

Computing a lower bound of the  
smallest eigenvalue of a symmetric  
positive definite Toeplitz matrix

*Teresa Laudadio      Nicola Mastronardi*  
*Marc Van Barel*

*Report TW 511, November 2007*



Katholieke Universiteit Leuven  
Department of Computer Science  
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

# Computing a lower bound of the smallest eigenvalue of a symmetric positive definite Toeplitz matrix

*Teresa Laudadio      Nicola Mastronardi*  
*Marc Van Barel*

*Report TW 511, November 2007*

Department of Computer Science, K.U.Leuven

## **Abstract**

In this paper several algorithms to compute a lower bound of the smallest eigenvalue of a symmetric positive definite Toeplitz matrix are described and compared in terms of accuracy and computational efficiency.

Exploiting the Toeplitz structure of the considered matrix, new theoretical insights are derived and an efficient implementation of some of the aforementioned algorithms is provided.

**Keywords :** Toeplitz matrix; eigenvalue problem; symmetric positive definite matrix.

**MSC :** Primary : 65F15.

# Computing a Lower Bound of the smallest Eigenvalue of a Symmetric Positive Definite Toeplitz Matrix

Teresa Laudadio<sup>1</sup>, Nicola Mastronardi<sup>1,2</sup> and Marc Van Barel<sup>2</sup>

<sup>1</sup>Istituto per le Applicazioni del Calcolo "M. Picone",  
Consiglio Nazionale delle Ricerche, Bari, 70126, Italy  
{t.laudadio, n.mastronardi}@ba.iac.cnr.it

<sup>2</sup>Department of Computer Science,  
Katholieke Universiteit Leuven, Leuven, 3001, Belgium  
{Nicola.Mastronardi, Marc.VanBarel}@cs.kuleuven.be

November 30, 2007

## Abstract

In this paper several algorithms to compute a lower bound of the smallest eigenvalue of a symmetric positive definite Toeplitz matrix are described and compared in terms of accuracy and computational efficiency.

Exploiting the Toeplitz structure of the considered matrix, new theoretical insights are derived and an efficient implementation of some of the aforementioned algorithms is provided.

**AMS Subject Classification:** 65F15

**Key Words and Phrases:** Toeplitz matrix; eigenvalue problem; symmetric positive definite matrix.

## 1 Introduction

Computing a lower bound of the smallest eigenvalue of a symmetric positive definite (SPD) Toeplitz matrix is of considerable interest in a variety of signal processing applications and estimation problems [1]. The aim of this paper is manifold: to provide a short survey of the most significant algorithms available in the literature for general SPD matrices, to compare their performance, to reveal new useful theoretical insights of the considered methods and, by exploiting the matrix Toeplitz structure, to derive an efficient implementation

of some of them. In particular, in our studies we consider the following algorithms: the method by Ma and Zarowski [4], the two algorithms proposed by W. Sun in [5], the Newton-based method proposed by Mastronardi and Boley [6], and the algorithm by C. Fassino [7]. We show that one of the methods proposed by Sun is equivalent to Fassino's method, and that both Sun's algorithms can be implemented in an efficient way. The paper is organized as follows: in sections II–V the aforementioned methods and their improved implementations are described; in section VI the numerical results are reported; in section VII the main conclusions are formulated; finally, in the appendix section, the equivalence between the method of Sun and the one of Fassino is proved.

## 2 Method of Ma and Zarowski

Let  $T_n$  be an  $n \times n$  SPD Toeplitz matrix:

$$T_n = \begin{bmatrix} t_0 & t_1 & \ddots & t_{n-2} & t_{n-1} \\ t_1 & t_0 & \ddots & \ddots & t_{n-2} \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ t_{n-2} & \ddots & \ddots & \ddots & t_1 \\ t_{n-1} & t_{n-2} & \ddots & t_1 & t_0 \end{bmatrix}.$$

We denote by

$$t^{(n-1)} = [ t_1 \quad t_2 \quad \dots \quad t_{n-1} ]^T$$

and by  $E_{n-1}$  the exchange matrix of order  $n - 1$ :

$$E_{n-1} = \begin{bmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & \dots & 0 & 1 & 0 \\ 0 & \dots & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 1 & 0 & \dots & \dots & 0 \end{bmatrix}$$

Let's partition the matrix  $T_n$  as follows:

$$T_n = \begin{bmatrix} T_{n-1} & E_{n-1}t^{(n-1)} \\ t^{(n-1)T}E_{n-1} & t_0 \end{bmatrix}, \quad (1)$$

where  $T_{n-1}$  is an  $(n-1) \times (n-1)$  matrix,  $t_0$  is the diagonal entry of  $T_n$ , and  $t^{(n-1)T}$  denotes the transpose of  $t^{(n-1)}$ . It is well known that the eigenvalues of  $T_n$  and  $T_{n-1}$  are real and satisfy the following interlacing property [8]:

$$\lambda_1(T_n) \leq \lambda_1(T_{n-1}) \leq \lambda_2(T_n) \leq \dots \leq \lambda_{n-1}(T_{n-1}) \leq \lambda_n(T_n).$$

As proved in [4], a lower bound of the smallest eigenvalue of  $T_n$  can be obtained in terms of a lower bound of the smallest eigenvalue of  $T_{n-1}$ . Indeed, the following theorem holds for any SPD matrix [4]:

**Theorem 1.** *Let  $\eta_1$  be a lower bound of the smallest eigenvalue of  $T_{n-1}$  and set  $b = E_{n-1}t^{(n-1)}$ . Then, a lower bound of the smallest eigenvalue of  $T_n$  is given by:*

$$\Delta_{MZ} = \frac{t_0 + \eta_1 - \sqrt{(t_0 + \eta_1)^2 - 4(t_0 - b^T T_{n-1}^{-1} b)\eta_1}}{2}. \quad (2)$$

It is proved that the bound  $\Delta_{MZ}$  is positive when  $T_n$  is Hermitian positive definite and, therefore, when it is a SPD Toeplitz matrix.

The computationally most intensive part of Eq.(2) is the computation of  $b^T T_{n-1}^{-1} b$ . The matrix-vector product  $T_{n-1}^{-1} b$  is efficiently computed by formulating the given problem in terms of Yule-Walker systems and by applying the Levinson-Durbin algorithm to solve them [8]. This algorithm requires  $2n^2$  flops. By taking into account the scalar product between  $b^T$  and the solution of the Yule-Walker system, the total number of flops required by the algorithm is  $3n^2$ .

### 3 Methods of Sun

More accurate lower bounds were introduced by W. Sun in [5]. The estimation of such bounds is based on the properties of the matrix series and the local monotonicity of the smallest eigenvalue. The following theorems hold [5]:

**Theorem 2.** *Let  $\eta_1$  be a lower bound of the smallest eigenvalue of  $T_{n-1}$ . Then, a lower bound of the smallest eigenvalue of  $T_n$  is given by:*

$$\Delta_{S1} = \frac{d_1 + d_2\eta_1 - \sqrt{(d_1 + d_2\eta_1)^2 - 4d_1\eta_1}}{2}. \quad (3)$$

**Theorem 3.** *Let  $\eta_1$  be a lower bound of the smallest eigenvalue of  $T_{n-1}$ . Then, a lower bound of the smallest eigenvalue of  $T_n$  is given by:*

$$\Delta_{S2} = \frac{d_1 + d_2\eta_1 - \sqrt{(d_1 + d_2\eta_1)^2 - 4d_1d_3\eta_1}}{2d_3}. \quad (4)$$

In the above theorems  $d_1$ ,  $d_2$  and  $d_3$  are defined as follows:

$$\begin{aligned} d_1 &= t_0 - b^T T_{n-1}^{-1} b, \\ d_2 &= 1 + b^T T_{n-1}^{-2} b, \\ d_3 &= d_2 - b^T T_{n-1}^{-3} b \eta_1. \end{aligned}$$

In the sequel the two methods based on (3) and (4) will be denoted by SUN1 and SUN2, respectively.

Concerning the computational cost, in [5] no analysis was carried out and the two methods, if roughly implemented, would require a big number of flops. In particular, SUN2 would exhibit  $O(n^4)$  cost. Here, we propose an efficient implementation of both algorithms. First of all, observe that in SUN1 the following products need to be computed:  $b^T T_{n-1}^{-1} b$  and  $b^T T_{n-1}^{-2} b$ . As already described in the previous section, the former needs  $3n^2$  flops. The latter only needs  $n^2$  flops as it can be written as follows:

$$b^T T_{n-1}^{-2} b = b^T T_{n-1}^{-1} T_{n-1}^{-1} b = (T_{n-1}^{-1} b)^T (T_{n-1}^{-1} b), \quad (5)$$

where  $T_{n-1}^{-1} b$  represents the solution of the Yule–Walker system and has already been computed. Therefore, it reduces to a scalar product that requires additional  $n^2$  flops. The total cost of the algorithm is  $4n^2$ . In the appendix we will prove that SUN1, that requires only  $4n^2$  flops, is mathematically equivalent to Fassino’s method [7] that requires only  $3n^2$  flops, based on the Cholesky factorization of  $T_n$ . Therefore, Fassino’s method can replace SUN1.

The algorithm SUN2 requires the computation of the aforementioned products as well as the computation of  $b^T T_{n-1}^{-3} b$ . This can be written as:

$$b^T T_{n-1}^{-3} b = b^T T_{n-1}^{-1} T_{n-1}^{-2} b = (T_{n-1}^{-1} b)^T (T_{n-1}^{-2} b). \quad (6)$$

Once again,  $T_{n-1}^{-1} b$  represents the solution of the Yule–Walker system. The product  $T_{n-1}^{-2} b$  can be efficiently computed by applying the following algorithm.

### 3.1 Solution of $T_{n-1}^2 x^{(n-1)} = -E_{n-1} t^{(n-1)}$

Consider the Toeplitz matrix  $T_i$ , with  $2 \leq i \leq n-1$ , partitioned as follows:

$$T_i = \begin{bmatrix} t_0 & t^{(i-1)T} \\ t^{(i-1)} & T_{i-1} \end{bmatrix}, \quad (7)$$

and suppose for some  $i$ ,  $2 \leq i \leq n-1$ , the solution of the following systems is known:

$$\begin{aligned} T_{i-2}y^{(i-2)} &= -E_{i-2}t^{(i-2)}, \\ T_{i-2}x^{(i-2)} &= y^{(i-2)}. \end{aligned}$$

We show how to solve the systems

$$T_{i-1}y^{(i-1)} = -E_{i-1}t^{(i-1)}, \quad (8)$$

$$T_{i-1}x^{(i-1)} = y^{(i-1)}. \quad (9)$$

Let, for  $i = 2, \dots, n-1$ ,

$$y^{(i-1)} = [\alpha_{i-1}, z^{(i-1)T}]^T, \quad (10)$$

$$x^{(i-1)} = [\gamma_{i-1}, w^{(i-1)T}]^T. \quad (11)$$

Then (8) and (9) can be written in this way:

$$\begin{aligned} \left[ \begin{array}{c|c} t_0 & t^{(i-2)T} \\ \hline t^{(i-2)} & T_{i-2} \end{array} \right] \left[ \begin{array}{c} \alpha_{i-1} \\ z^{(i-1)} \end{array} \right] &= -E_{i-1}t^{(i-1)} \\ &= - \left[ \begin{array}{c} t_{i-1} \\ E_{i-2}t^{(i-2)} \end{array} \right], \end{aligned}$$

$$\left[ \begin{array}{c|c} t_0 & t^{(i-2)T} \\ \hline t^{(i-2)} & T_{i-2} \end{array} \right] \left[ \begin{array}{c} \gamma_{i-1} \\ w^{(i-1)} \end{array} \right] = y^{(i-1)} = \left[ \begin{array}{c} \alpha_{i-1} \\ z^{(i-1)} \end{array} \right].$$

and from these, after simple manipulations, we obtain:

$$\alpha_{i-1} = - \frac{t_{i-1} + t^{(i-2)T} y^{(i-2)}}{t_0 + t^{(i-2)T} E_{i-2} y^{(i-2)}}, \quad (12)$$

$$z^{(i-1)} = \alpha_{i-1} E_{i-2} y^{(i-2)} + y^{(i-2)},$$

$$\gamma_{i-1} = \frac{\alpha_{i-1} \left( 1 - t^{(i-2)T} E_{i-2} x^{(i-2)} \right) - t^{(i-2)T} x^{(i-2)}}{t_0 + t^{(i-2)T} E_{i-2} y^{(i-2)}}, \quad (13)$$

$$w^{(i-1)} = \gamma_{i-1} E_{i-2} y^{(i-2)} + \alpha_{i-1} E_{i-2} x^{(i-2)} + x^{(i-2)}.$$

It is possible to reduce the amount of work since some scalar products in (12) and (13) can be computed by means of recurrence relations. More precisely, let

$$\beta_{i-1} = t_0 + t^{(i-2)T} E_{i-2} y^{(i-2)}$$

and

$$\delta_{i-1} = 1 - t^{(i-2)T} E_{i-2} x^{(i-2)}.$$

Then the following recurrence relations hold:

$$\beta_{i-1} = (1 - \alpha_{i-1}^2) \beta_{i-2}, \quad (14)$$

$$\delta_{i-1} = (1 - \alpha_{i-2}^2) \delta_{i-2} + 2\alpha_{i-2} \beta_{i-2} \gamma_{i-2}. \quad (15)$$

*Proof.* Let  $k = i - 3$ . Then

$$\begin{aligned} \beta_{k+2} &= t_0 + t^{(k+1)T} E_{k+1} y^{(k+1)} \\ &= t_0 + \begin{bmatrix} t^{(k)T} & t_{k+1} \end{bmatrix} \begin{bmatrix} E_k (\alpha_{k+1} E_k y^{(k)} + y^{(k)}) \\ \alpha_{k+1} \end{bmatrix} \\ &= t_0 + t^{(k)T} E_k y^{(k)} + \alpha_{k+1} \left( t^{(k)T} y^{(k)} + t_{k+1} \right). \end{aligned}$$

Then (14) follows since

$$t_0 + t^{(k)T} E_k y^{(k)} = \beta_{k+1} \quad (16)$$

and

$$t^{(k)T} y^{(k)} + t_{k+1} = -\alpha_{k+1} \beta_{k+1}. \quad (17)$$

To prove (15) we observe that

$$\begin{aligned}
\delta_{k+2} &= 1 - t^{(k+1)T} E_{k+1} x^{(k+1)} \\
&= 1 - [t^{(k)T}, t_{k+1}] E_{k+1} \begin{bmatrix} \gamma_{k+1} \\ w^{(k+1)} \end{bmatrix} \\
&= 1 - [t^{(k)T}, t_{k+1}] \begin{bmatrix} E_k w^{(k+1)} \\ \gamma_{k+1} \end{bmatrix} \\
&= 1 - [t^{(k)T}, t_{k+1}] \begin{bmatrix} \gamma_{k+1} y^{(k)} + \alpha_{k+1} x^{(k)} + E_k x^{(k)} \\ \gamma_{k+1} \end{bmatrix} \\
&= 1 - \gamma_{k+1} \left( t_{k+1} + t^{(k)T} y^{(k)} \right) - \alpha_{k+1} t^{(k)T} x^{(k)} \\
&\quad - t^{(k)T} E_k x^{(k)}.
\end{aligned}$$

Since

$$\alpha_{k+1} \beta_{k+1} = -(t_{k+1} + t^{(k)T} y^{(k)}),$$

and

$$\delta_{k+1} = 1 - t^{(k)T} E_k x^{(k)},$$

we have:

$$\delta_{k+2} = \delta_{k+1} + \alpha_{k+1} \beta_{k+1} \gamma_{k+1} - \alpha_{k+1} t^{(k)T} x^{(k)}.$$

Then (15) follows since, by (13),

$$-t^{(k)T} x^{(k)} = \beta_{k+1} \gamma_{k+1} - \alpha_{k+1} \delta_{k+1}.$$

□

The total number of flops required by SUN2 is  $7n^2$  flops.

## 4 Method of Mastronardi and Boley

In [6] the smallest eigenpair of a SPD Toeplitz matrix is computed by applying Newton's method to the characteristic polynomial of the matrix, when choosing as starting value  $\lambda = 0$ . This method is iterative and, at each step, computes an approximation of the smallest eigenvalue by means of the following relation:

$$\lambda = \lambda - \frac{p_i(\lambda)}{p_i'(\lambda)}, \quad (18)$$

where  $p_i(\lambda)$  and  $p_i'(\lambda)$  represent the characteristic polynomial and its first order derivative, respectively, of the matrix  $T_i$  of order  $i$ .

A lower bound of the smallest eigenvalue can be simply obtained by one step of Newton's method. It is worth observing that numerical cancellation may occur when computing  $p_i(\lambda)/p'_i(\lambda)$ . Therefore, as proved in [6], it is preferable to construct a recurrence relation  $\Phi_i(\lambda) = p_i(\lambda)/p'_i(\lambda)$  directly, and the lower bound is obtained by performing one step of the following recurrence relations:

$$\begin{aligned}\Phi_1(\lambda) &= -\beta_1, \\ \Phi_i(\lambda) &= \frac{\beta_i}{\frac{\beta_i}{\Phi_{i-1}(\lambda)} - (1 + \|y^{(i-1)}\|_2^2)}, \quad i = 2, \dots, n, \\ \underline{\lambda}_N &= -\Phi_n(\lambda),\end{aligned}$$

where  $\beta_i$  is defined in (14) and  $y^{(i-1)}$  is defined in (10).

The computational cost of the Newton-based algorithm is  $3n^2$ . In the sequel, this method will be denoted by N.

## 5 Method of Fassino

In [7] an algorithm to compute a lower bound of the smallest singular value of rectangular matrices is derived. This algorithm can be easily adapted in order to compute a lower bound of the smallest eigenvalue of a SPD Toeplitz matrix. Specifically, given  $T_n$  defined in (1) and  $R_n$  the  $R$  factor of the  $QR$  factorization of  $T_n$ , consider the partition:

$$R_n = \begin{bmatrix} R_{n-1} & r \\ 0 & \gamma \end{bmatrix},$$

with  $R_{n-1} = R(1:n-1, 1:n-1)$ ,  $r = R(1:n-1, n)$ , and  $\gamma = R(n, n)$ .

Fassino's algorithm computes a lower bound of the smallest eigenvalue of the SPD matrix  $T_n$  by means of the following scheme. Known  $\eta_1$ , a lower bound of the smallest eigenvalue of  $T_{n-1}$ , a lower bound  $\underline{\lambda}_F$  of  $T_n$  is given by:

$$\underline{\lambda}_F = \eta_1 \left( 1 + \frac{1}{2} \left( B - \sqrt{B^2 + 4\|x\|_2^2} \right) \right), \quad (19)$$

where  $B \equiv B(\eta_1, x, \rho) = \|x\|_2^2 + \rho^2/\eta_1 - 1$ ,  $x = R_{n-1}^{-1}r$ , and  $\rho = |\gamma|$ .

As described in [9], the computational cost of this algorithm is  $19n^2$ . Indeed, Fassino's method involves the computation of  $\|x\|_2$  and the inverse of the matrix  $R_n$ . The latter can be efficiently performed by applying the generalized

Schur algorithm and the required number of flops is  $18n^2$ . The former is performed by  $n^2$  flops.

An alternative and computationally less expensive algorithm, based on the method of Fassino, can be also derived. It is based on the Cholesky, rather than the  $QR$ , factorization of  $T_n$ . Although this alternative version only requires  $3n^2$  flops, the numerical tests show that it is less accurate than the  $QR$ -based implementation. Furthermore, it is possible to prove that the Cholesky-based version is mathematically equivalent to the algorithm SUN1 and is less intensive from the computational point of view since SUN1 requires  $4n^2$  flops. The proof of the equivalence between the aforementioned algorithms is reported in the appendix section.

In order to distinguish the two implementations of Fassino's method, we will refer to them as Fassino- $QR$  and Fassino-Cholesky. Moreover, the lower bound estimated by the Fassino- $QR$  implementation will be denoted by  $\underline{\Delta}_{FQR}$ , and that one provided by Fassino-Cholesky will be denoted by  $\underline{\Delta}_{FCho}$ .

## 6 Numerical tests

In this section the results of the numerical tests are described. All the computations were performed in MATLAB, a registered trademark of The MathWorks, Inc. In our studies, two examples of SPD Toeplitz matrices were considered.

**Example 1.** *The first example is represented by the SPD Toeplitz matrices [6], [9] defined as follows:*

$$T_n = m \sum_{k=1}^n \omega_k T_{2\pi\theta_k}, \quad (20)$$

where  $n$  is the dimension,  $m$  is chosen so that  $T$  is normalized in order to have the entries of the main diagonal of the matrix equal to 1,

$$T_\theta = t_{ij} = \cos((i-j)\theta),$$

and  $\omega_k$  and  $\theta_k$  are uniformly distributed random numbers taken from  $[0,1]$ .

We generated 100 SPD Toeplitz matrices of type (20) with dimension  $n = 128, 256, 512, 1024$ , and estimated a lower bound of the smallest eigenvalue by applying the algorithms described in the previous sections. Table 1 and 2 show the average of the absolute and relative errors of the estimates, respectively.

The results reported in the tables show that the Ma and Zarowski method is the least accurate. We can also observe that the methods SUN1 and Fassino-Cholesky provide the same lower bound, but the latter should be preferred

Absolute error mean	$n = 128$	$n = 256$	$n = 512$	$n = 1024$
$(\lambda_{min} - \underline{\lambda}_{MZ})$	0.1689	0.1046	0.0505	0.0218
$(\lambda_{min} - \underline{\lambda}_{S1})$	0.1273	0.0848	0.0394	0.0172
$(\lambda_{min} - \underline{\lambda}_{S2})$	0.0606	0.0438	0.0182	0.0085
$(\lambda_{min} - \underline{\lambda}_N)$	0.1384	0.0882	0.0406	0.0175
$(\lambda_{min} - \underline{\lambda}_{FQR})$	0.0674	0.0473	0.0191	0.0087
$(\lambda_{min} - \underline{\lambda}_{FCho})$	0.1273	0.0848	0.0394	0.0172

Table 1: Absolute error mean obtained when computing the lower bound of the smallest eigenvalue of (20) by the methods described in sections II-V.

Relative error mean	$n = 128$	$n = 256$	$n = 512$	$n = 1024$
$(\lambda_{min} - \underline{\lambda}_{MZ})/\lambda_{min}$	1.0000	1.0000	1.0000	1.0000
$(\lambda_{min} - \underline{\lambda}_{S1})/\lambda_{min}$	0.5293	0.5837	0.6093	0.6230
$(\lambda_{min} - \underline{\lambda}_{S2})/\lambda_{min}$	0.2154	0.2605	0.2780	0.3012
$(\lambda_{min} - \underline{\lambda}_N)/\lambda_{min}$	0.5630	0.6002	0.6195	0.6276
$(\lambda_{min} - \underline{\lambda}_{FQR})/\lambda_{min}$	0.2347	0.2739	0.2860	0.3055
$(\lambda_{min} - \underline{\lambda}_{FCho})/\lambda_{min}$	0.5293	0.5837	0.6093	0.6230

Table 2: Relative error mean obtained when computing the lower bound of the smallest eigenvalue of (20) by the methods described in sections II-V.

to the former because of its lower computational cost. The Newton-based method provides values which are very close to the ones provided by SUN1 and Fassino-Cholesky. The most accurate algorithms are represented by SUN2 and Fassino-QR, but SUN2 performs slightly better in accuracy and is significantly more efficient than Fassino-QR.

**Example 2.** The second example is given by the following tridiagonal Toeplitz matrices  $\bar{T}_n$ :

$$\bar{T}_n = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \quad (21)$$

The numerical results are reported in Tables 3 and 4.

Once again, the most accurate methods are SUN2 and Fassino-QR, while the least accurate is the method by Ma and Zarowski. Also in these tests, the Newton method provides results similar to SUN1 and Fassino-Cholesky.

For completeness' sake, a table showing the total number of flops required

Absolute error mean	$n = 128$	$n = 256$	$n = 512$	$n = 1024$
$(\lambda_{min} - \underline{\lambda}_{MZ})$	$5.9306 \times 10^{-4}$	$1.4943 \times 10^{-4}$	$3.7503 \times 10^{-5}$	$9.3940 \times 10^{-6}$
$(\lambda_{min} - \underline{\lambda}_{S1})$	$2.1674 \times 10^{-4}$	$5.6542 \times 10^{-5}$	$1.4443 \times 10^{-5}$	$3.6501 \times 10^{-6}$
$(\lambda_{min} - \underline{\lambda}_{S2})$	$2.2013 \times 10^{-5}$	$5.6698 \times 10^{-6}$	$1.4387 \times 10^{-6}$	$3.6235 \times 10^{-7}$
$(\lambda_{min} - \underline{\lambda}_N)$	$2.3248 \times 10^{-4}$	$5.8584 \times 10^{-5}$	$1.4704 \times 10^{-5}$	$3.6831 \times 10^{-6}$
$(\lambda_{min} - \underline{\lambda}_{FQR})$	$2.3004 \times 10^{-5}$	$5.7957 \times 10^{-6}$	$1.4545 \times 10^{-6}$	$3.6433 \times 10^{-7}$
$(\lambda_{min} - \underline{\lambda}_{FCho})$	$2.1674 \times 10^{-4}$	$5.6542 \times 10^{-4}$	$1.4443 \times 10^{-5}$	$3.6501 \times 10^{-6}$

Table 3: Absolute error mean obtained when computing the lower bound of the smallest eigenvalue of (21) by the methods described in sections II-V.

Relative error mean	$n = 128$	$n = 256$	$n = 512$	$n = 1024$
$(\lambda_{min} - \underline{\lambda}_{MZ})/\lambda_{min}$	1.0000	1.0000	1.0000	1.0000
$(\lambda_{min} - \underline{\lambda}_{S1})/\lambda_{min}$	0.3655	0.3784	0.3851	0.3886
$(\lambda_{min} - \underline{\lambda}_{S2})/\lambda_{min}$	0.0371	0.0379	0.0384	0.0386
$(\lambda_{min} - \underline{\lambda}_N)/\lambda_{min}$	0.3920	0.3921	0.3921	0.3921
$(\lambda_{min} - \underline{\lambda}_{FQR})/\lambda_{min}$	0.0388	0.0388	0.0388	0.0388
$(\lambda_{min} - \underline{\lambda}_{FCho})/\lambda_{min}$	0.3655	0.3784	0.3851	0.3886

Table 4: Relative error mean obtained when computing the lower bound of the smallest eigenvalue of (21) by the methods described in sections II-V.

by each method is reported (Table 5). It is possible to observe that all the considered algorithms exhibit  $O(n^2)$  cost.

Methods	Number of flops
Ma and Zarowski	$3n^2$
SUN1	$4n^2$
SUN2	$7n^2$
Newton	$3n^2$
Fassino-QR	$19n^2$
Fassino-Cholesky	$3n^2$

Table 5: Computational cost of the methods described in sections II-V.

## 7 Conclusion

In this paper the most significant algorithms for the computation of a lower bound of SPD Toeplitz matrices were considered. Comparison studies were carried out in terms of accuracy and computational efficiency. Furthermore, new theoretical relations between some of the algorithms under investigation were derived and, by exploiting the Toeplitz structure of the considered matrices,

more efficient implementations were provided. In particular, it was shown that the methods SUN1 and Fassino–Cholesky are mathematically equivalent, but the latter is more efficient; the method SUN2 can be efficiently implemented and exhibits  $O(n^2)$  cost rather than  $O(n^4)$ . In general, our numerical tests show that all the methods exhibit  $O(n^2)$  cost and that the method SUN2 performs best in terms of accuracy.

**Proof of the equivalence between SUN1 and Fassino–Cholesky**

Let  $\bar{A} \in \mathbb{R}^{m \times n}$ ,  $m \geq n$ ,  $\text{rank}(\bar{A}) = n$ , and let  $\bar{R} = \begin{bmatrix} R \\ 0 \end{bmatrix}$  be the  $R$  factor of the  $QR$  factorization of  $\bar{A}$ , with  $R \in \mathbb{R}^{n \times n}$ . Let  $A \equiv \bar{A}^T \bar{A}$ . We will show that the algorithm SUN1 described in [5], applied to  $A$  is essentially equivalent to the algorithm Fassino–Cholesky described in [7], applied to  $R$ .

**Lemma 1.** *Let*

$$A = \begin{bmatrix} \tilde{A} & b \\ b^T & c \end{bmatrix} \quad \text{and} \quad R = \begin{bmatrix} \tilde{R} & r \\ 0 & \gamma \end{bmatrix},$$

with  $\tilde{A} = A(1 : n-1, 1 : n-1)$ ,  $b = A(1 : n-1, n)$ ,  $c = A(n, n)$ , with  $\tilde{R} = R(1 : n-1, 1 : n-1)$ ,  $r = R(1 : n-1, n)$ ,  $\gamma = R(n, n)$ . Then

$$\tilde{A}^{-1}b = \tilde{R}^{-1}r. \tag{22}$$

*Proof:* Since  $A = R^T R$ ,

$$\begin{aligned} \begin{bmatrix} \tilde{A} & b \\ b^T & c \end{bmatrix} &= \begin{bmatrix} \tilde{R}^T & 0 \\ r^T & \gamma \end{bmatrix} \begin{bmatrix} \tilde{R} & r \\ 0 & \gamma \end{bmatrix} \\ &= \begin{bmatrix} \tilde{R}^T \tilde{R} & \tilde{R}^T r \\ r^T \tilde{R} & \gamma^2 + r^T r \end{bmatrix}. \end{aligned}$$

Hence,  $b = \tilde{R}^T r$ , i.e.,  $r = \tilde{R}^{-T} b$ . Moreover, since  $\tilde{A} = \tilde{R}^T \tilde{R}$ , (22) follows because

$$\tilde{A}^{-1}b = \tilde{R}^{-1} \tilde{R}^{-T} b = \tilde{R}^{-1}r. \tag{23}$$

Known  $\lambda_1^{(S)}$ , a lower bound of the smallest eigenvalue of  $\tilde{A}$ , the algorithm SUN1 yields  $\lambda_2^{(S)}$ , a lower bound of the smallest eigenvalue of  $A$ , based on the following scheme

$$\lambda_2^{(S)} = \frac{d_1 + d_2 \lambda_1^{(S)} - \sqrt{(d_1 + d_2 \lambda_1^{(S)})^2 - 4d_1 \lambda_1^{(S)}}}{2}, \tag{24}$$

where  $d_1 = c - b^T \tilde{A}^{-1} b$  and  $d_2 = 1 + b^T \tilde{A}^{-2} b$ .

Known  $\lambda_1^{(F)}$ , a lower bound of the smallest eigenvalue of  $\tilde{A}$ , the algorithm Fassino–Cholesky yields  $\lambda_2^{(F)}$ , a lower bound of the smallest eigenvalue of  $A$ , based on the following scheme,

$$\lambda_2^{(F)} = \lambda_1^{(F)} \left( 1 + \frac{1}{2} \left( B - \sqrt{B^2 + 4\|x\|_2^2} \right) \right), \quad (25)$$

where  $B \equiv B(\lambda_1^{(F)}, x, \rho) = \|x\|_2^2 + \rho^2/\lambda_1^{(F)} - 1$ ,  $x = \tilde{R}^{-1}r$ , and  $\rho = |\gamma|$ .

**Theorem 4.** *If  $\lambda_1^{(S)} = \lambda_1^{(F)}$ , then  $\lambda_2^{(S)} = \lambda_2^{(F)}$ .*

**Proof:**

Since

$$\begin{aligned} \rho^2 &= c - r^T r \\ &= c - b^T \tilde{A}^{-1} b \\ &= d_1, \end{aligned}$$

and

$$\begin{aligned} \|x\|_2^2 &= b^T \tilde{A}^{-2} b \\ &= d_2 - 1, \end{aligned}$$

we have

$$\begin{aligned} B(\lambda_1^{(F)}, x, \rho) &= \|x\|_2^2 + \rho^2/\lambda_1^{(F)} - 1 \\ &= b^T \tilde{A}^{-2} b + \frac{c - b^T \tilde{A}^{-1} b}{\lambda_1^{(F)}} - 1 \\ &= \frac{1}{\lambda_1^{(F)}} \left( d_1 + d_2 \lambda_1^{(F)} - 2\lambda_1^{(F)} \right). \end{aligned} \quad (26)$$

Replacing (26) in (25) we obtain (24) after simple manipulations.

Concerning the computational cost, the Fassino–Cholesky method requires  $3n^2$  flops.

## Acknowledgment

The research of the second author was partially supported by MIUR grant number 2004015437. The research of the last author was partially supported by the Research Council K.U.Leuven, project OT/05/40 (Large rank structured matrix computations), CoE EF/05/006 Optimization in Engineering (OPTEC), by the Fund for Scientific Research–Flanders (Belgium), Iterative methods in numerical Linear Algebra), G.0455.0 (RHPH: Riemann-Hilbert problems, random matrices and Padé-Hermite approximation), G.0423.05 (RAM: Rational modelling: optimal conditioning and stable algorithms), and by initiated by the Belgian State, IUAP V-22 Modelling). the Interuniversity Attraction Poles

Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization). The scientific responsibility rests with the authors.

## References

- [1] A. Dembo, Bounds on the extreme eigenvalues of positive-definite Toeplitz matrices, *IEEE Trans. Inform. Theory*, **34** (1988), 352-355.
- [2] M. Desai and V. Rao, A characterization of the smallest eigenvalue of a graph, *J. Graph Theory*, **18** (1994), 181-194.
- [3] E.X. Jiang, Bounds for the smallest singular value of a Jordan block with an application to eigenvalue perturbation, *Linear Alg. Appl.*, **197/198** (1994), 691-707.
- [4] E.M. Ma and C.J. Zarowski, On lower bounds for the smallest eigenvalue of a Hermitian matrix, *IEEE Trans. Inform. Theory*, **41** (1995) 539-540.
- [5] W. Sun, Lower bounds of the minimal eigenvalue of a Hermitian positive-definite matrix, *IEEE Trans. Inform. Theory*, **46** (2000), 2760-2762.
- [6] N. Mastronardi and D. Boley, Computing the smallest eigenpair of a symmetric positive definite Toeplitz matrix, *SIAM J. Sci. Comput.*, **20** (1999), 1921-1927.
- [7] C. Fassino, On updating the least singular value: a lower bound, *Calcolo*, **40** (2003), 213-229.
- [8] G.H. Golub and C. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins Univ. Press, Baltimore, MD, 1996.
- [9] N. Mastronardi, M. Van Barel and R. Vandebril, A Schur-based algorithm for computing the smallest eigenvalue of a symmetric positive definite Toeplitz matrix, *Lin. Algebra Appl.*, in press.