

Fractional-order Difference Decoupling and Pattern Extraction for Multivariate Time Series Forecasting

Chengli Zhou¹, Wuqi Chen¹, Yaqun Huang¹, Chunna Zhao^{1, #}

School of Information Science and Engineering, Yunnan University, South Waihuan Road, University City East, Kunming, 650500, Yunnan, P. R. China

Corresponding author: Chunna Zhao (Email: zhaochunna@ynu.edu.cn; ORCID: 0000-0002-0019-1041)

Abstract

Multivariate time series forecasting is essential in many real-world applications. However, inherent challenges such as non-stationarity, multi-scale temporal patterns, weak inter-variable correlations, and long-range dependencies limit the effectiveness of existing models. The modeling of heterogeneous temporal dynamics in a unified space, which often leads to structural interference and degraded learning efficiency. This study proposes a novel forecasting framework, termed the Fractional Differencing Decoupling and Pattern Extraction Model (FDDPEM). Built upon a structural decoupling–reconstruction paradigm, FDDPEM first separates trend and seasonal components using fractional differencing, which stabilizes the series while preserving long-memory characteristics. Each component is then modeled with specialized mechanisms through multi-scale segmentation, clustering, and two-dimensional convolution. Extensive experiments on multiple benchmark datasets demonstrate that FDDPEM consistently outperforms state-of-the-art methods, particularly in handling weakly seasonal data and complex inter-variable relationships. Ablation studies further validate the rationality and effectiveness of the proposed structural design.

Keywords: Fractional-order difference; Seasonal patch clustering; Trend patch clustering; Time series

1. Introduction

Multivariate time series data are widely used in applications such as in economy, manufacturing and transportation, and healthcare. Accurate forecasting is critical for decision-making in complex systems. However, intrinsic characteristics, including non-stationarity, multi-scale patterns, weak inter-variable correlations, and long-range dependencies, make reliable modeling difficult. More fundamentally, these challenges stem from the coexistence of global dependencies and local pattern heterogeneity, which remains insufficiently addressed in existing frameworks.

Early studies relied on statistical models assuming of linearity and stationarity, such as ARIMA [1] and its multivariate extensions [2]. While effective under limited conditions, they fail to model complex nonlinear dynamics and cross-variable interactions. Deep learning models, including recurrent neural networks [3, 4] and convolutional architectures [5, 6], improved nonlinear dependency modeling. Attention-based models [7, 8] and frequency-domain approaches [9] further enhanced long-range and seasonal modeling. Nevertheless, most approaches enhance

representation within a unified space, rather than reorganizing temporal structures through explicit decomposition.

Fractional Fourier transform–based models characterize signals in intermediate time–frequency domains [10, 11]. In particular, the multi-scale fractional Fourier transform convolutional neural network (MFFCNN) effectively captures seasonal patterns [12]. Despite these advances, existing methods still struggle with (i) weak or non-seasonal components, (ii) noise amplification under low inter-variable correlations, and (iii) insufficient cross-segment dependency modeling. These limitations indicate the absence of an explicit structural decoupling mechanism prior to modeling.

To address this issue, we propose the Fractional Differencing Decoupling and Temporal Pattern Extraction Model (FDDPEM). FDDPEM follows a structural decoupling–reconstruction paradigm rather than a conventional end-to-end enhancement strategy. It innovatively decouples trend and seasonal components via multi-order fractional differencing, which generalizes traditional integer-order differencing by introducing a continuum of fractional orders to control both sequence stabilization and then models them using component-specific mechanisms. This decoupling serves as structural reorganization rather than simple preprocessing, reducing inter-component interference and improving interpretability. By integrating decomposition with multi-scale pattern extraction and adaptive inter-variable modeling, FDDPEM enhances both accuracy and robustness.

Specifically, unlike integer-order differencing, which removes trend information at a fixed first-order scale, multi-order fractional differencing allows flexible adjustment of trend and seasonal contributions across multiple orders, providing a richer spectrum of temporal transformations. Fractional differencing separates the series into trend and seasonal components, providing a clearer modeling basis. It preserves long-memory characteristics while stabilizing the sequence, offering a principled alternative to integer-order differencing. For seasonal component, multi-scale segmentation, clustering, and two-dimensional convolution extract local and global features. The clustering introduces a grouping prior that improves interpretability of seasonal similarity. For trend component, sliding-window segmentation and clustering distinguish monotonic and oscillatory behaviors. This explicit categorization distinguishes structural growth patterns from short-term fluctuations. Furthermore, two-dimensional convolutional further connects local, global, and inter-variable trend features. Such interactions form a hierarchical dependency modeling scheme instead of flat feature aggregation. Together, these mechanisms strengthen long-term dynamic representation.

Experimental results demonstrate that FDDPEM significantly outperforms existing methods across multiple benchmarks. It performs particularly well on weakly seasonal data and complex inter-variable scenarios. Ablation studies confirm that each structural component contributes consistently, validating the decoupling–reconstruction design. This study details the FDDPEM architecture and demonstrates its effectiveness in multivariate forecasting.

This work extends MFFCNN [12]. While retaining multi-scale fractional Fourier analysis, it expands the framework as follows: (1) introducing structurally motivated temporal decoupling via fractional differencing; (2) stabilizing sequences while preserving long-term dependencies; (3) establishing hierarchical dependency reconstruction through component-wise clustering and two-dimensional convolution.

2. Related Work

2.1. *Difference in time series prediction*

The difference method is one of the most fundamental and effective tools in statistics and machine learning for dealing with non-stationary time series. The core idea is to eliminate the long-term trend or seasonal components in the data by performing subtraction operations on adjacent observations, thereby making the sequence meet the stationarity requirements. In classic statistical models such as ARIMA [1], difference is regarded as the standard preprocessing step for achieving data stabilization. Through first-order difference, the model can capture the instantaneous rate of change of the data. By means of seasonal difference, the cyclic interference of a fixed period can be eliminated. With the popularization of deep learning in the field of time series prediction, the application of the difference method has evolved from a simple preprocessing approach to an architectural design that enhances the feature extraction capability of the model. Firstly, the difference method is often used to assist models in capturing local dynamic features and alleviating prediction lag. For instance, when DeepAR [19] deals with strongly non-stationary data, it often combines differential operations to reduce the interference of global trends on probability prediction; Neural Basis Expansion Analysis for Interpretable Time Series Forecasting (N-BEATS) [20] adopts a deep decomposition architecture. Through the residual difference between forward and backward predictions, it filters the modeled components layer by layer to achieve refined modeling. Secondly, the difference method is integrated into the internal operators of the neural network to enhance the prediction sensitivity. Sample Convolution and Interaction Network (SCINet) [5] builds a recursive interactive learning structure through downsampling and differential operations, enhancing the model's perception ability of minor fluctuations in data. Informer [7] explored the possibility of reducing noise interference in long sequence modeling through differential operations in the preprocessing stage. In addition, sequence decoupling based on the idea of difference has become the mainstream trend. DLinear [21] uses a difference-like operation to separate the trend term from the residual term, demonstrating the high efficiency of linear models in processing component data. Autoformer [22] further proposed a deep decomposition architecture, which uses temporal decomposition smoothing processing to decouple the sequence into trend terms and seasonal terms. PatchTST [8] enhances the model's stability under complex fluctuations through independent channel and block operations, indirectly utilizing the idea of local stabilization.

2.2. *Decomposition of time series data*

Time series data, as a collection of data recorded in chronological order, usually contains rich time series patterns, including season, trend, and sudden change patterns, etc. These patterns provide valuable clues about future trends and also make time series data highly predictive. However, these complex patterns are usually superimposed and have dependencies at different time scales, which poses challenges to the precise analysis and prediction of data. Therefore, effectively decomposing time series data and extracting the features of each component have become important means to improve the accuracy of time series prediction. Firstly, in terms of time series decomposition, Autoformer [22] proposed a time series decomposition module based on moving averages, which decomposes complex time variations into season and trend components. Subsequently, Frequency Enhanced Decomposed Transformer (FEDformer) [23] enhanced the decomposition module through multi-core moving averages. Dlinear [21] decomposes the time series as a preprocessing step before linear regression, while Multi-scale Local and Global Context Modeling (MICN) [24] decomposes the input time series into season and trend terms and

integrates global and local context information for prediction. In terms of multi-season analysis, N-BEATS [20] uses multiple trigonometric function bases to fit time series; Frequency Improved Legendre Memory Model (FiLM) [25] projects the time series onto the Legendre polynomial space, where different basis functions correspond to different periodic components in the original sequence. Recently, Temporal 2D Variation Modeling Network (TimesNet) [26] utilized the Fourier transform to map time series into multiple components with different period lengths and proposed a modular architecture to handle these decomposed components.

3. Problem Formulation

3.1. Fractional-order Difference

Multivariate time series in the real world generally exhibit significant characteristics of co-existence of non-stationarity and long-term dependence. Although traditional integer-order differences can effectively weaken trend components and enhance sequence stationarity, they often overly suppress low-frequency information, thereby disrupting the long-term dependent structures with predictive value in time series. To strike a balance between stabilization processing and information retention, this study introduces fractional-order difference as the theoretical basis for sequence preprocessing and structural decoupling.

Let the time series be $X_{1D} \in \mathbb{R}^{C \times L}$, and trend component, namely fractional-order difference is defined as:

$$\begin{aligned} Tr(x_t) &= \Delta^d x_t = (1 - B)^d x_t \\ &= \sum_{k=0}^{\infty} g_k^{(d)} x_{t-k} \end{aligned} \quad (1)$$

where B represents the lag operator, and $g_k^{(d)}$ is the fractional-order difference coefficient defined by the Gamma function. This form extends the traditional discretized-order difference to the continuous-order space, enabling the difference intensity to be finely adjusted.

The core advantage of fractional-order difference lies in the power-law attenuation of its difference coefficients, which means that long-term historical information is not quickly erased but continuously participates in the construction of the current state with decreasing weights. Therefore, this operation can effectively preserve the long-term memory and low-frequency structure in the time series while weakening the non-stationary trend. Compared with the hard truncation feature of integer-order differences, fractional-order differences offer a more flexible trend suppression mechanism.

Based on the above characteristics, the model decouples the original sequence into trend-dominant components and seasonal dominant components, providing a clear structure and sufficient information input representation for the subsequent domain modeling. This continuous-order decoupling process not only reduces the interference of non-stationarity on the prediction model, but also lays a theoretical foundation for the independent modeling of season patterns and long-term evolution structures.

3.2. Seasonal Patch Clustering

Season components data usually contain obvious seasonal fluctuations and some irregular changes, as shown in Figure 1. Season components refer to stable and repeatable regular patterns in a time series, such as seasonal fluctuations, daily periodic changes, or equipment operation

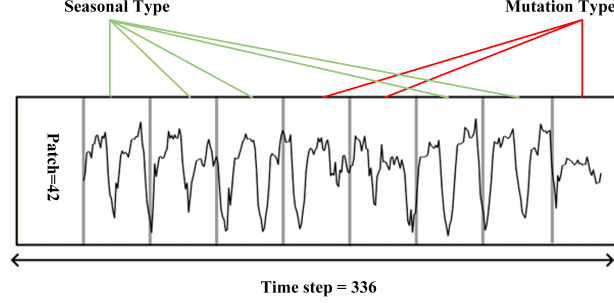


Figure 1: This example demonstrates the seasonal classes and mutant classes presented in the ETTh1 dataset when the seasonal patches are 42.

cycles, which have significant reference value for model predictions. Mutations, on the other hand, refer to the parts of a sequence that lack obvious periodicity, abnormal fluctuations or are affected by noise. They usually manifest as transient mutations or abnormal fluctuations, making limited contributions to model predictions and may even introduce interference. To model season patterns more effectively, we divide season components into multiple chunks (i.e., season feature fragments) and cluster them.

Although fractional-order difference can effectively highlight the fluctuation information in time series, the periodic term still simultaneously contains stable periodic patterns and irregular abrupt changes or noise fluctuations. Following the patching strategy of MFFCNN, two-dimensional patching data $X_s \in \mathbb{R}^{C \times S \times P}$ was obtained, S is the patch number. Applying the same treatment to all the fluctuation patches can easily amplify anomalies and interfere with periodic pattern modeling. Therefore, it is necessary to further distinguish the structural stability within the period, so as to achieve more targeted feature extraction.

For any two patches, the cosine similarity between them and its calculation method are:

$$sim_{sq} = \hat{p}_s^T \hat{p}_q, s \neq q \quad (2)$$

where p_s is the normalized patch vector.

And let $sim_{ss} = 0$, thereby obtaining the block similarity matrix $sim \in \mathbb{R}^{S \times S}$. To measure the consistency of each patch with the overall season structure, the average relative similarity of the s th patch is defined as:

$$mean_sim_s = \frac{1}{S-1} \sum_{s \neq q} sim_{sq} \quad (3)$$

where $mean_sim_s$ represents the average similarity of the s -th patch to all other patches. This quantity characterizes the season consistency strength of the block p_s in all patches.

3.3. Trend Patch Clustering

For one-dimensional trend term $Tr_{1D} \in \mathbb{R}^{C \times L}$ obtained for the fractional difference, we first use the window length step to slide the window and convert it into two-dimensional trend data. The implementation process is as follows:

$$Tr_{2D} = \text{Unfold}(\text{ReplicationPad}(Tr_{1D})) \quad (4)$$

where $\text{ReplicationPad}(\cdot)$ indicates filling in new areas by repeating boundary values. $Tr_{2D} \in \mathbb{R}^{C \times S \times P}$ is the two-dimensional trend components. The s th trend block is:

$$x_s \in \mathbb{R}^{C \times P}, s = 1, 2, \dots, S \quad (5)$$

To depict the overall change trend of each trend block in the time dimension, a linear fit is performed on the s th variable in the c th block, and its slope is defined as:

$$\begin{aligned} a_{c,s} &= \frac{\sum_{p=1}^P (x_{c,s,p} - \bar{x}_{c,s})(t_p - \bar{t})}{\sum_{p=1}^P (t_p - \bar{t})^2}, \\ \bar{x}_{c,s} &= \frac{1}{P} \sum_{p=1}^P x_{c,s,p}, \\ \bar{t} &= \frac{1}{P} \sum_{p=1}^P t_p, t_p = 1, 2, \dots, P. \end{aligned} \quad (6)$$

Then, the slopes of all variables are averaged to obtain the overall slope of the s th trend block $\text{slope}_s = \frac{1}{C} \sum_{c=1}^C a_{c,s}$. This quantity is used to quantify the overall upward or downward trend of this block. Then each trend block is hard partitioned based on the threshold δ :

$$\text{trend}_s = \begin{cases} Up, & \text{slope}_s > \delta \\ Down, & \text{slope}_s < -\delta \\ Flat, & |\text{slope}_s| \leq \delta \end{cases} \quad (7)$$

where δ is a learnable parameter used to suppress the small fluctuations caused by noise and avoid trend misjudgment. To block different trends into the corresponding convolutional branches, a trend mask is constructed:

$$M_{c,s,p}^{up} = \begin{cases} 1, & \text{trend}_s = Up \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

$$M_{c,s,p}^{down} = \begin{cases} 1, & \text{trend}_s = Down \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

$$M_{c,s,p}^{flat} = \begin{cases} 1, & \text{trend}_s = Flat \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

On the basis of the input trend item separated into $Tr_{2D}^{up} = M^{up} \odot Tr_{2D}$, $Tr_{2D}^{down} = M^{down} \odot Tr_{2D}$, $Tr_{2D}^{flat} = M^{flat} \odot Tr_{2D}$. Here, $\odot$ represents element-by-element multiplication. Then, for different trend categories, three two-dimensional convolutional branches with independent parameters are introduced:

$$\begin{cases} Y^{up} &= \text{Conv}_{up}(Tr_{2D}^{up}), \\ Y^{down} &= \text{Conv}_{down}(Tr_{2D}^{down}), \\ Y^{flat} &= \text{Conv}_{flat}(Tr_{2D}^{flat}) \end{cases} \quad (11)$$

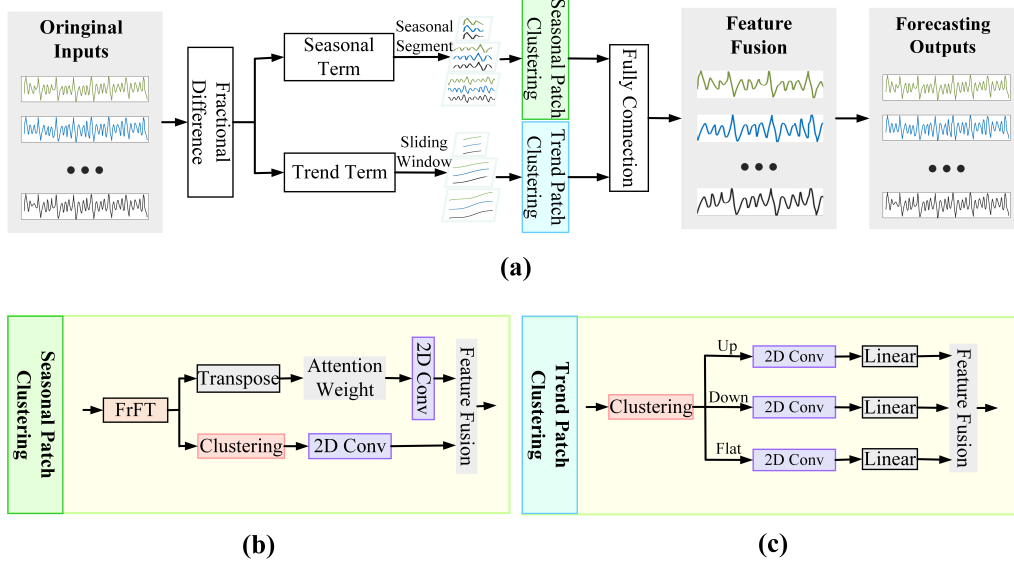


Figure 2: Overall structure of FDDPEM. (a) shows the overall framework, which first achieves season-trend decoupling through fractional-order difference, then performs feature extraction on the season components and trend components respectively, fuses the feature information, and finally completes the prediction. (b) is the seasonal patch clustering module; (c) presents the trend patch clustering module.

Each branch respectively models the local temporal structure characteristics under the upward, downward and oscillating trends.

The lightweight clustering design, when integrated with fractional-order decomposition, yields a pattern selection mechanism that is computationally efficient, interpretable, and applicable across diverse datasets, demonstrating the practical innovation of the proposed method.

4. Model Structure

To fully exploit the multi-scale structures and complex patterns inherent in multivariate time series, FDDPEM adopts a systematic decompose–model–fuse pipeline at the architectural level. The overall structure of FDDPEM is shown in Figure 2. FDDPEM consists of three key modules: the fractional-order difference module, the seasonal processing module, and the trend processing module.

4.1. Fractional Differencing Module

Time series decomposition is important to decompose the time series into different components (trend terms and seasonal terms), better revealing the inherent structure. The classical difference method directly calculates adjacent difference, accurately reflecting local trends but ignoring long-term low-frequency information [16, 13]. Fractional-order differencing extends the difference to non-integer orders, suppressing trends while preserving long-term dependencies and slowly varying low-frequency structures [27, 17], achieving a refined balance between trend removal and information retention.

Unlike the hard truncation trend elimination approach of integer-order difference, fractional orders form a continuous spectrum, enabling frequency response characteristics. The frequency response $H(\omega) = (1 - e^{-i\omega})^\alpha$ shows how low-frequency (trend) and high-frequency (seasonal) components are selectively retained, explaining why different α emphasize distinct temporal dependencies. Multi-order complementary decomposition fuses multiple α values to capture multi-scale temporal structures.

Applying multiple fractional orders and fusing results yields trend and seasonal components with enhanced expressive power, forming a hierarchical dependency reconstruction mechanism rather than simple feature stacking.

4.2. Seasonal Processing Module

For season component modeling, FDDPEM introduces the fractional Fourier transform to capture fine-grained, multi-scale season components, enabling adaptive extraction in the joint time–frequency domain. A multi-scale segmentation strategy constructs local temporal fragments, which are clustered based on season response strength via inter-variable correlation analysis, filtering noise from weakly correlated variables. Adaptive weights are assigned according to variable importance to strengthen key representations. Finally, a two-dimensional convolutional architecture models dependencies within segments, across segments, and along the variable dimension, achieving multi-dimensional feature fusion for the season component, as illustrated in Figure 2(a).

Specifically, when dealing with season components, we first transform the one-dimensional periodic term Se_{1D} into two-dimensional data of different scales through fractional Fourier transform. The implementation process is as follows:

$$Se_{2D}^i = FrFT(Se_{1D}), i = 1, 2, \dots, k \quad (12)$$

where $FrFT$ represents fractional Fourier transform, and $Se_{2D}^i \in \mathbb{R}^{C \times S \times P}$ is two-dimensional periodic data of different scales.

We empirically evaluated how cosine similarity captures phase-consistent patterns across temporal segments, demonstrating that high cosine similarity scores correspond to stable seasonal behaviors, while low scores indicate deviations or anomalies. This analysis confirms that cosine similarity is suitable as a lightweight yet effective metric for seasonal patch clustering within our fractional-decoupled framework, providing both interpretability and robust pattern selection.

Subsequently, the clustering output between season patches is calculated through the Section 3.2 method, and then the average similarity is mapped to continuous soft clustering weights using the sigmoid function:

$$w_s = \sigma(\text{mean_sim}_s), w_s \in (0, 1) \quad (13)$$

Here $\sigma(\cdot)$ is the sigmoid function. When $w_s \approx 1$, it indicates a seasonal pattern block; When $w_s \approx 0$, it indicates that the block deviates from the overall season structure and belongs to the mutation type of block. Consecutively, to maintain the original two-dimensional structure, dual two-dimensional convolutional branches are simultaneously applied to the input period term Se_{2D}^i :

$$\begin{aligned} \text{Season}(Se_{2D}^i) &= \text{Conv}_{\text{season}}(Se_{2D}^i); \\ \text{NonSeason}(Se_{2D}^i) &= \text{Conv}_{\text{nonseason}}(Se_{2D}^i) \end{aligned} \quad (14)$$

The two branches are structurally identical but have independent parameters, and are respectively used to model stable season patterns and local mutations or abnormal disturbances. For the p th block, the final output is expressed as:

$$Y_p = w_p \cdot \text{Conv}_{season}(x_p) + (1 - w_p) \cdot \text{nonseason}(x_p) \quad (15)$$

The output of all the blocks together constitutes the final season term feature representation $Se_{2D_f}^i = Y_1, Y_2, \dots, Y_P$. For the interaction between variables, we transpose $Se_{2D_f}^i$:

$$Se_{2D_p}^i = \text{Permutation}(Se_{2D_f}^i), i = 1, 2, \dots, k \quad (16)$$

where $\text{Permutation}(\cdot)$ It indicates the transposition of the data, but in order to reduce the noise problem caused by irrelevant variables, we used the attention mechanism to weight different variables. In particular, since the focus module adopts linear transformation W to get query ($Q = W_Q \cdot \hat{Se}_{2D_p}^i$), key ($K = W_K \cdot \hat{Se}_{2D_p}^i$) and values ($V = W_V \cdot \hat{Se}_{2D_p}^i$). Then, Q, K, V are used to calculate the attention weights and perform weighting. The process is as follows:

$$\begin{aligned} \text{Attention_score} &= \frac{Q \cdot K^T}{\sqrt{d_k}} \\ \alpha &= \text{softmax}(\text{Attention_score}) \\ Se_{2D_a}^i &= \alpha \cdot V \end{aligned} \quad (17)$$

where Q is the query vector, representing the information to be compared with other vectors; K is the key vector, which is used to calculate similarity with the query vector. V is the value vector, which is the output obtained by weighting the similarity between the query and the key. $\sqrt{d_k}$ is the scaling factor, which is the square root of the dimension of the key vector. softmax normalizes the weights. Then model the characteristics among the variables. The process is as follows:

$$\begin{aligned} \hat{Se}_{2D_a}^i &= \text{Conv}_{2D}(\hat{Se}_{2D_a}^i), i = 1, 2, \dots, k \\ \hat{Se}_{2D_f}^i &= \text{Conv}_{2D}(\hat{Se}_{2D_a}^i), i = 1, 2, \dots, k \end{aligned} \quad (18)$$

Through these operations, the final season component $\hat{Se}_{2D_f}^i$ is obtained.

It is divided into seasonal type and mutation type. This method offers two advantages: capturing stable seasonal patterns while suppressing sudden disturbances.

4.3. Trend Processing Module

In trend component modeling, FDDPEM employs a sliding window to generate time segments and clusters them by directional changes (upward, downward, oscillating) to explicitly depict the evolution structure of the trend term. Through variable correlation analysis and weighting mechanisms, the model emphasizes key variables that dominate trend changes. By integrating two-dimensional convolution, trend modeling no longer relies solely on one-dimensional temporal expansion but captures local patterns, long-term trend shapes, and inter-variable interactions, thereby enhancing long-term evolution capability, as illustrated in Figure 2(b).

The trend component is divided into multiple trend patches using a sliding window, where each patch represents the local change characteristics of the time series. To characterize trend differences, the overall change of each patch is quantified via the trend patch clustering method

(Section 3.3), and patches are classified into upward, downward, and flat trends according to their average slopes.

Based on the trend patch clustering results, the final trend feature is obtained by fusing the outputs of the three branches:

$$Tr_{2D_f} = Y^{up} + Y^{down} + Y^{flat} \quad (19)$$

Since each trend block belongs to only one category, this fusion process will not introduce interference between different trend patterns. Similarly, for interaction between variables, we transpose Tr_{2D_f} :

$$Tr_{2D_p} = \text{Permutation}(Tr_{2D_f}) \quad (20)$$

here $\text{Permutation}(\cdot)$ It indicates the transposition of data, but in order to reduce the noise problem caused by irrelevant variables, we used the attention mechanism to perform weighting processing on different variables:

$$Tr_{2D_a} = \text{Weighting}(Tr_{2D_p}) \quad (21)$$

where $\text{Weighting}(\cdot)$ is the same as the attention mechanism in Section. 3.3. Finally, model the trend terms among different variables:

$$\begin{aligned} \hat{Tr}_{2D_a} &= \text{Convolutional}_{2D}(Tr_{2D_a}), \\ \hat{Tr}_{2D_F} &= \text{Permutation}(\hat{Tr}_{2D_a}). \end{aligned} \quad (22)$$

Through the above steps, the processed trend term $\hat{Tr}_{2D_F}$ is obtained. This classification distinguishes blocks with significant monotonic changes from stationary blocks with local fluctuations, avoiding interference among different trend patterns during feature extraction. Independent convolutional branches are then designed for the three trend types to learn the structural features of ascending, descending, and oscillating trends. Finally, the branch outputs are fused in the original space to form a unified trend representation, enhancing modeling ability and expression accuracy for diverse trend patterns.

4.4. Forecasting Module

In the data prediction module, we comprehensively capture the long-term and short-term dependencies in the time series by fusing the features of trend terms and period terms, thereby generating the final prediction results. Specifically, this module is divided into three steps: feature flattening, feature fusion, and predictive output.

Trend term and cycle after dealing with the antecedent module, respectively for the two-dimensional tensor $\hat{Tr}_{2D_F} \in \mathbb{R}^{B \times C \times S \times P}$ and $\hat{Se}_{2D_F}^i \in \mathbb{R}^{B \times C \times S \times P}$, where B denotes the batch size, C denotes the number of variables, S denotes the number of blocks, and P denotes the block length. To facilitate subsequent fusion and prediction, we first perform flattening operations on $\hat{Tr}_{2D_F}$ and $\hat{Se}_{2D_F}^i$:

$$\begin{aligned} Tr_{1D} &= \text{Flatten}(\hat{Tr}_{2D_F}), \\ Se_{1D_F}^i &= \text{Flatten}(\hat{Se}_{2D_F}^i) \end{aligned} \quad (23)$$

where $Tr_{1D} \in \mathbb{R}^{B \times B \times L}$, $Se_{1D}^i \in \mathbb{R}^{B \times B \times L}$ represents the one-dimensional trend term and period term respectively, and L represents the input length. Then we fuse the period term data of different periods [12], and then fuse the period term data and the trend term data. The process is as follows:

$$\begin{aligned} Se_{1DC} &= \text{Concatenate}(Se_{1D}^i), i = 1, 2, \dots, k \\ Se_{1D} &= \text{Mapping}(Se_{1DC}) \\ f_c &= \text{Concatenate}(Se_{1D}, Tr_{1D}) \\ f &= \text{Mapping}(f_c) \end{aligned} \quad (24)$$

where $\text{Mapping}(\cdot)$ Represent a learnable feature mapping function for dimension transformation and information fusion of the concatenated features. f represents the time series feature obtained by fusing the season item and the trend item at a single fractional difference order. Since different fractional-order difference orders can characterize the dynamic behavior and long-term memory strength of sequences at different time scales, FDDPEM further constructs feature representations in parallel at multiple fractional-order orders and conducts fusion modeling of features at different scales. The process is as follows:

$$\begin{aligned} f_i &= \text{Concatenate}(f_1, f_2, \dots, f_i), i = 1, 2, \dots, j \\ f_L &= \text{Mapping}(f_i) \end{aligned} \quad (25)$$

Here j is a hyperparameter representing the number of scales, corresponding to the different fractional-order difference orders adopted. The final model uses a fully connected layer for prediction:

$$y = \text{FC}(f_L) \quad (26)$$

Here, $Y \in \mathbb{R}^{C \times F}$ is the final prediction result.

5. Experiments

To comprehensively evaluate the effectiveness of the FDDPEM model in multivariate time series prediction tasks, this chapter designs experiments based on multiple public benchmark datasets and conducts comparative analyses with current mainstream models. Further reveal the intrinsic mechanism and generalization ability of FDDPEM.

5.1. Datasets

To evaluate the performance of the FDDPEM model we proposed, we applied it to four commonly used long-term prediction benchmark datasets: (1) ETT [7] contains factor data of 7 power transformers from July 2016 to July 2018. The dataset is divided into four subsets, among which ETTh1 and ETTh2 are recorded once every hour, and ETTm1 and ETTm2 are recorded once every 15 minutes. (2) Weather [22] includes 21 meteorological factors collected every 10 minutes from the Max Planck Institute for Biogeochemistry's weather station during 2020. (3) The Exchange [22] dataset records the daily exchange rate data of eight different countries from 1990 to 2016. (4) Traffic [22] collected hourly road occupancy rate data measured by 862 sensors on highways in the San Francisco Bay Area from January 2015 to December 2016.

Table 1: Details of datasets in our experiments.

Dataset	Dim	Input Len	Output Len	Split	Frequency	Type
ETTh1	7	96	{96,192,336,720}	(6:2:2)	Hourly	Electricity
Weather	21	96	{96, 192, 336, 720}	(7:1:2)	10min	Weather
Traffic	862	96	{96, 192, 336, 720}	(7:1:2)	Hourly	Transportation
Exchange	8	96	{96, 192, 336, 720}	(7:1:2)	Daily	Economy

5.2. Experimental Settings

The statistical information of these datasets is shown in Table 1. Further, we selected some state-of-the-art models as benchmarks, including Multi-modal Entity Alignment Transformer (MEAformer) [15], iTransformer [28], MFFCNN [12], Crossformer [29], PatchTST [8], MS-GNet [14] and HDMixer [18]. To ensure fair comparison, all models use the same input length ($L = 336$) and prediction length ($F \in \{96, 192, 336, 720\}$). The selected evaluation indicators are two commonly used ones in time series prediction: mean Absolute Error (MAE) and Mean square Error (MSE). The block setting of period items is the same as that in MFFCNN [12]. The sliding window of the trend term selects a window size of 16 and a step size of 8 [8]. All experiments were conducted on PyTorch2.1.0 and executed on an NVIDIA RTX4090 24GB GPU.

5.3. Experimental Results

As shown in Table 2, the experimental results fully verify the significant improvement effect of FDDPEM compared with MFFCNN. Although MFFCNN has been able to model multi-scale season patterns quite well, it still has certain limitations in scenarios such as weak seasonal data, long-term predictions, and noise interference caused by low variable correlation. Based on this, FDDPEM further enhances the model’s performance and robustness in various complex prediction scenarios by introducing a trend-period decoupling mechanism of fractional-order difference and an attention-based variable weighting and time series pattern modeling strategy.

Overall, FDDPEM outperforms MFFCNN on multiple benchmark datasets and prediction settings, particularly for long prediction lengths and weak periodicity, where its advantages are more pronounced. For example, on the Exchange dataset, FDDPEM significantly surpasses MFFCNN without trend–season decoupling. As shown in Table 1, the results verify the substantial improvement of FDDPEM over MFFCNN. Although MFFCNN effectively models multi-scale periodic patterns, it remains limited in weak seasonality, long-term forecasting, and noise interference caused by low variable correlation. By introducing fractional-order trend–season decoupling and an attention-based variable weighting and pattern modeling strategy, FDDPEM further enhances performance and robustness in complex prediction scenarios.

5.4. Ablation Study

Research on design components: To confirm the impact of different modules in the FDDPEM model on model performance, we designed detailed ablation experiments. These experiments include component removal (*w/o*) experiments to evaluate the contribution of each module to the overall performance. Among them, *w/o – fod*: Remove the fractional difference (FOD) module; *w/o – pm*: Remove the season item processing (PM) module; *w/o-tm*: Remove the trend item processing (TM) module; *w/o – pc*: Remove the Block Clustering (PC) module;

FDDPEM

Table 2: Forecasting results of FDDPEM. All models use the same input length ($L = 96$), and prediction lengths ($T \in [96, 192, 336, 720]$). Lower MSE and MAE refer to better performance. The best results are in bold, and the second results are underlined.

Years		FDDPEM		MEAformer		iTransformer		MFFCNN		Crossformer		PatchTST		MSGNet		HDMixer	
		(Ours)		(2024)		(2024)		(2025)		(2023)		(2023)		(2024)		(2024)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.356	0.381	0.390	0.412	0.387	0.391	<u>0.358</u>	<u>0.384</u>	0.418	0.439	0.414	0.419	0.392	0.412	0.373	0.398
	192	0.375	0.404	0.454	0.450	0.441	0.436	<u>0.399</u>	<u>0.405</u>	0.471	0.474	0.460	0.445	0.443	0.442	0.412	0.420
	336	0.382	0.404	0.462	0.455	0.487	0.458	<u>0.387</u>	<u>0.406</u>	0.570	0.546	0.501	0.466	0.479	0.460	0.392	0.417
	720	0.415	0.437	0.528	0.508	0.503	0.491	<u>0.419</u>	<u>0.443</u>	0.653	0.621	0.500	0.488	0.490	0.486	0.448	0.463
	Avg	0.388	0.407	0.459	0.456	0.454	0.448	<u>0.391</u>	<u>0.410</u>	0.528	0.520	0.469	0.455	0.451	0.452	0.406	0.425
Weather	96	0.141	0.181	0.162	0.214	0.174	0.213	<u>0.144</u>	<u>0.186</u>	0.175	0.240	0.177	0.219	0.162	0.212	0.145	0.199
	192	0.183	0.226	0.201	0.247	0.224	0.257	<u>0.185</u>	<u>0.229</u>	0.233	0.303	0.222	0.256	0.210	0.253	0.194	0.241
	336	0.234	0.266	0.256	0.288	0.281	0.300	<u>0.237</u>	<u>0.269</u>	0.277	0.341	0.279	0.298	0.274	0.299	<u>0.237</u>	0.281
	720	0.311	0.319	0.321	0.334	0.360	0.351	<u>0.313</u>	<u>0.322</u>	0.371	0.411	0.354	0.349	0.368	0.359	0.317	0.333
	Avg	0.217	0.248	0.345	0.388	0.260	0.280	<u>0.220</u>	<u>0.252</u>	0.264	0.324	0.258	0.281	<u>0.254</u>	0.281	0.265	0.317
Traffic	96	0.360	0.241	0.369	0.258	0.392	0.268	<u>0.366</u>	<u>0.245</u>	0.511	0.264	0.459	0.298	0.601	0.352	<u>0.366</u>	0.249
	192	0.387	0.252	0.388	0.267	0.411	0.276	0.392	0.257	0.558	0.293	0.469	0.302	0.630	0.373	<u>0.385</u>	<u>0.258</u>
	336	0.406	0.267	0.402	0.273	0.423	0.282	0.407	<u>0.270</u>	0.575	0.307	0.483	0.309	0.664	0.389	<u>0.403</u>	0.276
	720	0.433	0.283	0.434	0.287	0.459	0.300	0.438	<u>0.285</u>	0.593	0.300	0.517	0.326	0.722	0.419	0.441	0.295
	Avg	0.397	0.261	<u>0.398</u>	0.271	0.421	0.282	0.401	<u>0.264</u>	0.559	0.291	0.482	0.309	0.654	0.383	0.399	0.265
Exchange	96	0.081	0.198	0.099	0.224	0.086	0.206	<u>0.083</u>	<u>0.200</u>	0.257	0.368	0.096	0.215	0.105	0.230	0.090	0.221
	192	0.172	0.294	0.225	0.352	0.180	0.303	<u>0.175</u>	<u>0.295</u>	0.471	0.510	0.182	0.303	0.195	0.317	0.192	0.311
	336	0.328	0.411	0.467	0.508	0.335	0.419	<u>0.329</u>	<u>0.413</u>	0.984	0.767	0.346	0.428	0.359	0.436	0.387	0.461
	720	0.876	0.689	1.094	0.801	<u>0.869</u>	0.705	0.878	0.701	1.637	1.034	0.868	<u>0.700</u>	0.938	0.737	1.040	0.722
	Avg	0.364	0.398	0.471	0.471	0.368	0.408	<u>0.366</u>	<u>0.402</u>	0.837	0.670	0.373	0.412	0.399	0.430	0.427	0.429
Rank First		39		0		0		0		0		1		0		0	

w/o-attention: Remove the Attention mechanism (Attention) module. Based on the experimental results in Table 3, we can identify the following points: Firstly, compared with *w/o - fod*, the prediction accuracy of the FDDPEM model is significantly improved, indicating the effectiveness of fractional-order difference in trend modeling, especially in the Weather and ETTh1 datasets. Secondly, compared with *w/o-pm*, the FDDPEM model performs significantly better on the Exchange and Traffic datasets than the version without this module, which indicates that season modeling is crucial in these datasets. Furthermore, the experimental results of *w/o - pc* show that in the Exchange and Traffic datasets, the prediction performance of FDDPEM is significantly better than that of the version without the PC module, indicating that block clustering can effectively enhance the ability to capture season patterns.

Comparative analysis of integer-order difference and fractional-order difference: To overcome the loss of low-frequency information caused by integer-order differencing, fractional-order difference is introduced. By employing non-integer orders, it removes trends while retaining long-term dependency information, as shown in Figure 3. As illustrated in Figure 3(a), both integer- and fractional-order differences achieve effective smoothing and similar trend weakening. However, spectral analysis in Figure 3(b) reveals their essential difference: integer-order differencing significantly suppresses long-term seasonal structures and eliminates key low-frequency components, whereas fractional-order differencing stabilizes the sequence while preserving evident low-frequency peaks. Therefore, fractional-order difference alleviates trend interference without sacrificing long-term information, providing more complete and discriminative structural features for subsequent season modeling and multi-scale feature extraction.

Parameter Sensitivity Experiment: To determine the optimal number of fractional-order scales in FDDPEM, parameter sensitivity experiments were conducted on ETTh1 and Weather. The goal is to balance temporal memory extraction and computational efficiency. As shown

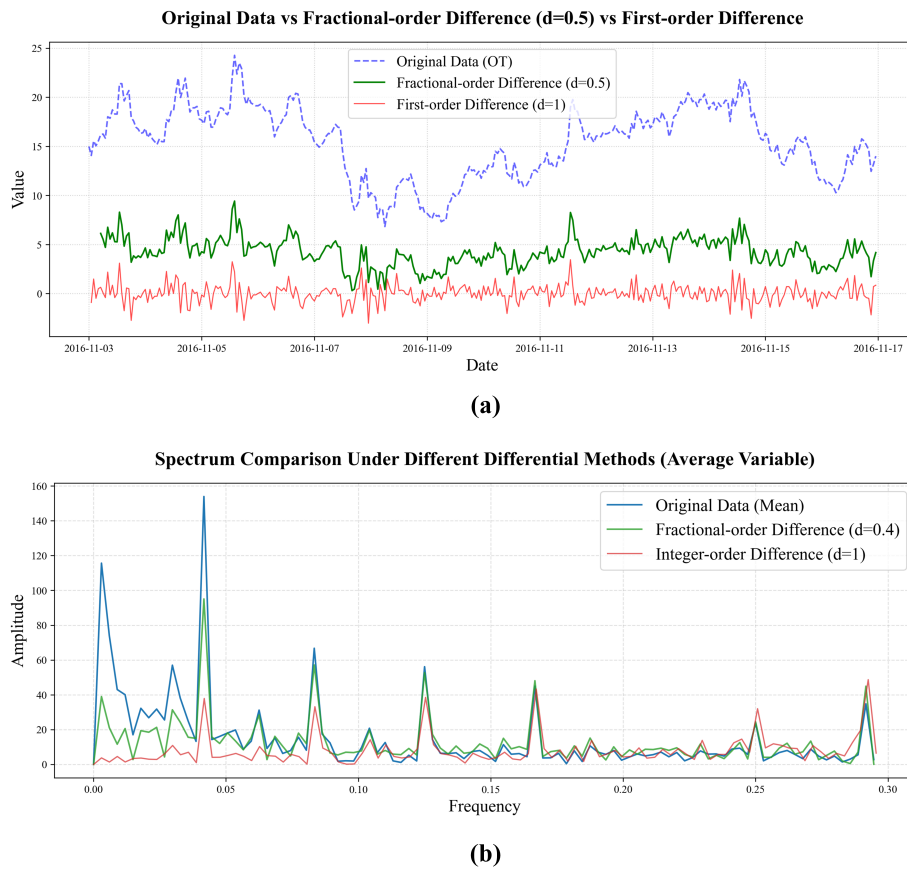


Figure 3: Comparison of fractional-order difference (0.5) and integer-order difference (1.0) in the ETTh1 dataset. (a) shows the temporal performance of the two difference methods after removing the trend; (b) presents the corresponding comparison of frequency domain features.

Table 3: Ablation results of FDDPEM, through removing the specific module of each one. The best results are in bold.

Module Datasets		FDDPEM		W/O-fod		W/O-pm		W/O-tm		W/O-pc		W/O-attention	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.356	0.381	0.362	0.387	0.358	0.387	0.359	0.385	0.359	0.386	0.360	0.388
	192	0.375	0.404	0.431	0.437	0.399	0.409	0.400	0.408	0.400	0.408	0.402	0.411
	336	0.382	0.404	0.410	0.435	0.401	0.424	0.398	0.421	0.395	0.417	0.395	0.416
	720	0.415	0.437	0.421	0.444	0.422	0.445	0.419	0.443	0.422	0.445	0.419	0.443
Weather	96	0.141	0.181	0.151	0.195	0.146	0.189	0.148	0.194	0.147	0.191	0.147	0.192
	192	0.182	0.226	0.192	0.235	0.191	0.232	0.187	0.231	0.188	0.231	0.186	0.229
	336	0.234	0.266	0.239	0.271	0.239	0.269	0.239	0.26	0.238	0.270	0.239	0.271
	720	0.311	0.319	0.316	0.324	0.315	0.323	0.313	0.322	0.314	0.322	0.314	0.323
Traffic	96	0.360	0.241	0.369	0.246	0.376	0.256	0.369	0.251	0.368	0.248	0.369	0.248
	192	0.387	0.252	0.392	0.261	0.394	0.264	0.394	0.264	0.392	0.260	0.393	0.260
	336	0.406	0.267	0.408	0.272	0.408	0.272	0.409	0.272	0.409	0.273	0.408	0.268
	720	0.433	0.283	0.439	0.288	0.439	0.290	0.439	0.290	0.440	0.287	0.440	0.290
Exchange	96	0.081	0.198	0.165	0.214	0.200	0.231	0.172	0.221	0.162	0.210	0.164	0.209
	192	0.172	0.294	0.275	0.238	0.271	0.261	0.241	0.230	0.244	0.228	0.244	0.231
	336	0.328	0.411	0.345	0.264	0.324	0.278	0.334	0.257	0.322	0.256	0.302	0.254
	720	0.876	0.689	0.886	0.711	0.858	0.715	0.838	0.685	0.831	0.683	0.838	0.685

in Figure 4, with increasing scale j , the prediction error (MSE) exhibits diminishing marginal returns.

From $j = 1$ to $j = 2$, prediction accuracy improves significantly on both datasets, indicating that a single scale cannot fully capture short-term dynamics and long-term memory, whereas a dual-scale design enriches feature representation through multi-order fusion. From $j = 2$ to $j = 3$, although inference time increases linearly (about 58% on Weather), accuracy gains nearly stagnate. This suggests that $j = 2$ sufficiently covers core fractional-order temporal features, while additional scales introduce redundancy and reduce deployment efficiency. Therefore, $j = 2$ is selected as the optimal configuration.

When determining the specific value of the order, an interval uniform distribution strategy is adopted to ensure unbiased sampling and capture intermediate temporal characteristics without prior knowledge. It also enhances feature complementarity by maximizing frequency response differences: lower orders emphasize trend retention, while higher orders focus on dynamic variations, forming effective information enhancement in the feature space. Overall, the dual-scale uniform design achieves deep temporal memory extraction under moderate computational cost and demonstrates strong cross-dataset generalization, providing stable parameter support for subsequent ablation studies.

5.5. Model Analysis

Robustness comparative analysis: As shown in Figure 5, MFFCNN heavily relies on seasonal stability. When seasonal patterns are clear, it maintains competitive under clear seasonal patterns but limited otherwise; however, under random information loss of 10%–20%, its sensitivity to data continuity causes a significant MSE increase, with a maximum growth rate of nearly 5.0%. Under Gaussian noise, the lack of effective trend modeling and noise separation further leads to performance degradation across scales ($L = 96$ to $L = 720$). In contrast, FDDPEM achieves decoupled modeling of seasonal laws and trend dynamics via fractional-order

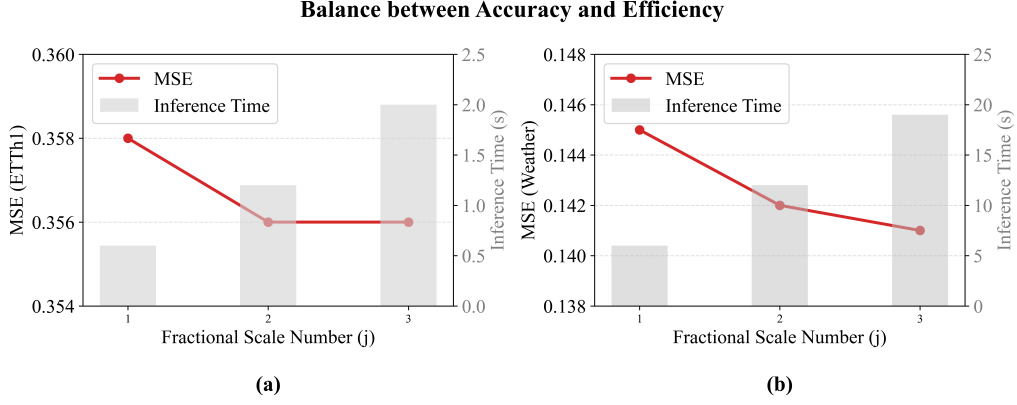


Figure 4: The variation trends of accuracy (MSE) and Time consumption (Time) with increasing scale under the ETTh1 and Weather datasets.

difference operators. Under the same interference conditions, its error deviation is substantially smaller. Even with 20% missing data, the MSE growth remains around 2.4%–2.8%, demonstrating strong robustness. In Gaussian noise environments, its performance is minimally affected, with error offsets close to zero. These results indicate that FDDPEM provides superior robustness and adaptability for complex, non-stationary, and incomplete data, offering more reliable support for long-term forecasting tasks.

Fractional order analysis: To study the effect of fractional orders on trend-season separation and long-term dependency retention, we evaluated ten orders $\alpha \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ on ETTh1 dataset. MSE was recorded under identical training and prediction settings. Results (Figure 6) show overall stable performance, with near-optimal MSE around $\alpha = 0.3$ and $\alpha = 0.6$, indicating robustness to order selection. The previously reported combination $\{0.3, 0.6\}$ lies within this stable range rather than being an absolute optimum. We updated the manuscript to remove imprecise statements and include this analysis, providing a clear empirical basis for fractional-order selection. To verify the fractional-order difference, we compared fractional-order and integer-order differences as well. While integer-order difference removes trends, it also weakens low-frequency seasonal structures, losing long-term dependencies. Fractional-order difference stabilizes the sequence while retaining low-frequency peaks, confirming its superior trend-season separation and long-range dependency preservation.

Visualization for Model Interpretability: To illustrate the FDDPEM model and clustering mechanisms, we conducted visualizations on the Weather dataset. The cosine similarity matrix among season patches are visualized (Figure 7). Strongly correlated patches cluster together, while weakly correlated or anomalous patches have lower similarity. This demonstrates how clustering identifies stable seasonal patterns and suppresses noise, enhancing intermediate feature extraction interpretability. These visualizations complement quantitative metrics, providing direct evidence of the effectiveness of fractional-order decomposition and clustering in FDDPEM.

Efficiency analysis: As shown in Figure 8, FDDPEM outperforms MFFCNN in predictive accuracy without significantly increasing resource consumption. Despite its low training overhead and memory usage, FDDPEM captures both global trends and local details more effectively,

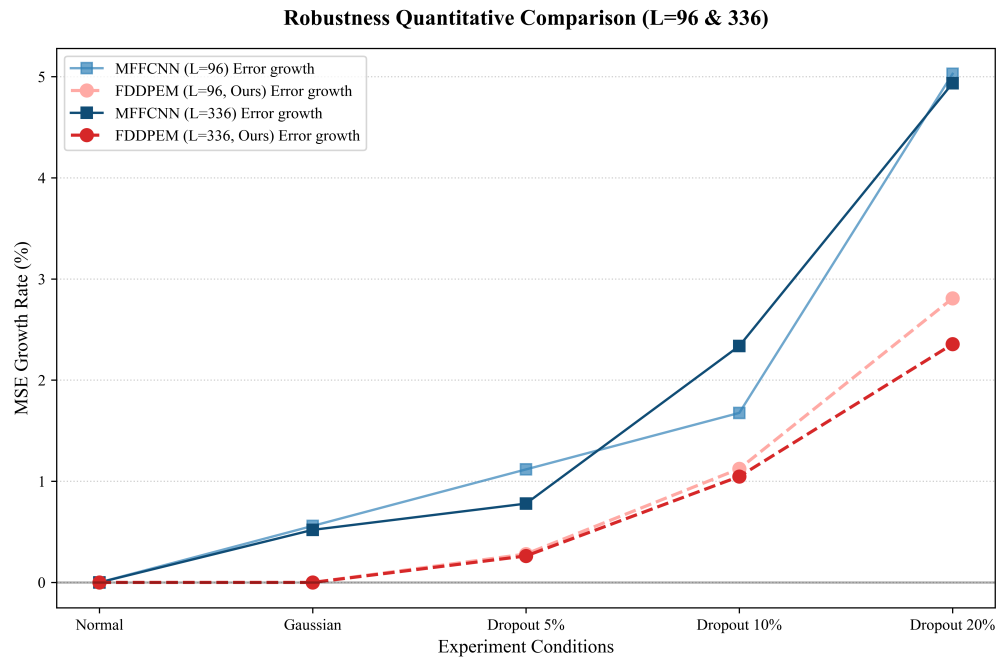


Figure 5: In the ETTh1 dataset, the evolution trend of the relative MSE growth rate of each model with the intensity of environmental interference when the prediction lengths are 96 and 336.

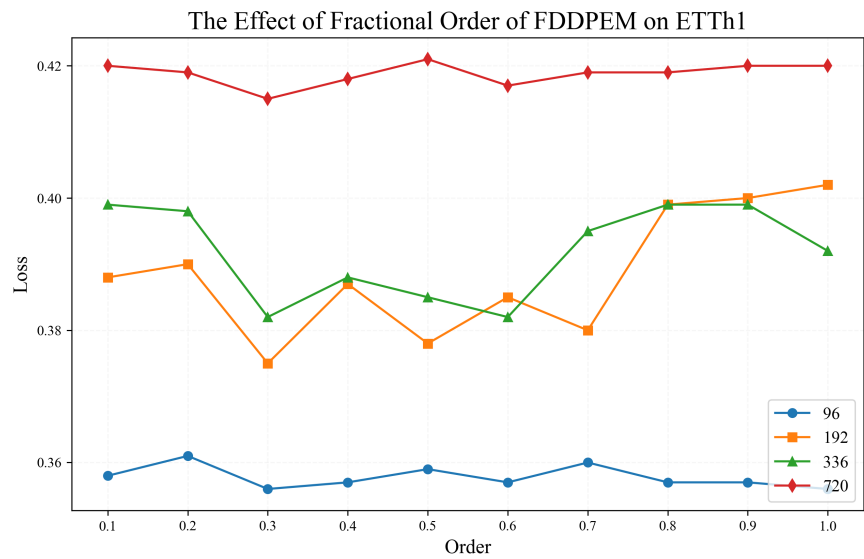


Figure 6: The effect of fractional order of FDDPEM on ETTh1

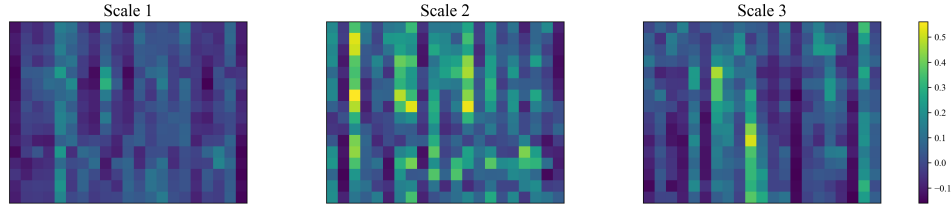


Figure 7: Cosine similarity matrix among season patches

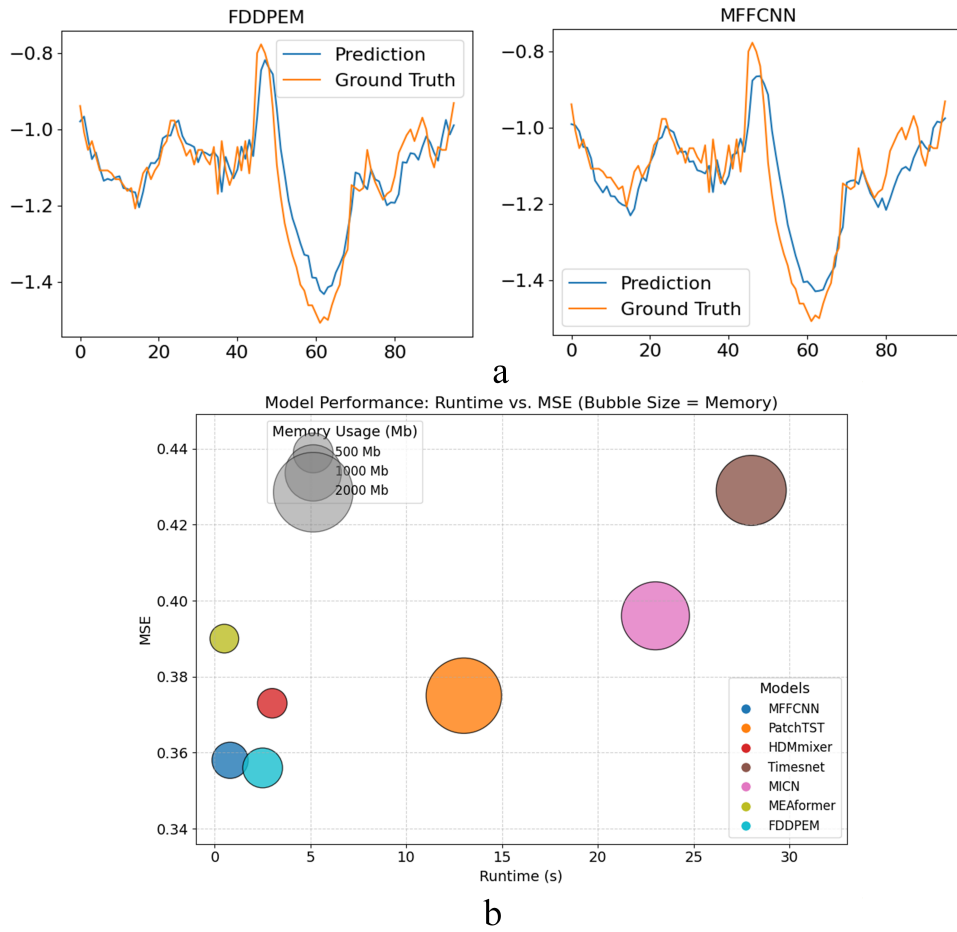


Figure 8: (a): The predictive fitting graphs of MFFCNN and FDDPEM output a length of 96 on the ETTh1 dataset. (b): Comparison of model efficiency when the output length of different methods on the ETTh1 dataset is 96.

particularly for complex fluctuations. This demonstrates that FDDPEM achieves a favorable balance between efficiency and accuracy, enhancing its applicability in scenarios requiring precise and resource-efficient predictions.

6. Conclusion

This work proposes the FDDPEM model to address complex temporal patterns and inter-variable coupling in multivariate time series forecasting. By structurally decoupling trend and seasonal components via fractional-order differencing and component-wise clustering, FDDPEM enhances both interpretability and long-term dependency modeling. Ablation studies confirm the contribution of each module, providing direct evidence of the model's effectiveness. Extensive experimental results show that FDDPEM consistently outperforms mainstream forecasting methods across benchmark datasets, while exhibiting strong robustness under noisy and missing-data conditions.

In summary, FDDPEM achieves substantial improvements in forecasting accuracy and robustness. Its strong performance on weakly seasonal data and in complex real-world environments provides an efficient and reliable solution for multivariate time series forecasting. Future research will explore the integration of causal modeling and adaptive frequency decomposition to further enhance interpretability and generalization in dynamic and large-scale forecasting scenarios.

Author Contributions

Chengli Zhou and Wuqi Chen contributed equally to this work and are joint first authors. Chengli Zhou was responsible for article writing, conceptualization and methodology development. Wuqi Chen was responsible methodology development and interpretation, and experimental implementation. Yaqun Huang contributed to theoretical analysis and supervision. Chunna Zhao contributed to experimental evaluation, supervision and correspondence for this study. All authors contributed to writing and revising the manuscript and approved the final version.

Acknowledgements

The authors thank the guide Prof. Chunna Zhao, all co-authors and reviewers for their careful review and valuable feedback.

This work is supported in part the 2026 Postgraduate Scientific Research and Innovation Project of the School of Information Science and Engineering, Yunnan University, in part by the Special Project (Post-Funding) of Yunnan Provincial Key Laboratory under Grant No. 202505AV340006.

References

- [1] C. F. Ansley and R. Kohn, "On the estimation of arima models with missing values," in *Time Series Analysis of Irregularly Observed Data: Proceedings of a Symposium Held at Texas A&M University, College Station, Texas, February 10–13, 1983*. New York, NY, USA: Springer, 1984, pp. 9–37.
- [2] X. Zhao, P. Zhao, L. Zhu, and G. Zhang, "A comparison of multivariate and univariate time series models applied in tree sap flux analyses," *Forest Sci.*, vol. 68, no. 5-6, pp. 473–486, 2022.
- [3] F. M. Salem, *Recurrent Neural Networks*. Berlin, Germany: Springer, 2022.

-
- [4] J. Park, J. Müller, B. Arora, B. Faybishenko, G. Pastorello, C. Varadharajan, R. Sahu, and D. Agarwal, “Long-term missing value imputation for time series data using deep neural networks,” *Neural Comput. Appl.*, vol. 35, no. 12, pp. 9071–9091, 2023.
 - [5] M. Liu, A. Zeng, M. Chen, Z. Xu, Q. Lai, L. Ma, and Q. Xu, “Scinet: Time series modeling and forecasting with sample convolution and interaction,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 35, 2022, pp. 5816–5828.
 - [6] A. A. Semenoglou, E. Spiliotis, and V. Assimakopoulos, “Image-based time series forecasting: A deep convolutional neural network approach,” *Neural Netw.*, vol. 157, pp. 39–53, 2023.
 - [7] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, 2021, pp. 11106–11115.
 - [8] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, “A time series is worth 64 words: Long-term forecasting with transformers,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Kigali, Rwanda, May 2023.
 - [9] K. Yi, Q. Zhang, W. Fan, S. Wang, P. Wang, H. He, N. An, D. Lian, L. Cao, and Z. Niu, “Frequency-domain mlps are more effective learners in time series forecasting,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 36, 2023, pp. 76656–76679.
 - [10] E. Koç and A. Koç, “Fractional Fourier transform in time series prediction,” *IEEE Signal Process. Lett.*, vol. 29, pp. 2542–2546, 2022.
 - [11] Z. Zhao and G. Li, “Synchrosqueezing-based short-time fractional Fourier transform,” *IEEE Trans. Signal Process.*, vol. 71, pp. 279–294, 2023.
 - [12] W. Chen, J. Ye, C. Zhao, and Y. Huang, “Mffcn: multi-scale fractional Fourier transform convolutional neural network for multivariate time series forecasting,” *J. Supercomput.*, vol. 81, no. 2, p. 416, 2025.
 - [13] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. Hoboken, NJ, USA: John Wiley & Sons, 2023.
 - [14] W. Cai, Y. Liang, X. Liu, J. Feng, and Y. Wu, “Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 10, 2024, pp. 11141–11149.
 - [15] Z. Chen, J. Chen, W. Zhang, L. Guo, Y. Fang, Y. Huang, Y. Zhang, Y. Geng, J. Z. Pan, and W. Song, “Meaformer: Multi-modal entity alignment transformer for meta modality hybrid,” in *Proc. ACM Int. Conf. Multimedia (MM)*, Ottawa, ON, Canada, 2023, pp. 3317–3327.
 - [16] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning, “Stl: A seasonal-trend decomposition,” *J. Off. Stat.*, vol. 6, no. 1, p. 3, 1990.
 - [17] J. R. M. Hosking, “Modeling persistence in hydrological time series using fractional differencing,” *Water Resour. Res.*, vol. 20, no. 12, pp. 1898–1908, 1984.

- [18] Q. Huang, L. Shen, R. Zhang, J. Cheng, S. Ding, Z. Zhou, and Y. Wang, “Hdmixer: Hierarchical dependency with extendable patch for multivariate time series forecasting,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, 2024, pp. 12608–12616.
- [19] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, “Deepar: Probabilistic forecasting with autoregressive recurrent networks,” *Int. J. Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.
- [20] B. N. Oreshkin, D. Carpo, N. Chapados, and Y. Bengio, “N-BEATS: Neural basis expansion analysis for interpretable time series forecasting,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Addis Ababa, Ethiopia, Apr. 2020.
- [21] A. Zeng, M. Chen, L. Zhang, and Q. Xu, “Are transformers effective for time series forecasting?,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 9, 2023, pp. 11121–11128.
- [22] H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: Decomposition transformers with autocorrelation for long-term series forecasting,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 34, 2021, pp. 22419–22430.
- [23] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, “Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, Baltimore, MD, USA, Jul. 2022, pp. 27268–27286.
- [24] H. Wang, J. Peng, F. Huang, J. Wang, J. Chen, and Y. Xiao, “Micn: Multi-scale local and global context modeling for long-term series forecasting,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Kigali, Rwanda, May 2023.
- [25] T. Zhou, Z. Ma, Q. Wen, L. Sun, T. Yao, W. Yin, and R. Jin, “Film: Frequency improved legendre memory model for long-term time series forecasting,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 35, 2022, pp. 12677–12690.
- [26] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, “Timesnet: Temporal 2d-variation modeling for general time series analysis,” 2022. [Online]. Available: [arXiv:2210.02186](https://arxiv.org/abs/2210.02186)
- [27] P. M. Robinson, “Time series with long memory,” in *Advanced Texts in Econometrics*. Oxford, U.K.: Oxford Univ. Press, 2003.
- [28] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, “iTransformer: Inverted transformers are effective for time series forecasting,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Vienna, Austria, May 2024.
- [29] Y. Zhang and J. Yan, “Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Kigali, Rwanda, May 2023.