# Kapitel 4
# Ähnlichkeitssuche in Zeitreihen

Skript zur Vorlesung: Spatial, Temporal, and Multimedia Databases
Sommersemester 2009, LMU München

# Überblick

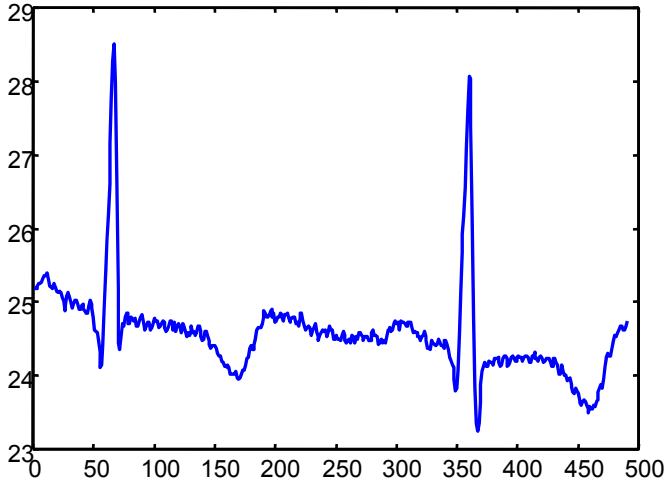
4.1 Einleitung

4.2 Matching-basierte Analyse

4.3 Threshold-basierte Analyse

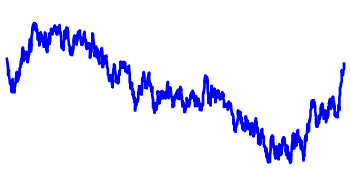
# 4.1 Einführung

- Zeitreihe = Sammlung von Beobachtungen die zeitlich sequentiell gemacht werden/wurden
  - Zeitreihe $o \in (\Re^d \times Time)$ mit $o = [(o_1,t_1), \ldots,(o_n,t_n)]$
    - Meist $d = 1$
    - *Time* ist die Zeitdomäne, meist diskret
  - Diskrete Zeitreihe daher $o = [o_1, \ldots, o_n]$
    - Oft werden die Werte zwischen zwei Zeitpunkten interpoliert

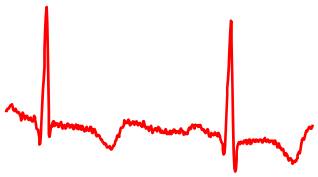


– Zeitreihen sind allgegenwärtig
  - Menschen messen alles Mögliche ...
    – Die Zustimmung der Bevölkerung zur Regierung
    – Den Blutdruck
    – Die Anzahl der Sonnenstunden in Freiburg pro Jahr
    – Der Wert der BVB-Aktie
    – Die Anzahl der Webhits pro Sekunde

    … und diese Dinge verändern sich über die Zeit hinweg
  - Zeitreihen werden in allen Anwendungsgebieten erzeugt



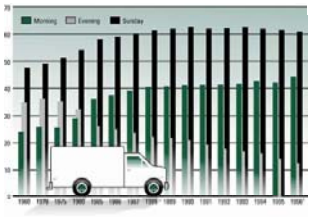
Naturwissenschaft



Medizin

Wirtschaft

...




...

- Herausforderungen
  - Große Datenvolumina
    - 1h EKG: ca. 1GB
    - Typischer Weblog: ca. 5GB pro Woche
    - Datenbank mit Space Shuttle Messwerten: ca. 160 GB
    - …
  - Heterogene Daten
    - Verschiedene Datenformate
    - Verschiedene Sampling Raten
    - Verrauschte Signale
    - Fehlende Werte
    - …
  - Subjektive Wahrnehmung der Ähnlichkeit abhängig von
    - User
    - Anwendung
    - Analyse-Art
      (Matching, Threshold-basiert, …)

# – Daten Vorverarbeitung

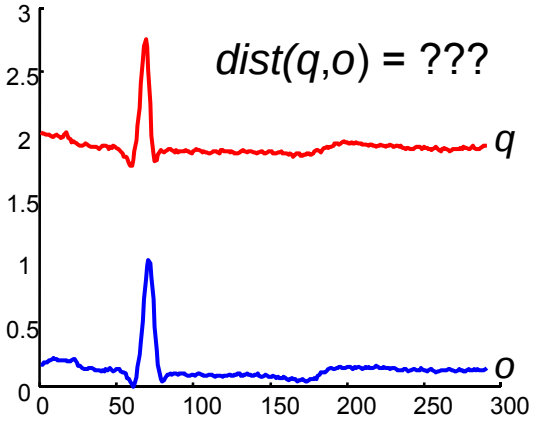
- Beseitigung von Verzerrungen in den Rohdaten
- Die wichtigsten Verzerrungen

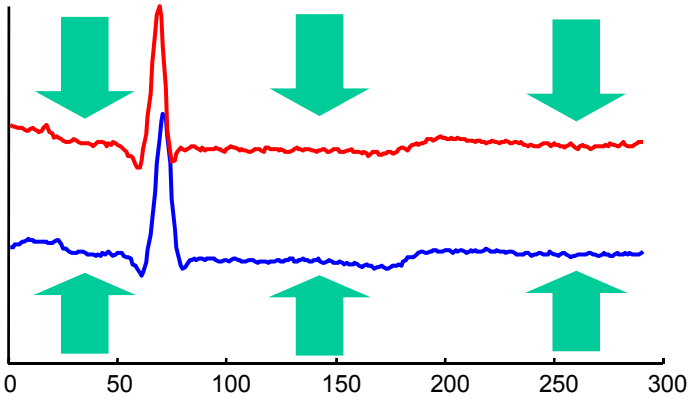
## – Offset Translation

» Ähnlich Zeitreihen mit unterschiedlichen Offsets

» Verschiebung aller Zeitreihen um den Mittelwert *MW*:

$\forall 1 \leq i \leq |o|: o_i = o_i - MW(o)$

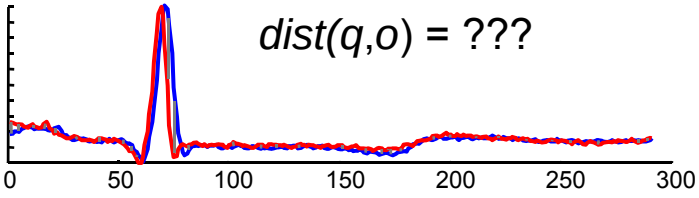



$q = q - MW(q)$

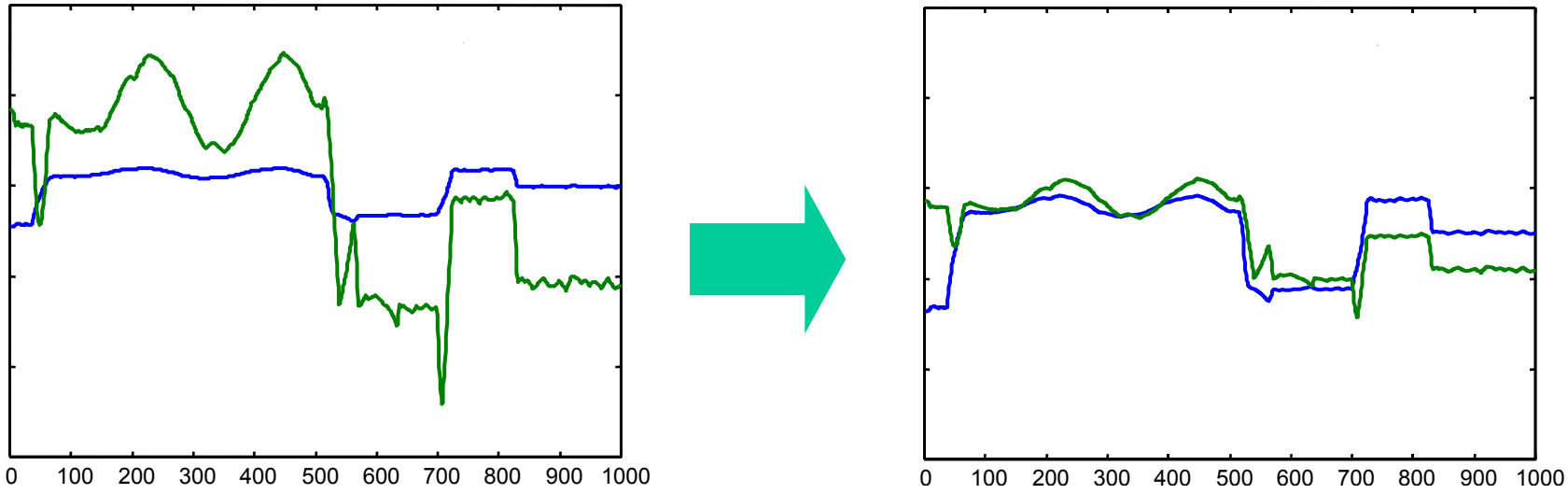
$o = o - MW(o)$

dist(q,o) = ???



– Amplituden Skalierung
  » Zeitreihen mit ähnlichem Verlauf aber unterschiedlichen Amplituden
  » Verschiebung der Zeitreihen um den Mittelwert (*MW*) und Normierung der Amplitude mittels der Standard Abweichung (StD):

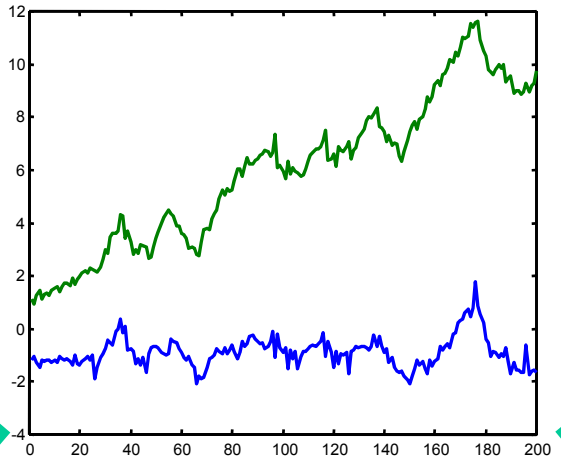
$$\forall 1 \leq i \leq |o|: o_i = (o_i - \mathrm{MW}(o)) / \mathrm{StD}(o)$$



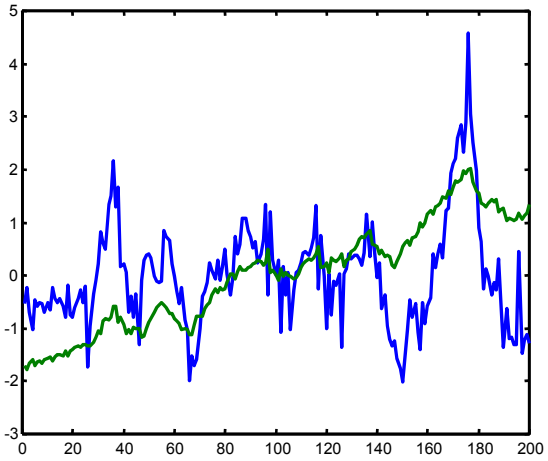

$$q = (q - MW(q)) / \mathrm{StD}(q)$$

$$o = (o - MW(o)) / \mathrm{StD}(o)$$

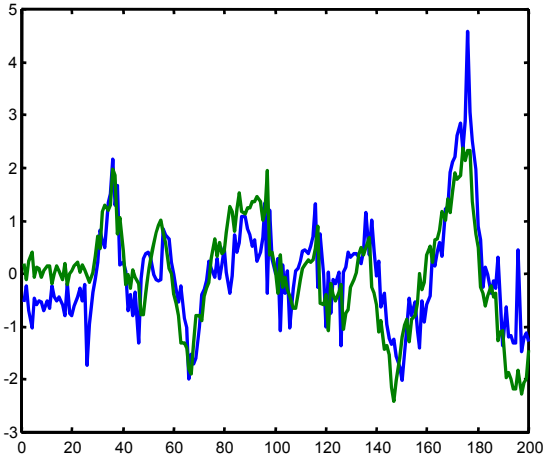
– Lineare Trends
  » Ähnliche Zeitreihen mit unterschiedlichen Trends
  » Intuition:
    - Bestimme Regressionslinie
    - Verschiebe Zeitreihe anhand dieser Linie



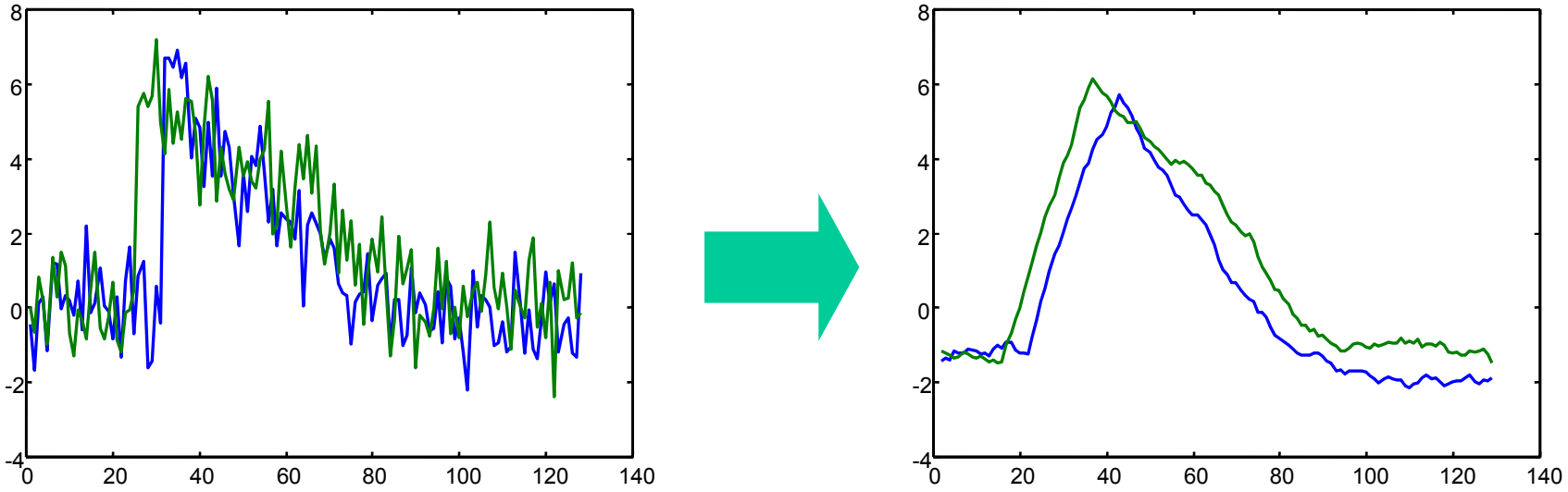

Offset Translation + Amplituden Skalierung




Offset Translation + Amplituden Skalierung
**+ Lineare Trend Beseitigung**

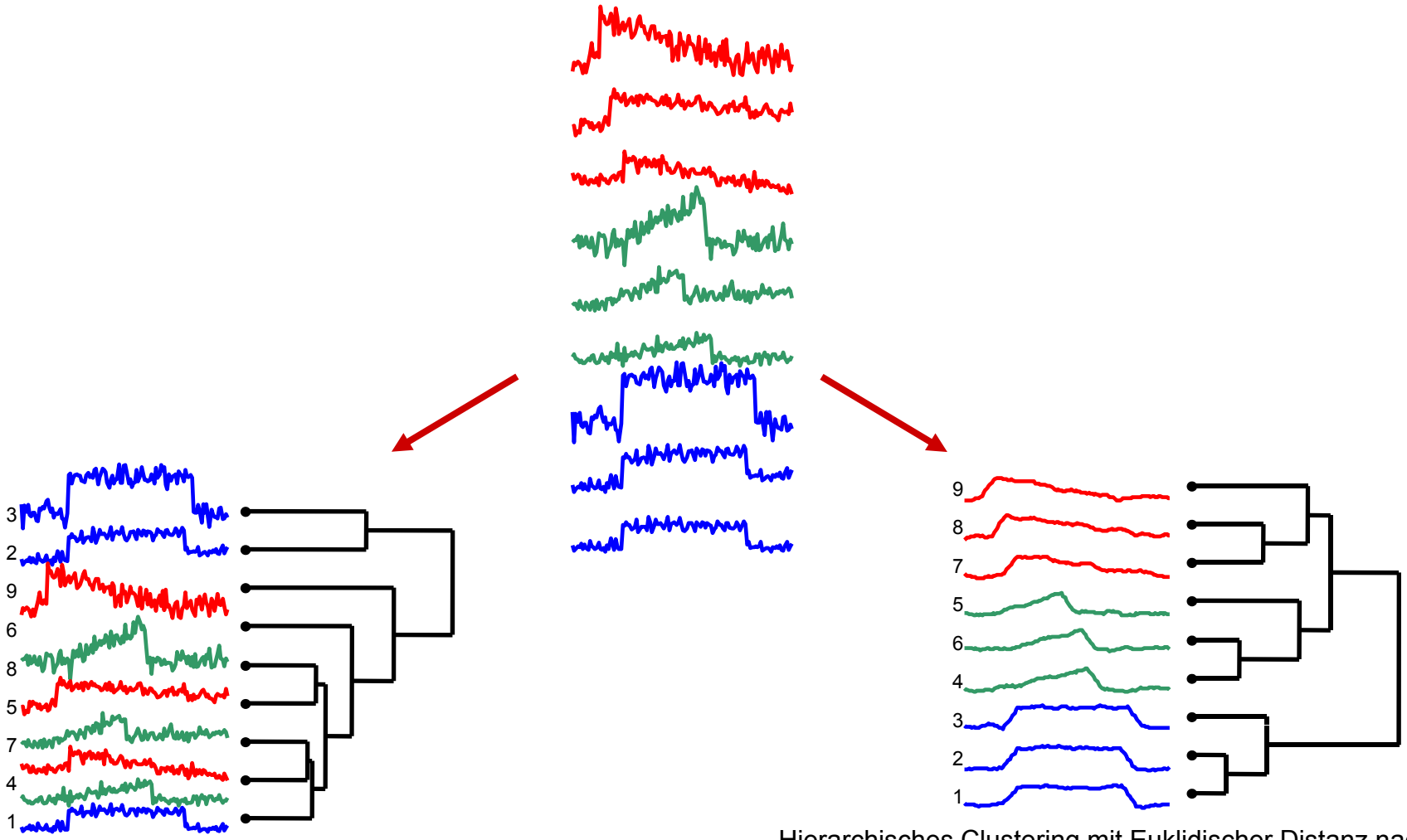


– Bereinigung von Rauschen
  » Ähnliche Zeitreihen mit hohem Rauschanteil
  » Glättung:
    Bilde für jeden Wert $o_i$ den Mittelwert über alle Werte $[o_{i-k}, \ldots, o_i, \ldots, o_{i+k}]$ für ein gegebenes $k$



- Beispiel

Rohdaten: Zeitreihen aus drei Klassen (erkennbar durch unterschiedliche Farbcodierung)



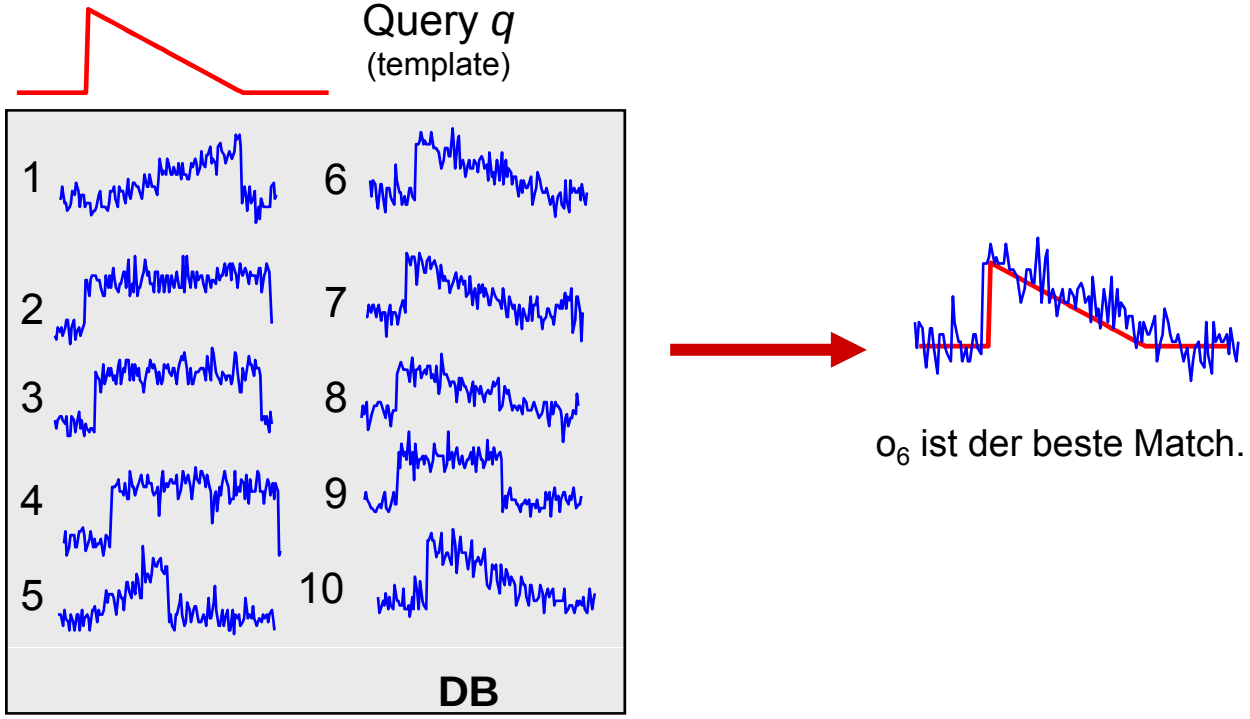

Hierarchisches Clustering der Rohdaten mit Euklidischer Distanz

Hierarchisches Clustering mit Euklidischer Distanz nach Bereinigung von Rauschen, Beseitigung linearer Trends, Amplituden Skalierung und Offset Translation

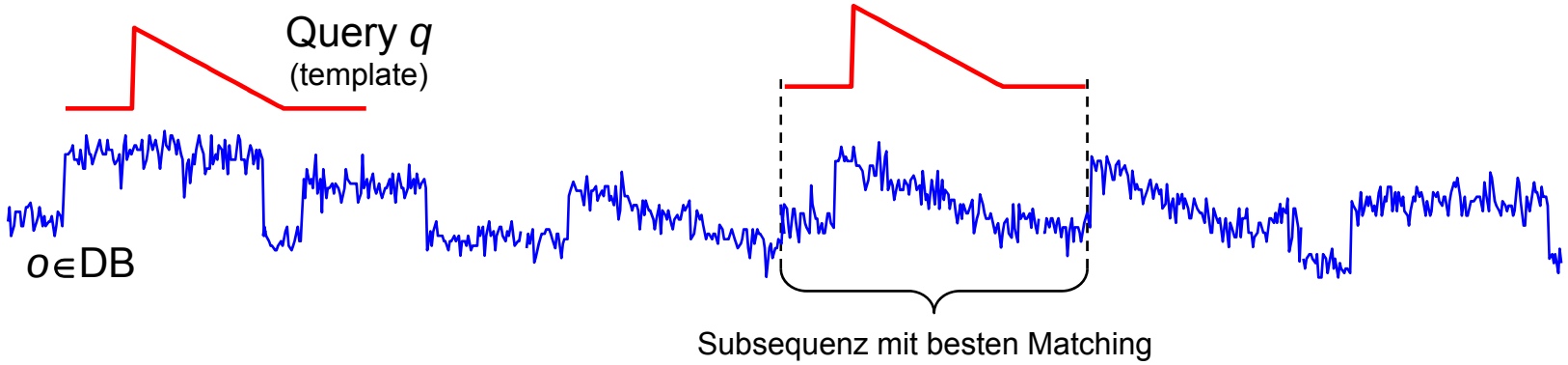
- Zusammenfassung
  - Die Rohdaten können unterschiedlichste Verzerrungen haben, die vor der Analyse beseitigt werden sollten
  - Welche Vorverarbeitungsschritte ausgeführt werden sollen, hängt typischerweise von der Anwendung ab
  - **ACHTUNG**: Oft sind die Verzerrungen die interessanten Informationen, die man durch eine Analyse finden will
  - Verschiedene „high-level“ Repräsentationen von Zeitreihen, die wir später kennen lernen, bieten elegante Möglichkeiten, diese Verzerrungen in den Griff zu bekommen

# 4.2 Matching-basierte Analyse

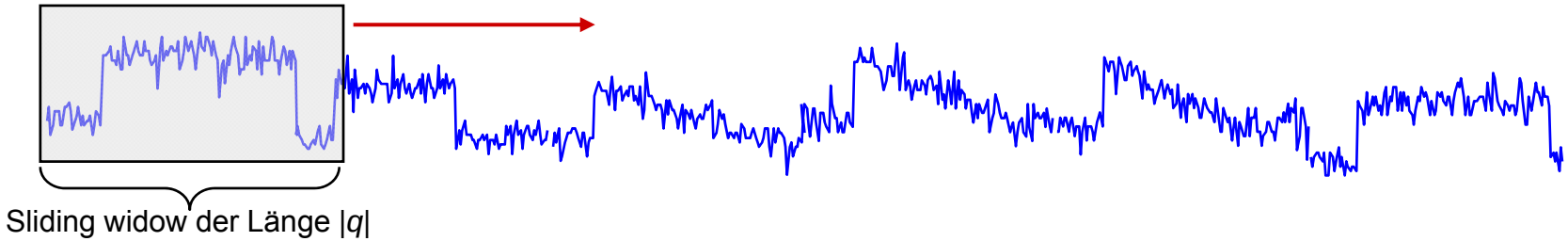
- Analyse-Arten
  - Gesamt-Suche („Whole-Matching“)
    - Gegeben: Query-Objekt *q* (Zeitreihe oder Template), Distanzfkt. *dist*
    - Gesucht: Zeitreihe $o \in$ DB, die am besten mit *q* „matched“ (als Ganzes)



- Subsequenz-Suche („Subsequence Matching“)
  - Gegeben: Query-Objekt *q* (Zeitreihe oder Template), Distanzfkt. *dist*
  - Gesucht: Zeitabschnitt, bei dem $o \in DB$ am besten mit *q* „matched“, bzw. das jenige $o \in DB$, bei dem dieser partielle Match am besten ist



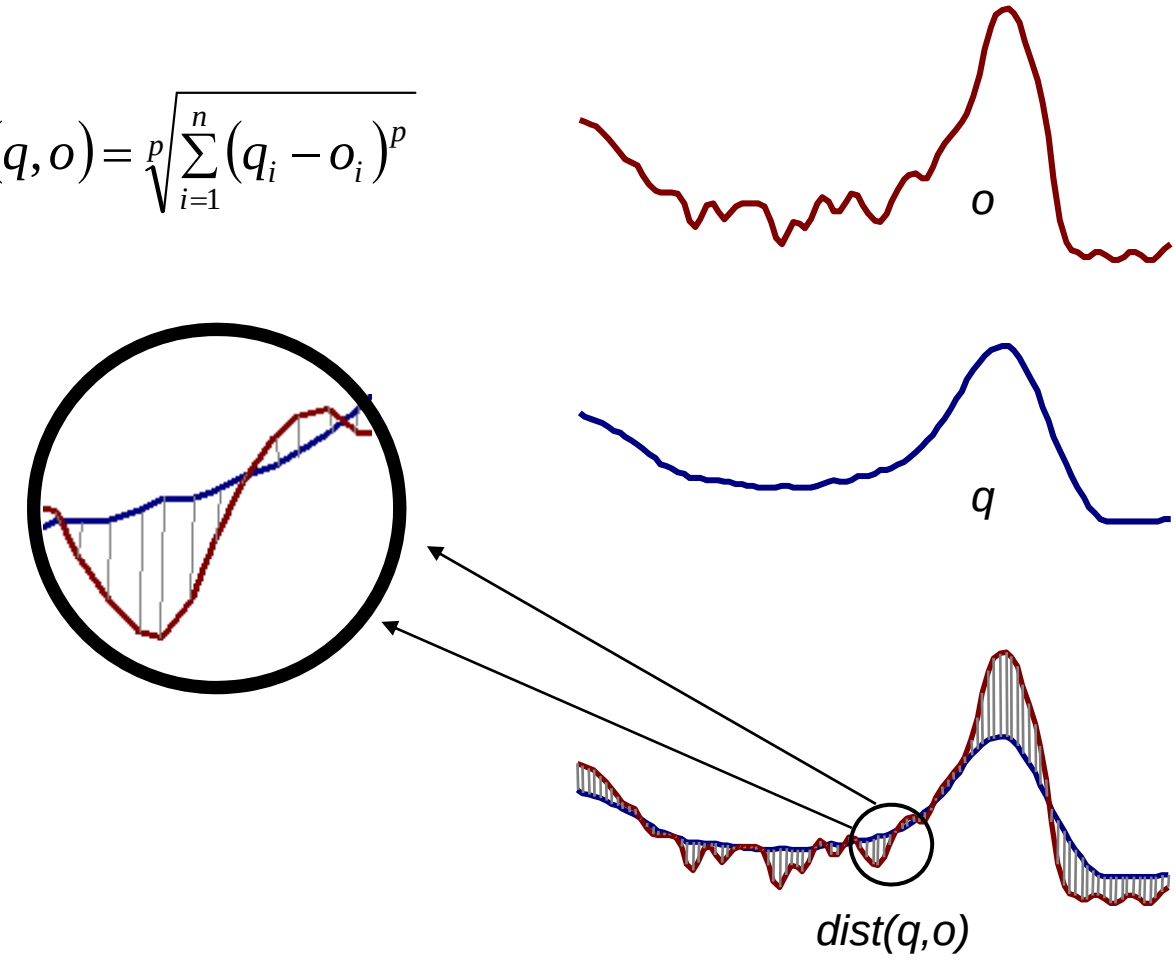

  - Bemerkung:
    - » ist Länge der Query-Subsequenz bekannt und fix, kann die Subsequenz-Suche immer in eine Gesamt-Suche überführt werden
    - » Verschiebe ein Fenster („sliding window“) über die Zeitreihen und materialisiere die Inhalte



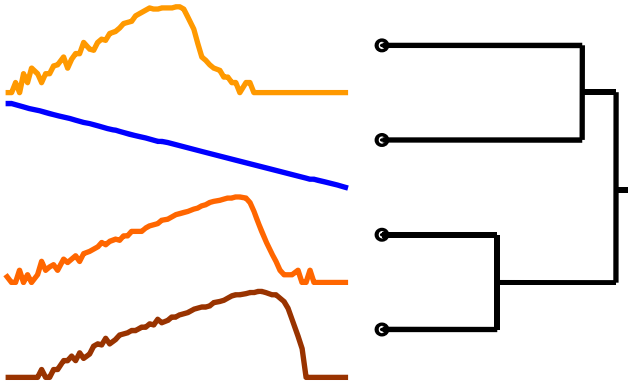
– Ähnlichkeit von Zeitreihen

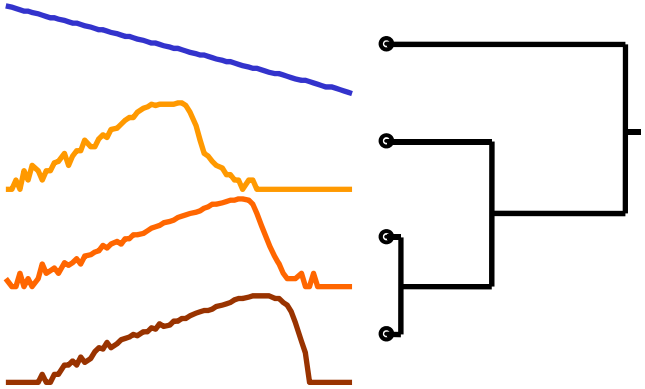
- Lp-Normen („Minkowski Metriken“)

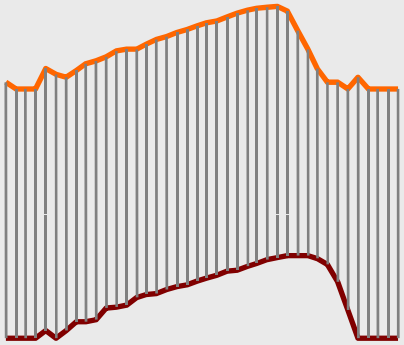
$$dist(q,o) = \sqrt[p]{\sum_{i=1}^{n} (q_i - o_i)^p}$$

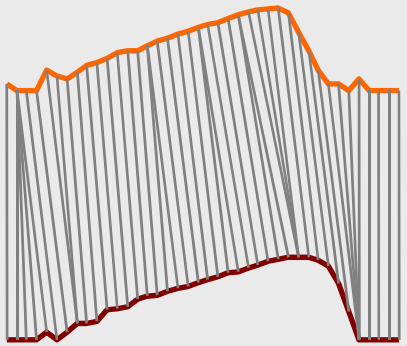


- Dynamic Time Warping (DTW)









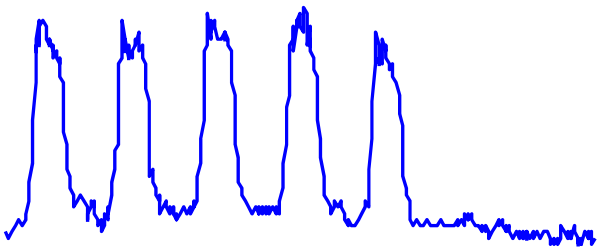
Fixierte Zeitachse (Lp-Norm)

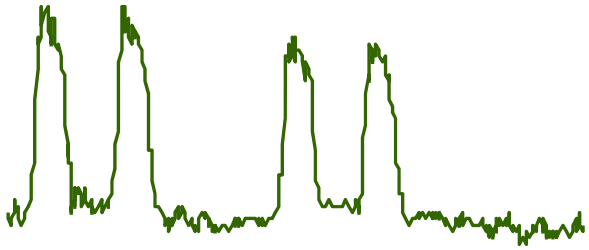
*Sequenzen werden „Eins-zu-Eins“ angeglichen und verglichen (Alignment)*

Verzogene („Gewarped”) Zeitachse

*Nichtlineare Alignments sind möglich*

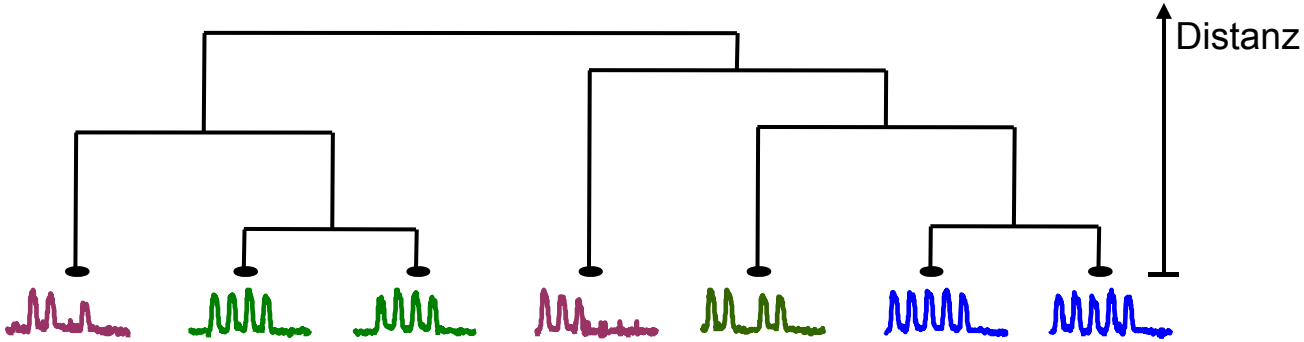
- Vergleich
  - DB enthält Zeitreihen, die den wöchentlichen Bedarf einer Firma messen (1 Zeitreihe entspricht 1 Woche)



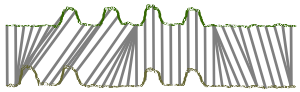




=> Mittwoch war Feiertag

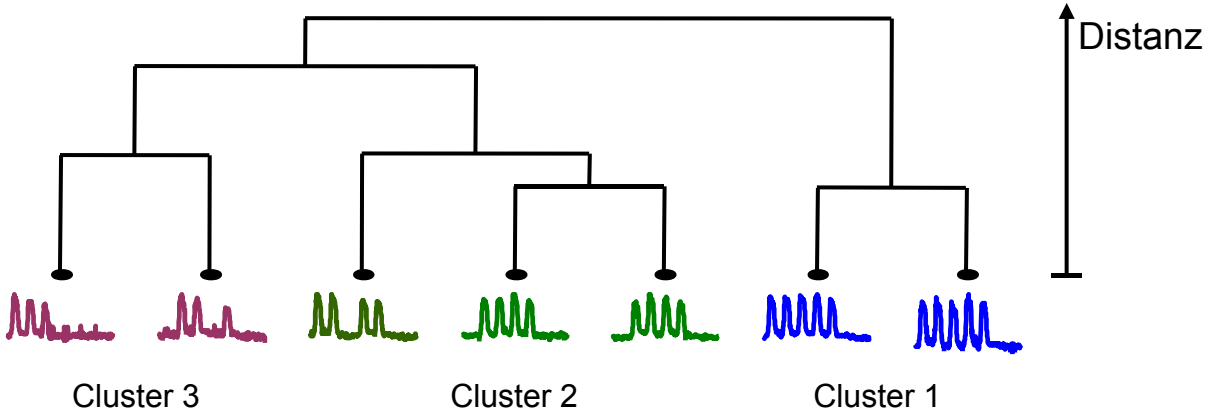
  - Ergebnis eines Clusterings mit Euklidischer Distanz
    - » Die beiden 5-Tage Wochen korrekt gruppiert
    - » Die drei 4-Tage Wochen und die beiden 3-Tage Wochen vermischt



– Ergebnis eines Clusterings mit Dynamic Time Warping



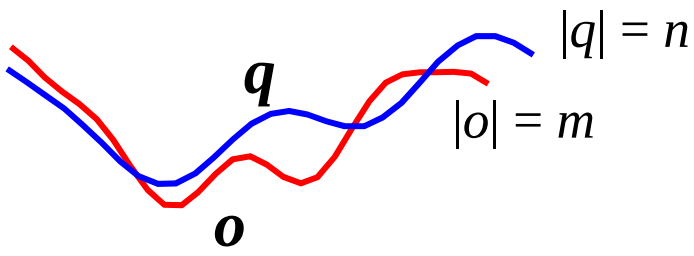
  - » Cluster 1: 5-Tage Wochen
  - » Cluster 2: 4-Tage Wochen
  - » Cluster 3: 3-Tage Wochen



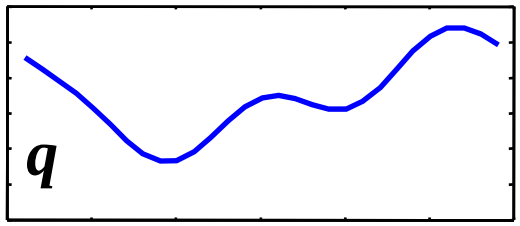

– Laufzeit:

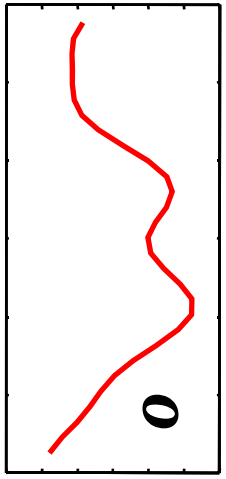
  - » Euklidische Distanz: ca. 1 Sekunde
  - » DTW: ca. 3,5 Stunden

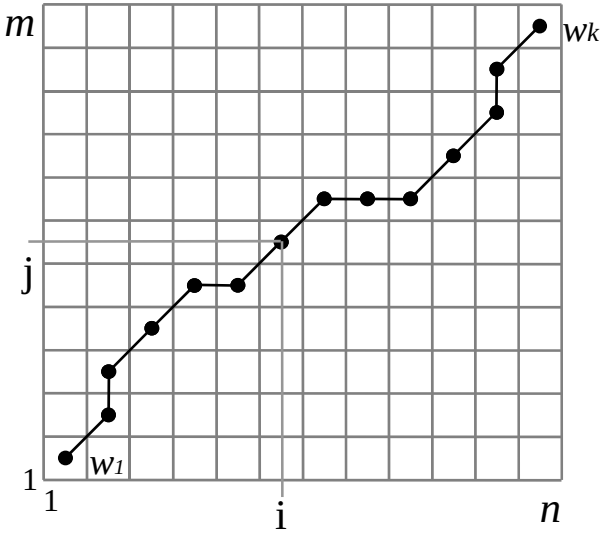
- Berechnung der DTW Distanz
  - Gegeben: Zeitreihen *q* und *o* unterschiedlicher Länge
  - Finde mapping von allen $q_i$ auf $o_j$ mit minimalen Kosten

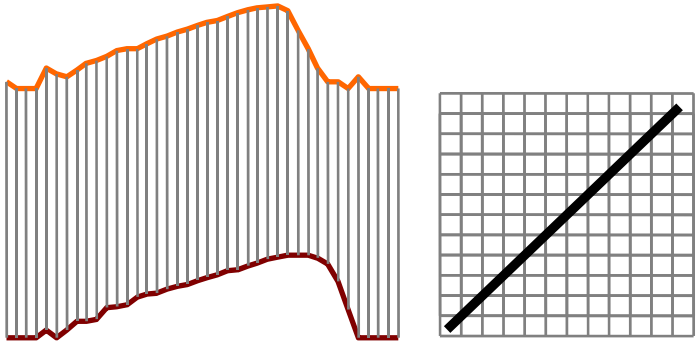


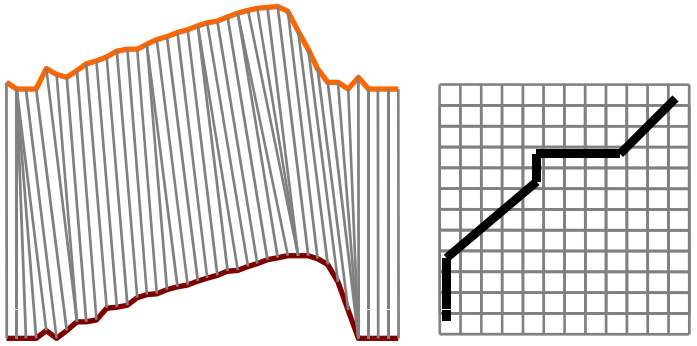

Suchmatrix



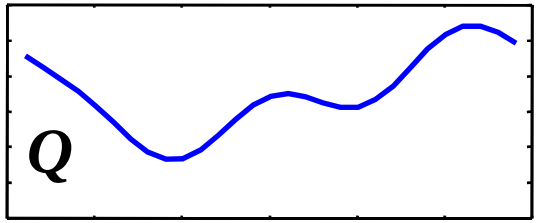



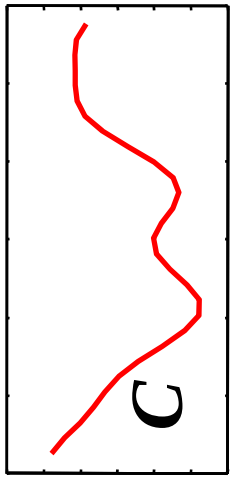


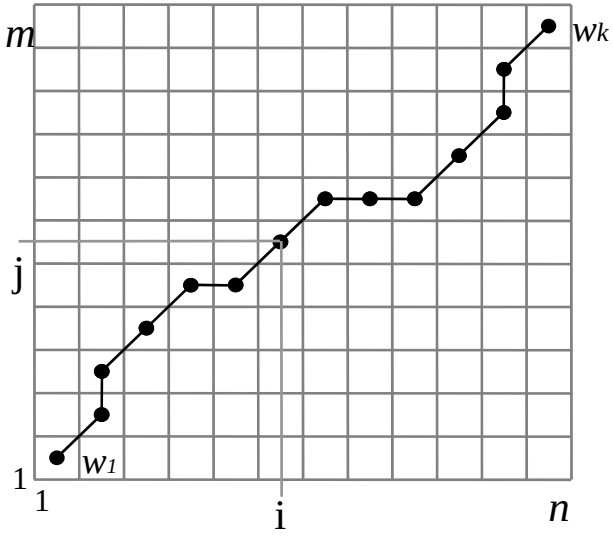
– Suchmatrix
  » Alle möglichen mappings von q auf o können als „warping“ Pfad in der Suchmatrix aufgefasst werden
  » Von all diesen Mappings suchen wir den Pfad mit den niedrigsten Kosten

$$DTW(q,o) = \min\left\{\sqrt{\sum_{k=1}^{K} w_k}\Big/K\right\}$$

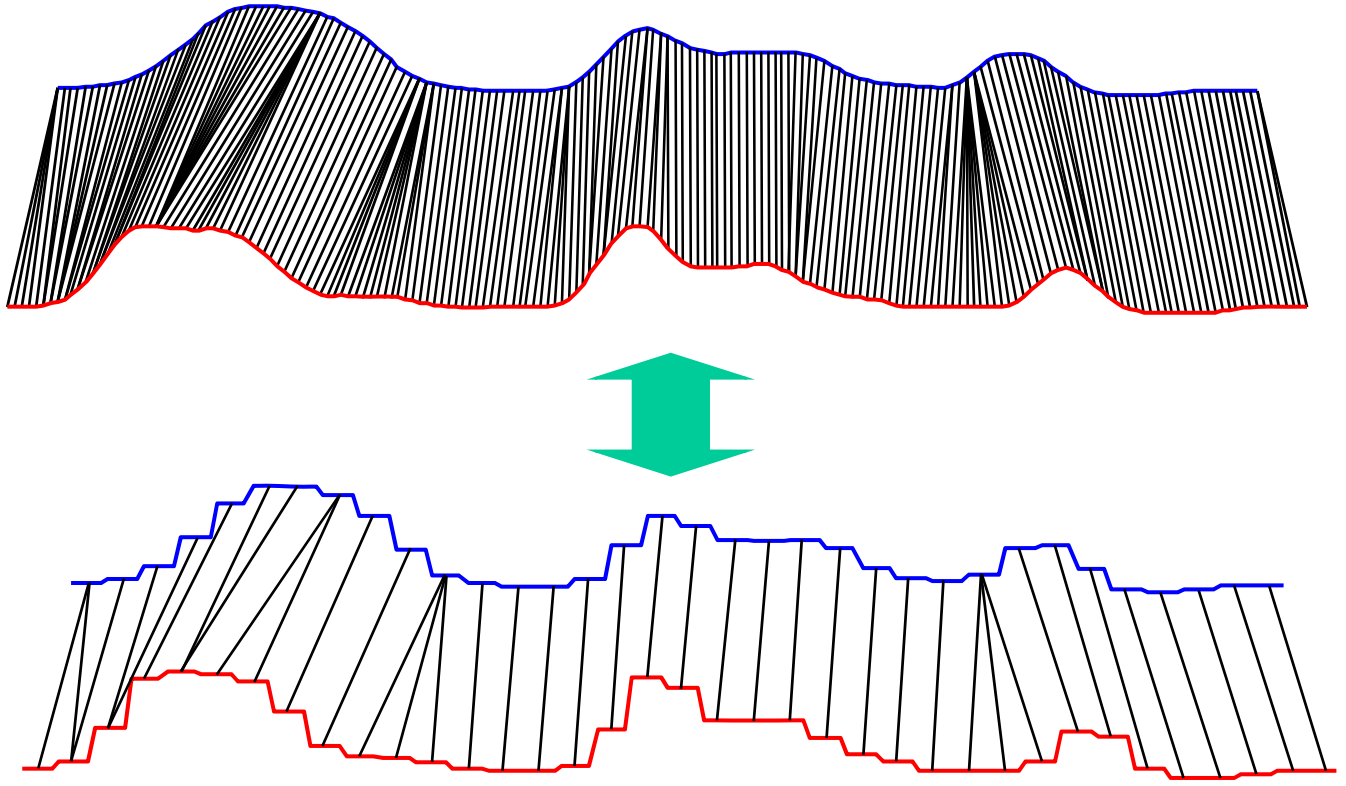
  » Theoretisch: exponentiell viele Pfade
  » Praktisch: Dynamisches Programmieren
  => Laufzeit ($n \cdot m$)



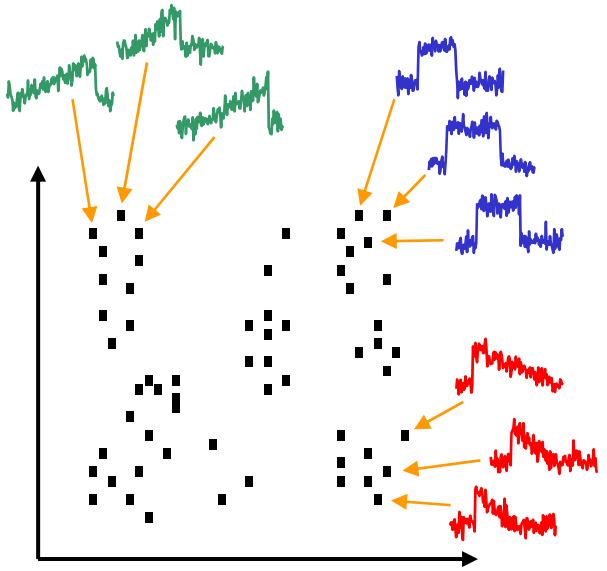



- Approximative Dynamic Time Warping Distanz
  - Idee
    - » Approximiere die Zeitreihen (komprimierte Repräsentation, Sampling, …)
    - » Berechne DTW auf den Approximationen



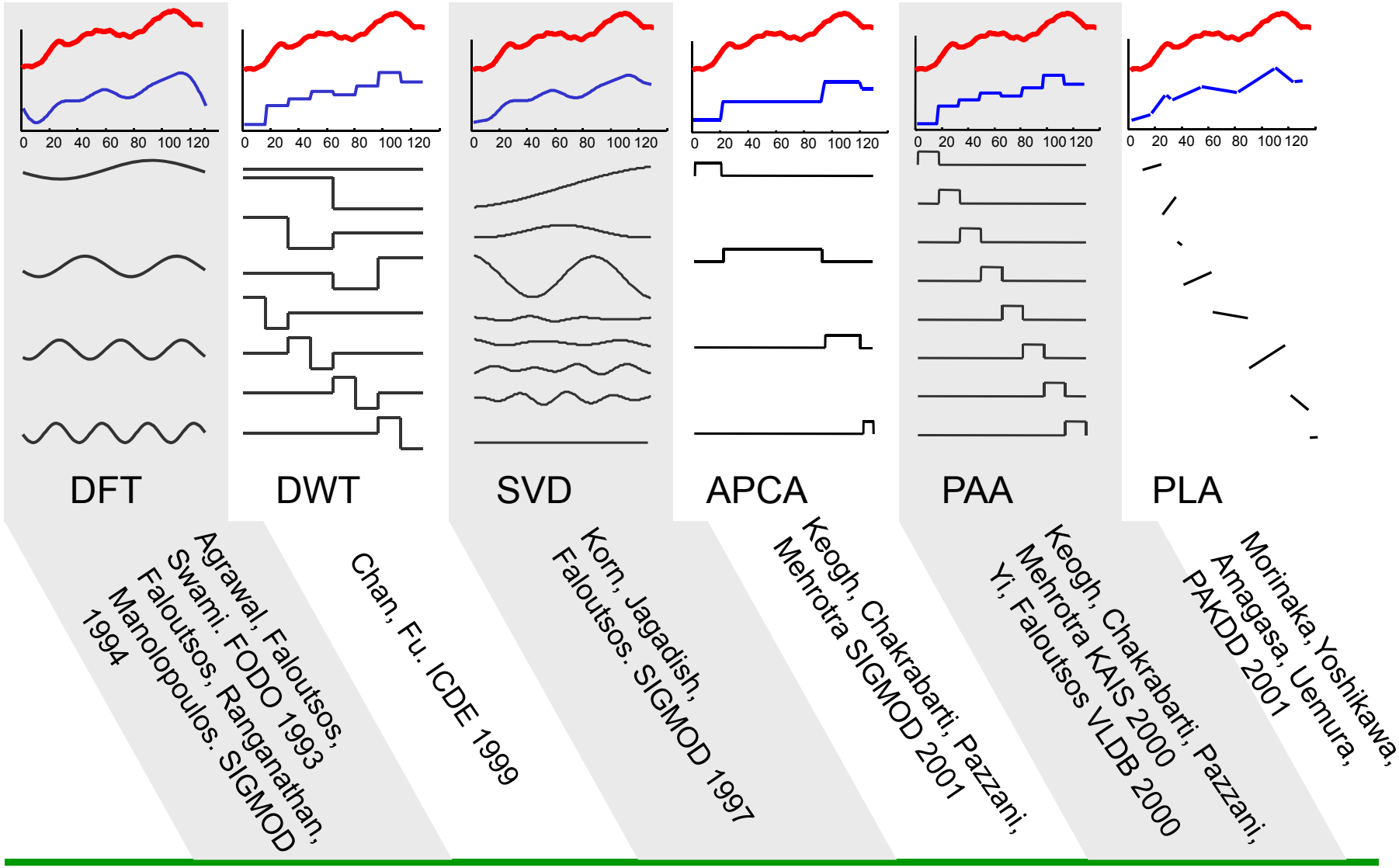
– Anfragebearbeitung
  - Indexierung:
    – eine Zeitreihe der Länge *n* kann als *n*-dimensionaler Feature-Vektor modelliert werden



  - Problem:
    – Zeitreihen meist sehr lang
    – Curse of Dimensionality
  - Lösung: GEMINI-Framework [Faloutsos, Ranganathan, Maolopoulos. SIGMOD 1994]
    – Basiert auf Dimensionsreduktion
    – Transformiere *n*-dimensionale Zeitreihen in *d*-dimensionale Zeitreihen (*d*<<*n*)
    – Definiere eine Distanzfunktion für die *d*-dimensionalen Zeitreihen, die die Untere Schranke Eigenschaft erfüllt

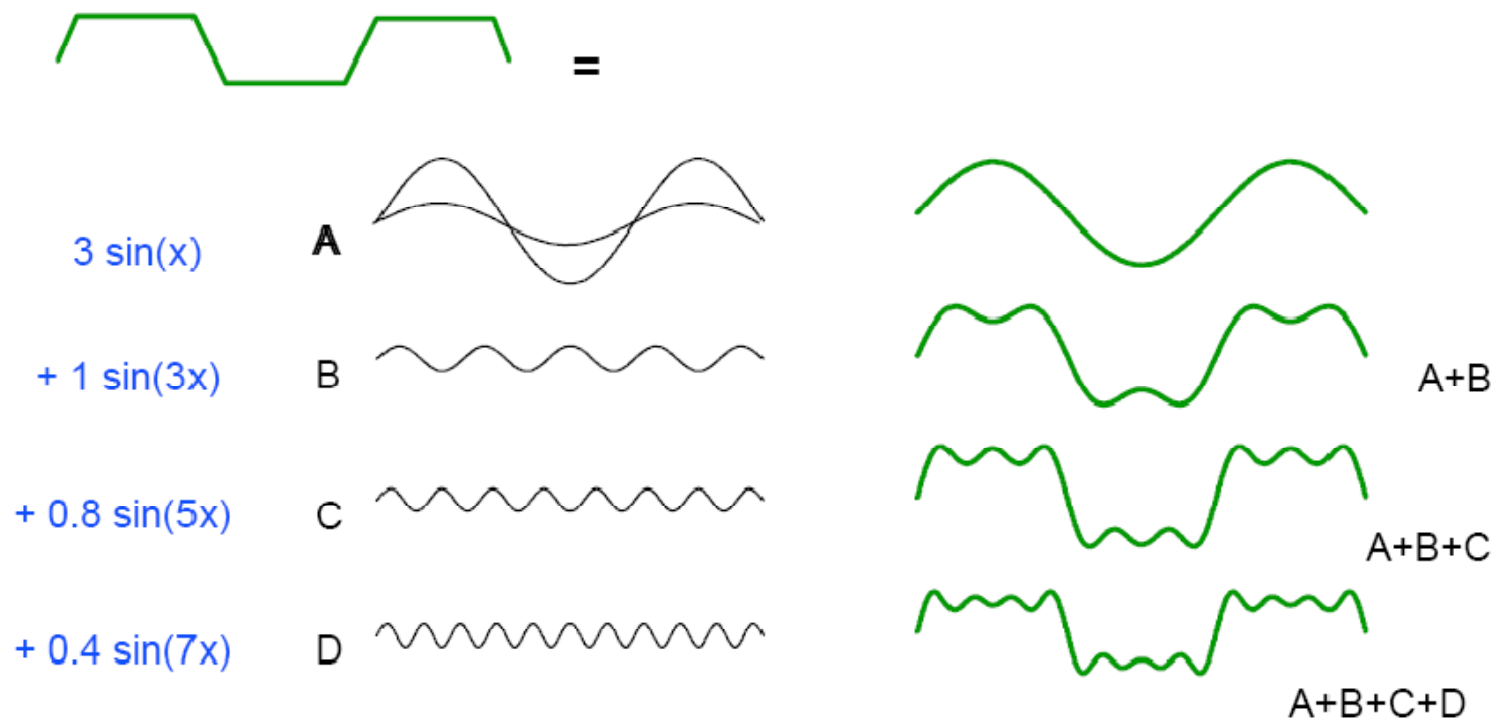
$$dist_{reduziert}(p,q) \leq dist_{original}(p,q)$$

- Techniken zur Dimensionsreduktion für Zeitreihen (Überblick)



# – Diskrete Fourier Transformation (DFT)

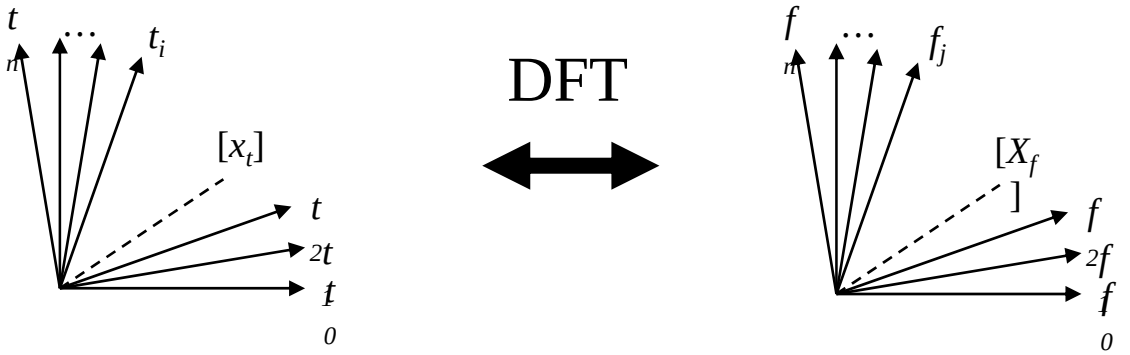
- Idee
  - Beschreibe beliebige periodische Funktion als gewichtete Summe periodischer Grundfunktionen (Basisfunktionen) mit unterschiedlicher Frequenz
  - Basisfunktionen: sin und cos



- DFT
  - Fouriers Theorem:

    Jede beliebige periodische Funktion lässt sich darstellen als Summe von Kosinus- und Sinus-Funktionen unterschiedlicher Frequenzen.

  - Transformation verändert eine Funktion nicht, sondern stellt sie nur anders dar
  - Transformation ist umkehrbar => inverse DFT
  - Analogie: Basiswechsel in der Vektorrechnung




  - Was ist diese andere „Basis“?

- Formal
  - Gegeben sei eine Zeitreihe der Länge $n$: $x = [x_t]$, $t = 0, \ldots, n-1$
  - Die DFT von $x$ ist eine Sequenz $X = [X_f]$ von $n$ komplexen Zahlen für die Frequenzen $f = 0, \ldots, n-1$ mit

$$X_f = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \cdot e^{\frac{-j2\pi ft}{n}} =$$

$$\underbrace{\frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \cos(\frac{2\pi ft}{n})}_{\text{Realteil}} - j \cdot \underbrace{\frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \sin(\frac{2\pi ft}{n})}_{\text{Imaginärteil}}$$

    wobei $j$ die imaginäre Einheit bezeichnet, d.h. $j^2 = -1$.
  - Der Realteil gibt den Anteil der Kosinus- und der Imaginärteil den Anteil der Sinusfunktionen in der jeweiligen Frequenz $f$ an.

– Durch die inverse DFT wird das ursprüngliche Signal *x* wieder hergestellt:

$$x_t = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} X_f \cdot e^{\frac{j 2\pi f t}{n}} \qquad t = 0, \ldots, n-1 \text{ (}t\text{: Zeitpunkte)}$$

$[x_t] \leftrightarrow [X_f]$ bezeichnet ein Fourier-Paar,
d.h. $DFT([x_t]) = [X_f]$ und $DFT^{-1}([X_f]) = [x_t]$.

– Die DFT ist eine lineare Abbildung, d.h. mit $[x_t] \leftrightarrow [X_f]$ und $[y_t] \leftrightarrow [Y_f]$ gilt auch:
  » $[x_t + y_t] \leftrightarrow [X_f + Y_f]$ und
  » $[ax_t] \leftrightarrow [aX_f]$ für ein Skalar $a \in IR$

– Energie einer Sequenz
  » Die Energie *E(c)* von *c* ist das Quadrat der Amplitude: $E(c) = |c|^2$.
  » Die Energie *E(x)* einer Sequenz *x* ist die Summe aller Energien über die Sequenz:

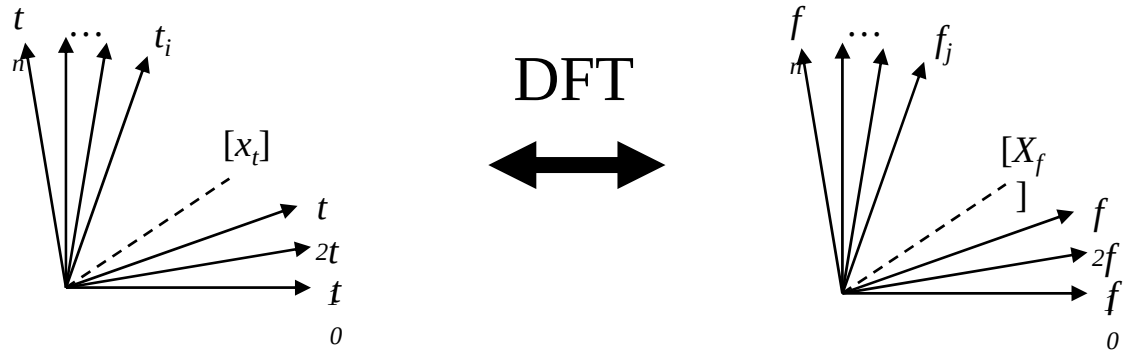
$$E(x) = \|x\|^2 = \sum_{t=0}^{n-1} |x_t|^2$$

– Satz von Parseval

  Die Energie eines Signals im Zeitbereich ist gleich der Energie im Frequenzbereich.
  Formal: Sei $X$ die DFT von $x$, dann gilt:

$$\sum_{t=0}^{n-1} |x_t|^2 = \sum_{t=0}^{n-1} |X_f|^2$$

– Folge aus Parsevals Satz und der Linearität der DFT: Die euklidische Distanz zweier Signale x und y stimmt im Zeit- und im Frequenzbereich überein: $\| x - y \|^2 = \| X - Y \|^2$




„Zeitbreich (-raum)“ „Frequenzbereich (-raum)“

- Grundidee der Anfragebearbeitung:
  - » Als Ähnlichkeitsfunktion für Sequenzen wird die euklidische Distanz verwendet:

$$dist(x,y) = \|x - y\| = \sqrt{\sum_{t=0}^{n-1} |x_t - y_t|^2}$$

  - » Der Satz von Parseval ermöglicht nun, die Distanzen im Frequenz- statt im Zeitbereich zu berechnen: $dist(x,y) = dist(X,Y)$
- Kürzen der Sequenzen für die Indexierung
  - » In der Praxis haben die tiefsten Frequenzen die größte Bedeutung.
  - » Die ersten Frequenz-Koeffizienten enthalten also die wichtigste Information.
  - » Für den Aufbau eines Index werden die transformierten Sequenzen gekürzt, d.h. von $[X_f]$, $f = 0, 1, \ldots, n-1$ werden nur die ersten $c$ Koeffizienten $[X_f < c]$, $c < n$, indexiert.
  - » Im Index kann dann eine untere Schranke der echten Distanz berechnet werden:

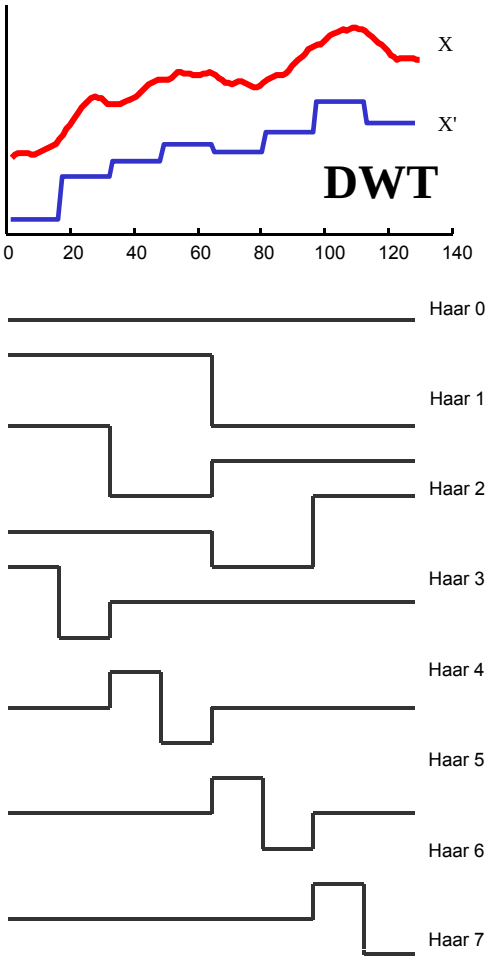
$$dist_c(x,y) = \sqrt{\sum_{f=0}^{c-1} |x_f - y_f|^2} \leq \sqrt{\sum_{f=0}^{n-1} |x_f - y_f|^2} = dist(x,y)$$

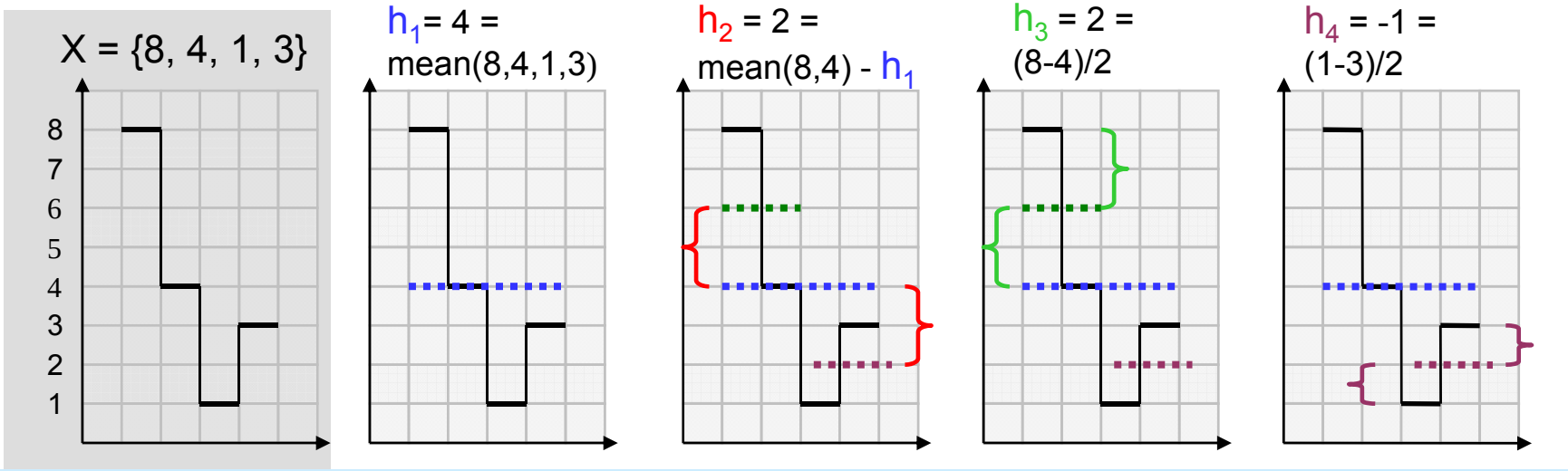
  - » Filter-Refinement: Filterschritt auf gekürzten Zeitreihen (mit Indexunterstützung), Refinement auf kompletten Zeitreihen

- Diskussion
  - Vorteile
    - » (Sehr) gute Komprimierung natürlicher Signale
    - » Effizient zu berechnen (O($n$ log $n$))
    - » Kann zeitliche Verschiebungs-invariante Anfragen unterstützen
  - Nachteile
    - » Zeitreihen müssen gleiche Länge haben
    - » Kein Support gewichteter Distanzfunktionen

# – Diskrete Wavelet Transformation (DWT)

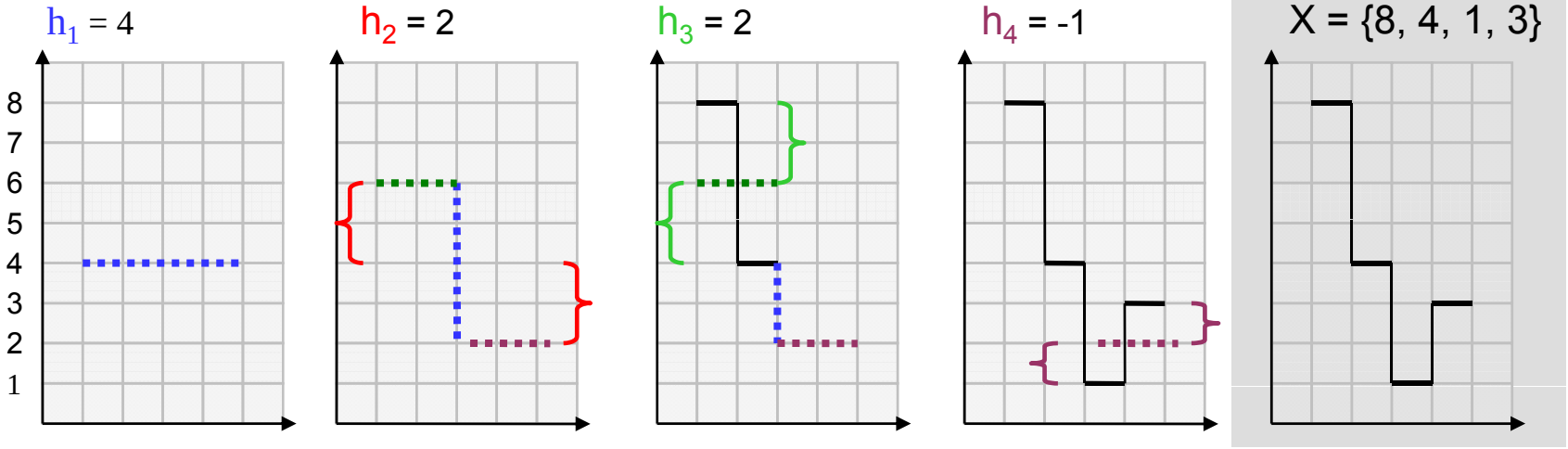
- Idee:
  - Repräsentiere Zeitreihe als Linearkombination von Wavelet-Basisfunktionen
  - Speichere nur die ersten Kooefizienten
  - Meist werden Haar-Wavelets als Basisfunktion gewählt (leicht zu implementieren)





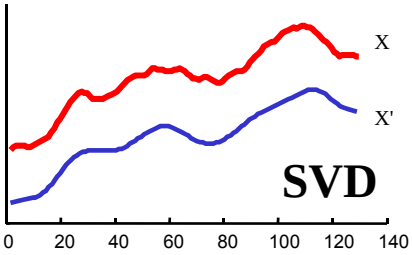

Schrittweise Transformation der Zeitreihe X = {8, 4, 1, 3} in die Haar Wavelet Darstellung H = [4, 2 , 2,-1]

Aus der Haar Wavelet Darstellung kann das ursprüngliche Signal verlustfrei wieder hergestellt werden.

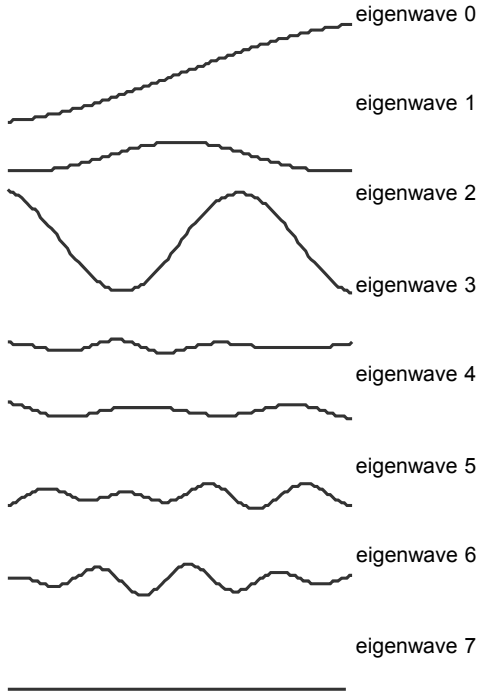


- Diskussion
  - Vorteile
    - » Gute Kompression v.a. bei stationären Signalen
    - » Kompression in linearer Laufzeit
    - » Einfache Unterstützung nicht-Euklidischer Distanzmaße (z.B. DTW)
  - Nachteile
    - » Länge der originalen Zeitreihen muss 2er-Potenz sein
    - » Länge der reduzierten Zeitreihen sollte 2er-Potenz sein
    - » Keine Unterstützung gewichteter Distanzmaße

- Singular Value Decomposition (SVD)
  - Idee
    - » Repräsentiere Zeitreihen als Linearkombination von Eigenwellen (Eigen Waves)
    - » Speichere nur die ersten (wichtigsten) Koeffizienten

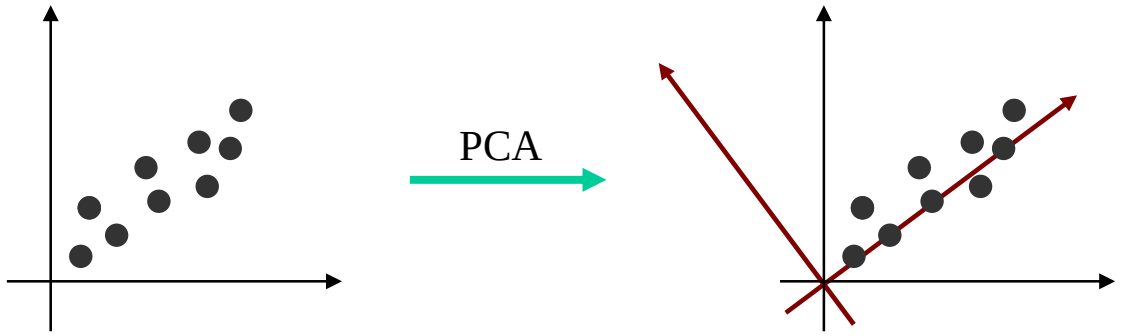



  - Vergleich SVD mit DFT und DWT
    - » Ähnlich: Linearkombination von Funktionen, die die Form der Zeitreihen modellieren
    - » Unterschied: SVD ist abhängig von den Daten (DFT: sin/cos, DWT: konstante Linien unterschiedlicher Amplituden)




  - SVD ist in der Textverarbeitung und dem Information Retrieval auch unter dem Acronym „Latent Semantic Indexing“

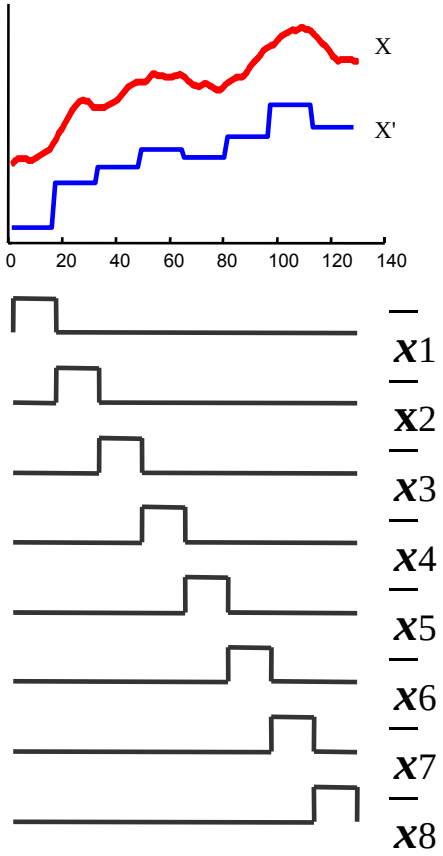
– Bestimmung der Eigenwellen
  - » Zeitreihen der Länge *n* = *n*-dimensionale Punkte
  - » PCA dieser Punkte
  - » Rotation der Zeitachsen auf die Hauptachsen aller Zeitreihen
  - » Erste Achse entlang der Richtung mit maximaler Varianz
  - » Zweite Achse entlang der Richtung mit maximaler Varianz orthogonal zur ersten Achse
  - » …
  - » Transformiere Zeitreihen in das neue Koordinatensystem (gegeben durch Hauptachsen) und speichere nur die ersten k Werte, da diese Dimensionen die meiste Varianz der Zeitreihen repräsentieren
  - » SVD minimiert den quadratischen Fehler, der durch das Weglassen von (*n*-*k*) Achsen gemacht wird



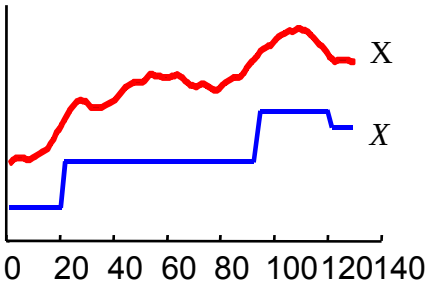

Alle Zeitreihenobjekte
als Punkte repräsentiert

- Vorteile
  - » Optimale Dimensionsreduktion durch Minimierung des kleinsten quadratischen Fehlers
  - » Daten-abhängig
- Nachteile
  - » Sehr aufwendig: $O(n^3)$
  - » Nur für einfache Euklidische Distanz (keine gewichtete oder nicht-Euklidische Distanzfunktion verwendbar)
  - » Schlecht für dynamische Daten: bei Einfügen/Löschen von Zeitreihen muss die gesamte SVD neu berechnet werden, da sich die Eigenwellen ändern könnten

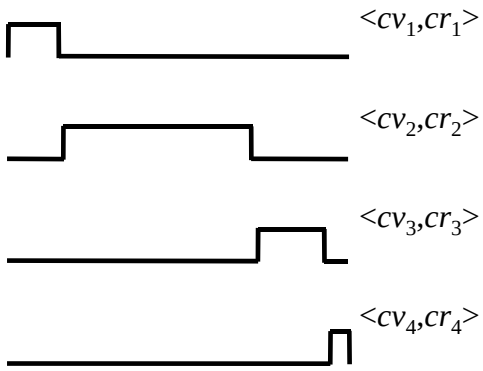
- Piecewise Aggregate Approximation (PAA)
  - Idee
    - » Repräsentiere Zeitreihen als Sequenz von Box-Basisfunktionen
    - » Jede Box hat dieselbe Länge
    - » Je länger die Boxen, desto niedriger die resultierende Approximation
  - Vorteile
    - » Schnell und einfach zu berechnen
    - » Unterstützt alle Arten von Distanzfunktionen
    - » Unterstützt Zeitreihen verschiedener Länge



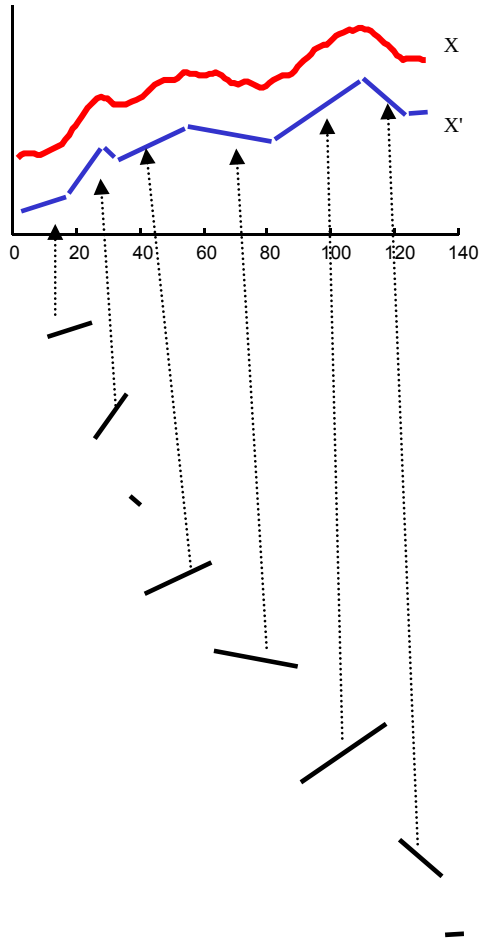
- Erweiterung: Adaptive Piecewise Constant Approximation (APCA)
  - Motivation
    - » Viele Zeitreihen haben Bereiche mit geringer Detailmenge (wenige Informationen) und Bereiche mit hoher Detailmenge
    - » PAA approximiert jeden Bereich mit derselben Detailtiefe




  - Idee
    - » Verwende Basisboxen mit unterschiedlicher Länge
    - » Speichere jedes Segment mit 2 Werten (vorher 1)
  - Vorteile
    - » Schnell und einfach zu berechnen
    - » Unterstützt alle Arten von Distanzfunktionen
    - » Unterstützt Zeitreihen verschiedener Länge
  - Nachteil
    - » Relativ komplexe Implementierung



- Piecewise Linear Approximation (PLA)
  - Idee
    - » Repräsentiere Zeitreihen als Sequenz von Linien-Segmenten
      S= (Länge,linke Höhe, rechte Höhe)
    - » Zwei aufeinanderfolgender Segmente können verbunden sein (müssen aber nicht), dann genügt
      S= (Länge,linke Höhe)
  - Berechnungskomplexität
    - » Optimale Lösung benötigt $O(n^2N)$ => für viele Anwendungen zu langsam
    - » Lineare Laufzeit durch Verwendung von Heuristiken möglich.
  - Vorteile
    - » Geeignet für „natürliche“ Signale
    - » Schnell und einfach zu berechnen (nicht optimal)
    - » Unterstützt alle Arten von Distanzfunktionen (insb. gewichtete Distanzfunktionen)
  - Nachteil:
    - » Keine Indexstruktur für PLA bekannt (aber: schneller sequentieller Scan möglich)



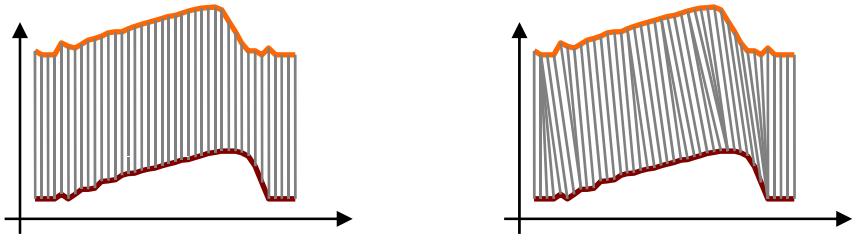
# 4.3 Threshold-Basierte Analyse

[Assfalg, Kriegel, Kröger, Kunath, Pryakhin, Renz.
Proc. 10th Int. Conf. on Extending Database Technology (EDBT), 2006]

– Bisherige Ansätze zur Repräsentation der Zeitreihen

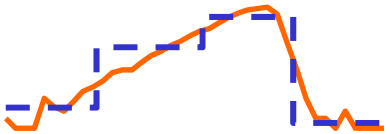
- Ziel:

  Effiziente Berechnung von Euklidischer Distanz und DTW, d.h. Ähnlichkeitssuche durch „matching“ der Amplitudengänge

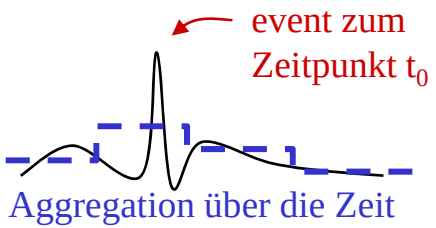


- Komprimierte Darstellung der Zeitreihen durch Dimensionsreduktion, d.h.

  **Aggregierung über die Zeit**



– Probleme bei Aggregierung über die Zeit:
  - Betrachtung signifikanter „Events“ schwierig




  - Fokusierung auf bestimmten Zeitpunkte/Zeitbereiche möglich aber nicht auf bestimmte Amplitudenwerte/Amplitudenbereiche
    – Sensoren sind oft Fehlerhaft in bestimmten Amplitudenspektren
      => Fokusierung auf relevante Amplituden(bereiche) wünschenswert.

    z.B.: Bilder von Digitalen Kameras sind oft in den extrem dunklen Regionen des Bildes und den extrem hellen Regionen des Bildes stark verauscht oder geben keine exakt authentischen Farbwerte wieder.

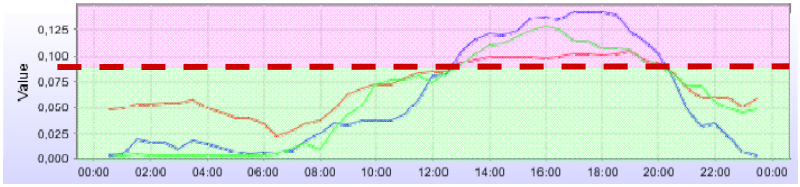
➔ Threshold-Basierte Repräsentation von Zeitreihen

Fokusierung auf relevante Amplituden(bereiche)

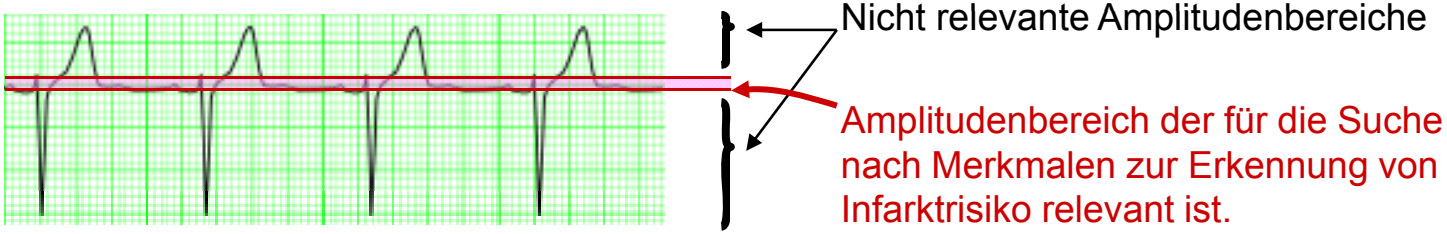
– Wozu Threshold-Basierte Analyse ?

Unterstützung von speziellen *Data Mining* Anwendungen

- Beispiel 1: Was sind die Ursachen für und was beeinflusst das Auftreten einer grenzwertigen Ozon-Belastung ?
  - Suche nach Regionen in denen die grenzwertige Ozon-Belastung ähnliches Verhalten bzgl. Zeitpunkt und Dauer aufweist.

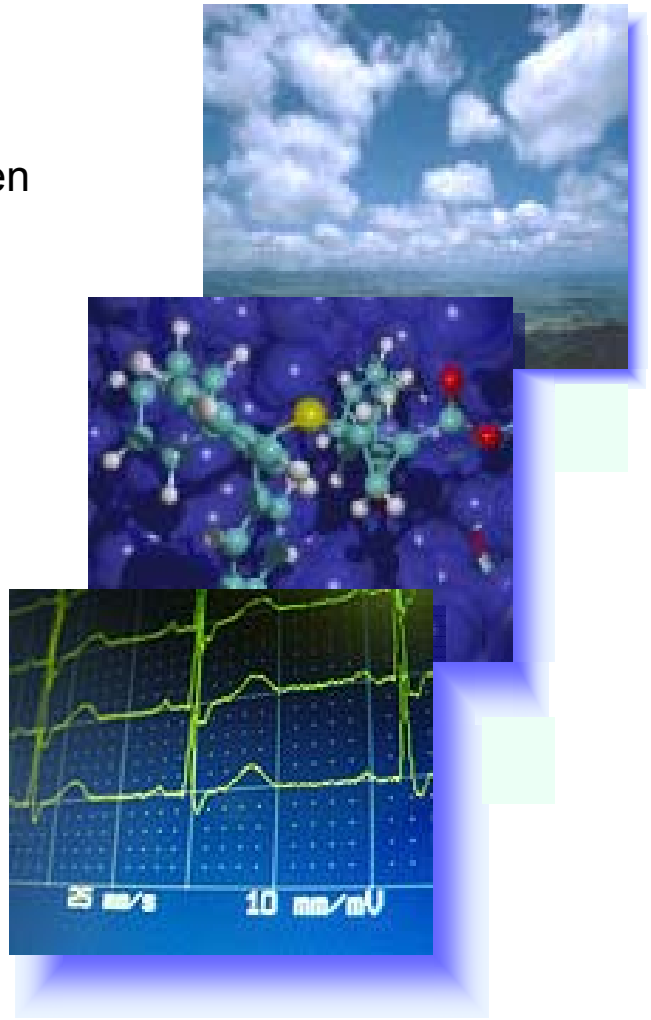



- Beispiel 2: Welche Patienten sind potentiell gefährdet einen Herzinfarkt zu bekommen ?
  - Ähnlichkeitssuche auf EKG-Daten fokusiert auf die relevanten Amplituden(bereiche)



# – Anwendungen für Threshold-Basierte Analyse

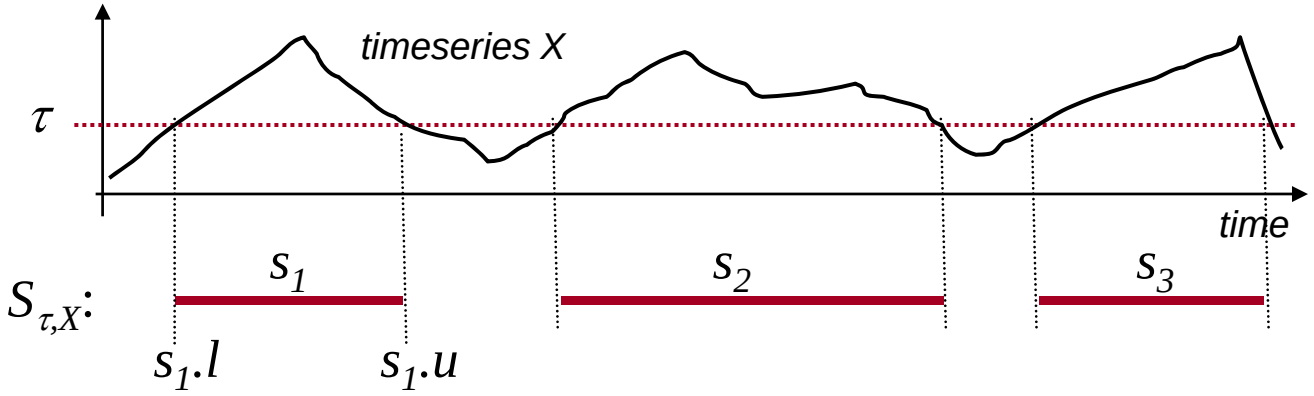
- Analyse von Umweltdaten
  - Untersuchung von Abhängigkeiten zwischen Ozon-Konzentration und anderen klimatischen und nicht-klimatischen Einflüssen.
- Analyse von Daten aus der Pharmaindustrie
  - Untersuchung der Blutwerte nach der Einnahme von bestimmten Medikamenten
- Analyse von Daten aus dem medizinischen Bereich
  - Erkennung von Anomalien in EKG-Kurven



# – Threshold-Basierte Ähnlichkeitssuche

- Prinzip:
  - Repräsentation einer Zeitreihe $X = \langle (x_i,t_i): i = 1..N \rangle$ durch eine Menge oder Sequenz von Zeitintervallen $S_{\tau,X} = \{s_j: j = 1..M\}$, für die gilt:

$$\forall t \in T : (\exists s_j \in S_{\tau,X} : s_j.l < t < s_j.u) \Leftrightarrow x(t) > \tau.$$

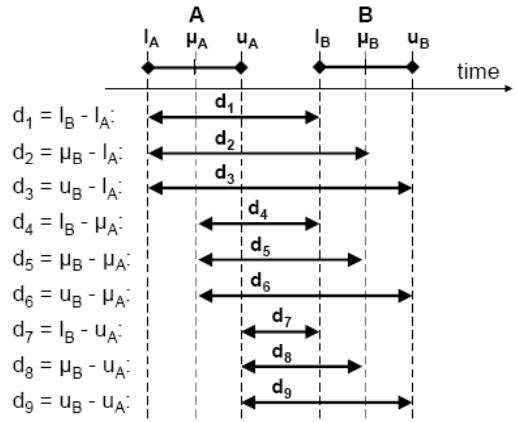



  - Die Ähnlichkeit zweier Zeitreihen wird über die Ähnlichkeit der entsprechenden Zeitintervalle bestimmt.


  - Frage: Wie ist die Ähnlichkeit von Zeit-intervall-Sequenzen definiert ?

    $S_{\tau,Q}$ ähnlicher zu $S_{\tau,A}$ oder zu $S_{\tau,B}$ ?

- Ähnlichkeit zwischen Zeit-Intervallen
  - Gesucht: Geeignete Distanzfunktion auf Intervalle die die intuitive Ähnlichkeit zweier Intervalle beschreibt.
  - Grundsätzlich: 9 Basis-Distanzen zwischen Intervalle




  - Problem: Intervalle sind durch zwei Attribute beschrieben, die Basis-Distanzfunktionen betrachten aber nur ein Attribut
    ➔ für die Ähnlichkeitsbestimmung sind die Basis-Distanzfunktionen nicht geeignet
    ➔ die Metrikeigenschaften sind nicht immer erfüllt, z.B. Dreiecksungleichung gilt bei $d_7$ nicht

  - Besser: Beide Parameter eines Intervals in das Ähnlichkeitsmass miteinbeziehen
    z.B.: **Euklidische Distanz**

- Ähnlichkeitsmaß für Zeit-Intervalle (Interval-Distance)
  - Seien $s_1=(l_1,u_1)$ und $s_2=(l_2,u_2)$ zwei Zeitintervalle. Die Ähnlichkeits-Distanz $d_{int}$ zwischen $s_1$ und $s_2$ ist folgendermaßen definiert:

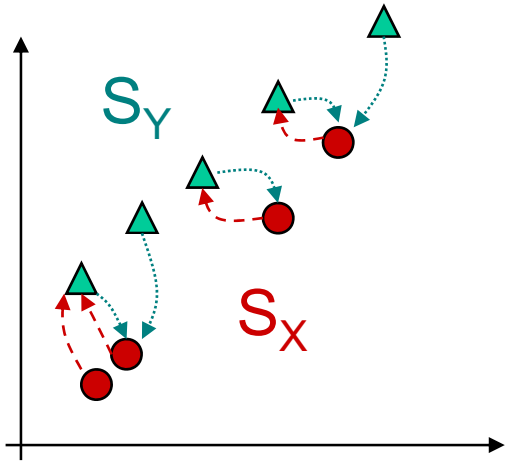
$$d_{\text{int}}(s_1, s_2) = \sqrt{(l_1 - l_2)^2 + (u_1 - u_2)^2}$$

- Ähnlichkeitsmaß für Mengen von Zeit-Intervallen (Threshold-Distance)
  - Vergleich mengenartiger Objekte

  ➔ wir verwenden die **Sum of Minimal Distance (SMD)** als Ähnlichkeitsmaß für unsere Mengen/Sequenzen von Zeit-Intervallen
  - Seien *X* und *Y* zwei Zeitreihen und $S_X$ und $S_Y$ die entsprechenden Intervall-Sequenz-Darstellungen von *X* und *Y* bzgl. eines bestimmten Thresholds $\tau$. Die Ähnlichkeits-Distanz $d_{TS}$ zwischen $S_X$ und $S_Y$ ist folgendermaßen definiert:

$$d_{TS}(S_X, S_Y) = \frac{1}{2} \cdot \left( \frac{1}{|S_X|} \cdot \sum_{s \in S_X} \min_{t \in S_Y} d_{\text{int}}(s,t) + \frac{1}{|S_Y|} \cdot \sum_{t \in S_Y} \min_{s \in S_X} d_{\text{int}}(t,s) \right)$$

- Beispiel für Threshold-Distance
  - Intervalle im folgenden durch Eck-Transformation als Punkte dargestellt:




$$d_{TS}(S_X, S_Y) = \frac{1}{2} \cdot \left( \underbrace{\frac{1}{|S_X|} \cdot \sum_{s \in S_X} \min_{t \in S_Y} d_{\text{int}}(s,t)}_{S_X \dashrightarrow S_Y} + \underbrace{\frac{1}{|S_Y|} \cdot \sum_{t \in S_Y} \min_{s \in S_X} d_{\text{int}}(t,s)}_{S_X \dashleftarrow S_Y} \right)$$

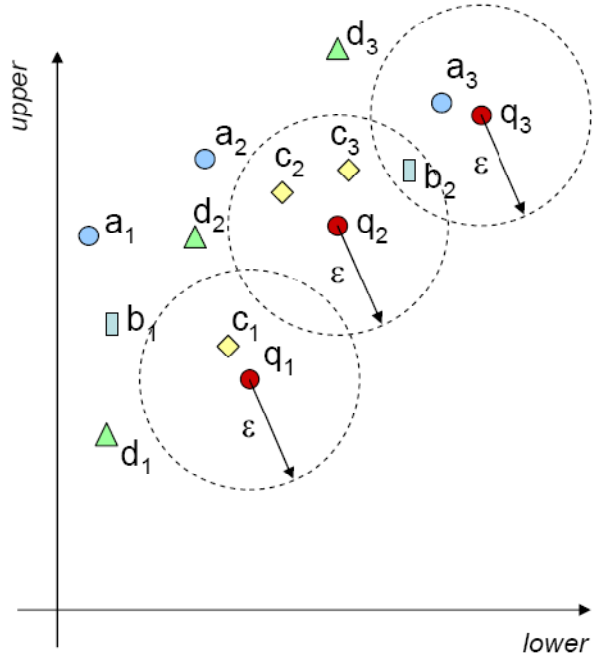
  - Intuitiv: Matche jedes Intervall aus $S_X$ mit dem am besten passenden Intervall aus $S_Y$, und umgekehrt, matche jedes Intervall aus $S_Y$ mit dem dazu am besten passenden Intervall aus $S_X$. Bilde jeweils die durchschnittliche matchingdistanz (Intervall-Distanz).
- Eigenschaften der Threshold-Distance
  - \+ Threshold-Distanz ist unabhängig von der Anzahl der betrachteten Zeitintervalle.
  - \+ Threshold-Distanz kann durch Räumliche Indexstrukturen (z.B. R-Baum) effizient unterstützt werden indem die einzelnen Zeit-Intervall-Instanzen indexiert werden.
  - – Threshold-Distanz (SMD) ist keine Metrik.

- Threshold Queries:
  - Anfragetypen:
    - Threshold-Basierte ε-Range-Anfrage $TQ_{\varepsilon}^{\varepsilon\text{-range}}$
    - Threshold-Basierte k-Nächste-Nachbar-Anfrage $TQ_{k}^{k\text{-NN}}$
  - Ziel:
    - Effiziente Unterstützung der Threshold-Queries da diese teuer zu berechnen sind
      ➔ Mittels Pruningstrategien möglichst die Kandidatenmenge einschränken.
  - Indexierung:
    - Verwaltung aller Zeit-Intervalle im R*-Baum
  - Unterstützung der Anfrage:
    - Unterstützung der Threshold-Query durch Verwendung von effizienten Methoden zur NN-Anfrage für die dementsprechenden Teilanfragen $\min_{t \in S_Y} d_{\text{int}}(q,t)$.
    - Konservative Schätzung der Threshold-Distance durch Betrachtung von möglichst wenig Objektinstanzen. Nicht alle Instanzen der Objekte müssen betrachtet werden ➔ (Filter)-Anfrage kann effizienter ausgeführt werden.

- Threshold-Basierte ε-Range-Anfrage $TQ_\varepsilon^{\varepsilon\text{-range}}$:

$$\forall X \in TQ_\varepsilon^{\varepsilon-range}(Q,\tau) : d_{TS}(S_{\tau,Q}, S_{\tau,X}) \leq \varepsilon$$

- Pruning-Strategie:
  - Alle Objekte *X* bei denen sich keine Zeit-Intervall-Instanz in eine der ε-range Umgebungen des Anfrageobjektes *Q* befindet, können keine Threshold-Distance kleiner gleich ε haben, d.h. $X \notin TQ_\varepsilon^{\varepsilon\text{-range}}(Q)$.



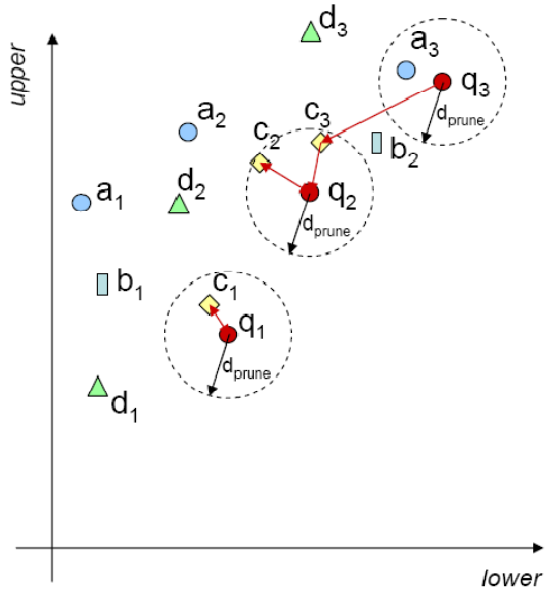

Für die Objekte A, B und C muss die Threshold-Distance (zumind. teilweise) berechnet werden.

Objekt D muss überhaupt nicht betrachtet werden.

- Threshold-Basierte k-Nächste-Nachbar-Anfrage $TQ_k^{k\text{-}NN}$:

$$\forall X \in TQ_k^{k-NN}(Q,\tau) : \forall Y \in D \setminus TQ_k^{k-NN}(Q,\tau) : d_{TS}(S_{\tau,Q}, S_{\tau,X}) < d_{TS}(S_{\tau,Q}, S_{\tau,Y}).$$

- Pruning-Strategie:
  - Sei $d_{prune}$ die Threshold-Distance zwischen den Objekten *Q* und *X*.
  - Alle Objekte *Y* bei denen sich keine Zeit-Intervall-Instanz in eine der $d_{prune}$-range Umgebungen des Anfrageobjektes *Q* befindet, können keine Threshold-Distance kleiner gleich $d_{prune}$ haben und sind somit nicht der NN von Q d.h. $Y \notin TQ_1^{k\text{-}range}(Q)$.




Objekte B und D müssen überhaupt nicht betrachtet werden.

# – Threshold-Basierte Indexierung

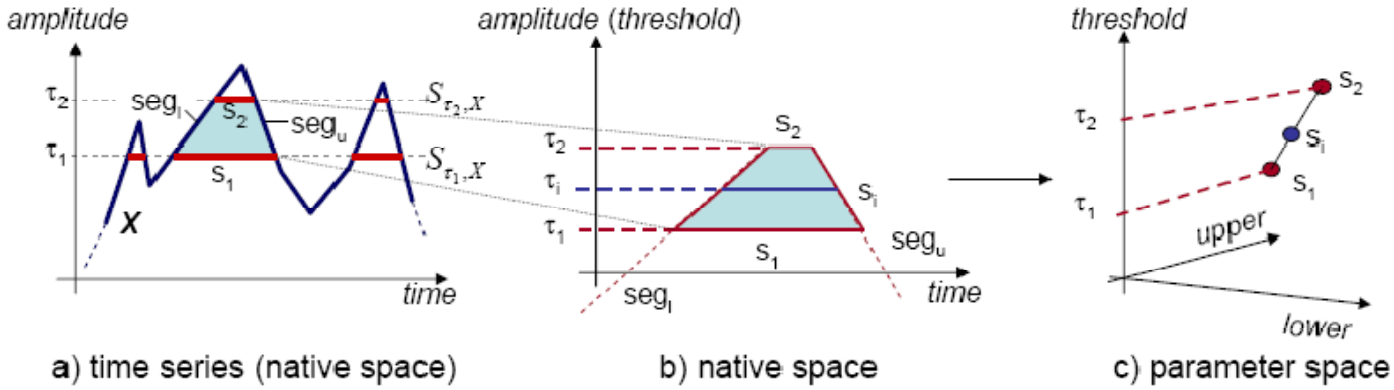
- Ziel:
  - Verwendung von Indexstrukturen zur effizienten Verwaltung der Zeitreihen-Objekte in einer Datenbank und zur Beschleunigung von Threshold-Queries
- Bisher:
  - Vorberechnung der Zeit-Intervall-Sequenzen für alle Objekte
  - Verwaltung der Zeit-Intervalle im R-Baum
  - R-Baum erlaubt effiziente NN-Suche

    ➔ Effiziente Unterstützung von Threshold-Queries

- Problem:
  - Threshold $\tau$ muss vorher (vor dem Einfügen eines Objektes in die Datenbank) bekannt sein.

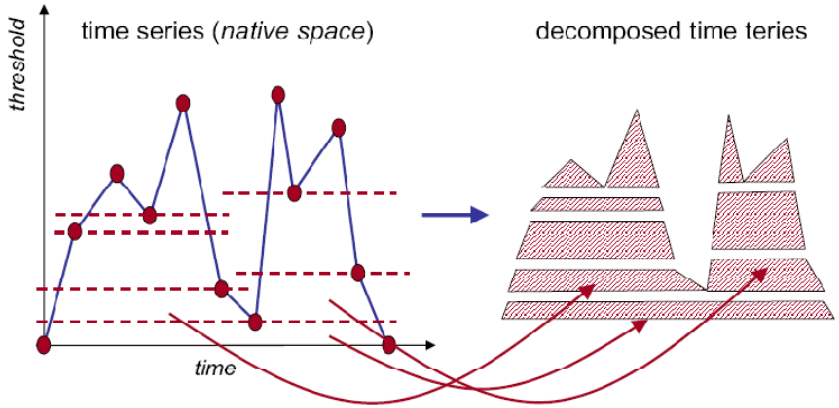
➔ Threshold-Invariante Verwaltung der Zeitreihen-Objekte

- Ansatz:
  - Berechne für ein Zeitreihen-Objekt die Zeit-Intervall-Sequenzen für alle möglichen Thresholds
  - Verwaltung aller Zeit-Intervall-Sequenzen in einer Indexstruktur
- Problem:
  - Die Anzahl der Zeit-Intervall-Sequenzen ist für ein Zeitreihen-Objekt unbestimmt
    ➔ „unendlich“ viele Zeit-Intervall Instanzen pro Objekt
    ➔ sehr hoher Speicherverbrauch
- Beobachtung:
  - Eine Menge von Zeit-Intervallen kann durch ein einziges Trapez repräsentiert werden.

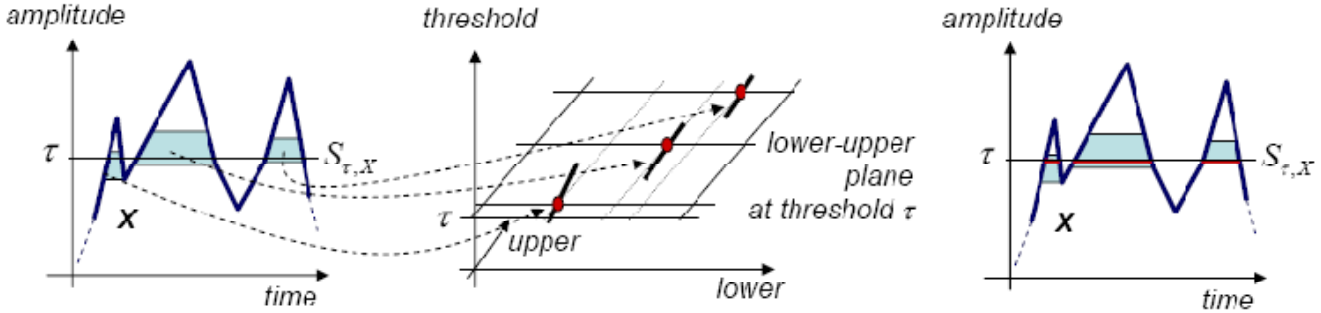



  - Für ein Zeitreihe der Länge N benötigt man höchstens N Trapeze für die Vollständige Repräsentation aller möglichen Zeit-Intervall-Sequenzen

- Idee:
  - Zerlege Zeitreihen vollständig in Trapeze




  - Effiziente Verwaltung der Trapeze in einer räumlichen Indexstruktur (R-Baum)
  - Berechnung der Zeit-Intervalle entsprechend eines bestimmten Thresholds durch einfache Schnittberechnung mit den jeweiligen Trapezen.



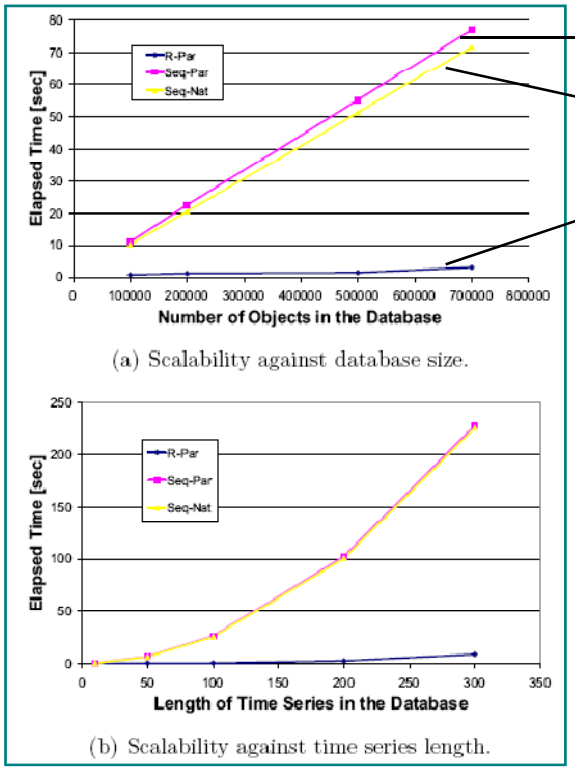

a) building time interval groups (native space)

b) determination of $S_{\tau,X}$ in parameter space

c) threshold-crossing time intervals (native space)

- Evaluation

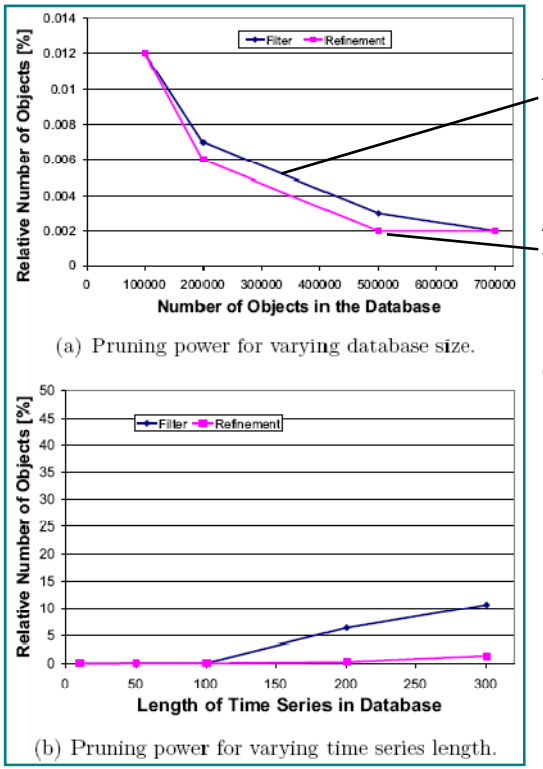
## Skalierbarkeit




sequentieller Scan
- auf Original-Daten (Seq-Par)
- auf zerlegte Zeitreihen (Seq-Par)

(R-Par)
Mit Index und Pruningstrategie

## Pruning-Power




Anzahl der im Filter-schritt zugegriffenen Kandidaten

Anzahl der verfeinerten Kandidaten

gemessen in % der Datenbankgrösse

- *R-Par* skaliert sehr gut auch bei grossen Datensätzen und sehr komplexen Objekten
- Distanzberechnung sehr teuer für grosse Zeitreihen
- Beobachtung: grössere Zeitreihen verzögern das Pruning