22nd International Symposium on
Mathematical Theory of Networks and Systems
July 11-15, 2016. Minneapolis, MN, USA

A subspace intersection based method for faster computation of the storage function for the lossless and all-pass cases

Ashish Kothyari, Cornelis Praagman, Madhu N. Belur and Debasattam Pal

Abstract— **The Algebraic Riccati Equation (ARE) cannot be formulated for the conservative/lossless and all-pass cases, though the notion of 'storage function' is well-defined for these cases too. New properties have been formulated recently about the storage function matrix for this case, which gave rise to new computational procedures. This paper targets improvement of this algorithm by avoiding some key computation intensive steps in minimal polynomial basis computation. We use the Zassenhaus method for basis computation for the sum and intersection of two subspaces. In addition to the conventional Zassenhaus method, for improved numerical accuracy, we propose LU and QR factorization methods with pivoting and compare the results.**

Keywords: **subspace intersection algorithms, Zassenhaus algorithm, minimal polynomial basis, LU factorization, QR factorization**

I. Introduction

The algebraic Riccati equation (ARE) has widespread applications in many optimal control problems, for example, in LQ control, H_∞ and H_2 control [15], [1]. Many conceptual and numerical methods for finding solutions of the ARE have been provided in [15], [3]. The link between storage functions of dissipative systems and solvability of AREs has been studied in [20], [22]. But, for a special class of dissipative systems, namely energy-conservative systems, the ARE is not defined. This is due to the fact that the formulation of ARE depends on a certain regularity condition which is not satisfied by conservative systems. For example, for lossless systems which are conservative with respect to 'positive real supply rate' i.e $u^T y$, where u is the input and y is the output of the system, the ARE is not defined. New properties that are satisfied for the storage function for conservative case have been formulated in [5]. In this paper we use these properties to develop an algorithm to compute the unique storage function for the conservative systems for which the ARE does not exist. Note that for lossless systems, due to the absence of energy-dissipation, the notions of 'available storage' and 'required supply', the two extremal storage functions, coincide: thus the storage function is unique for the lossless case.

The notation used in the paper is standard. The set $\mathbb{R}$ and $\mathbb{C}$ denote the fields of real and complex numbers respectively. The set $\mathbb{R}[\xi]$ denotes the ring of polynomials in ξ with real coefficients. The set $\mathbb{R}^{\mathtt{w}\times\mathtt{p}}[\xi]$ denotes all $\mathtt{w}\times\mathtt{p}$ matrices with entries from $\mathbb{R}[\xi]$. We use $\bullet$ when a dimension need not be specified: for example, $\mathbb{R}^{\mathtt{w}\times\bullet}$ denotes the set of real constant matrices having $\mathtt{w}$ rows. The space $\mathfrak{C}^\infty(\mathbb{R},\mathbb{R}^{\mathtt{w}})$ stands for the space of all infinitely often differentiable functions from $\mathbb{R}$ to $\mathbb{R}^{\mathtt{w}}$, and $\mathfrak{D}(\mathbb{R},\mathbb{R}^{\mathtt{w}})$ stands for the subspace of all compactly supported trajectories in $\mathfrak{C}^\infty(\mathbb{R},\mathbb{R}^{\mathtt{w}})$.

This paper is structured as follows: Section II summarizes preliminaries required in the paper. The properties of ARE solutions are presented in Section II-B. Section III uses the properties stated in Section II-B and provides a numerical algorithm to compute the storage function of conservative systems. Section IV contains a time comparison between the algorithms given in [5] and the algorithms presented in this paper. Section V contains comparison of the numerical accuracy for the algorithms presented in this paper. Some concluding remarks are presented in Section VI. Section VII contains numerical examples to illustrate the algorithm. Due to the sizes of intermediate matrices, these examples are reported on the last page of this paper.

II. Preliminaries

In this section we give a brief introduction to various concepts that are required to formulate and solve the problem addressed in the paper.

A. The behavioral approach

We begin with some essentials of the behavioral approach in control systems. A more detailed explanation can be found in [16].

Definition 2.1: A linear differential behavior $\mathfrak{B}$ is defined as the subspace of infinitely often differentiable functions $\mathfrak{C}^\infty(\mathbb{R},\mathbb{R}^{\mathtt{w}})$ consisting of all solutions to a set of linear ordinary differential equations with constant coefficients, i.e.,

*This work was support in part by IRCC (IITB), SERB (DST) and BRNS, India.

A. Kothyari, M.N. Belur and D. Pal are in the Department of Electrical Engineering, Indian Institute of Technology Bombay, India. Cornelis Praagman is with the Faculty of Economics and Business, University of Groningen, the Netherlands. Email: ashkothyari@ee.iitb.ac.in, c.praagman@rug.nl, belur@iitb.ac.in, debasattam@ee.iitb.ac.in

for $R(\xi) \in \mathbb{R}^{\bullet \times \mathtt{w}}[\xi]$

$$\mathfrak{B} := \left\{ w \in \mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathtt{w}}) \mid R\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{1}$$

The variable w in equation (1) is known as *manifest variable* of the behavior $\mathfrak{B}$. The set of linear differential behaviors with $\mathtt{w}$ manifest variables is denoted as $\mathfrak{L}^{\mathtt{w}}$. Equation (1) is called a *kernel representation* of the behavior $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}$ and sometimes written as $\mathfrak{B} = \ker R(\frac{d}{dt})$. The polynomial matrix $R(\xi)$ is assumed to have full row rank without loss of generality (see [16, Chapter 6]). This assumption guarantees existence of a nonsingular block $P(\xi)$ such that $R(\xi) = \begin{bmatrix} P(\xi) & Q(\xi) \end{bmatrix}$. Conforming to this partition of $R(\xi)$, partitioning w into $w = \begin{bmatrix} y \\ u \end{bmatrix}$, u, y are the input and output of the behavior $\mathfrak{B}$ respectively. Note that this partition is not unique. Such a partition is called an *input-output* partition of the behavior. The input-output partition is called *proper* if $P^{-1}Q$ is a matrix of *proper* rational functions. But the number of components of the input depends only on $\mathfrak{B}$: and that number is denoted as $\mathtt{m}(\mathfrak{B})$, and is called the *input cardinality* of the behavior. The number of components in the output is called the *output cardinality* represented as $\mathtt{p}(\mathfrak{B})$. It is well-known that $\mathtt{p}(\mathfrak{B}) = \operatorname{rank} R(\xi)$ and $\mathtt{m}(\mathfrak{B}) = \mathtt{w} - \mathtt{p}(\mathfrak{B})$.

In the behavioral approach, a system is nothing but its behavior and thus the terms behavior/system will be used interchangeably. There are numerous ways of representing a behavior depending on the modeling of the system. One representation which is useful is the *latent variable representation*: for $R(\xi) \in \mathbb{R}^{\bullet \times \mathtt{w}}$ and $M(\xi) \in \mathbb{R}^{\bullet \times \mathtt{m}}[\xi]$,

$$\mathfrak{B} := \{ w \in \mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathtt{w}}) \mid \exists \ell \in \mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathtt{m}}) \text{ s.t } R(\frac{d}{dt}) w = M(\frac{d}{dt}) \ell \}.$$

Here ℓ is called a latent variable.

Controllability is another important concept required for this paper.

Definition 2.2: A behavior $\mathfrak{B}$ is said to be *controllable* if for every pair of trajectories w_1, $w_2 \in \mathfrak{B}$ there exists $w_3 \in \mathfrak{B}$ and $\tau > 0$ such that

$$w_3(t) = \begin{cases} w_1(t) & \text{for} \quad t \leqslant 0, \\ w_2(t) & \text{for} \quad t \geqslant \tau. \end{cases}$$

We represent the set of all controllable behaviors with $\mathtt{w}$ variables as $\mathfrak{L}^{\mathtt{w}}_{\mathrm{cont}}$. The familiar PBH rank test for controllability has been generalized: a behavior $\mathfrak{B}$ with minimal kernel representation $\mathfrak{B} = \ker R(\frac{d}{dt})$ is controllable if and only if $R(\lambda)$ has constant rank for all $\lambda \in \mathbb{C}$. One of the ways by which a behavior $\mathfrak{B}$ can be represented if (and only if) $\mathfrak{B}$ is controllable is the *image representation*: for $M(\xi) \in \mathbb{R}^{\mathtt{w} \times \mathtt{m}}[\xi]$

$$\mathfrak{B} := \left\{ w \in \mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathtt{w}}) \middle| \exists \ell \in \mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathtt{m}}) \text{ s.t } w = M(\frac{d}{dt}) \ell \right\}. \tag{2}$$

If $M(\xi)$ is such that $M(\lambda)$ has full column rank for all $\lambda \in \mathbb{C}$, then the image representation is said to be an *observable* image representation: this can be assumed without loss of generality (see [16, Section 6.6]).

For a behavior $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}$ we define a Quadratic Differential Form (QDF) $Q_{\Sigma}(w) = w^T \Sigma w$. Such quadratic expressions of the manifest and/or latent variables of the behavior $\mathfrak{B}$ are very common in literature. A detailed explanation on QDFs can be found in [22]. The function $Q_{\Sigma}(w)$ is also called the supply rate. The supply rate is the rate of supply of energy to the system. Dissipative systems are those where the net energy exchange is always an *absorption* when the trajectories are considered which start-from-rest and end-at-rest, i.e. compactly supported. The link with existence of a storage function is well-known for the controllable system case in literature. See [22].

In this paper, we shall be dealing with supply rates Q_{Σ} induced by real symmetric constant nonsingular matrices $\Sigma = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}$ only. For a behavior $\mathfrak{B}$, this with a input/output partition as $w = \begin{bmatrix} u \\ y \end{bmatrix}$, this corresponds to positive real supply rate $2u^T y$. This work focuses on the conservative systems' case for which the algebraic Riccati equation does *not* exist. A conservative system is defined as:

Definition 2.3: Consider a symmetric and nonsingular matrix $\Sigma \in \mathbb{R}^{\mathtt{w} \times \mathtt{w}}$ and a behavior $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}_{\mathrm{cont}}$. The system $\mathfrak{B}$ is called Σ-conservative if

$$\int_{\mathbb{R}} 2u^T y \, dt = 0 \text{ for all } w \in \mathfrak{B} \cap \mathfrak{D}$$

where u and y are the input/output of the system respectively.

Systems conservative with respect to the positive real supply rate are known in the literature as lossless systems. We will be dealing only with lossless systems in this paper. The results in the paper can be extended to systems conservative with respect to other supply rates also.

B. The algebraic Riccati equation (ARE)

Consider a proper input-output partition (u, y) for a controllable dissipative behavior $\mathfrak{B}$ which has the following minimal i/s/o representation:

$$\dot{x} = Ax + Bu, \qquad y = Cx + Du, \tag{3}$$

with $A \in \mathbb{R}^{\mathtt{n} \times \mathtt{n}}, B, C^T \in \mathbb{R}^{\mathtt{n} \times \mathtt{p}}$ and $D \in \mathbb{R}^{\mathtt{p} \times \mathtt{p}}$ and (C, A) observable. We assume here that the number of inputs $\mathtt{m}(\mathfrak{B})$ = number of outputs $\mathtt{p}(\mathfrak{B})$. One of the results relating the Linear Matrix Inequality (LMI), controllable behaviors and storage functions is the *Kalman-Yakubovich-Popov (KYP)* lemma: details in [8, Section 5.6]. For easy reference we

present the *KYP* lemma, in a behavioral context, in the next proposition.

Proposition 2.4: [5] A behavior $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}_{\text{cont}}$, with a controllable and observable minimal i/s/o representation as in equation (3), is Σ-dissipative if and only if there exists a solution $K = K^T \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$ to the LMI

$$\begin{bmatrix} A^TK + KA & KB - C^T \\ B^TK - C & -(D+D^T) \end{bmatrix} \leqslant 0. \tag{4}$$

For systems with $D + D^T > 0$, the Schur complement with respect to $D+D^T$ in LMI (4) provides us with the algebraic Riccati inequality

$$A^TK + KA + (KB - C^T)(D+D^T)^{-1}(B^TK - C) \leqslant 0. \tag{5}$$

The corresponding equation to the inequality (5) is called the algebraic Riccati equation (ARE).

C. Storage functions for lossless systems

For lossless systems $D + D^T = 0$, hence the storage function for such systems cannot found using conventional methods. However for lossless systems the LMI (4) still exists (in fact with equality) and the unique solution to this LMI can be interpreted as storage functions even in the absence of the ARE. The LMI is equivalent to the following matrix equations for lossless systems [5].

$$A^TK + KA = 0 \qquad \text{and} \qquad B^TK - C = 0 \tag{6}$$

New properties of the solution for ARE in terms of the set of trajectories of 'minimal dissipation' have been formulated recently in [21]. We call these set of trajectories 'a Hamiltonian system'. For a lossless system this 'Hamiltonian system' $\mathfrak{B}_{\text{Ham}} = \mathfrak{B} \subseteq \mathfrak{B} \cup \mathfrak{B}^{\perp_\Sigma}$. (See [2, Lemma 11] and [10]). It has been shown in [5] that the ARE solution is closely linked to the static relations that hold between the states in a first order representation of this system. For a lossless behavior $\mathfrak{B}$, the first order representation of the Hamiltonian system $\mathfrak{B}_{\text{Ham}}$ is [5]

$$\begin{bmatrix} \xi I_n - A & 0 & -B \\ 0 & \xi I_n + A^T & -C^T \\ -C & B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ z \\ y \end{bmatrix} = 0. \tag{7}$$

The next result helps to extract the static relations of the first order representation (7) of behavior $\mathfrak{B}_{\text{Ham}}$ and in the process yields the unique storage function for the lossless behavior $\mathfrak{B}$.

Proposition 2.5: [5] Consider a controllable, lossless behavior $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}_{\text{cont}}$ with minimal state space representation as in equation (3). Assuming the McMillan degree of $\mathfrak{B}$ is $\mathtt{n}$, the corresponding Hamiltonian behavior $\mathfrak{B}_{\text{Ham}} = \ker R(\frac{d}{dt})$ where $R(\xi) := \xi E - H$ is the Hamiltonian pencil with $E := \begin{bmatrix} I_n & 0 & 0 \\ 0 & I_n & 0 \\ 0 & 0 & 0 \end{bmatrix}$ and $H := \begin{bmatrix} A & 0 & B \\ 0 & -A^T & C^T \\ C & -B^T & D+D^T \end{bmatrix}$ with $D + D^T = 0$. Then the following statements hold.

1) The Hamiltonian behavior $\mathfrak{B}_{\text{Ham}}$ is not autonomous, i.e. $\det R(\xi) = 0$.
2) There exists a unique symmetric matrix $K \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$ that satisfies
$$\frac{d}{dt}x^TKx = 2u^Ty \qquad \text{for all} \qquad \begin{bmatrix} u \\ y \end{bmatrix} \in \mathfrak{B}_{\text{Ham}} = \mathfrak{B}. \tag{8}$$
3) This matrix $K \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$ satisfies:
$$\text{rank} \begin{bmatrix} & R(\xi) & \\ -K & I & 0 \end{bmatrix} = \text{rank } R(\xi). \tag{9}$$

In [5], it is shown that equation (9) is a necessary and sufficient condition for $K = K^T \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$ to be a storage function for $\mathfrak{B}$. Thus, by equation (9) $[-K \quad I \quad 0]$ is in the row-span of the polynomial matrix $R(\xi)$. This fact is used to develop an algorithm to compute the storage function $K \in \mathbb{R}^{n\times n}$ for lossless systems.

In this paper we compare the time taken by the algorithms presented here with the time taken by the algorithm given in [5, Algorithm 5.5.1]. The algorithm given in [5] requires calculation of the minimal polynomial basis (MPB) *twice*: once for MPB $M(\xi)$ of the matrix $R(\xi)$ and then MPB again for an appropriate submatrix of $M(\xi)$. We briefly review the definition of an MPB. For a polynomial matrix $R(\xi) \in \mathbb{R}^{m\times n}[\xi]$ and rank m, let $Z(\xi) \in \mathbb{R}^{n\times(n-m)}[\xi]$ be a full column rank polynomial matrix such that $R(\xi)Z(\xi) = 0$. For the purpose of this paper, columns of $Z(\xi)$ are called a *minimal polynomial basis* (MPB) if the sum of the column degrees is the least amongst all such $Z(\xi)$. This paper focusses on elimination of the computation of the MPB completely in order to compute K much faster.

D. The row reduced echelon form and Gauss transformations

A matrix $A \in \mathbb{R}^{n\times n}$ is said to be in the *row reduced echelon* form if:

1) All nonzero rows are above all zero rows.
2) For each row, the leading nonzero element (called the pivot) is strictly to the right of the leading nonzero element of all rows above that row.

For example, the matrix $A \in \mathbb{R}^{4\times4}$ below is in the row reduced echelon form:

$$A = \begin{bmatrix} 2 & 3 & 7 & 8 \\ 0 & 3 & 12 & 1 \\ 0 & 0 & 2 & 9 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

A matrix can be converted to a row reduced echelon form with the help of elementary row operations [12]. These operations are represented using Gauss transformation matrices. We write:

$$PA = LU$$

where $P \in \mathbb{R}^{n\times n}$ is a permutation matrix, L and U being lower and upper triangular respectively. This is also called *LU factorization* of the matrix A using partial pivoting. Permutation matrix helps to achieve numerical stability. It also ensures that unreasonably large entries do not appear in L and U when A has small entries. More details on LU factorization can be found in [12, Chapter 3].

E. Zassenhaus sum-intersection algorithm for vector spaces

The Zassenhaus sum-intersection algorithm is used to calculate a basis for the intersection of two vector spaces. Let $S \in \mathbb{R}^{\bullet\times n}$ and $T \in \mathbb{R}^{\bullet\times n}$ be two matrices. Let the dimension of the row span of S (denoted by $< S >_R$) be k_1 and dimension of $< T >_R$ be k_2. Also let the dimension of the vector space $< S >_R + < T >_R$ be n_1 and the dimension of the intersection of the row spaces i.e. $< S >_R \cap < T >_R$ n_2, then, $k_1 + k_2 = n_1 + n_2$. The following steps calculate a basis for $< S >_R \cap < T >_R$.

1) $W = \begin{bmatrix} S & S \\ T & 0 \end{bmatrix}$ is constructed using matrices S and T.
2) Apply successive Gauss transformations on W to bring it to row reduced echelon form.
3) The submatrix $W(n_1+1:n_2, n+1:2n)$ contains a basis for $< S >_R \cap < T >_R$.

For more details about the Zassenhaus algorithm, see [6].

III. Algorithms to find storage functions for lossless/all-pass systems

In this section we discuss the algorithms to calculate the storage functions for lossless/all-pass systems for which the ARE does not exist based on the properties of the storage function we studied in Proposition 2.5 and the Zassenhaus algorithm. By using equation (9), we see that $\begin{bmatrix} -K & I & 0 \end{bmatrix}$ lies in the row span of $R(\lambda) \in \mathbb{R}^{(2\mathrm{n}+\mathrm{p})\times(2\mathrm{n}+\mathrm{p})}$ for all $\lambda \in \mathbb{C}$. Thus, by finding intersection of the row spans of $R(\lambda_i)$, $\lambda_i \in \mathbb{C}, i = 1,2,...,k$, we obtain $\begin{bmatrix} -K & I & 0 \end{bmatrix}$.

A. LU Zassenhaus algorithm implementation

One way to implement the Zassenhaus algorithm is by obtaining LU factorization of the matrix W as discussed in Section II-E. In an LU factorization, a matrix $A \in \mathbb{R}^{n\times n}$ is represented as a product of a lower and upper triangular matrix where $A = LU$, where $L \in \mathbb{R}^{n\times n}$ is a lower triangular matrix and $U \in \mathbb{R}^{n\times n}$ is an upper triangular matrix. The matrix A is converted to an upper triangular form (U) by premultiplying A by Gauss transformation matrices. The product of all Gauss transformation matrices is a lower triangular matrix whose inverse is also a lower triangular matrix. For more on LU factorization see [12, Chapter 3].

In order to extract $\begin{bmatrix} -K & I & 0 \end{bmatrix}$ from the row span of $R(\xi) \in \mathbb{R}^{(2\mathrm{n}+\mathrm{p})\times(2\mathrm{n}+\mathrm{p})}[\xi]$ we propose the following algorithm based on LU factorization within the Zassenhaus method:

Algorithm 1 LU based Zassenhaus implementation

Input: $R(\xi) := \xi E - H \in \mathbb{R}[\xi]^{(2\mathrm{n}+\mathrm{p})\times(2\mathrm{n}+\mathrm{p})}$, a rank 2n polynomial matrix and tolerance $\varepsilon > 0$.

Output: $K \in \mathbb{R}^{\mathrm{n}\times\mathrm{n}}$ with $x^T Kx$ the storage function.

Require: Evaluate $R(\lambda_i) \in \mathbb{R}^{(2\mathrm{n}+\mathrm{p})\times(2\mathrm{n}+\mathrm{p})}$, at

$$\lambda_i \in \mathbb{C}, i = 1,2,\ldots,k \text{ which are the roots of } \xi^k - 1 \tag{10}$$

for k suitably chosen for accuracy ε.

$W := \begin{bmatrix} R(\lambda_1) & R(\lambda_1) \\ R(\lambda_2) & 0 \end{bmatrix}$, $[L,U,P] := lu(W)$

$D := U(2n+p+1:4n+2p-\ell, 2n+p+1:4n+2p)$

Define ℓ as the largest integer such that

$$\|U(4n+2p-\ell+1:4n+2p,:)\|_2 < \varepsilon \tag{11}$$

Let c be the number of rows of D

while $c > n$ **do**

$W := \begin{bmatrix} R(\lambda_d) & R(\lambda_d) \\ D & 0 \end{bmatrix}$, $[L,U,P] := LU(W)$

$D := U(2n+p+1:2n+p+c-\ell, 2n+p+1:4n+2p)$

ℓ is the largest integer such that the condition in equation (11) is satisfied and $d = 3,4,\ldots,k$

end while

$X := D(1:n, n+1:2n)$

$K := -X^{-1}D(1:n, 1:n)$

B. QR Zassenhaus implementation algorithm

A QR factorization of a matrix $A \in \mathbb{R}^{n\times n}$ is given by $A = QR$ where $Q \in \mathbb{R}^{n\times n}$ is an orthogonal matrix and $R \in \mathbb{R}^{n\times n}$ is an upper triangular matrix. The matrix A is brought to upper triangular form by premultiplying A by successive Householder/Givens transformations. More on QR factorization can be found in [12, Chapter 5]. Thus, by factoring the matrix W of Algorithm 1 successively into a product of an orthogonal matrix Q and an upper triangular matrix R, we obtain a basis for the intersection of the two vector spaces by extracting the appropriate rows of the matrix R.

Thus, in order to extract $\begin{bmatrix} -K & I & 0 \end{bmatrix}$ from the row span of $R(\xi) \in \mathbb{R}^{(2\mathrm{n}+\mathrm{p})\times(2\mathrm{n}+\mathrm{p})}$ we propose the following QR factorization based Zassenhaus algorithm:

Algorithm 2 QR based Zassenhaus implementation

Input: $R(\xi) := \xi E - H \in \mathbb{R}[\xi]^{(2\mathtt{n}+\mathtt{p})\times(2\mathtt{n}+\mathtt{p})}$, a rank $2\mathtt{n}$ polynomial matrix and tolerance $\varepsilon > 0$.

Output: $K \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$ with $x^T Kx$ the storage function.

Require: Evaluate $R(\lambda_i) \in \mathbb{R}^{(2\mathtt{n}+\mathtt{p})\times(2\mathtt{n}+\mathtt{p})}$, at

$$\lambda_i \in \mathbb{C}, i = 1,2,\ldots,k \text{ which are the roots of } \xi^k - 1 \tag{12}$$

for k suitably chosen for accuracy ε.

1: $W := \begin{bmatrix} R(\lambda_1) & R(\lambda_1) \\ R(\lambda_2) & 0 \end{bmatrix}$, $[Q,R] := qr(W)$
$D := R(2n+p+1:4n+2p-\ell, 2n+p+1:4n+2p)$
Define ℓ as the largest integer such that

$$\|U(4n+2p-\ell+1:4n+2p,:)\|_2 < \varepsilon \tag{13}$$

Let c be the number of rows of D

2: **while** $c \geq n$ **do**
3: $W := \begin{bmatrix} R(\lambda_d) & R(\lambda_d) \\ D & 0 \end{bmatrix}$, $[Q,R] := qr(W)$
$D := R(2n+p+1:2n+p+c-\ell, 2n+p+1:4n+2p)$
ℓ is the largest integer such that the condition in equation (13) is satisfied and $d = 3,4,...,k$
4: **end while**
5: $X := D(1:n, n+1:2n)$
6: $K := -X^{-1}D(1:n, 1:n)$

IV. Comparison of computation time

The plot in Fig 1 shows the time taken by both LU and QR methods to compute K. Their time is compared with the time taken by the minimal polynomial basis extraction-based algorithm given in [5, Algorithm 5.5.1] for different orders. The experiments were carried out on an Intel Core i3 computer of 3.40 GHz with 4 GB RAM using Ubuntu 14.04 LTS operating system. The relative machine precision ε is 2.22×10^{-16}. Open source numerical computational package Scilab 5.5 has been used to implement the algorithms. In Fig 1, the time taken by the algorithms is plotted against the order of the system. It can be seen from the plot that the LU and QR based methods take the same time for computing K for lossless systems and are ten times faster as compared to the time taken by minimal polynomial basis-based algorithm which is given in [5]. The main drawback of the minimal polynomial based algorithm is that it is not applicable for orders greater than ten, while the LU and QR based algorithms are more stable and can evaluate K for higher order systems. We also compare how accurately the LU and QR based methods calculate K as compared to the minimal polynomial basis-based method in the next section.

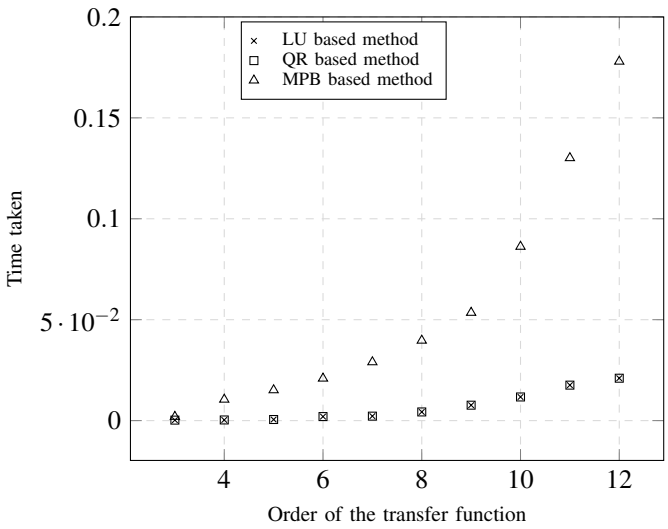

Fig. 1. Plot for time taken by algorithms versus system's order

V. Computational error

In this section we compare the proposed algorithms for accuracy of the obtained K. We compare how accurately the LU and QR based methods calculate K as compared to the minimal polynomial basis-based method. As discussed in Section II-C, the matrix K calculated for the lossless case satisfies the LMI given in equation (4) with equality. Thus, K satisfies the following equation:

$$\begin{bmatrix} A^T K + KA & KB - C^T \\ B^T K - C & 0 \end{bmatrix} = 0.$$

Also, according to Proposition 2.5, the matrix K is symmetric and $\begin{bmatrix} -K & I & 0 \end{bmatrix}$ is in the row span of $R(\xi)$. Thus we consider the following errors associated with computation of K.

$$\mathrm{Err}_{\mathrm{LMI}}(K) := \| \begin{bmatrix} A^T K + KA & KB - C^T \\ B^T K - C & 0 \end{bmatrix} \|_2 \tag{14}$$

$$\mathrm{Err}_{\mathrm{Sym}}(K) := \|K - K^T\|_2 \tag{15}$$

and

$$\mathrm{Err}_{\mathrm{rowspan}}(K) := \max_{\lambda_i} \left\{ \sigma_{2n+1}(\begin{bmatrix} & R(\lambda) & \\ -K & I & 0 \end{bmatrix}) \right\} \tag{16}$$

where the $\lambda_i's$ are the roots of the polynomials $\xi^k - 1$ which are also used to evaluate K from $R(\xi)$ as explained in Algorithm 1 and Algorithm 2. We calculate the errors $\mathrm{Err}_{\mathrm{LMI}}(K), \mathrm{Err}_{\mathrm{Sym}}(K)$ and $\mathrm{Err}_{\mathrm{rowspan}}(K)$ for randomly generated lossless systems. From figures 2, 3 and 4 we see that for most of the cases, error from LU based methods and the minimal polynomial basis-based methods are almost the same though the minimal polynomial basis-based algorithm provides a more symmetric K. The QR based algorithm

provides a less accurate K as compared to the LU based and the minimal polynomial based algorithms.

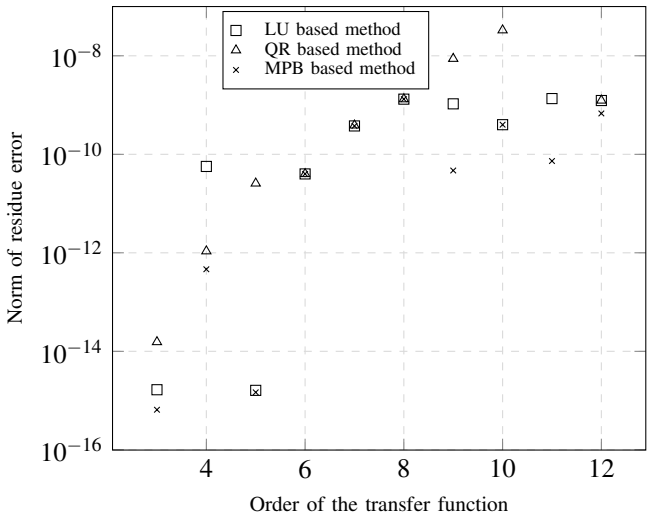

Fig. 2. Plot of $\mathrm{Err}_{\mathrm{LMI}}(K)$ (see equation (14)) residue versus system's order

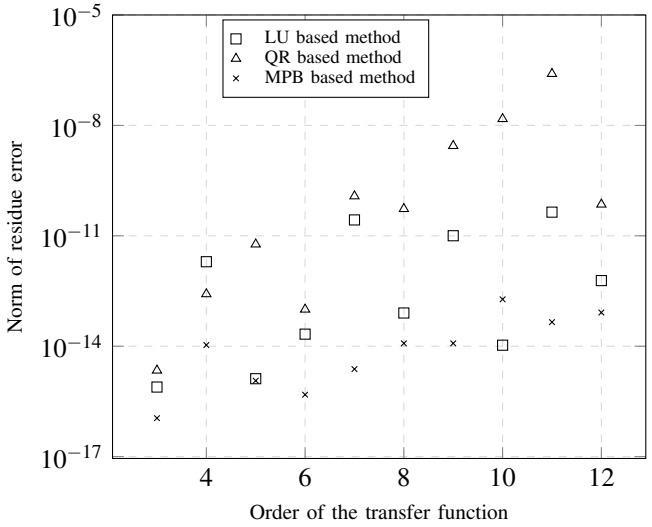

Fig. 3. Plot of $\mathrm{Err}_{\mathrm{Sym}}(K)$ (see equation (15)) residue versus system's order

Both the LU and the QR based Zassenhaus implementation allow explicit control of the tolerance when using the factorizations to reveal the rank (see equation (10) and (12) in Algorithms 1 and 2). In this section we also plot the variation of k of equation (10) of against the tolerance level ε used in equation (11) which specifies the number digits of precision required for identifying the 'almost zero' rows. In Figure 5 horizontal axis represents the value of the negative of the logarithm (in base 10) of the ε used versus the value of k.

The same plot is done in Figure 6 for the k in equation (12) of Algorithm 2 against the tolerance level ε used in equation (13). In Figure 6 horizontal axis represents the value of the negative of the logarithm (in base 10) of the ε used

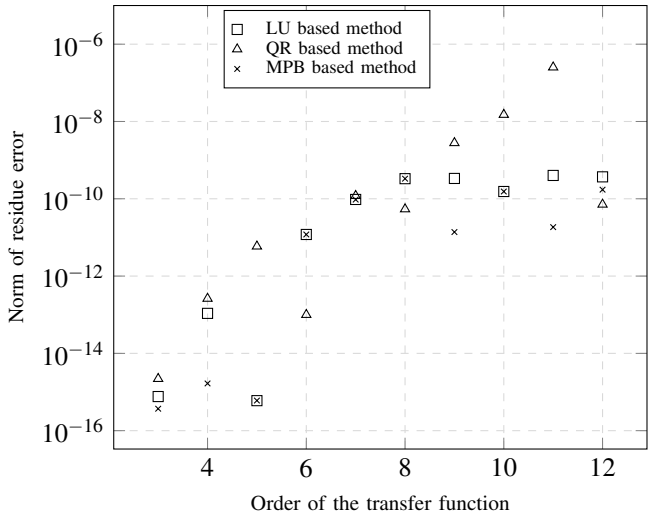

Fig. 4. Plot of $\mathrm{Err}_{\mathrm{Sym}}(K)$ (see equation (15)) residue versus system's order

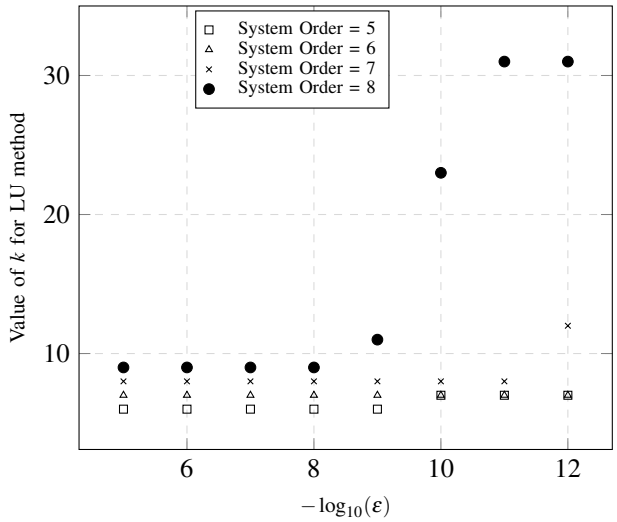

Fig. 5. Plot of value of k (see (10)) versus $-\log_{10}(\varepsilon)$ (see (11))

versus the value of k. From both Figure 5 and 6, we conclude that LU method requires less number of λ's for obtaining K as compared to the QR method for the same tolerance level.

VI. Conclusions

This paper dealt with the problem of calculating the storage function for systems for which the regularity condition required for the existence of the ARE is not satisfied i.e. for lossless systems. Algorithms were developed so as to calculate the storage functions for lossless systems based upon their properties. These algorithms implement Zassenhaus sum-intersection algorithm using LU factorizations and QR factorizations. We studied the computational error for both LU based and QR based algorithms and plotted these for both algorithms against the order of the system. Also, the computation time for obtaining the storage function (for

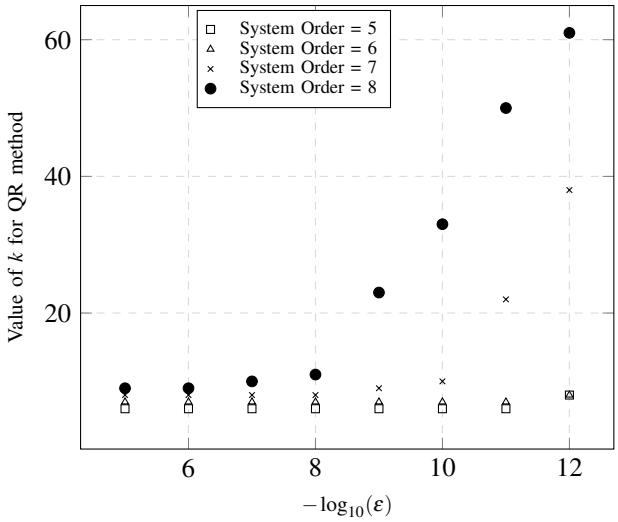

Fig. 6. Plot of value of k (see (12)) versus $-\log_{10}(\varepsilon)$ (see (13))

both LU and QR based methods) was compared with the time taken by the minimal polynomial basis extraction-based algorithm given in [5, Algorithm 5.5.1] and we found that both LU and QR based methods perform faster than minimal polynomial basis extraction-based algorithm. We studied three types of errors: $\text{Err}_{\text{LMI}}(K)$ (see (14)) which measures how well the K obtained satisfies the LMI in equation (4), Err_{Sym} (see (15)) which measures how symmetric is the obtained K and $\text{Err}_{\text{rowspan}}$ (see (16)) which checks whether $\begin{bmatrix}-K & I & 0\end{bmatrix}$ lies in the row span of $R(\xi)$ or not. We compare how accurately K is obtained using LU and QR based algorithms as compared to the minimal polynomial basis-based algorithm. We found that LU based algorithms and minimal polynomial basis-based algorithm calculate K with almost the same accuracy. In Section VII below, the storage function K is calculated for two examples using both the LU and QR based algorithms.

ACKNOWLEDGEMENT

The authors thank C. Bhawal and S. Kumar for sharing the Scilab codes and test cases for the algorithm proposed in [5], against which we compare the methods proposed in this paper.

REFERENCES

[1] A.C. Antoulas, *Approximation of Large-scale Dynamical Systems*, Advances in Design and Control, SIAM, Philadelphia, 2005.

[2] M.N. Belur, H.K. Pillai and H.L. Trentelman, Synthesis of dissipative systems: a linear algebraic approach, *Linear Algebra and its Applications*, vol. 425, no. 2-3, pages 739-756, 2007.

[3] D.A. Bini, B. Iannazzo and B. Meini, *Numerical Solution of Algebraic Riccati Equations*, SIAM, Philadelphia, 2012.

[4] S. Bittanti, A.J. Laub and J.C. Willems (Eds.), *The Riccati Equation*, Springer-Verlag, New York, Inc., USA, 1991.

[5] C. Bhawal, S. Kumar, D. Pal and M. N. Belur, *New Properties of ARE Solutions for Strictly Dissipative and Lossless Systems*, Springer, 2015.

[6] M. Künzer, A. Martin, M. Tentler & M. Wahrheit, Zassenhaus-Algorithms, *mo.mathematik.uni-stuttgart.de/inhalt/beispiel/beispiel1105*

[7] I. Gohberg, P. Lancaster and L. Rodman, *Matrix Polynomials*, Academic Press, New York, 2009.

[8] W.M. Haddad and V. Chellaboina, *Nonlinear Dynamical Systems and Control: a Lyapunov-based Approach*, Princeton University Press, 2008.

[9] S.C. Jugade, D. Pal, R.K. Kalaimani and M.N. Belur, Stationary trajectories, singular Hamiltonian systems and ill-posed interconnection, *Proceedings of the European Control Conference (ECC), Zurich, Switzerland*, pages 1740-1745, 2013.

[10] S.C. Jugade,D. Pal, R.K. Kalaimani and M.N. Belur, Fast modes in the set of minimal dissipation trajectories, *Systems & Control Letters*, 2016.

[11] T. Kailath, *Linear Systems*, N.J. Englewood Cliffs, Prentice-Hall, 1980.

[12] G.H. Golub, and C.F. Van Loan, *Matrix Computations*, Vol. 3., John Hopkins University Press., 2012.

[13] S.R. Khare, H.K. Pillai and M.N. Belur, An algorithm to compute a minimal nullspace basis of a polynomial matrix, *Proceedings of the Mathematical Theory of Networks and Systems (MTNS), Budapest, Hungary*, vol. 5, no. 9, pages 219-224, 2010.

[14] V. Kučera, Algebraic Riccati equation: Hermitian and definite solutions, *In: The Riccati Equation*, Eds. S. Bittanti, A.J. Laub and J.C. Willems, Springer Berlin Heidelberg, pages 53-88, 1991.

[15] P. Lancaster and L. Rodman, *Algebraic Riccati Equations*, Oxford University Press, 1995.

[16] J.W. Polderman and J.C. Willems, *Introduction to Mathematical Systems Theory: a Behavioral Approach*, Springer-Verlag, 1998.

[17] P. Rapisarda, *Linear Differential Systems*, Ph.D. thesis, University of Groningen, The Netherlands, 1998.

[18] P. Rapisarda and H.L. Trentelman, Linear Hamiltonian behaviors and bilinear differential forms, *SIAM Journal on Control and Optimization*, vol. 43, no. 3, pages 769-791, 2004.

[19] D.C. Sorensen, Passivity preserving model reduction via interpolation of spectral zeros, *Systems & Control Letters*, vol. 54, no. 4, pages 347-360, 2005.

[20] H.L. Trentelman and J.C. Willems, The dissipation inequality and the algebraic Riccati equation, *In: The Riccati Equation*, Eds. S. Bittanti, A.J. Laub and J.C. Willems, Springer Berlin Heidelberg, pages 197-242, 1991.

[21] H.L. Trentelman, H.B. Minh and P. Rapisarda, Dissipativity preserving model reduction by retention of trajectories of minimal dissipation, *Mathematics of Control, Signals, and Systems*, vol. 21, no. 3, pages 171-201, 2009.

[22] J.C. Willems and H.L. Trentelman, On quadratic differential forms, *SIAM Journal on Control and Optimization*, vol. 36, no. 5, pages 1703-1749, 1998.

[23] J.C. Willems and H.L. Trentelman, Synthesis of dissipative systems using quadratic differential forms: Parts I & II, *IEEE Transactions on Automatic Control*, vol. 47, no. 1, pages 53-69 & 70-86, 2002.

[24] J.C. Zúniga Anaya and D. Henrion, An improved Toeplitz algorithm for polynomial matrix null-space computation, *Applied Mathematics and Computation*, vol. 207, no. 1, pages 256-272, 2009.

VII. Examples

We consider two examples of lossless systems, both of order 4, and we use both Algorithms 1 and 2 to calculate K for these cases.

Example 7.1: For the system (A,B,C) with $A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -0.000258 & 0 & -0.0357 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ and $C = 10^{-4} \times \begin{bmatrix} 0 & 10.854 & 0 & 541.4 \end{bmatrix}$, the $\lambda_i's$ at which $R(\xi)$ is evaluated are taken from the set of roots of the polynomial $\xi^{40} - 1$. For the above system, $n = 4$ and $p = 1$. The storage function $K \in \mathbb{R}^{4\times4}$ as calculated using Algorithm 1 is:

$$K = \begin{bmatrix} 0.003 & 0 & 0.140 & 0 \\ 0 & 0.248 & 0 & 10.85 \\ 0.140 & 0 & 8.503 & 0 \\ 0 & 10.85 & 0 & 541.43 \end{bmatrix} \times 10^{-4}.$$

Similarly, the storage function K using Algorithm 2 is:

$$K = \begin{bmatrix} 0.003 & 0 & 0.140 & 0 \\ 0 & 0.248 & 0 & 10.85 \\ 0.140 & 0 & 8.503 & 0 \\ 0 & 10.85 & 0 & 541.43 \end{bmatrix} \times 10^{-4}.$$

Example 7.2: For the system (A,B,C) with $A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -0.00112 & 0 & -0.08787 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ and $C = 10^{-3} \times \begin{bmatrix} 0 & 1.407 & 0 & 47.96 \end{bmatrix}$, the $\lambda_i's$ at which $R(\xi)$ is evaluated are taken from the set of roots of the polynomial $\xi^{40} - 1$. For the above system, $n = 4$ and $p = 1$. The storage function $K \in \mathbb{R}^{4\times4}$ as calculated using Algorithms 1 and 2 are:

$$K = \begin{bmatrix} 0.016 & 0 & 0.540 & 0 \\ 0 & 0.69 & 0 & 14.074 \\ 0.540 & 0 & 28.07 & 0 \\ 0 & 14.074 & 0 & 479.695 \end{bmatrix} \times 10^{-4} \text{ and } K = \begin{bmatrix} 0.016 & 0 & 0.540 & 0 \\ 0 & 0.69 & 0 & 14.074 \\ 0.540 & 0 & 28.07 & 0 \\ 0 & 14.074 & 0 & 479.695 \end{bmatrix} \times 10^{-4}$$

respectively.

For each of the two examples above, Algorithms 1 and 2 yield the same storage function (within typical machine precision).