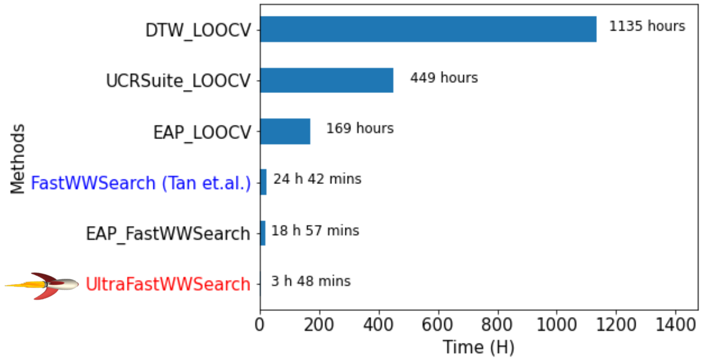
# Ultra Fast Warping Window Optimization for Dynamic Time Warping



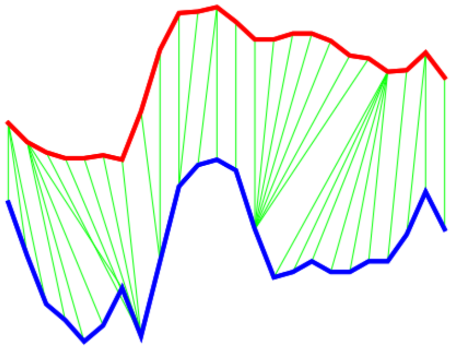

CW. Tan M. Herrmann G.I. Webb

MONASH University
Information Technology



ICDM 2021
IEEE International Conference on Data Mining
7 – 10 DECEMBER 2021
AUCKLAND NEW ZEALAND

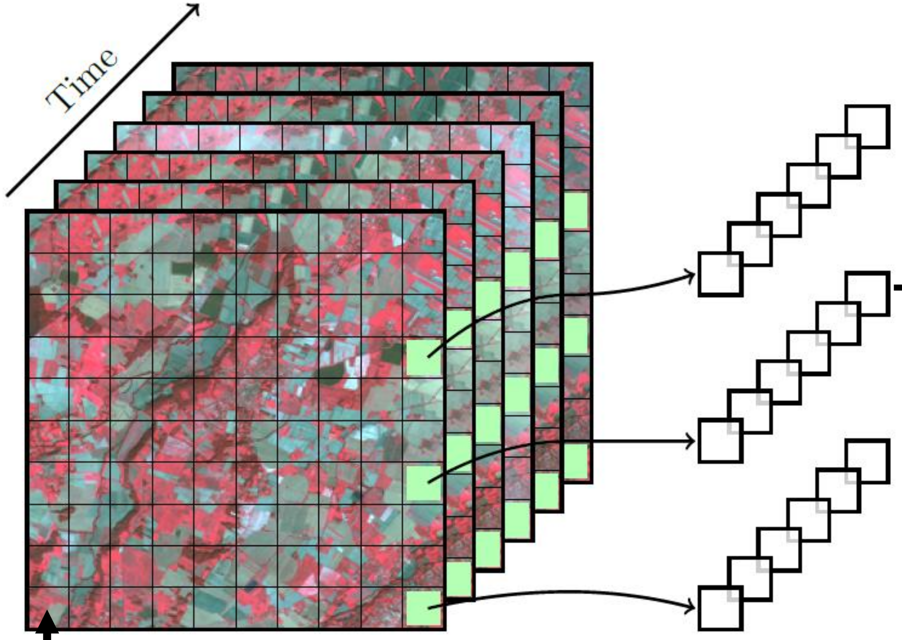



# Outline

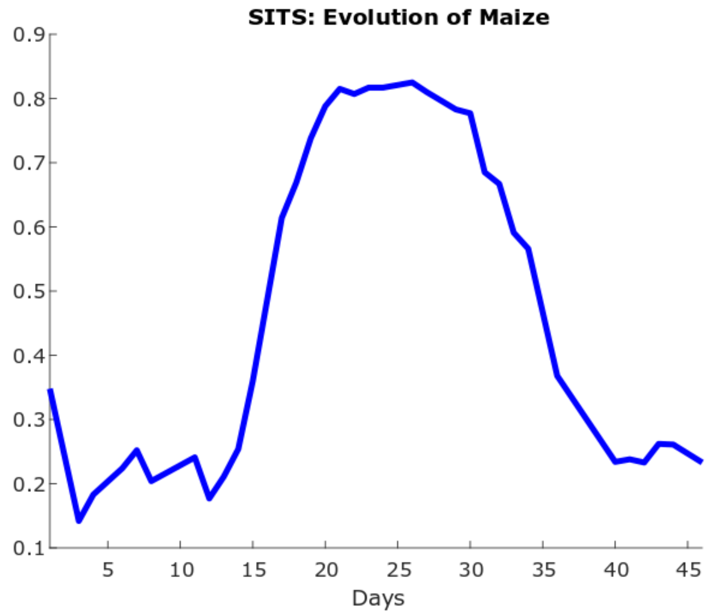
1. Introduction
2. Why learn the best warping window?
3. How to learn the warping window?
4. Current state of the art
5. Our method – Ultra Fast Warping Window Search
6. Results
7. Conclusion and future work

# What is a Time Series?

- Collection of observations made sequentially, more intuitive **visually**
- Many data can be transformed into time series → **Satellite Image Time Series**




**0.348,0.245,0.142,0.183,0.203,**
**0.224,0.252,0.204,0.216,0.229,**
**0.241,0.177,0.211,0.254,0.360,**
**0.487,0.614,0.669,0.738,0.788,**
**0.815,0.807,0.817,0.817,0.821,**
**0.825,0.810,0.796,0.783,0.777,**
**0.685,0.667,0.591,0.566,0.467,**
**0.368,0.335,0.301,0.268,0.234,**
**0.238,0.233,0.262,0.261,0.247,**
**0.233**




Satellite Images

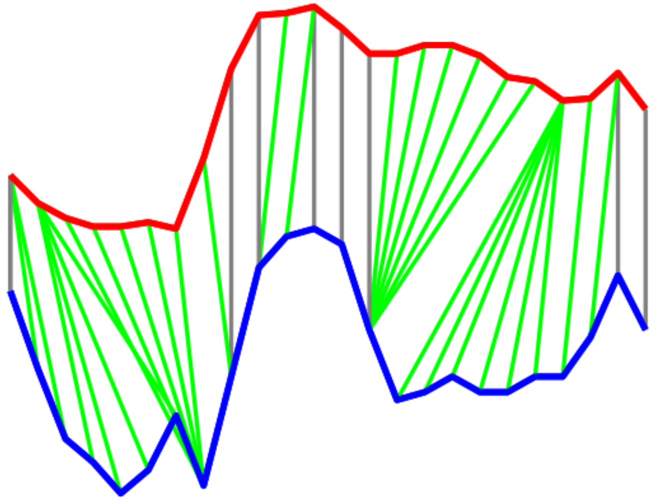
Array of numbers

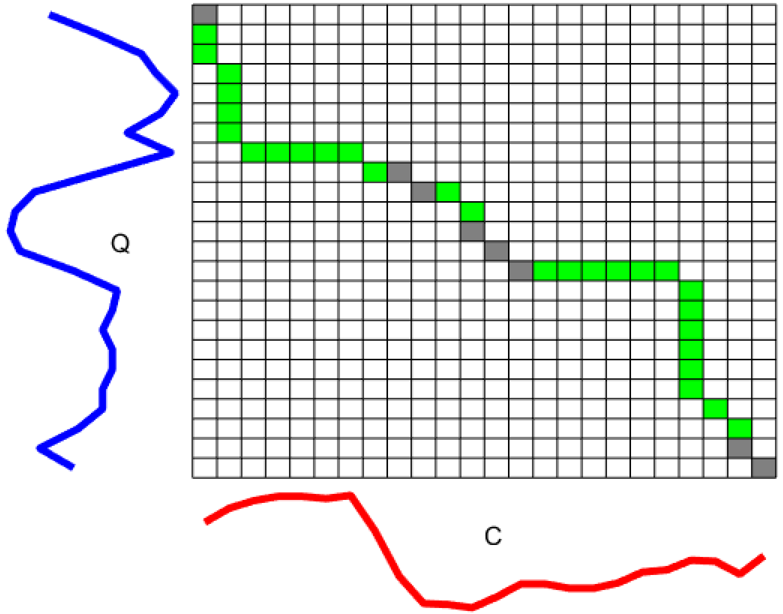
Time series

Every pixel represents a geographic area (Lat, Lon) on Earth

# Dynamic Time Warping

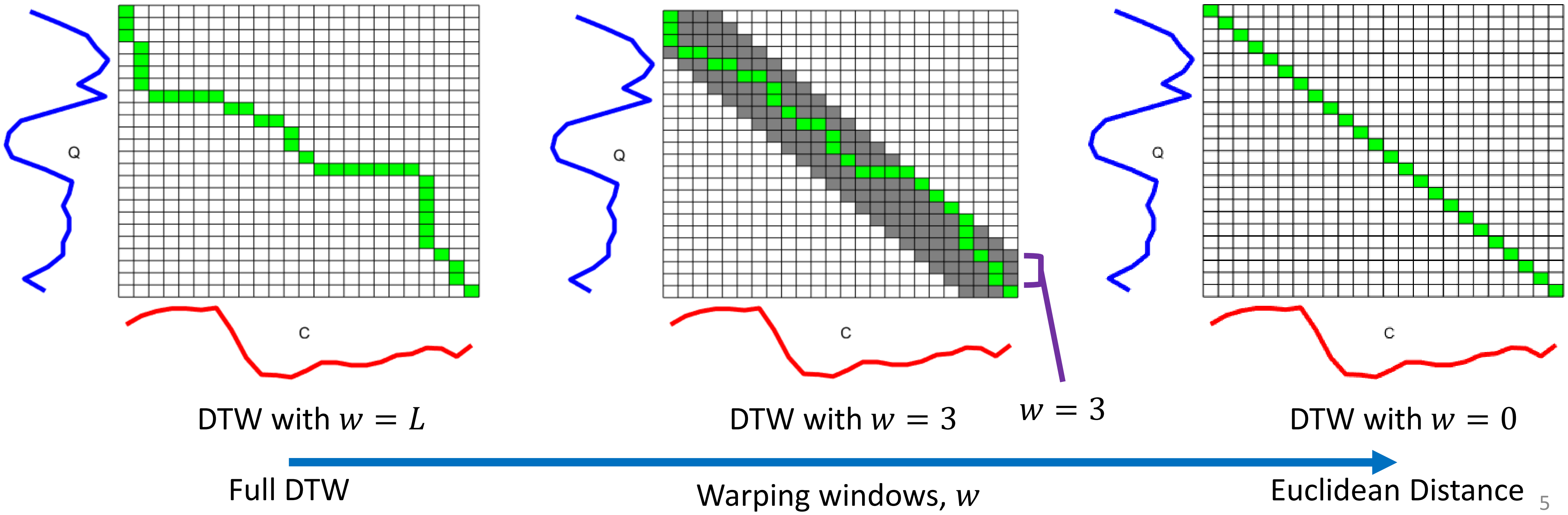
- a.k.a. **DTW** – similarity function to align time series $O(L^2)$
  - Every possible alignment of $Q$ and $C$ is a warping path, $\vec{p}$
    $$\vec{p} = [w_1, \ldots, w_K]$$
  - $w_k = (i, j)$ represents an association of $q_i \leftrightarrow c_j$ aligned by DTW
  - $\mathrm{DTW}(Q, C)$ finds the cheapest warping path ("*best*")
- Used in many fields: Classification, Regression, Clustering, …
  - Nearest Neighbour Algorithm (NN-DTW)





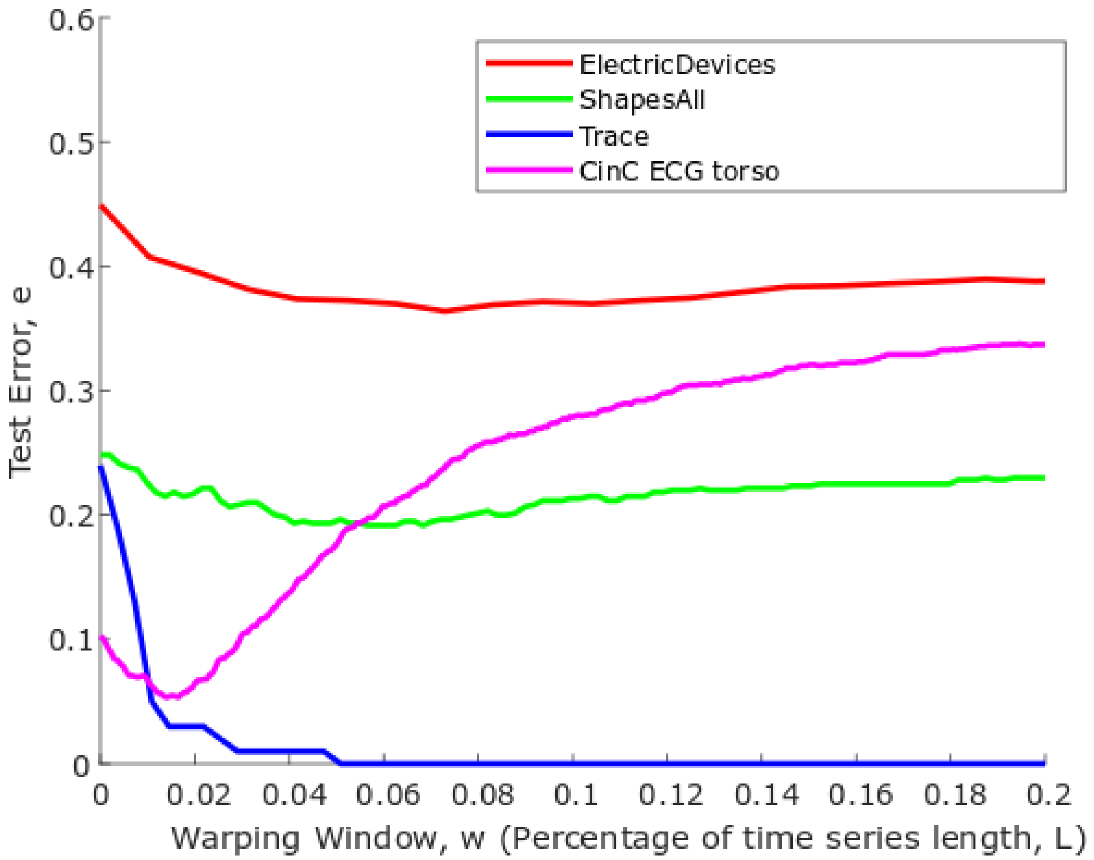
# Warping Window

- Warping Window, $w$ is a global constraint on the alignment of DTW such that the elements of $Q$ and $C$ can only be mapped if they are less than $w$ apart, $w = \{0, \dots, L\}$



# Why learn the best warping window?

- **Strong** influence on accuracy
  - On **CinC ECG torso** dataset, error rate reduced from 35% to 7%
  - NN-DTW with learnt warping window performs better
- **Speedup** DTW
  - Smaller $w$ means we don't need to compute the full DTW matrix
  - Reduce the complexity down to $O(w \cdot L)$




[1] Tan, C. W., Herrmann, M., Forestier, G., Webb, G. I., & Petitjean, F. (2018, May). Efficient search of the best warping window for dynamic time warping. In *Proceedings of the 2018 SIAM International Conference on Data Mining* (pp. 225-233). Society for Industrial and Applied Mathematics.

# How to learn the best warping window?

```
for  w = 0 to L  do
        error = 0
        for each  s in T  do
                nn_s = nn_search(s, T\s, w)
                if  nn_s.class ≠ s.class  then  error++

        if  error < bestError  then
                bestWW = w
                bestError = error
```

**Parameter** to NN-DTW algorithm

***Leave One Out*** Cross Validation (LOO-CV)

Naïvely learns the best warping window requires $O(N^2L^3)$ operations [1]

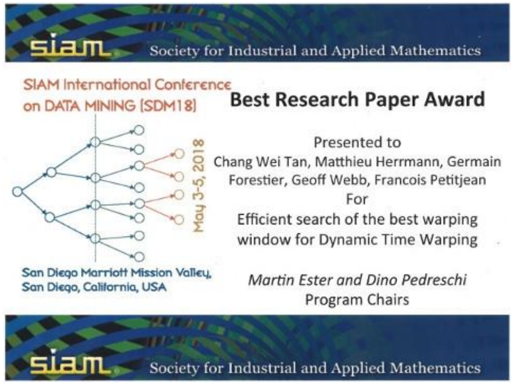
[1] Tan, C. W., Herrmann, M., Forestier, G., Webb, G. I., & Petitjean, F. (2018, May). Efficient search of the best warping window for dynamic time warping. In *Proceedings of the 2018 SIAM International Conference on Data Mining* (pp. 225-233). Society for Industrial and Applied Mathematics.

# Fast Warping Window Search

- Efficiently fill up a NN table, giving the nearest neighbour of every time series for all windows
- Naïvely create the table using DTW, requires $\theta(N^2L^3)$ operations

Prior approaches typically go from **smallest** to **largest** with a subset of windows

| | Nearest neighbor at warping windows | | | | |
|---|---|---|---|---|---|
| | 0 | 1 | $\cdots$ | $L-2$ | $L-1$ |
| $T_1$ | $T_{24}(2.57)$ | $T_{55}(0.98)$ | $\cdots$ | $T_{55}(0.98)$ | $T_{55}(0.98)$ |
| $\vdots$ | | | $\vdots$ | | |
| $T_N$ | $T_{60}(4.04)$ | $T_{47}(1.61)$ | $\cdots$ | $T_{47}(1.61)$ | $T_{47}(1.61)$ |



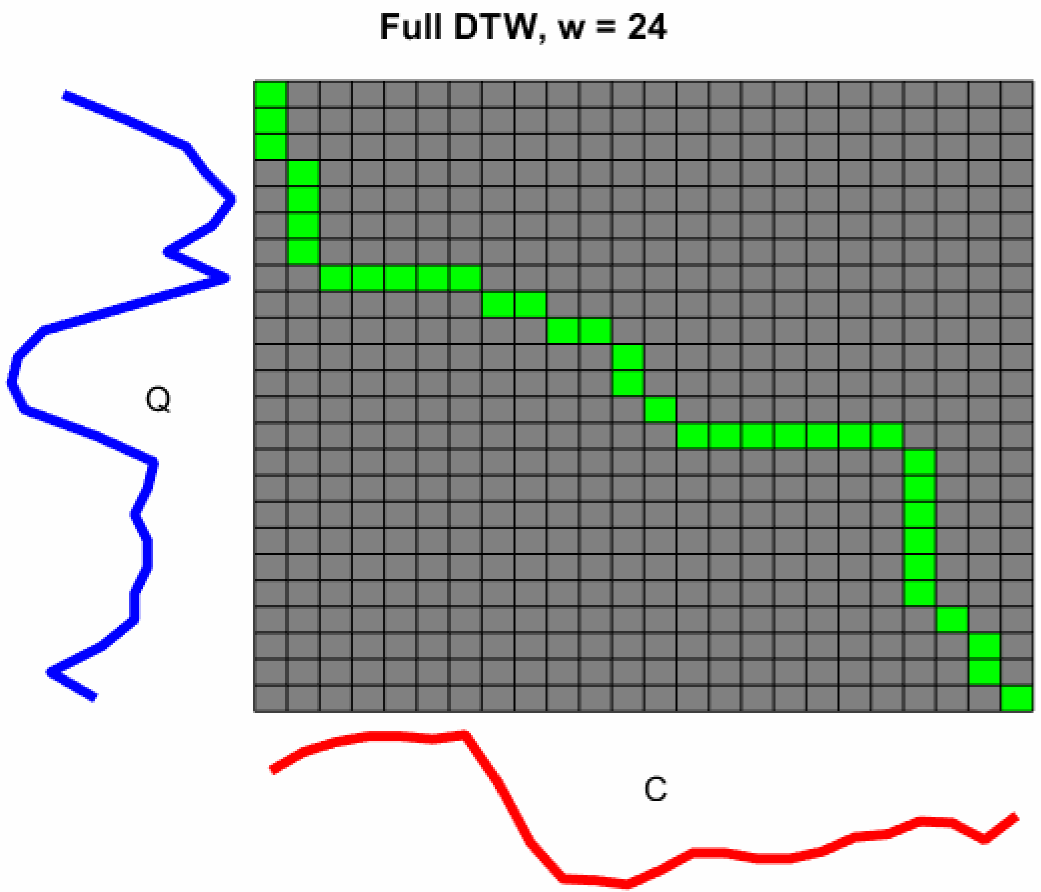

**FastWWS** goes from **largest** to **smallest**, fast enough to test all windows

# FastWWS Intuition

Warping path can be valid for several windows

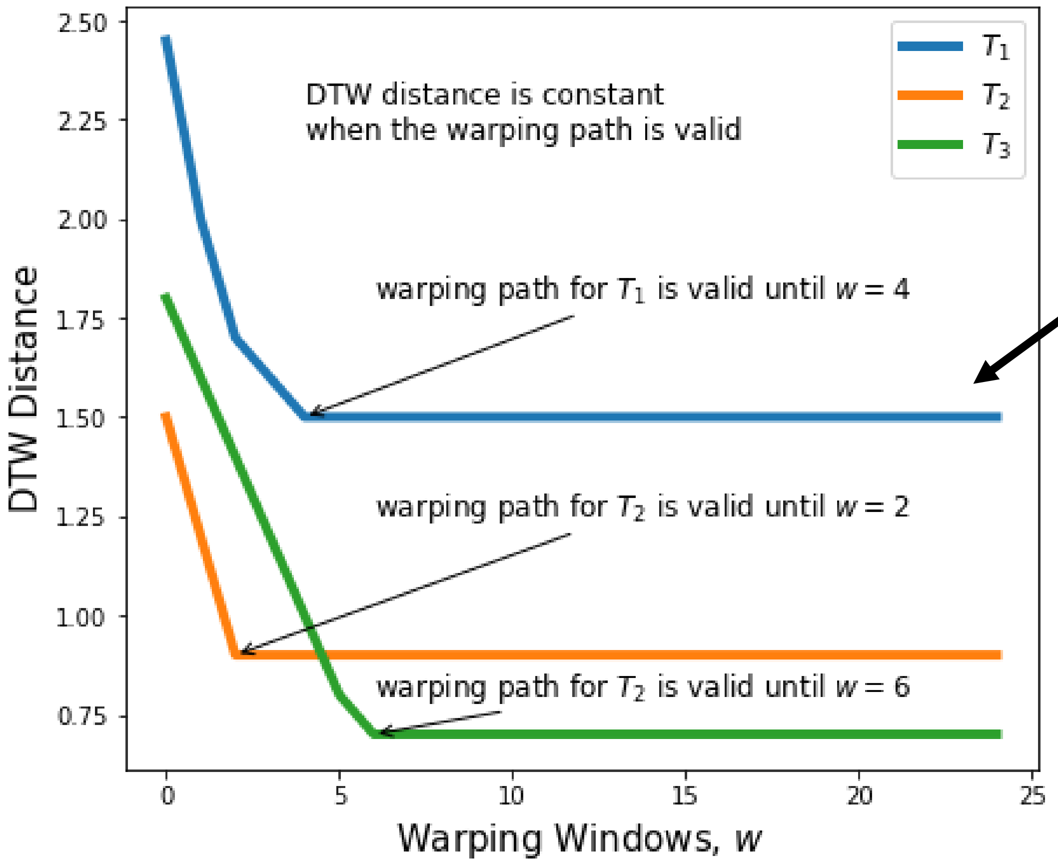
- $w$ has a "validity"
- skip computations of all valid $w$
- Example:
  - Warping path is valid to $w = 6$
  - $\text{DTW}_{24}(Q, C) = \text{DTW}_6(Q, C)$
  - Skip all DTW from $w = [24, \ldots, 6]$

| $\mathbf{w}$ | … | 4 | 5 | 6 | 7 | … | 23 | 24 |
|---|---|---|---|---|---|---|---|---|
| $\mathbf{DTW_w(Q, C)}$ | … | 8.82 | 8.36 | 8.04 | 8.04 | … | 8.04 | 8.04 |



# FastWWS Intuition

- **FastWWS** goes from largest to smallest, applies to all or a subset of windows




- Large window validity for $\mathrm{DTW}_L$ (Most of the time)
- No bounds are necessary
- DTW has not changed

Thus obtain $\mathrm{DTW}_{w+k}$ and/or $\mathrm{LB\_Keogh}_{w+k}$ for **"FREE"** as the lower bound for $\mathrm{DTW}_w$

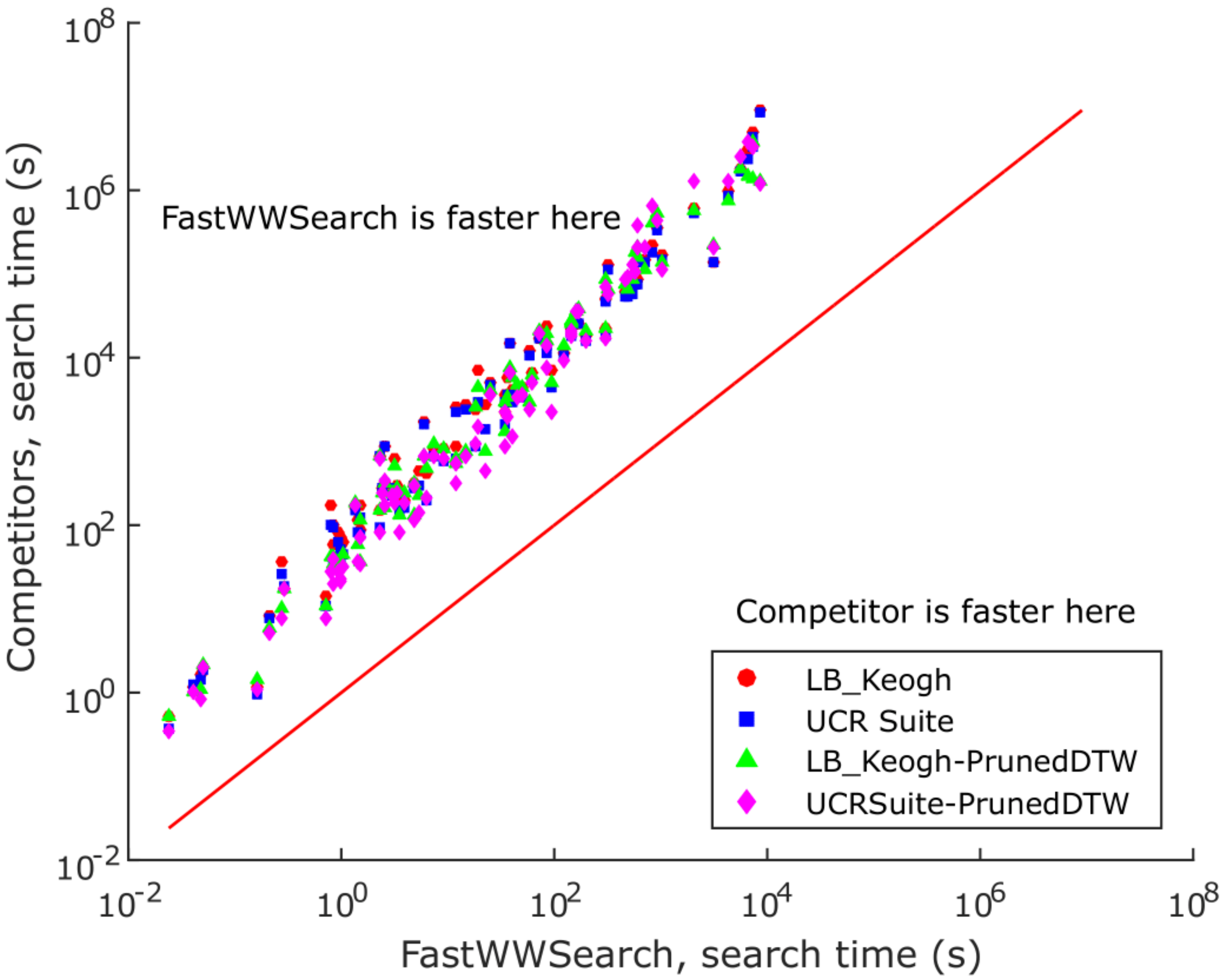
**<u>Tighter bounds for pruning</u>**

$$\mathrm{DTW}_w(Q, C) \geq \mathrm{DTW}_{w+1}(Q, C)$$

$$\mathrm{LB\_Keogh}_w(Q, C) \geq \mathrm{LB\_Keogh}_{w+1}(Q, C) \geq \mathrm{LB\_Kim}(Q, C)$$

Only use the value at $w + k$ when available, no point in computing $\mathrm{DTW}_{w+k}$ for $\mathrm{DTW}_w$

# **FastWWS** is **FASTER** and more **EFFICIENT** than previous methods!




**Average speed up**

On the $N^2$ Term

$N \leq 200 \Rightarrow 106\text{x}$

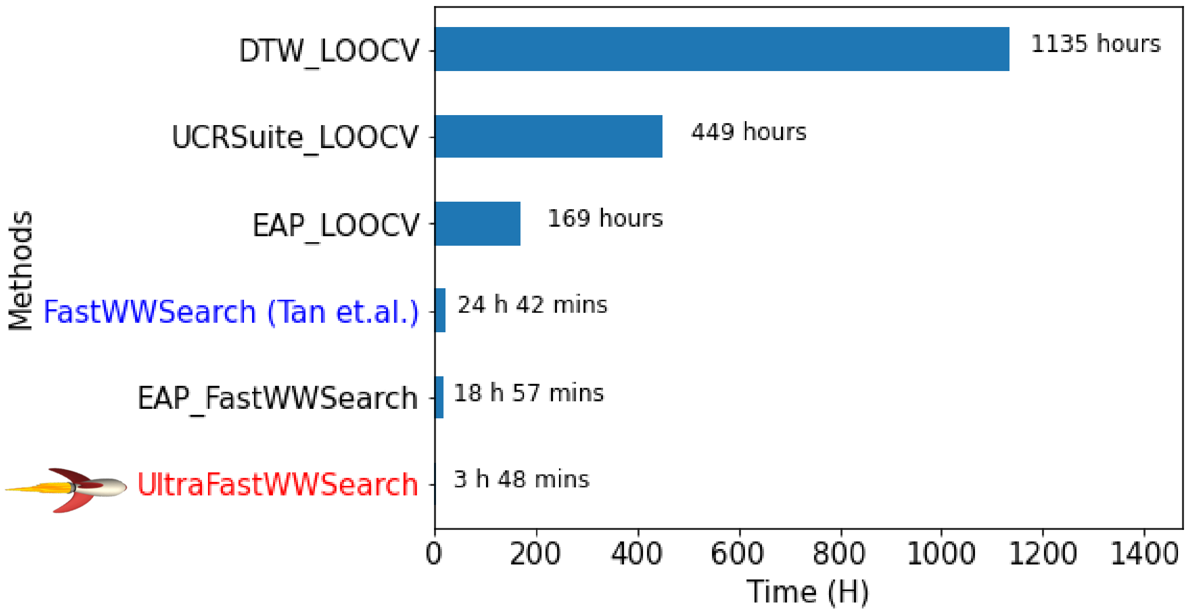
$\boldsymbol{N > 200 \Rightarrow 184\text{x}}$

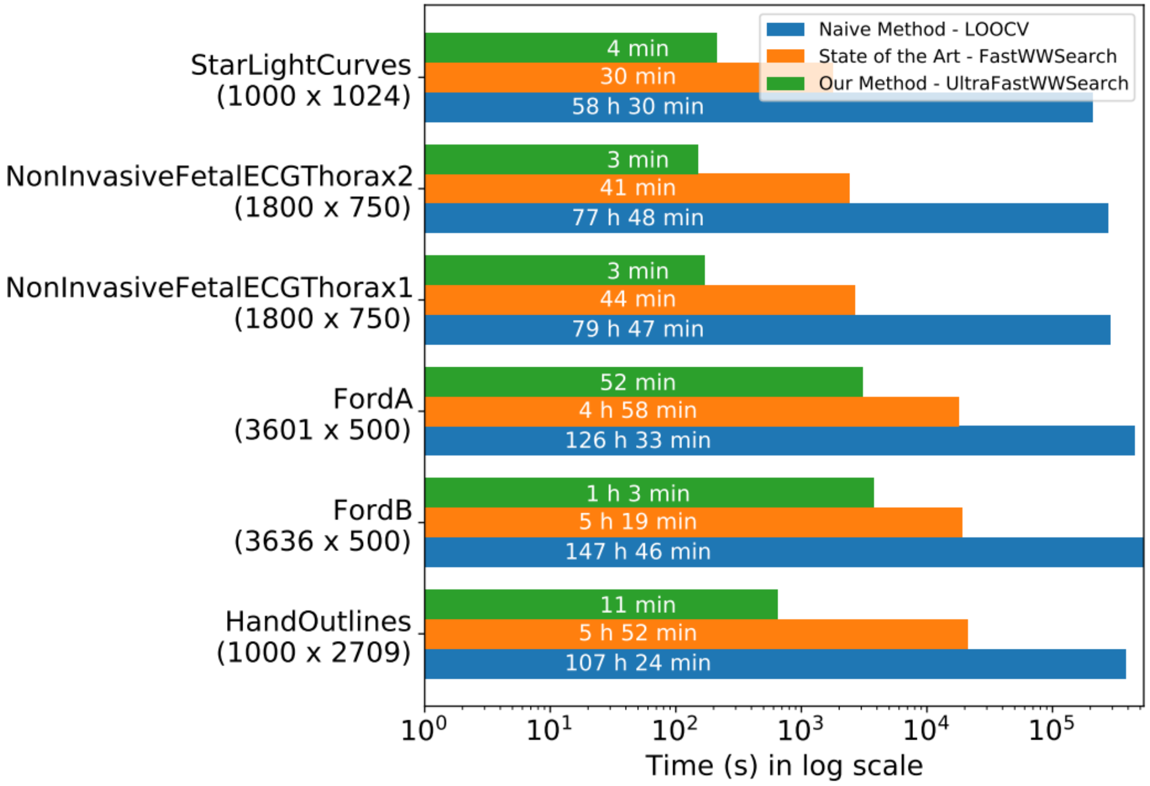
On the $L^3$ Term

$L \leq 300 \Rightarrow 67\text{x}$

$\boldsymbol{L > 300 \Rightarrow 250\text{x}}$

# FastWWS is still slow



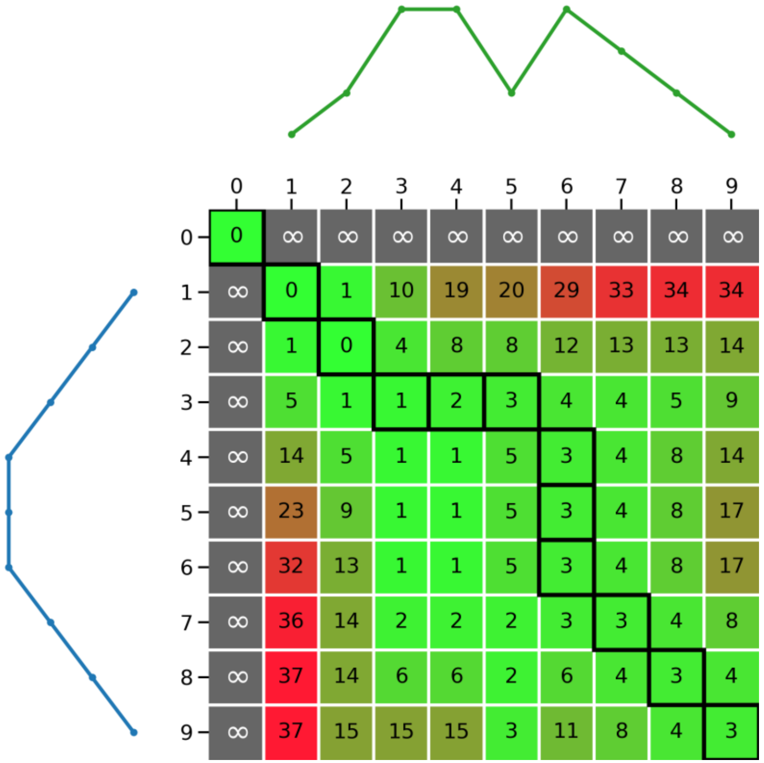



# Ultra Fast Warping Window Search for DTW

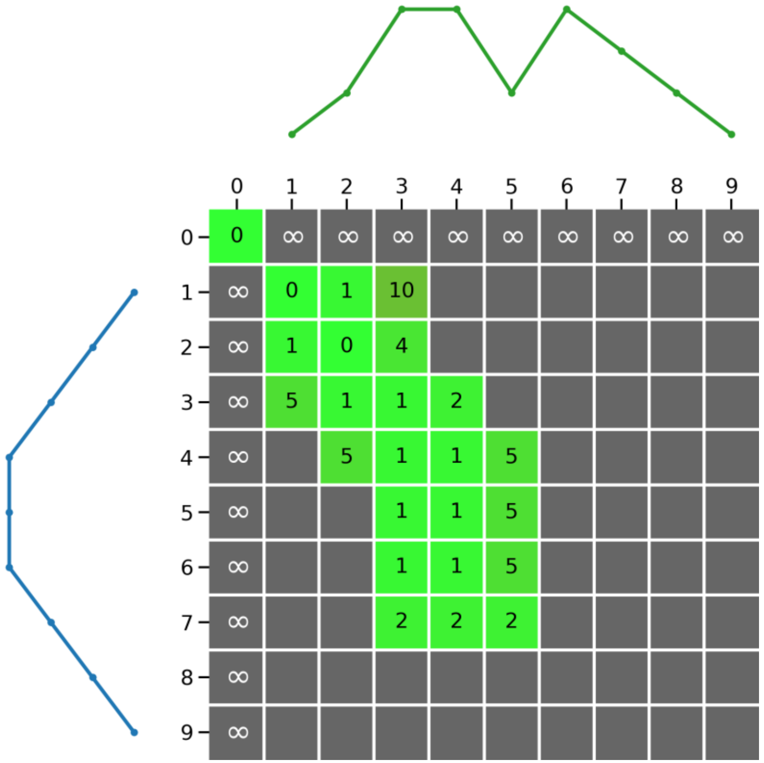
- Built upon FastWWSearch
- Inspired by EAP-DTW [1] to speed up FastWWSearch
- 6 primary strategies
  1. Replaces DTW with a more efficient EAP-DTW
  2. Uses early abandoning of EAP-DTW
  3. **Establishes tighter upper bound for EAP-DTW**
  4. **Ordering the time series to best exploit the efficient pruning and early abandoning power of EAP-DTW**
  5. Removes DTW lower bounds, which 1-4 render redundant
  6. Using the window validity to skip most window sizes altogether

[1] Herrmann, M., Webb, G.I. Early abandoning and pruning for elastic distances including dynamic time warping. *Data Min Knowl Disc* **35,** 2577–2601 (2021).

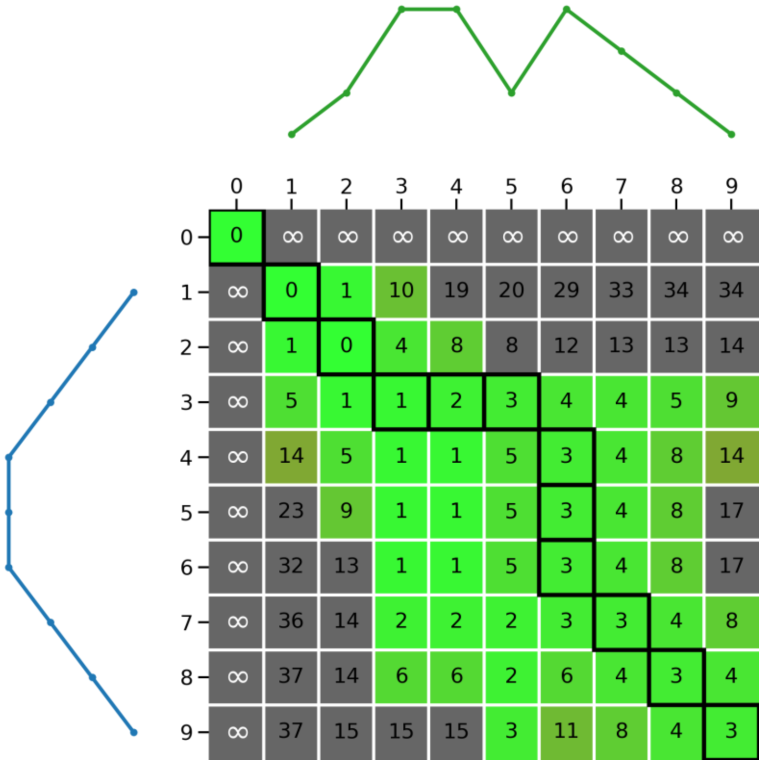
# Early Abandoning and Pruning




Original DTW cost matrix
81 computations




Early abandoned DTW cost matrix
23 computations
cut-off = 1




Pruned DTW cost matrix
57 computations
cut-off = 5

# Tighter Upper Bounds for EAP-DTW

- Minimise $DTW_L$ - the most expensive operation of UltraFastWWS

ensures that we always and only calculate the actual $DTW$ when we need to

$$\mathrm{UB} = \begin{cases} \max(\mathrm{NN}_w^Q, \mathrm{NN}_w^C), & \mathrm{NN}_w^Q \text{ and } \mathrm{NN}_w^C \text{is computed} \\ \mathrm{EuclideanDistance}(Q, C), & \mathrm{NN}_w^Q \text{ and } \mathrm{NN}_w^C \text{not computed} \end{cases}$$

Euclidean Distance is a better upper bound for $DTW$ than the commonly used $\infty$.

It corresponds to the parameter $w = 0$, that gives the largest distance between $(Q, C)$

# Filling the NN table

Specific ordering to exploit efficient pruning and early abandoning power of EAP-DTW

- Minimize the upper bound used in each call to $DTW$
- More productive to initially pair with $T_T$ with large NN distance

| $w$ | 0 | 1 | 2 | 3 | 4 | 5 | … | $L-3$ | $L-2$ | $L-1$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $T_1$ | $T_3(2.21)$ | $T_3(1.89)$ | $T_2(1.57)$ | $T_2(1.35)$ | $T_2(1.22)$ | $T_2(0.98)$ | $T_2(0.98)$ | $T_2(0.98)$ | $T_2(0.98)$ | $T_2(0.98)$ |
| $T_2$ | $T_5(2.78)$ | $T_5(2.54)$ | $T_5(2.35)$ | $T_5(2.14)$ | $T_5(1.92)$ | $T_5(1.92)$ | $T_5(1.92)$ | $T_5(1.92)$ | $T_5(1.92)$ | $T_5(1.92)$ |
| $T_3$ | $T_4(1.15)$ | $T_4(1.01)$ | $T_4(0.81)$ | $T_4(0.59)$ | $T_4(0.34)$ | $T_4(0.19)$ | $T_4(0.19)$ | $T_4(0.19)$ | $T_4(0.19)$ | $T_4(0.19)$ |
| $T_4$ | $T_2(1.09)$ | $T_2(0.99)$ | $T_2(0.76)$ | $T_2(0.56)$ | $T_2(0.56)$ | $T_2(0.56)$ | $T_2(0.56)$ | $T_2(0.56)$ | $T_2(0.56)$ | $T_2(0.56)$ |
| $T_5$ | $T_1(2.51)$ | $T_1(2.46)$ | $T_1(2.37)$ | $T_1(2.21)$ | $T_3(1.83)$ | $T_3(1.83)$ | $T_3(1.83)$ | $T_3(1.83)$ | $T_3(1.83)$ | $T_3(1.83)$ |
| $S$ | | | | | | | | | | |

4. Keep track of the candidate with the smallest NN distance at $w+1$ and start $w$ from this candidate

3. Process in ascending order

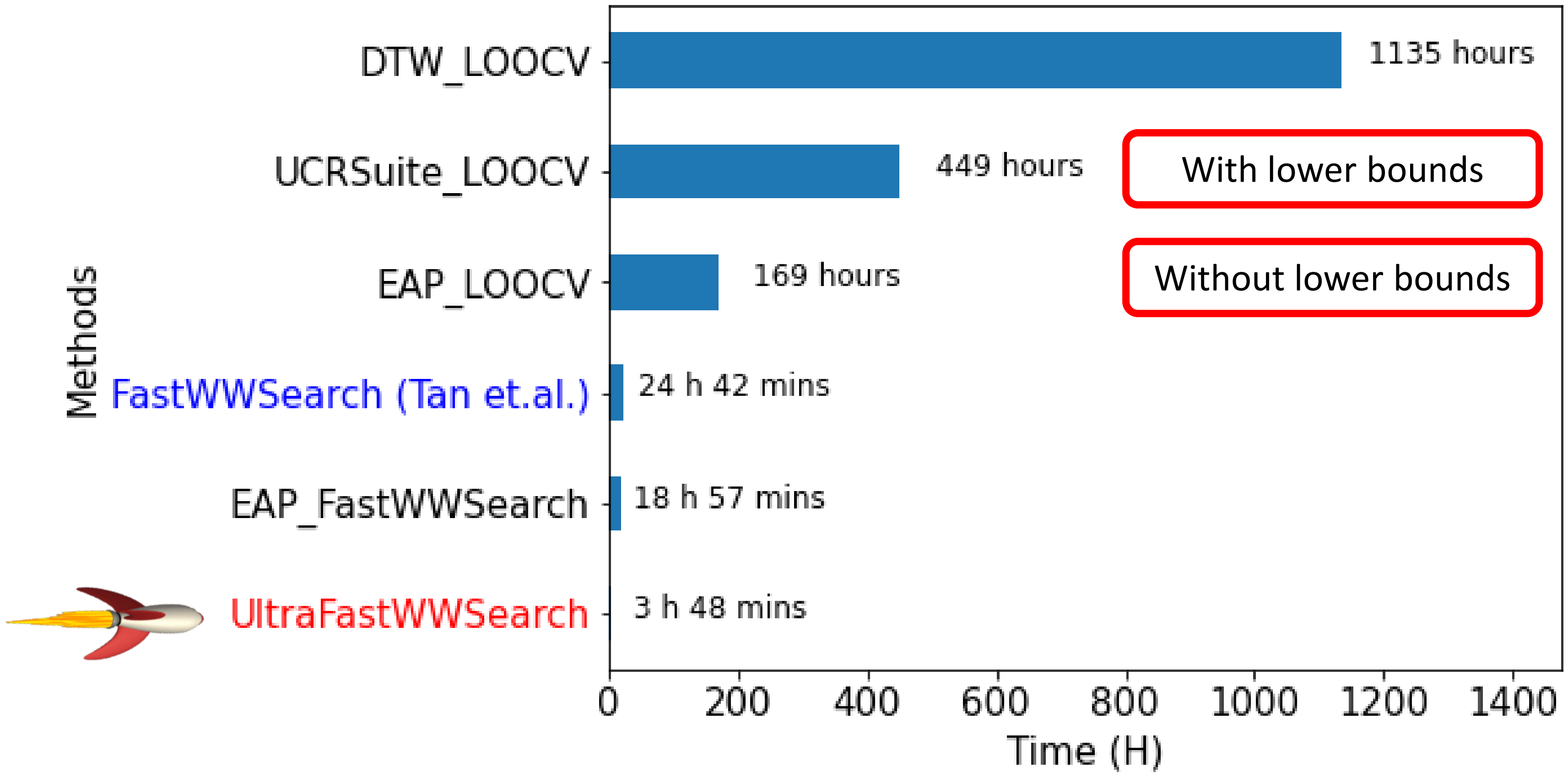
2. Skip to the maximum window validity

1. Process in descending order

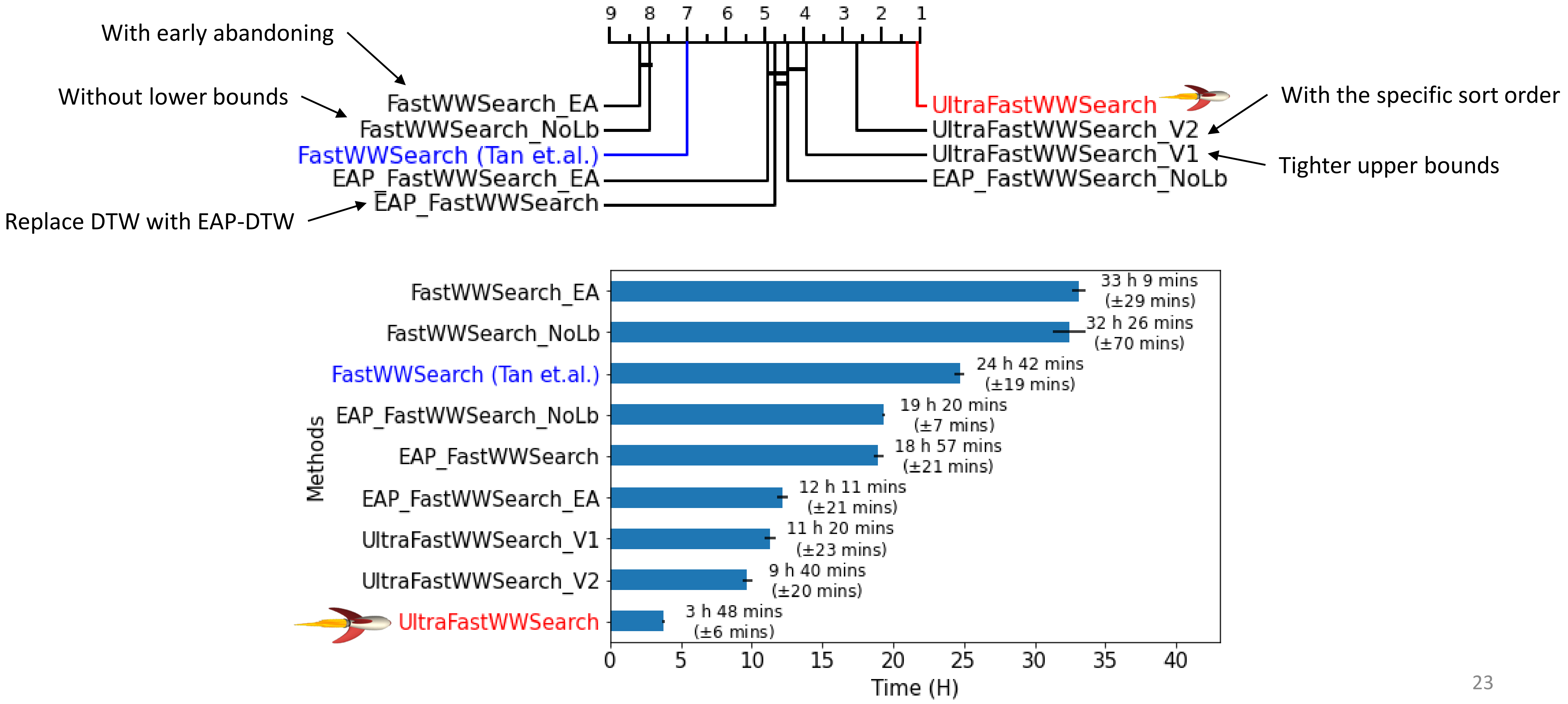
# Experimental Evaluation

- Evaluate the efficiency of UltraFastWWS 🚀
    1. LOOCV with DTW
    2. LOOCV with EAP-DTW
    3. LOOCV with UCR Suite
    4. FastWWS with DTW
    5. FastWWS with EAP-DTW
- Exhaustive search over 101 window sizes on all methods
- Average results over 5 runs for different reshuffling of $T$
- 128 benchmark time series datasets

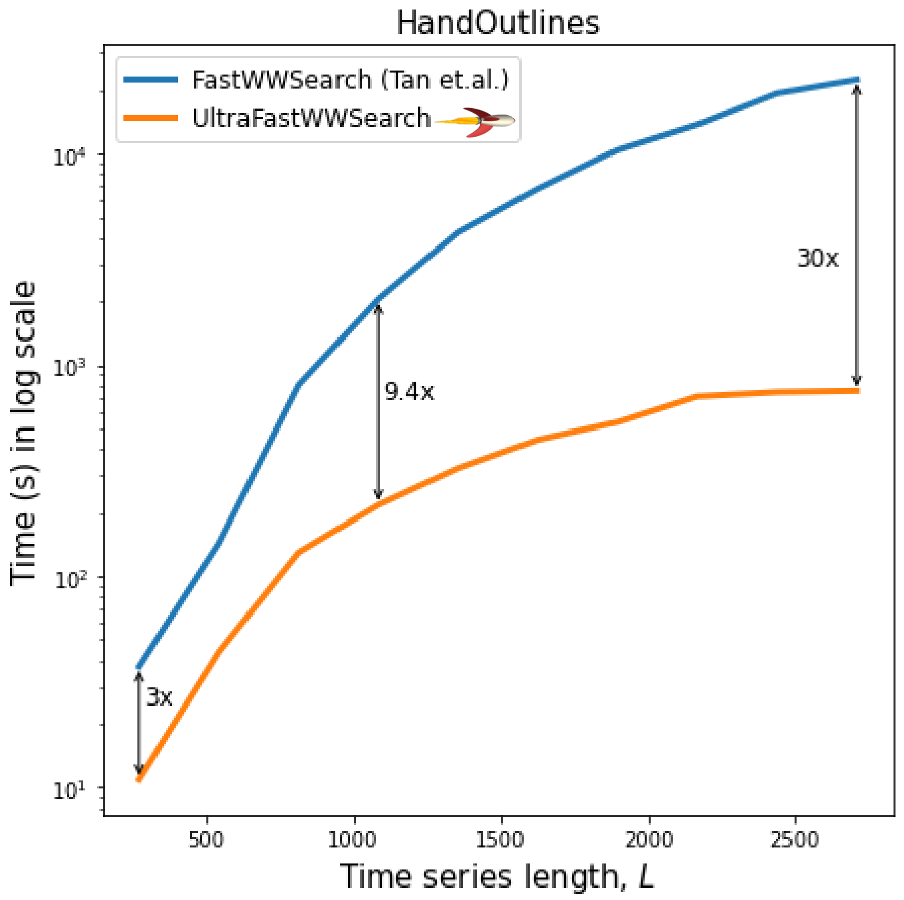
# Speeding up State of the Art

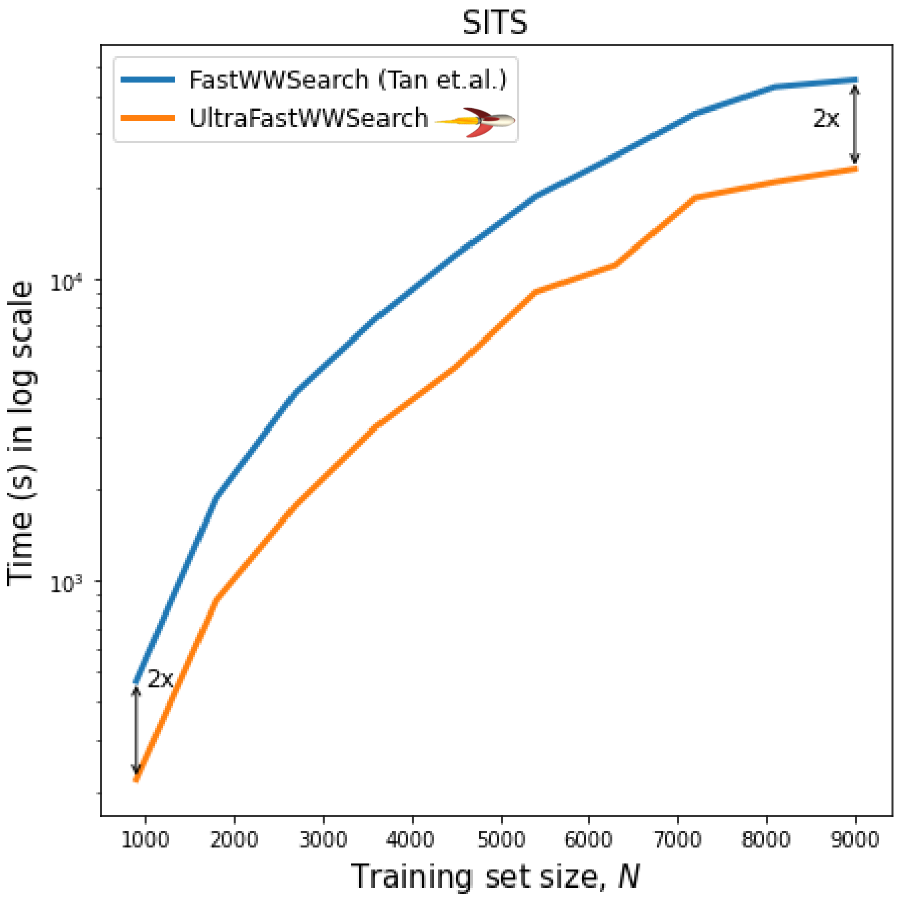


# Ablation Study



# Scalability to large and long datasets



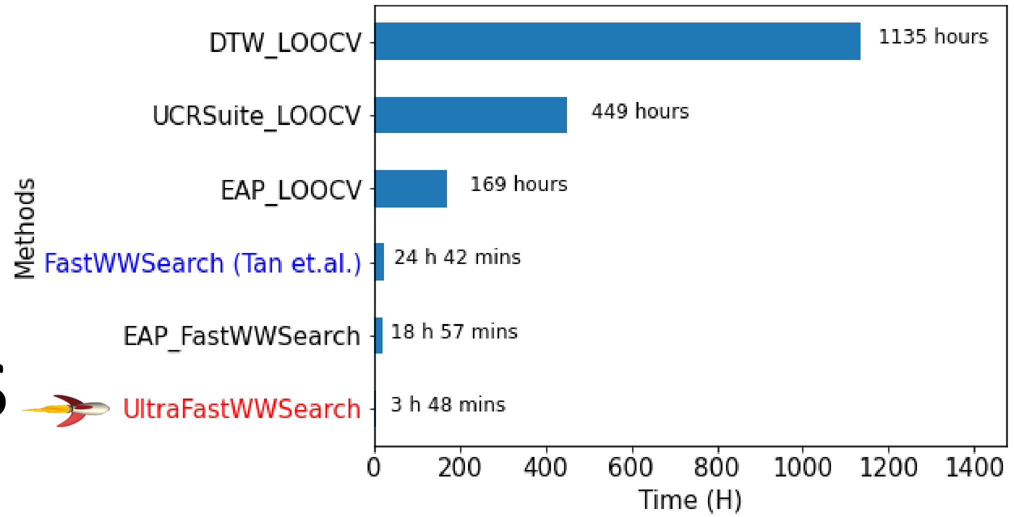
# Conclusion and Future work

- An exact algorithm to speedup the search for the best parameter (warping window) for NN-DTW
  - **UltraFastWWS** is more **EFFICIENT** and **FASTER**
  - **UltraFastWWS** completes the training on all 128 datasets within 4 hours, a time comparable to one of the most scalable TSC algorithm (ROCKET [1])
- Improve scalability and accuracy of Ensemble of Elastic Distances

- Our results and source code are online at
  - https://github.com/ChangWeiTan/UltraFastWWS
  - Slides: https://changweitan.com/research/ICDM21-slides.pdf

[1] Dempster, A., Petitjean, F., & Webb, G. I. (2020). ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, *34*(5), 1454-1495.

# Thank you!
# Questions and Answers




CW. Tan

M. Herrmann

G.I. Webb

chang.tan@monash.edu

github.com/ChangWeiTan/UltraFastWWS