L'attracteur de Lorenz avec PSTricks

12 juin 2012

1 Présentation

Ceci n'est pas un cours sur l'attracteur de Lorenz, il existe sur internet un grand nombre de sites qui lui sont consacrés très complets et d'une grande qualité pédagogique. Il s'agit, plus simplement, de montrer que l'utilisation de PSTricks permet d'illustrer ce modèle très convenablement.

Pour résoudre numériquement le système de Lorenz :

$$\begin{cases} \dfrac{\mathrm{d}x}{\mathrm{d}t} &= \sigma(x-y) \\ \dfrac{\mathrm{d}y}{\mathrm{d}t} &= rx-y-z \\ \dfrac{\mathrm{d}z}{\mathrm{d}t} &= xy-\beta z \end{cases}$$

j'utilise une version très simplifiée de la macro `\psplotDiffEqn` de Dominique Rodriguez qui est incluse dans le package `pstricks-add`, mais qui était, à l'origine, un package '`pst-eqdf`' à elle seule. Dominique m'a donné la permission de modifier sa macro afin de pouvoir sauvegarder dans un tableau `postscript`, et/ou dans un fichier, toutes les valeurs nécessaires au dessin des figures et en particulier à la représentation en 3D du *papillon* de Lorenz. Cette macro simplifiée, qui utilise la méthode Runge-Kutta 4 est incluse dans un tout petit package dont j'ai gardé le même nom `pst-eqdf`, mais la macro a changé de nom `\psequadiff` car elle ne trace plus les fonctions directement.

Pour les couleurs, j'utilise une macro `\listplotHSB` qui est un complément à celle-ci : `\parametricplotHSB`[1], que nous avions mise au point avec Denis Girou.

2 Les étapes de la construction des figures

2.1 Équations, paramètres et conditions initiales

Il faut d'abord définir les équations en notation algébrique pour la macro `\psequadiff` :

```
% y[0] y[1] y[2]
% x    y    z
\def\Lorenz{
Pr*(y[1]-y[0])|%
Rh*y[0]-y[1]-y[0]*y[2]|%
y[0]*y[1]-b*y[2]%
}
```

ainsi que les paramètres :

```
\def\parametres{
    /Pr 10 def
    /Rh 28 def
    /b 8 3 div def
}%
```

puis les conditions initiales :

```
                          x0 y0 z0
\def\conditionsInitiales{10 10 30}
```

Ce sont ces deux dernières définitions qu'il faudra modifier si on veut changer les conditions initiales et les paramètres.

[1] `http://melusine.eu.org/syracuse/G/pstricks/courbes-arc-en-ciel/`

2.2 L'utilisation de la commande \psequadiff

```
\pstVerb{\parametres}%
  \psequadiff[% x,y
  tabname=tabXY,
  method=rk4,
  whichabs=0,whichord=1,
  saveData,filename=LorenzXY.dat,
  plotpoints=2501,algebraic]{0}{25}{\conditionsInitiales}{\Lorenz}
```

Dans cet exemple, l'intervalle de temps choisi est de 0 à 25 s. Les grandeurs prises sur la pile sont (x,y). Le nombre de points calculés vaut 2501. Les valeurs, sur cet intervalle de temps, sont stockées dans le tableau `postscript` : [**tabname**=tabXY] avec tabXY qui est le nom que l'on choisit soi-même.

Pour ne pas refaire le calcul, on peut choisir d'enregistrer la liste de valeurs du couple (x,y), en positionnant le booléen [saveData] à true, on choisit un nom pour le fichier avec : [**filename**=LorenzXY.dat]. Par défaut, aucun fichier n'est enregistré.

```
saveData,filename=LorenzXY.dat,
```

Pour utiliser ce fichier, il suffit de faire :

```
\pstVerb{/XY {(LorenzXY.dat) run } def}%
\listplotHSB{XY}
```

Vous pouvez utiliser la commande **lisplot** du package pst-plot qui offre davantage d'options que **listplotHSB**.

2.3 Le tracé de la courbe

```
\listplotHSB[unit=0.25]{tabXY aload pop}%
```

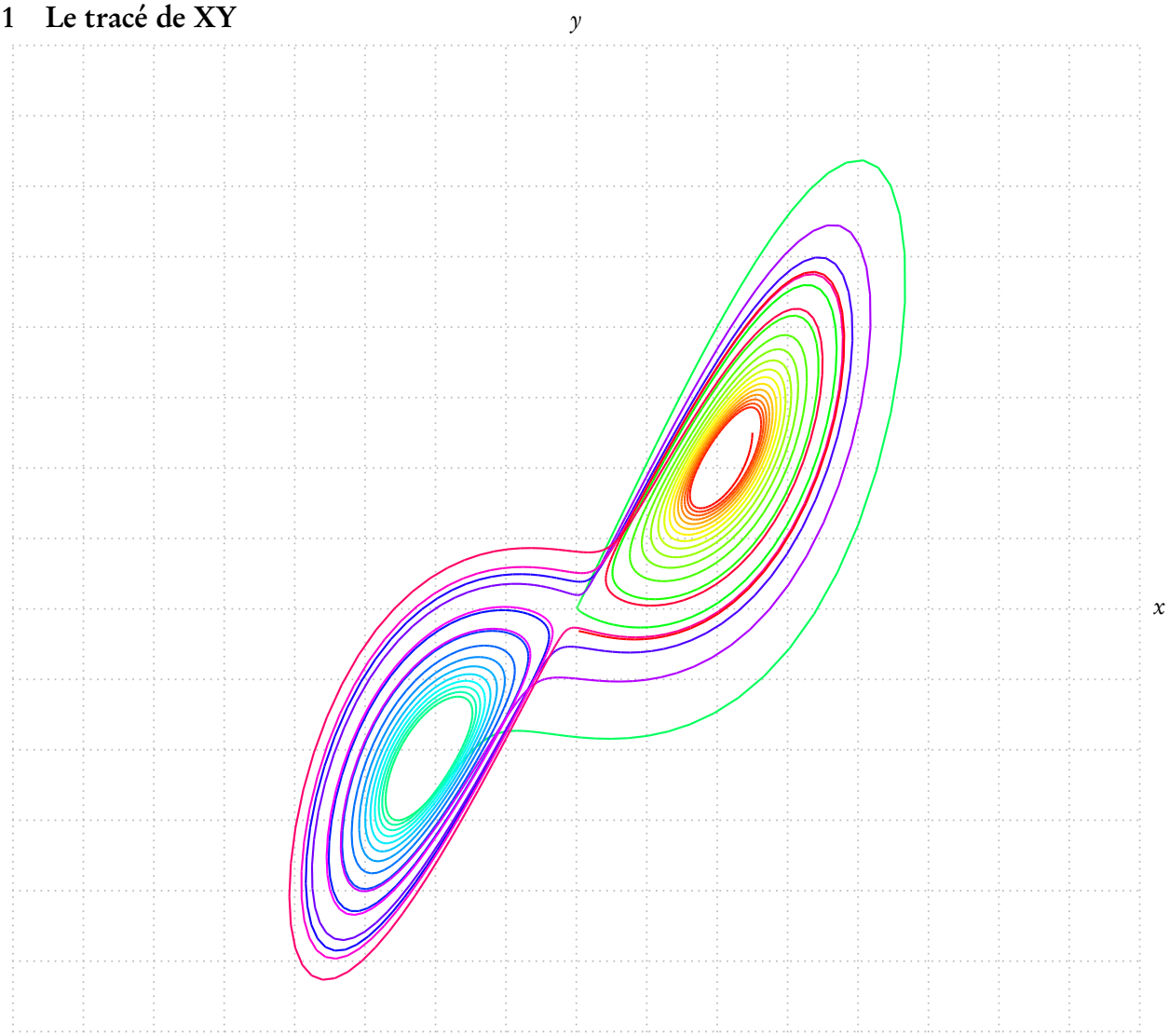
`\listplotHSB` permet de faire varier la couleur tout au long du tracé. Par défaut le paramètre hue de la définition d'une couleur dans le système hsb varie de 0 à 1. On peut fixer la plage de valeurs avec les 2 options suivantes :

- **HueBegin** avec une valeur comprise entre 0 et 1.
- **HueEnd** avec une valeur comprise entre 0 et 1.

Pour avoir une couleur uniforme tout au long du tracé, il faut désactiver l'option [**HSB**=false] ainsi le tracé prendra la couleur que vous avez choisie avec l'option habituelle linecolor.

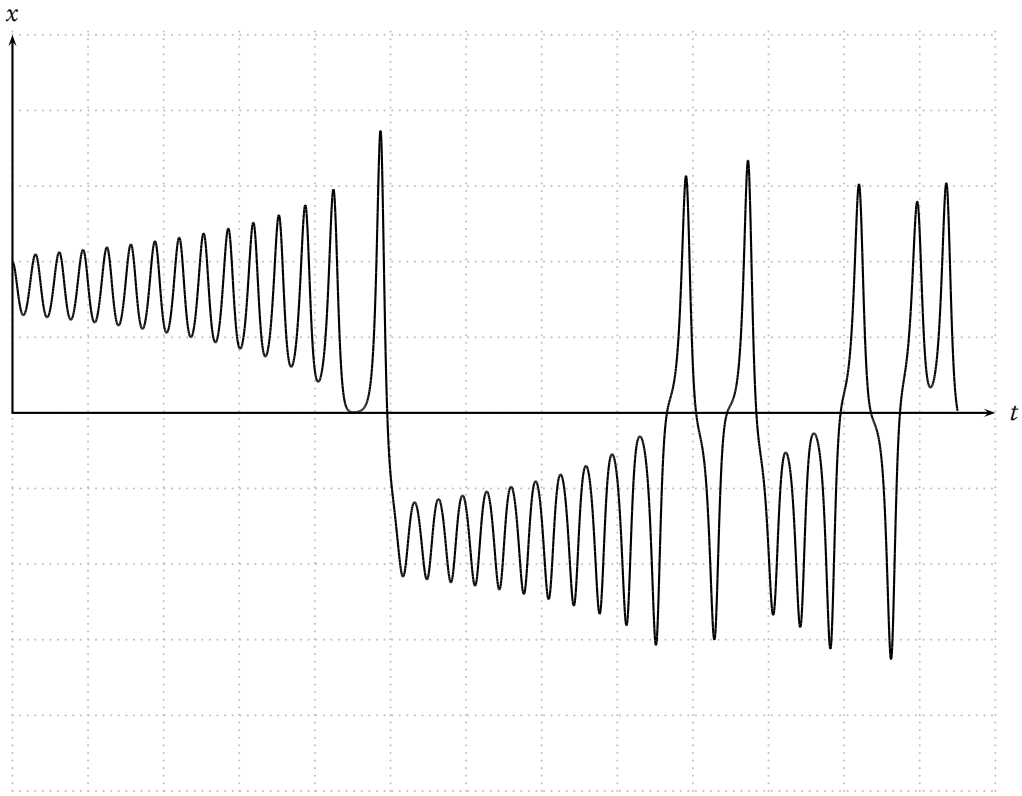
3 Le tracé des courbes

3.1 Le tracé de XY



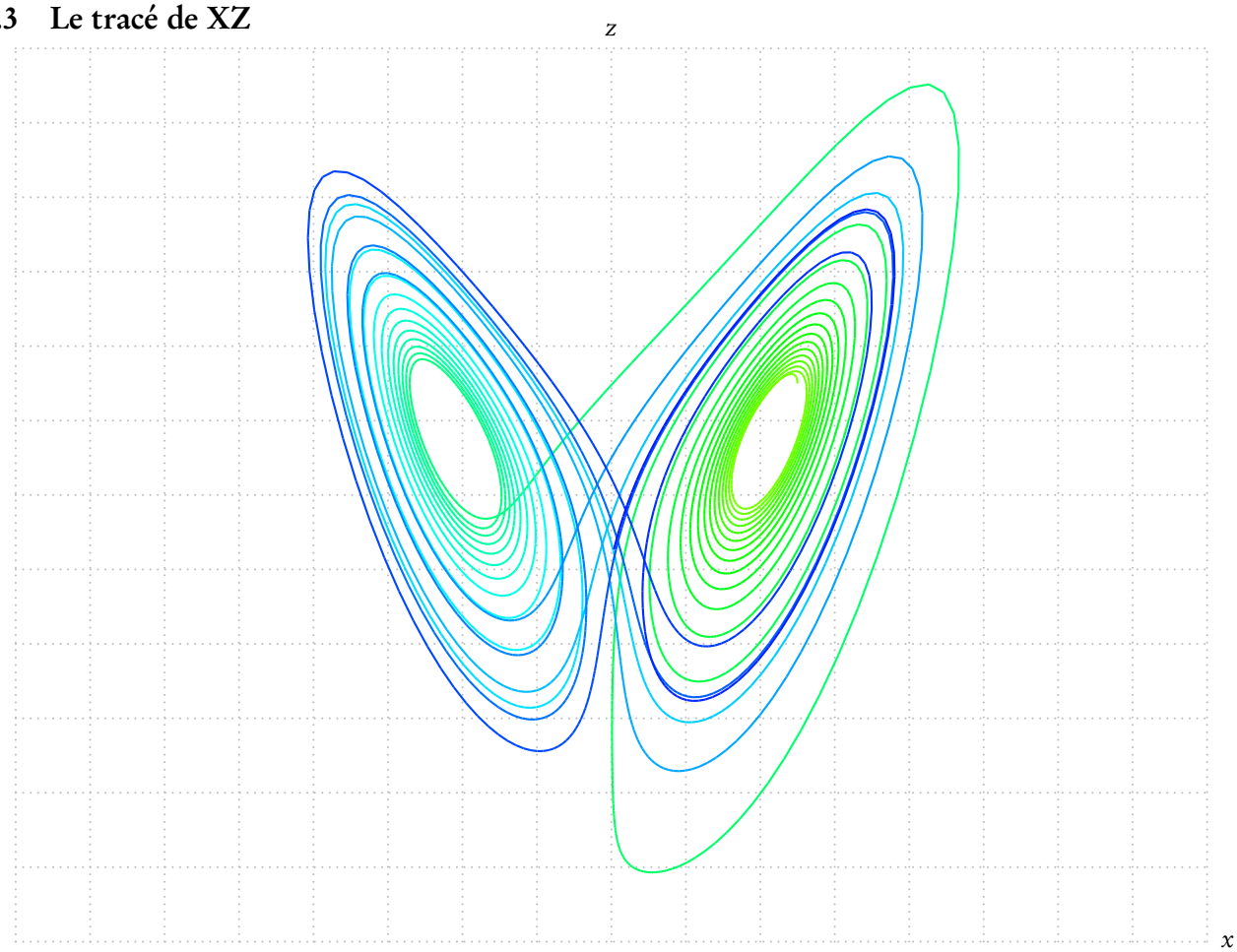

```
\begin{pspicture}(-8,-6)(8,8)
\psgrid[subgriddiv=0,gridcolor=lightgray,griddots=10,gridlabels=0pt]%
\pstVerb{\parametres}%
  \psequadiff[% x,y
  tabname=tabXY,
  method=rk4,
  whichabs=0,whichord=1,
  saveData,filename=LorenzXY.dat,
  plotpoints=2501,algebraic]{0}{25}{\conditionsInitiales}{\Lorenz}
\listplotHSB[unit=0.25]{tabXY aload pop}%
\uput[r](8,0){$x$}
\uput[u](0,8){$y$}
\end{pspicture}
```

3.2 Le tracé de X(t)



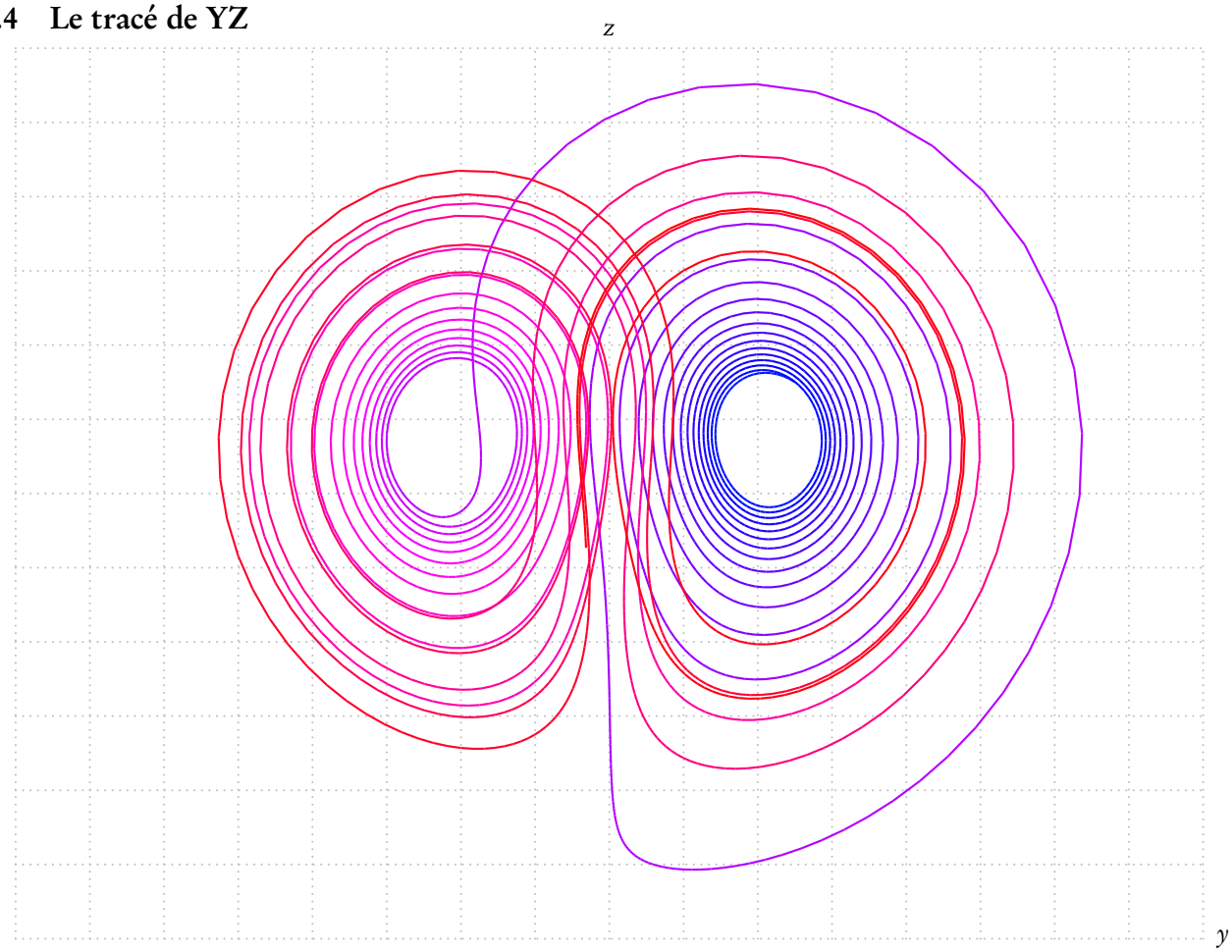

```
\begin{pspicture}(0,-5)(13,5)
\psgrid[subgriddiv=0,gridcolor=lightgray,griddots=10,gridlabels=0pt]%
\pstVerb{\parametres}%
  \psequadiff[% t,x
  tabname=tabtX,
  method=rk4,
  whichord=0,
  saveData,filename=LorenztX.dat,
  plotpoints=2501,algebraic]{0}{25}{\conditionsInitiales}{\Lorenz}
\listplotHSB[yunit=0.2,xunit=0.5,HSB=false]{tabtX aload pop}%
\uput[r](12.5,0){$t$}
\uput[u](0,5){$x$}
\end{pspicture}
```

3.3 Le tracé de XZ




```
\begin{pspicture}(-8,0)(8,12)
\psgrid[subgriddiv=0,gridcolor=lightgray,griddots=10,gridlabels=0pt]%
\pstVerb{\parametres}%
  \psequadiff[% x,z
  tabname=tabXZ,
  method=rk4,
  whichabs=0,
  whichord=2,
  saveData,filename=LorenzXZ.dat,
  plotpoints=2501,algebraic]{0}{25}{\conditionsInitiales}{\Lorenz}
\listplotHSB[linecolor=green,unit=0.25,HueBegin=.25,HueEnd=0.65]{tabXZ aload pop}%
\uput[r](8,0){$x$}
\uput[u](0,12){$z$}
\end{pspicture}
```

3.4 Le tracé de YZ




```
\begin{pspicture}(-8,0)(8,12)
\psgrid[subgriddiv=0,gridcolor=lightgray,griddots=10,gridlabels=0pt]%
\pstVerb{\parametres}%
  \psequadiff[% y,z
  tabname=tabYZ,
  method=rk4,
  whichabs=1,
  whichord=2,
  saveData,filename=LorenzYZ.dat,
  plotpoints=2501,algebraic]{0}{25}{\conditionsInitiales}{\Lorenz}
\listplotHSB[linecolor=magenta,unit=0.25,HueBegin=0.65,HueEnd=1]{tabYZ aload pop}%
\uput[r](8,0){$y$}
\uput[u](0,12){$z$}
\end{pspicture}
```

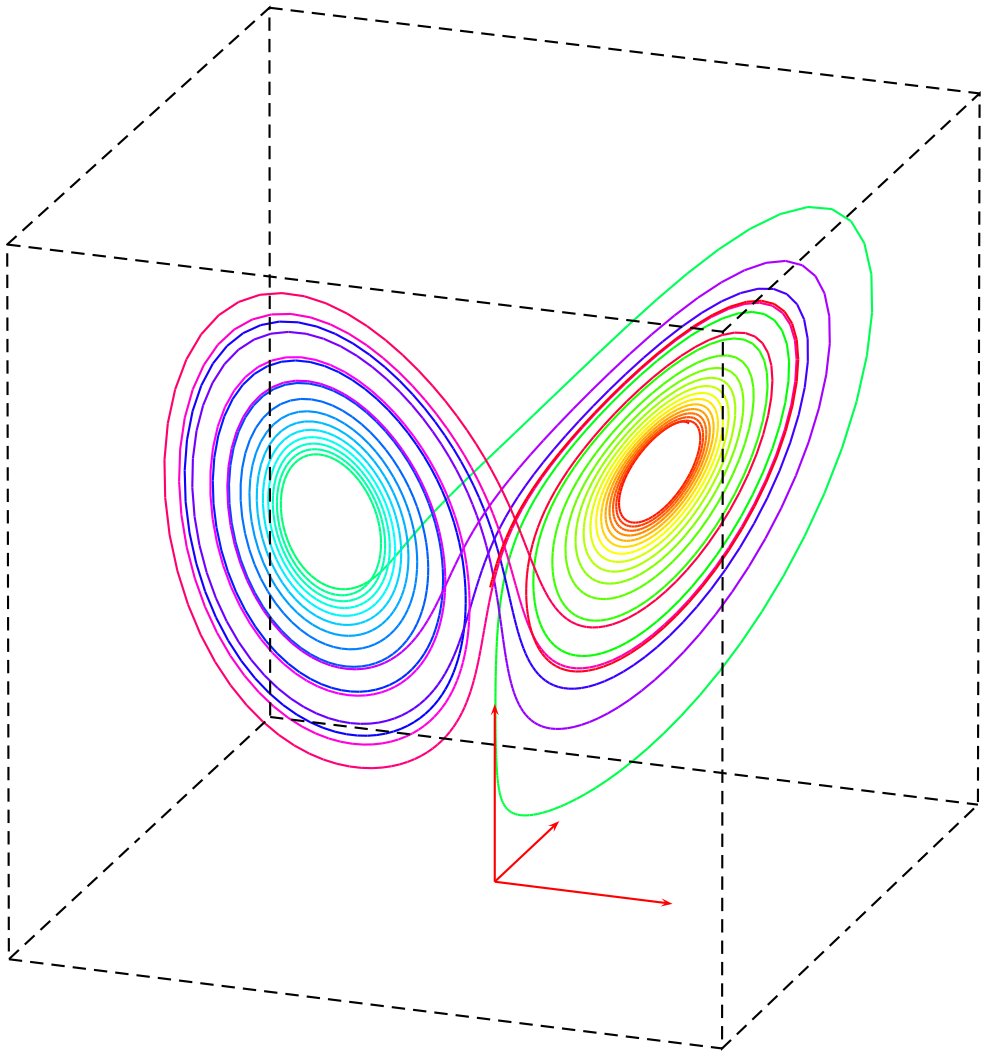
3.5 Le tracé du papillon de Lorenz en 3D

Le tracé en 3D est l'étape la plus délicate, puisqu'il faut réunir les fichiers (x,y), (x,z) et (y,z) en un seul fichier (x,y,z). Ce qu'on réalise ainsi :

```
\pstVerb{/XY [(LorenzXY.dat) run ] def
         /XZ [(LorenzXZ.dat) run ] def
         /YZ [(LorenzYZ.dat) run ] def
/tabXYZ [
    0 2 XY length 2 sub {/i exch def
    [ XY i get
      XY i 1 add get
      YZ i 1 add get
    ]
    } for
   ]def
```

```
/fichierpoints (XYZLorenz.dat) (w) file def
0 1 tabXYZ length 1 sub  {/i exch def
    tabXYZ i get aload pop
    /zi exch def
    /yi exch def
    /xi exch def
      fichierpoints xi 15 string cvs writestring
      fichierpoints 32 write %% espace
      fichierpoints yi 15 string cvs writestring
      fichierpoints 32 write %% espace
      fichierpoints zi 15 string cvs writestring
      fichierpoints 10 write %% CR
} for
fichierpoints closefile
```

Les dernières lignes permettent d'enregistrer, si on le souhaite, le fichier de points.



Le tracé s'effectue grâce à la commande \pnodeTroisD et ses diverses options.

```
\makeatletter
\pstVerb{
 /THETA \psk@TroisD@Theta def
 /PHI \psk@TroisD@Phi def
 /Dobs \psk@TroisD@Dobs def
 /DScreen \psk@TroisD@Ecran def
 /Sin1 THETA sin def
 /Sin2 PHI sin def
 /Cos1 THETA cos def
 /Cos2 PHI cos def
 /Cos1Sin2 Cos1 Sin2 mul def
 /Sin1Sin2 Sin1 Sin2 mul def
 /Cos1Cos2 Cos1 Cos2 mul def
 /Sin1Cos2 Sin1 Cos2 mul def
 /IIID {
 0 1 tabXYZ length 1 sub {/i exch def
 /TAB tabXYZ i get def
 /abscisse TAB 0 get def
 /ordonnee TAB 1 get def
 /cote TAB 2 get def
 \formules} for
 } def
 }%
 \makeatother
\listplotHSB[unit=1]{IIID}
\psset{unit=2.5,linestyle=dashed}
\pnodeTroisD(0,0,0){O}
\pnodeTroisD(0,0,5){Z}
\pnodeTroisD(5,0,0){X}
\pnodeTroisD(0,5,0){Y}
\pnodeTroisD(-10,-10,0){A}
\pnodeTroisD(-10,-10,20){B}
\pnodeTroisD(-10,10,20){C}
\pnodeTroisD(-10,10,0){D}
\pnodeTroisD(10,-10,0){E}
\pnodeTroisD(10,-10,20){F}
\pnodeTroisD(10,10,20){G}
\pnodeTroisD(10,10,0){H}
\pspolygon(A)(B)(C)(D)
\pspolygon(E)(F)(G)(H)
\psline(A)(E)
\psline(B)(F)
\psline(D)(H)
\psline(C)(G)
\psset{linestyle=solid,linecolor=red}
\psline{->}(O)(X)
\psline{->}(O)(Y)
\psline{->}(O)(Z)
```

4 Animations avec le package animate

En préparation.