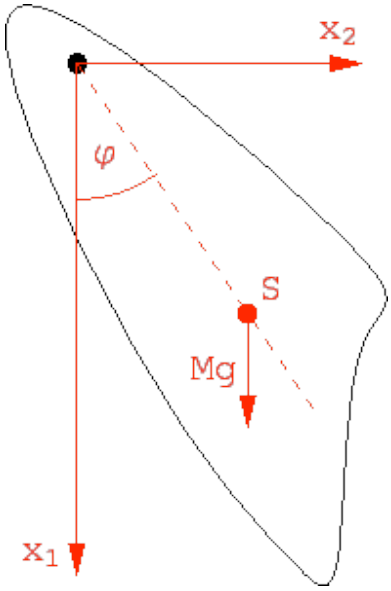
# Physikalisches Pendel im dreidimensionalen Raum



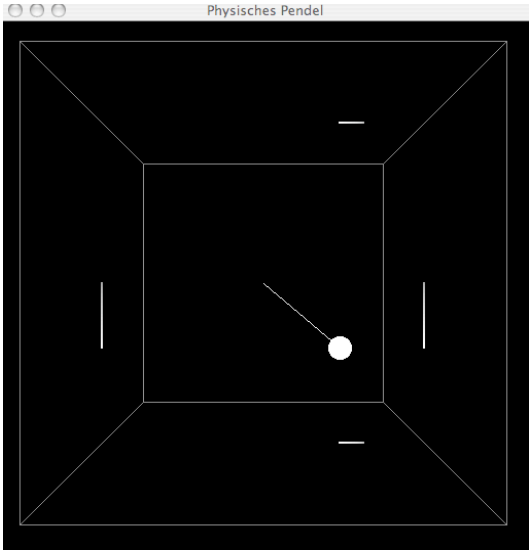

**Eine Semesterarbeit im Fach Numerik an der FH Zürich**

**Eingereicht bei Albert Heuberger**

**von**

**Gisela Dymorz**
**Adrian Fischer**
**Dominik Scherrer**
**Klasse 3ic**

**Zürich, 05.03.2007**



## Beschreibung

Der Dozent im Fach Numerik, Herr Heuberger, gab uns die Aufgabe, ein reelles Problem der Natur anhand einer Differentialgleichung zu erforschen. Mit den verschiedenen Verfahren und Funktionen in Java-Code, die wir bei ihm in den Vorlesungen studiert und implementiert hatten, sollten wir ein Programm entwickeln, welches das zu erforschende Problem abbildet.

## Aufgabenstellung

Wir stellten uns die Aufgabe, ein physikalisches Pendel im dreidimensionalen Raum zu programmieren. Wir setzten uns zum Ziel, ein möglichst realitätsnahes Ergebnis zu liefern. Dabei gingen wir zuerst von nur zwei Dimensionen aus und hofften, die dritte Dimension einzubauen sei dann ein leichtes Unterfangen. Unser Programm sollte eine möglichst gute Annäherung an die Realität sein. Zur eigentlichen Berechnung wollten wir auch eine einfache grafische Darstellung des Pendels haben.

## Java-Code für die Berechnung der Differentialgleichung

```
public double[] w(double t, double[] y) {
1.    double[] res, a, x, v;
2.    double kraft_seil, ags_betrag, faktor, kraft_luftwiderstand, ar_betrag;
3.    res = new double[6];
4.    x = new double[3];
5.    v = new double[3];

6.    x[0] = y[0];
7.    x[1] = y[1];
8.    x[2] = y[2];
9.    v[0] = y[3];
10.   v[1] = y[4];
11.   v[2] = y[5];

12.   kraft_seil = ((ds.getLaenge() - calcAbs(ds.getP(), x)) * ds.getFederkonstante())
13.   + ds.getDämpferkonstante() * calcAnteil(v, calcDiff(ds.getP(), x));
14.   ags_betrag = kraft_seil / ds.getMasse();
15.   if (ags_betrag > 0) { ags_betrag = 0;}
16.   faktor = ags_betrag / calcAbs(ds.getP(), x);
17.   if (Double.isNaN(faktor)) faktor = 0;
18.   a = calcProd(faktor, calcDiff(x, ds.getP()));

19.   a[1] -= 9.81;

20.   kraft_luftwiderstand = ds.getCw() / 2 * ds.getDichte() * Math.pow(calcAbs(v), 2) * Math.PI * Math.pow(ds.getRadius(),
      2);
21.   ar_betrag = kraft_luftwiderstand / ds.getMasse();
22.   faktor = ar_betrag / calcAbs(v);
23.   if (Double.isNaN(faktor)) faktor = 0;
24.   a = calcSum(a, calcProd(faktor, calcProd(-1, v)));

25.   res[0] = y[3];
26.   res[1] = y[4];
27.   res[2] = y[5];
28.   res[3] = a[0];
29.   res[4] = a[1];
30.   res[5] = a[2];

31.   return res;
}
```

# Berechnungen für die Endversion der Differentialgleichung

Beschreibung der verwendeten Funktionen:
Richtung(Punkt 1, Punkt 2) Abstand zwischen den Punkten als Vektor (Im Programm: calcDiff())
Anteil(Vektor 1, Vektor 2) Anteil von Vektor 1 in Richtung Vektor 2 (Im Programm: calcAnteil())
Abstand(Punkt 1, Punkt 2) Abstand zwischen den Punkten als Skalar (Im Programm: calcAbs())

```
1.  double[] res, a, x, v;
2.  double kraft_seil, ags_betrag, faktor, kraft_luftwiderstand, ar_betrag;
3.  res = new double[6];
4.  x = new double[3];
5.  v = new double[3];
```

Linie 1-5: Deklaration und Initialisierung der benötigten Variablen

```
6.  x[0] = y[0];
7.  x[1] = y[1];
8.  x[2] = y[2];
9.  v[0] = y[3];
10. v[1] = y[4];
11. v[2] = y[5];
```

Linie 6-8: Speicherung der Pendelposition (x-,y-,z-Richtung) im Array x
Linie 9-11: Speicherung der Pendelgeschwindigkeit (x-,y-,z-Richtung) im Array v

```
12. kraft_seil = ((ds.getLaenge() - calcAbs(ds.getP(), x)) * ds.getFederkonstante())
13. + ds.getDämpferkonstante() * calcAnteil(v, calcDiff(ds.getP(), x));
14. ags_betrag = kraft_seil / ds.getMasse();
15. if (ags_betrag > 0) { ags_betrag = 0;}
16. faktor = ags_betrag / calcAbs(ds.getP(), x);
17. if (Double.isNaN(faktor)) faktor = 0;
18. a = calcProd(faktor, calcDiff(x, ds.getP()));
```

Linie 12,13: Berechnung des Absolutwerts (Skalar) der Kraft im Seil [1]
$F_{Seil}$ = ($Länge_{Seil}$ – Abstand(Punkt P, Punkt X)) * Federkonstante
+ Dämpferkonstante * Anteil(v, Richtung(Punkt P, Punkt X))
Linie 14: F = m * a ➔ a = F / m;
Linie 15: Wenn der auf der vorhergehenden Linie berechneten Betrag der Beschleunigung durch das Seil grösser 0 ist, wird die Beschleunigung 0 gesetzt. Ein Seil kann Impuls nur in eine Richtung leiten, weshalb die Kraft nur in eine Richtung wirken kann.
Linie 16: Den Faktor berechnen, mit dem der Abstand zwischen den Punkten P und X (Vektor) multipliziert werden muss, um die Beschleunigung zu erhalten
Linie 17: Ist dieser Faktor keine Zahl (passiert, wenn der Abstand zwischen Punkt P und X 0 ist) den Faktor 0 setzen
Linie 18: Die Beschleunigung (in alle Richtungen, als Vektor) ist der Abstand zwischen Punkt P und X multipliziert mit dem Faktor von Linie 16

```
19. a[1] -= 9.81;
```

Linie 19: Die Gravitationsbeschleunigung (-9.81m/s^2) ist unabhängig der Masse, der Geschwindigkeit und Umgebung und wird zur Beschleunigung addiert

[1] Formel aus Taschenbuch der Physik, Horst Kuchling, Seite 103, Formel M 7.7

```
20.  kraft_luftwiderstand = ds.getCw() / 2 * ds.getDichte() * Math.pow(calcAbs(v), 2) * Math.PI *
     Math.pow(ds.getRadius(), 2);
21.  ar_betrag = kraft_luftwiderstand / ds.getMasse();
22.  faktor = ar_betrag / calcAbs(v);
23.  if (Double.isNaN(faktor)) faktor = 0;
24.  a = calcSum(a, calcProd(faktor, calcProd(-1, v)));
```

Linie 20: Berechnung des Absolutwerts (Skalar) der Kraft, die durch die Luftreibung verursacht wird [2]

$F_{Strömung} = c_w * A * (\delta / 2) * v^2$

| | | |
|---|---|---|
| $c_w$ | Luftwiderstandsbeiwert | bei einer Kugel 0.45 |
| A | Projektionsfläche | |
| δ | Dichte | Dichte von Luft 1.204 kg / m^3 |

Linie 21: F = m * a ➔ a = F / m;

Linie 22: Den Faktor berechnen, mit dem die Geschwindigkeit v multipliziert werden muss, um die Beschleunigung, die durch die Luftreibung verursacht wird, zu erhalten

Linie 23: Ist dieser Faktor keine Zahl (passiert, wenn die Geschwindigkeit 0 ist) den Faktor 0 setzen

Linie 24: Die Gesamtbeschleunigung ist die Summe aus der alten Beschleunigung und dem Produkt des Faktors und der Geschwindigkeit (negativ, da die Reibung entgegen der Richtung der Geschwindigkeit wirkt)

```
25.  res[0] = y[3];
26.  res[1] = y[4];
27.  res[2] = y[5];
28.  res[3] = a[0];
29.  res[4] = a[1];
30.  res[5] = a[2];
```

Linie 25-26: Speicherung der Geschwindigkeit im ersten Bereich des Resultat-Arrays

Linie 28-30: Speicherung der errechneten Beschleunigung im zweiten Bereich des Resultat-Arrays

```
31.  return res;
```

Linie 31: Rückgabe des Resultat-Arrays

Da unsere Darstellung möglichst genaue Werte benötigt, wird nicht der Euler-Algorithmus sondern der Runge-Kutta-Algorithmus vierter Ordnung verwendet. Mit dem Euler-Algorithmus summiert sich der Berechnungsfehler so extrem, dass die Geschwindigkeit des Pendels immer grösser wird.

Die Differentialgleichungen vereinfacht dargestellt:

$$x0' = v0$$

$$x1' = v1$$

$$x2' = v2$$

$$v0' = a0 = (faktor1(v,x,p) * |x0 - p0|) + (faktor2(v) * v0)$$

$$v1' = a1 = (faktor1(v,x,p) * |x1 - p1|) - g + (faktor2(v) * v1)$$

$$v2' = a2 = (faktor1(v,x,p) * |x2 - p2|) + (faktor2(v) * v2)$$

[2] Formel aus Taschenbuch der Physik, Horst Kuchling, Seite 173, Formel M 10.20

## Dateiablage

Auf http://adrian-fischer.ch/pendel befindet sich das Programm, diese Dokumentation, die Java-Dokumentation, der Java-Code und die verschiedenen Versionen. Der Link um das Programm zu starten ist ebenfalls auf dieser Seite, das Programm startet nach einem Klick automatisch. Für das Programm wird die JDK 5 (JDK 1.5) benötigt (für alle Betriebssysteme unter www.sun.com erhältlich).

## Bedienung des Programms

In der Bedieneroberfläche der grafischen Darstellung gibt es drei verschiedene Abschnitte.
Im obersten Abschnitt können alle Einstellungen mit einem Auswahlknopf gesetzt werden. Unter dieser Auswahl sind vier verschiedene Default-Einstellungen für vorkonfigurierte Abläufe vorhanden.
Im mittleren Abschnitt können alle Einstellungen manuell verändert werden: die Startposition des Pendels, die Startgeschwindigkeit in jede Richtung, das Gewicht der Kugel, der Radius der Kugel, die Dämpferkonstante des Seils, die Federkostante des Seils, das Umfeld und eine eingebaute Störung auf dem Aufhängepunkt P. Mit “Start“ wird der Vorgang gestartet. Sind irgendwelche Werte nicht in den gegebenen Toleranzen wird eine Fehlermeldung ausgegeben. Mit “Stopp“ wird der Vorgang in der aktuellen Position gestoppt und mit “Aktuelle Werte holen“ können die aktuellen Werte vom Programm abgerufen werden.
Im untersten Abschnitt kann mit den gesetzten Werten eine Tabelle mit den Ausgabewerten erstellt werden. Die Tabelle kann kopiert und in ein anderes Programm eingefügt werden (z.B. Excel). Diese Ausgabe basiert auf der Funktion fTable von Herrn Heuberger.

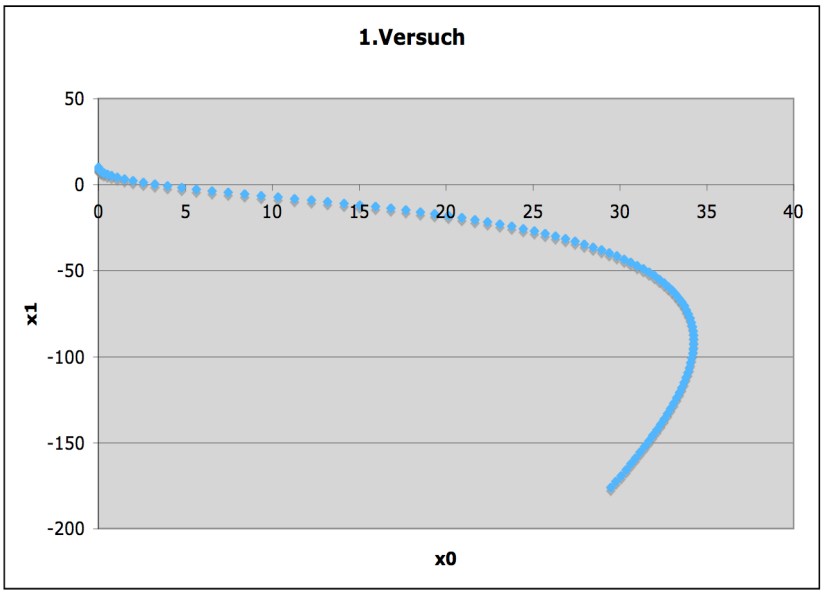
## Anhang - Erste Schritte

Am Anfang mussten wir die Differentialgleichung eines Pendels herausfinden. Als Systemgrössen definierten wir nur die Koordinatenpunkte für die Aufhängung und die Erdanziehungskraft. Die Position des Pendels und die Geschwindigkeitsanteile in jede Richtung definierten wir als Zustandsgrössen. Die Ideen, die wir hatten, schienen uns so einleuchtend und richtig, dass wir schnell ans Programmieren gingen. Mit den Programmbauteilen, die wir aus dem Kurs hatten, wollten wir die Auflösung der Differentialgleichung programmieren. Leider schien die von uns ausgedachte Differentialgleichung nicht die Richtige zu sein.

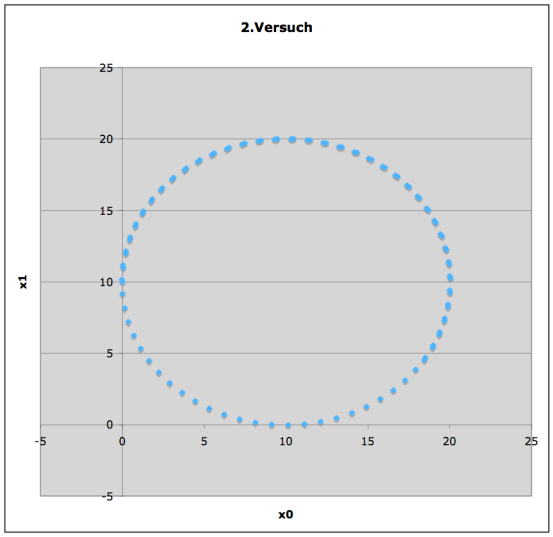
Wir gingen von folgenden Formeln aus:

$$x0' = v0$$
$$x1' = v1$$
$$x2' = v2$$
$$v0' = a0 = |v|^2 * |x0 - p0| \div |x - p|^2$$
$$v1' = a1 = (|v|^2 * |x1 - p1| \div |x - p|^2) - g$$
$$v2' = a2 = |v|^2 * |x2 - p2| \div |x - p|^2$$

Anstatt hin und her zu pendeln sauste unser Pendel ins negative davon. Was war falsch?




Beim zweiten Versuch nahmen wir die gleichen Formeln, liessen aber die Erdanziehung (g) weg und gaben dafür eine Anfangsgeschwindigkeit in der x1 Achse der Stärke 10 mit. Gross war unsere Freude über das Ergebnis. Unser Pendel lief an einer Stange im Kreis. Noch nie hatten wir uns über einen Kreis so sehr gefreut! So sollte es doch ein Einfaches sein, die Erdanziehung einzubauen und unser Projekt voranzutreiben.

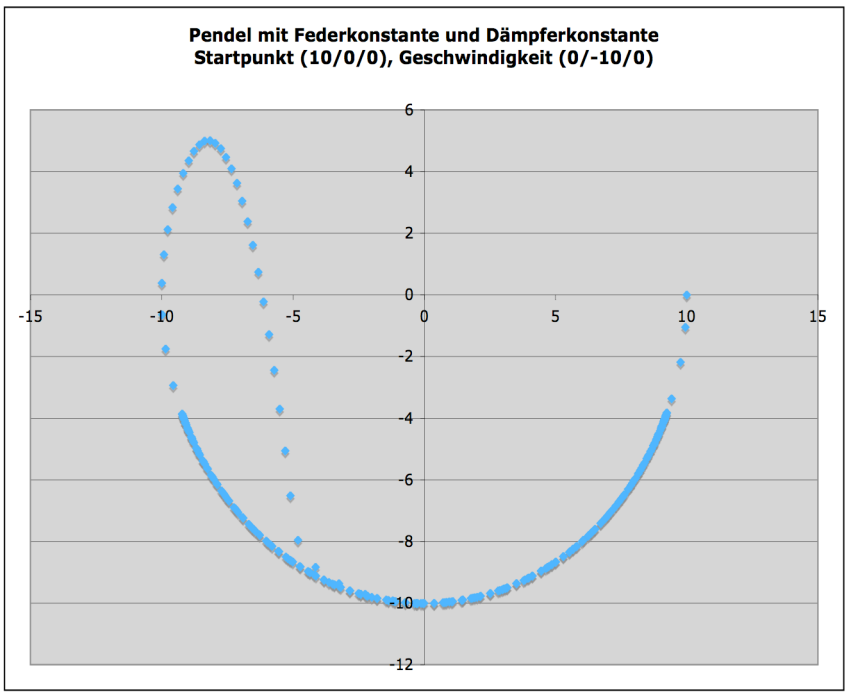



Bald fanden wir jedoch heraus, dass die Formeln, die wir uns erarbeitet hatten, nicht stimmen konnten.
Wir hatten bei der Berechnung die Kräfte falsch eingeteilt, deshalb mussten wir neue Formeln berechnen. Nun ermittelten wir zuerst die Beschleunigung, welche die Schwerkraft auf die Stange hat. Anschliessend addierten wir die Beschleunigung, die sich aus der Zentripetalkraft ergibt, zur schon vorhandenen Beschleunigung dazu.

Mit den Formeln[3] $\varphi = a\cos(\vec{a} \times \vec{b} / |\vec{a}| * |\vec{b}|)$ und $ags\_betrag = \cos(\varphi) * a\_gravitation$ berechneten wir die Beschleunigung, die durch die Schwerkraft entsteht. Ausgehend von $faktor = \frac{ags\_betrag}{\sqrt{(x0)^2 + (x1)^2 + (x2)^2}}$ berechneten wir den Faktor, den die Beschleunigung in Richtung der drei Achsen hat. Nun pendelte unser

[3] Formel aus Taschenbuch der Physik, Horst Kuchling, Seite 31, Formel G 1.5

Pendel endlich!
Mit dieser Version setzten wir uns mit Herrn Heuberger zusammen und diskutierten das weitere Vorgehen. Wir beschlossen, noch eine Störfunktion einzubauen. Herr Heuberger ermutigte uns, zu versuchen den Aufhängepunkt variieren zu lassen. So könnte man eine Bewegung simulieren, wie wenn man das Pendel mit der Hand ins Schwingen bringt.
Mit unseren Formeln funktionierte dies jedoch nicht so einfach, da wir keine Zustandsgrösse definiert hatten, welche den Abstand zwischen den zwei Punkten berechnet. Um dieses Problem zu lösen definierten wir die Verbindung zwischen dem Aufhängepunkt und der Kugel als "Feder" mit einer sehr hohen Federkonstante und einer sehr hohen Dämpfung.




Weiter hatten wir Mühe, dass das Pendel bei zu knapper Geschwindigkeit nicht nach unten fiel, sondern weiter eine Kreisbewegung ausführte. Es hing somit noch immer an einer Stange und nicht an einem Seil. Für dieses Problem führten wir eine neue Berechnung ein, die errechnete, in welche Richtung der Geschwindigkeits-vektor zeigte. Mit dieser Erweiterung waren wir schon ziemlich nahe an dem, was wir uns vorstellten.
Was uns für ein reelles Pendel noch fehlte, war die Dämpfung, die durch die Reibung der Kugel mit der Luft oder anderen Umgebungen entsteht. In der Gruppe suchten wir im Internet nach den richtigen Formeln für die Reibung. Nachträglich haben wir diese Formel auch im Taschenbuch der Physik von Horst Kuchling gefunden.[4]

Die Formel $F_{Strömung} = c_w * A * (\delta / 2) * v^2$ war für uns die richtige. Wobei folgende Variablen gelten:

- $c_w$ ist der Luftwiderstandsbeiwert => für eine Kugel beträgt dieser 0,45
- δ ist die Dichte der Luft => bei 20° beträgt diese 1,204 und bei 0° 1,292
- A ist die Projektionsfläche => für eine Kugel ist dies $A_{PF} = \Pi * r^2$
- v ist die Windgeschwindigkeit => bei uns ist dies der Absolutwert der Kugel-Geschwindigkeit

Mit diesen Formeln berechneten wir erneut eine Beschleunigung in Richtung der drei Achsen, welche wir zu den bereits berechneten Werten dazuaddierten.

So hatten wir unsere Differentialgleichung endlich auf einem reellen Stand.

[4] Formel aus Taschenbuch der Physik, Horst Kuchling, Seite 173, Formel M 10.20