INPUT RETRIEVAL FOR STATE STEERING USING LYAPUNOV MATRIX DIFFERENTIAL EQUATION

Cristian Flutur[1]

For a continuous linear time invariant system (LTI) we approach the matter of numerically computing an input function that steers the state trajectory from the initial state $x(0) = 0$ to a prescribed final state in finite time. The method relies on the numerical solution of a Lyapunov matrix differential equation. Numerical algorithms are provided for the computation of both the solution of this equation and the input function. We prove that all matrices involved in the algorithms are obtained from a $2n \times 2n$ matrix exponential (n is the state space dimension). For the subclass of systems with no symmetric eigenvalues with respect to the imaginary axis we develop a method that relies on a $n \times n$ matrix exponential and we provide the numerical algorithm.

Keywords: Lyapunov matrix differential equation, numerical algorithm, dynamical system, input retrieval

1. Introduction

Matrix differential equations appear in flieds like optimal control (Riccati differentiual equations) or model reduction (Sylvester and Lyapunov differential equations). Recent research addressed both closed form formulas and solution computation for matrix differential equations, either through the Sylvester operator ([1]), ODE integrators with LDL^T factorizations ([2]) or Krylov projection followed by the backward differentiation formula ([3]). In this paper, we use the Lyapunov matrix differential equation (LMDE) to compute suitable input functions that ensure desired state space trajectory steering for dynamical systems.

State trajectory steering has played a central role in control engineering ever since Kalman first addressed the issues of *control function* computation and *controllability* ([4]). For a given dynamical system, the question of how to retrieve the adequate input to generate a certain behavior of either the output or state of said system has drawn the interest of many researchers. Some of these research efforts have approached the matter of *input retrieval* through the inverse of the transfer matrix function ([5]). More recent work has also addressed the case in which invertibility of the transfer matrix is not necessary, but rather lateral invertibility is exploited ([6]). In [7] the output measurement is known and the unknown input is reconstructed via an estimator, exploiting the observability of the system.

[1] "Gheorghe Mihoc – Caius Iacob" Institute of Mathematical Statistics and Applied Mathematics of the Romanian Academy, Romania, e-mail: `cristian.flutur@upb.ro`. The results presented in this article were been funded by the Ministry of Investments and European Projects through the Human Capital Sectoral Operational Program 2014-2020, Contract no. 62461/03.06.2022, SMIS code 153735.

Input function computation is of major importance to optimal control theory, where the goal is to minimize certain criteria involving state variables and/or input functions ([4]). Recent work ([9]) focuses on minimum-energy input functions, computed using only measurement data, without knowing the system dynamics.

Controllability and observability gramians play a key part in many control theory issues, such as model reduction ([10], [11], [12]), optimal control ([4], [13]), network systems ([14], [15]). Whether it is involved in balanced truncation ([16]), optimal control or the investigation of the controllability property, the numerical computation of the controllability gramian plays a key role in many of these applications.

Finite time gramians are, essentially, solutions to LMDEs. Most efforts have focused on the numerical computation of the *infinite* time gramians ([17], [18]), rather than the *finite* time gramians. The lack of such a numerical method is due, in part, to the existence of a certain integral term, whose computation by numerical integration (see [19]) is avoided. This is one aspect that we remedy in this paper.

Contributions. We present two distinct numerical methods for the computation of the integral term, one involving a $2n \times 2n$ matrix exponential and one involving a $n \times n$ matrix exponential and a symmetric Lyapunov equation (n is the dimension of the system). Furthermore, the iterative construction of the solution of the LMDE does not require the system to be stable, as is the case of Lyapunov matrix algebraic equations. The proposed algorithms are compared with IVP (initial value problem) solvers (see [19]). Our methods for input retrieval do not require the inversion or invertibility of the matrix transfer function (as in [6] or [5], nor do they require observability and the construction of an estimator (as in [7]). The input we compute is guaranteed to minimize a certain energy function cost, therefore it has a physical meaning as the minimum energy required to steer the state trajectory from the initial null state to a desired final state, in a given time span.

Notation. The sets of real and complex numbers are denoted by $\mathbb{R}$ and $\mathbb{C}$, respectively, while $\mathbb{R}^n$ and $\mathbb{C}^n$ denote the vector spaces of n-dimensional vectors with real and complex elements, respectively. $A \in \mathbb{R}^{m\times n}$ denotes a $m \times n$ real matrix, A^T denotes its transpose. For a square invertible matrix $A \in \mathbb{R}^{n\times n}$ the inverse is denoted by A^{-1} and A^{-T} denotes the transpose of the inverse, i.e., $(A^{-1})^T$. The Moore-Penrose pseudoinverse of square matrix A is denoted by $A^\dagger$ (see [20]). $\lambda(A)$ denotes the spectrum of A, i.e., the set of its n eigenvalues. The $n-$dimensional identity matrix is denoted by I_n and 0_n denotes the $n \times n$ null matrix. The matrix exponential of a square matrix A is denoted by e^A. The vector subspace spanned by the the columns of matrix A is denoted by Im A. The real and imaginary parts of a complex number $\lambda \in \mathbb{C}$ are denoted by Real(λ) and Imag(λ), respectively. The vector norm in the Hilbert space ℓ^2 is denoted by $|| \cdot ||_2$ (see [21]).

Overview. The paper is organised as follows. In Section 2 we recall basic notions related to LTI systems, controllability of a state, Lyapunov matrix differential equations, controllability gramian and how it hinges on input retrieval for specific state trajectory steering. Section 3 presents the main results, consisting of two methods of the computation of the integral term, the algorithms that compute the solution of the LMDE and the input function, respectively. These results are illustrated on a numerical example in section 4. Section 5 presents the concluding remarks and some directions for future work.

2. Preliminaries

In this section we recall theoretical notions and technical details, necessary throughout the paper. The setup where the discussion takes place is that of LTI systems, described by state-space equations

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad (1)$$
$$y(t) = Cx(t) + Du(t),$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $u(t) \in \mathbb{R}^m$ is the input vector and $y(t) \in \mathbb{R}^p$ is the output vector. Indices n, m and p indicate the dimensions of the system: the number of state variables, the number of inputs and the number of outputs, respectively. Matrices A, B, C and D are constant and have appropriate dimensions. The solution of the first equation in (1) is

$$x(t) = e^{At}x(0) + \int_0^t e^{(t-\tau)A}Bu(\tau)d\tau. \qquad (2)$$

2.1. Controllability gramian

For a continuous LTI system (1) we have the finite-time controllability gramian defined as

$$W(t) = \int_0^t e^{(t-\tau)A}BB^T e^{(t-\tau)A^T} d\tau, \qquad (3)$$

i.e., the solution of the LMDE

$$\dot{W}(t) = AW(t) + W(t)A^T + BB^T. \qquad (4)$$

Matrix $W(t)$ is square, symmetric and semipositive definite, for all $t > 0$. If the system is controllable, then $W(t)$ is strictly positive definite, for all $t > 0$ (see [22]).

2.2. State controllability

We present on the notions of state controllability used in [22].

Definition 2.1. *For a LTI continuous-time system* (1) *a state* $x(t_f) \in \mathbb{R}^n$ *is called* **controllable** *if there exists an input function* $u(\tau)$ *that steers the state trajectory from the initial state* $x(0) = 0$ *to the final desired state* $x_f := x(t_f)$ *in* t_f *time. More precisely, plugging* $u(\tau)$ *in* (2), *yields* $x(t_f)$,

$$x(t_f) = \int_0^{t_f} e^{(t_f-\tau)A}Bu(\tau)d\tau. \qquad (5)$$

Such an input function is expressed as (see [22, Theorem 2.13.1])

$$u(\tau) = B^T e^{(t_f-\tau)A^T}[W(t_f)]^\dagger x(t_f), \qquad (6)$$

where $W(t_f)$ is the solution of the LMDE (4) evaluated at t_f. The use of the pseudoinverse here is due to the fact that, even if $x(t_f)$ is controllable, system (1) may not be controllable and, consequently, $W(t_f)$ may not be invertible. If all states are controllable (i.e., system (1) is controllable) then the pseudoinverse is substituted by the inverse. For a $n-$dimensional system (1) we call the pair (A, B) controllable if controllability matrix $R = \begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix}$ has full row rank.

Equation (6) is a direct consequence of the fundamental result given by [22, Lemma 2.13.1], stating that a state $x(t_f) \in \mathbb{R}^n$ is controllable in a finite time span

$[0, t_f]$ if and only if it belongs to the column space of the solution of the LMDE (4) evaluated at t_f, i.e., $x(t_f) \in \mathrm{Im}\, W(t_f)$.

The input function in (6) minimizes the energy cost function

$$E(u) = \int_{-\infty}^{\infty} |u(\tau)|^2 d\tau$$

and thus it is called *optimal* with respect to the energy cost function (see [22] and [4]).

2.3. **Problem formulation**

We approach the matter of numerically computing $u(\tau)$ that steers system (1) from the initial state $x(0) = 0$ to a prescribed finite $x(t_f) =: x_f$ in t_f time. In other words, we compute $u(\tau)$ from (6) such that (5) holds. Thus, we need to numerically compute

$$W(t_f) = e^{At_f} W(0) e^{A^T t_f} + \int_0^{t_f} e^{(t_f-\tau)A} BB^T e^{(t_f-\tau)A^T} d\tau, \quad \forall t_f > 0. \tag{7}$$

3. **Main result**

Let the interval $[0, t_f]$ be divided in N equal subintervals, each of length h and assume $W(t)$ is constant on each of these subintervals, i.e.,

$$W(t) = W_k, \text{ for } t \in [kh, (k+1)h).$$

The dynamic of $W(t)$ on the k-th subinterval has $t = kh$ as starting time and $W(kh)$ as initial condition (see [8, Section 3.1]). Thus, $W(t)$ becomes

$$W(t) = e^{(t-kh)A} W(kh) e^{(t-kh)A^T} + \int_{kh}^{t} e^{(t-\tau)A} BB^T e^{(t-\tau)A^T d\tau}.$$

Writing the dynamics on the entire interval, i.e., taking $t = (k+1)h$, yields (after some manipulation on the integrating variable) the discrete version of equation (7),

$$W[(k+1)h] = e^{hA} W(kh) e^{hA^T} + \int_0^h e^{\eta A} BB^T e^{\eta A^T} d\eta. \tag{8}$$

Hence that $W(t_f)$ can be numerically computed by iterating formula (8), with the initial condition $W(0) = 0$. The aim is to avoid the computation of

$$G := \int_0^h e^{\eta A} BB^T e^{\eta A^T} d\eta \tag{9}$$

in (8) by numerical integration. The method we propose for the computation follows arguments in [23] and is formulated in the following theorem.

Theorem 3.1. *Consider a continuous time dynamical system* (1) *and a time interval* $[0, t_f]$ *with* N *equal subintervals, each of dimension* h*. Consider now the solution of equation* (4) *evaluated at* t_f*, i.e., formula* (7) *and its discretized version with respect to the equally spaced time interval* $[0, t_f]$, (8)*. Denote*

$$A_0 := \begin{bmatrix} A & BB^T \\ 0_n & -A^T \end{bmatrix}, F := e^{hA} \text{ and } F_0 := e^{hA_0} = \begin{bmatrix} F & M \\ 0_n & F^{-T} \end{bmatrix}. \tag{10}$$

Then the integral term in (9) *satisfies* $G = MF^T$, *where* F *and* M *are submatrices of* F_0, *partitioned as in* (10).

Proof. The non-homogeneous $n-$dimensional dynamic in (4) is immersed in a $2n-$dimensional homogeneous dynamic. We denote

$$X(t) := \begin{bmatrix} W(t) & 0_n \\ I_n & 0_n \end{bmatrix},$$

In terms of this notation, equation (4) is rewritten as

$$\dot{X}(t) = A_0 X(t) + X(t) A_0^T.$$

This equation describes homogeneous (free) dynamics with solution

$$X(t) = e^{A_0 t} X(0) e^{A_0^T t},$$

where the matrix exponential

$$e^{A_0 t} = \begin{bmatrix} e^{At} & M(t) \\ 0_n & e^{-A^T t} \end{bmatrix}$$

keeps the upper block-triangular structure of A_0, its diagonal blocks are the matrix exponentials of the diagonal blocks of A_0 (see [24]) and $M(t) \in \mathbb{R}^{n\times n}$ denotes the right upper block, corresponding to the BB^T block in A_0. Now, considering the dynamics of $X(t)$ on the k-th subinterval of the interval $[0, t_f]$ and following similar arguments as with equations (7) and (8), we obtain

$$X[(k+1)h] = e^{hA_0} X(kh) e^{hA_0^T},$$

which, written explicitly, becomes

$$\begin{bmatrix} W[(k+1)h] & 0_n \\ I_n & 0_n \end{bmatrix} = \begin{bmatrix} e^{hA} & M(h) \\ 0_n & e^{-hA^T} \end{bmatrix} \begin{bmatrix} W(kh) & 0_n \\ I_n & 0_n \end{bmatrix} \begin{bmatrix} e^{hA^T} & 0_n \\ M^T(h) & e^{-hA} \end{bmatrix}.$$

Matrix $M(h)$ will be referred to as M. With notations (10) we obtain $F^{-1} = e^{-hA}$, $F^T = e^{hA^T}$ and $F^{-T} = e^{-hA^T}$. We check the equality in block-position $(1, 1)$ and get

$$W[(k+1)h] = FW(kh)F^T + MF^T. \tag{11}$$

Equations (11) and (8) allow us to identify the integral term in (9), i.e.,

$$G = \int_0^h e^{\eta A} BB^T e^{\eta A^T} d\eta = MF^T. \tag{12}$$

This concludes the proof. □

We are now able to formulate the algorithm to numerically compute the LMDE solution.

Algorithm 3.1. *Given a LTI system* (1) *and a time interval with* N *equal subintervals, each of dimension* h, *compute the LMDE solution defined in* (3) *at* $t_f = Nh$.

(1) *Construct* A_0 *according to* (10) *;*
(2) *Compute* $F_0 := e^{hA_0}$ *using the Padé approximation (see* [26]*);*
(3) *Extract matrices* $F = F_0(1:n, 1:n)$ *and* $M = F_0(1:n, n:2n)$ *and compute* $G = MF^T$.
(4) *Initialise* $W = 0$;

(5) *For* $k = 0 : N - 1$
 (a) $W = FWF^T + G$;

The computation of the matrices F and $G = MF^T$ in Algorithm 3.1 relies on the matrix exponential of the $2n \times 2n$ matrix A_0. We now provide a method that computes matrix G from the matrix exponential of a $n \times n$ matrix and the solution of a continuous algebraic matrix Lyapunov equation.

3.1. **The case of non symmetric eigenvalues**

The method works for systems where matrix A has no symmetric eigenvalues with respect to the imaginary axis. This class of systems also includes strictly stable systems and strictly antistable systems, for which input retrieval and the finite time controllability gramian are often required. For instance, after a stabilizing control law has designed and implemented, the input for a certain trajectory of the (now stable) closed loop system is required ([25]). The result is presented in the following theorem.

Theorem 3.2. *Consider a system* (1) *with no symmetric eigenvalues with respect to the imaginary axis* ($\lambda(A) \cap \lambda(-A) = \emptyset$)*. Let* $W(kh)$ *be the solution to the discrete dynamic Lyapunov equation* (8)*. The integral term* G *in* (9) *is the unique symmetric solution of the continuous matrix algebraic Lyapunov equation*

$$AG + GA^T = (FB)(FB)^T - BB^T. \tag{13}$$

Proof. The product between a square matrix and its matrix exponential is commutative (see [24, Theorem 1.3]). We apply this property on matrix A_0 from (10) and its matrix exponential e^{hA_0}, i.e., $A_0 e^{hA_0} = e^{hA_0} A_0$. Writing this explicitly yields (with the same notations as in (11))

$$\begin{bmatrix} A & BB^T \\ 0 & -A^T \end{bmatrix} \begin{bmatrix} F & M \\ 0 & F^{-T} \end{bmatrix} = \begin{bmatrix} F & M \\ 0 & F^{-T} \end{bmatrix} \begin{bmatrix} A & BB^T \\ 0 & -A^T \end{bmatrix}.$$

More specifically, for block-position $(1, 2)$, we obtain $AM + MA^T = F(BB^T) - (BB)^T F^{-T}$, which multiplied to the right by F^T and substituting $G = MF^T$ (see equation (12)) yields

$$AG + G(F^{-T} A^T F^T) = (FB)(FB)^T - BB^T. \tag{14}$$

We now look at the second term of the left hand side. The product between matrix A and its matrix exponential F is commutative, i.e., $AF = FA$, which can be written as $A = FAF^{-1}$ or (after transposing) $A^T = F^{-T} A^T F^T$. Thus, equation (14) becomes

$$AG + GA^T = (FB)(FB)^T - BB^T,$$

a continuous algebraic matrix Lyapunov equation, where solution G is unique, since $\lambda(A) \cap \lambda(-A) = \emptyset$. Solution G is symmetric, since the free term $(FB)(FB)^T - BB^T$ is also symmetric (see [17]). This concludes the proof. □

3.2. **Input computation**

We now tackle the numerical explicit computation of the input function $u(\tau)$ that steers the state trajectory of a given system (1) from the initial state $x(0) = 0$ to the prescribed final state $x_f = x(t_f)$ in t_f time. The detailed steps are given in the following algorithm.

Algorithm 3.2. *Given a LTI system* (1), *a prescribed final state* $x_f = x(t_f) \in \mathbb{R}^n$, *a time span* $[0, t_f]$ *divided in* N *subintervals, each of length* h, *compute the input matrix* $U \in \mathbb{R}^{m \times N}$, *with* $U(:,k) = u(kh) = u(t)$, *for* $t \in [kh, (k+1)h]$.

(1) *Compute matrices* $F = e^{hA}$ *and* G *from* (9) *either by following the steps in Algorithm 3.1, or by solving equation* (13) *if conditions of Theorem 3.2 are met;*
(2) *Compute* $W(t_f)$ *using Algorithm 3.1;*
(3) *If the system is controllable and* $W(t_f)$ *is invertible, solve the system of linear equations* $W(t_f)z = x_f$ *for* $z \in \mathbb{R}^n$*;*
Alternatively, use the Moore-Penrose pseudoinverse to compute
$z := [W(t_f)]^{\dagger} x_f$*;*
(4) *Initialise* $U = 0_{m \times N}$*;*
(5) *For* $k = 1 : N$
  (a) $z = F^T z$*;*
  (b) $U(:, N-k+1) = B^T z$.

4. **Numerical results**

The section is divided into two parts. In the first part we present a 3−state numerical example, suitable for graphic representation and interpretation. For such an example the LMDE solution in (3) can be computed by hand and so the results of the algorithms can be verified. In the second part we compare the proposed algorithms with Runge-Kutta methods (see [19]) for different simulation steps and different state space dimensions. All simulations have been done on a machine with Apple M2 Pro processor, 16 GB of RAM and MATLAB® R2023b.

4.1. **Graphical illustration**

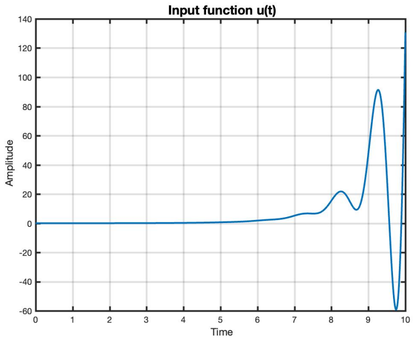
Let system (1) be defined by matrices

$$A = \begin{bmatrix} -2 & 6 & 0 \\ -6 & -2 & 0 \\ 0 & 0 & -1 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}.$$

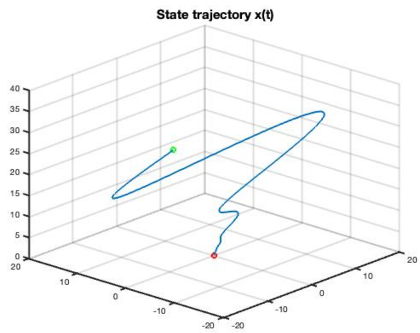
Matrices C and D are not involved in our discussion. The system is controllable, with 3 states and 1 input channel. The system is also stable, with one pair of complex-conjugate eigenvalues and one real eigenvalue. We want the state trajectory to be steered towards the final state $x_f = \begin{bmatrix} 3 & 11 & 20 \end{bmatrix}^T$ in $t_f = 10$ time. The chosen simulation step is $h = 10^{-5}$, resulting in $N = 10^6$ subintervals. The LMDE solution evaluated at t_f is

$$W(10) = \begin{bmatrix} 0.137 & -0.037 & 0.066 \\ -0.037 & 0.112 & -0.133 \\ 0.066 & -0.133 & 0.499 \end{bmatrix}$$

with an invertibility condition number $\kappa(W(10)) = 8.318$. This value allows for a numerically safe solution of the linear equations system $W(10)z = x_f$ and the use of the Moore-Penrose pseudoinverse is not required. We get the sampled version of the input function $u(\tau)$, namely $U(kh)$. With this input we compute and simulate the state trajectory, using the zero order hold discretisation on the original system (the numerical method is described in [27]). The final state obtained with this simulation is $\hat{x}_f = \begin{bmatrix} 2.998 & 11 & 19.998 \end{bmatrix}^T$, with an error equal to $||x_f - \hat{x}_f||_2 = 0.002$. The solution of the LMDE was computed in $t^* = 0.25$ seconds. We provide the graphical representations of both the input function and the state trajectory (Figure 1).



(A) Input function $u(\tau)$



(B) State trajectory $x(t)$ from $x(0) = 0$ to $\hat{x}_f = \begin{bmatrix} 2.998 & 11 & 19.998 \end{bmatrix}^T$

FIGURE 1. Input function and state trajectory

For this example we compute the solution to the LMDE (4) using the 4 stage Runge-Kutta method and the obtained error is $||x_f - \hat{x}_{f_{RK_4}}||_2 = 0.002$, while the computation time was $t_{RK_4} = 10.13$ seconds.

4.2. **Comparison with IVP solver**

We now consider solutions of LMDE (4), computed with both the proposed algorithms based on matrix exponential (reffered to as EXPM in Table 1) and IVP solvers (reffered to as IVPS in Table 1), for different values of dimension n and of simulation step h. The system is generated with MATLAB function **rss**. We use execution time as a performance measure, denoted t^* for our algorithms and t for the IVP solver. The results are presented in Table 1.

h / n	10^{-3}		10^{-4}		10^{-5}	
Method	EXPM	IVPS	EXPM	IVPS	EXPM	IVPS
$n = 10$	$\mathbf{t^* = 0.01}$	$t = 0.21$	$\mathbf{t^* = 0.08}$	$t = 1.75$	$\mathbf{t^* = 0.84}$	$t = 17.42$
$n = 30$	$\mathbf{t^* = 0.07}$	$t = 0.91$	$\mathbf{t^* = 0.67}$	$t = 6.60$	$\mathbf{t^* = 6.50}$	$t = 115.41$
$n = 200$	$\mathbf{t^* = 6.19}$	$t = 42.81$	$\mathbf{t^* = 62.47}$	$t = X$	$\mathbf{t^* = 635}$	$t = X$

TABLE 1. Execution times of simulations for simulation steps $h = 10^{-k}, k = 3, 4, 5$ and dimensions $n = 10$, $n = 30$, $n = 200$.

Entries marked $'X'$ in Table 1 are execution times which the computer was unable to compute due to "out of memory" errors. The results obtained with our proposed algorithm are far superior to the ones obtained with IVP solvers. Both methods return similar distances between the desired final state and the final state obtained with the computed input function.

5. **Conclusions**

In this paper, we have derived numerical methods for the input retrieval and the computation of Lyapunov matrix differential equation solutions. The computation of the LMDE solution is based on an iterative process and we provided two distinct ways to compute the integral term (9), avoiding numerical integration. The first method relies on the matrix exponential of A_0 in (10), while the second uses the matrix exponential of matrix A and the Lyapunov equation (13). The second approach is based on the Schur-Parlett method for function of matrices (see [24]) and avoids the matrix exponential of A_0, whose spectrum is comprised of both the spectrum of A and of $(-A)$, and thus may generate an ill-conditioned matrix exponential F_0. The input given by equation (6) and computed by Algorithm 3.2 has the minimum energy required to steer the state trajectory from the initial null state to a desired final state, in a given time span.

The numerical results have shown that, in terms of execution time, the proposed algorithms outperform traditional IVP solvers.

The current work provides the setup for investigating how state feedback relates to certain input functions and state trajectories. More precisely, it would be interesting to see if a feedback parameter K, aside from placing the eigenvalues of $A - BK$, is able to enforce certain state trajectories, through the finite time controllability gramian of the closed loop system.

REFERENCES

[1] *M. Behr, P. Benner and J. Heiland*, Solution formulas for differential Sylvester and Lyapunov equations. Calcolo, **56** (2019) No. 4, 51.

[2] N. Lang, H. Mena and J. Saak, On the benefits of the LDLT factorization for large-scale differential matrix equation solvers. Linear Algebra and its Applications, **480** (2015), 44-71.

[3] *M. Hached, and K. Jbilou* (2018). Numerical solutions to large-scale differential Lyapunov matrix equations. Numerical Algorithms, **3** (2018) No. 79, 741-757.

[4] *R. E. Kalman*, Contributions to the theory of optimal control. Bol. soc. mat. mexicana, **2** (1960), No. 5, 102-119.

[5] *P. G. Howlett*, Input retrieval in finite dimensional linear systems. The ANZIAM Journal, **4** (1982), No. 23, 357-382.

[6] *M. B. Estrada, M. F. Garcia, M. Malabre, and J. C. M. García*, Left invertibility and duality for linear systems. Linear algebra and its applications, **2-3** (2007), No. 425, 345-373.

[7] *M. Hou, and R. J. Patton*, Input observability and input reconstruction. Automatica, **6** (1998), No. 34 789-794.

[8] *K. Zhou, J. C. Doyle, and K. Glover*, Robust and optimal control, Prentice hall New Jersey, 1996.

[9] *G. Baggio, V. Katewa and F. Pasqualetti*, Data-driven minimum-energy controls for linear systems. IEEE Control Systems Letters, **3** (2019), No. 3, 589-594.

[10] *P. M. Van Dooren*, Gramian based model reduction of large-scale dynamical systems. Chapman and Hall CRC Research Notes in Mathematics, (2000) 231-248.

[11] *S. Gugercin and A. C. Antoulas*, A survey of model reduction by balanced truncation and some new results. International Journal of Control, **8** (2004), No. 77, 748-766.

[12] *P. Kürschner*, Balanced truncation model order reduction in limited time intervals for large systems. Advances in Computational Mathematics, **6** (2018), No. 44, 1821-1844.

[13] *O. Napolitano, D. Fontanelli, L. Pallottino and P Salaris*, Gramian-based optimal active sensing control under intermittent measurements. In 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE (2021, May) 9680-9686.

[14] *Y. Zhao and J. Cortés*, Gramian-based reachability metrics for bilinear networks. IEEE Transactions on Control of Network Systems, **3** (2016), No. 4, 620-631.

[15] *S. Zhao and F. Pasqualetti*, Networks with diagonal controllability Gramian: Analysis, graphical conditions, and design algorithms. Automatica, **102** (2019), 10-18.

[16] *H. Sandberg and A. Rantzer*, Balanced truncation of linear time-varying systems. IEEE Transactions on automatic control, **49** (2004), No. 2, 217-229.

[17] *S. J. Hammarling*, Numerical solution of the stable, non-negative definite lyapunov equation lyapunov equation, IMA Journal of Numerical Analysis, Oxford University Press, **2**(1982), No. 3, 303-323.

[18] *V. B. Larin and F. A. Aliev*, Construction of square root factor for solution of the Lyapunov matrix equation. Systems & control letters, **2** (1993), No. 20, 109-112.

[19] *D. F. Griffiths and D. J. Higham.* Numerical methods for ordinary differential equations: initial value problems (Vol. 5). London: Springer, 2010.

[20] *G. H. Golub and C. F. Van Loan*, Matrix computations, JHU press, 2013.

[21] *R. L. Allen and D. Mills*, Signal analysis:time, frequency, scale, and structure, John Wiley & Sons, 2004.

[22] *R. W. Brockett* Finite Dimensional Linear Systems, SIAM, 2015.

[23] *C. Van Loan* Computing Integrals Involving the Matrix Exponential, IEEE transactions on automatic control, **23**(1978), No. 3, 395-404

[24] *N. J. Higham*, Functions of matrices: theory and computation, SIAM, 2008.

[25] *R. Gupta, W. Zhao and R. K. Kapania*, Controllability gramian as control design objective in aircraft structural design optimization. AIAA Journal, **7** (2020), No. 58, 3199-3220.

[26] *C. Moler and C. Van Loan*, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later, SIAM review, **45**(2003), No. 1, 3-49.

[27] *G. F. Franklin, D. J. Powell and M. L. Workman* Digital Control of Dynamic Systems (3rd Edition), Prentice Hall, 1997