Rückblick

CLIQUE:

- Datenraum wird in Zellen der Breite ξ zerlegt.
- Eine Zelle ist dicht, wenn sie mind. τ Punkte enthält.
- Zusammenhängende Zellen bilden Cluster
- Unterraumsuche:
  - bottom-up (ähnlich Apriori)
  - Monotoniekriterium für dichte Zellen:

    Wenn *k*-dimensionale Zelle *C* nicht dicht, dann alle (*k*+1)-dimensionalen Zellen, in denen *C* als „Unterzelle“ enthalten ist, nicht dicht

Nachfolgeverfahren: ENCLUS, MAFIA

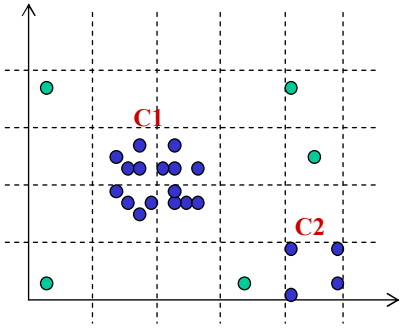
Dichte-verbundenes Subspace Clustering

Motivation:

- Nachteil der gitterbasierten Ansätze

  Wahl von ξ und τ

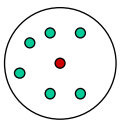
  Cluster für $\tau = 4$
  (ist C2 Cluster?)
  Für $\tau > 4$: keine Cluster
  (insb. C1 geht verloren!)



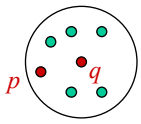
⇒ Verwende dichte-verbundenes Clustering (DBSCAN)

Subspace Clustering

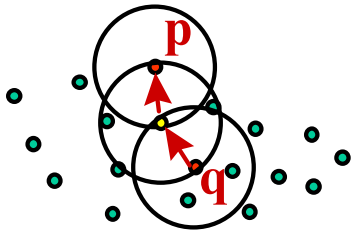
Rückblick: Dichte-verbundene Cluster (DBSCAN)



- Kernpunkt:
  Mehr als MinPts Punkte in der ε-Nachbarschaft

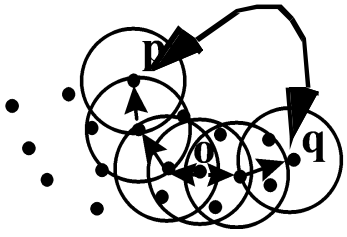


- Direkt dichte-erreichbar (*p* von *q*):
  q Kernpunkt und *p* in der ε-Nachbarschaft von *q*
- Dichte-erreichbar (*p* von *q*):
  Es gibt eine Kette direkt dichte-erreichbaren Punkte von *q* nach *p*



Subspace Cluster

- Dichte-verbunden (*p* und *q*):
  Es gibt Punkt *o*, sodass sowohl *p* als auch *q* dichte-erreichbar von *o*



- Dichte-verbundene Menge:
  Menge von (miteinander) dichte-verbundenen Punkten
- Dichte-verbundene Cluster:
  Dichte-verbundene Menge, die maximal ist bzgl. Dichte-Erreichbarkeit, d.h.
  $\forall p,q$: wenn $p \in C$ und q dichte-erreichbar von p ist, dann ist auch $q \in C$.

Subspace Clustering

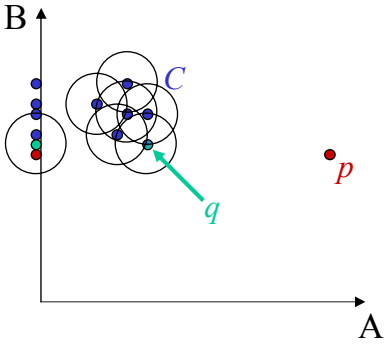
SUBCLU (Dichte-verbundenes Subspace Clustering) [Kailing, Kriegel, Kröger 2004]

- Berechne dichte-verbundene Subspace Cluster
- Vorteile:
  - Clusterbegriff mathematisch sauber formuliert
  - Zuordnung der (Kern-) Punkte zum Cluster eindeutig
  - Erkennen von Clustern unterschiedlicher Größe und Form
- Gesucht:
  - Effiziente Strategie, um die dichte-verbundenen Cluster in allen Unterräumen (bzgl. ε und *MinPts*) zu berechnen
  - Nutze Greedy-Ansatz wie bei CLIQUE: generiere bottom-up alle Subspace Cluster
  - Dazu notwendig: Monotoniekriterium für dichte-verbundene Cluster

Subspace Clustering

Monotonie dichte-verbundener Cluster

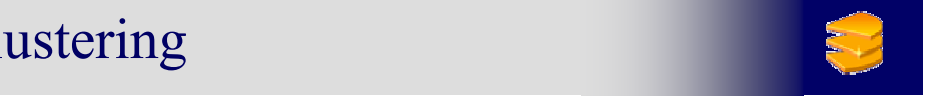
- Gilt leider nicht:
  - Sei C ein dichte-verbundener Cluster im Unterraum S
  - Sei $T \subset S$ ein Unterraum von S
  - C muss nicht mehr maximal bzgl. Dichte-Erreichbarkeit sein
  - Es kann Punkte geben, die nicht in C sind, aber im Unterraum T dichte-erreichbar von einem Objekt in C sind



C ist ein dichte-verbundener Cluster im Unterraum {A,B}

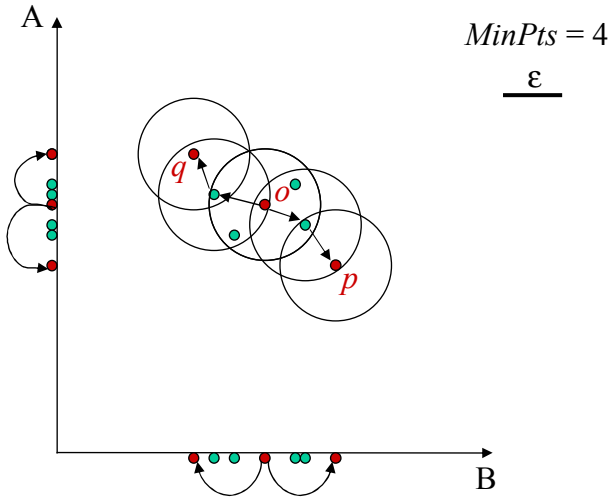
$p \notin C$ und $q \in C$

Im Unterraum {B} ist p (direkt) dichte-erreichbar von $q \in C$

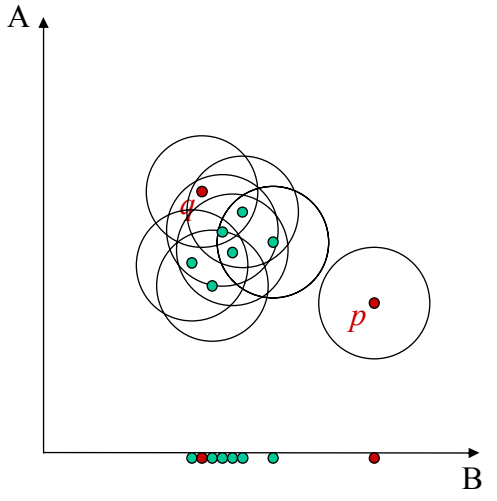


Monotonie dichte-verbundener Mengen

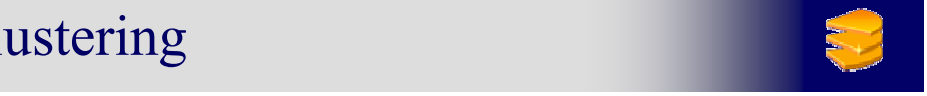
Wenn C eine dichte-verbundene Menge im Unterraum S ist, so ist C auch eine dichte-verbunden Menge in allen Teilräumen $T \subset S$



p und q dichte-verbunden in {A,B}, {A} und {B}



p und q nicht dichte-verbunden in {B} und {A,B}



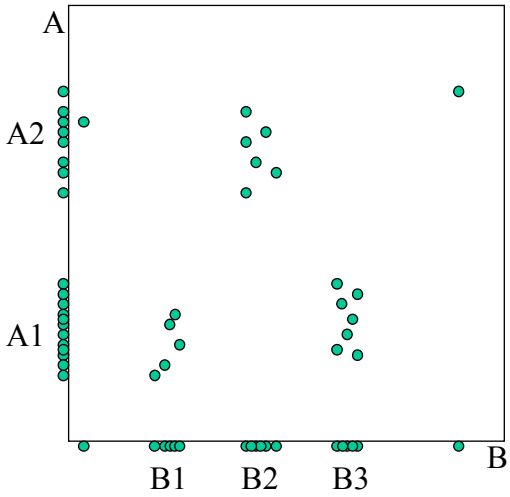
Algorithmus

- Generiere alle 1-dimensionalen dichte-verbundenen Cluster
- Für jeden k-dimensionalen Cluster muss nun geprüft werden, ob er in einem (k+1)-dimensionalen Oberraum noch vorhanden ist:
  - Gegeben:
    - Sk: Menge der k-dimensionale Unterräume in denen Cluster existieren
    - CS: Menge der Cluster im Unterraum S
    - Ck: Menge aller Mengen von Cluster in k-dimensionalen Unterräumen

      Ck = {CS | S ist k-dimensionaler Unterraum}
  - Vorgehen:
    - Bestimme (k+1)-dimensionale Kandidatenunterräume *Cand* aus Sk
    - Für einen beliebigen k-dimensionalen Unterraum $U \subset Cand$:

      Bestimme für alle k-dimensionalen Cluster c in U ($c \in CU$) die (k+1)-dimensionalen Fortsetzungen durch die Funktion DBSCAN(c, U, ε, *MinPts*)

Subspace Clustering



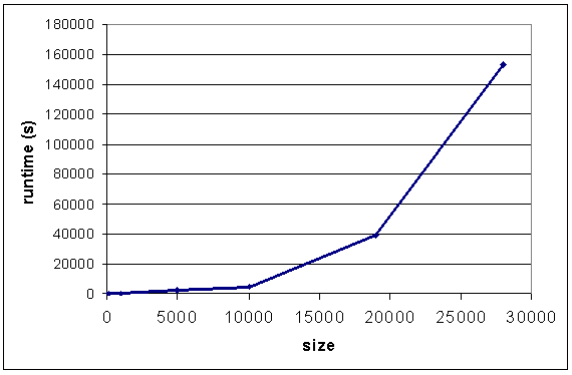
Funktion DBSCAN(*D*, *U*, ε, *MinPts*) berechnet alle dichte-verbundenen Cluster bzgl. ε und *MinPts* einer Datenmenge *D* im Unterraum *U*

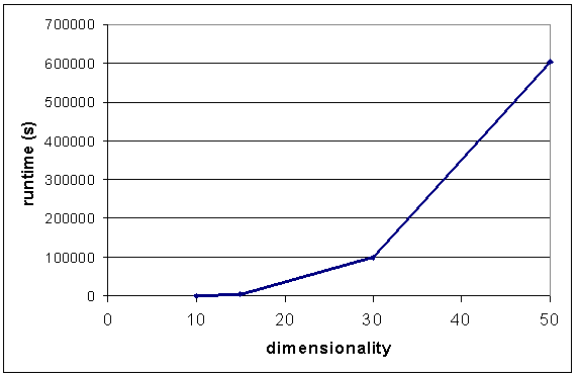
S1 = {{A}, {B}}
C{A} = {A1, A2}
C{B} = {B1, B2, B3}
C1 = {C{A}, C{B}}

- Heuristische Optimierungsmöglichkeit:
  - DBSCAN(*c*, *U*, ε, *MinPts*) nicht für zufälligen $U \subset Cand$ aufrufen, sondern für den Unterraum *U*, in dem die Gesamtanzahl der Punkte in den Clustern (also der Punkte in C*U*) am geringsten ist (im Beispiel: *U* = {B})
  - Dadurch wird die Anzahl der Range-Queries beim DBSCAN-Lauf minimiert (im Beispiel um 2)

Subspace Clustering

Experimente





Skalierbarkeit: superlinear in Anzahl der Dimensionen und Anzahl der Objekte

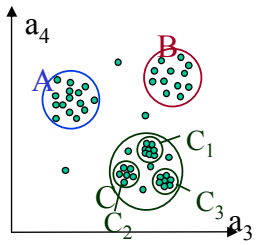
ABER: Findet mehr Cluster als CLIQUE 

Subspace Clustering

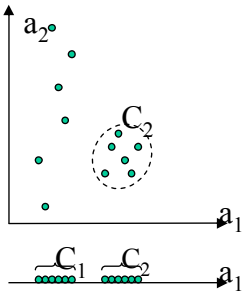
RIS (Ranking Interesting Subspaces) [Kailing, Kriegel, Kröger, Wanka 2003]

Probleme von SUBCLU:

- Verschiedene Cluster in einem Unterraum können verschieden dicht sein



- Cluster aus verschiedenen Unterräumen können verschieden dicht sein



Subspace Clustering

Idee von RIS:

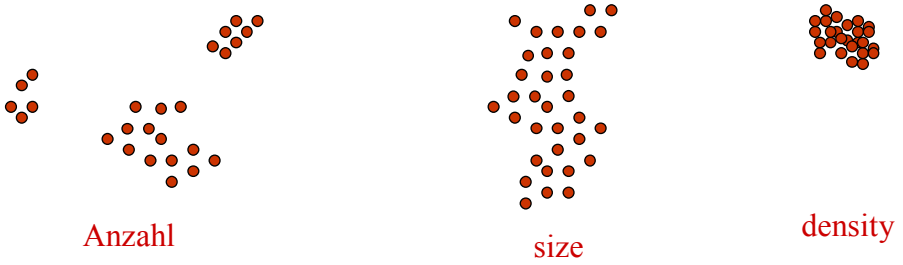
- Berechne nicht mehr direkt die Subspace Cluster
- Sondern: berechne nur die Unterräume, die interessante Cluster enthalten
  - Was sind interessante Cluster/Unterräume?
  - Qualitätskriterium für Unterräume
- RIS gibt eine Liste von Unterräumen aus, sortiert nach Qualität
- Die eigentlichen Cluster können durch ein beliebiges Cluster-Verfahren für die interessanten Unterräume erzeugt werden

Subspace Clustering

Interessante Unterräume:

- Cluster enthalten mindestens einen Kernpunkt

  ⇒ Unterraum, der keinen Kernpunkt enthält, kann nicht interessant sein
- Anzahl der Kernpunkte ist proportional zur
  - Anzahl der verschiedenen Cluster und/oder
  - Größe der Cluster und/oder
  - Dichte der Cluster



Subspace Clustering

Algorithmus RIS:

1. Berechne für jeden Punkt *p* der Datenbank die Unterräume, in denen *p* noch Kernpunkt ist

   ⇒ Berechnet alle relevanten Unterräume

2. Sammle für jeden berechneten Unterraum statistische Informationen um über die „Interessantheit“ des Unterraumes entscheiden zu können

   ⇒ Qualität der Unterräume (z.B. Anzahl der Kernpunkte)

   ⇒ Sortierung der Unterräume nach „Interessantheit“ möglich

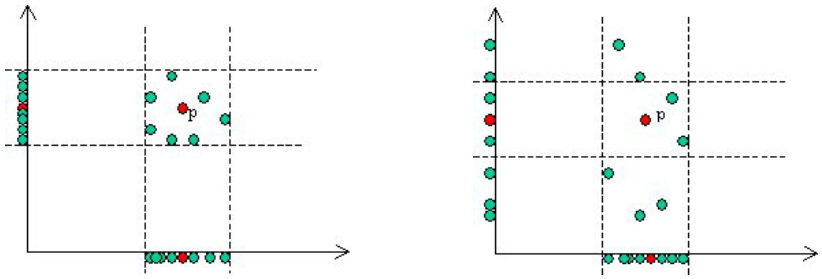
3. Entferne Unterräume, die redundante Informationen enthalten

   ⇒ Cluster in einem Unterraum S sind in allen Unterräumen T $\subseteq$ S enthalten

Schritt 1

Suche Unterräume, die mindestens einen Kernpunkt enthalten:

- Monotonie der Kernpunkteigenschaft:
  Wenn p ein Kernpunkt in Featureraum S ist, dann ist p auch ein Kernpunkt in allen Unterräumen $T \subseteq S$



Wenn p in T kein Kernpunkt ist, kann p auch in allen $S \supset T$ kein Kernpunkt sein.
$\Rightarrow$ Suchstrategie von CLIQUE und SUBCLU wieder verwendbar

Schritt 2

Qualität der gefundenen Unterräume:

- count[S] = Summe (der Anzahl) aller Punkte, die in der ε-Nachbarschaft aller Kernpunkte eines Unterraumes S liegen
- NaiveQuality(S) = count[S] – Kernpunkte(S)
  - Anzahl der erwarteten Punkte in einer ε-Nachbarschaft sinkt mit steigender Dimension
  - NaiveQuality favorisiert niedrig dimensionale Unterräume
- Skalierung in Abhängigkeit der Dimensionalität:

$$\text{Quality}(S) = \frac{\text{count}[S] - \text{Kernpunkte}(S)}{n(n-1)(\frac{2\epsilon}{\text{Attr.bereich}})^{\dim(S)}}$$

  - Periodische Randbedingungen um Punkte, die am Rand des Datenraumes liegen, nicht zu benachteilen

Schritt 3

Entfernen redundanter Unterräume:

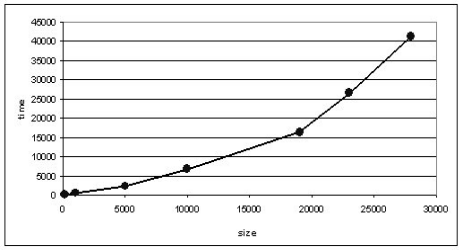
- „Überflüssige“ Unterräume:
  - Cluster im Raum S haben eine Projektion in Unterräumen von S
  - Durch die Hinzunahme von irrelevanten Dimensionen muss ein Cluster zunächst noch nicht verschwinden
- Pruning-Schritte:
  - Abwärts-Pruning:

    Wenn es einen (k-1)-dimensionalen Unterraum S mit einer höheren Qualität als ein k-dimensionaler Unterraum T ($T \subset S$) gibt, lösche T.
  - Aufwärts-Pruning:

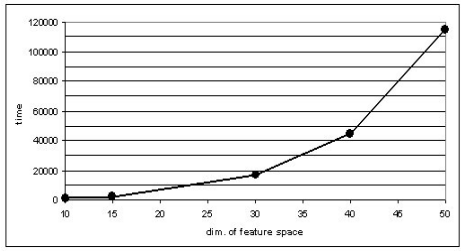
    Wenn der Count-Wert eines echten (k-1)-dimensionaler Unterraumes von S „besonders stark“ vom Mittelwert der Count-Werte aller echten (k-1)-dimensionalen Unterräume von S abweicht, lösche S

Experimentelle Untersuchung

Laufzeit in Abhängigkeit von n



Laufzeit in Abhängigkeit von d

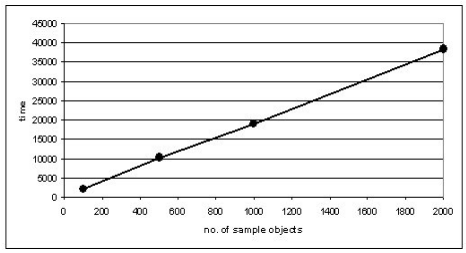


Skaliert superlinear in n und d

$\Rightarrow$ Random Sampling

auch bei kleinen Samplegrößen hohe Qualität

Laufzeit in Abhängigkeit der Samplegröße



Diskussion

Vorteile:

- Findet alle Unterräume, in denen interessante Cluster vorhanden sind
- Erzeugen von Subspace Clustern unterschiedlicher Dichte möglich (z.B. indem man in den gefundenen Unterräumen mit OPTICS „clustert“)

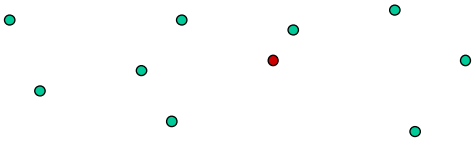
Nachteile:

- Problem, das Cluster in verschieden dimensionalen Unterräumen meist unterschiedlich dicht sind, ist immer noch nicht gelöst
- Trotz Dimensions-Anpassung des Qualitätskriteriums:
  ε begrenzt die Dimension der gefunden Unterräume nach oben:
  je kleiner ε desto niedriger dimensional die Unterräume, die gefunden werden

SURFING *(Subspaces Relevant for Clustering)* [Kailing, Kriegel, Kröger (subm.)]

- Idee: Berechne interessante Unterräume
  - Unabhängigkeit von einem globalen Dichteparameter für verschiedene Cluster und verschiedene Unterräume
  - ohne die dichte-basierte Vorstellung von Clustern komplett aufzugeben
  - OPTICS:
    - Unabhängig von einem globalen Dichteparameter
    - Dichte-basiertes Cluster-Modell
    - Kerndistanz (Distanz zum *k*-nächsten Nachbarn) und Erreichbarkeitsdistanz als Maß für lokale Dichte
    - Je kleiner Kerndistanz, desto dichter sind die Punkte lokal
    - Je größer Kerndistanz, desto weniger dicht sind die Punkte lokal

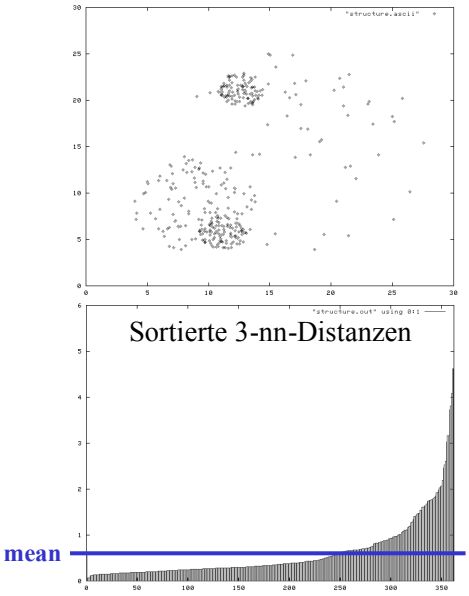
Kleine 10-nächste Nachbarn Distanz

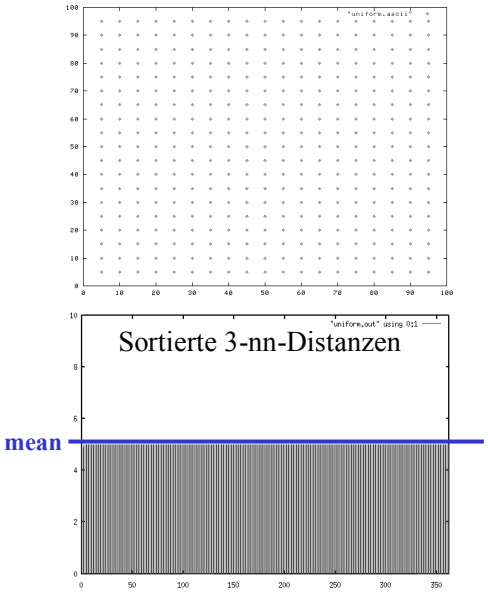


Grosse 10-nächste Nachbarn Distanz

Subspace Clustering

- Die Qualität der hierarchischen Clusterstruktur eines Unterraumes kann anhand der *k*-nn-Distanzen aller Punkte vorhergesagt werden:
  - Viele unterschiedliche *k*-nn-Distanzen ⇒ signifikante (hierarchische) Clusterstrukturen
  - Viele ähnliche *k*-nn-Distanzen ⇒ kaum (hierarchische) Clusterstrukturen





Subspace Clustering

Qualititätskriterium für Unterräume

- Varianz der *k*-nn-Distanzen in einem Unterraum:
  - Nachteil: berücksichtigt die quadrierten Differenzen zum Mittelwert
- Summe der Differenzen *DIFF* unterhalb des Mittelwertes:
  - Nachteil: nicht unabhängig von der Dimension
- Verhältnis aus *DIFF* zum Mittelwert μ:
  - Nachteil: Mittelwert ist nicht vollständig robust gegenüber Ausreißern und kleinen sehr dichten Clustern
    - Mittelwert wird durch einige wenige Ausreißer nach oben verschoben
      ⇒ *DIFF* unverhältnismäßig hoch ⇒ *DIFF*/μ unverhältnismäßig zu hoch
    - Mittelwert wird durch wenige kleine sehr dichte Cluster nach unten verschoben
      ⇒ *DIFF* unverhältnismäßig klein ⇒ *DIFF*/μ unverhältnismäßig zu klein

⇒ Skalierung mit der relativen Anzahl der Punkte, deren k-nn-Distanz unterhalb des Mittelwertes liegt (bezeichnet als *Below*)

Subspace Cluster

Qualität eines Unterraums:

$$\text{Quality} = \frac{\frac{DIFF}{\mu}}{\frac{Below}{N}} = \frac{DIFF \cdot N}{Below \cdot \mu}$$

DIFF = Summe der Differenzen der k-nn-Distanzen unterhalb von μ zum Mittelwert

μ = Mittelwert der *k*-nn-Distanzen

Below = Anzahl der Punkte, die eine *k*-nn-Distanz unterhalb von μ haben

N = Anzahl der Datensätze

Subspace Clustering

Algorithmus SURFING

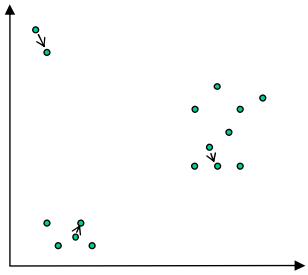
- Qualitätskriterium ist nicht monoton!!!
- ABER: Qualität steigt, wenn relevante Attribute hinzu kommen bzw. sinkt, wenn irrelevante Attribute hinzukommen
- Bottom-up Unterraum Generierung ähnlich wie *Apriori*, aber kein Pruning bei der Kandidatengenerierung

  $\Rightarrow$ mehr Kandidaten in jeder Iteration zu Testen
- Heuristisches Pruningkriterium um möglichst viele Unterräume zu löschen (dadurch wird Anzahl der Kandidaten reduziert)
- Komplexität: $O(N^2 \cdot m)$ m = # generierter Unterräume

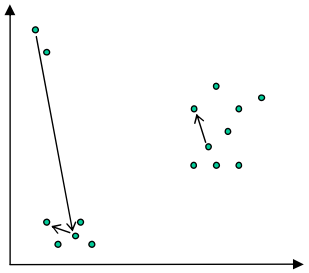
Subspace Clustering

Parameterwahl

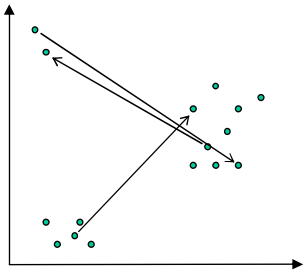
- SURFING hängt nur noch von *k* ab!!!
- Wahl von *k* relativ einfach:



k = 1 (zu klein)



k = 5 (richtig)



k = 10 (zu groß)

Subspace Clustering

Fazit

- SURFING ist dank der Pruning-Heuristik sehr effizient (meist werden nur knapp 1% aller möglichen Unterräume erzeugt)
- SURFING ist mehr oder weniger parameterfrei (Wahl von k relativ einfach und bei großen, hochdimensionalen Daten typischerweise nicht kritisch)
- SURFING erzielt (in Zusammenarbeit mit einem hierarchischen Clustering-Algorithmus) bessere experimentelle Ergebnisse als CLIQUE, SUBCLU oder RIS, speziell wenn:
  - Cluster in stark verschieden dimensionalen Unterräumen existieren
  - Hierarchische und unterschiedlich dichte Cluster existieren

Subspace Clustering

Zusammenfassung

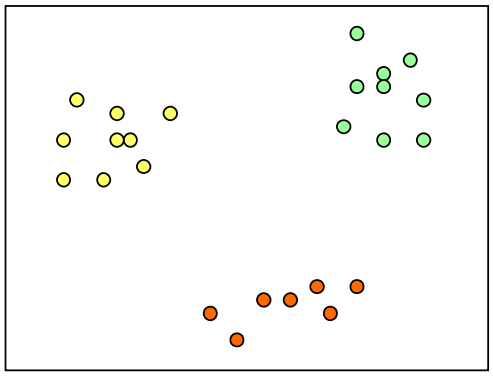
- CLIQUE, ENCLUS, MAFIA
  - Grid-basiertes Clustermodell
  - Direkte Berechnung der Cluster
- SUBCLU
  - Dichte-verbundenes Clustermodell
  - Direkte Berechnung der Cluster
- RIS
  - Dichte-verbundenes Clustermodell
  - Ranking der Unterräume anhand ihrer Qualität (flaches Clustering)
- SURFING
  - Dichte-verbundenes Clustermodell
  - Ranking der Unterräume anhand ihrer Qualität (hierarchisches Clustering)

Globaler Dichteparameter

Lokal adaptiver Dichteparameter

5.5 Correlation Clustering

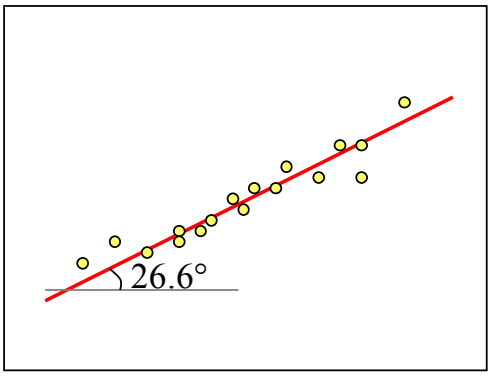
Clustering...



Einteilung der Punktemenge in Gruppen (Cluster), so dass...

- Maximale Ähnlichkeit der Punkte innerhalb der Cluster
- Minimale Ähnlichkeit der Punkte versch. Cluster

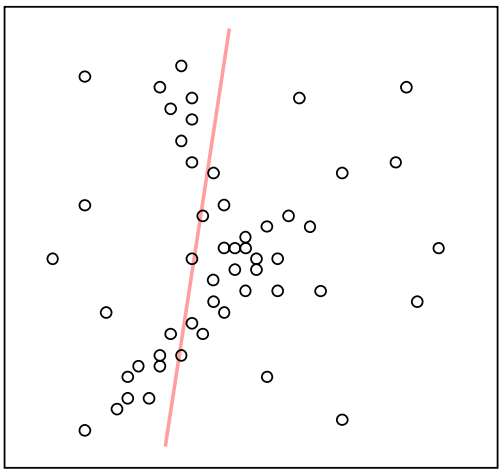
Korrelation...



$y \approx 0.5\, x + \ldots$

(lineare) Abhängigkeit zwischen den einzelnen Attributen (Dimensionen) einer Punktemenge

Probleme der Korrelation

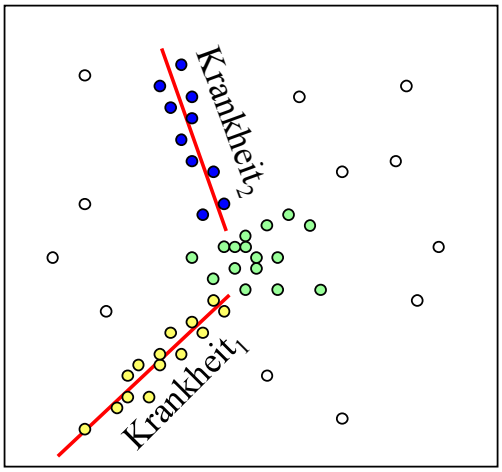


Rausch-Punkte

Verschiedene Teilmengen weisen unterschiedliche Korrelationen auf

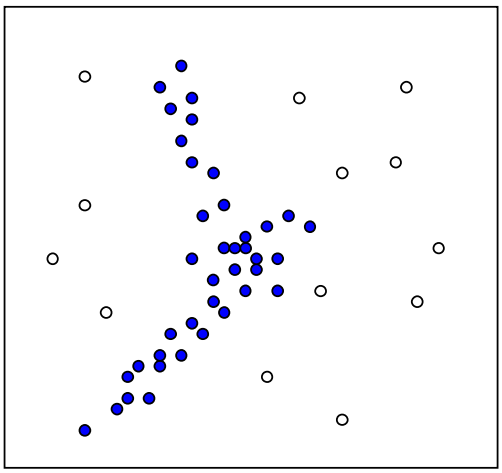
→ schwache Gesamt-Korrelation

Probleme der Korrelation



Ziel:
Suche nach Teilmengen von Punkten mit einheitlicher Korrelation

Dichtebasiertes Clustering



Trennt grundsätzlich auch
Correlation Cluster
von Rauschpunkten

Separiert aber nicht
nach unterschiedlicher
Regressionslinie

Informelle Definition

Ein Korrelations-verbundener Cluster ist eine Punktmenge mit...

- einheitlicher Punktdichte (bzw. Dichte-Schwellwert)
- einheitlicher Korrelation (Regressionslinie)

d.h. ein Correlation-Clustering-Verfahren soll

- Punktdichte und
- Korrelation

innerhalb von Clustern maximieren

zwischen separierten Clustern minimieren

Idee für Correlation Clustering Algorithmus

Erweiterung von dichtebasiertem Clustering

- DBSCAN
- OPTICS
- Oder eines anderen Verfahrens

ggf. unter Einführung neuer Parameter (Dimension der Korrelation)

Möglichkeiten, Korrelation ins Spiel zu bringen

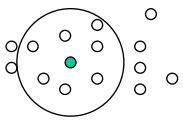
- Adaptives Ähnlichkeitsmaß
- Fraktale Dimension
- Hough-Transformation

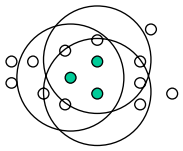
Adaptives Ähnlichkeitsmaß

[Böhm, Kailing, Kröger, Zimek: *Computing Clusters of Correlation Connected Objects*, subm.]

DBSCAN beruht im wesentlichen auf zwei Konzepten:

- Kernpunkte:
  Punkte, in deren ε-Umgebung sich mindestens *MinPts* Punkte befinden
- Dichte-Verbundenheit:
  Kernpunkte werden mit Nachbarn in der ε-Umgebung vereinigt





Idee von 4C (Computing Correlation Connected Clusters):

- Anpassung dieser Konzepte von DBSCAN, so dass nach korrelierten Punktmengen gesucht wird
- Dimension λ der Korrelation durch den Benutzer vorgegeben:
  - λ = 1 für Korrelations-Linien
  - λ = 2 für Korrelations-Ebenen usw.

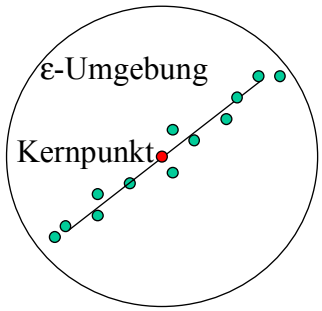
Kernpunkte bei 4C

Zusätzlich zur Forderung, dass sich in der ε-Umgebung mindestens *MinPts* Nachbarn befinden müssen:

Die Punkte in der ε-Umgebung eines Kernpunktes müssen sich

- ...auf (bzw. in der Nähe) einer gemeinsamen Linie (im Fall $\lambda=1$),
- ...einer gemeinsamen Ebene (im Fall $\lambda=2$),
- ...einer gemeinsamen λ-dimensionalen Hyperebene (im Fall $\lambda>2$)

durch den Kernpunkt befinden.



Dies lässt sich mathematsch wie folgt bestimmen:

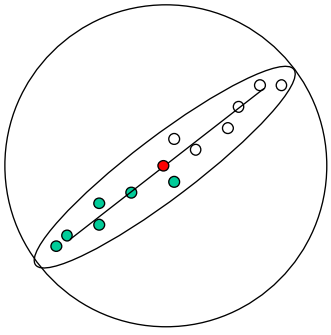
- Berechnung Kovarianzmatrix Σ der Nachbarn
- Eigenwert-Zerlegung (Principal Components) $V \cdot E \cdot V^T = \Sigma$
- Mindestens d–λ Eigenwerte müssen ≈ 0 sein

Hierdurch wird jedem Kernpunkt eine Kovarianzmatrix zugeordnet

Dichte- (bzw. Korrelations-) Verbundenheit

Prinzip:

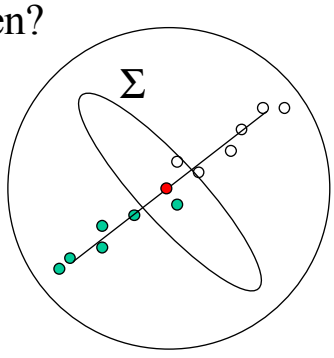
- Nur solche Punkte sollen mit einem Cluster vereinigt werden, die auch in der bisherigen Ausdehungsrichtung (d.h. nahe zur Korrelationslinie, -Ebene usw.) liegen



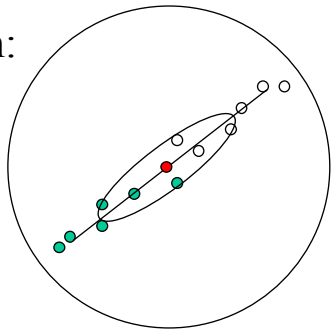
Wie kann dieses Ähnlichkeitsmaß erreicht werden?

- Ähnlichkeitsmaß entspricht Kovarianzmatrix: $\text{dist}^2(P,Q) = (P-Q) \cdot \Sigma \cdot (P-Q)^T$

  Richtungen starker Varianz werden durch das Ähnlichkeitsmaß stark gewichtet.
  $\Rightarrow$ Ansatz genau kontraproduktiv!



Dichte- (bzw. Korrelations-) Verbundenheit



- Kovarianzmatrix mit invertierten Eigenwerten:
  $\text{dist}^2(P,Q) = (P-Q) \cdot V \cdot E^{-1} \cdot V^T \cdot (P-Q)^T$

  Anmerkung: Diagonalmatrizen werden elementweise invertiert:
  $\text{diag}(a_1,a_2,...)^{-1} = \text{diag}(1/a_1, 1/a_2, ...)$

  Ausrichtung des Ellipsoids nun korrekt!

  Probleme:
  - Was macht man mit Eigenwerten =0 (also *keine* Varianz in dieser Richtung)?
  - Ausdehnung des Ellipsoids in allen Richtungen verschieden und nicht klar definiert

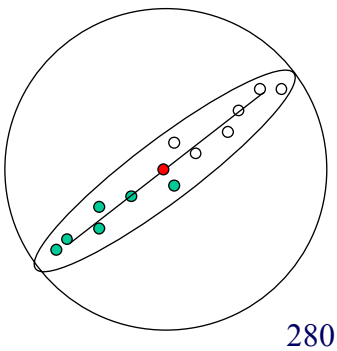
Dichte- (bzw. Korrelations-) Verbundenheit

Gewünscht: Ellipsoid mit folgenden Eigenschaften

- Ausrichtung gemäß den stärksten Eigenvektoren
- Ausdehnung ε in λ Richtungen
- Eine einheitliche, wesentlich geringere Ausdehung, die eine gewisse Toleranz erlaubt, in den verbleibenden d–λ Richtungen

Die Eigenwertmatrix wird wie folgt modifiziert:

- Die ersten λ Eigenwerte werden auf 1 gesetzt (Ellipsoid ist definiert als $\{x \mid \text{dist}(P,x) \leq \varepsilon\}$)
- Die verbleibenden d–λ Eigenwerte auf κ ($\kappa >> 1$ Benutzer-definierter Wert)
- Distanzmaß mit modifiziertem E′:
  $\text{dist}^2(P,Q) = (P-Q) \cdot V \cdot E' \cdot V^T \cdot (P-Q)^T$



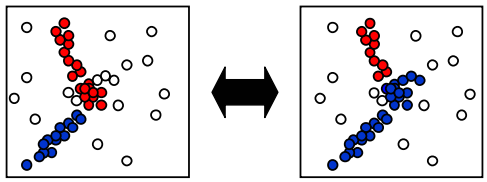
Unsymmetrische Metrik

Beobachtung:

Das Abstandsmaß ist nicht symmetrisch, da immer die modifiz. Kovarianzmatrix, die einem der beiden beteiligten Kernpunkte zugeordnet ist, das Abstandsmaß definiert.

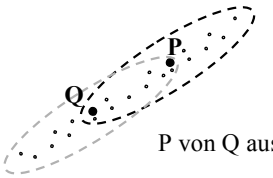
Problem:

Hierdurch wird das Clusterverfahren Reihenfolge-abhängig

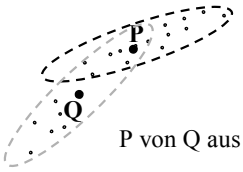


Lösung:

Vereinige Punkte nur dann, wenn sie sich „gegenseitig“ finden, also: $dist_P(P,Q) \leq \varepsilon$ **und** $dist_Q(Q,P) \leq \varepsilon$



P von Q aus direkt dichte-erreichbar



P von Q aus nicht direkt dichte-erreichbar

Algorithmus 4C (ε, *MinPts*, λ)

Für alle Objekte *o* aus der Datenbank:

Schritt 1: Test auf Korrelations-Kernobjekt

berechne ε-Umgebung $N_\varepsilon(o)$ von *o*;

Wenn $|N_\varepsilon(o)| \geq MinPts$

berechne Σ;

Wenn (*d*-λ) Eigenwerte ≈0

berechne E´;

berechne ε-Umgebung $N'_\varepsilon(o)$ von *o* bzgl. E´;

teste $|N'_\varepsilon(o)| \geq MinPts$;

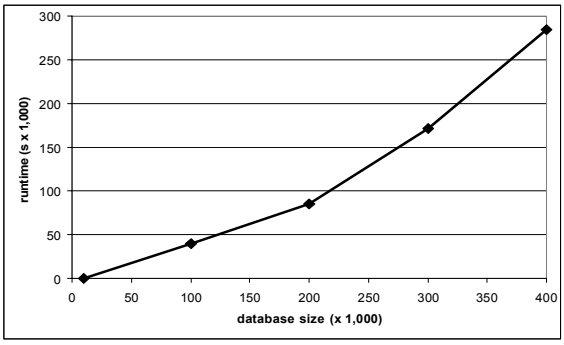
Schritt 2: Expandiere Cluster

berechne alle Punkte, die korrelations-dichte-erreichbar von *o* sind;

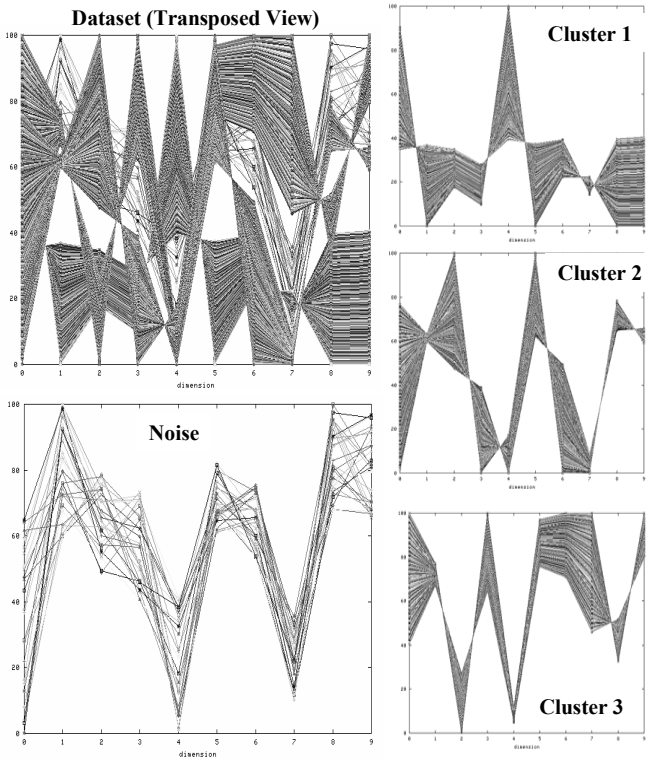
- ähnlich wie DBSCAN
- benutze dabei E´ als Distanzmaß
- achte auf Symmetrie

Ergebnisse: Accuracy

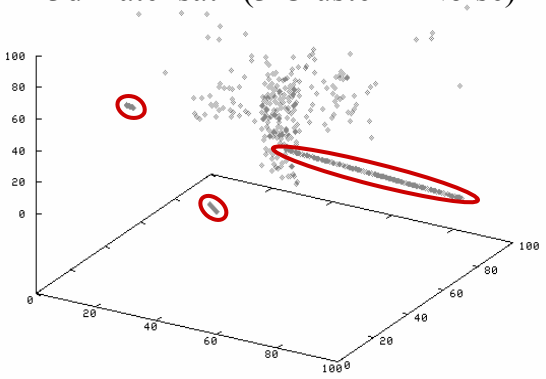
Laufzeit



10d Datensatz (3 Cluster + Noise)

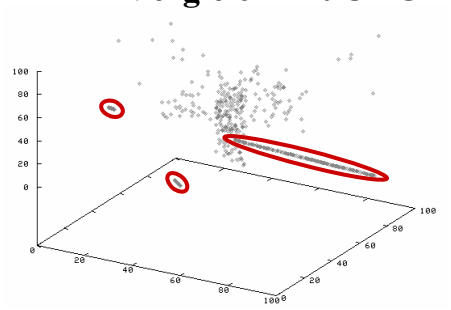


3d Datensatz (3 Cluster + Noise)

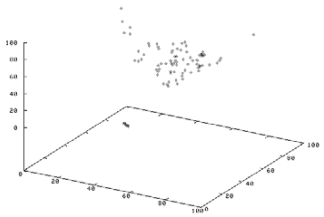


Ergebnisse: Accuracy

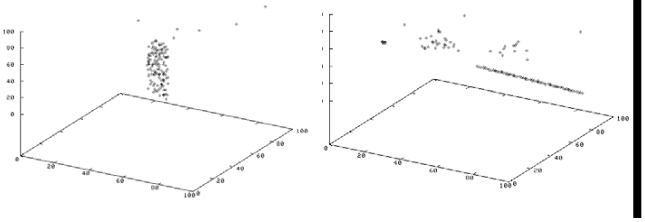
Vergleich mit ORCLUS



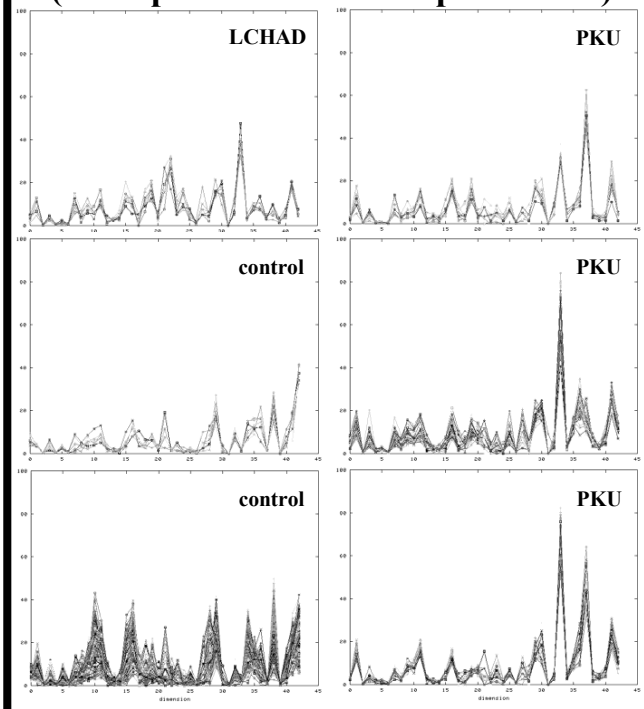
4C



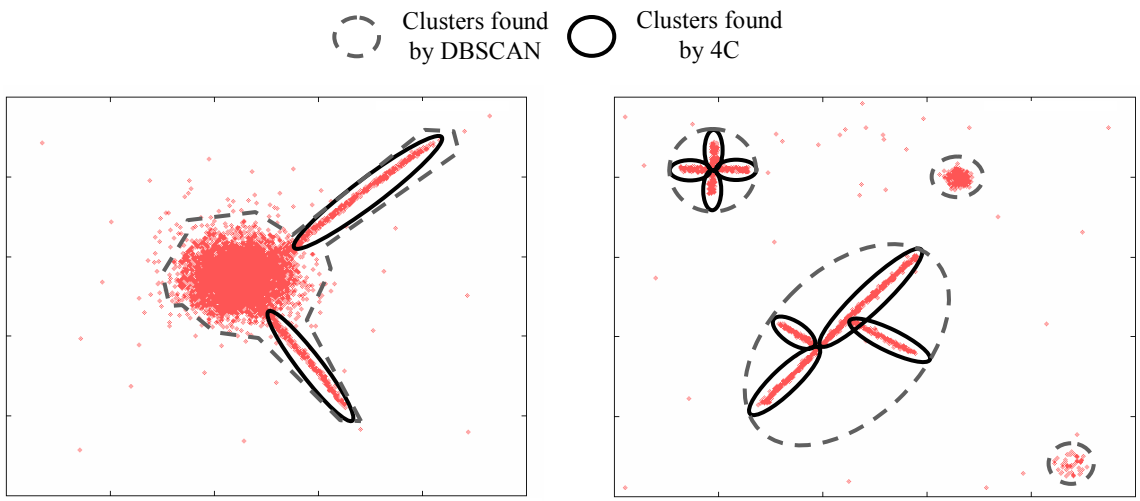
ORCLUS (k=3)



43d Metabolome Data
(6 sample cluster in transposed view)



Ergebnisse: Vergleich mit DBSCAN



Performanz

Komplexität ohne Indexunterstrützung:

- Für jeden (Kern-) Punkt ist das zugeordnete Ähnlichkeitsmaß (die modifizierte Kovarianzmatrix) zu ermitteln:
  - Ermittlung der Kovarianzmatrix: $O(nd^2)$
  - Eigenwert-Zerlegung der Kovarianzmatrix: $O(d^3)$
- DBSCAN wertet je eine Bereichsanfrage pro Punkt aus:
  - Auswertung mit modifizierter Kovarianzmatrix: $O(nd^2)$
- Gesamtkomplexität: $O(n^2d^2+d^3n)$

Komplexität mit Indexunterstützung:

- Bereichsanfrage reduziert sich auf $O(d^2 \log n)$
- Gesamt-Komplexität: $O(d^2n \log n + d^3n)$

Diskussion

Stärken

- Erstes Verfahren, das Teilmengen in einer Menge von Merkmalsvektoren ermittelt, die einheitliche Korrelation aufweisen (mit Ausnahme von ORCLUS, dessen „orientierte Cluster“ ähnlich funktionieren)
- Wesentlich bessere Ergebnisse als ORCLUS (*k*-Means)

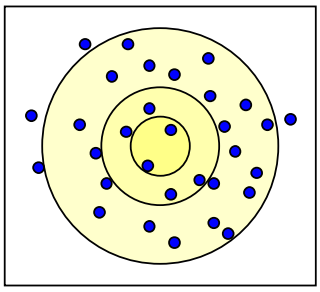
Schwächen

- Mengen müssen zusätzlich zur Korrelation auch Dichte-verbunden sein (Parameter ε)
- (zurzeit noch) nicht hierarchisch → 4C-OPTICS?
- Dimensionalität λ der Korrelation muss vorgegeben werden
- Findet nur lineare Abhängigkeiten
- Punkte können nur einem Cluster zugeordnet sein

Alternativ-Ansatz: Fraktale Dimension

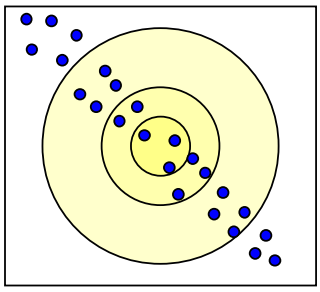
Bei Korrelationen ergibt sich charakteristische Abhängigkeit zwischen Volumen und Anzahl eingeschlossener Punkte:

Ohne Korrelation:



$N \sim r^2$

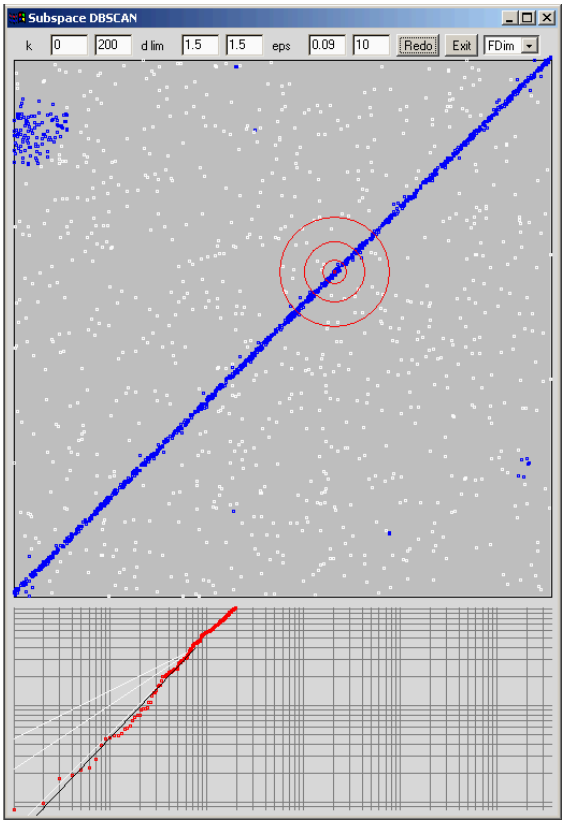
Mit Korrelation:



$N \sim r^1$

Dieser Effekt ist unabhängig davon, ob die Abhängigkeit linear oder nicht-linear ist.

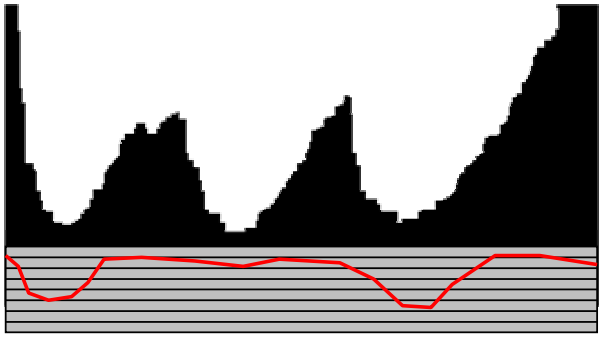
Ermittlung der fraktalen Dimension



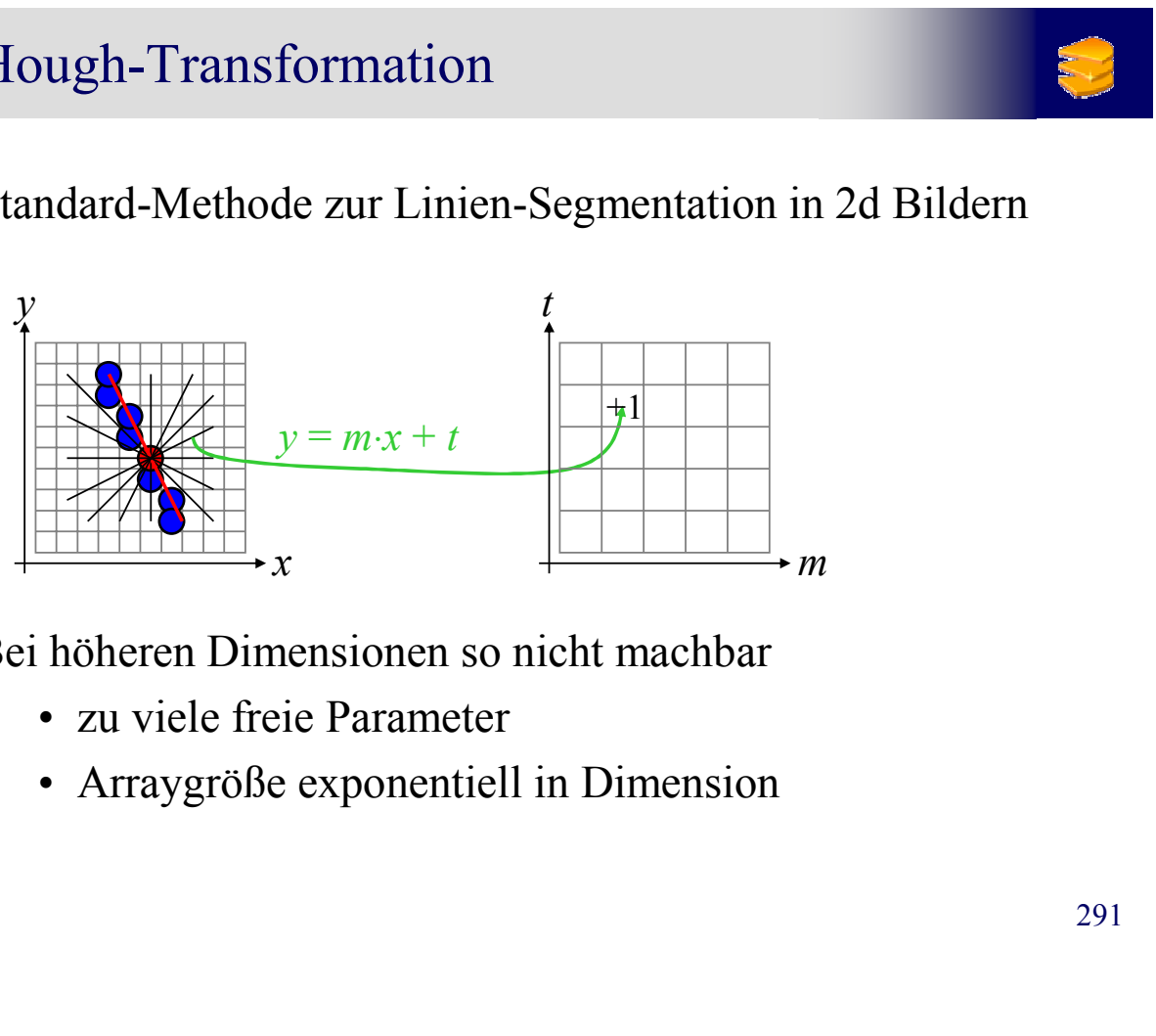
- Auswertung einer k-Nearest-Neighbor-Query für $k = \{1,2,...,k_{max}\}$
- Auftragen der k-NN-Distanzen in doppelt logarithmischem Maßstab
- Ergibt sich annähernd eine Gerade, dann entspricht die Steigung der Gerade der fraktalen Dimension

Fazit zur fraktalen Dimension

- Klare Unterscheidung zwischen korrelierten und nicht korrelierten Punktmengen nur in der Theorie
- Für die Clustering-Anwendung als Zusatz-Kriterium evtl. brauchbar
- Idee: OPTICS-Plot um fraktale Dimension erweitern:



Hough-Transformation

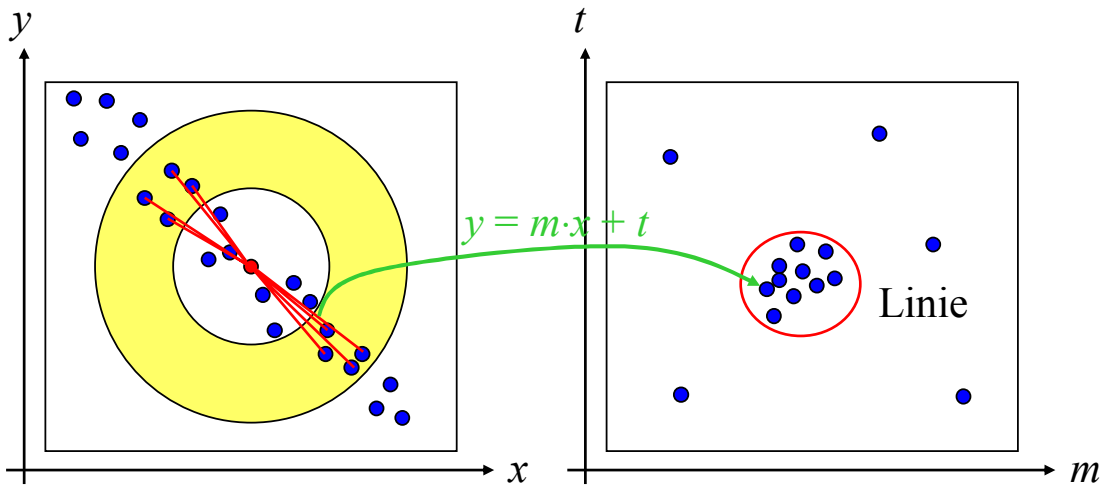


Standard-Methode zur Linien-Segmentation in 2d Bildern

Bei höheren Dimensionen so nicht machbar

- zu viele freie Parameter
- Arraygröße exponentiell in Dimension

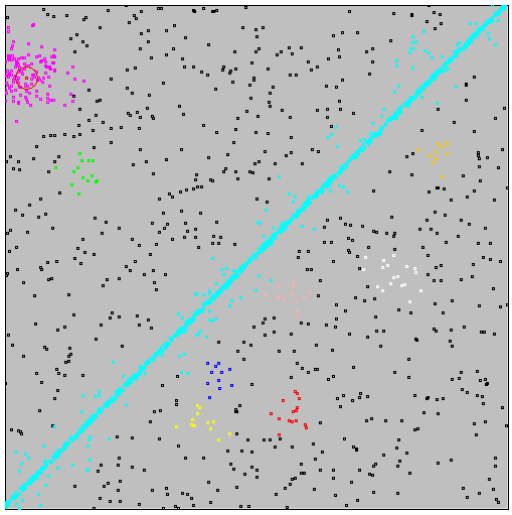
Modifikation des Verfahrens



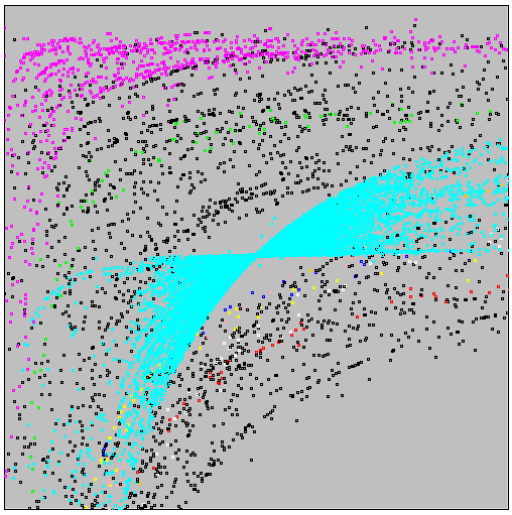
- Betrachte Paare (bzw. Tripel, Quadrupel) von Punkten
  - nahe beieinander liegend
  - oder zufallsbasiert
- Transformiere in Parameterraum
- Konventionelles Clustering im transformierten Raum

Modifizierte Hough-Transformation

Merkmals-Raum:



Transformierter Raum:



Zusätzliche Erweiterungsmöglichkeiten

- DBSCAN durch OPTICS ersetzen, um Hierarchien von Korrelations-verbundenen Punktmengen zu ermitteln
- Oder alternative Cluster-Methoden einsetzen
  (eigentlich sind beliebige Shapes hier nur begrenzt erwünscht)
- Transformation in polynomialen Vektorraum, um auch nicht-lineare Abhängigkeiten zu finden:
  $(x,y,z) \rightarrow (x,y,z,x^2,y^2,z^2,xy,xz,yz)$, wie bei SVMs
- Nutzung der Algorithmen auch für Subspace-Clustering, indem man PCA durch eine Methode der Merkmals-*Selektion* ersetzt

Literatur

C. Aggarwal and P. Yu. *Finding Generalized Projected Clusters in High Dimensional Space*. In Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'00), Dallas, TX, 2000.

C. C. Aggarwal and C. Procopiuc. *Fast Algorithms for Projected Clustering*. In Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'99), Philadelphia, PA, 1999.

R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. *Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications*. In Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'98), Seattle, WA, 1998.

C.-H. Cheng, A.-C. Fu, and Y. Zhang. *Entropy-Based Subspace Clustering for Mining Numerical Data*. In Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery in Databases (SIGKDD'99), San Diego, CA, 1999.

S. Goil, H. Nagesh, and A. Choudhary. *MAFIA: Efficient and Scalable Subspace Clustering for Very Large Data Sets*. Tech. Report No. CPDC-TR-9906-010, Center for Parallel and Distributed Computing, Dept. of Electrical and Computer Engineering, Northwestern University, 1999.

A. Hinneburg and D. Keim. *Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering*. In Proc. 25th Int. Conf. on Very Large Databases (VLDB'99), 1999.

I. Joliffe. Principal Component Analysis. Springer-Verlag, New York, 1986.

K. Kailing, H.-P. Kriegel, P. Kröger. *Density-Connected Subspace Clustering for High-Dimensional Data*. To appear in Proc. SIAM Int. Conf. on Data Mining (SDM'04), Orlando, FL, 2004.

K. Kailing, H.-P. Kriegel, P. Kröger. *Selecting Subspaces Relevant for Clustering High-Dimensional Data*. Submitted for publication at SIGMOD Conference 2004.

K. Kailing, H.-P Kriegel, P. Kröger, and S. Wanka. *RIS: Ranking Interesting Subspaces of High Dimensional Data*. In Proc. 7th Europ. Conf. On Principles and Practice of Knowledge Discovery and Data Mining (PKDD´03), Cavtat, Kroatien, 2003.

H. Nagesh, S. Goil, and A. Choudhary. *Adaptive Grids for Clustering Massive Data Sets*. In 1st SIAM Int. Conf. on Data Mining, Chicago, IL, 2001.

C. M. Procopiuc, M. Jones, P. K. Agarwal, T. M. Murali. *A Monte Carlo Algorithm for Fast Projective Clustering*. In Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'02), Madison, WN, 2002.