An Improved Jacobi Algorithm for Frequency-Domain Critically Sampled Graph Filter Banks (Postprint)

Authors: Li Weijing, Jiang Junzheng, Jiang Junzheng

Date: 2023-02-15T00:00:00+00:00

Abstract

Frequency-domain critically-sampled graph filter banks require eigendecomposition of the Laplacian matrix, which results in prohibitively high computational complexity for this framework. To address this issue, an improved Jacobi algorithm is employed to approximately solve for the eigenmatrix of this framework, thereby reducing computational complexity. The improved Jacobi algorithm reformulates the problem of approximating the eigenmatrix as a constrained optimization problem, using the approximation error of the Laplacian matrix as the objective function and the sparse orthogonality of the approximate eigenmatrix as the constraint, thereby solving for the approximate eigenmatrix. Theoretical and simulation results demonstrate that using the approximate eigenmatrix in frequency-domain critically-sampled graph filter banks does not alter their perfect reconstruction condition, and compared with existing frequency-domain critically-sampled graph filter banks, the improved Jacobi algorithm maintains favorable denoising performance while reducing computational complexity.

Full Text

Preamble

Journal of Guilin University of Electronic Technology
ChinaXiv Partner Journal, Vol. 43, No. 2, April 2023

An Improved Jacobi Algorithm for Frequency-Domain Critically Sampled Graph Filter Banks

Authors: LI Weijing, JIANG Junzheng
(School of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: Frequency-domain critically sampled graph filter banks require eigenvalue decomposition of the Laplacian matrix, which leads to excessively high computational complexity. To address this problem, an improved Jacobi algorithm is employed to approximately solve for the eigenmatrix of the framework, thereby reducing computational complexity. The improved Jacobi algorithm formulates the problem of approximating the eigenmatrix as a constrained optimization problem, using the approximation error of the Laplacian matrix as the objective function and the sparse orthogonality of the approximate eigenmatrix as the constraint condition to solve for the approximate eigenmatrix. Theoretical and simulation results demonstrate that using the approximate eigenmatrix in frequency-domain critically sampled graph filter banks does not alter their perfect reconstruction conditions. Compared with existing frequency-domain critically sampled graph filter banks, the improved Jacobi algorithm reduces computational complexity while maintaining excellent denoising performance.

Keywords: graph filter banks; improved Jacobi algorithm; graph Fourier transform; TP301.6

1. Introduction

High-dimensional irregular data from sensor networks and brain neural networks [1-3] can be modeled as graph signals. Unlike traditional discrete signals defined on regular time and spatial domains, graph signals are typically defined on irregular non-Euclidean domains, making conventional signal processing methods unsuitable. Processing such high-dimensional irregular data has become an urgent problem to solve. Researchers have proposed graph signal processing frameworks [4] that combine traditional signal processing methods with graph theory, providing powerful tools for handling these data. Graph filter banks [5-7], extended from traditional filter banks, have gained widespread attention due to their sparse characteristics and multiresolution analysis capabilities.

Design methods for graph filter banks mainly include vertex-domain sampling and frequency-domain sampling approaches. Two-channel critically sampled graph filter banks in the vertex domain satisfy orthogonal properties but lack compact support. Biorthogonal graph wavelet filter banks have been proposed to achieve both perfect reconstruction and compact support. However, these vertex-domain sampling filter banks are essentially only applicable to bipartite graphs; for non-bipartite graphs, approximation is required. Spline-like graph wavelet filter banks satisfy perfect reconstruction and graph invariance, making them applicable to graph signals with arbitrary topological structures. Nevertheless, these vertex-domain sampling graph filter banks have limitations: they require selecting appropriate sampling sets to ensure perfect reconstruction, the sampling sets for perfect reconstruction are not unique, and different sampling sets affect the overall performance of the graph filter banks.

To overcome these limitations, two-channel critically sampled graph filter banks in the frequency domain have been proposed [11]. Frequency-domain sampling graph filter banks address the shortcomings of vertex-domain sampling approaches: their perfect reconstruction sampling set is unique and satisfies perfect reconstruction characteristics for graph signals with arbitrary topological structures. However, because the sampling operation is performed in the frequency domain, eigenvalue decomposition is required to obtain the eigenvector matrix of the graph model, resulting in excessively high computational complexity.

To address the computational complexity issues in frequency-domain critically sampled graph filter banks, this paper employs improved Jacobi algorithms—including truncated Jacobi and parallel truncated Jacobi algorithms—to approximately solve for the eigenmatrix. The improved Jacobi algorithm approximates the eigenmatrix by multiplying Givens rotation matrices obtained in each iteration, thereby approximating the frequency-domain representation of graph signals. Theoretical analysis shows that the improved Jacobi algorithm is a greedy algorithm that belongs to the class of sparse orthogonal matrices, reducing the computational complexity of frequency-domain critically sampled graph filter banks while satisfying perfect reconstruction conditions.

2. Fundamentals of Graph Signal Processing

Consider an undirected graph $G = (V, E)$, where V represents the vertex set and E represents the edge set. The graph G can be represented by a weight matrix $W \in \mathbb{R}^{N\times N}$, whose element w_{ij} represents the association degree between vertex i and vertex j. If $w_{ij} = 0$, it indicates no edge connection between vertices i and j. The degree matrix D is a diagonal matrix whose diagonal elements equal the sum of the corresponding row elements in the weight matrix W. The Laplacian matrix L is defined as the difference between the degree matrix and the weight matrix: $L = D - W$.

A graph signal can be represented as an N-dimensional vector $f = [f_0, f_1, \cdots, f_N]^T$, where the i-th element corresponds to the signal value at the i-th node on the graph. Similar to traditional discrete signals, graph signals have corresponding frequency-domain representations. The graph Fourier transform of graph signal f is defined as:

$$\hat{f} = U^T f \quad (2)$$

The inverse graph Fourier transform is defined as:

$$f = U\hat{f} \quad (3)$$

Equations (2) and (3) are called the graph Fourier transform pair, where U is the eigenmatrix obtained by eigenvalue decomposition of the Laplacian matrix L.

Performing eigenvalue decomposition on the Laplacian matrix L yields:

$$L = U\Lambda U^T$$

where $\Lambda \in \mathbb{R}^{N\times N}$ is a diagonal matrix whose diagonal elements are the eigenvalues of L arranged in ascending order, and $U \in \mathbb{R}^{N\times N}$ is the eigenmatrix whose column vectors correspond to the eigenvectors of L.

Graph filtering can be considered from both vertex-domain and frequency-domain perspectives. In the vertex domain, graph filtering can be expressed as a polynomial of the Laplacian matrix, and the graph filtering process for graph signal f can be represented as:

$$f_{\text{out}} = \sum_{k=0}^{K} h_k L^k f \quad (5)$$

where h_k represents the filter coefficients. After graph filtering, the signal value at each vertex is a linear combination of signal values in its neighborhood. In the frequency domain, the graph filtering process can be represented as:

$$\hat{f}_{\text{out}} = H(\Lambda)\hat{f}$$

where $H(\Lambda) = \text{diag}(H(\lambda_0), H(\lambda_1), \dots, H(\lambda_{N-1}))$ is the frequency response of the graph filter, and $H(\lambda)$ is a polynomial of graph frequencies.

3. Two-Channel Frequency-Domain Critically Sampled Graph Filter Banks

Figure 1 shows the basic structure of a two-channel critically sampled graph filter bank in the frequency domain [11]. The input-output relationship of the two-channel frequency-domain critically sampled graph filter bank is given by:

$$\tilde{f} = G_0(\Lambda)S_bS_aH_0(\Lambda)f + G_1(\Lambda)S_bS_aH_1(\Lambda)f$$

where $H_k(\Lambda) = \text{diag}(H_k(\lambda_0), H_k(\lambda_1), \dots, H_k(\lambda_{N-1}))$ represents the frequency response of the k-th filter in the analysis filter bank, $G_k(\Lambda) = \text{diag}(G_k(\lambda_0), G_k(\lambda_1), \dots, G_k(\lambda_{N-1}))$ represents the frequency response of the k-th filter in the synthesis filter bank, $S_a = [I_{N/2}, \mathbf{0}; \mathbf{0}, J_{N/2}]$ is the frequency-domain downsampling matrix, and $S_b = [I_{N/2}, \mathbf{0}; \mathbf{0}, J_{N/2}]$ is the frequency-domain upsampling matrix, with $I_{N/2}$ and $J_{N/2}$ being the identity matrix and reverse identity matrix, respectively [12].

According to the input-output relationship, when the frequency responses of the subband filters satisfy:

$$H_0^2(\lambda_i) + H_0^2(\lambda_{N-i-1}) = c^2$$

$$H_1(\lambda_i) = H_0(\lambda_{N-i-1})$$

$$G_0(\lambda_i) = H_0(\lambda_i)$$
$$G_1(\lambda_i) = H_1(\lambda_i)$$

for $i = 0, 1, \ldots, N/2 - 1$, the filter bank satisfies the perfect reconstruction condition [11] with transfer function $T = c^2 I$.

Existing design methods mainly include orthogonal [8] and biorthogonal [9] approaches. The orthogonal design of graph filter banks depends on $H_0(\lambda_i)$ satisfying:

$$H_0^2(\lambda_i) + H_0^2(\lambda_{N-i-1}) = c^2$$

The design of two-channel frequency-domain sampling graph filter banks borrows from classical signal processing methods [8]. To ensure perfect reconstruction characteristics of graph filter banks, the remaining filters $H_1(\lambda_i)$, $G_0(\lambda_i)$, and $G_1(\lambda_i)$ are defined by $H_0(\lambda_i)$. For biorthogonal frequency-domain sampling graph filter banks, $H_0(\lambda_i)$ is defined by the low-pass filter [9]:

$$H_0(\lambda_i) + H_0(\lambda_{N-i-1}) = c^2$$

Similar to biorthogonal wavelet transforms in classical signal processing [13], the perfect reconstruction condition for biorthogonal frequency-domain sampling graph filter banks is:

$$H_0(\lambda_i)G_0(\lambda_i) + H_0(\lambda_{N-i-1})G_0(\lambda_{N-i-1}) = c^2$$

4. Fast Implementation Methods for Frequency-Domain Critically Sampled Graph Filter Banks

Frequency-domain critically sampled graph filter banks require computing the graph Fourier transform of graph signals, necessitating eigenvalue decomposition of the Laplacian matrix L to obtain its eigenmatrix U. This operation has a computational complexity of $O(N^3)$. When the graph scale is small, the eigenmatrix can be obtained quickly, but when the graph scale is large, the computational cost becomes prohibitive.

To reduce the computational complexity of solving for eigenmatrix U, this paper employs improved Jacobi algorithms to approximately solve for U. The purpose of approximately solving for eigenmatrix U is to find an approximate eigenmatrix $\hat{U}$ such that:

$$\hat{U} \approx U$$

where $\hat{U}$ can be expressed as a product of sparse orthogonal matrices:

$$\hat{U} = \prod_{j=1}^{J} S_j$$

with S_j being sparse orthogonal matrices belonging to set S.

To measure the computational complexity between approximate eigenmatrix $\hat{U}$ and eigenmatrix U, the relative complexity gain (RCG) is defined as [14]:

$$\mathrm{RCG} = \frac{\mathrm{nnz}(U)}{\sum_{j=1}^{J} \mathrm{nnz}(S_j)}$$

where $\mathrm{nnz}(U)$ represents the number of non-zero elements in matrix U. The relative complexity gain is the ratio of non-zero elements in eigenmatrix U to the sum of non-zero elements in the sparse orthogonal factor matrices S_j.

4.1 Truncated Jacobi Algorithm

The truncated Jacobi algorithm is an improvement of the Jacobi algorithm [15]. While the traditional algorithm stops iterating when the approximate diagonal matrix L_j reaches a certain precision, the truncated Jacobi algorithm pre-sets the number of iterations and constrains the sparse orthogonal matrix S_j to the set of Givens rotation matrices R_G, yielding the truncated Jacobi algorithm [14].

A Givens rotation matrix can be expressed as $G_{p,q,\theta}$, where p and q are rotation coordinates and $\theta \in [0, 2\pi]$ is the rotation angle. The Givens rotation matrix is determined by three parameters (p, q, θ). Each iteration aims to find a Givens rotation matrix that minimizes the cost function most rapidly [14].

The truncated Jacobi algorithm is a greedy algorithm. Even when the relative complexity gain is large, the time between using approximate $\hat{U}$ and exact eigenmatrix U may be relatively long. The algorithm proceeds as follows:

Algorithm 1: Truncated Jacobi Algorithm 1. Input: Laplacian matrix L, number of Givens rotations J 2. Initialize: $L_0 = L$, $S = I$ 3. For $j = 1$ to J: a. Find the element with maximum absolute value $|l_{pq}|$ in matrix L_j and determine its row coordinate p and column coordinate q b. Compute rotation angle: $\theta = \frac{1}{2} \arctan\left(\frac{2l_{pq}}{l_{qq} - l_{pp}}\right)$ c. Construct Givens rotation matrix $G_{p,q,\theta}$ d. Update: $L_{j+1} = G_{p,q,\theta}^T L_j G_{p,q,\theta}$, $S = SG_{p,q,\theta}$ e. If off-diagonal elements are sufficiently small, terminate iteration 4. Output: Approximate eigenmatrix $\hat{U} = S$, approximate eigenvalues $\hat{\Lambda} = \mathrm{diag}(L_J)$

4.2 Parallel Truncated Jacobi Algorithm

The relative complexity gain provided by the truncated Jacobi algorithm is only a theoretical measure. In practical computation, the time overhead of using approximate eigenmatrix $\hat{U}$ is not entirely proportional to its relative complexity gain. In programming environments (e.g., MATLAB), multiplying S_j with vectors is a serial process, while multiplying eigenmatrix U with vectors is a parallel process [14].

The parallel truncated Jacobi algorithm [14], improved from the truncated Jacobi algorithm, better addresses this issue. While the truncated Jacobi algo-

rithm performs only one Givens rotation per iteration, the parallel truncated Jacobi algorithm can perform n Givens rotations simultaneously. For J required Givens rotations, the parallel truncated Jacobi algorithm only needs $K = J/n$ iterations. Each rotation factor S_k is a matrix composed of n Givens rotations, expressed as:

$$S_k = \prod_{i=1}^{n} G_{p_i,q_i,\theta_i}$$

Similar to the truncated Jacobi algorithm, the parallel truncated Jacobi algorithm also determines rotation coordinates by finding elements with maximum absolute values in matrix L_k. However, the selected n elements must ensure they are not in the same row or column. Although the solution obtained by the parallel truncated Jacobi algorithm is not optimal, its actual runtime is much lower than that of the truncated Jacobi algorithm [14].

Algorithm 2: Parallel Truncated Jacobi Algorithm 1. Input: Laplacian matrix L, number of Givens rotations J 2. Initialize: $L_0 = L$, $S = I$, $t = J/n$ 3. For $k = 1$ to t: a. Extract upper triangular elements of L_k and sort them in descending order b. Select n maximum elements ensuring no two share the same row or column c. For each selected element, compute corresponding rotation angle θ_i and construct G_{p_i,q_i,θ_i} d. Assemble S_k by placing rotations in appropriate positions e. Update: $L_{k+1} = S_k^T L_k S_k$, $S = SS_k$ f. If termination condition met, output approximate eigenmatrix $\hat{U} = S$ and eigenvalues $\hat{\Lambda} = \text{diag}(L_k)$

5. Computational Complexity Analysis

For the truncated Jacobi algorithm, finding the element with maximum absolute value in matrix L_j is the most computationally expensive operation with complexity $O(N^2)$. In the worst case, the elements in rows p, q and columns p, q of L_j before and after Givens rotation are different. If the current iteration' s rotation coordinates (p, q) differ from the previous selection, this operation' s complexity is $O(N^2)$. However, in actual computation, most cases involve different coordinates between iterations, so the average computational complexity of the truncated Jacobi algorithm is $O(JN^2)$.

For the parallel truncated Jacobi algorithm, sorting elements of matrix L_k is the most expensive operation with complexity $O(N^2 \log N)$. Since only $K = J/n$ iterations are needed, the complexity of multiplying the matrix with vectors is $O(JN)$. Each rotation factor S_k in the parallel truncated Jacobi algorithm contains n Givens rotations, making it a parallel computation process. The computational complexity of these approximate diagonalization algorithms is lower than exact diagonalization. To ensure convergence, the iteration count is typically set to $J = N(N-1)/2$. After obtaining approximate eigenmatrix $\hat{U}$, the overall computational complexity is $O(N^2 \log N)$.

6. Simulation Results and Analysis

6.1 Graph Signal Denoising

We compare the performance of truncated and parallel truncated Jacobi algorithms with algorithms from references [8], [9], and [18] for graph signal denoising. Simulations are conducted in the same environment on three graph topologies constructed using GSPBOX [18]: sensor, roll, and Community. Signals are formed by summing the first few eigenvectors of the graph Laplacian matrix. Additive white Gaussian noise with standard deviation σ is added to the signals. The threshold for the low-pass channel of the graph filter banks is set to $\lambda_{N/2}$.

Experimental results lead to the following conclusions:

1. **Performance Comparison:** Under the same number of Givens rotations, the denoising effect of the truncated Jacobi algorithm is generally better than that of the parallel truncated Jacobi algorithm. This is because each iteration of the truncated Jacobi algorithm obtains a locally optimal solution, while the parallel version does not, resulting in slower convergence.

2. **Frequency vs. Vertex Domain:** For frequency-domain sampling graph filter banks, as the number of iterations increases, the approximate eigenmatrix obtained by both algorithms continuously approaches the exact eigenmatrix. Graph signals and noise signals are effectively separated in the frequency domain. The denoising effect of frequency-domain sampling graph filter banks [11] is superior to that of vertex-domain sampling graph filter banks [8-9].

3. **Low Iteration Performance:** At low iteration counts, the denoising performance of truncated and parallel truncated Jacobi algorithms is better than vertex-domain sampling graph filter banks and close to traditional frequency-domain critically sampled graph filter banks.

[FIGURE 1] shows the signal-to-noise ratio (SNR) of denoised graph signals under different algorithms for three different graph topologies. As the number of Givens rotations increases, the SNR improves accordingly.

6.2 Image Denoising

Images can be modeled as graphs where pixels correspond to graph nodes and pixel values correspond to node signals. Each pixel connects to its four neighboring pixels (up, down, left, right), with edge weights determined by adjacent node pixel values [20]. The 4-neighborhood graph representation is shown in [FIGURE 2].

Experiments use the coin image shown in [FIGURE 3]. Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) serve as performance

metrics for image denoising evaluation. [TABLE 1] presents PSNR and SSIM values before and after denoising by different algorithms.

Key Findings: - At low noise levels, the improved Jacobi algorithm achieves better denoising performance - As Givens rotations increase, the denoising effect of the improved Jacobi algorithm approaches that of reference [11] - Some graph filter banks [8-9] perform poorly because high-frequency components removed during filtering contain not only noise but also image edge information, destroying original image content while failing to effectively process true noise information

[FIGURE 3] shows the coin image before and after denoising by different algorithms. [TABLE 1] summarizes the quantitative results.

7. Conclusion

Eigenvalue decomposition of the Laplacian matrix constitutes the most computationally expensive operation in frequency-domain critically sampled graph filter banks. When graph scale is large, computing the exact eigenmatrix becomes prohibitively expensive. This paper employs improved Jacobi algorithms —truncated Jacobi and parallel truncated Jacobi—to approximately solve for the eigenmatrix in frequency-domain critically sampled graph filter banks, achieving computational complexity reduction while guaranteeing perfect reconstruction.

Simulation results demonstrate that at low iteration counts, the denoising performance of truncated and parallel truncated Jacobi algorithms surpasses vertex-domain sampling graph filter banks and approaches that of traditional frequency-domain critically sampled graph filter banks. Future work will focus on optimizing the rotation coordinate selection strategy to further enhance algorithm efficiency.

References

[1] MICHELLE G, NEWMAN E. Community structure in social and biological networks[J]. Proceedings of the National Academy of Sciences of the United States of America, 2002, 99(12): 7812-7826.

[2] SANKARASUBRAMANIAM Y, AKYILDIZ I. Wireless sensor networks[J]. IEEE Network Magazine, 2002, 40(8): 102-114.

[3] STAM C. Graph theoretical analysis of brain networks[J]. Nonlinear Biomedical Physics, 2007, 1(3): 1-19.

[4] SHUMAN D, NARANG S, FROSSARD P, et al. Signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains[J]. IEEE Signal Processing Magazine, 2013, 30(3): 83-98.

[5] TEKE O, VAIDYANATHAN P. Extending multirate signal processing theory to graphs, part I: Fundamentals[J]. IEEE Transactions on Signal Processing,

2017, 65(2): 409-422.

[6] TEKE O, VAIDYANATHAN P. Extending multirate signal processing theory to graphs, part II: M-channel filter banks[J]. IEEE Transactions on Signal Processing, 2017, 65(2): 423-437.

[7] NARANG S, ORTEGA A. Perfect reconstruction two-channel wavelet filter banks for graph structured data[J]. IEEE Transactions on Signal Processing, 2012, 60(6): 2786-2799.

[8] NARANG S, ORTEGA A. Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs[J]. IEEE Transactions on Signal Processing, 2013, 61(19): 4673-4685.

[9] EKAMBARAM V, FANTI G, AYAZIFAR B, et al. Spline-like wavelet filterbanks for multiresolution analysis of graph-structured data[J]. IEEE Transactions on Signal Processing, 2015, 1(4): 258-278.

[10] SAKIYAMA A, WATANABE K, TANAKA Y. Two-channel critically-sampled graph filter banks with spectral domain sampling[J]. IEEE Transactions on Signal Processing, 2019, 67(6): 1447-1460.

[11] TANAKA Y. Spectral domain sampling of graph signals[J]. IEEE Transactions on Signal Processing, 2018, 66(14): 3752-3767.

[12] COHEN A, DAUBECHIES I, FEAUVEAU J. Biorthogonal bases of compactly supported wavelets[J]. Communications on Pure and Applied Mathematics, 1992, 45(5): 485-560.

[13] MAGOAROU L, GRIBONVAL R, TREMBLAY N. Approximate fast graph Fourier transforms via multi-layer sparse approximations[J]. IEEE Transactions on Signal and Information Processing over Networks, 2017, 4(2): 407-420.

[14] JACOBI C. Uber ein leichtes verfahren, die in der theorie der sakularstorungen vorkommenden gleichungen numerisch aufzulosen[J]. Journal fur die Reine und Angewandte Mathematik, 1846, 1846(30): 51-94.

[15] GIVENS W. Computation of plane unitary rotations transforming a general matrix to triangular form[J]. Journal of the Society for Industrial and Applied Mathematics, 1958, 6(1): 26-50.

[16] GOLUB G, VANLOAN C. Matrix Computations[M]. Washington D.C: Johns Hopkins University Press, 2012.

[17] PERRAUDIN N, SHUMAN D, PARATTE J, et al. GSPBOX: A toolbox for signal processing on graphs[EB/OL]. https://arxiv.org/pdf/1408.5781v1.pdf, 2014.

[18] NARANG S, ORTEGA A, CHAO H. Graph-wavelet filterbanks for edge-aware image processing[C]//2012 IEEE Statistical Signal Processing Workshop. Piscataway, NJ: IEEE Press, 2012: 141-144.

[19] SHEKKIZHAR S, ORTEGA A. Efficient graph construction for image representation[EB/OL]. https://arxiv.org/pdf/2002.06662.pdf, 2020.

Editor: Zhang Suobin

Note: Figure translations are in progress. See original paper for figures.