Toupie pesante à point fixe

1. Équations du mouvement

On suppose que la pointe de la toupie (point O) reste fixe par rapport au référentiel inertiel. En pratique, l'expérience est réalisée avec une liaison de type Cardan, c'est-à-dire un gyroscope dont le centre de masse est non confondu avec le point fixe du mouvement.

La mise en équation en vue d'une intégration numérique est faite selon la méthode exposée dans équations du mouvement d'un solide.

Soit $R_s = (0x_sy_sz_s)$ le repère lié au solide coïncidant avec les axes propres du tenseur d'inertie au point O. L'axe Oz_s est l'axe de symétrie de la toupie, de moment d'inertie I_3. Les moments d'inertie I_1 et I_2 sont égaux.

Considérons tout d'abord le moment des forces de pesanteur au point O. Le centre de masse étant sur l'axe Oz_s, on pose $L = OG$. Soit le repère d'espace $R = (0xyz)$ (lié au référentiel) où Oz est la direction verticale. La matrice Q orthogonale définie l'orientation de la toupie. On a tout d'abord :

$$\overrightarrow{O}G = L(q_{13}\overrightarrow{e}_x + q_{23}\overrightarrow{e}_y + q_{33}\overrightarrow{e}_z) \tag{1}$$

Le moment du poids s'écrit :

$$\overrightarrow{N} = mgL(-q_{23}\overrightarrow{e}_x + q_{13}\overrightarrow{e}_y) \tag{2}$$

La matrice colonne correspondante est donc :

$$n = mgL \begin{pmatrix} -q_{23} \\ q_{13} \\ 0 \end{pmatrix} \tag{3}$$

Pour écrire les équations d'Euler, nous avons besoin des composantes du moment dans le repère R_s du solide. Ceci est obtenu par la transformation suivante :

$$n_s = Q^T n \tag{4}$$

Pour modéliser les frottements dus à la rotation de la toupie, on introduit un moment dans la direction de l'axe de symétrie de la toupie :

$$f_s = -k \begin{pmatrix} 0 \\ 0 \\ \omega_{s3} \end{pmatrix} \tag{5}$$

3 paramètres constants interviennent dans les équations d'Euler :

$$r = \frac{I_3}{I_1} = \frac{I_3}{I_2} \tag{6}$$

$$a = \omega_A^2 = \frac{mgL}{I_3} \tag{7}$$

$$b = \frac{k}{I_3} \tag{8}$$

Soit ω_0 la vitesse angulaire initiale de la toupie, c'est-à-dire la dérivée par rapport au temps de ψ à $t=0$ (à ne pas confondre avec la valeur initiale de ω_{s3}). Cette vitesse angulaire servira d'unité pour les variables ω_k. Pour une toupie rapide, on a :

$$\omega_A \ll \omega_0 \tag{9}$$

Écrivons les équations d'Euler avec Mathematica :

```
Q={{q11[t],q12[t],q13[t]},{q21[t],q22[t],q23[t]},{q31[t],q32[t],q33[t]}};
n={-q23[t],q13[t],0};
ns=Transpose[Q].n;
fs={0,0,omegaS3[t]};
euler={omegaS1'[t]==(1-r)*omegaS2[t]*omegaS3[t]+a*r*ns[[1]]-b*r*fs[[1]],
       omegaS2'[t]==(r-1)*omegaS3[t]*omegaS1[t]+a*r*ns[[2]]-b*r*fs[[2]],
       omegaS3'[t]==a*ns[[3]]-b*fs[[3]]}
```

$$\begin{pmatrix} \text{omegaS1}'(t) = ar(\text{q13}(t)\text{q21}(t) - \text{q11}(t)\text{q23}(t)) + (1-r)\text{omegaS2}(t)\text{omegaS3}(t) \\ \text{omegaS2}'(t) = ar(\text{q13}(t)\text{q22}(t) - \text{q12}(t)\text{q23}(t)) + (r-1)\text{omegaS1}(t)\text{omegaS3}(t) \\ \text{omegaS3}'(t) = -b\text{omegaS3}(t) \end{pmatrix}$$

Bien que la dernière équation soit directement intégrable, on la laissera dans le système pour plus de généralité.

Les équations différentielles vérifiées par les composantes de la transformation orthogonale :

```
OmegaS={{0,-omegaS3[t],omegaS2[t]},{omegaS3[t],0,-omegaS1[t]},{-omegaS2[t],omegaS1[t]
d = Q.OmegaS;
dQ = {q11'[t]==d[[1,1]],q12'[t]==d[[1,2]],q13'[t]==d[[1,3]],
      q21'[t]==d[[2,1]],q22'[t]==d[[2,2]],q23'[t]==d[[2,3]],
      q31'[t]==d[[3,1]],q32'[t]==d[[3,2]],q33'[t]==d[[3,3]]}
```

$$\begin{pmatrix} \text{q11}'(t) = \text{omegaS3}(t)\text{q12}(t) - \text{omegaS2}(t)\text{q13}(t) \\ \text{q12}'(t) = \text{omegaS1}(t)\text{q13}(t) - \text{omegaS3}(t)\text{q11}(t) \\ \text{q13}'(t) = \text{omegaS2}(t)\text{q11}(t) - \text{omegaS1}(t)\text{q12}(t) \\ \text{q21}'(t) = \text{omegaS3}(t)\text{q22}(t) - \text{omegaS2}(t)\text{q23}(t) \\ \text{q22}'(t) = \text{omegaS1}(t)\text{q23}(t) - \text{omegaS3}(t)\text{q21}(t) \\ \text{q23}'(t) = \text{omegaS2}(t)\text{q21}(t) - \text{omegaS1}(t)\text{q22}(t) \\ \text{q31}'(t) = \text{omegaS3}(t)\text{q32}(t) - \text{omegaS2}(t)\text{q33}(t) \\ \text{q32}'(t) = \text{omegaS1}(t)\text{q33}(t) - \text{omegaS3}(t)\text{q31}(t) \\ \text{q33}'(t) = \text{omegaS2}(t)\text{q31}(t) - \text{omegaS1}(t)\text{q32}(t) \end{pmatrix}$$

Les conditions initiales sont données à l'aide des angles d'Euler :

$$\psi(0) = 0,\ \psi'(0) = \omega_0 \tag{10}$$

$$\theta(0) = \theta_0,\ \theta'(0) = 0 \tag{11}$$

$$\phi(0) = 0,\ \phi'(0) = \omega_p \tag{12}$$

ω_p est la vitesse de précession initiale. Les conditions initiales doivent être exprimées pour les composantes du vecteur vitesse angulaire sur les axes de R_s et pour les éléments de la matrice Q :

```
Q0=RotationMatrix[phi,{0,0,1}].RotationMatrix[theta,{1,0,0}].RotationMatrix[psi,{0,0,
dQ0 = D[Q0/.{phi->phi[t],theta->theta[t],psi->psi[t]},t];
Qinit = Q0/.{phi->0,theta->theta0,psi->0};
dQinit = dQ0/.{phi'[t]->omegaP,psi'[t]->omega0,theta'[t]->0,phi[t]->0,
               theta[t]->theta0,psi[t]->0};
OmegaS = Transpose[Qinit].dQinit;
init = {omegaS1[0]==OmegaS[[3,2]],omegaS2[0]==OmegaS[[1,3]],omegaS3[0]==OmegaS[[2,1]]
        q11[0]==Qinit[[1,1]],q12[0]==Qinit[[1,2]],q13[0]==Qinit[[1,3]],
        q21[0]==Qinit[[2,1]],q22[0]==Qinit[[2,2]],q23[0]==Qinit[[2,3]],
        q31[0]==Qinit[[3,1]],q32[0]==Qinit[[3,2]],q33[0]==Qinit[[3,3]]}
```

$$\begin{pmatrix} \text{omegaS1}(0) = 0 \\ \text{omegaS2}(0) = \text{omegaP}\sin(\text{theta0}) \\ \text{omegaS3}(0) = \cos(\text{theta0})(\text{omega0}\cos(\text{theta0}) + \text{omegaP}) + \text{omega0}\sin^2(\text{theta0}) \\ \text{q11}(0) = 1 \\ \text{q12}(0) = 0 \\ \text{q13}(0) = 0 \\ \text{q21}(0) = 0 \\ \text{q22}(0) = \cos(\text{theta0}) \\ \text{q23}(0) = -\sin(\text{theta0}) \\ \text{q31}(0) = 0 \\ \text{q32}(0) = \sin(\text{theta0}) \\ \text{q33}(0) = \cos(\text{theta0}) \end{pmatrix}$$

Le système de 12 équations différentielles à intégrer est finalement :

```
equations=Join[euler,dQ]
```

2. Intégration avec Mathematica

2.a. Toupie conservative

Les frottements sont négligés et la vitesse de précession initiale est nulle. L'unité de temps est la période de rotation de la toupie.

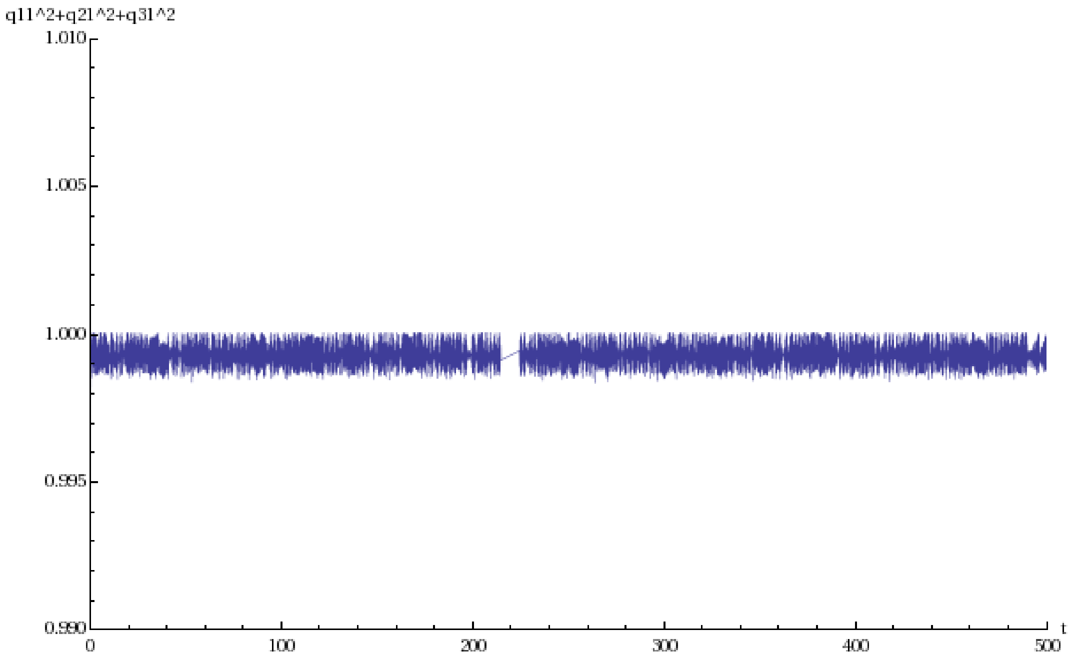
```
systeme=Join[equations,init]/.{r->0.2,a->N[2*Pi*0.1],b->0,omegaP->0,omega0->N[2*Pi],
                               theta0->N[30*Pi/180]}
```

On effectue l'intégration numérique :

```
tmax=500;
variables={omegaS1,omegaS2,omegaS3,q11,q12,q13,q21,q22,q23,q31,q32,q33};
solution = NDSolve[systeme,variables,{t,0,tmax},AccuracyGoal->5,
                   Method->'ImplicitRungeKutta',MaxSteps->Infinity]
```

Tout d'abord, on vérifie (partiellement) l'orthogonalité de Q :

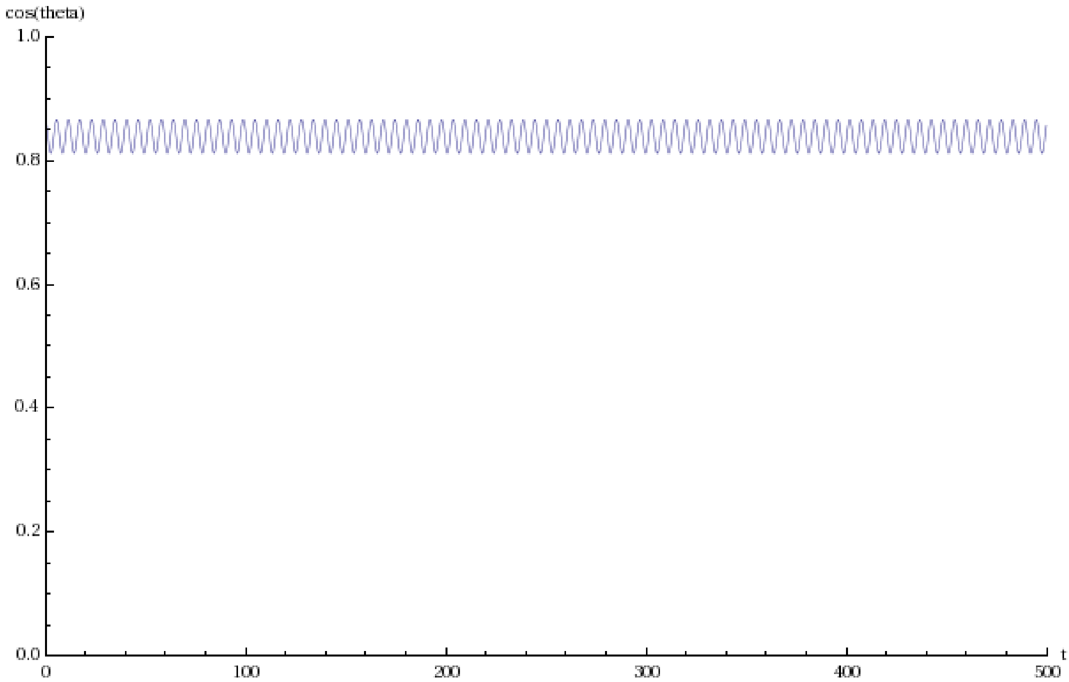
```
Plot[q11[t]*q11[t]+q21[t]*q21[t]+q31[t]*q31[t]/.solution,{t,0,tmax},
        PlotRange->{{0,tmax},{0.99,1.01}},AxesLabel->{'t','q11^2+q21^2+q31^2'}]
```



Les variations sont de l'ordre de 1 pour 1000.

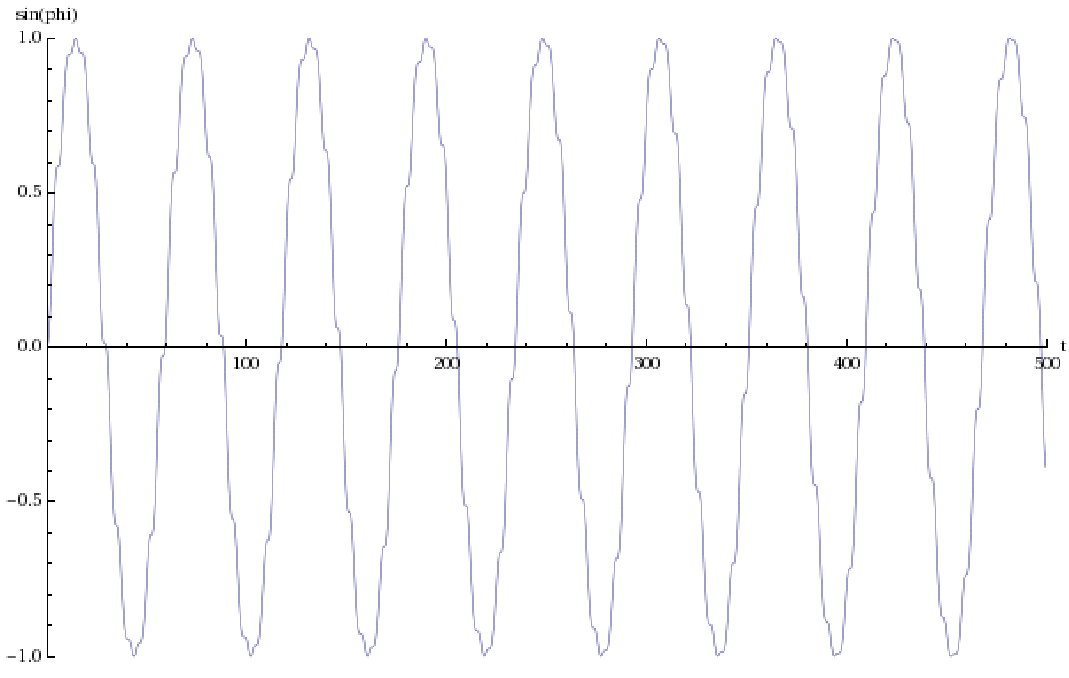
Traçons le cosinus l'angle $\theta(t)$:

```
Plot[q33[t]/.solution,{t,0,tmax},
        PlotRange->{{0,tmax},{0,1}},AxesLabel->{'t','cos(theta)'},PlotPoints->10^4]
```



On constate un mouvement de nutation. On trace le sinus de l'angle de précession $\varphi(t)$:

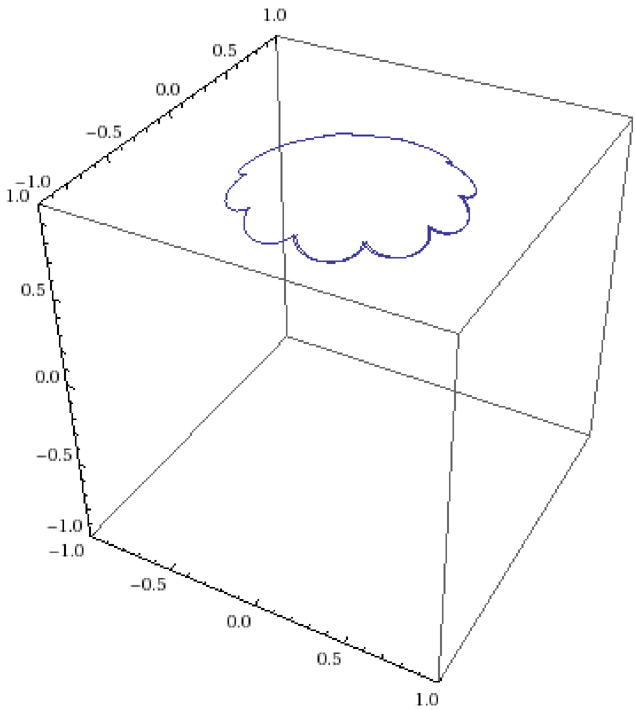
```
Plot[q13[t]/Sqrt[1-q33[t]*q33[t]]/.solution,{t,0,tmax},
        PlotRange->{{0,tmax},{-1,1}},AxesLabel->{'t','sin(phi)'},PlotPoints->10^4]
```



Cette courbe montre le mouvement de précession.

Représentation dans l'espace d'un point de l'axe de la toupie :

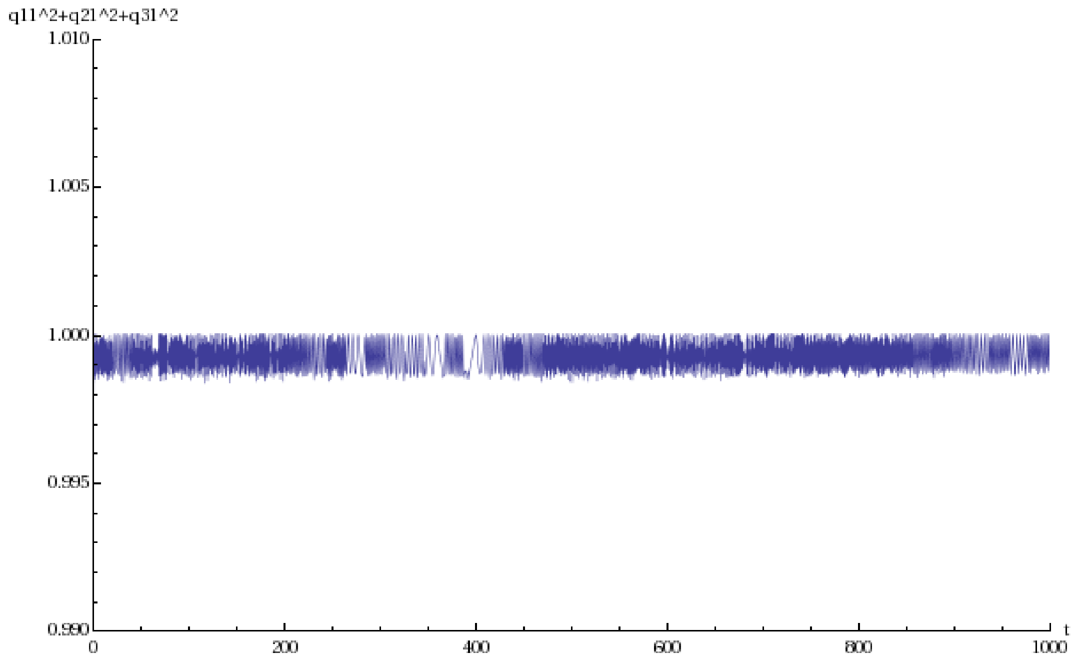
```
ParametricPlot3D[{q13[t],q23[t],q33[t]}/.solution,{t,0,100},PlotRange->{{-1,1},{-1,1}
```



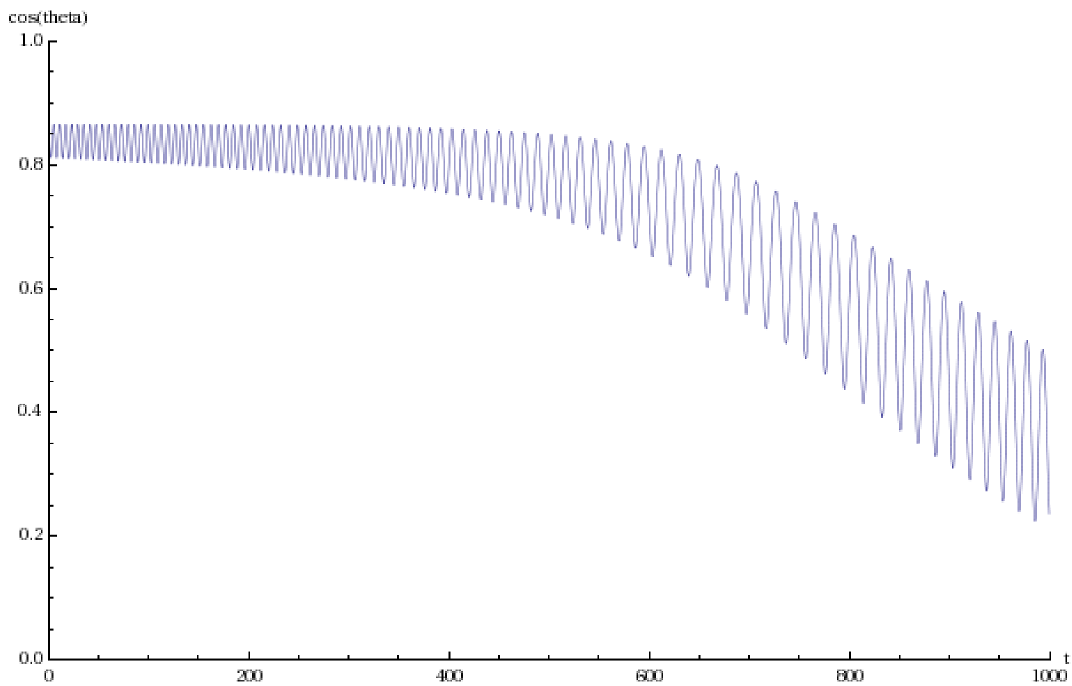
2.b. Toupie avec frottement

On reprend les mêmes paramètres avec en plus un léger couple de frottement sur l'axe de la toupie :

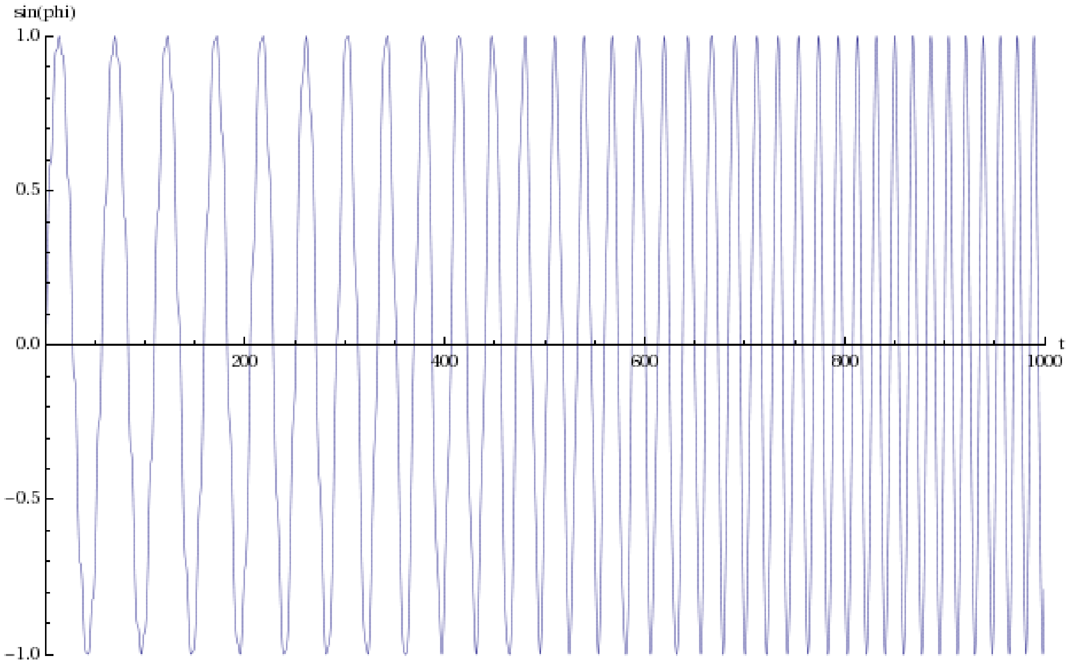
```
systeme=Join[equations,init]/.{r->0.2,a->N[2*Pi*0.1],b->0.001,omegaP->0,
                               omega0->N[2*Pi],theta0->N[30*Pi/180]}
tmax=1000;
solution = NDSolve[systeme,variables,{t,0,tmax},AccuracyGoal->5,
                   Method->'ImplicitRungeKutta',MaxSteps->Infinity]


Plot[q11[t]*q11[t]+q21[t]*q21[t]+q31[t]*q31[t]/.solution,{t,0,tmax},
        PlotRange->{{0,tmax},{0.99,1.01}},AxesLabel->{'t','q11^2+q21^2+q31^2'}]
```




```
Plot[q33[t]/.solution,{t,0,tmax},
       PlotRange->{{0,tmax},{0,1}},AxesLabel->{'t','cos(theta)'},PlotPoints->10^4]
```




```
Plot[q13[t]/Sqrt[1-q33[t]*q33[t]]/.solution,{t,0,tmax},
       PlotRange->{{0,tmax},{-1,1}},AxesLabel->{'t','sin(phi)'},PlotPoints->10^4]
```



3. Intégration avec Scilab

La fonction suivante effectue l'intégration numérique pour des paramètres et des conditions initiales donnés :

```
function [t,y]=solution(r,a,b,omegaP,omega0,theta0,tmax,te),
 c=cos(theta0),
 s=sin(theta0),
 function deriv=systeme(t,y),
  deriv(1)=(1-r)*y(2)*y(3)+a*r*(y(6)*y(7)-y(4)*y(9)), // omegaS1
  deriv(2)=(r-1)*y(1)*y(3)+a*r*(y(6)*y(8)-y(5)*y(9)), // omegaS2
  deriv(3)=-b*y(3), // omegaS3
  deriv(4)=y(3)*y(5)-y(2)*y(6), // q11
  deriv(5)=y(1)*y(6)-y(3)*y(4), // q12
  deriv(6)=y(2)*y(4)-y(1)*y(5), // q13
  deriv(7)=y(3)*y(8)-y(2)*y(9), // q21
  deriv(8)=y(1)*y(9)-y(3)*y(7), // q22
  deriv(9)=y(2)*y(7)-y(1)*y(8), // q23
  deriv(10)=y(3)*y(11)-y(2)*y(12), // q31
  deriv(11)=y(1)*y(12)-y(3)*y(10), // q32
  deriv(12)=y(2)*y(10)-y(1)*y(11), // q33
 endfunction
 init=[0;omegaP*s;omega0+omegaP*c;1;0;0;0;c;-s;0;s;c];
 t=[0:te:tmax];
    tolA=1d-7;
    tolR=1d-10;
 y=ode(init,0,t,tolR,tolA,systeme);
```

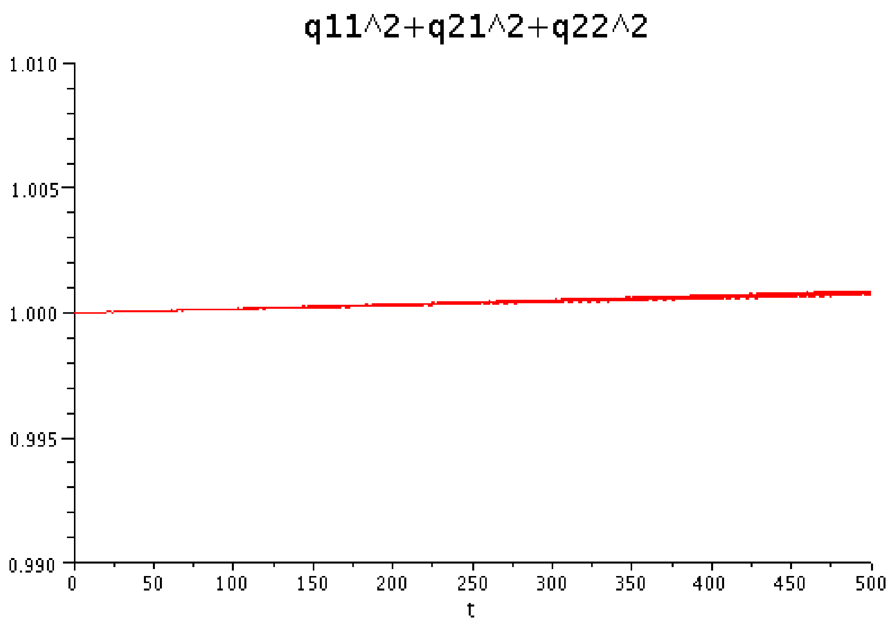
```
endfunction
```

La fonction suivante calcule les différentes quantités à tracer :

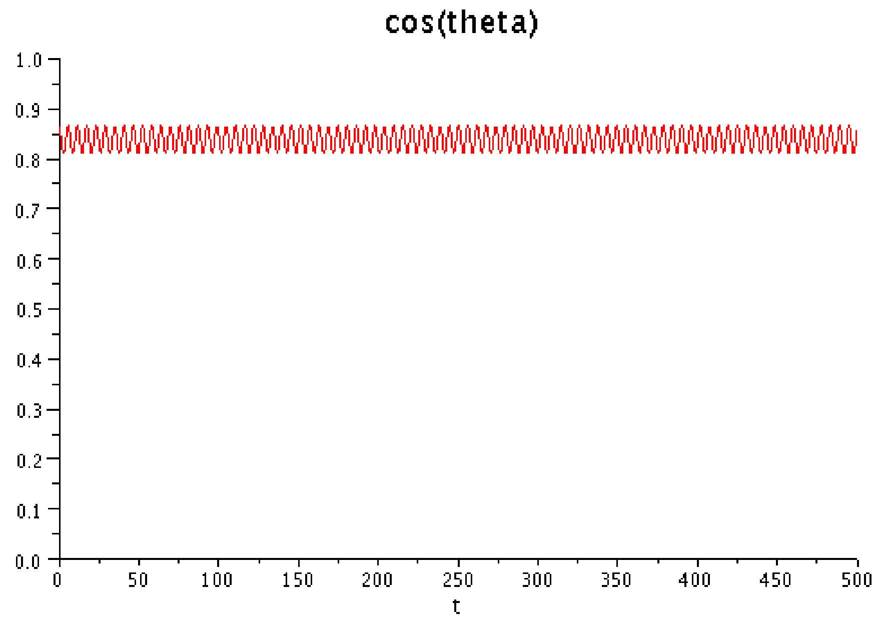
```
function [a1,a2,a3]=calcul(y)
 a1 = y(1,:);
 a2 = a1
 a3 = a1;
 s=size(a1);
 for i=1:s(2),
  a1(i)=y(4,i)*y(4,i)+y(7,i)*y(7,i)+y(10,i)*y(10,i),
  a2(i)=y(12,i),
  a3(i)=y(6,i)/sqrt(1-y(12,i)*y(12,i)),
 end;
endfunction
```

Voici par exemple la toupie sans frottements déjà considérée plus haut :

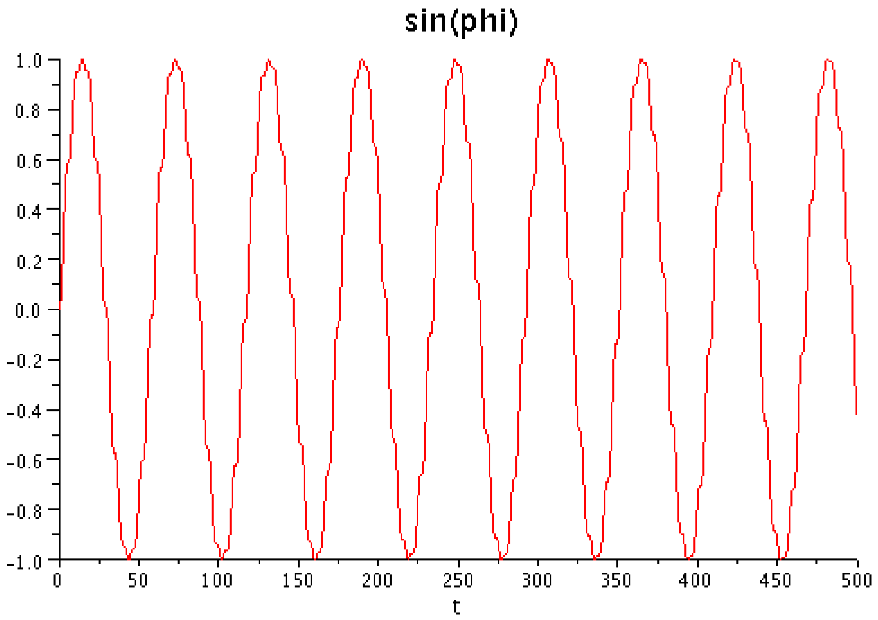
```
tmax=500;
[t,y]=solution(0.2,2*%pi*0.1,0,0,2*%pi,30*%pi/180,tmax,0.1);
[a1,a2,a3]=calcul(y);

plotH=scf();
plot2d(t,a1,style=5,rect=[0,0.99,tmax,1.01]);
xtitle('q11^2+q21^2+q22^2','t','');
```



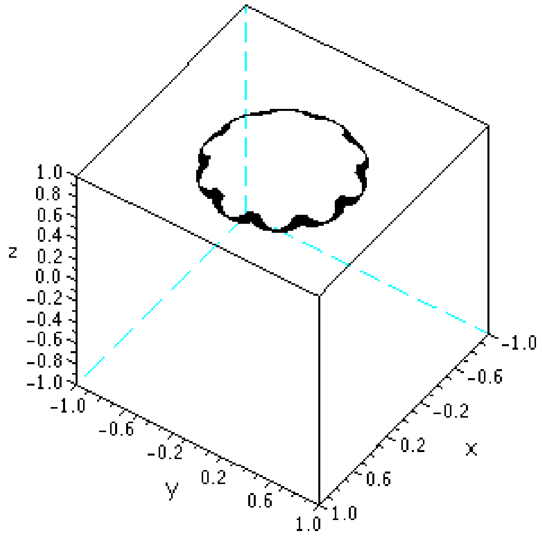
```
plotI=scf();
plot2d(t,a2,style=5,rect=[0,0,tmax,1]);
xtitle('cos(theta)','t','');
```




```
plotJ=scf();
plot2d(t,a3,style=5,rect=[0,-1,tmax,1]);
xtitle('sin(phi)','t','');
```




```
plotK=scf();
param3d(y(6,:),y(9,:),y(12,:),35,45,'x@y@z',[3,4],[-1,1,-1,1,-1,1]);
```


1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
z
-1.0
-0.6
-0.2
0.2
0.6
1.0
y
-1.0
-0.6
-0.2
0.2
0.6
1.0
x