# lektion10

December 5, 2024

## 1 LR-Zerlegung

[1]:
```
import numpy as np
np.set_printoptions(legacy = '1.25')
```

### 1.1 Instabilitäten bei Gauß-Elimination

[6]:
```
# Variante 1
A = np.array([[1e-20, 1],
              [1,     1]])

L = np.array([[1,    0],
              [1e20, 1]])

R = np.array([[1e-20,       1],
              [0,     1-1e20]])

np.linalg.norm(A - L@R)
```

[6]:
```
1.0
```

[3]:
```
L@R
```

[3]:
```
array([[1.e-20, 1.e+00],
       [1.e+00, 0.e+00]])
```

[4]:
```
# Variante 2
A = np.array([[1e-20, 1],
              [1,     1]])

P = np.array([[0, 1],
              [1, 0]])

L = np.array([[1,    0],
              [1e-20, 1]])

R = np.array([[1,       1],
```

```
              [0, 1-1e-20]])

np.linalg.norm(P@A - L@R)
```

[4]: 0.0

### 1.2 Vergleich verschiedener Verfahren zur Lösung eines linearen Gleichungssystems

[5]:
```
import numpy as np
import scipy as sc
import time

# Funktion zum Erzeugen eines zufälligen linearen Gleichungssystems
def generate_linear_system(size):
    A = np.random.randn(size, size) + 10*np.eye(size)
    b = np.random.randn(size)
    return A, b

# Funktion zum Messen der Ausführungszeit einer Methode
def measure_time(method, A, b, runs = 3):
    zeit = []
    for i in range(runs):
        start_time = time.time()
        method(A, b)
        zeit.append(time.time() - start_time)
    return min(zeit)

# Hauptfunktion zum Vergleichen der Methoden
def compare_methods(size):
    A, b = generate_linear_system(size)

    # Numpy's linalg.solve
    numpy_time = measure_time(np.linalg.solve, A, b)
    numpy_res  = np.linalg.norm(A@np.linalg.solve(A,b) - b)

    # Inverse Matrix Methode
    inverse_time = measure_time(lambda A, b: np.linalg.inv(A), A, b)
    inverse_res  = np.linalg.norm(A@(np.linalg.inv(A)@b) - b)

    # LU-Zerlegung
    lu_time = measure_time(lambda A, b: sc.linalg.lu_solve(sc.linalg.
 ↪lu_factor(A), b), A, b)
    lu_res  = np.linalg.norm(A@sc.linalg.lu_solve(sc.linalg.lu_factor(A), b) -␣
 ↪b)

    return [numpy_time, inverse_time, lu_time, numpy_res, inverse_res, lu_res]
```

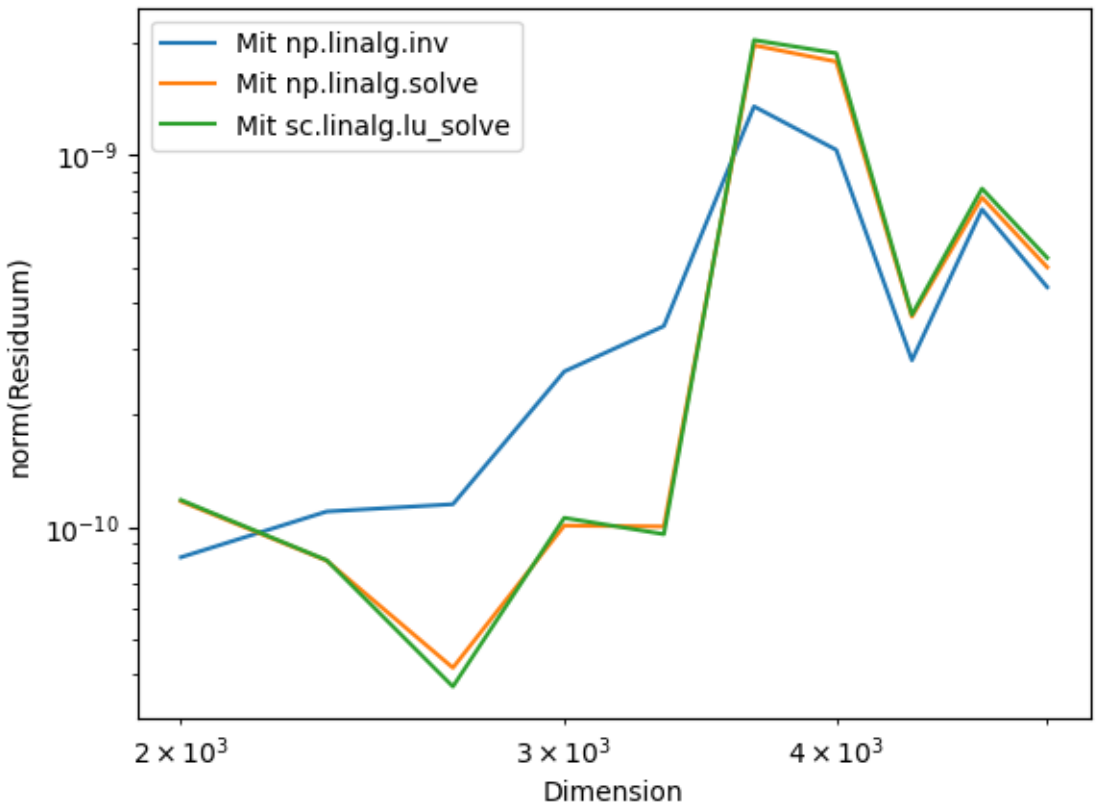
```
# Beispielaufruf mit einer Systemgröße von 100
compare_methods(100)
```

[5]:
```
[9.393692016601562e-05,
 0.00015997886657714844,
 0.00012922286987304688,
 1.936933379254351e-13,
 2.1881304448409068e-13,
 1.936933379254351e-13]
```

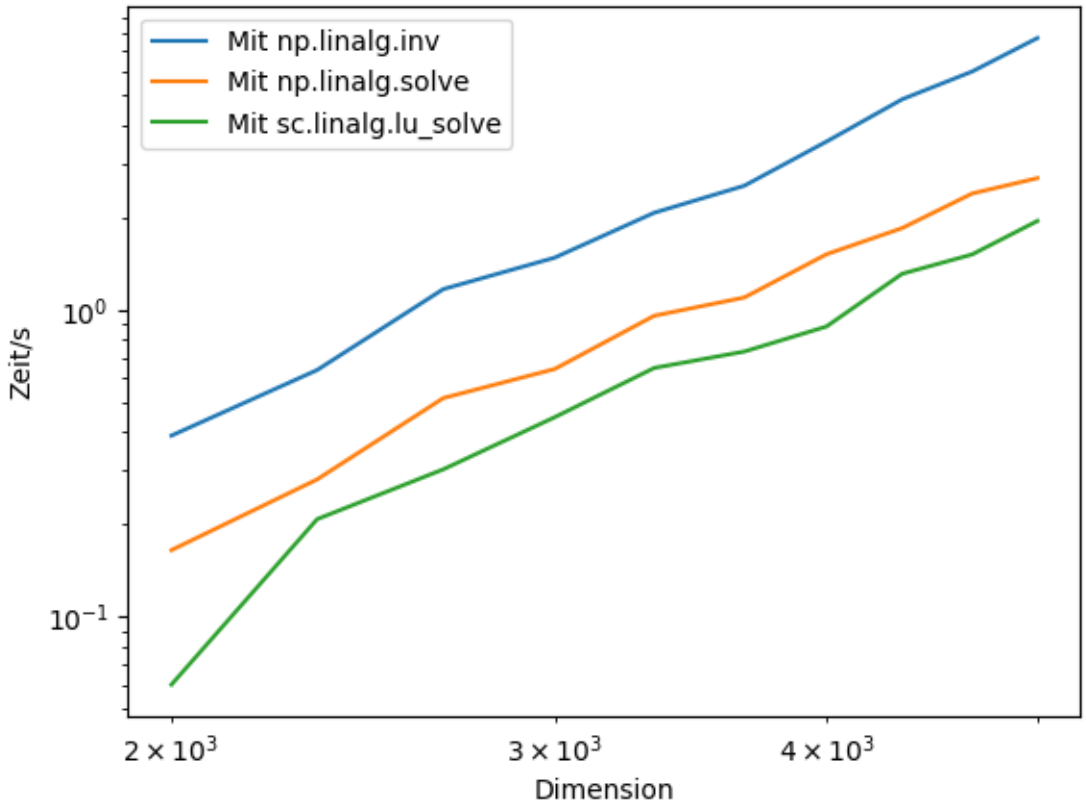
[6]:
```
N =  np.linspace(2e3, 5e3, 10)
zeiten = np.zeros((len(N), 6))

for i, n in enumerate(N):
    print(i+1,'/',len(N))
    zeiten[i,:] = compare_methods(int(n))

numpy_time, inverse_time, lu_time, numpy_res, inverse_res, lu_res = zeiten[:
 ↪,0], zeiten[:,1], zeiten[:,2], zeiten[:,3], zeiten[:,4], zeiten[:,5]
```

```
1 / 10
2 / 10
3 / 10
4 / 10
5 / 10
6 / 10
7 / 10
8 / 10
9 / 10
10 / 10
```

[7]:
```
import matplotlib.pyplot as plt

plt.figure()
plt.loglog(N, inverse_res, label = 'Mit np.linalg.inv');
plt.loglog(N, numpy_res,   label = 'Mit np.linalg.solve');
plt.loglog(N, lu_res,      label = 'Mit sc.linalg.lu_solve')
plt.ylabel('norm(Residuum)')
plt.xlabel('Dimension')
plt.legend();
```

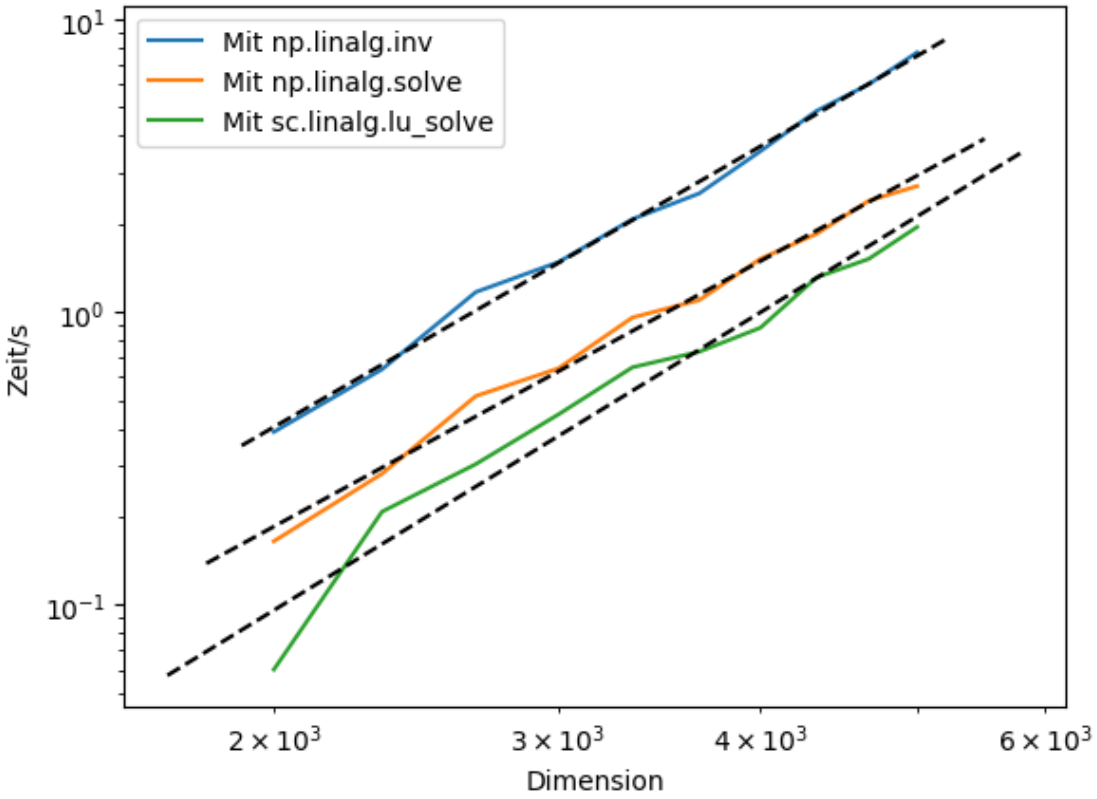



[8]:
```
plt.figure()
plt.loglog(N, inverse_time, label = 'Mit np.linalg.inv');
plt.loglog(N, numpy_time,   label = 'Mit np.linalg.solve');
plt.loglog(N, lu_time,      label = 'Mit sc.linalg.lu_solve')
plt.ylabel('Zeit/s')
plt.xlabel('Dimension')
plt.legend();
```




[9]:
```
def abline(slope, intercept):
    """Plot a line from slope and intercept"""
    axes = plt.gca()
    x_vals = np.array(axes.get_xlim())
    y_vals = np.exp(intercept)*x_vals**slope
    plt.loglog(x_vals, y_vals, 'k--')


plt.figure()
plt.loglog(N, inverse_time, label = 'Mit np.linalg.inv');
plt.loglog(N, numpy_time,   label = 'Mit np.linalg.solve');
plt.loglog(N, lu_time,      label = 'Mit sc.linalg.lu_solve')

slope, intercept = np.polyfit(np.log(N), np.log(inverse_time), deg = 1)
abline(slope, intercept)
slope, intercept = np.polyfit(np.log(N), np.log(numpy_time), deg = 1)
abline(slope, intercept)
slope, intercept = np.polyfit(np.log(N), np.log(lu_time), deg = 1)
abline(slope, intercept)

plt.ylabel('Zeit/s')
```

```
plt.xlabel('Dimension')
plt.legend();
```




[ ]: