Innovativity in Modeling and Analytics Journal of Research
vol. 4, 2019, pp. 38-43,
http://imajor.info/

ISSN 2534-9619

Iterative Solution of the Nonsymmetric Nash-Riccati Equations

Vladislav Krasimirov Tanov
Faculty of Economics and Business Administration,
Sofia University St.Kl.Ohridski, Bulgaria
vtanov@yahoo.com

Abstract. We investigate a nonsymmetric Nash-Riccati equation which has arisen in linear quadratic games for positive systems. There are papers where the stabilising solution of the nonsymmetric Nash-Riccati equation is computed applying the Newton procedure in the literature. Here we introduce the decoupled modification of the linearized Newton method. We provide numerical experiments with the new iteration and compare the results with the classical linearized Newton method.

Key Words: game models, Riccati equation.

1 Introduction

We investigate the nonsymmetric matrix Riccati equation in the special form:

$$\begin{aligned} 0 = \;& \mathcal{R}(X_1, X_2) \\ = \;& -\begin{pmatrix} A^T & 0 \\ 0 & A^T \end{pmatrix}\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} - \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} A - \begin{pmatrix} Q_1 \\ Q_2 \end{pmatrix} \\ & + \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}\begin{pmatrix} S_1 & S_2 \end{pmatrix}\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \end{aligned} \tag{1}$$

where $-A$ is a $n \times n$ Z-matrix, B_j is an $n \times m_j$ nonnegative matrix, $S_j = B_j R_{jj}^{-1} B_j^T$ ($S_j = S_j^T$) is a nonpositive matrix, Q_j is an $n \times n$ symmetric nonnegative matrix, R_{jj} is an $m_j \times m_j$ negative definite matrix for $j = 1, 2$ and X_1, X_2 are $n \times n$ unknown matrices.

The stabilizing solution of (1) is defined in [10] as follows. A left-right stabilizing solution $\begin{pmatrix} \tilde{X}_1 \\ \tilde{X}_2 \end{pmatrix}$ of (1) satisfies that the matrices $A - S_1\tilde{X}_1 - S_2\tilde{X}_2$ and $\begin{pmatrix} A^T - \tilde{X}_1 S_1 & -\tilde{X}_1 S_2 \\ -S_1\tilde{X}_2 & A^T - \tilde{X}_2 S_2 \end{pmatrix}$ are both stable. The Newton method under some conditions for computing the stabilizing solution of (1) is proposed in [10]. Authors in [1] present a study of the Nash equilibria on positive systems, described by the concept of deterministic feedback Nash equilibrium and the concept of open loop Nash equilibrium. The linearized Newton method (LNM) for computing the stabilizing solution of (1) is introduced in [9].

In this paper, we propose a decoupled modification of the linearized Newton method (DMLNM) for computing the stabilizing nonnegative solution of (1). Compared to LNM iteration, the new iteration is more efficient because the algorithm needs less matrix computations in the iterative process. Thus, it requires less CPU time for solving (1) than LNM and it is easy to construct a parallel version of DMLNM. Some numerical experiments are provided to confirm the effectiveness of the DMLNM.

The notation $\mathbf{R}^{s\times q}$ stands for $s \times q$ real matrices. In this investigation we exploit the properties of nonnegative matrices. A matrix $A = (a_{ij}) \in \mathbf{R}^{m\times n}$ is a nonnegative matrix if the inequalities $a_{ij} \geq 0$ are satisfied for all $1 \leq i \leq m$ and $1 \leq j \leq n$. We use an elementwise order relation. The inequality $P \geq Q (P > Q)$ for $P = (p_{ij}), Q = (q_{ij})$ means that $p_{ij} \geq q_{ij} (p_{ij} > q_{ij})$ for all indexes i and j. A matrix $A = (a_{ij}) \in \mathbf{R}^{n\times n}$ is said to be a Z-matrix if it has nonpositive off-diagonal entries. Any Z-matrix A can be written in the form $A = \alpha I - N$ with N being a nonnegative matrix. Each M-matrix is a Z-matrix with if $\alpha \geq \rho(N)$, where $\rho(N)$ is the spectral radius of N. It is called a nonsingular M-matrix if $\alpha > \rho(N)$ and a singular M-matrix if $\alpha = \rho(N)$.

2 Iterative Methods

2.1 The Linearized Newton method (LNM)

The linearized Newton method (LNM) is numerically investigated by Baeva [2] and introduced by C. Ma and H. Lu in [9]. Set $Z_0 = 0$, we compute matrix sequences $\{Y_i\} = \begin{pmatrix} Y_1^{(i)} \\ Y_2^{(i)} \end{pmatrix}\}$ and $\{Z_i = \begin{pmatrix} Z_1^{(i)} \\ Z_2^{(i)} \end{pmatrix}\}$ via following iterations

$$Y_{i+1}(\gamma I_n + A - SZ_i) = (\gamma I_{2n} - D)Z_i - Q \tag{2}$$

$$(\gamma I_{2n} + D - Y_{i+1}S)Z_{i+1} = Y_{i+1}(\gamma I_n - A) - Q, \tag{3}$$

for $i = 0, 1, 2, ...$ and $\gamma < 0$, as sequence $\{Z_i\}$ converge to the solution $\tilde{K}$, when i converge to the infinity. [9]

We consider the decoupled improvement modification of the linearized Newton method by formulas

$$Y_1^{(k)}(\gamma I_n + A) = (\gamma I_n - A^T + Z_1^{(k)}S_1 + Z_2^{(k)}S_2)\, Z_1^{(k)} - Q_1 \tag{4}$$

$$Y_2^{(k)}(\gamma I_n + A) = (\gamma I_n - A^T + Z_1^{(k)}S_1 + Z_2^{(k)}S_2)\, Z_2^{(k)} - Q_2 \tag{5}$$

$$(\gamma I_n + A')Z_1^{(k+1)} = Y_1^{(k)}(\gamma I_n - A + S_1Y_1^{(k)} + S_2Y_2^{(k)}) - Q_1 \tag{6}$$

$$(\gamma I_n + A')Z_2^{(k+1)} = Y_2^{(k)}(\gamma I_n - A + S_1Y_1^{(k)} + S_2Y_2^{(k)}) - Q_2\,. \tag{7}$$

We provide an algorithm to apply iteration (4)-(7):

Algorithm DMLNM.

1. Define the matrix coefficients $A, S1 = S_1, S2 = S_2, Q_1, Q_2, I_n = eye(n)$. Choose the parameters γ and tol. Initialize the initial points $Z_i^{(0)} = 0, i = 1, 2$.

2. Compute T=inverse of (γI_n + A).
3. For k= 0, 1, ..., until $\|\mathcal{R}(Z_1^{(k)}, Z_2^{(k)})\| > \text{tol}$ we compute :

   3.1. Compute $f1 = \gamma I_n - A' + Z_1^{(k)} S1 + Z_2^{(k)} S2$.

   3.2. Compute $Y_1^{(k)} = (f1\, Z_1^{(k)} - Q_1)\, T$ and $Y_2^{(k)} = (f1\, Z_2^{(k)} - Q_2)\, T$.

   3.3. Compute $f2 = \gamma I_n - A + S1\, Y_1^{(k)} + S2\, Y_2^{(k)}$.

   3.4. Compute $Z_1^{(k+1)} = T'\,(Y_1^{(k)}\, f2 - Q_1)$ and $Z_2^{(k+1)} = T'\,(Y_2^{(k)}\, f2 - Q_2)$.

   3.4. Compute $\|\mathcal{R}(Z_1^{(k)}, Z_2^{(k)})\|$ and increase the parameter k.

4. Print $Z_1^{(k)}, Z_2^{(k)}$. End.

3 Numerical examples

We consider a two-players game and we apply iterative methods LNM (2)-(3) and Decoupled Modification of the Linearized Newton Method (DMLNM) (4)-(7) on two numerical examples. We have compared via numerical experiments the Newton method and the ALIDI method for computing the stabilizing solution of (1). The matrix coefficients A, B_i, Q_i and R_{ii} for $i = 1, 2$ are defined using the Matlab description. The numerical experiments are constructed following the approach applied in [2].

Example 1. The matrix coefficients of (1) are:

$$A = \begin{pmatrix} -2.74 & 0.06 & 0.015 & 0.099 \\ 0.2 & -2.5 & 0.064 & 0.08 \\ 0.004 & 0.15 & -2.56 & 0.09 \\ 0.14 & 0.12 & 0.21 & -2.57 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 0.5938 \\ 0.2985 \\ 0.49 \\ 0.98 \end{pmatrix},$$

$$B_2 = \begin{pmatrix} 2.8 & 0 & 0 & 0 \\ 0 & 2.9 & 0 & 0 \\ 0 & 0 & 2.84 & 1.5 \\ 0 & 0 & 1.5 & 1.3 \end{pmatrix},$$

Q_1=eye(4,4)/2; $Q_1(1,1)$=2.0; $Q_1(4,4)$=1.5;
$Q_2 = 0.5 * Q_1$;
R_{11}=-1.909;
R_{22} = -eye(4,4); $R_{22}(1,1)$=-50; $R_{22}(4,4)$=-30;

We apply the above iterative methods for computing the stabilizing solution of (1) with the stop criteria $\|\mathcal{R}_i(Z_1^{(k)}, Z_2^{(k)})\| \leq tol = 1.0e-14$, $i = 1, 2$ and different values of μ. It takes the following values: $\gamma = -5, \gamma = -3$ and $\gamma = -1$. Table 1 presents the computational results for different values of μ. The average CPU time (avCPU) is computed for 100 runs for each value of γ. The average number of iterations (avIt) is computed for 100 runs.

The iterations require the same number of iteration steps while finding the stabilising nonnegative solution of (1) for big values of $|\gamma|$. Yet, the conclusion is that the DMLNM iteration is more effective than LNM method for the lower values of $|\gamma|$.

Table 1. Experiments for Example 1.

	LNM (2)-(3)		DMLNM (4)-(7)	
γ	avIt	avCPU	avIt	avCPU
-5	40	0.0015s	51	0.0017s
-3	24	0.0011s	35	0.0011s
-1	21	0.0008s	21	0.0007s

Example 2. The matrix coefficients are:
A=abs(randn(n))/99; s=max(abs(eig(A)))+2.95;
for i=1:n, A(i,i)=-(A(i,i))-s; end
B_1 = zeros(n,1); B_1(1)=abs(randn(1,1))/5;
B_2 =eye(n,n); B_2(n,n)=n/3; B_2(1,1)=n/5;
Q_1=zeros(n,n); Q_1(1,1)=n/5; Q_1(n,n)=1.5;
R_{11}=-1; Q_2=0.76 Q_1;
R_{22}=-eye(n,n); R_{22}(1,1)=-48; R_{22}(n,n)=-28;

To make experiments, we use different values of n, i.e. $n = 15$ and $n = 35$. We have 500 runs for each value of n, and we calculate average values for iteration numbers "avIt" and computational time "avCPU". The results are described in Table 2.

Table 2. Experiments for Example 2. $\gamma = -4$

	LNM (2)-(3)		DMLNM (4)-(7)	
n	avIt	avCPU	avIt	avCPU
15	9	0.0017s	9	0.0012s
35	13.4	0.0124s	17.6	0.0110s

Example 3. The matrix coefficients are:
A=abs(randn(n))/10; s=max(abs(eig(A)))+1.5;
for i=1:n, A(i,i)=-(A(i,i))-s; end
B_1=abs(randn(n,1))/6;
B_2=eye(n,n); B_2(n,n)=n/5; B_2(1,1)=n/10; B_2(1,n)=randn/10;
Q_1=zeros(n,n); Q_1(1,1)=n/5; Q_1(n,n)=1/n;
Q_2=0.25 Q_1; R_{11}=-1.5;
R_{22}=-eye(n,n); R_{22}(1,1)=-57; R_{22}(n,n)=-27;

We change the dimension n: $n = 15, 35, 60, 80, 100$. We have 100 runs for each value of n with each method LNM and DMLNM. The results are described in Table 3.

Table 3. Experiments for Example 3.

	LNM (2)-(3)		DMLNM (4)-(7)	
n	avIt	avCPU	avIt	av CPU
	$\gamma = -1.5$			
15	14.35	0.0026s	14	0.0019s
35	24	0.0236s	24	0.0147s
60	35.4	0.0744s	34	0.0531s
80	44.2	0.1706s	42	0.1196s
100	53	0.3352s	49	0.2281s
	$\gamma = -3$			
80	22	0.0924s	21	0.0546s
100	26	0.1680s	24.8	0.1052s

Example 4. The matrix coefficients are:
A=abs(randn(n))/50; s=max(abs(eig(A)))+3.25;

for i=1:n, A(i,i)=-(A(i,i))-s; end

B_1=abs(randn(n,1))/10;

B_2=eye(n,n); B_2(n,n)=n/15;

Q_1=1.75 eye(n,n); Q_1(1,n)=n/10; Q_1(n,1)=n/10;

Q_2=0.85 Q_1';

R_{11}=-1.5;

R_{22}=-10 eye(n,n); R_{22}(1,1)=-30; R_{22}(n,n)=-19;

Table 4. Experiments for Example 4. $\gamma = -5$

	LNM (2)-(3)		DMLNM (4)-(7)	
n	avIt	avCPU	avIt	av CPU
15	11	0.0023s	11	0.0016s
35	11	0.0107s	11	0.0081s
60	12	0.0281s	13	0.0251s
80	13	0.0520s	14	0.0410s

We are executing this example for different values of n, and 100 runs are completed for each value of n.

4 Conclusion

Results from experiments show that iteration DMLNM (4)-(7) is more effective than the LNM.

References

[1] T. Azevedo-Perdicoulis and G. Jank, Linear Quadratic Nash Games on Positive Linear Systems, European Journal of Control, 11 (2005), 1–13.

[2] N. Baeva, Improved Iterative Methods for Computing the Nash Equilibrium in Positive Systems, Innovativity in Modeling and Analytics Journal of Research, 4 (2019), 12–27.

[3] J. Engwerda, On the open-loop Nash equilibrium in LQ-games, Journal of Economic Dynamics and Control, 22(5) (1998), 729–762.

[4] Z.-Z. Bai, X.-X. Guo, and S.-F. Xu, Alternately linearized implicit iteration methods for the minimal nonnegative solutions of the nonsymmetric algebraic Riccati equations, Numer. Linear Algebra Appl., 13 (2006), 655–674.

[5] J. Guan and L. Lu, New alternately linearized implicit iteration for M-matrix algebraic Riccati equations, J. Math. Study, 50 (2017), 54–64.

[6] C.-H. Guo, On algebraic Riccati Equations Associated with M-Matrices, Linear Algebra Appl., 439(10) (2013), 2800–2814.

[7] J. Juang and W.-W. Lin, Nonsymmetric algebraic Riccati equations and Hamiltonian-like matrices, SIAM J. Matrix Anal. Appl. , 20 (1999), 228–243.

[8] C.-H. Guo and A. Laub, On the iterative solution of a class of nonsymmetric algebraic Riccati equations, SIAM J. Matrix Anal. Appl., 22(2) (2000), 376–391.

[9] C. Ma and H. Lu, Numerical study on nonsymmetric algebraic Riccati equations, *Mediterranean Journal of Mathematics*, 13(6), 2016, 4961-4973.

[10] G. Jank and D. Kremer, Open loop Nash games and positive systems - solvability conditions for nonsymmetric Riccati equations, Proceedings of MTNS 2004, Katolieke Universiteit, Leuven, Belgium, 2004 (in CD ROM).