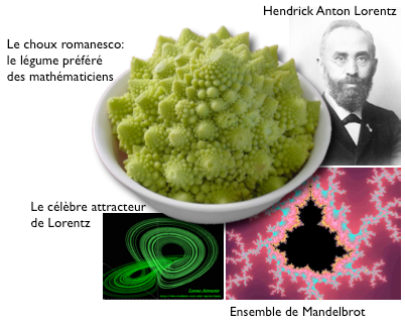
Matlab: Applications en mécanique

LA207, Université Pierre et Marie Curie.

`www.lmm.jussieu.fr/~hoepffner/enseignement`

Ce TP est à rendre sous la forme d'un compte-rendu: toutes les parties du sujet sont à rendre: "Manipulations", "Etude" ainsi que "Pour aller plus loin".

2.9 TP5: Un système chaotique modèle



Pour étudier la convection thermique dans l'atmosphère, Hendrick Anton Lorenz, en 1963 a dérivé un modèle mathématique très simplifié:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = rx - y - xy$$
$$\dot{z} = xy - bz$$

ou x,y et z sont trois variables qui évoluent dans le temps. Ici le point $\dot{x}$ signifie la dérivée temporelle. Lorenz s'est rendu compte que ce système en apparence si simple pouvait se comporter de manière très inattendue, cette surprise se traduit dans le titre de son article: "Deterministic nonperiodic flow", (J. Atmos. Sci, 20, 130). La solution oscille régulièrement, mais ne se répète jamais et reste toujours dans une zone bien définie. C'est ainsi que Lorenz a construit et étudié le premier exemple de système simple chaotique, et a mis en évidence des propriétés fondamentales: son évolution sur un attracteur étrange. Dans ce TP, nous allons suivre pas à pas les étapes qui ont mené Lorenz à ses découvertes. Ce TP est inspiré du chapitre 9 du livre (en anglais) "Nonlinear dynamics and chaos", de Steven H. Strogatz. Pour en savoir plus sur les systèmes chaotiques, vous pouvez aller feuilleter cet excellent livre.

Une fonction vous est fournie, que vous pouvez utiliser pour modéliser ce système: lorenz.m

```
[x,y,z,t]=lorentz(p0,tmax,r,sigma,b);
```

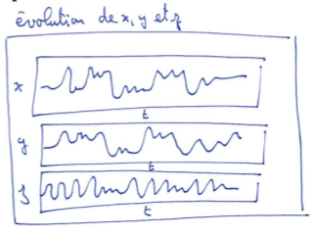

Les arguments d'entrée sont `p0`, un tableau à trois éléments: valeurs initiales de x, y, z; `tmax` est le temps final de la simulation; `r,sigma,b` qui sont trois paramètres numériques. On prendra $\sigma = 10, b = 8/3$. Les arguments de sortie sont de longs tableaux dans lesquels sont mémorisées les trajectoires de x, y, z dans le temps, ainsi que le vecteur temps `t`.

2.9.1 Manipulations

- **Fonction de deux variables:** Tracer la fonction $f(x, y) = sin(r)/r$, avec $r = \sqrt{x^2 + y^2}$ avec la fonction `mesh`. Vous choisirez les limites des axes selon x et y de sorte à ce qu'on voit bien la structure de la fonction: ses oscillations et le fait qu'elle tend vers zero lorsqu'on s'éloigne de l'origine.
- **Opérations logiques:** On construit un tableau de nombres aléatoires de distribution Gaussienne avec la fonction `randn` (à ne pas confondre avec la fonction `rand` qui construit un tableau de nombre aléatoires de distribution uniforme entre 0 et 1). En utilisant une opération logique de tableaux, montrer que pour un tableau aléatoire avec un grand nombre d'éléments, la moitié à peu près des éléments sont positifs et l'autre moitié sont négatifs.

2.9.2 Etude

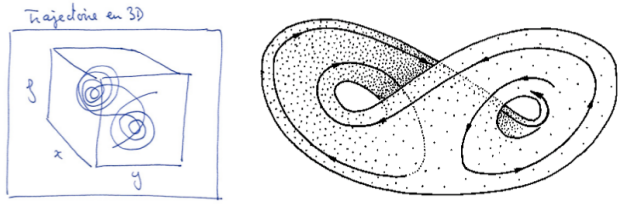
1. **Représentation d'une trajectoire**: Calculer une trajectoire avec condition initiale en $(10, 10, 10)$, du temps 0 au temps 10, avec $r = 35$. Tracez dans trois sous-graphiques l'évolution de x, y, et z en fonction du temps. Cette représentation nous donne une idée de l'évolution de chaque variable.



2. **Trajectoire en 3D**: Pour avoir une vue maintenant plus globale sur la trajectoire du système, nous allons tracer la trajectoire en trois dimensions, en considérant x,y, et z comme les coordonnées spatiales d'un point qui se déplace dans l'espace avec `plot3`. Vous pouvez utiliser `box on` pour voir plus clairement les limites du graph.

   Vous pouvez utiliser l'outil de rotation avec la souris pour observer cette trajectoire sous différents angles. Pour cela, utiliser le bouton du menu de la fenêtre graphique:

On observe que cette trajectoire se promène sur une surface qui ressemble à un papillon, voici une représentation de la structure de cette surface. En fait ce n'est pas exactement une surface, car cette structure a une épaisseur. C'est un attracteur étrange, qui "attire" la trajectoire.



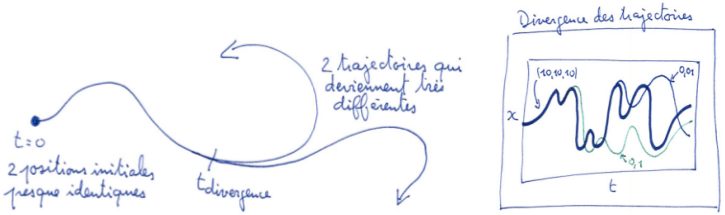
3. **Animation de la trajectoire**: Nous allons maintenant réaliser une animation de la trajectoire. Pour cela, faire une boucle sur l'indice temporel `i`, et à chaque pas de cette boucle, tracer la trajectoire du temps initial jusqu'au temps i. Pour mieux voir la position du système, on peut rajouter un cercle rouge aux coordonnées $(x(i), y(i), z(i))$. Pour fixer l'angle de vue on peut utiliser la commande `view(18,22)`.

4. **Comportements différents en fonction du paramètre** r:

   Maintenant, nous allons observer ce qui se passe pour différentes valeurs du paramètre r. Tracer dans quatre sous-graphiques l'évolution dans le temps de z pour successivement $r = 0.5, 5, 15, 25$. Qu'observez-vous? Comment qualifierez vous l'effet du paramètre r sur les propriétés de l'évolution du système. On trace maintenant l'évolution de z pour $r = 350$, quelque chose est maintenant différent, que ce passe-t'il?

5. **Sensibilité à la position initiale**: On se remet maintenant dans la plage de valeurs de r telles que le système se comporte de manière chaotique, par exemple $r = 35$. Les systèmes chaotiques ont la propriété que deux conditions initiales extrêmement proches peuvent donner engendrer des trajectoires divergentes: au bout d'un certain temps, les trajectoires ne se ressemblent plus. Nous allons tester cette propriété pour le système de Lorenz.

   Pour cela, tracez l'évolution dans le temps de z pour les conditions initiales $(10, 10, 10), (10, 10, 10.001), (10, 10, 10.01), (10, 10, 10.1)$ dans le même graphique. Vous choisirez un temps final `tmax` tels que on observe bien deux régimes: tout d'abord les trajectoires sont superposées, puis ensuite elles sont différentes.



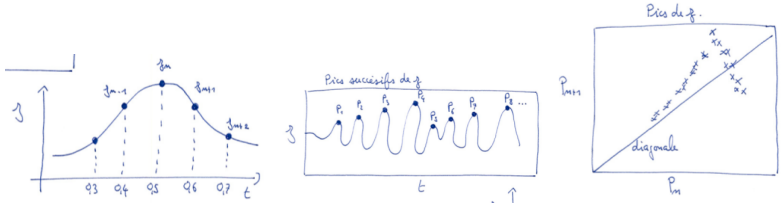
2.9.3 Pour aller plus loin

1. **Analyse des propriétés chaotiques**: Nous allons maintenant commencer à chercher l'*ordre dans le chaos*, et analyser les propriétés de l'attracteur étrange.

   Lorenz s'est rendu compte que les pics de l'évolution de la coordonnée z semblaient très réguliers, et l'idée lui est venue de tracer le graph de la valeur d'un pic donné en fonction de la valeur du pic précédent, p_{n+1} en fonction de p_n, et miracle, un graphique très simple s'est dessiné.

   Pour cela, la première étape consiste à obtenir à partir de la trajectoire selon z, les pics successifs. Un pic est une valeur z telle que au temps précédent, z est plus petit et que au temps suivant, z est aussi plus petit. Dans le schéma ci dessous, il y a clairement un pic au temps 0.5.

   Pour trouver ces pics, on eut faire une boucle sur i, l'indice temporel, et on garde les valeurs de z telles que `z(i)>z(i+1) & z(i)>z(i-1)`



   Nous avons maintenant une série p_n, et on trace pour chaque pic, sa valeur en fonction du pic précédent, c'est à dire pour tous les n, on trace un point d'abscisse p_n et d'ordonnée p_{n+1}.

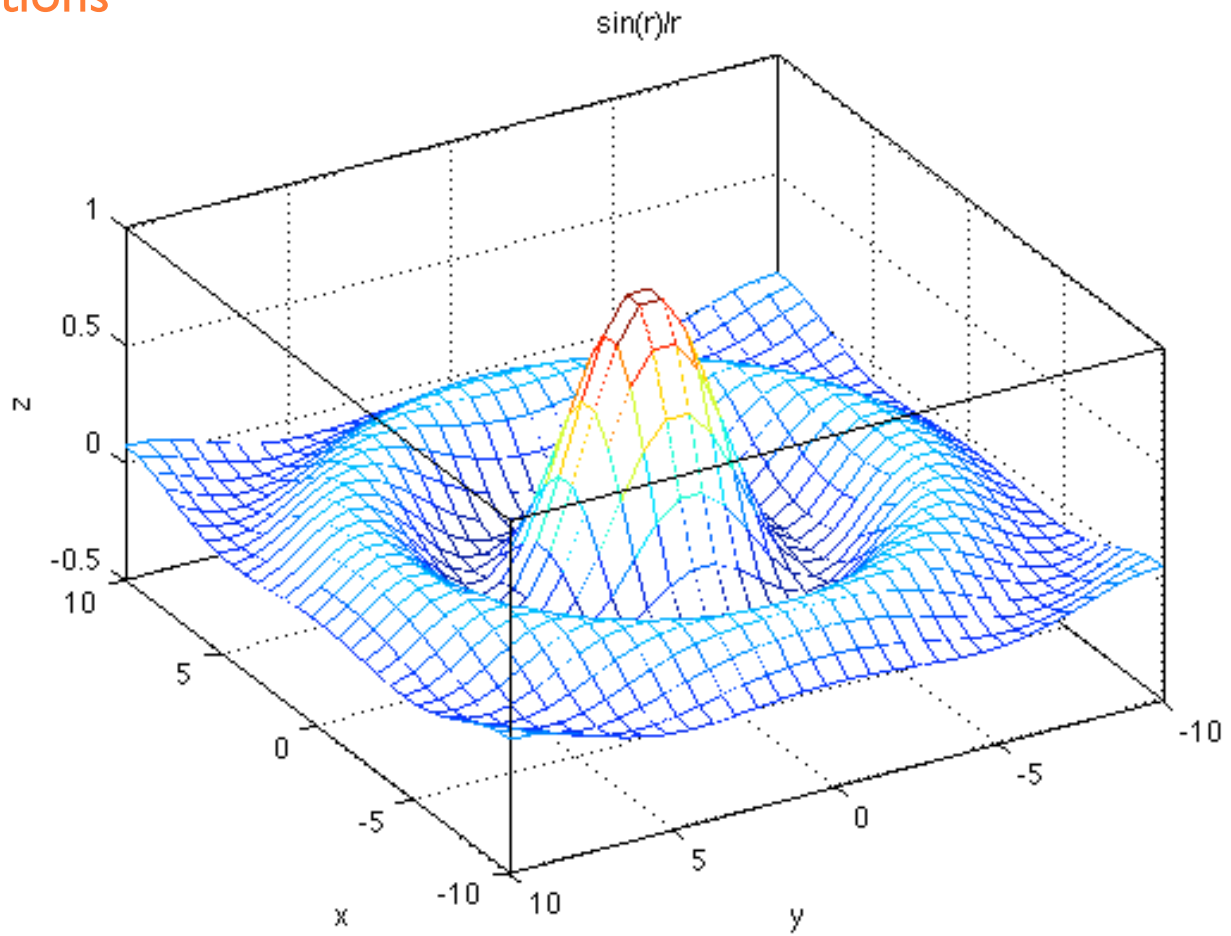
   Voici le graph qui a convaincu Lorenz que dans le chaos de son système, il y avait de l'ordre... Idée qui s'est avérée par la suite très féconde...

Matlab : applications en mécanique
LA207, 2010

http://www.lmm.jussieu.fr/~hoepffner/enseignement

Compte rendu TP6: attracteur de Lorenz

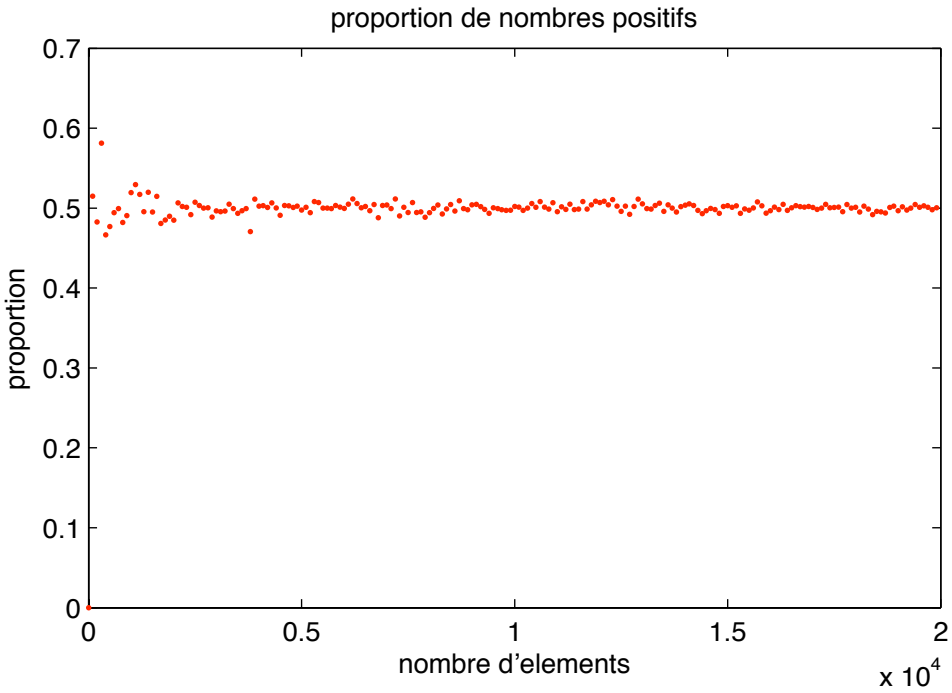
0) Manipulations




```
% manipulations
l=10;
x=linspace(-l,l,30);
y=linspace(-l,l,30);

[X,Y]=meshgrid(x,y);

R=sqrt(X.^2+Y.^2);
f=sin(R)./R;

mesh(X,Y,f);

xlabel('x');
ylabel('y');
zlabel('z');
title('sin(r)/r')
grid on
box on
```

Tracé en 3D de la fonction sin(r)/r avec la fonction “mesh”.
Ici x et y sont dans l’intervale [-10,10].

1) Statistiques et probabilités



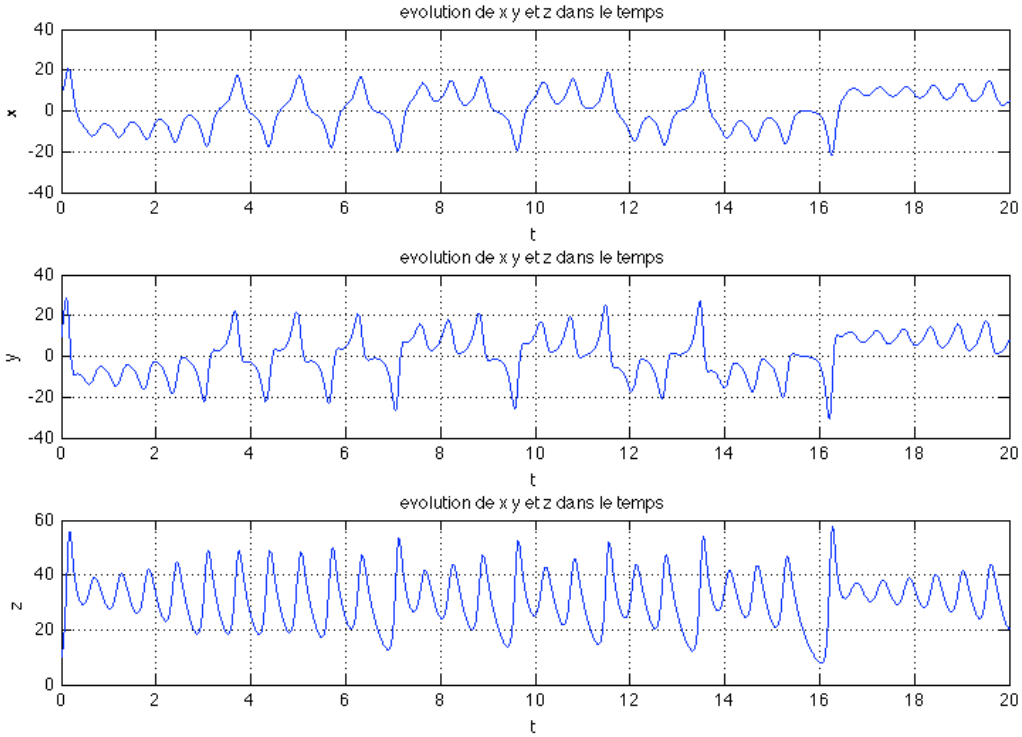
Script

```
% on compte les elements positifs
% boucle sur le nombre d'elements
for n=1:100:20000;
a=randn(n,1); % on construit le tableau aleatoire
c=sum(a>0)/n;% on compte
plot(n,c,'r.')% on trace
hold on
end
% annotations du graphique
title('proportion de nombres positifs')
xlabel('nombre d''elements');
ylabel('proportion')
break
```

On voit clairement que lorsque le nombre d'éléments du tableau aléatoire devient grand, on se retrouve avec la moité des éléments qui sont positifs (et l'autre moitié negatifs)

Attracteur de Lorenz

1) Simulation et trajectoires



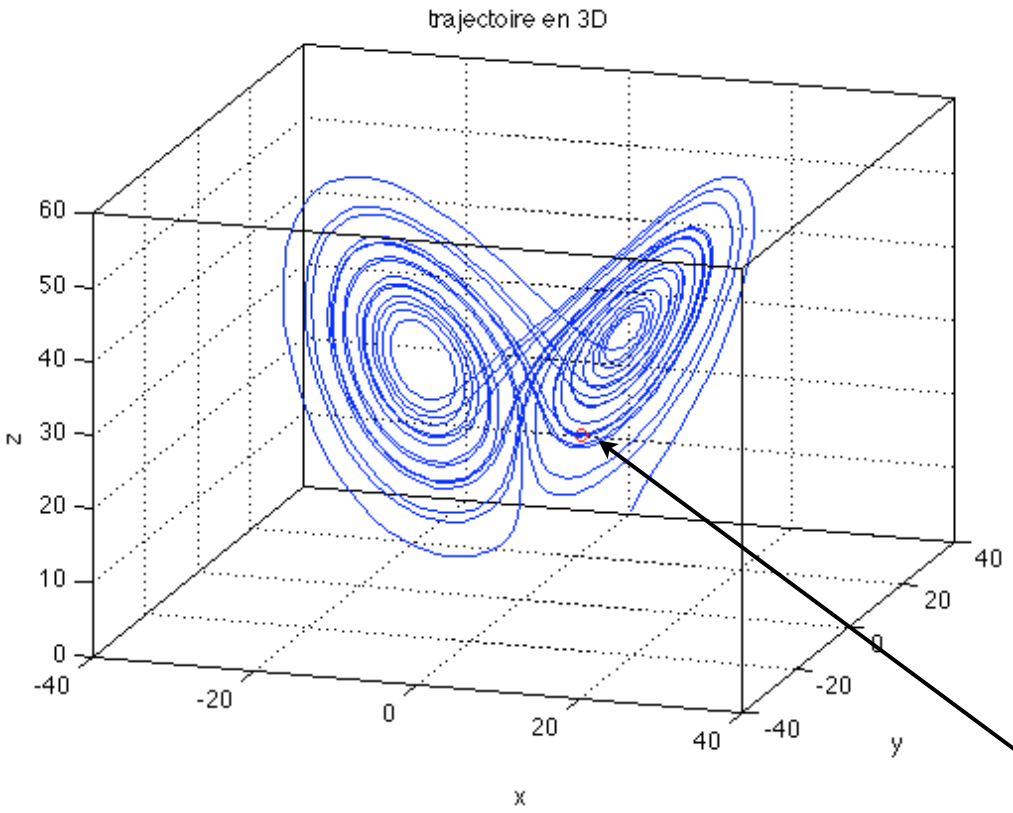
Nous avons tracé dans trois sous-graphiques l'évolution dans le temps des coordonnées x y et z pour t de 0 à 20, avec la condition initiale (10,10,10), et le paramètre r=35.

On observe que les trois coordonnées oscillent dans le temps, sans périodicité apparente.

Script

```
% tp6: attracteur de Lorenz
clear all; clf



r=35;
sigma=10;
b=8/3;

% position initiale
p0=[10 10 10];

% temps final
tmax=20;

% simulation
[x,y,z,t]=lorentz(p0,tmax,r,sigma,b);




% évolution des trois coordonnées
subplot(3,1,1); plot(t,x);
xlabel('t')
ylabel('x')
title('évolution de x y et z dans le temps');
grid on

subplot(3,1,2); plot(t,y);
xlabel('t')
ylabel('y')
title('évolution de x y et z dans le temps');
grid on

subplot(3,1,3); plot(t,z);
xlabel('t')
ylabel('z')
title('évolution de x y et z dans le temps');
grid on
```

2) Trajectoire en 3D



Ici pour la même condition initiale et les même paramètres, nous avons tracé la trajectoire en 3D avec la fonction plot3. On observe une structure en papillon.

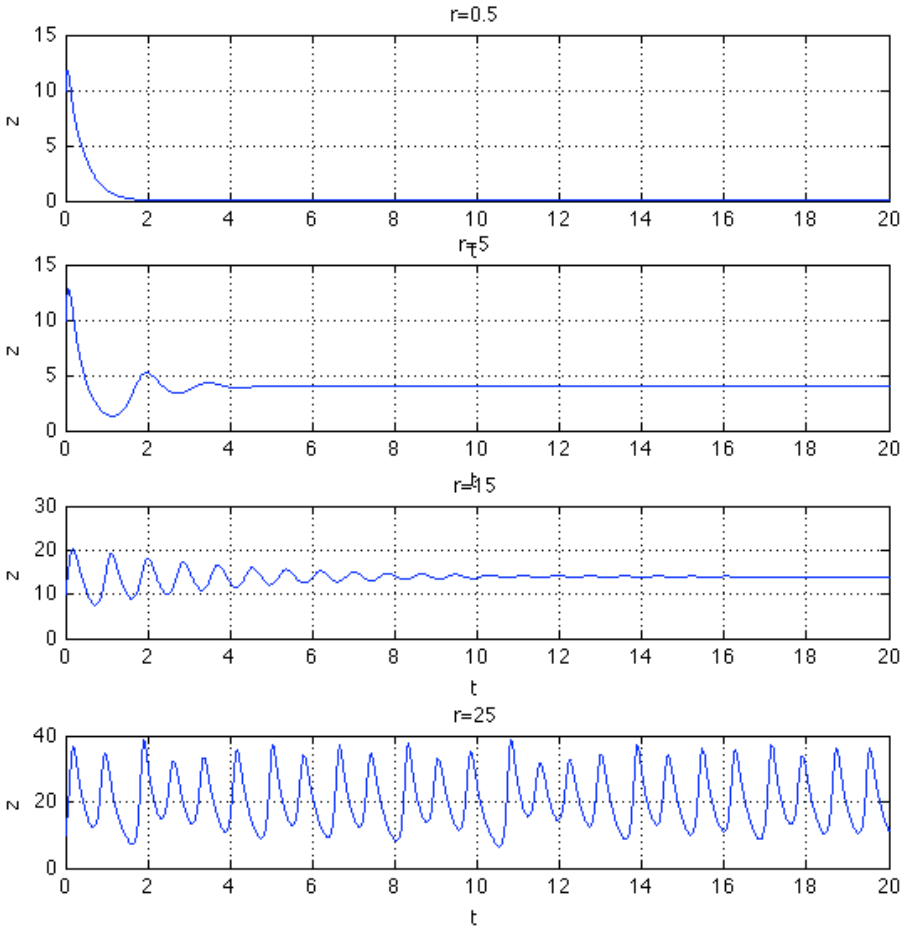
L'animation de la trajectoire montre bien que le point (x,y,z) tourne pendant un certain temps autour de chaque aile de ce papillon, et change de côté lorsque le rayon de rotation devient grand.

Pour l'animation, on trace la trajectoire depuis le temps zéro jusqu'à l'indice courant de la boucle "ind", et on ajoute un cercle rouge qui positionne la position courante

Script

```
% la trajectoire statique
clf
plot3(x,y,z)
box on
grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('trajectoire en 3D');
view(18,22)


% animation de la trajectoire
for ind=1:5:length(t)

    sel=1:ind;
    plot3(x(sel),y(sel),z(sel))
    hold on
    plot3(x(ind),y(ind),z(ind),'ro')
    hold off

    box on
    grid on
    xlabel('x')
    ylabel('y')
    zlabel('z')
    title('trajectoire en 3D');
    xlim([-40,40]);ylim([-40,40]);zlim([0,60]);
    view(18,22)
    drawnow
end
```

3) Effet du paramètre r



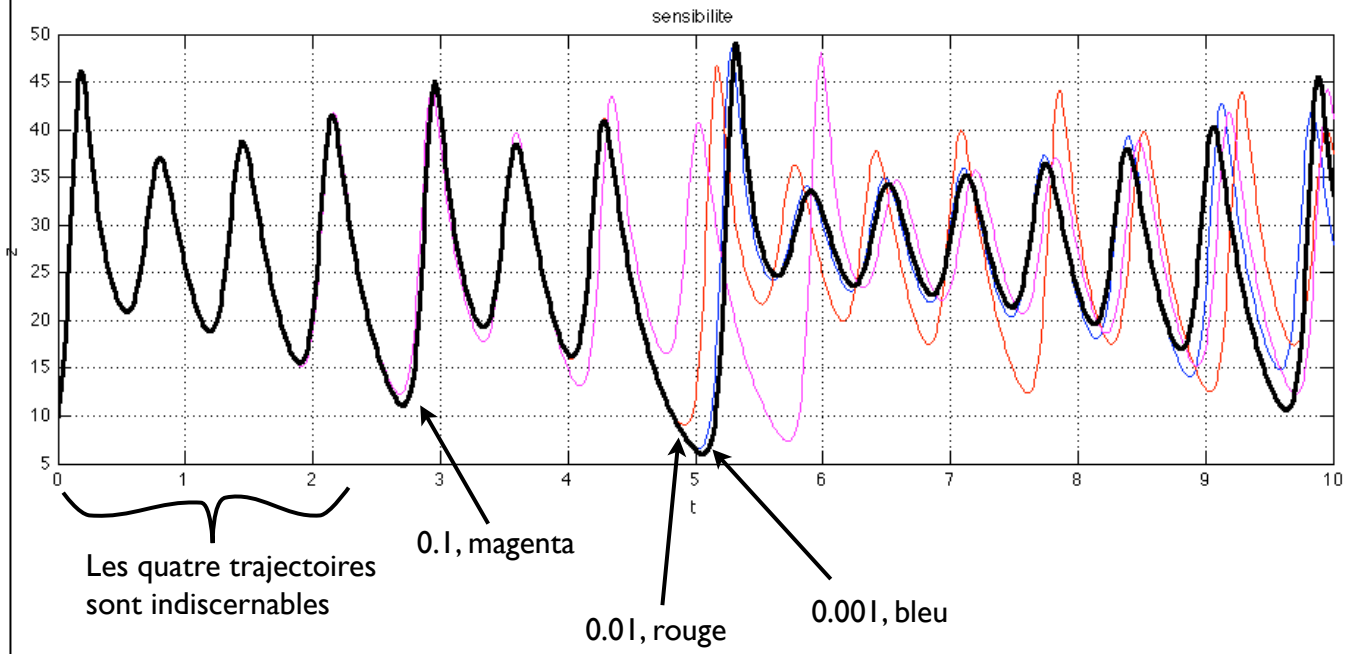
Script

```
% etude du paramètre r
r=0.5
[x,y,z,t]=lorentz(p0,tmax,r,sigma,b);
subplot(4,1,1);
plot(t,z)
grid on
xlabel('t'); ylabel('z');
title('r=0.5');

r=5
[x,y,z,t]=lorentz(p0,tmax,r,sigma,b);
subplot(4,1,2);
plot(t,z)
grid on
xlabel('t'); ylabel('z');
title('r=5');

r=15
[x,y,z,t]=lorentz(p0,tmax,r,sigma,b);
subplot(4,1,3);
plot(t,z)
grid on
xlabel('t'); ylabel('z');
title('r=15');

r=25
[x,y,z,t]=lorentz(p0,tmax,r,sigma,b);
subplot(4,1,4);
plot(t,z)
grid on
xlabel('t'); ylabel('z');
title('r=25');
```

Voici l'évolution dans le temps de la coordonnée z pour quatre valeurs du paramètre r: 0.5, 5, 15 et 25. On observe que pour 0.5, z tend rapidement vers 0: il n'y a pas d'oscillations. Pour 5, le système tend vers a peu près 4. Pour 15, le système oscille longtemps avant de tendre vers à peu près 13. Enfin, pour r=25, le système oscille de manière chaotique comme ce que nous avons observé dans les graphiques précédents.

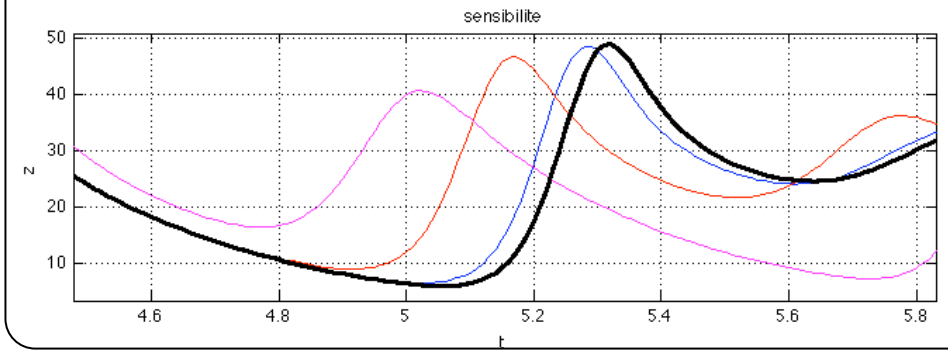
On peut en déduire que r est un paramètre clé pour le comportement du système: plus r est grand plus le comportement est chaotique. Pour r=350, le système diverge.

4) Sensibilité aux conditions initiales



Quatre trajectoires très similaires

Ici nous avons représenté l'évolution de la coordonnée z pour quatre conditions initiales proches de (10,10,10). Jusqu'à t=2.5, les quatre trajectoires sont indiscernables. Ensuite, les trajectoires s'éloignent de la trajectoire de référence. A t=2.7 pour 0.1, à t=4.9 pour 0.01, et à t=5.1 pour 0.001. Plus la perturbation est petite, plus la trajectoire dévie tard de la trajectoire de référence.

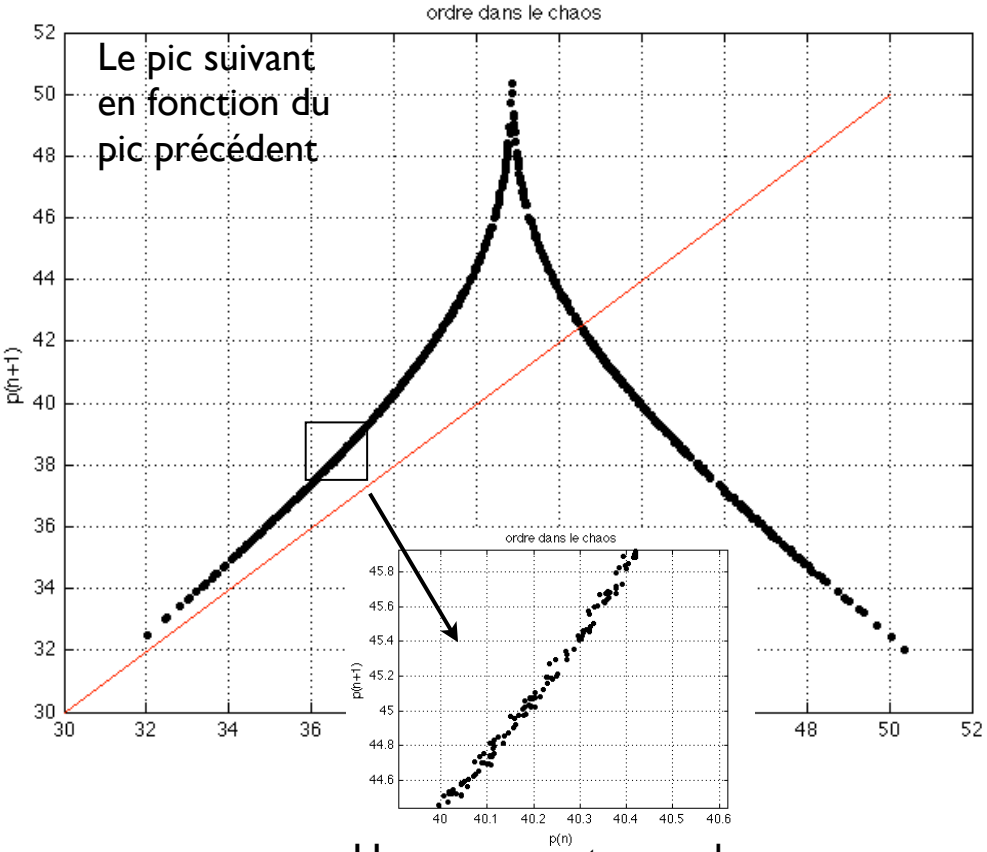


zoom autour de t=5, lorsque les trois trajectoires perturbées s'éloignent de la trajectoire de référence.

Script

```
% sensibilité aux conditions initiales

r=30
tmax=10;

p0=[10 10 10];
[x1,y1,z1,t]=lorentz(p0,tmax,r,sigma,b);

p0=[10 10 10.001];
[x2,y2,z2,t]=lorentz(p0,tmax,r,sigma,b);

p0=[10 10 10.01];
[x3,y3,z3,t]=lorentz(p0,tmax,r,sigma,b);

p0=[10 10 10.1];
[x4,y4,z4,t]=lorentz(p0,tmax,r,sigma,b);


subplot(1,1,1);
plot(t,z2,'b',t,z3,'r',t,z4,'m')
hold on
plot(t,z1,'k','linewidth',2);
hold off
grid on
xlabel('t'); ylabel('z');
title('sensibilite');
```

5) Pour aller plus loin



Un zoom montre que la ligne à bien une épaisseur: c'est en fait un ensemble fractal.

Script

```
% ordre dans le chaos
r=30
tmax=500;
p0=[10 10 10];
[x,y,z,t]=lorentz(p0,tmax,r,sigma,b);

% trouver les maximums
p=[];
n=length(t);
for ind=2:n-1;
if (z(ind)>z(ind-1))&(z(ind)>z(ind+1));
    p=[p,z(ind)];
end
end

% tracer les points
for ind=1:length(p)-1;
    plot(p(ind),p(ind+1),'k.','markersize',5); hold on
end

plot([0,50],[0,50],'r');
xlim([30,52]); ylim([30,52]);
hold off
grid on

xlabel('p(n)');
ylabel('p(n+1)');
title('ordre dans le chaos');
```

Nous avons calculé les pics de l'évolution de la coordonnée z, pour t de 0 à 300. Ces pics sont tracés dans le graphique par des points: p(n) en abscisse et p(n+1) en ordonnée. On observe une structure très nette et symétrique.

La ligne rouge représente la diagonale: les points au dessus de cette ligne correspondent à des pics tels que le pic suivant est plus grand, les points au dessous correspondent à des pics tels que le pic suivant est plus petit.

5) La fonction Lorenz.m

On utilise une méthode de discrétisation très simple dans le temps pour simuler l'évolution de ce système d'équations nonlinéaires: Euler explicite.

Voici ou sont codée les équations du système de Lorentz:

Script

```
function [xvec,yvec,zvec,tvec]=lorentz(p0,tmax,r,sigma,b);
% une fonction qui calcule l'?volution dans
% le temps pour le systeme de Lorentz, avec condition
% initiale donn?e, et param?tre r


% le pas de temps
dt=0.001;
tvec=0:dt:tmax;
n=length(tvec);

% initialisation pour le resultat
xvec=zeros(n,1);
yvec=zeros(n,1);
zvec=zeros(n,1);

% condition initiale
x=p0(1); xvec(1)=x;
y=p0(2); yvec(1)=y;
z=p0(3); zvec(1)=z;

% boucle de simulation
% discretisation de 'Euler explicite'
for ind=2:n
    % calcul des derivee temporelles
    xdot=sigma*(y-x);
    ydot=x*(r-z)-y;
    zdot=x*y - b*z;

    % marche un pas de temps
    x=x+dt*xdot;
    y=y+dt*ydot;
    z=z+dt*zdot;

    % memorise l'evolution
    xvec(ind)=x;
    yvec(ind)=y;
    zvec(ind)=z;
end
```