lektion9

January 23, 2020

Table of Contents

1 Lektion 9

1.1 Vektoren und Matrizen

1.1.1 Matrizen erstellen

[1]:
```
from sympy import *
init_printing()
```

[2]:
```
v = Matrix([1, -1, 1]) # 3 x 1 Matrix
v_ = Matrix(3, 1, [1, -1, 1])
display(v), display(v_);
```

$$\begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

[3]:
```
u = Matrix(1, 3, [1, 2, 3]) # 1 x 3 Matrix
u
```

[3]:
$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

[4]:
```
w = Matrix([[4, 5, 6]]) # 1 x 3 Matrix
w
```

[4]:
$$\begin{bmatrix} 4 & 5 & 6 \end{bmatrix}$$

[5]:
```
A = Matrix([[10, 2, 13],\
            [4, 15, 6],\
            [17, 8, 19]])
A
```

[5]:
$$\begin{bmatrix} 10 & 2 & 13 \\ 4 & 15 & 6 \\ 17 & 8 & 19 \end{bmatrix}$$

[6]:
```
B = Matrix(3, 3, range(1,10))
B
```

[6]:
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

[7]:
```
As = symbols('a:3:3')
Asymb = Matrix(3,3,As)
Asymb
```

[7]:
$$\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}$$

Spezielle Matrizen

[8]:
```
E = eye(3)
E
```

[8]:
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

[9]:
```
C = zeros(3)
C
```

[9]:
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

[10]:
```
D = diag(1, 2, 3)
D
```

[10]:
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

[11]:
```
D_ = diag(range(1,4)) #Achtung
D_
```

[11]:
$$\left[\{1, 2, 3\}\right]$$

[12]:
```
DD = diag(A,B)
DD
```

[12]:
$$\begin{bmatrix} 10 & 2 & 13 & 0 & 0 & 0 \\ 4 & 15 & 6 & 0 & 0 & 0 \\ 17 & 8 & 19 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 4 & 5 & 6 \\ 0 & 0 & 0 & 7 & 8 & 9 \end{bmatrix}$$

[13]:
```
DDD = diag(u,v)
DDD
```

[13]:

$$\begin{bmatrix} 1 & 2 & 3 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

1.1.2 Matrizen zusammenfügen

[14]:
```
A.col_join(B)
```

[14]:
$$\begin{bmatrix} 10 & 2 & 13 \\ 4 & 15 & 6 \\ 17 & 8 & 19 \\ 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

[15]:
```
AB = A.row_join(B)
AB
```

[15]:
$$\begin{bmatrix} 10 & 2 & 13 & 1 & 2 & 3 \\ 4 & 15 & 6 & 4 & 5 & 6 \\ 17 & 8 & 19 & 7 & 8 & 9 \end{bmatrix}$$

alternativ (vgl. numpy)

[16]:
```
Matrix.vstack(A,B) # vertikales Stapeln
```

[16]:
$$\begin{bmatrix} 10 & 2 & 13 \\ 4 & 15 & 6 \\ 17 & 8 & 19 \\ 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

[17]:
```
Matrix.hstack(A,B) # horizontales Stapeln
```

[17]:
$$\begin{bmatrix} 10 & 2 & 13 & 1 & 2 & 3 \\ 4 & 15 & 6 & 4 & 5 & 6 \\ 17 & 8 & 19 & 7 & 8 & 9 \end{bmatrix}$$

[18]:
```
AB.reshape(1,len(AB)) # Matrix AB zeilenweise in eine 1 x 18 Matrix umwandeln
```

[18]:
$$\begin{bmatrix} 10 & 2 & 13 & 1 & 2 & 3 & 4 & 15 & 6 & 4 & 5 & 6 & 17 & 8 & 19 & 7 & 8 & 9 \end{bmatrix}$$

1.1.3 Matrizen mit Bildungsvorschrift erzeugen

[19]:
```
def element(i, j):
    return 1/(i+j+1)
```

[20]:
```
H = Matrix(3, 3, element) # Hilbertmatrix
H
```

[20]:
$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

[21]:
```
M = Matrix(3,3, lambda i,j : i-j)
M
```

[21]:
$$\begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$$

1.1.4 Matrixplots

[22]:
```
Hl = Matrix(15, 15, element)
Hl
```

[22]:
$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} \\ \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} \\ \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} \\ \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} \\ \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} & \frac{1}{23} \\ \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} & \frac{1}{23} & \frac{1}{24} \\ \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} & \frac{1}{23} & \frac{1}{24} & \frac{1}{25} \\ \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} & \frac{1}{23} & \frac{1}{24} & \frac{1}{25} & \frac{1}{26} \\ \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} & \frac{1}{23} & \frac{1}{24} & \frac{1}{25} & \frac{1}{26} & \frac{1}{27} \\ \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} & \frac{1}{23} & \frac{1}{24} & \frac{1}{25} & \frac{1}{26} & \frac{1}{27} & \frac{1}{28} \\ \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} & \frac{1}{21} & \frac{1}{22} & \frac{1}{23} & \frac{1}{24} & \frac{1}{25} & \frac{1}{26} & \frac{1}{27} & \frac{1}{28} & \frac{1}{29} \end{bmatrix}$$

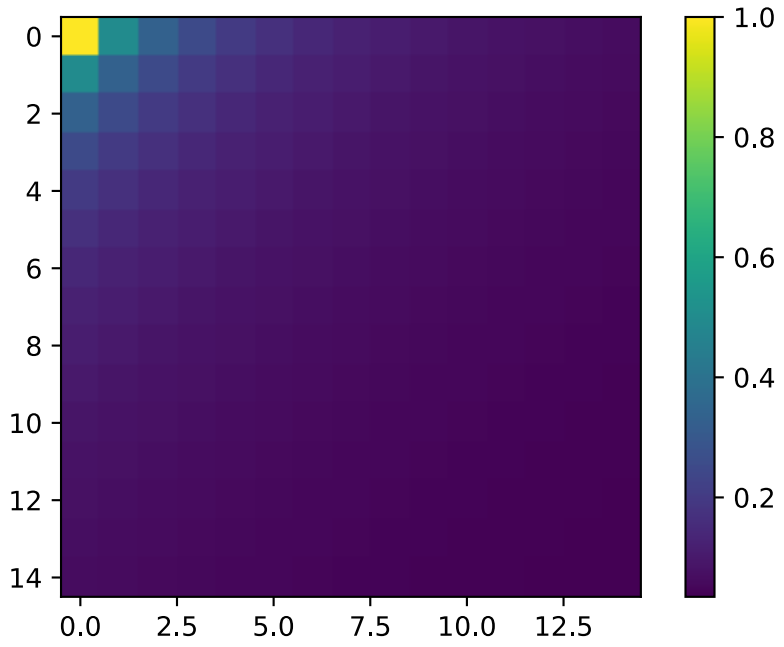
Plots sind Numerik

[23]:
```
import numpy as np
Hn = np.array(Hl).astype(float)
Hin = np.array(Hl.inv()).astype(float)
```

[24]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

```
%config InlineBackend.figure_format = 'svg'
plt.imshow(Hn)
plt.colorbar()
```

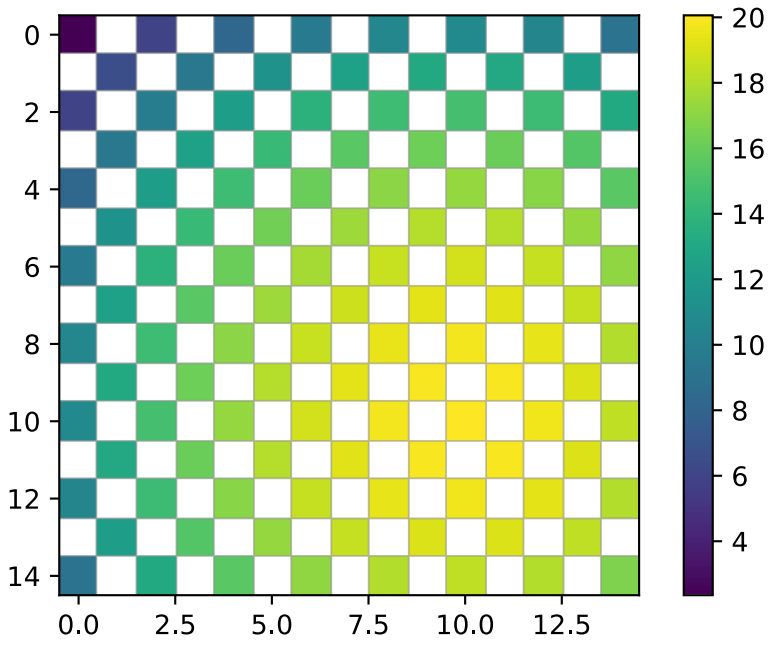
[24]: <matplotlib.colorbar.Colorbar at 0x7f6fae4a9150>



[25]:
```
plt.imshow(np.log10(Hin))
plt.colorbar()
```

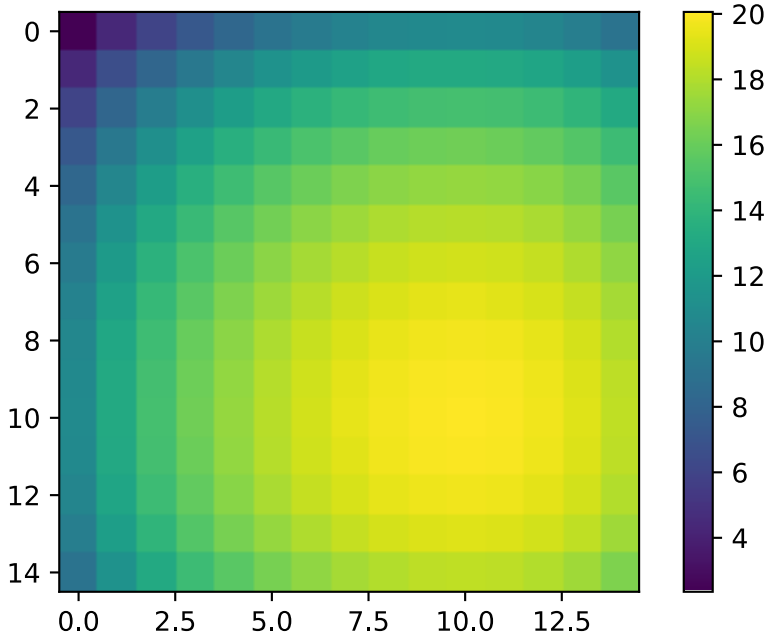
```
/local/schaedle/miniconda/lib/python3.7/site-packages/ipykernel_launcher.py:1:
RuntimeWarning: invalid value encountered in log10
  """Entry point for launching an IPython kernel.
```

[25]: <matplotlib.colorbar.Colorbar at 0x7f6fae37a4d0>



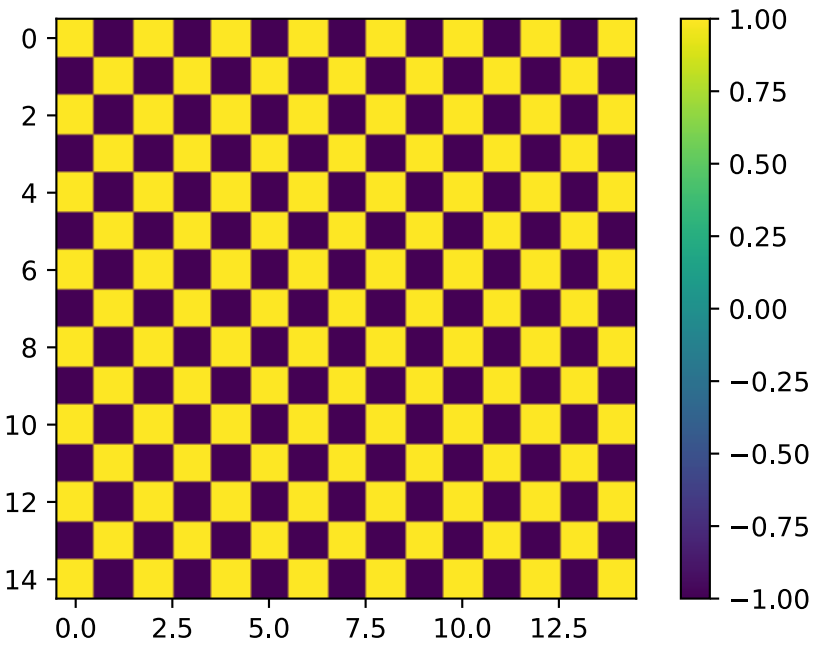
[26]:
```
import matplotlib.pyplot as plt
plt.imshow(np.log10(abs(Hin)))
plt.colorbar()
```

[26]: <matplotlib.colorbar.Colorbar at 0x7f6fae24da50>



[27]:
```
plt.imshow(np.sign(Hin))
plt.colorbar()
```

[27]:
```
<matplotlib.colorbar.Colorbar at 0x7f6fae19aa90>
```



1.1.5 Zugriff auf Matrixeinträge / Slicing

[28]:
```
A
```

[28]:
$$\begin{bmatrix} 10 & 2 & 13 \\ 4 & 15 & 6 \\ 17 & 8 & 19 \end{bmatrix}$$

[29]:
```
A[0]
```

[29]:
$$10$$

[30]:
```
A[3]
```

[30]:
$$4$$

[31]:
```
A[0, 0]
```

[31]:
$$10$$

[32]:
```
A[1, 2]
```

[32]: 6

[33]:
```
A[:, 1]
```

[33]: $\begin{bmatrix} 2 \\ 15 \\ 8 \end{bmatrix}$

[34]:
```
A[-1, :]  # letzte Zeile
```

[34]: $\begin{bmatrix} 17 & 8 & 19 \end{bmatrix}$

[35]:
```
AB[:, 1:-2:2] # von Spalte 1 bis zur vorletzten (-2) jede zweite Spalte
```

[35]: $\begin{bmatrix} 2 & 1 \\ 15 & 4 \\ 8 & 7 \end{bmatrix}$

1.1.6 Format/Dimension einer Matrix

[36]:
```
A.shape # 3 x 3 Matrix
```

[36]: $(3,\ 3)$

1.1.7 Matrixoperationen

Addition und Skalarmultiplikation

[37]:
```
a = symbols('a')
a*u
```

[37]: $\begin{bmatrix} a & 2a & 3a \end{bmatrix}$

[38]:
```
u + v
```

```
␣
↪---------------------------------------------------------------------------

ShapeError                                 Traceback (most recent call last)

<ipython-input-38-358b80487ff4> in <module>
----> 1 u + v
```

```
/local/schaedle/miniconda/lib/python3.7/site-packages/sympy/core/
↪decorators.py in binary_op_wrapper(self, other)
     127                     if f is not None:
     128                         return f(self)
--> 129             return func(self, other)
     130         return binary_op_wrapper
     131     return priority_decorator


/local/schaedle/miniconda/lib/python3.7/site-packages/sympy/matrices/
↪common.py in __add__(self, other)
    2135             if self.shape != other.shape:
    2136                 raise ShapeError("Matrix size mismatch: %s + %s" % (
-> 2137                     self.shape, other.shape))
    2138
    2139         # honest sympy matrices defer to their class's routine


ShapeError: Matrix size mismatch: (1, 3) + (3, 1)
```

[39]:
```
u + w
```

[39]: $\begin{bmatrix}5 & 7 & 9\end{bmatrix}$

[40]:
```
u - 2*w
```

[40]: $\begin{bmatrix}-7 & -8 & -9\end{bmatrix}$

[41]:
```
A -  2*B
```

[41]: $\begin{bmatrix}8 & -2 & 7\\-4 & 5 & -6\\3 & -8 & 1\end{bmatrix}$

Matrixmultiplikation $*$

[42]:
```
A*u
```

```
␣
↪---------------------------------------------------------------------------

ShapeError                                Traceback (most recent call last)

<ipython-input-42-49c5415e54c8> in <module>
----> 1 A*u
```

```
/local/schaedle/miniconda/lib/python3.7/site-packages/sympy/core/
↪decorators.py in binary_op_wrapper(self, other)
    127                     if f is not None:
    128                         return f(self)
--> 129             return func(self, other)
    130         return binary_op_wrapper
    131     return priority_decorator


/local/schaedle/miniconda/lib/python3.7/site-packages/sympy/matrices/
↪common.py in __mul__(self, other)
   2199             if self.shape[1] != other.shape[0]:
   2200                 raise ShapeError("Matrix size mismatch: %s * %s." % (
-> 2201                     self.shape, other.shape))
   2202
   2203         # honest sympy matrices defer to their class's routine


ShapeError: Matrix size mismatch: (3, 3) * (1, 3).
```

[43]:
```
u*A
```

[43]: $\begin{bmatrix} 69 & 56 & 82 \end{bmatrix}$

[44]:
```
u*v # euklidisches Skalarprodukt von u und v
```

[44]: $\begin{bmatrix} 2 \end{bmatrix}$

[54]:
```
z1 = Matrix(2, 1, [1, I])
z2 = Matrix(1, 2, [1, I])
z2*z1 # euklidisches Skalarprodukt von z mit sich selbst, kein innere Produkt!
```

[54]: $\begin{bmatrix} 0 \end{bmatrix}$

[45]:
```
v*u
```

[45]: $\begin{bmatrix} 1 & 2 & 3 \\ -1 & -2 & -3 \\ 1 & 2 & 3 \end{bmatrix}$

[46]:
```
xs = symbols('x:3')
xs
```

[46]: $(x_0,\ x_1,\ x_2)$

[55]:
```
x = Matrix(xs)
x
```

[55]:
$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$$

[56]:
```
A * x
```

[56]:
$$\begin{bmatrix} 10x_0 + 2x_1 + 13x_2 \\ 4x_0 + 15x_1 + 6x_2 \\ 17x_0 + 8x_1 + 19x_2 \end{bmatrix}$$

[57]:
```
P = Matrix(3, 3, [0, 1, 0, 1, 0, 0, 0, 0, 1])
P, A
```

[57]:
$$\left(\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 10 & 2 & 13 \\ 4 & 15 & 6 \\ 17 & 8 & 19 \end{bmatrix} \right)$$

[58]:
```
P*A  # Hintereinanderausführung linearer Abbildungen
```

[58]:
$$\begin{bmatrix} 4 & 15 & 6 \\ 10 & 2 & 13 \\ 17 & 8 & 19 \end{bmatrix}$$

[59]:
```
P@A  # wie in numpy
```

[59]:
$$\begin{bmatrix} 4 & 15 & 6 \\ 10 & 2 & 13 \\ 17 & 8 & 19 \end{bmatrix}$$

[60]:
```
matrix_multiply_elementwise(P, A) # Hadamard- oder Schurprodukt oder␣
↪elementweises Produkt
                                  # (entspricht * in numpy)
```

[60]:
$$\begin{bmatrix} 0 & 2 & 0 \\ 4 & 0 & 0 \\ 0 & 0 & 19 \end{bmatrix}$$

[61]:
```
A, B
```

[61]:
$$\left(\begin{bmatrix} 10 & 2 & 13 \\ 4 & 15 & 6 \\ 17 & 8 & 19 \end{bmatrix}, \quad \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \right)$$

[62]:
```
matrix_multiply_elementwise(A, B)
```

[62]:

$$\begin{bmatrix} 10 & 4 & 39 \\ 16 & 75 & 36 \\ 119 & 64 & 171 \end{bmatrix}$$

Transponieren

[63]:
```
x.T
```

[63]:
$$\begin{bmatrix} x_0 & x_1 & x_2 \end{bmatrix}$$

[64]:
```
x.T*x  # Achtung, das ergibt eine 1x1 Matrix
```

[64]:
$$\begin{bmatrix} x_0^2 + x_1^2 + x_2^2 \end{bmatrix}$$

[65]:
```
x.C # komplex konjugieren
```

[65]:
$$\begin{bmatrix} \overline{x_0} \\ \overline{x_1} \\ \overline{x_2} \end{bmatrix}$$

[66]:
```
x.H  # transponieren und komplex konjugieren
```

[66]:
$$\begin{bmatrix} \overline{x_0} & \overline{x_1} & \overline{x_2} \end{bmatrix}$$

[67]:
```
display(A), display(A.T);
```

$$\begin{bmatrix} 10 & 2 & 13 \\ 4 & 15 & 6 \\ 17 & 8 & 19 \end{bmatrix}$$

$$\begin{bmatrix} 10 & 4 & 17 \\ 2 & 15 & 8 \\ 13 & 6 & 19 \end{bmatrix}$$

1.2 Lineare Algebra

1.2.1 Determinate und Spur

[68]:
```
A = Matrix(3, 3, range(0, 9))
A
```

[68]:
$$\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}$$

[69]:
```
A[1,1] = 5
```

[70]:
```
A.det()
```

[70]:
$$-12$$

[71]:
```
A.trace()
```

[71]:
$$13$$

1.2.2 charakteristisches Polynom

[72]:
```
A
A[1,1] = 4
```

[73]:
```
A.charpoly()
```

[73]:
$$\text{PurePoly}\left(\lambda^3 - 12\lambda^2 - 18\lambda, \lambda, domain = \mathbb{Z}\right)$$

[74]:
```
A.det()
```

[74]:
$$0$$

1.2.3 Kern und Rang

[75]:
```
A.nullspace(), A.rank()
```

[75]:
$$\left(\left[\begin{bmatrix}1\\-2\\1\end{bmatrix}\right], \ 2\right)$$

[76]:
```
A[1,1] = 5
CS = A.columnspace()
RS = A.rowspace()
CS
```

[76]:
$$\left[\begin{bmatrix}0\\3\\6\end{bmatrix}, \ \begin{bmatrix}1\\5\\7\end{bmatrix}, \ \begin{bmatrix}2\\5\\8\end{bmatrix}\right]$$

1.2.4 Gram Schmidt Orthogonalisierung und Orthonormalisierung

klappt nur in $\mathbb{R}^n$ für das euklidische Skalarprodukt

[77]:
```
#CS[0][0] = 1+I
CSO = GramSchmidt(CS) # Orthogonalisierung
CSO
```

[77]:

$$\left[\begin{bmatrix}0\\3\\6\end{bmatrix}, \begin{bmatrix}1\\\frac{6}{5}\\-\frac{3}{5}\end{bmatrix}, \begin{bmatrix}\frac{6}{7}\\-\frac{4}{7}\\\frac{2}{7}\end{bmatrix}\right]$$

[78]:
```
[simplify(CSO[i].T*CSO[j]) for i in range(3) for j in range(3)]
```

[78]:
$$[[45], [0], [0], [0], [\frac{14}{5}], [0], [0], [0], [\frac{8}{7}]]$$

[79]:
```
CSON = GramSchmidt(CS, True) # Orthogonormalisierung
CSON
```

[79]:
$$\left[\begin{bmatrix}0\\\frac{\sqrt{5}}{5}\\\frac{2\sqrt{5}}{5}\end{bmatrix}, \begin{bmatrix}\frac{\sqrt{70}}{14}\\\frac{3\sqrt{70}}{35}\\-\frac{3\sqrt{70}}{70}\end{bmatrix}, \begin{bmatrix}\frac{3\sqrt{14}}{14}\\-\frac{\sqrt{14}}{7}\\\frac{\sqrt{14}}{14}\end{bmatrix}\right]$$

[80]:
```
[CSON[i].T*CSON[j] for i in range(3) for j in range(3)]
```

[80]:
$$[[1], [0], [0], [0], [1], [0], [0], [0], [1]]$$

1.2.5 Matrixinverse

[81]:
```
Hinv = H.inv()
Hinv
```

[81]:
$$\begin{bmatrix}9 & -36 & 30\\-36 & 192 & -180\\30 & -180 & 180\end{bmatrix}$$

[82]:
```
Hinv1 = H**(-1)
Hinv1
```

[82]:
$$\begin{bmatrix}9 & -36 & 30\\-36 & 192 & -180\\30 & -180 & 180\end{bmatrix}$$

[83]:
```
Hinv2 = H.inv(method="LU")
Hinv2
```

[83]:
$$\begin{bmatrix}9 & -36 & 30\\-36 & 192 & -180\\30 & -180 & 180\end{bmatrix}$$

[84]:
```
H*Hinv
```

[84]:
$$\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix}$$

[85]:
```
cancel(Asymb.inv())
```

[85]:
$$\begin{bmatrix} \frac{a_{11}a_{22}-a_{12}a_{21}}{a_{00}a_{11}a_{22}-a_{00}a_{12}a_{21}-a_{01}a_{10}a_{22}+a_{01}a_{12}a_{20}+a_{02}a_{10}a_{21}-a_{02}a_{11}a_{20}} & -\frac{a_{01}a_{22}-a_{02}a_{21}}{a_{00}a_{11}a_{22}-a_{00}a_{12}a_{21}-a_{01}a_{10}a_{22}+a_{01}a_{12}a_{20}+a_{02}a_{10}a_{21}-a_{02}a_{11}a_{2}} \\ -\frac{a_{10}a_{22}-a_{12}a_{20}}{a_{00}a_{11}a_{22}-a_{00}a_{12}a_{21}-a_{01}a_{10}a_{22}+a_{01}a_{12}a_{20}+a_{02}a_{10}a_{21}-a_{02}a_{11}a_{20}} & \frac{a_{00}a_{22}-a_{02}a_{20}}{a_{00}a_{11}a_{22}-a_{00}a_{12}a_{21}-a_{01}a_{10}a_{22}+a_{01}a_{12}a_{20}+a_{02}a_{10}a_{21}-a_{02}a_{11}a_{2}} \\ \frac{a_{10}a_{21}-a_{11}a_{20}}{a_{00}a_{11}a_{22}-a_{00}a_{12}a_{21}-a_{01}a_{10}a_{22}+a_{01}a_{12}a_{20}+a_{02}a_{10}a_{21}-a_{02}a_{11}a_{20}} & -\frac{a_{00}a_{21}-a_{01}a_{20}}{a_{00}a_{11}a_{22}-a_{00}a_{12}a_{21}-a_{01}a_{10}a_{22}+a_{01}a_{12}a_{20}+a_{02}a_{10}a_{21}-a_{02}a_{11}a_{2}} \end{bmatrix}$$

1.3 Lösung linearer Gleichungssysteme

1.3.1 Gausselimination vgl. Comp LA VL 7

[86]:
```
A = Matrix([[1, -1, 2], [0, 0, -1], [0, 2, -1]])
A
```

[86]:
$$\begin{bmatrix} 1 & -1 & 2 \\ 0 & 0 & -1 \\ 0 & 2 & -1 \end{bmatrix}$$

[87]:
```
A, v
```

[87]:
$$\left(\begin{bmatrix} 1 & -1 & 2 \\ 0 & 0 & -1 \\ 0 & 2 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \right)$$

[88]:
```
x = A.LUsolve(v)
x,  A@x-v
```

[88]:
$$\left(\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \right)$$

1.3.2 Lösung mit ‘solve’ und ‘linsolve’

[89]:
```
A, v
```

[89]:
$$\left(\begin{bmatrix} 1 & -1 & 2 \\ 0 & 0 & -1 \\ 0 & 2 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \right)$$

[90]:
```
x = Matrix(xs)
x
```

[90]:
$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$$

[91]:
```
Eq(A*x, v)
```

[91]:

$$\begin{bmatrix} x_0 - x_1 + 2x_2 \\ -x_2 \\ 2x_1 - x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

[92]:
```
solve(Eq(A*x, v), x)
```

[92]: $\{x_0 : 0,\ x_1 : 1,\ x_2 : 1\}$

[93]:
```
solve(Eq(A*x, v))
```

[93]: $\{x_0 : 0,\ x_1 : 1,\ x_2 : 1\}$

[94]:
```
 A.solve(v)
```

[94]: $\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$

[95]:
```
linsolve((A, v), xs)
```

[95]: $\{(0,\ 1,\ 1)\}$

[96]:
```
a = symbols("a")
B[0,1] = a
B
```

[96]: $\begin{bmatrix} 1 & a & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

[97]:
```
x = B.solve(v)
cancel(x)
```

[97]: $\begin{bmatrix} \frac{5a-14}{2a-4} \\ \frac{4}{a-2} \\ -\frac{11a-10}{6a-12} \end{bmatrix}$

[98]:
```
cancel(x)
```

[98]: $\begin{bmatrix} \frac{5a-14}{2a-4} \\ \frac{4}{a-2} \\ -\frac{11a-10}{6a-12} \end{bmatrix}$

[99]:
```
linsolve((B,v),xs)
```

[99]: $\left\{\left(\frac{5a-14}{2\,(a-2)},\ \frac{4}{a-2},\ \frac{10-11a}{6\,(a-2)}\right)\right\}$

[100]:
```
x = Matrix(3, 1, xs)
solve(Eq(B*x, v), xs)
```

[100]:
$$\left\{x_0 : \frac{5a-14}{2\,(a-2)},\ \ x_1 : \frac{4}{a-2},\ \ x_2 : \frac{10-11a}{6\,(a-2)}\right\}$$

1.3.3 LR Zerlegung vgl. CompLA VL 8

[101]:
```
A
```

[101]:
$$\begin{bmatrix} 1 & -1 & 2 \\ 0 & 0 & -1 \\ 0 & 2 & -1 \end{bmatrix}$$

[102]:
```
L,U,perm = A.LUdecomposition() # Zeile 2 und 3 muessen vertauscht werden
L, U, perm
```

[102]:
$$\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ \begin{bmatrix} 1 & -1 & 2 \\ 0 & 2 & -1 \\ 0 & 0 & -1 \end{bmatrix},\ [[1,\ 2]]\right)$$

[103]:
```
P = eye(A.shape[0]).permuteBkwd(perm)
P, L, U
```

[103]:
$$\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix},\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ \begin{bmatrix} 1 & -1 & 2 \\ 0 & 2 & -1 \\ 0 & 0 & -1 \end{bmatrix}\right)$$

[104]:
```
P@A-L@U
```

[104]:
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

[105]:
```
L,U,perm = H.LUdecomposition()
P = eye(H.shape[0]).permuteBkwd(perm)
P@H - L@U
```

[105]:
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

[106]:
```
# die Inverse von H ist bekannt. Die E
def element(i, j, n):
    return (-1)**(i+j)*(i+j+1)*binomial(n+i, n-j-1)*binomial(n+j,␣
 ↪n-i-1)*binomial(i+j, i)**2

Hi = Matrix(3, 3, lambda i, j : element(i, j, 3))
Hi
```

[106]:

$$\begin{bmatrix} 9 & -36 & 30 \\ -36 & 192 & -180 \\ 30 & -180 & 180 \end{bmatrix}$$

[107]:
```
H*Hi
```

[107]:
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

[108]:
```
Hnp = matrix2numpy(H)
Hnp
```

[108]:
```
array([[1, 1/2, 1/3],
       [1/2, 1/3, 1/4],
       [1/3, 1/4, 1/5]], dtype=object)
```

[109]:
```
Hnp = matrix2numpy(H, dtype='float')
Hnp
```

[109]:
```
array([[1.        , 0.5       , 0.33333333],
       [0.5       , 0.33333333, 0.25      ],
       [0.33333333, 0.25      , 0.2       ]])
```

1.4 Matrixfunktionen

[110]:
```
sin(H)
```

[110]:
$$\sin\left(\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}\right)$$

[111]:
```
H.applyfunc(sin) # sinus angewandt auf jeden Eintrag der Matrix
```

[111]:
$$\begin{bmatrix} \sin(1) & \sin\left(\frac{1}{2}\right) & \sin\left(\frac{1}{3}\right) \\ \sin\left(\frac{1}{2}\right) & \sin\left(\frac{1}{3}\right) & \sin\left(\frac{1}{4}\right) \\ \sin\left(\frac{1}{3}\right) & \sin\left(\frac{1}{4}\right) & \sin\left(\frac{1}{5}\right) \end{bmatrix}$$

[112]:
```
exp(A)  # Matrixexponetialfunktion
```

[112]:

$$\begin{bmatrix} e & 2\,\mathrm{re}\left(\frac{e^{-\frac{1}{2}+\frac{\sqrt{7}i}{2}}}{\left(\frac{1}{2}-\frac{\sqrt{7}i}{2}\right)\left(\frac{3}{2}-\frac{\sqrt{7}i}{2}\right)}\right)+\frac{\sqrt{7}\,\mathrm{im}\left(\frac{e^{-\frac{1}{2}-\frac{\sqrt{7}i}{2}}}{\frac{3}{2}+\frac{\sqrt{7}i}{2}}\right)}{2}+\frac{\mathrm{re}\left(\frac{e^{-\frac{1}{2}-\frac{\sqrt{7}i}{2}}}{\frac{3}{2}+\frac{\sqrt{7}i}{2}}\right)}{2}-\frac{2\sqrt{7}e\left(\frac{7\,\mathrm{im}\left(\frac{1}{\left(\frac{1}{2}+\frac{\sqrt{7}i}{2}\right)\left(\frac{3}{2}+\frac{\sqrt{7}i}{2}\right)}\right)}{4}+\frac{7\sqrt{7}\,\mathrm{re}\left(\frac{1}{\left(\frac{1}{2}+\frac{\sqrt{7}i}{2}\right)\left(\frac{3}{2}+\frac{\sqrt{7}i}{2}\right)}\right)}{4}\right.}{7} \\ 0 & \frac{\cos\left(\frac{\sqrt{7}}{2}\right)}{2e^{\frac{1}{2}}}+\frac{\sqrt{7}\sin\left(\frac{\sqrt{7}}{2}\right)}{14e^{\frac{1}{2}}}+\frac{2\sqrt{7}\,\mathrm{im}\left(\frac{e^{-\frac{1}{2}+\frac{\sqrt{7}i}{2}}}{\frac{1}{2}-\frac{\sqrt{7}i}{2}}\right)}{7} \\ 0 & \frac{4\sqrt{7}\sin\left(\frac{\sqrt{7}}{2}\right)}{7e^{\frac{1}{2}}} \end{bmatrix}$$

[113]:
```
M = randMatrix(2,3) # zufällige ganze Zahle zwischen 0 .. 99
M
```

[113]:
$$\begin{bmatrix} 19 & 17 & 53 \\ 69 & 62 & 52 \end{bmatrix}$$

1.5 Kopieren von Matrizen

[114]:
```
BB = B
```

[115]:
```
BB,B
```

[115]:
$$\left(\begin{bmatrix} 1 & a & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix},\ \begin{bmatrix} 1 & a & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right)$$

[116]:
```
BB[0,0]=9
```

[117]:
```
BB,B
```

[117]:
$$\left(\begin{bmatrix} 9 & a & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix},\ \begin{bmatrix} 9 & a & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right)$$

[118]:
```
BBB = B.copy()
```

[119]:
```
BBB[2,2] = 99
```

[120]:
```
BBB,B
```

[120]:
$$\left(\begin{bmatrix} 9 & a & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 99 \end{bmatrix},\ \begin{bmatrix} 9 & a & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right)$$