A Notations

Table 5: Notation.

$\mathbf{X}_t$	multivariate time series with a lookback window of L at timestamps t, $\mathbf{X}_t \in \mathbb{R}^{N \times L}$
X_t	the multivariate values of N distinct series at timestamp t, $X_t \in \mathbf{R}^N$
$\mathbf{Y}_t$	the prediction target with a horizon window of length τ at timestamps t, $\mathbf{Y}_t \in \mathbb{R}^{N \times \tau}$
$\mathbf{H}_t$	the hidden representation of $\mathbf{X}_t$, $\mathbf{H}_t \in \mathbb{R}^{N \times L \times d}$
$\mathbf{Z}_t$	the output of the frequency channel learner, $\mathbf{Z}_t \in \mathbb{R}^{N \times L \times d}$
$\mathbf{S}_t$	the output of the frequency temporal learner, $\mathbf{S}_t \in \mathbb{R}^{N \times L \times d}$
$\mathcal{H}_{chan}$	the domain conversion of $\mathbf{H}_t$ on channel dimensions, $\mathcal{H}_{chan} \in \mathbb{C}^{N \times L \times d}$
$\mathcal{Z}_{chan}$	the FreMLP output of $\mathcal{H}_{chan}$, $\mathcal{Z}_{chan} \in \mathbb{C}^{N \times L \times d}$
$\mathcal{Z}_{temp}$	the domain conversion of $\mathbf{Z}_t$ on temporal dimensions, $\mathcal{Z}_{temp} \in \mathbb{C}^{N \times L \times d}$
$\mathcal{S}_{temp}$	the FreMLP output of $\mathcal{Z}_{temp}$, $\mathcal{S}_{temp} \in \mathbb{C}^{N \times L \times d}$
$\mathcal{W}^{chan}$	the complex number weight matrix of FreMLP in the frequency channel learner, $\mathcal{W}^{chan} \in \mathbb{C}^{d \times d}$
$\mathcal{B}^{chan}$	the complex number bias of FreMLP in the frequency channel learner, $\mathcal{B}^{chan} \in \mathbb{C}^{d}$
$\mathcal{W}^{temp}$	the complex number weight matrix of FreMLP in the frequency temporal learner, $\mathcal{W}^{temp} \in \mathbb{C}^{d \times d}$
$\mathcal{B}^{temp}$	the complex number bias of FreMLP in the frequency temporal learner, $\mathcal{B}^{temp} \in \mathbb{C}^{d}$

B Experimental Details

B.1 Datasets

We adopt thirteen real-world benchmarks in the experiments to evaluate the accuracy of short-term and long-term forecasting. The details of the datasets are as follows:

Solar[5]: It is about the solar power collected by National Renewable Energy Laboratory. We choose the power plant data points in Florida as the data set which contains 593 points. The data is collected from 01/01/2006 to 31/12/2016 with the sampling interval of every 1 hour.

Wiki[6]: It contains a number of daily views of different Wikipedia articles and is collected from 1/7/2015 to 31/12/2016. It consists of approximately $145k$ time series and we randomly choose $5k$ from them as our experimental data set.

Traffic[7]: It contains hourly traffic data from 963 San Francisco freeway car lanes for short-term forecasting settings while it contains 862 car lanes for long-term forecasting. It is collected since 01/01/2015 with a sampling interval of every 1 hour.

ECG[8]: It is about Electrocardiogram(ECG) from the UCR time-series classification archive. It contains 140 nodes and each node has a length of 5000.

---

[5] `https://www.nrel.gov/grid/solar-power-data.html`

[6] `https://www.kaggle.com/c/web-traffic-time-series-forecasting/data`

[7] `https://archive.ics.uci.edu/ml/datasets/PEMS-SF`

[8] `http://www.timeseriesclassification.com/description.php?Dataset=ECG5000`

Electricity[9]: It contains electricity consumption of 370 clients for short-term forecasting while it contains electricity consumption of 321 clients for long-term forecasting. It is collected since 01/01/2011. The data sampling interval is every 15 minutes.

COVID-19[10]: It is about COVID-19 hospitalization in the U.S. state of California (CA) from 01/02/2020 to 31/12/2020 provided by the Johns Hopkins University with the sampling interval of every day.

METR-LA[11]: It contains traffic information collected from loop detectors in the highway of Los Angeles County. It contains 207 sensors which are from 01/03/2012 to 30/06/2012 and the data sampling interval is every 5 minutes.

Exchange[12]: It contains the collection of the daily exchange rates of eight foreign countries including Australia, British, Canada, Switzerland, China, Japan, New Zealand, and Singapore ranging from 1990 to 2016 and the data sampling interval is every 1 day.

Weather[13]: It collects 21 meteorological indicators, such as humidity and air temperature, from the Weather Station of the Max Planck Biogeochemistry Institute in Germany in 2020. The data sampling interval is every 10 minutes.

ETT[14]: It is collected from two different electric transformers labeled with 1 and 2, and each of them contains 2 different resolutions (15 minutes and 1 hour) denoted with m and h. We use ETTh1 and ETTm1 as our long-term forecasting benchmarks.

B.2 Baselines

We adopt eighteen representative and state-of-the-art baselines for comparison including LSTM-based models, GNN-based models, and Transformer-based models. We introduce these models as follows:

VAR [23]: VAR is a classic linear autoregressive model. We use the Statsmodels library (`https://www.statsmodels.org`) which is a Python package that provides statistical computations to realize the VAR.

DeepGLO [36]: DeepGLO models the relationships among variables by matrix factorization and employs a temporal convolution neural network to introduce non-linear relationships. We download the source code from: `https://github.com/rajatsen91/deepglo`. We use the recommended configuration as our experimental settings for Wiki, Electricity, and Traffic datasets. For the COVID-19 dataset, the vertical and horizontal batch size is set to 64, the rank of the global model is set to 64, the number of channels is set to [32, 32, 32, 1], and the period is set to 7.

LSTNet [10]: LSTNet uses a CNN to capture inter-variable relationships and an RNN to discover long-term patterns. We download the source code from: `https://github.com/laiguokun/LSTNet`. In our experiment, we use the recommended configuration where the number of CNN hidden units is 100, the kernel size of the CNN layers is 4, the dropout is 0.2, the RNN hidden units is 100, the number of RNN hidden layers is 1, the learning rate is 0.001 and the optimizer is Adam.

TCN [11]: TCN is a causal convolution model for regression prediction. We download the source code from: `https://github.com/locuslab/TCN`. We utilize the same configuration as the polyphonic music task exampled in the open source code where the dropout is 0.25, the kernel size is 5, the number of hidden units is 150, the number of levels is 4 and the optimizer is Adam.

Informer [13]: Informer leverages an efficient self-attention mechanism to encode the dependencies among variables. We download the source code from: `https://github.com/zhouhaoyi/Informer2020`. We use the recommended configuration as the experimental settings where the dropout is 0.05, the number of encoder layers is 2, the number of decoder layers is 1, the learning rate is 0.0001, and the optimizer is Adam.

---

[9] `https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014`

[10] `https://github.com/CSSEGISandData/COVID-19`

[11] `https://github.com/liyaguang/DCRNN`

[12] `https://github.com/laiguokun/multivariate-time-series-data`

[13] `https://www.bgc-jena.mpg.de/wetter/`

[14] `https://github.com/zhouhaoyi/ETDataset`

Reformer [18]: Reformer combines the modeling capacity of a Transformer with an architecture that can be executed efficiently on long sequences and with small memory use. We download the source code from: `https://github.com/thuml/Autoformer`. We use the recommended configuration as the experimental settings.

Autoformer [14]: Autoformer proposes a decomposition architecture by embedding the series decomposition block as an inner operator, which can progressively aggregate the long-term trend part from intermediate prediction. We download the source code from: `https://github.com/thuml/Autoformer`. We use the recommended configuration as the experimental settings.

FEDformer [29]: FEDformer proposes an attention mechanism with low-rank approximation in frequency and a mixture of expert decomposition to control the distribution shifting. We download the source code from: `https://github.com/MAZiqing/FEDformer`. We use FEB-f as the Frequency Enhanced Block and select the random mode with 64 as the experimental mode.

SFM [28]: On the basis of the LSTM model, SFM introduces a series of different frequency components in the cell states. We download the source code from: `https://github.com/z331565360/State-Frequency-Memory-stock-prediction`. We follow the recommended configuration as the experimental settings where the learning rate is 0.01, the frequency dimension is 10, the hidden dimension is 10 and the optimizer is RMSProp.

StemGNN [16]: StemGNN leverages GFT and DFT to capture dependencies among variables in the frequency domain. We download the source code from: `https://github.com/microsoft/StemGNN`. We use the recommended configuration of stemGNN as our experiment setting where the optimizer is RMSProp, the learning rate is 0.0001, the number of stacked layers is 5, and the dropout rate is 0.5.

MTGNN [15]: MTGNN proposes an effective method to exploit the inherent dependency relationships among multiple time series. We download the source code from: `https://github.com/nnzhan/MTGNN`. Because the experimental datasets have no static features, we set the parameter load_static_feature to false. We construct the graph by the adaptive adjacency matrix and add the graph convolution layer. Regarding other parameters, we follow the recommended settings.

GraphWaveNet [27]: GraphWaveNet introduces an adaptive dependency matrix learning to capture the hidden spatial dependency. We download the source code from: `https://github.com/nnzhan/Graph-WaveNet`. Since our datasets have no prior defined graph structures, we use only adaptive adjacent matrix. We add a graph convolutional layer and randomly initialize the adjacent matrix. We adopt the recommended setting as its experimental configuration where the learning rate is 0.001, the dropout is 0.3, the number of epochs is 50, and the optimizer is Adam.

AGCRN [17]: AGCRN proposes a data-adaptive graph generation module for discovering spatial correlations from data. We download the source code from: `https://github.com/LeiBAI/AGCRN`. We follow the recommended settings where the embedding dimension is 10, the learning rate is 0.003, and the optimizer is Adam.

TAMP-S2GCNets [4]: TAMP-S2GCNets explores the utility of MP to enhance knowledge representation mechanisms within the time-aware DL paradigm. We download the source code from: `https://www.dropbox.com/sh/n0ajd5l0tdeyb80/AABGn-ejfV1YtRwjf_L0A0sNa?dl=0`. TAMP-S2GCNets require a pre-defined graph topology and we use the California State topology provided by the source code as input. We adopt the recommended settings as the experimental configuration for COVID-19.

DCRNN [37]: DCRNN uses bidirectional graph random walk to model spatial dependency and recurrent neural network to capture the temporal dynamics. We download the source code from: `https://github.com/liyaguang/DCRNN`. We use the recommended configuration as our experimental settings with the batch size is 64, the learning rate is 0.01, the input dimension is 2 and the optimizer is Adam. DCRNN requires a pre-defined graph structure and we use the adjacency matrix as the pre-defined structure provided by the METR-LA dataset.

STGCN [39]: STGCN integrates graph convolution and gated temporal convolution through spatial-temporal convolutional blocks. We download the source code from: `https://github.com/VeritasYin/STGCN_IJCAI-18`. We follow the recommended settings as our experimental configuration where the batch size is 50, the learning rate is 0.001 and the optimizer is Adam. STGCN requires a pre-defined graph structure and we leverage the adjacency matrix as the pre-defined structure provided by the METR-LA dataset.

LTSF-Linear [34]: LTSF-Linear proposes a set of embarrassingly simple one-layer linear models to learn temporal relationships between input and output sequences. We download the source code from: `https://github.com/cure-lab/LTSF-Linear`. We use it as our long-term forecasting baseline and follow the recommended settings as experimental configuration.

PatchTST [38]: PatchTST proposes an effective design of Transformer-based models for time series forecasting tasks by introducing two key components: patching and channel-independent structure. We download the source code from: `https://github.com/PatchTST`. We use it as our long-term forecasting baseline and adhere to the recommended settings as the experimental configuration.

B.3 Implementation Details

By default, both the frequency channel and temporal learners contain one layer of FreMLP with the embedding size d of 128, and the hidden size d_h is set to 256. For short-term forecasting, the batch size is set to 32 for Solar, METR-LA, ECG, COVID-19, and Electricity datasets. And for Wiki and Traffic datasets, the batch size is set to 4. For the long-term forecasting, except for the lookback window size, we follow most of the experimental settings of LTSF-Linear [34]. The lookback window size is set to 96 which is recommended by FEDformer [29] and Autoformer [14]. In Appendix F.2, we also use 192 and 336 as the lookback window size to conduct experiments and the results demonstrate that FreTS outperforms other baselines as well. For the longer prediction lengths (e.g., 336, 720), we use the channel independence strategy and contain only the frequency temporal learner in our model. For some datasets, we carefully tune the hyperparameters including the batch size and learning rate on the validation set, and we choose the settings with the best performance. We tune the batch size over $\{4, 8, 16, 32\}$. The codes have been uploaded as supplementary and will be publicly available soon.

B.4 Visualization Settings

The Visualization Method for Global View. We follow the visualization methods in LTSF-Linear [34] to visualize the weights learned in the time domain on the input (corresponding to the left side of Figure 1(a)). For the visualization of the weights learned on the frequency spectrum, we first transform the input into the frequency domain and select the real part of the input frequency spectrum to replace the original input. Then, we learn the weights and visualize them in the same manner as in the time domain. The right side of Figure 1(a) shows the weights learned on the Traffic dataset with a lookback window of 96 and a prediction length of 96, Figure 9 displays the weights learned on the Traffic dataset with a lookback window of 72 and a prediction length of 336, and Figure 10 is the weights learned on the Electricity dataset with a lookback window of 96 and a prediction length of 96.

The Visualization Method for Energy Compaction. Since the learned weights $\mathcal{W} = \mathcal{W}_r + j\mathcal{W}_i \in \mathbb{C}^{d\times d}$ of the frequency-domain MLPs are complex numbers, we visualize the corresponding real part $\mathcal{W}_r$ and imaginary part $\mathcal{W}_i$, respectively. We normalize them by the calculation of $1/\max(\mathcal{W}) * \mathcal{W}$ and visualize the normalization values. The right side of Figure 1(b) is the real part of $\mathcal{W}$ learned on the Traffic dataset with a lookback window of 48 and a prediction length of 192. To visualize the corresponding weights learned in the time domain, we replace the frequency spectrum of input $\mathcal{Z}_{temp} \in \mathbb{C}^{N\times L\times d}$ with the original time domain input $\mathbf{H}_t \in \mathbb{R}^{N\times L\times d}$ and perform calculations in the time domain with a weight $W \in \mathbb{R}^{d\times d}$, as depicted in the left side of Figure 1(b).

B.5 Ablation Experimental Settings

DLinear decomposes a raw data input into a trend component and a seasonal component, and two one-layer linear layers are applied to each component. In the ablation study part, we replace the two linear layers with two different frequency-domain MLPs (corresponding to DLinear (FreMLP) in Table 4), and compare their accuracy using the same experimental settings recommended in LTSF-Linear [34]. NLinear subtracts the input by the last value of the sequence. Then, the input goes through a linear layer, and the subtracted part is added back before making the final prediction. We replace the linear layer with a frequency-domain MLP (corresponding to NLinear (FreMLP) in Table 4), and compare their accuracy using the same experimental settings recommended in LTSF-Linear [34].

C Complex Multiplication

For two complex number values $\mathcal{Z}_1 = (a + jb)$ and $\mathcal{Z}_2 = (c + jd)$, where a and c is the real part of $\mathcal{Z}_1$ and $\mathcal{Z}_2$ respectively, b and d is the imaginary part of $\mathcal{Z}_1$ and $\mathcal{Z}_2$ respectively. Then the multiplication of $\mathcal{Z}_1$ and $\mathcal{Z}_2$ is calculated by:

$$\mathcal{Z}_1\mathcal{Z}_2 = (a + jb)(c + jd) = ac + j^2bd + jad + jbc = (ac - bd) + j(ad + bc) \tag{10}$$

where $j^2 = -1$.

D Proof

D.1 Proof of Theorem 1

Theorem 1. *Suppose that* $\mathbf{H}$ *is the representation of raw time series and* $\mathcal{H}$ *is the corresponding frequency components of the spectrum, then the energy of a time series in the time domain is equal to the energy of its representation in the frequency domain. Formally, we can express this with above notations by:*

$$\int_{-\infty}^{\infty} |\mathbf{H}(v)|^2 \mathrm{d}v = \int_{-\infty}^{\infty} |\mathcal{H}(f)|^2 \mathrm{d}f \tag{11}$$

where $\mathcal{H}(f) = \int_{-\infty}^{\infty} \mathbf{H}(v)e^{-j2\pi fv}\mathrm{d}v$, v *is the time/channel dimension,* f *is the frequency dimension.*

Proof. Given the representation of raw time series $\mathbf{H} \in \mathbb{R}^{N\times L\times d}$, let us consider performing integration in either the N dimension (channel dimension) or the L dimension (temporal dimension), denoted as the integral over v, then

$$\int_{-\infty}^{\infty} |\mathbf{H}(v)|^2 \mathrm{d}v = \int_{-\infty}^{\infty} \mathbf{H}(v)\mathbf{H}^*(v)\mathrm{d}v$$

where $\mathbf{H}^*(v)$ is the conjugate of $\mathbf{H}(v)$. According to IDFT, $\mathbf{H}^*(v) = \int_{-\infty}^{\infty} \mathcal{H}^*(f)e^{-j2\pi fv}\mathrm{d}f$, we can obtain

$$\begin{aligned}
\int_{-\infty}^{\infty} |\mathbf{H}(v)|^2 \mathrm{d}v &= \int_{-\infty}^{\infty} \mathbf{H}(v)[\int_{-\infty}^{\infty} \mathcal{H}^*(f)e^{-j2\pi fv}\mathrm{d}f]\mathrm{d}v \\
&= \int_{-\infty}^{\infty} \mathcal{H}^*(f)[\int_{-\infty}^{\infty} \mathbf{H}(v)e^{-j2\pi fv}\mathrm{d}v]\mathrm{d}f \\
&= \int_{-\infty}^{\infty} \mathcal{H}^*(f)\mathcal{H}(f)\mathrm{d}f \\
&= \int_{-\infty}^{\infty} |\mathcal{H}(f)|^2 \mathrm{d}f
\end{aligned}$$

Proved. □

Therefore, the energy of a time series in the time domain is equal to the energy of its representation in the frequency domain.

D.2 Proof of Theorem 2

Theorem 2. *Given the time series input* $\mathbf{H}$ *and its corresponding frequency domain conversion* $\mathcal{H}$*, the operations of frequency-domain MLP on* $\mathcal{H}$ *can be represented as global convolutions on* $\mathbf{H}$ *in the time domain. This can be given by:*

$$\mathcal{H}\mathcal{W} + \mathcal{B} = \mathcal{F}(\mathbf{H} * W + B) \tag{12}$$

where $*$ *is a circular convolution,* $\mathcal{W}$ *and* $\mathcal{B}$ *are the complex number weight and bias,* W *and* B *are the weight and bias in the time domain, and* $\mathcal{F}$ *is* DFT.

Proof. Suppose that we conduct operations in the N (i.e., channel dimension) or L (i.e., temporal dimension) dimension, then

$$\mathcal{F}(\mathbf{H}(v) * W(v)) = \int_{-\infty}^{\infty} (\mathbf{H}(v) * W(v)) e^{-j2\pi f v} \mathrm{d}v$$

According to convolution theorem, $\mathbf{H}(v) * W(v) = \int_{-\infty}^{\infty} (\mathbf{H}(\tau) W(v-\tau)) \mathrm{d}\tau$, then

$$\begin{aligned}\mathcal{F}(\mathbf{H}(v) * W(v)) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (\mathbf{H}(\tau) W(v-\tau)) e^{-j2\pi f v} \mathrm{d}\tau \mathrm{d}v \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} W(v-\tau) e^{-j2\pi f v} \mathrm{d}v \mathbf{H}(\tau) \mathrm{d}\tau\end{aligned}$$

Let $x = v - \tau$, then

$$\begin{aligned}\mathcal{F}(\mathbf{H}(v) * W(v)) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} W(x) e^{-j2\pi f (x+\tau)} \mathrm{d}x \mathbf{H}(\tau) \mathrm{d}\tau \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} W(x) e^{-j2\pi f x} e^{-j2\pi f \tau} \mathrm{d}x \mathbf{H}(\tau) \mathrm{d}\tau \\ &= \int_{-\infty}^{\infty} \mathbf{H}(\tau) e^{-j2\pi f \tau} \mathrm{d}\tau \int_{-\infty}^{\infty} W(x) e^{-j2\pi f x} \mathrm{d}x \\ &= \mathcal{H}(f) \mathcal{W}(f)\end{aligned}$$

Accordingly, $(\mathbf{H}(v) * W(v))$ in the time domain is equal to $(\mathcal{H}(f)\mathcal{W}(f))$ in the frequency domain. Therefore, the operations of FreMLP ($\mathcal{H}\mathcal{W} + \mathcal{B}$) in the channel (i.e., $v = N$) or temporal dimension (i.e., $v = L$), are equal to the operations ($\mathbf{H} * W + B$) in the time domain. This implies that frequency-domain MLPs can be viewed as global convolutions in the time domain. Proved. $\square$

E Further Analysis

E.1 Ablation Study

In this section, we further analyze the effects of the frequency channel and temporal learners with different prediction lengths on ETTm1 and ETTh1 datasets. The results are shown in Table 6. It demonstrates that with the prediction length increasing, the frequency temporal learner shows more effective than the channel learner. Especially, when the prediction length is longer (e.g., 336, 720), the channel learner will lead to worse performance. The reason is that when the prediction lengths become longer, the model with the channel learner is likely to overfit data during training. Thus for long-term forecasting with longer prediction lengths, the channel independence strategy may be more effective, as described in PatchTST [38].

Table 6: Ablation studies of the frequency channel and temporal learners in long-term forecasting. 'I/O' indicates lookback window sizes/prediction lengths.

Dataset	ETTm1								ETTh1							
I/O	96/96		96/192		96/336		96/720		96/96		96/192		96/336		96/720	
Metrics	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
FreCL	0.053	0.078	0.059	0.085	0.067	0.095	0.097	0.125	0.063	0.089	0.067	0.093	0.071	0.097	0.087	0.115
FreTL	0.053	0.078	0.058	0.084	**0.062**	**0.089**	**0.069**	**0.096**	**0.061**	**0.087**	**0.065**	**0.091**	**0.070**	**0.096**	**0.082**	**0.108**
FreTS	**0.052**	**0.077**	**0.057**	**0.083**	0.064	0.092	0.071	0.099	0.063	0.089	0.066	0.092	0.072	0.098	0.086	0.113

E.2 Impacts of Real/Imaginary Parts

To investigate the effects of real and imaginary parts, we conduct experiments on Exchange and ETTh1 datasets under different prediction lengths $L \in \{96, 192\}$ with the lookback window of 96. Furthermore, we analyze the effects of $\mathcal{W}_r$ and $\mathcal{W}_i$ in the weights $\mathcal{W} = \mathcal{W}_r + j\mathcal{W}_i$ of FreMLP. In this experiment, we only use the frequency temporal learner in our model. The results are shown in

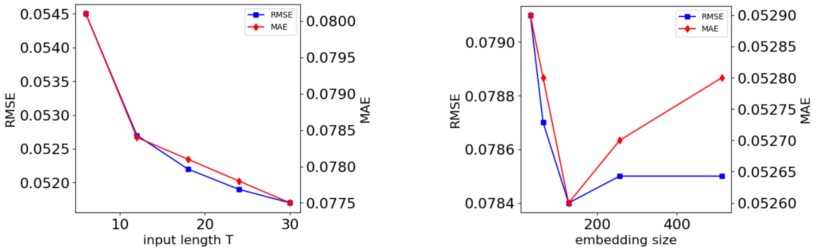
Table 7. In the table, Input$_{real}$ indicates that we only feed the real part of the input into the network, and Input$_{imag}$ indicates that we only feed the imaginary part of the input into the network. $\mathcal{W}(\mathcal{W}_r)$ denotes that we set $\mathcal{W}_i$ to 0 and $\mathcal{W}(\mathcal{W}_i)$ denotes that we set $\mathcal{W}_r$ to 0. From the table, we can observe that both the real part and imaginary part of input are indispensable and the real part is more important to the imaginary part, and the real part of $\mathcal{W}$ plays a more significant role for the model performances.

Table 7: Investigation the impacts of real/imaginary parts

Dataset	Exchange				ETTh1			
I/O	96/96		96/192		96/96		96/192	
Metrics	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Input$_{real}$	0.048	0.062	0.058	0.074	0.080	0.111	0.083	0.113
Input$_{imag}$	0.143	0.185	0.143	0.184	0.130	0.156	0.130	0.156
$\mathcal{W}(\mathcal{W}_r)$	0.039	0.053	0.051	0.067	0.063	0.089	0.067	0.093
$\mathcal{W}(\mathcal{W}_i)$	0.143	0.184	0.142	0.184	0.116	0.138	0.117	0.139
FreTS	0.037	0.051	0.050	0.067	0.061	0.087	0.065	0.091

E.3 Parameter Sensitivity

We further perform extensive experiments on the ECG dataset to evaluate the sensitivity of the input length L and the embedding dimension size d. (1) *Input length*: We tune over the input length with the value $\{6, 12, 18, 24, 30, 36, 42, 50, 60\}$ on the ECG dataset and the prediction length is 12, and the result is shown in Figure 6(a). From the figure, we can find that with the input length increasing, the performance first becomes better because the long input length may contain more pattern information, and then it decreases due to data redundancy or overfitting. (2) *Embedding size*: We choose the embedding size over the set $\{32, 64, 128, 256, 512\}$ on the ECG dataset. The results are shown in Figure 6(b). It shows that the performance first increases and then decreases with the increase of the embedding size because a large embedding size improves the fitting ability of our FSTN but may easily lead to overfitting especially when the embedding size is too large.



(a) Input window length

(b) Embedding size

Figure 6: The parameter sensitivity analyses of FreTS.

F Additional Results

To further evaluate the performance of our FreTS in multi-step forecasting, we conduct more experiments on METR-LA and COVID-19 datasets with the input length of 12 and the prediction lengths of {3, 6, 9, 12}, and the results are shown in Tables 8 and 9, respectively. In this experiment, we only select the state-of-the-art (i.e., GNN-based and Transformer-based) models as the baselines since they perform better than other models, such as RNN and TCN. Among these baselines, STGCN, DCRNN, and TAMP-S2GCNets require pre-defined graph structures. The results demonstrate that FreTS outperforms other baselines, including those models with pre-defined graph structures, at all steps. This further confirms that FreTS has strong capabilities in capturing channel-wise and time-wise dependencies.

F.1 Multi-Step Forecasting

Table 8: Multi-step short-term forecasting results comparison on the METR-LA dataset with the input length of 12 and the prediction length of $\tau \in \{3, 6, 9, 12\}$. We highlight the best results in **bold** and the second best results are underline.

Length	3		6		9		12	
Metrics	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Reformer	0.086	0.154	0.097	0.176	0.107	0.193	0.118	0.206
Informer	0.082	0.156	0.094	0.176	0.108	0.193	0.125	0.214
Autoformer	0.087	0.149	0.091	0.162	0.106	0.178	0.099	0.184
FEDformer	0.064	0.127	0.073	0.145	0.079	0.160	0.086	0.175
DCRNN	0.160	0.204	0.191	0.243	0.216	0.269	0.241	0.291
STGCN	0.058	0.133	0.080	0.177	0.102	0.209	0.128	0.238
GraphWaveNet	0.180	0.366	0.184	0.375	0.196	0.382	0.202	0.386
MTGNN	0.135	0.294	0.144	0.307	0.149	0.328	0.153	0.316
StemGNN	0.052	0.115	0.069	0.141	0.080	0.162	0.093	0.175
AGCRN	0.062	0.131	0.086	0.165	0.099	0.188	0.109	0.204
FreTS	**0.050**	**0.113**	**0.066**	**0.140**	**0.076**	**0.158**	**0.080**	**0.166**

Table 9: Multi-step short-term forecasting results comparison on the COVID-19 dataset with the input length of 12 and the prediction length of $\tau \in \{3, 6, 9, 12\}$. We highlight the best results in **bold** and the second best results are underline.

Length	3		6		9		12	
Metrics	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Reformer	0.212	0.282	0.139	0.186	0.148	0.197	0.152	0.209
Informer	0.234	0.312	0.190	0.245	0.184	0.242	0.200	0.259
Autoformer	0.212	0.280	0.144	0.191	0.152	0.201	0.159	0.211
FEDformer	0.246	0.328	0.169	0.242	0.175	0.247	0.160	0.219
GraphWaveNet	0.092	0.129	0.133	0.179	0.171	0.225	0.201	0.255
StemGNN	0.247	0.318	0.344	0.429	0.359	0.442	0.421	0.508
AGCRN	0.130	0.172	0.171	0.218	0.224	0.277	0.254	0.309
MTGNN	0.276	0.379	0.446	0.513	0.484	0.548	0.394	0.488
TAMP-S2GCNets	0.140	0.190	0.150	0.200	0.170	0.230	0.180	0.230
FreTS	**0.071**	**0.103**	**0.093**	**0.131**	**0.109**	**0.148**	**0.124**	**0.164**

F.2 Long-Term Forecasting under Varying Lookback Window

In Table 10, we present the long-term forecasting results of our FreTS and other baselines (PatchTST [38], LTSF-linear [34], FEDformer [29], Autoformer [14], Informer [13], and Reformer [18]) under different lookback window lengths $L \in \{96, 192, 336\}$ on the Exchange dataset. The prediction lengths are $\{96, 192, 336, 720\}$. From the table, we can observe that our FreTS outperforms all baselines in all settings and achieves significant improvements than FEDformer [29], Autoformer [14], Informer [13], and Reformer [18]. It verifies the effectiveness of our FreTS in learning informative representation under different lookback window.

G Visualizations

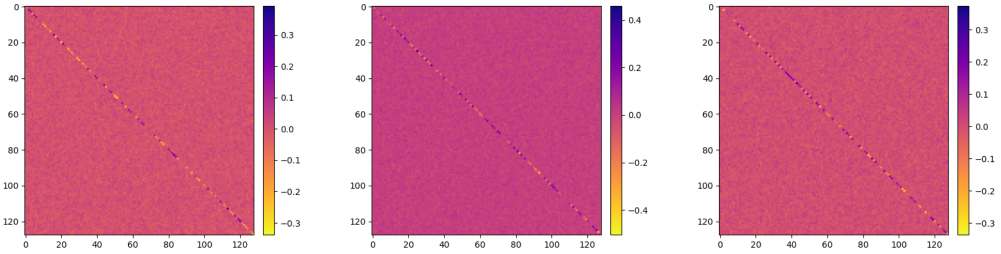
G.1 Weight Visualizations for Energy Compaction

We further visualize the weights $\mathcal{W} = \mathcal{W}_r + j\mathcal{W}_i$ in the frequency temporal learner under different settings, including different lookback window sizes and prediction lengths, on the Traffic and Electricity datasets. The results are illustrated in Figures 7 and 8. These figures demonstrate that

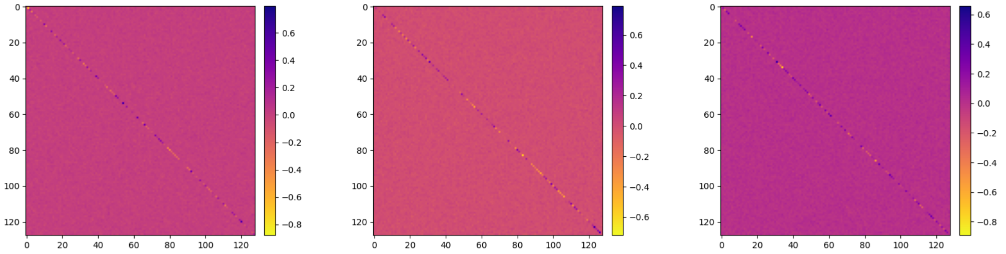
Table 10: Long-term forecasting results comparison with different lookback window lengths $L \in \{96, 192, 336\}$. The prediction lengths are as $\tau \in \{96, 192, 336, 720\}$. The best results are in **bold** and the second best results are underlined.

Models		FreTS		PatchTST		LTSF-Linear		FEDformer		Autoformer		Informer		Reformer	
Metrics		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
96	96	**0.037**	**0.051**	0.039	0.052	0.038	0.052	0.050	0.067	0.050	0.066	0.066	0.084	0.126	0.146
	192	**0.050**	**0.067**	0.055	0.074	0.053	0.069	0.064	0.082	0.063	0.083	0.068	0.088	0.147	0.169
	336	**0.062**	**0.082**	0.071	0.093	0.064	0.085	0.080	0.105	0.075	0.101	0.093	0.127	0.157	0.189
	720	**0.088**	**0.110**	0.132	0.166	0.092	0.116	0.151	0.183	0.150	0.181	0.117	0.170	0.166	0.201
192	96	**0.036**	**0.050**	0.037	0.051	0.038	0.051	0.067	0.086	0.066	0.085	0.109	0.131	0.123	0.143
	192	**0.051**	**0.068**	0.052	0.070	0.053	0.070	0.080	0.101	0.080	0.102	0.144	0.172	0.139	0.161
	336	**0.066**	**0.087**	0.072	0.097	0.073	0.096	0.093	0.122	0.099	0.129	0.141	0.177	0.155	0.181
	720	**0.088**	**0.110**	0.099	0.128	0.098	0.122	0.190	0.222	0.191	0.224	0.173	0.210	0.159	0.193
336	96	**0.038**	**0.052**	0.039	0.053	0.040	0.055	0.088	0.113	0.088	0.110	0.137	0.169	0.128	0.148
	192	**0.053**	**0.070**	0.055	0.071	0.055	0.072	0.103	0.133	0.104	0.133	0.161	0.195	0.138	0.159
	336	**0.071**	**0.092**	0.074	0.099	0.077	0.100	0.123	0.155	0.127	0.159	0.156	0.193	0.156	0.179
	720	**0.082**	**0.108**	0.100	0.129	0.087	0.110	0.210	0.242	0.211	0.244	0.173	0.210	0.168	0.205

the weight coefficients of the real or imaginary part exhibit energy aggregation characteristics (clear diagonal patterns) which can facilitate frequency-domain MLPs in learning the significant features.



(a) $\mathcal{W}_r$ under I/O=48/192 (b) $\mathcal{W}_r$ under I/O=48/336 (c) $\mathcal{W}_r$ under I/O=72/336

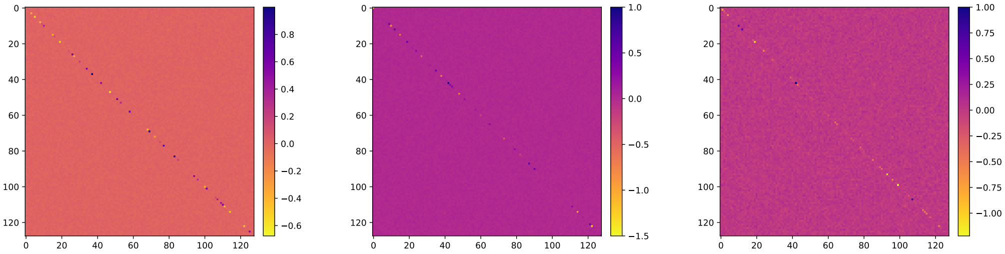


(d) $\mathcal{W}_i$ under I/O=48/192 (e) $\mathcal{W}_i$ under I/O=48/336 (f) $\mathcal{W}_i$ under I/O=72/336

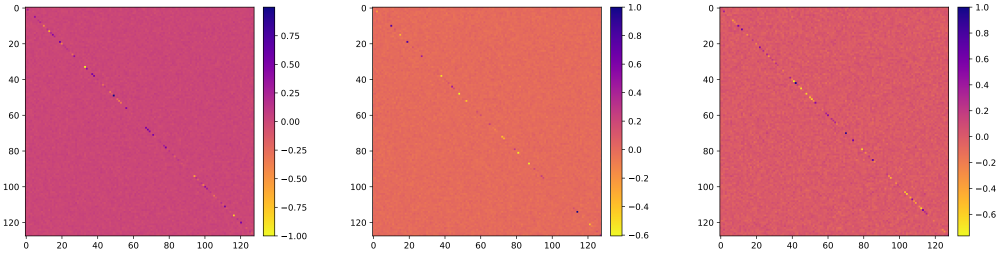
Figure 7: The visualizations of the weights $\mathcal{W}$ in the frequency temporal learner on the Traffic dataset. 'I/O' denotes lookback window sizes/prediction lengths. $\mathcal{W}_r$ and $\mathcal{W}_i$ are the real and imaginary parts of $\mathcal{W}$, respectively.

G.2 Weight Visualizations for Global View

To verify the characteristics of a global view of learning in the frequency domain, we perform additional experiments on the Traffic and Electricity datasets and compare the weights learned on the input in the time domain with those learned on the input frequency spectrum. The results are presented in Figures 9 and 10. The left side of the figures displays the weights learned on the input in the time domain, while the right side shows those learned on the real part of the input frequency spectrum. From the figures, we can observe that the patterns learned on the input frequency spectrum exhibit more obvious periodic patterns compared to the time domain. This is attributed to the global view characteristics of the frequency domain. Furthermore, we visualize the predictions of FreTS on



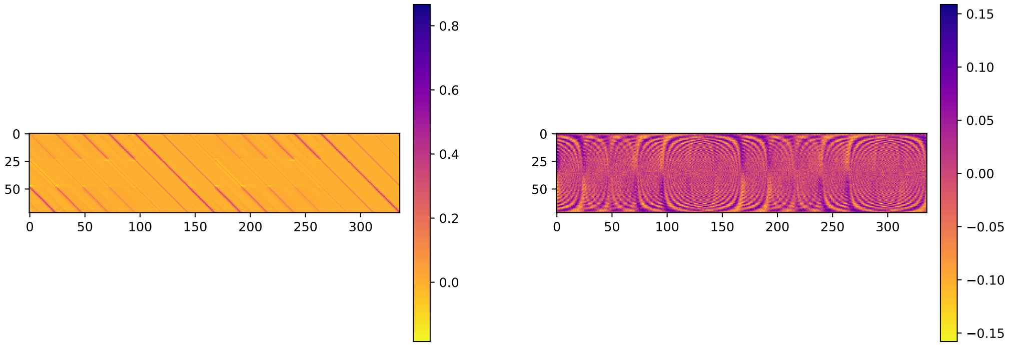
(a) $\mathcal{W}_r$ under I/O=96/96 (b) $\mathcal{W}_r$ under I/O=96/336 (c) $\mathcal{W}_r$ under I/O=96/720



(d) $\mathcal{W}_i$ under I/O=96/96 (e) $\mathcal{W}_i$ under I/O=96/336 (f) $\mathcal{W}_i$ under I/O=96/720

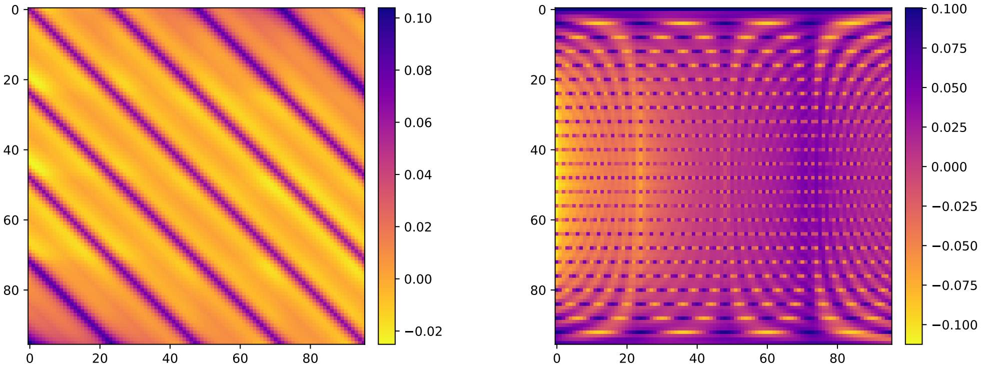
Figure 8: The visualizations of the weights $\mathcal{W}$ in the frequency temporal learner on the Electricity dataset. 'I/O' denotes lookback window sizes/prediction lengths. $\mathcal{W}_r$ and $\mathcal{W}_i$ are the real and imaginary parts of $\mathcal{W}$, respectively.

the Traffic and Electricity datasets, as depicted in Figures 11 and 12, which show that FreTS exhibit a good ability to fit cyclic patterns. In summary, these results demonstrate that FreTS has a strong capability to capture the global periodic patterns, which benefits from the global view characteristics of the frequency domain.



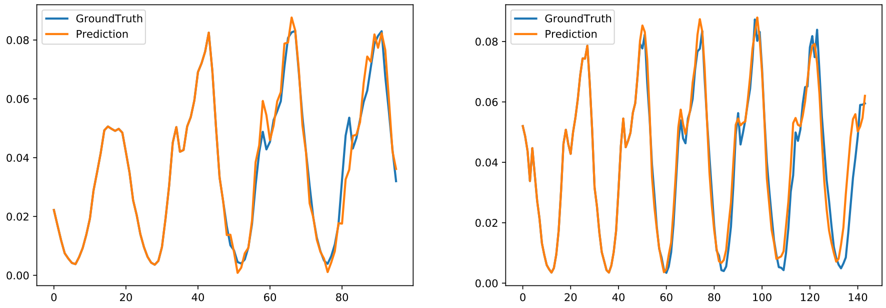
(a) Learned on the input (b) Learned on the frequency spectrum

Figure 9: Visualization of the weights ($L \times \tau$) on the Traffic dataset with lookback window size of 72 and prediction length of 336.

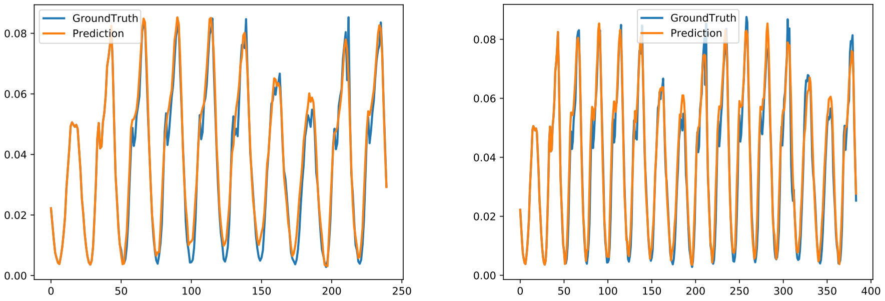


(a) Learned on the input (b) Learned on the frequency spectrum

Figure 10: Visualization of the weights ($L \times \tau$) on the Electricity dataset with lookback window size of 96 and prediction length of 96.

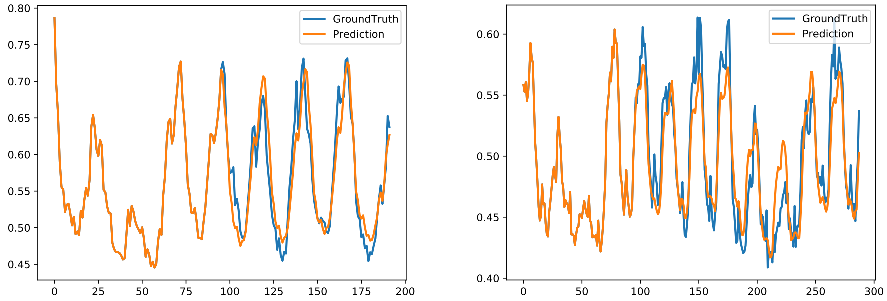


(a) I/O=48/48 (b) I/O=48/96

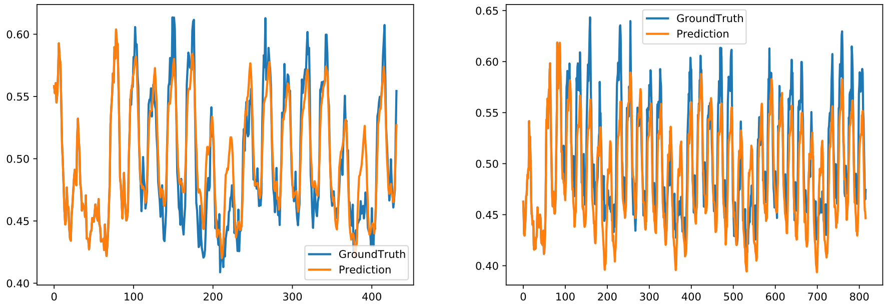


(c) I/O=48/192 (d) I/O=48/336

Figure 11: Visualizations of predictions (forecast vs. actual) on the Traffic dataset. 'I/O' denotes lookback window sizes/prediction lengths.



(a) I/O=96/96 (b) I/O=96/192



(c) I/O=96/336 (d) I/O=96/720

Figure 12: Visualizations of predictions (forecast vs. actual) on the Electricity dataset. ’I/O’ denotes lookback window sizes/prediction lengths.