Das Chaospendel-Skript

von Dr. rer. nat. Andreas Ernst

Version 05/2021

1 Einführung

Abbildung 1 zeigt eine Skizze des chaotischen Pendels. Am Aufhängungsbalken ist über ein Kugellager eine masselose drehbare Stange der Länge $2r$ mittig aufgehängt. An dessen Enden befinden sich wiederum Kugellager, an denen masselose Stangen der Länge l rotieren können. Am anderen Ende der Stangen befindet sich jeweils ein Gewicht der Masse m. Für die Herleitung der Bewegungsgleichungen werden die folgenden Vektoridentitäten verwendet:

$$(\vec{a}+\vec{b})^2 = |\vec{a}|^2 + 2\vec{a}\cdot\vec{b} + |\vec{b}|^2 \tag{1}$$
$$(\vec{a}\times\vec{b})\cdot(\vec{c}\times\vec{d}) = (\vec{a}\cdot\vec{c})(\vec{b}\cdot\vec{d}) - (\vec{b}\cdot\vec{c})(\vec{a}\cdot\vec{d}) \tag{2}$$
$$(\vec{a}\times\vec{b})^2 = |\vec{a}|^2|\vec{b}|^2 - (\vec{a}\cdot\vec{b})^2 \tag{3}$$

Die Gliederung der Broschüre ist wie folgt: In Abschnitt 2 wird ein Ausdruck für die kinetische Energie hergeleitet. In Abschnitt 3 wird ein Ausdruck für die potentielle Energie hergeleitet. In Abschnitt 4 werden die Bewegungsgleichungen mit Hilfe des Lagrangeformalismus aufgestellt. In Abschnitt 5 werden Hinweise zur numerischen Lösung des aus drei Bewegungsgleichungen bestehenden Systems gegeben. In Abschnitt 6 werden Schlussfolgerungen diskutiert.

2 Kinetische Energie

Die kinetische Energie der beiden Massen ist gegeben durch

$$T = \frac{1}{2}m\left(\vec{v}_1^2 + \vec{v}_2^2\right), \tag{4}$$

wobei $\vec{v}_1$ und $\vec{v}_2$ die Geschwindigkeiten der beiden Massenpunkte in kartesischen Koordinaten sind. Wir definieren die vektoriellen Winkelgeschwindigkeiten

$$\dot{\vec{\alpha}} = \begin{pmatrix} 0 \\ 0 \\ \dot{\alpha} \end{pmatrix}, \qquad \dot{\vec{\beta}}_{1/2} = \begin{pmatrix} 0 \\ 0 \\ \dot{\beta}_{1/2} \end{pmatrix}. \tag{5}$$

Dann gilt

$$\vec{v}_1^2 = \left(\dot{\vec{\alpha}}\times\vec{r} + \dot{\vec{\beta}}_1\times\vec{l}\right)^2 \tag{6}$$
$$= (\dot{\vec{\alpha}}\times\vec{r})^2 + 2(\dot{\vec{\alpha}}\times\vec{r})\cdot(\dot{\vec{\beta}}_1\times\vec{l}) + (\dot{\vec{\beta}}_1\times\vec{l})^2 \tag{7}$$

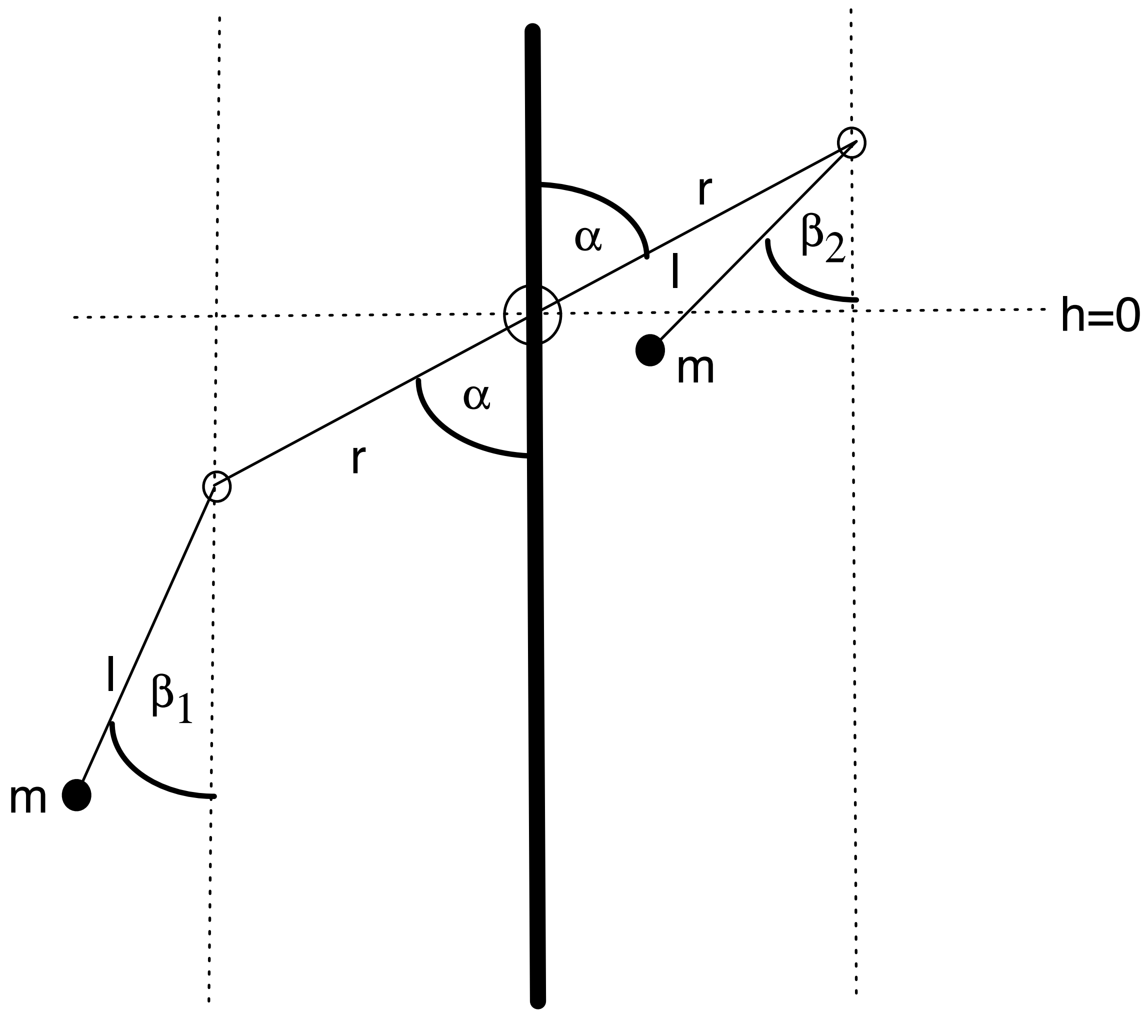

Abbildung 1: Skizze des chaotischen Pendels.

$$
\begin{aligned}
&= \dot{\alpha}^2 r^2 - \underbrace{(\dot{\vec{\alpha}} \cdot \vec{r})^2}_{=0} + \dot{\beta}_1^2 l^2 - \underbrace{(\dot{\vec{\beta}}_1 \cdot \vec{l})^2}_{=0} + 2 \underbrace{(\dot{\vec{\alpha}} \cdot \dot{\vec{\beta}}_1)}_{\dot{\alpha}\dot{\beta}_1} \underbrace{(\vec{r} \cdot \vec{l})}_{=rl\cos[\pi/2-(\alpha-\beta_1)])} - 2 \underbrace{(\vec{r} \cdot \dot{\vec{\beta}}_1)}_{=0} \underbrace{(\dot{\vec{\alpha}} \cdot \vec{l})}_{=0} && (8) \\
&= \dot{\alpha}^2 r^2 + \dot{\beta}_1^2 l^2 + 2rl\dot{\alpha}\dot{\beta}_1 \sin(\alpha - \beta_1) && (9) \\
\vec{v}_2^2 &= \dot{\alpha}^2 r^2 + \dot{\beta}_2^2 l^2 + 2rl\dot{\alpha}\dot{\beta}_2 \sin(\alpha - \beta_2) && (10)
\end{aligned}
$$

und wir erhalten die kinetische Energie

$$
T = m\dot{\alpha}^2 r^2 + \frac{1}{2} m(\dot{\beta}_1^2 + \dot{\beta}_2^2) l^2 + mrl\dot{\alpha} \left[\dot{\beta}_1 \sin(\alpha - \beta_1) + \dot{\beta}_2 \sin(\alpha - \beta 2)\right]. \qquad (11)
$$

3 Potentielle Energie

Wir können den Nullpunkt des Gravitationspotentials beliebig wählen. Am einfachsten ist es, die Höhe des großen Kugellagers in der Skizze als Nullpunkt des Potentials zu wählen.

Wenn sich alle Stangen in waagrechter Lage befinden, verschwindet die potentielle Energie identisch. Die potentielle Energie der beiden Massen ist gegeben durch

$$V = mg(h_1 + h_2) \tag{12}$$

wobei g die Erdbeschleunigung ist und h_1 und h_2 die beiden Höhen über (oder unter) dem Nullpunkt des Potentials ($h = 0$). Für die Höhen gilt

$$h_1 = -r\cos\alpha - l\cos\beta_1 \tag{13}$$
$$h_2 = r\cos\alpha - l\cos\beta_2. \tag{14}$$

Somit hängt die potentielle Energie nicht von α ab und ist gegeben durch

$$V = -mgl(\cos\beta_1 + \cos\beta_2) \tag{15}$$

4 Bewegungsgleichungen

Die spezifische Lagrangefunktion des Systems ist folglich gegeben durch

$$\begin{aligned} L &= (T - V)/m & (16)\\ &= \dot\alpha^2 r^2 + \frac{1}{2}(\dot\beta_1^2 + \dot\beta_2^2)l^2 + rl\dot\alpha\left[\dot\beta_1\sin(\alpha-\beta_1) + \dot\beta_2\sin(\alpha-\beta_2)\right] \\ &\quad + gl(\cos\beta_1 + \cos\beta_2) & (17) \end{aligned}$$

Die aus der Variationsrechnung folgende Lagrangegleichung für die generalisierte Koordinate q lautet allgemein

$$\frac{d}{dt}\frac{\partial L}{\partial \dot q} - \frac{\partial L}{\partial q} = 0. \tag{18}$$

Einsetzen von (17) und den drei Winkeln α, β_1 und β_2 liefert drei Bewegungsgleichungen 2. Ordnung:

$$\ddot\alpha + \frac{l}{2r}\left[\ddot\beta_1\sin(\alpha-\beta_1) - \dot\beta_1^2\cos(\alpha-\beta_1) + \ddot\beta_2\sin(\alpha-\beta_2) - \dot\beta_2^2\cos(\alpha-\beta_2)\right] = 0 \tag{19}$$
$$\ddot\beta_1 + \frac{r}{l}\left[\ddot\alpha\sin(\alpha-\beta_1) + \dot\alpha\dot\beta_1\cos(\alpha-\beta_1)\right] + gl\sin\beta_1 = 0 \tag{20}$$
$$\ddot\beta_2 + \frac{r}{l}\left[\ddot\alpha\sin(\alpha-\beta_2) + \dot\alpha\dot\beta_2\cos(\alpha-\beta_2)\right] + gl\sin\beta_2 = 0. \tag{21}$$

5 Numerische Lösung

Für eine numerische Lösung der Bewegungsgleichungen (19)-(21) kann das in den nachfolgenden Listings 1-5 angegebene Runge-Kutta-Integrationsprogramm 8. Ordnung verwendet werden.

```
g++ rk8p.C -o rk8p
./rk8p 0.0001 100.0 0.01 > out
```

Nach Ausführen des Integrationsprogramms kann die Lösung mit zwei in den Listings 6 und 7 angegebenen `gnuplot`-Skripten grafisch visualisiert werden. Hierzu muss zunächst das Programm `gnuplot` auf dem Rechner installiert werden.

```
gnuplot
gnuplot> load "anim.gnu"
```

6 Schlussfolgerungen

Aufgrund der Nichtlinearität der Bewegungsgleichungen kann es zu überraschenden Effekten kommen. Im Vergleich zum einfachen Fadenpendel kann zwischen den drei Rotationsfreiheitsgraden des chaotischen Pendels Energie übertragen werden, wobei die Winkelgeschwindigkeit eines einzelnen kugelgelagerten Pendels massiv wachsen kann. Dies geschieht hochgradig in der Nähe von Resonanzen. Je näher sich das System an einer Resonanz befindet, umso mehr Energie kann von einem auf ein anderes der kugelgelagerten Pendel übertragen werden. Mathematisch können die Resonanzen mit Hilfe von Poincaréschnitten im Phasenraum detektiert werden.

```
//------------------------------------------------------------------
// rk8p.C
// Code: Runge-Kutta-Integrator 8. Ordnung
//       mit konstanten Zeitschritten fuer die Bewegungs-
//       gleichungen des chaotischen Pendels
// Autor: Andreas Ernst
//------------------------------------------------------------------

#include <iostream>
#include <cmath>
#include <cstdlib>
using namespace std;

const double A1 = 10.0;
const double B1 = 5.0;
const double B2 = 5.0;
const double a1 = 10.0;
const double b1 = 5.0;
const double b2 = 5.0;
const double l = 10.0;
const double r = 10.0;
const double g = 9.81;

// Werte Vektorfunktion aus

void F(double m[], double in[][6], double out[][6], int n) {
  int i, j, k;
  double tmp0, tmp1, tmp2;

// Berechne Beschleunigungen

  double (*a)[3] = new double[n][3];
  for (i=0; i<n ; i++) {
     tmp0 = a[i][0];
     tmp1 = a[i][1];
     tmp2 = a[i][2];
     a[i][0] = -(l/2/r)*(tmp1*sin(a1-b1) - B1*B1*cos(a1-b1)
                 + tmp2*sin(a1-b2) - b2*b2*cos(a1-b2));
     a[i][1] = -(r/l)*(tmp0*sin(a1-b1) + A1*B1*cos(a1-b1))
                  - g*l*sin(b1);
     a[i][2] = -(r/l)*(tmp0*sin(a1-b2) + A1*B2*cos(a1-b2))
                  - g*l*sin(b2);
  }
```

Listing 1: C++ code `rk8p.C` des chaotischen Pendels

```
// Return output vector

  for (i=0; i<n ; i++) {
    for (k=0; k<3; k++) {
      out[i][k] = in[i][k+3];
      out[i][k+3] = a[i][k];
    }
  }

}

int main(int argc, char *argv[]) {

// Lese Anfangsbedingungen von stdin; Deklaration der Variablen

  int i, j, k;

  if (argc < 3 ) {
    cout << "Usage: rk8p <dt> <t_end> <dt_opt>" << endl;
    return 0;
  }

  double dt = atof(argv[1]); // time step
  double t_end = atof(argv[2]);
  double dt_opt = atof(argv[3]);
  double t_opt = 0.001;

  const int n = 1;

  double * m = new double[n];
  double (* Y)[6] = new double[n][6];
  double (* k0)[6] = new double[n][6];
  double (* k1)[6] = new double[n][6];
  double (* k2)[6] = new double[n][6];
  double (* k3)[6] = new double[n][6];
  double (* k4)[6] = new double[n][6];
  double (* k5)[6] = new double[n][6];
  double (* k6)[6] = new double[n][6];
  double (* k7)[6] = new double[n][6];
  double (* help)[6] = new double[n][6];

  double dt_out = dt_opt;
  double t_out = dt_out;

  Y[0][0] = 1.0;
  Y[0][1] = 1.0;
  Y[0][2] = 1.0;
  Y[0][3] = 0.5;
  Y[0][4] = 0.5;
  Y[0][5] = -0.5;
```


Listing 2: C++ code `rk8p.C` des chaotischen Pendels

```
// Integrationsschleife

  for (double t = 0; t < t_end; t += dt) {

    F(m, Y, k0, n);

    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      help[i][k] = Y[i][k] + dt*k0[i][k]/6;
    }

    F(m, help, k1, n);

    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      help[i][k] = Y[i][k] + dt*(k0[i][k]*4/75 + k1[i][k]*16/75);
    }

    F(m, help, k2, n);

    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      help[i][k] = Y[i][k] + dt*(k0[i][k]*5/6 - k1[i][k]*8/3
                             + k2[i][k]*5/2);
    }

    F(m, help, k3, n);

    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      help[i][k] = Y[i][k] + dt*(-k0[i][k]*8/5 + k1[i][k]*144/25
                             -k2[i][k]*4
+ k3[i][k]*16/25);
    }

    F(m, help, k4, n);

    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      help[i][k] = Y[i][k] + dt*(k0[i][k]*361/320 - k1[i][k]*18/5
                             +k2[i][k]*407/128 - k3[i][k]*11/80
                             +k4[i][k]*55/128);
    }

    F(m, help, k5, n);
```
Listing 3: C++ code `rk8p.C` des chaotischen Pendels

```
    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      help[i][k] = Y[i][k] + dt*(-k0[i][k]*11/640
                                           +k2[i][k]*11/256
                                           -k3[i][k]*11/160
                                           +k4[i][k]*11/256);
    }

    F(m, help, k6, n);

    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      help[i][k] = Y[i][k] + dt*(k0[i][k]*93/640 - k1[i][k]*18/5
                                 +k2[i][k]*803/256
                                 - k3[i][k]*11/160
                                 +k4[i][k]*99/256 + k6[i][k]);
    }

    F(m, help, k7, n);

// Integrationsschritt

    for (i=0; i<n; i++) for (k=0; k<6; k++) {
      Y[i][k] += dt*(k0[i][k]*7/1408 + k2[i][k]*1125/2816
                       +k3[i][k]*9/32 +k4[i][k]*125/768
                       +k6[i][k]*5/66 + k7[i][k]*5/66);
    }

// Datenausgabe

    if (t >= t_out) {
      cout << "DATA: ";
      cout << t << " ";
      for (int i = 0; i < n; i++) {
        cout << -r*sin(Y[0][0]) << " "
             <<  r*sin(Y[0][0]) << " "
             << -r*cos(Y[0][0]) << " "
             <<  r*cos(Y[0][0]) << " "
             << -r*sin(Y[0][0]) -l*sin(Y[0][1]) << " "
             <<  r*sin(Y[0][0]) -l*sin(Y[0][2]) << " "
             << -r*cos(Y[0][0]) -l*cos(Y[0][1]) << " "
             <<  r*cos(Y[0][0]) -l*cos(Y[0][2]);
      }
      cout << endl;
      t_out += dt_out;
    }
  }
```
Listing 4: C++ code `rk8p.C` des chaotischen Pendels

```cpp
  delete[] m;
   delete[] Y;
   delete[] k0;
   delete[] k1;
   delete[] k2;
   delete[] k3;
   delete[] k4;
   delete[] k5;
   delete[] k6;
   delete[] k7;
   delete[] help;

   return 0;

//-----------------------------------------------------------------------------
}
```

Listing 5: C++ code `rk8p.C` des chaotischen Pendels

```gnuplot
set nokey
set pointsize 3
set xrange [-30:30]
set yrange [-30:30]
n=1
c=0.001
load 'loop.gnu'
```

Listing 6: Gnuplot code `anim.gnu` für die Visualisierung des chaotischen Pendels

```gnuplot
plot 'out' ev ::n::n u 3:5 lt 3 pt 7,\
     'out' ev ::n::n u 4:6 lt 4 pt 7,\
     'out' ev ::n::n u 7:9 lt 3 pt 7,\
     'out' ev ::n::n u 8:10 lt 4 pt 7,\
     'out' ev ::n::n u 3:5:($4-$3):($6-$5) with vectors lt -1,\
     'out' ev ::n::n u 3:5:($7-$3):($9-$5) with vectors lt -1,\
     'out' ev ::n::n u 4:6:($8-$4):($10-$6) with vectors lt -1
print "Step ",n+1
pause c
n=n+1;
reread
```

Listing 7: Gnuplot code `loop.gnu` für die Visualisierung des chaotischen Pendels