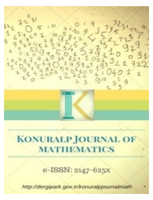
Konuralp Journal of Mathematics, 7 (2) (2019) 486-491

Konuralp Journal of Mathematics

Journal Homepage: www.dergipark.gov.tr/konuralpjournalmath
e-ISSN: 2147-625X



Computation of Sylvester and Stein Matrix Equations' Solutions by Iterative Decreasing Dimension Method

Oğuzer Sinan[1*]

[1]*Necmettin Erbakan University, Eregli Kemal Akman Vocational School, Department of Computer Technology and Programming, Konya-Turkey*
[*]*Corresponding author E-mail: osinan@erbakan.edu.tr, ORCID:0000-0002-8742-5372*

Abstract

Sylvester matrix equation and the Stein matrix equation which have very important applications in the stability analysis of continuous-time and discrete-time linear systems, respectively are studied. The solutions of these equations be possible demonstrated by a number of methods. Been one of them transformation to the system of linear equation was examined. Algorithms that provide transformation to $Ax = f$ system of linear equation was introduced. Iterative Decreasing Dimension Method (*IDDM*) algorithm was applicated for the solution of the obtained system. The *IDDM* algorithm has been evaluated according to the Gaussian elimination type algorithms. The *IDDM* algorithm is evade from divisor of zero based on its algorithmic structure. Depending on the study, codes for calculation were prepared in the MAPLE program code development environment.

Keywords: *Sylvester matrix equation, Stein matrix equation, IDDM, Kronecker product*
2010 Mathematics Subject Classification: *65F05,65Y20*

1. Introduction

The linear matrix equations

$$XA + BX = C \quad , \tag{1.1}$$
$$BYA - Y = C \tag{1.2}$$

where, $A \in \mathscr{M}_N(\mathbb{C})$, $B \in \mathscr{M}_P(\mathbb{C})$ and $C \in \mathscr{M}_P^N(\mathbb{C})$ are called Sylvester matrix equation and Stein matrix equation respectively.

$$BY - YA^{-1} = CA^{-1} \Leftrightarrow BYA - Y = C,$$

by that notification, the equation 1.2 is nothing but a special Sylvester equation with non-singular choices of the A coeffcient matrix. The continuous-time Sylvester matrix equation 1.1 and the discrete-time Stein matrix equation 1.2 have very important applications in the stability analysis of continuous-time and discrete-time linear systems, respectively. These matrix equations enacts a significant role in several applications in science and engineering such as assessment of implicit numerical schemes for partial differential equations, solving techniques for ordinary differential equations, control theory image restoration, signal processing, model reduction, block-diagonalization of matrices and computation of the matrix functions. There are various approaches either direct methods or iterative methods to the solution of these equations. The Bartels-Stewart method [4] and the Hessenberg-Schur method [11] are based on the transforming the coefficient matrices to Schur and Hessenberg form respectively, and then solving the corresponding linear system of equations by a backward substitution process. These approaches are categorized as direct methods. Some iterative methods to solve Sylvester matrix equation have also been proposed that are more appropriate to large sparse[14]. The solutions of X in the equation 1.1 and Y in the equation 1.2 be possible demonstrated in closed form in a number of different methods(see [15]). A salient pair of them is

$$X = -\int_0^\infty e^{tB}Ce^{tA}\,dt, \qquad Y = \sum_{i=0}^\infty B^iCA^i.$$

$\sigma(\cdot)$ denote the spectrum of a matrix and $\lambda_l(\cdot)$ is eigenvalues of a matrix. Precisely, $\lambda_i(B) \in \sigma(B)$ and $\lambda_j(A) \in \sigma(A)$. For all C, the Sylvester matrix equation has a unique solution X if $\sigma(B) \cap \sigma(-A) = \emptyset$ and the Stein matrix equation has a unique solution Y if $\lambda_i(B)\ \lambda_j(A^*) - 1$ has no zero elements[14],[2]. Equation 1.1 and equation 1.2 has special case named as Lyapunov matrix equations. Previous study of us titled as "Computation of the Solutions of Lyapunov Matrix Equations with IDDM", make evaluation respectively differential equation systems and difference equation systems stability. These stuations are called respectivly Hurwitz stability[9] and Schur stability[8] in the literature. The computation of X in the equation 1.1 and Y in the equation 1.2 can be made by reconstructing both matrix equations to the $Ax = f$ system of linear equation. This is a known way. For replacing, definitions and operations required for the system of linear equation are listed as follows.

Let $D \in \mathscr{M}_N^Q(\mathbb{C})$ and $T \in \mathscr{M}_P^M(\mathbb{C})$ be two matrices. The Kronecker product of D and T, denoted as $D \otimes T$, is defined as the partitioned matrix of order ${}_{P\cdot N} \times_{Q\cdot M}$ given by

$$D \otimes T = \begin{pmatrix} d_{11}T & d_{12}T & \dots & d_{1Q}T \\ d_{21}T & d_{22}T & \dots & d_{2Q}T \\ \vdots & \vdots & \ddots & \vdots \\ d_{N1}T & d_{N2}T & \dots & d_{NQ}T \end{pmatrix},$$

the Kronecker sum of A and B is $A \oplus B = I_N \otimes B + A \otimes I_P$[2]. The *vec* operator is a vector valued function of a matrix, denoted by $vec(\cdot)$.

$$vec(C) = \left[\, c_{1,1},\ c_{2,1},\ \cdots,\ c_{P,1},\ c_{1,2},\ \cdots,\ c_{P,N} \,\right]^T.$$

An operation that takes a vector and turns it a matrix is reverse of *vec* operation. This operation is performed with an operator called as *mat*.

$$C = mat(c,P,N) \in \mathscr{M}_P^N(\mathbb{C}) \Leftrightarrow vec(C) = c.$$

$$C_{i,j} = c_{(j-1)P+i}, \quad i : 1,2,\cdots,P, \quad j : 1,2,\cdots,N.$$

Let be taken $Z \in \mathscr{M}_M^Q(\mathbb{C})$, the following property of Kronecker product in [12] will gets to assist for transition to system of linear equation

$$C = TZD^* \Leftrightarrow vec(C) = (D \otimes T)\, vec(Z).$$

When the matrix equation $XA + BX = C$ is considered,

$$\begin{aligned} C &= I_P XA + BXI_N \\ vec(C) &= vec(I_P XA + BXI_N) \\ &= vec(I_P XA) + vec(BXI_N) \\ &= (A^* \otimes I_P + I_N \otimes B)\, vec(X) \\ vec(C) &= (A^* \oplus B)\, vec(X) \end{aligned} \tag{1.3}$$

is obtained. $U = (A^* \oplus B)$, $x = vec(X)$ and $c = vec(C)$ form the $Ux = c$ system of linear equation. Sylvester matrix equation has a unique solution if U is non-singular.

$$U = \begin{pmatrix} \overline{a_{11}}I_P + B & \overline{a_{21}}I_P & \dots & \overline{a_{N1}}I_P \\ \overline{a_{21}}I_P & \overline{a_{22}}I_P + B & \dots & \overline{a_{N2}}I_P \\ \vdots & \vdots & \ddots & \vdots \\ \overline{a_{1N}}I_P & \overline{a_{N2}}I_P & \dots & \overline{a_{NN}}I_P + B \end{pmatrix},$$

then the matrix equation $BYA - Y = C$ is taken into account,

$$\begin{aligned} vec(C) &= vec(BYA - Y) \\ &= vec(BYA) - vec(Y) \\ &= ((A^* \otimes B)\ vec(Y)) - vec(Y) \\ vec(C) &= (A^* \otimes B - I_{NP})\, vec(Y) \end{aligned} \tag{1.4}$$

is came by. On this situation, the system of linear equation $Vy = c$ is composed by $V = (A^* \otimes B - I_{NP})$, $y = vec(Y)$ and $c = vec(C)$. Stein matrix equation has a unique solution if V is non-singular.

$$V = \begin{pmatrix} \overline{a_{11}}B - I_P & \overline{a_{21}}B & \dots & \overline{a_{N1}}B \\ \overline{a_{12}}B & \overline{a_{22}}B - I_P & \dots & \overline{a_{N2}}B \\ \vdots & \vdots & \ddots & \vdots \\ \overline{a_{1N}}B & \overline{a_{N2}}B & \dots & \overline{a_{NN}}B - I_P \end{pmatrix}.$$

1.1. Computer Numbers

At computer computations input data and also intermediate and final results are placed in the computer memory by definite technique. That $\hat{z} = \pm\gamma^{p(z)} m_\gamma(z)$, γ is a base of number system, $p(z)$ is an exponent, $m_\gamma(z)$ is a mantissa of the computer number $\hat{z}$ [6][10]. If $\hat{z} \neq 0$, mantissa of the number satisfies the condition $1/\gamma \leq m_\gamma(z) < 1$ which make presentation $\hat{z}$ unique [6][10]. All numbers which can be placed in the memory their exponents are bounded $p_- \leq p(x) \leq p_+$. The mantissa is decomposed as $m_\gamma = a_1/\gamma + a_2/\gamma + \cdots + a_k/\gamma$, $1 \leq a_1 \leq \gamma - 1$ and $0 \leq a_j \leq \gamma - 1$, $(2 \leq j \leq k)$. Computer numbers are described by quantities of ε_0, ε_1 and ε_∞. 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. There exists the least positive element $\varepsilon_0 = \gamma^{(p_- -1)}$ of computer numbers discreate set [6]. The interval $(-\varepsilon_0, \varepsilon_0)$ has 0 as the unique element. Let be $z \in [\varepsilon_0, \varepsilon_\infty] \cup [-\varepsilon_\infty, -\varepsilon_0]$, any memorizable computer number is $\hat{z} = z(1+\alpha) + \beta$, $|z - \hat{z}| \leq \varepsilon_1 |z| + \varepsilon_0$, $|\alpha| \leq \varepsilon_1$, $|\beta| \leq \varepsilon_0$, $\alpha \cdot \beta = 0$ [6][10].

2. Preparing Linear Equation

The algorithms that perform the composition of U and V matrices formed in equation 1.3 and equation 1.4 are listed below. The U matrix's computation algorithm entitled as *SylvConv* is follows.

SylvConv **algorithm:**

$$\omega_i = (i-1)\,\mathtt{mod}(P)+1, \qquad \omega_j = (j-1)\,\mathtt{mod}(P)+1,$$
$$k = \frac{i-\omega_i}{P}+1, \qquad l = \frac{j-\omega_j}{P}+1, \quad k,l \in \mathbb{N},$$

$$u_{i,j} = \begin{cases} \overline{a_{k,l}} + b_{\omega_i,\omega_j} & \text{if } i=j \\ \overline{a_{l,k}} & \text{if } i=j \text{ and } k\neq l, \\ b_{\omega_i,\omega_j} & \text{if } i\neq j \text{ and } k=l, \\ 0 & \text{if } i\neq j \text{ and } k\neq l, \end{cases} \quad i,j:1,2,\cdots,P\cdot N,$$

MAPLE subroutine for *SylvConv*:

```
> SylvConv := proc(B :: Matrix, A :: Matrix) :: Matrix
local i, j, P, N, bi, bj, U;    bi := 0; bj := 0;
P := Dimension(B);    N := Dimension(A);
U := Matrix($N_1 \cdot P_1$, $N_2 \cdot P_2$, datatype = $complex_8$);
for i from 1 to $P_1 \cdot N_1$ do
if $bi \geq P_1$ then bi := 1 else bi := bi + 1 fi :
for j from 1 to $P_2 \cdot N_2$ do
if $bj \geq P_2$ then bj := 1 else bj := bj + 1 fi :
if i = j then
$U_{i,j}$ := conjugate$\left(A_{\frac{i-bi}{P_1}+1,\ \frac{j-bj}{P_2}+1}\right) + B_{bi,bj}$; next; fi :
if ub = vb then
$U_{i,j}$ := conjugate$\left(A_{\frac{j-bj}{P_2}+1,\ \frac{i-bi}{P_1}+1}\right)$ fi :
if $(i-bi) = (j-bj)$ then $U_{i,j} := B_{bi,bj}$ fi :
od : od : return U; end proc :
```

V matrix's computation algorithm entitled as *SteinConv* is follows.

SteinConv **algorithm:**

$$\omega_i = (i-1)\,\mathtt{mod}(P)+1, \qquad \omega_j = (j-1)\,\mathtt{mod}(P)+1,$$
$$k = \frac{i-\omega_i}{P}+1, \qquad l = \frac{j-\omega_j}{P}+1, \quad k,l \in \mathbb{N},$$

$$v_{i,j} = \begin{cases} \overline{a_{k,l}} \cdot b_{\omega_i,\omega_j} - 1 & \text{if } i=j, \\ \overline{a_{l,k}} \cdot b_{\omega_i,\omega_j} & \text{if } i\neq j. \end{cases} \quad i,j:1,2,\cdots,P\cdot N,$$

MAPLE subroutine for *SteinConv*:

```
> SteinConv := proc(B :: Matrix, A :: Matrix) :: Matrix
local i, j, P, N, bi, bj, V;    bi := 0; bj := 0;
P := Dimension(B);    N := Dimension(A);
V := Matrix($N_1 \cdot P_1$, $N_2 \cdot P_2$, datatype = $complex_8$);
for i from 1 to $P_1 \cdot N_1$ do
if $bi \geq P_1$ then bi := 1 else bi := bi + 1 fi :
for j from 1 to $P_2 \cdot N_2$ do
if $bj \geq P_2$ then bj := 1 else bj := bj + 1 fi :
if i = j then
$V_{i,j}$ := conjugate$\left(A_{\frac{i-bi}{P_1}+1,\ \frac{j-bj}{P_2}+1}\right) \cdot B_{bi,bj} - 1$
else
$V_{i,j}$ := conjugate$\left(A_{\frac{j-bj}{P_2}+1,\ \frac{i-bi}{P_1}+1}\right) \cdot B_{bi,bj}$ fi :
od : od : return V; end proc :
```

3. System of Linear Equation Solving with *IDDM*

The canonical algorithms for solving a system of linear equations is bottomed on Gaussian elimination such as Gauss Jordan, LU Decomposition with some change. For these type algorithms which contain division operations for computation, essential to avoid division by too small numbers, which may throw erroneous conclusion. *IDDM* (*Iterative Decreasing Dimension Method*) algorithm had been prefered for the calculation of equation1.3 and equation1.4. It has been said that the basis of at source of *IDDM* in [3] is same as the well known domain decomposition technique based on Schur complement 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 [3] and [13]. Hereinafter are pseudo codes that serves the implementation of *IDDM* algorithm for solving any $Ax = f$ directly to the computer.

3.1. *IDDM* algorithm

```
Step 1.  Settlementing of input values
$f^{(1)} = f, \quad A^{(1)} = A.$
Step 2.  Checking input matrix at initial situation
    $s = min(j)$ as provided by $a^{(k)}_{1,j} \neq 0, \quad j : 1,2,\cdots,N.$
    If $s$ is not available, there is no exist or unique solution, the algorithm is terminated.
Step 3.  Establishing initial values for the auxiliary $\widehat{R}$ matrix and $x$ solution vector
    $\xi^{(1)} = \frac{f^{(1)}_1}{a^{(1)}_{1,s}}$
    $x^{(1)}_j = \begin{cases} \xi^{(1)} & \text{if } j = s, \\ 0 & \text{if } j \neq s, \end{cases} \quad j : 1,2,\cdots,N.$
    $r^{(1)}_j = -\frac{a^{(1)}_{1,s+j}}{a^{(1)}_{1,s}}, \quad j : 1,2,\cdots,N-s.$
    $\widehat{r}^{(1)}_{i,j} = 0, \quad i : 1,2,\cdots,N, \quad j : 1,2,\cdots,N-1.$
    $\left.\begin{array}{lll} \widehat{r}^{(1)}_{j,j} & = 1 & \text{if } j < s, \\ \widehat{r}^{(1)}_{j+1,j} & = 1 & \text{if } j \geq s, \\ \widehat{r}^{(1)}_{s,j} & = r^{(1)}_{j-s+1} & \text{if } j \geq s, \end{array}\right\} \quad j : 1,2,\cdots,N-1.$
Step 4.  Iterative computation of $x$ solution vector for $k : 2,3,\cdots,N-1$
Step 4.1.  Dimension decreasing for $f$ vector and $A$ matrix
    $f^{(k)}_i = f^{(k-1)}_{i+1} - a^{(k-1)}_{i+1,s} \cdot \xi^{(k-1)}, \quad i : 1,2,\cdots,N-k+1.$
    $a^{(k)}_{i,j} = \begin{cases} a^{(k-1)}_{i+1,j} & \text{if } j < s, \\ a^{(k-1)}_{i+1,s} \cdot r^{(k-1)}_{j-s+1} + a^{(k-1)}_{i+1,j+1} & \text{if } j \geq s, \end{cases} \quad i,j : 1,2,\cdots,N-k+1.$
  Step 4.2.  Checking decreased matrix
    $s = min(j)$ as provided by $a^{(k)}_{1,j} \neq 0, \quad j : 1,2,\cdots,N-k+1.$
    if $s$ is not available, there is no exist or unique solution, the algorithm is terminated.
  Step 4.3.  Accumulating the solution and successive multiplication of $\widehat{R}$
    $\xi^{(k)} = \frac{f^{(k)}_1}{a^{(1)}_{1,s}}.$
    $x^{(k)}_i = x^{(k-1)}_i + \widehat{r}^{(k-1)}_{i,s} \cdot \xi^{(k)}, \quad i : 1,2,\cdots,N.$
    if $k = N-1$ algorithm complete with output $x$ else
    $r^{(k)}_j = -\frac{a^{(k)}_{1,s+j}}{a^{(k)}_{1,s}}, \quad j : 1,2,\cdots,N-k-s+1.$
    $\widehat{r}^{(k)}_{i,j} = \begin{cases} \widehat{r}^{(k-1)}_{i,j} & \text{if } j < s, \\ \widehat{r}^{(k-1)}_{i,s} \cdot r^{(k)}_{j-s+1} + \widehat{r}^{(k-1)}_{i,j+1} & \text{if } j \geq s, \end{cases} \quad i : 1,2,\cdots,N, \; j : 1,2,\cdots,N-k.$
  End of IDDM.
```

Example 3.1. *Applying for solve an Ax=f problem with prepared codes at MAPLE using by IDDM and Gauss elimination algorithms.*
$> A := Matrix(3,3,[[10,2,0],[10.00001,2,0.0001],[2,0,0]], datatype = complex_8)$

$$A := \begin{bmatrix} 10.0+0.I & 2.0+0.I & 0.0+0.I \\ 10.0000099999999996+0.I & 2.0+0.I & 0.000100000000000000006+0.I \\ 2.0+0.I & 0.0+0.I & 0.0+0.I \end{bmatrix}$$

$> f := Vector(3,[5,4.4,5.0001], datatype = complex_8)$

$$f := \begin{bmatrix} 5.0+0.I \\ 4.40000000000000034+0.I \\ 5.00009999999999977+0.I \end{bmatrix}$$

$> Tolerance := 1\cdot 10^{-6}$;x:=IDDM(A,f)

$$x := \begin{bmatrix} 2\ .500049999999999988+0.I \\ -10\ .000249999999999940+0.I \\ -6000\ .25000499999987000+0.I \end{bmatrix}$$

$> Tolerance := 1\cdot 10^{-6}$;x:=GaussE(A,f)

$$x := \begin{bmatrix} 2\ .500049999999999988+0.I \\ -10\ .000249999999999940+0.I \\ -6000\ .25000499999987000+0.I \end{bmatrix}$$

$> Tolerance := 1\cdot 10^{-5}$;x:=IDDM(A,f)

$$x := \begin{bmatrix} 2\ .500049999999999988+0.I \\ -10\ .000249999999999940+0.I \\ -6000\ .05000000000018000+0.I \end{bmatrix}$$

$> Tolerance := 1\cdot 10^{-5}$;x:=GaussE(A,f)

GaussE terminated very close to 0

The same problem was solved by both *IDDM* and Gauss elimination algorithm at example 3.1. Gaussian elimination type algorithms as constructional have divide operations. The fatal termination in the ongoing computing process in progress is the division by 0. For be flowable of progress at practically, test be done with a Tolerance value at the Gaussian algorithms. If $a^{(GaussE)}$ is a divisor as altered element of matrix, $\left|a^{(GaussE)}\right| > Tolerace$ must be satisfied for continuing. In which case the following evaluations can be made for the Gaussian elimination algorithms.
If $0 < \varepsilon_0 \leq Tolerance < \left|a^{(GaussE)}\right|$ then computation is continuing.
If $0 < \varepsilon_0 \leq \left|a^{(GaussE)}\right| \leq Tolerance$ then computation is terminated without solution.
Considering the *IDDM* algorithm written in section 3.1, the $a^{(k)}_{1,j} \neq 0$ expression in *step 2* and *step 4.2* should be rewritten as $\left|a^{(k)}_{1,j}\right| < Tolerance$, in the native computational code. Instead of $a^{(k)}_{1,j}$ will be indicate $a^{(IDDM)}$, both expressions are provision of same altered matrix element. The following evaluations can be made for the *IDDM* algorithm.
If $0 \leq \left|a^{(IDDM)}\right| < Tolerance$ then computation is continuing and result is reasonable. This situation if can provided by all of the line elements of the decreased matrix, algorithm terminate without solution.

Example 3.2. *Let consider $XA + BX = C$ and $BYA - Y = C$ which Sylvester and Stein matrix equations respectively. Let B, A and C matrices be*

$$B = \begin{pmatrix} 4 & 1 & 1 \\ 1+2i & 3 & 3 \\ 2 & 3 & 1 \end{pmatrix}, A = \begin{pmatrix} 0 & 2+i \\ 1 & 1 \end{pmatrix}, C = \begin{pmatrix} 3 & 1 \\ 2 & 0+i \\ 0 & 1 \end{pmatrix}.$$

Following is the computation results of both matrix equations with MAPLE procedures that use the algorithms in this paper. ComputeSylvester and ComputeStein was generated by author at MAPLE environment. Its contain sequential calls to procedures for defined algorithms.
$> Tolerance := 1\cdot 10^{-11}$;X:=ComputeSylvester(B,A,C)

$$X := \begin{bmatrix} 0.65714549+0.20929975I & 0.09229634-0.31185534I \\ -0.05142541-0.16217363I & 0.92408894+0.05244026I \\ 0.33054710-0.36317003I & -1.49056186+0.43109143I \end{bmatrix}$$

$> Tolerance := 1\cdot 10^{-11}$;Y:=ComputeStein(B,A,C)

$$Y := \begin{bmatrix} -0.05875801-0.12168347I & 0.54426145+0.22542349I \\ 0.38600297-1.75618587I & -1.24653039+1.61684455I \\ -0.64034170+2.66115865I & 2.01072655-2.64022197I \end{bmatrix}$$

Conclusion

The IDDM algorithm continues to run when the tolerance is set to large, ie when the range that should be accepted as 0 is expanded. However, since it sees the numbers that can make fine calculations as 0, it will not make calculations with these numbers. In this case, the deterioration of the results will increase. The results are sensible if the Tolerance as control parameter is selected as close to 0. Λ is the successive calculation error associated with the structure of computer numbers, if $Tolerance \leq \Lambda$ then either computation completed with a result or terminated for non uniqueness result, both of situation is unreliable. MVC (Matrix Vector Calculator) computer algebraic system which is present in [7] able to return computation error quantity for own performed calculus.
IDDM is a notable method of calculating the solution of the equations that can be converted into a system of linear equations based on the principles of matrix algebra like Sylvester equation or Stein equation or Lyapunov matrix equations which are the special case of these matrix equations.
The MAPLE worksheet file created during this study process can be accessed at *http://www.oguzersinan.net.tr/maple_worksheets*.

Acknowledgement

The author would like to thank Kemal Aydın (Selcuk University) for a number of productive discussions about computer aided methods of studying mathematical analysis. Also would like to thank responsible editor and anonymous reviewers.

References

[1] Akın, O. and Bulgak, H., *Linear Difference Equations and Stability Theory* [in Turkish], Selçuk University, Research Center of Applied Mathematics, Konya, (1998).
[2] Alexander, G., *Kronecker Products and Matrix Calculus with Applications*, John Wiley & Sons, N.Y., (1981).
[3] K. Aydın, G.Ç. Kızıltan, A.O. Çıbıkdiken,Generalized iterative decreasing method., European Journal of Pure and Aplied Mathematics, Vol:3, No:5,(2010), 819-830.
[4] R. Bartels and G. Stewart, Solution Of The Matrix Equation AX+XB=C, Communications of the ACM., Vol.15, No.9, (1972), 820-826.
[5] A. Bulgak, G. Demidenko, ,I. Matveeva, On location of the matrix spectrum inside an ellipse., Selcuk Journ. Appl. Math., Vol:4, No.1 (2003), 35-41.
[6] Bulgak, H., Pseudo eigenvalues, spectral portrait of a matrix and their connections with different criteria of stability, in: *Error Control and Adaptivity in Scientific Computing*, H. Bulgak and C. Zenger, eds., NATO Science Series, Kluwer Academic Publishers, (1999), 95-124.
[7] H. Bulgak, and D. Eminov, Computer dialogue system MVC, Selcuk Journ. Appl. Math., Vol:2, No.2 (2001), 17-38.
[8] A. Duman, K. Aydın, Sensitivity of Shur stability of systems of linear difference equations with constant coefficients, Scientific Research and Essays, Vol.6, No.28, (2011), 5846-5854.
[9] A. Duman, K. Aydın, Sensitivity of Hurwitz stability of linear differential equation systems with constant coefficients, International Journal of Geometric Methods in Modern Physics, Vol.14, No.6, (2017), 1750084
[10] Godunov, S.K., Antonov, A.G, Kiriljuk O.P and Kostin, V.I.,*Guaranteed Accuracy in Numerical Linear Algebra*, Kluwer Acaden'c Publishers, Dordrecht, Netherlands, (1993).
[11] G.H. Golub, S. Nash, Van C.F.Loan, A Hessenberg-Schur Method For The Problem AX +XB = C, IEEE TransAutomat Control, Vol.24, No.6, (1979), 909-913.
[12] Golub, G.H. and Van Loan, C.F., *Matrix Computations*, The Johns Hopkins University Press, Baltimore,MD, (2013).
[13] T. Keskin and K. Aydın, Iterative decreasing dimension algorithm., Computers and Mathematics with Appl., Vol:53, No.1 (2007), 1153-1158.
[14] A. Sadeghi, A new approach for computing the solution of Sylvester matrix equation., Journal of Interpolation and Approximation in Scientific Computing.,Vol.2016, No.2, (2016), 66-76.
[15] V. Simoncini, Computational Methods for Linear Matrix Equations., SIAM Review., Vol.58, No.3, (2016), 377–441.