# A Explanations and Proofs

## A.1 Proof of Theorem 1

**Theorem 1.** *The matrix $F$ can be obtained by solving the eigenvalue decomposition of $D^{-1/2}AD^{-1/2}$ and selecting the eigenvectors corresponding to the d largest eigenvalues.*

*Proof.*

$$\begin{aligned} Tr(F^T LF) &= Tr(F^T(I_n - D^{-1/2}AD^{-1/2})F) \\ &= Tr(F^T F - F^T D^{-1/2}AD^{-1/2}F) \\ &= Tr(I_n) - Tr(F^T D^{-1/2}AD^{-1/2}F) \end{aligned} \tag{1}$$

where $Tr(I_n) = n$ is a constant, the objective function in Definition 1 can be equivalently transformed into the following optimization problem:

$$\begin{aligned} &Max \;\; Tr(F^T D^{-1/2}AD^{-1/2}F) \\ &s.t. \qquad\quad F^T F = I \end{aligned} \tag{2}$$

The solution to this optimization problem can be obtained by solving the eigenvalue problem of $D^{-1/2}AD^{-1/2}$, specifically by selecting the eigenvectors corresponding to the k largest eigenvalues. □

## A.2 Proof of Theorem 2

**Theorem 2.** *The Stiefel Graph Spectral Convolution can be viewed as the following specific filtered graph spectral convolution:*

$$x *_s g = P g_\theta(\Lambda) P^T x \tag{3}$$

*where $\Lambda = diag(\lambda_1, \lambda_2, \cdots, \lambda_n)$, and $\lambda_1, \lambda_2, \cdots, \lambda_n$ are the eigenvalues of $A$ sorted in descending order. $P = (p_1, p_2, \cdots, p_n)$, where $p_i \in R^{n\times 1}$ is the eigenvector corresponding to $\lambda_i$. $g_\theta(\Lambda) = diag(\theta)$, and $\theta = (\theta_1, \theta_2, \cdots, \theta_n)$ satisfies:*

$$\theta_i = \begin{cases} p_i^T g & if\ \lambda_i \geq \lambda_d \\ 0 & if\ \lambda_i < \lambda_d \end{cases} \tag{4}$$

*Proof.* According to the conditions, $\lambda_1, \lambda_2, \cdots, \lambda_n$ are the eigenvalues of $A$ sorted in descending order. Let $f = (f_1, \cdots, f_d)$. By Theorem 1, $f_i (i = 1, \cdots, d)$ is the eigenvector corresponding to $\lambda_i$, then:

$$\begin{aligned} x *_s g &= F(F^T x \odot F^T g) \\ &= (f_1, f_2, \cdots, f_d)\left(\begin{pmatrix} f_1^T x \\ f_2^T x \\ \vdots \\ f_d^T x \end{pmatrix} \odot \begin{pmatrix} f_1^T g \\ f_2^T g \\ \vdots \\ f_d^T g \end{pmatrix}\right) \\ &= (f_1, f_2, \cdots, f_d)\begin{pmatrix} (f_1^T x)\cdot(f_1^T g) \\ (f_2^T x)\cdot(f_2^T g) \\ \vdots \\ (f_d^T x)\cdot(f_d^T g) \end{pmatrix} \\ &= \sum_{i=1}^{d} f_i((f_2^T x)\cdot(f_2^T g)) \end{aligned} \tag{5}$$

Given $g_\theta(\Lambda) = diag(\theta)$, where $\theta = (\theta_1, \theta_2, \cdots, \theta_n)$ and $\theta_i = \begin{cases} p_i^T G, \lambda_i \geq \lambda_d \\ 0, \lambda_i < \lambda_d \end{cases}$. Let $P = (p_1, p_2, \cdots, p_n)$, where $p_i (i = 1, \cdots, n)$ is the eigenvector corresponding to $\lambda_i$. It is clear that $f_i = p_i, i = 1, \cdots, d$. Let $g_\theta(\Lambda) = diag(\theta)$, then we have:

$$\begin{aligned} &P g_\theta(\Lambda) P^T x \\ &= (p_1, p_2, \cdots, p_n)\begin{pmatrix} p_1^T g & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & p_d^T g & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} p_1^T x \\ p_2^T x \\ \vdots \\ p_d^T x \\ \vdots \\ p_n^T x \end{pmatrix} \\ &= (p_1, p_2, \cdots, p_n)\begin{pmatrix} (p_1^T x)\cdot(p_1^T g) \\ (p_2^T x)\cdot(p_2^T g) \\ \vdots \\ (p_d^T x)\cdot(p_d^T g) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ &= \sum_{i=1}^{d} p_i((p_i^T x)\cdot(p_i^T g)) \end{aligned} \tag{6}$$

Given $f_i = p_i, i = 1, \cdots, d$, we can derive that $x *_S g = P g_\theta(\Lambda) P^T x$. □

## A.3 Proof of Algorithm 1

**Algorithm 1** Stiefel manifold optimization algorithm

**Input**: $X \in R^{n\times d}$, $E \in R^{n\times d}$
**Output**: $W \in R^{d\times d}$

1: Perform eigenvalue decomposition on $B = X^T X$ to obtain eigenvector matrix $D$ and eigenvalue diagonal matrix $\Lambda$;
2: Let $M = D\Lambda^{-1/2}D^T$;
3: Compute $C = E^T X$;
4: Compute $H = B + C^T C$;
5: Perform eigenvalue decomposition on $M^T HM$ to obtain eigenvector matrix $U$;
6: **return** $W = MU$;

*Proof.* Let $D$ and $\Lambda$ be the eigenvector matrix and the diagonal matrix of eigenvalues of the matrix $X^T X$, respectively, such that $X^T X = D\Lambda D^T$. Taking $M = D\Lambda^{-1/2}D^T$, and computing $W = MU$, we have:

$$U = M^{-1}W = D\Lambda^{1/2}D^T W \tag{7}$$

and:

$$
\begin{aligned}
U^TU &= W^TD\Lambda^{1/2}D^TD\Lambda^{1/2}D^TW = W^TD\Lambda D^TW \\
&= W^TX^TXW = I
\end{aligned}
\tag{8}
$$

Substituting $W = MU$ and $U^TU = I$ into the objective function (corresponding to equation (11) in the main manuscript):

$$
\begin{aligned}
\max \quad & \mathrm{Tr}(W^T(X^TX + X^TEE^TX)W) \\
\text{s.t.} \quad & W^TX^TXW = I
\end{aligned}
\tag{9}
$$

The optimization problem for $W$ is transformed into the following optimization problem for $U$:

$$
\begin{aligned}
&\max Tr(U^TM^T(X^TX + X^TEE^TX)MU) \\
&s.t.\ U^TU = I
\end{aligned}
\tag{10}
$$

Let $H = X^TX + X^TEE^TX$, then the above optimization problem is equivalent to finding the eigenvector matrix $U$ of $M^THM$. □

### A.4 Proof of Theorem 3

To prove Theorem 3, we first introduce two lemmas concerning the properties of the Stiefel Graph Fourier Transform.

**Lemma 1.** *The Stiefel Graph Fourier Transform satisfies:*

$$
S\left(X *_s G_1 *_s G_2 *_s \cdots *_s G_i\right) = S(X) \odot \prod_{j=1}^{i} S(G_j) \tag{11}
$$

*Proof.* Given $X *_S G = F(F^TX \odot F^TG)$, it follows that $S(X *_S G) = S(X) \odot S(G)$, therefore:

$$
\begin{aligned}
&S\left(X *_s G_1 *_s G_2 *_s \cdots *_s G_i\right) \\
&= S\left(X *_s G_1 *_s G_2 *_s \cdots *_s G_{i-1}\right) \odot S(G_i) \\
&= \cdots = S(X) \odot \prod_{j=1}^{i} S(G_j)
\end{aligned}
\tag{12}
$$

□

**Lemma 2.** *The Stiefel Graph Fourier Transform satisfies:*

$$
S\left(\sum_{i=1}^{m} Y_i\right) = \sum_{i=1}^{m} S(Y_i) \tag{13}
$$

*Proof.*

$$
S\left(\sum_{i=1}^{m} Y_i\right) = F^T\left(\sum_{i=1}^{m} Y_i\right) = \sum_{i=1}^{m} F^TY_i = \sum_{i=1}^{m} S(Y_i) \tag{14}
$$

□

**Theorem 3.** *The MSGSC has the following equivalent computational form:*

$$
MSGSC(X, G) = S^{-1}(\sum_{i=1}^{m} S(X) \odot \prod_{j=1}^{i} S(G_j)) \tag{15}
$$

*Proof.*

$$
\begin{aligned}
&S\left(\sum_{i=1}^{m} X *_s G_1 *_s G_2 *_s \cdots *_s G_i\right) \\
&= \sum_{i=1}^{m} S(X *_s G_1 *_s G_2 *_s \cdots *_s G_i) \\
&= \sum_{i=1}^{m} S(X) \odot \prod_{j=1}^{i} S(G_j)
\end{aligned}
\tag{16}
$$

Therefore,

$$
\begin{aligned}
&MSGSC(X, G) \\
&= \sum_{i=1}^{m} X *_s G_1 *_s G_2 *_s \cdots *_s G_i \\
&= S^{-1}\left(\sum_{i=1}^{m} S(X) \odot \prod_{j=1}^{i} S(G_j)\right)
\end{aligned}
\tag{17}
$$

□

## B Parameter Analysis

### B.1 The impact of the dimension of matrix $F$

| Methods | CSI300 | | Solar | |
|---|---|---|---|---|
| | MAE | MSE | MAE | MSE |
| d_model=64 | 0.3438 | 0.3419 | 0.2027 | <u>0.2132</u> |
| d_model=128 | **0.2987** | **0.3259** | **0.1802** | **0.2007** |
| d_model=256 | <u>0.3022</u> | <u>0.3281</u> | 0.1919 | 0.2150 |
| d_model=512 | 0.3057 | 0.3281 | <u>0.1916</u> | 0.2077 |

Table 1: The impact of the dimension of matrix $F$ on the results under a prediction length of 96, evaluated on the CSI300 and Solar datasets. The best results are in bold and the second best are underlined.

As shown in Table 1, the impact of the dimension of matrix $F$ on the results under a prediction length of 96 was evaluated on the CSI300 and Solar datasets. Among the tested dimensions (64, 128, 256, and 512), a dimension of 128 was found to be optimal. This suggests that a moderate dimensionality of 128 provides a good balance between model capacity and computational efficiency, allowing the model to capture essential features without excessive complexity or computational overhead.

### B.2 The impact of the number of $HP$ stacking layers

As shown in Table 2, under a prediction length of 96, the impact of the number of $HP$ stacking layers on the results was evaluated on the CSI300 and Solar datasets. Among the tested configurations (1, 2, 3, and 4 layers), two stacking layers were found to be optimal. This indicates that a moderate number of stacking layers can effectively capture the complex spatio-temporal dependencies in the data without introducing unnecessary model complexity or computational burden. Two layers provide sufficient depth to model the underlying

| Methods | CSI300 | | Solar | |
|---|---|---|---|---|
| | MAE | MSE | MAE | MSE |
| 1 | 0.3501 | 0.3457 | 0.2036 | 0.2307 |
| 2 | 0.2987 | **0.3259** | **0.1802** | **0.2007** |
| 3 | **0.2914** | 0.3308 | 0.1918 | 0.2031 |
| 4 | 0.3330 | 0.3407 | 0.1936 | 0.2107 |

Table 2: The impact of the number of $HP$ stacking layers on the results under a prediction length of 96, evaluated on the CSI300 and Solar datasets. The best results are in bold and the second best are underlined.

patterns while maintaining efficiency.

# C Time Consumption Analysis

| Methods | | Solar | | | |
|---|---|---|---|---|---|
| | | 96 | 192 | 336 | 720 |
| DST-SGNN | Train | **58.09** | **62.05** | **62.86** | **66.53** |
| | Inference | **15.18** | **17.59** | **17.65** | **21.1** |
| | Parameter | **3318K** | **3417K** | **3564K** | **3958K** |
| MiTSformer | Train | 444.37 | 439.62 | 436.22 | 415.8 |
| | Inference | 130.44 | 123.9 | 125.62 | 90.06s |
| | Parameter | 6848K | 6922K | 7033K | 7329K |
| Fredformer | Train | 229.37 | 216.84 | 311.81 | 890.46 |
| | Inference | 19.05 | 21.8 | 31.32 | 58.89 |
| | Parameter | 160741K | 321375K | 562567K | 605942K |
| iTransformer | Train | 98.5 | 98.73 | 97.55 | 106.78 |
| | Inference | 56.31 | 67.3 | 111.35 | 182.18 |
| | Parameter | 6411K | 6461K | 6534K | 6534K |

Table 3: Experimental time consumption (seconds per epoch) and parameter analysis on Solar dataset. The best results are in bold and the second best are underlined.

| Methods | | PEMS03 | | | |
|---|---|---|---|---|---|
| | | 96 | 192 | 336 | 720 |
| DST-SGNN | Train | 171.94 | 177.98 | 186.97 | 200.65s |
| | Inference | 45.14 | 54.8 | 67.81 | 93.17 |
| | Parameter | **2594K** | **2692K** | **2840K** | **3233K** |
| MiTSformer | Train | 1105.19 | 1118.38 | 1101.96 | 1098.73 |
| | Inference | 318.67 | 334.34 | 319.64 | 310.16 |
| | Parameter | 6848K | 6922K | 7033K | 7329K |
| Fredformer | Train | 598.41 | 558.08 | 557.65 | 602.74 |
| | Inference | 39.54 | 47.69 | 52.95 | 69.71 |
| | Parameter | 48688K | 62153K | 108995K | 367347K |
| iTransformer | Train | **22.93** | **23.45** | **24.67** | **28.73** |
| | Inference | **15.19** | **23.57** | **35.16** | **65.35** |
| | Parameter | 6411K | 6461K | 6534K | 6731K |

Table 4: Experimental time consumption (seconds per epoch) and parameter analysis on PEMS03 dataset. The best results are in bold and the second best are underlined.

| Methods | | exchange_rate | | | |
|---|---|---|---|---|---|
| | | 48 | 96 | 192 | 336 |
| DST-SGNN | Train | 37.75 | 37.08 | 36.36 | 34.23 |
| | Inference | 8.65 | 8.37 | 7.78 | 6.93 |
| | Parameter | 1465K | 1514K | **1613K** | **1760K** |
| MiTSformer | Train | 23.28 | 23.13 | 22.91 | 22.16 |
| | Inference | 10.5 | 10.56 | 9.64 | 8.8 |
| | Parameter | 6811K | 6848K | 6922K | 7033K |
| Fredformer | Train | 11.79 | 11.77 | 11.71 | 10.43 |
| | Inference | **0.87** | **0.73** | **0.72** | **0.8** |
| | Parameter | **680K** | **1278K** | 2572K | 4754K |
| iTransformer | Train | **10.81** | **10.91** | **10.71** | **10.41** |
| | Inference | 1.59 | 1.39 | 1.68 | 1.23 |
| | Parameter | 6387K | 6411K | 6461K | 6534K |

Table 5: Experimental time consumption (seconds per epoch) and parameter analysis on exchange_rate dataset. The best results are in bold and the second best are underlined.

As Table 3, Table 4, and Table 5 show, our method demonstrates unique advantages in terms of the number of parameters and time efficiency. In terms of the number of parameters, our method shows an advantage across all the tested datasets. Regarding time efficiency, its performance is closely related to the variable scale of the dataset. Especially on datasets with a rich number of variables, the advantage in operational efficiency is even more prominent.

Taking the Solar dataset as an example, our method is relatively leading in terms of time consumption and has the fastest running speed. During the training process on the PEMS03 dataset, compared with the two spatiotemporal models, Fredformer and MiTSformer, the training time of our method is significantly shortened, which also proves its high efficiency. However, on the exchange_rate dataset, our method performs relatively average. This is mainly because this dataset has only 8 variables and is small in scale, so it is unable to fully utilize the advantages of our method in handling large-scale variables and complex graph structures.

It can be seen that when facing datasets with a large number of variables, our method can fully unleash its performance potential, achieve more efficient operation and processing, and has significant value in practical applications.