Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

FAST VERIFICATION FOR THE PERRON PAIR OF AN IRREDUCIBLE NONNEGATIVE MATRIX*

SHINYA MIYAJIMA†

Abstract. Fast algorithms are proposed for calculating error bounds for a numerically computed Perron root and vector of an irreducible nonnegative matrix. Emphasis is put on the computational efficiency of these algorithms. Error bounds for the root and vector are based on the Collatz–Wielandt theorem, and estimating a solution of a linear system whose coefficient matrix is an M-matrix, respectively. We introduce a technique for obtaining better error bounds. Numerical results show properties of the algorithms.

Key words. Perron pair, Irreducible nonnegative matrix, M-matrix, Verified computation.

AMS subject classifications. 15A18, 15B48, 65F15, 65G20.

1. Introduction.

For $v \in \mathbb{R}^n$ and $M \in \mathbb{R}^{n\times n}$, let v_i and M_{ij} be the ith component and (i,j) element of v and M, respectively. For $v, w \in \mathbb{R}^n$ and $M, N \in \mathbb{R}^{n\times n}$, let $v \leq w$ and $M \leq N$ denote $v_i \leq w_i$ for all i and $M_{ij} \leq N_{ij}$ for all i, j, respectively. We say M is nonnegative if $M \geq 0$. We also say $A \in \mathbb{R}^{n\times n}$ is reducible if and only if for some permutation matrix P, the matrix P^TAP is block upper triangular. If A is not reducible, we say A is irreducible. The following, called the Perron–Frobenius theorem, states a well-known property of irreducible nonnegative matrices.

THEOREM 1 (Frobenius [1, 2]). *Let $A \in \mathbb{R}^{n\times n}$ be irreducible and nonnegative, $\rho(A)$ be the spectral radius of A, and λ_i, $i = 1, \ldots, n$ be eigenvalues of A such that $|\lambda_1| \geq \cdots \geq |\lambda_n|$. Then, the following are true:*

(a) $\lambda_1 = \rho(A) > 0$.
(b) *There exists an eigenvector x^* corresponding to λ_1 such that $x^* > 0$.*
(c) *If $A \geq B \geq 0$ for $B \in \mathbb{R}^{n\times n}$, then $\lambda_1 \geq \rho(B)$ holds, with equality holding if and only if $A = B$.*
(d) *The algebraic multiplicity of λ_1 is one.*

The positive eigenvalue λ_1 and corresponding positive eigenvector x^* are called the Perron root and vector, respectively, and we call the pair (λ_1, x^*) the Perron pair. In this paper, we are concerned with the accuracy of a numerically computed Perron pair of a nonnegative irreducible matrix.

Perron pairs have many applications. For example, the Perron pair is required in finite-state Markov chains such as branching processes [3] and Markov rewards processes with exponential utility [4]. In information theory, Stanczak and Boche [5] studied the relationship between Kullback–Leibler distance and a Perron root and used it to develop power control algorithms for providing a desired quality of service. Mathematical analysis with respect to a Perron root and vector has been extensively studied (see, e.g., [6, 7, 8]).

*Received by the editors on January 15, 2020. Accepted for publication on May 2, 2021. Handling Editor: Dario Bini.
†Faculty of Science and Engineering, Iwate University, Morioka, 020-8551, Japan (miyajima@iwate-u.ac.jp). This work was partially supported by JSPS KAKENHI Grant Numbers JP16K05270, JP21K03363, and the Research Institute for Mathematical Sciences, a Joint Usage/Research Center located in Kyoto University.

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

The work presented herein addresses the problem of verified computations for λ_1 and x^*, specifically, computing rigorous bounds for λ_1 and x^* based on approximations. There are many verification methods for eigenvalues and eigenvectors. For all eigenvalues, see [9, 10, 11, 12, 13, 14, 15]. For all eigenpairs, see [16, 17, 18]. For a few specified eigenvalues, see, e.g., [19, 20, 21, 22, 23, 24, 25]. For a few specified eigenvectors, see [26]. For computing bounds for λ_1 and x^*, we can apply verification methods for a few specified eigenpairs (e.g., [27, 28, 29, 30, 31, 32, 33, 34]), which require $\mathcal{O}(n^3)$ operations in general. Only Nagato and Ishii [23] have proposed a verification algorithm designed specifically for the Perron root of a nonnegative matrix. On the other hand, it does not compute a bound for the Perron vector.

The purpose of the present paper is to propose two verification algorithms for the Perron pair of an irreducible nonnegative matrix. Emphasis is put on the computational efficiency of these algorithms. Let $\tilde{\lambda}$ and $\tilde{x}$ be numerical results for λ_1 and x^*, respectively. We show that a subvector of $x^* - \tilde{x}$ can be regarded as the solution of a linear system with M-matrix (see Section 4). Verification algorithms for the solution of a linear system with M-matrix have already been proposed (e.g., [35, 36, 37, 38]). However, herein is reported the first attempt at a verification algorithm for an *eigenvector* that exploits the M-matrix property. We introduce a technique for improving the accuracy of $\tilde{\lambda}$ and $\tilde{x}$, reducing error bounds (see Sections 3, 5, and 6).

This paper is organized as follows: Section 2 introduces the notation and two necessary theorems used in this paper. Section 3 discusses methods for computing $\tilde{\lambda}$ and $\tilde{x}$ and improving their accuracy. Section 4 establishes a theory for computing error bounds and proposes the first algorithm. Section 5 proposes a second algorithm that uses the improved approximations and gives better error bounds. Section 6 reports numerical results. Finally, Section 7 highlights future work.

2. Preliminaries. For $M \in \mathbb{R}^{n\times n}$, let $\rho(M)$ be the spectral radius of M, and $|M| := (|M_{ij}|)$. Denote the principal submatrix of M consisting of rows and columns in $\mu \subseteq \{1, \ldots, n\}$ by $M[\mu]$. Analogously, let $x[\mu]$ be the subvector of $x \in \mathbb{R}^n$ which consists of components in μ. For $v \in \mathbb{R}^n$, let $\min(v) := \min_i(v_i)$, $\max(v) := \max_i(v_i)$, and $|v| := [|v_1|, \ldots, |v_n|]^T$. Let $\mathbb{R}_{\geq 0} := [0, \infty)$, $\mathbb{R}_{>0} := (0, \infty)$, $\mathbb{R}^n_{\geq 0} := \{v \in \mathbb{R}^n : v \geq 0\}$, $\mathbb{R}^n_{>0} := \{v \in \mathbb{R}^n : v > 0\}$, $\mathbb{R}^{n\times n}_{\geq 0} := \{M \in \mathbb{R}^{n\times n} : M \geq 0\}$, and $\mathbb{F}$ be the set of all floating-point real numbers. Let I be the identity matrix with compatible size, $\circ$ and $./$ be element-wise multiplication and division, respectively, and $\mathbb{1} := [1, \ldots, 1]^T$ with proper dimension. For $c \in \mathbb{R}^n$ and $r \in \mathbb{R}^n_{\geq 0}$, let $\langle c, r\rangle$ denote the real interval vector whose midpoint and radius are c and r, respectively. We write $\mathrm{fl}(\cdot)$, $\mathrm{fl}_{\triangle}(\cdot)$, and $\mathrm{fl}_{\triangledown}(\cdot)$ to denote results of floating-point computations, where all the operations inside the parentheses are executed using the to-the-nearest, toward -+∞, and toward --∞ rounding modes, respectively. Let `realmin` be the smallest positive normalized floating-point number (specifically `realmin` $= 2^{-1022}$ in IEEE 754 double precision). A real square matrix A is called a Z-matrix if $A_{ij} \leq 0$ for all $i \neq j$. It is clear that any Z-matrix can be written as $\mu I - B$ with $B \geq 0$. A Z-matrix $\mu I - B$ is called an M-matrix if $\mu > \rho(B)$. We say A has the "M-property" if A is an M-matrix.

In the proof of Theorem 5, we will apply Theorem 2, which is called the Collatz–Wielandt theorem.

THEOREM 2 (Collatz [39] and Wielandt [40]). *Let $A \in \mathbb{R}^{n\times n}_{\geq 0}$ and $x \in \mathbb{R}^n_{>0}$. It then follows that* $\min((Ax)./x) \leq \rho(A) \leq \max((Ax)./x)$.

In Section 4, we will use the following properties of an M-matrix.

THEOREM 3 (Fiedler and Pták [41]). *For a Z-matrix A, the following are equivalent:*

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

1. *A is an M-matrix.*
2. *A is nonsingular and $A^{-1} \geq 0$.*
3. *There exists $v \in \mathbb{R}^n_{>0}$ satisfying $Av > 0$.*

3. Computing an approximation and improving its accuracy. Let λ_1, x^*, $\tilde{\lambda}$, and $\tilde{x}$ be as in Section 1. We are interested in verified bounds for (λ_1, x^*) based on $(\tilde{\lambda}, \tilde{x})$; however, we want to add some remarks on how to obtain $(\tilde{\lambda}, \tilde{x})$. In Section 3.1, we thus discuss methods for computing $(\tilde{\lambda}, \tilde{x})$ and present a procedure based on the discussion. Section 3.2 introduces a method for improving its accuracy.

3.1. Computing an approximation. Let λ_i be as in Theorem 1, i.e., $|\lambda_1| \geq \cdots \geq |\lambda_n|$. If the cyclic index of A is one, then $|\lambda_2|/\rho(A) < 1$ follows, so that we can compute $(\tilde{\lambda}, \tilde{x})$ via the power method $x^{(k+1)} = Ax^{(k)}$, $k = 0, 1, \ldots$, where $x^{(0)} \in \mathbb{R}^n_{>0}$ is an initial guess. On the other hand, if the cyclic index of A is two or more, then $|\lambda_2|/\rho(A) = 1$, which means the power method does not work. One may consider that the issue can be overcome by introducing a shift. However, introducing the shift is not generally working.

Therefore, using the power method only seems to be problematic. A stable way for computing eigenvalues and eigenvectors in MATLAB is to call the routine `eig`. On the other hand, `eig` computes all eigenvalues and eigenvectors, requiring large CPU times when n is large, though we need only the Perron pair. In 2007, an iterative algorithm for computing $(\tilde{\lambda}, \tilde{x})$ has been proposed in [42]. Although it requires $\mathcal{O}(n^3)$ operations per iteration, it is reported in [42, Section 3.5] that the iteration converges even in the problem where the power method does not converge. However, there is a case where the iteration does not converge (see Example 5 in Section 6).

We thus propose the following procedure:

1. Execute the power method. If it converged, then terminate.
2. Execute the iteration in [42]. If it converged, then terminate.
3. Call `eig`.

Let $\widehat{x} \in \mathbb{R}^n$ be a numerical result for an eigenvector corresponding to $\rho(A)$. Then, $\widehat{x} < 0$ may occur. In such a case, we must put $\tilde{x} = -\widehat{x}$.

3.2. Improving the accuracy. Let $k \in \{1, \ldots, n\}$ satisfy $\tilde{x}_k = \max(\tilde{x})$, and $x^* \in \mathbb{R}^n_{>0}$ be the Perron vector of A such that $x^*_k = \tilde{x}_k$. Hereafter, in this paper, let $e \in \mathbb{R}^n$ be the kth column of I and $U \in \mathbb{R}^{(n-1)\times n}$ be I without the kth row. It is possible to improve the accuracy of $\tilde{\lambda}$ and $\tilde{x}$ via a Newton method applied to a nonlinear system derived from $(\rho(A)I - A)x^* = 0$. This technique is a subset of [34, Algorithm 2.1]. Since $x^*_k = \tilde{x}_k$, we correct the unknown quantities Ux^* and $\rho(A)$ into $y \in \mathbb{R}^n$, i.e., we consider finding y with $y_k = \rho(A)$ and $Uy = Ux^*$. Let $\tilde{y} \in \mathbb{R}^n_{>0}$ satisfy $U\tilde{y} = U\tilde{x}$ and $\tilde{y}_k = \tilde{\lambda}$, and $r := A\tilde{x} - \tilde{\lambda}\tilde{x}$. Suppose that $\tilde{x}e^T - (A - \tilde{\lambda}I)U^TU$ is nonsingular. From the analysis in [34], a correction term for $\tilde{y}$ obtained by the Newton method is $(\tilde{x}e^T - (A - \tilde{\lambda}I)U^TU)^{-1}r$, which is the solution $z^* \in \mathbb{R}^n$ to the linear system

$$(\tilde{x}e^T - (A - \tilde{\lambda}I)U^TU)z^* = r. \tag{1}$$

We cannot assert that $\tilde{x}e^T - (A - \tilde{\lambda}I)U^TU$ is nonsingular. As Theorem 4 implies, however, we can expect nonsingularity if $\tilde{\lambda}$ and $\tilde{x}$ are not too far from $\rho(A)$ and x^*, respectively.

THEOREM 4. *Let $A \in \mathbb{R}^{n\times n}_{\geq 0}$ be irreducible and x^* be a Perron vector of A. Then, $x^*e^T - (A - \rho(A)I)U^TU$ is nonsingular for any $k \in \{1, \ldots, n\}$.*

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

Proof. Let $C := A-\rho(A)I$ and $C_{[i,j]}$ be the (i,j) cofactor of C. From Theorem 1 (d), we have $\mathrm{rank}(C) = n-1$, so that $\mathrm{rank}(\mathrm{adj}(C)) = 1$. Therefore, there exist $\alpha \in \mathbb{R}\setminus\{0\}$ and a Perron vector $v \in \mathbb{R}^n_{>0}$ of A^T such that $\mathrm{adj}(C) = x^*(\alpha v)^T$, which gives $\det(x^*e^T - CU^TU) = (-1)^{n-1}\sum_{i=1}^n x_i^* C_{[i,k]} = (-1)^{n-1}\alpha x_k^*(v^Tx^*) \neq 0$. □

We numerically solve (1) and obtain an approximate solution $z \in \mathbb{R}^n$. Then, we can expect $\tilde{\lambda} + z_k$ and $\tilde{x} + U^TUz$ are more accurate than $\tilde{\lambda}$ and $\tilde{x}$, respectively. In order to obtain effective z, it is necessary to compute r in extended precision.

4. Verification theory. Let $\tilde{\lambda}$ and $\tilde{x}$ be as in Section 1. We present Theorem 5, which gives an error bound for $\tilde{\lambda}$.

THEOREM 5. *Let $A \in \mathbb{R}^{n\times n}_{\geq 0}$, $\tilde{x} \in \mathbb{R}^n_{>0}$, $\tilde{\lambda} \in \mathbb{R}_{>0}$, $r := A\tilde{x} - \tilde{\lambda}\tilde{x}$, and $\varepsilon := \max(|r|./\tilde{x})$. It then follows that $|\tilde{\lambda} - \rho(A)| \leq \varepsilon$.*

Proof. Define $y := (A\tilde{x})./\tilde{x}$. Then, $r_i = (y_i - \tilde{\lambda})\tilde{x}_i$, so that $|y_i - \tilde{\lambda}| \leq \max_j |r_j|/|\tilde{x}_j| = \varepsilon$ for all i. Theorem 2 moreover shows $\min(y) \leq \rho(A) \leq \max(y)$. Hence, $|\tilde{\lambda} - \rho(A)| \leq \max_j |y_j - \tilde{\lambda}| \leq \varepsilon$. □

REMARK 6. Let $\overline{\lambda} := \max(y)$, $\underline{\lambda} := \min(y)$, $\omega := (\overline{\lambda} + \underline{\lambda})/2$, $\delta := (\overline{\lambda} - \underline{\lambda})/2$, $\lambda_\triangle := \mathrm{fl}_\triangle((\overline{\lambda} + \underline{\lambda})/2)$, $\lambda_\triangledown := \mathrm{fl}_\triangledown((\overline{\lambda}+\underline{\lambda})/2)$, $\delta_\triangle := \mathrm{fl}_\triangle(\lambda_\triangle - \underline{\lambda})$, and $\delta_\triangledown := \mathrm{fl}_\triangle(\overline{\lambda} - \lambda_\triangledown)$. From Theorem 2, we have $\rho(A) \in \langle \lambda_\triangle, \delta_\triangle\rangle$ and $\rho(A) \in \langle \lambda_\triangledown, \delta_\triangledown\rangle$, so that

$$\rho(A) \in \langle \lambda^*, \delta^*\rangle, \quad \text{where } \lambda^* := \begin{cases} \lambda_\triangle \ (\text{if } \delta_\triangle \leq \delta_\triangledown) \\ \lambda_\triangledown \ (\text{otherwise}) \end{cases}, \text{ and } \delta^* := \min(\delta_\triangle, \delta_\triangledown).$$

This is an alternative way for enclosing $\rho(A)$. Moreover, $\delta^* \leq \varepsilon$ follows if $\underline{\lambda}/2 \leq \overline{\lambda} \leq 2\underline{\lambda}$; otherwise the enclosure is wide. Its proof given in the following is due to the reviewer. If the assumption is true, then Sterbenz' lemma implies $\overline{\lambda} - \underline{\lambda} \in \mathbb{F}$ and $\delta = \mathrm{fl}((\overline{\lambda} - \underline{\lambda})/2) \in \mathbb{F}$. We distinguish two cases.

1. If $\overline{\lambda} + \underline{\lambda} \in \mathbb{F}$, then $\lambda^* = \lambda_\triangle = \lambda_\triangledown = \omega \in \mathbb{F}$, and $\delta^* = \delta_\triangle = \delta_\triangledown = \delta$ implies
$$\varepsilon - \delta^* = \max_i |y_i - \tilde{\lambda}| - \delta = \max(\overline{\lambda} - \tilde{\lambda}, \tilde{\lambda} - \underline{\lambda}) - \delta = |\omega - \tilde{\lambda}| \geq 0.$$
2. If $\overline{\lambda} + \underline{\lambda} \notin \mathbb{F}$, then ω is the midpoint of two adjacent floating-point numbers. Denote by η the difference to its neighbors. Then, $\lambda_\triangle = \omega + \eta$ and $\lambda_\triangledown = \omega - \eta$.
   (a) If $\tilde{\lambda} \geq \omega$, then $\tilde{\lambda} \in \mathbb{F}$ implies $\tilde{\lambda} \geq \lambda_\triangle$, $\delta_\triangle = \lambda_\triangle - \underline{\lambda} = \delta + \eta \in \mathbb{F}$, and
   $$\varepsilon - \delta^* \geq \varepsilon - \delta_\triangle = \max(\overline{\lambda} - \tilde{\lambda}, \tilde{\lambda} - \underline{\lambda}) - \delta - \eta = \tilde{\lambda} - (\underline{\lambda} + \delta + \eta) = \tilde{\lambda} - \lambda_\triangle \geq 0.$$
   (b) If $\tilde{\lambda} < \omega$, then, similarly, $\tilde{\lambda} \leq \lambda_\triangledown$, $\delta_\triangledown = \overline{\lambda} - \lambda_\triangledown = \delta + \eta \in \mathbb{F}$, and $\varepsilon - \delta^* \geq \varepsilon - \delta_\triangledown = \overline{\lambda} - (\tilde{\lambda} + \delta + \eta) = \lambda_\triangledown - \tilde{\lambda} \geq 0$.

In Algorithm 11, therefore, we compute δ^* as an error bound.

One may consider that computing ε is unnecessary. When we use an accurate dot product algorithm (e.g., the INTLAB [43] routine `Dot_`), however, there is a case where a numerically obtained ε is smaller than a numerically obtained δ^*. This is because ε contains the residual r, and $|r_i|$, $i = 1, \ldots, n$ are small quantities computed from relatively large quantities. Therefore, calculating r with the accurate dot product algorithm gives smaller $|r_i|$. In contrast, the quantities $\underline{\lambda}$ and $\overline{\lambda}$ are not small in general, so that calculating them with the accurate algorithm is not advantageous. Although δ^* is a small quantity computed from relatively large quantities, we can compute it only after the calculations of $\underline{\lambda}$ and $\overline{\lambda}$ are completed. When the

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

calculations of $\underline{\lambda}$ and $\overline{\lambda}$ are completed, the computation of δ^* involves only few operations. Consequently, it is difficult to make the accurate dot product algorithm advantageous in the calculation of δ^*. Let $\hat{\varepsilon}$ be ε obtained via the accurate algorithm. We thus design Algorithm 12 such that the minimum of δ^* and $\hat{\varepsilon}$ is adopted as an error bound.

We next consider computing an error bound for $\tilde{x}$. Let $k \in \{1, \ldots, n\}$ satisfy $\tilde{x}_k = \max(\tilde{x})$, $x^* \in \mathbb{R}^n_{>0}$ be a Perron vector of A such that $x^*_k = \tilde{x}_k$, and $\mu := \{1, \ldots, n\} \setminus \{k\}$. Since U^TU coincides with I except $(U^TU)_{kk} = 0$, we have $x^* = \tilde{x}_k e + U^TUx^*$ and $\tilde{x} = \tilde{x}_k e + U^TU\tilde{x}$, so that $x^* - \tilde{x} = U^TU(x^* - \tilde{x})$. Thus, we estimate an upper bound for $|U(x^* - \tilde{x})| = |(x^* - \tilde{x})[\mu]|$.

For the estimation, we give a representation for $(x^* - \tilde{x})[\mu]$. Note that $U^TU(x^* - \tilde{x}) = x^* - \tilde{x}$ and $A[\mu] = UAU^T$. Using $IU = U$ and $Ax^* = \rho(A)x^*$ implies

$$(\rho(A)I - A[\mu])U(x^* - \tilde{x}) = \rho(A)U(x^* - \tilde{x}) - UA(x^* - \tilde{x}) = U(A\tilde{x} - \rho(A)\tilde{x}).$$

Therefore, if $\rho(A)I - A[\mu]$ is nonsingular, then we obtain the representation

$$(x^* - \tilde{x})[\mu] = U(x^* - \tilde{x}) = (\rho(A)I - A[\mu])^{-1}U(A\tilde{x} - \rho(A)\tilde{x}). \tag{2}$$

In order to show nonsingularity, we prove that $\rho(A)I - A[\mu]$ is an M-matrix (see Theorem 3). The M-property of $\rho(A)I - A[\mu]$ enables us to calculate the upper bound for $|(x^* - \tilde{x})[\mu]|$ at low computational cost (see Lemma 8).

LEMMA 7. *Let $k \in \{1, \ldots, n\}$, $\mu := \{1, \ldots, n\} \setminus \{k\}$, and $A \in \mathbb{R}^{n\times n}_{\geq 0}$ be irreducible. Then, $\rho(A)I - A[\mu]$ is an M-matrix.*

Proof. It is obvious that $\rho(A)I - A[\mu]$ is a Z-matrix. Since A is irreducible, moreover, we have $\rho(A[\mu]) < \rho(A)$, which completes the proof. ∎

Let $\underline{\lambda} \in \mathbb{R}_{>0}$ satisfy $\underline{\lambda} \leq \rho(A)$. If we have $v \in \mathbb{R}^{n-1}_{>0}$ such that $(\underline{\lambda}I - A[\mu])v > 0$, then the upper bound for $|(x^* - \tilde{x})[\mu]|$ can be computed efficiently. Lemma 8 is an analogue of the results in [35, 36, 37, 38].

LEMMA 8. *Let $A \in \mathbb{R}^{n\times n}_{\geq 0}$ be irreducible, $\tilde{\lambda} \in \mathbb{R}_{>0}$, $\tilde{x} \in \mathbb{R}^n_{>0}$, $v, w \in \mathbb{R}^{n-1}_{>0}$, and $k \in \{1, \ldots, n\}$. Define $r := A\tilde{x} - \tilde{\lambda}\tilde{x}$, and $\mu := \{1, \ldots, n\} \setminus \{k\}$. Let also $x^* \in \mathbb{R}^n_{>0}$ be a Perron vector of A such that $x^*_k = \tilde{x}_k$. Suppose $\underline{\lambda} \in \mathbb{R}_{>0}$ and $\varepsilon \in \mathbb{R}_{\geq 0}$ satisfy $\underline{\lambda} \leq \rho(A)$ and $\varepsilon \geq |\tilde{\lambda} - \rho(A)|$, respectively, and $w \leq (\underline{\lambda}I - A[\mu])v$. Define $s := (|r| + \varepsilon\tilde{x})[\mu]$, and $t := (s + \max(s./w)A[\mu]v)/\underline{\lambda}$. It then follows that $|(x^* - \tilde{x})[\mu]| \leq t$.*

Proof. Let $D := \rho(A)I$ and $E := A[\mu]$. Theorem 3 and Lemma 7 show

$$|(D - E)^{-1}| = (D - E)^{-1} = D^{-1}(I + E(D - E)^{-1}). \tag{3}$$

From $\varepsilon \geq |\tilde{\lambda} - \rho(A)|$, it follows that

$$|U(A\tilde{x} - \rho(A)\tilde{x})| = U|r + (\tilde{\lambda} - \rho(A))\tilde{x}| \leq U(|r| + |\tilde{\lambda} - \rho(A)||\tilde{x}|) \leq U(|r| + \varepsilon\tilde{x}) = s. \tag{4}$$

From $\underline{\lambda} \leq \rho(A)$ and $w \leq (\underline{\lambda}I - E)v$, we obtain

$$(D - E)^{-1}w \leq (D - E)^{-1}(\underline{\lambda}I - E)v \leq (D - E)^{-1}(D - E)v = v. \tag{5}$$

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

The relations (2) to (5) and $D^{-1} \leq (1/\underline{\lambda})I$ prove

$$\begin{aligned}|(x^* - \tilde{x})[\mu]| &\leq |(D-E)^{-1}||U(A\tilde{x} - \rho(A)\tilde{x})| \leq D^{-1}(I + E(D-E)^{-1})s \\ &= D^{-1}(s + E(D-E)^{-1}(s./w \circ w)) \\ &\leq D^{-1}(s + \max(s./w)E(D-E)^{-1}w) \leq D^{-1}(s + \max(s./w)Ev) \leq t. \end{aligned}$$

□

Lemma 8 gives an error bound for $\tilde{x}$.

THEOREM 9. *Let $\tilde{x}$, x^*, and t be as in Lemma 8, and $u := U^T t$. If the assumptions in Lemma 8 are true, then $|x^* - \tilde{x}| \leq u$.*

Proof. As mentioned above, $x^* - \tilde{x} = U^T U(x^* - \tilde{x})$ follows, so that $|x^* - \tilde{x}| = U^T|(x^* - \tilde{x})[\mu]|$. This and Lemma 8 prove the result. □

Algorithm 11 computes λ^* and δ^* in Remark 6 and an upper bound on u. To obtain the upper bounds on u, we need to compute v and w in Lemma 8. We can obtain v by the following procedure:

1. Numerically solve the linear system $(\underline{\lambda}I - A[\mu])v^* = b$, where $b \in \mathbb{R}^{n-1}_{>0}$, and v^* is the solution, via an iterative method. Let v be the numerical solution.
2. Update v such that $v_i = \max(v_i, \texttt{realmin})$, $i = 1, \ldots, n-1$.

If v is not far from v^*, then it approximately follows that

$$(\rho(A)I - A[\mu])v \;\geq\; (\underline{\lambda}I - A[\mu])v \approx b > 0. \tag{6}$$

The choice of b is arbitrary, and a natural one is $b = \mathbb{1}$. In Algorithm 11, we adopt the natural choice. If $\underline{\lambda}I - A[\mu]$ is well-conditioned, then the iterative method stops after few iterations, so that v can be obtained efficiently.

If we have v, then we can obtain w by the following procedure:

1. Compute a lower bound on $(\underline{\lambda}I - A[\mu])v$. Let w be the result.
2. Check $w > 0$.

We then have $w \leq (\underline{\lambda}I - A[\mu])v$. If v is not far from v^*, then we can expect $w > 0$ from (6). However, a case exists that the iterative method gives inaccurate v, so that $w > 0$ does not follow. In such a case, we compute v again via a direct method, update w, and check $w > 0$ once more.

REMARK 10. If the leading eigenvalues are close together, the case $\underline{\lambda} < \rho(A[\mu])$ may arise. If so, $\underline{\lambda}I - A[\mu]$ is not an M-matrix, so there does not exist $v \in \mathbb{R}^{n-1}_{>0}$ which satisfies $(\underline{\lambda}I - A[\mu])v > 0$ (see Theorem 3). Consequently, Algorithm 11 does not work. A similar remark applies to Algorithm 12.

Before computing λ^*, δ^*, and an upper bound on u, we need to verify that A is irreducible. If $A > 0$, then the irreducibility is obvious. Otherwise, the irreducibility can be verified by some standard graph algorithm (e.g., [44]).

Based on the discussion above, we propose Algorithm 11.

ALGORITHM 11. *Let $\tilde{x} \in \mathbb{R}^n_{>0}$ be as in Section 1, $v, w \in \mathbb{R}^{n-1}_{>0}$, and $\underline{\lambda}$, λ^*, and δ^* be as in Remark 6. Let also $k \in \{1, \ldots, n\}$ satisfy $\tilde{x}_k = \max(\tilde{x})$, and $x^* \in \mathbb{R}^n_{>0}$ be the Perron vector of A such that $x^*_k = \tilde{x}_k$. Define $r := A\tilde{x} - \lambda^*\tilde{x}$, $\mu := \{1, \ldots, n\} \setminus \{k\}$, $s := (|r| + \delta^*\tilde{x})[\mu]$, $t := (s + \max(s./w)A[\mu]v)/\underline{\lambda}$, and $u := U^T t$. This*

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

algorithm computes $\lambda^* \in \mathbb{R}_{>0}$, $\delta^* \in \mathbb{R}_{\geq 0}$, *and* $\overline{u} \in \mathbb{R}^n_{\geq 0}$ *such that* $\delta^* \geq |\lambda^* - \rho(A)|$ *and* $\overline{u} \geq |\tilde{x} - x^*|$. *The irreducibility of* A *is moreover proved if successful.*

Step 1. *If* $A > 0$, *then go to Step* 3. *Otherwise, go to Step* 2.
Step 2. *Verify the irreducibility of* A *via a standard graph algorithm. If it cannot be verified, then terminate with failure.*
Step 3. *Calculate* λ^* *and* δ^*.
Step 4. *Compute* v *by numerically solving* $(\underline{\lambda} I - A[\mu])v^* = \mathbb{1}$ *via an iterative method. Update* v *such that* $v_i = \max(v_i, \texttt{realmin})$, $i = 1, \ldots, n-1$.
Step 5. *Compute a lower bound on* $(\underline{\lambda} I - A[\mu])v$. *Let* w *be the result. If* $w > 0$ *cannot be verified, then compute* v *again via a direct method, update* w, *and check* $w > 0$ *once more. If* $w > 0$ *cannot be verified again, then the computation of* $\overline{u}$ *fails. Terminate.*
Step 6. *Compute an upper bound on* u. *Let* $\overline{u}$ *be the result. Terminate.*

If $\rho(A[\mu]) < \underline{\lambda}$, then the Neumann series gives

$$v^* = (\underline{\lambda} I - A[\mu])^{-1}\mathbb{1} = \frac{1}{\underline{\lambda}}\left(I + \frac{1}{\underline{\lambda}}A[\mu] + \frac{1}{\underline{\lambda}^2}(A[\mu])^2 + \cdots\right)\mathbb{1}. \tag{7}$$

We thus use $\mathrm{fl}((\mathbb{1} + A[\mu]\mathbb{1}/\underline{\lambda})/\underline{\lambda})$ as an initial guess of the iterative method in Step 4.

5. A technique for obtaining better error bounds. We can obtain better error bounds by using the improved approximation mentioned in Section 3.2. Let $\tilde{\lambda}$, $\tilde{x}$, k, x^*, r, and z be as in Section 3.2 and ε and u be as in Theorems 5 and 9, respectively. As mentioned in Section 3.2, we can expect $\tilde{\lambda} + z_k$ and $\tilde{x} + U^T U z$ to be more accurate than $\tilde{\lambda}$ and $\tilde{x}$, respectively. We store z independently from $\tilde{\lambda}$ and $\tilde{x}$. Then, $\tilde{\lambda} + z_k$ and $\tilde{x} + U^T U z$ are stored as if in multiple-precision floating-point numbers. Let $\hat{r}$, $\hat{\varepsilon}$, and $\hat{u}$ be r, ε, and u, respectively, where $\tilde{\lambda}$ and $\tilde{x}$ are replaced by $\tilde{\lambda} + z_k$ and $\tilde{x} + U^T U z$, respectively. We compute an upper bound on $|\hat{r}|$ executing an accurate dot product algorithm. In the first stage, we obtain z via an iterative method to reduce computational cost. However, a case exists for which the iterative method gives inaccurate z. This case occurs when the coefficient matrix of (1) is ill-conditioned. Consequently, we cannot verify $\min(|r| - |\hat{r}|) \geq 0$. In such a case, we compute z again via a direct method.

If $\tilde{\lambda} + z_k > 0$ and $\tilde{x} + U^T U z > 0$, then Theorems 5 and 9 applied to $\tilde{\lambda} := \tilde{\lambda} + z_k$ and $\tilde{x} := \tilde{x} + U^T U z$ give $|\tilde{\lambda} + z_k - \rho(A)| \leq \hat{\varepsilon}$ and $|x^* - (\tilde{x} + U^T U z)| \leq \hat{u}$. Let $\overline{\varepsilon}$ and $\overline{u}$ be upper bounds on $\hat{\varepsilon}$ and $\hat{u}$, respectively. Define $\lambda_I := \mathrm{fl}_\nabla(\tilde{\lambda} + z_k - \overline{\varepsilon})$, $\lambda_S := \mathrm{fl}_\Delta(\tilde{\lambda} + z_k + \overline{\varepsilon})$, $x_I := \mathrm{fl}_\nabla(\tilde{x} + U^T U z - \overline{u})$, $x_S := \mathrm{fl}_\Delta(\tilde{x} + U^T U z + \overline{u})$, $\lambda_C := \mathrm{fl}_\Delta((\lambda_I + \lambda_S)/2)$, $\varepsilon_C := \mathrm{fl}_\Delta(\lambda_C - \lambda_I)$, $x_C := \mathrm{fl}_\Delta((x_I + x_S)/2)$, and $u_C := \mathrm{fl}_\Delta(x_C - x_I)$. Then, $|\lambda_C - \rho(A)| \leq \varepsilon_C$ and $|x_C - x^*| \leq u_C$ follows. Since $\lambda_I \leq \rho(A)$, we update $\underline{\lambda}$ in Remark 6 such that $\underline{\lambda} = \lambda_I$ if $\lambda_I \geq \underline{\lambda}$, which occurs when $\overline{\varepsilon}$ is small (cf. Remark 6).

We summarize our strategy in Algorithm 12.

ALGORITHM 12. *Let* $\tilde{\lambda}$, $\tilde{x}$, k, x^*, r, *and* z *be as in Section* 3.2, $\underline{\lambda}$, λ^*, *and* δ^* *be as in Remark* 6, *and* ε *and* u *be as in Theorems* 5 *and* 9, *respectively. Let also* $\hat{r}$, $\hat{\varepsilon}$, *and* $\hat{u}$ *be* r, ε, *and* u, *respectively, where* $\tilde{\lambda}$ *and* $\tilde{x}$ *are replaced by* $\tilde{\lambda} + z_k$ *and* $\tilde{x} + U^T U z$, *respectively. This algorithm computes* $\lambda_C \in \mathbb{R}_{>0}$, $\varepsilon_C \in \mathbb{R}_{\geq 0}$, $x_C \in \mathbb{R}^n_{>0}$, *and* $u_C \in \mathbb{R}^n_{\geq 0}$ *such that* $|\lambda_C - \rho(A)| \leq \varepsilon_C$ *and* $|x_C - x^*| \leq u_C$. *The irreducibility of* A *is moreover proved if successful.*

Steps 1 and 2. *Similar to those in Algorithm* 11.

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

Step 3. *Calculate r by executing an accurate dot product algorithm. Compute z via an iterative method. Calculate an upper bound on $|\hat{r}|$ utilizing the accurate dot product algorithm. If* $\min(|r| - |\hat{r}|) \geq 0$ *cannot be verified, then compute z again via a direct method and calculate an upper bound on $|\hat{r}|$ again utilizing the accurate algorithm.*

Step 4. *If $\tilde{\lambda} + z_k > 0$ cannot be verified, then update z_k such that $z_k = 0$. If the positiveness of some components in $\tilde{x} + U^T U z$ cannot be verified, update the corresponding components of z similarly.*

Step 5. *Calculate λ^* and δ^*. Compute an upper bound on $\hat{\varepsilon}$. Let $\overline{\varepsilon}$ be the result. If $\delta^* \leq \overline{\varepsilon}$, then update $\tilde{\lambda}$, z_k, and $\overline{\varepsilon}$ such that $\tilde{\lambda} = \lambda^*$, $z_k = 0$, and $\overline{\varepsilon} = \delta^*$, respectively (see Remark 6). Calculate λ_I, λ_C, and ε_C as in the previous paragraph. Update $\underline{\lambda}$ such that $\underline{\lambda} = \max(\underline{\lambda}, \lambda_I)$.*

Steps 6 and 7. *Similar to Steps 4 and 5 in Algorithm 11, respectively.*

Step 8. *Compute an upper bound on $\hat{u}$. Let $\overline{u}$ be the result. Calculate x_C and u_C as in the previous paragraph. Terminate.*

6. Numerical results. We used a computer with Intel Core 1.51 GHz CPU, 16.0 GB RAM, and MATLAB R2012a with the Intel Math Kernel Library and IEEE 754 double precision. We denote the compared algorithms as follows:

`M1:` combination of the procedure in Section 3.1 and Algorithm 11,
`M2:` combination of the procedure and Algorithm 12, and
`V:` combination of the procedure and INTLAB routine `verifyeig`.

The comparison to `verifyeig` is not fair because it is applicable to general real or complex, point or interval matrices, computes inner inclusions, as well as bounds for invariant subspaces. However, lacking an algorithm tailored specifically to the Perron pair, we use `verifyeig`. In Step 2 of `M1` and `M2`, we called `tarjan.m` in https://jp.mathworks.com/matlabcentral/fileexchange/50707-tarjan-e, which executes the algorithm in [44]. In `M1` and `M2`, we adopted the MATLAB routine `bicgstab` as the iterative method. We used $\mathrm{fl}((\mathbb{1} + A[\mu]\mathbb{1}/\underline{\lambda})/\underline{\lambda})$ as the initial guess of `bicgstab` (cf. (7)). For A sparse, we used numerically computed incomplete LU factors of A as a preconditioner in `bicgstab`. In `M2`, the INTLAB routine `Dot_` was invoked as the accurate dot product algorithm. See http://web.cc.iwate-u.ac.jp/~miyajima/PP.zip for details of the implementations, where INTLAB codes of `M1`, `M2`, and `V` (denoted by `M1.m`, `M2.m`, and `V.m`) are uploaded. Let $\langle \tilde{\lambda}, \varepsilon \rangle$ and $\langle \tilde{x}, u \rangle$ contain the Perron root and vector, respectively. To assess qualities of enclosures, define the relative radius for the root (RRR) and relative radius for the vector (RRV) as $\varepsilon/|\tilde{\lambda}|$ and $\|u\|_2/\|\tilde{x}\|_2$, respectively. In some problems, `M1` and/or `M2` failed to compute u. The reason for the failure is that $w > 0$ could not be verified even in the second stage. In Examples 1 to 5, the routine `eig` within the procedure in Section 3.1 was invoked once (see Example 5).

Example 1. We observe the RRR, RRV, and CPU times of the algorithms when $A > 0$. Consider the cases where A is generated by the MATLAB code `A = gallery('cauchy',1:n,2*(1:n));`. In this case, $A > 0$. The 'cauchy' matrix is nonsymmetric. Table 1 displays the RRR, RRV, and CPU times (s) of the algorithms. This table shows that `M1` and `M2` were faster than `V`, and the RRR and RRV by `M2` were smaller than those by `V`.

Remark 13. When we used the 'lotkin' and 'minij' matrices instead, we observed tendencies similar to those in Table 1.

Example 2. We observe properties of the algorithms when $\min_{i,j} A_{ij} = 0$ and A is irreducible. Consider the cases where A is generated by

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

Table 1
RRR, RRV, and CPU times (s) for the 'cauchy' matrix.

n	M1 RRR RRV	M2 RRR RRV	V RRR RRV	M1 Time	M2 Time	V Time
500	4.6e–14 3.3e–12	3.0e–16 3.2e–16	5.6e–15 1.7e–14	0.019	0.15	0.19
1000	9.3e–14 9.2e–12	2.9e–16 3.2e–16	9.4e–15 3.2e–14	0.086	0.36	0.46
3000	2.9e–13 4.7e–11	2.7e–16 3.2e–16	2.2e–14 9.0e–14	0.49	3.0	10
6000	5.9e–13 1.3e–10	2.7e–16 3.2e–16	4.0e–14 1.8e–13	2.3	13	91

```
A = gallery('circul',[0;1;zeros(n-2,1)]);
A = gallery('toeppen',n,2,1,0,1,2);
A = gallery('tridiag',ones(n-1,1),zeros(n,1),[2;ones(n-2,1)]);
A = gallery('frank',n,K);
```

with K being a parameter. The 'circul,' 'tridiag,' and 'frank' matrices are nonsymmetric, whereas the 'toeppen' matrix is symmetric. Tables 2, 3, and 4 report the RRR, RRV, and CPU times (s) for the 'circul,' 'toeppen,' and 'tridiag' matrices. Let κ be an approximate condition number of $\underline{\lambda}I - A[\mu]$ obtained by the routine cond. Table 5 lists κ, and the RRR, RRV, and CPU times (s) of the algorithms for the 'frank' matrix. We see from Table 5 that M1 and M2 failed to enclose the Perron vector in many cases and were slow. The reason for the failure is that a large κ caused an inaccurate v, so that $w > 0$ could not be verified. The large κ also caused the slow execution of M1 and M2 because bicgstab required many iterations. Even when they succeeded, the RRV was large because $\|A[\mu]v/\underline{\lambda}\|_2$ was large. In fact, $\|A[\mu]v/\underline{\lambda}\|_2$ was $\mathcal{O}(10^4)$ and $\mathcal{O}(10^8)$ when $n = 60$ and K $= 0$ and $n = 100$ and K $= 0$, respectively.

Table 2
RRR, RRV, and CPU times (s) for the 'circul' matrix.

n	M1 RRR RRV	M2 RRR RRV	V RRR RRV	M1 Time	M2 Time	V Time
500	7.9e–12 4.6e–9	2.2e–16 4.3e–16	3.3e–16 3.3e–16	0.13	0.15	0.16
1000	2.9e–11 3.3e–8	2.2e–16 4.4e–16	3.3e–16 3.3e–16	0.63	0.66	0.96
2000	2.0e–12 4.7e–9	2.2e–16 4.4e–16	3.3e–16 3.3e–16	3.3	4.3	6.4
5000	4.2e–12 2.4e–8	2.2e–16 1.7e–15	3.3e–16 3.3e–16	44	49	92

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

Table 3
RRR, RRV, and CPU times (s) for the 'toeppen' matrix.

	M1	M2	V	M1	M2	V
n	RRR RRV	RRR RRV	RRR RRV	Time	Time	Time
500	2.1e–11 3.1e–7	3.0e–16 9.4e–15	3.0e–16 6.8e–13	0.10	0.11	0.11
1000	5.9e–16 3.0e–11	3.0e–16 2.7e–16	1.5e–16 2.7e–12	0.42	0.42	0.60
2000	7.4e–16 1.7e–10	3.0e–16 2.7e–16	1.5e–16 1.1e–11	1.9	2.0	4.0
5000	6.5e–12 9.3e–6	3.0e–16 6.7e–13	3.0e–16 6.8e–11	25	26	63

Table 4
RRR, RRV, and CPU times (s) for the 'tridiag' matrix.

	M1	M2	V	M1	M2	V
n	RRR RRV	RRR RRV	RRR RRV	Time	Time	Time
500	5.6e–16 8.5e–11	2.2e–16 2.7e–16	2.2e–16 2.2e–12	0.12	0.13	0.13
1000	3.6e–13 2.5e–7	2.2e–16 2.7e–16	1.1e–16 8.7e–12	0.49	0.51	0.72
2000	1.3e–15 3.7e–9	2.2e–16 1.5e–13	1.1e–16 3.5e–11	2.8	2.7	4.6
5000	1.1e–14 1.9e–7	2.2e–16 5.7e–16	1.1e–16 2.2e–10	37	38	74

Example 3. We consider the case where eigenvalues are closely clustered around $\rho(A)$. We generated A by the code `A = [2 e 1;e 2 1;e e 1];` with `e` being a parameter. Then, two eigenvalues are closely clustered around $\rho(A)$ for `e` close to 0. Let λ_i be defined as in Theorem 1, i.e., $|\lambda_1| \geq \cdots \geq |\lambda_n|$, and $\tilde{\lambda}_1$ and $\tilde{\lambda}_2$ denote numerical results for λ_1 and λ_2, respectively. Table 6 displays $\mathrm{fl}(|\tilde{\lambda}_1 - \tilde{\lambda}_2|)$, and the RRR, RRV, and CPU times (s) for various `e`. We see that `M1` and `M2` are not robust when eigenvalues are closely clustered around $\rho(A)$.

Example 4. We observe properties of the algorithms when $\min(\tilde{x})/\|\tilde{x}\|_2$ is small. Consider the case where A is generated by `n = 10; A = gallery('circul', [0;1; zeros(n-2,1)]); A(n,n) = s;` with `s` being a parameter. As far as we tested, $\min(\tilde{x})/\|\tilde{x}\|_2$ decreased as `s` increased. Table 7 reports $\mathrm{fl}(\min(\tilde{x})/\|\tilde{x}\|_2)$, and the RRR, RRV, and CPU times (s) for various `s`. We see that `M1` and `M2` worked well even when $\min(\tilde{x})/\|\tilde{x}\|_2$ is small.

Example 5. Let $\underline{\lambda}$ be as in Remark 6, and $\underline{\rho} \in \mathbb{F}$ be a lower bound for $\rho(A)$ obtained by `V`. We observe the properties when $\underline{\lambda}$ is not a tight lower bound. Consider the case where A is generated by the INTLAB code `A = band(rand(n),p,p);`. Then, $\underline{\lambda}$ tends to be nontight when n is large, or `p` is small. The nontight

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

Table 5
RRR, RRV, and CPU times (s), and κ for the 'frank' matrix.

n	K	κ	M1 RRR RRV	M2 RRR RRV	V RRR RRV	M1 Time	M2 Time	V Time
60	0	4.9e+6	2.7e–11 2.3e–4	2.7e–16 5.1e–14	2.7e–16 2.7e–16	0.036	0.049	0.010
60	1	3.3e+14	4.2e–15 failed	2.7e–16 failed	2.7e–16 1.6e–15	0.050	0.055	0.011
100	0	4.2e+10	2.5e–14 1.1e–3	3.1e–16 2.1e–16	1.6e–16 3.2e–16	0.091	0.11	0.012
100	1	1.7e+17	7.5e–15 failed	3.1e–16 failed	1.6e–16 2.2e–15	0.10	0.12	0.011
200	0	4.6e+18	2.1e–14 failed	3.0e–16 failed	3.0e–16 3.6e–16	0.32	0.34	0.022
200	1	1.6e+18	1.7e–14 failed	3.0e–16 failed	3.0e–16 3.4e–15	0.32	0.35	0.022
500	0	6.3e+19	1.0e+0 failed	1.0e+0 failed	2.4e–16 4.0e–16	1.9	2.1	0.39
500	1	4.4e+18	5.0e–12 failed	2.4e–16 failed	2.4e–16 5.3e–15	1.9	2.0	0.17

Table 6
RRR, RRV, and CPU times (s), and $\mathrm{fl}(|\tilde{\lambda}_1 - \tilde{\lambda}_2|)$ *in Example* 3.

e	$\mathrm{fl}(\lvert\tilde{\lambda}_1 - \tilde{\lambda}_2\rvert)$	M1 RRR RRV	M2 RRR RRV	V RRR RRV	M1 Time	M2 Time	V Time
1.0e–8	4.0e–8	1.1e–15 3.0e–7	4.4e–16 2.2e–15	2.2e–16 1.6e–16	0.0054	0.010	0.0051
1.0e–12	4.0e–12	1.1e–12 3.9e+0	4.4e–16 3.4e–4	8.7e–15 5.2e–6	0.0050	0.012	0.0080
1.0e–15	4.4e–15	6.7e–16 failed	4.4e–16 1.5e+0	2.2e–16 3.5e–4	0.0042	0.011	0.0070

$\underline{\lambda}$ is caused by an inaccurate $\tilde{x}$. Table 8 reports $\mathrm{fl}(\underline{\rho} - \underline{\lambda})$, and the RRR, RRV, and CPU times (s) for two choices of n and p. Only when $n = 1000$ and `p = 1`, the iteration in [42] did not converge, so that `eig` is called. It can be seen that M1 and M2 do not work when $\underline{\lambda}$ is nontight, i.e., $\tilde{x}$ is inaccurate.

7. Future work. As mentioned in Section 3.2, a generalization of the technique in the section is presented in [34, Algorithm 2.1]. One may consider that [34, Algorithm 2.1] can be utilized when eigenvalues are closely clustered around $\rho(A)$. We denote the number of the clustered eigenvalues by $\ell \in \{2, \ldots, n\}$. Then, [34, Algorithm 2.1] gives an accurate approximation $\tilde{X} \in \mathbb{C}^{n\times\ell}$ to a basis of invariant subspaces corresponding to the ℓ eigenvalues. Therefore, $\mathrm{span}(\tilde{X})$ contains an accurate approximation to the Perron vector. However, it is not clear how to construct the approximate Perron vector from $\tilde{X}$. The construction is our future work.

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

TABLE 7
RRR, RRV, and CPU times (s), and $\mathrm{fl}(\min(\tilde{x})/\|\tilde{x}\|_2)$ *in Example* 4.

s	$\mathrm{fl}(\min(\tilde{x})/\|\tilde{x}\|_2)$	M1 RRR RRV	M2 RRR RRV	V RRR RRV	M1 Time	M2 Time	V Time
100	1.0e–18	1.4e–16 1.7e–18	2.8e–16 3.5e–18	1.4e–16 1.7e–18	0.0035	0.011	0.0054
200	2.0e–21	1.4e–16 8.7e–19	2.8e–16 1.7e–18	1.4e–16 8.7e–19	0.0029	0.0085	0.0053
300	5.1e–23	1.9e–16 8.7e–19	3.8e–16 8.7e–19	1.9e–16 4.3e–19	0.0029	0.0095	0.0054
400	3.8e–24	1.4e–16 4.3e–19	2.8e–16 8.7e–19	1.4e–16 4.3e–19	0.0025	0.0091	0.0055
500	5.1e–25	2.3e–16 8.7e–19	2.3e–16 8.7e–19	1.1e–16 4.3e–19	0.0038	0.0093	0.0059

TABLE 8
RRR, RRV, and CPU times (s), and $\mathrm{fl}(\underline{\rho} - \underline{\lambda})$ *in Example* 5.

n	p	$\mathrm{fl}(\underline{\rho} - \underline{\lambda})$	M1 RRR RRV	M2 RRR RRV	V RRR RRV	M1 Time	M2 Time	V Time
500	5	2.7e–15	1.1e–15 2.6e–12	3.1e–16 3.0e–16	4.6e–16 3.1e–15	0.15	0.17	0.13
500	1	2.7e–15	8.4e–14 1.2e–10	4.3e–16 2.5e–16	2.1e–16 3.1e–16	0.16	0.18	0.21
1000	5	4.4e–15	1.3e–15 2.5e–13	3.0e–16 2.9e–16	6.0e–16 6.2e–15	0.42	0.45	0.68
1000	1	2.0e+0	1.0e+0 failed	7.8e–1 failed	2.0e–16 5.1e–16	2.1	2.2	2.5

Lemma 8 and Theorem 9 are more complicated than Theorem 2. Establishing a simpler theory for the verification of x^* is also our future work.

As observed in Example 5, Algorithms 11 and 12 are sensitive to the accuracy of $\tilde{x}$. Hence, developing a robust approximation method is also our future work.

Acknowledgment. The author sincerely thanks the reviewer for the comments, which significantly improved the first version. In particular, adding with the proof of $\delta^* \leq \varepsilon$ in Remark 6, the concise proofs of Theorems 4 and 5, (2), and Lemma 7 presented in this paper are also due to the reviewer. The author especially acknowledges for giving them.

REFERENCES

[1] G. Frobenius, Über Matrizen aus positiven Elementen, 1. *Sitzungsber. Königl. Preuss. Akad. Wiss.*, 471–476, 1908.
[2] G. Frobenius. Über Matrizen aus positiven Elementen, 2. *Sitzungsber. Königl. Preuss. Akad. Wiss.*, 514–518, 1909.

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

[3] T.E. Harris. *The Theory of Branching Processes*. Spring-Verlag, Berlin, 1963.
[4] R.A. Howard and J.E. Matheson. Risk sensitive Markov decision processes. *Manag. Sci.*, 8:356–369, 1972.
[5] S. Stanczak and H. Boche. Information theoretic approach to the Perron root of nonnegative irreducible matrices. In: *Proceedings of IEEE Information Theory Workshop*, San Antonio, 254–259, 2004.
[6] L. Lu and M.K. Ng. Localization of Perron roots. *Linear Algebra Appl.*, 392:103–117, 2004.
[7] G.-X. Huang, F. Yin, and K. Guo. The lower and upper bounds on Perron root of nonnegative irreducible matrices. *J. Comput. Appl. Math.*, 217:259–267, 2008.
[8] C.D. Meyer. Uncoupling the Perron eigenvector problem. *Linear Algebra Appl.*, 114–115:69–94, 1989.
[9] R. Lohner. Enclosing all eigenvalues of symmetric matrices. In: Ch. Ullrich and J. Wolff von Gudenberg (editors), *Accurate Numerical Algorithms: A Collection of Research Papers*, Research Reports ESPRIT, Project 1072, DIAMOND, Springer-Verlag, Berlin, vol. 1, 87–103, 1989.
[10] S. Oishi. Fast enclosure of matrix eigenvalues and singular values via rounding mode controlled computation. *Linear Algebra Appl.*, 324:133–146, 2001.
[11] K. Maruyama, T. Ogita, Y. Nakaya, and S. Oishi. Numerical inclusion method for all eigenvalues of real symmetric definite generalized eigenvalue problem. *IEICE Trans.*, J87-A(8):1111–1119, 2004 (in Japanese).
[12] S. Miyajima, T. Ogita, and S. Oishi. Fast verification for respective eigenvalues of symmetric matrix. *Lect. Notes Comput. Sci.*, 3718:306–317, 2005.
[13] S. Miyajima. Fast enclosure for all eigenvalues in generalized eigenvalue problems. *J. Comput. Appl. Math.*, 233:2994–3004, 2010.
[14] S. Miyajima. Numerical enclosure for each eigenvalue in generalized eigenvalue problem. *J. Comput. Appl. Math.*, 236:2545–2552, 2012.
[15] T. Hoshi, T. Ogita, K. Ozaki, and T. Terao. An a posteriori verification method for generalized Hermitian eigenvalue problems in large-scale electronic state calculations. *J. Comput. Appl. Math.*, 376:112830, 2020.
[16] S. Miyajima, T. Ogita, and S. Oishi. Numerical verification for each eigenpair of symmetric matrix. *Trans. JSIAM*, 16:535–552, 2006 (in Japanese).
[17] S. Miyajima, T. Ogita, S.M. Rump, and S. Oishi. Fast verification for all eigenpairs in symmetric positive definite generalized eigenvalue problems. *Reliab. Comput.*, 14:24–45, 2010.
[18] S. Miyajima. Fast enclosure for all eigenvalues and invariant subspaces in generalized eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 35:1205–1225, 2014.
[19] H. Behnke. Inclusion of eigenvalues of general eigenvalue problems for matrices. *Computing Suppl.*, 6:69–78, 1988.
[20] H. Behnke. The calculation of guaranteed bounds for eigenvalues using complementary variational principles. *Computing*, 47:11–27, 1991.
[21] Y. Watanabe, N. Yamamoto, and M.T. Nakao. Verification methods of generalized eigenvalue problems and its applications. *Trans. JSIAM*, 9(3):137–150, 1999 (in Japanese).
[22] N. Yamamoto. A simple method for error bounds of eigenvalues of symmetric matrices. *Linear Algebra Appl.*, 324:227–234, 2001.
[23] K. Nagatou and Y. Ishii. Validated computation tool for the Perron-Frobenius eigenvalues. *Kyushu Univ. Preprint Series in Math.*, 2008. https://catalog.lib.kyushu-u.ac.jp/opac_download_md/10751/2008-1.pdf
[24] K. Toyonaga. Numerical enclosure for multiple eigenvalues of an Hermitian matrix whose graph is a tree. *Linear Algebra Appl.*, 431:1989–1999, 2009.
[25] A. Imakura, K. Morikuni, and A. Takayasu. Verified partial eigenvalue computations using contour integrals for Hermitian generalized eigenproblems. *J. Comput. Appl. Math.*, 369:112543, 2020.
[26] S.M. Rump and J. Zemke. On eigenvector bounds. *BIT*, 43:823–837, 2004.
[27] T. Yamamoto. Error bounds for computed eigenvalues and eigenvectors. *Numer. Math.*, 34:189–199, 1980.
[28] T. Yamamoto. Error bounds for computed eigenvalues and eigenvectors II. *Numer. Math.*, 40:201–206, 1982.
[29] G. Alefeld and H. Spreuer. Iterative improvement of componentwise error bounds for invariant subspaces belonging to a double or nearly double eigenvalue. *Computing*, 36:321–334, 1986.
[30] G. Mayer. Enclosures for eigenvalues and eigenvectors. In: L. Atanassova and J. Herzberger (editors), *Computer Arithmetic and Enclosure Methods*, Elsevier Science, Amsterdam, 49–68, 1992.
[31] G. Mayer. Result verification for eigenvectors and eigenvalues. In: J. Herzberger (editor), *Topics in Validated Computations*, Elsevier Science, Amsterdam, 209–276, 1994.
[32] G. Mayer. A unified approach to enclosure methods for eigenpairs. *Z. Angew. Math. Mech.*, 74:115–128, 1994.
[33] S.M. Rump. Guaranteed inclusions for the complex generalized eigenproblem. *Computing*, 42:225–238, 1989.
[34] S.M. Rump. Computational error bounds for multiple or nearly multiple eigenvalues. *Linear Algebra Appl.*, 324:209–226, 2001.
[35] A. Neumaier. A simple derivation of the Hansen-Bliek-Rohn-Ning-Kearfott enclosure for linear interval equations. *Reliab. Comput.*, 5:131–136, 1999.

Electronic Journal of Linear Algebra, ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 37, pp. 402-415, May 2021.

[36] S.M. Rump. Accurate solution of dense linear systems, Part II: Algorithms using directed rounding. *J. Comput. Appl. Math.*, 242:185–212, 2013.
[37] A. Minamihata, K. Sekine, T. Ogita, and S. Oishi. Fast verified solutions of sparse linear systems with H-matrices. *Reliab. Comput.*, 15:127–141, 2013.
[38] A. Minamihata, K. Sekine, T. Ogita, S.M. Rump, and S. Oishi. Improved error bounds for linear systems with H-matrices. *NOLTA, IEICE*, 6(3):377–382, 2015.
[39] L. Collatz. Einschliessungssatz fuer die charakteristischen Zahlen von Matrizen. *Math. Z.*, 48:221–226, 1942–43.
[40] H. Wielandt. Unzerlegbare, nicht negative Matrizen. *Math. Z.*, 52:642–648, 1950.
[41] M. Fiedler and V. Pták. On matrices with non-positive off-diagonal elements and positive principal minors. *Czechoslovak Math. J.*, 12:382–400, 1962.
[42] P. Chanchana. An algorithm for computing the Perron root of a nonnegative irreducible matrix. *Ph.D thesis, North Carolina State University*, 2007. https://repository.lib.ncsu.edu/handle/1840.16/3756
[43] S.M. Rump. INTLAB - INTerval LABoratory. In: T. Csendes (editor), *Developments in Reliable Computing*, Kluwer Academic Publishers, Dordrecht, 77–104, 1999.
[44] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.