Yugoslav Journal of Operations Research
34 (2024), Number 1, 73–92
DOI: https://doi.org/10.2298/YJOR230315019Z

BOUNDS ON EIGENVALUES OF REAL SYMMETRIC INTERVAL MATRICES FOR αBB METHOD IN GLOBAL OPTIMIZATION

Djamel ZERROUKI
Laboratoire de Recherche Opérationnelle et de Mathématiques de la Décision, Faculté des Sciences, Université Mouloud Mammeri de Tizi Ouzou, 15000 Tizi-Ouzou, Algeria
djamel.zerrouki@ummto.dz

Mohand OUANES
Laboratoire de Recherche Opérationnelle et de Mathématiques de la Décision, Faculté des Sciences, Université Mouloud Mammeri de Tizi Ouzou, 15000 Tizi-Ouzou, Algeria
mohand.ouanes@ummto.dz

Received: March 2023 / Accepted: August 2023

Abstract: In this paper, we investigate bounds on eigenvalues of real symmetric interval matrices. We present a method that computes bounds on eigenvalues of real symmetric interval matrices. It outperforms many methods developed in the literature and produces as sharp as possible bounds on eigenvalues of real symmetric interval matrices. The aim is to apply the proposed method to compute lower bounds on eigenvalues of a symmetric interval hessian matrix of a nonconvex function in the αBB method and use them to produce a tighter underestimator that improves the αBB algorithm for solving global optimization problems. In the end, we illustrate by example, the comparison of various approaches of bounding eigenvalues of real symmetric interval matrices. Moreover, a set of test problems found in the literature are solved efficiently and the performances of the proposed method are compared with those of other methods.

Keywords: Global optimization, αBB method, eigenvalues bounds, Hessian matrix, interval matrices, interval analysis.

MSC: 65G40, 65F15, 65K05, 90C30.

1. INTRODUCTION

We consider the following problem

$$(Pb) \quad \begin{cases} \min f(x) \\ x \in [\underline{x}, \overline{x}] \subset \mathbb{R}^n \end{cases}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a nonconvex and C^2-continuous function and variable $x \in \mathbb{R}^n$, with $x_i \in \mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$, $\underline{x}_i < \overline{x}_i$, for $i = 1, \ldots, n$, is the interval domain. (Pb) is called a problem of global optimization. Several methods have been studied in the literature for solving global optimization problems. The αBB branch and bound algorithm [1, 2, 3, 4] is based on constructing a convex underestimator for general C^2 functions. The efficiency of the algorithm αBB depends on the tightness of the underestimator. In the αBB method [4], the underestimator has been defined as follows

$$F(x) = f(x) - \frac{1}{2} \sum_{i=1}^{n} \alpha_i (x_i - \underline{x_i})(\overline{x_i} - x_i),$$

with

$$\alpha_i \geq \max \left\{ 0, - \min_{x \in [\underline{x}, \overline{x}]} \lambda_i(x) \right\},$$

where $\lambda_i(x)$ is the ith eigenvalue of the hessian matrix of $f(x)$. The rule choice of α_i is to ensure the convexity of the underestimator. If we assume that the convexity property is guaranteed, then selecting a smaller value of α_i leads to a tighter underestimator. Since we deal with a symmetric interval hessian matrix of $f(x)$, it's too difficult to calculate exact bounds on eigenvalues of general symmetric interval matrices, in fact, it is revealed in [5, 6, 7] that such a problem is, in general, NP-hard. Various methods in [2, 8, 9, 10, 11, 12, 13, 14] have investigated the calculation of eigenvalues of interval matrices. Also, methods in [15, 16, 17] have studied the stability of interval dynamic systems, general parametric interval matrices and more related results to eigenvalues bounds. For the αBB method, the most favorable method for bounding eigenvalues of interval matrices is the scaled Gerschgorin method [2]. It's an easy method to compute each α_i parameter. Otherwise, methods in [11, 10, 12, 14] are less easy to compute bounds on eigenvalues of interval matrices, but they can produce sharper bounds compared to the ones produced by the scaled Gerschgorin method.

Our aim in this paper is to develop a useful method from a practical point of view, to compute as sharp as possible bounds on eigenvalues of real symmetric interval matrices.

An interval matrix is defined by

$$\mathbf{A} := [\underline{A}, \overline{A}] = \left\{ A \in \mathbb{R}^{n \times n}; \underline{A} \leq A \leq \overline{A} \right\},$$

where $\underline{A}, \overline{A} \in \mathbb{R}^{n\times n}$, are given matrices and $A =: (a_{ij})$, with $a_{ij} \in \mathbb{R}$ for $i, j = 1, \dots, n$. The inequality $\underline{A} \le \overline{A}$, is considered element-wise.
We also define a symmetric interval matrix by

$$\mathbf{A}^s := \left\{A \in \mathbf{A} | A^T = A\right\},$$

where $A \in \mathbb{R}^{n\times n}$, is a real symmetric matrix. We denote the midpoint matrix and the radius matrix of the interval matrix $\mathbf{A}$, respectively by

$$A_c := \frac{1}{2}(\underline{A} + \overline{A}), \quad A_\Delta := \frac{1}{2}(\overline{A} - \underline{A}).$$

A real symmetric matrix has n real eigenvalues sorted in a decreasing order as follows

$$\lambda_1(A) \ge \lambda_2(A) \ge \dots \ge \lambda_n(A).$$

And for a real symmetric interval matrix, the eigenvalues are defined as follows

$$\lambda_i(\mathbf{A}^s) = [\underline{\lambda_i}(\mathbf{A}^s), \overline{\lambda_i}(\mathbf{A}^s)] := \{\lambda_i(A) | A \in \mathbf{A}^s\}, \quad i = 1, \dots, n.$$

The structure of this paper is as follows. In section 2, several methods for bounding eigenvalues of symmetric interval matrices are presented. In section 3, a new approach and main results are stated. Computational results are reported in section 4.

2. RELATED WORKS

The Rohn method developed in [14], computes bounds on eigenvalues of the symmetric interval matrix $\mathbf{A}^s$, as follows

Theorem 1. *Bounds on eigenvalues of the symmetric interval matrix* $\boldsymbol{A}^s$*, are given by*

$$\lambda_i(\boldsymbol{A}^s) \in [\underline{\lambda_i}(A_c) - \rho(A_\Delta), \overline{\lambda_i}(A_c) + \rho(A_\Delta)], \quad i = 1, \dots, n.$$

$\rho(A_\Delta)$ *is the spectral radius of the matrix* A_Δ*.*

See [12] for the proof of the above theorem.

The scaled Gerschgorin method developed in [2], computes α_i values (lower bounds on eigenvalues of a real symmetric interval matrix) as follows

$$\alpha_i = \max\left\{0, -\left(\underline{a_{ii}} - \sum_{i\ne j}^{n} \max\left\{|\underline{a_{ij}}|, |\overline{a_{ij}}|\right\} \frac{d_j}{d_i}\right)\right\} \tag{1}$$

with $d_i > 0, \forall i = 1, \ldots, n$. The scaling vector is $d = \overline{x} - \underline{x}$, which means that variables with a wider range have a greater impact on the quality of the underestimator compared to variables with a smaller range. For more details see [2].

The Hladik method developed in [12], is based on Cauchy's interlacing property for eigenvalues of a symmetric matrix [18, 19, 20]. This method involves two versions of interlacing. The first is the direct version which is based directly on the Cauchy theorem. The second is the indirect version which is based on the Weyl theorem [19, 20].

The Hladik diagonal maximization (DM) method developed in [12], is the same as the previous method except that in this method the authors use a diagonal maximization symmetric interval matrix instead of the original symmetric interval matrix. For more details see [12].

The Hertz method developed in [21], is based on constructing a set of 2^{n-1} vertex matrices, so the smallest eigenvalues of this set of matrices are the exact bounds on eigenvalues of the original symmetric interval matrix.

3. NEW APPROACH AND MAIN RESULTS

3.1. New approach

In the following, we present a general proposition that proves valid bounds on eigenvalues for any selection of a symmetric interval matrix from the original symmetric interval matrix $\mathbf{A}^s$. The selection of such a symmetric interval matrix is based on the diagonal elements of the original symmetric interval matrix $\mathbf{A}^s$.

Definition 2. *Let $LV(m) = \{i \in \mathbb{N}\}$, a set of indices, and m is the index of the set LV. For example, $LV(1) = \{2, 4, 5\}$, $LV(2) = \{1, 2, 4, 6\}$, for $m = 1, 2$.*

$LV(m)$ is an identification of a symmetric interval matrix, where all its diagonal elements whose indices are in the set $LV(m)$, are fixed at their lower bounds.

Similar definition, $UV(m) = \{i \in \mathbb{N}\}$ is an identification of a symmetric interval matrix, where all its diagonal elements whose indices are in the set $UV(m)$, are fixed at their upper bounds.

We define a symmetric interval matrix $\mathbf{A}^s_{LV(m)}$ derived from the original symmetric interval matrix $\mathbf{A}^s$, by

$$\mathbf{A}^s_{LV(m)} = \begin{cases} \underline{a_{ij}} & \text{if } i = j = k \quad \text{with } k \in LV(m) \\ \mathbf{a}_{ij} & \text{else} \end{cases}$$

Proposition 3. *For any selection of a symmetric interval matrix $\boldsymbol{A}^s_{LV(m)}$ derived from $\boldsymbol{A}^s$, for each $i \in \{1, \ldots, n\}$, we have*

$$\underline{\lambda_i}(\boldsymbol{A}^s) \geq \underline{\lambda_i}(\boldsymbol{A}^s_{LV(m)}).$$

Proof. For every $A \in \mathbf{A}^s$, with $a_{ij} \in [\underline{a_{ij}}, \overline{a_{ij}}]$. The formula of eigenvalue from The Courant-Fischer theorem [18, 19], is given by

$$\lambda_i(A) = \max_{V \subseteq \mathbb{R}^n; \dim V = i} \left\{ \min_{x^T x = 1} x^T A x \right\}$$

$$= \max_{V \subseteq \mathbb{R}^n; \dim V = i} \left\{ \min_{x^T x = 1} \left(\sum_{i=1}^{n} a_{ii} x_i^2 + \sum_{i \neq j}^{n} a_{ij} x_i x_j \right) \right\}.$$

Now, all diagonal elements whose indices are included in the set $LV(m)$, are fixed at their lower bounds. Hence, the above equality becomes as follows

$$= \max_{V \subseteq \mathbb{R}^n; \dim V = i} \left\{ \min_{x^T x = 1} \left(\sum_{i=1}^{n} a_{ii} x_i^2 + \sum_{i \neq j}^{n} a_{ij} x_i x_j \right) \right\}$$

$$\geq \max_{V \subseteq \mathbb{R}^n; \dim V = i} \left\{ \min_{x^T x = 1} \left(\sum_{k \in LV(m)} \underline{a_{kk}} x_k^2 + \sum_{i \notin LV(m)}^{n} a_{ii} x_i^2 + \sum_{i \neq j}^{n} a_{ij} x_i x_j \right) \right\}$$

$$= \max_{V \subseteq \mathbb{R}^n; \dim V = i} \left\{ \min_{x^T x = 1} x^T A_{LV(m)} x \right\} = \lambda_i(A_{LV(m)}), \quad \forall A_{LV(m)} \in \mathbf{A}^s_{LV(m)}.$$

Therefore,

$$\underline{\lambda_i}(\mathbf{A}^s) \geq \underline{\lambda_i}(\mathbf{A}^s_{LV(m)}), \quad \forall i = 1, \ldots, n, \quad \forall m$$

□

In the same way, we can prove that $\overline{\lambda_i}(\mathbf{A}^s) \leq \overline{\lambda_i}(\mathbf{A}^s_{UV(m)}), \quad \forall i = 1, \ldots, n.$

Definition 4. *Similarly with definition 2, $LV1, LV2$, $UV1$ and $UV2$ are sets of indices.*

$LV1(m) = \{i \in \mathbb{N}\}$ *is an identification of a symmetric interval matrix, where all its diagonal elements whose indices are in the set $LV1(m)$, are fixed at their lower bounds.*

$LV2(m) = \{i \in \mathbb{N}\}$ *is an identification of a symmetric interval matrix, where all its diagonal elements whose indices are in the set $LV2(m)$, are intervals.*

$UV1(m) = \{i \in \mathbb{N}\}$ *is an identification of a symmetric interval matrix, where all its diagonal elements whose indices are in the set $UV1(m)$, are fixed at their upper bounds.*

$UV2(m) = \{i \in \mathbb{N}\}$ *is an identification of a symmetric interval matrix, where all its diagonal elements whose indices are in the set $UV2(m)$, are intervals.*

Now, we define two types of symmetric interval matrices for the calculation of the lower bounds on eigenvalues and two types of symmetric interval matrices for the upper bounds on eigenvalues of the symmetric interval matrix $\mathbf{A}^s$. These symmetric interval matrices are obtained from $\mathbf{A}^s$ as follows

$$\mathbf{A}^s_{LV1(m)} = \begin{cases} \underline{a_{ij}} & \text{if } i = j = k, \quad \text{with } k \in LV1(m) \\ \mathbf{a}_{ij} & \text{else} \end{cases},$$

$$\mathbf{A}^s_{UV1(m)} = \begin{cases} \overline{a_{ij}} & \text{if } i = j = k, \quad \text{with } k \in UV1(m) \\ \mathbf{a}_{ij} & \text{else} \end{cases};$$

$$\mathbf{A}^s_{LV2(m)} = \begin{cases} \underline{a_{ij}} & \text{if } i = j \neq l, \quad \text{with } l \in LV2(m) \\ \mathbf{a}_{ij} & \text{else} \end{cases},$$

$$\mathbf{A}^s_{UV2(m)} = \begin{cases} \overline{a_{ij}} & \text{if } i = j \neq l, \quad \text{with } l \in UV2(m) \\ \mathbf{a}_{ij} & \text{else} \end{cases};$$

with $\mathbf{a}_{ij} = [\underline{a_{ij}}, \overline{a_{ij}}]$, $\underline{a_{ij}} < \overline{a_{ij}}$ and $\underline{a_{ij}}, \overline{a_{ij}} \in \mathbb{R}$, for $i, j = 1, \ldots, n$ and $\forall m$.

We use $LV1$ and $LV2$ for the calculation of the lower bounds on eigenvalues of symmetric interval matrices, and we use $UV1$ and $UV2$ for the calculation of the upper bounds on eigenvalues of symmetric interval matrices.

The main contribution of this work is based on the diagonal elements of the symmetric interval matrix $\mathbf{A}^s$, which is the selection of specific symmetric interval matrices obtained from the original symmetric interval matrix $\mathbf{A}^s$, to produce valid bounds on eigenvalues for $\mathbf{A}^s$. Therefore, we will use the well-known result given in [19, 20], to identify a better selection of symmetric interval matrices that, in general, can produce as sharp as possible bounds on eigenvalues of the original symmetric interval matrix $\mathbf{A}^s$.

3.2. Algorithm

The result used from [19, 20], is the sum of squares of entries of a normal matrix equals the sum of squares of its eigenvalues. Hence, we present an algorithm based on that result, for the selection of the specific symmetric interval matrices. Noted in the presented algorithm that $r = \frac{n}{2}$ if n is even or $r = \frac{n-1}{2}$ if n is odd, and that because the sets $LV1(m)$ and $LV2(m)$ will be reconstructed starting from $i = r$ until $i = (n-2)$. In other words, the symmetric interval matrices $\mathbf{A}^s_{LV1(m)}$ and $\mathbf{A}^s_{LV2(m)}$ for $i = r, \ldots, (n-2)$ are already generated when i variates from 0 to $r-1$. Also, if $i = (n-1)$ and $j = n$, then all indices are added in the set $LV1(m)$, so all diagonal elements of $\mathbf{A}^s_{LV1(m)}$ are fixed at their lower bounds, and that work is already done in [12]. In step 2 of the proposed algorithm, S is a vector of real numbers and Λ is a list of sets that contain indices. In steps 6 and 9, the sum is computed by interval arithmetic as follows

$$S(m) = \sum_{i=1}^{n} |\mathbf{A}^s_{LV1(m),ii}|^2 \qquad \text{if } m \text{ is odd},$$

Algorithm 1 Interval matrices selection $O(n^3)$

1. Set $r = \dfrac{n}{2}$ if n is even or $r = \dfrac{n-1}{2}$ if n is odd.
2. Set $m = 1$, a vector S and a list $\Lambda = \{\emptyset\}$.
3. **for** $i = 0, \ldots, r-1$ **do**
4. **for** $j = i+1, \ldots, n$ **do**
5. Construct the set $LV1(m)$ and if $i > 0$ add all indices $1, \ldots, i$ into the set $LV1(m)$.
6. Add the index j into the set $LV1(m)$ and calculate the sum of squares of the diagonal elements of $\mathbf{A}^s_{LV1(m)}$. Store the result in $S(m)$.
7. Add the set $LV1(m)$ into the list Λ and set m=m+1.
8. Construct the set $LV2(m)$ and if $i > 0$ add all indices $1, \ldots, i$ into the set $LV2(m)$.
9. Add the index j into the set $LV2(m)$ and calculate the sum of squares of the diagonal elements of $\mathbf{A}^s_{LV2(m)}$. Store the result in $S(m)$.
10. Add the $LV2(m)$ into the list Λ and set m=m+1.
11. **end for, end for**
12. Select from Λ, the n sets corresponding to the n minimum values of the vector S. Also select the n sets corresponding to the n maximum values of the vector S.

and

$$S(m) = \sum_{i=1}^{n} |\mathbf{A}^s_{LV2(m),ii}|^2 \qquad \text{if } m \text{ is even,}$$

where $\mathbf{A}^s_{LV1(m),ii}$ is the diagonal element of the symmetric interval matrix $\mathbf{A}^s_{LV1(m)}$ (same for $\mathbf{A}^s_{LV2(m)}$). Considering only diagonal elements because all symmetric interval matrices $\mathbf{A}^s_{LV1(m)}$ have the same nondiagonal elements with the original symmetric interval matrix $\mathbf{A}^s$ and they differ only on the diagonal elements. Finally, in step 12, we select the $2n$ sets (or matrices) corresponding to the n minimum and maximum values, to produce as sharp as possible bounds on eigenvalues. For example, for negative lower bound eigenvalues, it is wise to choose the minimum value of the sum of squares, otherwise, for positive lower bound eigenvalues, choosing the maximum value of the sum of squares would be a good choice. The outputs of the proposed algorithm are $2n$ symmetric interval matrices, but not necessarily n matrices $\mathbf{A}^s_{LV1(m)}$ and n matrices $\mathbf{A}^s_{LV2(m)}$.

To more understand how to use the vector S, the sets $LV1$, $LV2$ and the list Λ in the proposed algorithm, we present a simple execution (without an interval matrix) of the algorithm with $n = 4$. At steps 1 and 2, we have $r = 2$, $m = 1$ and $\Lambda = \{\emptyset\}$. At steps 5 and 6, we construct the set $LV1(1) = \{1\}$, we calculate the sum (noted s_1) and we store it in the vector S, so $S = (s_1)$. At step 7 we add the set $LV1(1)$ into the list Λ, so $\Lambda = \{LV1(1)\}$. Here we have a remark that the sum s_1 in the vector S is an identification of the set $LV1(1)$ in the list Λ. In similar way for steps 8, 9 and 10, we have $m = 2$, we construct the set $LV2(2) = \{1\}$, we calculate the sum (noted s_2) and we store it in the vector S, so $S = (s_1, s_2)$. We add the set $LV2(2)$ into the list Λ, so $\Lambda = \{LV1(1), LV2(2)\}$. At step 12 we will have the following:

$$S = (s_1, s_2, s_3, \dots, s_{m-1}),$$

$$\Lambda = \{LV1(1), LV2(2), LV1(3), \dots, LV2(m-1)\}.$$

Finally, for example, if the n minimum values in S are s_1, s_2, s_4 and s_5, then the sets that must be chosen are $LV1(1)$, $LV2(2)$, $LV2(4)$ and $LV1(5)$. Similar way for the n maximum.

For the complexity, the algorithm involves two nested loops which depend on n. Inside these two loops, in the worst cases, we have: $\frac{n}{2}$ statements in step 5, $(2n+2)$ statements ($2n$ elementary operations) in step 6, 2 statements in step 7, $\frac{n}{2}$ statements in step 8, $(2n+2)$ statements ($2n$ elementary operations) in step 9 and 2 statements in step 10. Therefore, the total complexity without step 12 is $O(n^3)$. The number of statements in step 12 is $m-1$ which can be computed as follows

$$m - 1 = \sum_{i=0}^{r-1} \sum_{j=i+1}^{n} 2 = -r(-2n + r - 1).$$

If n is odd we have $m - 1 = \frac{3}{4}(n-1)(n+1)$, and if n is even we have $m - 1 = \frac{1}{4}n(3n+2)$, which mean that the complexity of step 12 is $O(n^2)$. Hence, the complexity of the algorithm is the dominant term between $O(n^3)$ and $O(n^2)$, which is $O(n^3)$.

3.3. Main results

In this part of this section, results of calculation bounds on eigenvalues of the symmetric interval matrix $\mathbf{A}^s$ are presented. These results are valid and the number of fixed elements at their lower bounds, has no effect on the correctness of these results since we have already proved in proposition 3 that for any selection of symmetric interval matrices $\mathbf{A}^s_{LV(m)}$, the bounds on eigenvalues produced from these matrices are valid. Therefore, for simplicity, we assume that the sets $LV1(m)$ and $LV2(m)$ contain one index, also we assume that the outputs of the proposed algorithm are n symmetric interval matrices $\mathbf{A}^s_{LV1(m)}$ and n symmetric interval matrices $\mathbf{A}^s_{LV2(m)}$ with $m = 1, \dots, n$.

Theorem 5. *For each $i \in \{1, \dots, n\}$, we have*

$$\underline{\lambda_i}(\boldsymbol{A}^s) \geq \max_{m=1,\dots,n} \underline{\lambda_i}(\boldsymbol{A}^s_{LV1(m)})$$

Proof. For every $A \in \mathbf{A}^s$, with $a_{ij} \in [\underline{a_{ij}}, \overline{a_{ij}}]$, we can start with The Courant-Fischer theorem [18, 19]

$$\lambda_i(A) = \max_{V\subseteq\mathbb{R}^n;\dim\ V=i} \left\{ \min_{x^Tx=1} x^T Ax \right\}$$

$$= \max_{V\subseteq\mathbb{R}^n;\dim\ V=i} \left\{ \min_{x^Tx=1} \left(\sum_{i=1}^{n} a_{ii}x_i^2 + \sum_{i\neq j}^{n} a_{ij}x_ix_j \right) \right\},$$

now, for $i = j = k$ with $k \in LV1(m)$, we assume that $a_{ij} = \underline{a_{ij}} = \underline{a_{kk}}$, else we have $\forall a_{ij} \in [\underline{a_{ij}}, \overline{a_{ij}}]$, then

$$\max_{V\subseteq\mathbb{R}^n;\dim\ V=i} \left\{ \min_{x^Tx=1} \left(\sum_{i=1}^{n} a_{ii}x_i^2 + \sum_{i\neq j}^{n} a_{ij}x_ix_j \right) \right\}$$

$$\geq \max_{V\subseteq\mathbb{R}^n;\dim\ V=i} \left\{ \min_{x^Tx=1} \left(\underline{a_{kk}}x_k^2 + \sum_{i\neq k}^{n} a_{ii}x_i^2 + \sum_{i\neq j}^{n} a_{ij}x_ix_j \right) \right\}$$

$$= \max_{V\subseteq\mathbb{R}^n;\dim\ V=i} \left\{ \min_{x^Tx=1} x^T A_{LV1(m)} x \right\} = \lambda_i(A_{LV1(m)}), \quad \forall A_{LV1(m)} \in \mathbf{A}^s_{LV1(m)},$$

therefore,

$$\underline{\lambda_i}(\mathbf{A}^s) \geq \underline{\lambda_i}(\mathbf{A}^s_{LV1(m)}), \quad \forall i = 1,\dots,n, \quad \forall m = 1,\dots,n.$$

Hence,

$$\underline{\lambda_i}(\mathbf{A}^s) \geq \max_{m=1,\dots,n} \underline{\lambda_i}(\mathbf{A}^s_{LV1(m)}), \quad \forall i = 1,\dots,n.$$

□

In similar way, for $i = j = k$ with $k \in UV1(m)$, we assume that $a_{ij} = \overline{a_{ij}} = \overline{a_{kk}}$, else we have $a_{ij} \in [\underline{a_{ij}}, \overline{a_{ij}}]$, we can prove that

$$\overline{\lambda_i}(\mathbf{A}^s) \leq \min_{m=1,\dots,n} \overline{\lambda_i}(\mathbf{A}^s_{UV1(m)}), \qquad i = 1,\dots,n.$$

The next theorem is analogous to theorem 5 since the difference between them is only in the sets $LV1(m)$ and $LV2(m)$.

Theorem 6. *For each $i \in \{1,\dots,n\}$, we have*

$$\underline{\lambda_i}(\boldsymbol{A}^s) \geq \max_{m=1,\dots,n} \underline{\lambda_i}(\boldsymbol{A}^s_{LV2(m)})$$

In similar way, for $i = j \neq l$ with $l \in UV2(m)$, we assume that $a_{ij} = \overline{a_{ij}}$, we have

$$\overline{\lambda_i}(\mathbf{A}^s) \leq \min_{m=1,\dots,n} \overline{\lambda_i}(\mathbf{A}^s_{UV2(m)}), \qquad i = 1,\dots,n.$$

At this point, we can present our main result by resuming theorems 5 and 6 in the following corollary.

Corollary 7. *Let the symmetric interval matrix* $\boldsymbol{A}^s \in \mathbb{IR}^{n\times n}$, *for* $i, m = 1,\dots,n$, *then we have*

$$\underline{\lambda_i}(\boldsymbol{A}^s) \geq \max\left\{\max_m \underline{\lambda_i}(\boldsymbol{A}^s_{LV1(m)}), \max_m \underline{\lambda_i}(\boldsymbol{A}^s_{LV2(m)})\right\},$$

$$\overline{\lambda_i}(\boldsymbol{A}^s) \leq \min\left\{\min_m \overline{\lambda_i}(\boldsymbol{A}^s_{UV1(m)}), \min_m \overline{\lambda_i}(\boldsymbol{A}^s_{UV2(m)})\right\},$$

Proof. Let $\mathbf{A}^s \in \mathbb{IR}^{n\times n}$. From theorems 5 and 6, for $i, m = 1,\dots,n$, we have

$$\underline{\lambda_i}(\mathbf{A}^s) \geq \max_{m=1,\dots,n} \underline{\lambda_i}(\mathbf{A}^s_{LV1(m)}) \quad \text{and} \quad \underline{\lambda_i}(\mathbf{A}^s) \geq \max_{m=1,\dots,n} \underline{\lambda_i}(\mathbf{A}^s_{LV2(m)}),$$

therefore,

$$\underline{\lambda_i}(\mathbf{A}^s) \geq \max\left\{\max_m \underline{\lambda_i}(\mathbf{A}^s_{LV1(m)}), \max_m \underline{\lambda_i}(\mathbf{A}^s_{LV2(m)})\right\}.$$

□

In the same way, we can prove the second inequality presented in corollary 7.

We denote respectively by $A^{(m)}_{l1c}, A^{(m)}_{u1c}, A^{(m)}_{l2c}, A^{(m)}_{u2c}$, the midpoint matrices of the symmetric interval matrices $\mathbf{A}^s_{LV1(m)}, \mathbf{A}^s_{UV1(m)}, \mathbf{A}^s_{LV2(m)}, \mathbf{A}^s_{UV2(m)}$. And $A^{(m)}_{l1\Delta}$, $A^{(m)}_{u1\Delta}, A^{(m)}_{l2\Delta}, A^{(m)}_{u2\Delta}$, the radius matrices of the same symmetric interval matrices for $m = 1,\dots,n$.

Corollary 7 presents valid lower and upper bounds on eigenvalues of the real symmetric interval matrix $\mathbf{A}^s$. Moreover, the symmetric interval matrices derived from $\mathbf{A}^s$, can produce tight bounds on eigenvalues of the original real symmetric interval matrix. By applying Rohn's bounds from theorem 1 on the generated symmetric interval matrices, we get the following corollary.

Corollary 8. *Let $\boldsymbol{A}^s \in \mathbb{IR}^{n\times n}$. Then for $i, m = 1, \ldots, n$ we have*

$$\underline{\lambda_i}(\boldsymbol{A}^s) \geq \max\left\{\max_m \left(\lambda_i(A_{l1c}^{(m)}) - \rho(A_{l1\Delta}^{(m)})\right), \max_m \left(\lambda_i(A_{l2c}^{(m)}) - \rho(A_{l2\Delta}^{(m)})\right)\right\},$$

$$\overline{\lambda_i}(\boldsymbol{A}^s) \leq \min\left\{\min_m \left(\lambda_i(A_{u1c}^{(m)}) + \rho(A_{u1\Delta}^{(m)})\right), \min_m \left(\lambda_i(A_{u2c}^{(m)}) + \rho(A_{u2\Delta}^{(m)})\right)\right\}.$$

3.4. Example

Consider an example found in [12]:

$$\mathbf{A}^s = \begin{pmatrix} [2975, 3025] & [-2015, -1985] & 0 & 0 \\ [-2015, -1985] & [4965, 5035] & [-3020, -2980] & 0 \\ 0 & [-3020, -2980] & [6955, 7045] & [-4025, -3975] \\ 0 & 0 & [-4025, -3975] & [8945, 9055] \end{pmatrix}$$

Next, bounds on eigenvalues of the symmetric interval matrix $\mathbf{A}^s$ are computed. The signification items are as follows

- (Q) Bounds computed by Leng method [11].
- (R) Bounds computed by Rohn method [14].
- (D1) Bounds computed by method 1 rule 1 in [12].
- (D2) Bounds computed by method 1 rule 2 in [12].
- (I1) Bounds computed by method 2 rule 1 in [12].
- (I2) Bounds computed by method 2 rule 2 in [12].
- (B) Best bounds computed in [12].
- (P) Bounds computed by the proposed method.
- (O) Optimal bounds found in [10].

Table 1: Comparison results of the proposed method bounds with different methods bounds on eigenvalues for example 1.

	$[\underline{\lambda_1}(\mathbf{A}^s),\overline{\lambda_1}(\mathbf{A}^s)]$	$[\underline{\lambda_2}(\mathbf{A}^s),\overline{\lambda_2}(\mathbf{A}^s)]$	$[\underline{\lambda_3}(\mathbf{A}^s),\overline{\lambda_3}(\mathbf{A}^s)]$	$[\underline{\lambda_4}(\mathbf{A}^s),\overline{\lambda_4}(\mathbf{A}^s)]$
(Q)	[12550.5313,12730.531]	[6974.459,7154.459]	[3299.848,3479.848]	[815.161,995.161]
(R)	[12560.629,12720.433]	[6984.557,7144.360]	[3309.946,3469.750]	[825.259,985.063]
(D1)	[8945.000,12720.227]	[4945.000,9055.000]	[2924.504,6281.721]	[825.259,3025.000]
(D2)	[8945.000,12720.227]	[2945.000,9453.444]	[1708.932,6281.721]	[825.259,3025.000]
(I1)	[12560.629,12720.4333]	[6984.557,7144.360]	[3309.946,3469.750]	[825.259,985.063]
(I2)	[12560.629,12720.4333]	[6984.557,7144.360]	[3309.946,3469.750]	[825.259,985.063]
(B)	[12560.629,12720.227]	[6990.761,7138.180]	[3320.286,3459.432]	[837.063,973.199]
(P)	[12560.685,12720.378]	[6994.418,7134.511]	[3329.404,3450.475]	[841.923,968.096]
(O)	[12560.837,12720.227]	[7002.282,7126.828]	[3337.078,3443.312]	[842.925,967.108]

Results in table 1 show that the proposed method produces sharper and better bounds (P) compared to Rohn's bounds (R) and Leng's bounds (Q). Noted in [12] that the developed methods are not so sharp when the intervals $[\underline{\lambda_i}(\mathbf{A}^s), \overline{\lambda_i}(\mathbf{A}^s)]$ for $i = 1, \ldots, n$, don't overlap, therefore, the authors choose the best combination bounds (B) of all methods. However, the proposed method produces always sharper bounds compared to Rohn's bounds and even to the best bounds (B) in [12] in this case. $\overline{\lambda_i}(\mathbf{A}^s) = 12720.227$ from (B) is obtained from proposition 3.2 in [12], and that value is by chance the optimal value. We can easily apply that proposition on the proposed method to obtain the same value.

3.5. Complexity

Methods for the calculation of bounds on eigenvalues of real symmetric interval matrices can be classified by their complexities. Hertz method in [21] computes exact bounds on eigenvalues of symmetric interval matrices. However, this accuracy of the bounds comes with a cost of time complexity. The Hertz method generates a number of 2^{n-1} vertex matrices to arrive at the exact bounds on eigenvalues, that cost may become prohibitive in the computations for high dimensional matrices. The scaled Gerschgorin method in [2] sacrifices the accuracy of the bounds on eigenvalues of symmetric interval matrices, for the speed of the computations. It is a method with $O(n^2)$ complexity. The complexity of Rohn and Hladik methods is $O(n^3)$.

The proposed method involves two procedures. The first procedure is the algorithm of interval matrices selection, with complexity equal to $O(n^3)$ as described in section 3.2. The second procedure is the calculation of bounds on eigenvalues of the interval matrices obtained from the first procedure. We have mentioned in section 3.2, that the outputs of the first procedure are $2n$ interval matrices. Since we have applied the Rohn method on the $2n$ interval matrices, then the complexity of the second procedure would be $2n * O(n^3)$, which is $O(n^4)$. Hence, the complexity of the proposed method is the dominant term among the complexities of the two procedures, which in our case is $O(n^4)$.

The proposed method has a simple structure and its complexity is polynomial. Furthermore, the proposed method produces, in general, sharper bounds on eigenvalues of symmetric interval matrices. Therefore, these factors can be seen as implementation advantages for the proposed method in comparison with the existing methods in the literature.

4. COMPUTATIONAL RESULTS

The aim of this paper is to produce tighter lower bounds on eigenvalues $\underline{\lambda_i}$ for $i = 1, \ldots, n$, of symmetric interval hessian matrices, then tighter underestimators in the αBB method, for solving global optimization problems.

In the next experiment, we compare the efficiency of the algorithm αBB when using five different methods for the calculation of each α_i value. The first is the scaled Gerschgorin method that computes α_i by equation (1). The second is the

Hladik method developed in [12]. The third is Hladik diagonal maximization (DM) developed in [12]. The fourth is the proposed method and the last is the Hertz method [21].

The αBB algorithm is implemented in C++ programs and executed on a Dell computer with an Intel(R) Core(TM) i5-4210U CPU with a speed of 3.40 GHz and 8GB RAM. The computational results of the αBB algorithm using the proposed method are summarised in table 3. The performances comparison results of the proposed method with the other methods are reported in tables 4 and 5. The following criteria are taken in consideration:

- $N^o iter$ is the number of iteration to reach to ϵ-global optimum value.
- Tcpu is the computing time in seconds. The values reported are the average values of 1000 times run for each function in table 2.
- σ is the standard deviations of computing time of 1000 times run for each function in table 2.
- Fopt is the minimum ϵ-global optimum value.

Table 3 shows that the minimum optimal solution is obtained by the αBB algorithm when using the proposed method, in a significantly small computational time. Notice from table 4, that the results differ for each function. For functions (1,2,3,4 and 7), there was no improvement when using the Hladik diagonal maximization (DM) or the proposed lower bounds on eigenvalues of the hessian interval matrices of the corresponding functions. However, in comparison with the other approach methods, there was an improvement of up to 156 iterations for function (4) with less computational time as well. For the rest of the functions, the improvement was up to 1886 iterations between using the proposed method and using the other approach methods. On average of 14 functions, using the proposed method, the αBB algorithm requires 776 iterations to reach the optimal minimum value while it requires 1430 iterations with the Hladik method, 1184 iterations with the scaled Gerschgorin method, and 846 iterations with the Hladik diagonal maximization (DM) method. We can measure in percentage, the average improvement on the number of iterations between the proposed method and the other approach methods. Therefore, for the Hladik method there was an improvement of approximately 45.7%. For the scaled Gerschgorin method there was an improvement of 34.5%. Finally, for the Hladik (DM) method there was an improvement of 8.3%. Using the Hertz method [21] that computes exact bounds on the eigenvalues, the αBB algorithm requires 507 iterations on average, to reach the optimal minimum value. Results reported in table 5 present the study of the time speed of the algorithm αBB with different methods of computing each α_i value. Table 5 shows that, for most cases, the algorithm αBB runs faster when using the proposed method than when using other approach methods. Furthermore, the low standard deviations of multiple runs of the algorithm αBB with the proposed

Table 2: Collection of multivariate test functions.

	source	Function	Domain
1	[22]	$sin(x_1 + x_2) + (x_1 - x_2)^2 - 1.5x_1 + 2.5x_2 + 1$	[-1.5,4]$\times[-3, 3]$
2	[22]	$-sin((x_1 - 1)(x_1 - 2)(x_2 + 1))$	[-1,1]$\times[-2, 0]$
3	[22]	$(x_2 - 5x_1^2/(4\pi^2) + 5x_1/\pi - 6)^2 + 10(1 - 1/(8\pi))cos(x_1) + 10$	[-5,10]$\times[0, 15]$
4	[22]	$100(x_2 - x_1^2)^2 + (x_1 - 1)^2$	[-3,3]$\times[-1.5, 4.5]$
5	[22]	$0.5(x_1^2 + x_2^2) - cos(10ln(2x_1))cos(10ln(3x_2)) + 1$	[0.01,1]^2
6	[22]	$(1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2))$	[-2,2]^2
		$(30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2))$	
7	[9]	$x_1^4 + x_2 - (x_1 + x_2^2)^2$	[1,3]$\times[-1, 1]$
8	[9]	$(x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$	[0,1]^4
9	[9]	$(2x_1 + x_2 - 3)^2 + (x_1x_2 - 1)^2$	[0,4]^2
10	[9]	$100(x_2 - x_1^2) + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2$	[1,2]$\times[-0.5, 0.5] \times [1, 2]^2$
		$+10.1((1 - x_2)^2 + (1 - x_4)^2) + 19.8((1 - x_2) + (1 - x_4))$	
11	[8]	$0.4x_1^{2/3}x_3^{-2/3} + 0.4x_2^{2/3}x_4^{2/3} + 10 - x_1 - x_2$	[0.1,10]^4
12	[23]	$\sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	[-3,3]^5
13	[23]	$\sum_{i=2}^{n-2} \left[(x_{i-1} + 10x_i)^2 + 5(x_{i+1} - x_{i+2})^2 + (x_i - 2x_{i+1})^4 + 10(x_{i-1} - x_{i+2})^4\right]$	[-4,5]^5
14	[23]	$(x_1 - 1)^2 + \sum_{i=2}^{n} \left[i(2x_i^2 - x_{i-1})^2\right]$	[-10,10]^5

Table 3: Computational results of using the proposed method in αBB algorithm for solving the test functions listed in table 2.

Function	ϵ	N^o*iter*	Tcpu.10^{-4}	Fopt
1	1.10^{-8}	41	4.8259	1.913223
2	1.10^{-3}	146	17.3072	-0.999994
3	1.10^{-6}	120	14.3239	0.399250
4	1.10^{-7}	275	29.8433	0
5	1.10^{-4}	3261	948.007	0.000179
6	5.10^{-5}	829	116.375	3.000003
7	5.10^{-5}	752	120.295	-4
8	1.10^{-5}	118	24.8714	0
9	5.10^{-5}	76	7.27851	0
10	1.10^{-7}	32	3.72991	27.8846
11	5.10^{-4}	2068	562.093	-9.2
12	1.10^{-5}	1487	381.041	0
13	1.10^{-4}	510	107.736	0
14	1.10^{-4}	1159	320.712	0

method, indicate that the results are more consistent compared to those of the other approach methods.

5. CONCLUSION

In this paper, we considered bounds on eigenvalues of real symmetric interval matrices. Since it's hard to calculate exact or better bounds on eigenvalues of symmetric interval matrices for all cases, we presented a method with several improvements for computing as sharp as possible bounds on eigenvalues of symmetric interval matrices. We have used the lower bounds on eigenvalues from the proposed method to show the efficiency effect of our lower bounds on eigenvalues of symmetric interval hessian matrices, on the αBB algorithm for solving global optimization problems.

Acknowledgements. Authors would like to thank the anonymous referee and the editor for suggestions which has improved the paper. This work is supported by the General Directorate of Scientific Research and Technological Development (DGRSDT)/MESRS, Algeria.

Funding. This research received no external funding.

Table 4: Performance comparison results of using the proposed method with four different methods to compute lower bounds on eigenvalues in αBB algorithm for solving the test functions listed in table 2.

	Gerschgorin [2]		Hladik [12]		Hladik (DM) [12]		Proposed		Hertz [21]	
	N^o$iter$	Tcpu.10^{-4}	N^o$iter$	Tcpu.10^{-4}	N^o$iter$	Tcpu.10^{-4}	N^o$iter$	Tcpu.10^{-4}	N^o$iter$	Tcpu.10^{-4}
1	43	4.9736	41	4.9966	41	4.8267	41	4.8259	41	4.8466
2	222	27.2429	150	17.6776	146	17.3216	146	17.3072	126	14.7452
3	120	13.0445	140	16.7857	120	14.334	120	14.3239	106	11.104
4	403	38.096	431	54.825	275	31.7039	275	29.8433	210	17.9807
5	3907	1294.21	4257	1454.67	3506	1082.82	3261	948.007	3131	908.694
6	1174	172.018	1087	168.992	1078	165.23	829	116.375	782	104.63
7	752	96.208	754	128.682	752	125.748	752	120.295	752	114.691
8	682	144.321	155	32.098	144	29.1104	118	24.8714	60	12.6049
9	110	11.3822	79	7.8877	77	7.61806	76	7.27851	58	5.27939
10	157	19.062	73	10.7122	29	3.89415	28	3.72991	28	3.68711
11	1546	292.617	4819	1627.08	2080	581.392	2068	562.093	622	110.437
12	3812	864.817	4287	1623.55	1803	488.402	1487	381.041	593	125.109
13	1360	268.78	710	161.567	585	127.218	510	107.736	172	41.875
14	2292	499.195	3045	1118.77	1216	308.414	1159	320.712	422	86.3169
Avg.	1184.3	267.569	1430.6	459.164	846.6	213.431	776.4	189.889	507.4	111.571

Table 5: computational time comparison results of using the proposed method with four different methods to compute lower bounds on eigenvalues in αBB algorithm for solving the test functions listed in table 2.

	Gerschgorin [2]		Hladik [12]		Hladik (DM) [12]		Proposed		Hertz [21]	
	Tcpu.10^{-4}	$\sigma.10^{-5}$	Tcpu.10^{-4}	$\sigma.10^{-5}$	Tcpu.10^{-4}	$\sigma.10^{-5}$	Tcpu.10^{-4}	$\sigma.10^{-5}$	Tcpu.10^{-4}	$\sigma.10^{-5}$
1	4.9736	11.261	4.9966	8.174	4.8267	6.818	4.8259	6.301	4.8466	6.571
2	27.2429	37.149	17.6776	13.512	17.3216	14.747	17.3072	14.880	14.7452	15.226
3	13.0445	23.613	16.7857	17.790	14.334	20.001	14.3239	14.301	11.104	18.012
4	38.096	53.312	54.825	50.653	31.7039	36.673	29.8433	33.762	17.9807	22.269
5	1294.21	231.617	1454.67	251.445	1082.82	217.401	948.007	215.236	908.694	207.908
6	172.018	198.379	168.992	175.478	165.23	117.566	116.375	108.769	104.63	113.165
7	96.208	122.545	128.682	150.892	125.748	124.768	120.295	161.507	114.691	88.316
8	144.321	182.166	32.098	16.499	29.1104	19.490	24.8714	24.966	12.6049	12.525
9	11.3822	29.728	7.8877	10.268	7.61806	9.876	7.27851	10.559	5.27939	8.671
10	19.062	28.666	10.7122	11.122	3.89415	5.258	3.72991	5.273	3.68711	5.746
11	292.617	112.001	1627.08	221.993	581.392	208.696	562.093	186.571	110.437	96.274
12	864.817	180.026	1623.55	216.893	488.402	87.490	381.041	50.127	125.109	24.306
13	268.78	263.734	161.567	39.620	127.218	30.2015	107.736	30.026	41.875	58.625
14	499.195	300.813	1118.77	473.363	308.414	59.508	320.712	115.257	86.3169	20.649
Avg.	267.569	126.786	459.164	118.407	213.431	68.464	189.889	69.824	111.571	49.876

REFERENCES

[1] C. D. Maranas and C. A. Floudas, "A deterministic global optimization approach for molecular structure determination," *The Journal of chemical physics*, vol. 100, no. 2, pp. 1247–1261, 1994. doi: 10.1063/1.467236

[2] C. S. Adjiman, S. Dallwig, C. A. Floudas, and A. Neumaier, "A global optimization method, αbb, for general twice-differentiable constrained nlps—i. theoretical advances," *Computers & Chemical Engineering*, vol. 22, no. 9, pp. 1137–1158, 1998. doi: 10.1016/S0098-1354(98)00027-1

[3] C. S. Adjiman, I. P. Androulakis, and C. A. Floudas, "A global optimization method, αbb, for general twice-differentiable constrained nlps—ii. implementation and computational results," *Computers & chemical engineering*, vol. 22, no. 9, pp. 1159–1179, 1998. doi: 10.1016/S0098-1354(98)00218-X

[4] I. P. Androulakis, C. D. Maranas, and C. A. Floudas, "αbb: A global optimization method for general constrained nonconvex problems," *Journal of Global Optimization*, vol. 7, no. 4, pp. 337–363, 1995. doi: 10.1007/BF01099647

[5] A. Nemirovskii, "Several np-hard problems arising in robust stability analysis," *Mathematics of Control, Signals and Systems*, vol. 6, pp. 99–105, 1993. doi: 10.1007/BF01211741

[6] J. Rohn, "Checking positive definiteness or stability of symmetric interval matrices is np-hard," *Commentationes Mathematicae Universitatis Carolinae*, vol. 35, no. 4, pp. 795–797, 1994.

[7] V. Kreinovich, A. V. Lakeyev, J. Rohn, and P. Kahl, *Computational complexity and feasibility of data processing and interval computations.* Springer Science & Business Media, 2013, vol. 10.

[8] C. E. Gounaris and C. A. Floudas, "Tight convex underestimators for c^2-continuous problems: Ii. multivariate functions," *Journal of Global Optimization*, vol. 42, no. 1, pp. 69–89, 2008. doi: 10.1007/s10898-008-9288-8

[9] M. Hladík, "On the efficient gerschgorin inclusion usage in the global optimization αBB method," *Journal of Global Optimization*, vol. 61, no. 2, pp. 235–253, 2015. doi: 10.1007/s10898-014-0161-7

[10] Q. Yuan, Z. He, and H. Leng, "An evolution strategy method for computing eigenvalue bounds of interval matrices," *Applied mathematics and computation*, vol. 196, no. 1, pp. 257–265, 2008. doi: 10.1016/j.amc.2007.05.051

[11] H. Leng and Z. He, "Computing eigenvalue bounds of structures with uncertain-but-non-random parameters by a method based on perturbation theory," *Communications in Numerical Methods in Engineering*, vol. 23, no. 11, pp. 973–982, 2007. doi: 10.1002/cnm.936

[12] M. Hladík, D. Daney, and E. Tsigaridas, "Bounds on real eigenvalues and singular values of interval matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 4, pp. 2116–2129, 2010. doi: 10.1137/090753991

[13] H. Leng, Z. He, and Q. Yuan, "Computing bounds to real eigenvalues of real-interval matrices," *International journal for numerical methods in engineering*, vol. 74, no. 4, pp. 523–530, 2008. doi: 10.1002/nme.2179

[14] J. Rohn, "A handbook of results on interval linear problems," *Internet text available at http://www. cs. cas. cz/rohn/handbook*, 2005.

[15] L. Kolev and S. Petrakieva, "Assessing the stability of linear time-invariant continuous interval dynamic systems," *IEEE Transactions on Automatic Control*, vol. 50, no. 3, pp. 393–397, 2005. doi: 10.1109/TAC.2005.843857

[16] J. Rohn, "Stability of interval matrices: the real eigenvalue case," *IEEE transactions on automatic control*, vol. 37, no. 10, pp. 1604–1605, 1992. doi: 10.1109/9.256393

[17] L. Kolev, "Eigenvalue range determination for interval and parametric matrices," *International Journal of Circuit Theory and Applications*, vol. 38, no. 10, pp. 1027–1061, 2010. doi: 10.1002/cta.609

[18] G. H. Gloub and C. F. Van Loan, "Matrix computations," *Johns Hopkins Universtiy Press, 3rd edtion*, 1996.

[19] R. A. Horn and C. R. Johnson, *Matrix analysis.* Cambridge university press, 2012.

[20] J. H. Wilkinson, *The algebraic eigenvalue problem.* Oxford University Press, Inc., 1988.

[21] D. Hertz, "The extreme eigenvalues and stability of real symmetric interval matrices," *IEEE Transactions on automatic control*, vol. 37, no. 4, pp. 532–535, 1992. doi: 10.1109/9.126593

[22] M. Ouanes, H. A. Le Thi, T. P. Nguyen, and A. Zidna, "New quadratic lower bound for multivariate functions in global optimization," *Mathematics and Computers in Simulation*, vol. 109, pp. 197–211, 2015. doi: 10.1016/j.matcom.2014.04.013

[23] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 2, pp. 150–194, 2013. doi: 10.1504/IJMMNO.2013.055204