Interpolation

Abgabe in Moodle über die Schaltfläche *Übungsaufgaben → Übungsblatt 5: Abgabe*. Sie können bis 23:59 Uhr des o.g. Datums abgeben. Achten Sie darauf, dass die letzte Abgabe bewertet wird.

Aufgabe 1 Programmieren: Kontinuierliche Repräsentation (5 Punkte)

In `task5_1.py` sind vier Punkte in `points` (x- und y-Koordinate) und deren zugehörige Werte in `values` gegeben. Es soll eine kontinuierliche 2D-Funktion durch die Punkte gefittet werden, um eine Dateninterpolation zu ermöglichen.

a) (1 Punkt)
Implementieren Sie die Funktion `phi(r)`, welche die radiale Basisfunktion $\phi(r) = e^{-r^2}$ berechnen soll.

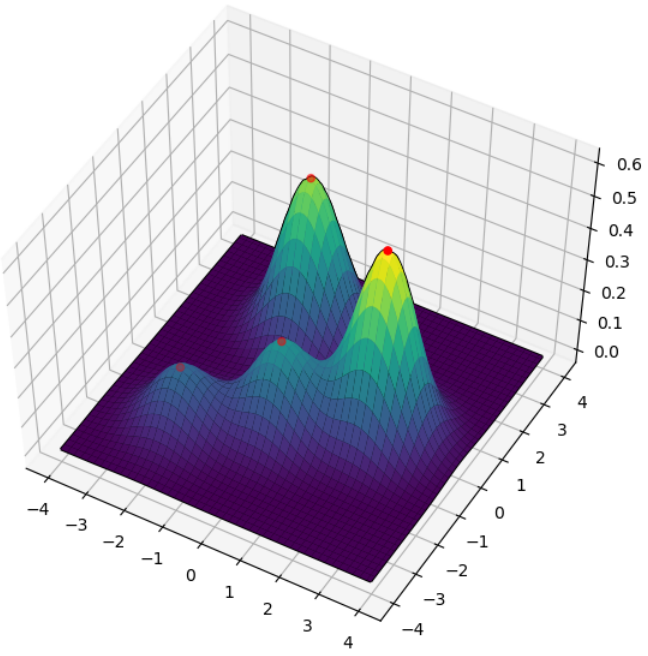
b) (1 Punkt)
Zeichnen Sie die Punkte in Rot in einen 3D Scatterplot. Die x-Koordinaten, also die erste Spalte der Punkte-Matrix, kann man über `points[:,0]` auslesen.

c) (3 Punkte)
Berechnen Sie die Gewichte w_i, indem Sie ein lineares Gleichungssystem lösen. Erstellen Sie im Anschluss die Funktion und zeigen Sie das Ergebnis in einem 3D Surfaceplot. Die eingezeichneten Punkte sollen lokale Maxima der Funktion sein. Die Höhe, bzw. der Funktionswert soll mit der *viridis*-Colormap dargestellt werden. Verwenden Sie den Funktionsbereich $[-4, 4]$ in x- und y-Richtung.

Hinweise:

- Siehe Cheatsheet *Creating Arrays* und *Vectors and Matrices*.

Das Ergebnis sieht so aus:



Aufgabe 2 Programmieren: Delaunay Triangulierung (5 Punkte)

In `task5_2.py` werden eine Reihe von Datenpunkten in einen vtkPoints-Container geladen. In dieser Aufgabe sollen die Punkte mittels VTK's Filterpipeline stückweise linear interpoliert werden.

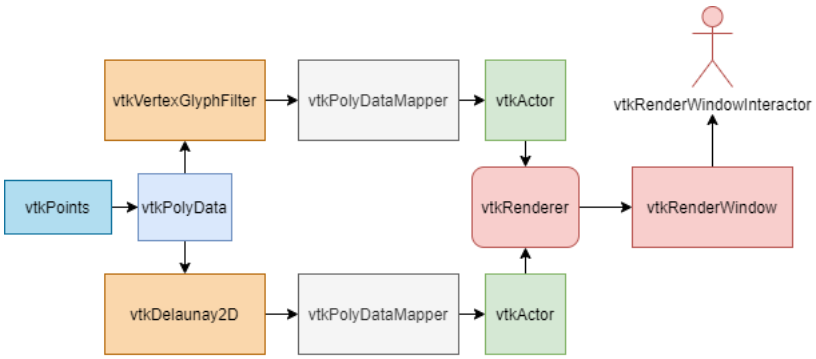
a) (3 Punkte)
Visualisieren Sie die Punkte als rote Kugeln.

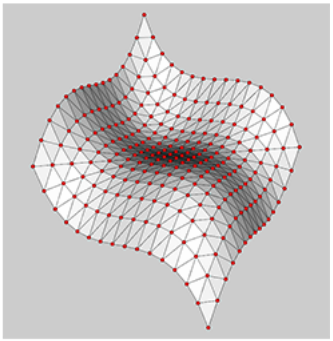
1. Fügen Sie die Punkte einer PolyData-Datenstruktur hinzu. Dieser Datentyp ist ein sehr allgemeiner Container, welcher Punkte, Linien und beliebige Polygone speichern kann:
```
polydata = vtk.vtkPolyData()
polydata.SetPoints(points)
```
2. Erstellen Sie nun einen VertexFilter, der die PolyData als Input erhält und Glyphen aus den Punkten (vertices) generiert:
```
vertexFilter = vtk.vtkVertexGlyphFilter()
vertexFilter.SetInputData(polydata)
vertexFilter.Update()
```
3. Zeigen Sie das Ergebnis an, analog zu z.B. Übungsblatt 2. D.h. Sie benötigen einen PolyDataMapper und einen Actor, die sie mit der Ausgabe des VertexFilter verknüpfen. Im Anschluss müssen Sie das angezeigte Bild mit einem renderer, einem RenderWindow und einem RenderWindowInteractor erstellen. Für den Actor verwenden Sie folgende Einstellungen:
```
actor.GetProperty().SetPointSize(8)
actor.GetProperty().SetRenderPointsAsSpheres(True)
actor.GetProperty().SetColor(1.0, 0.0, 0.0)
```

b) (2 Punkte)
Erstellen Sie nun eine zweite Datenansicht, welche die Punkte über den DelaunayFilter trianguliert:

1. Der DelaunayFilter wird wie der VertexFilter erstellt. Er kann dieselbe PolyData-Struktur als Input erhalten. Die Initialisierung geschieht mit: `delaunayFilter = vtk.vtkDelaunay2D()`
2. Sie benötigen einen zweiten Mapper und Actor. Der Actor soll derselben Szene hinzugefügt werden. Folgende Einstellung soll zudem übernommen werden, damit die Kanten des resultierenden Meshes sichtbar werden:
```
meshActor.GetProperty().SetEdgeVisibility(True)
```

Die zu erstellende Pipeline als Übersicht (links) und das Ergebnis (rechts):

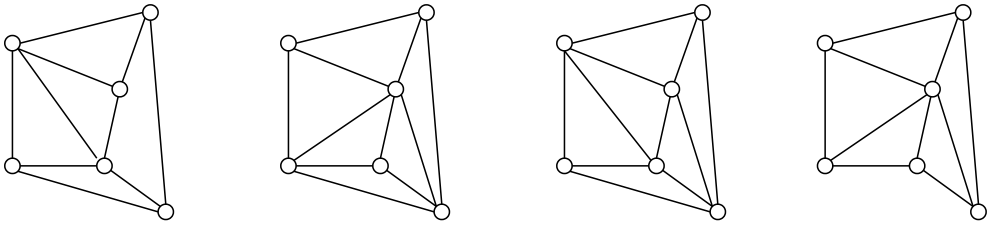




Aufgabe 3 Theorie: Interpolation (5 Punkte)

Geben Sie die Antworten auf die Theorieaufgaben direkt in Moodle ein.

a) (2 Punkte)
Welche der folgenden Darstellungen ist eine Delaunay Triangulierung?



(a) (b) (c) (d)

b) (1 Punkt)
Bei der Konstruktion eines Voronoi-Diagramms liegt der Schnittpunkt der Mittelsenkrechten, also ein Eckpunkt der Voronoi-Zelle, immer innerhalb eines Dreiecks der Delaunay Triangulierung.

(a) Wahr.

(b) Falsch.

c) (2 Punkte)
Gegeben seien folgende Punkte eines regulären Gitters (z.B. Pixel in einem Bild):

$$f(15,3) = 255 \qquad f(16,3) = 218 \qquad f(16,4) = 196 \qquad f(15,4) = 173$$

Bestimmen Sie den Wert der Stelle $f(15.5, 3.25)$ durch bilineare Interpolation.