Aufgabe 13 - Fallender Stein

Clemens Ibrom

May 2024

Teil 1 - Die Theorie

Wir betrachten in der Aufgabe einen im Gravitationsfeld der Erde fallenden Stein. Dabei gehen wir ein Mal von einer konstanten Beschleunigung, und ein Mal von einer höhenabhängigen Beschleunigung aus. Die konstante Beschleunigung ist gegeben durch:

$$a_y = g = -G\frac{M_E}{R_E} \tag{1}$$

wo $G = 6.67430 \cdot 10^{-11}\,\frac{\mathrm{Nm}^2}{\mathrm{kg}^2}$ die Gravitationskonstante [1] ist, $M_E = 5.9722 \cdot 10^{24}\mathrm{kg}$ die Erdmasse [2] ist und $R_E = 6.3781 \cdot 10^6\mathrm{m}$ der Radius am Äquator [3] ist. Der Einfachheit halber nähern wir die Erde mit einer Sphäre, weil warum nicht. Die variierende Beschleunigung ist gegeben durch

$$a_y = -G\frac{M_E}{y^2}. \tag{2}$$

Hier ist $y = y(t)$ die Koordinate von dem Stein. Wir setzten später $y = R_E + h$, wo h die Fall Höhe ist.

Die Bewegungsgleichung

Die Bewegungsgleichung für den fallenden Stein mit Masse m, ohne Reibung ist

$$m\ddot{y} = -ma_y.$$

Hier ist a_y wie in 1 oder 2. Sie kann in ein System von DGLs 1. Ordnung umgewandelt werden in dem wir den Vektor $\vec{y}(t)$ definieren durch:

$$\vec{y}(t) = \begin{pmatrix} y(t) \\ v_y(t) \end{pmatrix}. \tag{3}$$

[1] von `https://en.wikipedia.org/wiki/Gravitational_constant`, am 13.05.24 um 16:30 abgelesen

[2] von `https://en.wikipedia.org/wiki/Earth`, am 13.05.24 um 16:30 abgelesen

[3] von `https://en.wikipedia.org/wiki/Earth`, am 13.05.24 um 16:30 abgelesen

Die Ableitung von 3 bzgl. der Zeit ist dann:

$$\dot{\vec{y}}(t) = \begin{pmatrix} v_y(t) \\ a_y(t) \end{pmatrix}. \tag{4}$$

für das Euler-Verfahren diskretisieren wir die Zeit und führen die Notation $y_i = y(t_i)$ ein. Zusätzlich nähern wir die Ableitung mit der Vorwärts-Differenz und benutzten $t_{i+1} = t_i + \Delta t_i$ und erhalten mit 4

$$\frac{d\vec{y_i}}{dt_i} =: d\vec{y_i} \approx \frac{\vec{y}_{i+1} - \vec{y_i}}{\Delta t} = \begin{pmatrix} \dot{y_i} \\ a_{y_i} \end{pmatrix} \tag{5}$$

Nun nutzen wir aus, dass im Euler-Verfahren der Zustandsvektor $\vec{y_i}$ bekannt ist und somit auch $\dot{y_i}$ um 5 nach y_{i+1} umzuformen:

$$\vec{y}_{i+1} = \vec{y_i} + d\vec{y_i} \cdot \Delta t. \tag{6}$$

Diese Vektorielle Gleichung wird komponentenweise implementiert. Die $d\vec{y_i}$ werden mit Hilfe von 5 und den Ausdrücken für die Beschleunigung berechnet. Dabei ist $\vec{y_i}$ als Array gegeben und die Ableitung ist dann:

$$d\vec{y_i}[0] = \vec{y_i}[1] \tag{7}$$

$$d\vec{y_i}[1] = -a_y. \tag{8}$$

Im Code sind das die Funktionen

```
void ODE_falling_stone_const(double t, const double y[], double dy[], int dim, void *params)
```

```
void ODE_falling_stone_varying(double t, const double y[], double dy[], int dim, void *params)
```

Die übergebenen Parameter sind wie in der Aufgabe beschrieben. Mit Methoden um die dy_i zu berechnen kann nun der Euler Step implementiert werden. Hier unterscheidet sich meine Deklaration von der Vorgeschlagenen, da ich auch das Array dy[] übergebe:

```
void euler_step(double t, double dt, double y[], double dy[], ODE_FUNC ode_func, int dim, void* params)
```

Das mache ich, weil ich zusätzlich zu typedef void ODE FUNC auch ein typedef void STEPPER definiere um die Simulation als eigen stehende Methode außerhalb von main definiere und dort das Array dy[] übergebe um alles zusammen zu haben. Hier ist der Code für die Simulation Methode:

```
void simul(double t, double dt, double y[], double dy[], ODE_FUNC ode_func,
    STEPPER stepper, int dim, void* params){
    while(y[0] >= Rad_E){
        stepper(t,dt,y,dy,ode_func,dim,params);
        t += dt;
    }
}
```

Die Funktion simul führt einen Euler-Verfahren Schritt aus bis der Stein auf der Erdoberfläche ankommt, d.h. $y[0] = R_E$ (im Code ist $R_E = Rad_E$, da es bei M_E gemeckert hat, dass es schon eine solche benannte Konstante irgendwo gibt und dann habe ich alle umbenannt um sicher zu sein).

Teil 5 - Fall aus 200km

Bei dem Fall aus 200 km werden die Geschwindigkeiten beim Aufprall bei konstanter und Höhen-abhängiger Beschleunigung erfragt. Zusätzlich, als sanity check, habe ich die analytisch berechneten Werte ausgeben lassen. Alle Werte sind in m/s.

```
const a; v at impact: 1979.7425119304746204
const a; v analytical at impact is: 1979.7424935827741592
var a; v at impact: 1949.4142400814926077
var a; v_analytical at impact is: 1949.4142274398934660
ratio von v_const/v_var: 1.0155576332753751
```

Wie man sehen kann, sind die Werte aus der Simulation sehr nah aneinander, und an der analytischen Lösung. Da das Ratio von v_{const}/v_{var} größer als 1 ist, beschleunigt die Konstante Erdbeschleunigung den Stein mehr.

Teil 6 - Fall aus 20m

Im letzten Teil fällt der Stein aus 20m Höhe und die Simulation wird für 600 logaritmisch-gleichverteilte Werte von dt durchgeführt. Dann wurde der Betrag vom Fehler zur analytischen Lösung in Abhängigkeit von dt geplottet. Das Ergebnis ist sehr verrauscht, aber die maximale Abweichung ist nicht sehr groß:

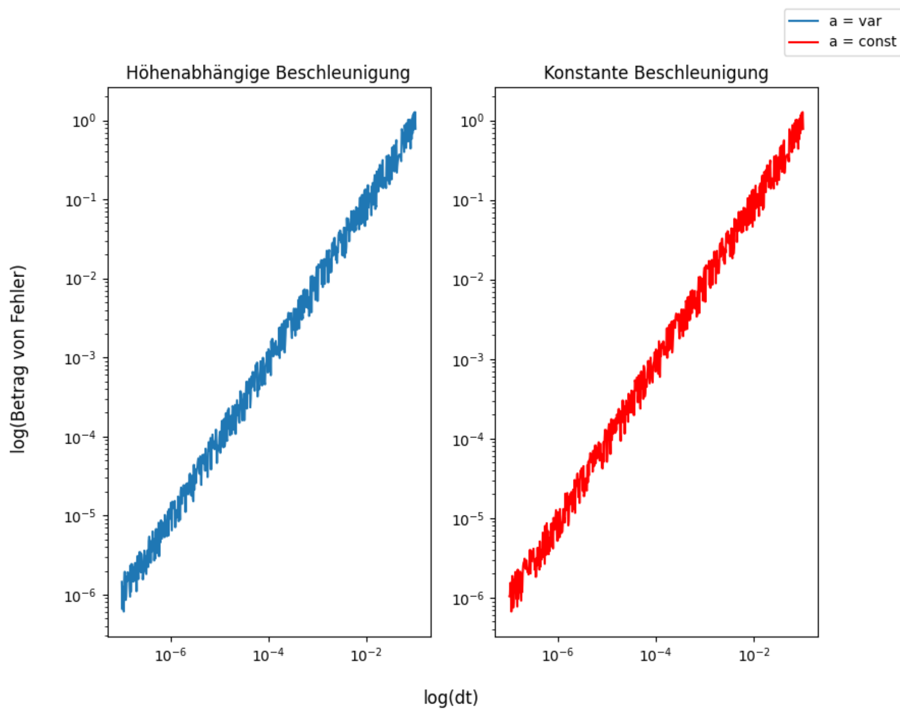

Figure 1: Log-Log Plot von Betrag von Fehler der numerischen Lösung

Konvergenzverhalten

Das Konvergenzverhalten ist hier durch die Steigung von Geraden die zu den Daten gefittet werden gegeben. Hier wird scipy.optimize.curve_fit benutzt um eine lineare Funktion in die Daten zu fitten. Die gefitteten Geraden sind: Hier

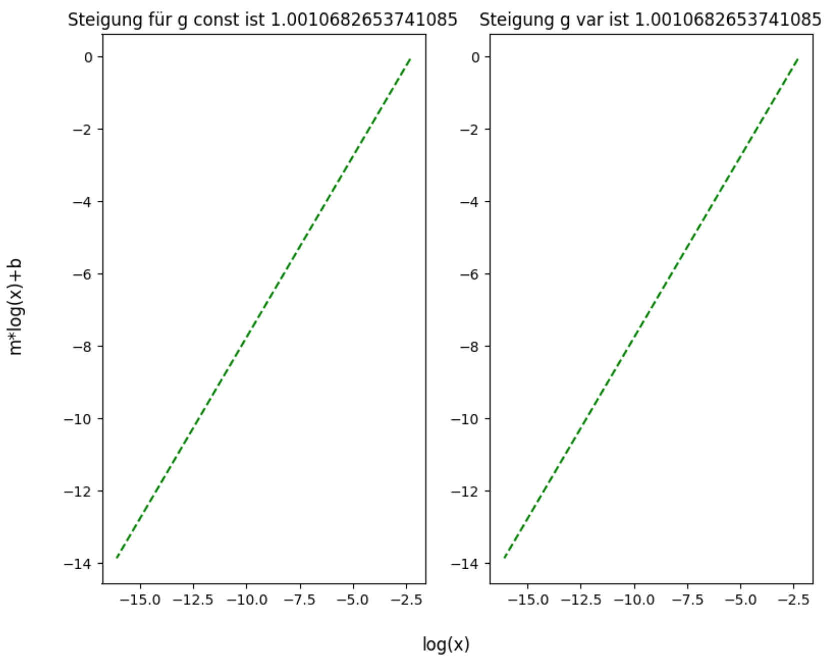

Figure 2: Zum Log-Log Plot Gefittete Geraden

ist zu bemerken, dass die Steigung gleich für beide Beschleunigungen ist. Die Konvergenz-Ordnung ist definiert als der Exponent p in der Abschätzung:

$$\max_{dt} \mid \text{num\_sol}(dt) \; - \; \text{analytic\_sol}(dt) \; \mid \leq C \cdot (dt)^p.$$

Da wir Log-Log Plots betrachten wird der Exponent p zu Faktor vor $log(dt)$, also zu dem m in 2. Das bedeutet, dass wir eine Konvergenz-Ordnung von 1.0010682653741085 erreicht haben. Diese ist unabhängig von der Beschleunigung, wie man in 2 sehen kann.