1. 10\*10 Komponenten Modularisierung

## chiwan1.py

import numpy as np

matrixSize = 10

Qa= np.random.rand(matrixSize, matrixSize)

Q\_1 = np.dot(Qa, Qa.transpose())

print(Q\_1)

np.linalg.cholesky(Q\_1) ##check if a matrix is positive definite

p\_1= np.random.uniform(0, 9, size=10)

G\_1 = np.diag(-1\*np.random.uniform(0, 9, size=10))

h\_1 = np.random.uniform(0, 9, size=10)

l\_1 = np.random.uniform(-9, 0, size=10)

np.savez\_compressed(r'C:\Users\skqkr\Desktop\Semesterarbeit/Chiwan\_Q1', Q=Q\_1, p=p\_1, G=G\_1, h=h\_1, l=l\_1)

##2. 50\*50 Modularisierung

import numpy as np

matrixSize = 50

Qa= np.random.rand(matrixSize, matrixSize)

Q\_2 = np.dot(Qa, Qa.transpose())

print(Q\_2)

np.linalg.cholesky(Q\_2) ##check if a matrix is positive definite

p\_2= np.random.uniform(0, 9, size=50)

G\_2 = np.diag(-1\*np.random.uniform(0, 9, size=50))

h\_2 = np.random.uniform(0, 9, size=50)

l\_2 = np.random.uniform(-9, 0, size=50)

np.savez\_compressed(r'C:\Users\skqkr\Desktop\Semesterarbeit/Chiwan\_Q2', Q=Q\_2, p=p\_2, G=G\_2, h=h\_2, l=l\_2)

3. 100\*100

import numpy as np

matrixSize = 100

Qa= np.random.rand(matrixSize, matrixSize)

Q\_3 = np.dot(Qa, Qa.transpose())

print(Q\_3)

np.linalg.cholesky(Q\_3) ##check if a matrix is positive definite

p\_3= np.random.uniform(0, 9, size=100)

G\_3 = np.diag(-1\*np.random.uniform(0, 9, size=100))

h\_3 = np.random.uniform(0, 9, size=100)

l\_3 = np.random.uniform(-9, 0, size=100)

np.savez\_compressed(r'C:\Users\skqkr\Desktop\Semesterarbeit/Chiwan\_Q3', Q=Q\_3, p=p\_3, G=G\_3, h=h\_3, l=l\_3)

4. 500\*500

import numpy as np

matrixSize = 500

Qa= np.random.rand(matrixSize, matrixSize)

Q\_4 = np.dot(Qa, Qa.transpose())

print(Q\_4)

np.linalg.cholesky(Q\_4) ##check if a matrix is positive definite

p\_4= np.random.uniform(0, 9, size=500)

G\_4 = np.diag(-1\*np.random.uniform(0, 9, size=500))

h\_4 = np.random.uniform(0, 9, size=500)

l\_4 = np.random.uniform(-9, 0, size=500)

np.savez\_compressed(r'C:\Users\skqkr\Desktop\Semesterarbeit/Chiwan\_Q4', Q=Q\_4, p=p\_4, G=G\_4, h=h\_4, l=l\_4)

5. 1000\*1000

import numpy as np

matrixSize = 1000

Qa= np.random.rand(matrixSize, matrixSize)

Q\_5 = np.dot(Qa, Qa.transpose())

print(Q\_5)

np.linalg.cholesky(Q\_5) ##check if a matrix is positive definite

p\_5= np.random.uniform(0, 9, size=1000)

G\_5 = np.diag(-1\*np.random.uniform(0, 9, size=1000))

h\_5 = np.random.uniform(0, 9, size=1000)

l\_5 = np.random.uniform(-9, 0, size=1000)

np.savez\_compressed(r'C:\Users\skqkr\Desktop\Semesterarbeit/Chiwan\_Q5', Q=Q\_5, p=p\_5, G=G\_5, h=h\_5, l=l\_5)

6. 2000\*2000

import numpy as np

matrixSize = 2000

Qa= np.random.rand(matrixSize, matrixSize)

Q\_6 = np.dot(Qa, Qa.transpose())

print(Q\_6)

np.linalg.cholesky(Q\_6) ##check if a matrix is positive definite

p\_6= np.random.uniform(0, 9, size=2000)

G\_6 = np.diag(-1\*np.random.uniform(0, 9, size=2000))

h\_6 = np.random.uniform(0, 9, size=2000)

l\_6 = np.random.uniform(-9, 0, size=2000)

np.savez\_compressed(r'C:\Users\skqkr\Desktop\Semesterarbeit/Chiwan\_Q6', Q=Q\_6, p=p\_6, G=G\_6, h=h\_6, l=l\_6)

7. 5000\*5000

import numpy as np

matrixSize = 5000

Qa= np.random.rand(matrixSize, matrixSize)

Q\_7 = np.dot(Qa, Qa.transpose())

print(Q\_7)

np.linalg.cholesky(Q\_7) ##check if a matrix is positive definite

p\_7= np.random.uniform(0, 9, size=5000)

G\_7 = np.diag(-1\*np.random.uniform(0, 9, size=5000))

h\_7 = np.random.uniform(0, 9, size=5000)

l\_7 = np.random.uniform(-9, 0, size=5000)

np.savez\_compressed(r'C:\Users\skqkr\Desktop\Semesterarbeit/Chiwan\_Q4', Q=Q\_7, p=p\_7, G=G\_7, h=h\_7, l=l\_7)