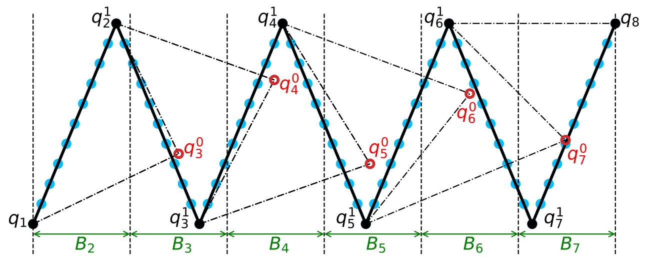

Figure 21: Optimal and convergent special case

A BASELINES

If not stated otherwise, **ILTS** uses average initialization and four iterations in the experiments. This choice is made because the results with random initialization are similar, and ILTS empirically converges within four iterations. **LTTB** is a special case of ILTS, using average initialization and single iteration. **MinMaxLTTB** is a two-step method that uses MinMax to preselect a small ratio of points (e.g., $2 * m$ used in the following experiments) and then applies LTTB on the preselected time series to sample m points. **Visval** works by iteratively removing the point that leads to the least areal difference (i.e., the smallest triangle area formed with its two adjacent points) in a bottom-up manner.

MinMax intuitively selects the min and max points per bucket. **M4** selects the first, last, min, and max points per bucket to cater to the semantics of line rasterization. **Uniform** is a straightforward method that samples points at fixed intervals.

FSW is an angle-based greedy PLA method that attempts to maximize the segment length as long as its feasible space is nonempty. **Sim-Piece** is a disjoint-segment PLA method that merges similar segments to achieve a better compression ratio.

DFT decomposes data into a sum of sinusoidal components and captures frequency components with the highest amplitudes. **PCA** transforms data into a set of orthogonal principal components and captures those that account for the most significant variance.

B SSIM FORMULA

Let X and Y denote the pixel matrices of two images with the same dimensions $w \times h$. Formally,

$$SSIM(X,Y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{4}$$

where μ_x, μ_y are the average intensities for luminance comparison, σ_x^2, σ_y^2 are the variances for contrast comparison, σ_{xy} is the covariance of X and Y for structure comparison, and C_1, C_2 are small constants to stabilize division operations.

C PROOF OF PROPOSITION 1

Proof. As shown in Figure 21, during the first iteration, due to the characteristics of the triangle wave, the slope of the anchor-floater line for bucket $B2$ is less than that of the upward segment. Therefore, the farthest point from this anchor-floater must be the turning point where the wave starts to descend. Conversely, in bucket $B3$, the slope of the anchor-floater line is greater than that of the downward segment, meaning the farthest point from this anchor-floater is the turning point where the wave begins to ascend. As a result, the sampled points from the first iteration correspond to the turning points of the triangle wave, which represent the optimal solution for the largest triangle sampling. In subsequent iterations, the sampling results remain unchanged, achieving convergence. □

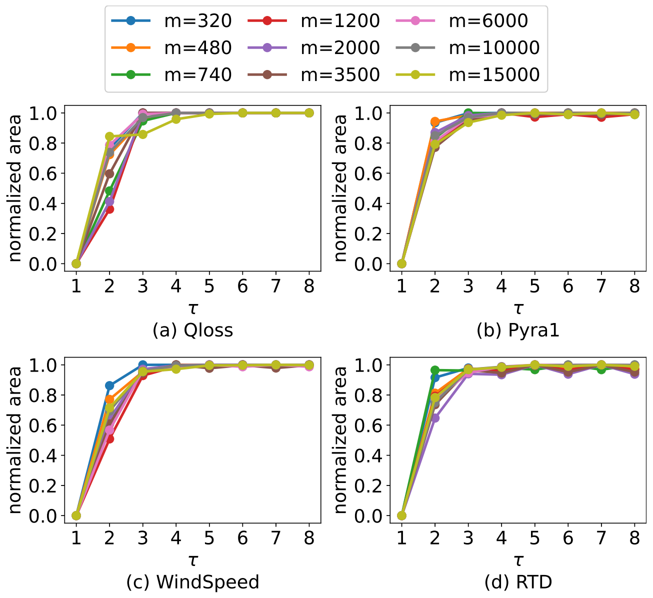

Figure 22: ILTS convergence on effective area

D CONVERGENCE EVALUATION

Each line in Figure 22 is the change curve of the effective area of the sampled time series by ILTS during eight iterations under a fixed number of sampled points m. We normalize the effective areas in order to observe the common trend of different m. It can be seen that the effective area increases significantly since the second iteration, and then convergences quickly within four iterations.

E VOTING RESULTS FROM USER STUDY

Table 4 shows the dataset-specific voting results from our user study. To ensure that users' choices were not influenced by method names or SSIM scores, participants were not informed of this information for each image in the study.

F VARYING THE NUMBER OF INPUT POINTS n BY QUERY SELECTIVITY

We adjust the selectivity of the query to vary the input points n in Figure 23. Compared with Section 6.4.3, the results of ILTS are similar, regardless of whether the query selectivity or dataset size is varied. It is because a convex hull is not pre-computed for the entire dataset but on each page with a fixed number of points.

G OVERHEAD EVALUATION

Figure 24 shows the write latency and disk space occupation before and after adding convex hull precomputation. The convex hull

Table 4: Dataset-specific voting results

Algorithm (%)	Qloss	Pyra1	WindSpeed	RTD
ILTS	25	25	23	26
LTTB	22	17	23	22
MinMaxLTTB	21	25	20	21
MinMax	22	16	20	13
M4	1	6	9	3
Visval	3	7	3	9
Uniform	1	0	0	0
FSW	4	0	1	1
Sim-Piece	0	0	1	0
DFT	1	3	0	5
PCA	0	1	0	0

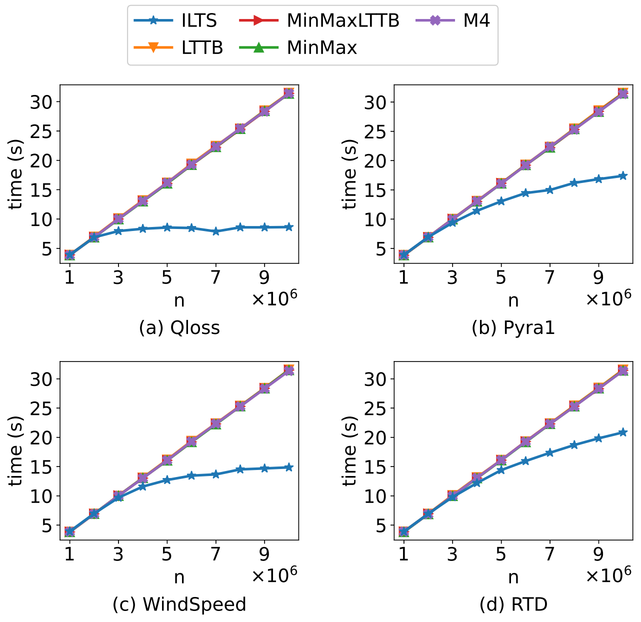

Figure 23: Scalability on the number of input points n by varying the query selectivity

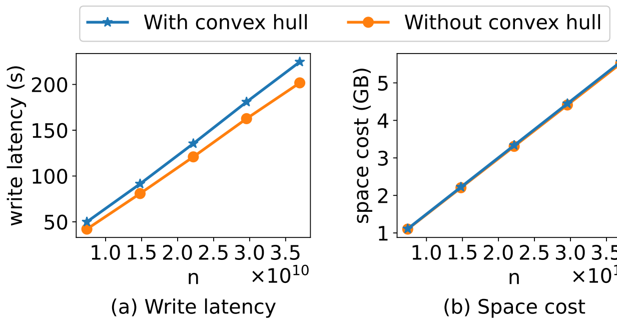

Figure 24: Overhead of convex hull precomputation

construction algorithm implemented is Quickhull [11] with a time complexity of $O(nlogn)$ given n input points. Therefore the write latency increases after adding convex hull precomputation. On the other hand, convex hull precomputation has little impact on space consumption as shown in Figure 24(b). It is because we use the bitmap for representing the convex hull (Section 5.2.2) and four numbers for representing the bounding box of the convex hull (Section 5.2.3). Given the acceleration effect of convex hulls on ILTS, we believe it is worthwhile to precompute convex hulls.