



Why are Dendrograms Useful?

If someone tells us they have a new similarity measure for DNA, and it produces an *intuitive* dendrogram...

... but if their new similarity measure gives us a very *unintuitive* dendrogram, we should view it with suspicion...

Piecewise Linear Approximation I

Basic Idea: Represent the time series as a sequence of straight lines.

Lines could be **connected**, in which case we are allowed $N/2$ lines

If lines are **disconnected**, we are allowed only $N/3$ lines

Personal experience on dozens of datasets suggest disconnected is better. Also only disconnected allows a lower bounding Euclidean approximation

Each line segment has

- 1 length
- left_hasRight (or right_hasLeft can be inferred by looking at the next segment)

Each line segment has

- 1 length
- left_hasRight
- right_hasLeft

Karl Friedrich Gauss
1777 - 1855

Indexování a dobývání znalostí z časových řad (*time series*)

Eamonn Keogh

eamonn@cs.ucr.edu

Defining Distance Measures

Definition: Let O_1 and O_2 be two objects from the universe of possible objects. The distance (dissimilarity) is denoted by $D(O_1, O_2)$

What properties are desirable in a distance measure?

- $D(A,B) = D(B,A)$ Symmetry
- $D(A,A) = 0$ Constancy
- $D(A,B) = 0$ iff $A=B$ Positivity
- $D(A,B) \leq D(A,C) + D(B,C)$ Triangular Inequality



Lowland Gorilla
Gorilla gorilla gorilla

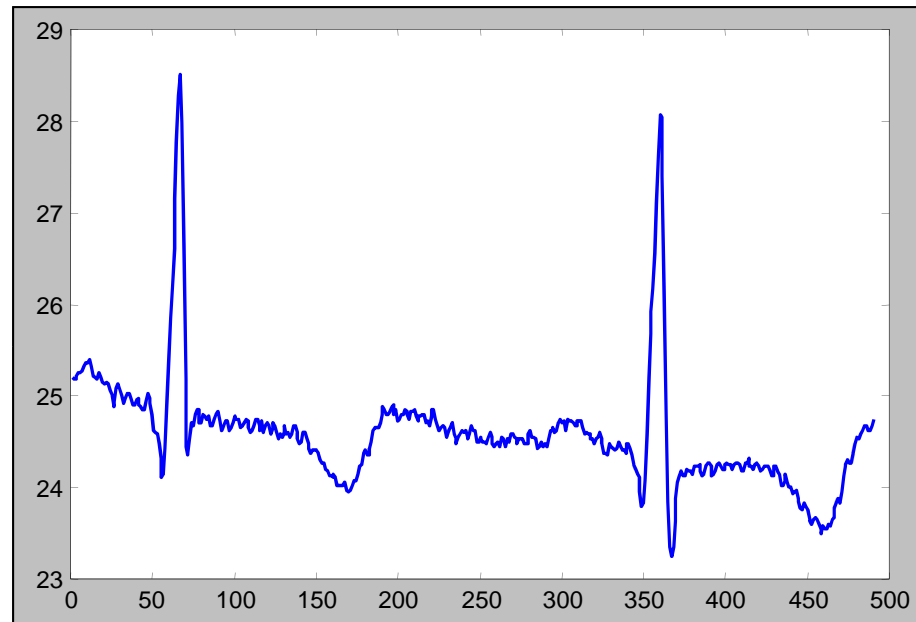
DTW is needed for most natural objects...

Mountain Gorilla
Gorilla gorilla beringei

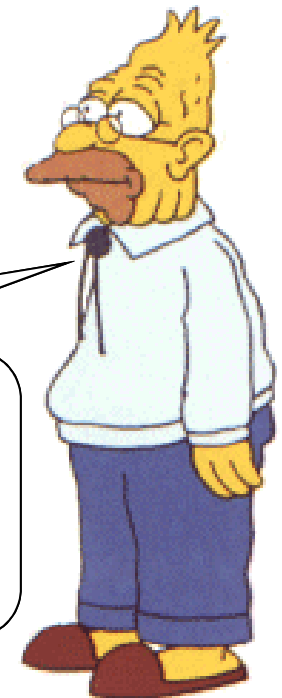
25.1750
25.2250
25.2500
25.2500
25.2750
25.3250
25.3500
25.3500
25.4000
25.4000
25.3250
25.2250
25.2000
25.1750
..
..
24.6250
24.6750
24.6750
24.6250
24.6250
24.6250
24.6750
24.7500

Co jsou časové řady?

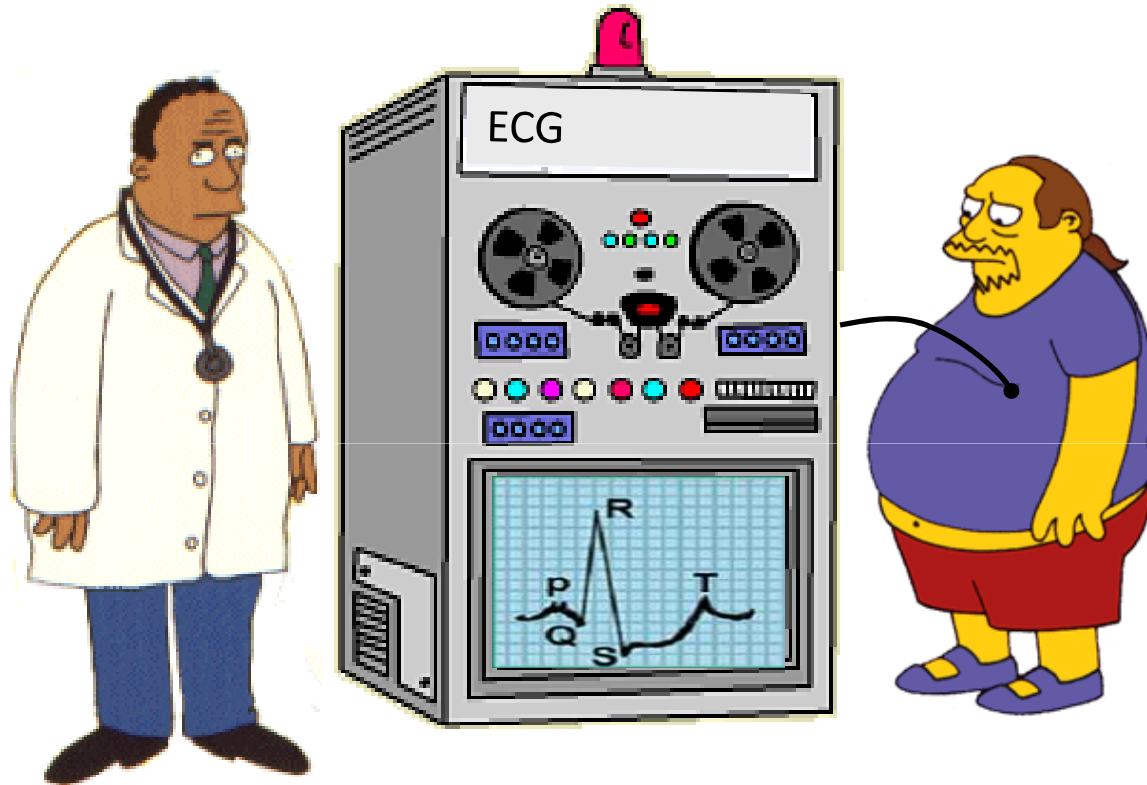
Časové řady jsou posloupnosti dat (vektorů dat), které jsou výsledkem pozorování/měření uskutečněných postupně v čase.



Většinu technik (míry podobnosti, indexování, postupy pro redukci dimenzionality), se kterými se setkáme v této přednášce, lze použít i pro další typy dat.



Motivační příklad ...



Pacient si stěžuje na bolest na prsou. Jeho EKG vypadá neobvykle ...

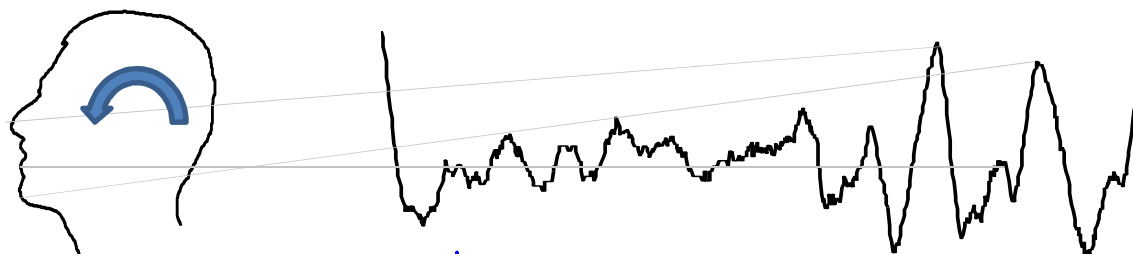
Lékař by rád našel v archivu nějaký podobný případ, jehož EKG vypadá **podobně**. Doufá totiž, že to by ho mohlo varovat před možnou chybou, inspirovat při návrhu řešení ...

2 otázky:

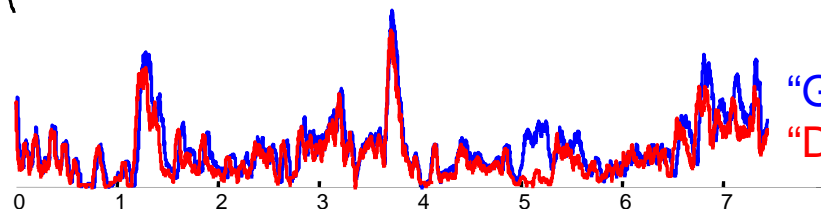
- Jak poznat podobné křivky? – definice míry vzdálenosti
- Jak realizovat vyhledávání RYCHLE?

Jaké další typy dat lze zpracovávat podobně?

a) Obrázky,
video, ...



b) Texty –
počet
výskytů slov
na stránkách

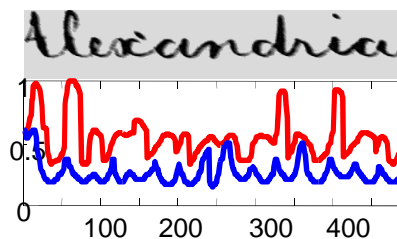


“God” – Bible v angličtině
“Dios” – Bible ve španělštině



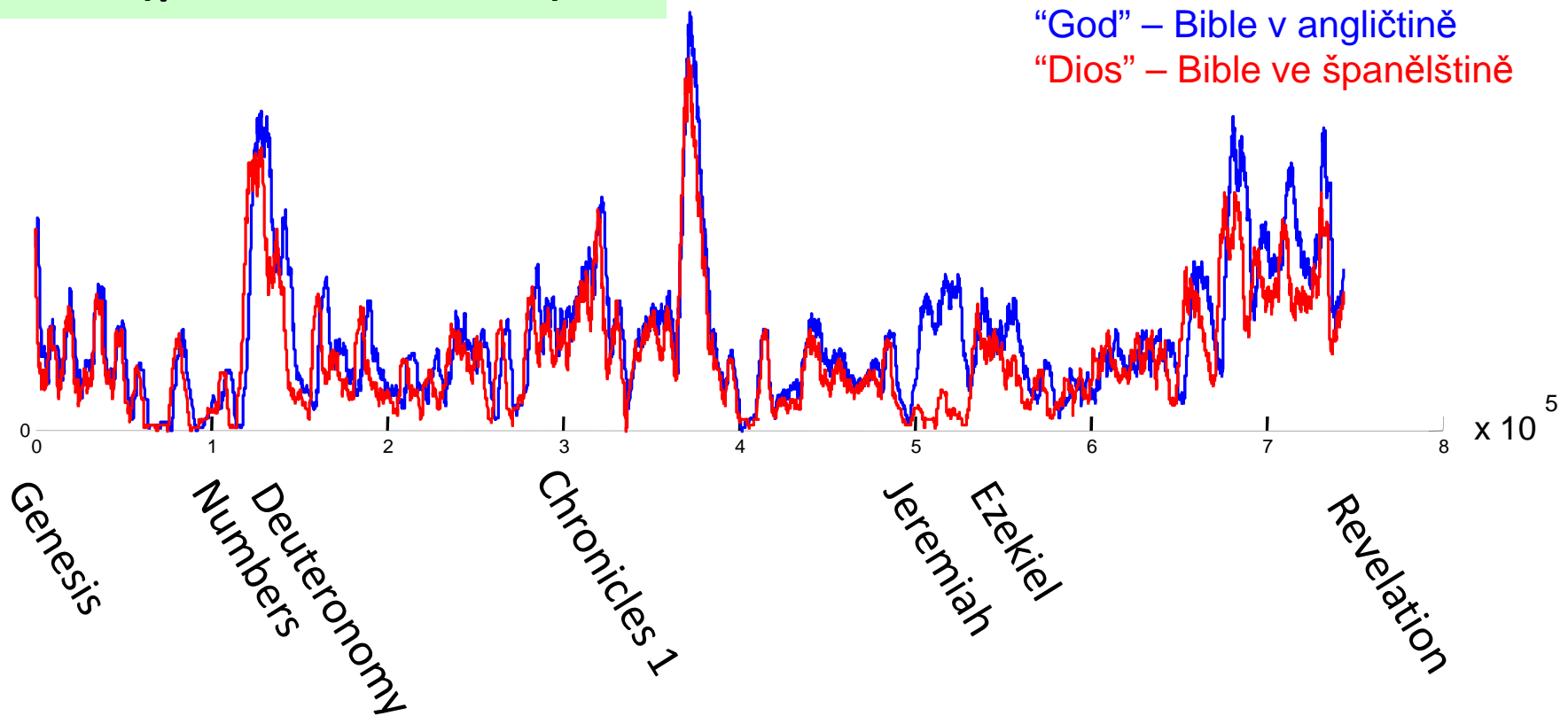
“El Señor”- Bible
ve španělštině

c) rukopis
(roztaženo v
čase)



Textová data jako časové řady...

Lokální frekvence slov (po stránkách)

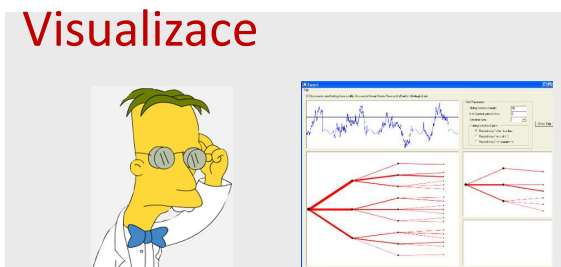
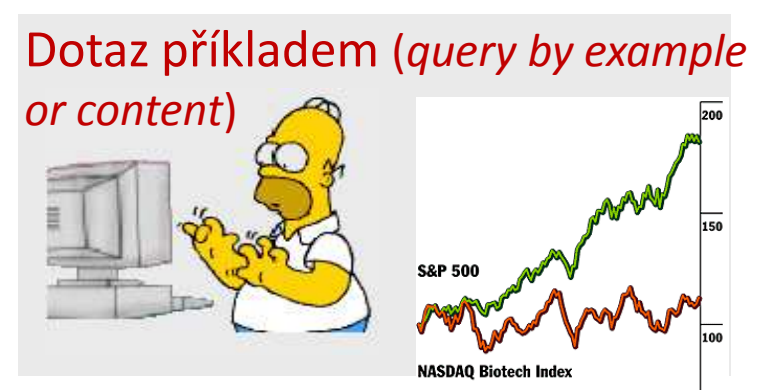
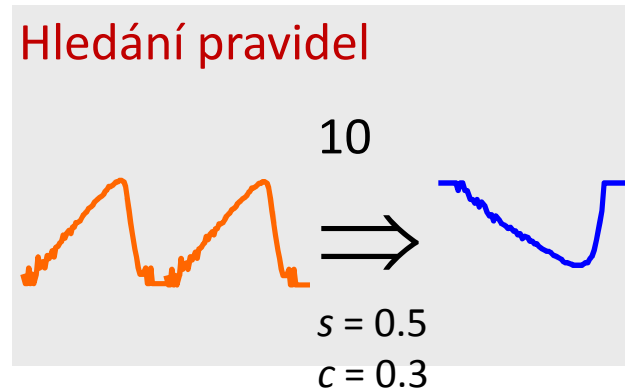
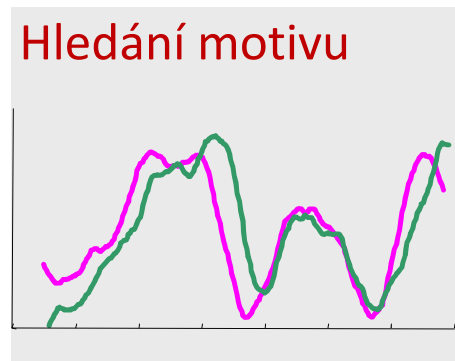
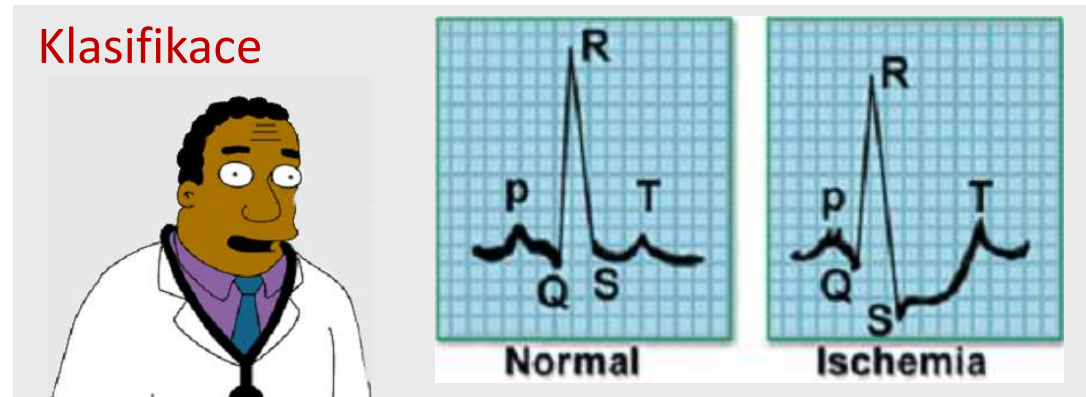
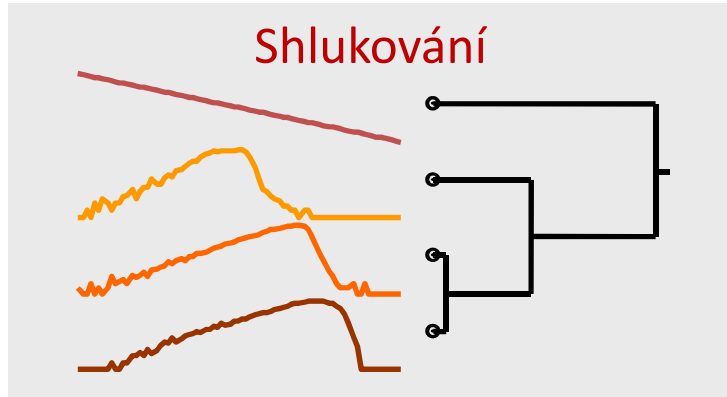


“El Senor” – Bible ve španělštině



Jaké **typy úloh** jsou pro časové řady důležité?

Ve všech potřebujeme charakterizaci *podobnosti (Similarity)*



Co je těžkého na práci s časovými řadami?

Část I

1. Velký objem dat → vysoké požadavky na efektivní reprezentaci i implementaci používaných algoritmů (např. tak, aby se minimalizovaly přístupy na externí paměť).

- 1 hodina dat z EKG: 1 Gigabyte.
- Typický Weblog: 5 Gigabytes/týden.
- Space Shuttle Database: 200 Gigabytes

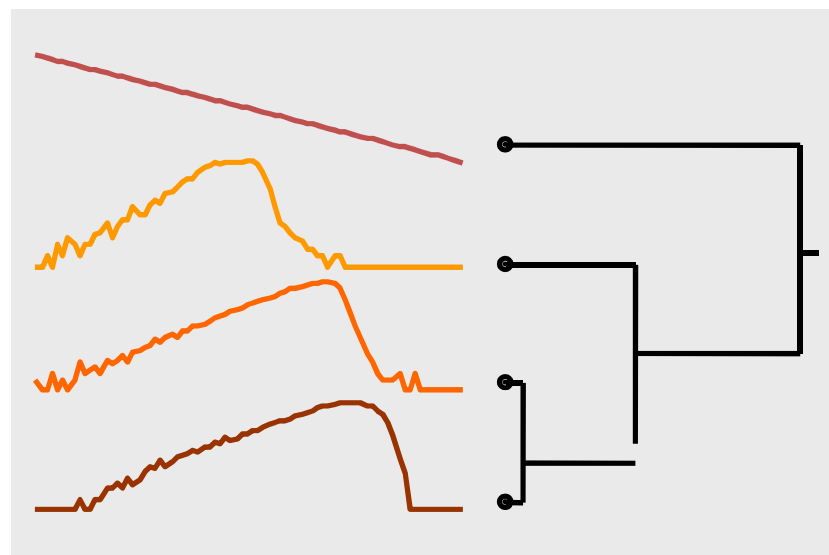
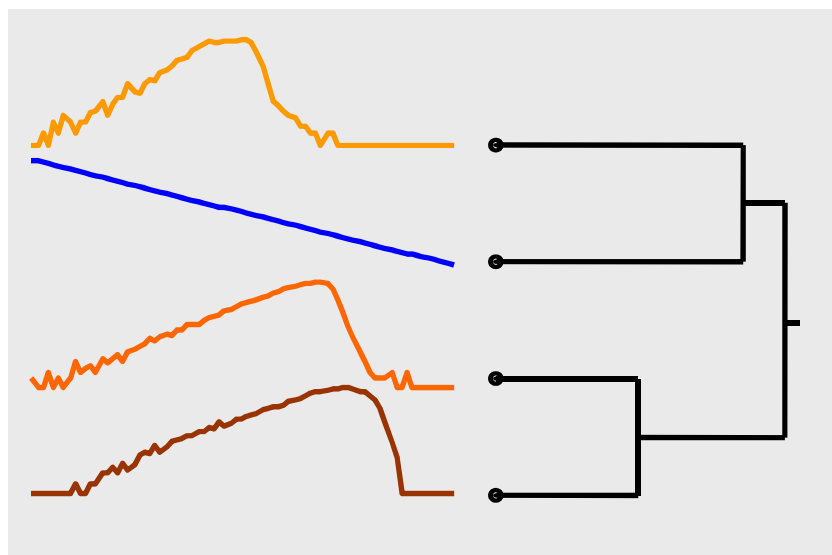
2. Problémy spojené se slučováním dat z více zdrojů (*merging data from various sources*):

- Různé formáty.
- Rozdílné vzorkovací frekvence.
- Šum, chybějící hodnoty,

Co je těžkého na práci s časovými řadami?

Část II

Odpověď: Hodnocení podobnosti je subjektivní!



Hodnocení podobnosti závisí na uživateli, na zdroji dat i na řešené úloze. Tyto aspekty je nutné respektovat při definici odpovídající míry podobnosti.

Definice míry vzdálenosti (*Distance Measures*)

Definice: Necht' O_1 a O_2 jsou dva objekty z množiny všech možných objektů (univerzum). Jejich vzájemná vzdálenost (nepodobnost - *dissimilarity*) se označuje $D(O_1, O_2)$

Jaké vlastnosti by měla mít míra vzdálenosti (nepodobnosti)? Má být

- $D(A, B) = D(B, A)$ *symetrická*
- $D(A, A) = 0$ *konstantní*
- $D(A, B) = 0$ iff $A = B$ *kladná*
- $D(A, B) \leq D(A, C) + D(B, C)$ *splňovat trojúhelníkovou nerovnost*



Proč požadujeme trojúhelníkovou nerovnost?

Požadují ji prakticky všechny metody na indexování dat.

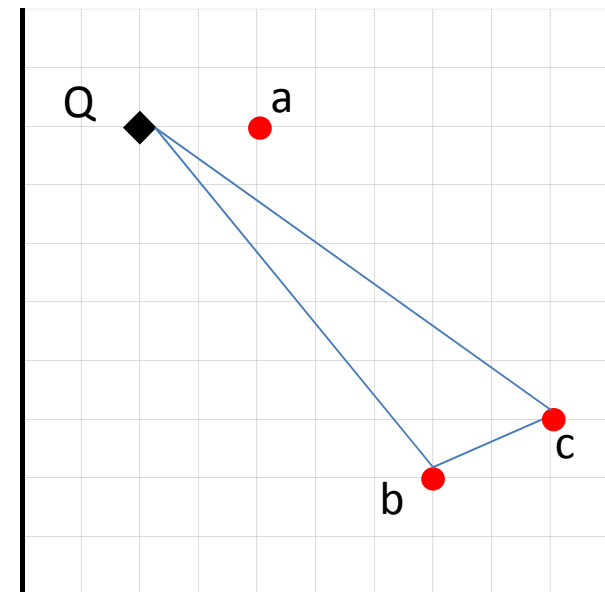
Důvod? Předpokládejme, že k danému bodu **Q** máme vybrat mezi 3 body **a**, **b** a **c** ten, který je od **Q** nejméně vzdálen.

Nejdřív zjistíme, že bod **a** je od **Q** vzdálen **2** jednotky: **a** se stane *zatím-nejlepší*. Pro další bod **b** vypočteme vzdálenost od **Q** **7.81** jd. Teď už nemusíme zjišťovat vzdálenost mezi **Q** a **c** – stačí využít trojúhelníkovou nerovnost !

Víme, že

$$D(Q,b) \leq D(Q,c) + D(b,c)$$
$$D(Q,b) - D(b,c) \leq D(Q,c)$$
$$7.81 - 2.30 \leq D(Q,c)$$
$$5.51 \leq D(Q,c)$$

Dolní odhad pro vzdálenost mezi **Q** a **c** je 5.51 jd. Není tedy lepší než náš *zatím-nejlepší* bod **a**.



	a	b	c
a		6.70	7.07
b			2.30
c			

Eucleidova míra vzdálenosti

Je pro dvě synchronně měřené časové řady

$$Q = q_1 \dots q_n$$

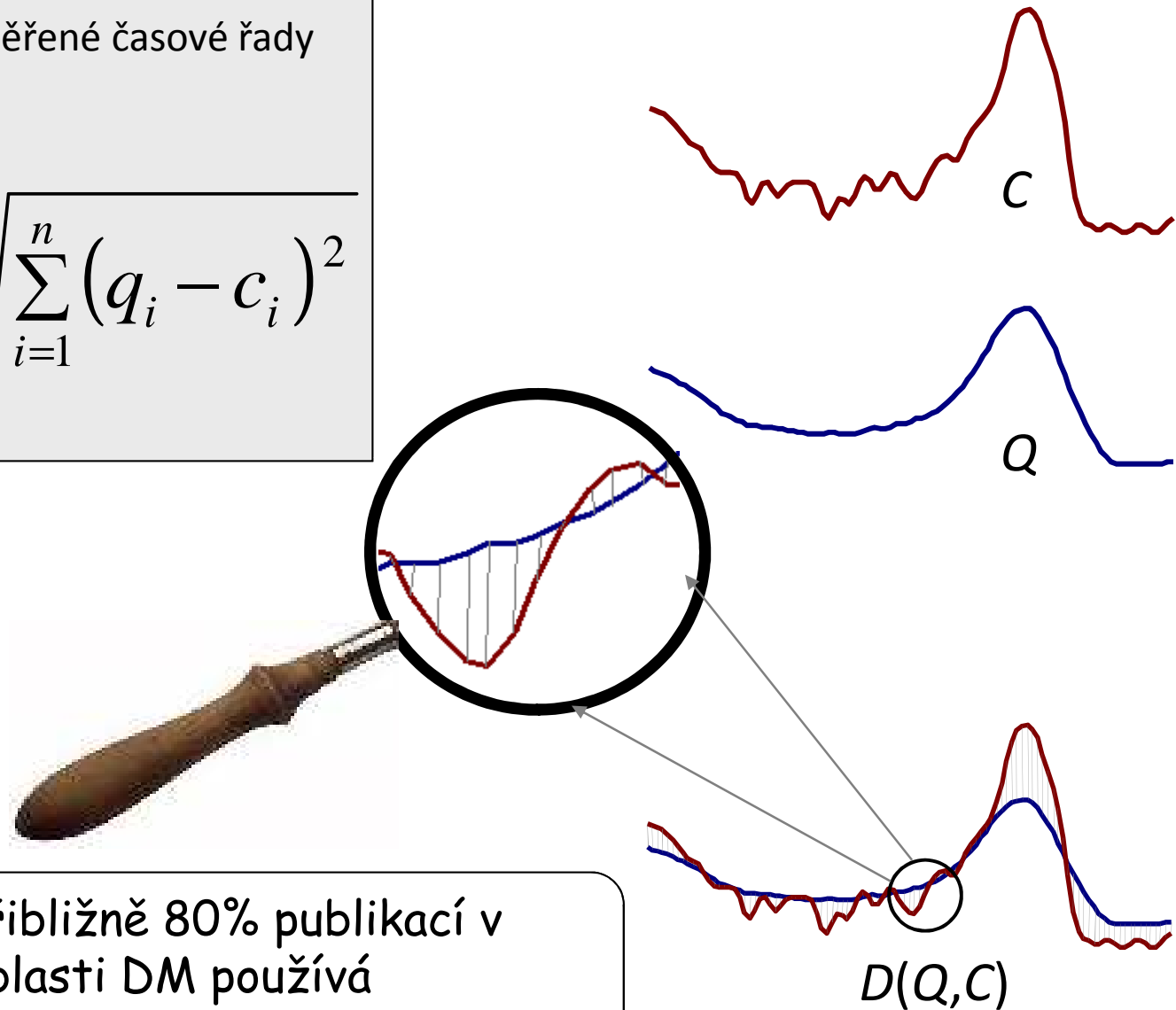
$$C = c_1 \dots c_n$$

definována takto:

$$D(Q, C) \equiv \sqrt{\sum_{i=1}^n (q_i - c_i)^2}$$



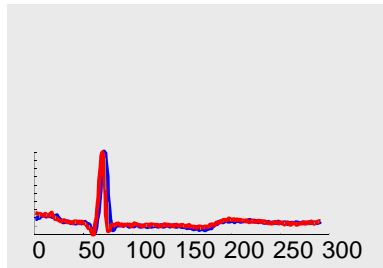
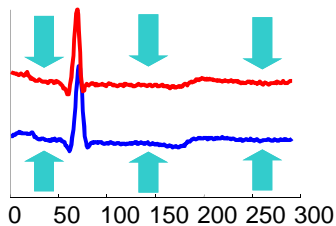
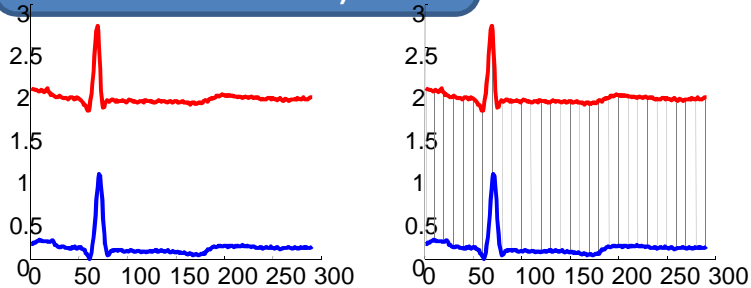
Přibližně 80% publikací v oblasti DM používá Eucleidovu vzdálenost



Předzpracování pomocí lineární transformace

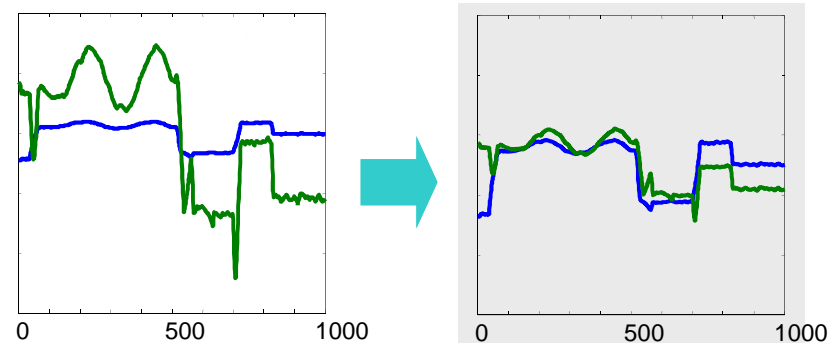
T_I: vyvážení (Offset Translation)

$$Q = Q - \text{průměr}(Q)$$



T_{II}: úprava amplitud (Amplitude Scaling)

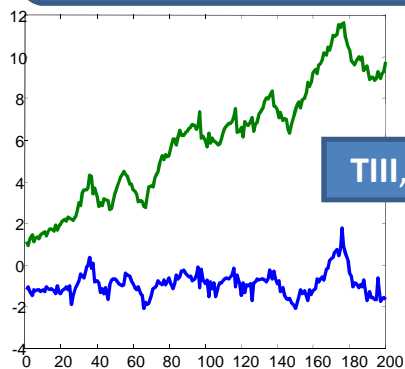
$$Q = (Q - \text{průměr}(Q)) / \text{st_odchylka}(Q)$$



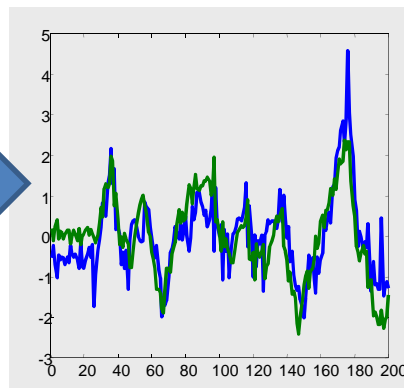
T_{IV}: odstranění šumu

T_{III}: odstranění lin. trendu

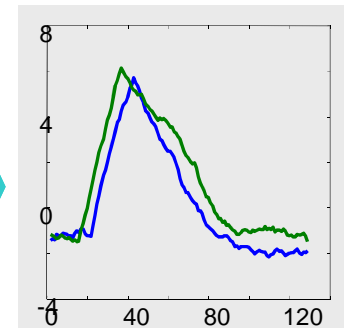
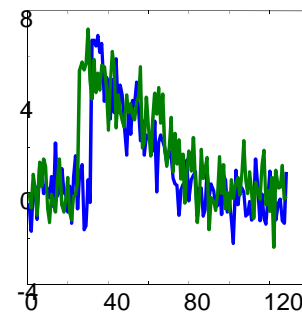
Původní signál **s1** je proložen přímkou **l1** a výsledným signálem se stane jejich rozdíl (**s1 - l1**).



T_{III}, T_I a T_{II}

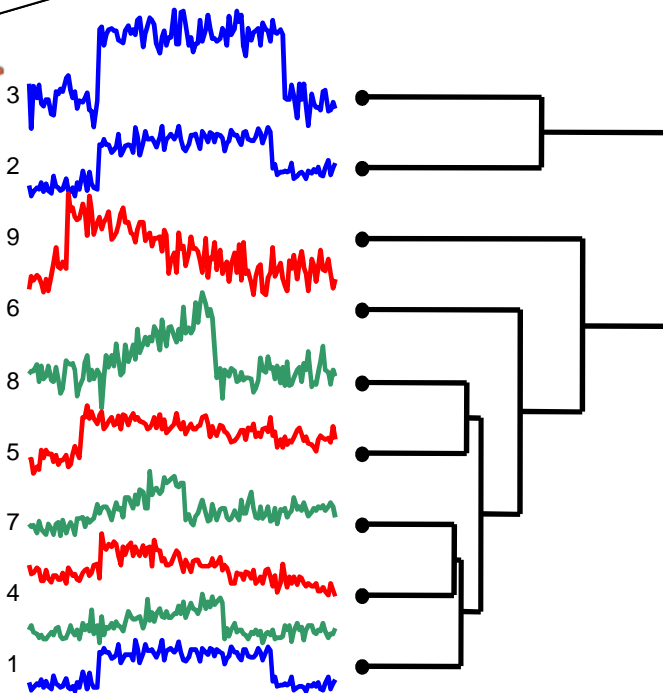


Vyhlazení (smoothing): Každá hodnota je nahrazena průměrem okolních hodnot.

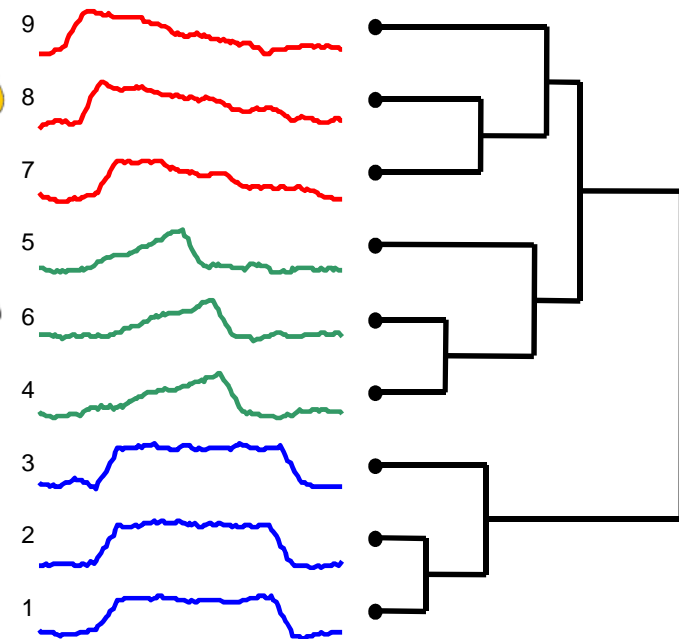


Rychlý experiment demonstrující užitečnost předzpracování

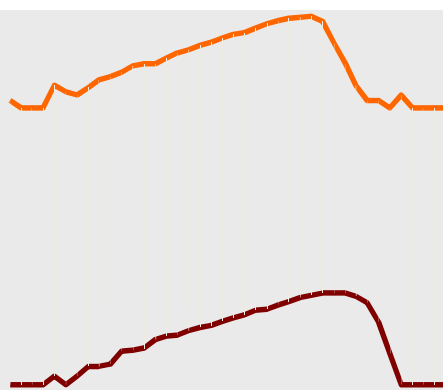
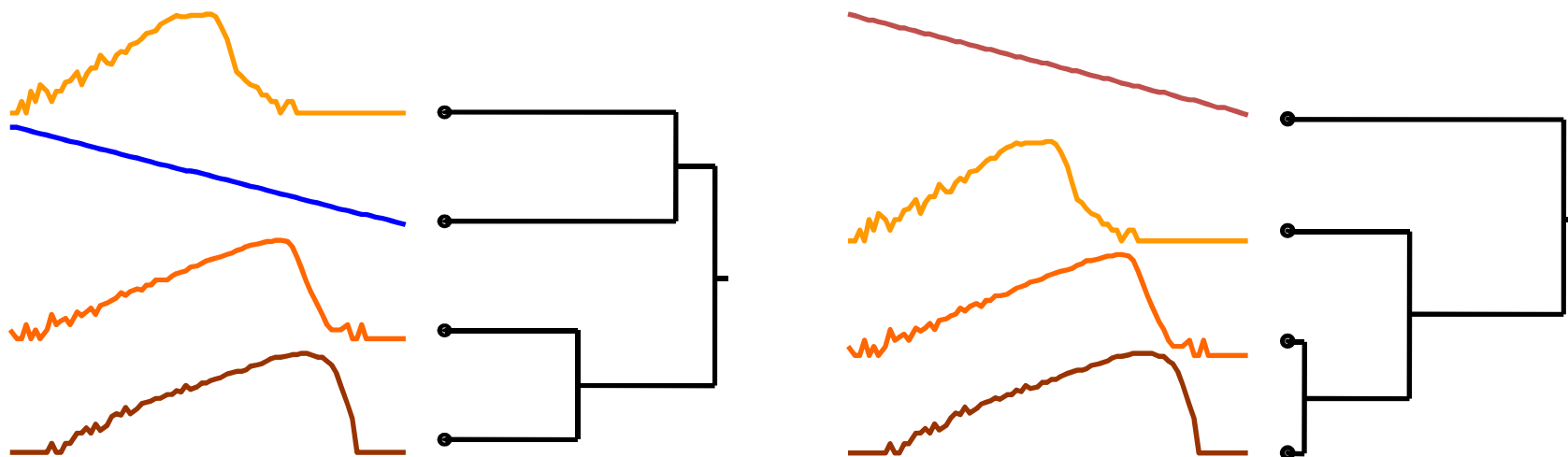
Výsledek shlukování původních dat s použitím Eukleidovy vzdálenosti.



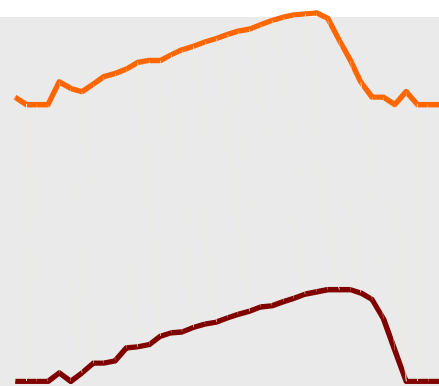
Shlukování používající Eucl. vzdálenost pro předzpracovaná data (odstranění šumu, lineárních trendů, vyvážení a úprava amplitud).



Dynamické borcení času (*Dynamic Time Warping: DTW*)



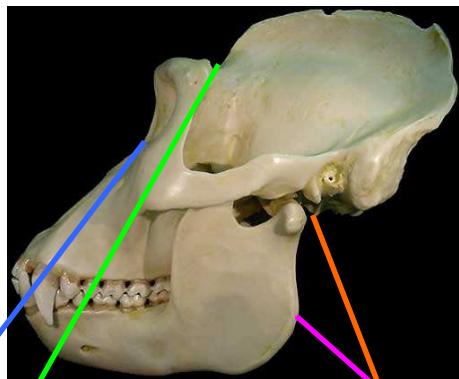
Pevná časová osa (*Fixed Time Axis*)
Předpokládá, že posloupnosti jsou srovnávány „bod po bodu“ ve stejných intervalech.



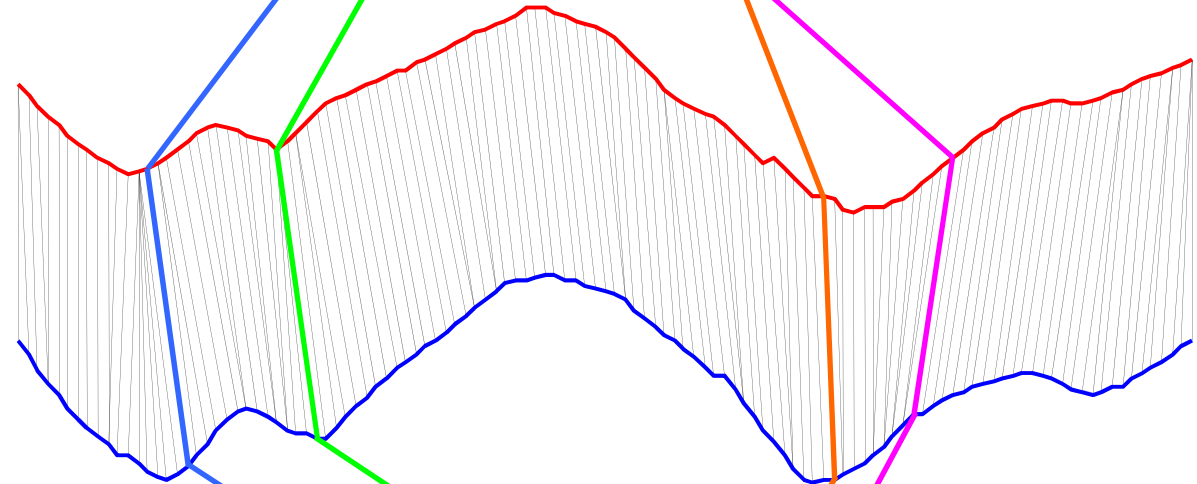
“Zborcená” časová osa
Umožňuje „nesynchronizované“ plynutí času, tj. nelineární skoky při srovnávání bodů dvou signálů.

Poznámka: Nejdřív ukážeme pár příkladů, pak se vrátíme k metodě výpočtu.

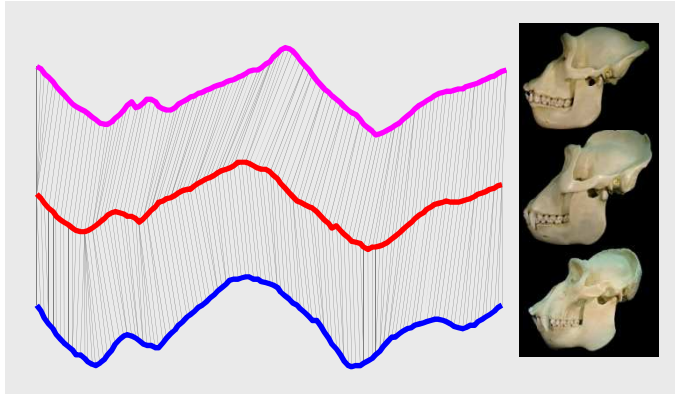
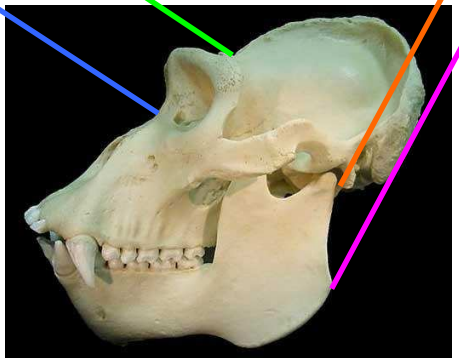
Gorila nížinná
Gorilla gorilla gorilla



DTW je vhodné
pro reálná
přirozená data...

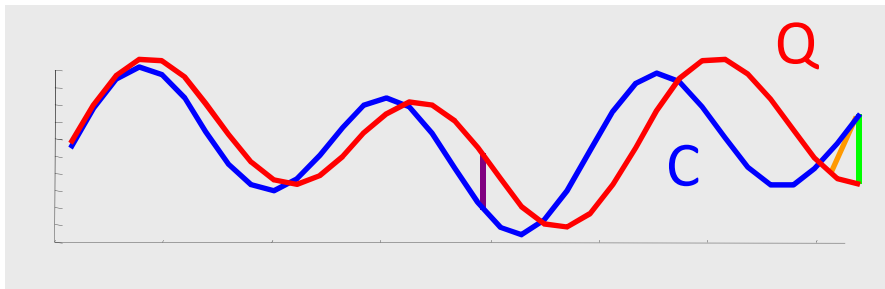


Gorila horská
Gorilla gorilla beringei



Postup výpočtu DTW I

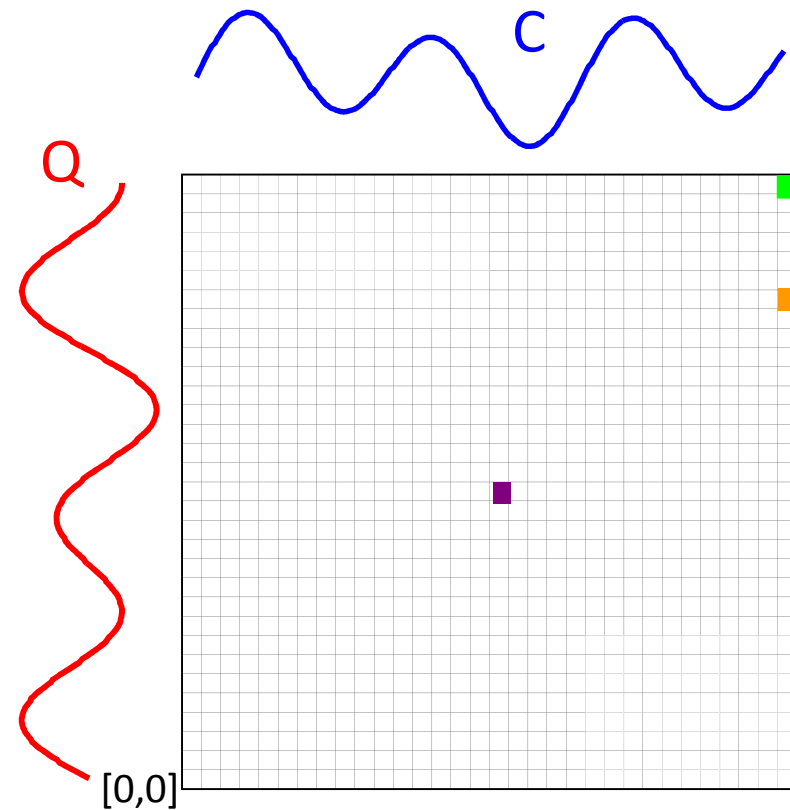
Vytvoříme matici $|Q|$ krát $|C|$,
všech hodnot vzdáleností $d(i,j)$
mezi indexovanými body obou
křivek a hledáme “nejlevnější”
cestu z $[0,0]$ do $[m,n]$.



Jaký je odhad složitosti výpočtu (počet
možných cest WPs) v matici $m \times n$?

Označme $\max(m,n) = n$.

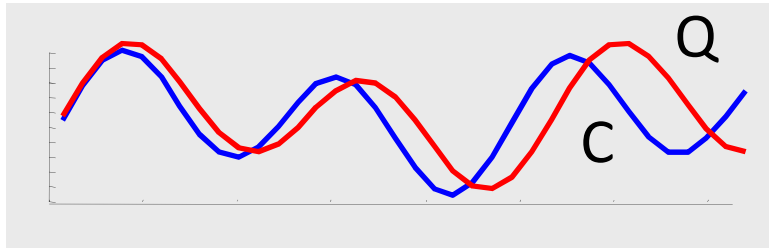
Velmi hrubý horní odhad je 3^n
Daleko přesnější je odhad $O(n^2)$, viz
rekurzivní rovnice na dalším listu



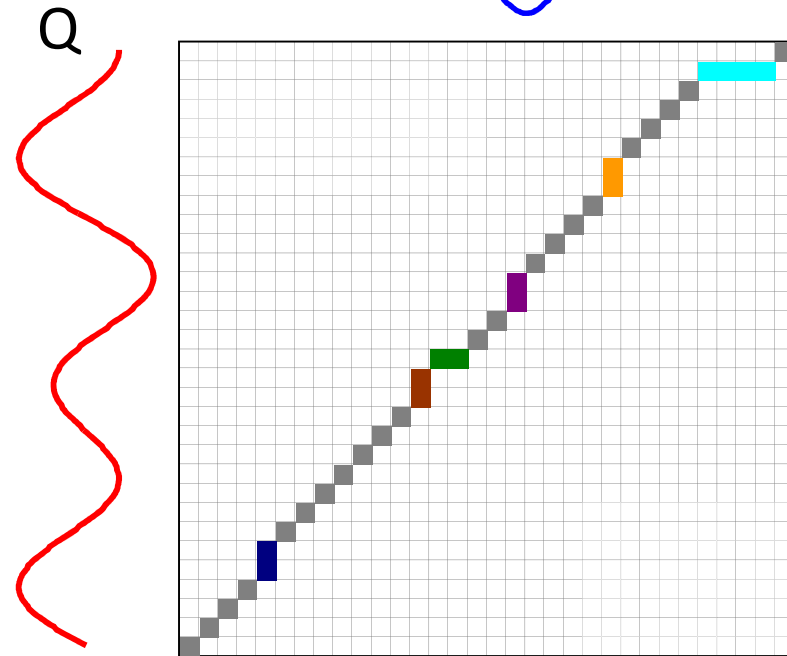
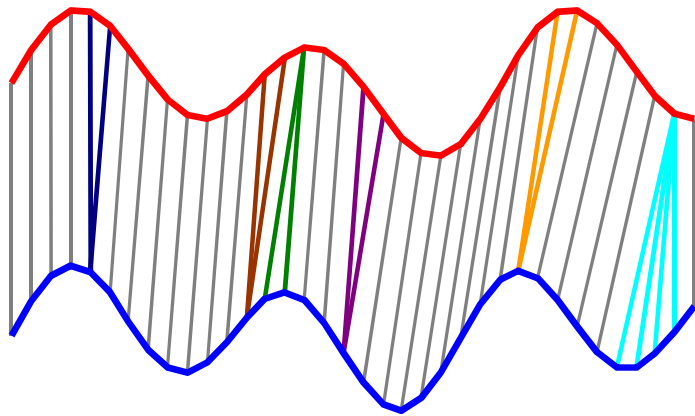
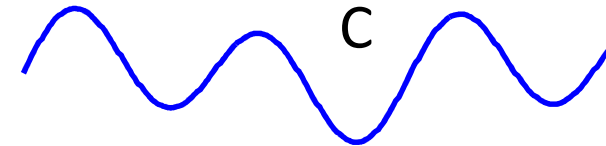
Postup výpočtu DTW II.

Nechť w je DTW cesta w_1, w_2, \dots, w_K tvořená body matice. Hledáme takovou posloupnost DTW, která bude celkovou vzdálenost přiřazených bodů

$$\text{minimalizovat } \dots \text{DTW}(Q, C) = \min \left\{ \sqrt{\sum_{k=1}^K w_k} / K \right\}$$



Označme $\gamma(i, j)$ nejmenší vzdálenost z bodu (i, j) do „konce“, pak řešení lze charakterizovat



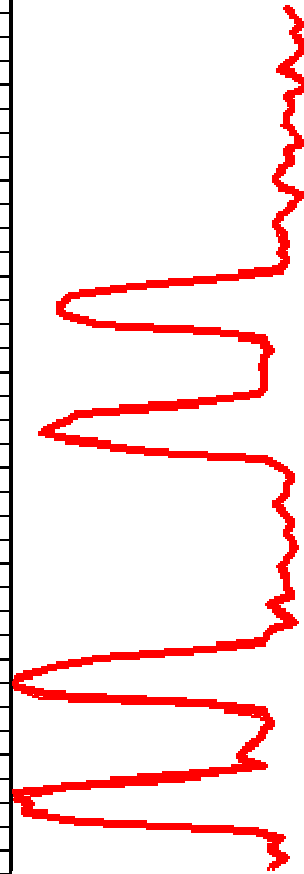
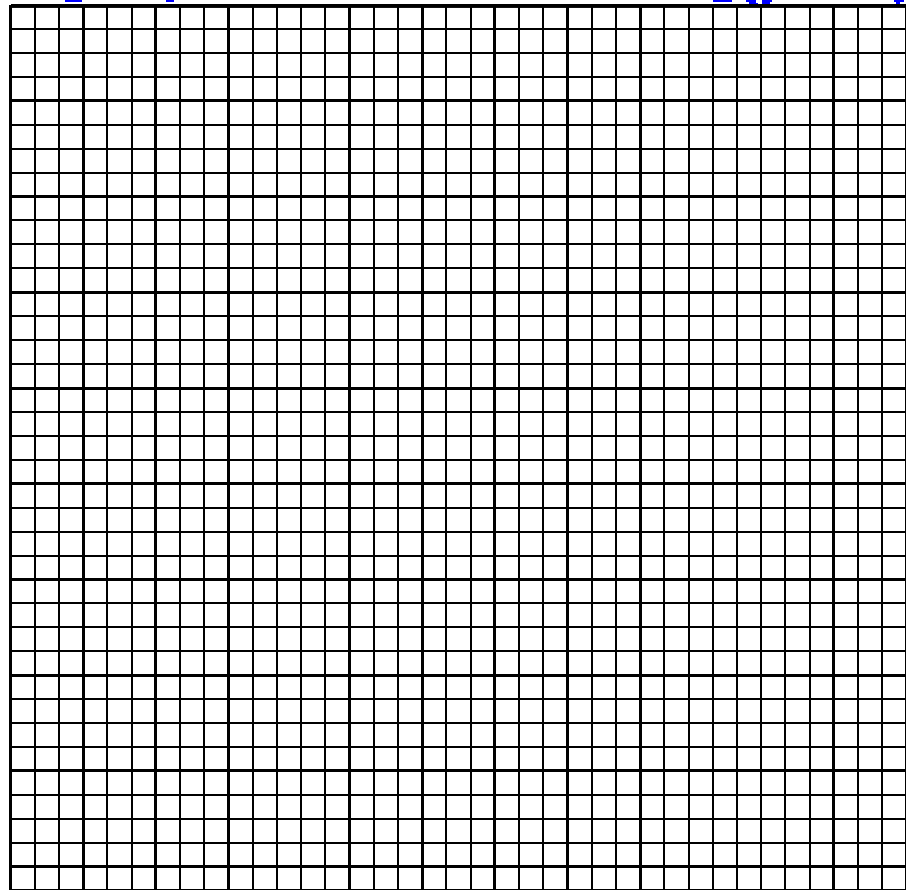
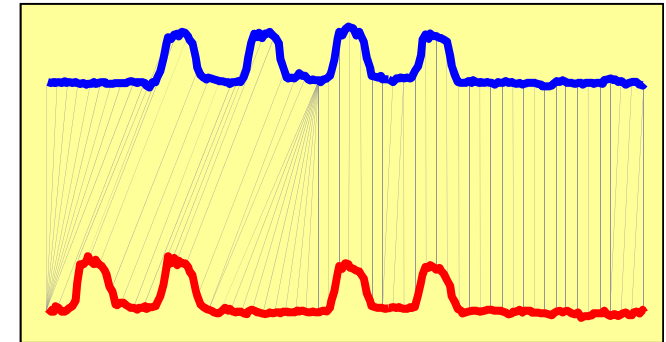
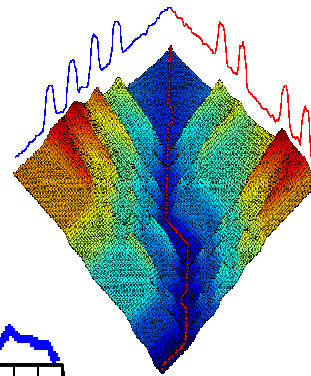
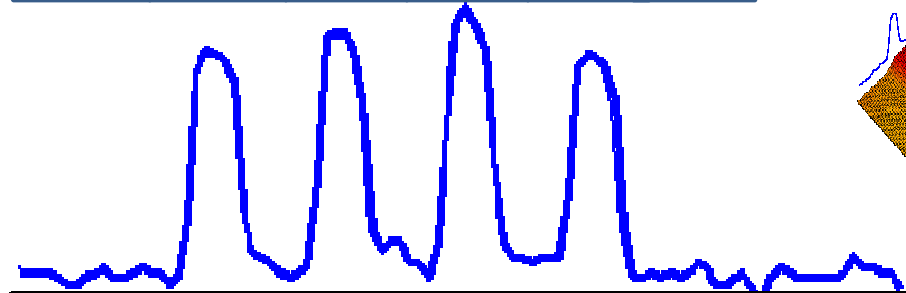
rekursivním vztahem (lze jej použít pro výpočet pomocí dynamického prog.)

$$\gamma(i, j) = d(q_i, c_j) + \min \{ \gamma(i+1, j+1), \gamma(i+1, j), \gamma(i, j+1) \}$$

W: cesta DTW

Podívejme se na data z reálné úlohy

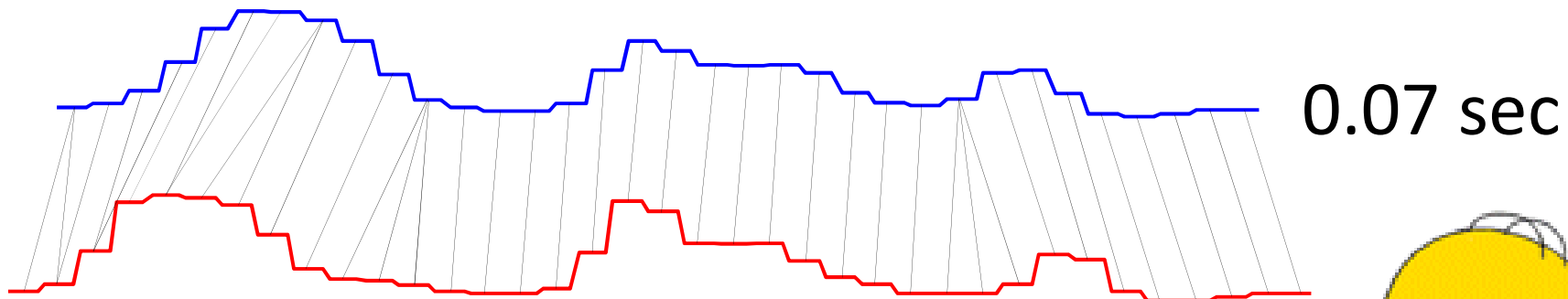
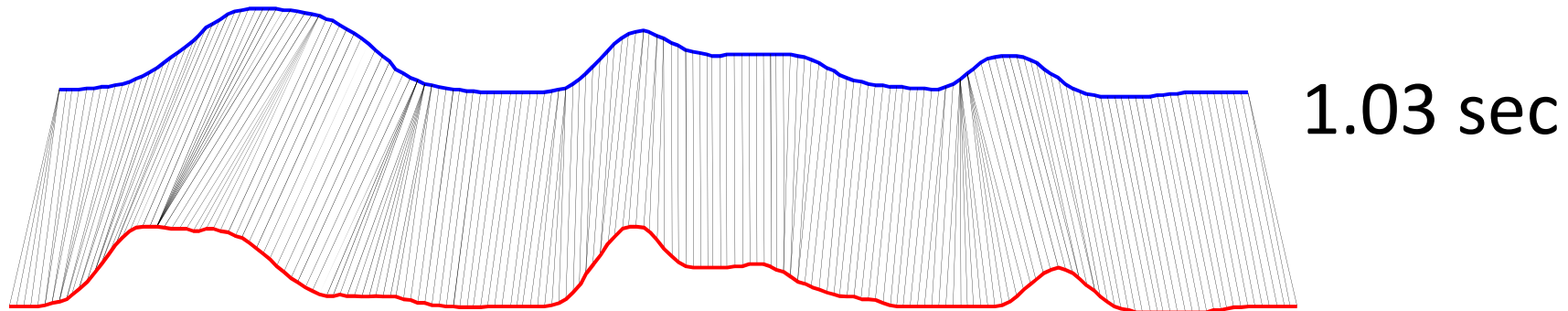
Po Út St Čt Pá So



Data o odběrech energie během 2 period délky jeden týden.

Povšimněme si, že oba týdny mají jen 4 pracovní dny: **modrá** posloupnost začíná svátkem (pondělí), **červená** má svátek ve středu.

$O(n^2)$ může být dost veliké. Nemohl by se výpočet DTW urychlit pomocí **Rychlých Aproximací**?



... Použití aproximací se na první pohled zdá být rozumné.

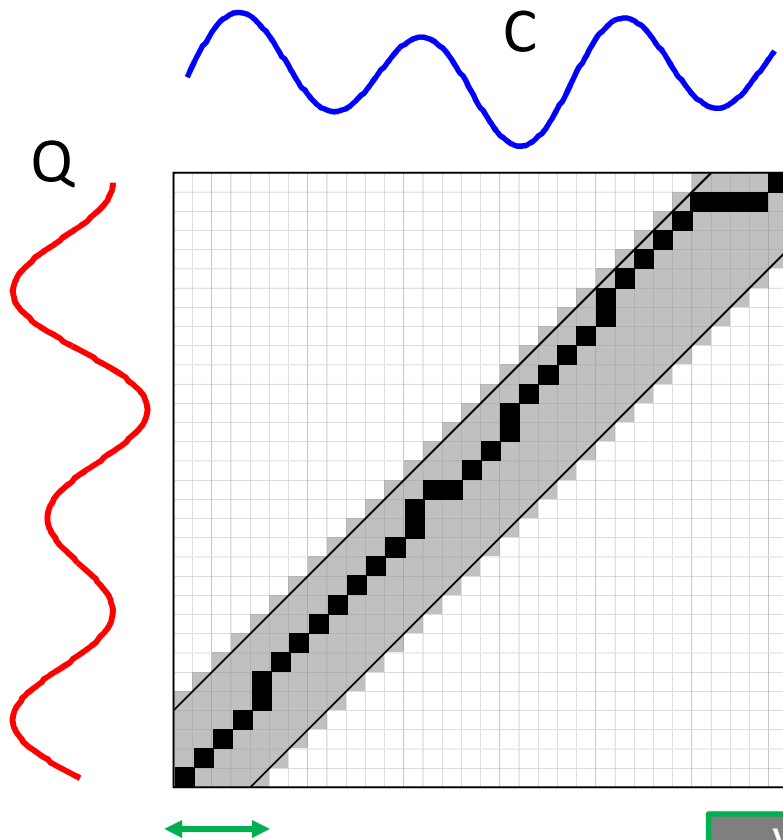
Řada experimentů potvrzuje, že tento postup dává dobré výsledky při shlukování, klasifikaci, atd.



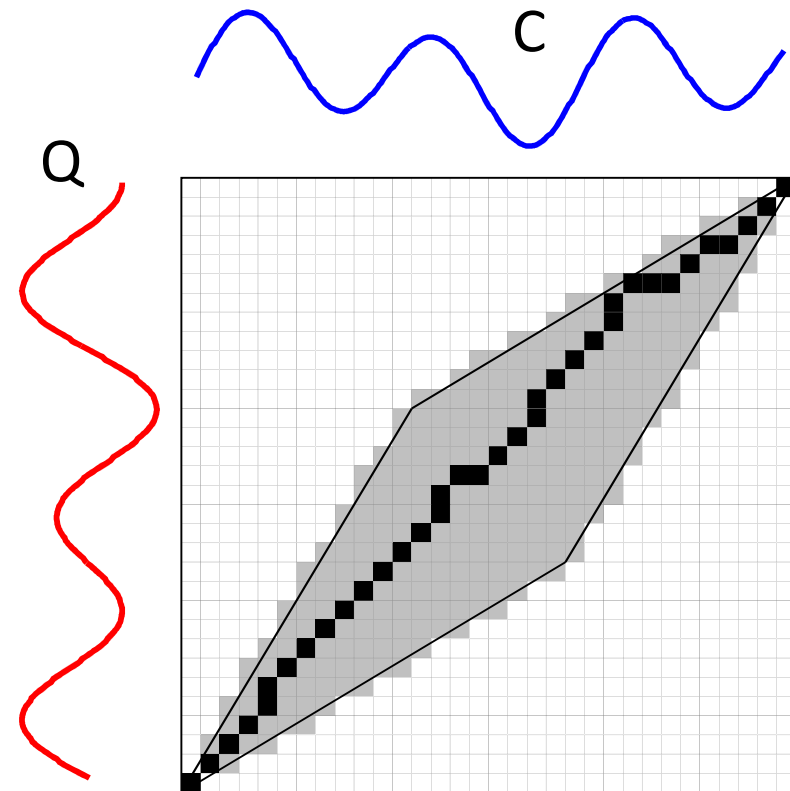
Další urychlení: Globální omezující podmínky

- umožňují mírně zrychlit výpočet
- zabraňují vzniku „patologických“ řešení

Sakoe-Chibaův pruh



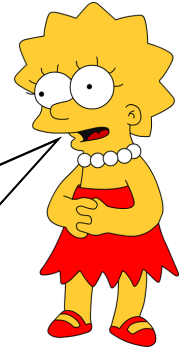
Itakurůva Paralelogram



Velikost W-okna
(*Warping Window*)

Jak lze urychlit nalezení nejméně vzdálené posloupnosti od posloupnosti Q?

Využití dolních odhadů



Snaž se co nejčastěji využít **levné postupy** pro výpočet dolní meze!

Algorithm Lower_Bounding_Sequential_Scan(Q)

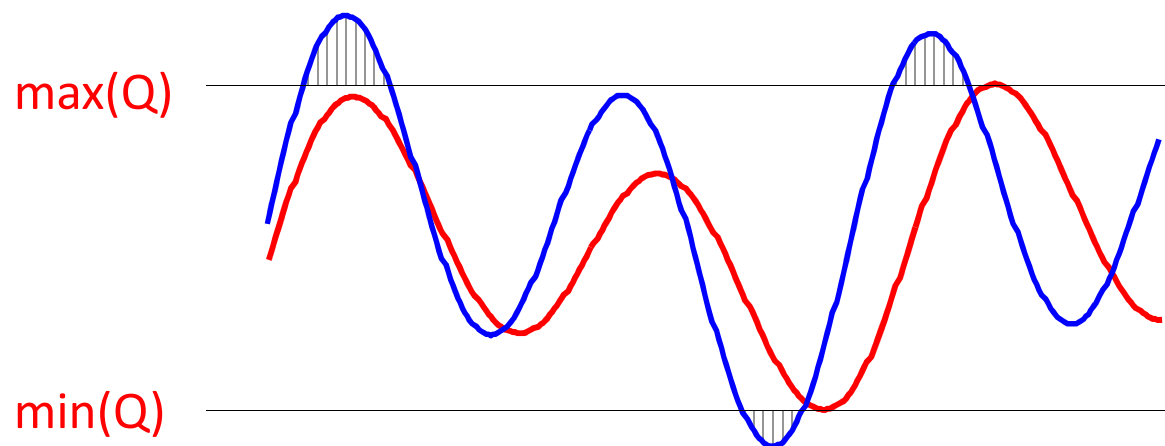
```
1.  best_so_far = infinity;
2.  for all sequences in database
3.    LB_dist = lower_bound_distance( Ci, Q);
4.    if LB_dist < best_so_far
5.      true_dist = DTW( Ci, Q);
6.      if true_dist < best_so_far
7.        best_so_far = true_dist;
8.        index_of_best_match = i;
9.      endif
10.   endif
11. endfor
```



A náročné **kompletní výpočty** prováděj jen když to jinak opravdu nejde !!!

Výpočet dolní meze podle Yi

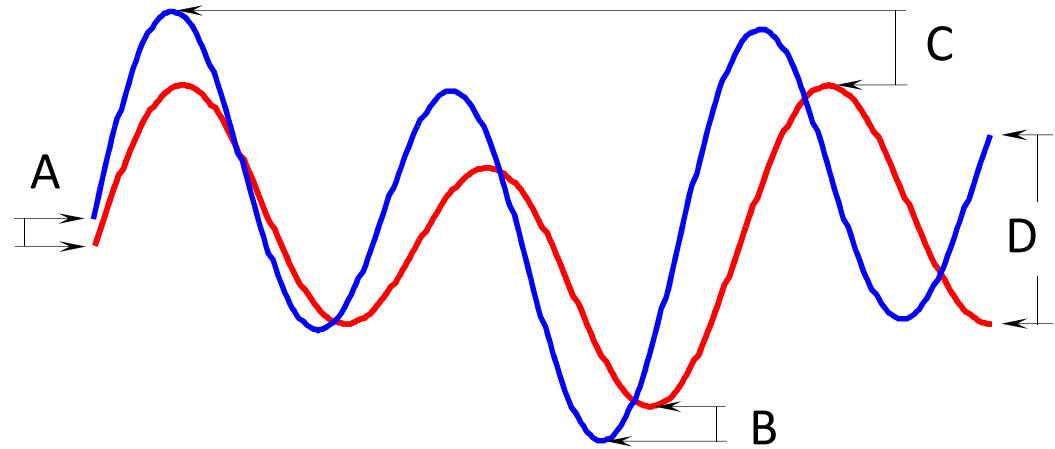
 LB_Yi



Yi, B, Jagadish, H & Faloutsos, C.
*Efficient retrieval of similar time
sequences under time warping.*
ICDE 98, pp 23-27.

Suma druhých mocnin šedých čar reprezentuje minimální hodnotu, kterou odpovídající body musí přispět do výpočtu vzdálenosti obou křivek. Nabízí proto také hodnotu pro stanovení dolní meze pro DTW.

Dolní odhad podle Kima

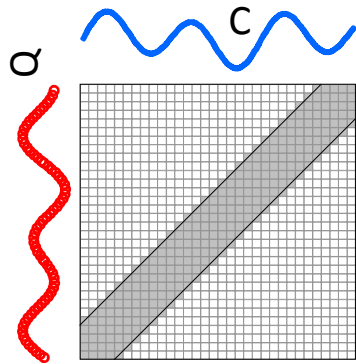


 LB_Kim

Kim, S, Park, S, & Chu, W. *An index-based approach for similarity search supporting time warping in large sequence databases*. ICDE 01, pp 607-614

Součet druhých mocnin vzdáleností následujících 4 bodů na obou křivkách: **začátek (A)**, **konec (D)**, **minimum (B)** a **maximum (C)** reprezentuje také dolní odhad vzdálenosti obou křivek.

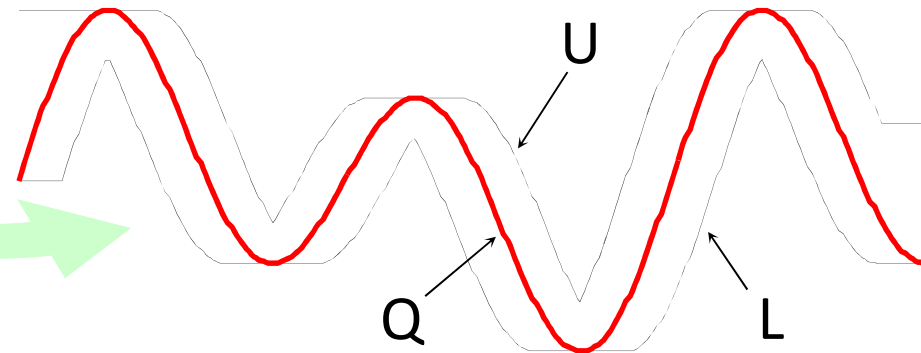
Dolní odhad podle Keogha



Sakoe-Chiba Band

$$U_i = \max(q_{i-r} : q_{i+r})$$

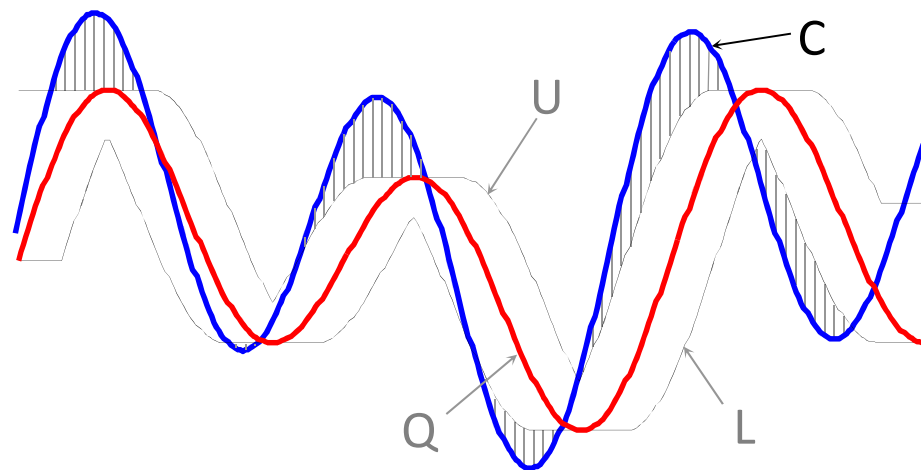
$$L_i = \min(q_{i-r} : q_{i+r})$$



LB_Keogh

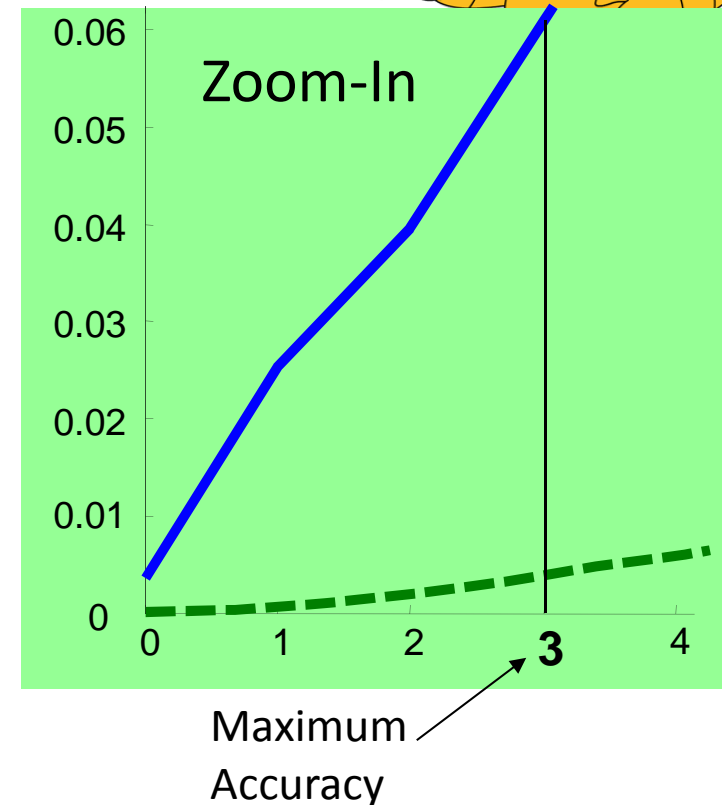
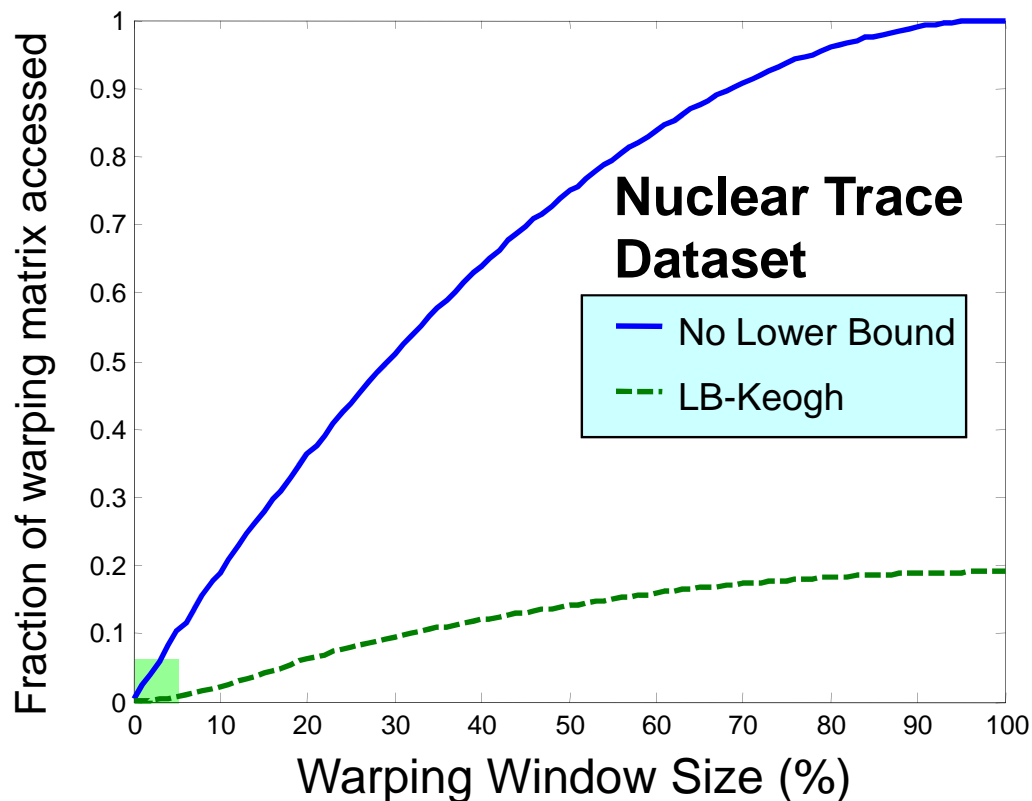
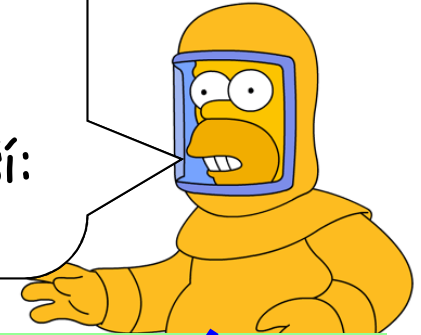
Obálková metoda
Metoda dolní meze

$$LB_Keogh(Q, C) = \sum_{i=1}^n \begin{cases} (q_i - U_i)^2 & \text{if } q_i > U_i \\ (q_i - L_i)^2 & \text{if } q_i < L_i \\ 0 & \text{otherwise} \end{cases}$$



Vyplatí se používat kombinaci „w-okno a dolní meze“?

DTW je sice $O(n^2)$, ale podaří-li se zvolit dobře velikost w-okna pro danou úlohu, musíme provést jen 6% práce (počet navštívených políček matice). A s použitím dolního odhadu podle Keogha je úspora ještě výraznější: jen 0.3% práce!



Časté předsudky o DTW ...

- *“DTW má vysoké požadavky na CPU”¹*
- *“DTW je vhodné jen pro malé soubory časových řad”²*
- *“(DTW) má kvadratickou složitost, a proto jej nelze použít na dlouhé časové řady”³*
- *“výkon DTW se výrazně zhoršuje s velikostí databáze”⁴*

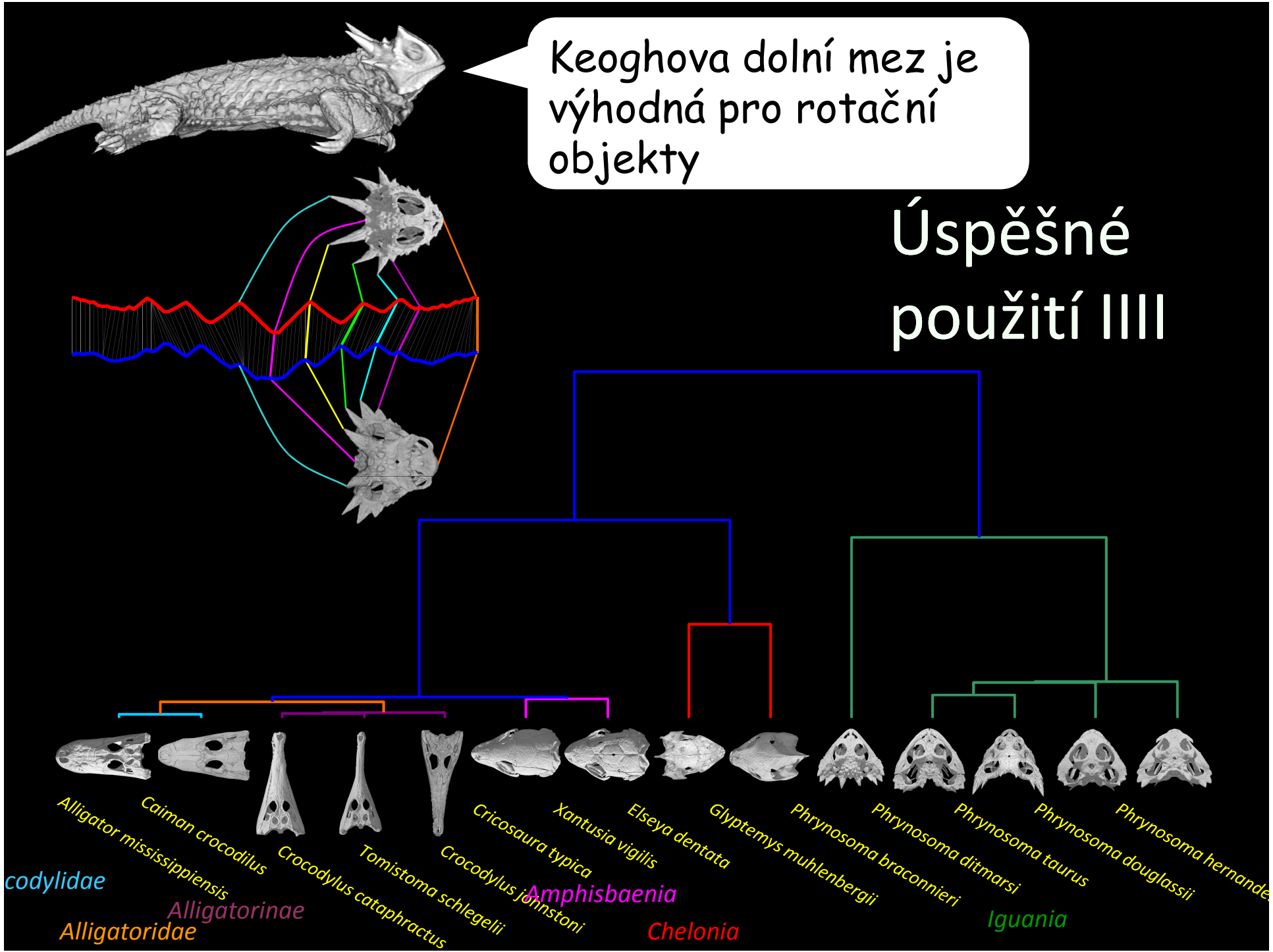
Toto není pravda!

...DTW má při použití „šikovných omezení“ pro DM lineární složitost!



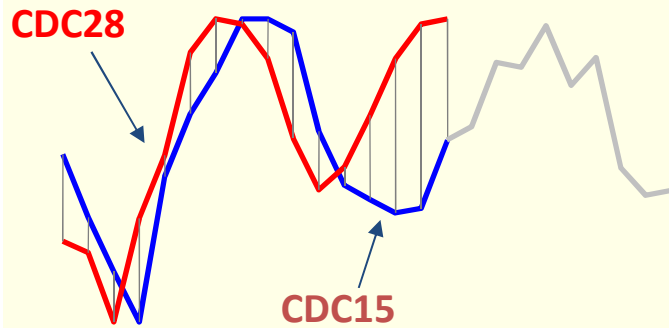
Keoghova dolní mez je výhodná pro rotační objekty

Úspěšné použití III

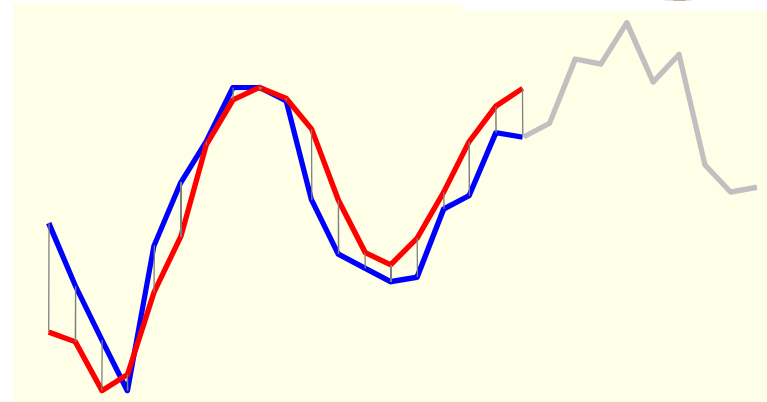
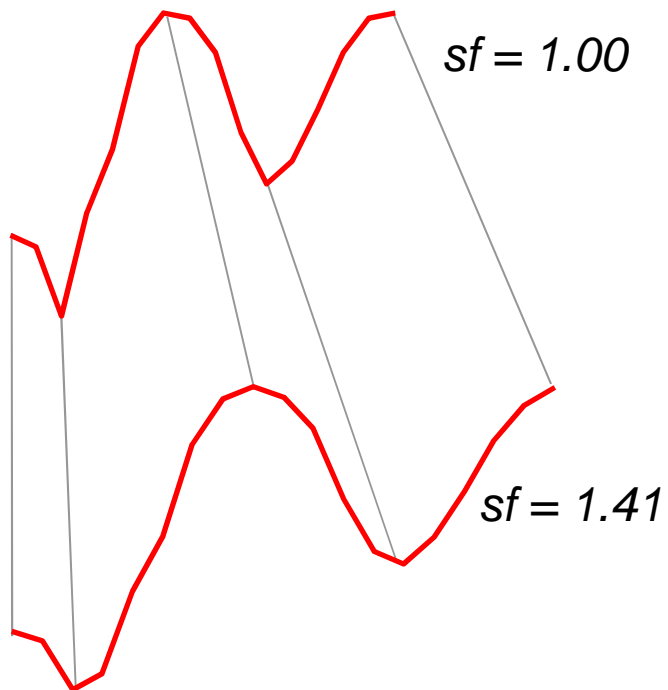


Uniformní škálování (uniform scaling)

Dva geny, které jsou funkčně spřízněné ...

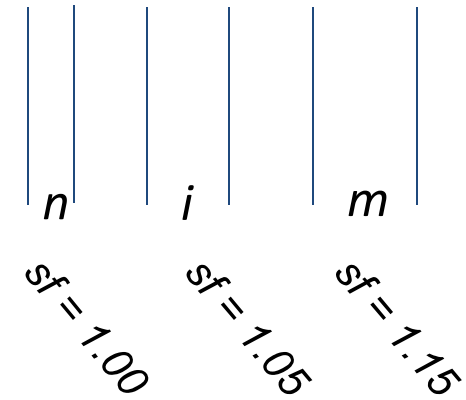


Někdy je globální nebo *uniformní* škálování stejně užitečné jako DTW

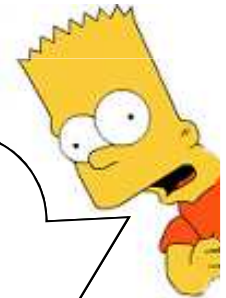




Here is some notation, the shortest scaling we consider is length n , and the largest is length m . The *scaling factor* (sf) is the ratio i/n , $n \leq i \leq m$



```
Algorithm: Test_All_Scalings(Q,C)
    best_match_val      = inf;
    best_scaling_factor  = null;
    for p = n to m
        QP = rescale(Q,p);
        distance = squared_Euclidean_distance(QP,C);
        if distance < best_match_val
            best_match_val = distance;
            best_scaling_factor = p/n;
        end;
    end;
    return(best_match_val, best_scaling_factor)
```



Here is the code to **Test_All_Scalings**, the time complexity is only $O((m-n) * n)$, but we may have to do this many times...

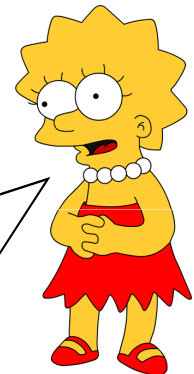
Lower Bounding Revisited!

We can speed up similarity search under uniform scaling by using a lower bounding function, just like we did for DTW.

Algorithm: Lower_Bounding_Sequential_Scan(Q,C)

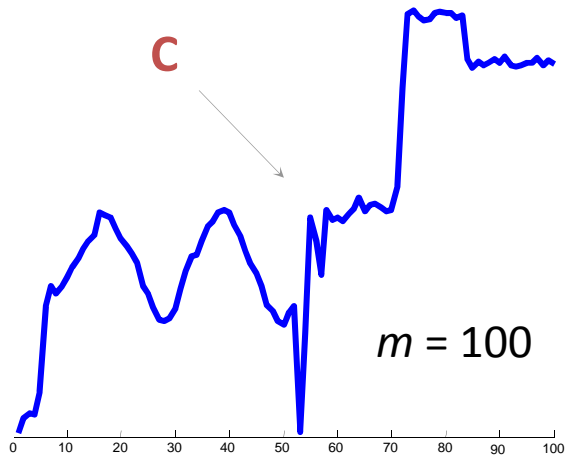
```
overall_best_time_series = null;
overall_best_match_val = inf;
for i = 1 to number_of_time_series_in_(C)
  if lower_bound_distance(Q,Ci) < overall_best_match_val
    [dist, scale] = Test_All_Scalings(Q,Ci)
    if dist < overall_best_match_val
      overall_best_time_series = i;
      overall_best_match_val = dist;
    end;
  end;
end;
end;
```

You have already seen this idea for DTW!



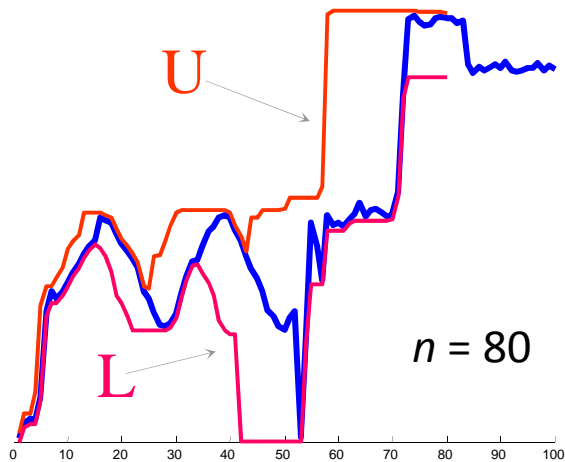
But is there a lower bound for uniform scaling?





Assume that you have a database of time series C_i , all of length 100.

You have a query Q , of length 80, and you want to find the best match in the database under any scaling of Q , from 80 to 100.

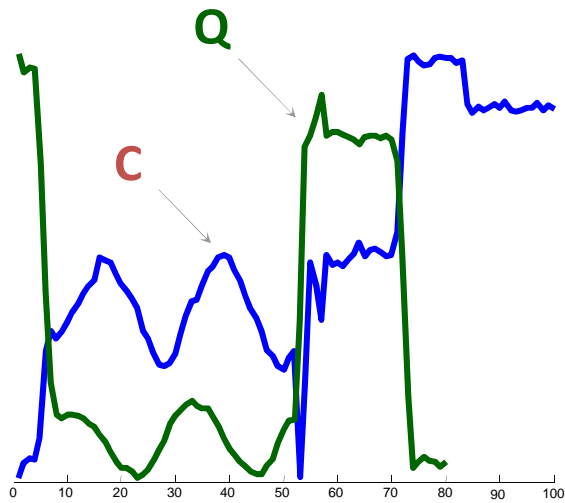


We can build envelopes around all candidates time series C_i , in our database, just like we did for DTW, except the definition of the envelopes is different.

$$U_i = \max(c_{\lfloor (i-1)*m/n \rfloor + 1}, \dots, c_{\lfloor i*m/n \rfloor})$$

$$L_i = \min(c_{\lfloor (i-1)*m/n \rfloor + 1}, \dots, c_{\lfloor i*m/n \rfloor})$$





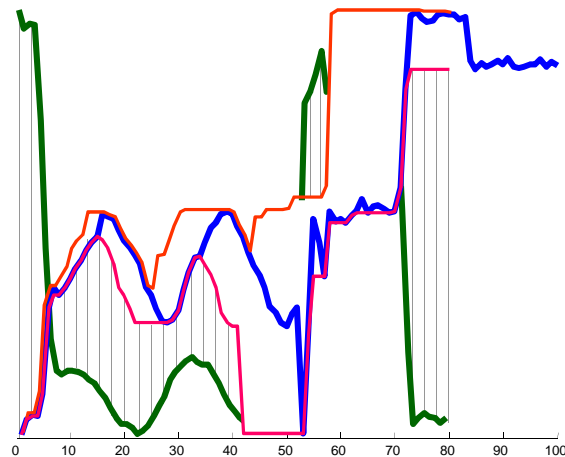
Once the envelopes have been built, we can lower bound **Test_All_Scalings**.

What's more, the lower bound is one we have already seen!



 **LB_Keogh**

Envelope-Based Lower Bound



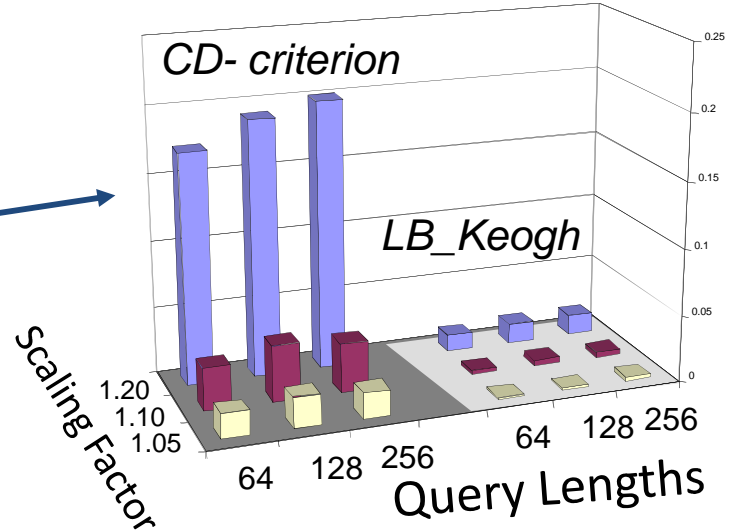
$$LB_Keogh(Q, C) = \sum_{i=1}^n \begin{cases} (q_i - U_i)^2 & \text{if } q_i > U_i \\ (q_i - L_i)^2 & \text{if } q_i < L_i \\ 0 & \text{otherwise} \end{cases}$$

An experiment to test the utility of lower bounding uniform scaling, over different scaling factors (Y-axis) and scaling lengths (X-axis). The dataset was a “mixed bag” of 10,000 assorted time series.

This is the time taken by brute force search



CD-criterion is the only other lower bound for uniform scaling





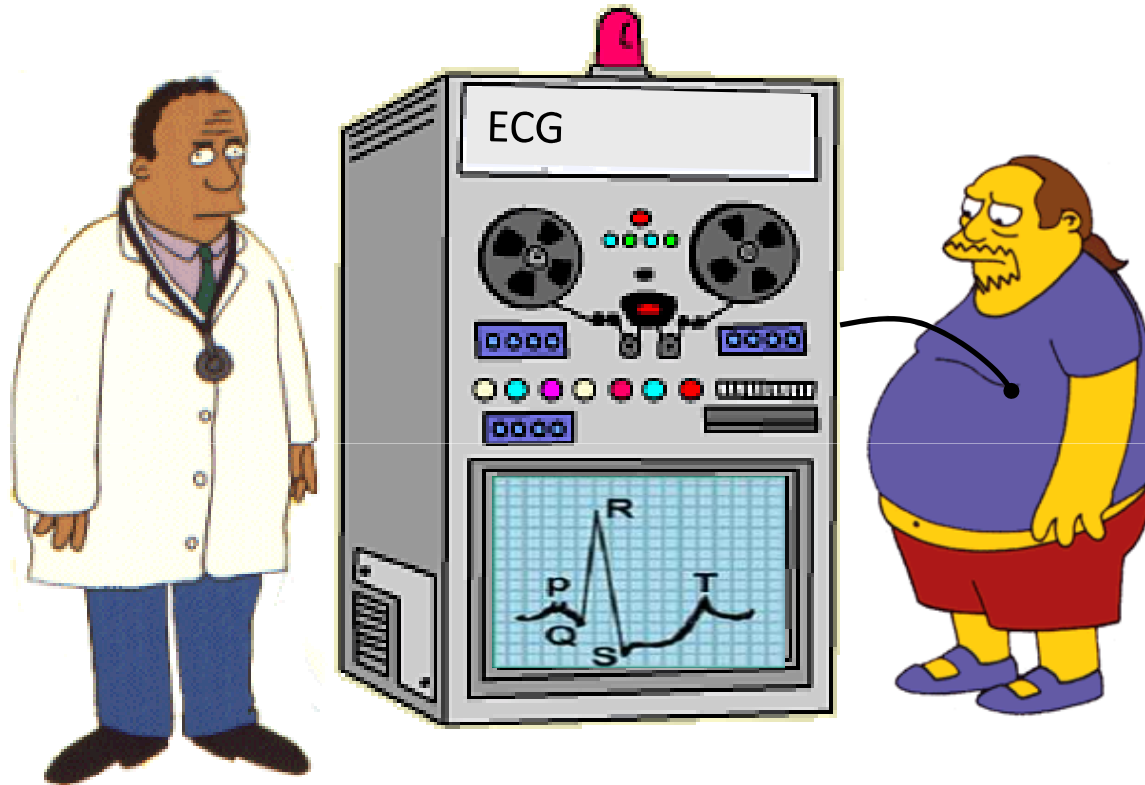
Apart from making DTW tractable for data mining for the first time, *envelope based techniques* also allow...

1. More accurate classification (SDM04)
2. Indexing with uniform scaling (VLDB04)
3. Faster Euclidean indexing (TKDE04)
4. Subsequence matching (IDEAS03)
5. Multivariate time series indexing (SIGKDD03)
6. Rotation invariant indexing (SIGKDD04)
7. DTW on Streaming time series (to appear)
8. Indexing of Images (TPAMI-04, VIS-05)

We strongly feel that envelope based techniques are the best solutions for time series similarity



Nový pohled na motivační příklad ...

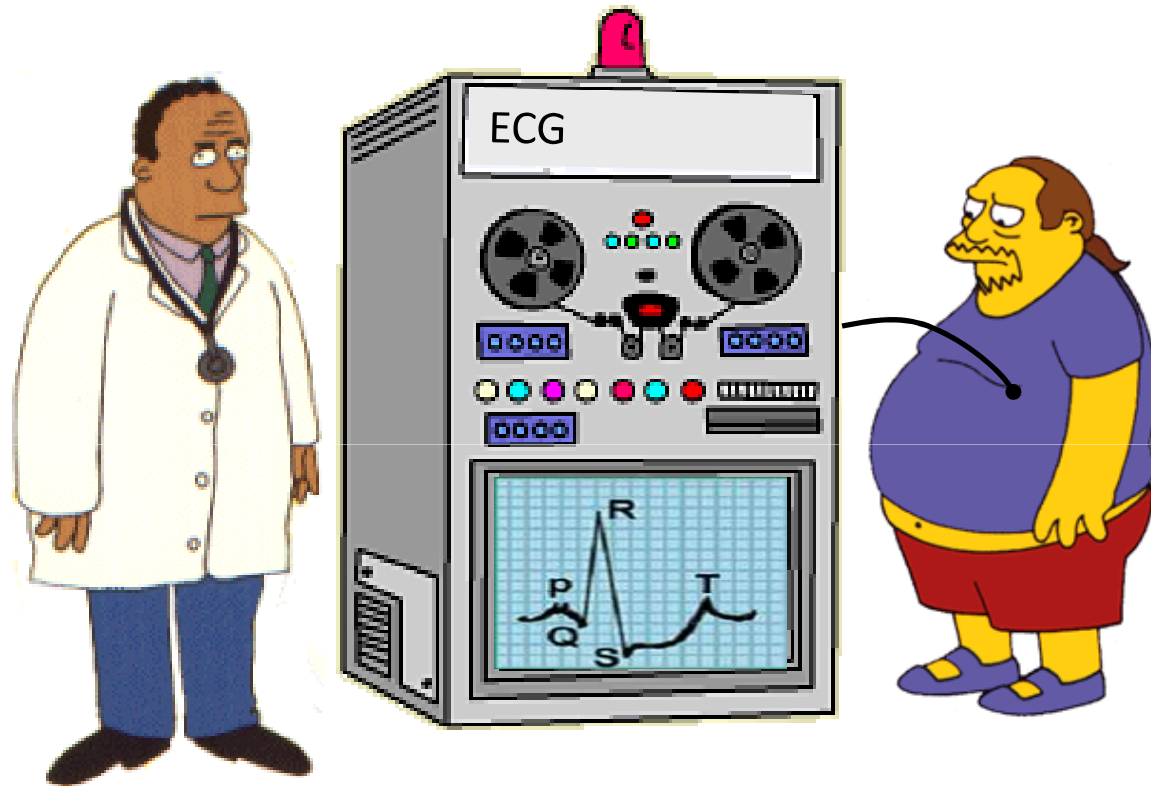


Pacient si stěžuje na bolest na prsou. Jeho EKG vypadá neobvykle ...

Lékař by rád našel v archivu nějaký podobný případ, jehož EKG vypadá **podobně**. Doufá totiž, že to by ho mohlo varovat před možnou chybou, inspirovat při návrhu řešení ...

2 otázky:

- Jak poznat podobné křivky? – definice míry vzdálenosti
- **Jak realizovat vyhledávání RYCHLE?**



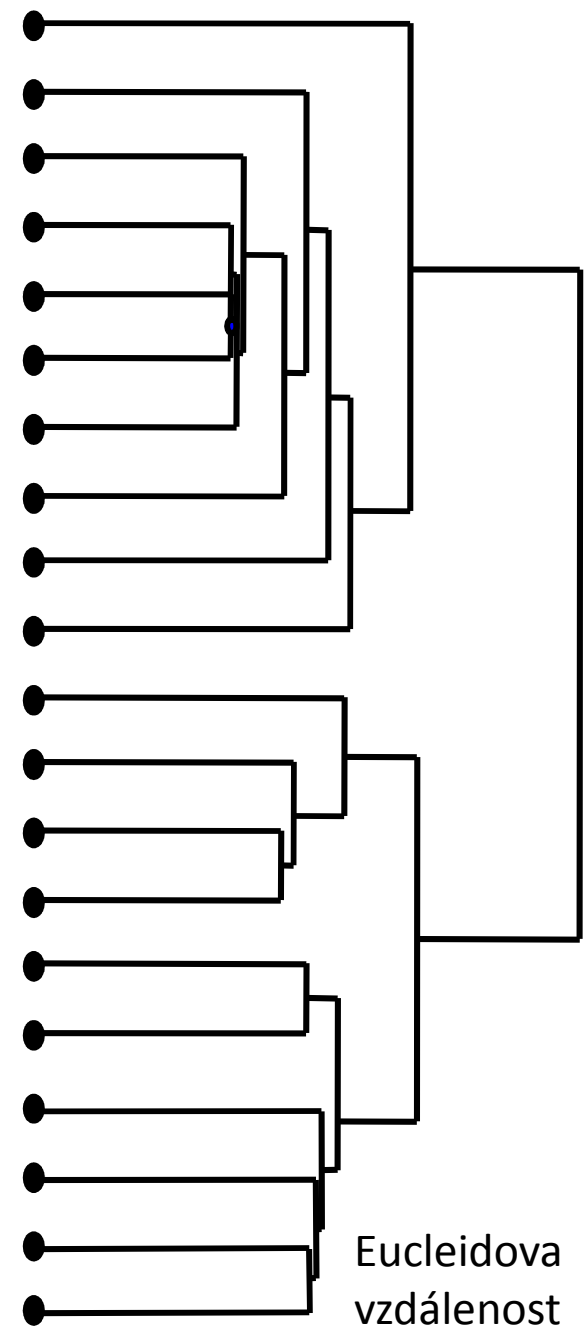
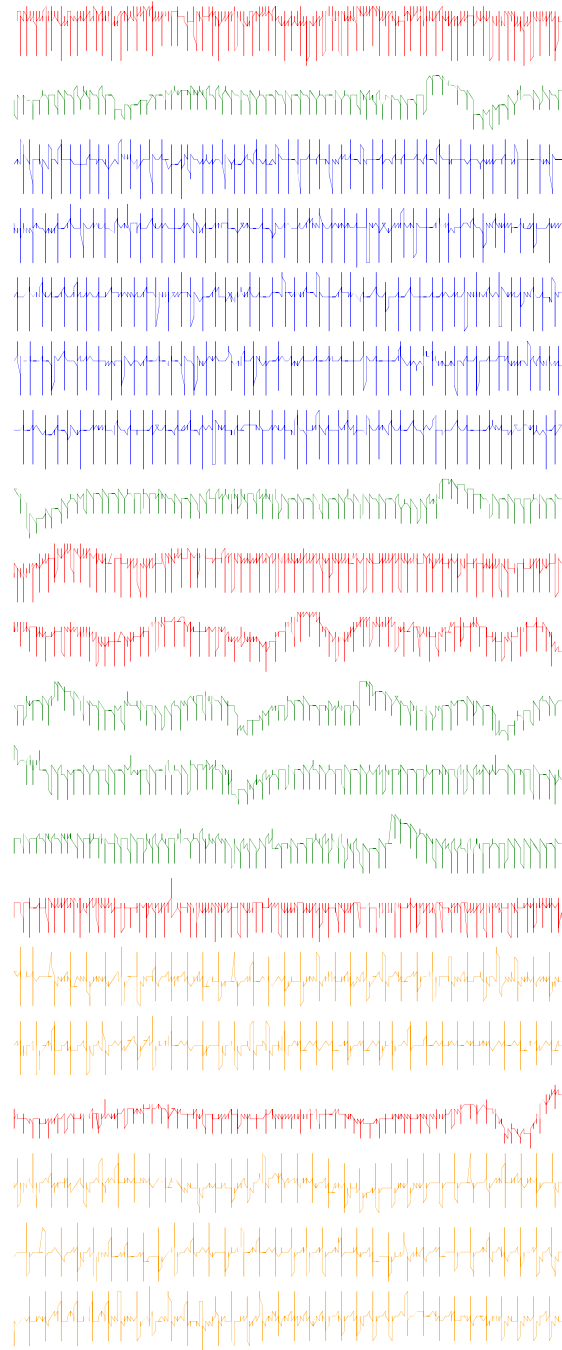
You go to the doctor because of chest pains. Your ECG looks strange...

Your doctor wants to search a database to find **similar** ECGs, in the hope that they will offer clues about your condition...

Two questions:

- How do we define similar?
- **How do we search quickly?**

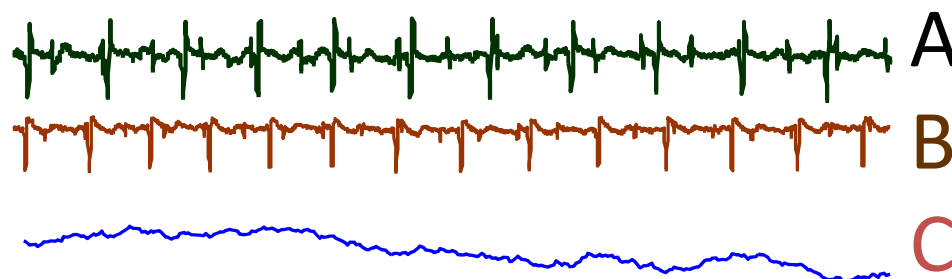
U velmi dlouhých čas.řad dostáváme s použitím *podobnosti založené na tvaru* poměrně špatné výsledky. Podobnost musí brát v úvahu *strukturu vyšších řádů*.



Euclidova vzdálenost

Využití struktury/podobnost vycházející z modelu

Základem je extrakce více *globálních* atributů, jejichž vektor dobře **reprezentuje** přísl. časové řady. Míra podobnosti se pak počítá z tohoto vektoru!



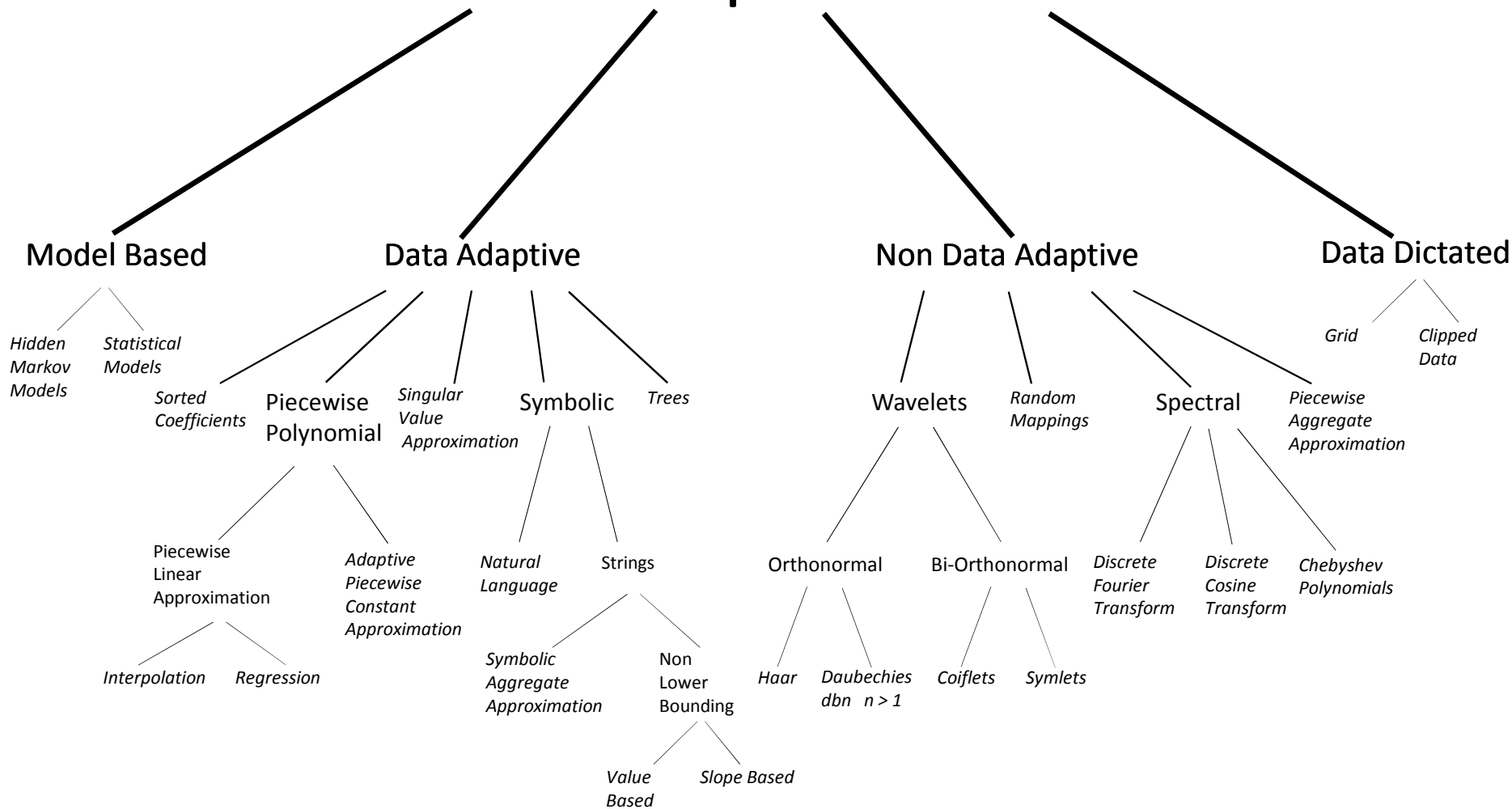
Time Series \ Feature	A	B	C
Max Value	11	12	19
Autocorrelation	0.2	0.3	0.5
Zero Crossings	98	82	13
ARIMA	0.3	0.4	0.1
...



Jak vybrat vhodné

- **atributy?**
- **míry vzdálenosti?**
- **algoritmy učení?**

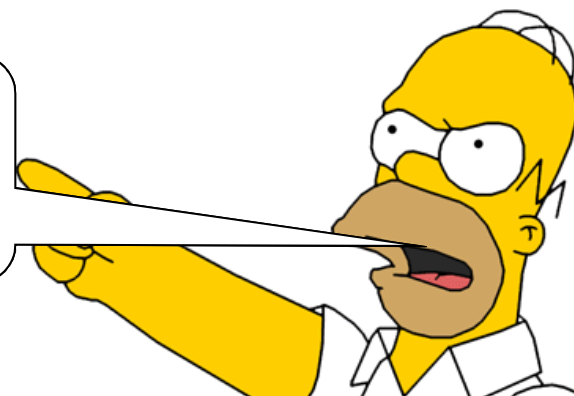
Time Series Representations



Obecný DM algoritmus pro práci s velkými daty

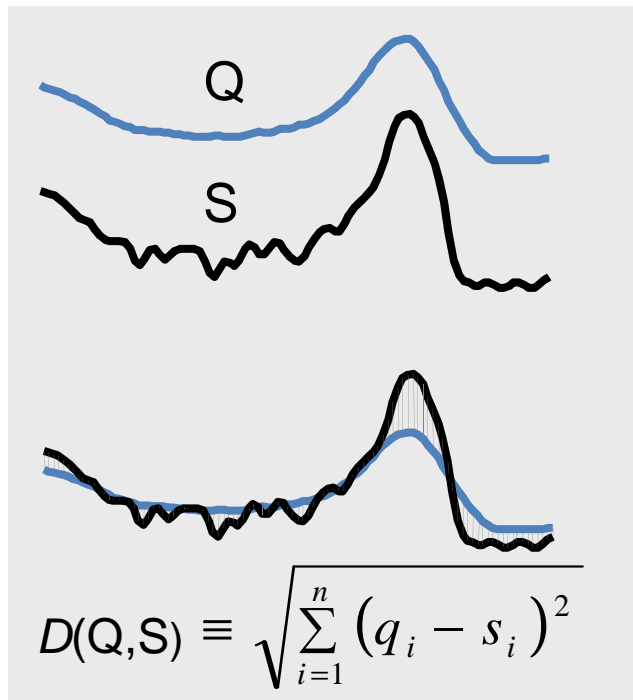
1. Vytvoř si *aproximaci* zpracovávaných dat tak, aby se vešla do paměti počítače aniž by ztratila základní vlastnosti původní úlohy.
2. Najdi pomocí těchto dat **přibližné řešení**
3. Otestuj, zda řešení získané v bodě 2 je řešením i pro původní data nebo toto řešení modifikuj, aby z něj vzniklo řešení pro původní data. Minimalizuj při tom počet přístupů na externí paměťová média.

Tento postup funguje jen když použité aproximace umožňují získat dolní hranice!



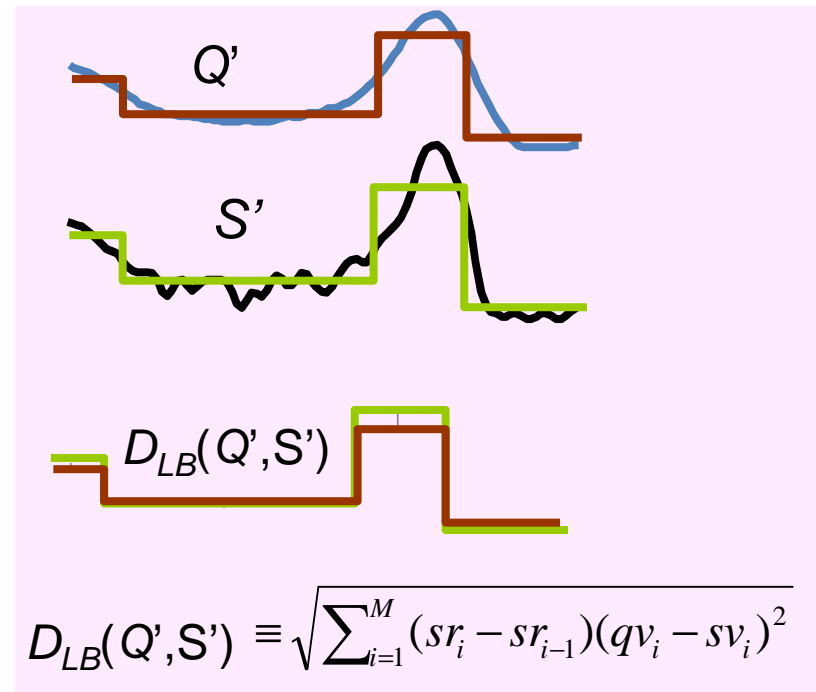
Nová reprezentace jako zdroj dolních mezí

- Dolní meze se osvědčily při výpočtu míry vzdálenosti. U reprezentací můžeme postupovat podobně ...



Původní data

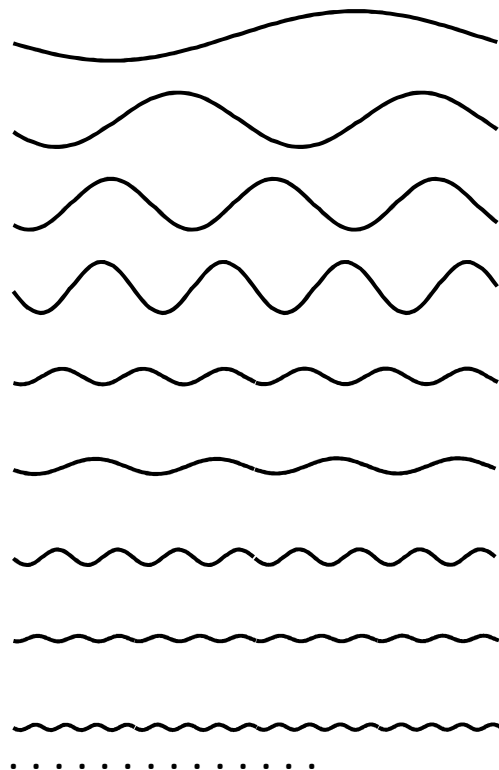
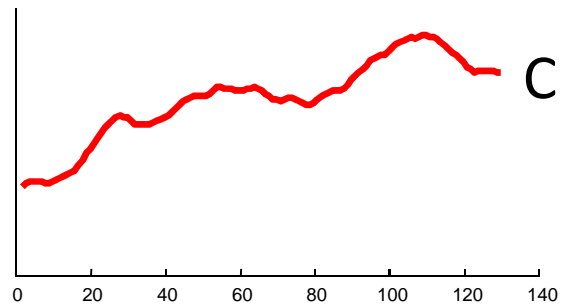
Aproximace či
"nová"
reprezentace



Lower bounding means that for all Q and S, we have:

$$D_{LB}(Q',S') \leq D(Q,S)$$

Příklad postupu pro redukci dimensionality II



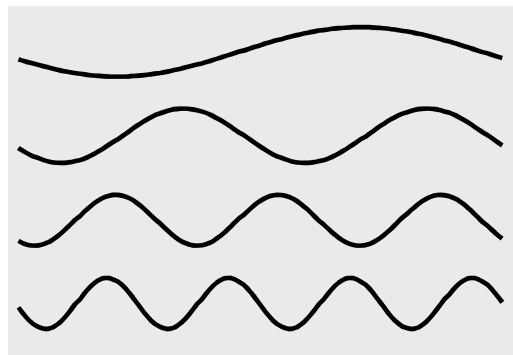
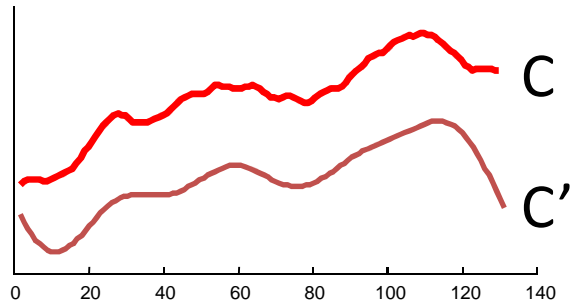
Raw Data **Fourier Coefficients**

0.4995	1.5698
0.5264	<u>1.0485</u>
0.5523	0.7160
0.5761	<u>0.8406</u>
0.5973	0.3709
0.6153	<u>0.4670</u>
0.6301	0.2667
0.6420	<u>0.1928</u>
0.6515	0.1635
0.6596	<u>0.1602</u>
0.6672	0.0992
0.6751	<u>0.1282</u>
0.6843	0.1438
0.6954	<u>0.1416</u>
0.7086	0.1400
0.7240	<u>0.1412</u>
0.7412	0.1530
0.7595	<u>0.0795</u>
0.7780	0.1013
0.7956	<u>0.1150</u>
0.8115	0.1801
0.8247	<u>0.1082</u>
0.8345	0.0812
0.8407	<u>0.0347</u>
0.8431	0.0052
0.8423	<u>0.0017</u>
0.8387	0.0002
...	...
...	...
...	...

Původní data mohou být rozložena pomocí Diskrétní Fourierovy Transformace (DFT) na 64 čistých sinusovek – několik prvních je na obrázku.

Fourierovy koeficienty představují novou reprezentaci původních dat!

Další postup



Podařilo se nám výrazně
snížit objem dat: 15/16
nepotřebujeme

Raw Data

0.4995
0.5264
0.5523
0.5761
0.5973
0.6153
0.6301
0.6420
0.6515
0.6596
0.6672
0.6751
0.6843
0.6954
0.7086
0.7240
0.7412
0.7595
0.7780
0.7956
0.8115
0.8247
0.8345
0.8407
0.8431
0.8423
0.8387

...
...

Fourier Coefficients

1.5698
1.0485
0.7160
0.8406
0.3709
0.4670
0.2667
0.1928
0.1635
0.1602
0.0992
0.1282
0.1438
0.1416
0.1400
0.1412
0.1530
0.0795
0.1013
0.1150
0.1801
0.1082
0.0812
0.0347
0.0052
0.0017
0.0002

...
...
...

Vybrané (*Truncated*) Fourierovy koeficienty

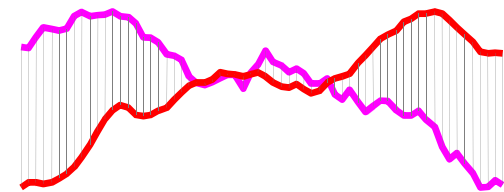
1.5698
1.0485
0.7160
0.8406
0.3709
0.4670
0.2667
0.1928

$$n = 128$$

$$N = 8$$

$$C_{\text{ratio}} = 1/16$$

Od redukce dimenze k dolním mezím



$$D(Q, C) \equiv \sqrt{\sum_{i=1}^n (q_i - c_i)^2}$$

Raw
Data 1 Raw
Data 2

0.4995	-	0.7412
0.5264	-	0.7595
0.5523	-	0.7780
0.5761	-	0.7956
0.5973	-	0.8115
0.6153	-	0.8247
0.6301	-	0.8345
0.6420	-	0.8407
0.6515	-	0.8431
0.6596	-	0.8423
0.6672	-	0.8387
0.6751	-	0.4995
0.6843	-	0.5264
0.6954	-	0.5523
0.7086	-	0.5761
0.7240	-	0.5973
0.7412	-	0.6153
0.7595	-	0.6301
0.7780	-	0.6420
0.7956	-	0.6515
0.8115	-	0.6596
0.8247	-	0.6672
0.8345	-	0.6751
0.8407	-	0.6843
0.8431	-	0.6954
0.8423	-	0.7086
0.8387	-	0.7240
...		...
...		...



Vybrané
Fourierovy
koeficienty 1

1.5698
<u>1.0485</u>
0.7160
<u>0.8406</u>
0.3709
<u>0.4670</u>
0.2667
<u>0.1928</u>

Vybrané
Fourierovy
koeficienty 2

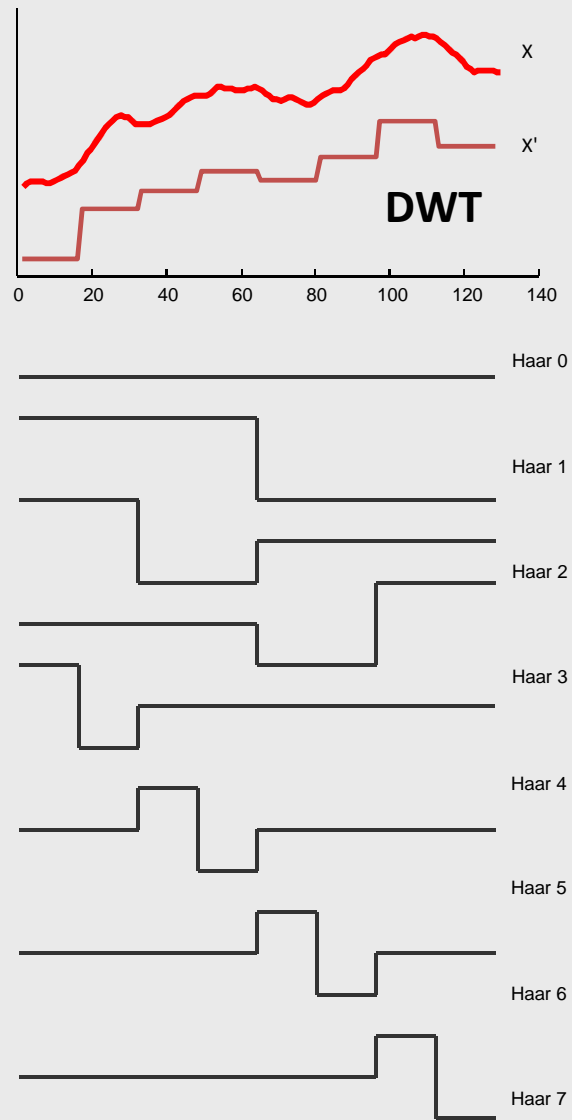
1.1198
<u>1.4322</u>
1.0100
<u>0.4326</u>
0.5609
<u>0.8770</u>
0.1557
<u>0.4528</u>

Eukleidova vzdálenost mezi dvěma vektory vybraných Fourierových koeficientů je vždy menší než vzdálenost mezi původními řadami*.

Tedy **DFT** přináší možnost zjistit **dolní meze!**

*Parsevalova věta

Discrete Wavelet Transform I



Basic Idea: Represent the time series as a linear combination of Wavelet basis functions, but keep only the first N coefficients.

Although there are many different types of wavelets, researchers in time series mining/indexing generally use Haar wavelets.

Haar wavelets seem to be as powerful as the other wavelets for most problems and are very easy to code.

[Excellent free Wavelets Primer](#)

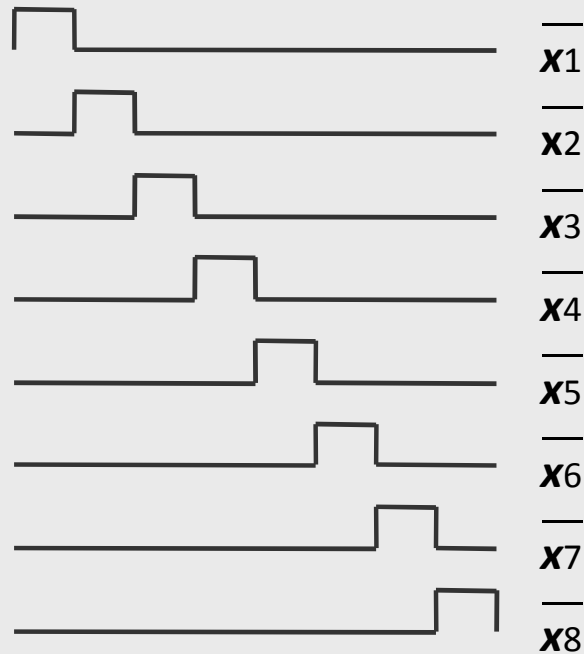
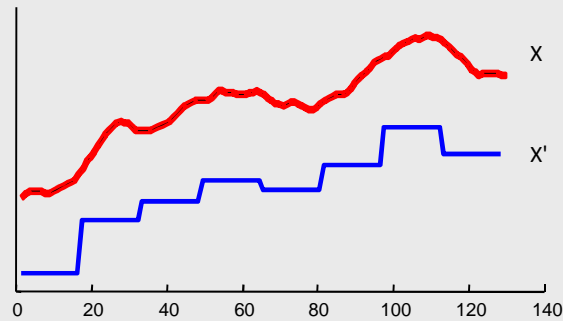
Stollnitz, E., DeRose, T., & Salesin, D. (1995). *Wavelets for computer graphics A primer: IEEE Computer Graphics and Applications*.



Alfred Haar

1885-1933

Piecewise Aggregate Approximation I



Basic Idea: Represent the time series as a sequence of N box basis functions.

Note that each box is of the same length (n/N).

$$\bar{x}_i = \frac{N}{n} \sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} x_j$$

Given the reduced dimensionality representation we can calculate the approximate Euclidean distance as...

$$DR(\bar{X}, \bar{Y}) \equiv \sqrt{\frac{n}{N}} \sqrt{\sum_{i=1}^N (\bar{x}_i - \bar{y}_i)^2}$$

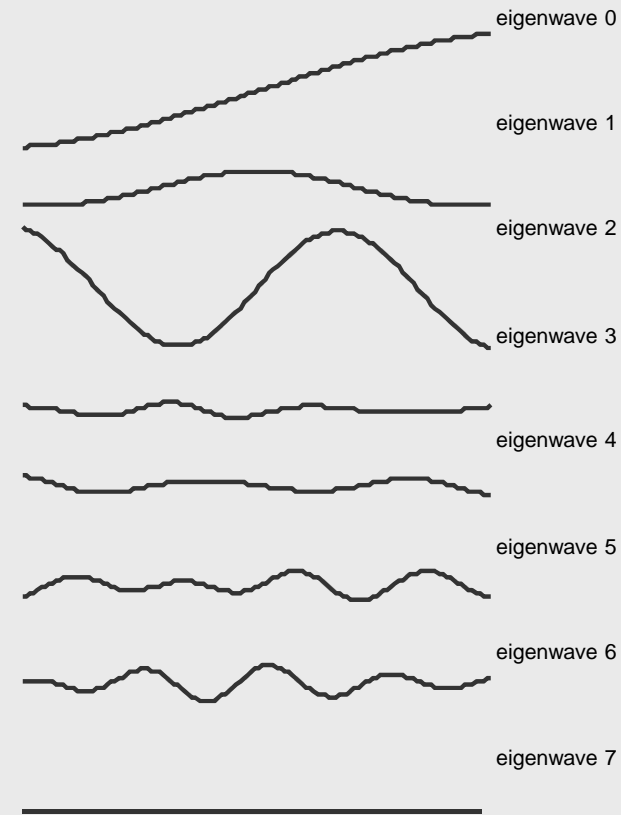
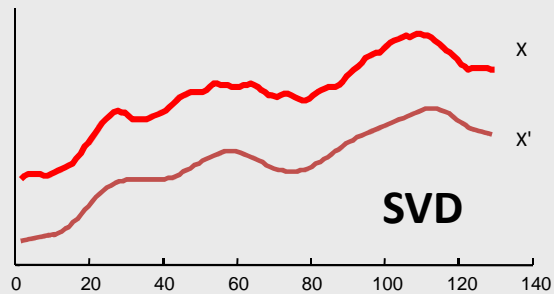
This measure is provably lower bounding.

Independently introduced by two authors

- Keogh, Chakrabarti, Pazzani & Mehrotra, KAIS (2000) / Keogh & Pazzani PAKDD April 2000

- Byoung-Kee Yi, Christos Faloutsos, VLDB September 2000

Singular Value Decomposition I



Basic Idea: Represent the time series as a linear combination of *eigenwaves* but keep only the first N coefficients.

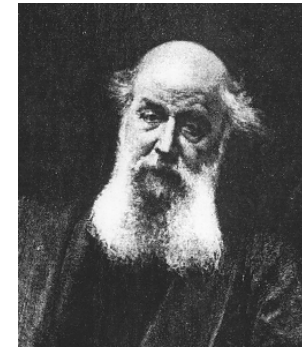
SVD is similar to Fourier and Wavelet approaches is that we represent the data in terms of a linear combination of shapes (in this case *eigenwaves*).

SVD differs in that the *eigenwaves* are data dependent.

SVD has been successfully used in the text processing community (where it is known as *Latent Symantec Indexing*) for many years.

[Good free SVD Primer](#)

Singular Value Decomposition - A Primer.
Sonia Leach



James Joseph Sylvester
1814-1897

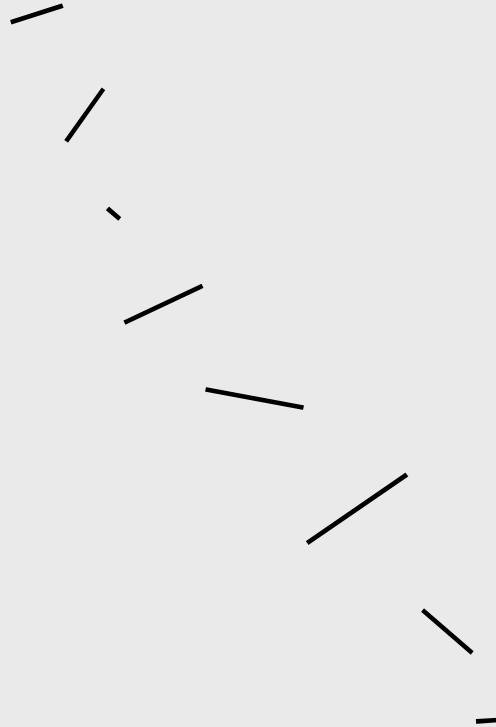
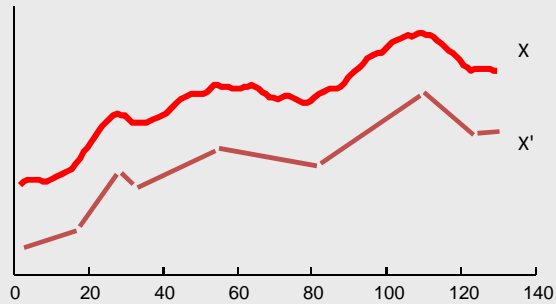


Camille Jordan
(1838--1921)



Eugenio Beltrami
1835-1899

Piecewise Linear Approximation



Basic Idea: Represent the time series as a sequence of straight lines.

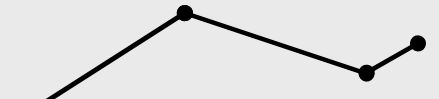
Lines could be **connected**, in which case we are allowed $N/2$ lines

If lines are **disconnected**, we are allowed only $N/3$ lines

Personal experience on dozens of datasets suggest **disconnected** is better. Also only **disconnected** allows a lower bounding Euclidean approximation



Karl Friedrich Gauss, 1777 - 1855



Each line segment has

- length
- left_height

(right_height can be inferred by looking at the next segment)

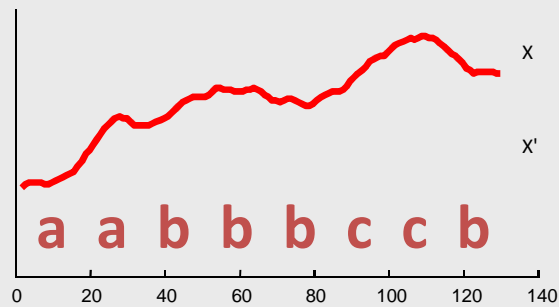


Each line segment has

- length
- left_height
- right_height

- Good ability to compress natural signals.
- Fast linear time algorithms for PLA exist.
- Already widely accepted in some communities (e.g. biomedical)

Symbolic Approximation I



a

0

a

1

b

2

b

3

b

4

c

5

c

6

b

7

Basic Idea: Convert the time series into an alphabet of discrete symbols. Use string indexing techniques to manage the data.

Potentially an interesting idea, but all work thus far are very ad hoc.

Pros and Cons of Symbolic Approximation as a time series representation.

- Potentially, we could take advantage of a wealth of techniques from the very mature field of string processing and bioinformatics.
- It is not clear how we should discretize the times series (discretize the values, the slope, shapes? How big of an alphabet? etc).
- There are more than 210 different variants of this, at least 35 in data mining conferences.

Shrnutí

- Pro práci s **krátkými časovými řadami** je vhodné používat DTW, u kterého je
 - je vhodně nastavena velikost a tvar w-okna
 - a použijí se dolní meze pro zrychlení.
- V případě **dlouhých časových řad**, o nichž nic nevíme, je užitečné použít **kompresní techniky**.
- Pokud o zpracovávaných řadách existují nějaké expertní znalosti, využijte je pro **extrakci vhodných atributů**.