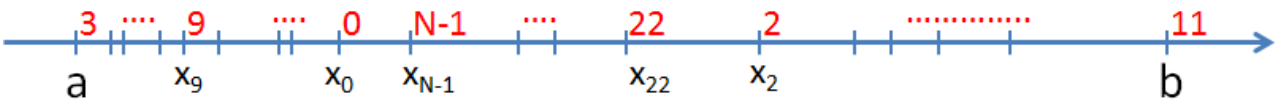
Übungsaufgaben zu Python

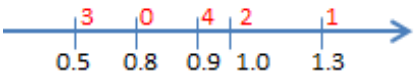
Aufgabe 1

In der Abbildung ist ein Zahlenstrahl zu sehen, auf dem N x-Werte markiert sind. Die unterschiedlichen Positionen seien von 0 bis N-1 zufällig nummeriert (rote Zahlen). Die x-Werte zu den Positionen heißen entsprechend ihrer Nummer x_k. In einer Python Liste p=[] sollen diese Werte hinterlegt werden, hierbei soll gelten $p[k] = x_k$. In der zweiten Abbildung ist ein Beispiel zu sehen.

Allgemein:



Ein Beispiel:



p=[0.8,1.3,1,0.5,0.9]
t=[[4,2],[3,0],[0,4],[2,1]]

a) Erstellen Sie für ein N Ihrer Wahl und $a = \sqrt{2}$, $b = 5e$ eine Liste p0, in der die Positionen fortlaufend und äquidistant von links nach rechts positioniert sind.

b) Erstellen Sie eine Liste p1, in der die Positionsnummern und die Abstände zufällig verteilt sind.

c) Benutzen Sie nun die x_k-Werte aus Teil a), nummerieren Sie die Positionen aber zufällig und speichern Sie das Ergebnis in einer Liste p2 ab.

d) Definieren Sie eine neue Liste t mit der Eigenschaft, dass diese Zweierlisten (Elemente) enthält, welche immer zwei nebeneinanderliegende Nummern beinhaltet. Die Elemente müssen nicht fortlaufend eingetragen werden. Als Beispiel diene obige Skizze mit vier Elementen (erstellen Sie eine Funktion).
Berechnen Sie hiermit für p0, p1, p2 den mittleren Abstand, den maximalen Abstand sowie den kleinsten Abstand der Teilintervalle. Erstellen Sie eine Liste mit den Mittelpunkten und zeichen Sie für $N = 10000$ ein Balkendiagramm mit der Häufigkeitsverteilung der auftretenden Abstände.

e) Löschen Sie in p2 alle x-Werte, die zu Positionsnummern gehören, die durch 3 teilbar sind und erstellen Sie anschließend die neue t-Liste.

f) Berechnen Sie mittels der p-Listen die Funktion $y = ln(x^2)$ und zeichnen Sie diese auf dem Intervall $[\sqrt{2}, 5e]$.

g) Bestimmen Sie numerisch die Fläche unter der Kurve, mittels der t-Liste und der p-Liste und zum Vergleich mit Hilfe einer Python-Funktion und vergleichen Sie mit dem exakten Ergebnis.

Aufgabe 2

In dieser Aufgabe soll es darum gehen Gleichungssysteme zu lösen. Die Dimension der Matrix sei $N_d \times N_d$.

a) Erstellen Sie für $N_d = 10$ eine Matrix A und einen Vektor $\vec{b}$ aus Zufallszahlen und lösen Sie die Gleichung $A \cdot \vec{x} = \vec{b}$.

b) Variieren Sie nun N_d von 500 bis 15000 in Schritten von 1000 und messen Sie die Zeit, die Sie für die Lösung des LGS benötigen (benutzen Sie zur Lösung `numpy.linalg.solve`). Tragen Sie schließlich die Zeit für die Lösung über N_d auf.

c) Erstellen Sie nun eine Matrix B, die nur von Null verschiedene Elemente auf der Diagonalen, sowie oberhalb und unterhalb der Diagonalen hat. Alle anderen Elemente seien null. Wiederholen Sie für diese Matrix die Aufgabe b).

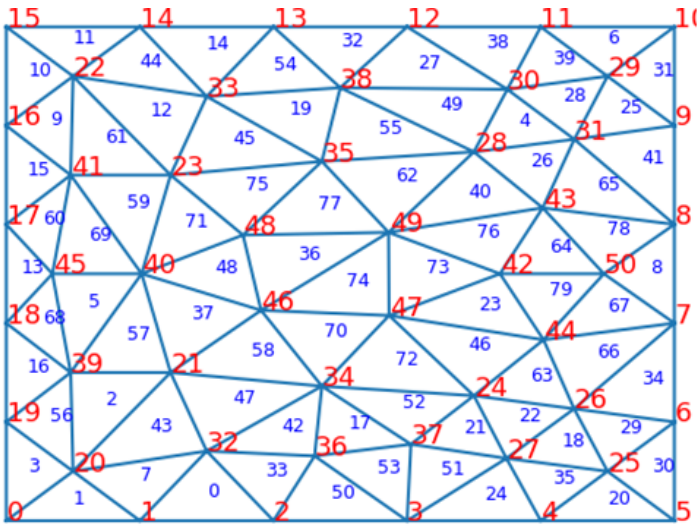
d) Benutzen Sie wieder die Matrix B aus Teilaufgabe c), speichern Sie aber die Matrix als sogenannte dünnbesetze Matrix (sparse matrix) ab. Die entsprechende Definition einer solchen Matrix kann mit `B=scipy.sparse.lil_matrix( (Nd,Nd) )` erfolgen. Gelöst werden dünnbesetzte Gleichungssysteme mit dem Befehl `x=scipy.sparse.linalg.spsolve(B.tocsc(),b)`. Bestimmen Sie erneut die Rechenzeit zur Lösung des LGS bei verschiedenen N_d wie in b). Tragen Sie alle Zeiten aus b), c) und d) in einem Graphen auf.

e) Passen Sie die Kurven aus b) und d) mit einem Polynom an. Mit welcher Potenz wachsen die Lösungszeiten als Funktion von N_d?

f) Zeichnen Sie den Speicherbedarf für die Matrizen aus b) und d) als Funktion von N_d.

Aufgabe 3

In der Aufgabe 1 wurde ein eindimensionales Intervall in Abschnitte aufgeteilt. Hier wollen wir nun ein zweidimensionales Gebiet in Teilgebiete unterteilen. Analog zum 1D-Beispiel definiert man auch hier eine p-Liste und eine t-Liste. Die p-Liste enthält nun Punkte in Form einer Liste oder eines Tupels und die Elemente in der t-Liste bestehen aus einer Liste dreier Nummern, da es sich bei den Teilgebieten um Dreiecke handelt. Der Umlaufsinn innerhalb eines Elementes sei immer im Gegenuhrzeigersinn, die Startnummer ist beliebig. Neben Punkten und Elementen gibt es noch Kanten. Die meisten Kanten gehören zu zwei Dreiecken. Die Skizze zeigt ein Beispiel für ein Rechteck.



Positionen
p=[[x_0,y_0] , [x_1,y_1] , ...]

Elemente
t=[[1,2,32] , [20,0,1] , [21,39,20] , ...]

Kanten
e=[[20,32],[47,42],....]

a) Unterteilen Sie ein Rechteck der (Länge 3, Höhe 4) in N=8 Dreiecke und erstellen Sie mit elementaren Befehlen die Liste p und die Liste t mittels eines Python-Programmes. Testen Sie Ihr Programm auch mit größeren N-Werten. Sie können die Dreiecke mit `plt.triplot` zeichnen (siehe Listing des b)-Teils).

b) Folgendes Python Programm erfüllt die Aufgabe aus der Teilaufgabe a) unter Verwendung von geeigneten Python-Modulen

```
import numpy as np
from scipy.spatial import Delaunay
import matplotlib.pyplot as plt

#Anfangs und Endpunkt
y0=1; yN=5
x0=-1; xN=2
#Anzahl der Positionen N=N0*N0 Anzahl Dreiecke etwa N0*N0/2
N0=7

# Lege Punkte fest
points=[]
for i in  range(N0):
  for j in  range(N0):
    x=x0+i*(xN-x0)/(N0-1)
    y=y0+j*(yN-y0)/(N0-1)
    points+=[[x,y]]
```

```
points=np.array(points)

# Nummeriere und lege Elemente fest
tri = Delaunay(points)
# entnehme p und t-Liste
t=tri.simplices
p=1.0*points


print("p-Liste: ",p)
print("t-Liste: ",t)

#Male die Dreiecke
plt.triplot(p[:,0], p[:,1], t)
#male Kreise an den Positionen
plt.plot(p[:,0], p[:,1], 'o')
plt.show()
```

Berechnen Sie eine Liste aller Mittelpunkte und Flächen der Dreiecke.

c) Erstellen Sie für b) eine Liste e aller Kanten (siehe auch Skizze), eine Liste aller Kanten r, die am Rand liegen und eine Liste aller Dreiecke rt, die am Rand liegen.

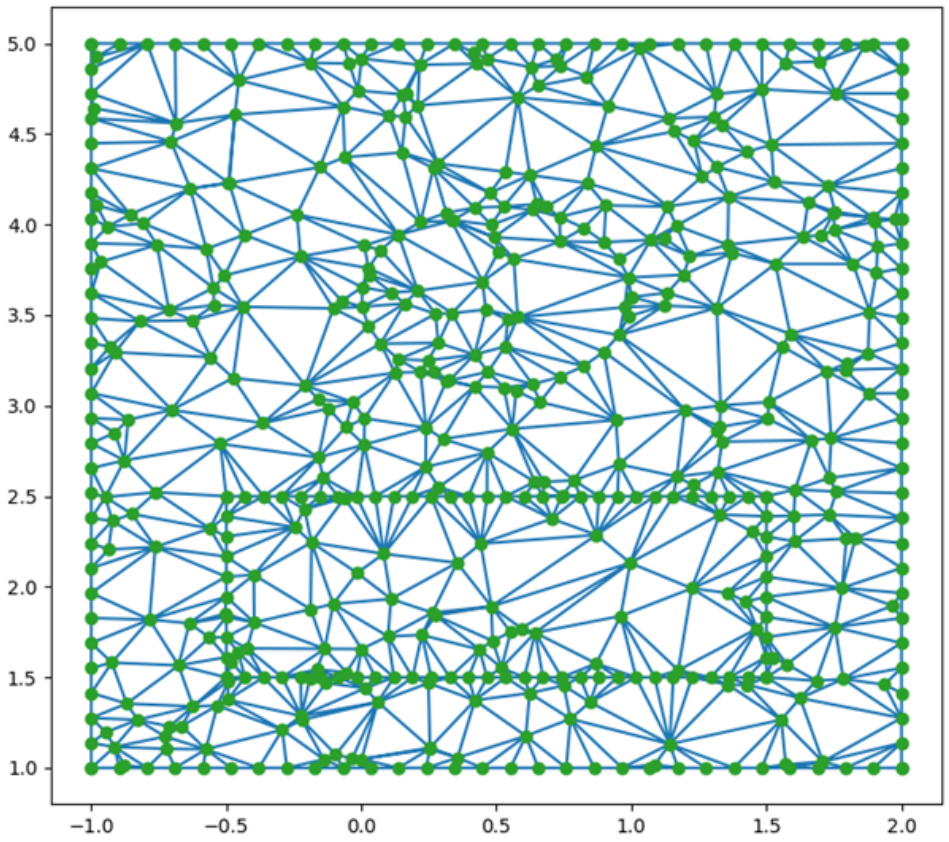
d) Berechnen Sie die Funktion $f(x, y) = x/(x^2 + y^2)$ auf dem Rechteck (d.h. an den Positionen) und zeichnen Sie die Höhenlinien der Funktion (benutzen Sie hierzu `plt.tricontourf`, bzw `plt.tricontour`).

e) Malen Sie ein Kreuz in den Schwerpunkt jener Dreiecke, die im oberen rechten Viertel des Rechtecks liegen.

f) Im Rechteck von Teilaufgabe b) mit etwa N=200 soll ein Kreis ausgeschnitten werden. Erstellen Sie hierzu eine neue t-Liste. Wie könnten Sie den inneren Kreisrand besser approximieren?

Aufgabe 4

In der Abbildung ist ein Rechteck gezeigt, welches in Dreiecke unterteilt ist. Im großen Rechteck sind weiterhin ein Kreis und ein kleines Rechteck zu erkennen. Alle Punkte, die nicht auf den Rändern der Rechtecke oder des Kreises liegen sind zufällig erzeugt.



a) Versuchen Sie das obige Bild annähernd zu reproduzieren. Verwenden Sie wieder die Funktion `scipy.spatial.Delaunay`, wie in der vorangegangenen Aufgabe.

b) Erstellen Sie eine Häufigkeitsverteilung der Winkel und der Flächen der Dreiecke.

c) Malen Sie an der Stelle der Schwerpunkte der Dreiecke
– ein rotes Kreuz für diejenigen Dreiecke innerhalb des Kreises.
– einen grünen Kreis für diejenigen Dreiecke innerhalb des kleinen Rechteckes.

d) Berechnen Sie die Summe aller Dreiecksflächen innerhalb des Kreises und des kleinen Rechteckes.

e) Berechnen Sie anhand der p und t-Liste den Umfang des großen Rechteckes.

f) Erstellen Sie eine Liste der Positionen, die auf dem Rand zwischen den Punkten $(-1, 5)$ und $(-1, 1)$ liegen.

g) Erstellen Sie eine Liste der Kanten, die auf dem Rand zwischen den Punkten $(2, 1)$ und $(2, 5)$ liegen.

h) Berechnen Sie die Funktion $\Phi(x, y)$ auf dem gesamten Gebiet (d.h. an den Positionen) und machen Sie einen Höhenlinienplot (benutzen Sie hierzu `plt.tricontourf`, bzw `plt.tricontour`).

$$\Phi(x,y) = \begin{cases} xy & (x,y) \text{ im Kreis} \\ x+y & (x,y) \text{ im kleinen Rechteck} \\ x^2+y^2 & \text{sonst} \end{cases}$$