FH GR Fachhochschule Graubünden
University of Applied Sciences

Übungsblatt LA 1

Computational and Data Science BSc
HS 2023

Lösungen

Mathematik 1

1. Aussagen über Python/Numpy

Welche der folgenden Aussagen sind wahr und welche falsch?	**wahr**	**falsch**
a) Python/Numpy wurde speziell für den Unterricht an Schulen entwickelt.	○	●
b) Python/Numpy ist die Abkürzung von "Numerical Python".	●	○
c) Python/Numpy ist ein CAS (*Computer-Algebra-Systeme*).	○	●
d) Die Kernkompetenzen von Python/Numpy sind *Numerik*, *Datenverarbeitung* und *Datenvisualisierung*.	●	○
e) Python/Numpy ist für die gängigen Betriebssysteme Windows, Mac und Linux erhältlich.	●	○

2. Terme mit Hilfe von Python/Numpy numerisch auswerten

Wir werten jeweils den Term mit Python/Numpy *numerisch* aus. Dazu implementieren wir den folgenden Code, den wir für jede Teilaufgabe ergänzen.

```
# Python initialisieren:
import numpy as np;
# Berechnungen:
...
```

a) 45.6^3 durch `45.6**3`

b) $\sqrt{3}$ durch `np.sqrt(3)`

c) $\sqrt[5]{2}$ durch `2**(1/5)`

d) $\sqrt[7]{5.654 \cdot 10^{24}}$ durch `5.654e24**(1/7)`

e) $44.5^{\frac{5}{9}}$ durch `44.5**(5/9)`

f) 4π durch `4*np.pi`

g) $\sin(5\pi/4)$ durch `sin(5*np.pi/4)`

h) $\cot(-3\pi/5)$ durch `1/tan(-3*np.pi/5)`

i) $\tan(77°)$ durch `tan(77*np.pi/180)`

j) $\arccos(-0.45)$ durch `np.arccos(-0.45)`

k) $\operatorname{arccot}(34.1)$ durch `np.arctan(1/34.1)`

l) $e^{-3.34}$ durch `np.exp(-3.34)`

m) e durch `np.exp(1)`

n) $\ln(13.2)$ durch `np.log(13.2)`

o) $\log_{10}(23'456)$ durch `np.log(23)/np.log(10)`

p) $\log_2(69.6)$ durch `np.log(69.6)/np.log(2)`

3. Skriptvorlage für Berechnungen mit Python/Numpy

Wir betrachten den folgenden Code für Python/Numpy zur Berechnung der *Hypotenuse* eines *rechtwinkligen Dreiecks* aus den beiden *Katheten* $a \approx 12.3\,\text{cm}$ und $b \approx 8.14\,\text{cm}$.

```
# Python initialisieren:
import numpy as np;
# Parameter:
a=12.3; b=8.14; pr=3; ME='cm';
# Berechnungen:
c=np.sqrt(a**2+b**2);
# Ausgabe:
print(__file__);

print(f"Seite a = {a:#.{pr}g} {ME}");
print(f"Seite b = {b:#.{pr}g} {ME}");
print(f"Seite c = {c:#.{pr}g} {ME}");
```

a) Wir implementieren den Code in Python/Numpy, speichern Sie das Skript unter einem geeigneten Dateinamen und führen es aus. Gemäss Output beträgt die *Länge* der *Hypotenuse* des *Dreiecks*

$\underline{\underline{c \approx 14.7\,\text{cm}.}}$ (1)

b) Die Zeile `print(__file__);` bewirkt die Ausgabe des Dateinamens der Skript-Datei.

c) Wir variieren den Wert des Parameters `pr` und beobachten die Wirkung dieser Variationen. Offensichtlich ist `pr` jeweils gerade die Anzahl Dezimalstellen im ausgegebenen Wert.

d) Wir modifizieren den Code für die *Katheten* mit *Längen* $a \approx 0.85\,\text{km}$ und $b \approx 234\,\text{m}$.

```
# Python initialisieren:
import numpy as np;
# Parameter:
a=0.85e3; pr_a=2; sc_a=1.0e-3; ME_a='km';\
b=234.;   pr_b=3; sc_b=1.0;    ME_b='m';\
          pr_c=2; sc_c=1.0e-3; ME_c='km';\
# Berechnungen:
c=np.sqrt(a**2+b**2);
# Ausgabe:
 print(__file__);
 print(f"Seite a = {a*sc_a:#.{pr_a}g} {ME_a}");
print(f"Seite b = {b*sc_b:#.{pr_b}g} {ME_b}");
print(f"Seite c = {c*sc_c:#.{pr_c}g} {ME_c}");
```

Gemäss Output beträgt die *Länge* der *Hypotenuse* des *Dreiecks* in diesem Fall

$\underline{\underline{c \approx 0.88\,\text{km}.}}$ (2)

4. Aussagen über Python/Sympy

Welche der folgenden Aussagen sind wahr und welche falsch?	**wahr**	**falsch**
a) Python/Sympy wurde speziell für den Unterricht an Schulen entwickelt.	○	●
b) Python/Sympy ist die Abkürzung von "Symbolic Python".	●	○
c) Python/Sympy ist ein CAS (*Computer-Algebra-Systeme*).	●	○
d) Die Kernkompetenzen von Python/Sympy sind *Numerik*, *Datenverarbeitung* und *Datenvisualisierung*.	○	●
e) Python/Sympy ist für die gängigen Betriebssysteme Windows, Mac und Linux erhältlich.	●	○

5. Elementare Algebra mit Python/Sympy

Wir betrachten den folgenden Code für Python/Sympy.

```
# Python initialisieren:
import IPython.display as dp;
import sympy as sp;
# Python konfigurieren:
sp.init_printing();
x,y,z=sp.symbols('x,y,z');
# Parameter:
p=x*x**2*(x**2-4)*(x**2-2*x-15);
# Berechnungen:
f=sp.expand(p);
g=sp.factor(p);
h=sp.simplify(p);
# Ausgabe:
dp.display(f);
dp.display(g);
dp.display(h);
```

a) Wir implementieren den Code in Python/Sympy und führen ihn aus. Im Output erhalten wir die Terme

$$60\,x^3 + 8\,x^4 - 19\,x^5 - 2\,x^6 + x^7 \tag{3}$$

$$(-5+x)\,(-2+x)\,x^3\,(2+x)\,(3+x) \tag{4}$$

$$x^3\,(-4-x^2)\,(-15-2x+x^2) \tag{5}$$

$$(-5+x)\,x^3\,(3+x)\,(-4+x^2). \tag{6}$$

b) Wir fassen die Wirkungen der drei Befehle `expand`, `factor` und `simplify` in einer Tabelle zusammen.

Befehl	Wirkung
`expand`	Vollständiges *Ausmultiplizieren*
`factor`	Vollständiges *Faktorisieren*
`simplify`	Einfaches *Vereinfachen*

(7)

Zusätzlich fällt auf, dass die Terme in jedem auftretenden *Polynom* in absteigender Reihenfolge nach *Potenzen* sortiert werden.

c) Keine Lösung verfügbar.

6. Terme Faktorisieren mit Python/Sympy

Wir *faktorisieren* jeweils den Term mit Python/Sympy. Dazu implementieren wir den folgenden Code, den wir für jede Teilaufgabe modifizieren.

```
# Python initialisieren:
import IPython.display as dp;
import sympy as sp;
# Python konfigurieren:
sp.init_printing();
...=sp.symbols('...');
# Parameter:
p=...;
# Berechnungen:
q=sp.factor(p);
# Ausgabe:
dp.display(q);
```

a) Wir betrachten den Term

$$p = x^3 - 4\,x. \tag{8}$$

Wir modifizieren den Code.

```
# Python konfigurieren:
x=sp.symbols('x');
# Parameter:
p=x**3-4*x;
```

Gemäss Ausgabe gilt

$$\underline{\underline{p = (x+2)\cdot x\cdot (x-2).}} \tag{9}$$

b) Wir betrachten den Term

$$p = 2\,xy - 3\,x^3y - 2\,y^3 + 3\,x^2y^3. \tag{10}$$

Wir modifizieren den Code.

```
# Python konfigurieren:
x,y=sp.symbols('x y');
# Parameter:
p=2*x*y-3*x**3*y-2*y**3+3*x**2*y**3;
```

Gemäss Ausgabe gilt

$$\underline{\underline{p = y\cdot \left(3\,x^2-2\right)\cdot \left(y^2-x\right).}} \tag{11}$$

c) Wir betrachten den Term

$$p = b^4 + a^3b - a^2b^2 - ab^3. \tag{12}$$

Wir modifizieren den Code.

```
# Python konfigurieren:
a,b=sp.symbols('a b');
# Parameter:
p=b**4+a**3*b-a**2*b**2-a*b**3;
```

Gemäss Ausgabe gilt

$$\underline{\underline{p = b \cdot (a-b)^2 \cdot (a+b).}} \tag{13}$$

7. Terme Vereinfachen mit Python/Sympy

Wir *vereinfachen* jeweils den Term mit Python/Sympy. Dazu implementieren wir den folgenden Code, den wir für jede Teilaufgabe modifizieren.

```
# Python initialisieren:
import IPython.display as dp;
import sympy as sp;
# Python konfigurieren:
sp.init_printing();
x=sp.symbols('x');
# Parameter:
p=...;
# Berechnungen:
q=sp.simplify(p);
# Ausgabe:
dp.display(q);
```

a) Wir betrachten den Term

$$p = 5\,x^2 + 3\,x^3 - 11\,x^2 + 3\,x. \tag{14}$$

Wir modifizieren den Code.

```
# Parameter:
p=5*x**2+3*x**3-11*x**2+3*x;
```

Gemäss Ausgabe gilt

$$\underline{\underline{p = 3\,x\,(x-1)^2.}} \tag{15}$$

b) Wir betrachten den Term

$$p = \frac{13 - 3\,x^4 - 3\,x^2 - 7}{2\,x^2 + 6 + x^2}\,. \tag{16}$$

Wir modifizieren den Code.

```
# Parameter:
p=(13-3*x**4-3*x**2-7)/(2*x**2+6+x**2);
```

Gemäss Ausgabe gilt

$$\underline{\underline{p = 1 - x^2.}} \tag{17}$$

c) Wir betrachten den Term

$$p = 2\left(3 + 3\tan^2(x)\right) \cdot \cos^2(x). \tag{18}$$

Wir modifizieren den Code.

```
# Parameter:
p=2*(3+3*sp.tan(x)**2)*sp.cos(x)**2;
```

Gemäss Ausgabe gilt

$$\underline{\underline{p = 6.}} \tag{19}$$

8. Gleichungen lösen mit Python/Sympy

Wir *vereinfachen* jeweils die Gleichung mit Python/Sympy. Dazu implementieren wir den folgenden Code, den wir für jede Teilaufgabe modifizieren.

```
# Python initialisieren:
import IPython.display as dp;
import sympy as sp;
# Python konfigurieren:
sp.init_printing();
x=sp.symbols('x');
# Parameter:
l=...; r=...;
# Berechnungen:
L=sp.solve(l-r,x);
# Ausgabe:
dp.display(L);
```

a) Wir betrachten die *Gleichung*

$$3\,x + 5 = 17. \tag{20}$$

Wir modifizieren den Code.

```
# Parameter:
l=3*x+5; r=17;
```

Gemäss Ausgabe ist die *Lösungsmenge*

$$\underline{\underline{\mathbb{L} = \{4\}.}} \tag{21}$$

b) Wir betrachten die *Gleichung*

$$3\,x^2 - 36\,x + 107 = 2. \tag{22}$$

Wir modifizieren den Code.

```
# Parameter:
l=3*x**2-36*x+107; r=2;
```

Gemäss Ausgabe ist die *Lösungsmenge*

$$\underline{\underline{\mathbb{L} = \{5, 7\}.}} \tag{23}$$

c) Wir betrachten die *Gleichung*

$$x^3 + 2\,x^2 - 5\,x - 2 = 4. \tag{24}$$

Wir modifizieren den Code.

```
# Parameter:
l=x**3+2*x**2-5*x-2; r=4;
```

Gemäss Ausgabe ist die *Lösungsmenge*

$$\underline{\underline{\mathbb{L} = \{-3, -1, 2\}}}. \tag{25}$$