Hochschule München
Fakultät für Informatik und Mathematik

Praktikumsaufgabe 4

in der Vorlesung

Computational Geometry

Konvexe Hüllen mit qhull

Team:	Christopher Hinz, Tobias Gruber
Studiengruppe:	Master Informatik
Studiensemester:	1. Semester
Schwerpunkt:	Embedded Computing

19. Juni 2022

Sommersemester 2022

1 Einführung

Installieren Sie das Programm qhull, erzeugen Sie zufällige Punktmengen und berechnen Sie mit qhull konvexe Hüllen, auch in höheren Dimensionen (qhull bringt ein Werkzeug zur Erzeugung von Punktmengen mit). Plotten Sie die Zeiten für zunehmende Punktanzahlen bei unterschiedlichen Dimensionen (2-8). Versuchen Sie, die Ausgaben von qhull bei "geschwätzigsterËinstellung nachzuvollziehen, zu verstehen und ggf. mit Inhalten dieser Lehrveranstaltung in Einklang zu bringen.

2 Funktionen von qhull

Qhull bietet 6 verschiedene Programme an:

- qconvex: convex hulls
- qdelaunay: Delaunay triangulations and furthest-site Delaunay triangulations
- qhalf: halfspace intersections about a point
- qhull: all structures with additional options
- qvoronoi: Voronoi diagrams and furthest-site Voronoi diagrams
- rbox: generate point distributions for qhull

Im Zuge dieses Praktikums werden wir sowohl rbox, zur Erzeugung von Punktmengen nutzen, als auch qconvex, zur Berechnen der konvexe Hüllen, nutzen.

2.1 rbox (http://www.qhull.org/html/rbox.htm)

Das Programm rbox generiert zufällige oder reguläre Punkte. Standardmäßig innerhalb eines Würfels (es sei denn die Optionen 's', 'x', oder 'y' werden übergeben). Einige Beispiele zur Erzeugung von Punktmengen sind:

- rbox 10 D3: erzeugt 10 Punkte in 3D
- rbox 15 D4: erzeugt 15 Punkte in 4D
- rbox 10 D2: erzeugt 10 Punkte auf einem 2D Kreis
- rbox 100 W0: erzeugt 100 Punkte auf der Oberfläche eines Würfels

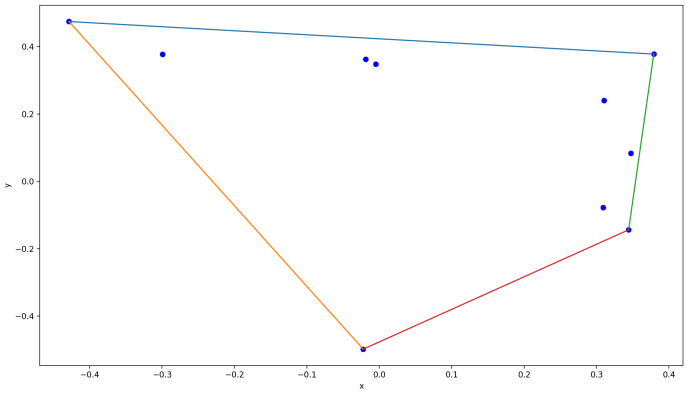
2.2 Beispiele

Bevor auf Details der Berechnungen von qconvex eingegangen wird, sollen zwei kurze Beispiele die Eingabedaten und die Ergebnisse der Berechnung visualisieren.
Zur Demonstration der Funktion von qconvex sollen Zufallsdaten und deren konvexe Hülle visualisiert werden. Hierzu wurde ein Python-Skript implementiert das die Datenpunkte plottet und aus den Ergebnissen von qconvex die Hyperebenen zeichnet. Die

Visualisierung kann entweder auf Basis der Normalenvektoren der Hyperebene (Option 'n') oder mit Hilfe der Vertices der Facetten (Option 'o') erfolgen

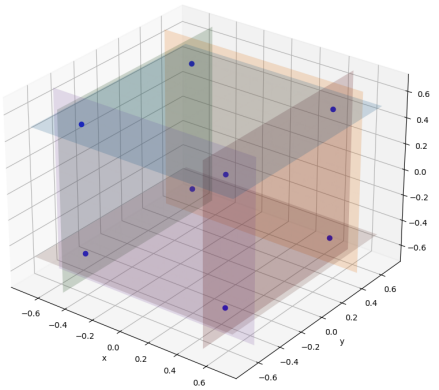
Punkte und konvexe Hülle in 2 Dimensionen

Im erste Beispiel werden die Datenpunkte in 2 Dimensionen erzeugt und eingelesen und deren Hyperebenen (Geraden) mit qconvex berechnet. Hierbei wurde die qconvex Option zur Ausgabe der Vertices genutzt. Mit diesen lassen sich die Hyperebenen (Geraden) zeichnen. Der nachfolgende Plot zeigt dies:



Punkte und konvexe Hülle in 3 Dimensionen

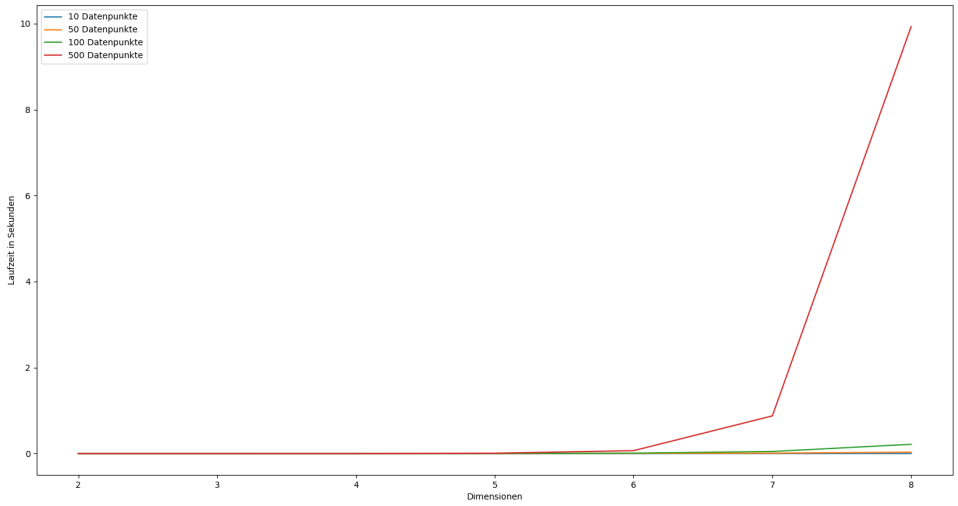
Im zweiten Beispiel werden die Datenpunkte in 3 Dimensionen erzeugt und eingelesen und ebenfalls die Hyperebenen (Ebenen) mit qconvex berechnet. Hier wurde ebenfalls die berechneten Vertices genutzt. Auch dies ist untenstehend mit matplotlib visualisiert:



2.3 Zeitmessung

Es soll die Laufzeit des Algorithmus für verschiedene Punktmengen in unterschiedlichen Dimensionen gemessen werden. Eine Zeitangabe ist bei der Ausgabe der Zusammenfassung des Ergebnisses enthalten. Die Zusammenfassung erhält man durch die qconvex Option 's'. Das Ergebnis der Messungen ist untenstehend grafisch und tabellarisch dargestellt.

Dimension	2D	3D	4D	5D	6D	7D	8D
Laufzeit in s	2.3e-05	3.4e-05	3.5e-05	4.2e-05	5.8e-05	6.2e-05	4.8e-05
	3e-05	6.3e-05	0.000186	0.000832	0.003005	0.009608	0.03245
	3.6e-05	7.8e-05	0.000298	0.001947	0.009298	0.04809	0.2151
	0.000116	0.00021	0.000907	0.007799	0.06736	0.876	9.933



2.4 qconvex (http://www.qhull.org/html/qconvex.htm)

Konvexe Hülle berechnen:

- qconvex s: s = print summary

Punktmengen erzeugen und Konvexe Hülle plotten:

- rbox 10 D3 — qconvex s: 10 Punkte in 3D und konvexe Hülle

TODO:

- Source code: https://github.com/qhull/qhull