Fundamental Journal of Mathematics and Applications, 5 (2) (2022) 98-105
Research Article

FUJMA

Fundamental Journal of Mathematics and Applications

Journal Homepage: *www.dergipark.org.tr/en/pub/fujma*
ISSN: 2645-8845
doi: https://dx.doi.org/10.33401/fujma.996668

Computation of the Solutions of Lyapunov Matrix Equations with Iterative Decreasing Dimension Method

Oğuzer Sinan[1*], Sefa Baydak[2], Ahmet Duman[3] and Kemal Aydın[4]

[1] *Department of Computer Technology, Eregli Kemal Akman Vocational School, Necmettin Erbakan University, Konya, Turkey*
[2] *Graduate School of Natural Sciences, Necmettin Erbakan University, Konya, Turkey*
[3] *Department of Matematics, Faculty of Science, Necmettin Erbakan University, Konya, Turkey*
[4] *Department of Mathematics, Faculty of Science, Selcuk University, Konya, Turkey*
[*] *Corresponding author*

Article Info

Keywords: *Hurwitz stability, IDDM, Lyapunov matrix equations, Schur stability*
2010 AMS: *39A30, 34K20, 65F45*
Received: *20 September 2021*
Accepted: *19 March 2022*
Available online: *28 March 2022*

Abstract

The existence of a solution of continuous and discrete-time Lyapunov matrix equations was studied. Both Lyapunov matrix equations are transformed into a matrix-vector equation and the solution of the obtained new system was examined. The iterative decreasing dimension method (IDDM) was implemented for solving the generated matrix-vector equation. Computations have been done with Maple procedures that run the constituted algorithms.

1. Introduction

The systems are

$$y'(t) = Ay(t) \tag{1.1}$$

and

$$y(n+1) = Ay(n), \quad n \in \mathbb{Z}, \tag{1.2}$$

respectively differential equation system and difference equation system considered.
In the system (1.1) $y(t) = \left(y_1(t), y_2(t), \ldots, y_N(t)\right)^T$, $y_i(t)\,(i = 1,2,\cdots,N)$ are differentiable functions. The coefficient matrix of systems is $A \in M_N(\mathbb{C})$. $M_N^P(\mathbb{C})$ and $M_N(\mathbb{C})$ respectively will denote the set of all $N \times P$ matrices and set of square matrices of size $N \times N$ that matrices elements are complex numbers.
Hurwitz stability is well known in the literature. Regarding the equation system (1.1), in order for system to be Hurwitz stable, the real part of all the eigenvalues of A must be less than zero. Another qualification of Hurwitz stability of equation system (1.1) is concerning with continuous-time Lyapunov matrix equation

$$A^*H + HA = -I \tag{1.3}$$

that has a unique solution under $H = H^* > 0$ condition where I is unit matrix and A^* is adjoint of the matrix A.
Schur stability is well known in the literature too. In accordance with the spectral criterion, all eigenvalues of the matrix A must fall into the unit disc so that the equation system (1.2) get Schur stable [1, 2]. Another occurrence for equation system (1.2) is that there exists and unique positive definite $H = H^*$ matrix satisfying the discrete-time Lyapunov matrix equation

$$A^*HA - H = -I. \tag{1.4}$$

Email addresses and ORCID numbers: **osinan@erbakan.edu.tr, 0000-0002-8742-5372 (O. Sinan), sbaydak@erbakan.edu.tr, 0000-0002-0337-3342 (S. Baydak), aduman@erbakan.edu.tr, 0000-0002-4022-5285 (A. Duman), kaydin@selcuk.edu.tr, 0000-0001-7822-3384 (K. Aydın)**

2. From Lyapunov matrix equations to linear algebraic equation

The *Kronecker product* of B and C denoted as $B\otimes C$ and the *Kronecker sum* of A and D, denoted by $A\oplus D$, is defined in [3] as the expression $A\oplus D = A\otimes I_S + I_N\otimes D$. The *VEC* operator is a vector valued function of the U matrix, denoted by $VEC(U)$ which represent a $N\cdot M$ dimensional vector defined in [3] as follows

$$VEC(U) = [u_{11}, u_{21}, \cdots, u_{N1}, u_{12}, \cdots, u_{NS}]^T .$$

A property of Kronecker product that, in [4] is

$$U = CXB^* \Leftrightarrow VEC(U) = (B\otimes C)VEC(X)$$

where $B\in M_S^Q(\mathbb{C})$, $C\in M_N^P(\mathbb{C})$, $D\in M_S(\mathbb{C})$, $U\in M_N^S(\mathbb{C})$ and $X\in M_P^Q(\mathbb{C})$.

2.1. Transformation for continuous-time Lyapunov matrix equation

When matrix equation (1.3) is considered,

$$\begin{aligned}
-I &= A^*HI + IHA\\
VEC(-I) &= VEC(A^*HI + IHA)\\
&= VEC(A^*HI) + VEC(IHA)\\
&= (I\otimes A^* + A^*\otimes I)VEC(H)\\
VEC(-I) &= (A^*\oplus A^*)VEC(H)
\end{aligned}$$

is obtained. $G\in M_{N^2}(\mathbb{C})$ and $G = (A^*\oplus A^*)$, $h = VEC(H)$ and $z = VEC(-I)$ is formed the

$$Gh = z \tag{2.1}$$

matrix-vector equation. This linear algebraic equation has a unique solution if G is non-singular. As well this linear algebraic equation is affair with continuous-time Lyapunov matrix equation. Let $G = (g_{ij})$, $g_{ij}\in\mathbb{C}$ and $A = (a_{ij})$, $a_{ij}\in\mathbb{C}$. The G matrix's computation algorithm entitled as *LyapunovC* is follows.

LyapunovC algorithm

$$g_{(i-1)N+k,\,(j-1)N+l} = \begin{cases} \overline{a_{kl}+a_{ij}} & i=j\,;k=l,\\ \overline{a_{lk}} & i=j\,;k\neq l,\\ \overline{a_{ji}} & i\neq j\,;k=l,\\ 0 & i\neq j\,;k\neq l,\end{cases}$$

for $i,j,k,l = 1,2,\cdots,N$.

2.2. Transformation for discrete-time Lyapunov matrix equation

If the matrix equation (1.4) is taken into account,

$$\begin{aligned}
VEC(-I) &= VEC(A^*HA - H)\\
&= VEC(A^*HA) - VEC(H)\\
&= (A^*\otimes A^*VEC(H)) - VEC(H)\\
VEC(-I) &= (A^*\otimes A^* - I)VEC(H)
\end{aligned}$$

is obtained. On this situation, the matrix vector-equation (2.1) is composed by $G = (A^*\otimes A^* - I)$, $h = VEC(H)$ and $z = VEC(-I)$. This equation is affair with discrete-time Lyapunov matrix equation and has a unique solution if G is invertible. The matrix G computation algorithm entitled as *LyapunovD* is follows.

LyapunovD algorithm

$$g_{(i-1)N+k,\,(j-1)N+l} = \begin{cases} \overline{a_{kl}\,a_{ij}} - 1 & i=j\,;k=l,\\ \overline{a_{lk}\,a_{ji}} & i\neq j\,;k\neq l,\end{cases}$$

for $i,j,k,l = 1,2,\cdots,N$.

3. Solving the $Gh = z$ linear algebraic equation

The equation (2.1) may be solved by varied methods. Iterative decreasing dimension method(IDDM) is one of them which is decreases by one dimension at every step for get to the solution without any pre-processing. This method and the algorithm that processes this method is given in detail in [5, 6]. As synopsis, framework computation of this method has given with equation (3.1) by [5, 6].

$$h = \sum_{k=1}^{N^2} \left(\prod_{l=1}^{k-1} \widehat{R}^{(l)} \right) h_0^{(k)} \tag{3.1}$$

$h_0^{(k)}$ is a special solution that $h_0^{(k)} = \begin{pmatrix} 0 \cdots 0 & \dfrac{z_1^{(k)}}{g_{1s}^{(k)}} & 0 \cdots 0 \end{pmatrix}^T$, where $g_{1s}^{(k)}$ which is the first non-zero elements of first row of matrix $G^{(k)}$. $G^{(k)}$ and $z^{(k)}$ are reduced matrix and vectors, of equation (2.1).

$$G^{(k)} = \begin{cases} G & \text{if } k = 1, \\ G_2^{(k-1)} \widehat{R}^{(k-1)} & \text{if } k \neq 1 \end{cases} \; ; \quad G_2^{(k-1)} = g_{ij}^{(k-1)} \; \begin{matrix} j = 1, \cdots, N^2 - k, \\ i = 2, \cdots, N^2 - k; \end{matrix}$$

$$z^{(k)} = \begin{cases} z & \text{if } k = 1, \\ v^{(k-1)} - G_2^{(k-1)} h_0^{(k-1)} & \text{if } k \neq 1 \end{cases} \; ; \quad v^{(k-1)} = z_j^{(k-1)}, j = 2, \cdots, N^2 - k.$$

$\widehat{R}^{(k)} \in M_{(N^2-k)}^{(N^2-k+1)}(\mathbb{C})$ are matrices which are composed of the base vectors of solution space as

$$\widehat{R}^{(k)} = \left(\begin{array}{c:c} I & 0 \\ \hdashline 0 & r^{(k)} \\ \hdashline 0 & I \end{array} \right) ; \; \widehat{r}_{s,s+j-1}^{(k)} = r_j^{(k)} = -\frac{g_{1j}^{(k)}}{g_{1s}^{(k)}}; \; j = 1, 2, \cdots, N^2 - k - s + 1.$$

In the condition of $s = N^2 - k + 1$, particular cases of $\widehat{R}^{(k)} = \left(\begin{array}{c} I \\ \hdashline 0 \end{array} \right)$ are evident. This method has been arranged for equation (2.1) and has been prepared for computer aided computation. The algorithm named *IDDMforLyapunov* that calculates matrix H with IDDM has been given follows.

IDDMforLyapunov algorithm

`Step 1. Settlementing of input matrix;` $G^{(1)} = G$.

`Step 2. Checking input matrix at initial situation;`

$s = min(j)$ as provided by $g_{1j}^{(1)} \neq 0$, for $j = 1, 2, \cdots, N^2$, if s is not available,there is no exist or unique solution for the G, the algorithm is terminated.

`Step 3. Establishing initial values;`

$$z_{(i-1)N+j}^{(1)} = \begin{cases} -1 & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \quad \text{for } i, j = 1, 2, \cdots, N,$$

$$h_{ij}^{(1)} = \begin{cases} \eta^{(1)} = \dfrac{1}{g_{1s}^{(1)}} & \text{if } N(j-1) + i = s, \\ 0 & \text{if } j \neq s, \end{cases} \quad \text{for } i, j = 1, 2, \cdots, N,$$

$$r_j^{(1)} = -\frac{g_{1,s+j}^{(1)}}{g_{1s}^{(1)}}, \qquad \text{for } j = 1, 2, \cdots, N^2 - s,$$

$$\widehat{R}^{(1)} = \begin{cases} \widehat{r}_{jj}^{(1)} & = 1 & \text{if } j < s, \\ \widehat{r}_{j+1,j}^{(1)} & = 1 & \text{if } j \geq s, \\ \widehat{r}_{sj}^{(1)} & = r_{j-s+1}^{(1)} & \text{if } j \geq s, \end{cases} \quad \text{for } j = 1, 2, \cdots, N^2 - 1.$$

`Step 4.  Iterative computation of solution of matrix` H;

Overall iteration of substeps on hereinafter is continuing for $k = 2,3,\cdots,N^2-1$;

`Step 4.1.  Dimension decreasing for vector` z `and matrix` G;

$$z_i^{(k)} = z_{i+1}^{(k-1)} - g_{i+1,s}^{(k-1)} \cdot \eta^{(k-1)},$$
$$g_{ij}^{(k)} = \begin{cases} g_{i+1,j}^{(k-1)} & \text{if } j < s, \\ g_{i+1,s}^{(k-1)} \cdot r_{j-s+1}^{(k-1)} + g_{i+1,j+1}^{(k-1)} & \text{if } j \geq s, \end{cases} \qquad \text{for } i,j = 1,2,\cdots,N^2-k+1.$$

`Step 4.2.  Checking reduced matrix;`

$s = min(j)$ as provided by $g_{1,j}^{(k)} \neq 0$, for $j = 1,2,\cdots,N^2-k+1$, if s is not available, the algorithm is terminated.

`Step 4.3.  Accumulating the solution;`

$$\eta^{(k)} = \frac{z_1^{(k)}}{g_{1s}^{(k)}}, h_{ij}^{(k)} = h_{ij}^{(k-1)} + \widehat{r}_{N(j-1)+i,s}^{(k-1)} \cdot \eta^{(k)}, \qquad \text{for } j = 1,2,\cdots,N.$$

`Step 4.4.  Successive multiplication of` $\widehat{R}$;

$$r_j^{(k)} = -\frac{g_{1,s+j}^{(k)}}{g_{1s}^{(k)}}, \qquad \text{for } j = 1,2,\cdots,N^2-k-s+1,$$
$$\widehat{r}_{ij}^{(k)} = \begin{cases} \widehat{r}_{ij}^{(k-1)} & \text{if } j < s, \\ \widehat{r}_{is}^{(k-1)} \cdot r_{j-s+1}^{(k)} + \widehat{r}_{i,j+1}^{(k-1)} & \text{if } j \geq s, \end{cases} \qquad \begin{array}{l} \text{for } i = 1,2,\cdots,N^2, \\ \text{for } j = 1,2,\cdots,N^2-k. \end{array}$$

`Step 5.  Computation of IDDMforLyapunov algorithm is completed with the output solution matrix` H.

Thus, if the *IDDMforLyapunov* algorithm gives a solution, this solution require be a symmetric positive defined matrix.

4. Maple procedures

The *ComputeSystem* main procedure calls the some procedures according to the sequence. These procedures are executed the algorithms defined previous sections. This procedure takes four parameter. The first parameter named *ErrTolerance* is a small number that describes the tolerance of comparison with respect to zero in *Step 2* and *Step 4.2* in the $Gh = z$ computation. The second parameter, *TestTolerance*, is a small number used to corroborate that a unique H solution matrix was symmetrically and positively defined. Distinctly, this tolerance value was defined an acceptability limit by any one for special purpose. The third parameter, is allows the choice either continuous time system or discrete-time system computation. At last, fourth parameter is being the coefficient matrix that belong the system (1.1) or (1.2).

```
> restart
> with (Linear Algebra, Dimension, Eigenvalues)
> Continuous :: integer_1 := 0; Discrete :: integer_1 := 1;

> LyapunovC := proc(A :: Matrix) :: Matrix
local i, j, k, l, N, G; N := Dimension(A);
G := Matrix(N_1 · N_2, N_1 · N_2, datatype = complex_8);
for i from 1 to N_1 do for j from 1 to N_2 do
for k from 1 to N_1 do for l from 1 to N_2 do
if i ≠ j and k ≠ l then next fi: if i = j and k = l then
G_{(i−1)·N_1+k,(j−1)·N_2+l} := conjugate(A_{k,l} + A_{i,j}); next; fi :
if i = j then G_{(i−1)·N_1+k,(j−1)·N_2+l} := conjugate(A_{l,k});next;fi :
G_{(i−1)·N_1+k,(j−1)·N_2+l} := conjugate(A_{j,i});
od: od: od: od:return G; end proc:

> LyapunovD := proc(A :: Matrix) :: Matrix
local i, j, k, l, N, G; N := Dimension(A);
G := Matrix(N_1 · N_2, N_1 · N_2, datatype = complex_8);
for i from 1 to N_1 do for j from 1 to N_2 do
```

for k from 1 to N_1 do for l from 1 to N_2 do
if $i = j$ and $k = l$ then
$G_{(i-1)\cdot N_1+k,(j-1)\cdot N_2+l} :=$ conjugate$(A_{i,j} \cdot A_{k,l}) - 1$ else
$G_{(i-1)\cdot N_1+k,(j-1)\cdot N_2+l} :=$ conjugate$(A_{j,i} \cdot A_{l,k})$ fi :
od : od : od : od : return G; end proc :

$>$ IDDMforLyapunov := proc(W :: Matrix) :: Matrix
global $ConclusionSituation$; local $i, j, k, \eta, r, z, H, G, R, Temp$
$DimBase, DimVec, DimMat, RowNumber, CoulumnNumber,$
$IndexNONZERO$; $ConclusionSituation :=$
"Exist and unique solution been computed that provide the
Lyapunov equation."; $DimMat :=$ Dimension(W);
$RowNumber := DimMat_1$; $CoulumnNumber := DimMat_2$;
$DimVec := DimMat_1$; $DimBase :=$ sqrt($DimVec$);
$z :=$ Vector($DimVec$, datatype = complex$_8$);
for i from 1 to $DimBase$ do $z_{DimBase\cdot(i-1)+i} := -1$ od :
$G :=$ Matrix($DimMat$, datatype = complex$_8$);
for i from 1 to $DimMat_1$ do for j from 1 to $DimMat_2$ do
$G_{i,j} := W_{i,j}$; od : od :
$H :=$ Matrix($DimBase$, $DimBase$ datatype = complex$_8$);
$R :=$ Matrix($DimMat_1$, $DimMat_2 - 1$, datatype = complex$_8$);
for $IndexNONZERO$ from 1 to $DimMat_2$ do
if $|G_{1,IndexNONZERO}| > ErrTolerance$ then break fi : od :
if $IndexNONZERO > DimMat_2$ then $ConclusionSituation :=$
"Has no unique solution so system can't provide the
Lyapunov equation!"; return Matrix([0]); fi :
$\eta := z_1 \cdot G^{-1}_{1,IndexNONZERO}$;
for i from 1 to $DimBase$ do for j from 1 to $DimBase$ do
if $DimBase \cdot (i-1) + j = IndexNONZERO$ then $H_{i,j} := \eta$; fi : od : od :
$r :=$ Vector($DimMat_2 - IndexNONZERO$, datatype = complex$_8$);
for j from 1 to $DimMat_2 - IndexNONZERO$ do
$r_j := -G_{1,IndexNONZERO+j} \cdot G^{-1}_{1,IndexNONZERO}$ od :
for j from 1 to $DimMat_2 - 1$ do if $j < IndexNONZERO$ then
$R_{j,j} := 1$ else $R_{IndexNONZERO,j} := r_{j-IndexNONZERO+1}$; $R_{j+1,j} := 1$; fi : od:
$Temp :=$ Vector($DimMat_1$, datatype = complex$_8$);
for k from 1 to $DimMat_1 - 1$ do
for i from 1 to $RowNumber - 1$ do $z_i := z_{i+1} - G_{i+1,IndexNONZERO} \cdot \eta$ od :
for i from 1 to $RowNumber - 1$ do for j from 1 to $CoulumnNumber - 1$ do
if $j < IndexNONZERO$ then $G_{i,j} := G_{i+1,j}$ else
$G_{i,j} := G_{i+1,indexNONZERO} \cdot r_{j-IndexNONZERO+1} + G_{i+1,j+1}$ fi : od : od:
$RowNumber := RowNumber - 1$; $CoulumnNumber := CoulumnNumber - 1$;
for $IndexNONZERO$ from 1 to $CoulumnNumber$ do
if $|G_{1,IndexNONZERO}| > ErrTolerance$ then break fi : od :
if $IndexNONZERO > CoulumnNumber$ then $ConclusionSituation :=$
"Has no unique solution so that system can't provide the
Lyapunov equation!"; return Matrix([0]); fi :
$\eta := z_1 \cdot G^{-1}_{1,IndexNONZERO}$;
for i from 1 to $DimBase$ do for j from 1 to $DimBase$ do
$H_{i,j} := H_{i,j} + R_{DimBase\cdot(i-1)+j,IndexNONZERO} \cdot \eta$ od : od :
$r :=$ Vector($CoulumnNumber - IndexNONZERO$, datatype = complex$_8$);
for j from 1 to $CoulumnNumber - IndexNONZERO$ do
$r_j := -G_{1,IndexNONZERO+j} \cdot G^{-1}_{1,IndexNONZERO}$ od :
for i from 1 to $DimMat_1$ do $Temp_i := R_{i,IndexNONZERO}$ od:
for j from $IndexNONZERO$ to $CoulumnNumber - 1$ do
for i from 1 to $DimMat_1$ do $R_{i,j} := Temp_i \cdot r_{j-IndexNONZERO+1} + R_{i,j+1}$
od: od: od: return H; end proc :

$>$ IsCorroborate := proc(H :: Matrix) :: boolean
global $ValidationTest$; local $i, j, N, EigVal$; $N :=$ Dimension(H);
for i from 1 to $N_1 - 1$ do for j from $i+1$ to N_2 do
if $|H_{j,i} - H_{i,j}| < TestTolerance$ then next fi :

```
ValidationTest := "Symmetry situation is out of accepted tolerance
value!";return false; od: od:EigVal := Eigenvalues(H);
for i from 1 to N1 do if ℜ(EigVal_i) ≥ TestTolerance then next fi:
ValidationTest := "Positivity of solution matrix is out of
 accepted tolerance value!";return false; od: ValidationTest :=
"Both situations that symmetry and positivity of solution matrix
 been in accepted tolerance range."; return true; end proc:

> ComputeSystem := proc
(argErrTolerance :: float,argTestTolerance :: float,
EqType :: integer_1,A :: Matrix) :: boolean
global ErrTolerance,TestTolerance,boolResult,txtResult; local G,H;
if argErrTolerance < 10.^-13 then ErrTolerance := 10.^-13
 elif argErrTolerance > 10.^-4 then ErrTolerance := 10.^-4
 else ErrTolerance := argErrTolerance fi:
if argTestTolerance < 10.^-13 then TestTolerance := 10.^-13
 elif argTestTolerance > 10.^-4 then TestTolerance := 10.^-4
 else TestTolerance := argTestTolerance fi:
if EqType = 1 then G := LyapunovD(A) elif EqType = 0 then
G := LyapunovC(A) fi: print('G' = G);
H := IDDMforLyapunov(G); print(ConclusionSituation);
if H = [0] then return false fi: print('H' = H);
boolResult := IsCorroborate(H);
print(ValidationTest,tolerance value is = TestTolerance);
print("The related system is asymptotic stable.");
return boolResult; end proc:
```

Example 4.1.

$> A := Matrix(2, 2, [[1, -3], [2, -4]], datatype = complex_8)$

$$A := \begin{bmatrix} 1.0+0.I & -3.0+0.I \\ 2.0+0.I & -4.0+0.I \end{bmatrix}$$

$> ComputeSystem(10^{-13}, 10^{-10}, continuous, A)$

$$G = \begin{bmatrix} 2.0+0.I & 2.0+0.I & 2.0+0.I & 0.+0.I \\ -3.0+0.I & -3.0+0.I & 0.+0.I & 2.0+0.I \\ -3.0+0.I & 0+0.I & -3.0+0.I & 2.0+0.I \\ 0.+0.I & -3.0+0.I & -3.0+0.I & -8.0+0.I \end{bmatrix}$$

"Exist and unique solution been computed that provide the continuous-time Lyapunov matrix equation."

$$H = \begin{bmatrix} 1.83333333299999990+0.I & -1.16666666700000010+0.I \\ -1.16666666700000010+0.I & 1.0+0.I \end{bmatrix}$$

"Both situations that symmetry and positivity of solution matrixbeen in accepted tolerance range.",

tolerace value is $1.000000000 \cdot 10^{-10}$, *"The related system is asymptotic stable."*

$> ComputeSystem(10^{-13}, 10^{-10}, discrete, A)$

$$G = \begin{bmatrix} 0.0+0.I & 2.0+0.I & 2.0+0.I & 4.0+0.I \\ -3.0+0.I & -5.0+0.I & -6.0+0.I & -8.0+0.I \\ -3.0+0.I & -6.0+0.I & -5.0+0.I & -8.0+0.I \\ 9.0+0.I & 12.0+0.I & 12.0+0.I & 15.0+0.I \end{bmatrix}$$

"Has no unique solution so that can't provide the discrete-time Lyapunov matrix equation!"

Example 4.2.

$> A := Matrix(2, 2, [[0, 0], [0, 0]], datatype = complex_8)$

$$A := \begin{bmatrix} 0.0+0.I & 0.0+0.I \\ 0.0+0.I & 0.0+0.I \end{bmatrix}$$

$> \mathit{ComputeSystem}(10^{-13}, 10^{-10}, discrete, A)$

$$G = \begin{bmatrix} -1.0+0.I & 0.0+0.I & 0.0+0.I & 0.0+0.I \\ 0.0+0.I & -1.0+0.I & 0.0+0.I & 0.0+0.I \\ 0.0+0.I & 0.0+0.I & -1.0+0.I & 0.0+0.I \\ 0.0+0.I & 0.0+0.I & 0.0+0.I & -1.0+0.I \end{bmatrix}$$

"Exist and unique solution been computed that provide the discrete-time Lyapunov matrix equation."

$$H = \begin{bmatrix} 1.0+0.I & 0.0+0.I \\ 0.0+0.I & 1.0+0.I \end{bmatrix}$$

"Both situations that symmetry and positivity of solution matrix been in accepted tolerance range.",

tolerace value is $1.000000000 \cdot 10^{-10}$, *"The related system is asymptotic stable."*

5. Conclusion

On discrete set of double precision computer numbers, γ the base of number system, ε_0 the minimal positive number, ε_∞ the maximal number, and ε_1 is the step of computer numbers on the interval from 1 to γ. Thus, let be $v \in [-\varepsilon_\infty, -\varepsilon_0] \cup [\varepsilon_0, \varepsilon_\infty]$, any memorizable double precision computer number is $v_{dp} = v(1+\alpha)+\beta$, $|v - v_{dp}| \leq \varepsilon_1|v| + \varepsilon_0$, $|\alpha| \leq \varepsilon_1$, $|\beta| \leq \varepsilon_0$, $\alpha \cdot \beta = 0$ (see for example[1] and [2]). The selected tolerance values from a particular interval $[10^{-13}, 10^{-4}]$ were used in the evaluation of the inequalities. The lower bound of the interval was chosen to be a larger number than ε_1, depending on the ε_1 which determines the size of computation error. *TestTolerance* should always be larger than *ErrTolerance* so that the assessment be efficient.

DDM which is inspiration for IDDM is described as type of Schur complement domain decomposition method in [7]. Decreasing dimension method (DDM) divides a large system into two smaller systems to be solved separately. To give a general understanding of the computational quantum of DDM was used DDM and Gaussian elimination method to solve a system of n dimension linear algebraic equations in [7]. The explanation in [7] tell us that the computational quantum of the two methods are approximately the same to solve the system whose coefficient matrix is full, but the quantum of DDM is much less than that of Gaussian elimination to solve band matrix equations. IDDM have made an improvement by modifying the method in [7]. Notwithstanding DDM needed some pre-processing situations, without any pre-processing IDDM decreases the dimension of the linear systems, by one order in every step (see for example ([6]). By its very nature, IDDM performs division by a number away from 0 bound up with the *ErrTolerance* value. Thus inherently prevents the error of division by 0.

Acknowledgements

The authors would like to express their sincere thanks to the editor and the anonymous reviewers for their helpful comments and suggestions.

Funding

There is no funding for this work.

Availability of data and materials

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author's contributions

All authors contributed equally to the writing of this paper. All authors read and approved the final manuscript.

References

[1] O. Akın, H. Bulgak, *Linear Difference Equations and Stability Theory* [in Turkish], Selçuk University, Research Center of Applied Mathematics, Konya, 1998.

[2] H. Bulgak, *Pseudo Eigenvalues, Spectral Portrait of a Matrix and Their Connections with Different Criteria of Stability*, In: Error control in Adaptivity in Scientific Computing, H. Bulgak, C. Zenger (editors), NATO Science Series, Kluwer Academic Publishers, 1999, (pp. 95-124).

[3] G. Alexander, *Kronecker Products and Matrix Calculus with Applications*, John Wiley & Sons, N.Y, 1981.

[4] G. H. Golub, C. F. VanLoan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 2013.
[5] K. Aydın, G. C. Kızılkan, A. O. Çıbıkdiken, *Generalized iterative decreasing method*, European J. Pure Appl. Math., **3**(5)(2010), 819-830.
[6] T. Keskin, K. Aydın, *Iterative decreasing dimension algorithm*, Comput. Math. Appl., **53**(1)(2007), 1153-1158.
[7] H. Vang, J. Jiang, *Solution of the system of linear algebraic equations by decreasing dimension*, Appl. Math. Comput., **109**(1)(2000), 51-57.