Dynamic Time Warping

A Tool for Time Series Analysis

By: Hongpeng Zhang, Zihao Ding

Presentation Outline

- What is Time Series Data
- Basics of Dynamic Time Warping
- Code break down & analysis
- Experiment
- Result
- Reference

- How slow DTW actually is
- How to do DTW faster

Time Series Data

Sequential Data

- Sequential Data, as name implies, are sequences

- Difference between a sequence and a set:
    - A sequence X is a set of elements, together with a total order imposed on those elements

- Examples of sequential data:
    - Strings - sequences of characters
    - Time Series - sequences of vectors

Time Series Data

- A sequence of observations made over time

- Examples:
    - Stock market prices
    - Heart rate over time
    - Speech - represented as a sequence of audio measurements at discrete time intervals
    - Music - represented as sequence of pairs of note and duration

Time Series Data Classification Applications:

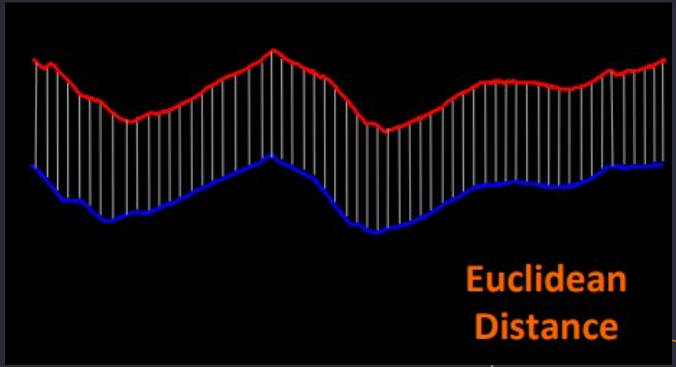
- Stock market prices
  - Price Prediction (stock price, oil price, currencies etc.)
- Heart rate
  - healthy heart or maybe has disease
- Speech
  - Speech recognition
- Music
  - Music recognition

Tool for Time Series Data Classification

Time Series Classification Tools:

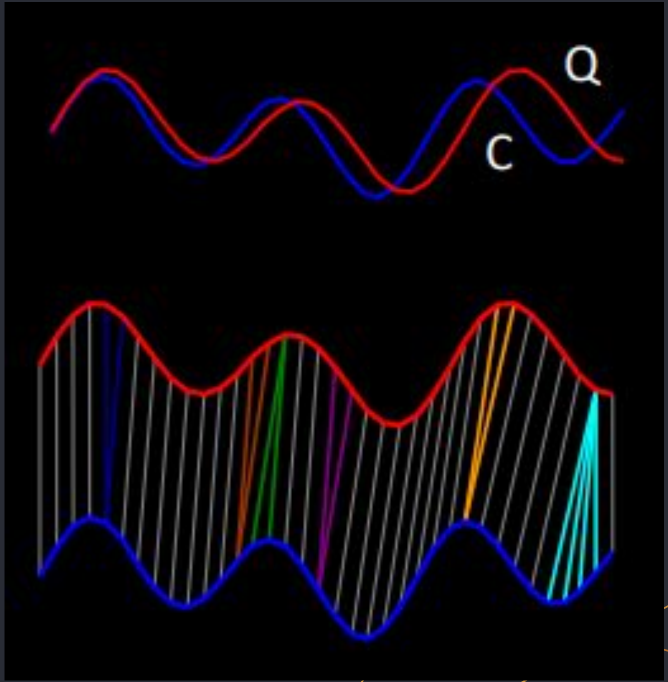
$$d(p, q) = \sqrt{(p - q)^2}.$$

- Euclidean Distance



Dynamic Time Warping (DTW)

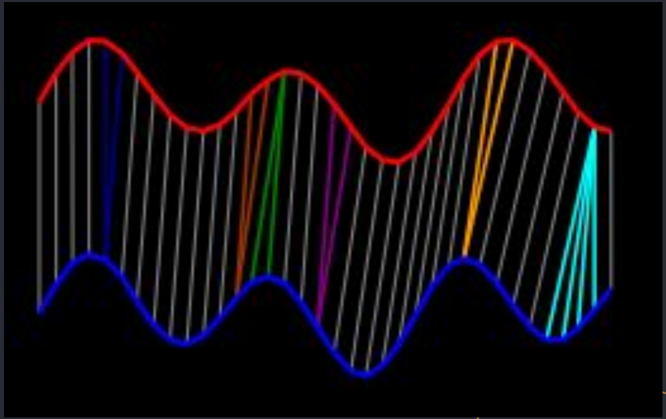
DTW is an algorithm for measuring similarity between two time series which may vary (i.e. warp) in timing.



Rules of DTW Alignment

DTW is a method that calculates an optimal match between two given sequences (e.g. time series) with certain restriction and rules:

- Boundary conditions
- Monotonicity
- Continuity



Finding Alignments

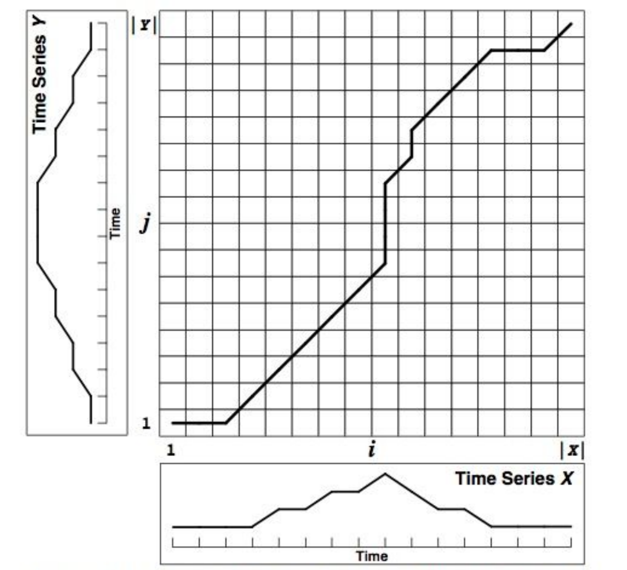
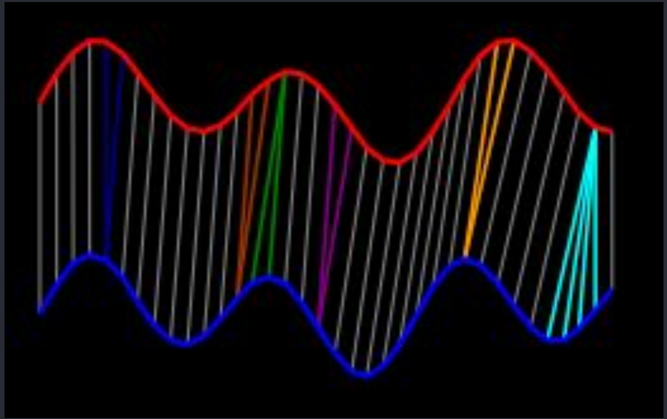


Figure 2. A cost matrix with the minimum-distance warp path traced through it.



The Cost of DTW Alignment

$$D(i,j) = Dist(i,j) + \min[D(i-1,j), D(i,j-1), D(i-1,j-1)]$$

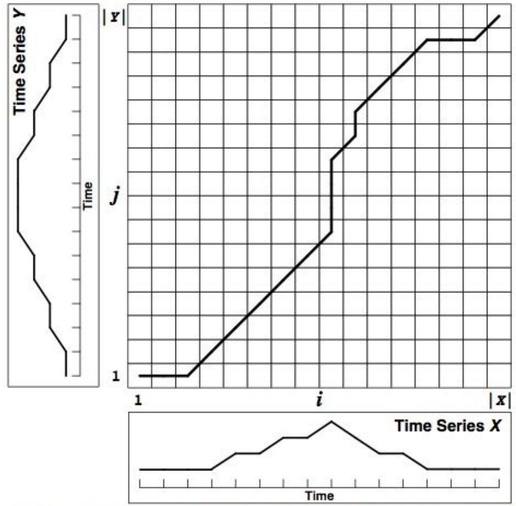


Figure 2. A cost matrix with the minimum-distance warp path traced through it.

DTW implementation

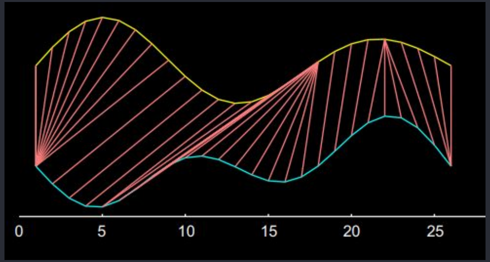
Pseudocode

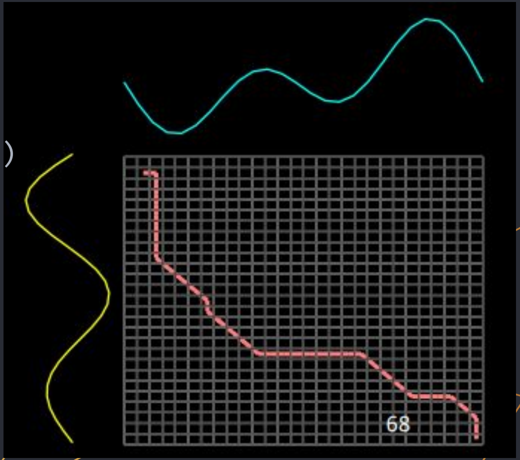
```
def dtw(x, y):
    # Initialization
    for i = 1..n
        for j = 1..m
            C[i, j] = inf

    C[0, 0] = 0.

   # Main loop
   for i = 1..n          # For Each Row
       for j = 1..m     # For Each Column
           dist = d(x_i, y_j) ** 2       # ED distance
           C[i, j] = dist + min(C[i-1, j], C[i, j-1], C[i-1, j-1])

   return sqrt(C[n, m])
```

Complexity:

O(mn) = O(n^2)   time

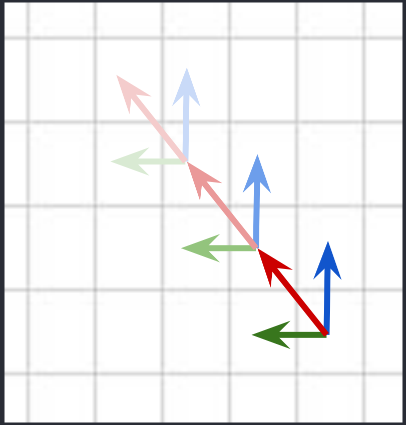
O(mn) = O(n^2)   space





Step Visualization

```
  def dtw(x, y):
 # Initialization
 ...

# Main loop
for i = 1..n            # For Each Row
     for j = 1..m       # For Each Column
         dist = d(x_i, y_j) ** 2  # ED distance
         C[i, j] = dist + min(C[i-1, j], C[i, j-1],
                              C[i-1, j-1])

return
```



Simple Example

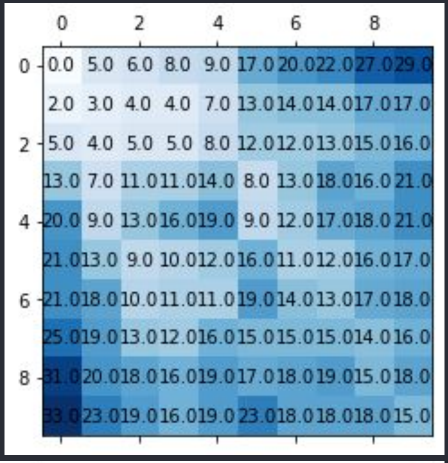
a = [1, 3, 4, 9, 8, 2, 1, 5, 7, 3]
b = [1, 6, 2, 3, 0, 9, 4, 3, 6, 3]

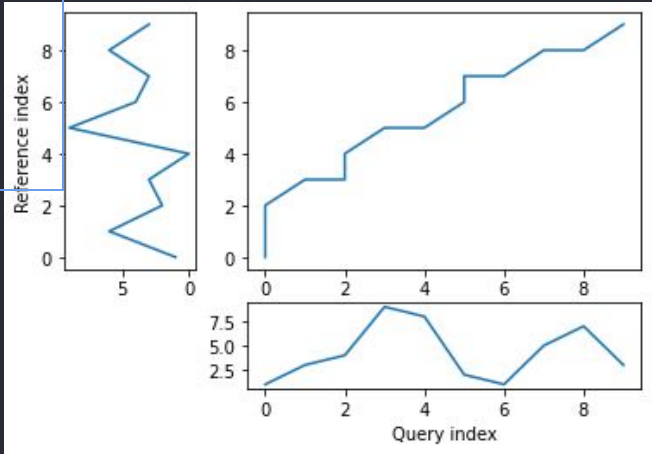
3	33	23	19	16	19	23	18	17	18	15
7	31	20	18	16	19	17	17	18	15	18
5	25	19	13	12	16	15	14	15	14	16
1	21	18	10	11	11	19	14	13	17	18
2	21	13	9	10	12	16	11	12	16	17
8	20	9	13	16	19	9	12	17	18	21
9	13	7	11	11	14	8	13	18	16	21
4	5	4	5	5	8	12	12	13	15	16
3	2	3	4	4	7	13	14	14	17	17
1	0	5	6	8	9	17	20	22	27	29
	1	6	2	3	0	9	4	3	6	3

= | Ai - Bj | + D [i-1, 0]
= | 8 - 1 | + 13 = 20

= | Ai - Bj | + min(D [i-1, j-1]
D[i-1, j]
D[i, j-1])
= | 8 - 0 | + 11 = 19

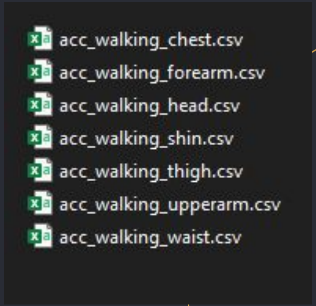
= | Ai - Bj | + D [0, j - 1]
= | 1 - 3 | + 6 = 8

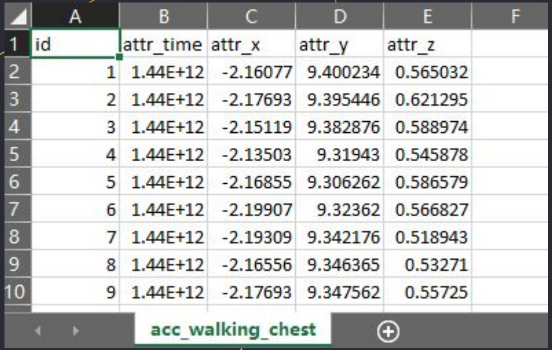




Experiment Setup

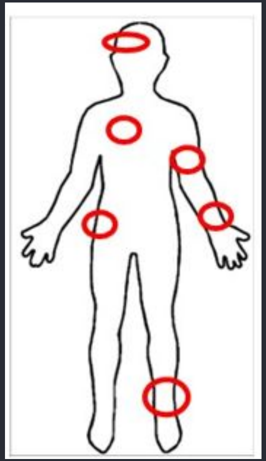
- Using RealWorld (HAR) Dataset[2] as Input

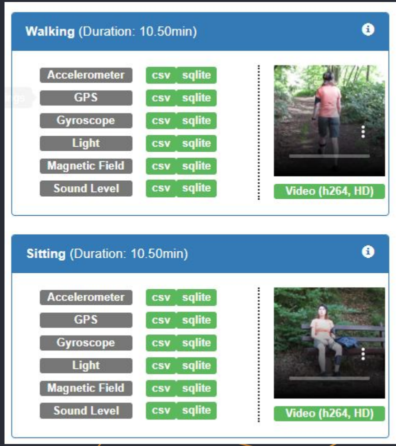




	A	B	C	D	E	F
1	id	attr_time	attr_x	attr_y	attr_z	
2	1	1.44E+12	-2.16077	9.400234	0.565032	
3	2	1.44E+12	-2.17693	9.395446	0.621295	
4	3	1.44E+12	-2.15119	9.382876	0.588974	
5	4	1.44E+12	-2.13503	9.31943	0.545878	
6	5	1.44E+12	-2.16855	9.306262	0.586579	
7	6	1.44E+12	-2.19907	9.32362	0.566827	
8	7	1.44E+12	-2.19309	9.342176	0.518943	
9	8	1.44E+12	-2.16556	9.346365	0.53271	
10	9	1.44E+12	-2.17693	9.347562	0.55725	

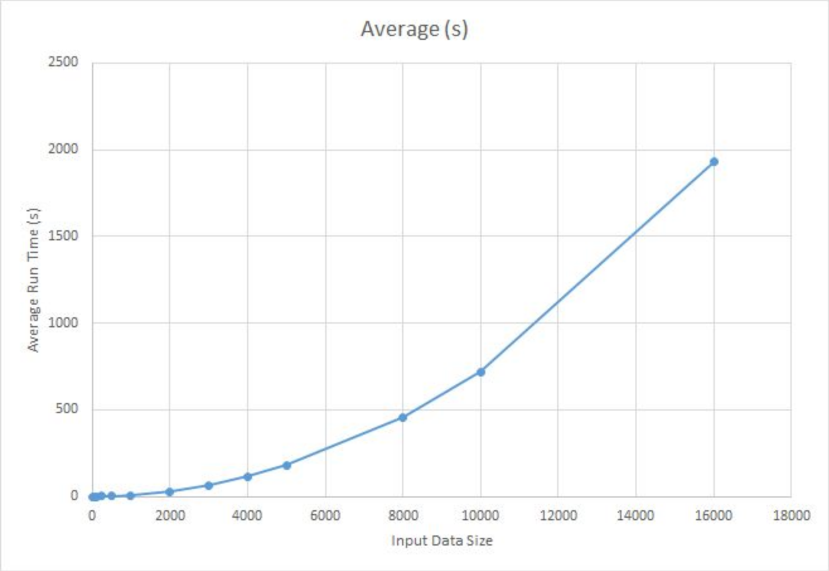
acc_walking_chest



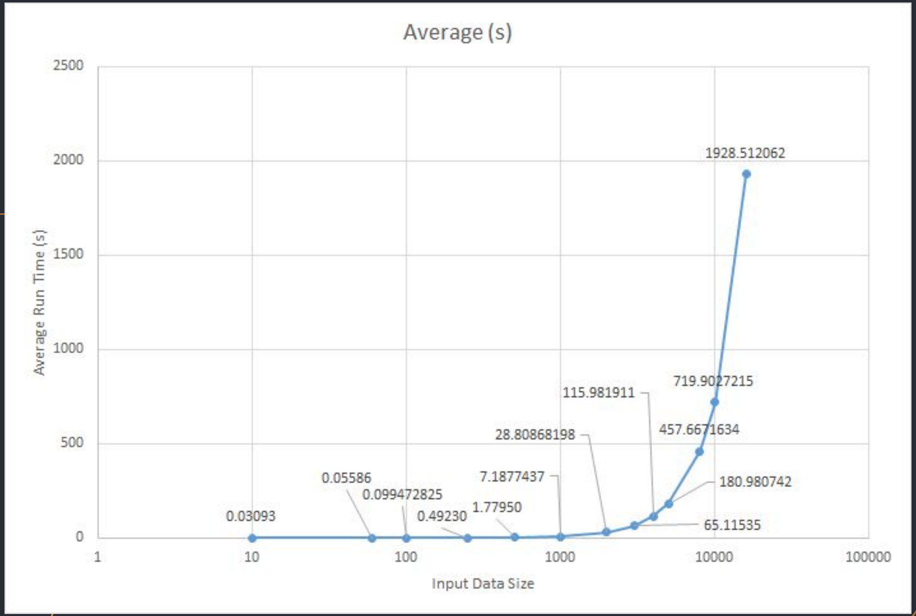


Result

Input Size	Run1 (s)	Run2 (s)	Run3 (s)	Run4 (s)	Average (s)	Log of itself	Meanful log	2 power
10	0.03095	0.03092	0.02992	0.03192	0.03093			
60	0.05388164	0.0538583	0.059841	0.055851	0.05586	3.139701214	0.832537	~64
100	0.09574	0.09574	0.09571	0.11070	0.099473	6.175092707	2.307165	~128
250	0.515125	0.451762	0.479715	0.5226	0.49230	5.870820936	1.853863	~256
500	1.721416	1.7802438	1.881963	1.734388	1.77950	4.01695757	2.014065	~512
1000	7.34934	7.23165	7.09291	7.07708	7.187744	2.002892786	2.002893	~1024
2000	30.67257	28.70869	27.71690	28.13657	28.80868	1.176493933	2.009324	~2048
3000	62.82902	63.09929	64.19027	70.34281	65.11535	0.832830318		
4000	113.64424	119.22028	114.30231	116.76081	115.9819	0.641936375	1.980399	~4096
5000	180.55012	181.41136			180.9807	1.991965783		
8000	455.33249	460.00183			457.6672	2.075117479	2.075117	~8192
10000	713.38994	726.41550			719.9027	#NUM!		
16000	1935.16011	1921.86401			1928.512	#NUM!		~16384



Same Result but in **Logarithmic scale**



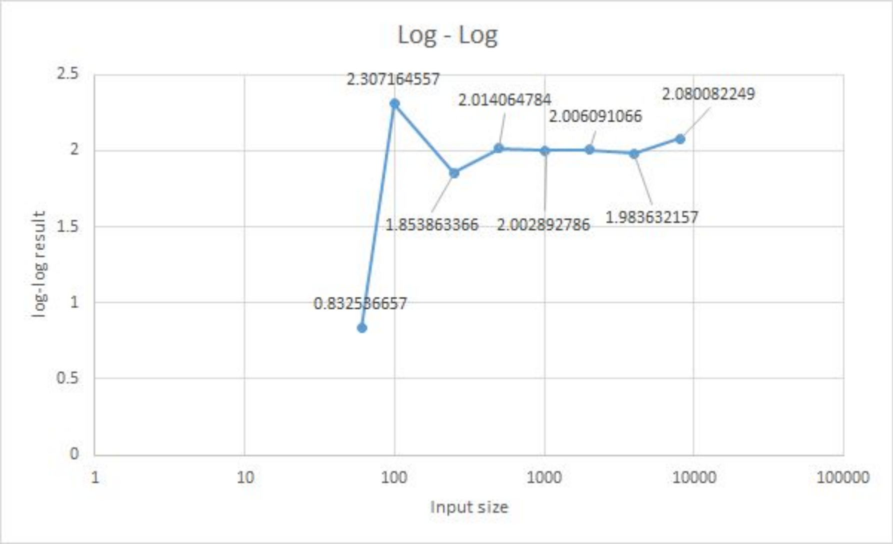
Result of log-log graph

- Meaningful log = Log2(avgtime next/ avgtime current)

Complexity:

O(mn) = O(n^2) time

Input Size	Average Run Time (s)	Meanful log	
10	0.030925324		
60	0.055858073	0.832536657	~64
100	0.099472825	2.307164557	~128
250	0.4923004	1.853863366	~256
500	1.779502775	2.014064784	~512
1000	7.1877437	2.002892786	~1024
2000	28.80868198	2.006091066	~2048
3000	65.11534608		
4000	115.722278	1.983632157	~4096
5000	180.980742		
8000	457.6671634	2.080082249	~8192
10000	719.9027215		
16000	1935.160115		~16384



Reference

- Giorgino, Toni. "Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package." Journal of Statistical Software [Online], 31.7 (2009): 1 - 24. Web. 26 Apr. 2021
- T. Sztyler and H. Stuckenschmidt, "On-body localization of wearable devices: An investigation of position-aware activity recognition," 2016 IEEE International Conference on Pervasive Computing and Communications (PerCom), 2016, pp. 1-9, doi: 10.1109/PERCOM.2016.7456521.
- Rakthanmanon, Thanawin et al. "Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping." Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Association for Computing Machinery

Thank you.

Flaws of DTW

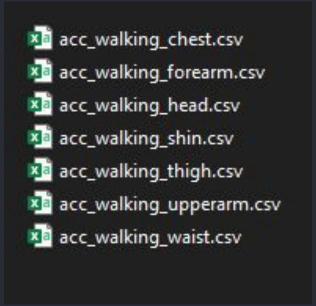
- Different of Size between datasets (eg. 10 vs 100, resulting too much stretching, the result might be meaningless)
- SLOW in real life use

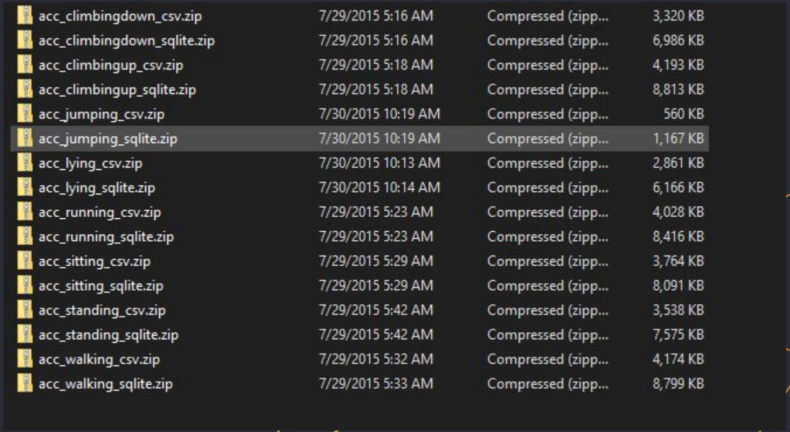
How slow ?

- Compare file wise DTW for 1st object in the HAR dataset:
  - 31950 per value capture in each file,
  - Total 1540 cross file comparison made,
    (1.572 * 10^12 total size)

  - Total Run Time: 5 hr 42min
  - Only for about 10 min of accelerometer data

- Longest test run: **23 hr 12 min**
  - For all senor comparison





acc_climbingdown_csv.zip	7/29/2015 5:16 AM	Compressed (zipp...	3,320 KB
acc_climbingdown_sqlite.zip	7/29/2015 5:16 AM	Compressed (zipp...	6,986 KB
acc_climbingup_csv.zip	7/29/2015 5:18 AM	Compressed (zipp...	4,193 KB
acc_climbingup_sqlite.zip	7/29/2015 5:18 AM	Compressed (zipp...	8,813 KB
acc_jumping_csv.zip	7/30/2015 10:19 AM	Compressed (zipp...	560 KB
acc_jumping_sqlite.zip	7/30/2015 10:19 AM	Compressed (zipp...	1,167 KB
acc_lying_csv.zip	7/30/2015 10:13 AM	Compressed (zipp...	2,861 KB
acc_lying_sqlite.zip	7/30/2015 10:14 AM	Compressed (zipp...	6,166 KB
acc_running_csv.zip	7/29/2015 5:23 AM	Compressed (zipp...	4,028 KB
acc_running_sqlite.zip	7/29/2015 5:23 AM	Compressed (zipp...	8,416 KB
acc_sitting_csv.zip	7/29/2015 5:29 AM	Compressed (zipp...	3,764 KB
acc_sitting_sqlite.zip	7/29/2015 5:29 AM	Compressed (zipp...	8,091 KB
acc_standing_csv.zip	7/29/2015 5:42 AM	Compressed (zipp...	3,538 KB
acc_standing_sqlite.zip	7/29/2015 5:42 AM	Compressed (zipp...	7,575 KB
acc_walking_csv.zip	7/29/2015 5:32 AM	Compressed (zipp...	4,174 KB
acc_walking_sqlite.zip	7/29/2015 5:33 AM	Compressed (zipp...	8,799 KB

How slow ?

```
print ("\n" + "Total number of comparison: " + str(len(final_result)-1))
```

```
Total number of sensor data : 56

Pairwise DTW distance:(low to high)

['name', 0]
['acc_standing_chest.csv vs. acc_standing_upperarm.csv', 672.7254962431782]
['acc_standing_shin.csv vs. acc_sitting_shin.csv', 1337.3869614844023]
['acc_lying_thigh.csv vs. acc_sitting_thigh.csv', 1454.055157626862]
['acc_standing_waist.csv vs. acc_sitting_waist.csv', 1500.4273600488173]
['acc_standing_shin.csv vs. acc_walking_thigh.csv', 1521.1157385703048]
['acc_lying_thigh.csv vs. acc_standing_thigh.csv', 1588.905622584762]
['acc_standing_chest.csv vs. acc_standing_shin.csv', 1658.618210077129]
['acc_walking_forearm.csv vs. acc_climbingup_forearm.csv', 1675.884265165052]
['acc_standing_head.csv vs. acc_walking_head.csv', 1713.8939870666318]
['acc_standing_chest.csv vs. acc_walking_upperarm.csv', 1755.3873267659974]
['acc_standing_shin.csv vs. acc_standing_upperarm.csv', 1821.2167763926864]
['acc_standing_shin.csv vs. acc_walking_upperarm.csv', 1845.5824112914459]
['acc_lying_thigh.csv vs. acc_lying_chest.csv', 1859.9768576225708]
['acc_sitting_chest.csv vs. acc_walking_thigh.csv', 1873.0156520861428]
['acc_standing_chest.csv vs. acc_sitting_shin.csv', 1875.5304303774747]
['acc_walking_head.csv vs. acc_sitting_head.csv', 1878.8742487601357]
['acc_walking_upperarm.csv vs. acc_standing_upperarm.csv', 1899.5446334021453]
['acc_standing_thigh.csv vs. acc_walking_chest.csv', 1953.241683857029]
['acc_standing_upperarm.csv vs. acc_sitting_shin.csv', 2022.9804906022143]
['acc_standing_shin.csv vs. acc_sitting_upperarm.csv', 2031.251319157789]
['acc_standing_head.csv vs. acc_sitting_head.csv', 2044.5335326062038]
['acc_sitting_thigh.csv vs. acc_lying_chest.csv', 2094.6645568965027]
['acc_standing_chest.csv vs. acc_walking_thigh.csv', 2099.913899885918]
['acc_standing_upperarm.csv vs. acc_walking_chest.csv', 2106.1649510507013]
['acc_walking_thigh.csv vs. acc_walking_upperarm.csv', 2122.712328383684]
['acc_standing_shin.csv vs. acc_sitting_chest.csv', 2125.1952452934834]
['acc_walking_waist.csv vs. acc_standing_waist.csv', 2131.7259606736206]
['acc_walking_thigh.csv vs. acc_sitting_shin.csv', 2146.876316143935]
['acc_walking_upperarm.csv vs. acc_sitting_shin.csv', 2152.1975526920805]
['acc_walking_thigh.csv vs. acc_sitting_upperarm.csv', 2237.4545408251934]
['acc_standing_chest.csv vs. acc_walking_chest.csv', 2250.075614479392]
```

Some ways to speed it up:

- Add a “Window” to the whole matrix, eliminate if path out of window

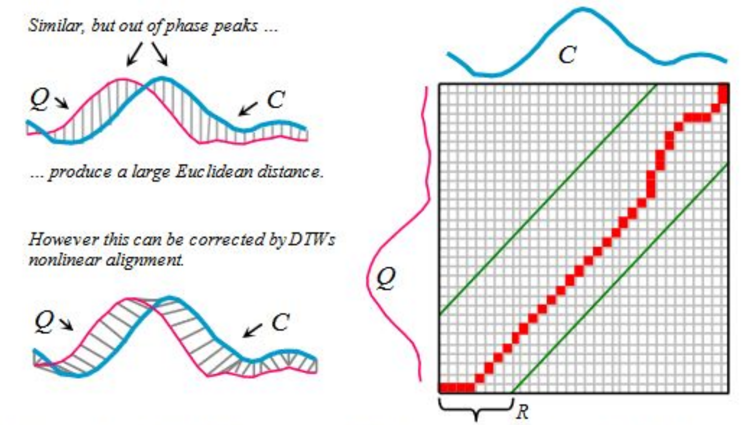


Figure 3: *left*) Two time series which are similar but out of phase. *right*) To align the sequences we construct a warping matrix, and search for the optimal warping path (red/solid squares). Note that Sakoe-Chiba Band with width *R* is used to constrain the warping path

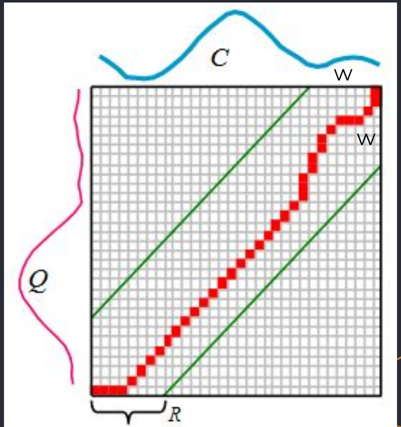
Pseudocode

```
DTW Distance(s: array [1..n], t: array [1..m], w: int) {
    DTW := array [0..n, 0..m]

    w := max(w, abs(n-m)) // adapt window size (*)

    for i := 0 to n
        for j:= 0 to m
            DTW[i, j] := infinity
    DTW[0, 0] := 0
    for i := 1 to n
        for j := max(1, i-w) to min(m, i+w)
            DTW[i, j] := 0

    for i := 1 to n
        for j := max(1, i-w) to min(m, i+w)
            cost := d(s[i], t[j])
            DTW[i, j] := cost + minimum(DTW[i-1, j  ],    // insertion
                                        DTW[i  , j-1],    // deletion
                                        DTW[i-1, j-1])    // match

    return DTW[n, m]
}
```

Complexity:

O(w(m+n-w)) time

O(w(m+n-w)) space



Possible ways to improve performance - fastdtw

Salvador, Stan & Chan, Philip. (2004). Toward Accurate Dynamic Time Warping in Linear Time and Space. Intelligent Data Analysis. 11. 70-80.

Thank you.