lektion6

November 18, 2021

Inhalt

1 Lektion 6

1.1 Wiederholung und Ergänzungen

[1]:
```
%matplotlib notebook
#%matplotlib inline
#%matplotlib qt
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt
sp.init_printing()
```

[77]:
```
bild = plt.figure(1)                # Bild
kors = bild.gca()                   # Koordinatensystem
p1 = kors.plot([0, 1], [0, 1], 'mp-'  ) #  Liste von Linien
xn = np.linspace(-1, 1, 1000)
p2 = kors.plot(np.sin(xn), np.cos(xn))
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

[78]:
```
kors.set_aspect('equal')
p1[0].set_markersize(30)
```

[79]:
```
p1[0].set_markerfacecolor('gold')
p1[0].set_markeredgecolor((1, 0, 0)) # RGB Werte
```

[10]:
```
import matplotlib.colors as mcolors
```

[13]:
```
list(mcolors.BASE_COLORS.keys())
```

[13]:
```
['b', 'g', 'r', 'c', 'm', 'y', 'k', 'w']
```

[14]:
```
mcolors.CSS4_COLORS
```

[14]:
```
{'aliceblue': '#F0F8FF',
 'antiquewhite': '#FAEBD7',
 'aqua': '#00FFFF',
 'aquamarine': '#7FFFD4',
 'azure': '#F0FFFF',
 'beige': '#F5F5DC',
 'bisque': '#FFE4C4',
 'black': '#000000',
 'blanchedalmond': '#FFEBCD',
 'blue': '#0000FF',
 'blueviolet': '#8A2BE2',
 'brown': '#A52A2A',
 'burlywood': '#DEB887',
 'cadetblue': '#5F9EA0',
 'chartreuse': '#7FFF00',
 'chocolate': '#D2691E',
 'coral': '#FF7F50',
 'cornflowerblue': '#6495ED',
 'cornsilk': '#FFF8DC',
 'crimson': '#DC143C',
 'cyan': '#00FFFF',
 'darkblue': '#00008B',
 'darkcyan': '#008B8B',
 'darkgoldenrod': '#B8860B',
```

```
'darkgray': '#A9A9A9',
'darkgreen': '#006400',
'darkgrey': '#A9A9A9',
'darkkhaki': '#BDB76B',
'darkmagenta': '#8B008B',
'darkolivegreen': '#556B2F',
'darkorange': '#FF8C00',
'darkorchid': '#9932CC',
'darkred': '#8B0000',
'darksalmon': '#E9967A',
'darkseagreen': '#8FBC8F',
'darkslateblue': '#483D8B',
'darkslategray': '#2F4F4F',
'darkslategrey': '#2F4F4F',
'darkturquoise': '#00CED1',
'darkviolet': '#9400D3',
'deeppink': '#FF1493',
'deepskyblue': '#00BFFF',
'dimgray': '#696969',
'dimgrey': '#696969',
'dodgerblue': '#1E90FF',
'firebrick': '#B22222',
'floralwhite': '#FFFAF0',
'forestgreen': '#228B22',
'fuchsia': '#FF00FF',
'gainsboro': '#DCDCDC',
'ghostwhite': '#F8F8FF',
'gold': '#FFD700',
'goldenrod': '#DAA520',
'gray': '#808080',
'green': '#008000',
'greenyellow': '#ADFF2F',
'grey': '#808080',
'honeydew': '#F0FFF0',
'hotpink': '#FF69B4',
'indianred': '#CD5C5C',
'indigo': '#4B0082',
'ivory': '#FFFFF0',
'khaki': '#F0E68C',
'lavender': '#E6E6FA',
'lavenderblush': '#FFF0F5',
'lawngreen': '#7CFC00',
'lemonchiffon': '#FFFACD',
'lightblue': '#ADD8E6',
'lightcoral': '#F08080',
'lightcyan': '#E0FFFF',
'lightgoldenrodyellow': '#FAFAD2',
```

```
'lightgray': '#D3D3D3',
'lightgreen': '#90EE90',
'lightgrey': '#D3D3D3',
'lightpink': '#FFB6C1',
'lightsalmon': '#FFA07A',
'lightseagreen': '#20B2AA',
'lightskyblue': '#87CEFA',
'lightslategray': '#778899',
'lightslategrey': '#778899',
'lightsteelblue': '#B0C4DE',
'lightyellow': '#FFFFE0',
'lime': '#00FF00',
'limegreen': '#32CD32',
'linen': '#FAF0E6',
'magenta': '#FF00FF',
'maroon': '#800000',
'mediumaquamarine': '#66CDAA',
'mediumblue': '#0000CD',
'mediumorchid': '#BA55D3',
'mediumpurple': '#9370DB',
'mediumseagreen': '#3CB371',
'mediumslateblue': '#7B68EE',
'mediumspringgreen': '#00FA9A',
'mediumturquoise': '#48D1CC',
'mediumvioletred': '#C71585',
'midnightblue': '#191970',
'mintcream': '#F5FFFA',
'mistyrose': '#FFE4E1',
'moccasin': '#FFE4B5',
'navajowhite': '#FFDEAD',
'navy': '#000080',
'oldlace': '#FDF5E6',
'olive': '#808000',
'olivedrab': '#6B8E23',
'orange': '#FFA500',
'orangered': '#FF4500',
'orchid': '#DA70D6',
'palegoldenrod': '#EEE8AA',
'palegreen': '#98FB98',
'paleturquoise': '#AFEEEE',
'palevioletred': '#DB7093',
'papayawhip': '#FFEFD5',
'peachpuff': '#FFDAB9',
'peru': '#CD853F',
'pink': '#FFC0CB',
'plum': '#DDA0DD',
'powderblue': '#B0E0E6',
```

```
 'purple': '#800080',
 'rebeccapurple': '#663399',
 'red': '#FF0000',
 'rosybrown': '#BC8F8F',
 'royalblue': '#4169E1',
 'saddlebrown': '#8B4513',
 'salmon': '#FA8072',
 'sandybrown': '#F4A460',
 'seagreen': '#2E8B57',
 'seashell': '#FFF5EE',
 'sienna': '#A0522D',
 'silver': '#C0C0C0',
 'skyblue': '#87CEEB',
 'slateblue': '#6A5ACD',
 'slategray': '#708090',
 'slategrey': '#708090',
 'snow': '#FFFAFA',
 'springgreen': '#00FF7F',
 'steelblue': '#4682B4',
 'tan': '#D2B48C',
 'teal': '#008080',
 'thistle': '#D8BFD8',
 'tomato': '#FF6347',
 'turquoise': '#40E0D0',
 'violet': '#EE82EE',
 'wheat': '#F5DEB3',
 'white': '#FFFFFF',
 'whitesmoke': '#F5F5F5',
 'yellow': '#FFFF00',
 'yellowgreen': '#9ACD32'}
```

1.2 Bilder abspeichern

[16]:
```
bild.savefig('einbild2.png')
```

[18]:
```
bild.canvas.get_supported_filetypes()
```

[18]:
```
{'eps': 'Encapsulated Postscript',
 'jpg': 'Joint Photographic Experts Group',
 'jpeg': 'Joint Photographic Experts Group',
 'pdf': 'Portable Document Format',
 'pgf': 'PGF code for LaTeX',
 'png': 'Portable Network Graphics',
 'ps': 'Postscript',
 'raw': 'Raw RGBA bitmap',
 'rgba': 'Raw RGBA bitmap',
 'svg': 'Scalable Vector Graphics',
```

```
 'svgz': 'Scalable Vector Graphics',
 'tif': 'Tagged Image File Format',
 'tiff': 'Tagged Image File Format'}
```

[17]:
```
bild.savefig('einbild.svg', format='svg')
```

1.3 3D Koordinatensysteme

[20]:
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')  # 3d Koordinatensystem
ax.plot(1, 1, 1, 'rx')
ax.set_xlim3d((-1, 1))
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

[20]:
$(-1.0,\ 1.0)$

1.4 Implizit gegebene Flächen Isoflächen (marching_cube + plot_trisurf)

Alle Punkte $(x, y, z) \in \mathbb{R}^3$ derart, dass

$$f(x, y, z) = 0 \quad \text{für} \quad f : \mathbb{R}^3 \to \mathbb{R}$$

[30]:
```
from skimage import measure

xn = np.linspace(-2 ,2, 100)
scalex = lambda x: xn[0]+(xn[-1]-xn[0])*x/(len(xn)-1)

X, Y, Z = np.meshgrid(xn, xn, xn)
vol = X**4+Y**4+Z**4+1000*(X**4+Y**4)*(X**4+Z**4)*(Y**4+Z**4)-10
#vol = X**2+Y**2+Z**2-4                    # Kugel
#vol = (np.sqrt(X**2+Y**2)-1)**2+Z**2-1/4  # Torus

verts, faces, _, __ = measure.marching_cubes(vol, level=0)
# 0-Isofläche beschrieben durch Dreiecksflächen

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_trisurf(scalex(verts[:, 0]), scalex(verts[:, 1]), scalex(verts[:, 2]), \
                triangles = faces, cmap = 'hsv' , alpha=0.5)

ax.set_xlim3d((xn[0], xn[-1]))
ax.set_ylim3d((xn[0], xn[-1]))
ax.set_zlim3d((xn[0], xn[-1]))
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

[30]: $(-2.0,\ 2.0)$

ohne Skalierung mit 'scalex'

[48]:
```
dx = xn[1]-xn[0]

verts, faces, normals, values = measure.marching_cubes(vol, level=0,␣
 ↪spacing=(3*[dx]))

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_trisurf(verts[:, 0]-2, verts[:, 1]-2, verts[:, 2]-2, \
    triangles = faces)
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

[48]: `<mpl_toolkits.mplot3d.art3d.Poly3DCollection at 0x7f3c25120f40>`

1.5 Nochmal Listen

1.5.1 Zugriff auf Elemente einer Liste (Slicing von Listen)

[31]:
```
a = [1, 2, [3, 4], [5, 6], 7]
a
```

[31]: $[1,\ 2,\ [3,\ 4],\ [5,\ 6],\ 7]$

[32]:
```
a[0] # erstes Element
```

[32]: 1

[42]:
```
a[-1]  # letztes Element
```

[42]: 7

[36]:
```
a[0:-1] # jedes Element vom Ersten (0) bis Vorletzten (-1)
```

[36]: $[1,\ 2,\ [3,\ 4],\ [5,\ 6]]$

[37]:
```
a[::2]  # jedes zweite Element vom Ersten (0) bis zum Letzten
```

[37]: $[1,\ [3,\ 4],\ 7]$

a[anfang:ende:increment] 'ende' ist exklusiv

[38]:
```
c = a[0:-1:2]
c
```

[38]:
```
[1, [3, 4]]
```

[43]:
```
b = a + c #  Zusammenfügen von Listen
b
```

[43]:
```
[1, 2, [3, 4], [5, 6], 7, 1, [3, 4]]
```

[44]:
```
b[2][0] = 'pi' #Achtung
```

[45]:
```
b
```

[45]:
```
[1, 2, ['pi', 4], [5, 6], 7, 1, ['pi', 4]]
```

[46]:
```
3*c
```

[46]:
```
[1, ['pi', 4], 1, ['pi', 4], 1, ['pi', 4]]
```

1.5.2 Flaches und tiefes kopieren (copy / deepcopy)

[52]:
```
al = [1, 21, [3, 4]]
```

[53]:
```
ac = al.copy()  # flache Kopie
ac[0] = 'hallo'
ac[2][0] = 2225
```

[54]:
```
al
```

[54]:
```
[1, 21, [2225, 4]]
```

[55]:
```
ac
```

[55]:
```
['hallo', 21, [2225, 4]]
```

[56]:
```
from copy import deepcopy
al = [1, 21, [3, 4]]
```

[58]:
```
ad = deepcopy(al)# tiefe Kopie
```

[59]:
```
ad[0] = 'hallo'
ad[2][0] = 2225
```

[60]:
```
ad
```

[60]:
```
['hallo', 21, [2225, 4]]
```

[61]:
```
al
```

[61]:

[1, 21, [3, 4]]

[62]:
```
b = al + deepcopy(al)
```

[63]:
```
b
```

[63]: [1, 21, [3, 4], 1, 21, [3, 4]]

[64]:
```
b[2][0]= 331
```

[65]:
```
b
```

[65]: [1, 21, [331, 4], 1, 21, [3, 4]]

1.6 Zu den aktuellen Aufgaben

1.6.1 zu Aufgabe 24

[67]:
```
def f (c):  # abschnittweise definierte Funktion
    if c > 1:
        c= 1
    elif c <-1:
        c =-1
    else:
        c = c**3
    return c
```

[68]:
```
f(2.3)
```

[68]: 1

[69]:
```
f(-2), f(-1), f(0), f(0.5), f(1), f(2)
```

[69]: (−1, −1, 0, 0.125, 1, 1)

1.6.2 Funktionen mit mehreren Ausgaben

[70]:
```
def g(c):
    q = c**2
    mc = -c
    return c, q, mc
```

[71]:
```
erg = g(2)
erg
```

[71]: (2, 4, −2)

[72]:
```
a, b, c = g(2)
```

[73]:
```
a, c
```

[73]: $(2, -2)$

1.7 Ergänzungen zu Schleifen

1.7.1 enumerate

[80]:
```
for e in [2, 5, 7, 9]:
    print(e)
```

```
2
5
7
9
```

[81]:
```
for i, e in enumerate([2, 5, 7, 9]):
    print(i, e)
```

```
0 2
1 5
2 7
3 9
```

1.7.2 zip (Reißverschluss)

[84]:
```
for e in zip([12, 23, 43, 2 , 3 ], ['a', 'b', 'c', 'd']):
    print(e[0], e[1])
```

```
12 a
23 b
43 c
2 d
```

1.7.3 Erzeugen von Wörterbüchern (dictionary)

[2]:
```
d = {}
for e in zip([12, 23, 43], ['a', 'b', 'c']):
    d |= {e[0] : e[1]}  # inplace 'oder' geht seit Python 3.9
d
```

[2]: {12: 'a', 23: 'b', 43: 'c'}

[3]:
```
d = {}
for e in zip([12, 23, 43], ['a', 'b', 'c']):
    d.update({e[0] : e[1]})
```

[88]:
```
d
```

[88]:
```
{12: 'a', 23: 'b', 43: 'c'}
```

[92]:
```
q = { -i: i**2 for i in range(2, 10)} # Wörterbuch der Quadratzahlen
```

[93]:
```
q
```

[93]:
$$\{-9 : 81,\ -8 : 64,\ -7 : 49,\ -6 : 36,\ -5 : 25,\ -4 : 16,\ -3 : 9,\ -2 : 4\}$$

[95]:
```
q[-2]
```

[95]:
$$4$$

1.8 Numerische Auswertung von Integralen

[96]:
```
x = sp.symbols('x')
I1 = sp.Integral(sp.sin(x**2+sp.sin(x)), (x, 0, 20))
I1
```

[96]:
$$\int_{0}^{20} \sin\left(x^{2} + \sin\left(x\right)\right)\, dx$$

[97]:
```
I1.doit()# das ist zu schwierig
```

[97]:
$$\int_{0}^{20} \sin\left(x^{2} + \sin\left(x\right)\right)\, dx$$

[98]:
```
%%timeit
I1.n()
```

```
249 ms ± 6.53 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

[99]:
```
I2 = sp.Integral(sp.sin(x**2)/x, (x, 0, 2))
I2
```

[99]:
$$\int_{0}^{2} \frac{\sin\left(x^{2}\right)}{x}\, dx$$

[100]:
```
I2.doit()
```

[100]:
$$\frac{\mathrm{Si}\left(4\right)}{2}$$

[101]:
```
%%timeit
I2.n()
```

```
7.64 ms ± 85.7 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

[102]:
```
%%timeit
I2.doit().n()
```

```
158 ms ± 10.1 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```