prob10_1__Baelle_werfen

Sage Notebook zur Aufgabe 10.1

der Vorlesung

Theoretische Physik 1. Mechanik
Uni Leipzig, Wintersemester 2018/19
Autor: Jürgen Vollmer (2018)

Sage ist ein OpenSource Projekt, das viele Methoden der computerbasierten Mathematik und Computer-Algebra in einem Python-basierten Notebook anbietet.
Dokumentation und Informationen zur **Installation** findet man auf
https://sagemath.org
Eine hervorragende Einführung in das Arbeiten mit Sage bietet das Buch
Paul Zimmermann, u.a.: "Computational Mathematics with SageMath"
http://sagebook.gforge.inria.fr/english.html

Allgemeine Definitionen, Variablen, Konstanten

Pfad und Stammname für Abbildungen

Bitte den Pfad editiert und die Kommentarzeichen vor den "save_image()"-Befehlen entfernen, um die erstellten Dateien zu speichern.

```
baseName = 'XXX--bitte editieren--XXX/2018W_Mechanik/Uebungen
/Sage/prob10_1__Baelle_werfen__'
```

Variablen deklarieren

```
n, Romega, Xvelo, Yvelo = var('n', 'Romega', 'Xvelo', 'Yvelo')
alpha = var('alpha')
```

Anfangswerte festlegen

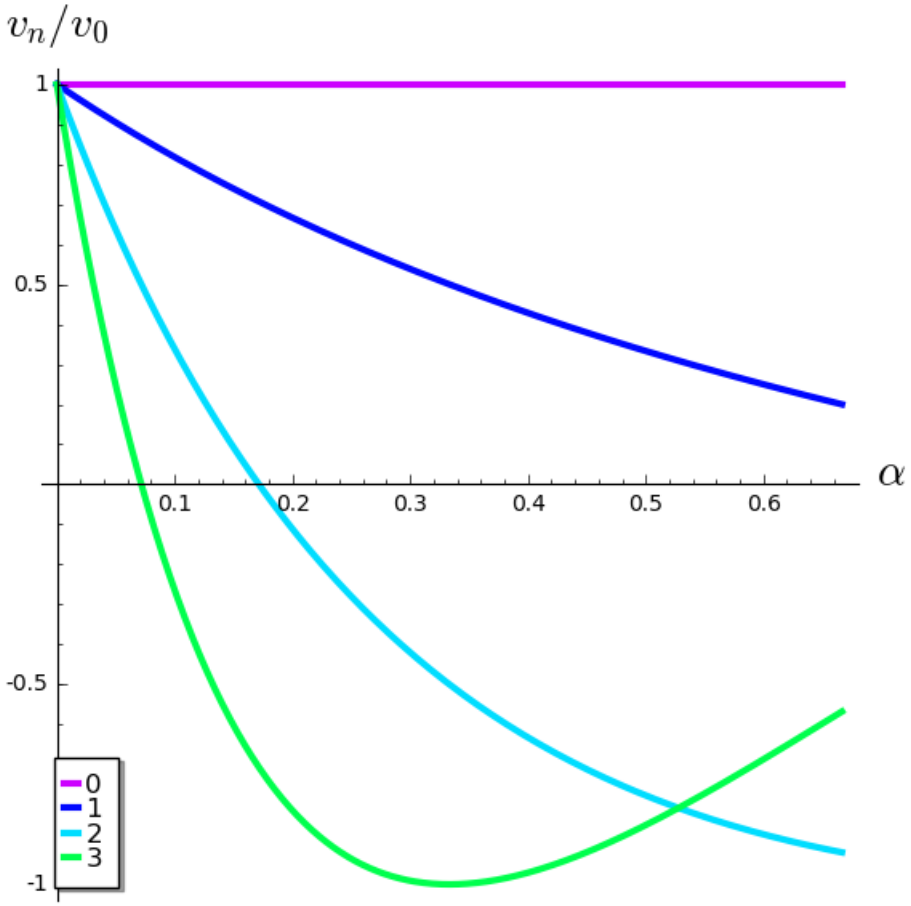
```
nIni = 0
RomegaIni =  0
XveloIni  =  1
YveloIni  = -1
```

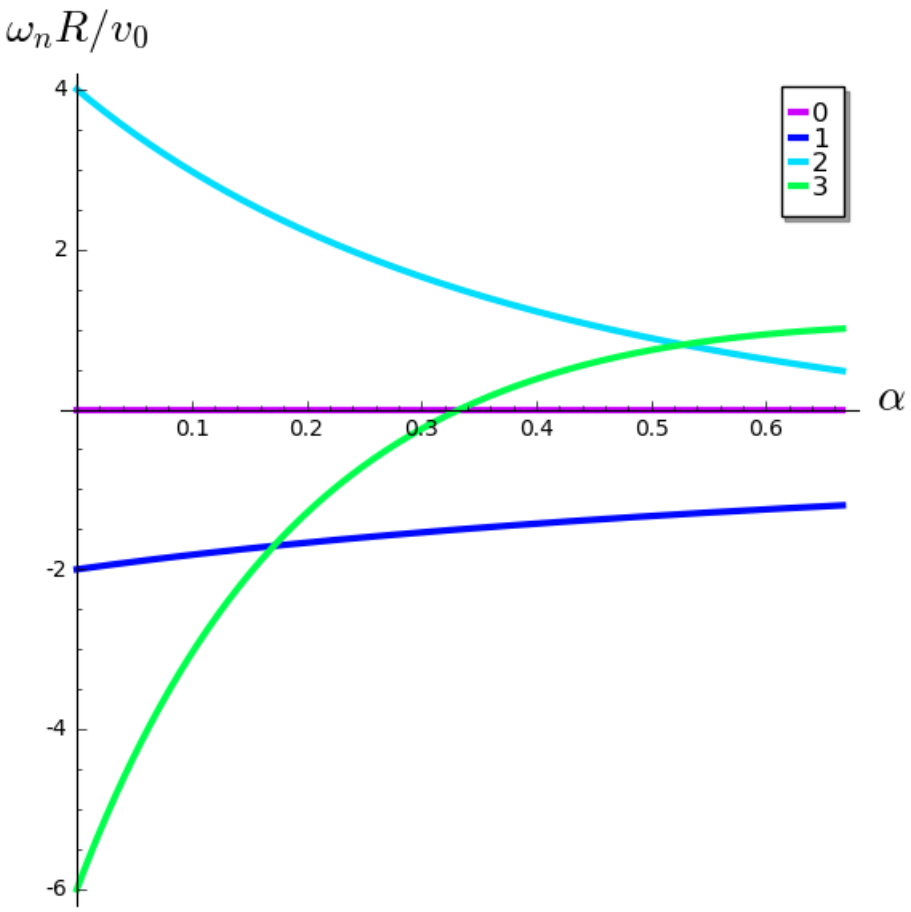
Kollisionsregel

```
def coordUpdate(n, Romega, Xvelo, Yvelo) :
    n += 1
    RomegaN = ( (-1)^n * 2 * Xvelo - (1-alpha) * Romega ) / (1+alpha)
    XveloN  = ( (1-alpha) * Xvelo + (-1)^n * 2 * alpha * Romega ) /  (1+alpha)
    YveloN  = -1 * Yvelo
    return n, RomegaN, XveloN, YveloN
```

α-Abhängigkeit von (v_n, ω_n)

```
nrCollisions = 4

n      = nIni
Romega = RomegaIni
Xvelo  = XveloIni
Yvelo  = YveloIni


p = plot([])
q = plot([])

for j in range(nrCollisions):
    p += plot( Xvelo,  (alpha, 0, 2/3), color=hue(.8-
float(j)/(1.8*nrCollisions)), legend_label=j, thickness=3 )
    q += plot( Romega, (alpha, 0, 2/3), color=hue(.8-
float(j)/(1.8*nrCollisions)), legend_label=j, thickness=3 )
    n, Romega, Xvelo, Yvelo = coordUpdate(n, Romega, Xvelo, Yvelo)

p.axes_labels( [r'$\alpha$', r'$v_n / v_0$'] )
p.axes_labels_size( 2 )
p.show(figsize=[6,6])
# p.save_image(baseName+'Xvelo.svg', figsize=[6,6])

q.axes_labels( [r'$\alpha$', r'$\omega_n R / v_0$'] )
q.axes_labels_size( 2 )
q.show(figsize=[6,6])

# q.save_image(baseName+'Romega.svg', figsize=[6,6])
```


v_n/v_0
1
0.5
-0.5
-1
0.1
0.2
0.3
0.4
0.5
0.6
α
0
1
2
3


$\omega_n R/v_0$
4
2
-2
-4
-6
0.1
0.2
0.3
0.4
0.5
0.6
α
0
1
2
3

Bahn des Balles

Anfangswerte festlegen

```
nrCollisions = 12
alphaVal = 0.4

Xcoord, Ycoord, t = var('Xcoord', 'Ycoord', 't')
Xcoord = 0
Ycoord = 0.75

n      = nIni
Romega = RomegaIni
Xvelo  = XveloIni
Yvelo  = YveloIni
```

Bahn zeichnen

```
slp  = (Yvelo / Xvelo).substitute(alpha=alphaVal)

X0 = Xcoord
if (slp > 0) :
    X1 = (1-Ycoord)/slp
else :
    X1 = Xcoord - Ycoord/slp
Xmin = 0
Xmax = Xvelo*X1

p = plot( Ycoord+slp*(x-X0), (x, X0, X1), color='red', thickness=3 )

for j in range(nrCollisions):

    n, Romega, Xvelo, Yvelo = coordUpdate(n, Romega, Xvelo, Yvelo)
    slp = (Yvelo / Xvelo).substitute(alpha=alphaVal)

    X0 = X1
    if (Yvelo > 0) :
        Ycoord = 0
        X1 += 1/slp
    else :
        Ycoord = 1
        X1 += -1/slp

    if ( X1 > Xmax ) : Xmax = X1
    if ( X1 < Xmin ) : Xmin = X1

    if  X0<X1 :
        xL, xR = X0, X1
    else :
        xL, xR = X1, X0

    p += plot( Ycoord+slp*(x-X0), (x, xL, xR), color=hue(.8-
float(j)/(1.8*nrCollisions)), thickness=3)

Xmin -= 0.05 * (Xmax - Xmin)
Xmax += 0.05 * (Xmax - Xmin)
p += plot( 1, (x, Xmin, Xmax), color='black', thickness=5)
p += plot( 0, (x, Xmin, Xmax), color='black', thickness=5)
```

```
p += text( r'$\alpha=$'+alphaVal.str(digits=2), (0.5, 1.1), fontsize=16,
color='black' )

p.axes_labels( [r'$x$', r'$y$'] )
p.axes_labels_size( 2 )
p.show(figsize=[6,4], ymin=0, ymax=1)

# p.save_image(baseName+'Bahn.svg', figsize=[6,4])
```