5.4 Bewegung unter Zentralkräften (IV)

5.4.1 Das N-Körperproblem

In diesem Kapitel untersuchen wir das Problem von drei und mehr Körpern, die untereinander unter Gravitationswechselwirkung stehen.

1894 hat Poincaré theoretisch gezeigt, dass es unmöglich ist, die Bewegungsgleichungen allgemein zu lösen, wenn die Zahl N der Körper >=3 ist. Diese Aussage hat dann den Weg geöffnet für die Suche nach verschiedenen **numerischen** Verfahren. Das Problem besteht mathematisch in der Lösung von N vektoriellen Differenzialgleichungen zweiter Ordnung unter Berücksichtigung der Tatsache, dass alle miteinander gekoppelt sind.

Eine numerische Lösung des Problems ist mit praktisch beliebiger Genauigkeit möglich. (Dennoch können analytische Lösungen gefunden werden, wenn man das Problem *einschränkt*, indem man vereinfachende Annahmen macht bezüglich der Art der Bewegung und der Wechselwirkungen.)

Unsere Untersuchungen werden rein numerisch sein –außer einigen theoretischen Überlegungen im Falle des *eingeschränkten Problems*, vgl. 5.4.5.

Der numerische Zugang zum N-Körperproblem liefert uns schnell graphische Lösungen von komplizierten physikalischen Problemen. Man muss nur einen Blick werfen auf die Abbildungen dieses Kapitels. In der folgenden Abbildung sehen wir vier Körper, die unter Gravitations-WW stehen.

Unser Problem lautet: Welches sind die Positionen und Geschwindigkeiten der 4 Körper zur Zeit t, wenn diese Werte zur Zeit t = 0 bekannt sind?

Die Gravitationskraft, die der Körper k auf den Körper i ausübt, ist gegeben durch Gleichung (1)

$$\vec{F}_{ik} = -G\frac{m_i m_k}{r_{ik}^3}\vec{r}_{ik} \quad , \quad (1)$$

worin $\mathbf{r}_{ik} = \mathbf{r}_i - \mathbf{r}_k = (x_i - x_k)\mathbf{i}_o + (y_i - y_k)\mathbf{j}_o + (z_i - z_k)\mathbf{k}_o$

und $r_{ik} = ((x_i - x_k)^2 + (y_i - y_k)^2 + (z_i - z_k)^2)^{1/2}$ sind.

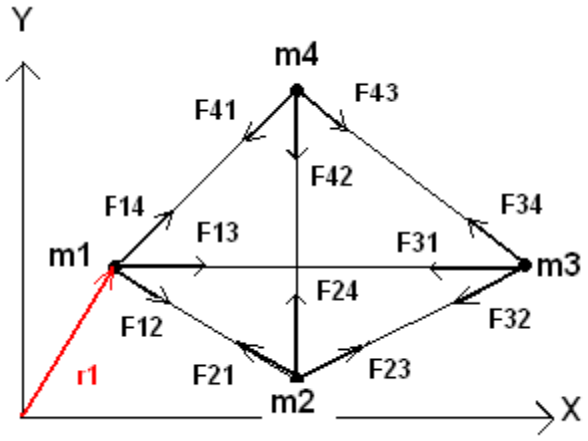

Fig.5.4-1

Die Gesamtkraft auf den i-ten Körper ist gegeben durch

$$\bar{F}_i = \sum_{k=1,k\neq i}^{N} \bar{F}_{ik} = -Gm_i \sum_{k=1,k\neq i}^{N} \frac{m_k}{r_{ik}^3}\bar{r}_{ik} \qquad (2)$$

Die Doppelsumme können wir durch einfache Summen ersetzen, z.B.:

$$m_i \frac{dv_{ix}}{dt} = \sum_{j=1}^{N} - \frac{Gm_i m_j (x_i - x_j)}{r_{ij}^3}$$

und zwei weitere für dv_{iy}/dt und dv_{iz}/dt. Für j = i gibt es keine Summe. Im folgenden Programm verwenden wir nur die beiden Koordinaten x und y.

Unsere Aufgabe besteht darin, zunächst Ort und Geschwindigkeit der 4 Körper am Ende des Intervalls h:= Δt zu berechnen -unter der Annahme, dass diese Werte am Anfang des Intervalls bekannt sind.

Wir beginnen mit der Berechnung der Geschwindigkeit v(t+h). (Zur Vereinfachung notieren wir nur die Gleichungen für die x-Koordinate, aber ohne den Index x zu schreiben, also $v_x(t+h) := v(t+h)$.)

Für den ersten Körper haben wir

$$v_1(t+h) = v_1(t) + a_1(t)h = v_1(t) + F_1\, h/m_1 \qquad (3)$$

Wir erinnern uns, dass $a(t) \approx (v(t+h) - v(t))/h$.

Aus Fig. 5.4-1 lesen wir ab, dass für die x-Koord. gilt $F_1 = F_{12} + F_{13} + F_{14}$ -und Gl. (39) muss ausführlicher folgendermaßen heißen:

$$v_1(t+h) = v_1(t) + F_{12}\, h/m_1 + F_{13}\, h/m_1 + F_{14}\, h/m_1 \qquad (4)$$

Für die numerische Rechnung zerlegen wir diese Gleichung wie folgt in drei Gleichungen der gleichen Gestalt:

$$v_1(t+h) = v_1^{(2)} + F_{14}\, h/m_{1,}$$

$$v_1^{(2)} = v_1^{(1)} + F_{13}\, h/m_{1,}$$

$$v_1^{(1)} = v_1 + F_{12}\, h/m_{1.}$$

Wir haben dann dreimal eine Gleichung von folgender Form auszuwerten:

$$v^{\,neu} = v^{\,alt} + \text{Korrekturterm} \qquad (5)$$

Wenn wir $R3:=R^3$ und $R:=(x^2 + y^2)^{1/2}$ setzen, können wir auf den ersten Körper das folgende Schema anwenden:

1➔ 2 $\quad x = x(1) - x(2)$
$\quad F = -GM(1)M(2)\, x/R3$
$\quad v(1) = v(1) + F{\cdot}h/M(1) := v_1^{(1)}$

1➔ 3 $\quad x = x(1) - x(3)$
$\quad F = -GM(1)M(3)\, x/R3$
$\quad v(1) = v(1) + F{\cdot}h/M(1) := v_1^{(2)}$

1➔ 4 $\quad x = x(1) - x(4)$
$\quad F = -GM(1)M(4)\, x/R3$
$\quad v(1) = v(1) + F{\cdot}h/M(1) := v1(t + h)$

Für die Körper 2, 3 und 4 haben wir das gleiche Schema anzusetzen. Beim zweiten Körper werden wir berücksichtigen, dass $\mathbf{F}_{12} = -\mathbf{F}_{21}$. Das bedeutet, dass wir in der Wechselwirkung 1➔ 2 die Gleichung $v(2) = v(2) - F{\cdot}h/M(2)$ zu berücksichtigen haben. Ebenso berücksichtigen wir $v(3) = v(3) - F{\cdot}h/M(3)$ bei der WW 1 ➔ 3. In der letzten Gleichung steht F für $F_{13,}$ usw.

Im Fall des vierten Körpers haben wir keine Kraft zu berechnen, denn alle WW wurden schon berücksichtigt.

Mit den folgenden Programmzeilen können wir alle Rechnungen ausführen:

```
for i from 1 to n-1 do
for k from i+1 to n do

    x:=x(i)-x(k):
    y:=y(i)-y(k):
    r:=sqrt(x^2 + y^2):
     r3:= r^3:
      F:= -g·m(i)·m(k)·x/r3:
       v(i):= v(i) + F·h/m(i):
        v(k):= v(k) - F·h/m(k):
     end_for:
     end_for:
```

Mit den folgenden Zeilen berechnen wir die Koordinaten x und y (z gibt es nicht)

```
for j from 1 to n do

x(j) := x(j) + v(j)·h:
y(j) := y(j) + v(j)·h:
        end_for:
```

Man kann die Genauigkeit des Verfahrens erhöhen, wenn man v im 1. Schritt im Mittelpunkt des Zeitintervalls berechnet: v(t + h/2) (*Feynman*-Methode). Daher führen wir zu Beginn die Rechnungen mit h/2 aus. In den weiteren Rechnungen benutzen wir dann $\Delta t = h$.

Wie ich in 5.2.1 erklärte, führen wir die Rechnungen mit einheitenfreien Variablen aus (reduzierte Variablen). Mithilfe des 1. Programms berechnen wir die Bewegung von 3 Körpern gleicher Massen, die sich am Anfang in den Eckpunkten eines gleichseitigen Dreiecks befinden: A(1|0), B(0|1,732), C(-1|0). Wir geben ihnen zum Start die im Programm angegebenen Geschwindigkeiten und werden beobachten, dass die Körper immerfort denselben Kreis beschreiben.

Wenn wir andere Anfangsgeschwindigkeiten wählen, ergeben sich recht exotische Bahnen.

Mit den Anfangsgeschwindigkeiten (0.4,0.6), (0.3,-0.6) und -0.7,0) ergeben sich völlig chaotische Trajektorien. Dies alles zeigt uns, dass es keineswegs einfach ist, ein harmonisches Universum zu erschaffen.

Programm 1:

```
reset()://n-Koerper mit Feynman-Methode
N_koerper:=proc(dt,schritte)// dt = Zeitintervall
begin
m:=[1,1,1]://Liste der Massen
X:=[1,-1,0]:Y:=[0,0,1.732]:// Koordinaten
U:=[.35355,.35355,-.7071]://x-Koord. der Geschw.
V:=[.61237,-.61237,0]:
n:=nops(m)://Zahl der Elemente in der Liste m
for p from 1 to n do
x[p]:=X[p]://Auszug der Elemente aus den Listen
y[p]:=Y[p]:
u[p]:=U[p]:
v[p]:=V[p]:
end_for:
x1[0]:=X[1]:y1[0]:=Y[1]://Fuer den Graphen
x2[0]:=X[2]:y2[0]:=Y[2]:
x3[0]:=X[3]:y3[0]:=Y[3]:
g:=1://bei red. Koord. sonst g = 6.67.10^(-11)
h:=dt/2://Anfangsschritt
for r from 0 to schritte do// Zahl der zu zeichnenden Punkte
for i from 1 to n-1 do//Geschwindigkeiten fuer h:=dt/2
for k from i+1 to n do
c:=g*m[i]:
xx:=x[i]-x[k]:
yy:=y[i]-y[k]:
r3:=(xx*xx+yy*yy)^(-3/2):
c1:=c*m[k]*r3:
fx:=-c1*xx:
fy:=-c1*yy:
u[i]:=u[i]+fx*h/m[i]:u[k]:=u[k]-fx*h/m[k]:
v[i]:=v[i]+fy*h/m[i]:v[k]:=v[k]-fy*h/m[k]:
end_for:
end_for:

h:=dt:
for j from 1 to n do// x,y-Koordinaten
x[j]:=x[j]+u[j]*h:
y[j]:=y[j]+v[j]*h:
end_for:
//Punkte des Graphen
x1[r]:=x[1];y1[r]:=y[1]:
x2[r]:=x[2]:y2[r]:=y[2]:
```

```
x3[r]:=x[3]:y3[r]:=y[3]:
x1[0]:=X[1]:y1[0]:=Y[1]:
x2[0]:=X[2]:y2[0]:=Y[2]:
x3[0]:=X[3]:y3[0]:=Y[3]:
end_for:
plot(plot::Group2d(
plot::Point2d(x1[r],y1[r],Color=RGB::Red)$ r=0..schritte,
plot::Point2d(x2[r],y2[r],Color=RGB::Green)$ r=0..schritte,
plot::Point2d(x3[r],y3[r],Color=RGB::Blue)$ r=0..schritte,
 ViewingBox=[-1.5..2,-1..2],Scaling=Constrained )): end_proc:
 N_koerper(0.1,30)
```

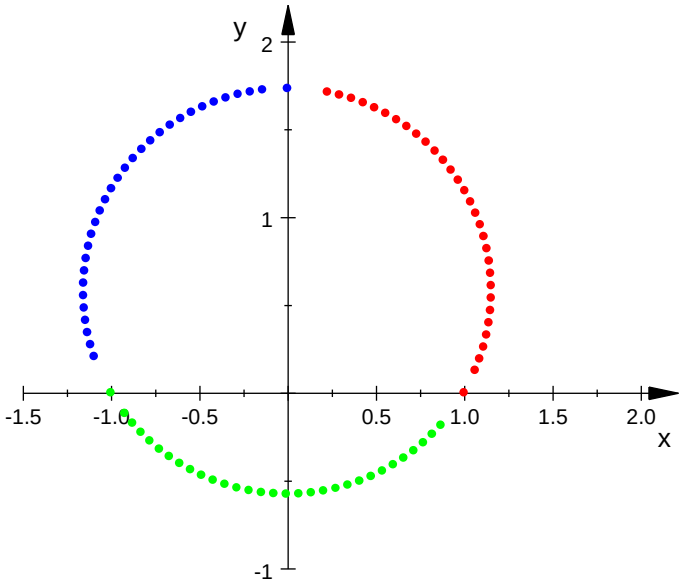

Fig.5.4-2

Im 7. Kapitel werden wir uns ausführlicher mit der Feynman-Methode beschäftigen. Wir haben sie hier schon eingesetzt, weil es nicht schwierig ist, sie zu verstehen. Die Methode **`numeric::odesolve2`** von MuPAD funktioniert sehr gut, viel besser als die Feynman-Methode, aber wir wissen nicht, was sich hinter der MUPAD- Methode genau verbirgt.

Im 3. Programm stecken alle Informationen des vorigen Programms, aber die Gleichungen werden sorgfältiger notiert. Wir können diese ausführliche Liste leicht mithilfe des 2. Programms erzeugen.

Programm 2:

```
reset():Beschleunigungen bei N-Koerpern
//r[i][j]:=sqrt((x[i]-x[j])^2+(y[i]-y[j])^2):Entfernungen
ax[1]:=_plus(_plus(-m[j]*(x[i]-x[j])/r[i][j]^3 $i=1..1)$
  j=1..4);
ay[1]:=_plus(_plus(-m[j]*(y[i]-y[j])/r[i][j]^3 $i=1..1)$
  j=1..4);
//Um ax[2],ay[2] etc.zu berechnen, muss man "$i=2..2",
  etc.setzen
//siehe 4.7.2
ax[2]:=_plus(_plus(-m[j]*(x[i]-x[j])/r[i][j]^3 $i=2..2)$
  j=1..4);
ay[2]:=_plus(_plus(-m[j]*(y[i]-y[j])/r[i][j]^3 $i=2..2)$
  j=1..4);
```

Ausgabe:

$$-\frac{(x[1]-x[2])\,m[2]}{r[1][2]^3}-\frac{(x[1]-x[3])\,m[3]}{r[1][3]^3}-\frac{(x[1]-x[4])\,m[4]}{r[1][4]^3}$$

$$-\frac{(y[1]-y[2])\,m[2]}{r[1][2]^3}-\frac{(y[1]-y[3])\,m[3]}{r[1][3]^3}-\frac{(y[1]-y[4])\,m[4]}{r[1][4]^3}$$

$$\frac{(x[1]-x[2])\,m[1]}{r[2][1]^3}-\frac{(x[2]-x[3])\,m[3]}{r[2][3]^3}-\frac{(x[2]-x[4])\,m[4]}{r[2][4]^3}$$

$$\frac{(y[1]-y[2])\,m[1]}{r[2][1]^3}-\frac{(y[2]-y[3])\,m[3]}{r[2][3]^3}-\frac{(y[2]-y[4])\,m[4]}{r[2][4]^3}$$

Wir können nicht einfach das Programm der beiden Tennisbälle aus 2.5.1 verwenden, denn diese bewegten sich ohne Wechselwirkung. Wir konnten daher 2 unabhängige IVP (Initial Value Problem) hinschreiben. Jetzt aber haben wir 3 Körper mit WW, und alle drei sind ein einziges Problem.

Wir betrachten zunächst nochmals die drei Körper aus dem ersten Programm, aber in einer Schreibweise, die sich leicht verallgemeinern lässt und mit Animation.

5.4.2 Bahn dreier Körper (Verallgemeinerung)

Programm 3:

```
reset():3 Koerper
m1:=1:m2:=1:m3:=1:
DIGITS=20:
IVP:={x1''(t)=
-m2*(x1(t)-x2(t))/sqrt((x1(t)-x2(t))^2+(y1(t)-y2(t))^2)^3
-m3*(x1(t)-x3(t))/sqrt((x1(t)-x3(t))^2+(y1(t)-y3(t))^2)^3,
y1''(t)=
-m2*(y1(t)-y2(t))/sqrt((x1(t)-x2(t))^2+(y1(t)-y2(t))^2)^3
-m3*(y1(t)-y3(t))/sqrt((x1(t)-x3(t))^2+(y1(t)-y3(t))^2)^3,
x2''(t)=
-m1*(x2(t)-x1(t))/sqrt((x2(t)-x1(t))^2+(y2(t)-y1(t))^2)^3
-m3*(x2(t)-x3(t))/sqrt((x2(t)-x3(t))^2+(y2(t)-y3(t))^2)^3,
y2''(t)=
-m1*(y2(t)-y1(t))/sqrt((x2(t)-x1(t))^2+(y2(t)-y1(t))^2)^3
-m3*(y2(t)-y3(t))/sqrt((x2(t)-x3(t))^2+(y2(t)-y3(t))^2)^3,
x3''(t)=
-m1*(x3(t)-x1(t))/sqrt((x3(t)-x1(t))^2+(y3(t)-y1(t))^2)^3
-m2*(x3(t)-x2(t))/sqrt((x3(t)-x2(t))^2+(y3(t)-y2(t))^2)^3,
y3''(t)=
-m1*(y3(t)-y1(t))/sqrt((x3(t)-x1(t))^2+(y3(t)-y1(t))^2)^3
-m2*(y3(t)-y2(t))/sqrt((x3(t)-x2(t))^2+(y3(t)-y2(t))^2)^3,
x1(0)=1,x2(0)=-1, x3(0)=0,//die 3 Koerper aus Programm 1
y1(0)=0,y2(0)=0,y3(0)=1.732,
x1'(0)=0.35355,x2'(0)=0.35355,x3'(0)=-0.7071,
y1'(0)=0.61237,y2'(0)=-0.61237,y3'(0)=0}:
/*x1(0)=-m2,x2(0)=m1, x3(0)=0, //Für Programm 4
y1(0)=0.7*m3,y2(0)=0,y3(0)=-0.7*m1,
x1'(0)=-1.01*m3,x2'(0)=0,x3'(0)=1.01*m1,
y1'(0)=-0.9*m2,y2'(0)=0.9*m1,y3'(0)=0}:*/
fields:=[x1(t),y1(t),x2(t),y2(t),x3(t),y3(t),
x1'(t),y1'(t),x2'(t),y2'(t),x3'(t),y3'(t)]:
ivp:=numeric::ode2vectorfield(IVP,fields):
Y:=numeric::odesolve2(ivp):
```

```
//Animation
dt:=0.05:imax:=60:
plot(//"Sonne" in Programm 4
plot::Point2d(Y(t)[1], Y(t)[2],
Color = RGB::Blue,
VisibleFromTo = t..t + 0.99*dt,
PointSize = 2*unit::mm)
$ t in [i*dt $ i = 0..imax],
plot::Line2d([Y(t - dt)[1], Y(t - dt)[2]],
[Y(t)[1], Y(t)[2]], Color = RGB::Blue,
VisibleAfter = t)
$ t in [i*dt $ i = 1..imax],
//Großer Planet
plot::Point2d(Y(t)[3], Y(t)[4],
Color = RGB::Red,
VisibleFromTo = t..t + 0.99*dt,
PointSize = 2*unit::mm)
$ t in [i*dt $ i = 0..imax],
plot::Line2d([Y(t - dt)[3], Y(t - dt)[4]],
[Y(t)[3], Y(t)[4]], Color = RGB::Red,
VisibleAfter = t)
$ t in [i*dt $ i = 1..imax],
//Kleiner Planet
plot::Point2d(Y(t)[5], Y(t)[6],
Color = RGB::Green,
VisibleFromTo = t..t + 0.99*dt,
PointSize = 2*unit::mm)
$ t in [i*dt $ i = 0..imax],
plot::Line2d([Y(t - dt)[5], Y(t - dt)[6]],
[Y(t)[5], Y(t)[6]], Color = RGB::Green,
VisibleAfter = t)
$ t in [i*dt $ i = 1..imax],
Scaling=Constrained):
```

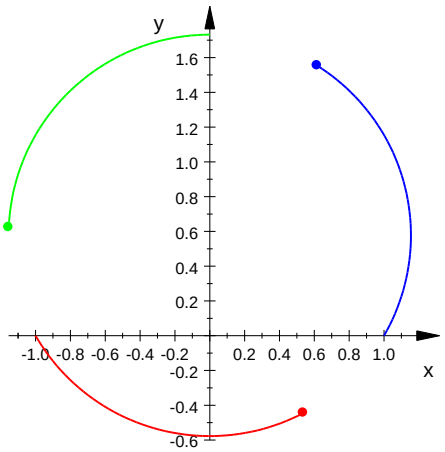

Fig.: 5.4-3

5.4.3 Bahnen von 4 Körpern

Wir untersuchen jetzt einen Fall mit 4 Körpern. Eine Sonde (m = 1000 kg) wird von der Erde aus in den Raum geschickt. Sie macht sich auf den Weg zur Venus, und nach vier Monaten wird sie ihren kürzesten Abstand von Venus erreicht haben. Die Flugdaten habe ich in der folgenden Tabelle zusammengestellt.

	Masse/kg	Ausgangsposition/m	Anfangsgeschw./ms^{-1}
Sonne	2E30	0,0	0,0
Venus	4.85E24	9.553E10, -5.08E10	16440, 30920
Erde	5.98E24	1.496E11,0	0, 0.29770
Sonde	1000	1.49E11, 1E10	-5000, 27200

Das folgende Programm ist eine kleine Variation des vorigen. Ich notiere also nur die wichtigsten Programmzeilen. Die Berechnungen der WW kann man bequem mithilfe von Programm 2 machen.

Programm 4:

```
 reset()://4Koerper
m1:=2E30*g:m2:=4.85E24*g:m3:=5.98E24*g:m4:=1E3*g:
g:=6.67E-11:
DIGITS=20:

IVP:={x1''(t)=

-m2*(x1(t)-x2(t))/sqrt((x1(t)-x2(t))^2+(y1(t)-y2(t))^2)^3
-m3*(x1(t)-x3(t))/sqrt((x1(t)-x3(t))^2+(y1(t)-y3(t))^2)^3
-m4*(x1(t)-x4(t))/sqrt((x1(t)-x4(t))^2+(y1(t)-y4(t))^2)^3,

y1''(t)=
-m2*(y1(t)-y2(t))/sqrt((x1(t)-x2(t))^2+(y1(t)-y2(t))^2)^3
-m3*(y1(t)-y3(t))/sqrt((x1(t)-x3(t))^2+(y1(t)-y3(t))^2)^3
-m4*(y1(t)-y4(t))/sqrt((x1(t)-x4(t))^2+(y1(t)-y4(t))^2)^3,
```

```
x2''(t)=
-m1*(x2(t)-x1(t))/sqrt((x2(t)-x1(t))^2+(y2(t)-y1(t))^2)^3
-m3*(x2(t)-x3(t))/sqrt((x2(t)-x3(t))^2+(y2(t)-y3(t))^2)^3
-m4*(x2(t)-x4(t))/sqrt((x2(t)-x4(t))^2+(y2(t)-y4(t))^2)^3,
...

x3''(t)=
-m1*(x3(t)-x1(t))/sqrt((x3(t)-x1(t))^2+(y3(t)-y1(t))^2)^3
-m2*(x3(t)-x2(t))/sqrt((x3(t)-x2(t))^2+(y3(t)-y2(t))^2)^3
-m4*(x3(t)-x4(t))/sqrt((x3(t)-x4(t))^2+(y3(t)-y4(t))^2)^3,
...
x4''(t)=
-m1*(x4(t)-x1(t))/sqrt((x4(t)-x1(t))^2+(y4(t)-y1(t))^2)^3
-m2*(x4(t)-x2(t))/sqrt((x4(t)-x2(t))^2+(y4(t)-y2(t))^2)^3
-m3*(x4(t)-x3(t))/sqrt((x4(t)-x3(t))^2+(y4(t)-y3(t))^2)^3,
...
x1(0)=0,x2(0)=9.553E10, x3(0)=1.496E11,x4(0)=1.49E11,
y1(0)=0,y2(0)=-5.08E10,y3(0)=0,y4(0)=1E10,
x1'(0)=0,x2'(0)=16440,x3'(0)=0,x4'(0)=-5E3,
y1'(0)=0,y2'(0)=30920,y3'(0)=29770,y4'(0)=27200}:

fields:=[x1(t),y1(t),x2(t),y2(t),x3(t),y3(t),x4(t),y4(t),
x1'(t),y1'(t),x2'(t),y2'(t),x3'(t),y3'(t),x4'(t),y4'(t)]:

ivp:=numeric::ode2vectorfield(IVP,fields):
Y:=numeric::odesolve2(ivp):
```

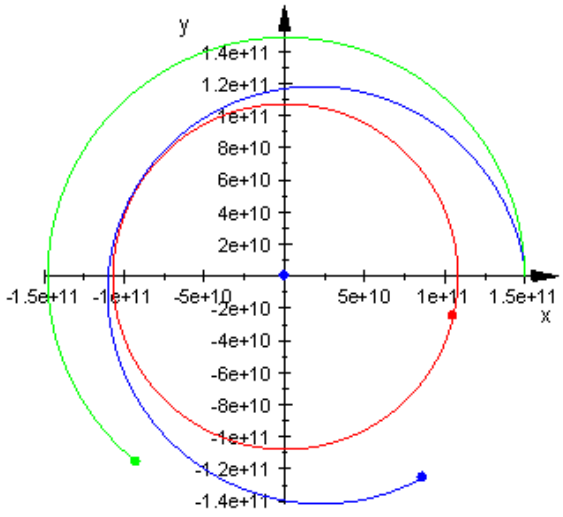

Fig.: 5.4-4

Die Sonne steht im Ursprung, und die Erde bewegt sich auf der äußeren (grünen) Bahn. Die Sonde bewegt sich auf der blauen Bahn. Für dt wurde 200000 gewählt und imax = 100.

5.4.4 Bahnen von 3 Körpern (Übungen)

Sie können mithilfe der vorigen Programme eigene "Sonnensysteme" aufbauen und untersuchen. Das einfachste wäre, die Ausgangsdaten der Programme zu variieren und die neuen Bahnen zu studieren.

Die folgende Figur zeigt, dass die Bahnen von Erde und Venus praktisch Kreise sind mit der Sonne im Zentrum.

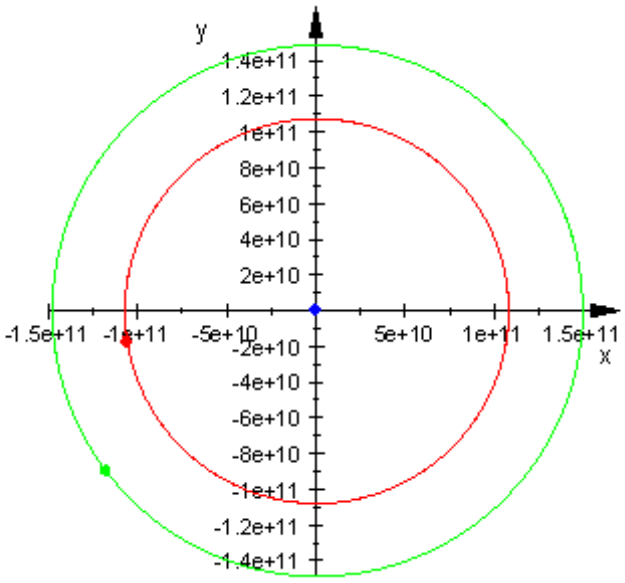

Fig.: 5.4-5

```
m1:=2E30*g:m2:=4.85E24*g:m3:=5.98E24*g:

g:=6.67E-11:DIGITS=30:
x1(0)=0,x2(0)=9.553E10, x3(0)=1.496E11,
y1(0)=0,y2(0)=-5.08E10,y3(0)=0,
x1'(0)=0,x2'(0)=16440,x3'(0)=0,
y1'(0)=0,y2'(0)=30920,y3'(0)=29770}:
```

dt:=2000000; imax := 100 ou mais

Im MUPAD-Manual finden Sie den Fall zweier Massen m_1, m_2 , die sich um eine "Sonne" von sehr großer Masse bewegen. Die folgeden Anfangsbedingungen wurden gewählt:

```
m1:=1:m2:=0.04:m3:=0.0001: (m1:= "Sonne")

x1(0)=-m2,x2(0)=m1, x3(0)=0,

y1(0)=0.7*m3,y2(0)=0,y3(0)=-0.7*m1,

x1'(0)=-1.01*m3,x2'(0)=0,x3'(0)=1.01*m1,

y1'(0)=-0.9*m2,y2'(0)=0.9*m1,y3'(0)=0}:
```

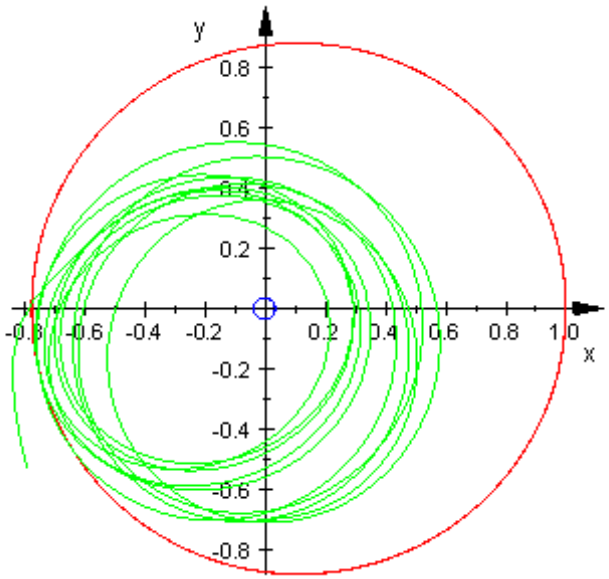

Fig.: 5.4-6

Die Figur zeigt, dass der kleine (grüne) Planet mit m3 = 0.0001 das System der 3 Körper verlassen wird. Die Bahn des größeren Körpers der Masse m1 = 1 erleidet nur geringe Störungen.

Programm 5 berechnet erneut die Bahn des Planeten Merkur, dessen Orbit in 5.2.1 mithilfe von **numeric::odesolve2** von MuPAD berechnet wurde.

An dieser Stelle werden wir uns nur auf die Definitionen von Geschwindigkeit und Beschleunigung stützen. Versuchen Sie das Programm zu verstehen, wobei Sie die folgenden Beziehungen benutzen:

$v(t) \approx (x(t+h) - x(t))/h$ und $a(t) \approx (v(t+h) - v(t))/h$, mit $h = \Delta t$, vgl. 5.4.1

Je kleiner das Zeitintervall h ist, desto besser nähern sich die Werte den Ableitungen x‘(t) = *v(t)* und x“(t) = *a(t)*. Dieses Verfahren heißt *Euler*-Algorithmus.

Programm 5

```
reset():
Euler:=proc(h,schritte)//Merkur /vergleiche 5.2.2
begin
t(0):=0:i:=0:
x(0):=0.3075:y(0):=0:u(0):=0:v(0):=1.982:
DIGITS:=6:
print("t= ",t(i),"x= ",x(i),"y= ",y(i)):

for i from 1 to schritte do
r(i):=sqrt(x(i-1)^2+y(i-1)^2):
ax:=-x(i-1)/r(i)^3:
ay:=-y(i-1)/r(i)^3:
u(i):=u(i-1)+ax*h:
v(i):=v(i-1)+ay*h:
x(i):=x(i-1)+u(i)*h:
y(i):=y(i-1)+v(i)*h:
t(i):=t(i-1)+h:
print("t= ",t(i),"x= ",x(i),"y= ",y(i)):

end_for:

//plot(plot::Point2d([x(i),y(i)])$
i=0..schritte,Color=RGB::Blue):
end_proc:
Euler(0.01,5)
```

Ergebnisse:

```
"t= ", 0, "x= ", 0.3075, "y= ", 0

"t= ", 0.01, "x= ", 0.306442, "y= ", 0.01982

"t= ", 0.02, "x= ", 0.304327, "y= ", 0.0395716

"t= ", 0.03, "x= ", 0.301158, "y= ", 0.0591862

"t= ", 0.04, "x= ", 0.296947, "y= ", 0.0785961

"t= ", 0.05, "x= ", 0.291713, "y= ", 0.0977349
```

5.4.5 Bahn von Mond und Erde (eingeschränktes 3-Körper-Problem)

Im "eingeschränkten 3-Körper-Problem" bewegen sich zwei massive Körper um das gemeinsame Massezentrum S während ein dritter, kleinerer Körper sich in derselben Ebene bewegt wie die beiden massiven. Wir können uns eine Sonde m_3 vorstellen, die sich im Gravitationsfeld von Erde m_1 und Mond m_2 bewegt. Der Einfluss der Sonne wird nicht berücksichtigt.

In Fig. 5.4-7 sehen wir Erde und Mond auf der x-Achse eines Koordinatensystems, das sich mit konstanter Winkelgeschwindigkeit ω dreht.

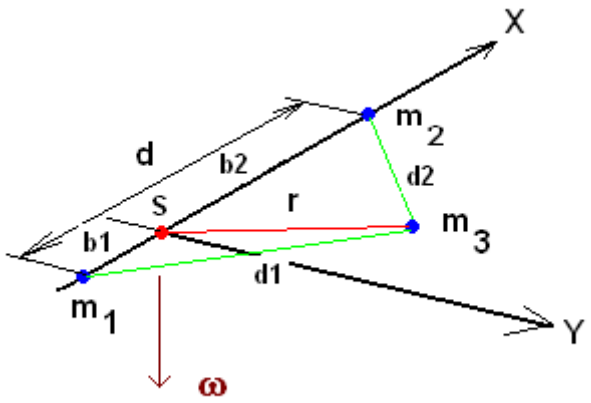

Fig.: 5.4-7

Die beiden Planeten beschreiben komplanare Kreise um ihren Schwerpunkt S. Damit haben wir ein 3-Körper-Problem auf ein 1-Körper-Problem reduziert, und unsere Aufgabe besteht darin, Position und Geschwindigkeit von m_3 im Ablauf der Zeit zu berechnen.

Die Erde hat von S den Abstand $b_1 = m \cdot d$, wobei $m := m_2/(m_1 + m_2)$. Die Entfernung zwischen S und Mond beträgt $b_2 = m' \cdot d$ mit $m' = 1-m$.

Die Winkelgeschwindigkeit hat die Richtung der z-Achse, und ihr Wert ist durch $\omega^2 = G\,(m_1 + m_2)/d^3$ gegeben.

In einem Inertialsystem wäre Newtons zweites Gesetz $m_3\mathbf{a} = \mathbf{F}_1 + \mathbf{F}_2$, worin $\mathbf{F}_1$ und $\mathbf{F}_2$ die Kräfte von m_1 und m_2 sind. In unserem nichtinertialen System haben wir zwei "Trägheitskräfte" einzuführen: Zentrifugalkraft $\mathbf{F}_c = -m_3\,\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})$ und CORIOLIS-Kraft $\mathbf{F}_{cor} = -2\,m_3\,\boldsymbol{\omega} \times \mathbf{v}_{rel}$.

Die Bewegungsgleichungen für die beiden Koordinaten von m_3 lauten (2.5.3)

$$\frac{d^2x}{dt^2} = x + 2\frac{dy}{dt} - \frac{m'(x+m)}{d_1^3} - \frac{m(x-m')}{d_2^3} \qquad (6)$$

$$\frac{d^2y}{dt^2} = y - 2\frac{dx}{dt} - \frac{m'y}{d_1^3} - \frac{my}{d_2^3} \qquad (7)$$

Die Zeiteinheit wurde so gewählt, dass $\omega = 1$,d.h. so, dass die Zeit für eine Drehung des Koordinatensystems $T = 2\pi$ ist.
Die Abstände d_1 und d_2 sind

$$d_1^2 = (b_1 + x)^2 + y^2 = (m \cdot d + x)^2 + y^2 \qquad (8)$$

$$d_2^2 = (b_2 - x)^2 + y^2 = (m \cdot d - x)^2 + y^2 \qquad (9)$$

Wenn wir $d = 1$ wählen, ist die Masse $m = 0.012277471$.

Die Anfangsbedingungen sind $x_0 = 0.994$ (d.h. rechte Seite des Mondes), $y_0 = 0$.
$dx(0)/dt := v_x(0) = 0$ und $dy(0)/dt := v_y(0) = -2.1138987966945$.

Diese große Anzahl von Dezimalstellen ist nötig, da die Berechnungen sehr empfindlich sind in Bezug auf Variationen von ihnen. In den frühen Tagen der Raumfahrt war es absolut notwendig, dass die "injection speed" von 10840 m/s um nicht mehr als 1 m/s variierte. Bei einem Unterschied von > 2 m/s hätte man den Mond nicht getroffen, denn es gab keine Möglichkeit, die Bahn während des Fluges zu korrigieren.

Mit dem folgenden Programm können Sie diese Aussagen nun selbst überprüfen. Ein durchschnittlicher PC berechnet heute (2014) eine Trajektorie in wenigen Sekunden. (Zur Zeit der ersten PCs, dauerte eine Flugbahnberechnung Stunden! Der Tischrechner HP-85 benötigte 1980 ca. 4 Stunden zur Berechnung eines Orbits!).

Programm 6:

```
x0:=0.994://Eingeschränktes 3-Körper-Problem
y0:=0:vx0:=0: vy0:=-2.1138987966945027:
m:=0.012277471: mu:=1-m:
r1:=((x+m)^2+y*y)^(3/2):
r2:=((x-mu)^2+y*y)^(3/2):
IVP:={x''(t)=x(t)+2*y'(t)-mu*(x(t)+m)/r1-m*(x(t)-mu)/r2,
y''(t)=y(t)-2*x'(t)-mu*y(t)/r1-m*y(t)/r2,
x(0)=x0,x'(0)=vx0,
y(0)=y0,y'(0)=vy0}:
fields:=[x(t),y(t),x'(t),y'(t)]:
ivp:=numeric::ode2vectorfield(IVP, fields):
Y := numeric::odesolve2(ivp):
//Animation
dt:=0.005:imax:=990:
plot(
plot::Point2d(-0.01228,0, Color=RGB::Black,
PointSize=3*unit::mm),//Erde
plot::Point2d(0.9877,0, Color=RGB::Green,
PointSize=2*unit::mm), //Mond
plot::Point2d(Y(t)[1], Y(t)[2],
Color = RGB::Blue,VisibleFromTo = t..t + 0.99*dt,
PointSize = 2*unit::mm) $ t in [i*dt $ i = 0..imax],
plot::Line2d([Y(t - dt)[1], Y(t - dt)[2]],
[Y(t)[1], Y(t)[2]], Color = RGB::Red,VisibleAfter = t)
$ t in [i*dt $ i = 1..imax]):
```

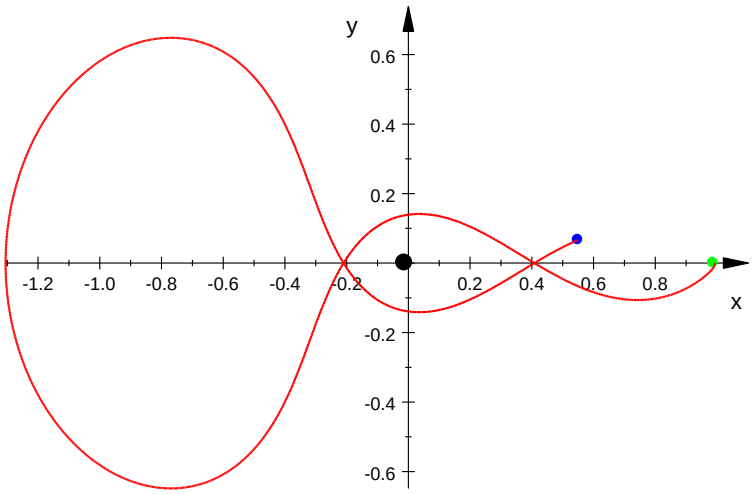

Fig.: 5.4-8

In der Figur wurde m_3 von der Mondoberfläche (grün) abgeschossen, um zur Erde zurückzukehren. Sie flog weit an der Erde (schwarz) vorbei und ist dabei, wieder zum Mond zurückzukehren.

Ort der Erde: x(0) = -0.01228, y(0) = 0; Mond: x(0) = 0.9887, y(0) = 0. Die Sonde startet in x(0) = 0.994 und y(0) = 0 (rechts vom Mond).

In Fig. 5.4.9 wurden dt = 0.002 und imax = 5500 benutzt. $v_y(0)$ = -2.0325. Die resultierende Bahn ist sehr kompliziert und nicht geschlossen.

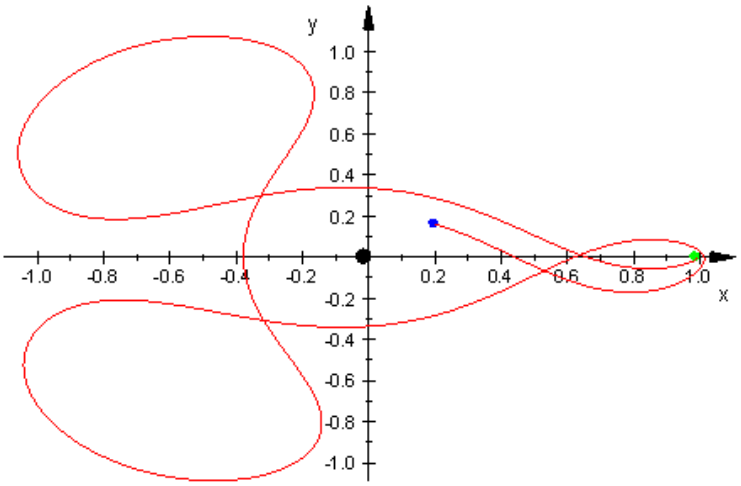

Fig.: 5.4-9 m