prob02_B__freier_Fall

Sage Notebook zur Aufgabe
2.B Freier Fall mit turbulenter Reibung

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/prob02_B__freier_Fall__'
```

Pakete laden für Plotten und Numerik

```
import scipy; from scipy import integrate
import numpy as np
```

Differentialgleichung als Funktion von t

```
t = var('t')
def dW_dt(X, t=0) :
    return  -1 - np.sign(X) * X * X
```

Trajektorien plotten

2 Trajektorien berechnen

```
t1 = srange(-1.4, 4, 0.1 )
X1ini = [ -1.4,  np.tan(1.4) ]
X1 = integrate.odeint( dW_dt, X1ini, t1 )
U,W1 = X1.T

t2 = srange(0.15, 4, 0.01 )
```

```
X2ini = [ 0.15, -1/np.tanh(0.15) ]
X2 = integrate.odeint( dW_dt, X2ini, t2 )
U,W2 = X2.T
```

Trajektorien plotten und Theoriekurven darüberlegen

```
p  = line( zip(t1, W1), color='red', thickness=2 )
p += text( r'$[ -1.4, -\tan(1.4) ]$', (2,5), fontsize=16,
horizontal_alignment='left', color='red' )
p += plot( -np.tan(x), x, -1.4, 1.4, color='green', thickness=5, linestyle=':'
)
p += text( r'$-\tan(\tau-\tau_0)$', (2,4), fontsize=16,
horizontal_alignment='left', color='green' )
p += plot( -np.tanh(x), x, -1.4, 4, color='gray', thickness=4, linestyle='--'
)
p += text( r'$-\tanh(\tau-\tau_0)$', (2,3), fontsize=16,
horizontal_alignment='left', color='gray' )

p += line( zip(t2, W2), color='blue', thickness=3 )
p += text( r'$[ 0.15, -1/\tanh(0.15) ]$', (2,2), fontsize=16,
horizontal_alignment='left', color='blue' )
p += plot( -1/np.tanh(x), x, 0.15, 4, color='cyan', thickness=4, linestyle='--'
)
p += text( r'$-1/\tanh(\tau-\tau_0)$', (2,1), fontsize=16,
horizontal_alignment='left', color='cyan' )

p.axes_labels( [r'$\tau - \tau_0$', r'$w$'] )
p.show( gridlines=True, figsize=[6,4] )

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