**PROPERTIES**

* ﻿It is **exact**: the Matrix Profile based methods provide no false positives or false dismissals. It can handle missing data: Even in the presence of missing data, we can provide answers which are guaranteed to have no false negatives.
* It is **simple and parameter-free**: In contrast, the more general algorithms in this space that typically require building and tuning spatial access methods and/or hash functions.
* It is **space efficient**: Matrix Profile construction algorithms requires an inconsequential space overhead, just linear in the time series length with a small constant factor, allowing massive datasets to be processed in main memory (for most data mining, disk is death).
* It **is incrementally maintainable**: Having computed the Matrix Profile for a dataset, we can incrementally update it very efficiently. In many domains this means we can effectively maintain exact joins/motifs/discords on streaming data forever. MP is extremely scalable, for extremely large datasets we can compute the Matrix Profile in an anytime fashion, allowing ultra-fast approximate solutions and **real-time data interaction**.
* **Simplicity and Intuitiveness**: Seeing the world through the MP lens often invites/suggests simple and elegant solutions.
* It can be constructed in **deterministic time**: given only the length of the time series, we can precisely predict in advance how long it will take to compute the Matrix Profile. (this allows resource planning)
* It can **leverage hardware**: Matrix Profile construction is embarrassingly parallelizable, both on multicore processors, GPUs, distributed systems etc.

**DEFINITIONS**

* **Definition 1:** A **time series** T ∈ ℝn is a sequence of real-valued numbers ti ∈ ℝ : T = {t1,t2 … tn} , where n is the length of T
* ﻿**Definition 2:** A **subsequence** Ti,m ∈ ℝm is a continuous subset of values from T of length m starting from position i. Formally defined as ti ∈ ℝ : Ti,m = {ti,ti+1 … ti+m-1}
* **Definition 3:** A **distance profile** D is a vector of the Euclidean distances between a given query and each subsequence in an all subsequence length. In the original matrix profile formulation ﻿is assumed that the distance is measured using the Euclidean distance between the z-normalized subsequences
* **Definition 4:** An **all-subsequences set** A of a time series T is an ordered set of all possible subsequences of T obtained by sliding a window of length m across T : A = {T1,m , T2,m … Tn-m+1,m} where m is a user-defined subsequence length. We use A[i] to denote Ti,m
* **Definition 5:** Given two all-subsequences sets A and B and two subsequences A[i] and B[j], a **1NN-join function** θ1NN(A[i], B[j]) is a Boolean function which returns “true” only if B[j] is the nearest neighbor of A[i] in the set B.
* **Definition 6:** Given all-subsequences sets A and B, a **similarity join set JAB** of A and B is a set containing pairs of each subsequence in A with its nearest neighbor in B : JAB = {〈 A[i], B[j] 〉 | θ1NN(A[i], B[j]). We denote this formally as JAB=A⋈ θ1NN B.
* **Definition 7:** A **matrix profile** (or just profile) PAB is a vector of the Euclidean distances between each pair in JAB . We call this vector the matrix profile because one (inefficient) way to compute it would be to compute the full distance matrix of all the subsequences in one time series with all the subsequence in another time series and extract the smallest value in each row (the smallest non-diagonal value for the self-join case).
* **Definition 9:** The ith element in the matrix profile tells us the Euclidean distance to the nearest neighbor of the subsequence of T, starting at i. However, it does not tell us where that neighbor is located. This information is recorded in matrix profile index. A **matrix profile index IAB** of a similarity join set JAB is a vector of integers where IAB[i] = j if {A[i], B[j]} ∈ JAB