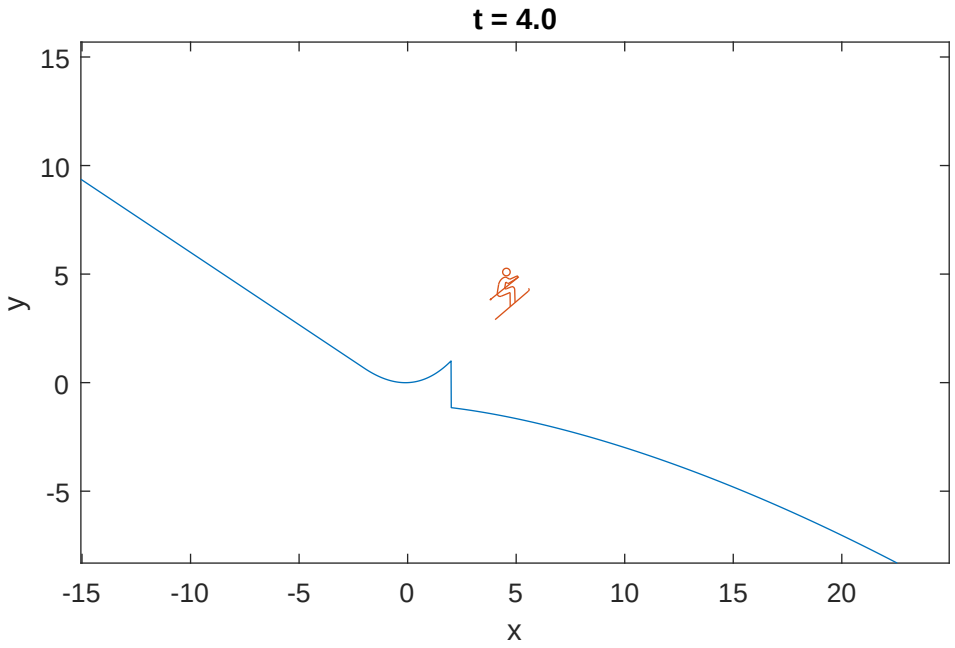
Skisprung

In diesem Projekt werden Sie einen Skisprung mithilfe von Differentialgleichungen modellieren. Für die numerische Berechnung wird Matlab verwendet. Das Projekt besteht aus drei Teilen: Im ersten Teil werden Sie das Verhalten eines Objekts simulieren, welches am Boden haftet, sich also nur hin und her bewegen kann, nicht aber auf und ab. Im zweiten Teil werden Sie das Verhalten eines Objekts im freien Fall simulieren. In beiden Fällen ignorieren wir jegliche Reibung, was das Problem vereinfacht und uns erlaubt, Resultate mithilfe der Energieerhaltung zu überprüfen. Im dritten Teil werden beide Modelle kombiniert: Abhängig davon, ob man gerade Anlauf nimmt oder durch die Luft fliegt, wird das entsprechende Modell verwendet. Zusätzlich werden Bedingungen benötigt, um zu entscheiden, wann von dem einem in das andere Modell gewechselt wird. Zum Abschluss des Projekts sollen Sie einen kurzen Bericht (2-3 Seiten) verfassen und abgeben.



1 Teil 1: Auf dem Boden

Wir betrachten ein Objekt, das immer auf dem Boden bleibt. Der Boden kann ein beliebiges Profil haben, er muss also nicht horizontal sein, sondern kann unterschiedliche Neigungen haben. Das Objekt wird von der Schwerkraft beeinflusst, welche es in Abhängigkeit von der Neigung des Bodens beschleunigt. In diesem Abschnitt leiten wir eine Differentialgleichung her, um diese Wechselwirkung zu beschreiben, wobei der Einfachheit halber Reibung ignoriert wird.

1.1 Grundlegende Differentialgleichung

Nehmen wir an, dass sich unser Objekt entlang einer Funktion $f : \mathbb{R} \to \mathbb{R}$, welche das Bodenprofil beschreibt, bewegt. Sei

$$\mathbf{P}(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix} \in \mathbb{R}^2 \tag{1}$$

die Position des Objekts, $\mathbf{v}(t) \in \mathbb{R}^2$ der Geschwindigkeitsvektor und $v(t) = \|\mathbf{v}(t)\| \geq 0$ die Geschwindigkeit des Objekts zum Zeitpunkt $t \geq 0$. Wie in der Physik üblich, verwenden wir die Notationen $\dot{x} = \frac{\partial x}{\partial t}$ für Zeitableitungen und $f' = \frac{\partial f}{\partial x}$ für Ortsableitungen. Es gilt

$$\mathbf{v}(t) = \dot{\mathbf{P}}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} \dot{x}(t) \\ f'(x(t))\dot{x}(t) \end{bmatrix} = \dot{x}(t) \begin{bmatrix} 1 \\ f'(x(t)) \end{bmatrix} = \frac{v(t)}{\sqrt{1 + f'(x(t))^2}} \begin{bmatrix} 1 \\ f'(x(t)) \end{bmatrix}, \tag{2}$$

wobei für die letzte Gleichung $v(t) = \|\mathbf{v}(t)\|$ verwendet wurde. Wir können nun die Position $\mathbf{P}(t)$ unseres Objekts mithilfe seiner Geschwindigkeit $v(t)$ berechnen:

$$\begin{cases} \dot{x}(t) = \frac{v(t)}{\sqrt{1+f'(x(t))^2}}, \\ y(t) = f(x(t)). \end{cases} \tag{3}$$

1.2 Tangente und Normale

Für jeden Vektor

$$\mathbf{a} = \begin{bmatrix} \mathbf{a}_x \\ \mathbf{a}_y \end{bmatrix} \in \mathbb{R}^2 \tag{4}$$

sei

$$\hat{\mathbf{a}} = \frac{\mathbf{a}}{\|\mathbf{a}\|} \in \mathbb{R}^2 \tag{5}$$

die normierte Vektorform von $\mathbf{a}$. Sei $\mathbf{t}(x) \in \mathbb{R}^2$ der Tangentenvektor und $\mathbf{n}(x) \in \mathbb{R}^2$ der Normalenvektor von f an der Stelle $x \in \mathbb{R}$. Es gilt

$$\mathbf{t}(x) = \begin{bmatrix} 1 \\ f'(x) \end{bmatrix} \quad \text{und} \quad \mathbf{n}(x) = \begin{bmatrix} f'(x) \\ -1 \end{bmatrix}. \tag{6}$$

Gemäss der vorherigen Notation definiere $\hat{\mathbf{t}}$ und $\hat{\mathbf{n}}$ als die normierten Vektorformen von $\mathbf{t}$ und $\mathbf{n}$. Aus (2) schliessen wir, dass die Geschwindigkeit immer in dieselbe Richtung wie die Tangente von f zeigt:

$$\mathbf{v}(t) = \frac{v(t)}{\sqrt{1+f'(x(t))^2}} \begin{bmatrix} 1 \\ f'(x(t)) \end{bmatrix} = v(t)\hat{\mathbf{t}}(x(t)). \tag{7}$$

Somit können wir (3) vereinfachen als

$$\begin{cases} \dot{x}(t) = v(t)\hat{\mathbf{t}}_x(x(t)), \\ y(t) = f(x(t)). \end{cases} \tag{8}$$

1.3 Schwerkraft

Wir nehmen nun an, dass unser Objekt von der Schwerkraft in negativer y-Richtung beeinflusst wird. Dazu sei

$$\mathbf{g} = \begin{bmatrix} 0 \\ -g \end{bmatrix} \in \mathbb{R}^2 \tag{9}$$

der nach unten gerichtete Schwerkraftsvektor, wobei $g \approx 9,8$ die Gravitationskonstante der Erde ist. Beachten Sie, dass in dieser Herleitung MKSA-Einheiten nicht explizit notiert werden. Auf einer schiefen Ebene wird der Schwerkraftsvektor in zwei Komponenten zerlegt, eine, die parallel zur Ebene wirkt, und eine, die senkrecht zur Ebene wirkt. Dies ist in Abbildung 1 zu sehen. Da wir Reibungseffekte ignorieren, benötigen wir nur die parallele Komponente. Es gilt

$$\mathbf{g}_{||} = \frac{\mathbf{g}^\top \mathbf{t}}{\mathbf{t}^\top \mathbf{t}} \mathbf{t} = (\mathbf{g}^\top \hat{\mathbf{t}})\, \hat{\mathbf{t}}. \tag{10}$$

Man beachte, dass $g_{||} = \left\|\mathbf{g}_{||}\right\| = \left|\mathbf{g}^\top \hat{\mathbf{t}}\right|$ unser Objekt beschleunigt, wenn die Steigung von f bei x negativ ist, und das Objekt abbremst, wenn die Steigung positiv ist. Dies lässt sich mit der folgenden Differentialgleichung ausdrücken:

$$\dot{v}(t) = \mathbf{g}^\top \hat{\mathbf{t}}(x(t)) = -g\, \hat{\mathbf{t}}_y(x(t)). \tag{11}$$

Kombiniert man (8) und (11) und fügt noch Anfangsbedingungen hinzu, erhält man

$$\begin{cases} \dot{x}(t) = v(t)\, \hat{\mathbf{t}}_x(x(t)) & \forall t > 0, \\ \dot{v}(t) = -g\, \hat{\mathbf{t}}_y(x(t)) & \forall t > 0, \\ x(0) = x_0, \\ v(0) = v_0, \end{cases} \tag{12}$$

und $y(t) = f(x(t))$ für alle $t \geq 0$, wobei x_0 die anfängliche x-Position des Objekts und v_0 die Anfangsgeschwindigkeit des Objekts ist.

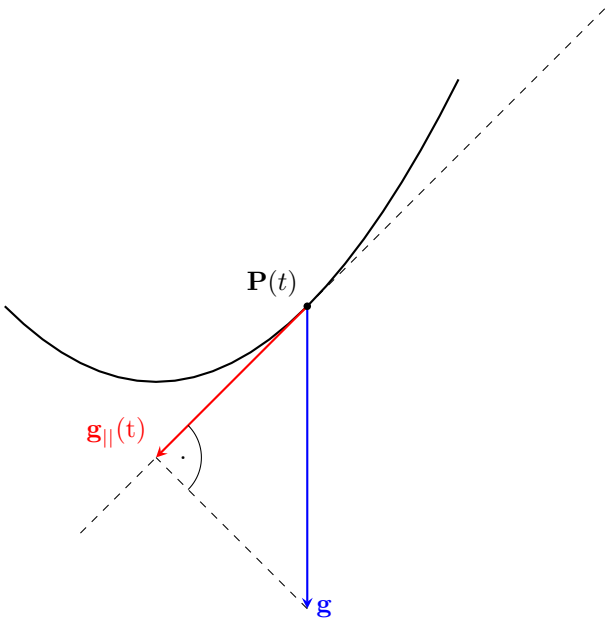


Abbildung 1: Kräfte an der schiefen Ebene.

1.4 Numerische Lösung

Wir werden das explizite Euler-Verfahren verwenden, um (12) numerisch zu lösen: Für eine Differentialgleichung der Form

$$\begin{cases} \dot{x}(t) = \Phi(t, x(t)) \\ x(0) = x_0 \end{cases} \tag{13}$$

mit dem Anfangswert x_0 berechnen wir x_k nach der Vorschrift

$$x_{k+1} := x_k + \Delta t\, \Phi(t_k, x_k) \quad \text{für} \quad k = 0, 1, \ldots, \tag{14}$$

wobei $\Delta t > 0$, $t_k = k\Delta t$ und $x_k \approx x(t_k)$ für alle $k \in \mathbb{N}$.

1.5 Aufgaben zu Teil 1

- Zeigen Sie, dass (12) den Energieerhaltungssatz nicht verletzt, und nehmen Sie den Beweis in Ihren Projektbericht auf. Dieser besagt, dass die Summe von kinetischer und potentieller Energie konstant bleibt:
$$E_{ges}(t) = E_{kin}(t) + E_{pot}(t) = \frac{1}{2}mv(t)^2 + mgy(t) \equiv \text{konstant}. \tag{15}$$
Beachten Sie, dass die Masse m des Objekts für diese Rechnung nicht relevant ist. Tipp: Leiten Sie die Gesamtenergie $E_{ges}(t)$ nach t ab und zeigen Sie, dass das Resultat unabhängig von t immer gleich 0 ist.
- Schreiben Sie ein Matlab-Programm, welches die Differentialgleichung (12) mithilfe des expliziten Euler-Verfahrens im Zeitintervall $[0, T]$ löst. Hier sei $T = 10$. Verwenden Sie dafür die Funktion $f(x) = x^2$ und die Anfangsbedingungen $x_0 = -2$ und $v_0 = 0$.
- Überprüfen Sie, dass auch die numerische Approximation dem Energieerhaltungssatz folgt. Das Objekt sollte genau zwischen $x = -2$ und $x = 2$ hin und her pendeln und immer auf dieselbe Höhe zurückkehren. Plotten Sie dazu die Bahn des Objekts und die Anfangshöhe des Objekts in dasselbe Diagramm. Die maximale Höhe des Objekts sollte nie (bis auf numerische Fehler) die Anfangshöhe überschreiten. Notieren Sie in Ihrem Bericht den Fehler zwischen maximaler Höhe des Objekts und Starthöhe für $\Delta t = 10^{-2}$, $\Delta t = 10^{-3}$ und $\Delta t = 10^{-4}$.

2 Teil 2: In der Luft

In diesem Kapitel werden wir ein Objekt im freien Fall betrachten. Auch hier werden wir wieder Reibung ignorieren.

2.1 Differentialgleichung

Nehmen wir an, ein Objekt bewegt sich nur unter Einwirkung der Schwerkraft durch die Luft. Die Position $\mathbf{P}(t) \in \mathbb{R}^2$ des Objekts erfüllt die Differentialgleichung

$$\begin{cases} \ddot{\mathbf{P}}(t) = \mathbf{g} & \forall t > 0, \\ \mathbf{P}(0) = p_0, \\ \dot{\mathbf{P}}(0) = \dot{p}_0, \end{cases} \tag{16}$$

wobei $p_0 \in \mathbb{R}^2$ die Anfangsposition, $\dot{p}_0 \in \mathbb{R}^2$ die Anfangsgeschwindigkeit und

$$\mathbf{g} = \begin{bmatrix} 0 \\ -g \end{bmatrix} \in \mathbb{R}^2 \tag{17}$$

der nach unten gerichtete Schwerkraftsvektor ist. Alternativ kann man (16) auch in Abhängigkeit von x und y schreiben:

$$\begin{cases} \ddot{x}(t) = 0 & \forall t > 0, \\ \ddot{y}(t) = -g & \forall t > 0, \\ x(0) = x_0, \\ y(0) = y_0, \\ \dot{x}(0) = \dot{x}_0, \\ \dot{y}(0) = \dot{y}_0. \end{cases} \tag{18}$$

2.2 Numerische Lösung

Hier verwenden wir das Leapfrog-Verfahren, um (18) numerisch zu lösen: Für eine Differentialgleichung der Form

$$\begin{cases} \ddot{x}(t) = \Phi(t, x(t)) \\ x(0) = x_0 \\ \dot{x}(0) = \dot{x}_0 \end{cases} \tag{19}$$

definieren wir den Wert x_1 als

$$x_1 := x_0 + \Delta t \dot{x}_0 + \frac{\Delta t^2}{2} \Phi(0, x_0) \approx x(\Delta t) \tag{20}$$

und berechnen dann x_k nach der Vorschrift

$$x_{k+1} := 2x_k - x_{k-1} + \Delta t^2 \Phi(t_k, x_k) \quad \text{für} \quad k = 1, 2, \ldots, \tag{21}$$

wobei $\Delta t > 0$, $t_k = k\Delta t$ und $x_k \approx x(t_k)$ für alle $k \in \mathbb{N}$.

2.3 Aufgaben zu Teil 2

- Berechnen Sie die Lösung von (18) per Hand und fügen sie diese in den Projektbericht ein. Integrieren Sie dafür zuerst $\ddot{x}$ und $\ddot{y}$ zweimal in der Zeit. Sie erhalten x und y in Abhängigkeit von jeweils zwei Konstanten. Wählen Sie die Konstanten so, dass die Anfangsbedingungen $x(0) = x_0$, $y(0) = y_0$, $\dot{x}(0) = \dot{x}_0$ und $\dot{y}(0) = \dot{y}_0$ erfüllt sind.
- Schreiben Sie ein Matlab-Programm, welches (18) mithilfe des Leapfrog-Verfahrens im Zeitintervall $[0, T]$ löst, mit $T = 1$. Verwenden Sie dafür die Anfangsbedingungen $x_0 = 0$, $y_0 = 1$, $\dot{x}_0 = 2$ und $\dot{y}_0 = 1$. Berechnen Sie nun den maximalen Fehler zwischen der numerischen und der exakten Lösung (einmal für x und einmal für y) zu allen Zeitpunkten t_k für $\Delta t = 10^{-3}$. Das Ergebnis sollte im Projektbericht vorhanden sein.

3 Teil 3: Kombination der Modelle

In den letzten zwei Abschnitten haben wir Differentialgleichungen sowohl für ein Objekt, das sich auf dem Boden bewegt, als auch für ein Objekt, das durch die Luft fliegt, hergeleitet. Ausserdem haben wir zwei Algorithmen gesehen, mit denen wir diese Gleichungen numerisch lösen können. Wir wollen jedoch ein einziges Programm implementieren, das automatisch zwischen beiden Modi umschaltet. Dies kann mit den Bedingungen, die wir in diesem Abschnitt vorstellen, erreicht werden.

Wir werden die folgenden Bezeichnungen verwenden:

Bodenschritt	$\equiv$ ein Euler-Schritt für die Bodendifferentialgleichung (12)
Luftschritt	$\equiv$ ein Leapfrog-Schritt für die Luftdifferentialgleichung (18)

3.1 Boden zu Luft

Nach jedem Bodenschritt wird ein "künstlicher"Luftschritt berechnet und dann geprüft, ob sich das Objekt über dem Boden befindet. Wenn dies der Fall ist, wechseln wir in den Luftmodus.

Algorithm 1: Boden zu Luft

```
// Bodenschritt
```
$x_{k+1} = x_k + \Delta t \dots$
$y_{k+1} = f(x_{k+1})$
$v_{k+1} = v_k + \Delta t \dots$

```
// Künstlicher Luftschritt
```
$x_{\text{artificial}} = 2x_{k+1} - x_k + \Delta t^2 \dots$
$y_{\text{artificial}} = 2y_{k+1} - y_k + \Delta t^2 \dots$

```
// Ist das Objekt über dem Boden?
```
if $y_{\text{artificial}} > f(x_{\text{artificial}})$ **then**
```
  // Wechsele in den Luftmodus
```
end

3.2 Luft zu Boden

Nach jedem Luftschritt wird getestet, ob sich das Objekt unterhalb des Bodens befindet. Wenn dies der Fall ist, wird das Programm beendet.

Algorithm 2: Luft zu Boden

```
// Luftschritt
```
$x_{k+1} = 2x_k - x_{k-1} + \Delta t^2 \dots$
$y_{k+1} = 2y_k - y_{k-1} + \Delta t^2 \dots$

```
// Ist das Objekt unter dem Boden?
```
if $y_{k+1} < f(x_{k+1})$ **then**
```
  // Beende das Programm
```
end

3.3 Aufgaben zu Teil 3

- Erstellen Sie ein Matlab-Programm, welches eine Skispringerin oder einen Skispringer auf einer Schanze simuliert. Dazu starten Sie mit Ihrem Progamm aus Teil 1. Wie zuvor iterieren wir über alle Zeitpunkte t_k und berechnen in jedem Schritt die Werte x_{k+1} und y_{k+1}. Führen Sie nun eine Boolsche Variable *onTheGround* ein, welche den aktuellen Modus (Luft oder Boden) darstellt. Abhängig vom aktuellen Wert dieser Variable (*true* oder *false*) wird in jeder Interation entweder ein Euler-Schritt für die Boden-DGL oder ein Leapfrog-Schritt für die Luft-DGL ausgeführt. Ausserdem soll in jeder Iteration überprüft werden, ob der Modus und damit der Wert der Variable *onTheGround* gewechselt werden muss. Verwenden Sie hierzu die Pseudo-Codes aus diesem Kapitel.

  Die Funktion *initializeRamp*, welche auf ADAM zur Verfügung steht, stellt die Schanze dar und berechnet f und $\frac{\partial f}{\partial x}$. Verwenden Sie als Inputs für *initializeRamp* die Werte $rampHeight = 1$ und $rampSlope = 1$. Als Anfangsbedingungen verwenden Sie $x_0 = -30$, $v_0 = 0$. Beachten Sie, dass wie zuvor $y_0 = f(x_0)$ gilt.

  Simulieren Sie Ihre Skispringerin oder Ihr Skispringer in dem Intervall $[0, T]$ mit $T = 10$ und Zeitschritt $\Delta t = 10^{-3}$ und plotten Sie die Bahn des Objekts und die Bodenfunktion in dasselbe Diagramm.

Da das Programm beim Übergang Luft-Boden terminiert wird, wird der Zeitpunkt T nicht notwendigerweise erreicht.

- Wir wollen nun die Form der Schanze experimentell optimieren. Verändern Sie dazu die Parameter *rampHeight* und *rampSlope* sodass Ihre Skispringerin oder Ihr Skispringer möglichst weit fliegt (die x-Koordinate vom Übergang Luft-Boden sollte möglichst gross sein). Nehmen Sie diese Werte in den Projektbericht auf.
- Verwenden Sie die Funktion *makeVideo*, welche auf ADAM verfügbar ist, um ein Video vom Sprung zu erstellen.

4 Projektbericht und Termine

Für dieses Projekt sollen Sie einen kurzen Projektbericht schreiben (2-3 Seiten). Dieser Projektbericht (als PDF) wird zusammen mit den Lösungen für die Aufgaben, Ihrem Code und dem Video aus Teil 3 über ADAM abgegeben. Ausserdem wird es drei Besprechungstermine geben, an denen Fragen gestellt werden können. Der Terminplan ist:

- 11.04. - Veröffentlichung
- 24.04. - Besprechung - Seminarraum 05.002, 12:15-13:00
- 08.05. - Besprechung - Seminarraum 05.002, 12:15-13:00
- 22.05. - Besprechung - Seminarraum 05.002, 12:15-13:00
- 23.06. - Abgabe
- 03.07. bis 07.07. - Projektkontrolle (in Präsenz, Termine werden später vereinbart)

5 Bonus

Dieses Kapitel ist optional.

Bisher wurde Ihr Programm terminiert, sobald das Objekt aus der Luft auf dem Boden aufkommt. Mit folgendem Pseudo-Code können Sie dafür sorgen, dass das Objekt stattdessen weiterfährt. Nun können Sie statt

Algorithm 3: Luft zu Boden - ohne Terminierung

```
// Luftschritt
x_{k+1} = 2x_k - x_{k-1} + Δt^2 ...
y_{k+1} = 2y_k - y_{k-1} + Δt^2 ...

// Ist das Objekt unter dem Boden?
if y_{k+1} < f(x_{k+1}) then
    // Berechne benötigte Variablen
    y_{k+1} = f(x_{k+1})
    v_{k+1} = 1/Δt [(x_{k+1} - x_k)   (y_{k+1} - y_k)] t̂(x_k)

    // Wechsele in den Bodenmodus
end
```

initializeRamp das Programm *initializeRandomizedGround* von ADAM verwenden, um eine Bodenfunktion mit zufällig erzeugten Hügeln zu erhalten.

Allgemeine Informationen zur Vorlesung befinden sich auf ADAM.
Zuletzt editiert am 12. April 2023.