Vorlesungen

„Einführung in die Datenvisualisierung“

„Angewandte Datenvisualisierung für Medizinphysiker“

Kapitel C.
C. Graphische Datenanalyse

Dr. Frank Weichert
Informatik VII
Technische Universität Dortmund
http://ls7-www.cs.uni-dortmunC.de

Nur zum persönlichen Gebrauch durch Mitglieder der Universität Dortmund

C.1. Dimensionsreduktion
- C.1.1. Hauptachsentransformation
- C.1.1.1. Lagrange-Multiplikatoren (Exkurs)
- C.1.1.2. Ausgleichsebenen
- C.1.1.3. Optimierungsproblem
- C.1.1.4. Hauptachsen
- C.1.1.5. Projektion
- C.1.1.6. Dimensionsreduktion
- C.1.1.7. Singularwertzerlegung
- C.1.2. Beispiel

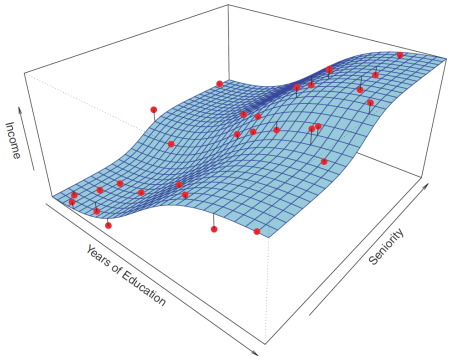
C.2. Kohonenkarten
- C,2.1. Einleitung
- C.2.2. Aufbau der Karte
- C.2.3. Anlernen
- C.2.4. Beispiele

C.3. Quantisierung
- C.3.1. Vektorquantisierung
- C.3.2. Hyper-Octree-Vektorquantisierung
- C.3.3. Median-Schnitt-Vektorquantisierung

C.4. Clustering
- C.4.1. k-Mittelwert-Clustering
- C.4.2. Hierarchisches Clustering
- C.4.3. Dichtebasiertes Clustering

Allgemeine Zielstellung

Bei Versuchen mit einer hohen Anzahl von Merkmalen diese in einer geschickten Form zusammenzufassen, sodass Unterschiede zwischen Merkmalsträgern in „wenigen Dimensionen“ sichtbar werden.



Formalere Definition

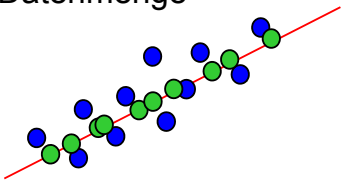
Über eine Dimensionsreduktion wird versucht, diejenigen Richtungen in einem hochdimensionalen Raum zu bestimmen, in denen die wesentlichen Strukturen in den Daten deutlich werden.

Exemplarische Teilfragestellungen

- Unterscheidung verschiedener Merkmalsträger
  ⇨ **Hauptachsentransformation**
- Ermittlung von Repräsentanten für Merkmalsträger
  ⇨ **Cluster- und Quantisierungsverfahren**

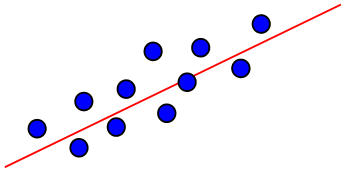
Gegenstand:

- Bestimmung der Hauptausdehnungsrichtungen einer Datenmenge
- Datenreduktion durch Projektion eines Punktes auf einen „wesentlichen“ Unterraum.

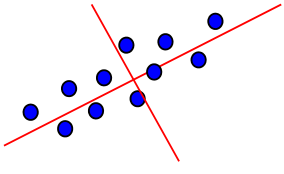


Anwendung bei der Visualisierung:

- Günstige Wahl des Koordinatensystems für die Visualisierung durch Projektionen und Schnitte
- Erkennung der wirklichen Dimension der Punktmenge zur Dimensionsreduktion durch Projektion



Ausgleichsgerade



Hauptachsen

Exkurs: Lagrange-Multiplikatoren

Gesucht: $f(x, y) = \min$ unter der Nebenbedingung $g(x, y) = 0$.

Lösung:

Lagrange-Ansatz: $f(x, y) - \lambda \cdot g(x, y) = \min$.

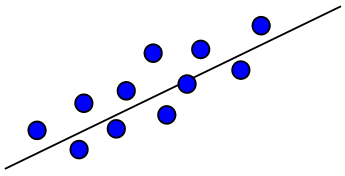
Partielles Ableiten nach den Variablen liefert

$$f_x - \lambda\, g_x = 0,\ f_y - \lambda\, g_y = 0,\quad g(x, y) = 0.$$

$\rightsquigarrow$ für das Minimum von $f(x, y) - \lambda \cdot g(x, y)$ ist $g(x, y) = 0$

$\rightsquigarrow$ Minimum von $f(x, y) - \lambda \cdot g(x, y)$ ist gleich dem Minimum von $f(x, y)$ unter der Bedingung $g(x, y) = 0$.

Ausgleichsebenen



Gegeben: n Punkte $\mathbf{a}_i$ der Dimension m.

Gesucht: Eine Hyperebene $\mathbf{e}^*\ \mathbf{x} = d$,

wobei

$\mathbf{e}$ einen m-dimensionalen Koeffizientenvektor mit $||\mathbf{e}||_2^2 = 1$ und

$\mathbf{x}$ m-dimensionale Punkte bezeichnet,

sodass die Summe der quadratischen Abstände der Punkte zu der Hyperebene minimal ist, d.h. $\sum_{i=1}^{n} (\mathbf{e}^* \mathbf{a}_i - d)^2$ minimal

Bemerkung: Es werden nur Ebenen betrachtet, deren Koeffizienten auf 1 normiert sind. Dies schränkt den Lösungsraum nicht ein, hat aber den Vorteil, dass dann $\mathbf{e}^*\ \mathbf{x} - d$ gleich dem Abstand von $\mathbf{x}$ zur Ebene ist.

Optimierungsproblem:

$$\min_{\mathbf{e}} \underbrace{\sum_{i=1}^{n}(\mathbf{e}^*\mathbf{a}_i - d)^2}_{f(\ldots)}, \text{ wobei } \underbrace{\|\mathbf{e}\|_2^2 = 1.}_{g(\ldots)}$$

bzw. $g(\ldots) = \|\mathbf{e}\|_2^2 - 1$

Lösungansatz:

Elimination der Nebenbedingung mit Hilfe von Lagrange-Multiplikatoren.

Beobachtung: Eine Hyperebene, welche die Quadratsumme der Abstände minimiert, geht durch den Schwerpunkt der Punkte $\mathbf{a}_i$.

Begründung: Notwendige Bedingung für das Optimum:

- Die Ableitung von $\sum_{i=1}^{n}(\mathbf{e}^*\mathbf{a}_i - d)^2 - \lambda(\|\mathbf{e}\|_2^2 - 1)$ nach d muss gleich 0 sein.
- Daraus folgt $d = \mathbf{e}^* \sum_{i=1}^{n} \mathbf{a}_i / n$, d.h. der Schwerpunkt $\sum_{i=1}^{n} \mathbf{a}_i / n$ liegt auf der Hyperebene.

Konsequenz: Möglichkeit der Vereinfachung des Optimierungsproblems durch Verschieben der gegebenen Punkte in den Schwerpunkt.

Lösungsverfahren:

1. Verschiebe die gegebenen Punkte, sodass ihr Schwerpunkt im Ursprung liegt.
2. Löse das Problem für den Spezialfall mit Ausgleichsebenen durch den Ursprung.
3. Verschiebe die gefundene Lösung in den Schwerpunkt.

Spezialfall in Schritt 2: Minimierung eingeschränkt auf Hyperebenen, die durch den Ursprung gehen, d.h. $\mathbf{e}^* \mathbf{x} = 0$:

$$\min_{\mathbf{e}} \sum_{i=1}^{n} (\mathbf{e}^* \mathbf{a}_i)^2, \quad \text{wobei } \|\mathbf{e}\|_2^2 = 1.$$

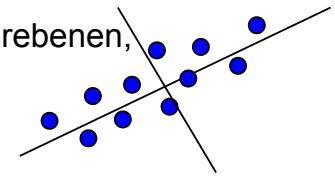
Bemerkung: Bezogen auf Schritt 2 bezeichnen nun $\mathbf{a}_i$ die in den Schwerpunkt verschobenen Punkte.

Lösung des Spezialfalls:

1. Elimination der Nebenbedingung durch Einführung eines Lagrange-Multiplikators λ: $\min_{\mathbf{e},\lambda} \underbrace{\sum_{i=1}^{n} (\mathbf{e}^* \mathbf{a}_i)^2}_{f} - \underbrace{\lambda(\|\mathbf{e}\|_2^2 - 1)}_{g}.$
2. Bestimmung der optimalen Lösungen durch partielles Ableiten nach $\mathbf{e}$ und λ und durch Nullsetzen. Dies ergibt

   $$\mathbf{A}\mathbf{A}^* \mathbf{e} = \lambda \mathbf{e}, \|\mathbf{e}\|_2^2 = 1$$

   wobei $\mathbf{A}$ die Matrix, deren i-ter Spaltenvektor $\mathbf{a}_i$ ist.
3. Lösungen des Gleichungssystems: Die auf 1 normierten Eigenvektoren der Matrix $\mathbf{C} := \mathbf{A}\mathbf{A}^*$, d.h. es m Lösungen gibt, deren Koeffizientenvektoren senkrecht aufeinander stehen. Sie entsprechen m Hyperebenen, die jeweils eine lokal optimale Lösung darstellen.



Wert der Quadratsumme der Abstände der Punkte von den Hyperebenen

Für die Quadratsumme aller Abstände gilt

$$\sum_{i=1}^{n} {d_{i,j}}^2 = \lambda_j.$$

Begründung:

Sei $d_{i,j}$ der Abstand des Punktes $\mathbf{a}_i$ von der Ebene von $\mathbf{e}_j$.
Dann gilt

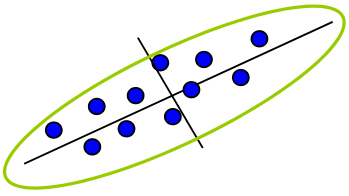
$$d_{i,j} = \mathbf{e}_j^* \, \mathbf{a}_i \text{ und } \mathbf{d}_j^* = \mathbf{e}_j^* \, \mathbf{A},$$

wobei $\mathbf{d}_j := (d_{i,j})$ der Vektor der Abstände aller $\mathbf{a}_i$ von $\mathbf{e}_j$.

Für die Quadratsumme aller Abstände gilt dann

$$\sum_{i=1}^{n} {d_{i,j}}^2 = \mathbf{d}_j^* \mathbf{d}_j = \mathbf{e}_j^* \mathbf{A}\mathbf{A}^* \mathbf{e}_j = \lambda_j.$$

Hauptachsen



Die Eigenvektoren $\mathbf{e}_j$ können als Richtungen der Hauptachsen eines Hyperellipsoids aufgefasst werden.

Natürliche Wahl der Länge der Hauptachse j aufgrund des Wertes der Quadratsumme:

Wurzel der Quadratsumme der Abstände

$\sqrt{\lambda_j}$, wobei λ_j den Eigenwert zum Eigenvektor $\mathbf{e}_j$ bezeichnet.

Dieses Ellipsoid charakterisiert die Verteilung der Punkte $\mathbf{a}_i$ im Raum.
Bilden die Punkte etwa eine langgestreckte Punktewolke, so wird das Ellipsoid analog in diese Richtung gestreckt sein.

Bemerkung: Die Quadratsumme der Abstände drückt die Streuung der Punkte aus. Die Division der Quadratsumme durch die Anzahl der Punkte ergibt die statistische Größe „Varianz“.

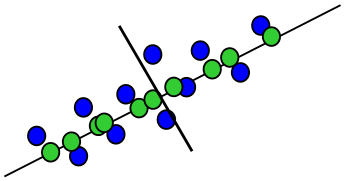
Hauptachsentransformation

Die Koordinatentransformation in das Koordinatensystem, das durch die Hauptachsen des Hyperellipsoids definiert wird, heißt *Hauptachsentransformation* (engl. *Principal Component Analysis (PCA)*).

Projektion auf einen Unterraum

Sei $\hat{\mathbf{a}}_i$ die Projektion von $\mathbf{a}_i$ auf den Unterraum, der von den ersten k Hauptachsen aufgespannt wirC. Dann gilt

$$\sum_{i=1}^{n} ||\mathbf{a}_i - \hat{\mathbf{a}}_i||^2 = \sum_{j=k+1}^{m} \lambda_j$$



Begründung:

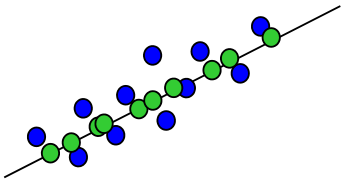
In der Darstellung von $\mathbf{a}_i$ bezüglich Basis der Hauptachsen fallen bei der Projektion die Komponenten von $\mathbf{a}_i$ weg, die nicht zu den ersten k Hauptachsen gehören. Die Summe der Quadrate der Länge der Komponenten aller $\mathbf{a}_i$ in Richtung einer solchen Hauptachse j ist gleich $\mathbf{d}_j^* \mathbf{d}_j = \lambda_j$.

Damit ist die Summe der Quadrate aller Längen gleich

$$\sum_{j=k+1}^{m} \lambda_j.$$

Dimensionsreduktion

Falls eine gegebene Punktmenge näherungsweise nur einen k-dimensionalen Unterraum aufspannt, kann die Dimension durch Projektion auf diesen Unterraum reduziert werden.



In diesem Fall werden die Eigenvektoren zu den k größten Eigenwerten die Basis dieses Unterraums bilden.

Der Fehler, ausgedrückt in der Summe der Quadrate der Abstände zwischen Originalpunkt und projiziertem Punkt, ist dann gemäß der Rechnung auf der vorigen Seite gleich $\sum_{j=k+1}^{m} \lambda_j$.

Anhand der Größe der Eigenwerte lässt sich also entscheiden, ob eine Dimensionsreduktion durch Projektion angebracht ist.

Alternativer Ansatz zur Hauptachsentransfomation

Singulärwertzerlegung einer $m \times n$-Matrix $\mathbf{A}$

Eine Darstellung der Form

$$\mathbf{U}^*\mathbf{AV} = \mathbf{D}$$

wobei

$$\mathbf{D}=\mathrm{diag}(\sigma_1, \ldots, \sigma_p),\ p = \min\{m, n\},\ \sigma_1 \geq \ldots \geq \sigma_p \geq 0,$$

eine Pseudodiagonalmatrix

$\mathbf{U}$, $\mathbf{V}$ orthogonale $m \times m$ beziehungsweise $n \times n$-Matrizen.

„Pseudodiagonalmatrix“ heißt, dass die linke obere $p \times p$-Untermatrix eine Diagonalmatrix mit Diagonalwerten λ_i ist und die gemäß Dimension noch verbleibenden restlichen Zeilen bzw. Spalten nur Nulleinträge enthalten.

Existenz der Singulärwertzerlegung U*AV = D einer $m \times n$-Matrix $\mathbf{A}$

Aus der Definition der Singulärwertzerlegung folgt

$\mathbf{A}\,\mathbf{v}_i = \sigma_i \mathbf{u}_i, \quad \mathbf{A}^*\,\mathbf{u}_i = \sigma_i \mathbf{v}_i$
für $i = 1, \ldots, \min\{m, n\}$.

Daraus ergibt sich

$$\mathbf{A}^*\mathbf{A}\mathbf{v}_i = \sigma_i^2 \mathbf{v}_i, \quad \mathbf{A}\mathbf{A}^*\mathbf{u}_i = \sigma_i^2 \mathbf{u}_i$$

Das bedeutet, dass $\mathbf{v}_i$ ein Eigenvektor der Matrix $\mathbf{A}^*\mathbf{A}$ zum Eigenwert σ_i^2 und $\mathbf{u}_i$ ein Eigenvektor der Matrix $\mathbf{A}\mathbf{A}^*$ zum Eigenwert σ_i^2 ist,
$i = 1, \ldots, \min\{m, n\}$.

Beziehung der Singulärwertzerlegung zur Hauptachsentransformation

Sei

$$\mathbf{A}_k = \sum_{j=1}^{k} \sigma_j \mathbf{u}_j \mathbf{v}_j^*$$

wobei

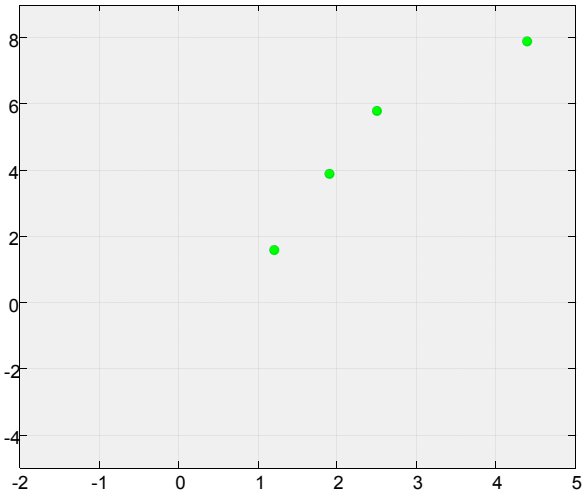
- $\mathbf{u}_j$, $\mathbf{v}_j$ der j-te Spaltenvektor von $\mathbf{U}$ beziehungsweise von $\mathbf{V}$ und k kleiner als der Rang von $\mathbf{A}$ ist und die Eigenwerte nach Größe fallend indiziert sinC.
- Die Spaltenvektoren von $\mathbf{A}_k$ sind Approximationen der Spaltenvektoren $\mathbf{a}_i$ von $\mathbf{A}$, die als Summe der Anteile von $\mathbf{a}_i$ längs der k ersten Hauptachsen des Ellipsoids der Hauptachsentransformation ergeben.
- Es gilt $||\mathbf{A} - \mathbf{A}_k||_2 = \sigma_{k+1}$,
  wobei $||\mathbf{A}||_2 := \max\{|\sigma| \mid \sigma \text{ Eigenwert von } \mathbf{A}\} = \max_{||\mathbf{x}||_2=1} ||\mathbf{A}\mathbf{x}||_2$
  Dies ist eine Approximation der Punkte $\mathbf{a}_i$ durch ihre Projektion in den Unterraum, der von den k Hauptachsen gebildet wirC.

1. Repräsentation der Messungen/Objekte in einer Datenmatrix

Matrix $\mathbf{A}_{m\,x\,n}$ repräsentiert n Vektoren $\mathbf{a}_i \in \Re^m$

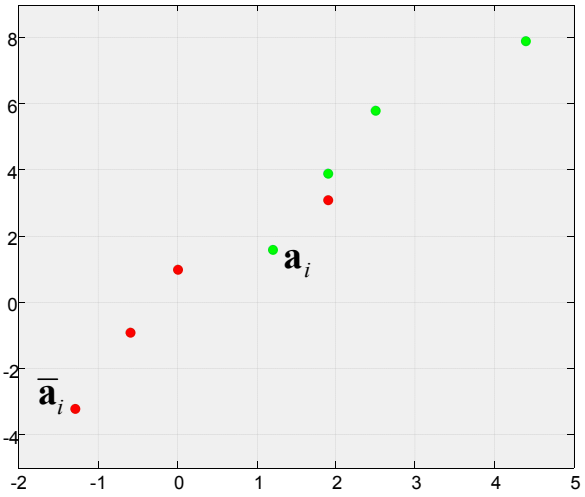
$$\mathbf{A}_{2x4} = \begin{pmatrix} 1.2 & 1.9 & 2.5 & 4.4 \\ 1.6 & 3.9 & 5.8 & 7.9 \end{pmatrix}$$

 # Objekte (n) (→) *# Merkmale (m)* (↓)



2. Messungen/Objekte in den Ursprung verschieben

- Bzgl. Schwerpunkt $\mathbf{m}_A = \frac{1}{n}\sum_{i=1}^{n}\mathbf{a}_i = \begin{pmatrix} 2.5 \\ 4.8 \end{pmatrix}$ Objekte verschieben $\overline{\mathbf{a}}_i = \mathbf{a}_i - \mathbf{m}_A$
- Datenmatrix der zentrierten Objekte $\overline{\mathbf{A}}_{2x4} = \begin{pmatrix} -1.3 & -0.6 & 0 & 1.9 \\ -3.2 & -0.9 & 1.0 & 3.1 \end{pmatrix}$



3. Berechnung der Eigenvektoren und Eigenwerte

a.) Nullstellen des charakteristischen Polynoms

b.) Singulärwertzerlegung (**S**ingular **V**alue **D**ecomposition)

Faktorisierung der zentrierten Datenmatrix $\overline{\mathbf{A}}_{m\,x\,n}$ gemäß:

$$\overline{\mathbf{A}}_{m\,x\,n} = \mathbf{U}_{m\,x\,m} \cdot \mathbf{\Sigma}_{m\,x\,n} \cdot \mathbf{V}^T_{n\,x\,n}$$

Bemerkungen

- Spalten von **U** sind orthogonale Eigenvektoren von $\mathbf{AA}^T$.
- Spalten von **V** sind orthogonale Eigenvektoren von $\mathbf{A}^T\mathbf{A}$
- Eigenwerte $\lambda_1, \ldots, \lambda_{Rang}$ von $\mathbf{AA}^T$ und $\mathbf{A}^T\mathbf{A}$ mit

$$\mathbf{\Sigma}_{m\,x\,n} = diag(\sqrt{\lambda_1}, \mathrm{K}\,, \sqrt{\lambda_{Rang}})$$

3. Berechnung der Eigenvektoren und Eigenwerte

b.) Singulärwertzerlegung (**S**ingular **V**alue **D**ecomposition)

$$\overline{\mathbf{A}}_{m\,x\,n} = \mathbf{U}_{m\,x\,m} \cdot \mathbf{\Sigma}_{m\,x\,n} \cdot \mathbf{V}^T_{n\,x\,n}$$

$$\begin{pmatrix} -1.3 & -0.6 & 0 & 1.9 \\ -3.2 & -0.9 & 1.0 & 3.1 \end{pmatrix} = \begin{pmatrix} -0.6636 & 0.4212 & 0.4789 & 0.3909 \\ -0.2068 & -0.2184 & -0.6579 & 0.6904 \\ 0.1725 & -0.7155 & 0.5687 & 0.3672 \\ 0.6979 & 0.5127 & 0.1199 & 0.4855 \end{pmatrix} \cdot$$

$$\begin{pmatrix} 5.1896 & 0 \\ 0 & 0.6228 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \cdot$$

$$\begin{pmatrix} 0.4457 & 0.8952 \\ 0.8952 & -0.4457 \end{pmatrix}$$

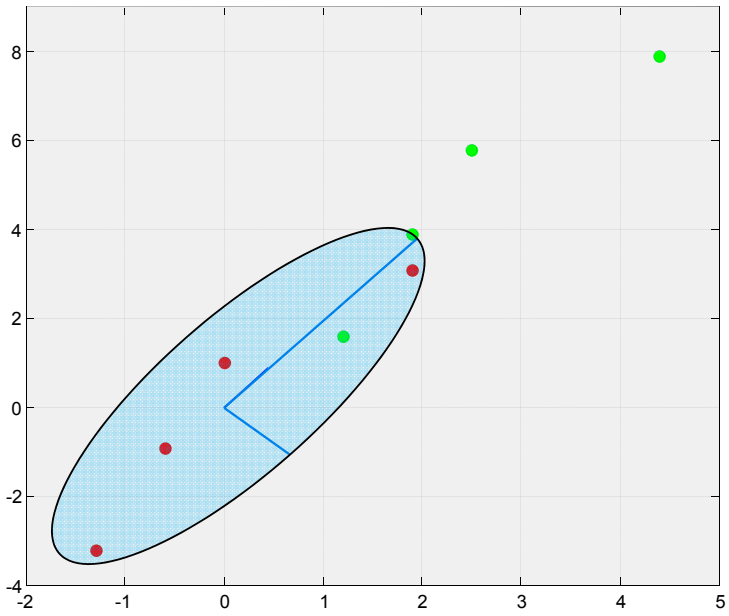
$\lambda_1 = 26.93 \leftarrow$ 5.1896

$\lambda_2 = 0.39 \leftarrow$ 0.6228

$\mathbf{e}_1 \leftarrow$ (0.4457, 0.8952) (0.8952, −0.4457) $\rightarrow \mathbf{e}_2$

3. Berechnung der Eigenvektoren und Eigenwerte

b.) Singulärwertzerlegung (**S**ingular **V**alue **D**ecomposition)



4. Projektion auf einen Unterraum

a.) Bestimmung der wesentlichen Komponenten

$$EW = \langle \underbrace{\lambda_1, \ldots, \lambda_l}_{\text{wesentliche Komponenten}}, \ldots, \lambda_{Rang} \rangle \qquad EW = \langle \underbrace{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_l}_{\text{wesentliche Komponenten}}, \ldots, \boldsymbol{e}_{Rang} \rangle$$

b.) Projektion auf Unterraum

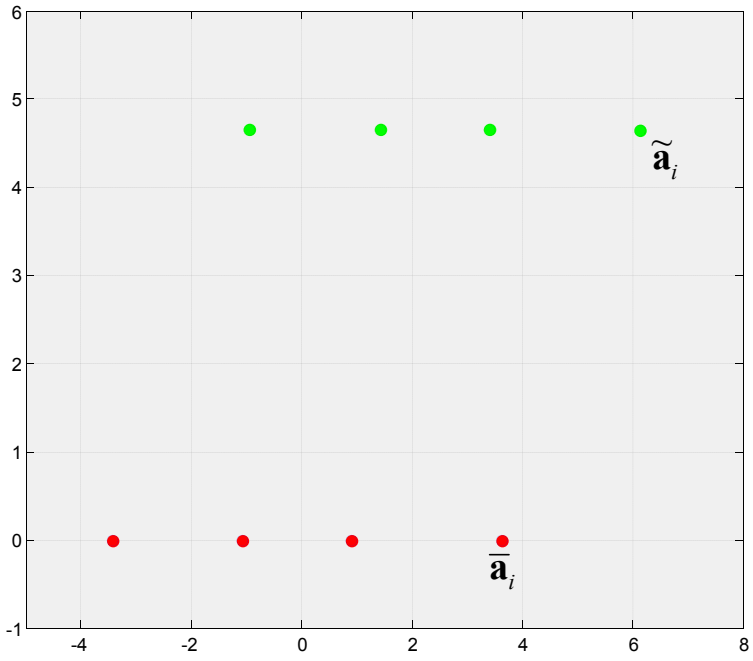
$$\widetilde{\mathbf{A}}_{l\,x\,n} = \mathbf{E}_{l\,x\,m} \cdot \overline{\mathbf{A}}_{m\,x\,n} \qquad \boldsymbol{E}_{l\times m} = \left(\boldsymbol{e}_1^T \ldots \boldsymbol{e}_l^T\right)^T$$

$$\begin{pmatrix} -3.44 & -1.07 & 0.89 & 3.62 \end{pmatrix} = \begin{pmatrix} 0.44 & 0.89 \end{pmatrix} \cdot \begin{pmatrix} -1.3 & -0.6 & 0 & 1.9 \\ -3.2 & -0.9 & 1.0 & 3.1 \end{pmatrix}$$

(c.) Verschiebung in den Schwerpunkt)

$$\widetilde{\mathbf{a}}_i = \overline{\mathbf{a}}_i + \mathbf{m}_A$$

4. Projektion auf einen Unterraum



C.1. Dimensionsreduktion
- C.1.1. Hauptachsentransformation
- C.1.1.1. Lagrange-Multiplikatoren (Exkurs)
- C.1.1.2. Ausgleichsebenen
- C.1.1.3. Optimierungsproblem
- C.1.1.4. Hauptachsen
- C.1.1.5. Projektion
- C.1.1.6. Dimensionsreduktion
- C.1.1.7. Singularwertzerlegung
- C.1.2. Beispiel

C.2. Kohonenkarten
- C,2.1. Einleitung
- C.2.2. Aufbau der Karte
- C.2.3. Anlernen
- C.2.4. Beispiele

C.3. Quantisierung
- C.3.1. Vektorquantisierung
- C.3.2. Hyper-Octree-Vektorquantisierung
- C.3.3. Median-Schnitt-Vektorquantisierung

C.4. Clustering
- C.4.1. k-Mittelwert-Clustering
- C.4.2. Hierarchisches Clustering
- C.4.3. Dichtebasiertes Clustering

Kohonenkarte (Self Organizing Map – SOM)

- definiert eine Abbildung eines d-dimensionalen Raums auf einen k-dimensionalen Raum, $k < C$.
- versucht, Nachbarschaften möglichst gut zu erhalten, indem dichter liegende Häufungen von Punkten möglichst benachbart abgebildet werden.

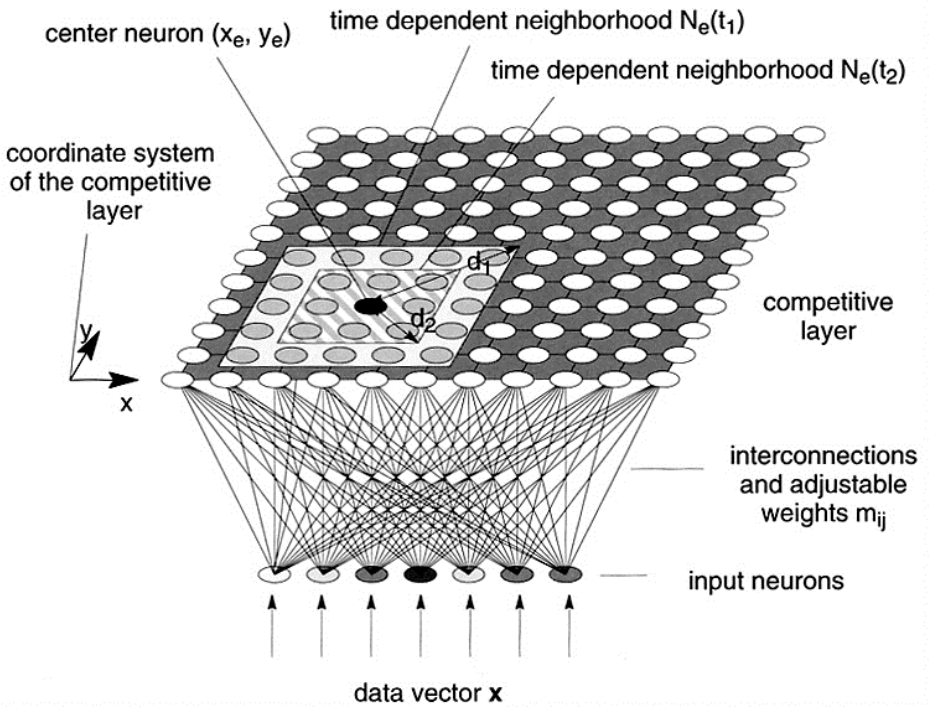
Anwendung bei der Visualisierung

- Reduktion der Dimension auf die Möglichkeiten der Darstellung ($k = 2$ oder $k = 3$)

Literatur:

- T. Kohonen, The Self-Organizing Map, Proceedings of the IEEE, Vol. 78(9), 1990, 1464-1480
- T. Kohonen, Self-Organization and Associative Memory, 3rd eC., Springer Series on Infomation Sciences 8, Springer-Verlag, Berlin, 1989

Aufbau einer Kohonenkarte



Kohonenkarte

- **Aufbau:** $n \times m$-Gitter aus Neuronen, jedem Neuron ist ein d-dimensionaler Punkt (auch „Gewicht“ genannt) zugeordnet
- **Initialisieren:** durch zufällige Zuordnung von d-dimensionalen Punkten an die Neuronen
- **Anlernen:** durch eine Menge von zufällig gewählten Anlerndatenpunkten aus einer gegebenen Punktmenge:

  Finde das bestpassende Neuron für einen Anlerndatenpunkt und verändere sein zugeordnetes Gewicht sowie den seiner Nachbarn etwas in Richtung des Anlerndatenpunktes.
- **Clustering der Neuronen:** nach Abstand der zugeordneten Punkte.

Algorithmus (Anlernen einer Kohonenkarte)

Eingabe: T Anlernpunkte, Konstanten η_0, d_0.
Ausgabe: Gewichte der Neuronen der Kohonenkarte

BEGIN
$t := 0;$

Für jeden Anlerndatenpunkt $\mathbf{x}$ {

bestimme das Neuron e, dessen Gewicht $\mathbf{m}_e$ den kleinsten Euklidschen Abstand zu $\mathbf{x}$ hat;

für alle Neuronen $i \in N_e(t)$ (s.u.) {

$\eta(t) := \eta_0(1 - \frac{t}{T});$
$\Delta\mathbf{m}_i := \eta(t)(\mathbf{x} - \mathbf{m}_i);$
$\mathbf{m}_i := \mathbf{m}_i + \Delta\mathbf{m}_i;$

$t := t + 1;$}
END.

Definition der Nachbarschaft $N_e(t)$

$$N_e(t) = \big(x_e - d(t), x_e + d(t)\big) \times \big(y_e - d(t), y_e + d(t)\big)$$

wobei

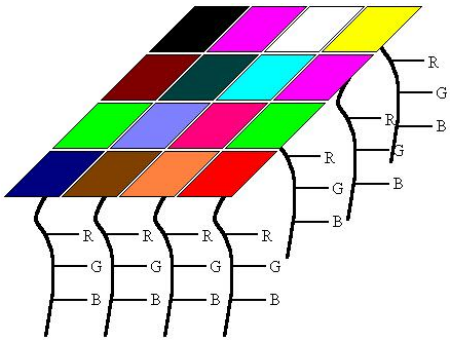
$$d(t) := d_0 \cdot (1 - \frac{t}{T}), \quad t \in [0, T]$$

Auswahl der Anlernpunkte:

Die Menge der Anlernpunkte kann etwa die gegebene Punktmenge oder eine zufällig ausgewählte Teilmenge davon sein.

Literatur:
M. Groß, Visual Computing, Springer-Verlag, Berlin, 1994, Kap. 6.3.2

Beispiel: Abbildung des dreidimensionalen RGB-Farbraums ins Zweidimensionale

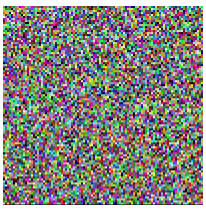


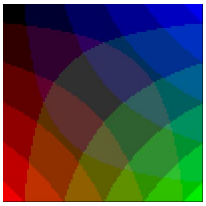
Aufbau des Netzes:

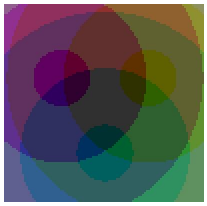
2D-Gitter aus Neuronen mit Gewichten, die Farben repräsentieren

Initialisierung des Netzes:

- *Möglichkeit A:* zufällig
  Nachteil: Hohe Anzahl von Iterationen notwendig, um am Ende eine gute Karte zu erhalten
- *Möglichkeit B:* Mischen von Grundfarben, die sich von je einer Quelle ausbreiten
  Vorteil: Bessere Ähnlichkeit benachbarter Farben bewirkt schnellere Konvergenz

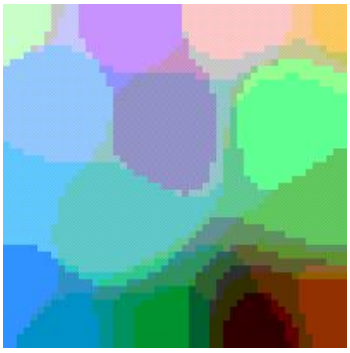






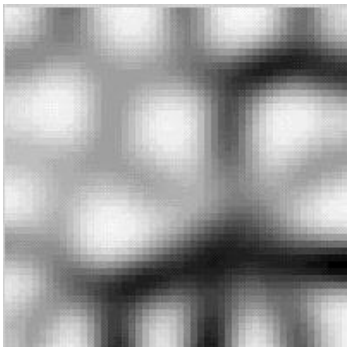
Iteration:

- Eine Ergebniskarte nach 500 Iterationen.
- Nicht überall entspricht die Anordnung der ursprünglichen Nachbarschaft (dies ist auch nicht möglich). Rechts unten ist die Anordnung besonders schlecht.



Ähnlichkeitsbild

- visualisiert den Grad der Nachbarschaft der Zuordnung der Neuronenkarte.
- *Nachbarschaftswert eines Neurons:* der mittlere Abstand seines Gewichts zu den Neuronen in der Umgebung
- *Visualisierung des Nachbarschaftswertes:* *dunkel:* großer mittlerer Abstand *hell:* kleiner mittlerer Abstand



Quelle: T. Germano, Self Organizing Maps, http://davis.wpi.edu/~matt/courses/soms

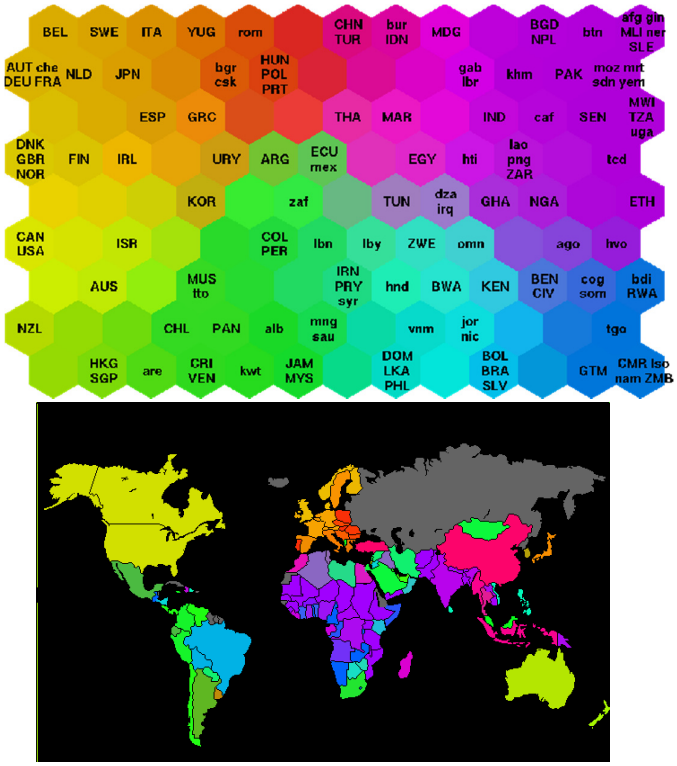
Beispiel: Anordnung von Ländern nach Lebensqualität

- Grundlage: 39 Merkmale, z.B. Gesundheit, Schule, etc.
- Anordnung von Länder nach Ähnlichkeit des Merkmalsvektors
- Farbkodierung der Cluster mit Farben unterschiedlicher Helligkeit, sodass die Helligkeit stetig auf der Karte verteilt ist.
- Darstellung der Länder mit der entsprechenden Farbe auf einer geographischen Landkarte.

Quelle:

http://www.cis.hut.fi/research/som-research/worldmap.html

Beispiel: Anordnung von Ländern nach Lebensqualität



The Country Names

ARG	Afghanistan	GTM	Guatemala	NZL	New Zealand
AGO	Angola	HKG	Hong Kong	OAN	Taiwan, China
ALB	Albania	HND	Honduras	OMN	Oman
ARE	United Arab Emirates	HTI	Haiti	PAK	Pakistan
ARG	Argentina	HUN	Hungary	PAN	Panama
AUS	Australia	HVO	Burkina Faso	PER	Peru
AUT	Austria	IDN	Indonesia	PHL	Philippines
BDI	Burundi	IND	India	PNG	Papua New Guinea
BEL	Belgium	IRL	Ireland	POL	Poland
BEN	Benin	IRN	Iran, Islamic Rep.	PRT	Portugal
BGD	Bangladesh	IRQ	Iraq	PRY	Paraguay
BGR	Bulgaria	ISR	Israel	ROM	Romania
BOL	Bolivia	ITA	Italy	RWA	Rwanda
BRA	Brazil	JAM	Jamaica	SAU	Saudi Arabia
BTN	Bhutan	JOR	Jordan	SDN	Sudan
BUR	Myanmar	JPN	Japan	SEN	Senegal
BWA	Botswana	KEN	Kenya	SGP	Singapore
CAF	Central African Rep.	KHM	Cambodia	SLE	Sierra Leone
CAN	Canada	KOR	Korea, Rep.	SLV	El Salvador
CHE	Switzerland	KWT	Kuwait	SOM	Somalia
CHL	Chile	LAO	Lao PDR	SWE	Sweden
CHN	China	LBN	Lebanon	SYR	Syrian Arab Rep.
CIV	Cote d'Ivoire	LBR	Liberia	TCD	Chad
CMR	Cameroon	LBY	Libya	TGO	Togo
COG	Congo	LKA	Sri Lanka	THA	Thailand
COL	Colombia	LSO	Lesotho	TTO	Trinidad and Tobago
CRI	Costa Rica	MAR	Morocco	TUN	Tunisia
CSK	Czechoslovakia	MDG	Madagascar	TUR	Turkey
DEU	Germany	MEX	Mexico	TZA	Tanzania
DNK	Denmark	MLI	Mali	UGA	Uganda
DOM	Dominican Rep.	MNG	Mongolia	URY	Uruguay
DZA	Algeria	MOZ	Mozambique	USA	United States
ECU	Ecuador	MRT	Mauritania	VEN	Venezuela
EGY	Egypt, Arab Rep.	MUS	Mauritius	VNM	Viet Nam
ESP	Spain	MWI	Malawi	YEM	Yemen, Rep.
ETH	Ethiopia	MYS	Malaysia	YUG	Yugoslavia
FIN	Finland	NAM	Namibia	ZAF	South Africa
FRA	France	NER	Niger	ZAR	Zaire
GAB	Gabon	NGA	Nigeria	ZMB	Zambia
GBR	United Kingdom	NIC	Nicaragua	ZWE	Zimbabwe
GHA	Ghana	NLD	Netherlands		
GIN	Guinea	NOR	Norway		
GRC	Greece	NPL	Nepal		

C.1. Dimensionsreduktion
C.1.1. Hauptachsentransformation
C.1.1.1. Lagrange-Multiplikatoren (Exkurs)
C.1.1.2. Ausgleichsebenen
C.1.1.3. Optimierungsproblem
C.1.1.4. Hauptachsen
C.1.1.5. Projektion
C.1.1.6. Dimensionsreduktion
C.1.1.7. Singularwertzerlegung
C.1.2. Beispiel
C.2. Kohonenkarten
C,2.1. Einleitung
C.2.2. Aufbau der Karte
C.2.3. Anlernen
C.2.4. Beispiele
C.3. Quantisierung
C.3.1. Vektorquantisierung
C.3.2. Hyper-Octree-Vektorquantisierung
C.3.3. Median-Schnitt-Vektorquantisierung
C.4. Clustering
C.4.1. k-Mittelwert-Clustering
C.4.2. Hierarchisches Clustering
C.4.3. Dichtebasiertes Clustering

Datenreduktion:

Ersetzung einer gegebenen Datenmenge durch eine kleinere Datenmenge mit ähnlichen Eigenschaften, z.B. ähnliche Dichteverteilung der Punkte

Verbreitete Methode:

Vektorquantisierung

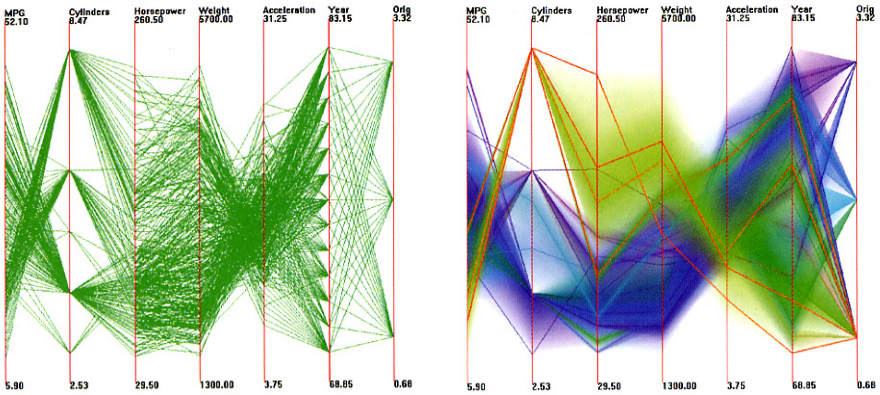
Clustering:

Gruppierung der Elemente einer gegebenen Punktmenge in Teilmengen (Cluster), so dass die Punkte innerhalb jedes Clusters zueinander in engerer Beziehung als zu Punkten in den anderen Clustern sinC.

Anwendung für die Visualisierung:

Strukturierung einer gegebenen Punktmenge zum Auffinden und visuellen Hervorheben interessanter Bereiche

Beispiel:



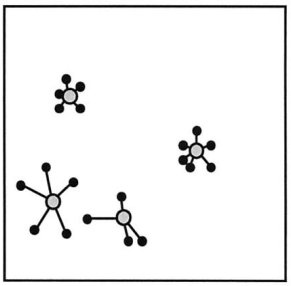
ungeclustert geclusterter Ausschnitt

Literatur:

J.C.Dubes, Algorithms for clustering data, Prentice Hall, Englewood Cliffs, NJ, 1988

Versionen der Vektorquantisierung

- **„k-Vektorquantisierung“:** Auffinden von k Punkten, $k > 0$ eine vorgegebene Konstante, deren räumliche Verteilung denen der gegebenen Punkte entspricht

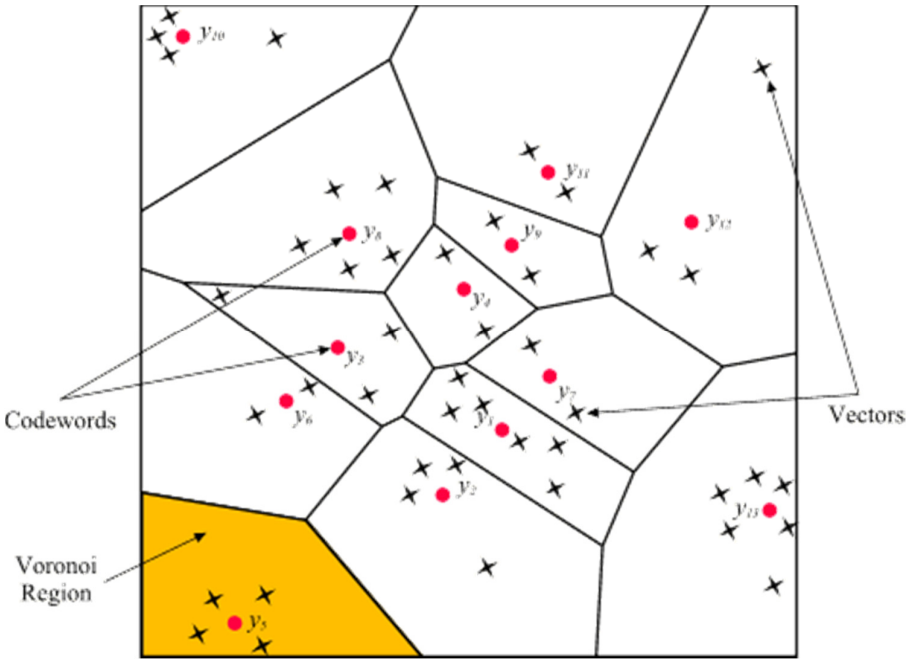


- **„Vektorquantisierung“:** Auffinden einer endlichen Menge von Punkten $\mathbf{q}$, sodass eine disjunkte Zuordnung von maximal p räumlich benachbarten Punkten aus der gegebenen Punktmenge zu jedem $\mathbf{q}$ existiert, $p > 0$ eine vorgegebene Konstante.

Literatur:

A. Gersho, R. Grey, Vector quantization and signal compression, Kluwer Publishing Company, Boston, 1992

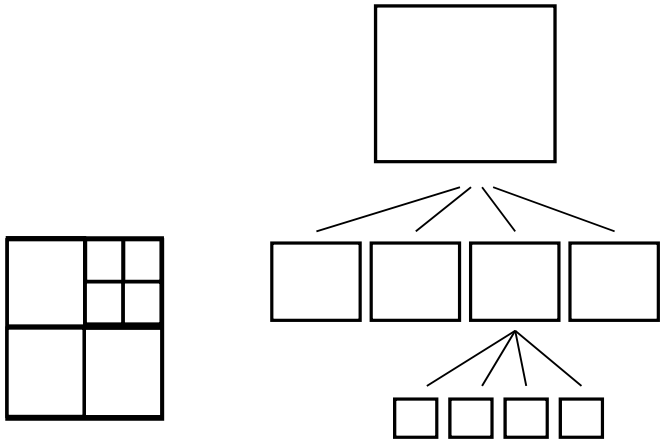
Vektorquantisierung



Hyperoctree-Struktur:

- Ein Baum mit maximal 2^d Nachfolgern an jeden Knoten.
- Jeder Knoten repräsentiert einen d-dimensionalen Quader.
- Die Quader der Nachfolger eines Knotens ergeben sich durch Teilung jeder Seite des Quaders des Knotens in der Mitte.

Beispiel: Quadtree



ALGORITHMUS (Hyper-Octree-Vektorquantisierung)

Eingabe: Eine d-dimensionale Punktmenge P, eine Schranke $p > 0$

Ausgabe: Ein d-dimensionaler Octree mit der Eigenschaft, dass jedes Blatt höchstens p Elemente der gegebenen Punktmenge enthält.

BEGIN

für alle Punkte $\mathbf{x} \in P$: {

füge $\mathbf{x}$ in den Hyper-Octree ein;

solange das Blatt von $\mathbf{x}$ mehr als p Punkte enthält: {

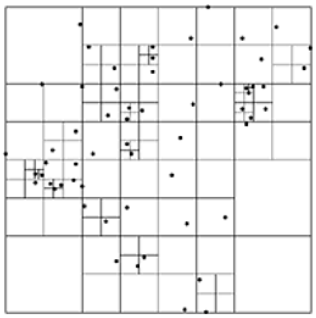
zerlege den Quader des Blattes in Unterquader, füge Knoten für die Unterquader ein, die mindestens einen der gegebenen Punkte enthalten und weise die Punkte des alten Blattes den neuen Blättern zu; }}

berechne den Mittelwert der Knoten in jedem Blatt und gib ihn aus;

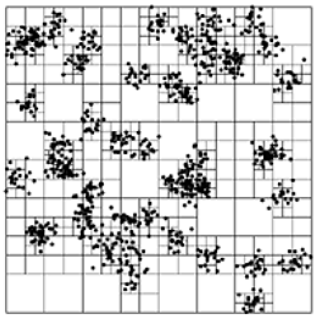
ENC.

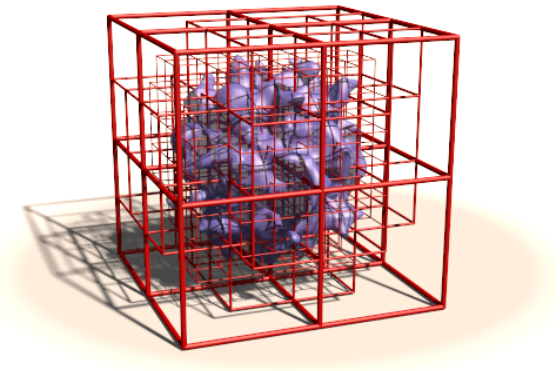
Bemerkung:

- Die Hyper-Octree-Vektorquantisierung generiert in Gebieten hoher Punktdichte eher Zellen kleiner Größe, in Gebieten geringer Punktdichte eher Zellen großer Größe. Dementsprechend entstehen in Gebieten hoher Punktdichte eher mehr Quantisierungspunkte als in Gebieten geringer Punktdichte.
- Auf diese Weise wird eine dichteangepasste Reduktion der gegebenen Punktmenge erzielt.



2D-Fall





3D-Fall

ALGORITHMUS (k-Hyper-Octree-Vektorquantisierung)

Eingabe: Eine d-dimensionale Punktmenge P, eine Schranke $k > 0$.

Ausgabe: Ein d-dimensionaler Octree mit der Eigenschaft, dass jedes Blatt genau einen Punkt sowie eine natürliche Zahl als Gewicht enthält.

BEGIN

für alle Punkte $\mathbf{x} \in P$: {

füge $\mathbf{x}$ in den Hyper-Octree ein;

solange das Blatt von $\mathbf{x}$ mehr als einen Punkt enthält {

zerlege den Quader des Blattes in Unterquader, füge Knoten für die Unterquader ein, die mindestens einen der gegebenen Punkte enthalten und weise die Punkte des alten Blattes den neuen Blättern zu; }

ALGORITHMUS (k-Hyper-Octree-Vektorquantisierung)

[…Fortsetzung…]

Wenn der Baum mehr als k Blätter enthält: {

entnehme zwei Punkte aus dem Baum, die in Blättern mit gleichen Vorfahren und auf der tiefstmöglichen Schicht liegen, und entferne deren Blätter;

berechne das gewichtete Mittel beider Punkte, wobei die beiden gespeicherten Gewichte verwendet werden;

berechne ein neues Gewicht als Summe der beiden Gewichte;

Füge den gemittelten Punkt und das Gewicht wie einen ursprünglichen Punkt in den Baum ein und teile die Blätter analog, bis der Punkt und sein Gewicht alleine in einem Blatt liegen; } }

ENC.

Bemerkungen:

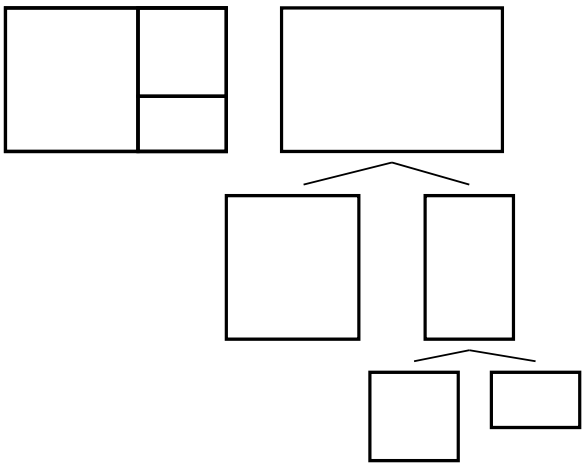
- Die bedingte Anweisung auf dieser Seite tritt ein, wenn der Baum $k+1$ Blätter enthält. Nach ihrer Ausführung hat der Baum k Blätter.

- Die k-Hyper-Octree-Vektorquantisierung reduziert insbesondere in Gebieten hoher Punktdichte die Anzahl der Punkte. Punkte in Gebieten geringer Punktdichte werden noch durch Quantisierungspunkte repräsentiert.

- Durch die mitgespeicherten Gewichte bleibt die ursprüngliche Dichte erkennbar.

Median-Schnitt-Baum - Struktur:

- Ein Baum mit maximal 2 Nachfolgern an jedem Knoten
- Jeder Knoten repräsentiert einen d-dimensionalen Quader
- Die Quader der Nachfolger eines Knotens ergeben sich durch Teilung der längsten Seite des Quaders des Knotens am Median der in ihm gespeicherten Punkte, d.h. links und rechts der induzierten Ebene liegen etwa gleich viele Punkte (es wird vorausgesetzt, dass keine zwei Punkte in einem Koordinatenwert übereinstimmen).



ALGORITHMUS (Median-Schnitt (median cut)-Vektorquantisierung)

Eingabe: Eine d-dimensionale Punktmenge P, eine Schranke $p > 0$

Ausgabe: Ein d-dimensionaler Median-Schnitt-Baum mit der Eigenschaft, dass jedes Blatt höchstens p Elemente der gegebenen Punktmenge enthält.

BEGIN

initialisiere die Wurzel des Baumes und füge die Punkte von P ein;

Solange es noch ein Blatt mit mehr als p Punkten gibt {

zerlege das Blatt nach der Medianregel in zwei neue Knoten, übernehme seine Punkte in diese und füge die beiden neuen Knoten als Nachfolger an das ehemalige Blatt an; }

berechne den Mittelwert der Knoten in jedem Blatt und gib ihn aus;

ENC.

Bemerkungen:

- Die Größe der Blattzellen der Median-Schnitt-Vektorquantisierung passt sich der Dichte der gegebenen Punktmenge an. Damit liegen in Gebieten höherer Punktdichte eher mehr Quantisierungspunkte als in Gebieten geringer Punktdichte.
- Auf diese Weise wird eine dichteangepasste Reduktion der gegebenen Punktmenge erzielt.

ALGORITHMUS (k-Median-Schnitt-Vektorquantisierung)

Eingabe: Eine d-dimensionale Punktmenge P, eine Schranke $k > 0$.

Ausgabe: Ein d-dimensionaler Median-Schnitt-Baum mit k Blättern.

BEGIN

initialisiere die Wurzel des Baumes und füge die Punkte von P ein;

Solange die Blattanzahl kleiner als k ist {

zerlege das Blatt mit der größten Anzahl von Punkten nach der Medianregel in zwei neue Knoten, übernehme seine Punkte in diese und füge die beiden neuen Knoten als Nachfolger an das ehemalige Blatt an; }

berechne den Mittelwert der Knoten in jedem Blatt und gib ihn aus;

ENC.

Bemerkung:

Falls k eine Zweierpotenz ist, können die Blätter des Baumes etwa gleich viele der gegebenen Punkte enthalten. In diesem Fall wird gleichzeitig eine dichtebezogene Vektorquantisierung erzielt.

Quelle:
Paul Debevec, A Median Cut Algorithm for Light Probe Sampling, *SIGGRAPH 2005*

C.1. Dimensionsreduktion
C.1.1. Hauptachsentransformation
C.1.1.1. Lagrange-Multiplikatoren (Exkurs)
C.1.1.2. Ausgleichsebenen
C.1.1.3. Optimierungsproblem
C.1.1.4. Hauptachsen
C.1.1.5. Projektion
C.1.1.6. Dimensionsreduktion
C.1.1.7. Singularwertzerlegung
C.1.2. Beispiel
C.2. Kohonenkarten
C,2.1. Einleitung
C.2.2. Aufbau der Karte
C.2.3. Anlernen
C.2.4. Beispiele
C.3. Quantisierung
C.3.1. Vektorquantisierung
C.3.2. Hyper-Octree-Vektorquantisierung
C.3.3. Median-Schnitt-Vektorquantisierung
C.4. Clustering
C.4.1. k-Mittelwert-Clustering
C.4.2. Hierarchisches Clustering
C.4.3. Dichtebasiertes Clustering

Ziel des Clustering

- Identifikation einer endlichen Menge von Kategorien, Klassen oder Gruppen (*Cluster*) in den Daten
- Objekte im *gleichen* Cluster sollen möglichst ähnlich sein
- Objekte aus *verschiedenen* Clustern sollen möglichst unähnlich zueinander sein

Herausforderungen

- Cluster unterschiedlicher Größe, Form und Dichte
- hierarchische Cluster

→ unterschiedliche Clustering-Algorithmen

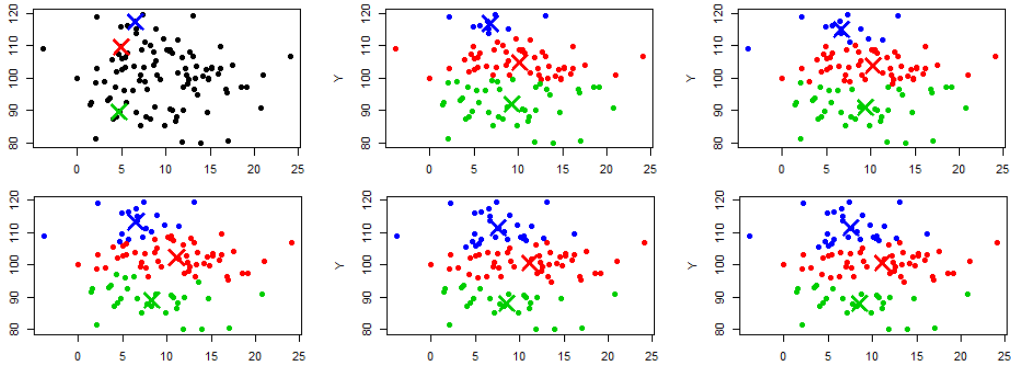
Clustering-Verfahren

- **Partitionierende Verfahren**
  - Parameter: Anzahl *k* der Cluster, Distanzfunktion
  - sucht ein „flaches“ Clustering in *k* Cluster mit minimalen Kosten
- **Hierarchische Verfahren**
  - Parameter: Distanzfunktion für Punkte und für Cluster
  - bestimmt Hierarchie von Clustern, mischt jeweils die ähnlichsten Cluster
- **Dichtebasierte Verfahren**
  - Parameter: minimale Dichte in einem Cluster, Distanzfunktion
  - erweitert Punkte um ihre Nachbarn solange Dichte groß genug
- **Andere Clustering-Verfahren**
  Fuzzy Clustering, Graph-theoretische Verfahren, neuronale Netze

Algorithmus

- Wähle zufällig k Clustermittelpunkte
- Iteriere
  - Für alle Objekte
    - Berechne Abstand jedes Objekts zu jedem Clustermittelpunkt
    - Weise Objekt seinem nächsten Clustermittelpunkt zu
  - Wenn sich keine Objektzuordnung mehr geändert hat, dann STOP
  - Für alle Cluster
    - Berechne neues Clusterzentrum

Beispiel



Clustering als Optimierung

Finde für eine Menge P von Objekten eine Zuordnung f in k Cluster, sodass $q_k(f)$ minimal ist

Clustergüte

Definition

Sei $f: P \to C$ mit $|C|$ beliebig. Sei $dist(\boldsymbol{p}, C_i)$ der mittlere Abstand von $\boldsymbol{p}$ zu allen Punkten eines Clusters C_i, dann gilt

- Abstand eines Objekts $\boldsymbol{p}$ zum Repräsentanten seines Clusters:
$$a(\boldsymbol{p}) = dist\big(\boldsymbol{p}, f(\boldsymbol{p})\big)$$
- Abstand zum Repräsentanten des „zweitnächsten“ Clusters:
$$b(\boldsymbol{p}) = \min dist(\boldsymbol{p}, C_i)) \text{ mit } C_i \neq f(\boldsymbol{p})$$
- Silhouette eines Punkte $\boldsymbol{p}$ ist $s(\boldsymbol{p}) = \frac{b(\boldsymbol{p}) - a(\boldsymbol{p})}{max\,(a(\boldsymbol{p}), b(\boldsymbol{p}))}$
- Silhouette von f ist $\sum s(\boldsymbol{p})$

Bedeutung

- $s(\boldsymbol{p}) \approx 0$: Punkt liegt zwischen zwei Clustern
- $s(\boldsymbol{p}) \to 1$: Punkt liegt näher am eigenen als am nächsten Cluster
- $s(\boldsymbol{p}) \to -1$: Punkt liegt näher am nächsten Cluster als am eigenen

Ähnlichkeitsmaße

- Numerische Werte
  - Abstandsmaße

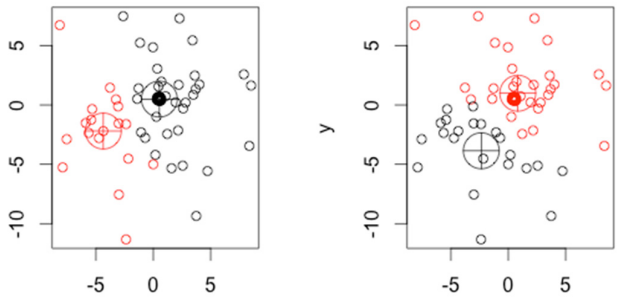
$$d_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}} = \|\boldsymbol{x} - \boldsymbol{y}\|_p$$

    - p=1: Manhattan Distanz
    - p=2: Euklidische Distanz

  - Winkelmaße
- Kategorielle Werte
- Mengenwertige Werte

Centroid eines Clusters (numerisch)

- Mittelwert
- Median
- Medoid:
  „mittlerste“ Punkt von *C*,
  der auch in *P* ist



ALGORITHMUS (k-Mittelwert (Means)-Clustering)

Eingabe: Eine d-dimensionale Punktmenge P, eine Schranke $k > 0$.

Ausgabe: k d-dimensionale Punkte, sogenannte Cluster-Zentren. Das Cluster eines Cluster-Zentrums $\mathbf{c}$ besteht aus den Punkten in P, die keinem anderen Cluster-Zentrum näher liegen als $\mathbf{c}$.

BEGIN

wähle eine Menge Q aus k initialen Cluster-Zentren;

Wiederhole:

Ordne jedem Punkt aus P einen Punkt in Q zu, von dem er nicht weiter entfernt ist als zu allen anderen Punkten in Q;

Berechne für alle dadurch entstehenden Cluster ein neues Cluster-Zentrum als Schwerpunkt der in dem Cluster liegenden Punkte;

ALGORITHMUS (k-Mittelwert (Means)-Clustering)

[…Fortsetzung…]

ersetze Q durch die Menge dieser neuen Cluster-Zentren;

bis keine merkliche Veränderung von Q mehr eintritt;

ENC.

Möglichkeiten der Initialisierung von Q:

- Wähle k Punkte in P zufällig.
- Berechne eine zufällige Zerlegung von P in k Cluster und nimm deren Schwerpunkte für Q.

Bemerkungen:

- Das Ziel der Iteration des k-Mittelwert-Clustering ist, die folgende Zielfunktion zu minimieren:

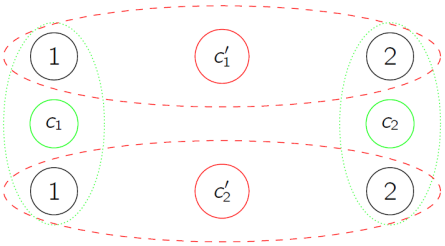
$$E(Q) = \sum_{\mathbf{q} \in Q} \sum_{\mathbf{p} \in V(\mathbf{q})} d(\mathbf{p}, \mathbf{q})^2$$

wobei

  - $V(\mathbf{q})$ die Menge der Punkte aus P bezeichnet, die im Voronoi-Gebiet von $\mathbf{q}$ liegen.
  - Das *Voronoi-Gebiet* von $\mathbf{q}$ bezüglich einer endlichen Punktmenge Q definiert ist als das Gebiet, bei dem $\mathbf{q}$ näher liegt als allen anderen Punkten in Q

- Das k-Mittelwert-Clustering kann auch als ein Verfahren zur k-Vektor-quantisierung aufgefasst werden.
- Zeitaufwand: $O(m\ n\ k)$, wobei m die Anzahl der Iterationen und n die Anzahl der Punkte in P bezeichnen.

Bemerkungen



- resultierende Cluster können stark von der Wahl der initialen Centroide abhängen

→ keine Garantie dafür, dass globales Minimum

- resultierende Anzahl an Clustern kann auch kleiner k sein

Verbesserungen

- mehrere Durchläufe mit unterschiedlichen Startkonfigurationen
- Abweichende Maße zur Abstands- und Centroidbestimmung
- initiale Centroide sollten möglichst weit voneinander entfernt sein

Clusterhierarchie - *Struktur*

- Ein Binärbaum, bei dem jedem Knoten eine Teilmenge der gegebenen d-dimensionalen Punktmenge zugeordnet ist
- Der Wurzel ist die gesamte gegebene Menge zugeordnet.
- Jedem Blatt ist genau ein Punkt der Eingabemenge zugeordnet und umgekehrt.
- Die zugeordnete Punktmenge eines inneren Knotens ergibt sich als Vereinigung der zugeordneten Punktmengen seiner Kinder.

Typen von Verfahren für hierarchisches Clustering

- **bottom-up** *(agglomerative)*: Fasse die Punkte der gegebenen Menge sukzessive zusammen
- **top-down** *(divisive)*: Teile die gegebene Punktmenge sukzessive auf

ALGORITHMUS (agglomeratives hierarchisches Clustering)

Eingabe: d-dimensionale Punktmenge P, Verschmelzungskostenfunktion C.

Ausgabe: Eine Clusterhierarchie für P

BEGIN

definiere für jeden Punkt in P ein eigenes Cluster, füge die Cluster in eine Clusterliste L ein und übernehme die Cluster als Blätter der Clusterhierarchie;

Solange L nicht leer ist: {

Finde ein Paar von Clustern in L, das die geringsten Verschmelzungskosten hat;

Entferne das gefundene Cluster-Paar aus L, vereinige die beiden Cluster zu einem neuen Cluster, füge das neue Cluster in L ein und übernimm das neue Cluster als Elternknoten der Knoten der beiden Cluster in die Clusterhierarchie; }

ENC.

Verschmelzungskostenfunktionen zweier Cluster C, C'

- ***kürzeste Punktdistanz:*** $d_{\text{min}}(C, C') := \text{min}_{\mathbf{p} \in C, \mathbf{p}' \varepsilon\, C'} d(\mathbf{p}, \mathbf{p}')$

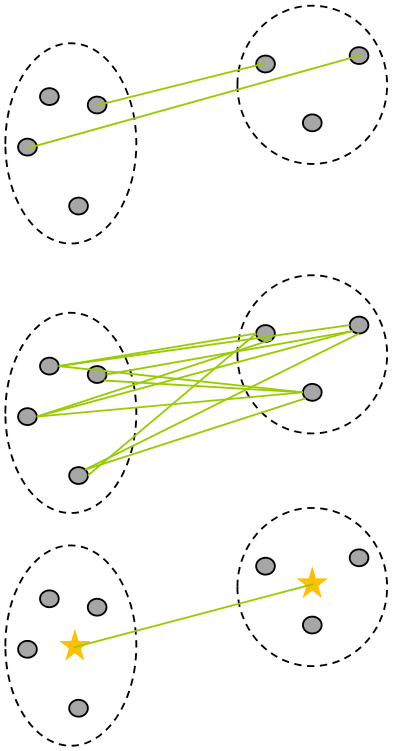
- ***mittlere Punktdistanz:*** $d_{\text{av}}(C, C') := \sum_{\mathbf{p} \in C, \mathbf{p}' \in C'} d(\mathbf{p}, \mathbf{p}'))/(|C| \cdot |C'|)$

  (manchmal werden auch die Quadrate der Distanzen genommen)

- ***maximale Punktdistanz:*** $d_{\text{max}}(C, C') := \text{max}_{\mathbf{p} \in C, \mathbf{p}' \in C'} d(\mathbf{p}, \mathbf{p}')$

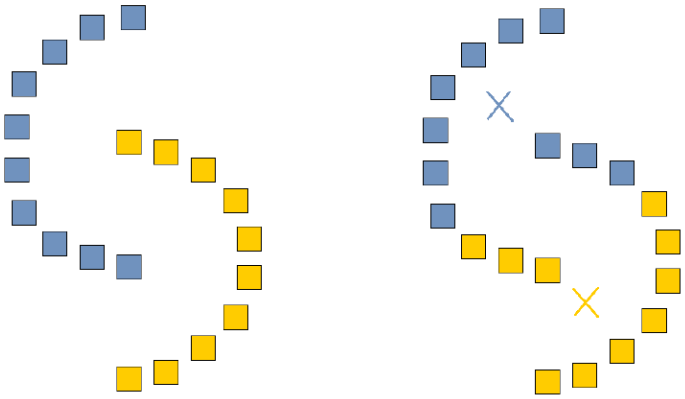
Abstandsmaße

- **Single Link:**
  Minimum aller Abstände
  zwischen zwei Objekten aus je einem Cluster
- **Complete Link:**
  Maximum aller Abstände
- **Average Link:**
  Durchschnittlicher Abstand
- **Centroid:**
  Abstand der Mittelpunkte



Problem:

k-Means Ansatz kann lediglich sphärische Cluster erkennen



Ground Truth k-Means Ergebnis

→ **Dichtebasierter Clusteringansatz**

Dichtebasiertes Clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- arbeitet dichtebasiert, ist in der Lage, mehrere Cluster zu erkennen. Rauschpunkte werden dabei ignoriert und separat zurückgeliefert
- in höherdimensionalen Räumen effizienter als vergleichbare raumzerlegende Verfahren, z.B. alpha-Shapes (s. Kapitel G. „Visualisierung räumlicher Daten"), liefert jedoch nur die Clustermengen und nicht deren Berandung.
- Cluster sind Regionen im d-dimensionalen Raum, deren Objekte dicht zusammenliegen
- Cluster können von beliebige Form sein

Jedes Objekt eines Clusters muss von einer gewissen Menge an anderen Objekten umgeben sein → **Dichte-Schwellenwert**.

LITERATUR:

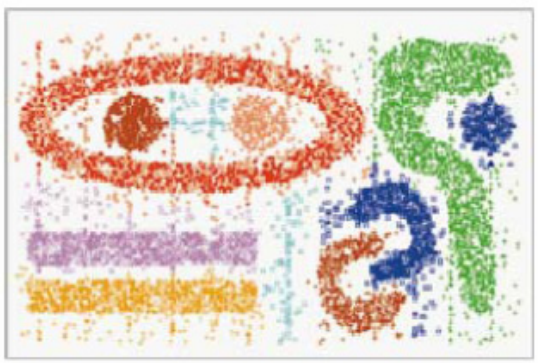
M. Ester, H.-P. Kriegel, J. Sander, X. Xu: A density-based algorithm for discovering Clusters in large spatial databases with noise. Proc. of the Second Int. Conf. on Knowledge Discovery and Data Mining (KDD-96). 1996, S. 226–231

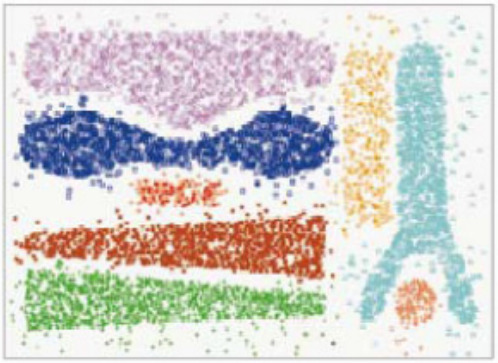
Parameter zur Steuerung des Dichte-Schwellenwerts

- Umgebungsradius ε
- MinPts = minimale #Punkte in ε -Umgebung

Punkte werden **Kernpunkte** genannt, wenn sie ≥MinPts Punkte in ihrer ε -Umgebung haben.

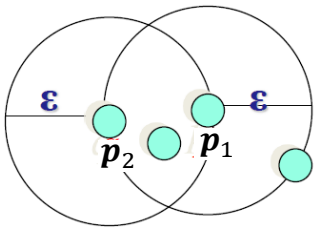
Cluster bestehen aus der Verkettung von Kernpunkten und den sich in ihren ε -Umgebungen befindlichen nicht Kernpunkten.

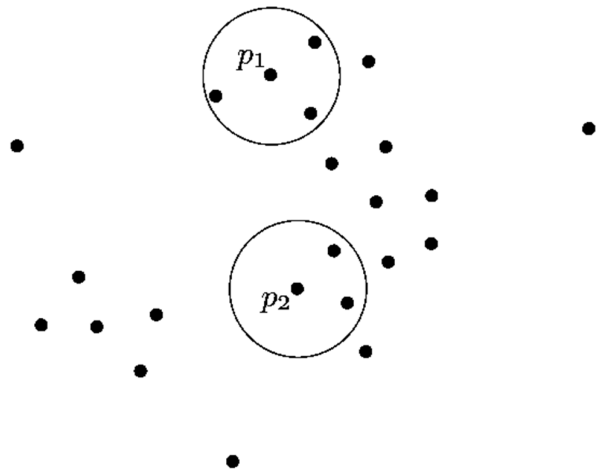




- ε **–Umgebungen** $N_\varepsilon(\boldsymbol{p})$ eines Punktes $\boldsymbol{p} \in P$: $N_\varepsilon(\boldsymbol{p}) = \{\boldsymbol{q} \in P : d(\boldsymbol{p}, \boldsymbol{q}) \leq \boldsymbol{\varepsilon}\}$
- $\boldsymbol{p} \in P$ ist ein **Kernpunkt** (hier $\boldsymbol{p}_1$), falls $|N_\varepsilon(\boldsymbol{p})| \geq MinPts$
- $\boldsymbol{p} \in P$ ist ein **Randpunkt** (hier $\boldsymbol{p}_2$), falls $|N_\varepsilon(\boldsymbol{p})| \leq MinPts$

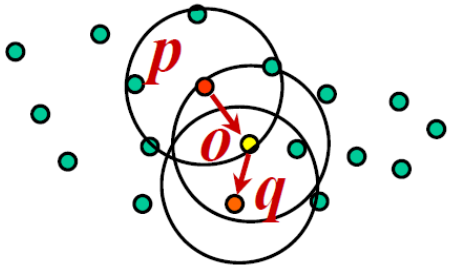
Beispiel: $MinPts = 4$





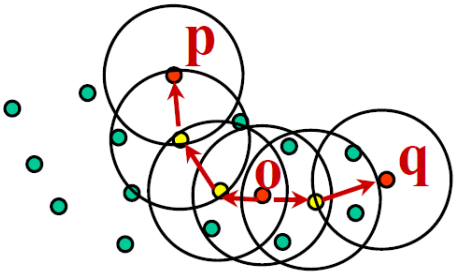
- Kernpunkt: $\boldsymbol{p}_1$
- Randpunkt: $\boldsymbol{p}_2$

- $\boldsymbol{p}$ ist ist **direkt Dichte-erreichbar** von einem Kern-Punkt $\boldsymbol{q}$, falls
  - $\boldsymbol{p} \in N_\varepsilon(\boldsymbol{q})$ und
  - $|N_\varepsilon(\boldsymbol{q})| \geq MinPts$

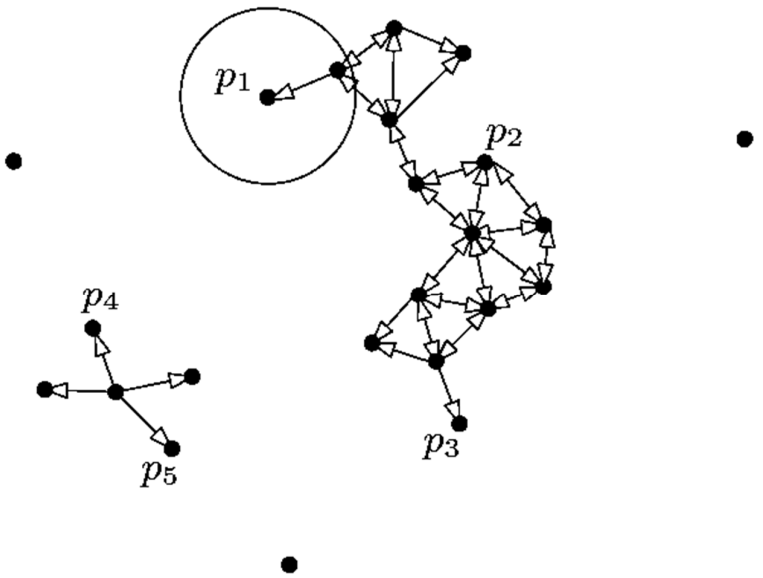


- $\boldsymbol{p}$ ist ist **Dichte-erreichbar** von einem Kern-Punkt $\boldsymbol{q}$, falls eine Punktesequenz $\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n$, $\boldsymbol{p}_1 = \boldsymbol{q}$, $\boldsymbol{p}_n = \boldsymbol{p}$ vorliegt, bei der $\boldsymbol{p}_1$ direkt Dichte-erreichbar $\boldsymbol{p}_n$ ist.

- $\boldsymbol{p}$ ist ist **Dichte-verbunden** von einem Kern-Punkt $\boldsymbol{q}$, falls ein Punkt $\boldsymbol{o}$ vorliegt, sodass $\boldsymbol{p}$ und $\boldsymbol{q}$ Dichte-erreichbar von $\boldsymbol{o}$ sinC.



Beispiel: $MinPts = 4$



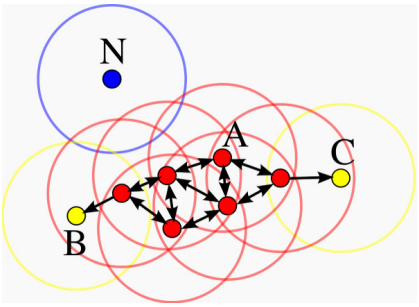
- $\boldsymbol{p}_1$ ist Dichte-erreichbar von $\boldsymbol{p}_2$, aber nicht von $\boldsymbol{p}_3$
- $\boldsymbol{p}_2$ ist nicht Dichte-erreichbar von $\boldsymbol{p}_1$und $\boldsymbol{p}_3$
- $\boldsymbol{p}_1$und $\boldsymbol{p}_3$ sind Dichte-verbunden, $\boldsymbol{p}_1$und $\boldsymbol{p}_4$ nicht

Cluster: Die durch dieselben Kernpunkte miteinander verbundenen Punkte

Sofern C eine Untermenge von P ist und die folgenden Bedingungen erfüllt sind, ist C ein Cluster:

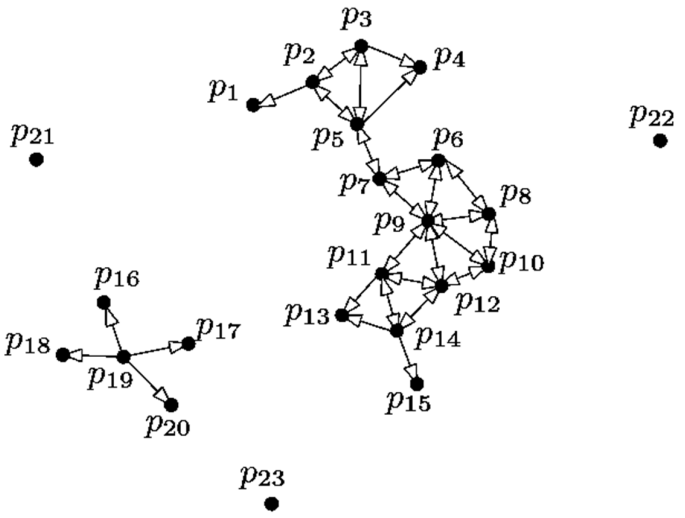
- **Maximalität**: Falls ein Punkt $\boldsymbol{p} \in C$ ein Kern-Punkt ist und $\boldsymbol{q}$ ist Dichte-erreichbar von $\boldsymbol{p}$, dann $\mathbf{q} \in C$
- **Konnektivität:** Alle Punktepaare $\boldsymbol{p}_i, \boldsymbol{p}_j \in C$ (eventuell $\boldsymbol{p}_i = \boldsymbol{p}_j$) muss Dichte-verbunden sein

Rauschpunkte: weder dicht, noch dichte-erreichbar.



- Punkte B und C sind Dichte-erreichbar von A und dadurch Dichte-verbunden und gehören zu dem gleichen Cluster.
- Punkt N ist weder ein Kernpunkt, noch Dichte-erreichbar, also Rauschen.

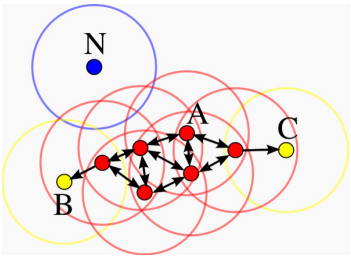
Beispiel: $MinPts = 4$



- $\{p_1, p_2, \cdots, p_{15}\}$ und $\{p_{16}, \cdots, p_{20}\}$ sind Cluster
- $\{p_1, p_2, \cdots, p_{20}\}$ ist kein Cluster
- Randpunkte: p_1, p_4, p_{13}, p_{15}, p_{16}, p_{17}, p_{18}, p_{20}
- Rauschpunkte: p_{21}, p_{22}, p_{23}

Algorithmus

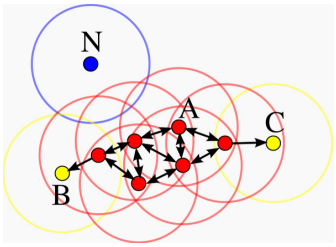
DBSCAN(D, eps, MinPts)
  C = 0
  for each unvisited point P in dataset D
    mark P as visited
    N = C.**regionQuery**(P, eps)
    if sizeof(N) < MinPts
      mark P as NOISE
    else
      C = next cluster
      expandCluster(P, N, C, eps, MinPts)



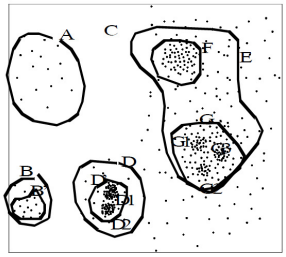
expandCluster(P, N, C, eps, MinPts)
  add P to cluster C
  for each point P' in N
    if P' is not visited
      mark P' as visited
      N' = C.**regionQuery**(P', eps)
      if sizeof(N') >= MinPts
        N = N joined with N'
    if P' is not yet member of any cluster
      add P' to cluster C

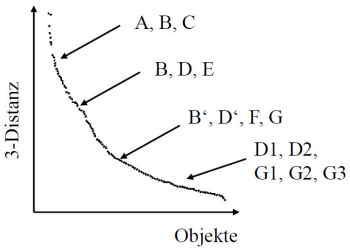
regionQuery(P, eps)
  return all points within P's eps-neighborhood (including P)

Bemerkungen:



- Dichte-erreichbare Punkte können von mehr als einem Cluster Dichte-erreichbar sein. Diese werden von dem Algorithmus nicht-deterministisch einem der möglichen Cluster zugeordnet: Dichteverbundenheit nicht transitiv; Dichteerreichbarkeit nicht symmetrisch.
- Zur Berechnung ist in $O(n)$ Rechenschritte möglich, n die Anzahl der Punkte. Der maximale Rechenaufwand pro Rechenschritt ist durch die Suche nach den ε -Nachbarn eines Punktes beschränkt.
- Zur Suche nach ε -Nachbarn können *k*-Distanz-Diagramme verwendet werden.





- Eine berandende Fläche der Cluster wird nicht berechnet.