CHAPTER 5
EIGENVECTOR COMPUTING ALGORITHMS

In Sec. 4.2, we learned that the real time calculation of the gain matrix that transforms sensor outputs into modal coordinates involves cross multiplying the segment outputs, averaging the resulting matrix, and computing the eigenvector matrix of the averaged matrix. The eigenvector matrix computation in section 4.3 was done in batches. In practice, computation of eigenvector matrices may be so extensive that it holds up other operations. A more efficient eigenvector computation strategy would involve recursive computation concurrent with other operations.

In this chapter we will discuss a method to compute eigenvector matrices, also called modal matrices, recursively using a version of Jacobi rotation technique. We will present a brief review of the Jacobi rotation algorithm including numerical examples, and develop special algorithms for solving our eigenvector problem.

5.1 Jacobi Rotation Algorithm

Jacobi rotation, also known as Givens rotation, is a very popular eigenvector computing algorithm in digital signal processing because it is simple and stable in general. It is even simpler and more robust in processing symmetric real matrices (Haykin, 1991). It has also been used extensively for computing eigenvectors in structural dynamic finite element problems (for example, Bradbary and Fletcher, 1968).

In Sec. 4.3, we showed that the matrix that transforms segment outputs into modal coordinates is the matrix that diagonalizes the segment output correlation matrix. Most square matrices can be diagonalized by similarity transformation. If $\mathbf{S}$ is a diagonalizable matrix, then its eigenvector matrix $\mathbf{\Psi}$ is a transformation matrix such that the result of similarity transformation

$$\mathbf{Q} = \mathbf{\Psi}^{\mathrm{T}}\mathbf{S}\mathbf{\Psi} \tag{5.1}$$

is diagonal. Jacobi rotation is a method to compute the transformation matrix, $\mathbf{\Psi}$. Basically, this method works by transforming the matrix to another matrix that has a zero component in selected positions. The derivation of this method can be found in many linear algebra or numerical analysis textbooks such as Press et al. (1986). Without derivation, the procedure can be described as follows.

First, initialize $\mathbf{Q}(1) = \mathbf{S}$, the matrix to be diagonalized. Also initialize the matrix $\mathbf{\Psi}$,

$$\boldsymbol{\Psi}(1) = \mathbf{I}\,. \tag{5.2}$$

1. At time step h, choose an off-diagonal element of matrix $\mathbf{Q}(h)$ to be set to zero. Call this element $\mathbf{Q}_{pq}(h)$, where p is the row position of the element and q is the column position. In the *cyclic* version of Jacobi rotation algorithms, all the off-diagonal elements in the upper triangular half of the matrix are chosen systematically row by row. For example, the (1,2) element is chosen for the first loop. Then the (1,3) element is chosen, and so on. The last loop chooses the (N-1,N) element of the N-by-N matrix. This systematic selection process is called *sweeping*.

2. Compute the rotation angle

$$\mathsf{q}(h) = \frac{1}{2}\tan^{-1}\left(\frac{2\,\mathbf{Q}_{pq}(h)}{\mathbf{Q}_{qq}(h) - \mathbf{Q}_{pp}(h)}\right) \text{ if } \mathbf{Q}_{qq}(h) \neq \mathbf{Q}_{qq}(h)\,, \tag{5.3}$$

   Or $\mathsf{q}\,(h) = 90^0$ if $\mathbf{Q}_{qq}\,(h) = \mathbf{Q}_{pp}(h)$.

3. Assemble the Jacobi rotation matrix, which is like an identity matrix except for the elements indicated below:

$$\mathbf{P}(h) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos(\mathsf{q}(h)) & & -\sin(\mathsf{q}(h)) & & \\ & & & 1 & & & \\ & & \sin(\mathsf{q}(h)) & & \cos(\mathsf{q}(h)) & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \begin{matrix} \\ \\ \leftarrow p^{\text{th}}\text{row} \\ \\ \leftarrow q^{\text{th}}\text{row} \\ \\ \\ \end{matrix}$$
$$\begin{matrix} & & \uparrow & & \uparrow & & \\ & & p^{\text{th}}\text{column} & & q^{\text{th}}\text{column} & & \end{matrix} \tag{5.4}$$

4. Apply similarity transformation to $\mathbf{Q}(h)$ according to

$$\mathbf{Q}(h+1) = \mathbf{P}(h)^{\mathrm{T}}\,\mathbf{Q}(h)\mathbf{P}(h) \tag{5.5}$$

5. Update the estimate for the modal matrix, $\boldsymbol{\Psi}(h)$, by post-multiplying it by the Jacobi rotation matrix $\mathbf{P}(h)$.

$$\boldsymbol{\Psi}(h+1) = \boldsymbol{\Psi}(h)\mathbf{P}(h)\,. \tag{5.6}$$

6. Repeat steps 2 through 5 until a suitable diagonality criterion is satisfied by **Q**.

Notice that step 5 means that the eigenvector matrix is the product of all the Jacobi rotation matrices obtained at each iteration, that is,

$$\mathbf{\Psi} = \mathbf{P}(1)\mathbf{P}(2)\mathbf{P}(3)\cdots \tag{5.7}$$

Step 4 in each iteration sets the (p,q) element of matrix **Q** to zero. In the next iteration, where the indices p and q are changed, the element that was set to zero in the previous iteration will again be nonzero. Nevertheless, the matrix **Q** will become progressively more diagonal with each iteration. From the analytic geometry point of view, this transformation is a rotational coordinate transformation in a plane, hence the name Jacobi rotation.

To present a numerical example, we would like to obtain the gain matrix that will transform the following matrix into a diagonal matrix.

$$\mathbf{S} = \mathbf{Q}(1) = \begin{bmatrix} 3.8 & 2.1 & -0.45 & 2 \\ 2.1 & 4.5 & -0.54 & 1 \\ -0.45 & -0.54 & 2.3 & -0.3 \\ 2 & 1 & -0.3 & -2.6 \end{bmatrix}$$

The first off-diagonal element to annihilate is the (1,2) element of **Q**, that is, 2.1. For this element, Eq. (5.3) gives $\mathsf{q}(1) = -0.703$; and the Jacobi rotation matrix according to Eq. (5.4) is

$$\mathbf{P}(1) = \begin{bmatrix} 0.763 & 0.646 & 0 & 0 \\ -0.646 & 0.763 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The first iteration transforms the matrix into

$$\mathbf{Q}(2) = \mathbf{P}(1)^{\mathrm{T}}\mathbf{Q}(1)\mathbf{P}(1) = \begin{bmatrix} 2 & 0 & 0.006 & 0.88 \\ 0 & 6 & -0.7 & 2.06 \\ 0.006 & -0.7 & 2.3 & -0.30 \\ 0.88 & 2.06 & -0.30 & 2.6 \end{bmatrix}$$

This transformation annihilated the (1,2) element of **Q**. The next transformation annihilates the (1,3) element, resulting in

$$\mathbf{Q}(3) = \mathbf{P}(2)^{\mathrm{T}}\mathbf{Q}(2)\mathbf{P}(2) = \begin{bmatrix} 2 & 0.01 & 0 & 0.89 \\ 0.01 & 6.3 & -0.7 & 2.06 \\ 0 & -0.7 & 2.3 & -0.30 \\ 0.89 & 2.06 & -0.30 & 2.6 \end{bmatrix}$$

This transformation annihilates the (1,3) element, but undid the annihilation of the (1,2) element of the previous transformation. However, the resulting matrix is relatively "more diagonal" than the previous matrix. The original matrix and the results of the first five transformations are illustrated in Fig. 5.1. We can see that the matrix becomes progressively more diagonal with each transformation.

The above 4-by-4 matrix has 6 off-diagonal elements to annihilate ((*p*,*q*) = (1,2), (1,3), (1,4), (2,3), (2,4), and (3,4)). After one sweep (which means six rotations to annihilate the six upper off-diagonal elements), the matrix is almost diagonal, as shown in Fig. 5.2.

The above sweep can be repeated indefinitely. However, Fig. 5.3 shows that the matrix is practically diagonal. The second sweep result is

$$\mathbf{Q}_{(second\ sweep)} = \begin{bmatrix} 1.02 & 0.028 & 0.014 & 0 \\ 0.028 & 7.34 & 0 & 0 \\ 0.014 & 0 & 2.18 & 0 \\ 0 & 0. & 0 & 2.66 \end{bmatrix}$$

Further sweeps will diagonalize the matrix even further, and eventually only maintain the diagonality. The eigenvector matrix is obtained by cumulatively multiplying the Jacobi rotation matrices according to Eq. (5.7).

Notice that as the matrix resulting from rotations becomes more and more diagonal, the rotation angle in Eq. (5.3) becomes closer and closer to zero, and the Jacobi rotation matrix in Eq. (5.4) becomes closer and closer to identity. Eventually, multiplying the estimate of the eigenvector matrix with more Jacobi rotation matrices does not change the estimate significantly. At this stage the estimate has converged to the true eigenvector matrix.

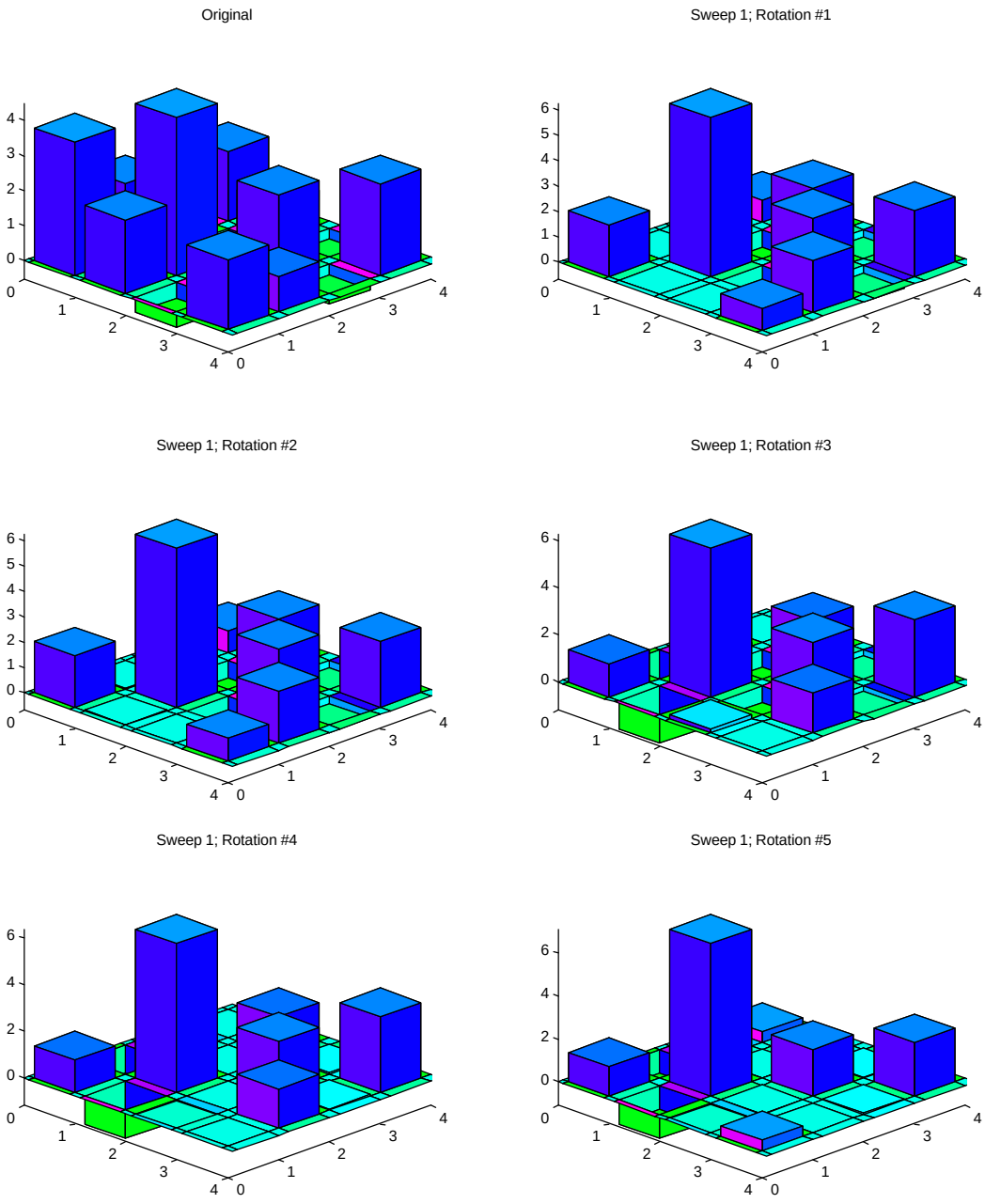

Figure 5.1 Jacobi rotation example.

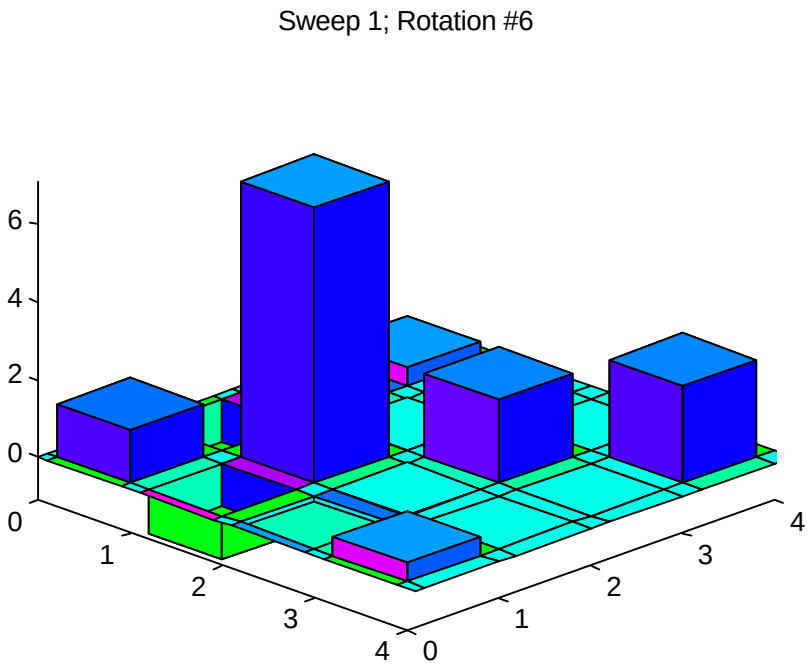

Figure 5.2 Result of first sweep.

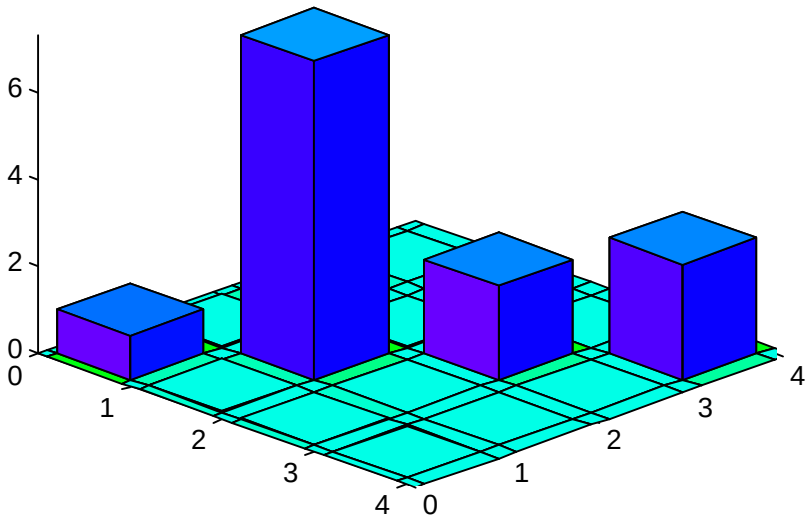

Figure 5.3 Result of second sweep.

The above example showed the effectiveness and simplicity of the Jacobi rotation algorithm. Equation (5.7) shows that cumulative multiplication of the Jacobi rotations improves the estimate of the modal matrix progressively. Equation (5.4) shows that multiplication with a Jacobi rotation matrix effectively changes only two rows and two columns of the previous estimate of the modal matrix. The transformed matrix in Eq. (5.4) becomes more and more diagonal with each rotation. The rotation angle θ becomes closer and closer to zero with each rotation. Therefore, the Jacobi rotation matrix **P** in Eq. (5.4) becomes closer and closer to identity. Eventually after sufficient rotations the rotation matrix will just oscillate around identity, effectively maintaining the diagonality of the rotated matrix **Q**.

5.2 Algorithm A, Convergence and Rotation Angle

The gain matrix calculation procedure shown in Fig. 4.7 can be done using any eigenvector computation algorithm. The modal matrix computation in the simulation examples in section 4.3 was done with the Jacobi rotation algorithm, sweeping the upper off-diagonal elements of the segment output correlation matrix **S** four times. For a ten-mode filter, each sweep annihilates 45 off-diagonal elements, with one Jacobi rotation for each element. For the purpose of discussion and comparison, we refer to this algorithm as *Algorithm A*.

The segment output correlation matrix is estimated by recursive averaging. The average $\mathbf{S}(k)$ is improved with each time step k. Each time a new average is obtained, its modal matrix is computed with Eqs. (5.2) through (5.6). Figure 5.4 shows the connection between the segments and the signal processing.

The rotation angle θ in Eq. (5.3) is a good indication of how much the modal matrix estimate will change with each rotation. A small rotation angle means that the Jacobi rotation matrix in Eq. (5.4) is close to the identity matrix, therefore, a small rotation angle does not change the modal matrix estimate significantly. Figure 5.5 shows the first 8 time steps of the algorithm used in Sec. 4.3. Each time step includes four sweeps, which mean 4*45 = 180 Jacobi rotations. There is a strong tendency that within each time step the rotation angle becomes smaller. However, each new time step finds a new correlation matrix, discards the modal matrix, and starts forming a new modal matrix by cumulatively multiplying the Jacobi rotation matrix all over again. The initial guess is the same for each time step. Improvement of the modal matrix (and hence the sensor gain matrix) is achieved only because the estimate of the segment output correlation matrix **S** improves with time steps. In other words, $\mathbf{W}_k$ is better than $\mathbf{W}_{k-1}$ only if $\mathbf{S}_k$ is better than $\mathbf{S}_{k-1}$.

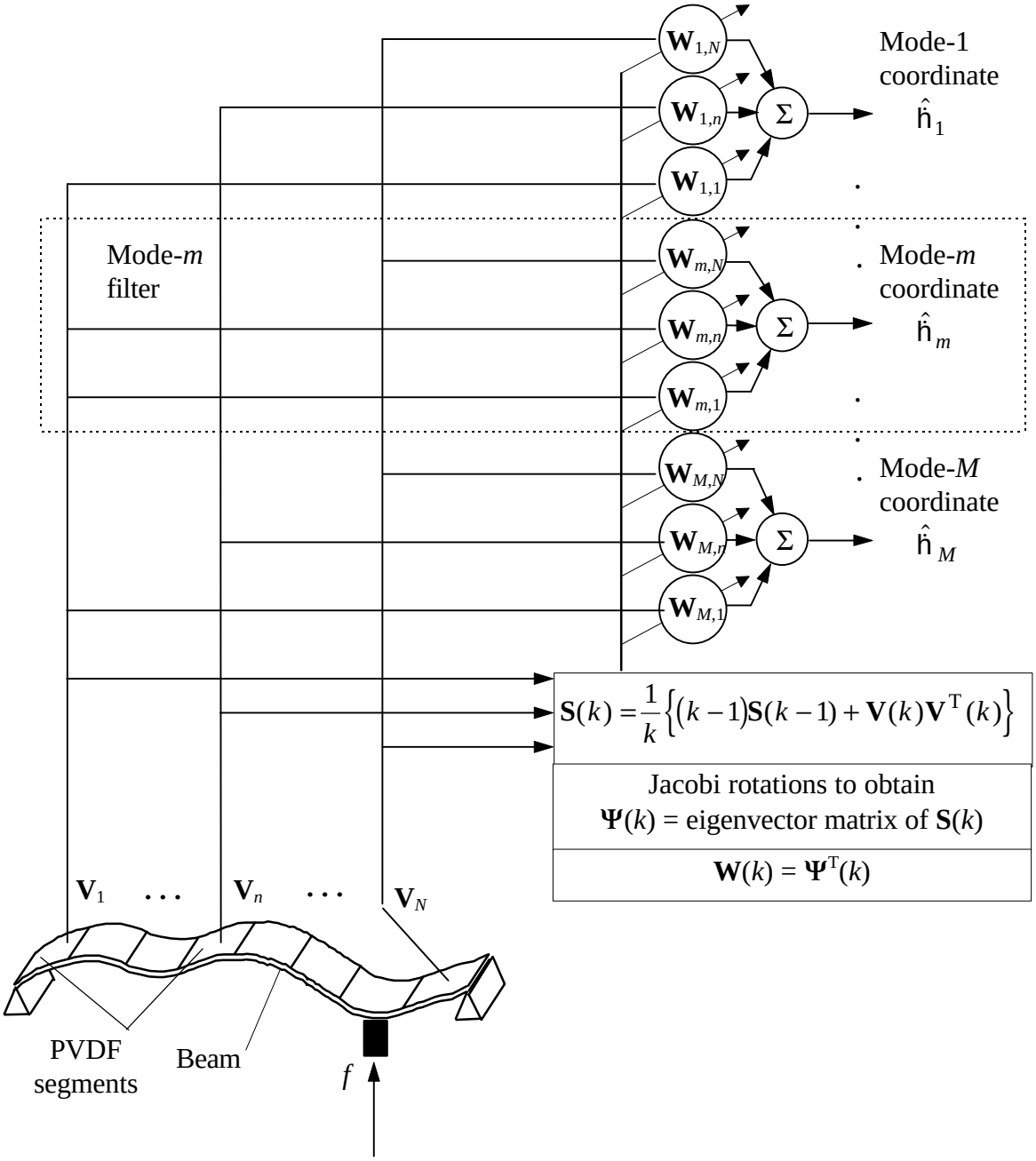

Figure 5.4 Algorithm A.

Jacobi rotations in Algorithm A are merely a tool to solve for the eigenvectors of the segment output correlation matrix **S** in each time step. For each **S**(*k*), the eigenvector estimate does not improve beyond the four sweeps of Jacobi rotations. Thus, the rotation angle does not necessarily become smaller with time step *k*. Because of the large number of operations per time step and the little improvement in the eigenvectors with time steps, Algorithm A is of little practical utility. We discussed it only to facilitate the understanding of the development of the two new algorithms that we will discuss in the next sections.

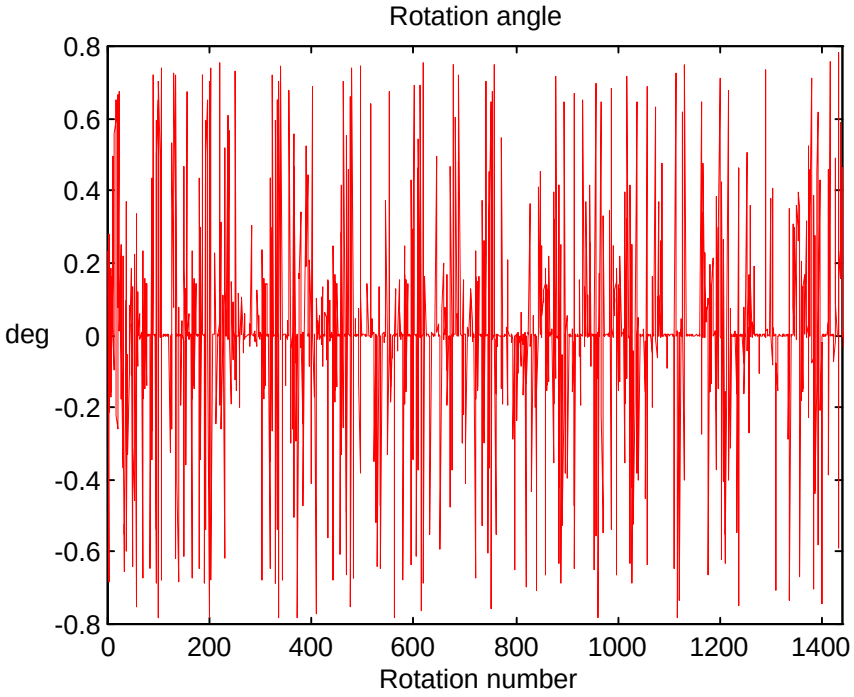

Figure 5.5 Typical rotation angle history of Algorithm A.

5.3 Algorithm B

5.3.1 Development

A much more economical method for computing the modal matrix will be devised below. The objective of the new algorithm is to retain the recent update for the modal matrix of **S** (the estimate for the segment output correlation matrix) and to track the slow changes in **S**. If the excitation force is stationary and if the structure does not change significantly with time, then after sufficient time steps **S** can be assumed to be constant. This assumption introduces error that will decrease with time. With this assumption, we can perform just one Jacobi rotation at each time

step instead of sweeping the entire upper off-diagonal elements. Instead of being discarded, the modal matrix is stored and improved gradually with each time step.

The selection of the element on which to perform the Jacobi rotation is done based on a variable which we will call the *coupling factor*. We define the coupling factor of the ij^{th} off-diagonal element of a real symmetric matrix **T** as

$$\gamma_{ij} = \frac{\mathbf{T}_{ij}^{\ 2}}{\mathbf{T}_{ii}\,\mathbf{T}_{jj}}. \tag{5.8}$$

The coupling factor can be used to determine which off-diagonal elements are to be annihilated. This helps prevent a Jacobi rotation from operating on an off-diagonal element that is already very close to zero. This strategy saves computational effort. In the *Threshold Jacobi method* (Bathe and Wilson, 1976), the element is annihilated only if the coupling factor is larger than a prescribed threshold. This method saves computational time if the eigenvector computation is implemented with a computer or a programmable digital signal processing board. Another method is to annihilate the element that has the largest coupling coefficient. In this section we adopt the latter method. With the above ideas, we develop the following algorithm, which we refer to as *Algorithm B*.

Before starting the algorithm, initialize a variable matrix

$$\mathbf{T}(0) = \mathbf{V}(0)\mathbf{V}^{\mathrm{T}}(0) \tag{5.9}$$

This matrix will later be the result of rotation of **S**.

Also initialize the sensor gain matrix **W**(0) to an initial guess. In the absence of such a guess, initialize **W**(0) to identity.

1. Compute the coupling factors of the upper off-diagonal elements of **T**(k), using

$$\gamma_{ij} = \frac{\mathbf{T}_{ij}^{\ 2}}{\mathbf{T}_{ii}\,\mathbf{T}_{jj}} \tag{5.10}$$

2. Find the indices (p,q) of the element with the largest coupling factor.

3. Calculate the rotation angle $\theta(k)$ using

$$\theta(k)=\frac{1}{2}\arctan\left(\frac{2\mathbf{T}_{pq}(k)}{\mathbf{T}_{pp}(k)-\mathbf{T}_{qq}(k)}\right) \text{ if } \mathbf{T}_{pp}(k)-\mathbf{T}_{qq}(k)\neq 0, \tag{5.11a}$$

or

$$\theta(k)=\frac{\pi}{4} \text{ if } \mathbf{T}_{pp}(k)-\mathbf{T}_{qq}(k)=0. \tag{5.11b}$$

4. Assemble a Jacobi rotation matrix

$$\mathbf{P}(k)=\begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos(\theta) & & -\sin(\theta) & & \\ & & & 1 & & & \\ & & \sin(\theta) & & \cos(\theta) & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \begin{matrix} \\ \\ \leftarrow p^{\text{th}}\text{row} \\ \\ \leftarrow q^{\text{th}}\text{row} \\ \\ \\ \end{matrix} \tag{5.12}$$

$$\begin{matrix} & & \uparrow & & \uparrow & & \\ & & p^{\text{th}}\text{column} & & q^{\text{th}}\text{column} & & \end{matrix}$$

5. Update the modal matrix according to Eq. (5.7).

$$\mathbf{\Psi}(k)=\mathbf{\Psi}(k-1)\mathbf{P}(k) \tag{5.13}$$

6. Estimate the segment output correlation matrix **S** by the recursive averaging process

$$\mathbf{S}(k)=\frac{1}{k}\left\{(k-1)\mathbf{S}(k-1)+\mathbf{V}(k)\mathbf{V}^{\mathrm{T}}(k)\right\} \tag{5.14}$$

7. Perform similarity transformation on $\mathbf{S}(k)$ with $\mathbf{\Psi}(k)$, store the result as $\mathbf{T}(k)$

$$\mathbf{T}(k)=\mathbf{\Psi}^{\mathrm{T}}(k)\mathbf{S}(k)\mathbf{\Psi}(k) \tag{5.15}$$

8. Obtain the sensor gain matrix **W** by transposing the modal matrix $\mathbf{\Psi}$

$$\mathbf{W}(k)=\mathbf{\Psi}^{\mathrm{T}}(k) \tag{5.16}$$

9. Repeat steps 1 through 8.

The repetition of this algorithm can be terminated after a chosen convergence criterion is satisfied. However, we can also run this algorithm indefinitely, especially if we implement it with a dedicated digital electronic circuit, as we intend to do in the future. In this case, if the structure's modal parameters change, the computation of the sensor gain matrix can track the changes. Therefore, the sensor gain matrix will always decorrelate the sensor outputs.

The intermediate matrix **T** in step 7 is a unique feature of the new algorithm. To understand the logic behind this algorithm, keep in mind that it is just a Jacobi rotation algorithm to solve for the modal matrix of the segment output correlation matrix **S**, as done in Eq. (4.10). Equation (5.15) shows that **T** is the estimate of the diagonal eigenvalue matrix of **S**. Substitution of Eq. (5.13) into Eq. (5.15) results in

$$\mathbf{T}(k)=\mathbf{P}^{\mathrm{T}}(k)\boldsymbol{\Psi}^{\mathrm{T}}(k-1)\mathbf{S}(k)\boldsymbol{\Psi}(k-1)\mathbf{P}(k). \tag{5.17}$$

If **S** is quasi-constant, as usually the case after enough time steps, then we can assume that

$$\mathbf{S}(k)\approx\mathbf{S}(k-1). \tag{5.18}$$

Under this assumption, Eq. (5.17) can be written as

$$\mathbf{T}(k)=\mathbf{P}^{\mathrm{T}}(k)\ \boldsymbol{\Psi}^{\mathrm{T}}(k-1)\mathbf{S}(k-1)\boldsymbol{\Psi}(k-1)\ \mathbf{P}(k). \tag{5.19}$$

The three middle terms on the right-hand side of the last equation can be simplified by applying Eq. (5.15)

$$\mathbf{T}(k-1)=\boldsymbol{\Psi}^{\mathrm{T}}(k-1)\mathbf{S}(k-1)\boldsymbol{\Psi}(k-1) \tag{5.20}$$

Thus, Eq. (5.19) becomes

$$\mathbf{T}(k)=\mathbf{P}^{\mathrm{T}}(k)\mathbf{T}(k-1)\mathbf{P}(k). \tag{5.21}$$

According to Eq. (5.5), the last equation is simply a Jacobi rotation to bring the matrix **T** closer to a diagonal matrix.

Equation (5.18) is a key assumption in the derivation of the algorithm because it effectively enables us to obtain the modal matrix of a slowly varying matrix. Consequently, this algorithm can

track changes in the modal parameters of the structure, since this algorithm uses the most recent updates of the segment output correlation matrix **S**.

Algorithm B is much faster than Algorithm A. In section 4.3, Algorithm A needs to sweep the upper off-diagonal elements of **S** four times at each time step to achieve a modal matrix $\boldsymbol{\Psi}$ that will transform **S** into a reasonably diagonal matrix. For a ten-mode sensor, there are 45 upper non-diagonal elements to annihilate with a Jacobi rotation. Four sweeps per time step involve 4*45 = 180 Jacobi rotations per time step. On the other hand, Algorithm B only does one rotation per time step.

Algorithm B also results in better sensor gain matrices than algorithm A. Equation (5.15) shows that Algorithm B keeps a matrix **T** that is stored and rotated towards a diagonal form at each time step. The estimate for $\boldsymbol{\Psi}$ (the modal matrix of **S**) is the cumulative product of the rotation matrices. This means that the initial guess for the modal matrix at each time step is the modal matrix at the previous time step. Therefore, the modal matrix improves with each time step. In Algorithm A, on the other hand, the initial guess for the modal matrix at each time step is the identity matrix as in Eq. (5.2).

5.3.2 Numerical Example

Algorithm B results in fast convergence with the number of rotations. After only 256 time steps, one rotation at each time step, the correlation matrix of the sensor output is already close to a diagonal matrix (See Fig. 5.6). After 32768 time steps, the sensor output correlation matrix, shown in Fig. 5.7, is practically diagonal.

The time history of the rotation angle, which is an indication of the convergence of the algorithm, is shown in Figure 5.8. This figure has been decimated by a factor of 8 for clarity. After the rotation angle has converged around zero, some ‘spikes’ appear occasionally. The largest rotations (close to $+ 45^{o}$ or $- 45^{o}$) are done because the random excitation causes the denominator of Eqs. (5.11) to be zero. However, these rotations are usually followed by rotations in the opposite direction immediately. Therefore, the modal matrix is quickly restored to its steady-state value.

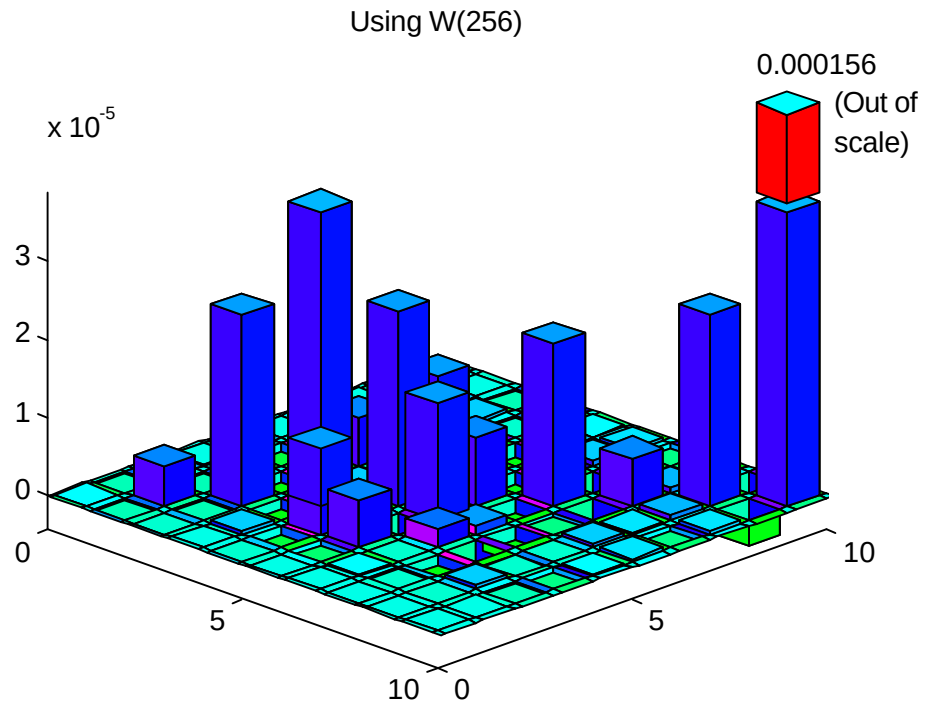

Figure 5.6 Sensor output correlation matrix (Algorithm B) after 256 time steps.

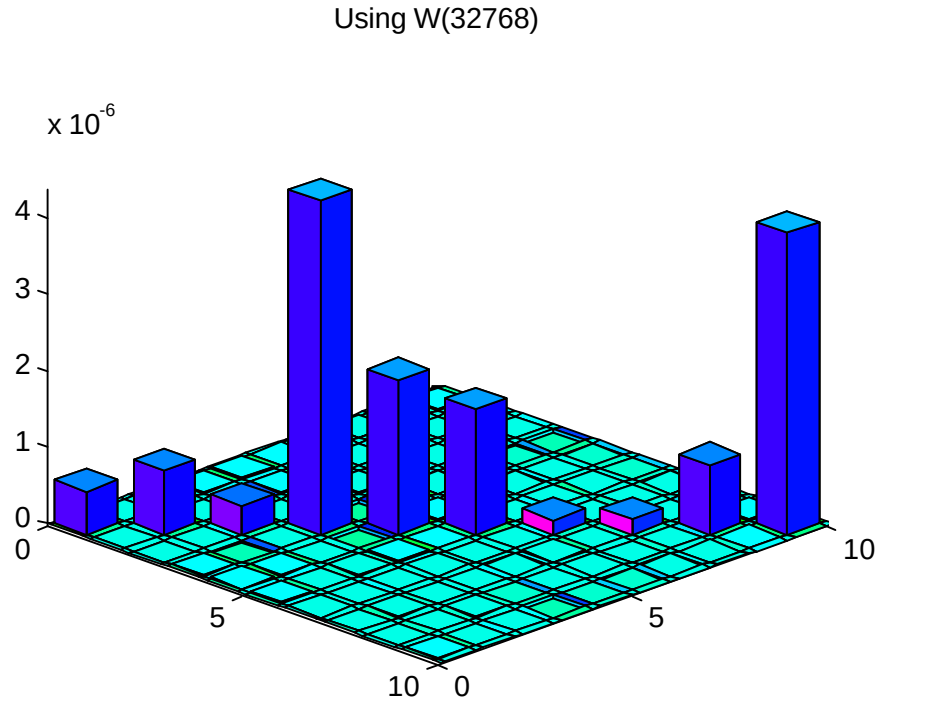

Figure 5.7 Sensor output correlation matrix (Algorithm B) after 32768 time steps.

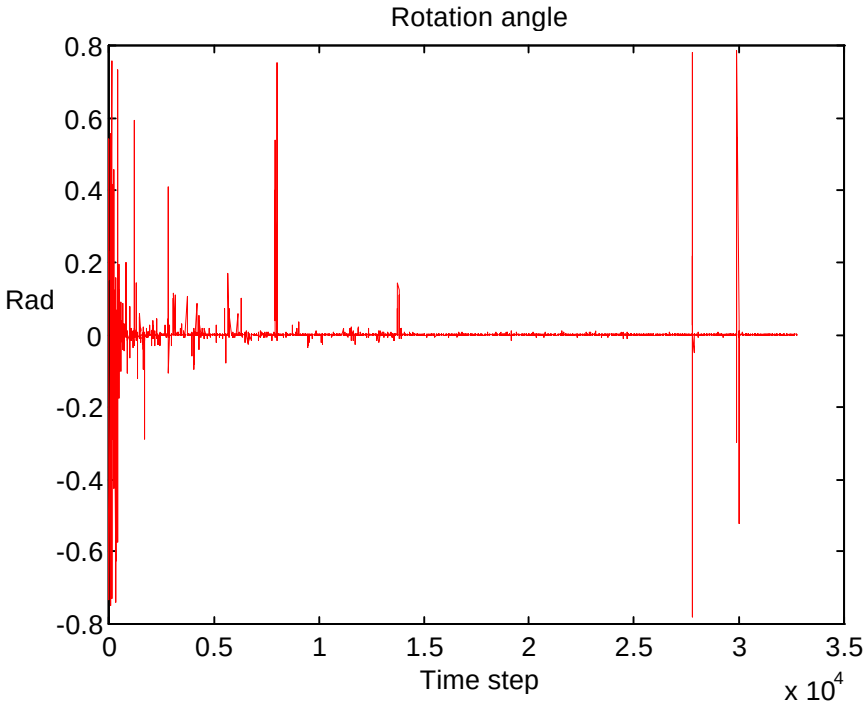

Figure 5.8 Rotation angle history, Algorithm B.

The sensor gain matrix resulting from Algorithm B with 32768 time steps is shown in Fig. 5.9. The magnitudes of the FRF's from force to sensor outputs are shown in Fig. 5.10.

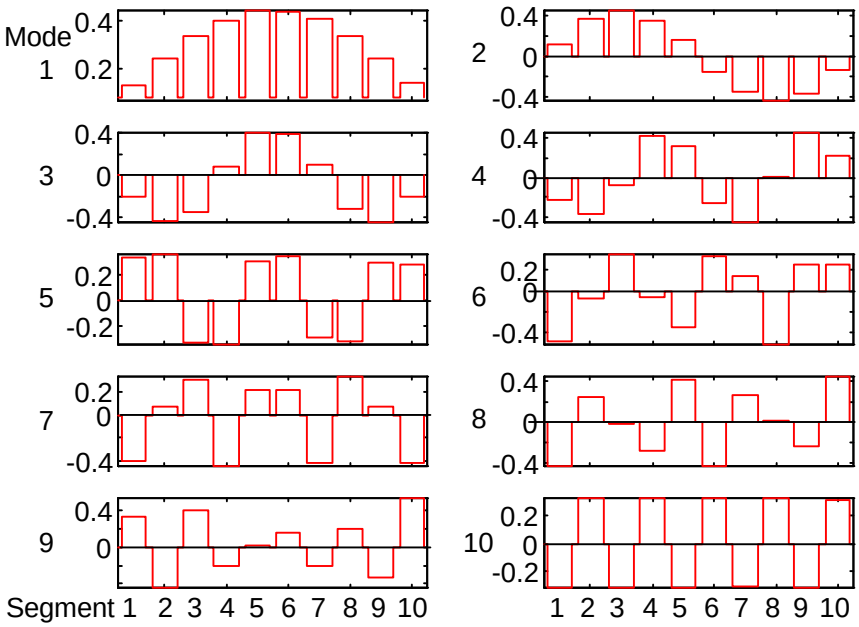

Figure 5.9 Sensor gain matrix (Algorithm B), 32768 time steps.

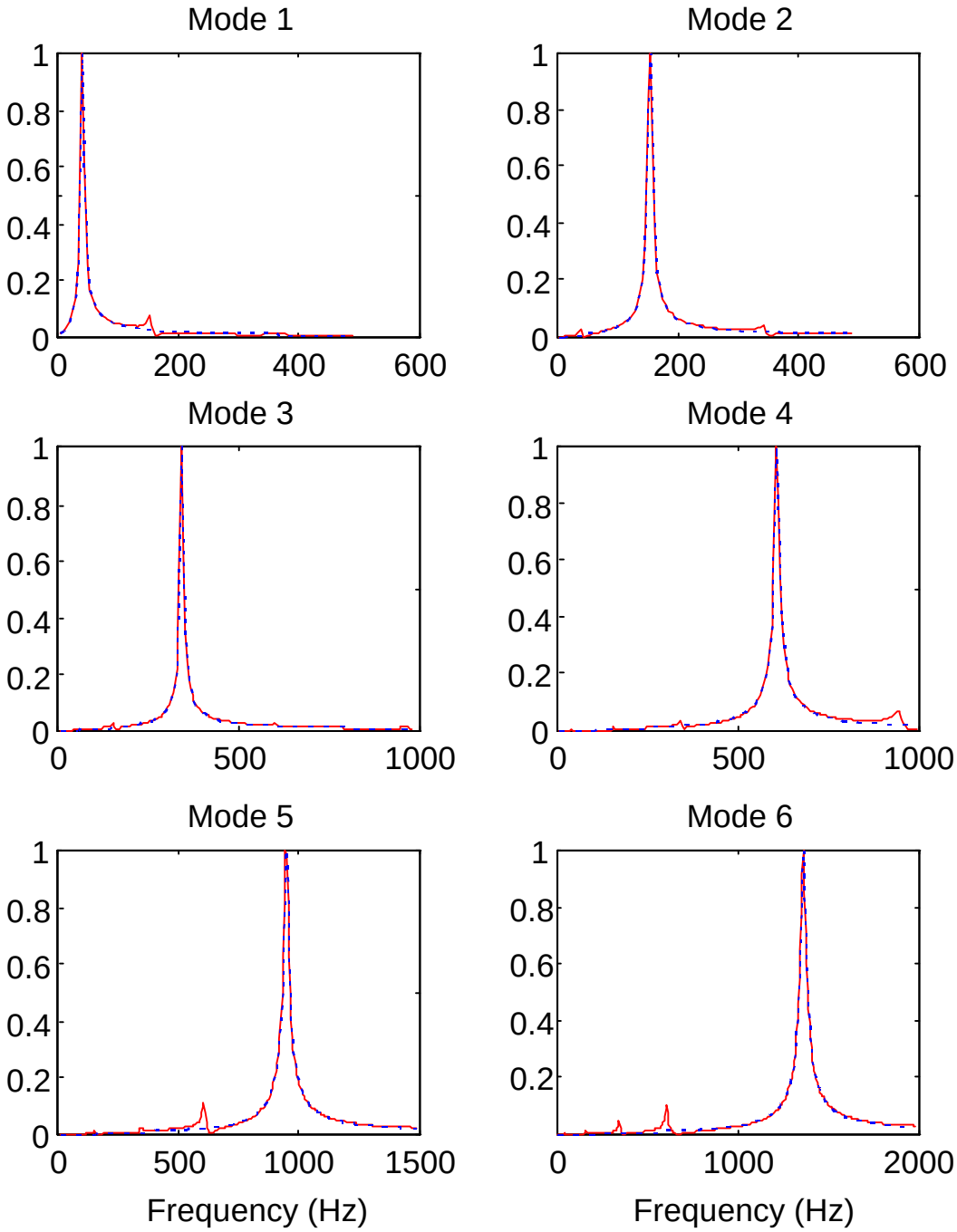

Figure 5.10 Normalized magnitude responses of modal filter (Algorithm B) after 32768 time steps.

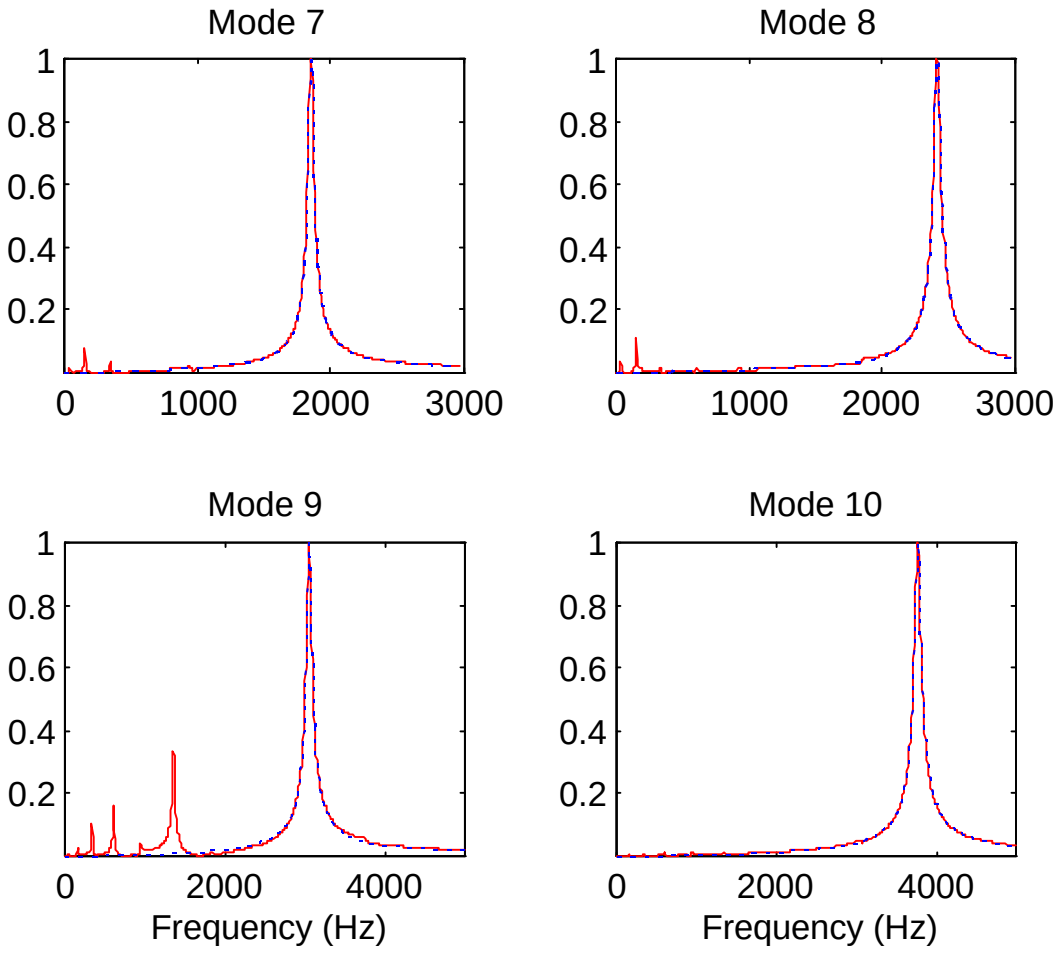

Figure 5.10 Normalized magnitude responses of modal filter (Algorithm B) after 32768 time steps.

5.4 Algorithm C

5.4.1 Development

The above reasoning and numerical example show that Algorithm B is very efficient in computing the gain matrix **W** of the adaptive modal sensor. There is, however, another fact to be considered in the adaptive computation of the gain matrix. For the outputs of the adaptive modal filter to be proportional to modal coordinates it is necessary that these outputs be orthogonal to each other. The discussion in section 4.2 verifies that this requirement is satisfied by the gain matrix obtained

by Algorithm B, since the gain matrix is the transpose of the modal matrix of **S**. However, we can exploit this orthogonality requirement to refine the modal matrix computing scheme as follows. As long as any off-diagonal element of the sensor output correlation matrix **R** is not zero, coupling still exists between the outputs of the sensor. Therefore, the sensor gain matrix **W** should be modified to eliminate this coupling. Thus, checking the diagonality of the output correlation matrix **R** provides a sort of 'performance feedback' to the gain matrix computation scheme. We will develop a new algorithm to incorporate performance feedback, which was called for in our discussion in Subsection 4.2.3.

To understand the intuitive derivation of the algorithm, imagine that we could rotate the output correlation matrix **R** with a Jacobi rotation matrix **P** to annihilate an off-diagonal nonzero p,q^{th} element $\mathbf{R}_{pq}$. The matrix **P** that would do this task can be obtained using Eqs. (5.11) and (5.12). In reality, we cannot rotate the output correlation matrix **R** directly. However, we can create the same effect by modifying the sensor gain matrix **W**. Recall that when **W** is quasi-constant and close to the ideal gain matrix, Eqs. (4.8) and (4.9) can be combined into

$$\mathbf{R} \approx \mathbf{W}\,\mathbf{S}\,\mathbf{W}^{\mathrm{T}} \tag{5.22}$$

and, since $\mathbf{W} = \mathbf{\Psi}^{\mathrm{T}}$,

$$\mathbf{R} \approx \mathbf{\Psi}^{\mathrm{T}}\mathbf{S}\,\mathbf{\Psi} \tag{5.23}$$

The last equation points us to Eq. (5.15) of Algorithm B, which has been proven successful in computing the correct modal matrix. In particular, the similarity of Eq. (5.23) to Eq. (5.15) shows that we can now use the matrix **T** in place or **R** to create a Jacobi rotation matrix. However, this time we take advantage of the performance feedback by aiming to annihilate the p,q^{th} element of **R** instead of **T** as in Algorithm B. The resulting algorithm, referred to as *Algorithm C*, is listed below.

Before starting the algorithm, initialize a variable matrix

$$\mathbf{T}(0) = \mathbf{V}(0)\mathbf{V}^{\mathrm{T}}(0) \tag{5.24}$$

This matrix will later be the result of rotation of **S**.

Also initialize the sensor gain matrix **W**(0) to an initial guess. In the absence of such a guess, initialize **W**(0) to identity.

1. Estimate the sensor output correlation matrix **R** recursively by

$$\mathbf{R}(k) = \frac{1}{k}\left\{(k-1)\mathbf{R}(k-1) + \hat{\mathsf{h}}(k)\hat{\mathsf{h}}^{\mathrm{T}}(k)\right\} \tag{5.25}$$

2. Compute the coupling factors of the upper off-diagonal elements of **R**(*k*), using

$$\mathsf{g}_{ij} = \frac{\mathbf{R}_{ij}^{\,2}}{\mathbf{R}_{ii}\,\mathbf{R}_{jj}} \tag{5.26}$$

3. Find the indices (*p*,*q*) of the element with the largest coupling factor.

4. Calculate the rotation angle $\mathsf{q}(k)$ using

$$\mathsf{q}(k) = \frac{1}{2}\arctan\left(\frac{2\,\mathbf{T}_{pq}(k)}{\mathbf{T}_{pp}(k) - \mathbf{T}_{qq}(k)}\right) \text{ if } \mathbf{T}_{pp}(k) - \mathbf{T}_{qq}(k) \neq 0, \tag{5.27a}$$

or

$$\mathsf{q}(k) = \frac{\mathsf{p}}{4} \text{ if } \mathbf{T}_{pp}(k) - \mathbf{T}_{qq}(k) = 0. \tag{5.27b}$$

5. Assemble the Jacobi rotation matrix

$$\mathbf{P}(k) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos(\mathsf{q}) & & -\sin(\mathsf{q}) & & \\ & & & 1 & & & \\ & & \sin(\mathsf{q}) & & \cos(\mathsf{q}) & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \begin{matrix} \\ \\ \leftarrow p^{\text{th}}\text{row} \\ \\ \leftarrow q^{\text{th}}\text{row} \\ \\ \\ \end{matrix}$$

$$\begin{matrix} & & \uparrow & & \uparrow & & \\ & & p^{\text{th}}\text{column} & & q^{\text{th}}\text{column} & & \end{matrix} \tag{5.28}$$

6. Update the modal matrix according to Eq. (5.7).

$$\mathbf{\Psi}(k) = \mathbf{\Psi}(k-1)\mathbf{P}(k) \tag{5.29}$$

7. Estimate the segment output correlation matrix **S** by averaging recursively

$$\mathbf{S}(k) = \frac{1}{k}\left\{(k-1)\mathbf{S}(k-1) + \mathbf{V}(k)\mathbf{V}^{\mathrm{T}}(k)\right\} \tag{5.30}$$

8. Perform similarity transformation on **S**(k) with **Ψ**(k), store the result as **T**(k)

$$\mathbf{T}(k) = \mathbf{\Psi}^{\mathrm{T}}(k)\mathbf{S}(k)\mathbf{\Psi}(k) \tag{5.31}$$

9. The sensor gain matrix **W** is the transpose of the modal matrix **Ψ**, as mentioned in Eq. (4.12). The sensor output is the segment output multiplied by the sensor gain matrix. Therefore,

$$\hat{\mathbf{n}}(k) = \mathbf{W}(k)\mathbf{V}(k) = \mathbf{\Psi}^{\mathrm{T}}(k)\mathbf{V}(k) \tag{5.31}$$

10. Repeat steps 1 through 9.

The repetition of this algorithm can be terminated after a chosen convergence criterion is satisfied. However, we can also run this algorithm indefinitely, especially if we implement it with a dedicated digital electronic circuit. In this case, if the structure's modal parameters change, the computation of the sensor gain matrix can track the changes. Therefore, the sensor gain matrix will always decorrelate the sensor outputs.

Figure 5.11 shows the connections between the segments and Algorithm C. This algorithm is more sophisticated than Algorithm B, particularly because of the performance feedback. Equation (5.25) constitute extra computation over algorithm B. This is the cost of performance feedback which does not exist in Algorithm B.

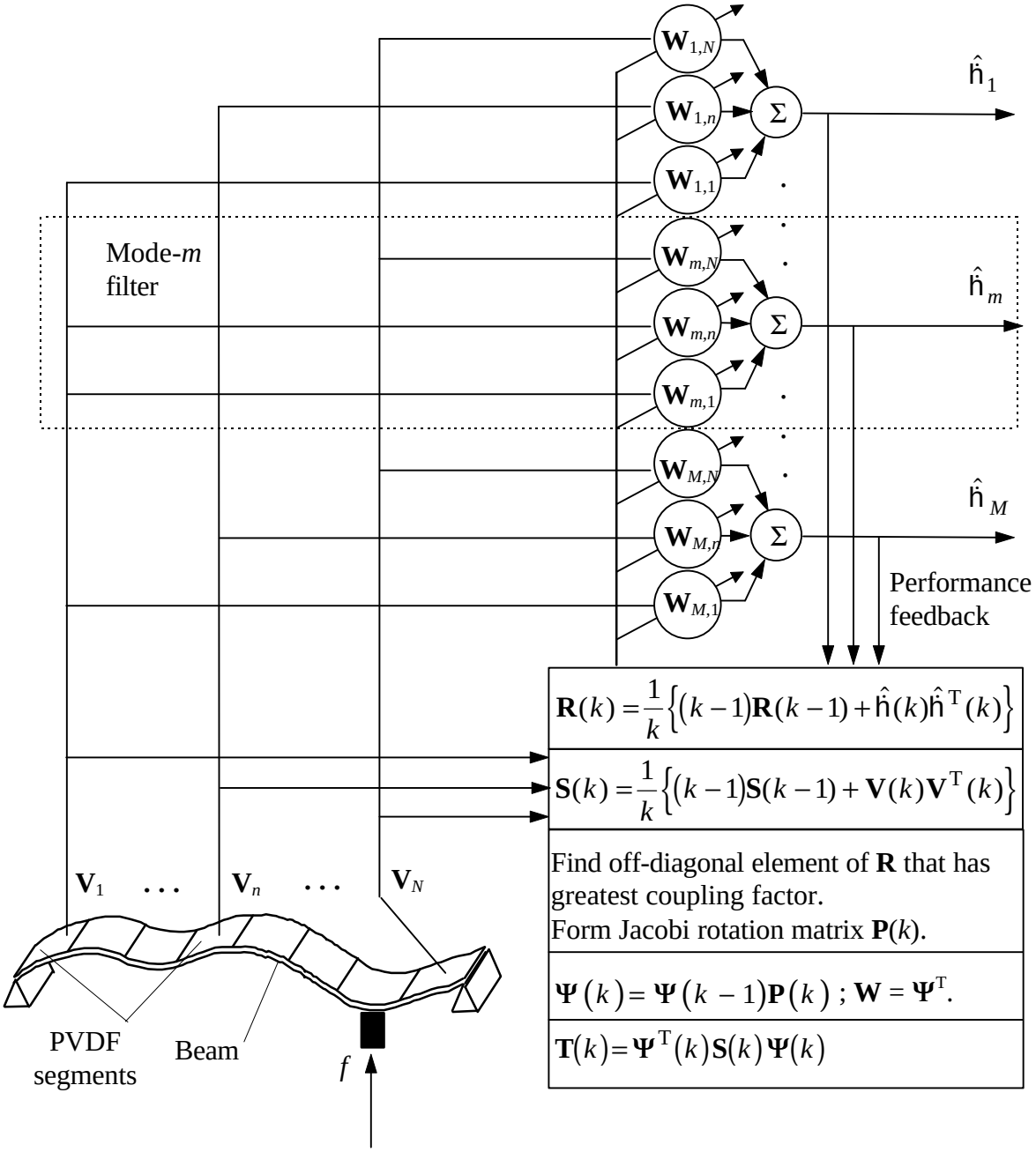

Figure 5. 11 Input connections to Algorithm C.

5.4.2 Numerical Example

Algorithm C results in a faster convergence and more robust gain matrix than Algorithm B. The eight-time decimated history of the rotation angle q in Figure 5.12 shows that the rotation angle converges to zero in fewer time steps than in Algorithm B. Moreover, after the rotation angle converges around zero, it does not 'spike' to a big value and flip to the opposite direction, as occurred with Algorithm B.

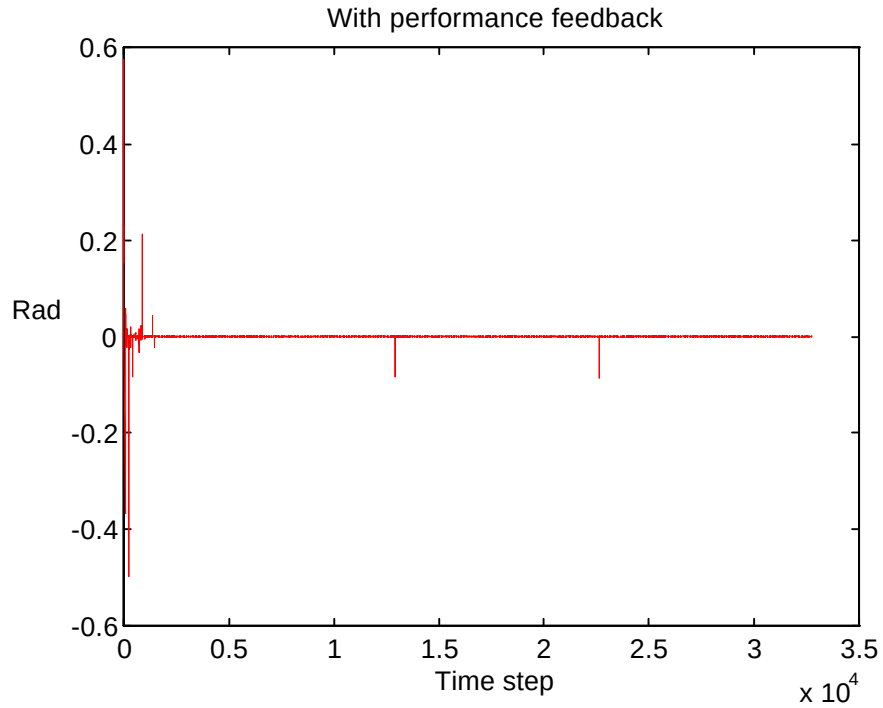

Figure 5.12 Rotation angle history, Algorithm C.

The sensor gain matrix **W** obtained after 32768 time steps with Algorithm C is shown in Fig. 5.13. We can see that this gain matrix is very close to the ideal gain matrix shown in Fig. 4.9. The sensor output correlation matrix is shown in Figure 5.14. This correlation matrix is practically diagonal, meaning that the sensor outputs are uncorrelated. From the last two figures, we can conclude that the modal filter with Algorithm C effectively decouples the segment output voltages **V** into modal coordinates.