Georg-August-Universität Göttingen
Institut für Theoretische Physik
Prof. Dr. F. Heidrich-Meisner
Dr. E. Bothmann
S. L. Villani
SoSe 2021

Übungen zur Vorlesung
Computergestütztes Wissenschaftliches Rechnen
Blatt 7

Lernziele dieses Übungsblattes

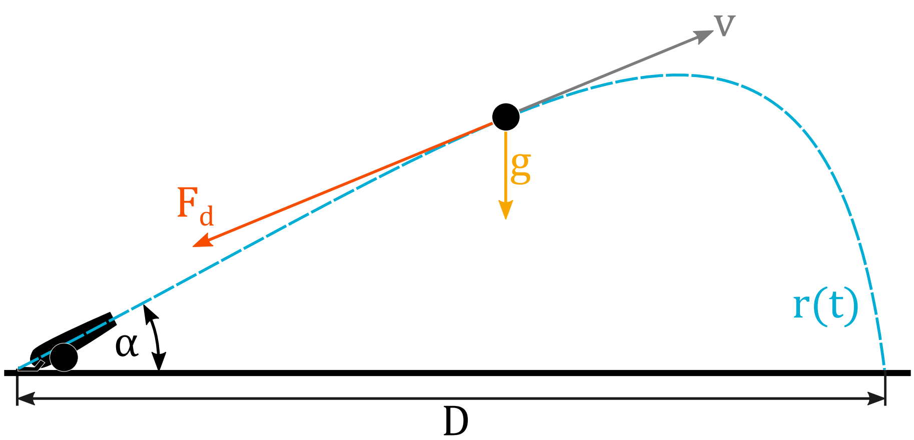
- Simulation der Trajektorie eines analytisch nicht geschlossen lösbaren Problems: schiefer Wurf mit Luftwiderstandskraft.
- Aufstellen und Implementieren der Bewegungsgleichung, Berechnung von Trajektorien mithilfe des RK2-Algorithmus.
- Allgemeine Herangehensweisen an Optimierungsprobleme: Interpolation und Formulierung als Nullstellensuche.
- Implementierung komplexerer Programmstrukturen.

Aufgabenmodus

Die Aufgabe dieser Woche verfügt über kein gesondertes Tutorial. Sie sollten sie mit Ihren bisherigen Kenntnissen bearbeiten können.

Auf diesem Blatt kommen keine neuen Konzepte vor, und der Umfang sollte auch geringer sein als bei den anderen Blättern. Dies gibt Ihnen die Gelegenheit, eventuelle Rückstände aufzuholen und/oder Ihre bisherige Arbeit zu sortieren und z.B. Ihre Numerik-Bibliothek ordentlich zu formatieren und zu kommentieren.

Übungsaufgabe 19



Auf diesem Arbeitsblatt führen wir einige der erarbeiteten Algorithmen und Funktionen der letzten Wochen zusammen, um ein komplexes Optimierungsproblem zu lösen. Wir schließen damit das Themengebiet der gewöhnlichen Differentialgleichungen aus dem Kontext der klassischen Physik ab.

Ziel ist es, den optimalen Abschusswinkel α_{opt} einer Kanonenkugel zu berechnen, so dass diese die größtmögliche horizontale Strecke D_{opt} zurücklegt. Wegen der Nicht-Linearität im Luftwiderstand ist dieses Problem analytisch nicht lösbar.

Sie benötigen folgende physikalische Parameter:

- $g = 9{,}81\,\mathrm{m/s^2}$ – Gravitationsbeschleunigung
- $c_w = 0{,}47$ – Widerstandsbeiwert einer glatten Kugel
- $\rho = 1{,}2041\,\mathrm{kg/m^3}$ – Dichte der Luft
- $R = 0{,}055\,\mathrm{m}$ – Radius der Kanonenkugel
- $v_0 = 250\,\mathrm{m/s}$ – Anfangsgeschwindigkeit der Kanonenkugel
- $m = 5{,}5\,\mathrm{kg}$ – Masse der Kanonenkugel.

Für die Startgeschwindigkeit ergibt sich eine Reynolds-Zahl von $\mathrm{Re} = 10^6$. Wir nehmen an, diese sei während der gesamten Simulation konstant, so dass auch c_w als Konstante gelte.

Die zum Luftwiderstand gehörende Kraft wirkt dem Geschwindigkeitsvektor $\boldsymbol{v}$ entgegengesetzt auf die Kanonenkugel (und verlangsamt sie so), mit dem Betrag

$$F_d = \frac{\rho v^2}{2} \cdot A \cdot c_w.$$

Als Referenzfläche $A = \pi R^2$ dient die frontal angeströmte Querschnittsfläche der Kanonenkugel.

Die Kugel wird aus einer Höhe von $y_0 = 0\,\mathrm{m}$ abgefeuert. Ihre Trajektorie ende, sobald Sie wieder den ebenen Boden erreicht. Sie dürfen (außer für die Berechnung des Luftwiderstandes) die Kugel als singulären Massenpunkt ohne Ausdehnung modellieren.

Aufgabe 19.1 *Aufstellen der Differentialgleichung*

Implementieren Sie die Funktion

```
int cannonball_ode(double t, const double y[],
                   double f[], void *params)
{
  ...
}
```

Stellen Sie hierfür das Differentialgleichungssystem auf, das die Trajektorie der Kanonenkugel unter Berücksichtigung aller wirkenden Kräfte beschreibt. Überlegen Sie sich eine geeignete Konvention für den Aufbau Ihres Zustandsvektors $\mathbf{y}$ und implementieren Sie dann in `cannonball_ode` die Systemfunktion $\mathbf{F}$.

Hinweis: Benutzen Sie keine trigonometrischen Funktionen! Projizieren Sie den Luftwiderstands-Kraftvektor stattdessen auf den momentanen Geschwindigkeitsvektor, um die x- und y- Komponenten zu erhalten. Bestimmen Sie zunächst den Betrag F_d der

Luftwiderstandskraft. Die beiden Komponenten des vektoriellen Luftwiderstands ergeben sich dann durch

$$F_{d,x} = -F_d \frac{v_x}{|\boldsymbol{v}|}, \qquad F_{d,y} = -F_d \frac{v_y}{|\boldsymbol{v}|}.$$

Aufgabe 19.2 *Simulation von Trajektorien*

Simulieren Sie das System mit den eingangs definierten Parametern mit Hilfe eines RK2-Algorithmus, der die Bewegung der Kugel als Funktion der Zeit ergeben soll. Berechnen Sie dabei 10 verschiedene Trajektorien für Abschusswinkel im Intervall $[0°, 90°]$ und plotten Sie diese Trajektorien in ein gemeinsames x-y-Diagramm.

Was ist die größte horizontale Entfernung D, die eine Ihrer 10 Kugeln erreicht?

Wie verändert sich die Form der Trajektorien, wenn die Kugel nur eine Masse von 50 g hat?

Aufgabe 19.3 *Distanzfunktion*

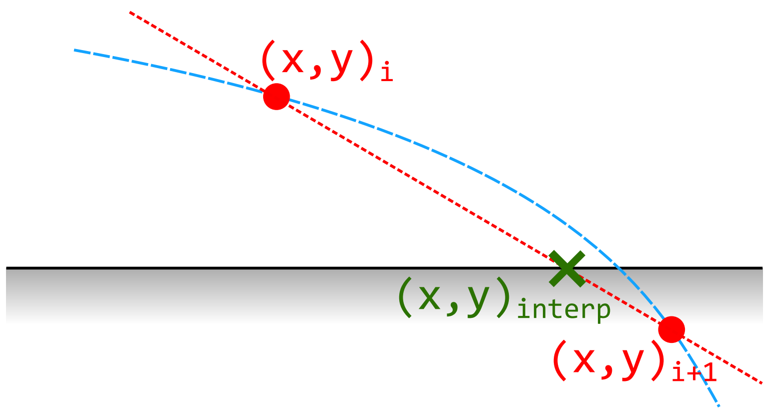
Schreiben Sie die Distanzfunktion

```
double cannon_distance(double angle)
{
  ...
}
```

Die Funktion nimmt einen Abschusswinkel `angle` entgegen und gibt die horizontale Distanz zurück, die eine Kanonenkugel unter diesem Abschusswinkel zurücklegt, bevor sie den Boden wieder erreicht. Verwenden Sie die eingangs definierten Parameterwerte.

Verwenden Sie für die Zeitintegration erneut das RK2-Verfahren. Ein Zeitschritt von 0,1 s ist ausreichend.

Es ist wichtig, dass Sie den exakten Schnittpunkt der Trajektorie mit dem Boden bestimmen. Benutzen Sie den letzten und den vorletzten Zeitschritt der Integration und interpolieren Sie den Durchgang durch den Boden linear:



$$D \approx x_{interp} = \frac{x_i \cdot y_{i+1} - y_i \cdot x_{i+1}}{y_{i+1} - y_i}.$$

Wenn Sie den Schnittpunkt nicht interpolieren, wäre die Distanzfunktion mit einem Diskretisierungsrauschen behaftet, das die nächste Teilaufgabe erheblich erschweren würde.

Plotten Sie die Distanz, die die Kanonenkugel zurücklegt für Abschusswinkel im Interval $[0°, 90°]$. Plotten Sie mindestens 200 verschiedene Winkel. Bei etwa welchem Winkel fliegt die Kugel am weitesten?

Aufgabe 19.4 *Optimierung mit dem Newton-Raphson-Verfahren*

Bestimmen Sie mit Ihrer Implementierung des Newton-Raphson-Verfahrens aus Ihrer `my_numerics`-Bibliothek die maximale horizontale Flugweite D_{opt} und den optimalen Abschusswinkel α_{opt}.

Das Maximum ist dadurch charakterisiert, dass die Ableitung der Distanzfunktion gerade verschwindet, d.h. wir suchen den Nulldurchgang der Ableitung von `cannon_distance()`. Berechnen Sie also zunächst die Ableitung mit `diff()` und verkapseln Sie die Berechnung in einer neuen Funktion

```
double D_cannon_distance(double angle) { ... }.
```

(Für Experten: das ist notwendig, weil C keine anonymen Funktionen (lambdas) unterstützt.)

Bestimmen Sie dann den Nulldurchgang mit `find_root()`.

Hinweis: Eine Winkel-Schrittweite von 10^{-3} ist für alle auftretenden Ableitungen geeignet. Orientieren Sie sich an Ihrem Plot aus der vorherigen Teilaufgabe, um einen geeigneten Startwert für die Iteration zu finden.

Selbsttest

- Wie bestimmt man, ob eine Differenzialgleichung (z.B. eine Bewegungsgleichung) analytisch lösbar ist? Wann entscheidet man sich für eine numerische Simulation des Problems?
- Empfiehlt es sich, eine numerische Behandlung einer analytisch lösbaren Differenzialgleichung vorzunehmen? Wann kann das ggf. sinnvoll sein?
- Wie gehe ich vor beim Aufstellen einer Differenzialgleichung, die ich in einem Programm implementieren möchte?
- Wie entscheide ich, welchen Algorithmus ich zur Behandlung der Differenzialgleichung nutze?
- Wie entscheide ich, ob die Simulation erfolgreich war, oder ob ihr Fehler zu groß war?
- Was ist die einfachste Vorgehensweise zum Interpolieren von Datenpunkten?
- Kann man ein Optimierungsproblem immer in eine Nullstellensuche umformulieren?