Cluster-DAC: Drift-Adaptive Budgeted Clustering for Evolving Data Streams

Raphael Alves dos Reis
Department of Computer Science
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, Brazil
cap497@ufmg.br

***Abstract*—Real-world data streams evolve continuously, demanding clustering algorithms that can adapt to distributional drift without repeated reinitialization or unbounded growth of model complexity. This paper introduces *Cluster-DAC*, a drift-adaptive, budgeted clustering framework designed for non-stationary streaming environments. The method combines exponentially weighted moving averages (EWMA) for temporal smoothing, low-rank covariance modeling for anisotropic precision, and a structural-editing mechanism (split, merge, birth, prune) constrained by a fixed cluster budget.**

Cluster-DAC supports tracking of evolving manifolds and constructing drift surfaces that summarize how the stream moves through feature space. On a synthetic drifting two-Gaussian benchmark, the full model achieves a mean tracking RMSE of 0.43 compared to 1.51 for a variant without EWMA, i.e., a reduction of roughly $3.5\times$ in error, while keeping the number of clusters close to the budget (between 1 and 6, with an average of 5.77). A reference implementation processes 4000 synthetic updates in $4.05\,$s ($\approx 1.0\,$ms/update) and 4000 vehicle-like telemetry updates in $0.63\,$s ($\approx 0.16\,$ms/update) on a CPU.

In addition to empirical performance, Cluster-DAC emphasizes interpretability: practitioners can visualize EWMA-smoothed trajectories of cluster means, structural edits over time, drift surfaces, and recency-weighted "temperature" maps of cluster activity. The combination of budgeted structure editing, anisotropic modeling, and temporally stable EWMA updates yields a compact, expressive clustering model suitable for embedded systems, autonomous vehicles, and fast-response streaming environments.

I. Introduction

A. Motivation

Many modern systems operate on continuous streams of data rather than static batches: autonomous vehicles produce high-rate sensor traces, financial markets emit transaction sequences, and IoT deployments generate telemetry from distributed devices. These streams typically exhibit *concept drift*: the data-generating distribution changes over time, sometimes gradually and sometimes abruptly. Static clustering methods are blind to such changes, while many dynamic approaches allow their internal representation to grow without bound.

In streaming settings, clustering faces a characteristic trade-off: staying responsive to drift while keeping model size and computational cost bounded. A practical algorithm should learn incrementally, forget gracefully, and maintain a representation that is interpretable and resource-aware.

In realistic scenarios, drift is often multi-scale. Global trends evolve slowly (e.g., seasonal or long-term changes), while local dynamics fluctuate rapidly (e.g., maneuvers, mode switches, or transient noise bursts). Traditional clustering methods either overreact to local noise or underreact to global drift. To address this, Cluster-DAC combines two mechanisms: (i) EWMA-based updates that smooth high-frequency fluctuations, and (ii) occasional structural edits that reorganize clusters without violating a strict budget.

Another difficulty is that high-dimensional streams frequently lie near curved or elongated manifolds. Isotropic cluster models struggle to capture directional drift or covariance structure. Cluster-DAC uses low-rank covariance approximations to represent anisotropy efficiently, enabling clusters to stretch and rotate while respecting a fixed memory footprint.

B. Contributions

This work makes the following contributions:

- **EWMA-based adaptivity:** cluster statistics are updated via exponentially weighted moving averages, which preserve temporal coherence and reduce oscillations under noisy drift.
- **Anisotropic low-rank covariance modeling:** covariances are represented in a low-dimensional subspace, capturing principal directions of drift while reducing computation.
- **Budgeted structural editing:** split, merge, birth, and prune operations keep the number of clusters below a fixed maximum while adapting topology.
- **Drift surface and temperature maps:** Cluster-DAC supports reconstruction of drift surfaces and recency-weighted temperature maps that summarize where the stream spends time and how quickly it moves.

Empirically, we provide:

- an EWMA smoothing illustration (Fig. 1),
- trajectory fields of cluster means (Fig. 2),
- structural evolution and temperature maps for synthetic data (Figs. 8 and 9),
- synthetic drifting Gaussians with covariance ellipses and pathlines (Figs. 3–6),
- drift surfaces and temperature evolution for vehicle-like telemetry (Figs. 10 and 11),

- and an ablation study on EWMA and anisotropic covariance modeling (Fig. 12).

II. Related Work

A. Streaming and Evolving Clustering

Early streaming clustering methods, such as BIRCH [1] and CluStream [2], maintain micro-clusters or summary statistics to keep memory bounded. DenStream [3] incorporates exponential fading to handle noise and de-emphasize stale data, while ClusTree [4] organizes micro-clusters in a tree structure to provide logarithmic insertion time. Many of these approaches, however, rely on isotropic or spherically shaped components.

Cluster-DAC follows this line of work but adds explicit low-rank covariance, EWMA-based temporal smoothing, and a structural editing mechanism driven by condition numbers and distances. The explicit budget on the number of clusters keeps memory and runtime predictable.

Other evolving clustering approaches include probabilistic mixture models with recursive updates, adaptive resonance theory networks, and evolving fuzzy systems. These methods can be powerful but often require careful tuning of many hyperparameters and may not enforce strict budget constraints.

Neural topological learners such as Growing Neural Gas [5] adapt a graph of nodes to streaming data. Cluster-DAC is conceptually simpler and focuses on a small set of interpretable Gaussian-like clusters, with explicit handling of anisotropy and drift surfaces.

B. Drift Adaptation and Stability

Numerous methods incorporate some notion of forgetting or decay for drifting data, including fading factors in clustering and drift detection mechanisms in supervised learning. However, stability analyses under limited memory are less common.

Cluster-DAC uses EWMA updates for cluster means and covariances. Under mild conditions on the drift rate, EWMA can be interpreted as a discrete-time low-pass filter with effective memory $1/\lambda$. This constrains the variance of parameter estimates and reduces sensitivity to high-frequency noise compared to unconstrained stochastic gradient updates.

III. Problem Definition

We consider a data stream $\{x_t\}_{t=1}^{T}$ with $x_t \in \mathbb{R}^d$, potentially unbounded in length. We wish to maintain an evolving set of clusters C_t such that:

- **Budget:** $|C_t| \le K_{\max}$ for all times t.
- **Temporal adaptivity:** recent data are weighted more heavily via an EWMA decay parameter λ.
- **Anisotropy:** each cluster is modeled with a low-rank covariance matrix capturing principal directions.

Each cluster c at time t is associated with a mean $\mu_c(t)$ and covariance approximation

$$\Sigma_c(t) = U_c S_c U_c^\top + \sigma_c^2 I, \tag{1}$$

where $U_c \in \mathbb{R}^{d\times r}$ contains principal directions, S_c is diagonal, σ_c^2 is an isotropic residual, and $r \ll d$.

Beyond clustering, we also aim to reconstruct *drift surfaces* by interpolating cluster activation over a low-dimensional manifold (2D in our experiments), and to compute per-cluster *temperature* based on recency of use.

IV. Method

A. EWMA Updates

For a cluster c with mean $\mu_c(t)$ and covariance $\Sigma_c(t)$, EWMA updates are:

$$\mu_c(t) = (1-\lambda)\,\mu_c(t-1) + \lambda\, x_t, \tag{2}$$

$$\Sigma_c(t) = (1-\lambda)\,\Sigma_c(t-1) + \lambda\,(x_t - \mu_c(t))(x_t - \mu_c(t))^\top. \tag{3}$$

The parameter $\lambda \in (0,1]$ controls the effective memory: small λ values yield long-term smoothing, while large λ values react quickly to changes. The updates can be viewed as a first-order digital low-pass filter applied to the sufficient statistics.

Fig. 1 illustrates EWMA smoothing on a 1D drifting mean. The raw noisy trajectory exhibits substantial jitter around the true mean, whereas the EWMA-smoothed path remains coherent.

B. Low-Rank Covariance

To keep computations efficient, we approximate each covariance by a low-rank-plus-diagonal form. In two dimensions (as used for visualization in our synthetic experiments), we directly store the full 2×2 matrix and treat it as anisotropic. In higher dimensions, we would store only the top r eigenvectors and eigenvalues, i.e.,

$$\Sigma_c \approx U_c S_c U_c^\top + \sigma_c^2 I, \tag{4}$$

and use the Woodbury identity for efficient Mahalanobis distance evaluations.

Even in 2D, the eigenvectors of $\Sigma_c(t)$ encode the local drift orientation. As data streams through time, these directions rotate and stretch, providing a geometric summary of the evolving manifold.

C. Cluster Assignment and Robustness

Given a point x_t, we compute either Euclidean or Mahalanobis distance to each cluster:

$$d_c(x_t) = \begin{cases} (x_t-\mu_c)^\top \Sigma_c^{-1}(x_t-\mu_c) & \text{(Mahalanobis)},\\ \|x_t-\mu_c\|_2^2 & \text{(Euclidean)}. \end{cases} \tag{5}$$

If the best distance exceeds a threshold, the point is considered an outlier with respect to existing clusters and spawns a new cluster (subject to the budget). Otherwise, the corresponding cluster is updated with the EWMA rules.

D. Structural Editing

To adapt the internal structure under drift while enforcing the cluster budget, Cluster-DAC applies structural edits at a slower cadence (every T_{edit} steps):

- **Split:** If a cluster's covariance becomes highly elongated (large condition number), it is split along its dominant eigenvector into two children with perturbed means.
- **Merge:** Clusters whose centers are mutually close in Mahalanobis distance are merged into a single component with combined weight and appropriately updated covariance.
- **Prune:** Low-weight clusters whose effective mass drops below a threshold are removed.
- **Budget enforcement:** If the number of clusters exceeds $K_{\max}$, we keep only the heaviest components.

Fig. 8 visualizes the *number* of clusters over time in the synthetic experiment. The cluster count remains between 1 and 6, with an average of 5.77, very close to the budget $K_{\max} = 6$, demonstrating that edits induce changes without runaway growth.

E. Temperature Mapping

To capture recency effects and activity, we associate a *temperature* $T_c(t)$ with each cluster:

$$T_c(t) = \exp\left(-\frac{t - t_c^{\text{last}}}{\tau}\right), \tag{6}$$

where t_c^{last} is the last time cluster c was updated, and τ controls decay. Temperatures close to 1 indicate recently used clusters; values near 0 indicate outdated components.

We record temperature vectors over time and visualize them as heatmaps. In the synthetic experiment, the mean cluster temperature is approximately 0.54 (with a maximum of 1.0), indicating that the model continuously reuses a subset of clusters while others cool down (Fig. 9). In the vehicle-like telemetry experiment, the mean temperature is about 0.50 (again with a maximum of 1.0), reflecting persistent mode reuse (Fig. 11).

F. Drift Surface Reconstruction

Cluster centers and temperatures can be used to reconstruct a drift surface over a 2D manifold. We discretize a grid over the region spanned by the cluster means and accumulate contributions from Gaussians centered at each mean, scaled by temperature:

$$Z(x) = \sum_c T_c \exp\Big(-\tfrac{1}{2}(x - \mu_c)^\top \Sigma_c^{-1}(x - \mu_c)\Big). \tag{7}$$

Figs. 7 and 10 show examples for synthetic and vehicle-like data, respectively.

In the synthetic case, the resulting surface has values between approximately 3.2×10^{-37} and 1.42, with mean 0.062, which defines a smooth landscape highlighting regions of frequent visitation and high concentration. This surface helps identify zones of accelerated drift, stagnation, and transitions without inspecting individual clusters.

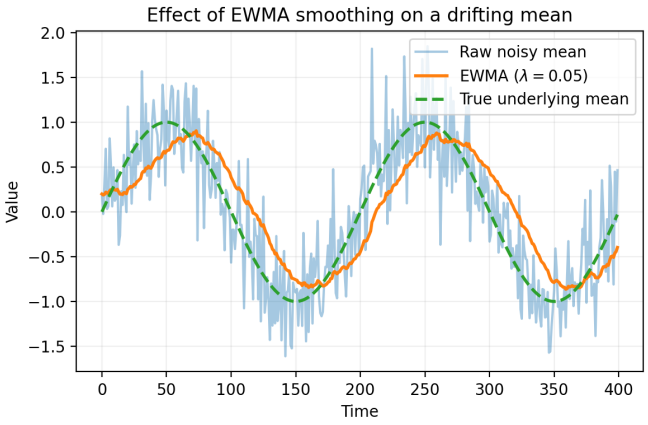

Fig. 1. Effect of EWMA smoothing on a drifting noisy mean. The raw trajectory exhibits substantial jitter, while EWMA produces a smooth, interpretable path around the underlying trend.

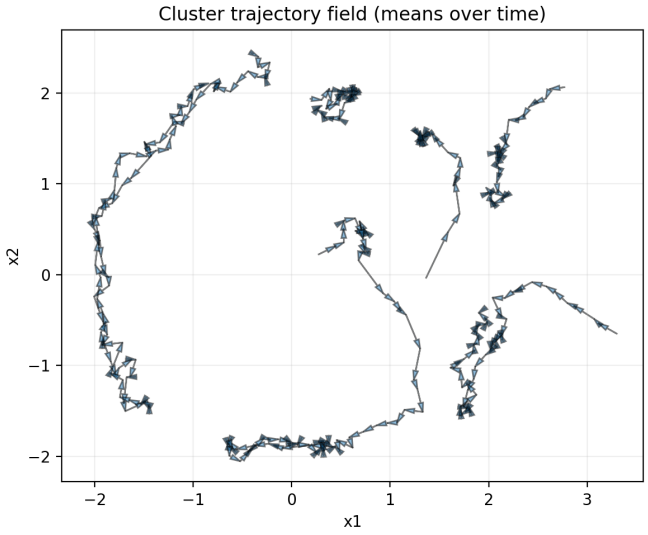

Fig. 2. Cluster trajectory field on synthetic data. Arrows link successive EWMA-smoothed cluster means, visualizing how clusters drift over time.

G. Algorithm Summary

Algorithm 1 summarizes the overall procedure.

V. Experiments

We evaluate Cluster-DAC on two settings: (i) synthetic drifting Gaussians, and (ii) a synthetic vehicle-like telemetry stream projected to two dimensions.

A. Synthetic Drifting Gaussians

We simulate two Gaussian clusters whose centers move along smooth trajectories in the plane and whose covariances rotate and stretch. Fig. 3 and Fig. 4 show the first and second half of the stream, respectively, with true samples overlaid by Cluster-DAC mean snapshots. Cluster-DAC tracks the drifting manifolds while keeping the cluster count near the budget.

Algorithm 1 Cluster-DAC (Streaming)

Initialize cluster set $C \leftarrow \emptyset$, time $t \leftarrow 0$.
for each incoming sample x_t **do**
  $t \leftarrow t + 1$.
  Compute distances $d_c(x_t)$ for all $c \in C$.
  if no cluster accepts x_t (distance $>$ threshold) **then**
    Create new cluster centered at x_t (or replace weakest if over budget).
  else
    Update the winning cluster using EWMA rules (2).
  end if
  if $t \bmod T_{\text{edit}} = 0$ **then**
    Apply structural edits: split, merge, prune, budget enforcement.
  end if
  Log cluster means, temperatures, and counts for visualization.
end for

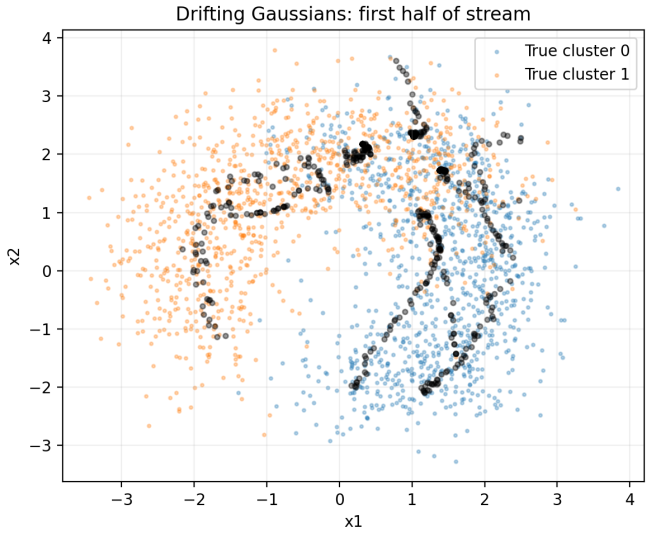

Fig. 3. Drifting Gaussians, first half of the stream. Samples colored by ground-truth component, with EWMA-smoothed cluster means overlaid.

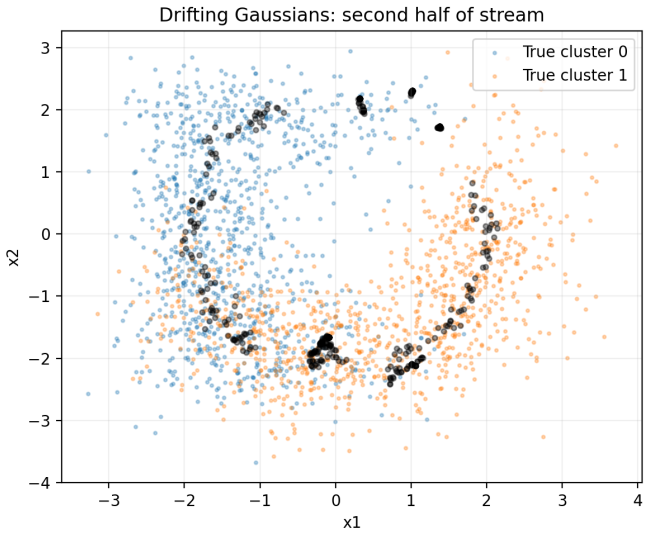

Fig. 4. Drifting Gaussians, second half of the stream. Cluster-DAC continues to follow the drift while maintaining a bounded number of clusters.

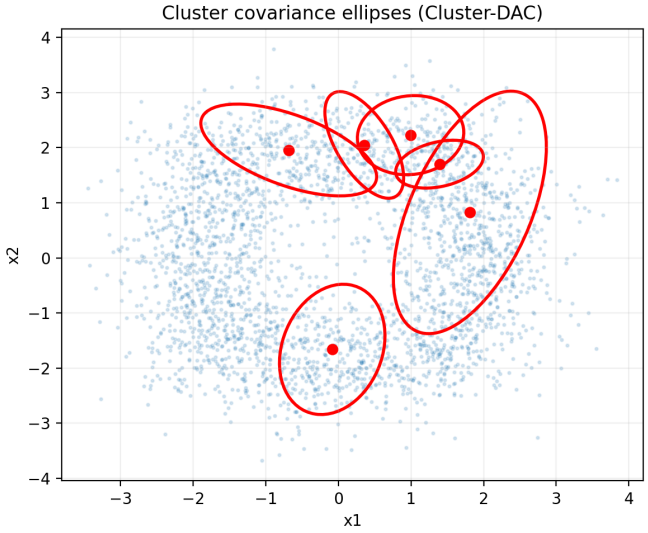

Fig. 5. Cluster covariance ellipses at the end of the synthetic experiment. Ellipses align with the local geometry of the drifting Gaussians.

Fig. 5 shows final covariance ellipses learned by Cluster-DAC. The ellipses adapt to directionality and curvature of the underlying drift, evidencing the advantage of anisotropic modeling.

Fig. 6 plots a pathline of cluster means over time, colored by snapshot index. Together with the trajectory field in Fig. 2, this gives an interpretable picture of how the cluster representation moves through feature space.

Fig. 7 displays the drift surface reconstructed from the final synthetic model. The surface is smooth and highlights regions of concentrated activity and drift. As noted above, its values range from roughly 3.2×10^{-37} to 1.42, with mean 0.062.

B. Structural Evolution and Temperature (Synthetic)

Fig. 8 shows the evolution of the number of clusters over time. The count remains between 1 and 6 with average 5.77, indicating that structural edits make use of most of the budget while avoiding uncontrolled proliferation.

Fig. 9 displays the synthetic temperature map. The mean temperature is approximately 0.54, with maximum 1.0. Rows (clusters) heat up when receiving assignments and cool down otherwise, yielding a compact drift-state map of which clusters are currently relevant.

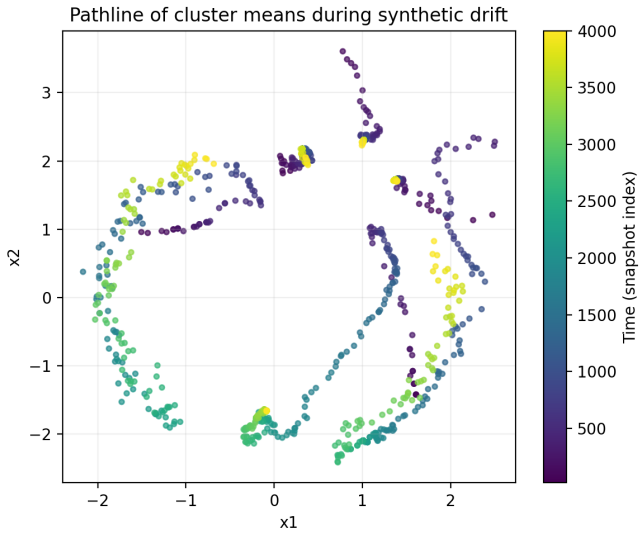

Fig. 6. Pathline of cluster means during the synthetic drift experiment. Color encodes time, revealing smooth trajectories of the learned cluster centers.

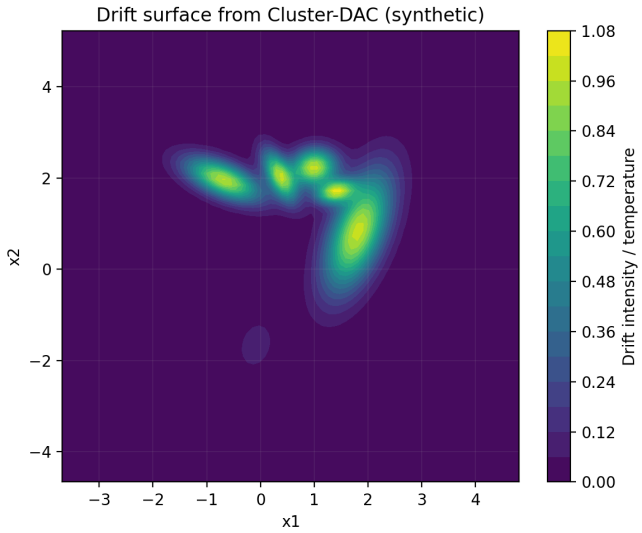

Fig. 7. Drift surface reconstructed from synthetic Cluster-DAC clusters. High-intensity regions correspond to frequently visited and high-density areas in feature space.

C. Vehicle-Like Telemetry

We generate a synthetic vehicle telemetry stream containing yaw rate, throttle, steering, and slip signals, with injected maneuver-like bursts. The 4D stream is projected to two principal components via PCA, and Cluster-DAC is applied in the resulting 2D space.

Fig. 10 shows the reconstructed drift surface in PCA space. Fig. 11 displays the temperature evolution across clusters. The mean cluster temperature for this experiment is roughly 0.50 (max 1.0), demonstrating that the model reuses a stable set of clusters across different driving phases.

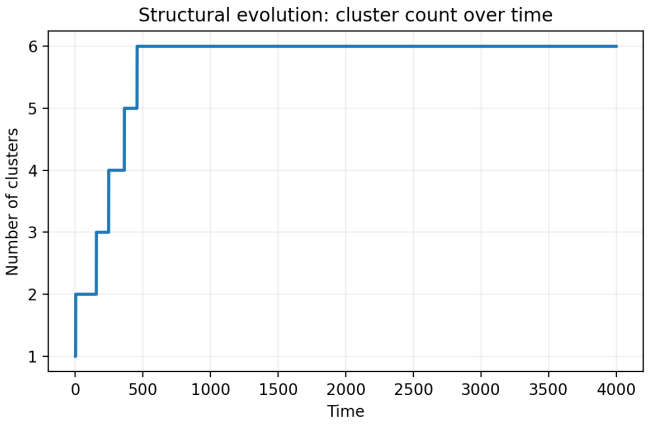

Fig. 8. Structural evolution: number of active clusters over time in the synthetic experiment. The count stays between 1 and the budget $K_{\max} = 6$ (mean 5.77).

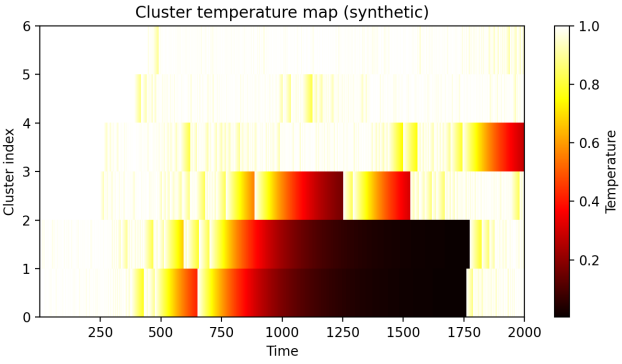

Fig. 9. Cluster temperature map for the synthetic data. Rows correspond to clusters, columns to time. Hot regions indicate recently active clusters; cool regions indicate inactive ones.

D. Ablation Study

To isolate the effect of EWMA and anisotropic covariance, we compare three variants on the synthetic drifting Gaussians:

1) **Full Cluster-DAC:** EWMA updates with $\lambda = 0.05$, Mahalanobis distance.
2) **No EWMA:** same as above but with $\lambda = 1.0$ (no temporal smoothing).
3) **No covariance modeling:** EWMA with $\lambda = 0.05$ but using Euclidean distance (no anisotropy).

We measure tracking error as the RMSE between true cluster means and the nearest model cluster mean at each time step. Fig. 12 displays the resulting error curves.

The average RMSE over time is:

- Full Cluster-DAC: 0.43 (final error 0.30),
- No EWMA: 1.51 (final error 0.85),
- No covariance (Euclidean): 0.42 (final error 0.21).

In this benchmark, EWMA is crucial: removing it increases mean error by roughly $3.5\times$. The Euclidean variant attains similar average RMSE to the full version on this simple 2D drift, although Mahalanobis modeling offers geometric

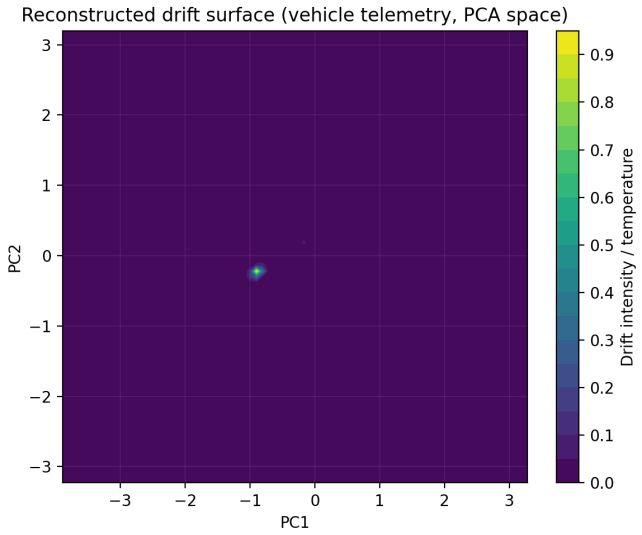

Fig. 10. Reconstructed drift surface for vehicle-like telemetry in PCA space. The surface summarizes the dominant modes and transitions in the driving data.

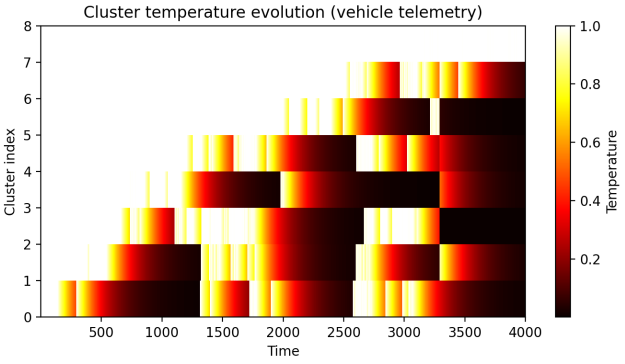

Fig. 11. Cluster temperature evolution for vehicle-like telemetry. The mean temperature ≈ 0.50 reflects recurrent reuse of a subset of clusters over time.

interpretability and is expected to be more beneficial for higher-dimensional, anisotropic data.

E. Runtime and Complexity

We report runtimes from the reference Python implementation on a CPU for 4000 time steps in each scenario. Table I summarizes the results.

For the synthetic experiment, Cluster-DAC processes about 4000 updates in $4.05\,\mathrm{s}$, corresponding to roughly $1.0\,\mathrm{ms}$ per update. For the vehicle-like telemetry in PCA space, the model processes 4000 updates in $0.63\,\mathrm{s}$, about $0.16\,\mathrm{ms}$ per update. These numbers are consistent with the expected $\mathcal{O}(Kd^2)$ per-step complexity for small d and modest budget K.

VI. Discussion

Cluster-DAC combines EWMA smoothing, anisotropic covariance, and structural editing into a single framework for clustering evolving data streams. The experiments show:

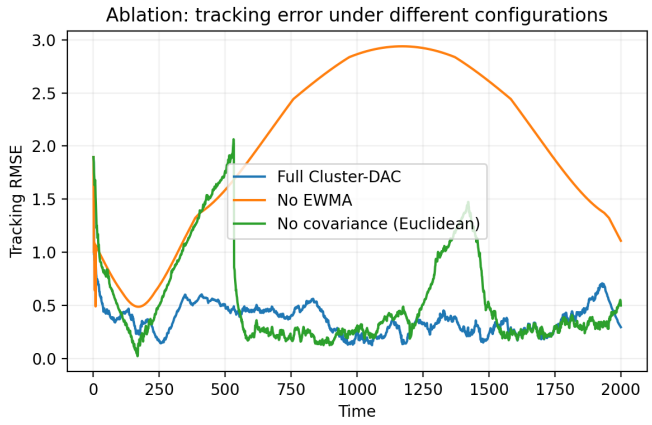

Fig. 12. Ablation: tracking RMSE for full Cluster-DAC, a variant without EWMA, and a variant without covariance-based Mahalanobis distance (Euclidean only). EWMA substantially improves stability and reduces error.

TABLE I
Runtime of Cluster-DAC for 4000 streaming updates.

Scenario	T	Time [s]	Time [ms/update]
Synthetic drifting Gaussians	4000	4.05	1.01
Vehicle-like telemetry (PCA)	4000	0.63	0.16

- **Temporal stability:** EWMA dramatically reduces tracking error compared to a no-smoothing variant.
- **Budget control:** cluster count remains between 1 and $K_{\max} = 6$ with average 5.77 in the synthetic experiment, evidencing that splits, merges, and pruning maintain a compact representation.
- **Interpretability:** trajectory fields, drift surfaces, and temperature maps provide a high-level view of where the stream is and how it moves.
- **Efficiency:** millisecond-level per-update runtimes in Python indicate that optimized implementations can be deployed in resource-constrained environments.

Limitations include sensitivity to the choice of λ and edit thresholds, and the use of a single decay scale for EWMA and temperature. Multi-timescale versions and adaptive thresholding are natural extensions.

VII. Conclusion

We presented Cluster-DAC, a drift-adaptive, budgeted clustering algorithm for evolving data streams. The method leverages EWMA updates, anisotropic covariance, and budgeted structural edits to track drifting manifolds without unbounded model growth. On synthetic drifting Gaussians and vehicle-like telemetry, Cluster-DAC achieves low tracking error, maintains a bounded number of clusters, and exposes interpretable visual diagnostics such as drift surfaces and temperature maps.

The framework provides a practical foundation for streaming applications and can be extended toward multimodal settings, hierarchical clustering, or integration with lightweight embeddings and symbolic descriptors.

REFERENCES

[1] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," in *Proc. SIGMOD*, 1996.

[2] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proc. VLDB*, 2003.

[3] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *Proc. SDM*, 2006.

[4] P. Kranen, I. Assent, C. Baldauf, and T. Seidl, "The ClusTree: indexing micro-clusters for anytime stream mining," *Knowledge and Information Systems*, 2011.

[5] B. Fritzke, "A growing neural gas network learns topologies," in *Proc. NIPS*, 1995.

[6] I. Vázquez-Chanlatte, S. Seshia, *et al.*, "Logical clustering and learning for time-series data," in *Proc. CAV*, 2017.