Chapter Six

A Numerical Procedure for Eigenvalues of Reducible Matrices

The generalized eigenmode of a square matrix has been introduced and studied in Chapter 3. More specifically, in Sections 3.2 and 3.3 the existence of a generalized eigenmode of a square regular matrix has been proved by making use of its normal form. As the proofs in Sections 3.2 and 3.3 are constructive, a conceptual algorithm has been obtained by which a generalized eigenmode in principle can be computed. See in particular the proof of Corollary 3.16. However, the obtained algorithm heavily relies on a normal form of the matrix involved.

In this chapter an alternative algorithm is presented. *Howard's algorithm*, as it is called, is used to compute the generalized eigenmode of a square regular matrix in a direct way, avoiding its normal form. Despite the differences, many similarities exist between the approaches and algorithms in Chapter 3 and this chapter. The notion of a bouquet matrix, studied in Section 3.4, will play a prominent role in the current chapter.

As noted, the existence of a generalized eigenmode of a square regular matrix has been treated in Sections 3.2 and 3.3, where it has been shown that for any regular matrix $A \in \mathbb{R}_{\max}^{n\times n}$, finite vectors $\eta, v \in \mathbb{R}^n$ exist such that for all $k \geq 0$

$$A \otimes (v + k \times \eta) = v + (k+1) \times \eta\,. \tag{6.1}$$

Note that if η and v satisfy (6.1) for all $k \geq 0$, then for each k, the maximum of $a_{ij}+v_j+k\eta_j, j \in \underline{n}$, in the matrix product on the left-hand side of (6.1), needs to be attained for just one suitable pair (j, i) per equation. Therefore, a first (brute force) attempt to determine a solution of (6.1) is to assign, independently of k, to each $i \in \underline{n}$ precisely one j from the set $\underline{n}$ for which the maximum in the ith equation of (6.1) might be attained. Then, with each $i \in \underline{n}$, precisely one j from the set $\underline{n}$ is associated. This type of association is called *policy* in the literature.

A policy Π can be seen as a mapping $\Pi : \mathcal{N}(A) \to \mathcal{D}(A)$, assigning to each node $i \in \mathcal{N}(A)$ an arc $\Pi_i \in \mathcal{D}(A)$ such that node i is the end node of arc Π_i. Let A^Π be the matrix obtained from A by keeping all the entries of A corresponding to the arcs $\Pi_1, \Pi_2, \ldots, \Pi_n$ and by replacing all the other entries of A by ε. The matrix A^Π is often referred to as a *policy matrix*. Note that a policy matrix has precisely one finite element in each row. Examples of policy matrices have therefore already been encountered in Section 3.4 in the form of sunflower and bouquet matrices. In fact, policy matrices and bouquet matrices are different names for the same type of matrices.

In Section 3.4, we gave a graph-theoretical method to compute a generalized eigenmode of the policy/bouquet matrix A^Π. The method can be implemented ef-

ficiently. The problem of finding a generalized eigenmode of A can therefore be reduced to that of finding a policy Π such that the generalized eigenmode of A^{Π} is also a generalized eigenmode of the overall matrix A. The solution of (6.1) is attained by at least one policy, and since there are only finitely many policies, it is clear that searching the space of policies will eventually lead to the solution of (6.1). Unfortunately, for a fully finite matrix A the number of policies is equal to n^n, indicating that in general the number of policies grows exponentially with n. Hence, a clever method has to be derived for improving a policy when it is not yet the right one.

Howard's algorithm, also known as the *policy iteration algorithm*, is an iterative algorithm for computing a generalized eigenmode. First, a policy is chosen and the eigenmode of the corresponding policy matrix is computed. This part of the overall algorithm is called *value determination* and is presented in algorithmic form in Section 6.1.1. Next, it is tested whether the generalized eigenmode of the policy matrix is already a generalized eigenmode of the original matrix. If so, a solution of (6.1) for all $k \geq 0$ has been found. If not, the policy has to be adapted. To obtain a clever adaptation scheme, first the equations for η and v in (6.1), which must hold for all $k \geq 0$, will be replaced by an equivalent set of equations for η and v that are independent of k. Based on the equivalent set of equations, a scheme will be derived on how to improve the chosen but incorrect policy. This scheme, called *policy improvement*, will be presented in algorithmic form in Section 6.1.2.

Howard's algorithm will be presented in Section 6.1.3. It is based on the algorithms developed and presented in Sections 6.1.1 and 6.1.2. Numerical examples are provided in Section 6.2.

6.1 HOWARD'S ALGORITHM

6.1.1 Value determination

Consider the matrix A, and let Π be a given policy. Our aim is to compute a generalized eigenmode of the policy matrix A^{Π}; that is, we wish to compute vectors $\eta, v \in \mathbb{R}^n$ such that for all $k \geq 0$

$$A^{\Pi} \otimes (v + k \times \eta) = v + (k+1) \times \eta \,.$$

The matrix A^{Π} is a bouquet matrix, and the communication graph $\mathcal{G}(A^{\Pi})$ is made up of one or more subgraphs, being sunflower graphs, with the associated matrices, being sunflower matrices. In Section 3.4 it has been shown that the eigenvalue of a sunflower matrix exists and equals the mean of the (only) circuit in the associated sunflower graph. A corresponding eigenvector follows by going along the circuit and paths, starting from an arbitrary chosen node in the circuit. Combining the obtained eigenvalues and eigenvectors, a generalized eigenmode of the matrix A^{Π} is obtained; see (3.15) and (3.16) in Section 3.4 for details.

The algorithm in Section 3.4 is going to be part of Howard's algorithm and as such is applied to compute a generalized eigenmode of a policy (matrix). Note that for each obtained eigenmode, the cycle-time vector η will be unique, whereas the

vector v will not be unique. Indeed, v may be changed by increasing each of its components (belonging to the same sunflower) with the same number.

In order to ensure that Howard's algorithm terminates, the new value of the vector v is partly based on its old value, thereby introducing a kind of ordering between the eigenmodes corresponding to subsequent policies. More specifically, in each sunflower graph a node is chosen in the associated (unique) circuit, and the corresponding component of v is kept at the value it had obtained previously. The components corresponding to the other nodes of the sunflower are given newly computed values. When starting Howard's algorithm, the vector v is set to the unit vector, i.e., $v := \mathbf{u}$. In the algorithm stated below, the direct predecessor of node j in the communication graph of A^{Π} is denoted by $\pi_{\Pi}(j)$.

Algorithm 6.1.1 VALUE DETERMINATION

1. *Find a circuit ζ in the graph $\mathcal{G}(A^{\Pi})$.*
2. *Compute the average weight of ζ and denote it by $\bar{\eta}_{\zeta}$.*
3. *Select a node in the circuit ζ, say, node j, and set $\eta_j := \bar{\eta}_{\zeta}$. Further, set $v_j := v_j$, i.e., let v_j keep the value it had obtained previously.*
4. *Visit all nodes in the graph $\mathcal{G}(A^{\Pi})$ that are reachable from node j. If node i is visited in this process, then set $\eta_i := \bar{\eta}_{\zeta}$ and calculate v_i from the iteration $v_i = a_{i,\pi_{\Pi}(i)} - \bar{\eta}_{\zeta} + v_{\pi_{\Pi}(i)}$.*
5. *If there remain nodes that are not reachable from node j in $\mathcal{G}(A^{\Pi})$, restart the algorithm with the steps 1 to 4 for the graph made up of these nodes and the associated arcs from $\mathcal{G}(A^{\Pi})$.*

The reason for dealing with the value of v_j as in step 3 is that monotonicity is then achieved with respect to the following lexicographical ordering. Given two candidate generalized eigenmodes (η, v) and (η', v'), with $\eta, v, \eta', v' \in \mathbb{R}^n$, it is said that $(\eta, v) \leq (\eta', v')$ if either $\eta \leq \eta'$ or $\eta = \eta'$ and $v \leq v'$. Here $\eta \leq \eta'$ means that η componentwise is less than or equal to η', with at least one strict inequality, and similarly for $v \leq v'$. This monotonicity can be used to prove (see Section 6.5) that the overall algorithm converges.

Given a policy Π and a generalized eigenmode (η, v) of the associated policy matrix A^{Π}, it can be checked whether (η, v) is also a generalized eigenmode of the matrix A itself. If so, we have found two finite vectors η and v such that (6.1) is satisfied for all $k \geq 0$ and an eigenmode has been found. If not, then the policy Π is not yet a correct policy for matrix A and should be replaced by finding a new (improved) policy Π'. How this is done is explained in the policy improvement algorithm in Section 6.1.2.

6.1.2 Policy improvement

As announced, in this section first the equations in (6.1), which must hold for all $k \geq 0$, will be shown to be equivalent to two sets of alternative equations that

are independent of k. For that purpose, consider a regular matrix $A \in \mathbb{R}_{\max}^{n\times n}$. If $\eta, v \in \mathbb{R}^n$ satisfy (6.1) for all $k \geq 0$, this means in conventional notation that η and v are such that for all $k \geq 0$ and all $i \in \underline{n}$

$$\max_{j\in\underline{n}} \{a_{ij} + v_j + k\eta_j\} = v_i + (k+1)\eta_i. \tag{6.2}$$

Note that for notational convenience the multiplication sign $\times$ is omitted in (6.2). From the definition of the arc set, it follows that $a_{ij} \neq \varepsilon$ if and only if $(j,i) \in \mathcal{D}(A)$. Therefore, the previous implies that for all $k \geq 0$ and all $i \in \underline{n}$

$$\max_{j\in\mathcal{D}(A)_i} \{a_{ij} + v_j + k\eta_j\} = v_i + (k+1)\eta_i,$$

where $\mathcal{D}(A)_i = \{j : (j,i) \in \mathcal{D}(A)\}$. Note that, for a fixed node i, $j \in \mathcal{D}(A)_i$ if and only if $(j,i) \in \mathcal{D}(A)$. Hence, the above maximization over nodes in $\mathcal{D}(A)_i$ can also be written as a maximization over arcs in $\mathcal{D}(A)$ that go to the specific node i. Therefore, it follows from (6.2) that for all $k \geq 0$ and all $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(A)} \{a_{ij} + v_j + k\eta_j\} = v_i + (k+1)\eta_i. \tag{6.3}$$

In the following the above notation with maximization over an arc set will used extensively.

Dividing both sides in (6.3) by $k > 0$ yields for all $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(A)} \left\{ \frac{a_{ij} + v_j + k\eta_j}{k} \right\} = \frac{v_i + (k+1)\eta_i}{k}.$$

Note that the components of both η and v are finite. Also all the entries a_{ij} with $(j,i) \in \mathcal{D}(A)$ are finite. Hence, in the last equality the limit exists for k to ∞ and satisfies for all $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(A)} \eta_j = \eta_i. \tag{6.4}$$

Given a finite vector η satisfying the above equations, define the matrix $\bar{A}$ as

$$[\bar{A}]_{ij} \stackrel{\text{def}}{=} \begin{cases} [A]_{ij} & \text{if } (j,i) \in \mathcal{D}(A) \text{ and } \eta_i = \eta_j, \\ \varepsilon & \text{otherwise.} \end{cases} \tag{6.5}$$

Note that due to the regularity of A the arc set corresponding to matrix $\bar{A}$ is not empty; see exercise 1. Clearly, this arc set is given by

$$\mathcal{D}(\bar{A}) = \{(j,i) \in \mathcal{D}(A) | \eta_i = \eta_j\}. \tag{6.6}$$

With (6.4) and (6.6) it is clear that for the above η

$$\begin{cases} \eta_i = \eta_j & \text{for all } (j,i) \in \mathcal{D}(\bar{A}), \\ \eta_i > \eta_j & \text{for all } (j,i) \in \mathcal{D}(A)\backslash\mathcal{D}(\bar{A}). \end{cases}$$

Now it follows that for k large enough and for any $i \in \underline{n}$ the expression

$$\max_{(j,i)\in\mathcal{D}(A)} \{a_{ij} + v_j + k\eta_j\} = v_i + (k+1)\eta_i$$

can be replaced by

$$\max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j + k\eta_j\} = v_i + (k+1)\eta_i. \tag{6.7}$$

Indeed, it is easy to see that since $\eta_i > \eta_j$ for all $(j,i) \in \mathcal{D}(A)\backslash\mathcal{D}(\bar{A})$, there exists an integer $K \geq 0$ such that for all $k \geq K$ and any $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(A)\backslash\mathcal{D}(\bar{A})} \{a_{ij} + v_j + k\eta_j\} \leq v_i + (k+1)\eta_i. \tag{6.8}$$

The reason for this is that the left-hand side in (6.8) grows at a rate of at most $\max\{\eta_j | (j,i) \in \mathcal{D}(A)\backslash\mathcal{D}(\bar{A})\}$, which is less than η_i, the rate at which the right-hand side grows.

Further, because $\eta_i = \eta_j$ for all $(j,i) \in \mathcal{D}(\bar{A})$, it follows now from (6.7) that for all $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j - \eta_j\} = v_i.$$

Hence, it follows from (6.2), with $\mathcal{D}(\bar{A})$ defined in (6.6), that for all $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(A)} \eta_j = \eta_i, \tag{6.9}$$

$$\max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j - \eta_j\} = v_i. \tag{6.10}$$

Conversely, given finite vectors η and v that satisfy (6.9) and (6.10) for all $i \in \underline{n}$, it follows directly for all $k \geq 0$ and all $i \in \underline{n}$ that

$$\begin{aligned} v_i + (k+1)\eta_i &\overset{(6.10)}{=} \max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j - \eta_j\} + (k+1)\eta_i \\ &= \max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j - \eta_j + (k+1)\eta_i\} \\ &\overset{(6.6)}{=} \max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j + k\eta_j\} \\ &\leq \max_{(j,i)\in\mathcal{D}(A)} \{a_{ij} + v_j + k\eta_j\}. \end{aligned} \tag{6.11}$$

Again by (6.8) it follows easily that for k large enough the inequality in (6.11) has to be an equality.

Hence, finding finite vectors η and v such that (6.9) and (6.10) are satisfied for all $i \in \underline{n}$, with $\mathcal{D}(\bar{A})$ defined in (6.6), is equivalent to finding finite vectors η and v such that (6.1) is satisfied for k large enough, say, for $k \geq K$, with K large enough. Then, by redefining $v := v + K \times \eta$, it is easy to see that the latter is also equivalent to finding finite vectors η and v such that (6.1) is satisfied for all $k \geq 0$.

Now using equations (6.9) and (6.10) it can be checked if a solution of the restricted problem, corresponding to a chosen policy, is a solution of the original problem. If not, the equations also can be used to obtain an improved policy. All this is explained below, where the starting point is a policy Π and two associated vectors η and v such that $A^{\Pi} \otimes (v + k \times \eta) = v + (k+1) \times \eta$, for all $k \geq 0$, or, equivalently, such that for all $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(A^{\Pi})} \eta_j = \eta_i, \qquad \max_{(j,i)\in\mathcal{D}(A^{\Pi})} \{a_{ij} + v_j - \eta_j\} = v_i.$$

The idea is now to check whether the vectors η and v are also such that for all $i \in \underline{n}$

$$\max_{(j,i)\in\mathcal{D}(\mathcal{A})} \eta_j = \eta_i, \qquad \max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j - \eta_j\} = v_i$$

and, if not, to give a better policy. This all can be done by the two steps described next.

The first step is to check whether η satisfies (6.9) (i.e., to check whether it holds that $\max\{\eta_j|(j,i) \in \mathcal{D}(A)\} = \eta_i$ for all $i \in \underline{n}$).

1. Determine the set

$$I_1 = \left\{ i \in \mathcal{N}(A) \,\middle|\, \eta_i < \max_{(j,i)\in\mathcal{D}(A)} \eta_j \right\}.$$

If $I_1 \neq \emptyset$, then determine for all $i \in I_1$ the sets

$$\mathcal{D}(A)_i^1 = \left\{ (k,i) \in \mathcal{D}(A) \,\middle|\, \eta_k = \max_{(j,i)\in\mathcal{D}(A)} \eta_j \right\}.$$

If $I_1 \neq \emptyset$, then the present η and the corresponding policy Π are not yet correct. A better policy Π' has to be found. This is done by adjusting Π for those nodes i that do not satisfy (6.9), i.e., for nodes in the set I_1. This can be done as follows. Define

$$\Pi_i' := \begin{cases} (k,i) & \text{for some } (k,i) \in \mathcal{D}(A)_i^1 \text{ if } i \in I_1, \\ \Pi_i & \text{if } i \notin I_1. \end{cases}$$

Next, return to the value determination in Section 6.1.1 to compute a new generalized eigenmode for the policy matrix $A^{\Pi'}$.

If $I_1 = \emptyset$, then the present η is such that (6.9) is satisfied for all $i \in \underline{n}$. Next, (6.10) has to be considered. Therefore, consider $\mathcal{D}(\bar{A})$ as defined in (6.6) and continue as follows.

2. Determine the set

$$I_2 = \left\{ i \in \mathcal{N}(A) \,\middle|\, v_i < \max_{(j,i)\in\mathcal{D}(\bar{A})} (a_{ij} + v_j - \eta_j) \right\}.$$

If $I_2 \neq \emptyset$, then determine for all $i \in I_2$ the sets

$$\mathcal{D}(A)_i^2 = \left\{ (k,i) \in D(A) \,\middle|\, a_{ik} + v_k - \eta_k = \max_{(j,i)\in\mathcal{D}(\bar{A})} (a_{ij} + v_j - \eta_j) \right\}.$$

With this second step it is checked whether the vectors η and v satisfy (6.10); that is, it is checked whether $\max\{a_{ij} + v_j - \eta_j|(j,i) \in \mathcal{D}(\bar{A})\} = v_i$ for all $i \in \underline{n}$.

If $I_2 \neq \emptyset$, then the current η and v, and consequently the current policy, are not yet correct. A new policy Π' is found by adjusting Π for those nodes that do not satisfy (6.10), i.e., for nodes in the set I_2. This can be done as follows. Define

$$\Pi_i' := \begin{cases} (k,i) & \text{for some } (k,i) \in \mathcal{D}(A)_i^2 \text{ if } i \in I_2, \\ \Pi_i & \text{if } i \notin I_2. \end{cases}$$

Next, return to the value determination in Section 6.1.1 to compute a new generalized eigenmode for the policy matrix $A^{\Pi'}$.

If $I_2 = \emptyset$, then the current η and v are such that (6.10) is satisfied for all $i \in \underline{n}$ and the algorithm terminates.

The above steps are summarized in the following algorithm.

Algorithm 6.1.2 POLICY IMPROVEMENT

1. *Determine the set* $I_1 = \left\{ i \in \mathcal{N}(A) \,\middle|\, \eta_i < \max_{(j,i)\in\mathcal{D}(A)} \eta_j \right\}$.

   • *If* $I_1 = \emptyset$, *the current* η *is such that (6.9) is satisfied for all* $i \in \underline{n}$. *Then compute* $\mathcal{D}(\bar{A}) = \{(j,i) \in \mathcal{D}(A) | \eta_i = \eta_j\}$ *and continue with step 2.*

   • *If* $I_1 \neq \emptyset$, *then determine for all* $i \in I_1$ *the sets*

   $$\mathcal{D}(A)_i^1 = \left\{ (k,i) \in \mathcal{D}(A) \,\middle|\, \eta_k = \max_{(j,i)\in\mathcal{D}(A)} \eta_j \right\}.$$

   Define a new policy Π' *as*

   $$\Pi_i' := \begin{cases} (k,i) & \text{for some } (k,i) \in \mathcal{D}(A)_i^1 \text{ if } i \in I_1, \\ \Pi_i & \text{if } i \notin I_1. \end{cases}$$

2. *Determine the set* $I_2 = \left\{ i \in \mathcal{N}(A) \,\middle|\, v_i < \max_{(j,i)\in\mathcal{D}(\bar{A})} (a_{ij} + v_j - \eta_j) \right\}$.

   • *If* $I_2 = \emptyset$, *then the current* η *and* v *are such that both (6.9) and (6.10) are satisfied for all* $i \in \underline{n}$. *Hence, the algorithm can stop.*

   • *If* $I_2 \neq \emptyset$, *then determine for all* $i \in I_2$ *the sets*

   $$\mathcal{D}(A)_i^2 = \left\{ (k,i) \in D(A) \,\middle|\, a_{ik} + v_k - \eta_k = \max_{(j,i)\in\mathcal{D}(\bar{A})} (a_{ij} + v_j - \eta_j) \right\}.$$

   Define a new policy Π' *as*

   $$\Pi_i' := \begin{cases} (k,i) & \text{for some } (k,i) \in \mathcal{D}(A)_i^2 \text{ if } i \in I_2, \\ \Pi_i & \text{if } i \notin I_2. \end{cases}$$

6.1.3 The overall algorithm

In this section we put together the ideas presented in the previous sections. This results in an algorithm that yields a generalized eigenmode (η, v) for square regular matrices.

Algorithm 6.1.3 HOWARD'S ALGORITHM

Choose an arbitrary policy $\Pi(1)$ *and set* $v^{\Pi(0)} := \mathbf{u}$. *Apply the following steps for* $k \geq 1$.

1. *Compute a generalized eigenmode* $(\eta^{\Pi(k)}, v^{\Pi(k)})$ *for* $A^{\Pi(k)}$ *via the value determination algorithm, i.e., Algorithm 6.1.1, where* $v_j^{\Pi(k)} := v_j^{\Pi(k-1)}$ *for selected nodes* j, *one in each of the circuits of* $\mathcal{G}(A^{\Pi(k)})$.

2. *Check whether* $(\eta^{\Pi(k)}, v^{\Pi(k)})$ *is also a generalized eigenmode of* A. *If so, the algorithm terminates. Otherwise, construct a new policy* $\Pi(k+1)$ *via the policy improvement algorithm, i.e., Algorithm 6.1.2. Set* $k := k+1$, *and return to step 1.*

Howard's algorithm is a very efficient algorithm for computing the generalized eigenmode of a regular matrix A. It has been reported to be on average almost linear in time; that is, in processing time, the algorithm is on average linearly proportional to the number of finite entries in the matrix A, though a tight upper bound has not yet been found. It is also conjectured that in the worst case the run time is polynomial in the number of finite entries of A. For more details on this, see [23].

6.2 EXAMPLES

In this section Howard's algorithm will be illustrated by means of the matrices in Examples 5.1.2 and 5.1.3.

Example 6.2.1 *Consider the matrix A studied in Example 5.1.2, given by*

$$A = \begin{pmatrix} 1 & 2 & \varepsilon & 7 \\ \varepsilon & 3 & 5 & \varepsilon \\ \varepsilon & 4 & \varepsilon & 3 \\ \varepsilon & 2 & 8 & \varepsilon \end{pmatrix}.$$

The communication graph of A is depicted in Figure 5.2. Recall that the graph is not strongly connected and that A is reducible.

Take as a starting point the policy $\Pi(1) = \{(1,1),(2,2),(2,3),(2,4)\}$ and $v^{\Pi(0)} := \mathbf{u}$. The graph $\mathcal{G}(A^{\Pi(1)})$ associated to $A^{\Pi(1)}$ is depicted in Figure 6.1. To obtain a generalized eigenmode of $A^{\Pi(1)}$, step 1 of Howard's algorithm will

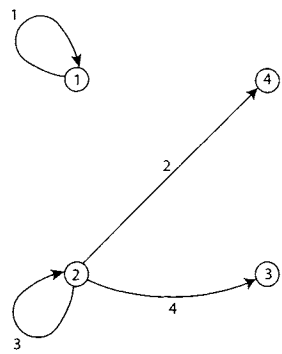

Figure 6.1: Communication graph of $A^{\Pi(1)}$.

be applied (i.e., the value determination according to Algorithm 6.1.1). Note that $\mathcal{G}(A^{\Pi(1)})$ contains two circuits, corresponding to two sunflowers. One circuit, denoted $\zeta_1^{\Pi(1)}$, is given by the arc $(1,1)$, with average weight 1. Clearly, the circuit consists of just one node, namely, node 1. Choose this node, and set $\eta_1^{\Pi(1)} := 1$ and $v_1^{\Pi(1)} := v_1^{\Pi(0)} (= 0)$. As there are no other vertices that can be reached from node 1 in $\mathcal{G}(A^{\Pi(1)})$, step 1 through 4 of Algorithm 6.1.1 are now completed for the circuit $\zeta_1^{\Pi(1)}$.

Next, steps 1 through 5 of Algorithm 6.1.1 have to be repeated for the remaining nodes and arcs of $\mathcal{G}(A^{\Pi(1)})$. *The second circuit, denoted* $\zeta_2^{\Pi(1)}$, *is given by the arc* $(2,2)$ *and has average weight* 3. *This circuit consists of just one node, namely, node* 2. *Choose this node, and set* $\eta_2^{\Pi(1)} := 3$ *and* $v_2^{\Pi(1)} := v_2^{\Pi(0)} (= 0)$. *Now note that there are vertices that can be reached from node* 2 *in* $\mathcal{G}(A^{\Pi(1)})$, *namely, nodes* 3 *and* 4. *Therefore, set* $\eta_3^{\Pi(1)} := 3$ *and* $\eta_4^{\Pi(1)} := 3$. *Further, compute* $v_3^{\Pi(1)}$ *and* $v_4^{\Pi(1)}$ *according to step 4 of Algorithm 6.1.1. Specifically,* $\pi_{\Pi(1)}(3) = 2$ *and* $\pi_{\Pi(1)}(4) = 2$, *which gives*

$$v_3^{\Pi(1)} = a_{32} + v_2^{\Pi(1)} - \eta_2^{\Pi(1)} = 1$$

and

$$v_4^{\Pi(1)} = a_{42} + v_2^{\Pi(1)} - \eta_2^{\Pi(1)} = -1.$$

Since all nodes have been treated, a generalized eigenmode $(\eta^{\Pi(1)}, v^{\Pi(1)})$ *of* $A^{\Pi(1)}$ *is obtained with*

$$\eta^{\Pi(1)} = \begin{pmatrix} 1 \\ 3 \\ 3 \\ 3 \end{pmatrix} \quad \text{and} \quad v^{\Pi(1)} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ -1 \end{pmatrix}.$$

Next it has to be checked whether $(\eta^{\Pi(1)}, v^{\Pi(1)})$ *is also an eigenmode of* A *itself, i.e., whether equations (6.9) and (6.10) are satisfied. This will be done by applying step 2 of Howard's algorithm (i.e., the policy improvement according to Algorithm 6.1.2). First, the set* I_1 *has to be determined for* $(\eta^{\Pi(1)}, v^{\Pi(1)})$ *as given above. It follows here easily that* $I_1 = \{1\}$. *The corresponding arc set* $\mathcal{D}(A)_1^1$ *is equal to* $\{(2,1),(4,1)\}$. *Using the improvement as in step 1 of Algorithm 6.1.2, it follows that, for instance,* $\Pi(2) = \{(2,1),(2,2),(2,3),(2,4)\}$ *can be taken; that is, to obtain* $\Pi(2)$ *from* $\Pi(1)$, *just replace* $(1,1)$ *by* $(2,1)$ *and keep the other arcs. The graph* $\mathcal{G}(A^{\Pi(2)})$ *associated to* $A^{\Pi(2)}$ *is depicted in Figure 6.2. To obtain a gen-*

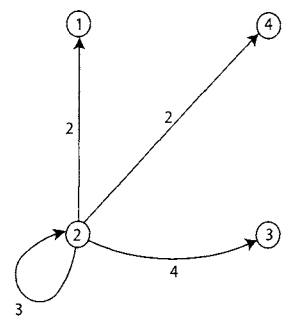

Figure 6.2: Communication graph of $A^{\Pi(2)}$.

eralized eigenmode of $A^{\Pi(2)}$, *steps 1 through 5 of Algorithm 6.1.1 will be applied. Note that the graph* $\mathcal{G}(A^{\Pi(2)})$ *contains just one circuit given by the arc* $(2,2)$. *Denote this circuit by* $\zeta^{\Pi(2)}$. *Its average weight is* 3, *and it consists of just one node,*

namely, node 2. Choose this node, and set $\eta_2^{\Pi(2)} := 3$ *and* $v_2^{\Pi(2)} := v_2^{\Pi(1)} (= 0)$. *Now note that all the vertices can be reached from node 2 in* $\mathcal{G}(A^{\Pi(2)})$. *Therefore, set* $\eta_1^{\Pi(2)} := 3$, $\eta_3^{\Pi(2)} := 3$, *and* $\eta_4^{\Pi(2)} := 3$. *Note that* $\pi_{\Pi(2)}(1) = 2$, $\pi_{\Pi(2)}(3) = 2$, *and* $\pi_{\Pi(2)}(4) = 2$. *Hence, it follows that*

$$v_1^{\Pi(2)} = a_{12} + v_2^{\Pi(2)} - \eta_2^{\Pi(2)} = -1,$$

$$v_3^{\Pi(2)} = a_{32} + v_2^{\Pi(2)} - \eta_2^{\Pi(2)} = 1,$$

and

$$v_4^{\Pi(2)} = a_{42} + v_2^{\Pi(2)} - \eta_2^{\Pi(2)} = -1,$$

which gives a generalized eigenmode $(\eta^{\Pi(2)}, v^{\Pi(2)})$ *of* $A^{\Pi(2)}$ *with*

$$\eta^{\Pi(2)} = \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \end{pmatrix} \quad \text{and} \quad v^{\Pi(2)} = \begin{pmatrix} -1 \\ 0 \\ 1 \\ -1 \end{pmatrix}.$$

It can be checked whether $(\eta^{\Pi(2)}, v^{\Pi(2)})$ *is also an eigenmode of* A *itself by applying steps 1 and 2 of Algorithm 6.1.2 again. In order to do so, first, set* I_1 *has to be determined for* $(\eta^{\Pi(2)}, v^{\Pi(2)})$ *as given above. Since all components of* $\eta^{\Pi(2)}$ *have the same value, it follows that* $I_1 = \emptyset$. *Also it follows that* $\mathcal{D}(\bar{A})$, *defined in (6.6), coincides with* $\mathcal{D}(A)$. *Hence, step 1 can be skipped and step 2 has to be done. So, set* I_2 *has to be determined. Therefore, determine the vector* $w^{\Pi(2)}$, *with components* $w_i^{\Pi(2)} = \max_{(j,i) \in \mathcal{D}(\bar{A})} \{a_{ij} + v_j^{\Pi(2)} - \eta_j^{\Pi(2)}\}$ *for all* $i \in \underline{4}$, *resulting in*

$$w^{\Pi(2)} = \begin{pmatrix} 3 \\ 3 \\ 1 \\ 6 \end{pmatrix}.$$

Comparing $w^{\Pi(2)}$ *with* $v^{\Pi(2)}$, *as done in step 2, it is clear that* $I_2 = \{1, 2, 4\}$. *The corresponding arc sets* $\mathcal{D}(A)_i^2$ *are given by* $\mathcal{D}(A)_1^2 = \{(4, 1)\}$, $\mathcal{D}(A)_2^2 = \{(3, 2)\}$, *and* $\mathcal{D}(A)_4^2 = \{(3, 4)\}$. *Using the improvement as in step 2, it follows that* $\Pi(3) = \{(4, 1), (3, 2), (2, 3), (3, 4)\}$ *has to be taken. The graph* $\mathcal{G}(A^{\Pi(3)})$ *associated to* $A^{\Pi(3)}$ *is depicted in Figure 6.3. Again steps 1 through 5 of Algorithm 6.1.1 will be applied to obtain a generalized eigenmode of* $A^{\Pi(3)}$. *Note that the graph* $\mathcal{G}(A^{\Pi(3)})$ *contains just one circuit given by the arcs* $\{(2, 3), (3, 2)\}$, *with average weight equal to* $4\frac{1}{2}$. *Choose node 2 on this circuit. Performing steps 3 and 4 of Algorithm 6.1.1 yields a generalized eigenmode* $(\eta^{\Pi(3)}, v^{\Pi(3)})$ *of* $A^{\Pi(3)}$, *with*

$$\eta^{\Pi(3)} = \begin{pmatrix} 4\frac{1}{2} \\ 4\frac{1}{2} \\ 4\frac{1}{2} \\ 4\frac{1}{2} \end{pmatrix} \quad \text{and} \quad v^{\Pi(3)} = \begin{pmatrix} 5\frac{1}{2} \\ 0 \\ -\frac{1}{2} \\ 3 \end{pmatrix}.$$

It can be checked whether $(\eta^{\Pi(3)}, v^{\Pi(3)})$ *is also an eigenmode of* A *itself by applying steps 1 and 2 of Algorithm 6.1.2. Therefore, the set* I_1 *and possibly set* I_2

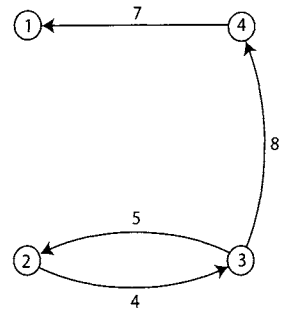

Figure 6.3: Communication graph of $A^{\Pi(3)}$.

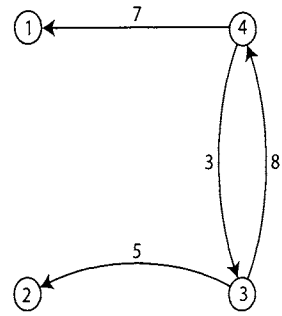

Figure 6.4: Communication graph of $A^{\Pi(4)}$.

have to be determined. It turns out that $I_1 = \emptyset$ *and that* $I_2 = \{3\}$. *The corresponding arc set* $\mathcal{D}(A)^2_3$ *is given by* $\mathcal{D}(A)^2_3 = \{(4,3)\}$. *Using the improvement as in step 2, it follows that* $\Pi(4) = \{(4,1),(3,2),(4,3),(3,4)\}$ *has to be taken. The graph* $\mathcal{G}(A^{\Pi(4)})$ *associated to* $A^{\Pi(4)}$ *is depicted in Figure 6.4. Again steps 1 through 5 of Algorithm 6.1.1 will be applied. Note that the graph* $\mathcal{G}(A^{\Pi(4)})$ *contains just one circuit given by the arcs* $\{(3,4),(4,3)\}$, *with average weight being* $5\frac{1}{2}$. *Pick node 3 on this circuit. Performing steps 3 and 4 of Algorithm 6.1.1 results in a generalized eigenmode* $(\eta^{\Pi(4)}, v^{\Pi(4)})$ *of* $A^{\Pi(4)}$, *with*

$$\eta^{\Pi(4)} = \begin{pmatrix} 5\frac{1}{2} \\ 5\frac{1}{2} \\ 5\frac{1}{2} \\ 5\frac{1}{2} \end{pmatrix} \quad \text{and} \quad v^{\Pi(4)} = \begin{pmatrix} 3\frac{1}{2} \\ -1 \\ -\frac{1}{2} \\ 2 \end{pmatrix}.$$

It can be checked whether $(\eta^{\Pi(4)}, v^{\Pi(4)})$ *is also an eigenmode of* A *itself by applying steps 1 and 2 of Algorithm 6.1.2. It turns out that both* $I_1 = \emptyset$ *and* $I_2 = \emptyset$. *Hence, the algorithm can stop and the pair* $(\eta^{\Pi(4)}, v^{\Pi(4)})$ *satisfies (6.9) and (6.10), for all* $i \in \underline{4}$, *so that*

$$A \otimes (v^{\Pi(4)} + k \times \eta^{\Pi(4)}) = v^{\Pi(4)} + (k+1) \times \eta^{\Pi(4)},$$

for all $k \geq K$, *with* $K \geq 0$ *large enough. By simple verification it follows that here* $K = 0$ *can be taken. Hence,* $(\eta^{\Pi(4)}, v^{\Pi(4)})$ *constitutes a generalized eigenmode of*

matrix A. In fact, since all entries of $\eta^{\Pi(4)}$ are equal, it follows that $A \otimes v^{\Pi(4)} = \frac{1}{2} \otimes v^{\Pi(4)}$, so that $v^{\Pi(4)}$ is an eigenvector of the matrix A for the eigenvalue $\lambda = 5\frac{1}{2}$.

Example 6.2.2 *Next consider the matrix A studied in Example 5.1.3, given by*

$$A = \begin{pmatrix} 6 & 2 & \varepsilon & 7 \\ \varepsilon & 3 & 5 & \varepsilon \\ \varepsilon & 4 & \varepsilon & 3 \\ \varepsilon & 2 & 8 & \varepsilon \end{pmatrix}.$$

The graph $\mathcal{G}(A)$ is depicted in Figure 5.2, where only the weight of the self-loop at node 1 has to be changed from 1 into 6. Recall that the graph is not strongly connected and that matrix A is reducible.

Take as a starting point policy $\Pi(1) = \{(1,1),(2,2),(2,3),(2,4)\}$ and $v^{\Pi(0)} := \mathbf{u}$. The graph $\mathcal{G}(A^{\Pi(1)})$ associated with $A^{\Pi(1)}$, depicted in Figure 6.5, contains two circuits. One circuit is given by the arc $(1,1)$ and has average weight equal to

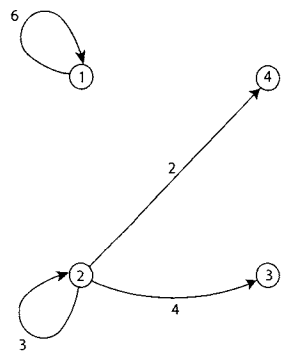

Figure 6.5: Communication graph of $A^{\Pi(1)}$.

6. The other circuit is given by the arc $(2,2)$ and has average weight equal to 3. Further note that nodes 3 and 4 can be reached from node 2. First, consider node 1 and set $\eta_1^{\Pi(1)} := 6$ and $v_1^{\Pi(1)} := 0$. Next, consider node 2 and set $\eta_2^{\Pi(1)} := 3$ and $v_2^{\Pi(1)} := 0$. Now, performing step 4 of Algorithm 6.1.1 starting from node 2 ultimately results in a generalized eigenmode $(\eta^{\Pi(1)}, v^{\Pi(1)})$ of $A^{\Pi(1)}$, with

$$\eta^{\Pi(1)} = \begin{pmatrix} 6 \\ 3 \\ 3 \\ 3 \end{pmatrix} \quad \text{and} \quad v^{\Pi(1)} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ -1 \end{pmatrix}.$$

Next, we check whether $(\eta^{\Pi(1)}, v^{\Pi(1)})$ is also an eigenmode of A itself by applying steps 1 and 2 of Algorithm 6.1.2. First, the set I_1 has to be determined for $(\eta^{\Pi(1)}, v^{\Pi(1)})$ as given above. It follows here easily that $I_1 = \emptyset$. Also, it follows that the arc set $\mathcal{D}(\bar{A})$ coincides with $\mathcal{D}(A)$ except for the arcs $(2,1)$ and $(4,1)$ (i.e., $\mathcal{D}(\bar{A}) = \mathcal{D}(A) \backslash \{(2,1) \cup (4,1)\}$). Hence, step 1 can be skipped and step 2 can

be started. So, the set I_2 has to be determined. Therefore, determine $w^{\Pi(1)}$ with components $w_i^{\Pi(1)} = \max_{(j,i)\in\mathcal{D}(\bar{A})} \{a_{ij} + v_j^{\Pi(1)} - \eta_j^{\Pi(1)}\}$, for all $i \in \underline{4}$. It follows that

$$w^{\Pi(1)} = \begin{pmatrix} 0 \\ 3 \\ 1 \\ 6 \end{pmatrix}.$$

Comparing $w^{\Pi(1)}$ with $v^{\Pi(1)}$, as done in step 2, yields $I_2 = \{2, 4\}$. The corresponding arc sets $\mathcal{D}(A)_i^2$ are given by $\mathcal{D}(A)_2^2 = \{(3, 2)\}$ and $\mathcal{D}(A)_4^2 = \{(3, 4)\}$. Using the improvement as in step 2, it follows that $\Pi(2) = \{(1, 1), (3, 2), (2, 3), (4, 3)\}$ has to be taken. The graph $\mathcal{G}(A^{\Pi(2)})$ associated with $A^{\Pi(2)}$ is depicted in Figure 6.6. Repeated application of the steps of Howard's algorithm now yields the

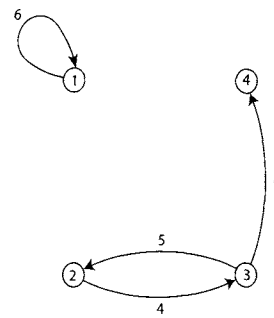

Figure 6.6: Communication graph of $A^{\Pi(2)}$.

following sequence of generalized eigenmodes, communication graphs, and improved policies. The details are left to the reader.

The generalized eigenmode $(\eta^{\Pi(2)}, v^{\Pi(2)})$ of $A^{\Pi(2)}$ is given by

$$\eta^{\Pi(2)} = \begin{pmatrix} 6 \\ 4\frac{1}{2} \\ 4\frac{1}{2} \\ 4\frac{1}{2} \end{pmatrix} \quad \text{and} \quad v^{\Pi(2)} = \begin{pmatrix} 0 \\ 0 \\ -\frac{1}{2} \\ 3 \end{pmatrix}.$$

The improved policy is given by $\Pi(3) = \{(1, 1), (3, 2), (4, 3), (3, 4)\}$. The graph $\mathcal{G}(A^{\Pi(3)})$ associated with $A^{\Pi(3)}$ is depicted in Figure 6.7. The generalized eigenmode $(\eta^{\Pi(3)}, v^{\Pi(3)})$ of $A^{\Pi(3)}$ is given by

$$\eta^{\Pi(3)} = \begin{pmatrix} 6 \\ 5\frac{1}{2} \\ 5\frac{1}{2} \\ 5\frac{1}{2} \end{pmatrix} \quad \text{and} \quad v^{\Pi(3)} = \begin{pmatrix} 0 \\ -1 \\ -\frac{1}{2} \\ 2 \end{pmatrix}.$$

It is now easily checked that the pair $(\eta^{\Pi(3)}, v^{\Pi(3)})$ satisfies (6.9) and (6.10) for all $i \in \underline{4}$, so that

$$A \otimes (v^{\Pi(3)} + k \times \eta^{\Pi(3)}) = v^{\Pi(3)} + (k + 1) \times \eta^{\Pi(3)},$$

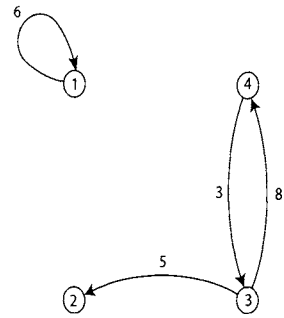

Figure 6.7: Communication graph of $A^{\Pi(3)}$.

for all $k \geq K$, with $K \geq 0$ large enough. By simple verification it follows that here $K = 6$ can be taken. Hence, $(\eta^{\Pi(4)}, v^{\Pi(4)} + 6 \times \eta^{\Pi(4)})$ constitutes a generalized eigenmode of matrix A.

6.3 HOWARD'S ALGORITHM FOR HIGHER-ORDER MODELS

The recurrence relation $x(k+1) = A \otimes x(k)$ for all $k \geq 0$, with $x(0) = x_0$, can also be formulated as

$$x(k) = A \otimes x(k-1), \tag{6.12}$$

for all $k \geq 0$, with $x(-1) = x_0$, and Howard's algorithm then yields vectors η and v in $\mathbb{R}^n$ such that for all $k \geq 0$

$$v + k \times \eta = A \otimes (v + (k-1) \times \eta). \tag{6.13}$$

In this section we are going to study the generalization of (6.12) to higher-order recurrence relations and sketch the development of a Howard-like algorithm for computing a generalized eigenmode. Consider the higher-order recurrence relation

$$x(k) = \bigoplus_{l=0}^{M} A_l \otimes x(k-l), \tag{6.14}$$

for all $k \geq 0$ and with $x(-1), x(-2), \ldots, x(-M)$ specified. For details, see Section 4.5. The matrices A_0, $A_l, \ldots, A_M$ are supposed to belong to the set $\mathbb{R}^{n \times n}_{\max}$, where the graph of matrix A_0 does not contain circuits with a nonnegative weight. The latter requirement ensures that the sequence $\{x(k) : k \geq 0\}$ is uniquely determined, given $x(-1), x(-2), \ldots, x(-M)$. The problem of finding the generalized eigenmode then is to find vectors η and v in $\mathbb{R}^n$ such that

$$v + k \times \eta = \bigoplus_{l=0}^{M} A_l \otimes (v + (k-l) \times \eta), \tag{6.15}$$

for all $k \geq 0$. As in Section 6.1.2, the latter equations for η and v can be replaced by two sets of alternative equations. In order to describe these sets we introduce the

so-called multigraph associated with (6.14). To abbreviate the notation, we write $\mathcal{A} = (A_0, \ldots, A_M)$ for the matrices in (6.15). The multigraph associated to (6.14) will be denoted by $G(\mathcal{A})$ and consists of a set of nodes $\mathcal{N}(\mathcal{A}) = \underline{n}$ and a set of directed edges $\mathcal{D}(\mathcal{A}) = \{(j,l,i) \in \mathcal{N}(\mathcal{A}) \times \mathbb{N} \times \mathcal{N}(\mathcal{A}) : [A_l]_{ij} \neq \varepsilon\}$. In the latter, (j,l,i) has to be interpreted as an edge from node j to node i with a delay l and a weight $[A_l]_{ij}$. It is therefore possible to have several arcs between node j and i, each with a different delay.

A concise way of writing (6.14) is by means of a shift operator. We formally set $\gamma^{\otimes l} \otimes x(k) = x(k-l)$, $l \geq 0$, and

$$\mathcal{A}(\gamma) = \bigoplus_{l=0}^{M} A_l \otimes \gamma^{\otimes l}.$$

Then, (6.14) reads as $x(k) = \mathcal{A}(\gamma) \otimes x(k)$.

Similar to the analysis put forward in Section 6.1.2, it can be shown that the existence of η and v in $\mathbb{R}^n$ satisfying (6.15) for all $k \geq 0$ is equivalent to the existence of η and v in $\mathbb{R}^n$ such that for all $i \in \underline{n}$

$$\max_{(j,l,i)\in\mathcal{D}(\mathcal{A})} \eta_j = \eta_i, \tag{6.16}$$

$$\max_{(j,l,i)\in\mathcal{D}(\bar{\mathcal{A}})} \left\{ [A_l]_{ij} + v_j - l \times \eta_j \right\} = v_i, \tag{6.17}$$

where $\bar{\mathcal{A}} = (\bar{A}_0, \ldots, \bar{A}_M)$ with $\bar{A}_0, \ldots, \bar{A}_M$ defined by

$$[\bar{A}_l]_{ij} \stackrel{\text{def}}{=} \begin{cases} [A_l]_{ij} & \text{if } (j,l,i) \in \mathcal{D}(\mathcal{A}) \text{ and } \eta_i = \eta_j, \\ \varepsilon & \text{otherwise.} \end{cases}$$

Note that the maximization in (6.16) over arcs in $\mathcal{D}(A)$ that go to node i can equivalently be formulated as a maximization over all nodes j and delays l such that $(j,l,i) \in \mathcal{D}(A)$. A similar remark can be made with respect to (6.17).

In the same spirit as in Section 6.1, a policy improvement algorithm can be developed to solve equations (6.16) and (6.17) for η and v in $\mathbb{R}^n$, where a policy again is a selection of edges such that each node is the end node of precisely one edge. Different edges may share the starting node and may have different delays. Formally, Π is a policy when for each $i \in \mathcal{N}(\mathcal{A})$ there exists an $l \in \mathbb{N}$ and a $j \in \mathcal{N}(\mathcal{A})$ such that $(j,l,i) \in \mathcal{D}(\mathcal{A})$, in which case the edge (j,l,i) is said to belong to Π. We write this as $(j,l,i) \in \Pi$.

Given a policy Π, define the polynomial matrix $\mathcal{A}^{\Pi}(\gamma) = \bigoplus_{l=0}^{M} A_l^{\Pi} \otimes \gamma^{\otimes l}$, where

$$[A_l^{\Pi}]_{ij} \stackrel{\text{def}}{=} \begin{cases} [A_l]_{ij} & \text{if } (j,l,i) \in \Pi, \\ \varepsilon & \text{if } (j,l,i) \notin \Pi. \end{cases}$$

As in Section 6.1.1, it turns out to be easy to compute the generalized eigenmode of $\mathcal{A}^{\Pi}(\gamma)$; that is, there is a straightforward version of the value determination algorithm, like Algorithm 6.1.1, to compute η and v in $\mathbb{R}^n$ such that for all $k \geq 0$

$$v + k \times \eta = \bigoplus_{l=0}^{M} A_l^{\Pi} \otimes (v + (k-l) \times \eta).$$

Such η and v satisfy for all $i \in \underline{n}$

$$\max_{(j,l,i)\in\Pi} \eta_j = \eta_i, \qquad \max_{(j,l,i)\in\Pi} \left\{ [A_l]_{ij} + v_j - l \times \eta_j \right\} = v_i.$$

The idea again is check whether the two vectors η and v also satisfy (6.16) and (6.17) for all $i \in \underline{n}$; that is, whether it holds that

$$\max_{(j,l,i)\in\mathcal{D}(\mathcal{A})} \eta_j = \eta_i, \qquad \max_{(j,l,i)\in\mathcal{D}(\bar{A})} \left\{ [A_l]_{ij} + v_j - l \times \eta_j \right\} = v_i,$$

for all $i \in \underline{n}$. If this is the case, a generalized eigenmode has been found. If not, an improved policy is required, as in Section 6.1.2.

The improvement is first based on whether η satisfies (6.16) for all $i \in \underline{n}$. If not, an improved policy can be found as in step 1 of Algorithm 6.1.2, whereupon the value determination algorithm has to be recalled to obtain a new candidate eigenmode and the process has to be repeated.

If η is such that (6.16) is satisfied for all $i \in \underline{n}$, then η and v have to be checked to satisfy (6.17) for all $i \in \underline{n}$. If this is not true, then an improved policy can be found as in step 2 of Algorithm 6.1.2, whereupon the value determination algorithm has to be recalled to obtain a new candidate eigenmode and the process has to be repeated.

6.4 EXERCISES

1. Show that regularity of A implies that the set $\mathcal{D}(\bar{A})$ is not empty.
2. Formulate a version of Howard's algorithm for higher-order descriptions.
3. Develop a version of the power algorithm for higher-order descriptions.
4. Investigate the possibilities to modify Karp's algorithm and the power algorithm so that reducible matrices can be treated. Assume that you have a method to compute the reduced graph and the maximal strongly connected components.
5. Verify that a generalized eigenmode (η, v) of matrix A in Example 2.1.3 is given by

$$\eta = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{4}{3}, \frac{4}{3}, \frac{4}{3}, \frac{4}{3}, \frac{4}{3}, \frac{1}{2}\right)^\top$$

and

$$v = \left(8\frac{1}{2}, 9, 12\frac{1}{2}, 8, 24, 30\frac{1}{3}, 31\frac{2}{3}, 30\frac{5}{6}, 35, 17\right)^\top.$$

6. Compute a generalized eigenmode (η, v) of the matrix

$$A = \begin{pmatrix} 2 & 1 & 0 & 1 & 1 \\ \varepsilon & 2 & 4 & \varepsilon & \varepsilon \\ \varepsilon & 4 & 1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 2 & \varepsilon \\ 5 & 0 & \varepsilon & 0 & 2 \end{pmatrix}.$$

(Answer: $\eta = (4,4,4,2,4)^\top$ and $v = (-3,0,0,0,-2)^\top$.)

7. Compute a generalized eigenmode (η, v) for the higher-order recurrence corresponding to equation (0.13);

$$x(k) = \begin{pmatrix} 2 & 5 \\ \varepsilon & 3 \end{pmatrix} \otimes x(k-1) \oplus \begin{pmatrix} \varepsilon & \varepsilon \\ 3 & \varepsilon \end{pmatrix} \otimes x(k-2), \qquad k \geq 2.$$

(Answer: $\eta = (3,3)^\top$ and $v = (2,0)^\top$.)

8. Compute a generalized eigenmode (η, v) for the higher-order recurrence

$$x(k) = \begin{pmatrix} \varepsilon & 1 \\ -4 & \varepsilon \end{pmatrix} \otimes x(k) \oplus \begin{pmatrix} 2 & 4 \\ -1 & \varepsilon \end{pmatrix} \otimes x(k-2), \qquad k \geq 2.$$

Do the computations by applying the ideas in Section 6.3. Verify the results by going over to a first-order recurrence relation and by applying Howard's algorithm as presented in Section 6.1. (Answer: $\eta = (1,1)^\top$ and $v = (0,-3)^\top$.)

6.5 NOTES

The present chapter is based upon [23]. There, details can also be found on the convergence of Howard's algorithm. However, no tight bounds can be given for the complexity of the algorithm. Numerical experiments indicate that Howard's algorithm is extremely efficient and should be preferred to other algorithms. We also refer to [23] for details on Howard's algorithm for higher-order descriptions. Howard's algorithm is named for the author of [54]. See [33] for a convergence proof.

All three authors of this book were involved in a European project called Alapedes (the ALgebraic Approach to Performance Evaluation of Discrete Event Systems). At one of the Alapedes meetings, in Waterford, Ireland, in 1997, another Alapedes member, Stéphane Gaubert, showed for the first time the power of Howard's algorithm on his laptop. The program calculated in virtually no time the eigenmodes of very large matrices (number of rows and columns in the thousands). We all were greatly impressed. See `www.maxplus.org` under `scilab` for information on the software concerned.