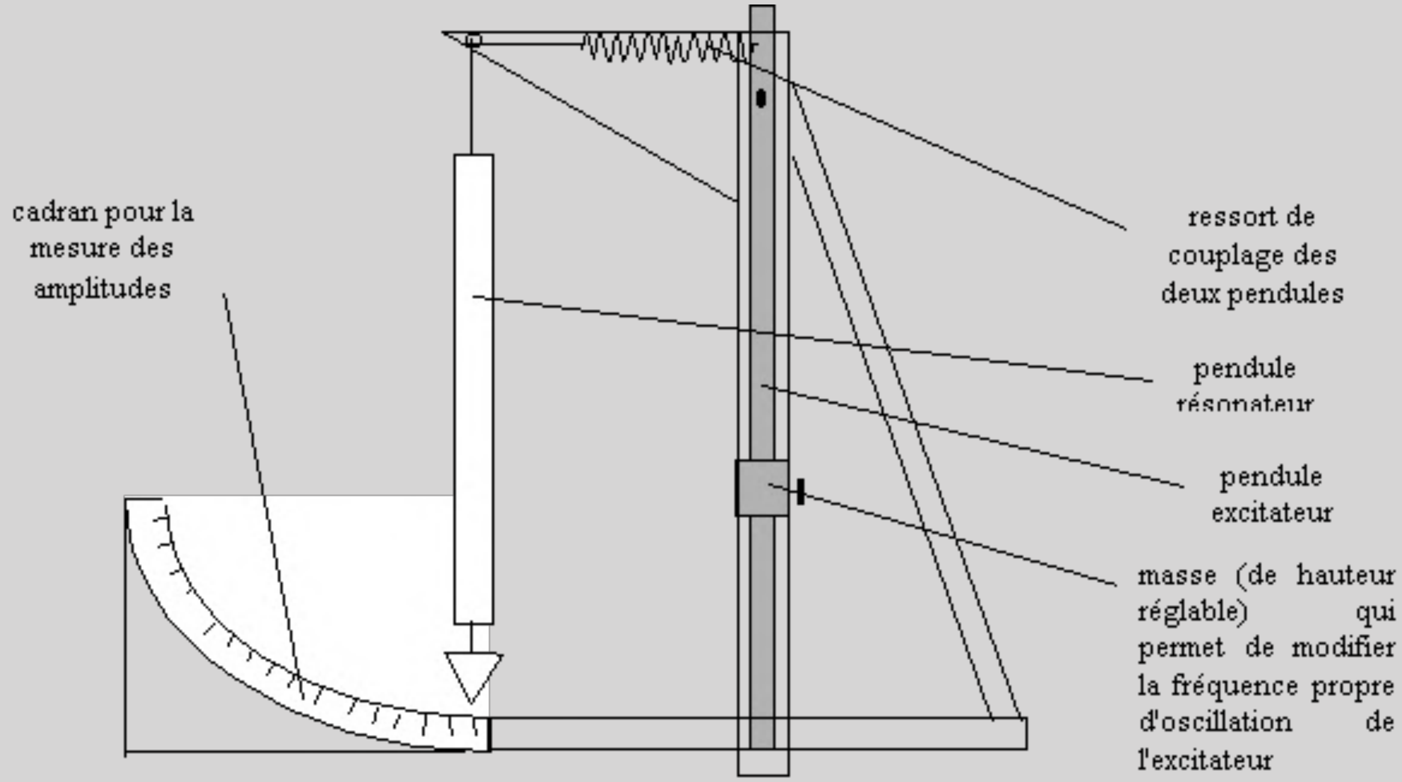
Sommaire

En quoi certains systèmes physiques ou mathématiques en apparence assez simples peuvent en réalité évoluer de façon quasi-imprévisible ?

Première partie : Le chaos discret étudié à travers la suite logistique

Deuxième partie : Le chaos continu dans le pendule à oscillations forcées





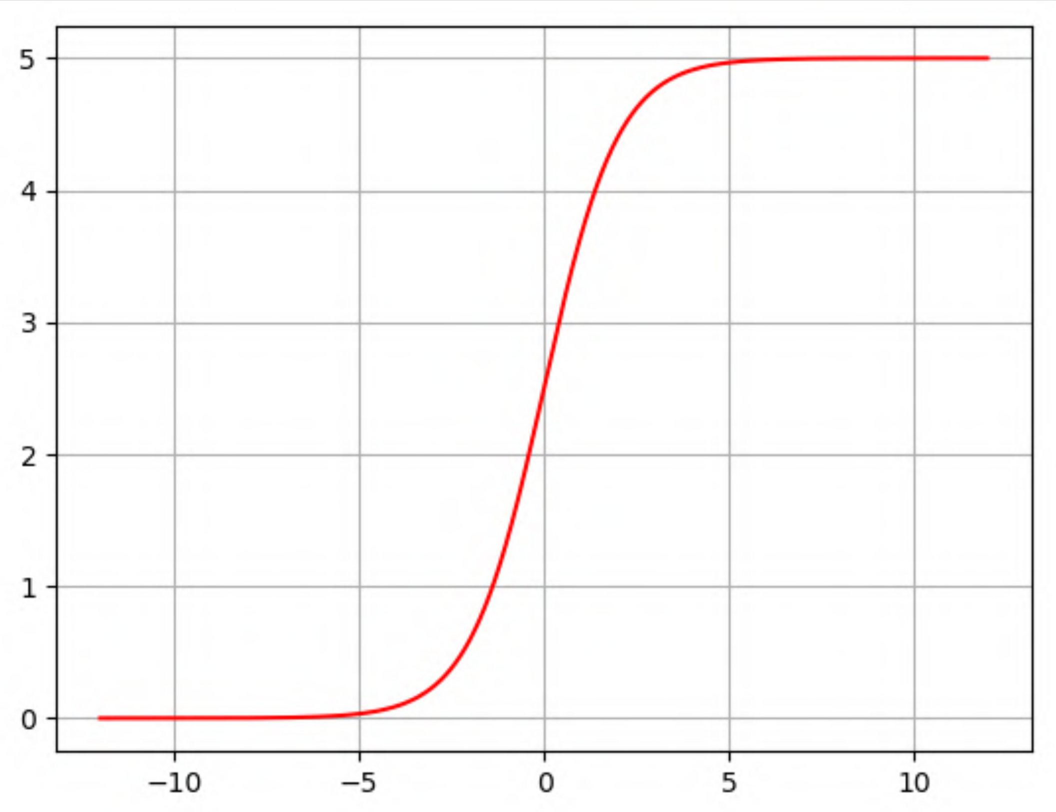
La suite logistique : les modèles continu et discret

Idée de ***Verhulst :*** $p'(t) = \mu p(t)(M - p(t))$

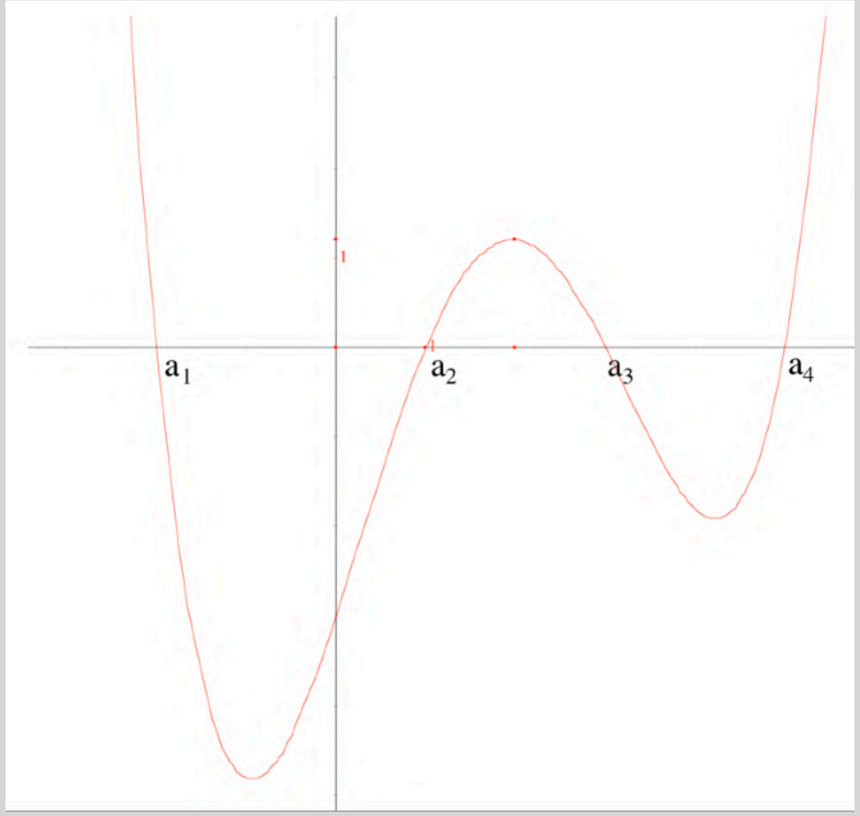
Solution ***analytique*** *:* $p(t) = \dfrac{M}{1 + \lambda e^{-\mu M t}}$

Suite logistique :

$$u_{n+1} = \mu u_n(1 - u_n)$$



Allure de la courbe logistique

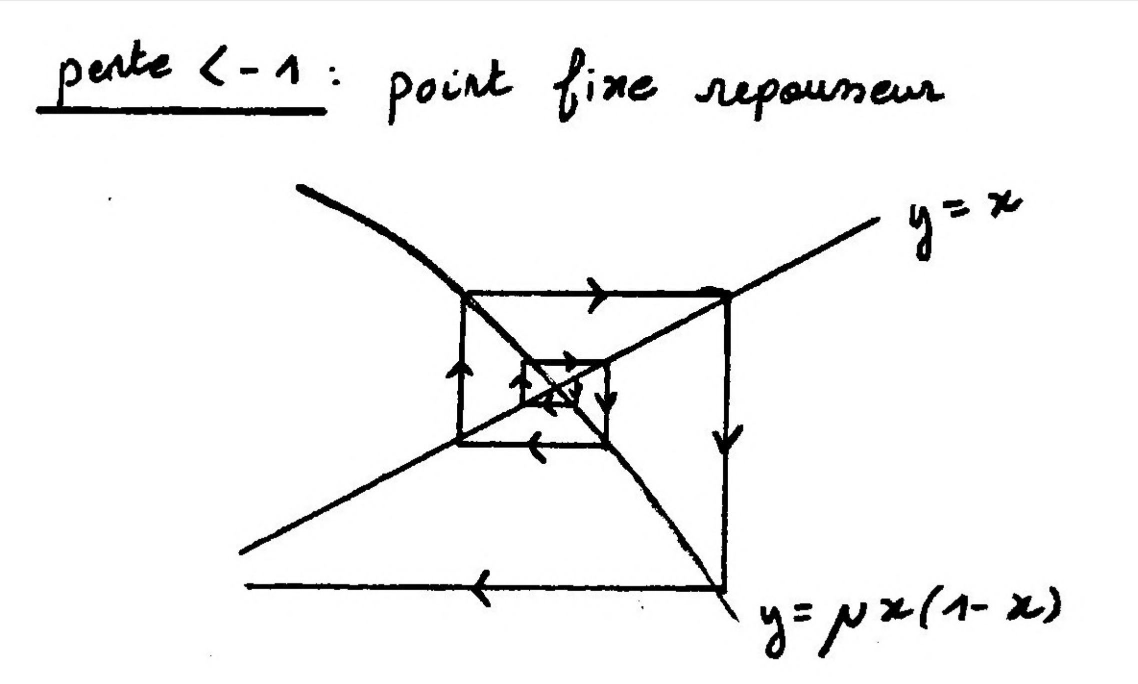


Points d'équilibres stables et instables

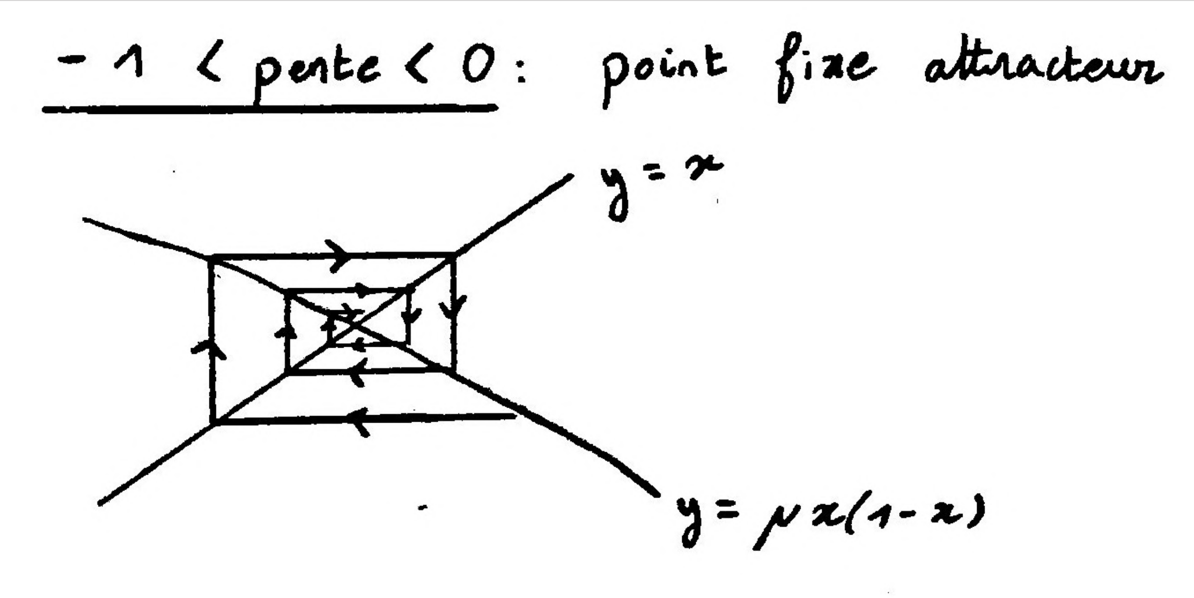
La suite logistique : première exploration

$$u_{n+1} = \mu u_n(1 - u_n) = f(u_n) \quad \text{où} \quad f(x) = \mu x(1 - x)$$

- ***Convergences*** : en escalier ou en escargot
- ***Points fixes*** : attractif, répulsif, super-attractif, parabolique



Exemple de comportement répulsif



Exemple de comportement attractif

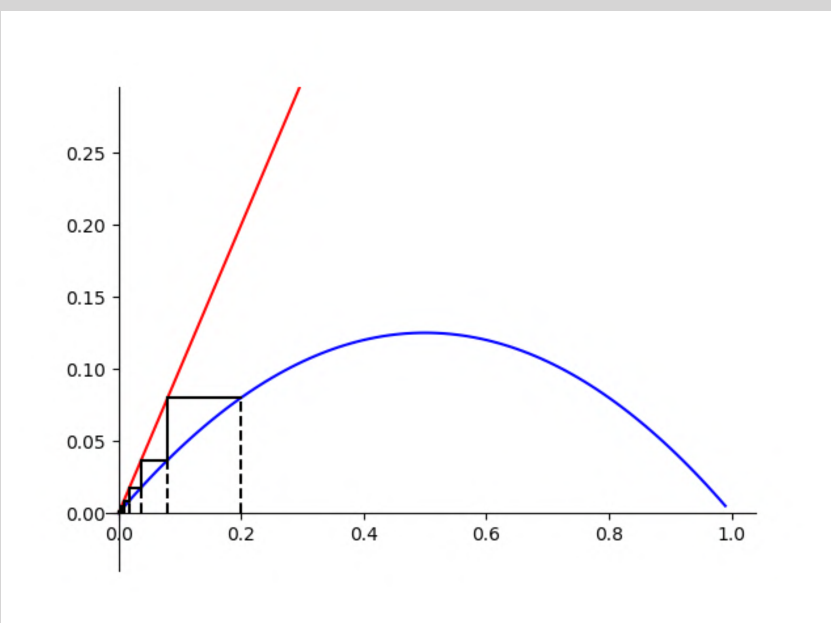
La suite logistique : première exploration

Théorème. *Si* $0 < \mu \leq 3$ *alors,*

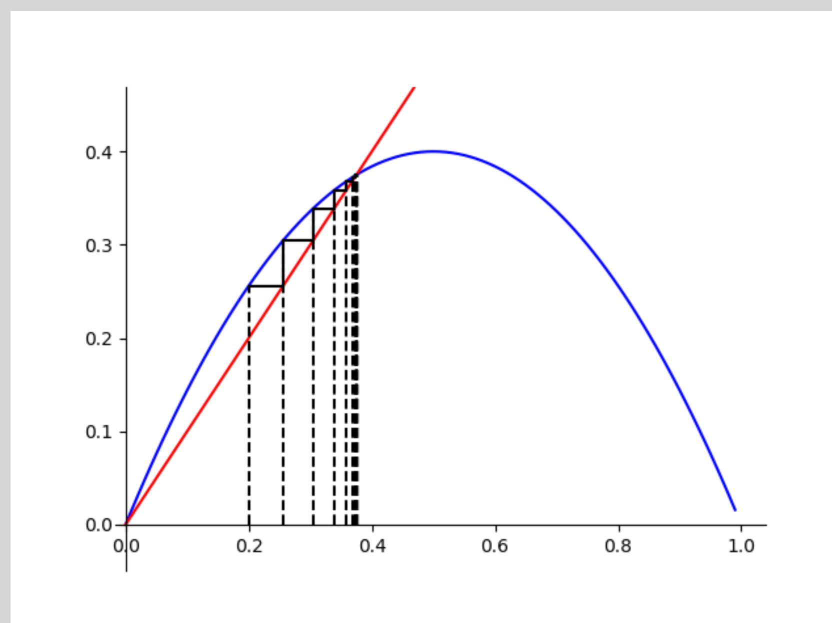
*pour toute **valeur initiale*** $u_0 \in [0,1]$

la suite (u_n) *est **convergente.***

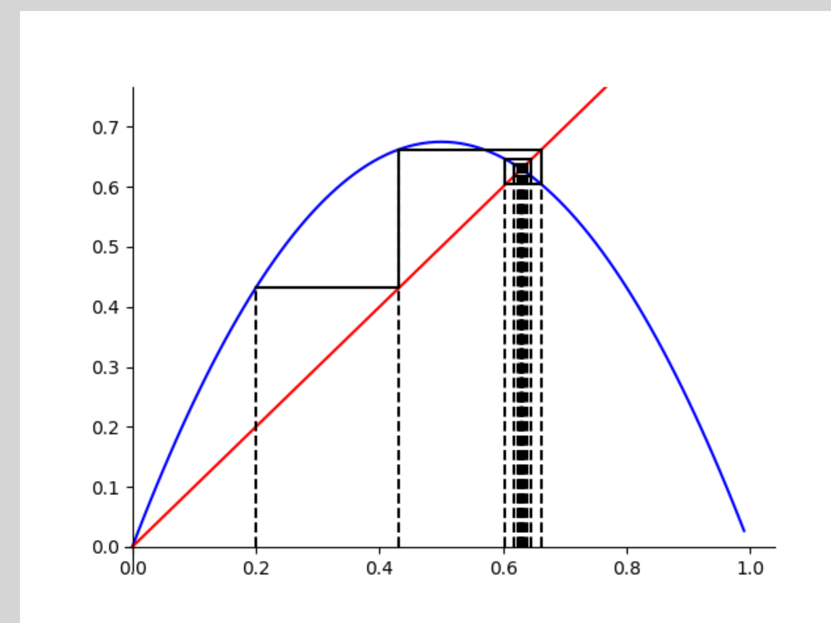
Le cas $0<\mu<1$



Le cas $1<\mu<2$



Le cas $2<\mu<3$



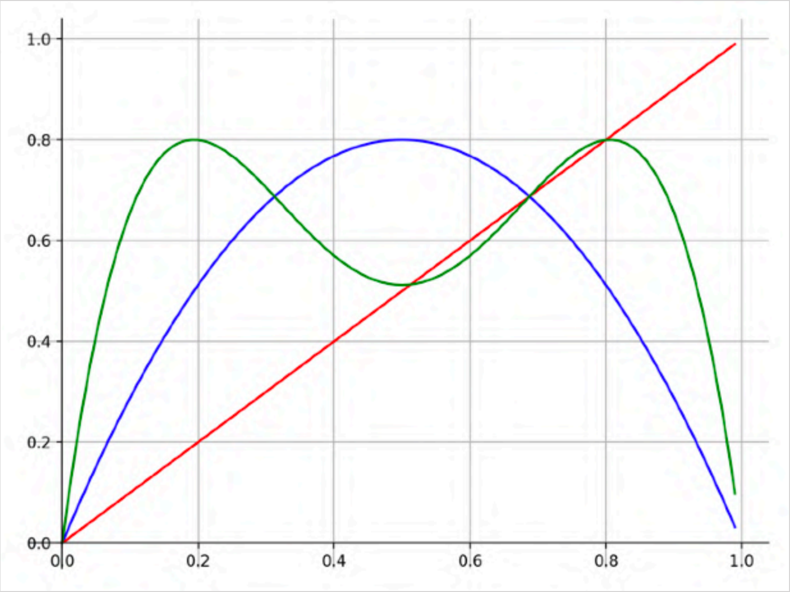
La suite logistique : doublement de la période

Théorème de Coppel. *Soit $I = [a,b]$ un intervalle compact et soit $f : I \to I$ une fonction continue. On suppose que f n'admet pas de 2-cycle. Alors, pour tout $u_0 \in I$, la suite récurrente $u_{n+1} = f(u_n)$ converge.*

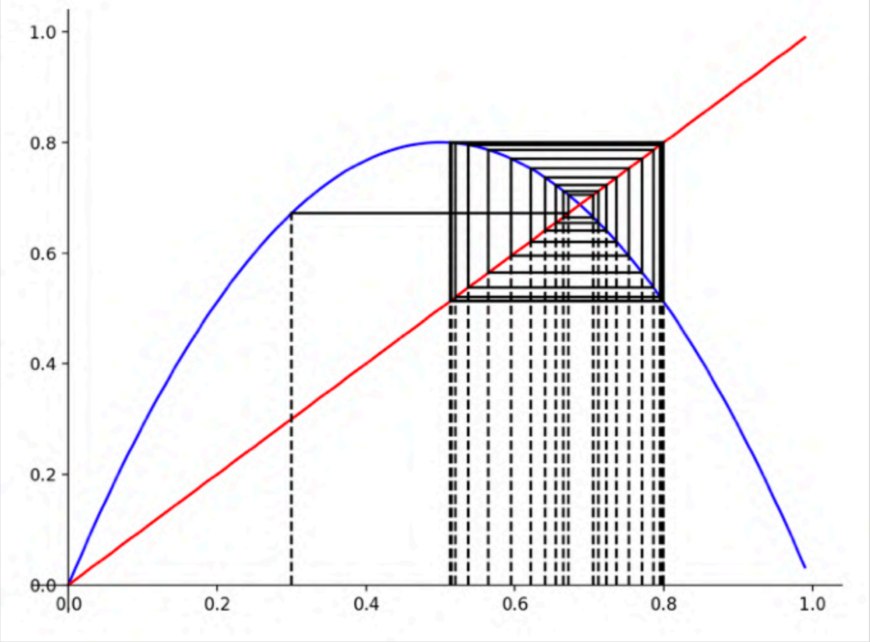
Étude du polynôme $$f^2(x) - x = -x(\mu x - \mu + 1)(\mu^2 x^2 - \mu(\mu+1)x + \mu + 1)$$

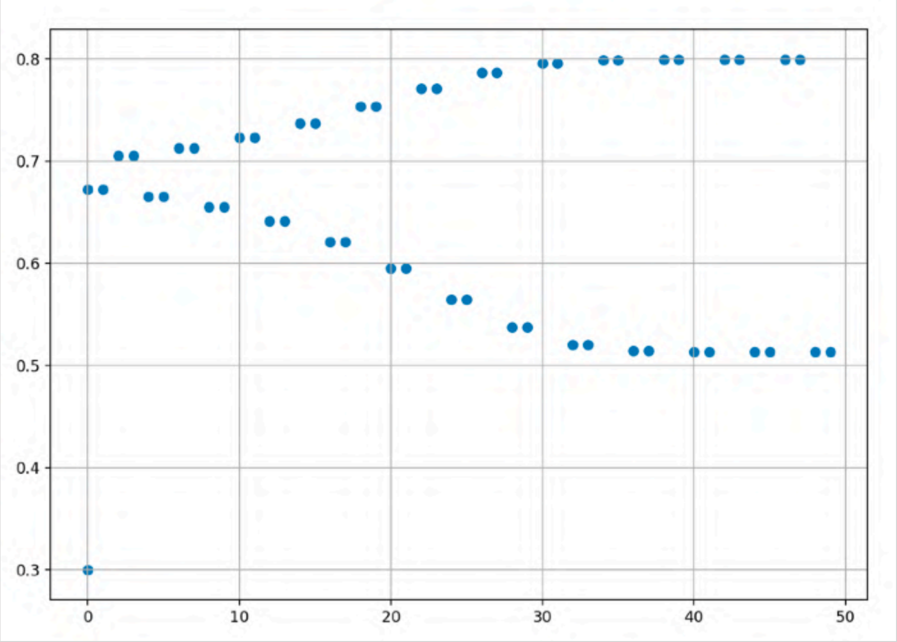
qui possède ***deux racines non triviales***.

Représentation graphique de ***f² et f***



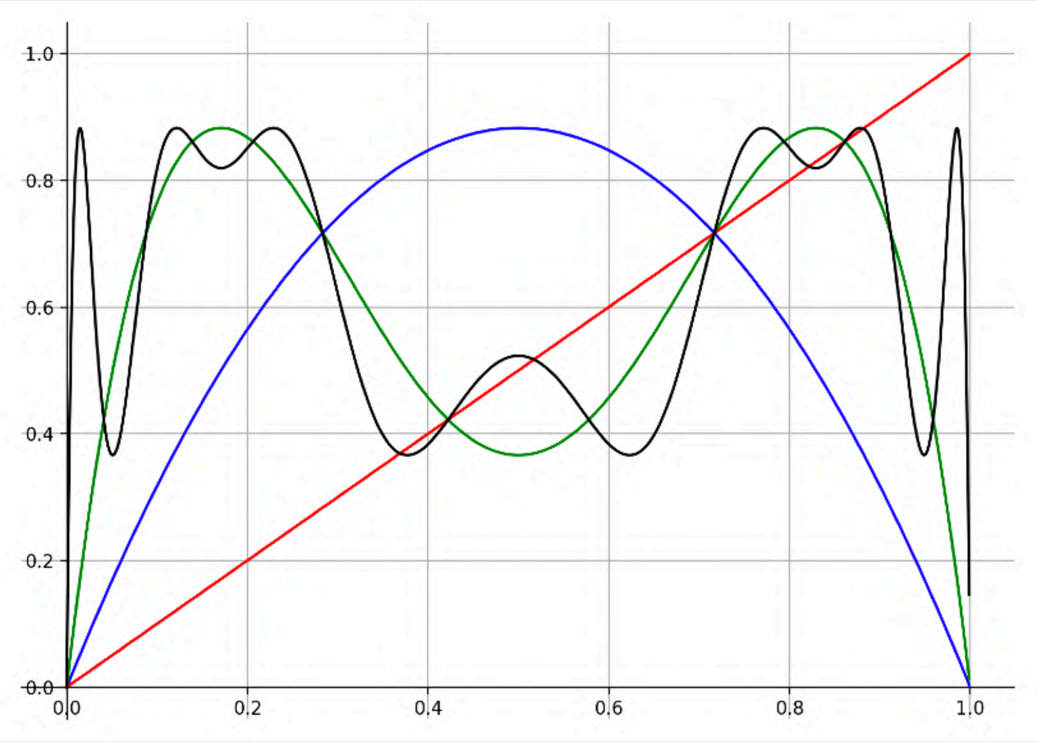
Exemple d'un ***2-cycle attractif (μ=3,2)***



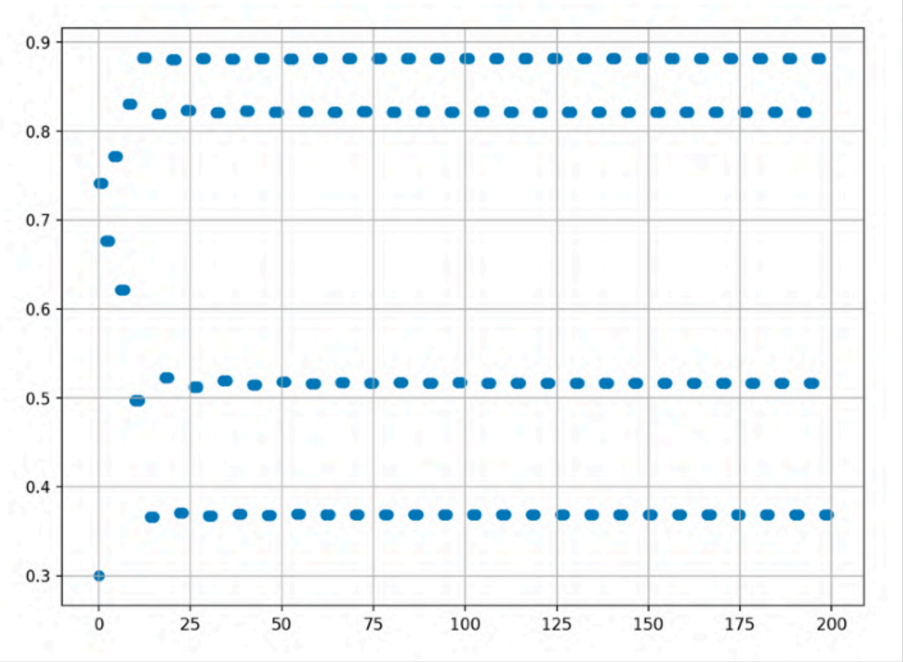


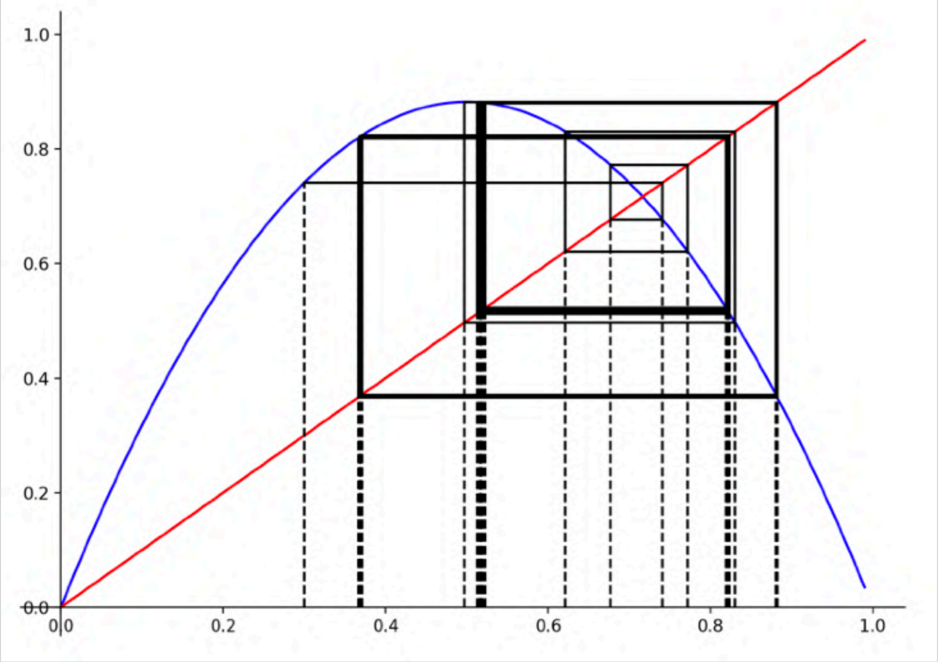
La suite logistique : doublement de la période

De la même manière, on étudie le polynôme $f^4(x) - x$



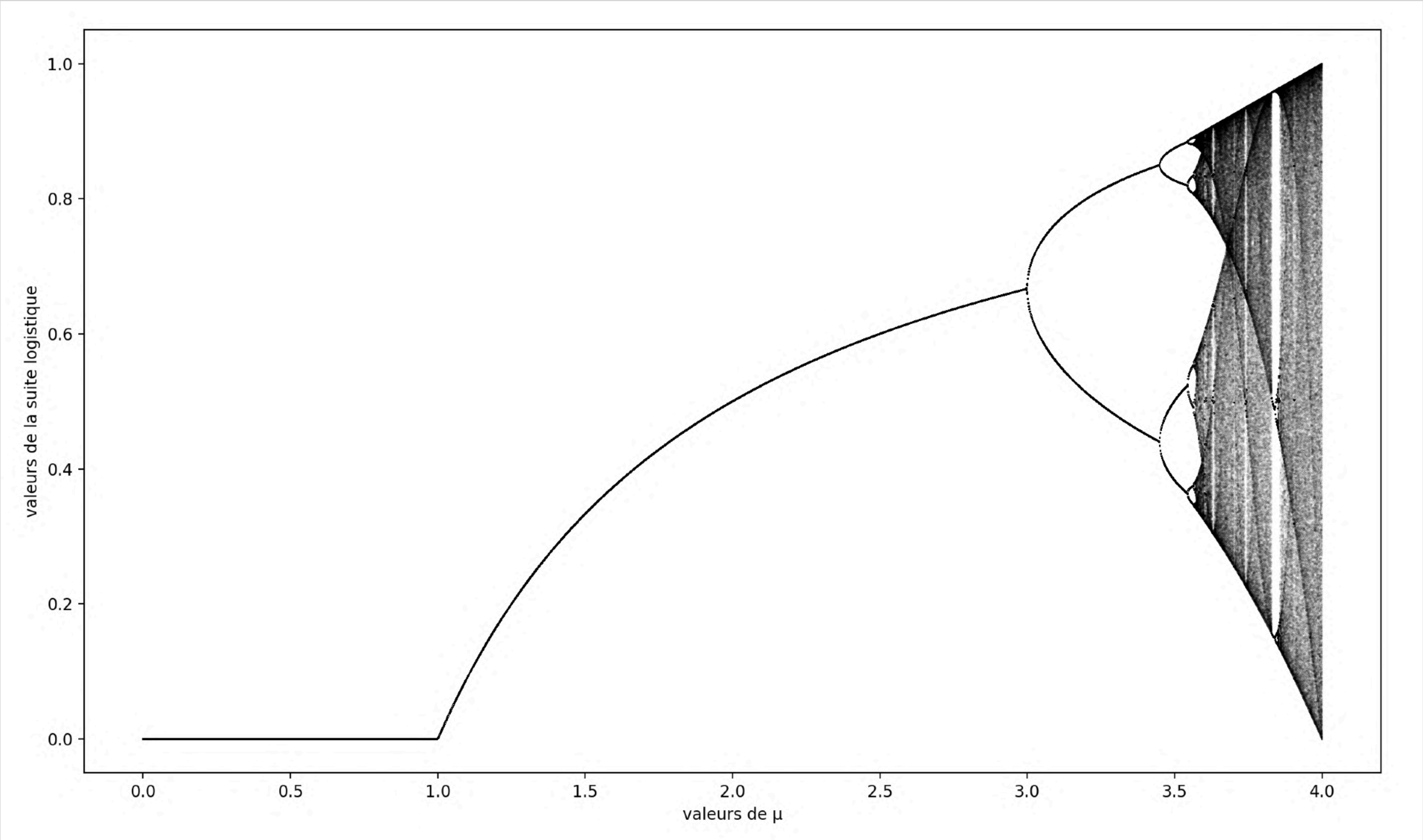
Apparition d'un 4-cycle (μ=3,53)





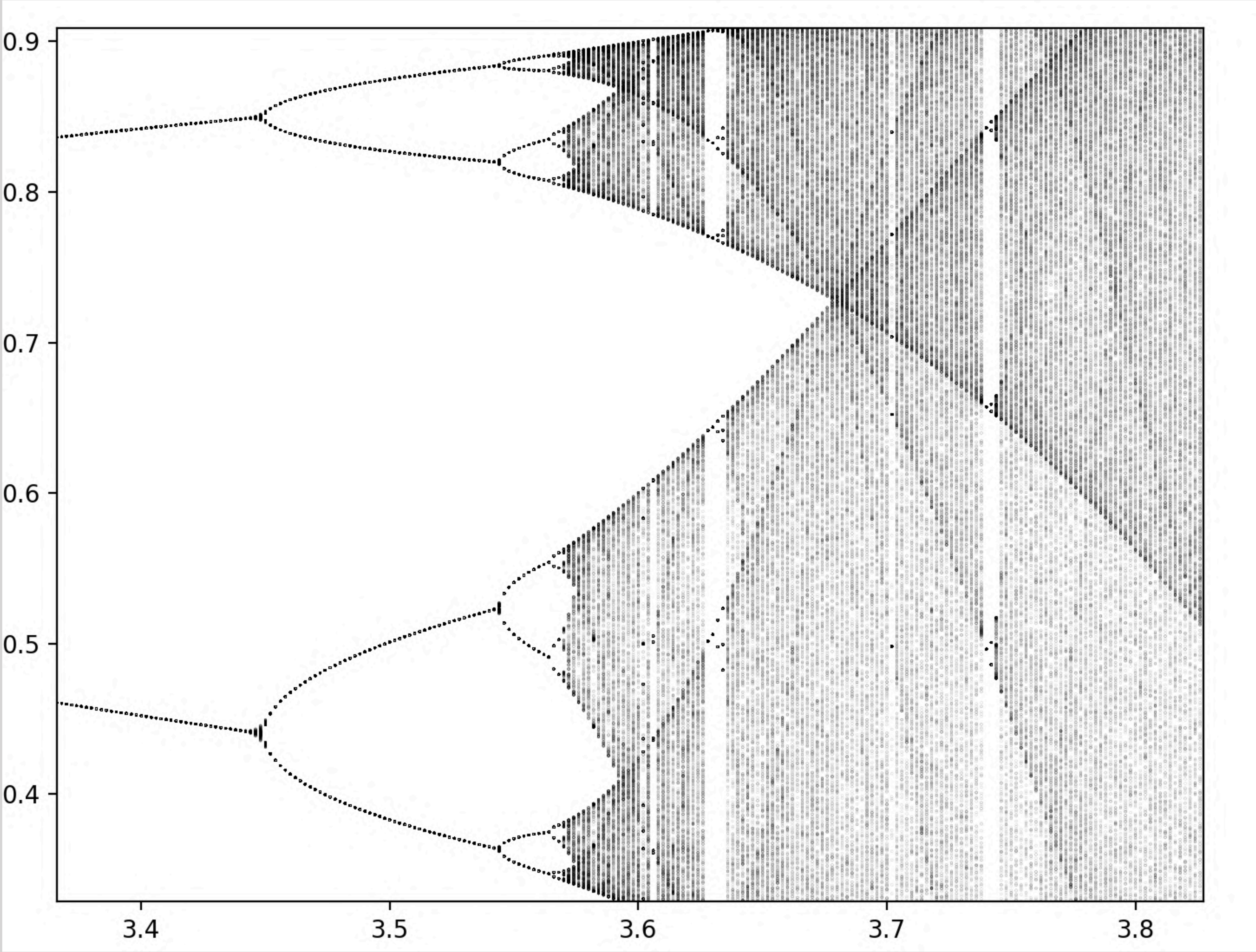
La suite logistique : doublement de la période

Et ainsi de suite, on réitère le même procédé. On obtient le ***diagramme de bifurcation*** *ci-dessous :*



La suite logistique : la zone intermédiaire $\mu_\infty \leq \mu < 4$

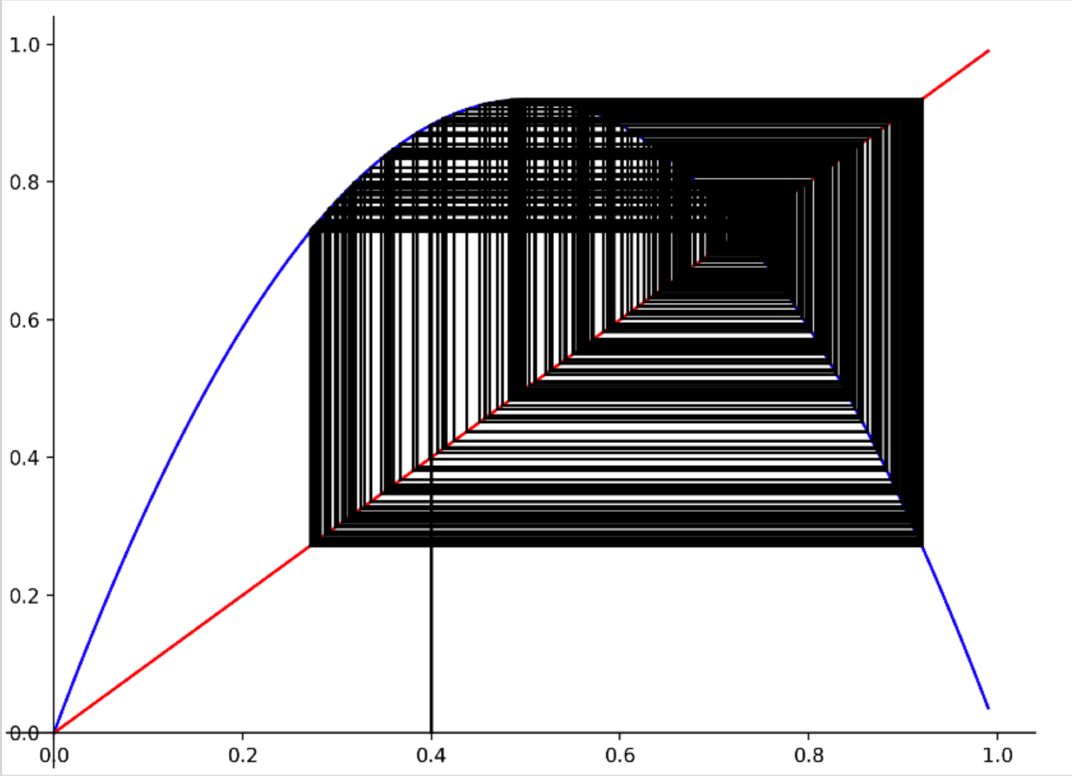
Le diagramme de bifurcation ***de plus près*** *:*



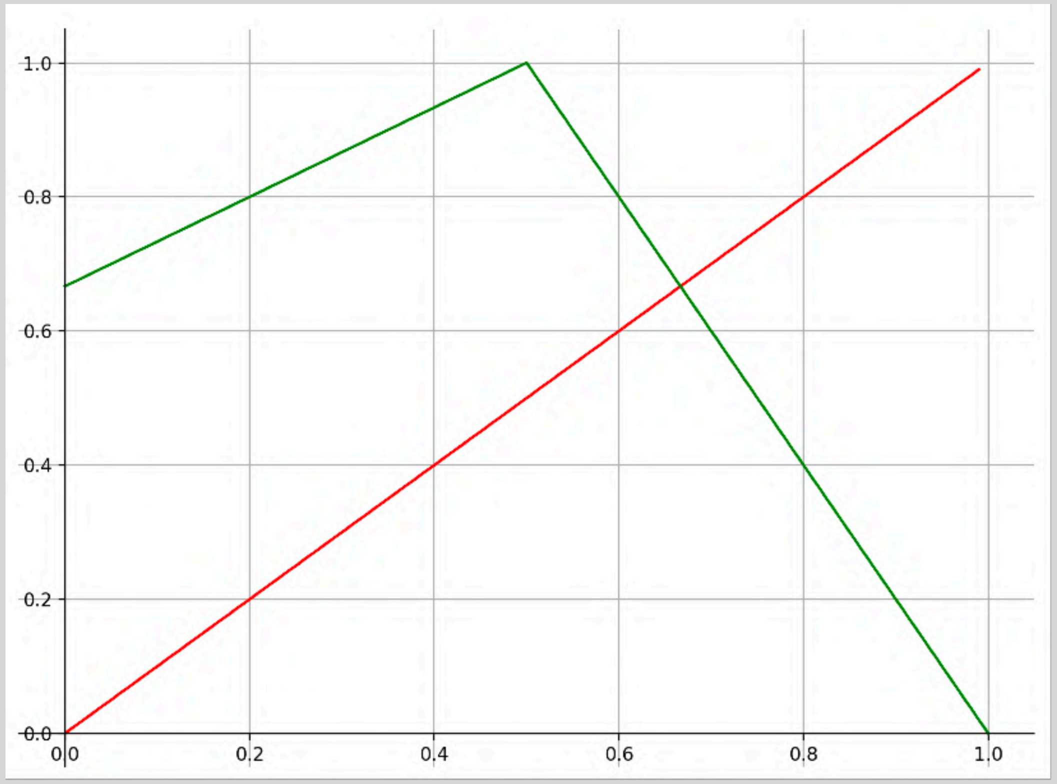
La suite logistique : la zone intermédiaire $\mu_\infty \leq \mu < 4$

La méthode de Ruelle *permet de trouver des valeurs de μ telles que la dynamique soit chaotique.*

par exemple μ = 3, 67857351043



*La « **TENTE** » de Ruelle*



La suite logistique : dynamique chaotique si μ = 4

Le **chaos** au sens de **Devaney**

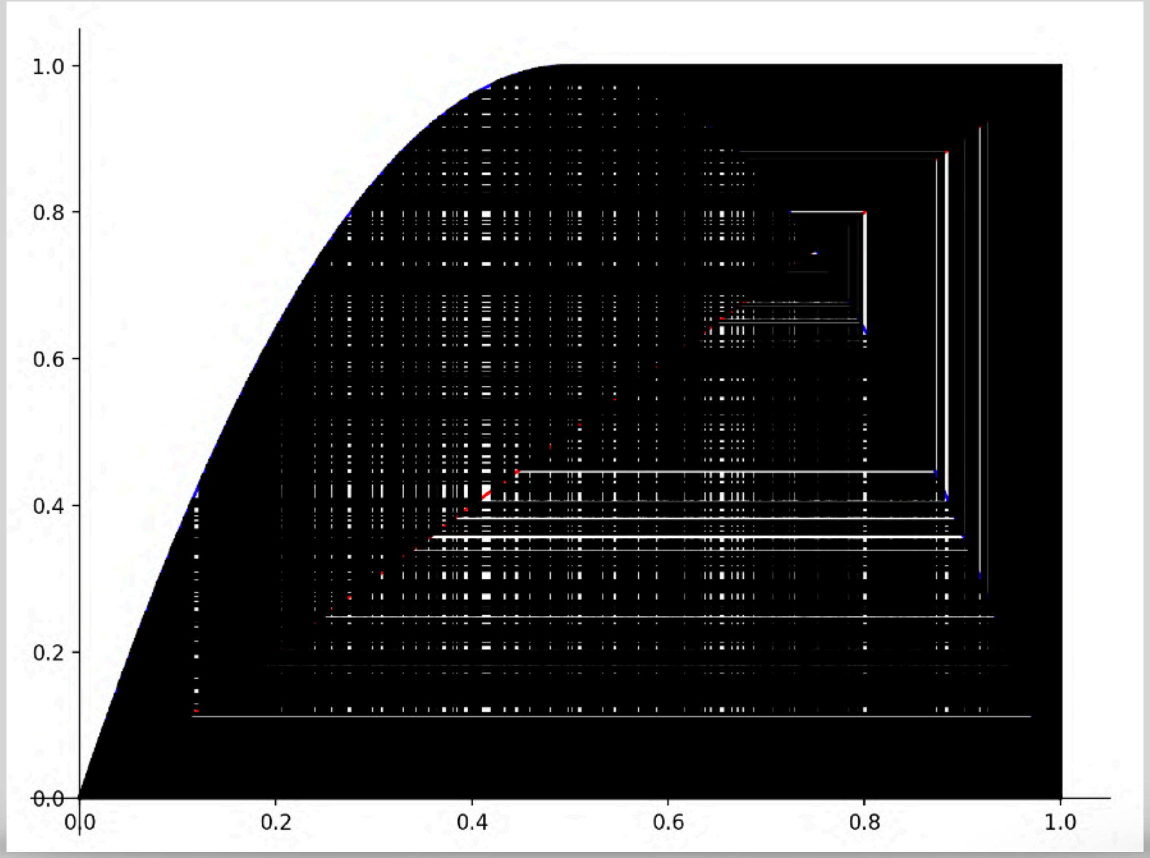
• **Transitivité** : *Soit* $f : I \to I$ *une fonction continue. On dit que* f *est topologiquement transitive ou simplement transitive si étant donnés deux ouverts non vides* U, V *de* I *il existe un entier* p *tel que* $f^p(U) \cap V \neq \emptyset$.

• **Sensibilité aux conditions initiales** : *On dit que* f *est sensible aux conditions initiales ou simplement sensitive, s'il existe* $\varepsilon > 0$ *tel que, pour tout* $x \in I$ *et pour tout* $\eta > 0$*, il existe* $y \in I$ *et* $n \in \mathbb{N}$ *avec* $|x - y| < \eta$ *et* $|f^n(x) - f^n(y)| > \varepsilon$.

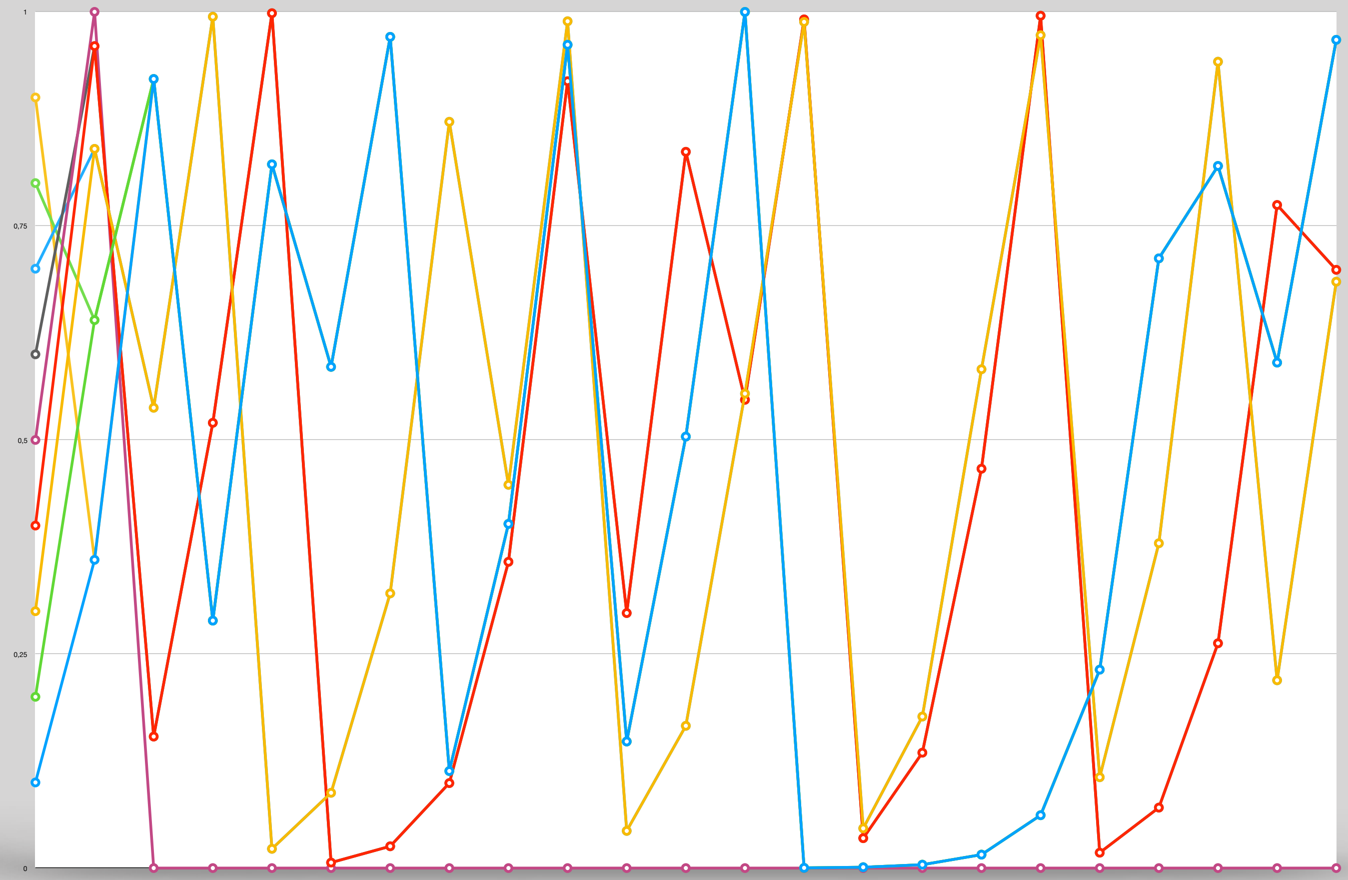
• **Densité des points périodiques** : *Les points périodiques de* f *sont denses dans* I *si pour tout* $(x, y) \in I$ *, il existe* $z \in]x, y[$ *tel que* z *soit un point périodique de* f.

La suite logistique : dynamique chaotique si $\mu = 4$

Construction *des termes de la suite logistique*

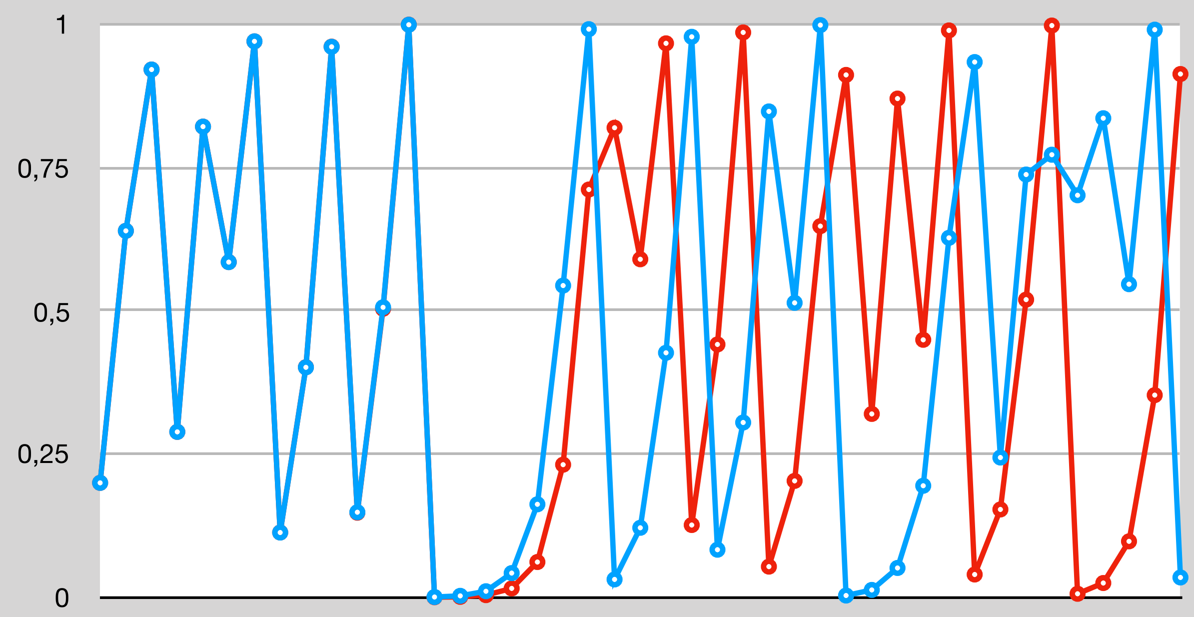


Dynamique chaotique pour différentes conditions initiales

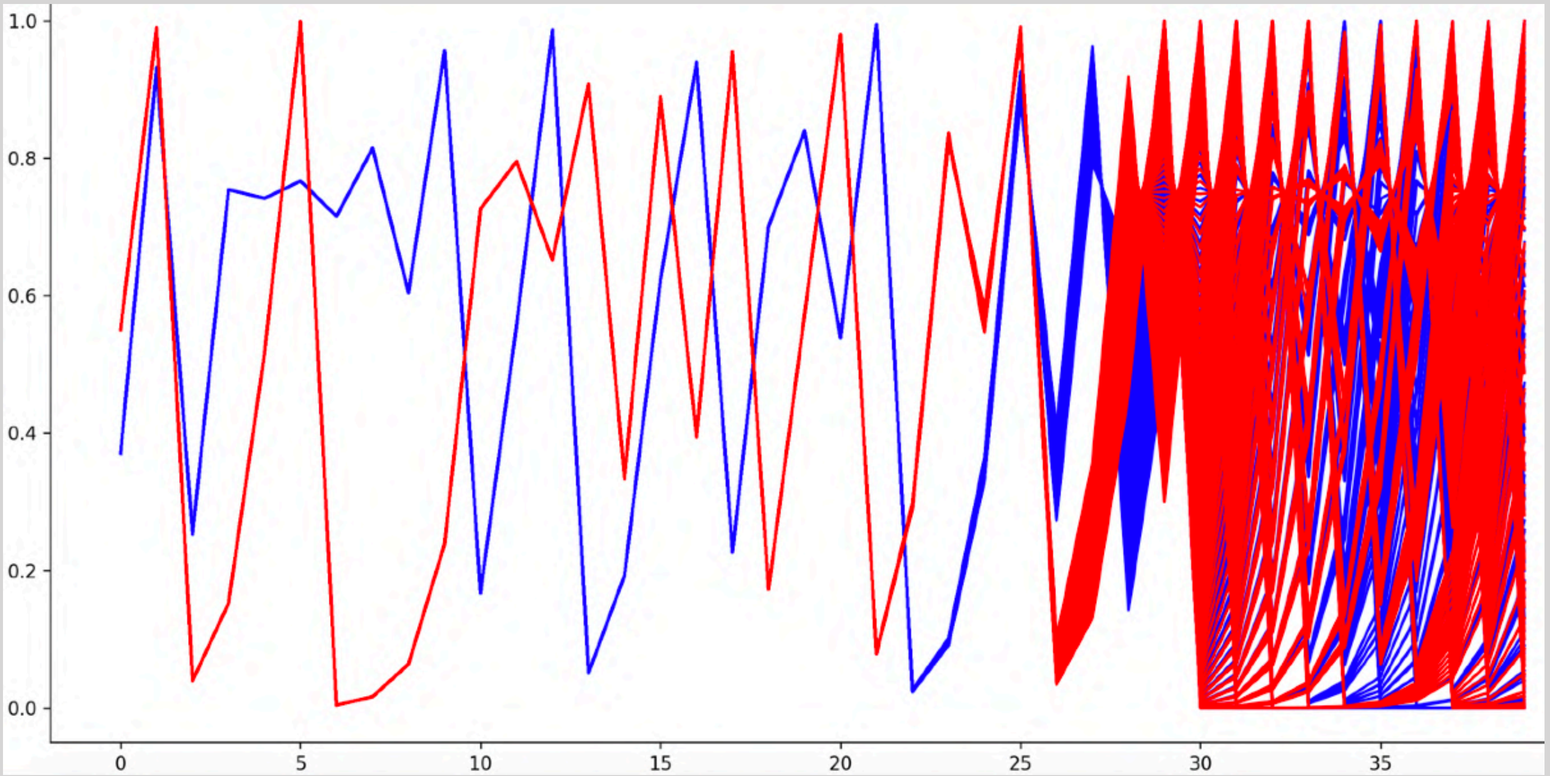


La suite logistique : dynamique chaotique si μ = 4

*Représentation de la **sensibilité** aux conditions initiales :*



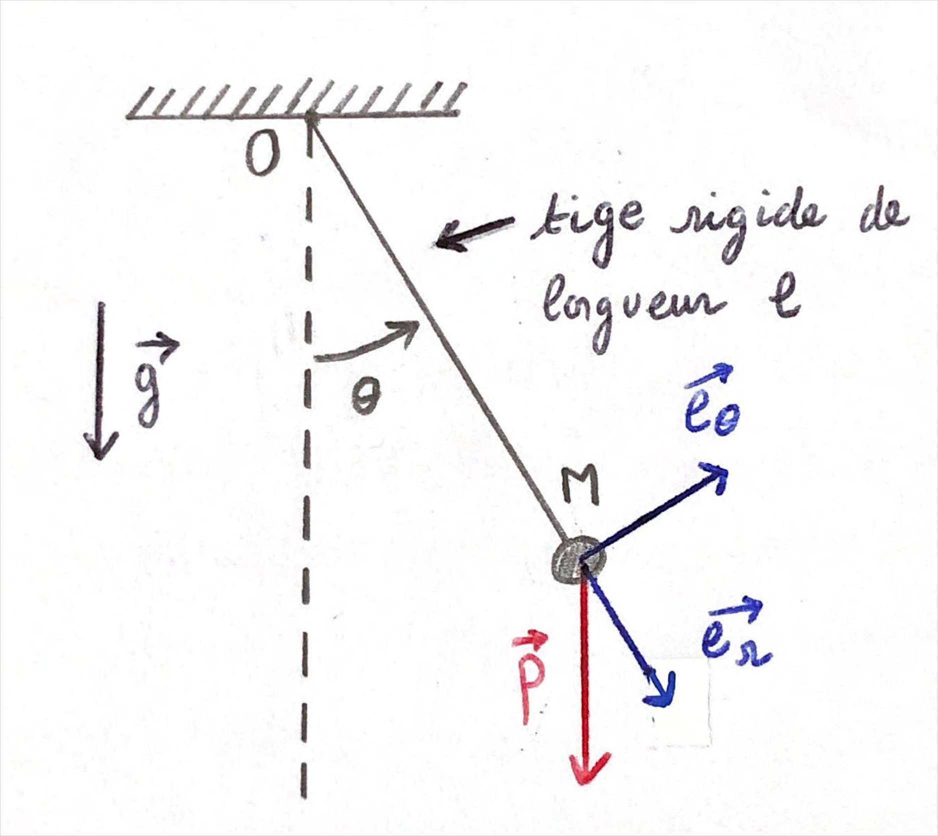
*Représentation du « **mélange** » des tracés (densité) :*



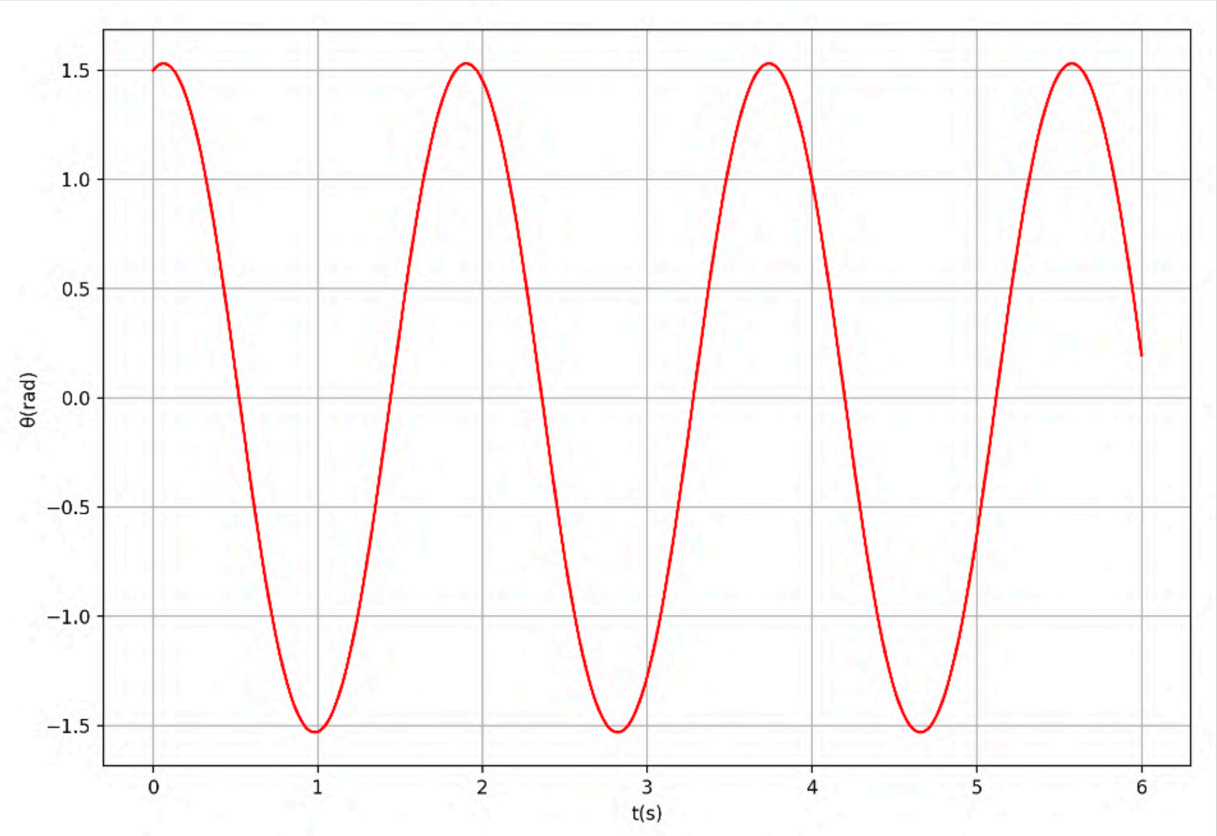
Le chaos continu : première approche

Équation du mouvement : $$\ddot{\theta} + \frac{g}{l} sin(\theta) = 0$$

Schéma du pendule libre

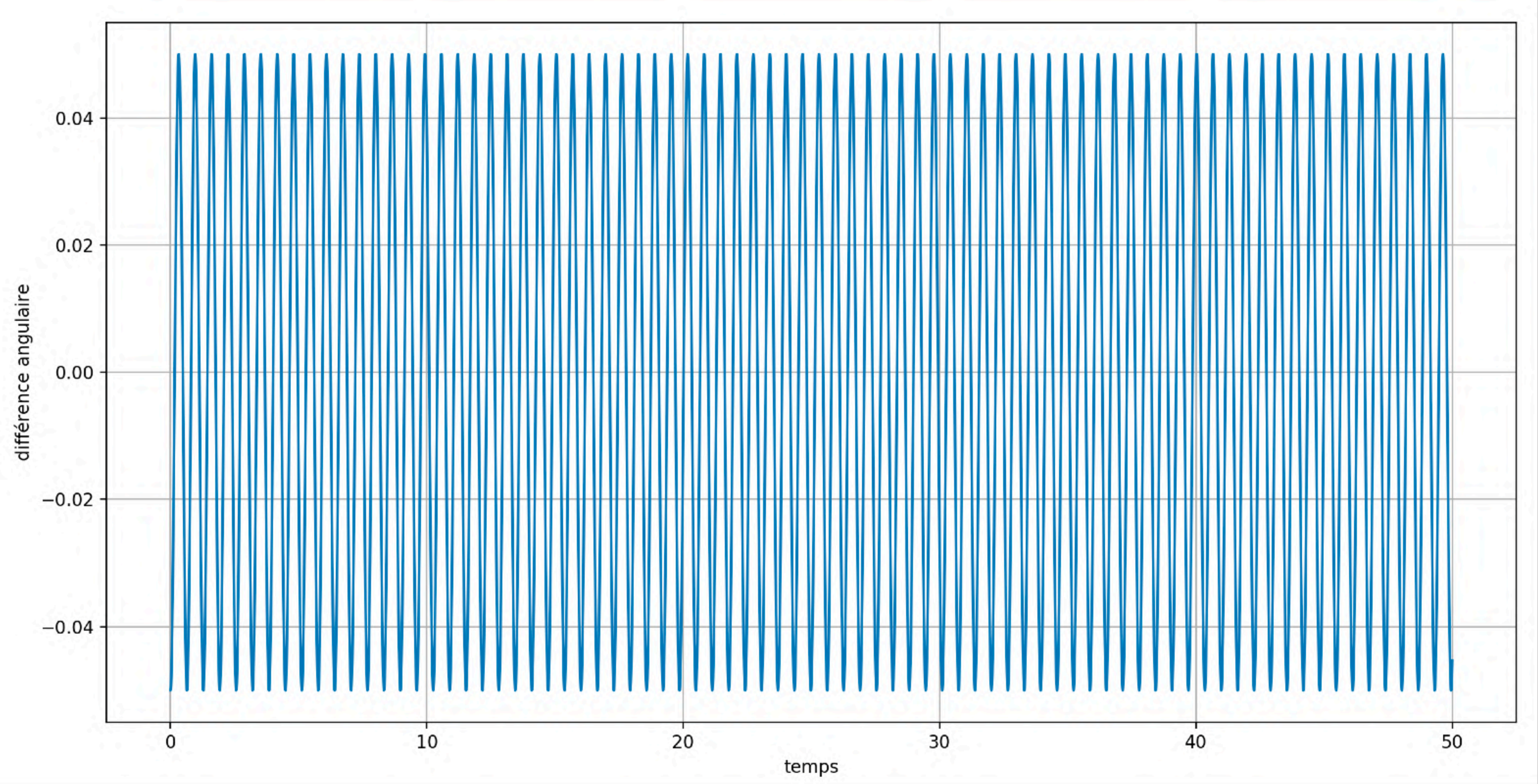


Évolution de θ(t)



Le chaos continu : première approche

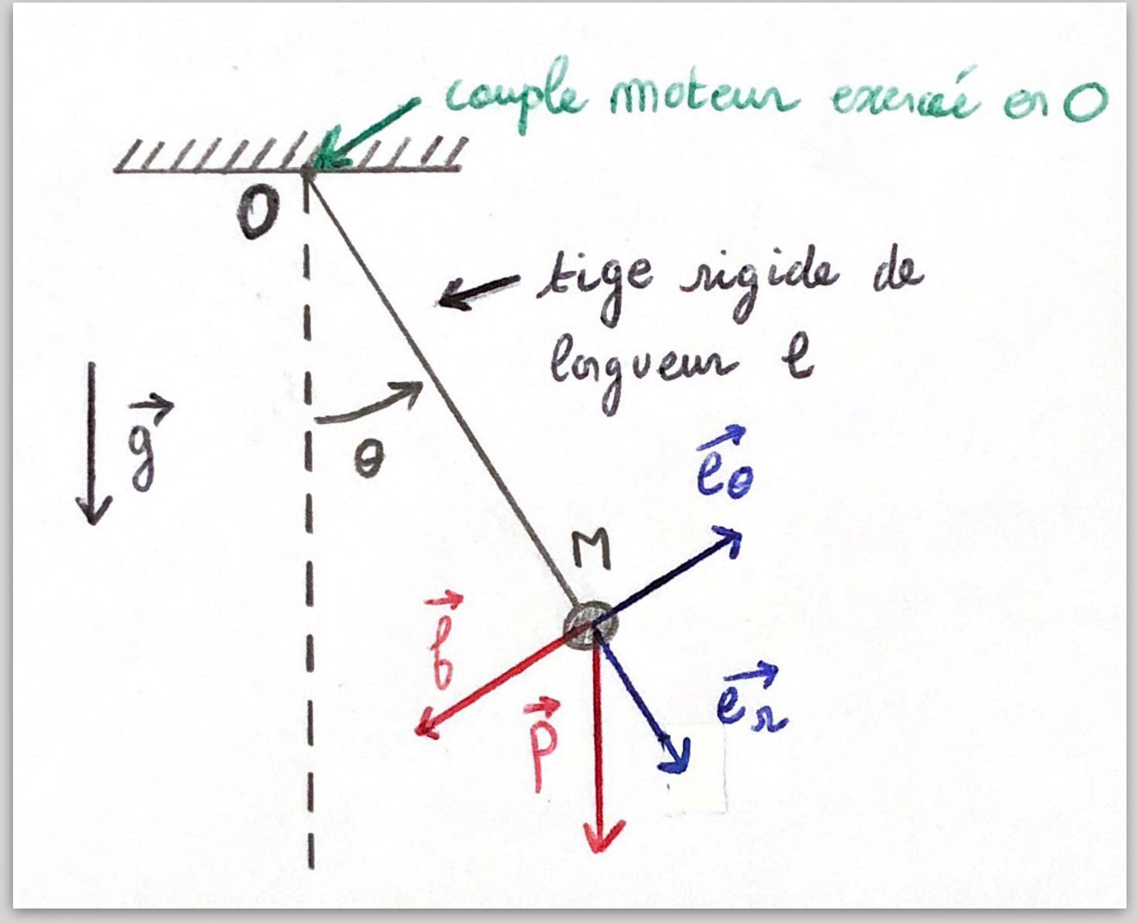
Évolution des trajectoires pour ***deux conditions initiales « proches »***



$\Delta\theta(t) \ll (\theta_1(t), \theta_2(t))$ *donc le système* ***n'est pas chaotique***

Le pendule à oscillations forcées : mise en équation

Schéma du pendule forcé



<u>Système :</u> {masse M + tige rigide}

<u>Référentiel :</u> terrestre supposé galiléen

<u>Bilan des forces :</u>
- poids
- frottements fluides exercés sur la tige
- couple moteur exercé en O

La liaison pivot est supposée idéale.

Application du théorème du moment cinétique en O.

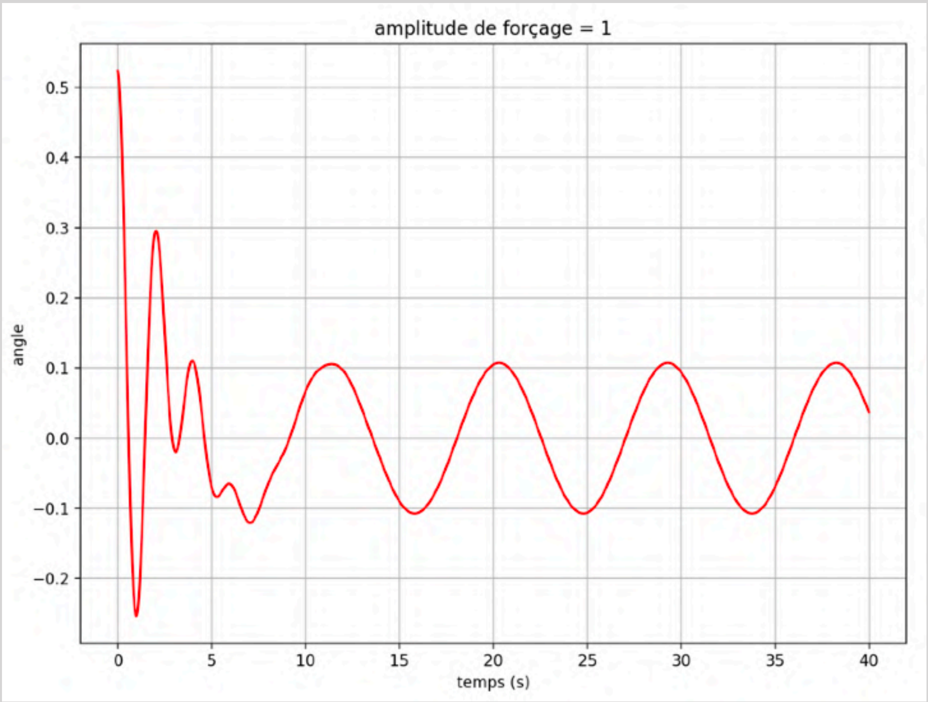
<u>Équation du mouvement :</u>

$$\ddot{\theta} + \frac{k}{m}\dot{\theta} + \frac{g}{l}sin(\theta) = \frac{\alpha}{ml^2}sin(\omega t)$$

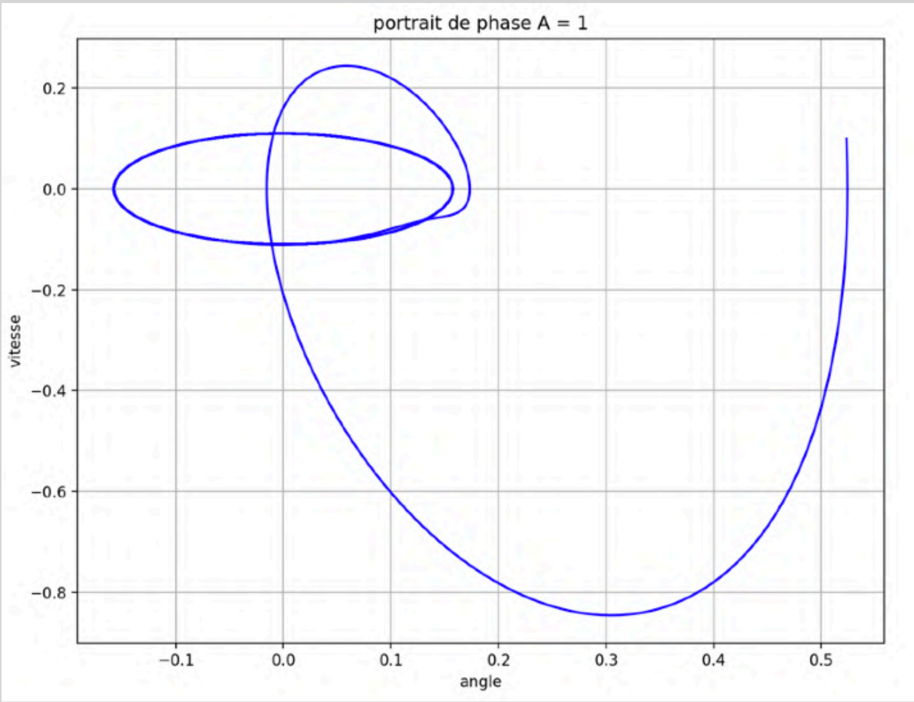
Le pendule à oscillations forcées : résolution du modèle

Dans le cas d’un écoulement visqueux, l’équation se réécrit : $$\ddot{\theta} + \frac{\omega_0}{Q}\dot{\theta} + \omega_0^2 sin(\theta) = Asin(\omega t)$$

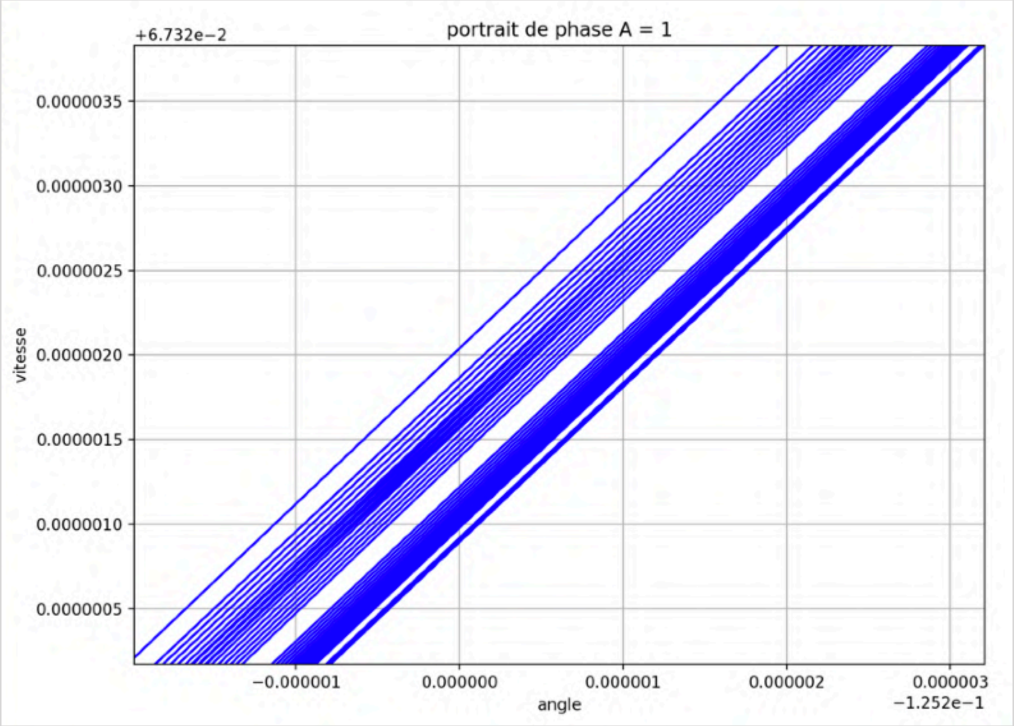
Courbe temporelle de θ



Pulsation de forçage ω = 0,7 rad/s



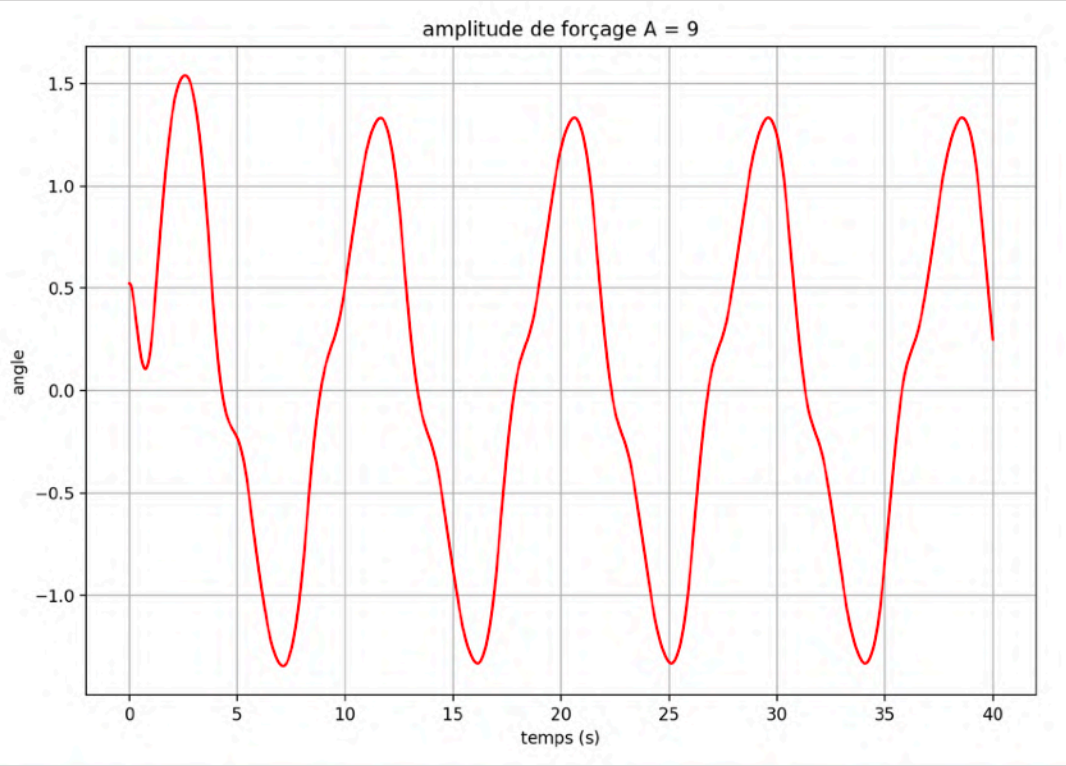
Dédoublement des périodes



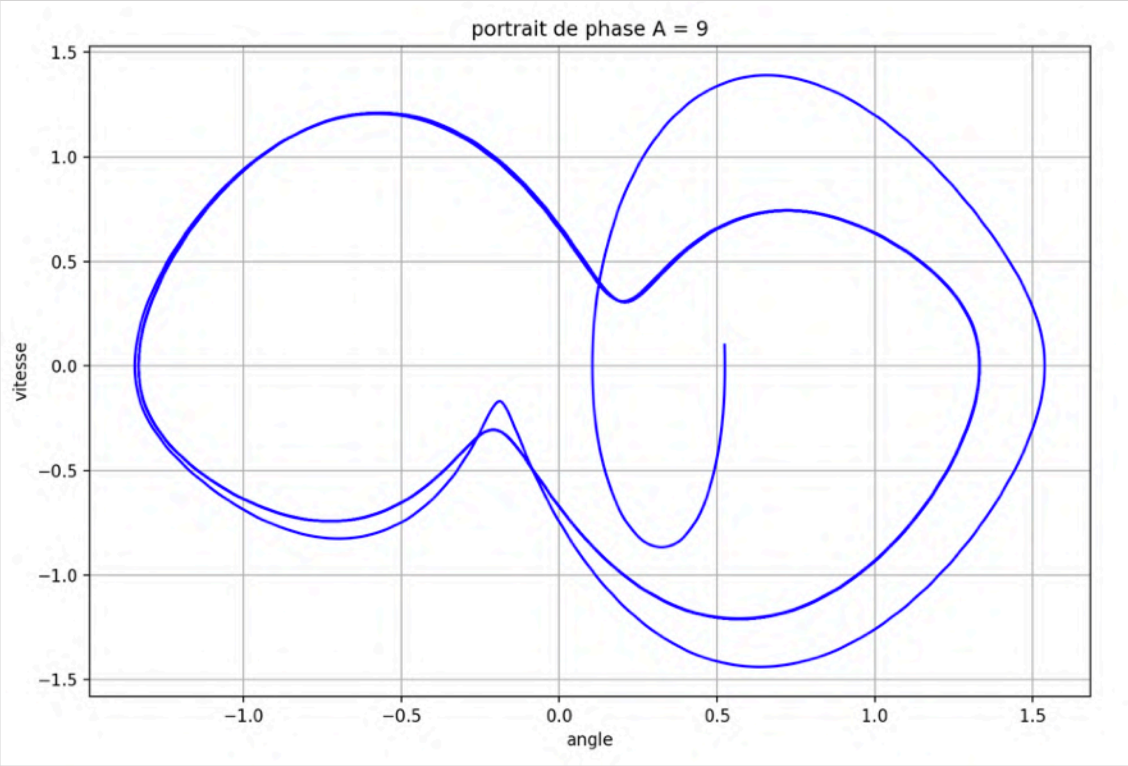
Le pendule à oscillations forcées : résolution du modèle

Augmentons l'amplitude de forçage :

Phénomène toujours périodique



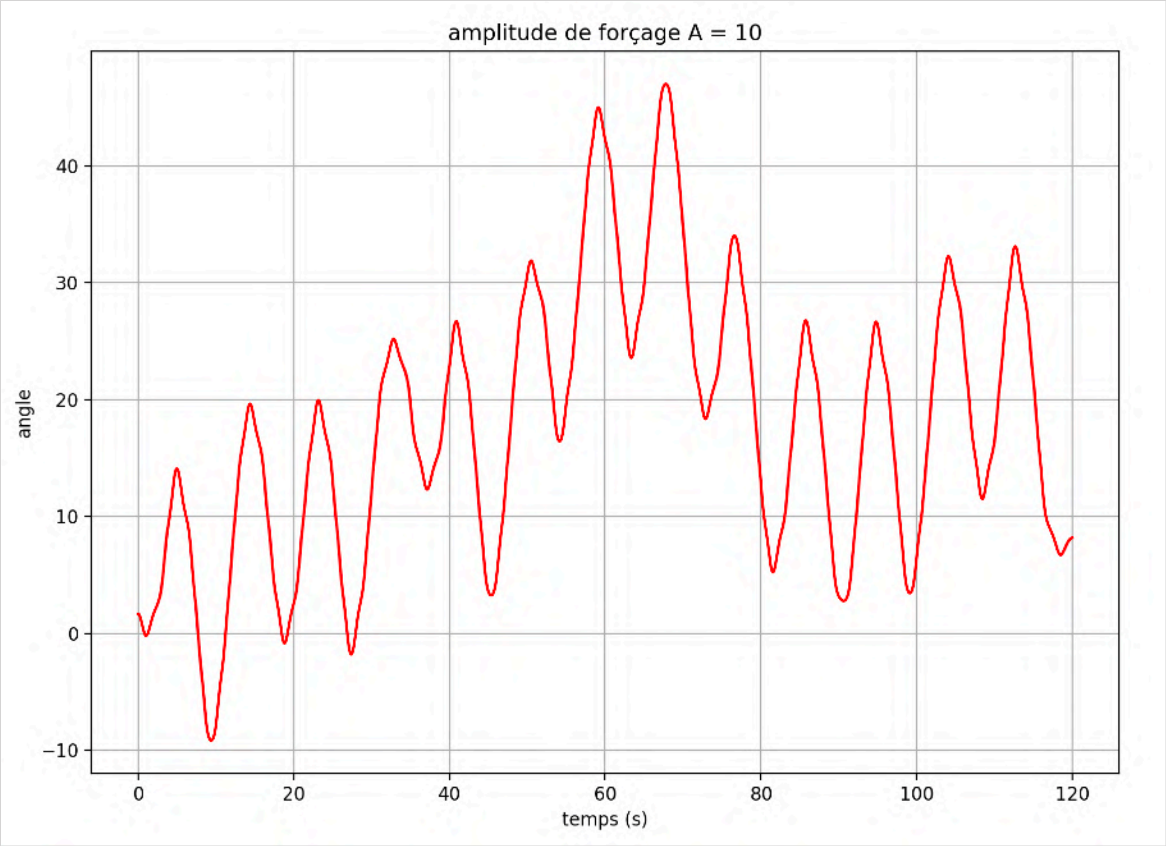
Dédoublement en deux ellipses



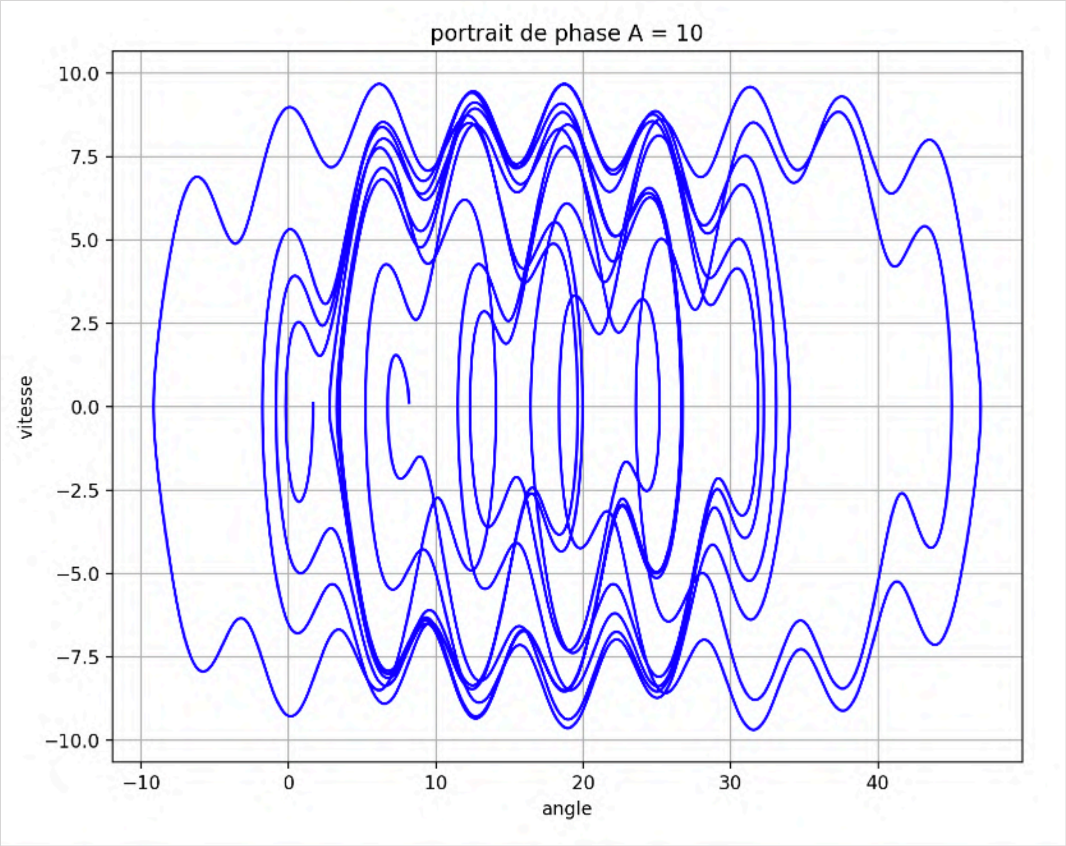
Le pendule à oscillations forcées : un système chaotique

Si on augmente à nouveau l'amplitude de forçage, on obtient des résultats surprenants.

Trajectoire chaotique



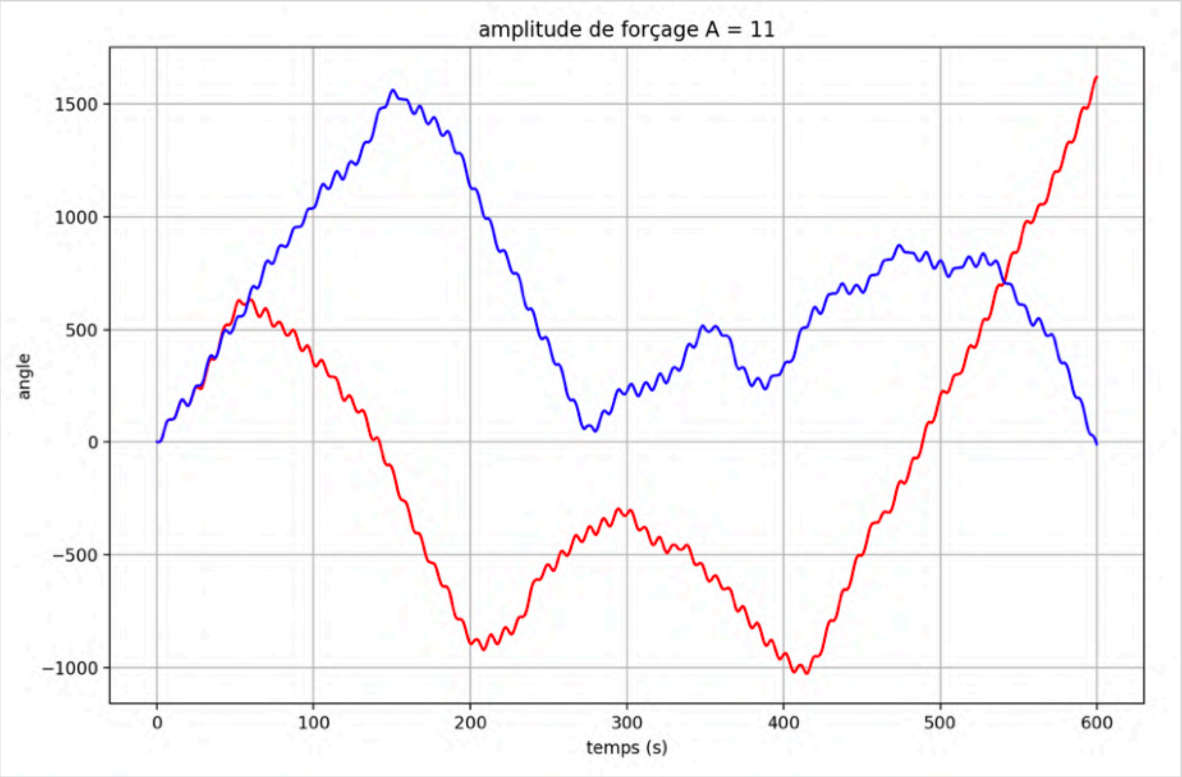
Portrait de phase chaotique



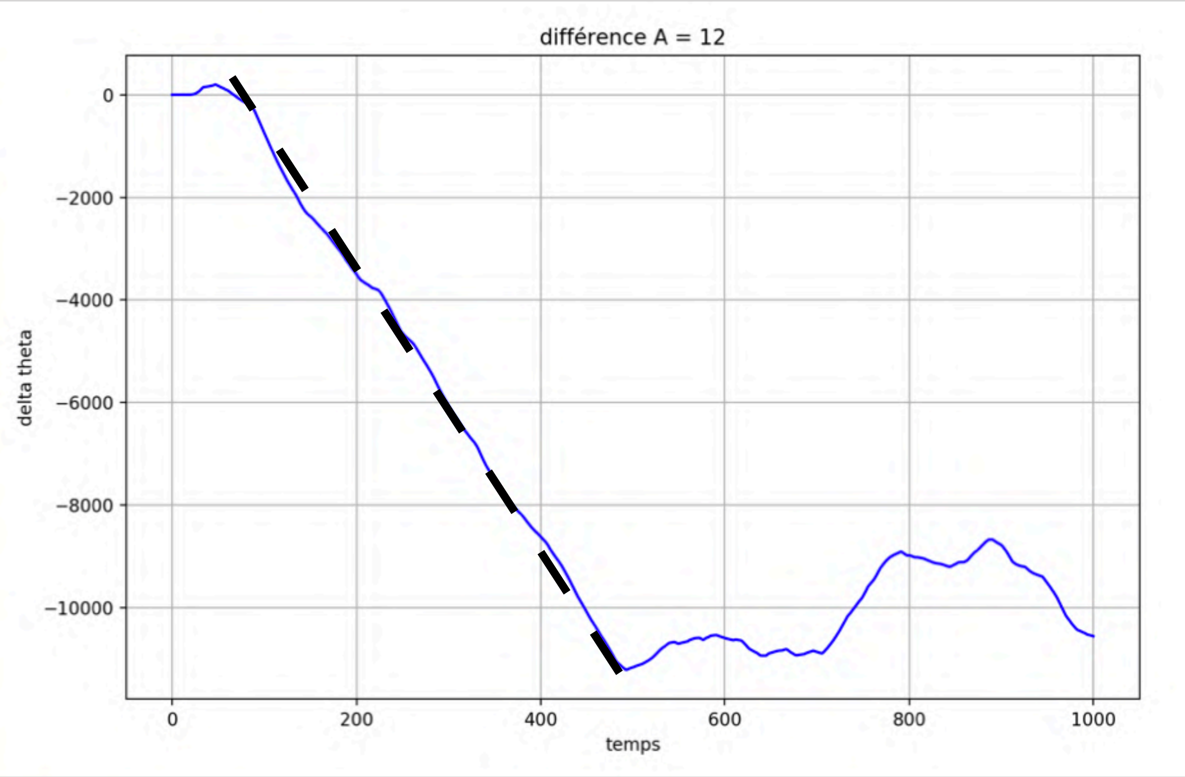
Existence d'un paramètre de chaos : le système est chaotique selon la valeur du paramètre A.

Le pendule à oscillations forcées : un système chaotique

Divergence de trajectoires pour deux conditions initiales « proches »



Différence angulaire pour deux conditions initiales « proches »

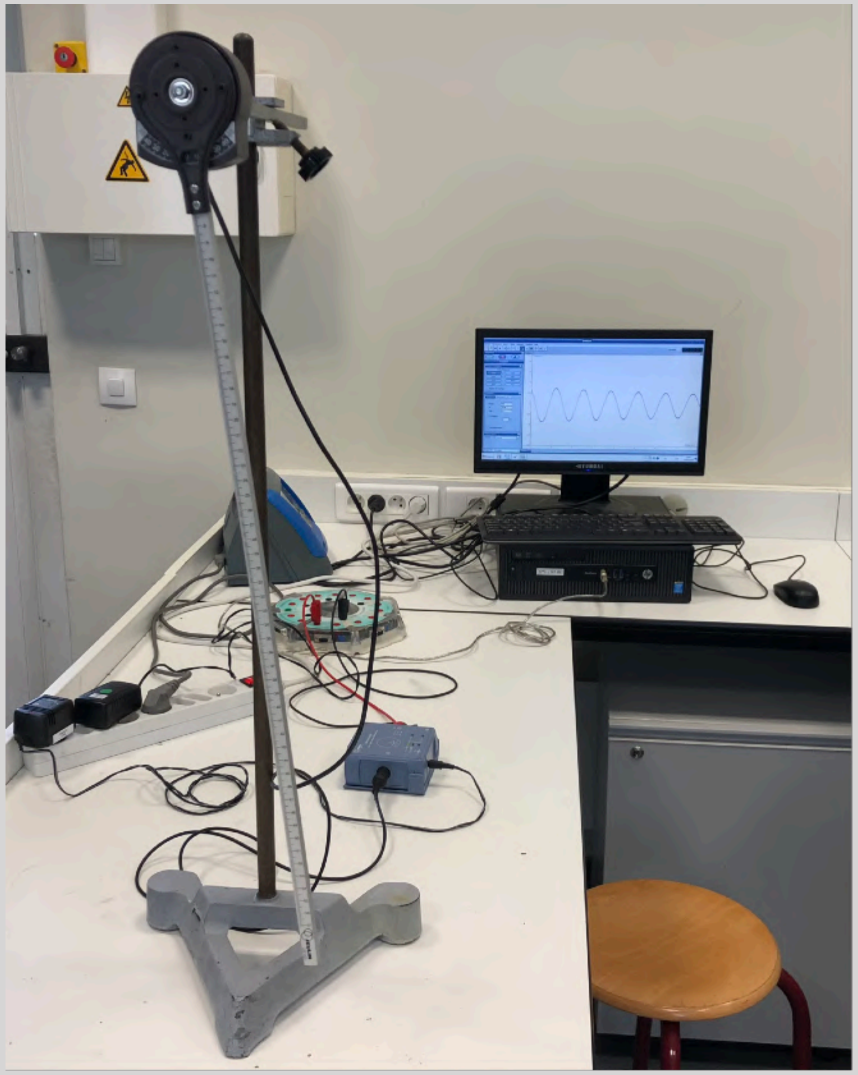


Nouvelle caractérisation du chaos dans les systèmes dynamiques : ***l'exposant de Lyapunov.***

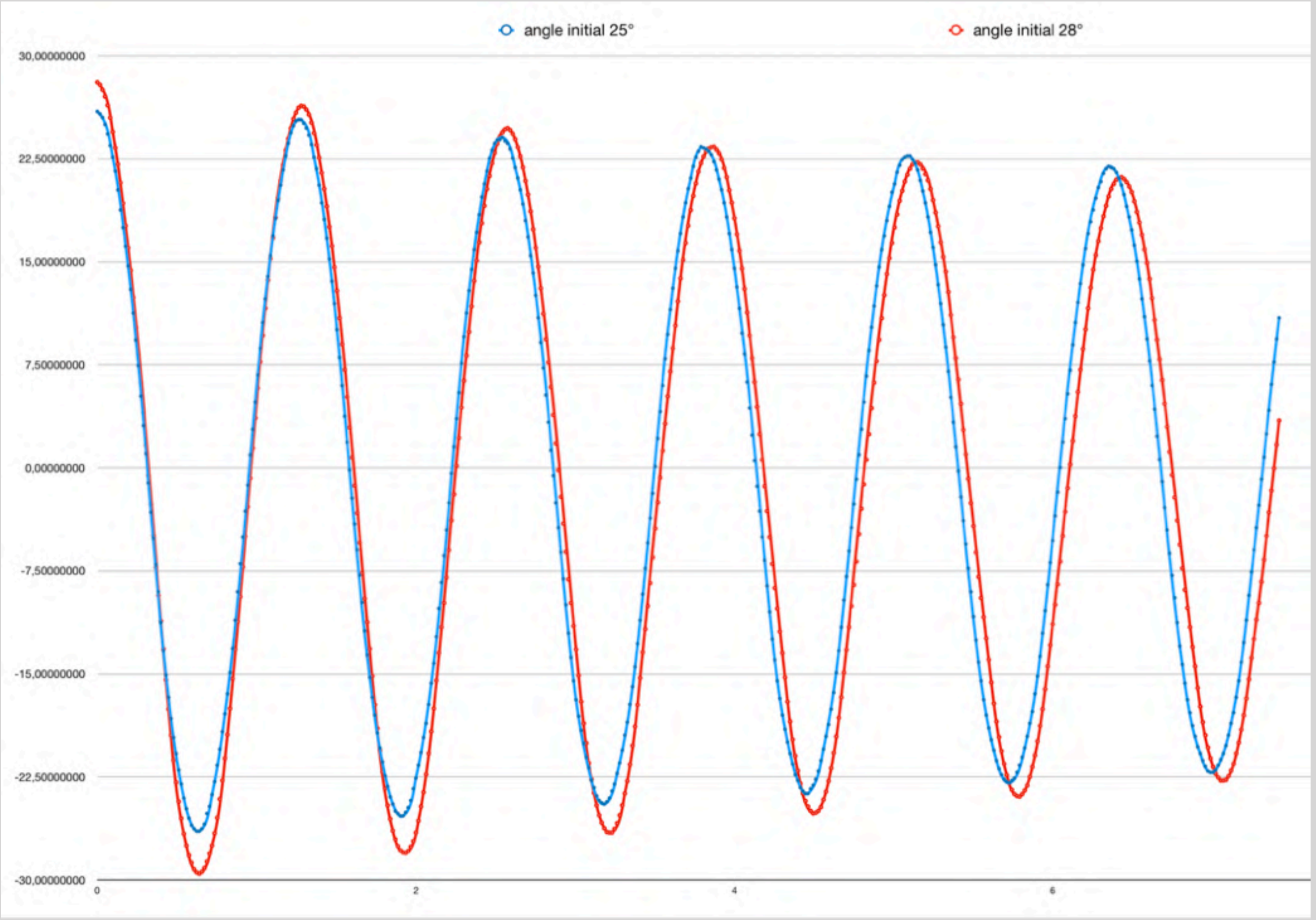
$$\Delta x(t) = \Delta x(0)e^{\lambda t}$$

Le pendule à oscillations forcées : application pratique

***Montage expérimental** du pendule libre*



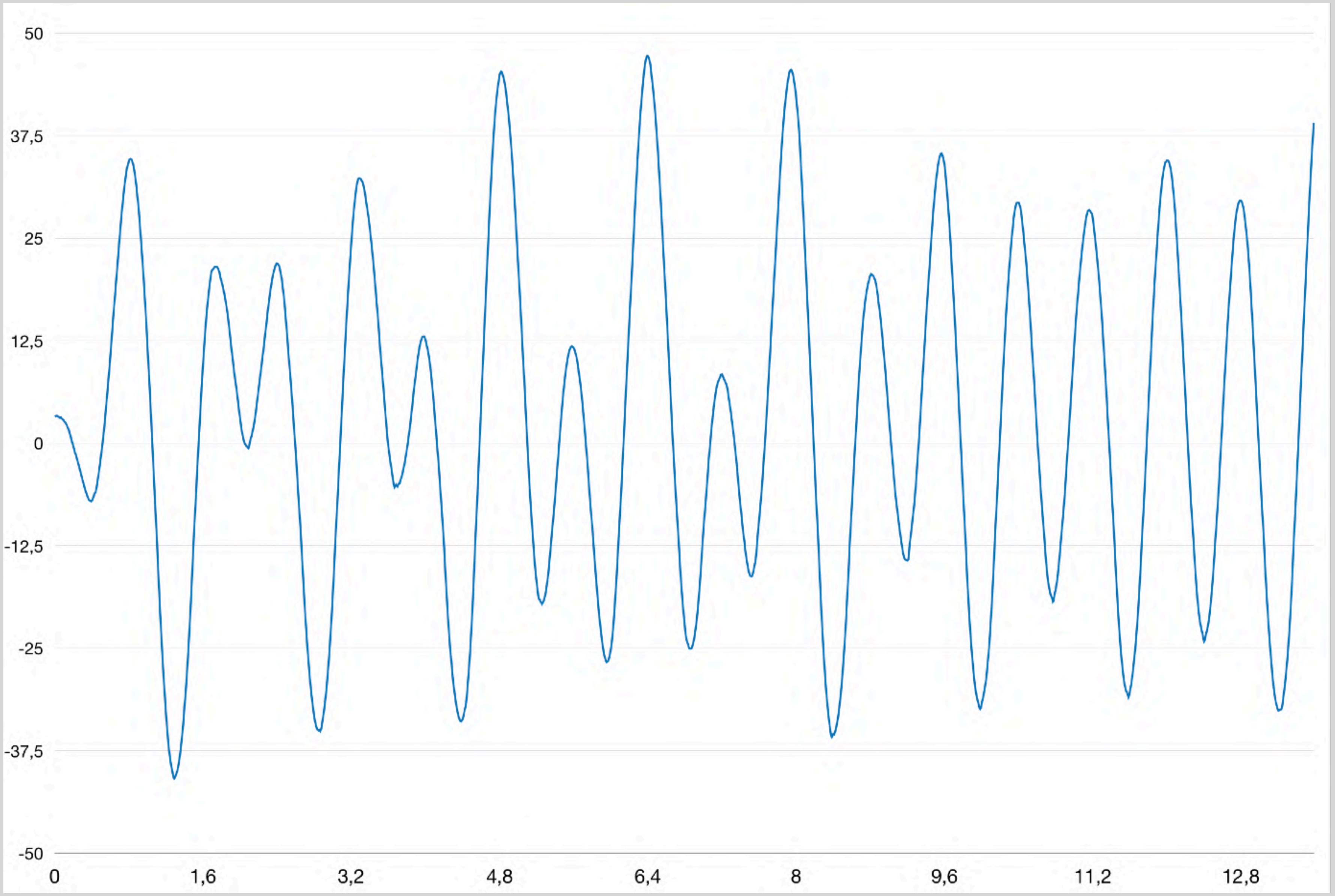
***Résultats** pour deux lancers avec des CI « proches »*



⇒ **Confirmation** des attentes théoriques : **système non chaotique**

Le pendule à oscillations forcées : application pratique

***Courbe angulaire** expérimentale pour un **système forcé** :*



⇒ **Confirmation** des attentes théoriques : **dédoublement** en plusieurs périodes, approche d'un **système chaotique**

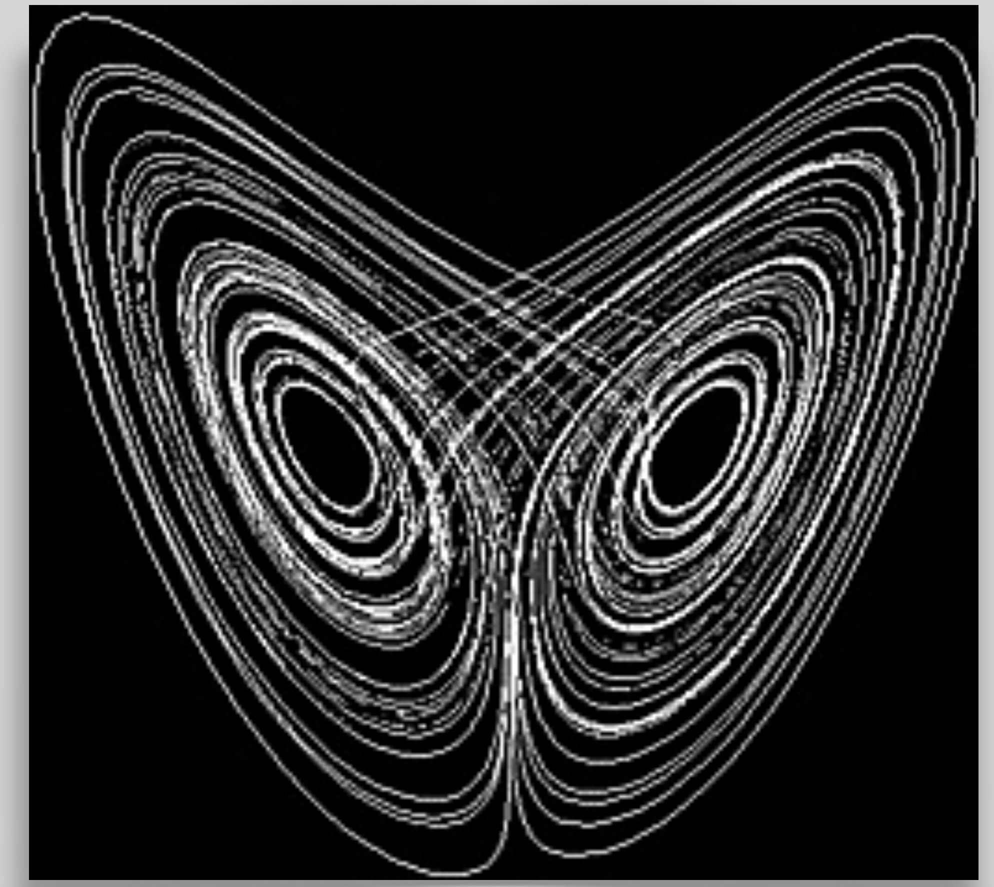
Limites de l'expérience : paramètre de chaos (amplitude) trop faible, le pendule n'effectue pas des tours entiers

Conclusion

« Une cause très petite, qui nous échappe, détermine un effet considérable que nous ne pouvons pas ne pas voir, et alors nous disons que cet effet est dû au hasard. »

Poincaré, *Science et Méthode*, 1908

La théorie du chaos est souvent assimilée à ***l'effet papillon*** *dans la presse et la littérature.*



```python
# paramètrage de la simulation :

g = 9.81     # accélération de la pesanteur
k = 0.25     # coefficient de frottement visqueux k>0
M = 0.5     # masse M (la masse de la tige rigide est négligeable devant M)
l = 1     # longueur de la tige
ampl = 0.216     # amplitude du couple moteur de forçage
w = 0.7     # pulsation de forçage
dt = 0.01    # pas temporel de la simulation


a = 0.001   #1
b = 9.81    #9            # on calcule les coefficients de l'équa diff
c = 30    #10       on remarquera que la valeur limite trouvée de manière empirique
                            #pour a = 0.001, b = 9.81 est c = 3.135 au dela il y a mouvement chaotique


def pend_forc_visq(theta0, omega0, tmax) :          # résout l'équa diff dans le cas d'un frottement visqueux
    t = t0 = 0                # on commence la simulation à t=0
    x = [t0]
    theta = [theta0]
    omega = [omega0]
    while t < tmax :
        x.append(t)         # on ajoute t à la liste des temps
        k1 = omega0*dt
        j1 = ( c*math.sin(w*t) - a*omega0 - b*math.sin(theta0) ) * dt                      # on calcule les coefficients de RK4
        k2 = ( omega0 + j1/2 ) * dt
        j2 = ( c*math.sin(w*t) - a*(omega0 + j1/2) - b*math.sin(theta0 + k1/2) ) * dt
        k3 = ( omega0 + j2/2 ) * dt
        j3 = ( c*math.sin(w*t) - a*(omega0 + j2/2) - b*math.sin(theta0 + k2/2) ) * dt
        k4 = ( omega0 + j3 ) * dt
        j4 = ( c*math.sin(w*t) - a*(omega0 + j3) - b*math.sin(theta0 + k3) ) * dt
        theta1 = theta0 + (1/6.0)*(k1 + 2*k2 + 2*k3 + k4)               # calcul de theta à t
        theta.append(theta1)                                            # on ajoute theta à la liste
        theta0 = theta1
        omega1 = omega0 + (1/6.0)*(j1 + 2*j2 + 2*j3 + j4)               # calcul de omega à t
        omega.append(omega1)                                            # on ajoute omega à la liste
        omega0 = omega1
        t = t + dt
    return x, theta, omega
```

```python
import numpy as np
import matplotlib.pyplot as plt
import math

l = 1
g = 9.81
w0 = g/l
m = 0.3

def f(y, v) :
    return [v, -(w0)**2*math.sin(y)]

def euler(f, t0, y0, v0, T, n) :
    dates = []
    Y = [[y0, v0]]
    y = y0
    v = v0
    for i in range(n) :
        t = t0 + i*T/n
        y = y + f(y, v)[0]*T/n
        v = v + f(y, v)[1]*T/n
        dates.append(t)
        Y.append([y, v])
    t = t0 + T
    dates.append(t)
    return dates, Y

temps = euler(f, 0, 1.5, 1, 10, 10000)[0]
X = euler(f, 0, 1.5, 1, 10, 10000)[1]

theta = []
vitesse = []
for i in range(len(X)) :
    theta.append(X[i][0])
    vitesse.append(X[i][1])



plt.plot(temps, theta)
plt.grid()
plt.show()
```

```python
import numpy as np
import matplotlib.pyplot as plt

r = 3.5

def f(x) :
    return r*x*(1-x)

u = [0.2]
n = 20

for i in range(n) :
    u.append(f(u[i]))

x = np.arange(0, 1, 0.01)
plt.plot(x, f(x), 'b-', x, x, 'r-')

for i in range(0,n-1):
    plt.plot([u[i], u[i+1]], [u[i+1], u[i+1]], 'k-')
    plt.plot([u[i+1], u[i+1]], [u[i+1], u[i+2]], 'k-')
    plt.plot([u[i], u[i]], [0, u[i+1]], 'k--')
plt.plot([u[n-1], u[n-1]], [0, u[n]], 'k--')
plt.show()
```

```python
import numpy, math, matplotlib # on importe les bibliothèques
import math
from pylab import *

# paramétrage de la simulation Un+1 = r*Un*(1-Un)
r = 3.6
u0 = 0.7
v0 = 0.71
n = 200

X=[0]
Y=[u0]
for i in range(n) :
    X.append(i)
    Y.append(r*Y[i-1]*(1-Y[i-1]))

plt.figure(1)
plt.scatter(X,Y)
plt.grid()
plt.show()
```

```python
import matplotlib.pyplot as plt
import numpy as np

size=[]
for i in range(200) :
    size.append(0.001)

R = np.linspace(0,4, 201)
for r in R:
    # calcul des points de la suite
    n = 200
    u = 0.5
    X = []
    Y = []
    for k in range(n):
        u = r*u*(1-u)
        if k >= 100:
            X.append(r)
            Y.append(u)
    plt.scatter(X,Y,size,'k')


plt.xlabel("valeurs de r")
plt.ylabel("valeurs de la suite logistique")
plt.show()
```