SIST: A Similarity Index for Storage Traffic

Lu Pang, Krishna Kant, and Jie Wu
CIS Department, Temple University
Philadelphia, PA
{lpang, kkant, jiewu}@temple.edu

Outline

- Introduction
- SIST
- Evaluation
- Conclusion and Future Work

Introduction

- Performance of storage systems is crucial for data intensive workloads
  - Constant evolution of storage technologies and systems
  - Evaluation of storage systems or features is crucial and often done using storage traces
- With increasing availability of traces, how do we select one or more traces that
  - Have certain characteristics
  - Cover a range of behaviors
  - Are easily distinguished from other similar traces
- Need notion of ***Trace Similarity***

Existing Similarity Measures (1)

- Image Similarity (IS)
  - Used to measure the similarity between two images
  - Focused on directly or indirectly trying to model the human visual system
- Image Similarity example
  - Structural similarity index (SSIM) : A product of three comparison measurements between the target and reference image
    - Mean value (luminance)
    - Variance (contrast)
    - Correlation (structure)

Existing Similarity Measures (2)

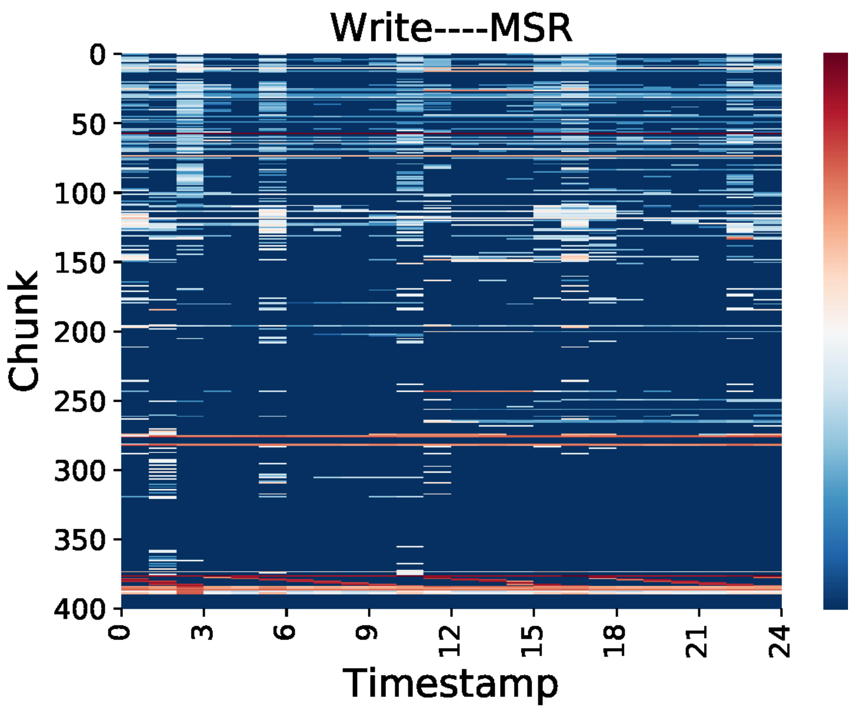
- Time Series Similarity (TSS) Measures
  - Used to measure the relationship between two time series
  - Focus on scenarios where direct comparison of values is important
    - Stock price
    - Gnomic structure
    - ECG signals
- TSS example
  - Euclidean (or Minkowski) distance
  - Edit distance
  - Longest Common SubSequence (LCSS)
  - Dynamic time warping (DTW) – provide better alignment

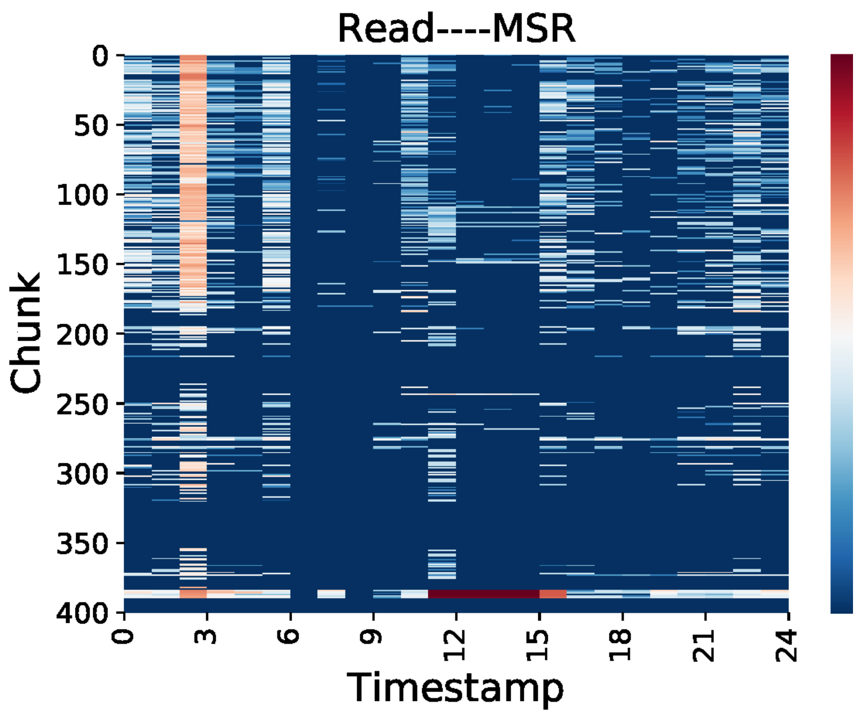
Why do we need a new measure?

- Trace Similarity needs
  - Comparison should be relevant to the storage performance perspective
  - Need to cover both temporal and spatial similarity aspects
  - Need to capture nonhomogeneity in storage behavior
    - Applications with different characteristics starting or ending at unpredictable times
    - Occasional unusually heavy load on some applications
    - Regular but distinctive activities such as backup
- Aggregate measures inadequate, e.g.,
  - Time aggregation – overall access frequency of each chunk
    - Inadequate for caching, short-term tiering, prefetching, etc.
  - Space aggregation – overall accesses in successive time-slots
    - Useful only for network bandwidth management
  - Aggregate variability measures also largely inadequate

Typical Trace Behavior

Number of read and write accesses of Friday for MSR usr workload (red represent more accesses, blue represent less accesses)





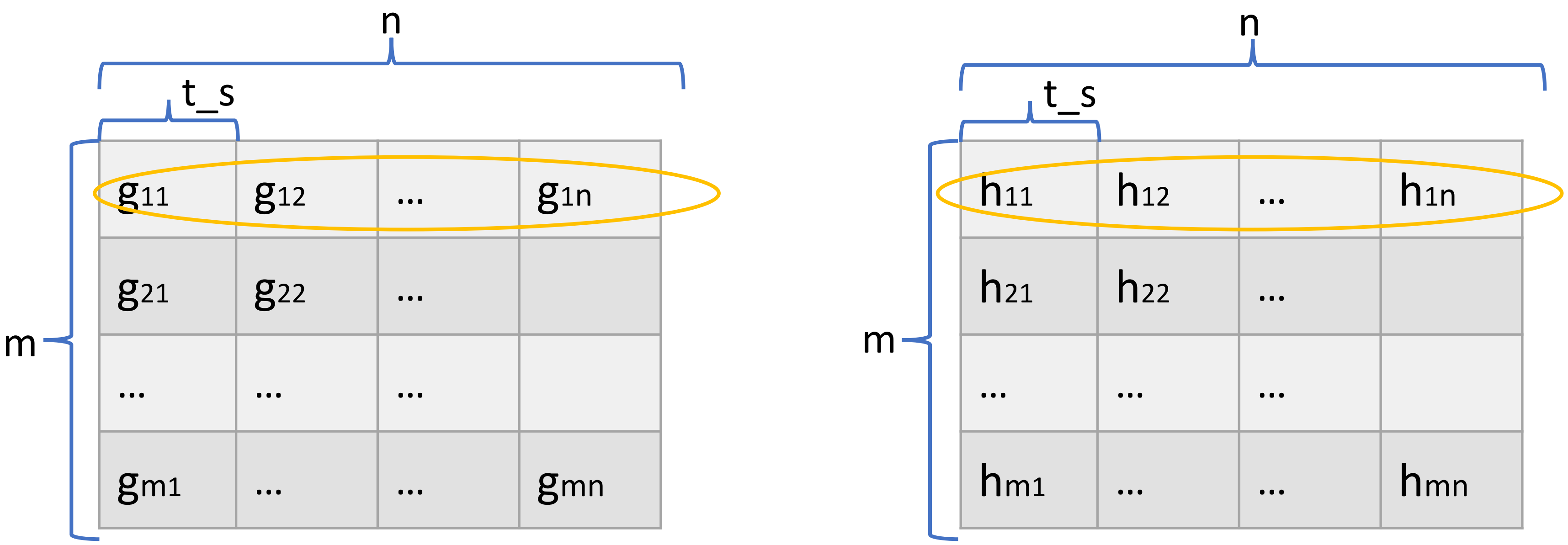
Similarity Index for Storage Traces

- Provide a similarity index called SIST
  - Accounts for storage performance aspects in general
  - However, avoid tying it to specific use-cases (e.g., tiering, caching, etc.)
- SIST is a triplet (S_M, S_A, S_D)
  - S_M : Overall (or Main) similarity based on discrete wavelet transform (DWT)
  - S_A : Similarity in activity level of two traces
  - S_D : High-frequency content (or Detail) similarity (that is ignored in the computation of S_M)
- SIST caters to storage performance
  - SIST captures access locality behavior, activity level, high-frequency behavior

Trace Pre-processing to compute SIST

- Original trace: Each access is a 4-tuple consisting of
  - Offset
  - Request size
  - Request type (read or write)
  - Timestamp
- Represent all requests in a time slot as a data-grid where
  - Rows: chunk number
  - Columns: time slot
  - Contents of a cell: #accesses with the specified chunk in the specific time slot
- May have multiple chunk accesses in the same time slot, which will be represented in different rows of the same column

Example of a Data Grid

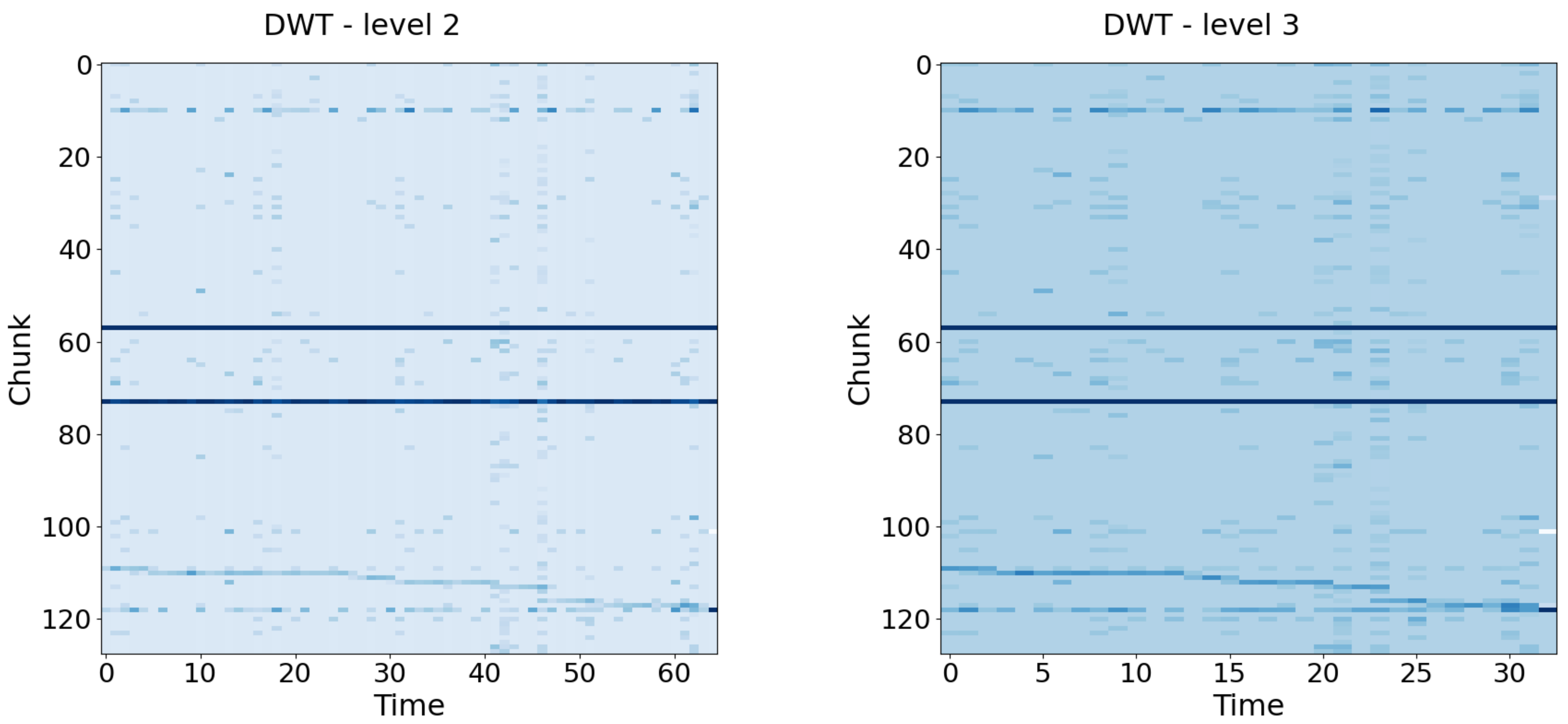


- Selection of time-slot duration
  - Depends on the intensity of the trace
    - Most storage operations occur in phases, with each phase accessing a rather small range of LBAs actively
  - Need to choose time-slot duration large enough to contain a significant number of accesses to these active ranges (e.g., 10's to 100’s)
  - The similarity is not meaningful (too few accesses)
  - The similarity is not granular enough (too many accesses)

SIST Implementation

- Use DWT on data-grids
  - Allows separation of level of detail (through scale factor)
  - Low-dimensional features used to compute S_M and S_A measures
  - Top k detail coefficients of DWT used to finer details S_{H}
- For main similarity, we do not use DWT coefficients directly
  - Use the approximation coefficients of level k to get a low dimensional representation of the grids
  - Distance metric (DM) for comparing traces
    - Use constrained dynamic time warping (DTW) to align each pair of the corresponding row vectors of the two grids
    - Obtained as sum of Euclidian distance between rows over columns (chunks)
- Compute S_A : the normalized difference between the root sum squares of the low dimensional representation of the grids
- Compute S_H : the normalized difference between the root mean square deviation of the fine detail representation of the grids

DWT at different scale factors



Evaluation – Object Storage Dataset

- HPC Trace
  - Publicly available server-side I/O request arrival traces
  - Use OrangeFS parallel file system
  - Generated on the Rennes site of Grid’5000 Workloads using MPI-IO Test benchmarking tool
- Workload
  - Several traces w/ different config. parms (File layout, Spatiality, Request size, #processes, etc.)
  - Trace A and trace B have different spatiality parameter (contiguous and non-contiguous), and different number of processes (64+64 and 64+32)
  - Trace C has a different request size than A and B

Evaluation – Block Storage Dataset

- MSR Trace
  - One-week public block I/O traces of enterprise servers at MSR, Cambridge
- Workloads
  - User home directories (usr : A), Project directories (proj : B), HW monitoring (hm: C)
  - Trace request details
    - Timestamp: the time the request is made, in Windows file time
    - Hostname: the hostname (ignored)
    - Disk number: the same disk number (ignored)
    - Type: "Read" or "Write"
    - Offset: the byte offset from the start of a disk to the requested LBA
    - Size: the number of bytes requested
    - Response time: the time needed for the request to complete (ignored)

Evaluation – Stack Distance

- Evaluate SIST with Stack Distance
  - Stack distance is an important measure of locality and hence performance
  - Use grids where a point in position (chunk#, time) denotes the average stack distance for chunk# within the specific time-slot
  - Similar behavior for stack distance confirms that SIST should also track the storage performance quite well
  - Shows the similarity comparisons for the four perturbations of the results with HPC stack distance grids

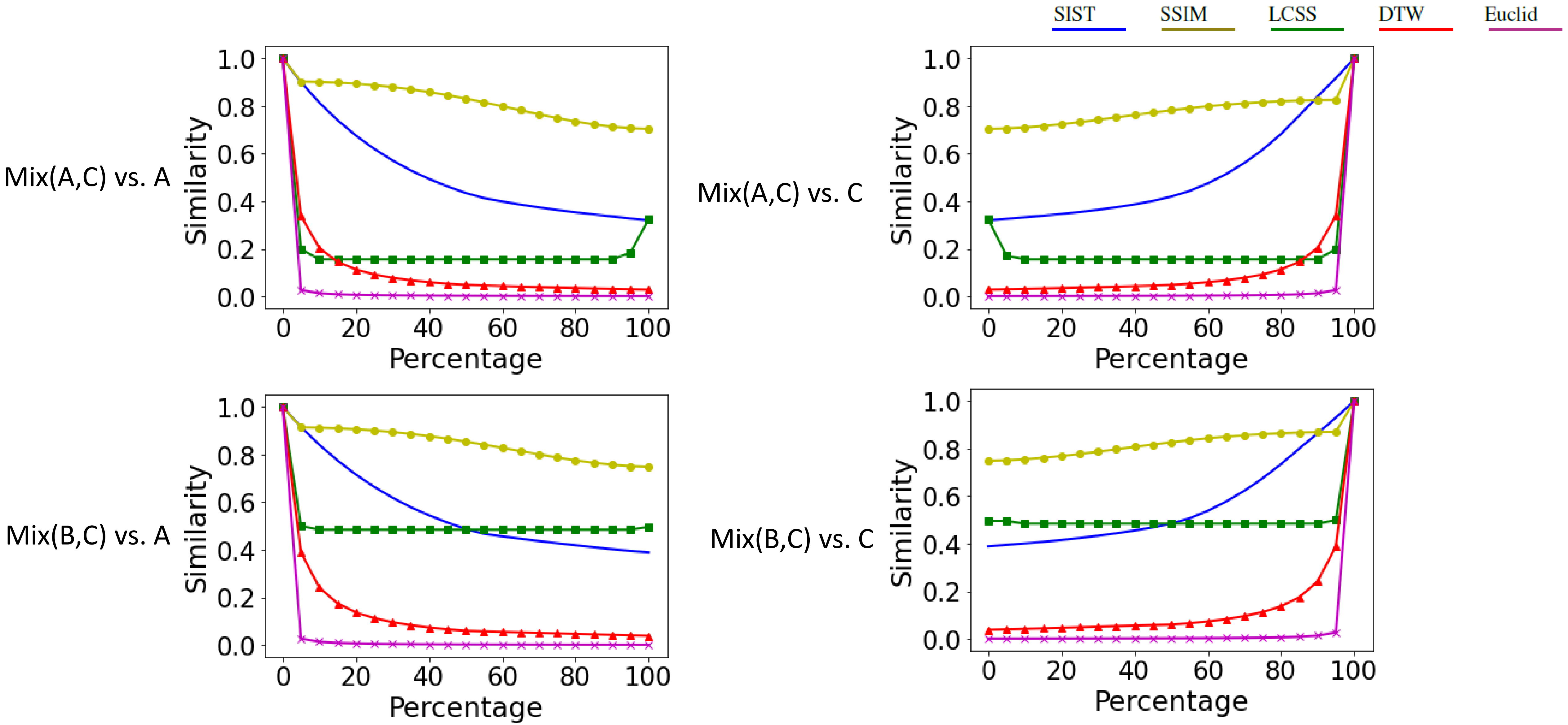
Evaluation - Experimental setup

- Evaluate SIST with 4 types of perturbations
- In all cases, uses a probability parm p in the range $0..1$
  - Mixing:
    - Random mixture of traces (say, A, B). Mixed trace M = $(1 - p\%) * A + p\% * B$
  - Shifting:
    - Shift the original trace in the left or right direction by p% (Any vacated places are filled with zeros)
  - Salt and pepper noise addition:
    - Replace (p/2)% of the accesses (in randomly chosen positions) have the minimum access count and the remaining (p/2)% of accesses to have the maximum access count
  - Traffic Thinning:
    - Remove randomly chosen p% of the accesses (i.e., make them zero) without changing the trace length

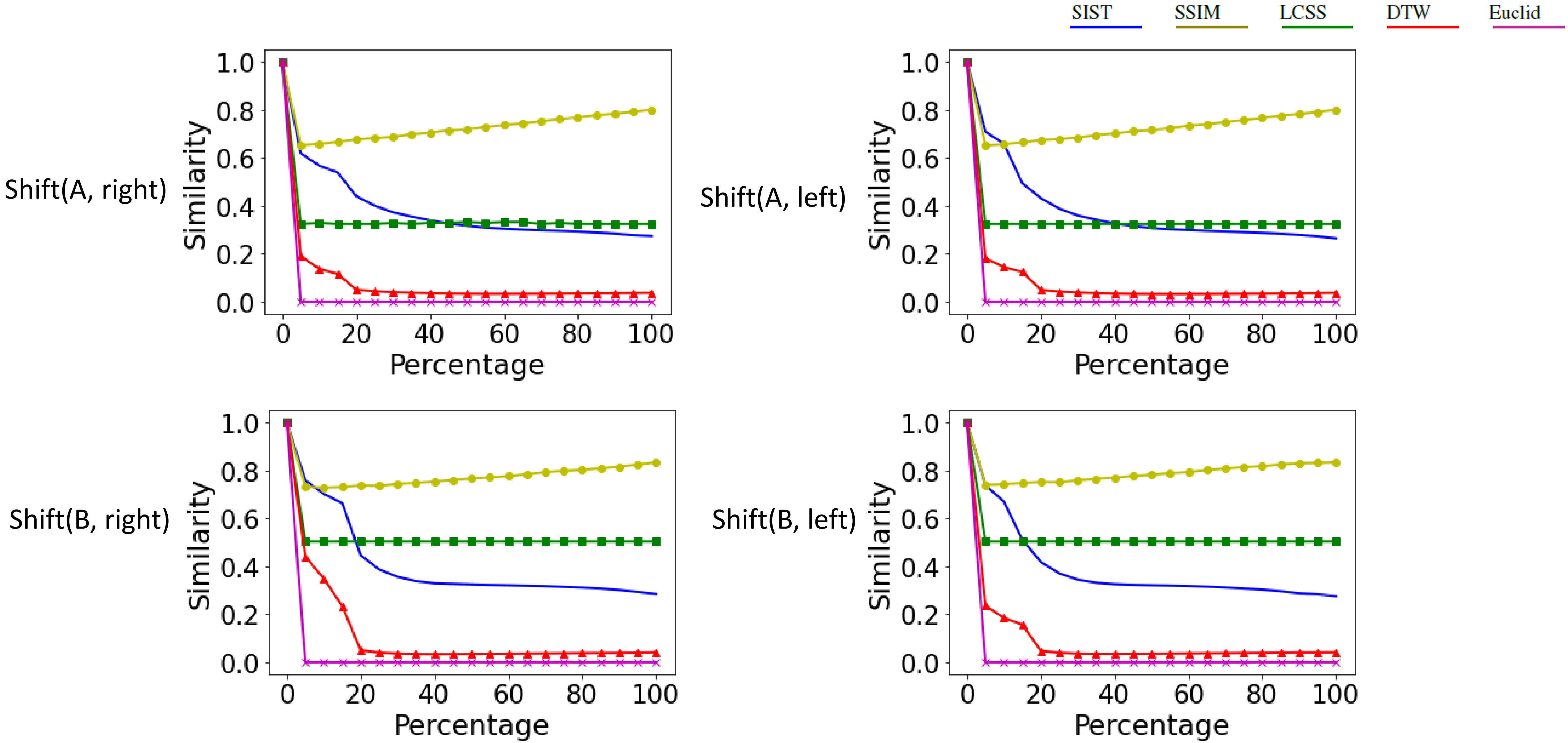
Evaluation - Experimental setup

- Evaluate SIST against the following four measures
  - Structural Similarity Index (SSIM):
    - Treat the trace grids as the input images to compute the similarity
  - Longest Common SubSequence (LCSS) similarity:
    - Computed as the length of the longest common subsequence divided by the time series length
  - Euclidean Distance (Euclid):
    - Computed as the normalized Euclidean distance between the two time-series
  - Dynamic Time Warping based similarity:
    - Calculated as the normalized mean DTW distance for row vectors of trace grids

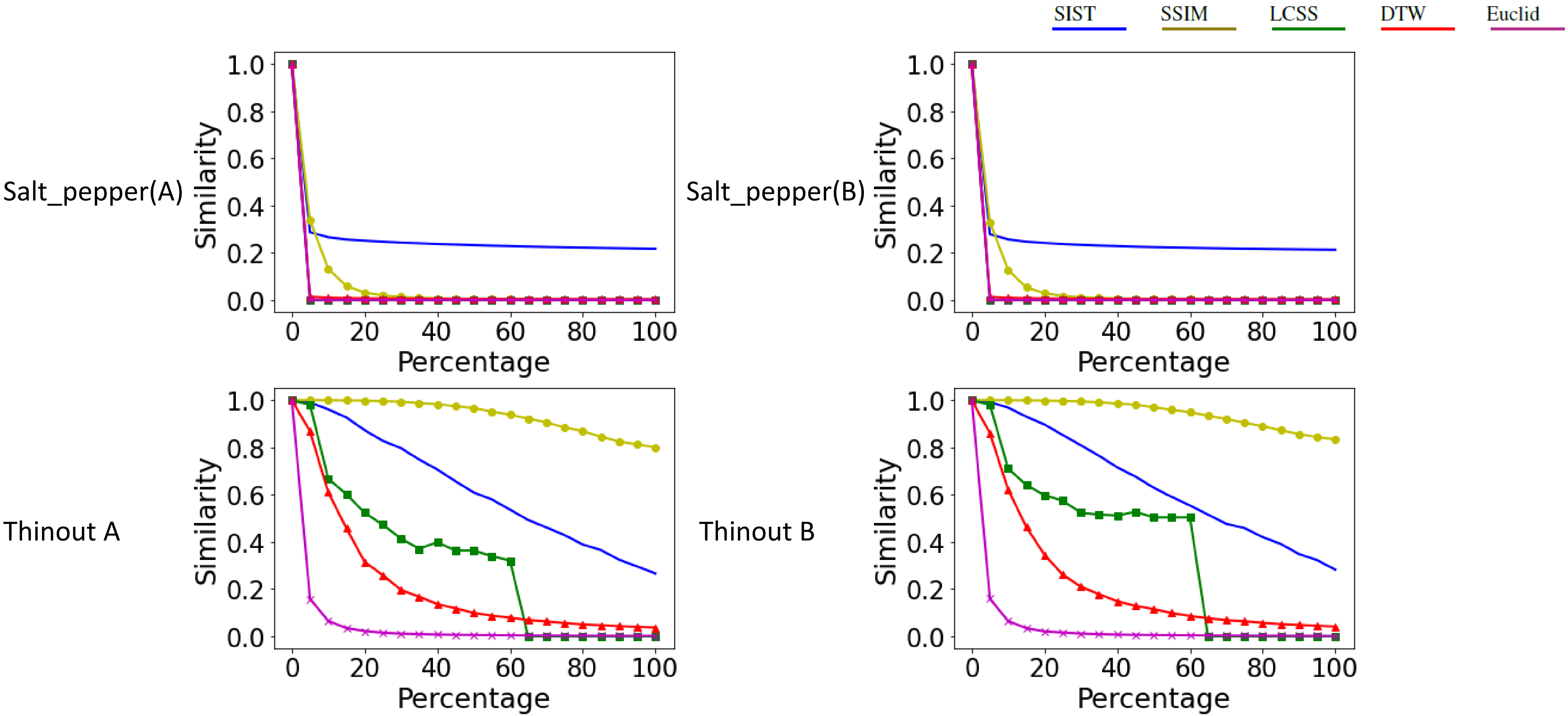
Evaluation – Mixing (HPC traces)



Evaluation – Shifting (HPC traces)

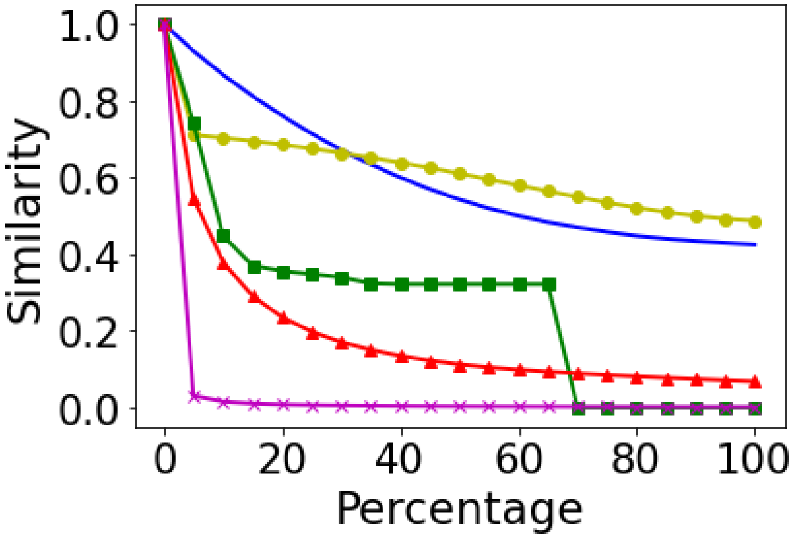


Evaluation – Salt-n-pepper & Thinout (HPC traces)

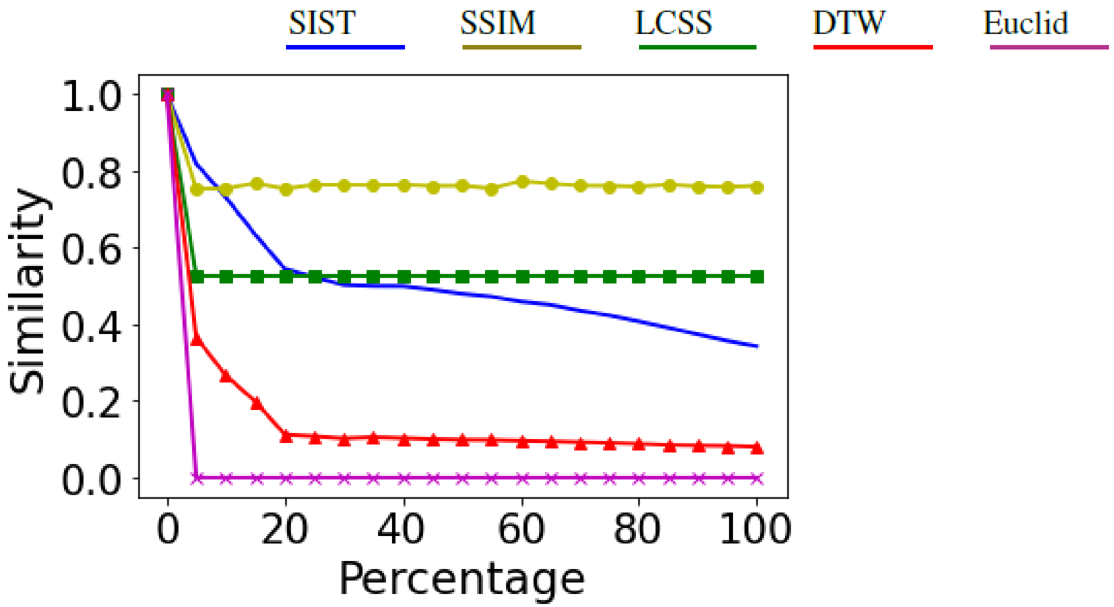


Evaluation – MSR traces

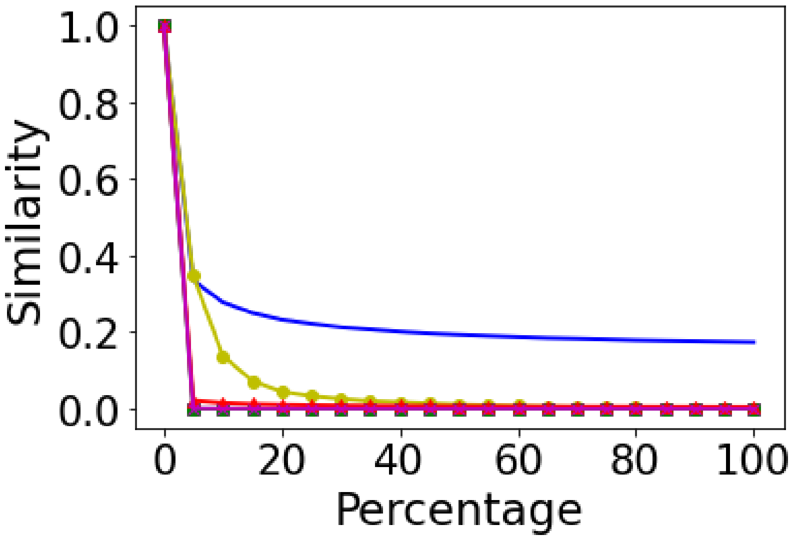
Mix(A,C) vs. A



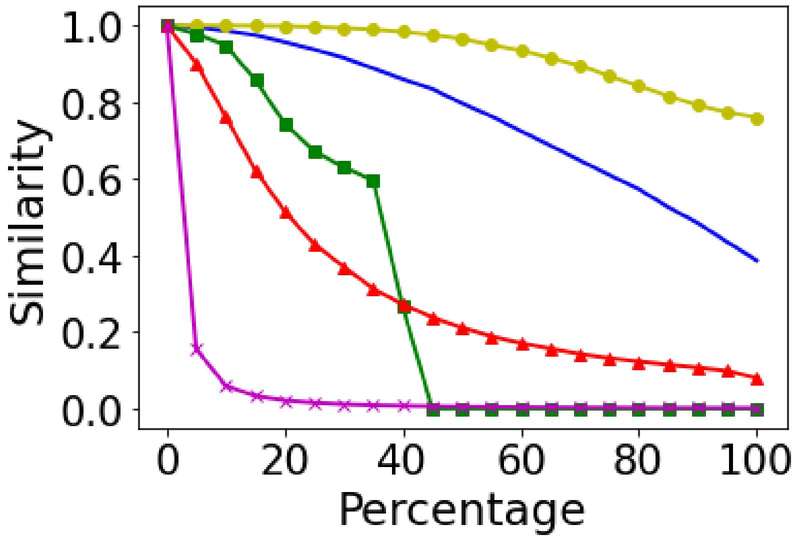
Shift(A,right)



Salt_pepper(A)



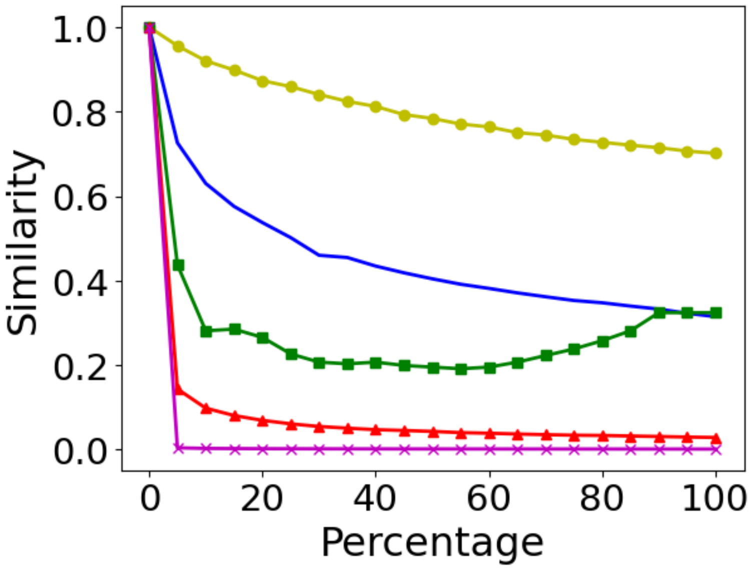
Thinout A



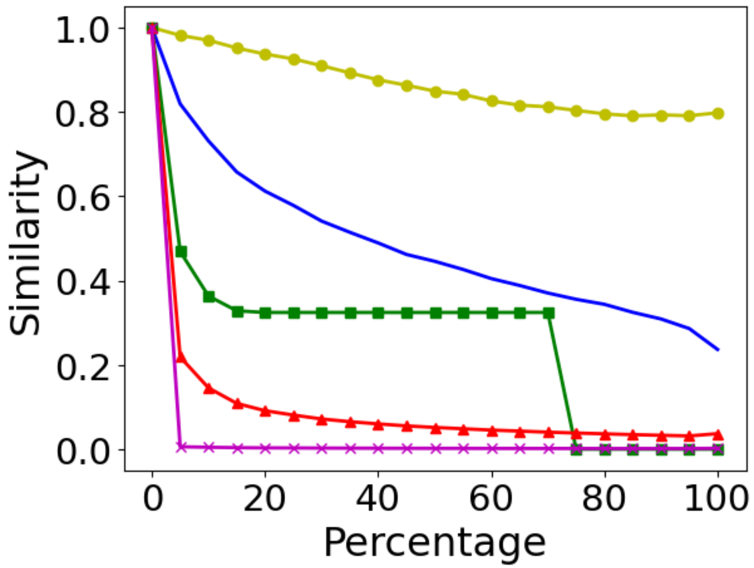
SIST SSIM LCSS DTW Euclid

Evaluation – Stack-distance (HPC traces)

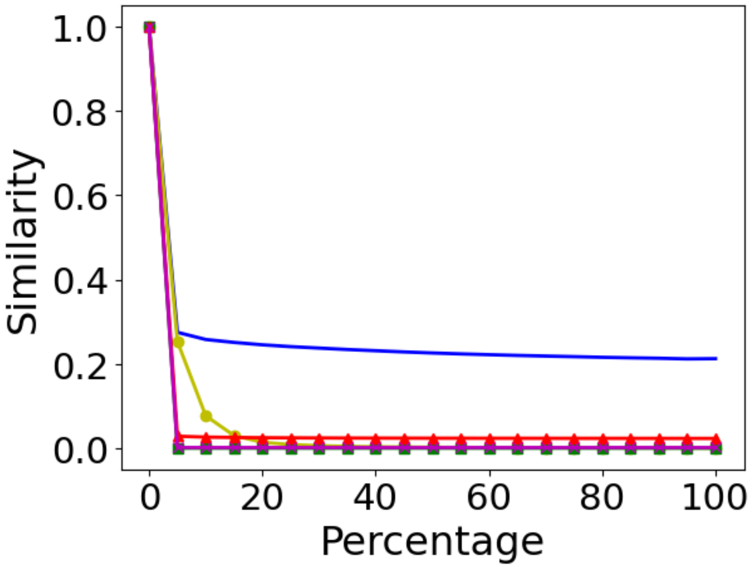
Mix(A,C) vs. A



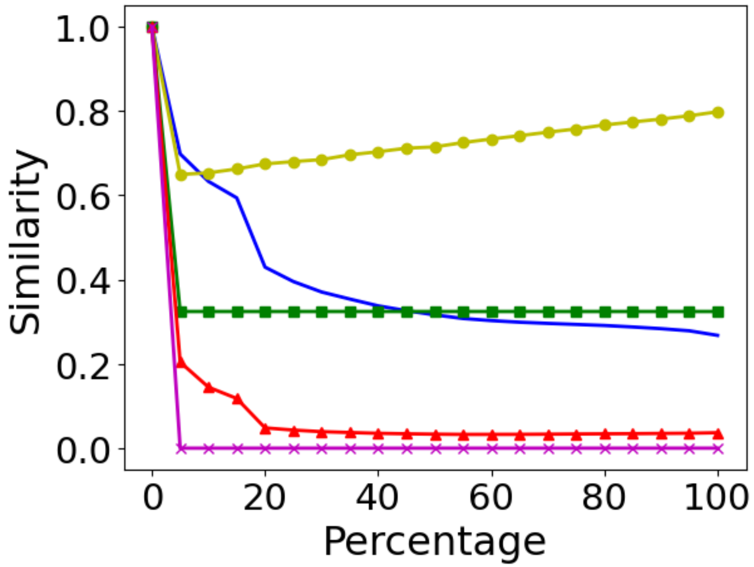
Shift(A,right)



Salt_pepper(A)

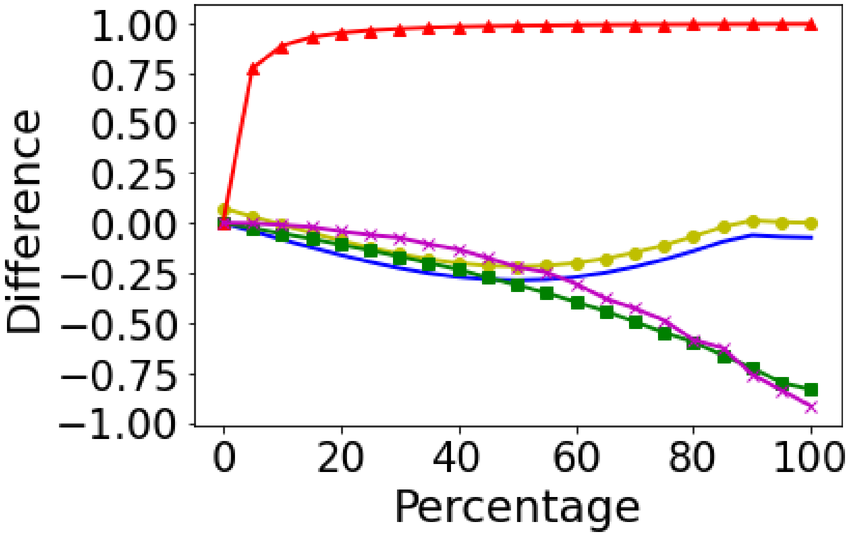


Thinout A

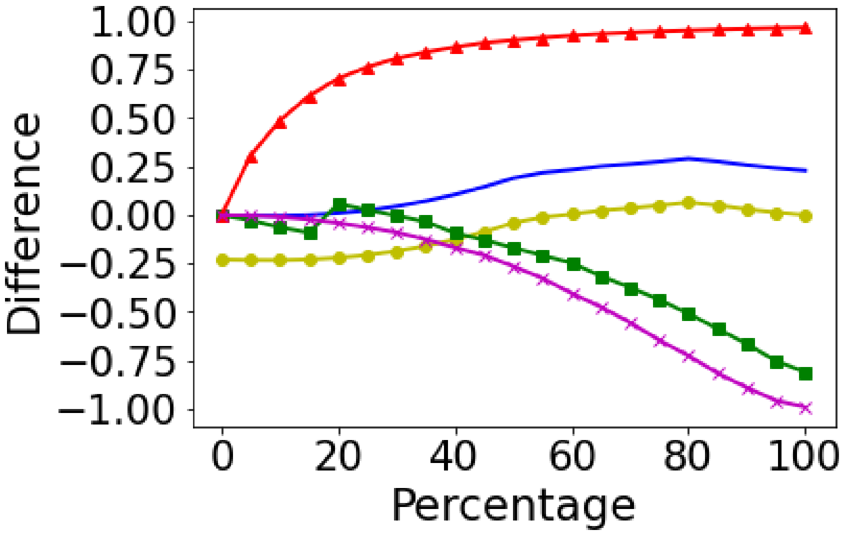


Evaluation – Activeness & Finer detail differences

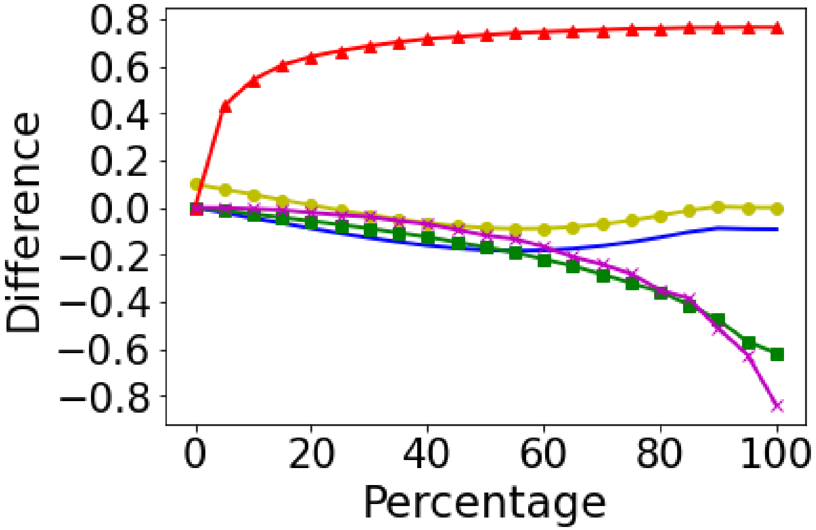
HPC A
Activeness
differences



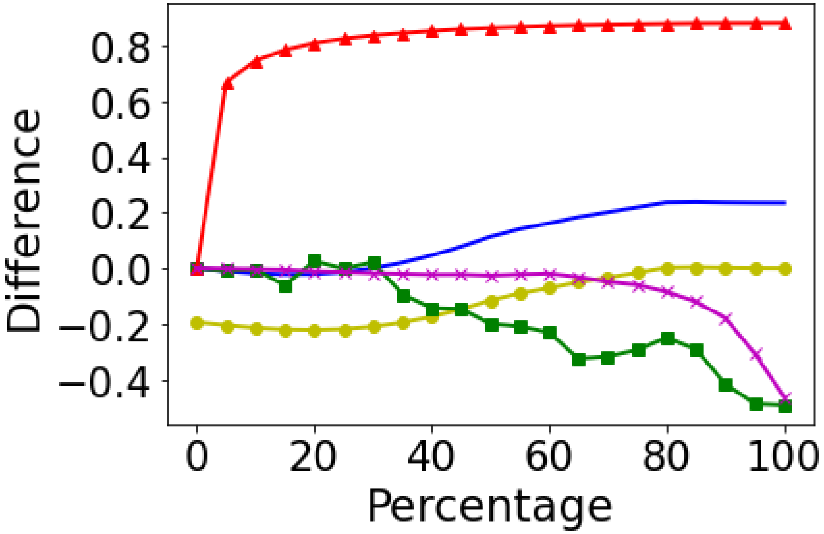
MSR A
Activeness
differences



HPC A
Finer detail
differences



MSR A
Finer detail
differences



Conclusion and Future Work

- Defined a similarity index called SIST for comparing storage traces and compared it against other commonly used measures for traces from both object and block storage
- We showed that SIST has a much better behavior compared to the commonly used image and time series similarity measures for storage trace
- Future work
  - Validate SIST and correlate it with actual storage system performance for various use cases
  - Devise efficient algorithms for clustering traces based on the similarity metric and potentially other criteria relevant to storage system performance