science, engineering and health studies

Research Article

Science, Engineering and Health Studies
https://lio1.tci-thaijo.org/index.php/sehs
ISSN (Online): 2630-0087

Volatility-informed representation for time-series clustering

*Prakarn Taranodom and Thapana Boonchoo**

Department of Computer Science, Faculty of Science and Technology, Thammasat University, Pathum Thani 12120, Thailand

ABSTRACT

Time series clustering is a challenging task due to the presence of noise, non-stationarity, and volatility in real-world data. Traditional methods that rely on raw data often fail to capture meaningful temporal patterns. To address this issue, a time series clustering framework called enhanced clustering using conditional volatility (ECCV) is proposed, which systematically integrates conditional volatility (CV) as an alternative feature representation to improve clustering accuracy. In the feature extraction stage, an autoregressive integrated moving average - generalized autoregressive conditional heteroskedasticity (ARIMA-GARCH) model is applied to generate CV from the residuals of time series data, filtering out noise, preserving local structural patterns, and capturing temporal fluctuations that are often underutilized by conventional methods. The extracted CV is then employed as input for a wide range of clustering algorithms, including partition-based, hierarchical, shape-based, spectral, and neural network-based approaches. The main novelty of this work lies in the systematic evaluation and integration of CV for time series clustering, which, to the best of our knowledge, has not been comprehensively investigated before. Its effectiveness is demonstrated across 26 benchmark datasets through both quantitative metrics (the Rand Index) and statistical validation using Wilcoxon signed-rank tests, as well as visualizations and domain-specific analyses to show selective but meaningful improvements. Experiments reveal that CV-enhanced features significantly improve clustering performance in several algorithms, particularly those leveraging latent representations or graph-based embeddings, although their impact varies depending on the dataset type and algorithm. Overall, ECCV provides a scalable, interpretable, and statistically grounded framework for leveraging volatility information in time series clustering, offering a novel perspective on representation learning in dynamic and volatile environments.

Keywords: conditional volatility; time-series; clustering; GARCH

***Corresponding author**:
Thapana Boonchoo
thapanab@tu.ac.th

***Received**: 9 June 2025*
***Revised**: 26 September 2025*
***Accepted**: 25 October 2025*
***Published**: 16 November 2026*

***Citation**:
Taranodom, P., & Boonchoo, T. (2026). Volatility-informed representation for time-series clustering. Science, Engineering and Health Studies, 20, 26020004.

1. INTRODUCTION

A time series is a series of data points indexed in time order used to analyze dynamic systems, such as trends and seasonal variations. In many applications, this data may be structured in such a way as to represent the time evolution of a particular phenomenon, and clustering such data usually involves forming meaningful clusters, where the time series sharing similar attributes are put into homogeneous clusters. This task is one of the most significant and challenging aspects of time series data mining, with applications across various domains, including finance (Kumar et al., 2002), biology (Fujita et al., 2012; Subhani et al., 2010), climate science (Steinbach et al., 2003), and medicine (Wismüller et al., 2002). One key challenge in time series clustering is measuring similarity

among sequences. Thus, the quality of clustering performance is affected most by the chosen similarity measures, which depend on how well features are extracted from the data. When feature extraction fails to capture meaningful temporal patterns, the similarity computations become unreliable, leading to poor clustering outcomes (Aghabozorgi et al., 2015; Wang et al., 2013). Current traditional approaches, such as Euclidean distance and dynamic time warping (DTW), generally do not deal well with high-dimensional noise or irregular time series data (Ding et al., 2008).

Regarding the presence of noise and non-stationarity in real-world time series, many financial and economic datasets exhibit fluctuations, trends, and structural shifts that make direct comparisons difficult (Fu, 2011). If a clustering approach relies solely on raw time-series data, it can be highly sensitive to these variations, resulting in distorted similarity measures. Standard preprocessing techniques, such as normalization and transformation, can mitigate some of these challenges, but they do not fully account for the complexity inherent in many real-world time series (Warren Liao, 2005). In particular, financial and economic data are often characterized by volatility patterns that influence underlying market dynamics, making it crucial to incorporate measures that capture these fluctuations more effectively (Cont, 2007).

Time series clustering methods are generally categorized into shape-based, feature-based, and model-based approaches (Aghabozorgi et al., 2015). Shape-based methods traditionally rely on Euclidean distance, limited to equal-length series (Petitjean et al., 2011); Paparrizos and Gravano (2016) improved this with k-shape clustering, using normalized cross-correlation to match DTW-based accuracy with better efficiency. However, such methods, including those using raw data directly, often overlook trends, seasonality, and time-dependent patterns, leading to suboptimal clustering on noisy or volatile real-world data, while raw-data input also brings noise sensitivity and computational inefficiency (Keogh & Ratanamahatana, 2005).

Feature-based clustering extracts key statistical properties from time series before clustering. Räsänen and Kolehmainen (2009) applied k-means to basic statistical features, while Li et al. (2013) used k-medoids with variation characteristics from piecewise aggregate approximation (PAA); such methods improve efficiency but often overlook temporal dependencies or the interplay between feature engineering and clustering. Conditional volatility offers an alternative feature representation: early work relied on Pearson correlation coefficients to measure similarity in stock returns (Mantegna, 1999; Bonanno et al., 2001), but neglected temporal dependencies, autocorrelation, volatility dynamics, and unequal sample sizes. More recent studies have turned to GARCH-based volatility—Ausloos et al. (2020) used GARCH and TGARCH models to reveal hidden market structures, though without extending to clustering; Wang et al. (2022) showed that GARCH distribution choice significantly affects volatility estimation accuracy, informing the model selection in this study; Lúcio and Caiado (2022) used TGARCH-based conditional volatility autocorrelations for clustering S&P 500 industries during COVID-19, though without benchmarking against raw returns; Sherly et al. (2025) showed ARIMA filtering improves volatility forecast accuracy; López-Oriona et al. (2025) applied ARIMA-based forecast-similarity measures to functional time series clustering; and Ma (2025) used ARIMA and ARMA–GARCH models to capture temporal dynamics, volatility clustering, and fat-tail behavior in Chinese GDP and stock data. Collectively, these studies show that feature-based and ARIMA–GARCH preprocessing can capture temporal and volatility patterns, but CV has rarely been systematically incorporated as a clustering feature.

Model-based approaches assume observed data is generated by an underlying model, treating two series as similar if produced by the same parameters; these include statistical feature extraction (Nanopoulos et al., 2001), ARMA models (Kalpakis et al., 2001), Markov Chains (Sebastiani et al., 1999), and Hidden Markov Models (Panuccio et al., 2002), with Markov Chains simpler but less expressive. Time-series bitmaps (Kumar et al., 2005), though primarily for visualization, can also be viewed as model-based. GARCH-based variants include Caiado and Crato (2007), who used GARCH parameters to estimate distances between return fluctuations; Otranto (2008), who separated volatility into constant and time-varying parts; and Caiado and Crato (2010), who combined TGARCH parameters with a periodogram of squared returns for DJIA clustering. However, model-based approaches suffer from scalability problems and degrade when clusters are close together (Mitsa, 2010).

Beyond clustering, several studies have directly compared GARCH-type models for volatility forecasting. Ali (2013) found TGARCH performed marginally better for pathogen indicator time series across diverse distributions; Lim and Sek (2013) found TGARCH excelled in pre- and post-crisis periods for Malaysian stock market volatility, while GARCH was more effective during the crisis; Sahiner (2022) found EGARCH outperformed at higher time frames and symmetric GARCH models at lower time frames across ten Asian markets, with GARCH-type models overall showing strong adaptability (confirmed by the Diebold-Mariano test); and Marisetty (2024) confirmed across major global indices (FTSE 100, Hang Seng, NIKKEI 225, NSE 50, S&P 500) that no single model consistently outperforms others, with TGARCH and higher-order GARCH models capturing asymmetric volatility more effectively under specific conditions.

These studies confirm that no single GARCH model consistently outperforms others across datasets, with performance depending on dataset characteristics, time frames, and market conditions; however, most focus on forecasting accuracy rather than using the resulting CV as a feature for downstream tasks such as clustering, leaving its systematic use as an unsupervised learning input rare or limited. Multiple GARCH models are therefore systematically compared, and the optimal model is selected based on AIC, diagnostics and use CV as a clustering feature—capturing time-dependent volatility variations to improve clustering accuracy and robustness across diverse datasets.

These issues are addressed using a two-stage ARIMA–GARCH modeling framework. ARIMA is first applied to remove trends and seasonality, ensuring stationarity and reducing noise (Box et al., 2016; Tsay, 2010), since applying volatility models directly to raw data can otherwise yield biased estimates due to autocorrelation, structural shifts, and non-stationarity (Hamilton, 1994); by extracting residuals containing primarily stochastic fluctuations, ARIMA ensures the subsequent GARCH model can accurately capture true volatility dynamics without interference from deterministic patterns. The

residuals are then passed into a GARCH model (Engle, 1982; Bollerslev, 1986), which models heteroskedasticity to produce conditional volatility (CV) as a more reliable feature of the series' underlying dynamics—a combination widely validated in financial econometrics (Tsay, 2010). While ARIMA–GARCH has been extensively applied to volatility forecasting (Srihari et al., 2024), its systematic use for generating CV as a clustering feature remains limited, and even recent benchmark evaluations of time series clustering (e.g., López-Oriona et al., 2025) have not explored volatility-based features such as CV. Incorporating CV instead of raw time-series data thus provides a less noisy, more representative signal that enhances similarity measures and improves clustering algorithms' ability to identify natural groupings in volatile, complex datasets.

2. MATERIALS AND METHODS

In this section, a framework to enhance the clustering performance is presented on time-series data using conditional volatility (ECCV for short). The ECCV framework consists of two main steps: feature extraction using time series modeling, and a clustering model. The overall framework is shown in Figure 1.

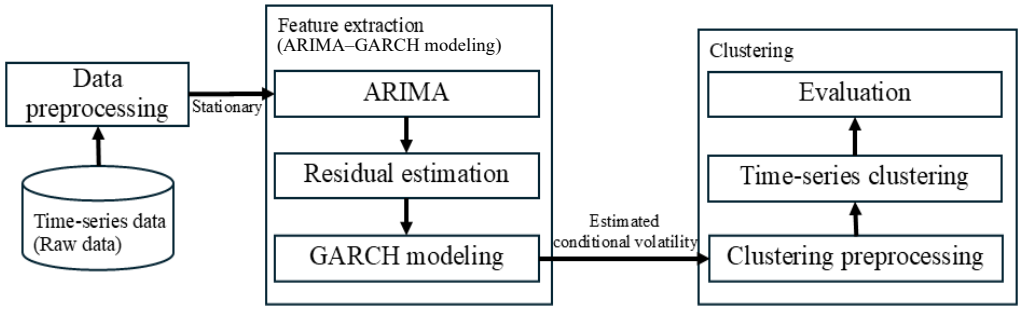


Figure 1. Overall ECCV Framework

2.1 Feature extraction using time series modeling

In this step, meaningful features are extracted from raw time series data to improve clustering performance. This process consists of 5 steps: pre-processing, stationarity check, SARIMA modeling, residual estimation, and GARCH modeling. Each step ensures that the extracted features capture the essential time-dependent dynamics of the data before proceeding to clustering. The following subsections provide a detailed description of each step.

2.1.1 Pre-processing before ARIMA model

Before applying any time series modeling, time series data requires pre-processing to ensure stationarity, which is essential to avoid misleading model estimates. The stationarity of each raw time series is tested using the augmented dickey–fuller (ADF) test (Dickey & Fuller, 1979), considering both constants and trends in the time series. The ADF test is formulated as follows:

$$\Delta Y_t = \alpha + \beta t + \gamma Y_{t-1} + \sum_{i=1}^{p} \delta_i \Delta Y_{t-i} + \epsilon_t \quad (1)$$

where Y_t is the time series value at time t, α is a constant term, β represents a time trend component, γ is the coefficient that determines whether the time-series is stationary, ϵ_t is the error term, ΔY_t denotes the first difference of the time series, and δ_i represents lagged differences up to order p. The null hypothesis (H_0) states that the time series has a unit root, meaning it is non-stationary, while the alternative hypothesis (H_1) suggests that the series is stationary. If the test fails to reject H_0, meaning the time series is non-stationary, then transformations, such as differencing, logarithmic transformation, or power transformations, are applied to stabilize the mean and variance. If stationarity is not achieved, subsequent models may produce unreliable inferences, leading to incorrect clustering results.

2.1.2 Time series modeling

To model the time series data effectively, the ARIMA-GARCH approach is employed in a two-step estimation process. The ARIMA model captures the underlying trend and autocorrelation structure, while the GARCH model estimates CV, a key component in this study for clustering analysis.

2.1.2.1 ARIMA model

Once stationarity is confirmed, the ARIMA model (Box et al., 2016) is applied to capture the underlying trend and autocorrelation structure of a time series, while the GARCH model estimates CV, which is crucial for clustering analysis. After confirming stationarity, the ARIMA model is applied to model the trend and autocorrelation structure. The general form is as follows:

$$\left(1 - \sum_{i=1}^{p} \phi_i B^i\right)(1 - B)^d y_t = \left(1 + \sum_{j=1}^{q} \theta_j B^j\right)\varepsilon_t \quad (2)$$

where y_t is the observed time series data at time t, ϕ_i and θ_j are the autoregressive and moving average parameters for the non-seasonal components, and B is the backshift operator $B^k y_t = y_{t-k}$, and k is the number of periods shifted back.

The model is typically denoted as $ARIMA(p, d, q)$ where p and q are the orders of the non-seasonal autoregressive and moving average components, and d is the non-seasonal differencing order. The error term ε_t represents a white noise process or residual.

When seasonality in time series data is considered, the model is extended to SARIMA (seasonal ARIMA), which incorporates seasonal autoregressive terms (P), seasonal differencing (D), and seasonal moving average components (Q). The general form of the SARIMA model is expressed as:

$$\left(1 - \sum_{i=1}^{p} \phi_i B^i\right)(1 - B)^d \left(1 - \sum_{i=1}^{P} \Phi_i B^{is}\right)(1 - B^s)^D Y_t = \left(1 + \sum_{j=1}^{q} \theta_j B^j\right)\left(1 + \sum_{j=1}^{Q} \Theta_j B^{js}\right)\epsilon_t \quad (3)$$

where the parameters Φ_i and Θ_j represent the seasonal autoregressive and seasonal moving average components, respectively. The term s denotes the seasonal period, such as 12 for monthly data with yearly seasonality. Importantly, when all seasonal orders P, D, and Q are zero, and there is no seasonal period s, the SARIMA model reduces to the standard ARIMA model. Therefore, ARIMA can be viewed as a special case of SARIMA in the absence of seasonal components.

Optimal ARIMA parameters are selected using the Autocorrelation function (ACF) and partial autocorrelation function (PACF), along with minimizing the Akaike information criterion (AIC) (Akaike, 1998), which evaluates model fit by balancing goodness of fit and complexity:

$$AIC = 2k - 2\ln(L) \quad (4)$$

where k is the number of parameters and L is the model likelihood. AIC is widely used in time-series analysis due to its ability to select models that fit well while avoiding excessive complexity, reducing the risk of overfitting. To ensure model adequacy, the Ljung-Box test (Ljung & Box, 1978) is performed on the residuals to check for autocorrelation. A high p-value suggests that residuals are independently distributed, confirming that the ARIMA model adequately captures time series dynamics.

2.1.2.2 GARCH model

When the residuals are obtained from the ARIMA model, they are analyzed using the GARCH model to estimate time-varying volatility. The GARCH model, proposed by Bollerslev (1986), generalizes the ARCH model by allowing both autoregressive (AR) and moving average (MA) components for modeling conditional volatility. The GARCH model equation is defined as follows:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2 \quad (5)$$

where σ_t^2 is the conditional variance at time t, α_0 is a constant, α_i represents the impact of past squared residuals, and β_j represents the influence of past conditional variances, and p and q are the lag orders of the model.

The error term ϵ_t follows a specific probability distribution known as the error distribution, which can take different forms, such as normal distribution, student's T-distribution, and generalized error distribution (GED). The choice of error distribution significantly affects the accuracy of conditional variance estimation, as different distributions handle heavy tails and skewness differently.

However, to account for the diverse characteristics of datasets and better capture complex volatility patterns, various GARCH-type models have been developed. These models address the limitations of the standard GARCH model, such as its inability to capture asymmetry or leverage effects. The model variations of GARCH include:

Threshold GARCH (TGARCH) – this was proposed by Zakoian (1994). The TGARCH model introduces an asymmetry term to differentiate the impact of positive and negative shocks on volatility. This is particularly useful for datasets where negative shocks have a more significant effect than positive ones, such as stock market downturns.

Exponential GARCH (EGARCH) – this was developed by Nelson (1991). The EGARCH model addresses the shortcomings of symmetric GARCH models by capturing leverage effects and allowing for asymmetric impacts of shocks on volatility. Unlike the standard GARCH model, EGARCH does not require non-negativity constraints on parameters, enhancing its flexibility in modeling financial data.

Fractionally Integrated GARCH (FIGARCH) – this was introduced by Baillie et al. (1996). The FIGARCH model is designed to capture long memory in volatility processes. This model is particularly useful for datasets where persistence in volatility is observed over extended periods, such as in commodity markets.

Other GARCH Variants: *Integrated GARCH (IGARCH)* was proposed by Engle and Bollerslev (1986), which was a special case of GARCH with a unit root in the variance equation, often used for highly persistent volatility. *GJR-GARCH*, a type of TGARCH model proposed by Glosten et al. (1993), specifically addresses leverage effects in asset returns.

The selection of the optimal GARCH model follows the same criteria as ARIMA, using the AIC, mean squared error (MSE), Root Mean Squared Error (RMSE), and mean absolute error (MAE) to evaluate model performance. Additionally, the Durbin-Watson statistic and Ljung-Box test are used to ensure residuals exhibit minimal autocorrelation. These GARCH-based models serve as key approaches for estimating CV, which will be used as an input feature for clustering analysis in the next step. For representation, the CV matrix obtained from the ARIMA-GARCH Model is defined as:

$$\Sigma = \left[\sigma_{c_1}^2\ \sigma_{c_2}^2\ \dots\ \sigma_{,c_N}^2\right], \text{where } \sigma_{c_i}^2 = [\sigma_1^2\ \sigma_2^2\ \dots\ \sigma_T^2]^T \quad (6)$$

where Σ denotes a matrix where each column vector $\sigma_{c_i}^2$ represents the CV of the company c_i over time. Each $\sigma_{c_i}^2$ is a $T \times 1$ vector composed of the estimated variances σ_t^2 for all time points $t = 1, \dots, T$. Let T be the total number of time points and N the number of companies. Hence, the matrix Σ is of size $T \times N$.

2.2 Clustering

This step discusses the clustering process, starting with the importance of pre-processing and continuing with the evaluation of various clustering models. Clustering methods are discussed in this section, each with its unique strengths and limitations for time series data clustering.

2.2.1 Pre-processing before clustering model

Pre-processing, including feature scaling and transformation, is essential for clustering, and similar steps should be applied to CV, just as they are for raw data. CV, like raw data, can exhibit varying scales and distributions, which may distort distance calculations in clustering algorithms. To ensure clustering effectiveness, scaling methods such as normalization, standardization, and robust scaling were applied to both raw data and CV. These methods help align the distributions, making sure that both data types contribute equally to the clustering process (Han et al., 2012).

Additionally, transformations like logarithmic adjustments and Yeo-Johnson were considered to address any skewness in the data. For time-series data, autocorrelation was applied to CV to capture time-dependent relationships and reduce noise. By applying the same pre-processing techniques to both CV and raw data, the data was ensured to be in a comparable form, leading to more accurate and stable clustering results.

2.2.2 Clustering models

Nine widely used clustering algorithms covering the main paradigms were selected in time series clustering—partition-based (K-means, KDBA, kernel-K-means, Clara, and Clarans), hierarchical, shape-based (K-shape), spectral (spectral clustering), and neural network-based (N2D) methods—to ensure broad coverage and evaluate the robustness and general applicability of the proposed CV feature across diverse clustering approaches. These algorithms are widely adopted in literature and readily available in standard libraries, allowing efficient implementation and ensuring that the conclusions of this study are not limited to a specific algorithm class. The following describes each clustering technique used, including its underlying principles, advantages, and limitations.

2.2.2.1 K-means

The K-means algorithm (MacQueen, 1967) partitions data into k clusters by minimizing within-cluster variance, typically using Euclidean distance:

$$d_{\text{Euclidean}}(x_i, \mu_j) = \sqrt{\sum_{d=1}^{D}(x_{i,d} - \mu_{j,d})^2} \quad (7)$$

While computationally efficient, Euclidean distance assumes spherical clusters and may not perform well for time-dependent data. To address this, dynamic time warping (DTW), which aligns sequences was incorporated by minimizing temporal distortions. Since standard K-means lacks a proper centroid update for DTW, K-means with DBA (KDBA) was adopted, using DTW barycenter averaging (DBA) originally proposed by Petitjean et al. (2011), to compute centroids by minimizing the sum of DTW distances between a candidate centroid and the cluster members:

$$\mu_j^{\text{DBA}} = \arg\min_{\mu} \sum_{i=1}^{N} d_{\text{DTW}}(x_i, \mu) \quad (8)$$

KDBA enhances clustering for sequential data, but at a higher computational cost. To improve centroid initialization and clustering stability, K-means++ (Arthur & Vassilvitskii, 2007) was applied to both Euclidean and KDBA-based approaches. By integrating these techniques, adaptability was ensured for both static and time-dependent datasets.

2.2.2.2 CLARA

CLARA, proposed by Kaufman and Rousseeuw (1990), reduces computational complexity by applying K-medoids to multiple randomly drawn subsets of the dataset instead of the entire dataset. By selecting the best clustering result from these samples based on the sum of pairwise dissimilarities, CLARA maintains robustness to noise and outliers while significantly improving scalability. However, its reliance on sampled subsets may lead to suboptimal clustering when important data points are excluded.

To address this limitation, CLARANS, introduced by Ng and Han (2002), enhances the medoid selection process through a randomized search. Instead of working with fixed subsets, CLARANS iteratively replaces medoids with randomly chosen candidates and evaluates whether the new configuration improves clustering quality. This adaptive search mechanism allows CLARANS to explore a larger solution space, helping to avoid local minima and find better cluster assignments. Despite its advantages, CLARANS can be more computationally intensive than CLARA, particularly for high-dimensional data.

Both methods aim to minimize the sum of dissimilarities within clusters, formulated as:

$$\sum_{i=1}^{N}\sum_{j=1}^{k} I(x_i \in C_j)\, d(x_i, m_j) \quad (9)$$

where $d(x_i, m_j)$ represents the dissimilarity between data point x_i and the medoid m_j.

While CLARA prioritizes computational efficiency by working with smaller samples, CLARANS improves clustering accuracy through a more dynamic and exploratory search process.

2.2.2.3 Spectral clustering

Spectral clustering (Ng et al., 2001) is an alternative method for partitioning data that constructs a similarity graph and projects data into a lower-dimensional space using eigenvalues of the graph Laplacian. The method starts by computing the Laplacian matrix, defined as:

$$L = D - W \quad (10)$$

where W is the similarity matrix (or adjacency matrix) that captures the pairwise relationships between data points and is the diagonal degree matrix with $D_{ii} = \sum_j W_{ij}$, where i and j are data points. The similarity graph encoded in L is then used to solve the eigenvalue problem $Lv = \lambda v$, where v represents the eigenvectors associated with the smallest eigenvalues λ. These eigenvectors serve as new representations of the data, onto which standard clustering algorithms, particularly K-means, are applied.

A common choice for constructing the similarity matrix W is the radial basis function (RBF) kernel, also known as the Gaussian affinity function. This function measures the similarity between two data points x and y based on their Euclidean distance, assigning higher similarity values to closer points and lower values to distant ones. The RBF kernel is defined as:

$$W_{ij} = \exp\left(-\gamma |x_i - x_j|^2\right) \quad (11)$$

where γ is a parameter that controls the scale of the similarity measure. The use of the RBF kernel helps spectral clustering capture non-linear structures in the data, making it effective for detecting clusters with arbitrary shapes. However, careful tuning of $\gamma$is necessary to balance sensitivity to local versus global structures, and the computational cost of eigenvalue decomposition makes spectral clustering expensive for large datasets.

2.2.2.4 K-shape

K-shape (Paparrizos & Gravano, 2016) is a clustering method designed specifically for time-series data. Unlike K-means, which uses Euclidean distance, K-shape utilizes a shape-based similarity metric that captures the temporal patterns of time-series data more effectively. The algorithm computes the centroids by aligning the time-series data before clustering. The objective function minimizes the dissimilarity between the data points and the centroids using a normalized cross-correlation (NCC) measure. Specifically, the dissimilarity between time-series x_i and the cluster centroid C is given by $\text{NCC}(x_i, C)$, and the algorithm attempts to minimize the sum of these dissimilarities

across all time-series in the dataset. K-shape is particularly well-suited for time-series data exhibiting periodicity or trend patterns. However, it can be computationally expensive due to the need for time-series alignment.

2.2.2.5 Kernel K-means

Kernel K-means (Dhillon et al., 2004) is an extension of traditional K-means that leverages a kernel function to implicitly map data onto a higher-dimensional feature space. This enables the algorithm to capture non-linear structures in the data. In the context of time-series clustering, the global alignment kernel (GAK), introduced by Cuturi (2011), is particularly effective. Unlike the RBF kernel, which relies on fixed Euclidean distances, GAK is derived from DTW and allows for temporal misalignments. Crucially, GAK is a positive-definite kernel, ensuring its compatibility with kernel-based methods like Kernel K-means.

GAK computes similarity by summing over all possible alignments between two time series, weighting each alignment path based on a smoothing parameter that controls sensitivity to misalignments. It is defined as:

$$K_{\mathrm{GAK}}(X,Y) = \sum_{\pi \in P(X,Y)} e^{-\lambda c(\pi)} \tag{12}$$

where $P(X,Y)$ represents the set of all alignment paths between time series X and Y, $c(\pi)$ is the cost of alignment path π and λ is a smoothing parameter controlling sensitivity to misalignments.

This flexibility makes GAK particularly suitable for datasets with variable-length sequences or phase variations. However, similar to other kernel-based approaches, GAK significantly increases computational complexity due to the need for pairwise similarity calculations across sequences.

2.2.2.6 Agglomerative clustering

Agglomerative clustering (Ward, 1963) is a bottom-up hierarchical method that iteratively merges clusters based on a predefined linkage criterion. The algorithm starts with each data point as an individual cluster and gradually merges with the closest clusters. The merging criterion is based on a measure of distance between clusters, such as Ward's method, which minimizes the variance within clusters. At each step, the algorithm calculates the pairwise distances between clusters and merges the two clusters that result in the smallest increase in total variance. Agglomerative clustering is flexible in that it does not require the number of clusters to be predefined. However, it is computationally expensive because it requires calculating pairwise distances at each iteration, which makes it less suitable for large datasets.

2.2.2.7 N2D

N2D (McConville et al., 2019) is a deep learning-based clustering method that utilizes an autoencoder to project high-dimensional data into a lower-dimensional latent space prior to clustering. The autoencoder is composed of two parts: an encoder function $f(\cdot)$ that maps an input x to a latent representation z, and a decoder function $g(\cdot)$ that reconstructs the input from z. The model is trained by minimizing the reconstruction loss between the original input and its reconstruction, formulated as:

$$L(x,\hat{x}) = \left|x - g\big(f(x)\big)\right|^2 \tag{13}$$

Once the autoencoder is trained, the latent representations z are extracted, and clustering is applied in this reduced space using standard algorithms, such as K-means. This approach enables effective clustering of complex or nonlinear data structures, such as images or text documents. However, N2D typically requires significant computational resources for model training and is sensitive to the choice of hyperparameters, which may limit its scalability in large real-world datasets.

3. RESULTS AND DISCUSSION

In this study, the performance of the proposed CV, derived from GARCH models, is examined across various clustering algorithms to investigate whether it enhances clustering performance compared to conventional time-series clustering on 26 benchmark datasets.

3.1 Experimental setup

To ensure a fair comparison, all experiments follow a consistent setup. The primary dataset consists of 26 time series datasets obtained from the UCR archive (Dau et al., 2019), which serves as the testbed for evaluating the proposed ECCV.

3.1.1 Time series model setup

The modeling process consists of two main components: ARIMA for capturing the trend and autocorrelation structure of a time series, and GARCH for modeling volatility dynamics. The selection and fitting of both models are conducted systematically using predefined statistical criteria to ensure optimal performance. For ARIMA model selection, the auto_arima function from the pmdarima library in Python is used to automatically determine the optimal ARIMA model by minimizing the AIC. To ensure stationarity, the ADF test is conducted. If the data is non-stationary, appropriate transformations, such as differencing, are applied. Seasonality is detected through decomposition, and an ADF test is performed on the seasonal components. If seasonality is present, auto_arima incorporates seasonal parameters P, D, Q, and s into the model selection process to account for seasonal effects. Residuals are validated using the Ljung-Box test at a significance level of α = 0.05 to ensure minimal autocorrelation. If residuals exhibit autocorrelation, indicating that the model does not fully capture the underlying data structure, alternative models with more appropriate statistical properties are considered, even if they yield slightly lower accuracy, thereby ensuring robustness and better adherence to theoretical assumptions.

After obtaining ARIMA residuals, various GARCH-type models were fitted. For selection, various variants are explored, including Standard GARCH, TGARCH, EGARCH, GJR-GARCH, FIGARCH, and FIAGARCH models. Different error distributions, such as normal, student's t, and GED, are also tested to evaluate which best fits the data. The models are compared using several evaluation metrics, including AIC, MSE, Durbin-Watson Statistic, and the Ljung-Box test. AIC is used to evaluate the model fit, where a lower AIC suggests a better model by balancing fit and complexity. MSE measures the average squared difference between predicted and actual values. A lower MSE indicates better predictive accuracy. The Durbin-Watson Statistic is used to check for autocorrelation in the residuals, with values between 1.5 and 2.5 indicating no significant autocorrelation, which is desirable for model validity. The Ljung-Box test tests the residuals for autocorrelation, and a p-value greater

than 0.05 suggests that no significant autocorrelation exists, confirming that the model assumptions are satisfied. The final model is selected based on the lowest AIC and MSE, while ensuring the Durbin-Watson statistic falls between 1.5 and 2.5 and the Ljung-Box test yields a p-value greater than 0.05. This structured approach ensures robust time series modeling and enhances clustering accuracy by leveraging volatility dynamics effectively.

3.1.2 Clustering model setup

Before applying clustering algorithms, the time series data is preprocessed using various scaling methods and transformations to optimize clustering performance. Data scaling is essential, as different clustering algorithms are sensitive to variations in data distribution. To evaluate the impact of conditional volatility on clustering performance, nine clustering methods, categorized into three main types with different distance measures, are employed in this study (Table 1). To ensure a fair and consistent evaluation, uniform parameter settings are maintained across all clustering experiments. By standardizing the experimental setup, improvements in clustering performance can be attributed to the incorporation of conditional volatility rather than variations in parameter settings.

Table 1. Summary table of clustering algorithms

Clustering algorithm	Distance measure	Category
K-means	Euclidean	Partition-based
K-means (DTW) / KDBA	DTW	Partition-based
CLARA	Euclidean	Partition-based
CLARANS	Euclidean	Partition-based
Kernel K-means	Kernel-based (GAK)	Partition-based
Spectral clustering	RBF	Spectral
K-shape	Shape-based	Shape-based
Agglomerative	Euclidean	Hierarchical
N2D (deep autoencoder)	Mahalanobis	Neural-network-based

K-means clustering is implemented using both Euclidean and DTW distance metrics. The Euclidean variant follows the standard k-means++ initialization and runs for a maximum of 50 iterations to ensure convergence. In contrast, K-means with DTW applied allows for a small warping window, such as a Sakoe Chiba radius of 25, which balances flexibility in time-series alignment with computational efficiency and prevents overfitting to noise or outliers. CLARA and CLARANS are both applied using Euclidean distance. These methods enhance the scalability of clustering by sampling subsets of data. CLARA performs multiple random samplings to approximate K-medoids clustering, while CLARANS further optimizes medoid selection through randomized local searches, reducing computational overhead compared to standard K-means. Spectral clustering is implemented with affinity 'nearest_neighbors', which constructs a similarity graph by connecting each data point to its closest neighbors. The clustering assignments are determined using orthogonal matrix factorization for clustering (ClusterQR), ensuring numerical stability and robustness, particularly in cases where standard k-means might fail due to non-convex cluster shapes. The code preprocesses the data to ensure proper dimensionality before fitting the model. K-shape clustering, specifically designed for time-series data, applies a shape-based distance metric to group similar temporal patterns. Kernel K-means utilizes the GAK, a similarity measure tailored for time-series data, capturing complex temporal patterns beyond Euclidean distance. This kernel method allows for nonlinear clustering structures by mapping data into a high-dimensional space. Agglomerative clustering employs different linkage methods depending on the chosen distance metric. When using Euclidean distance, linkage='ward' is applied, which minimizes the variance within clusters, leading to compact and well-separated groups. N2D transforms the original data into a lower-dimensional representation before clustering with a GMM. This approach enables soft clustering, where each point is assigned to multiple clusters probabilistically rather than deterministically. The implementation follows a structured process: UMAP-GMM is applied to reduce dimensionality using UMAP while clustering the transformed space with GMM. This combination enhances the separation of clusters in a way that retains the inherent structure of the data. The N2D model integrates an autoencoder with the UMAP-GMM clustering pipeline, leveraging the learned feature representation to improve clustering performance. In the final clustering step, the model is trained, and clustering assignments are generated. The method is optimized using batch processing and runs for a specified number of epochs to balance accuracy and efficiency. The other parameters besides the above will use the default values from the library.

3.2 Effect of CV on clustering performance

To evaluate the impact of CV, clustering results using raw data and CV-adjusted data are compared across 26 datasets. The performance is measured using the Rand index (RI), which assesses clustering similarity by quantifying how well data points are grouped together or separated in both the predicted and true labels. Each algorithm is implemented based on the previous section.

The RI is defined as:

$$RI = \frac{TP + TN}{TP + TN + FP + FNRI} \tag{14}$$

where TP (true positives) represents the number of pairs of points that are correctly clustered together in both the predicted and true clustering. TN (true negatives) is the number of pairs of points that are correctly placed in separate clusters in both the predicted and true clustering. FP (false positives) refers to the number of pairs of points that are clustered together in the predicted clustering but are actually in separate clusters in the true clustering. FN

(False Negatives) is the number of pairs of points that are clustered separately in the predicted clustering but belong to the same cluster in the true clustering.

The RI ranges from 0 to 1, with 1 indicating perfect clustering (exact match with true labels) and 0 indicating performance no better than random assignment; higher values thus reflect better clustering performance. Comparing RI scores between raw data and CV-adjusted data lets us assess whether CV adjustment improves clustering consistency with the ground truth.

Table 2 shows the clustering performance across the 26 datasets from the UCR archive that were evaluated using the RI, comparing raw data with CV-adjusted data. The results indicate that CV transformation affects different clustering methods in distinct ways—some methods benefit from the transformation, while others show slight decreases or remain unchanged.

Among all clustering methods, N2D demonstrated the highest performance gain following CV transformation. Its average RI improved from 0.600 to 0.654, corresponding to an 8.93% increase. This suggests that CV features enrich the internal representations learned by deep architecture, possibly by emphasizing temporal irregularities or structural variance that raw features may underrepresent. SC also showed notable improvement, with its average RI increasing from 0.605 to 0.657, a gain of 8.53%. This indicates that volatility-aware transformation enhances the graph structure used in spectral embeddings, leading to improved cluster separability in the transformed feature space. Clustering methods such as CLARA, K-means (Euclidean), K-Shape, and Kernel K-means also benefited, though to a lesser extent. CLARA improved by 6.87%, and K-means, a baseline method, improved from 0.588 to 0.620, a gain of 5.55%, suggesting moderate compatibility between CV and Euclidean distance-based partitioning. K-shape, which is sensitive to the shape of the time series, improved by 4.28%, while Kernel K-means exhibited a gain of 3.66%, implying that the addition of CV provides useful nonlinear separation characteristics. KDBA, which is based on DTW, also saw a positive gain of 3.35%. This is worth noting because in earlier tests with different datasets, KDBA's performance occasionally decreased when CV was applied. The gain observed in this benchmark suggests that the effectiveness of CV in alignment-based methods like KDBA may be sensitive to the specific structure of the dataset or the consistency of temporal patterns. CLARANS and agglomerative clustering showed smaller improvements of 3.78% and 0.80%, respectively, suggesting that volatility-adjusted features might be less aligned with hierarchical clustering paradigms or graph-based density methods.

Furthermore, the group of SPECTRO datasets observed the most benefit on average, with RI gains of approximately 1.33%, suggesting that frequency-domain volatility is particularly informative in distinguishing between clusters. For the TRAFFIC dataset (Chinatown), the experiment also showed a similar improvement (1.20%), because directional flows and peak variations—commonly captured through volatility—are critical to clustering in such data. ECG datasets showed inconsistent results, with an average change of -2.07%, indicating that CV sometimes decreased clustering performance in these highly regular and periodic signals. HAR datasets exhibited modest improvement (~1.3%), while MOTION datasets benefited substantially, with an average gain of approximately 8.85%, reflecting that temporal dynamics and short-term fluctuations in these domains can be better captured through CV.

By contrast, IMAGE datasets, which constitute the largest portion of the benchmark, showed small overall improvements (~0.23%), and SENSOR datasets experienced a slight decline (-3.53%). This suggests that for IMAGE and SENSOR types, either the raw data already captures most structural information, or the volatility signal adds limited or potentially disruptive information in these contexts. In addition, some individual datasets were found to vary widely in how much they benefited from the CV adjustment—even within the same type. For example, datasets such as Beef, BirdChicken, DistalPhalanxOutlineCorrect, and GunPoint consistently exhibited higher RI values across multiple clustering algorithms when CV was applied. This suggests that local volatility patterns—such as fluctuations in amplitude or variability near class transitions—are critical in enhancing cluster boundaries for these particular time series. Conversely, datasets such as SonyAIBORobotSurface2 and ToeSegmentation1 showed negligible gains or even slight declines in performance. This may be attributed to the fact that these datasets already exhibit strong class separability in raw form or lack significant volatility structure, making the contribution from CV minimal or even detrimental in some cases.

Taken together, these results affirm that CV adjustment serves as a powerful transformation in time series clustering—particularly when applied to methods that rely on latent representations or shape-based comparisons. The fact that N2D and Spectral Clustering responded strongly suggests that CV contributes critical information that enhances both deep and graph-based structural learning.

3.3 Visualization

In this section, the comparison between raw and CV-based clustering is visualized and quantified. Figure 2 presents scatter plots of adjusted RI scores, where the horizontal axis corresponds to raw data and the vertical axis to CV data, separated by algorithm.

The distribution of points in Figure 2 is consistent with the percentages summarized in Table 3. SC stands out with 69.2% of datasets showing improvements, and very few declines (7.7%), which is reflected by the cluster of circle markers above the diagonal. N2D (61.5% CV > Raw) and KDB (61.5% CV > Raw) similarly exhibit many points above the diagonal, confirming that volatility-based features are particularly beneficial for these algorithms. K-shape also shows 61.5% improvements, although with a higher proportion of declines (30.8%), consistent with the mixture of circles and crosses seen in its panel of Figure 2.

By contrast, Kernel K-means performs less consistently: only 34.6% of datasets improve with CV, while declines (30.8%) and near-equal cases (34.6%) are more common, as seen in its high density of diamond markers close to the diagonal. K-means, Clara, Clarans, and Agglomerative clustering all fall into a middle ground, with improvements in approximately 46–50% of cases and declines ranging from 19–27%. These mixed outcomes are also apparent in Figure 2, where many datasets cluster near the diagonal with a balance of circles, diamonds, and crosses.

From a domain perspective, IMAGE datasets, which constitute the largest group, appear frequently above the diagonal in multiple algorithms, suggesting consistent—though sometimes modest—gains from volatility modeling.

SENSOR, MOTION, and HAR datasets are more often represented by circles above the diagonal, indicating substantial benefits in domains characterized by short-term fluctuations. ECG datasets mostly lie near the diagonal, indicating limited improvements, while TRAFFIC datasets present a balanced mix, underscoring that CV does not universally dominate across all domains.

Taken together, the combination of Figure 2 and Table 3 provides both a visual and quantitative confirmation that incorporating conditional volatility as a clustering feature yields clear advantages for certain algorithms (notably SC, N2D, and KDB) and data domains, while its benefits remain more modest or inconsistent for centroid- and shape-based methods.

3.4 Statistical validation

To verify whether the observed performance differences between CV and raw data are systematic rather than coincidental, a formal statistical test was conducted. Specifically, the Wilcoxon signed-rank test (Wilcoxon, 1945), a widely used non-parametric test for paired data that does not assume normality (Demšar, 2006; García et al., 2010), was employed. In this case, each dataset contributes a paired observation consisting of RI from CV and RI from raw data. The null hypothesis (H_0) states that the median difference between CV and raw RI values is zero (no systematic advantage), while the alternative hypothesis (H_1) posits that CV differs consistently from raw. A significance level of 0.05 was adopted for all tests.

To capture performance variations across different granularities, the Wilcoxon test was conducted at two levels: First, algorithm level (Table 4): For each clustering algorithm, CV was compared to raw across all datasets. Second, type level (Table 5): Datasets were grouped by domain (e.g., IMAGE, SENSOR), and the test was applied within each group.

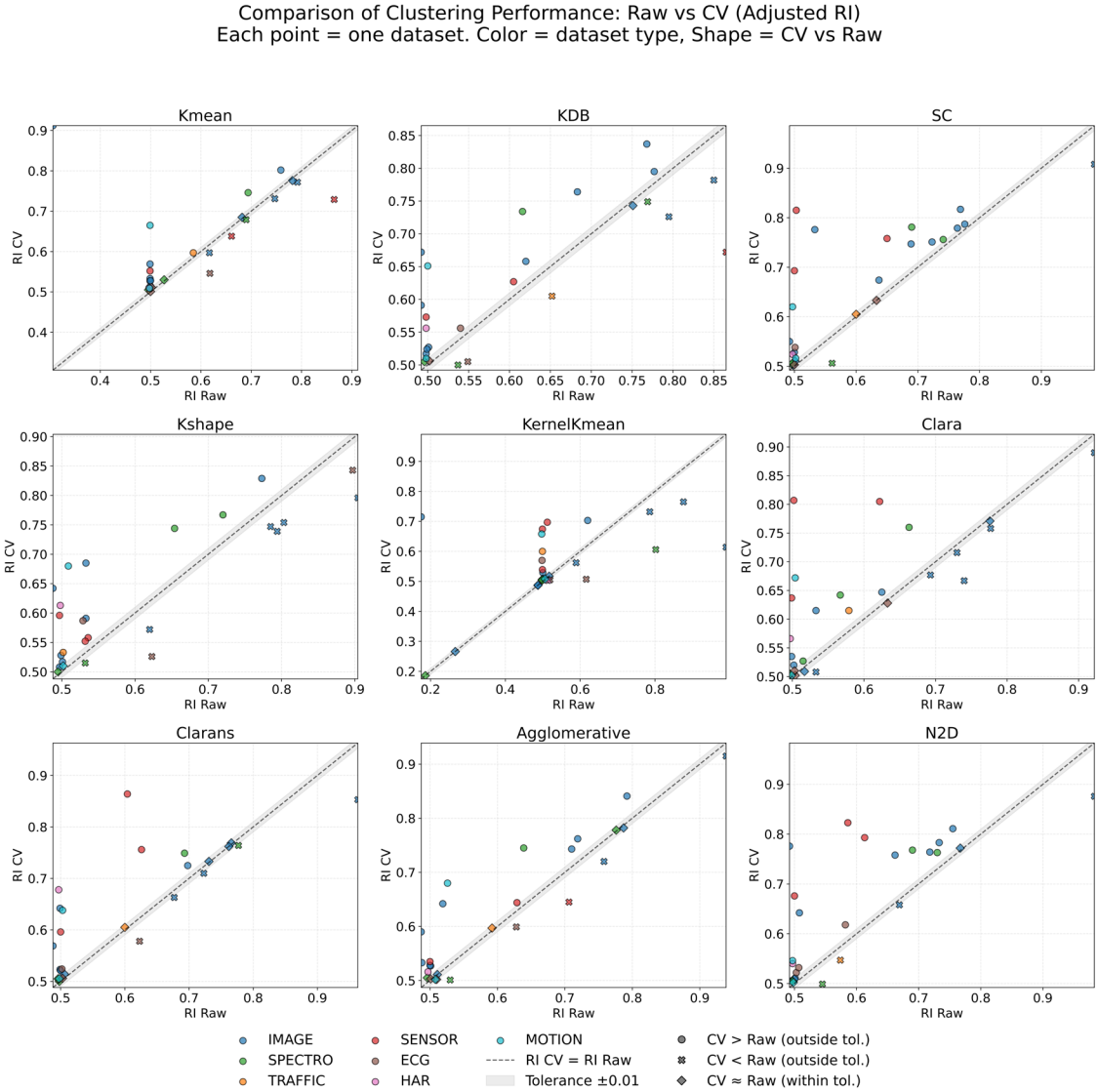


Figure 2. Comparison of clustering performance

Table 2. Performance comparison among clustering methods

DATASET(UCR)	TYPE	Kmean(EU)		KDBA		SC		Kshape		KernelKmean		Clara		Clarans		Agglomerative		N2D	
		raw	cv	raw	cv	raw	cv	raw	cv	raw	cv	raw	cv	raw	cv	raw	cv	raw	cv
ArrowHead	IMAGE	0.617	0.597	0.620	**0.658**	0.637	**0.674**	0.620	0.572	0.589	0.562	0.625	**0.647**	0.677	0.663	0.487	**0.590**	0.669	0.658
Beef	SPECTRO	0.694	**0.746**	0.616	**0.734**	0.741	**0.756**	0.654	**0.744**	0.186	**0.186**	0.663	**0.760**	0.693	**0.749**	0.639	**0.745**	0.730	**0.763**
BeetleFly	IMAGE	0.499	**0.533**	0.492	**0.591**	0.492	**0.550**	0.533	**0.591**	0.487	**0.487**	0.533	0.508	0.488	**0.569**	0.488	**0.533**	0.508	**0.642**
BirdChicken	IMAGE	0.499	**0.569**	0.492	**0.672**	0.533	**0.776**	0.488	**0.642**	0.487	**0.487**	0.533	**0.615**	0.499	**0.642**	0.519	**0.642**	0.492	**0.776**
Chinatown	TRAFFIC	0.585	**0.597**	0.652	0.605	0.600	**0.605**	0.502	**0.533**	0.499	**0.600**	0.579	**0.615**	0.600	**0.605**	0.592	**0.597**	0.574	0.547
DiatomSizeReduction	IMAGE	0.306	**0.912**	0.768	**0.837**	0.986	0.908	0.904	0.796	0.990	0.614	0.922	0.890	0.963	0.853	0.939	0.915	0.983	0.876
DistalPhalanxOutlineCorrect	IMAGE	0.499	**0.527**	0.501	**0.527**	0.501	**0.510**	0.501	**0.517**	0.500	**0.508**	0.499	**0.535**	0.499	**0.523**	0.501	**0.527**	0.500	**0.509**
DodgerLoopGame	SENSOR	0.499	**0.552**	0.498	**0.573**	0.500	**0.693**	0.497	**0.596**	0.499	**0.539**	0.499	**0.637**	0.500	**0.596**	0.500	**0.535**	0.500	**0.676**
ECG200	ECG	0.618	0.546	0.540	**0.556**	0.633	**0.633**	0.623	0.526	0.498	**0.570**	0.633	0.628	0.623	0.578	0.628	0.599	0.582	**0.618**
ECGFiveDays	ECG	0.500	**0.501**	0.502	**0.506**	0.501	**0.538**	0.897	0.843	0.519	0.505	0.504	0.502	0.502	**0.524**	0.510	0.502	0.503	**0.522**
FaceFour	IMAGE	0.759	**0.802**	0.777	**0.795**	0.776	**0.787**	0.785	0.747	0.620	**0.703**	0.740	0.667	0.723	0.710	0.719	**0.762**	0.755	**0.811**
GunPoint	HAR	0.497	**0.510**	0.498	**0.556**	0.497	**0.524**	0.498	**0.613**	0.514	0.503	0.497	**0.566**	0.497	**0.678**	0.497	**0.516**	0.497	**0.540**
Ham	SPECTRO	0.527	**0.530**	0.537	0.500	0.561	0.506	0.532	0.515	0.498	**0.505**	0.515	**0.527**	0.498	**0.500**	0.530	0.501	0.545	0.499
Herring	IMAGE	0.500	**0.514**	0.498	**0.517**	0.498	**0.502**	0.497	**0.508**	0.517	**0.517**	0.502	**0.520**	0.506	**0.514**	0.511	**0.511**	0.496	**0.506**
MiddlePhalanxOutlineCorrect	IMAGE	0.500	**0.528**	0.499	**0.524**	0.500	**0.528**	0.499	**0.528**	0.500	**0.529**	0.517	0.509	0.500	**0.521**	0.500	**0.528**	0.501	**0.511**
MiddlePhalanxTW	IMAGE	0.782	0.776	0.850	0.782	0.769	**0.817**	0.773	**0.829**	0.786	0.732	0.777	0.758	0.766	**0.769**	0.792	**0.841**	0.767	**0.772**
OSULeaf	IMAGE	0.747	0.731	0.751	0.743	0.723	**0.751**	0.794	0.739	0.175	**0.715**	0.730	0.716	0.731	**0.733**	0.758	0.720	0.733	**0.783**
ProximalPhalanxOutlineAgeGroup	SPECTRO	0.690	0.679	0.769	0.749	0.690	**0.781**	0.720	**0.767**	0.802	0.606	0.567	**0.642**	0.777	0.764	0.776	**0.778**	0.690	**0.768**
ProximalPhalanxTW	IMAGE	0.792	0.772	0.795	0.726	0.764	**0.779**	0.803	0.754	0.876	0.765	0.776	0.771	0.762	**0.762**	0.787	0.782	0.718	**0.764**
Rock	IMAGE	0.682	**0.685**	0.683	**0.764**	0.689	**0.747**	0.533	**0.685**	0.266	**0.266**	0.693	0.677	0.698	**0.725**	0.710	**0.743**	0.662	**0.758**
SonyAIBORobotSurface1	SENSOR	0.865	0.729	0.865	0.672	0.503	**0.815**	0.536	**0.558**	0.499	**0.674**	0.502	**0.807**	0.626	**0.756**	0.706	0.645	0.586	**0.823**
SonyAIBORobotSurface2	SENSOR	0.661	0.638	0.605	**0.627**	0.650	**0.758**	0.532	**0.552**	0.512	**0.697**	0.622	**0.805**	0.604	**0.864**	0.629	**0.644**	0.613	**0.793**
ToeSegmentation1	MOTION	0.498	**0.509**	0.498	**0.510**	0.502	**0.516**	0.502	**0.510**	0.506	**0.508**	0.499	**0.502**	0.498	**0.505**	0.508	0.502	0.498	**0.502**
ToeSegmentation2	MOTION	0.499	**0.665**	0.500	**0.651**	0.497	**0.620**	0.509	**0.680**	0.497	**0.658**	0.504	**0.672**	0.503	**0.638**	0.526	**0.680**	0.497	**0.546**
TwoLeadECG	ECG	0.502	**0.511**	0.549	0.505	0.500	**0.503**	0.529	**0.587**	0.616	0.507	0.503	**0.510**	0.503	**0.506**	0.501	**0.503**	0.507	**0.532**
Wine	SPECTRO	0.496	**0.505**	0.496	**0.505**	0.496	**0.505**	0.495	**0.500**	0.499	**0.505**	0.496	**0.505**	0.496	**0.505**	0.496	**0.505**	0.497	**0.505**
Average RI		0.589	**0.622**	0.610	**0.630**	0.605	**0.657**	0.606	**0.632**	0.536	**0.556**	0.594	**0.634**	0.605	**0.644**	0.606	**0.629**	0.600	**0.654**
Percentage change (%)		5.55%		3.35%		8.53%		4.28%		3.66%		6.87%		6.46%		3.78%		8.93%	

Table 3. Algorithm-wise performance: CV vs. raw data (% of datasets)

Algorithm	% CV > Raw	% CV < Raw	% ≈ (within tol)
Kmean	50.000	26.923	23.077
KDB	61.538	26.923	11.538
SC	69.231	7.692	23.077
Kshape	61.538	30.769	7.692
KernelKmean	34.615	30.769	34.615
Clara	50.000	23.077	26.923
Clarans	46.154	19.231	34.615
Agglomerative	50.000	19.231	30.769
N2D	61.538	15.385	23.077

At the algorithm level, the results show that SC ($p < 0.001$), N2D ($p = 0.001$), Clarans ($p = 0.005$), Clara ($p = 0.018$), and agglomerative ($p = 0.025$) all exhibit statistically significant improvements when CV is applied. These findings confirm the visual patterns in Figure 2, where these methods showed a higher proportion of points above the diagonal line (CV > raw). Conversely, K-means ($p = 0.055$), KDB ($p = 0.055$), K-shape ($p = 0.051$), and Kernel K-means ($p = 0.249$) do not reach significance, consistent with their more balanced or dispersed outcomes in Figure 2 and the percentages reported in Table 3.

At the domain level, the benefits of CV also vary across data types. IMAGE datasets ($p < 0.001$) show the strongest and most consistent gains, while SENSOR ($p < 0.001$), MOTION ($p < 0.001$), HAR ($p = 0.004$), and SPECTRO ($p = 0.023$) datasets also achieve significant improvements. In contrast, TRAFFIC ($p = 0.152$) and ECG ($p = 0.546$) do not show significant effects, indicating that CV does not provide systematic advantages for these types.

Taken together, the Wilcoxon results corroborate the visual analysis by confirming that CV yields statistically significant improvements in several clustering methods and dataset domains. At the same time, the results emphasize that the benefits of CV are not universal: they are strongest for methods such as Spectral Clustering and N2D, and for data domains such as IMAGE and SENSOR, while remaining more modest or inconsistent for centroid- and shape-based methods (e.g., K-means, K-shape) and certain domains (e.g., ECG, TRAFFIC).

Table 4. Wilcoxon test results by algorithm

Algorithm	Sample-size	W-statistic	*p*-value
Kmean	26	238.5	0.055
KDB	26	238.5	0.055
SC	26	**290.0**	**0.000**
Kshape	26	240.0	0.051
KernelKmean	26	135.0	0.249
Clara	26	**258.0**	**0.018**
Clarans	26	**259.0**	**0.005**
Agglomerative	26	**235.5**	**0.025**
N2D	26	**298.5**	**0.001**

Table 5. Wilcoxon test results by dataset type

Type	Sample-size	W-statistic	*p* -value
IMAGE	108	3665.5	0.000
SPECTRO	36	437.0	0.023
TRAFFIC	9	32.0	0.152
SENSOR	27	326.5	0.000
ECG	27	171.0	0.546
HAR	9	44.0	0.004
MOTION	18	167.0	0.000

3.5 Discussion

The combined evidence from all results highlights several important insights into the role of conditional volatility (CV) in time-series clustering.

At a broad level, Table 2 shows that CV leads to consistent average improvements across nearly all clustering algorithms, with percentage gains ranging from 3.35% (KDBA) to 8.93% (N2D). Notably, graph- and

representation-based methods, such as Spectral Clustering (8.53%) and N2D (8.93%), exhibit the largest relative improvements. Centroid- and shape-based methods, including K-means (5.55%), K-Shape (4.28%), and Kernel K-means (3.66%), benefit less consistently. These results suggest that algorithms leveraging pairwise similarities, graph structures, or latent embeddings are better positioned to exploit volatility-adjusted information.

Visual analysis (Figure 2 and Table 3) reinforces this conclusion. A majority of datasets cluster above the diagonal, confirming improvements from CV. However, the distribution of marker shapes (circles, diamonds, crosses) indicates that gains are concentrated in certain domains and algorithms. For instance, IMAGE datasets, which dominate the benchmark, generally exhibit modest improvements (~0.23%), while SENSOR, HAR, MOTION, and SPECTRO datasets show more pronounced upward shifts, reflecting the ability of CV to capture short-term volatility and structural variance in these domains. Conversely, ECG and TRAFFIC datasets display mixed or negligible improvements, suggesting that CV contributes less when temporal patterns are regular or class separability is already high.

Statistical validation (Tables 4 and 5) supports these observations. At the algorithm level, CV yields statistically significant improvements for Spectral Clustering, N2D, CLARA, CLARANS, and Agglomerative clustering, but not for K-means, K-Shape, or Kernel K-means. At the domain level, IMAGE, SENSOR, MOTION, HAR, and SPECTRO datasets demonstrate significant gains, whereas ECG and TRAFFIC do not. This aligns with row-level evidence: for example, within the IMAGE domain, DiatomSizeReduction shows substantial RI improvements under CV, whereas Herring and OSULeaf exhibit negligible changes, highlighting the role of dataset-level heterogeneity.

Taken together, these results indicate that the impact of CV is algorithm-dependent, domain-dependent, and dataset-specific:

Algorithm-dependence – Methods that utilize pairwise similarities, graph structures, or latent embeddings (e.g., SC and N2D) benefit most from CV. Centroid- and shape-based algorithms experience smaller or inconsistent gains, reflecting their limited ability to leverage structural variance.

Domain-dependence – Heterogeneous or noisy domains (SENSOR, MOTION, HAR, SPECTRO) display clearer benefits from CV, whereas highly regular domains (ECG, TRAFFIC, IMAGE) exhibit modest or inconsistent improvements.

Dataset-level heterogeneity – Even within a single domain, individual datasets respond differently depending on intrinsic characteristics such as volatility regimes, amplitude fluctuations, or temporal irregularities. For example, within IMAGE datasets, DiatomSizeReduction shows substantial improvements, while Herring and OSULeaf show minimal gains.

These insights suggest that CV should be considered a selective enhancement, rather than a universal solution. Practitioners should evaluate both the clustering algorithm and dataset characteristics before incorporating CV, while researchers may explore which dataset properties (e.g., volatility regimes, signal periodicity) drive improvements.

4. CONCLUSION

This article introduced ECCV, a time-series clustering framework that leverages ARIMA-GARCH to extract CV as an alternative representation of time series. By transforming raw sequences into volatility-based features—where each point reflects the model-estimated conditional variance—ECCV reduces noise, emphasizes structural changes, and preserves local temporal dynamics often underutilized by conventional clustering methods.

Experiments on 26 UCR benchmark datasets show that CV enhances clustering performance in several algorithms. The largest gains were observed for N2D and SC, which rely on latent representations and graph-based embeddings, while partitional methods (K-Means, CLARA, K-Shape) benefited moderately, and hierarchical/alignment-based methods (Agglomerative, KDBA) showed inconsistent improvements. These results indicate that the effectiveness of CV is selective, depending on both the clustering algorithm and dataset characteristics (e.g., volatility regimes, temporal patterns). Overall, ECCV provides a scalable, interpretable, and statistically grounded approach to improving time-series clustering, particularly in domains where volatility carries meaningful information (e.g., SENSOR, MOTION, SPECTRO).

Future work could explore several promising directions. One possible direction is integrating CV into advanced clustering algorithms to assess potential improvements in accuracy. Another is combining CV with raw data within multivariate clustering frameworks to harness complementary information. Additionally, alternative volatility models beyond GARCH, such as stochastic volatility, could be applied to capture more complex dynamics. Finally, future studies may focus on characterizing dataset properties—such as signal amplitude variability and periodicity—that maximize the benefits of CV. Ultimately, CV should be considered a selective enhancement, applied based on algorithm choice and dataset characteristics, offering a flexible tool for practitioners and a foundation for further research into volatility-informed time-series analysis.

ACKNOWLEDGMENT

This research was supported by the Scholarship for Talented Students, Graduate Program, Faculty of Science and Technology, Thammasat University, under Contract No. TB 10/2564.

AUTHOR CONTRIBUTION

P.T. and T.B. conceived and planned the experiments. P.T. carried out experiments and data collection. P.T. and T.B. contributed to the interpretation of the results. P.T. took the lead in writing the manuscript, while T.B. supervised the project. Both authors provided critical feedback and helped shape the research, analysis, and manuscript.

REFERENCES

Aghabozorgi, S., Shirkhorshidi, A. S., & Wah, T. Y. (2015). Time-series clustering – A decade review. *Information Systems*, *53*, 16–38. https://doi.org/10.1016/j.is.2015.04.007

Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. In E. Parzen, K. Tanabe, & G. Kitagawa (Eds.), *Selected papers of Hirotugu Akaike* (pp. 199–213). Springer. https://doi.org/10.1007/978-1-4612-1694-0_15

Ali, G. (2013). EGARCH, GJR-GARCH, TGARCH, AVGARCH, NGARCH, IGARCH and APARCH models for pathogens at marine recreational sites. *Journal of Statistical and Econometric Methods*, *2*(3), 57–73.

Arthur, D., & Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms* (pp. 1027–1035). Society for Industrial and Applied Mathematics. https://dl.acm.org/doi/10.5555/1283383.1283494

Ausloos, M., Zhang, Y., & Dhesi, G. (2020). Stock index futures trading impact on spot price volatility. The CSI 300 studied with a TGARCH model. *Expert Systems with Applications*, *160*, Article 113688. https://doi.org/10.1016/j.eswa.2020.113688

Baillie, R. T., Bollerslev, T., & Mikkelsen, H. O. (1996). Fractionally integrated generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *74*(1), 3–30. https://doi.org/10.1016/S0304-4076(95)01749-6

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327. https://doi.org/10.1016/0304-4076(86)90063-1

Bonanno, G., Lillo, F., & Mantegna, R. N. (2001). High-frequency cross-correlation in a set of stocks. *Quantitative Finance*, *1*(1), 96–104. https://doi.org/10.1080/713665554

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2016). *Time series analysis: Forecasting and control* (5th ed.). John Wiley & Sons.

Caiado, J., & Crato, N. (2007). A GARCH-based method for clustering of financial time series: International stock markets evidence. In C. H. Skiadas (Ed.), *Recent advances in stochastic modeling and data analysis* (pp. 542–551). World Scientific. https://doi.org/10.1142/9789812709691_0064

Caiado, J., & Crato, N. (2010). Identifying common dynamic features in stock returns. *Quantitative Finance*, *10*(7), 797–807. https://doi.org/10.1080/14697680903567152

Cont, R. (2007). Volatility clustering in financial markets: Empirical facts and agent-based models. In G. Teyssière & A. P. Kirman (Eds.), *Long memory in economics* (pp. 289–309). Springer. https://doi.org/10.1007/978-3-540-34625-8_10

Cuturi, M. (2011). Fast global alignment kernels. In L. Getoor & T. Scheffer (Eds.), *Proceedings of the 28th international conference on machine learning* (pp. 929–936). Omnipress. https://dl.acm.org/doi/10.5555/3104482.3104599

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., & Keogh, E. (2019). The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, *6*(6), 1293–1305. https://doi.org/10.1109/JAS.2019.1911747

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, *7*, 1–30. https://dl.acm.org/doi/10.5555/1248547.1248548

Dhillon, I. S., Guan, Y., & Kulis, B. (2004). Kernel k-means, spectral clustering and normalized cuts. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 551–556). Association for Computing Machinery. https://doi.org/10.1145/1014052.1014118

Dickey, D. A., & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American Statistical Association*, *74*(366), 427–431. https://doi.org/10.2307/2286348

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., & Keogh, E. (2008). Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, *1*(2), 1542–1552. https://doi.org/10.14778/1454159.1454226

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica*, *50*(4), 987–1007. https://doi.org/10.2307/1912773

Engle, R. F., & Bollerslev, T. (1986). Modelling the persistence of conditional variances. *Econometric Reviews*, *5*(1), 1–50. https://doi.org/10.1080/07474938608800095

Fu, T.-C. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, *24*(1), 164–181. https://doi.org/10.1016/j.engappai.2010.09.007

Fujita, A., Severino, P., Kojima, K., Sato, J. R., Patriota, A. G., & Miyano, S. (2012). Functional clustering of time series gene expression data by Granger causality. *BMC Systems Biology*, *6*(1), Article 137. https://doi.org/10.1186/1752-0509-6-137

García, S., Fernández, A., Luengo, J., & Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, *180*(10), 2044–2064. https://doi.org/10.1016/j.ins.2009.12.010

Glosten, L. R., Jagannathan, R., & Runkle, D. E. (1993). On the relation between the expected value and the volatility of the nominal excess return on stocks. *The Journal of Finance*, *48*(5), 1779–1801. https://doi.org/10.1111/j.1540-6261.1993.tb05128.x

Hamilton, J. D. (1994). *Time series analysis*. Princeton University Press.

Han, J., Kamber, M., & Pei, J. (2012). *Data mining: Concepts and techniques* (3rd ed.). Morgan Kaufmann.

Kalpakis, K., Gada, D., & Puttagunta, V. (2001). Distance measures for effective clustering of ARIMA time-series. In N. Cercone, T. Y. Lin, & X. Wu (Eds.), *Proceedings of the 2001 IEEE international conference on data mining* (pp. 273–280). IEEE. https://doi.org/10.1109/ICDM.2001.989529

Kaufman, L., & Rousseeuw, P. J. (1990). *Finding groups in data: An introduction to cluster analysis*. John Wiley & Sons.

Keogh, E., & Ratanamahatana, C. A. (2005). Exact indexing of dynamic time warping. *Knowledge and Information*

Systems, *7*(3), 358–386. https://doi.org/10.1007/s10115-004-0154-9

Kumar, M., Patel, N. R., & Woo, J. (2002). Clustering seasonality patterns in the presence of errors. In *Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 557–563). Association for Computing Machinery. https://doi.org/10.1145/775047.775129

Kumar, N., Lolla, V. N., Keogh, E., Lonardi, S., Ratanamahatana, C. A., & Wei, L. (2005). Time-series bitmaps: A practical visualization tool for working with large time series databases. In H. Kargupta, J. Srivastava, C. Kamath, & A. Goodman (Eds.), *Proceedings of the 2005 SIAM international conference on data mining (SDM)* (pp. 531–535). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9781611972757.55

Li, G., Bräysy, O., Jiang, L., Wu, Z., & Wang, Y. (2013). Finding time series discord based on bit representation clustering. *Knowledge-Based Systems*, *54*, 243–254. https://doi.org/10.1016/j.knosys.2013.09.015

Lim, C. M., & Sek, S. K. (2013). Comparing the performances of GARCH-type models in capturing the stock market volatility in Malaysia. *Procedia Economics and Finance*, *5*, 478–487. https://doi.org/10.1016/S2212-5671(13)00056-7

Ljung, G. M., & Box, G. E. P. (1978). On a measure of lack of fit in time series models. *Biometrika*, *65*(2), 297–303. https://doi.org/10.1093/biomet/65.2.297

López-Oriona, Á., Montero-Manso, P., & Vilar, J. A. (2025). Time series clustering based on prediction accuracy of global forecasting models. *Knowledge-Based Systems*, *323*, Article 113649. https://doi.org/10.1016/j.knosys.2025.113649

Lúcio, F., & Caiado, J. (2022). COVID-19 and stock market volatility: A clustering approach for S&P 500 industry indices. *Finance Research Letters*, *49*, Article 103141. https://doi.org/10.1016/j.frl.2022.103141

Ma, M. (2025). Time series analysis of regional GDP and stock market volatility using ARIMA and ARMA-GARCH models. *Highlights in Business, Economics and Management*, *61*, 151–162. https://doi.org/10.54097/4t750n32

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In L. M. Le Cam & J. Neyman (Eds.), *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (pp. 281–297). University of California Press.

Mantegna, R. N. (1999). Hierarchical structure in financial markets. *The European Physical Journal B*, *11*(1), 193–197. https://doi.org/10.1007/s100510050929

Marisetty, N. (2024). Evaluating the efficacy of GARCH models in forecasting volatility dynamics across major global financial indices: A decade-long analysis. *Journal of Economics, Management and Trade*, *30*(9), 16–33. https://doi.org/10.9734/jemt/2024/v30i91238

McConville, R., Santos-Rodriguez, R., Piechocki, R. J., & Craddock, I. (2019). *N2D: (Not Too) deep clustering via clustering the local manifold of an autoencoded embedding*. ArXiv. https://doi.org/10.48550/arXiv.1908.05968

Mitsa, T. (2010). *Temporal data mining*. Chapman and Hall/CRC.

Nanopoulos, A., Alcock, R., & Manolopoulos, Y. (2001). Feature-based classification of time-series data. In N. Mastorakis & S. D. Nikolopoulos (Eds.), *Information processing and technology* (pp. 49–61). Nova Science.

Nelson, D. B. (1991). Conditional heteroskedasticity in asset returns: A new approach. *Econometrica*, *59*(2), 347–370. https://doi.org/10.2307/2938260

Ng, A. Y., Jordan, M. I., & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Proceedings of the 15th international conference on neural information processing systems: Natural and synthetic* (pp. 849–856). MIT Press. https://dl.acm.org/doi/10.5555/2980539.2980649

Ng, R. T., & Han, J. (2002). CLARANS: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, *14*(5), 1003–1016. https://doi.org/10.1109/TKDE.2002.1033770

Otranto, E. (2008). Clustering heteroskedastic time series by model-based procedures. *Computational Statistics & Data Analysis*, *52*(10), 4685–4698. https://doi.org/10.1016/j.csda.2008.03.020

Panuccio, A., Bicego, M., & Murino, V. (2002). A hidden Markov model-based approach to sequential data clustering. In T. Caelli, A. Amin, R. P. W. Duin, D. de Ridder, & M. Kamel (Eds.), *Structural, syntactic, and statistical pattern recognition* (pp. 734–743). Springer. https://doi.org/10.1007/3-540-70659-3_77

Paparrizos, J., & Gravano, L. (2016). k-Shape: Efficient and accurate clustering of time series. *ACM SIGMOD Record*, *45*(1), 69–76. https://doi.org/10.1145/2949741.2949758

Petitjean, F., Ketterlin, A., & Gançarski, P. (2011). A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, *44*(3), 678–693. https://doi.org/10.1016/j.patcog.2010.09.013

Räsänen, T., & Kolehmainen, M. (2009). Feature-based clustering for electricity use time series data. In M. Kolehmainen, P. Toivanen, & B. Beliczynski (Eds.), *Adaptive and natural computing algorithms* (pp. 401–412). Springer. https://doi.org/10.1007/978-3-642-04921-7_41

Sahiner, M. (2022). Forecasting volatility in Asian financial markets: Evidence from recursive and rolling window methods. *SN Business & Economics*, *2*(10), Article 157. https://doi.org/10.1007/s43546-022-00329-9

Sebastiani, P., Ramoni, M., Cohen, P. R., Warwick, J., & Davis, J. (1999). Discovering dynamics using Bayesian clustering. In D. J. Hand, J. N. Kok, & M. R. Berthold (Eds.), *Advances in intelligent data analysis* (pp. 199–209). Springer. https://doi.org/10.1007/3-540-48412-4_17

Sherly, A., Christo, M. S., & Elizabeth, J. V. (2025). A hybrid approach to time series forecasting: Integrating ARIMA and Prophet for improved accuracy. *Results in Engineering*, *27*, Article 105703. https://doi.org/10.1016/j.rineng.2025.105703

Srihari, G., Kusuma, T., Chetanraj, D. B., Kumar, J. P. S., & Aluvala, R. (2024). Predictive modeling of return volatility in sustainable investments: An in-depth analysis of ARIMA, GARCH, and ARCH techniques. *Investment Management and Financial Innovations*, *21*(1), 213–228. https://doi.org/10.21511/imfi.21(1).2024.17

Steinbach, M., Tan, P.-N., Kumar, V., Klooster, S., & Potter, C. (2003). Discovery of climate indices using clustering. In *Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 446–455). Association for Computing Machinery. https://doi.org/10.1145/956750.956801

Subhani, N., Rueda, L., Ngom, A., & Burden, C. J. (2010). Multiple gene expression profile alignment for microarray time-series data clustering. *Bioinformatics*, *26*(18), 2281–2288. https://doi.org/10.1093/bioinformatics/btq422

Tsay, R. S. (2010). *Analysis of financial time series* (3rd ed.). John Wiley & Sons.

Wang, X., Mueen, A. Ding, H., Trajcevski, G., Scheuermann, P., & Keogh, E. (2013). Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, *26*(2), 275–309. https://doi.org/10.1007/s10618-012-0250-5

Wang, Y., Xiang, Y., Lei, X., & Zhou, Y. (2022). Volatility analysis based on GARCH-type models: Evidence from the Chinese stock market. *Economic Research–Ekonomska Istraživanja*, *35*(1), 2530–2554. https://doi.org/10.1080/1331677X.2021.1967771

Ward, J. H., Jr. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, *58*(301), 236–244. https://doi.org/10.1080/01621459.1963.10500845

Warren Liao, T. (2005). Clustering of time series data—A survey. *Pattern Recognition*, *38*(11), 1857–1874. https://doi.org/10.1016/j.patcog.2005.01.025

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, *1*(6), 80–83. https://doi.org/10.2307/3001968

Wismüller, A., Lange, O., Dersch, D. R., Leinsinger, G. L., Hahn, K., Pütz, B., & Auer, D. (2002). Cluster analysis of biomedical image time-series. *International Journal of Computer Vision*, *46*(2), 103–128. https://doi.org/10.1023/A:1013550313321

Zakoian, J.-M. (1994). Threshold heteroskedastic models. *Journal of Economic Dynamics and Control*, *18*(5), 931–955. https://doi.org/10.1016/0165-1889(94)900 39-6