

**Volume 86, Number 11,  
November 2025**

**ISSN 0005-1179  
CODEN: AURCAT**



# **AUTOMATION AND REMOTE CONTROL**

**Editor-in-Chief  
Andrey A. Galyaev**

<http://ait.mtas.ru>

Automation and Remote Control

Vol. 86, No. 11, November 2025

**Available via license: CC BY 4.0**

# Automation and Remote Control

ISSN 0005-1179

## Editor-in-Chief

Andrey A. Galyaev

**Deputy Editors-in-Chief** M.V. Khlebnikov and E.Ya. Rubinovich

**Coordinating Editor** A.S. Samokhin

## Editorial Board

F.T. Aleskerov, A.V. Arutyunov, N.N. Bakhtadze, A.A. Bobtsov, P.Yu. Chebotarev, A.G. Chkhartshvili, L.Yu. Filimonyuk, A.L. Fradkov, O.N. Granichin, M.F. Karavai, E.M. Khorov, M.M. Khrustalev, A.I. Kibzun, S.A. Krasnova, A.P. Krishchenko, A.G. Kushner, N.V. Kuznetsov, A.A. Lazarev, A.I. Lyakhov, A.I. Matasov, S.M. Meerkov (USA), R.V. Mescheryakov, A.I. Mikhal'skii, B.M. Miller, O.V. Morzhin, R.A. Munasypov, A.V. Nazin, A.S. Nemirovskii (USA), D.A. Novikov, A.Ya. Oleinikov, P.V. Pakshin, D.E. Pal'chunov, A.E. Polyakov (France), V.Yu. Protasov, L.B. Rapoport, I.V. Rodionov, N.I. Selvesyuk, P.S. Shcherbakov, A.N. Sobolevski, O.A. Stepanov, A.B. Tsybakov (France), D.V. Vinogradov, V.M. Vishnevskii, K.V. Vorontsov, and L.Yu. Zhilyakova

**Staff Editor** E.A. Martekhina

## SCOPE

*Automation and Remote Control* is one of the first journals on control theory. The scope of the journal is control theory problems and applications. The journal publishes reviews, original articles, and short communications (deterministic, stochastic, adaptive, and robust formulations) and its applications (computer control, components and instruments, process control, social and economy control, etc.).

*Automation and Remote Control* is abstracted and/or indexed in *ACM Digital Library*, *BFI List*, *CLOCKSS*, *CNKI*, *CNPIEC Current Contents/Engineering, Computing and Technology*, *DBLP*, *Dimensions*, *EBSCO Academic Search*, *EBSCO Advanced Placement Source*, *EBSCO Applied Science & Technology Source*, *EBSCO Computer Science Index*, *EBSCO Computers & Applied Sciences Complete*, *EBSCO Discovery Service*, *EBSCO Engineering Source*, *EBSCO STM Source*, *EI Compendex*, *Google Scholar*, *INSPEC*, *Japanese Science and Technology Agency (JST)*, *Journal Citation Reports/Science Edition*, *Mathematical Reviews*, *Naver*, *OCLC WorldCat Discovery Service*, *Portico*, *ProQuest Advanced Technologies & Aerospace Database*, *ProQuest-ExLibris Primo*, *ProQuest-ExLibris Summon*, *SCImago*, *SCOPUS*, *Science Citation Index*, *Science Citation Index Expanded (Sci-Search)*, *TD Net Discovery Service*, *UGC-CARE List (India)*, *WTI Frankfurt eG*, *zbMATH*.

Journal website: <http://ait.mtas.ru>

© The Author(s), 2025 published by Trapeznikov Institute of Control Sciences, Russian Academy of Sciences.

*Automation and Remote Control* participates in the Copyright Clearance Center (CCC) Transactional Reporting Service.

Available via license: CC BY 4.0

0005-1179/25. *Automation and Remote Control* (ISSN: 0005-1179 print version, ISSN: 1608-3032 electronic version) is published monthly by Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, 65 Profsoyuznaya street, Moscow 117997, Russia.

Volume 86 (12 issues) is published in 2025.

Publisher: Trapeznikov Institute of Control Sciences, Russian Academy of Sciences.

65 Profsoyuznaya street, Moscow 117997, Russia; e-mail: [redacsia@ipu.rssi.ru](mailto:redacsia@ipu.rssi.ru); <http://ait.mtas.ru>, <http://ait-arc.ru>

# Contents

---

---

## *Automation and Remote Control*

Vol. 86, No. 11, 2025

---

---

### Linear Systems

Numerical Method for Solving the Time-Optimization Problem for Linear Non-Stationary Discrete-Time Systems of General Form

*D. N. Ibragimov and K. A. Tsarkov* 989

PID Controller Design for Suppressing Bounded Exogenous Disturbances

*M. V. Khlebnikov* 1015

---

### Nonlinear Systems

Relay Controller Design in Self-Oscillating Control Systems Based on Training Examples

*V. A. Mozzhechkov* 1035

---

### Stochastic Systems

Model for Servicing Multiservice Traffic in an Access Node of a Satellite Communications Network with a Dynamically Variable Service Delivery Rate

*A. A. Maslov, G. V. Sebekin, M. S. Stepanov, S. N. Stepanov, and A. O. Shchurkov* 1046

Trajectory Countermeasures Against a Linear Observer

*A. Potapov and A. Galyaev* 1059

---

### Optimization, System Analysis, and Operations Research

A Distributed Graph Vertex Numbering Algorithm Combined with Breadth-First Search Tree Construction

*O. P. Kuznetsov* 1074

---

---



# Numerical Method for Solving the Time-Optimization Problem for Linear Non-Stationary Discrete-Time Systems of General Form

D. N. Ibragimov<sup>\*,a</sup> and K. A. Tsarkov<sup>\*\*,b</sup>

<sup>\*</sup>*Moscow Aviation Institute (National Research University), Moscow, Russia*

<sup>\*\*</sup>*Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia*

*e-mail: <sup>a</sup>rikk.dan@gmail.com, <sup>b</sup>k6472@mail.ru*

Received April 10, 2025

Revised June 6, 2025

Accepted July 21, 2025

**Abstract**—In the paper, we construct a software-implemented numerical procedure for solving the time-optimization problem for linear discrete-time systems with arbitrary variable matrices of the system and convex sets of geometric constraints on control. We also prove the convergence of the sequence of control processes produced by the algorithm to a solution to the problem. The efficiency is demonstrated on a number of examples.

*Keywords:* time-optimization problem, linear discrete-time systems, sequential global improvement, Krotov method

**DOI:** 10.7868/S1608303225110011

## 1. INTRODUCTION

The time-optimization problem is one of the classical problems of optimal control theory. In its most generality, it is posed as the problem of transferring a given dynamical object from a fixed initial state to some known set in phase space (target domain) in the least possible time. The corresponding dynamical system itself can operate in either continuous or discrete time, can be linear or nonlinear, and the target domain can be either a given smooth surface or a one-point set (singleton). In this paper, the time-optimization problem is understood as the problem of transferring a linear dynamical object operating in discrete time from a given initial state to the origin in the least possible number of steps. With the exception of some special cases, the main tool for studying problems of this kind has always been and remains the method of directly enumerating the moments of time of possible termination of the transient process and directly solving the resulting finite-dimensional problems of terminal stabilization. For these reasons, relatively few works are devoted directly to the time-optimization problem in the above sense. Among them, we highlight the fairly general considerations in [1–3], as well as their continuations [4–8].

In this paper, we develop a fundamentally different algorithm for solving the time-optimization problem, which is based on the method of constructing sequential global improvements of control processes proposed in the works of V.F. Krotov [9, 10]. This method is also applied to problems of terminal stabilization of a given linear discrete-time system at various fixed moments of time of the end of the transient process. However, unlike the classical approach, these problems are not supposed to be solved completely. Improvements are constructed until either an optimal process in terms of time-optimization is found, or the impossibility of reaching the origin in a given number of steps is proven. For these purposes, we construct an estimate of the optimal value of the quality functional based on the current approximation. A similar approach can be implemented using

other numerical methods of conditional finite-dimensional optimization and improvement, including those developed specifically for studying discrete optimal control problems. The latter include the methods from [11, Chapter 6] and some methods from [12, Part 2 and Part 4]. A distinctive feature of Krotov method is the speed of improvement, as well as the fact that, within the framework of the considered problems of terminal stabilization of linear discrete-time systems, it is possible to establish the presence of the key property of strict improvement of non-optimal processes. General results on non-strict improvement in optimal control problems for discrete and continuous systems were obtained by V.F. Krotov. Some particular results on strict improvement, close to those discussed here, but mainly in the case of continuous systems, are presented in [13].

The paper directly continues the research begun in [14], where we propose an approach to constructing two-sided optimal time estimates for stationary linear systems with a non-singular diagonalizable matrix of the system, and a numerical method for improving the upper bound and constructing a guaranteeing process. Here, the results obtained in [14] are further developed in the following directions. First, it will be shown that the iterative procedure from [14] not only allows one to improve the upper bounds for the optimal time estimate and construct guaranteeing processes, but is also actually capable of approximately finding optimal controls in the corresponding time-optimization problem without any additional assumptions. Second, a new procedure will be proposed that allows one to solve the time-optimization problem and construct optimal processes in the absence of any known optimal time estimates. Third, all results will be generalized to the case of non-stationary linear discrete-time systems of general form. In particular, all of them are applicable to stationary systems with a singular matrix.

## 2. PROBLEM STATEMENT

Consider a linear non-stationary system with discrete time

$$x(k+1) = A(k)x(k) + u(k), \quad k = 0, 1, \dots, \quad (1)$$

where  $x(k) \in \mathbb{R}^n$  is the state of the system,  $u(k) \in U(k)$  is the control,  $U(k)$  are convex compact sets in  $\mathbb{R}^n$  containing the origin,  $A(k) \in \mathbb{R}^{n \times n}$  are arbitrary given matrices. The initial condition for the system (1) is fixed:

$$x(0) = x_0 \in \mathbb{R}^n. \quad (2)$$

It is required to calculate the minimum number of steps  $N_{\min}$ , in which it is possible to transfer the system (1) from a given initial state  $x_0$  to the origin and to construct an optimal process  $\{x^*(k), u^*(k-1)\}_{k=1}^{N_{\min}}$ , satisfying the condition  $x^*(N_{\min}) = 0$ . The number  $N_{\min}$  will be called the optimal time of the system (1) with the initial condition (2) and we will assume that the problem is solvable, i.e.  $N_{\min} < \infty$ .

In the stationary case  $A(k) \equiv A \in \mathbb{R}^{n \times n}$  and  $U(k) \equiv U \subset \mathbb{R}^n$ , the well-known sufficient conditions for the solvability of the time-optimization problem for the system (1)–(2) are the Schur stability of the matrix  $A$  and the inclusion  $0 \in \text{int}U$  or the Kalman controllability of the system (1) and the inclusion  $0 \in \text{ri}U$ .

## 3. FIXED-TIME PROBLEM AND OPTIMALITY CONDITIONS

In [14], in the case of stationary systems with a non-singular matrix  $A(k) \equiv A$ , we propose an algorithm that allows one to construct guaranteeing processes, using known estimates for the optimal time, and improve the upper estimate. This includes solving several problems with a fixed operation time of the system sequentially. Let us discuss these problems.

First, introduce the following notation. Let  $N > 0$  be some fixed value of discrete time and  $k = 0, \dots, N - 1$  in the system (1). Let

$$\begin{aligned} \mathcal{U}_N &:= \{k \mapsto u(k) : \{0, \dots, N - 1\} \rightarrow \mathbb{R}^n \mid u(k) \in U(k)\}, \\ \mathcal{X}_N &:= \{k \mapsto x(k) : \{0, \dots, N\} \rightarrow \mathbb{R}^n\}. \end{aligned}$$

We will identify the set  $\mathcal{X}_N$  with the Euclidean space  $\mathbb{R}^{n(N+1)}$ , and the elements of the set  $\mathcal{U}_N$  with the corresponding vectors of the Euclidean space  $\mathbb{R}^{nN}$ .

Consider for the system (1)–(2) the problem

$$J_N(u) = \|x(N)\|^2 \rightarrow \min_{u \in \mathcal{U}_N}, \tag{3}$$

where  $x \in \mathcal{X}_N$  is the solution to (1)–(2) for  $k = 0, \dots, N - 1$  and fixed  $u \in \mathcal{U}_N$ . Here and below,  $\|\cdot\|$  denotes the Euclidean norm in  $\mathbb{R}^n$ .

The problem (3) for the system (1)–(2) satisfies the conditions [11, p. 124, conditions 1–4], which guarantee [11, Theorem 5.6.2] that for the optimality of control  $\hat{u} \in \mathcal{U}_N$  in this problem it is necessary and sufficient to require the fulfillment of the relations of the classical discrete maximum principle [11, Theorems 5.3.1 and 5.6.1]. Since the system (1) is linear, these relations reduce to testing  $N$  maximum conditions

$$\langle \hat{\psi}(k + 1), \hat{u}(k) \rangle = \max_{u \in U(k)} \langle \hat{\psi}(k + 1), u \rangle, \tag{4}$$

for  $k = 0, \dots, N - 1$ , where  $\langle \cdot, \cdot \rangle$  denotes the inner product in  $\mathbb{R}^n$ , and the values of the dual variable  $\hat{\psi}(k)$  are uniquely determined by

$$\hat{x}(k + 1) = A(k)\hat{x}(k) + \hat{u}(k), \quad \hat{x}(0) = x_0, \tag{5}$$

$$\hat{\psi}(k) = A(k)^T \hat{\psi}(k + 1), \quad \hat{\psi}(N) = -2\hat{x}(N). \tag{6}$$

It is easy to see that for  $N \geq N_{\min}$  the optimal controls in problem (3) are singular [15], since in this case  $\hat{\psi}(k) \equiv 0$ . For what follows, we will need an alternative form of the optimality conditions in this problem, with respect to which the optimal controls will no longer be singular for any value of  $N$ .

**Theorem 1.** *Let  $\hat{u} \in \mathcal{U}_N$ . Then for the optimality of  $\hat{u}$  in the problem (1)–(3) it is necessary and sufficient that for  $k = 0, \dots, N - 1$  the inclusions*

$$\hat{u}(k) \in \text{Arg} \min_{u \in U(k)} \|\mathcal{A}_N(k + 1)u + \mathcal{A}_N(k)\hat{x}(k) - \hat{\xi}(k + 1)\| \tag{7}$$

hold, where

$$\mathcal{A}_N(k) := A(N - 1) \dots A(k), \quad k = 0, \dots, N - 1, \quad \mathcal{A}_N(N) := I, \tag{8}$$

$I$  is the identity matrix of size  $n \times n$ , and the values  $\hat{x}(k)$  and  $\hat{\xi}(k)$  are defined by the equalities (5) and

$$\hat{\xi}(k) = \hat{\xi}(k + 1) - \mathcal{A}_N(k + 1)\hat{u}(k), \quad \hat{\xi}(N) = 0. \tag{9}$$

For the convenience of the reader, the proofs of the assertions presented in the paper are included in Appendix A.

Necessary conditions in Theorem 1 up to changes of variables and renaming were obtained in [14] for the case  $A(k) \equiv A, \det A \neq 0, U(k) \equiv U$ . Sufficient conditions are obtained here for the first time.

The set of conditions (5), (7) and (9), unlike (4)–(6), does not degenerate in the case  $N \geq N_{\min}$ . Thus, for example, for  $k = N - 1$  from (7) we have

$$\hat{u}(N - 1) \in \text{Arg} \min_{u \in U(N-1)} \|A(N - 1)\hat{x}(N - 1) + u\|,$$

and this condition is always meaningful and reflects the geometric meaning of optimal control in the problem (1)–(3).

As a direct consequence of Theorem 1, we can obtain the statement that the algorithm proposed in [14] allows one to approximately find optimal processes in the time-optimization problem for stationary non-degenerate linear systems. This strengthens the theoretical results presented in the indicated work. The algorithm for solving the time-optimization problem proposed below also relies heavily on the assertion of Theorem 1.

#### 4. IMPROVEMENT THEOREMS

Let  $\hat{u} \in \mathcal{U}_N$  be some arbitrary control. Let us set the goal of constructing a new control  $\tilde{u} \in \mathcal{U}_N$  that improves  $\hat{u}$  in the sense of the value of the functional  $J_N$  in the problem (3).

**Theorem 2.** *Let  $\hat{u} \in \mathcal{U}_N$  be an arbitrary control, and  $\hat{x}, \hat{\xi} \in \mathcal{X}_N$  be the corresponding solutions to the equations (5) and (9). Let us define the control  $\tilde{u} \in \mathcal{U}_N$  by the condition*

$$\tilde{u}(k) \in \text{Arg} \min_{u \in U(k)} \|\mathcal{A}_N(k + 1)u + \mathcal{A}_N(k)\tilde{x}(k) - \hat{\xi}(k + 1)\| \quad \forall k \in \{0, \dots, N - 1\}, \quad (10)$$

where  $\mathcal{A}_N(k)$  is determined from (8), and  $\tilde{x}(k)$  satisfies the equalities

$$\tilde{x}(k + 1) = A(k)\tilde{x}(k) + \tilde{u}(k), \quad \tilde{x}(0) = x_0. \quad (11)$$

Then, with respect to controls  $\hat{u}, \tilde{u} \in \mathcal{U}_N$ , there is a non-strict improvement in (3), i.e.

$$J_N(\tilde{u}) \leq J_N(\hat{u}).$$

Theorem 2 was proved up to notation in [14] under the assumptions  $A(k) \equiv A, \det A \neq 0, U(k) \equiv U$ , which are inessential. The proof given in Appendix 1 is entirely based on V.F. Krotov’s global improvement constructions.

**Theorem 3.** *Let  $\hat{u} \in \mathcal{U}_N, \hat{x} \in \mathcal{X}_N$  satisfy the equation (5) and for the pair  $(\tilde{x}, \tilde{u})$  the conditions (10), (11) hold. Then the equality  $J_N(\tilde{u}) = J_N(\hat{u})$  holds if and only if the control  $\hat{u}$  is optimal in (1)–(3).*

Theorem 3 states that any non-optimal control in (1)–(3) can be strictly improved with respect to the values of the functional  $J_N$  by constructing a new control according to the relations (10), (11). Since all sets  $U(k)$  are assumed to be compact, these relations are always solvable (possibly not uniquely).

#### 5. ESTIMATION OF THE OPTIMAL VALUE OF THE QUALITY FUNCTIONAL

When studying the problem (3) in the context of the original time-optimization problem for the system (1)–(2), it is important to be able to estimate from below the optimal value of the functional  $J_N$  with sufficient accuracy. For this purpose, one can use the information obtained in the process of constructing improvements. Here is one such estimate.

**Theorem 4.** Let  $\hat{u} \in \mathcal{U}_N$ , the values  $\hat{x}(k)$  and  $\hat{\psi}(k)$  be determined from (5) and (6), and  $J_N^*$  be the optimal value of the functional  $J_N$  in (3). Then

$$J_N^* \geq \|\hat{x}(N)\|^2 - \sum_{k=0}^{N-1} \max_{u \in U(k)} \langle \hat{\psi}(k+1), u - \hat{u}(k) \rangle. \tag{12}$$

From Theorem 4 we deduce the following: if for some  $N > 0$  and some control  $\hat{u} \in \mathcal{U}_N$  the right-hand side of inequality (12) is strictly positive, then  $N_{\min} > N$  is guaranteed to hold, since  $J_N^* = 0$  for all  $N \geq N_{\min}$  due to the conditions  $0 \in U(k)$ . In particular, substituting the control  $\hat{u} = 0 \in \mathcal{U}_N$  into (12), we obtain the following estimate of the optimal time:

$$N_{\min} \geq \min \left\{ N \geq 0 \mid \|\mathcal{A}_N(0)x_0\|^2 + 2 \sum_{k=0}^{N-1} \min_{u \in U(k)} \langle \mathcal{A}_N(k+1)^T \mathcal{A}_N(0)x_0, u \rangle \leq 0 \right\}.$$

We should note that a similar to (12) inequality is obtained in [11, p. 166].

### 6. ITERATIVE ALGORITHM FOR SOLVING THE TIME-OPTIMIZATION PROBLEM

Now we proceed to constructing a general algorithm for solving the initial time-optimization problem for the system (1)–(2). To do this, we will sequentially improve the control processes in the problem (3) for values  $N = 1, 2, \dots$ , applying the results of section 4, and use the inequality (12) for a quick estimate of the theoretical possibility of reaching the origin for the current value of  $N$ . Excluding the trivial case, we will assume that  $N_{\min} > 0$ .

We have the following algorithm:

0. Set the value of the permissible calculation error  $\varepsilon > 0$ , put  $N = 1$ .
1. Set/supplement the initial approximation with the equality  $u^{(0)}(N - 1) = 0$ , put  $l = 0$  and calculate the matrices  $\mathcal{A}_N(k) = A(N - 1) \dots A(k)$ ,  $k = 0, \dots, N - 1$ , set the matrix  $\mathcal{A}_N(N)$  to be unitary, of size  $n \times n$ .
2. Find the solution  $\xi^{(l)}$  to the system of equations

$$\xi(k) = \xi(k + 1) - \mathcal{A}_N(k + 1)u^{(l)}(k), \quad \xi(N) = 0.$$

3. For each  $k \in \{0, \dots, N - 1\}$ , sequentially find and fix some solution  $u^{(l+1)}(k)$  to the extremal problem

$$\|\mathcal{A}_N(k + 1)u + \mathcal{A}_N(k)x^{(l+1)}(k) - \xi^{(l)}(k + 1)\| \rightarrow \min_{u \in U(k)},$$

where the values of  $x^{(l+1)}(k)$  are calculated using the formulas

$$x^{(l+1)}(k + 1) = A(k)x^{(l+1)}(k) + u^{(l+1)}(k), \quad k = 0, \dots, N - 1, \quad x^{(l+1)}(0) = x_0.$$

4. Check the external stopping condition

$$\|x^{(l+1)}(N)\| < \varepsilon,$$

when executed, finish the calculations with the answer  $N_\varepsilon = N$ ,  $u_\varepsilon = u^{(l+1)}$ .

5. Find the solution  $\psi^{(l+1)}$  to the system of equations

$$\psi(k) = A(k)^T \psi(k + 1), \quad k = 0, \dots, N - 1, \quad \psi(N) = -2x^{(l+1)}(N).$$

6. Calculate the estimate  $E_N^{(l+1)}$  of the possible approximation to origin by the formula

$$E_N^{(l+1)} = \|x^{(l+1)}(N)\|^2 - \sum_{k=0}^{N-1} \max_{u \in U(k)} \langle \psi^{(l+1)}(k + 1), u - u^{(l+1)}(k) \rangle.$$

7. Check the internal stopping condition

$$E_N^{(l+1)} > 0,$$

when executing, fix the found  $u^{(l+1)}$  as a new initial approximation, i.e. set  $u^{(0)}(k) = u^{(l+1)}(k)$ ,  $k = 0, \dots, N - 1$ , increase  $N$  by one and go to step 1, otherwise increase  $l$  by one and go to step 2.

Denote by  $\mathcal{U}^*$  the set of all time-optimal controls for the system (1)–(2). By virtue of the assumptions made, the inclusions  $\mathcal{U}^* \subset \mathcal{U}_{N_{\min}} \subset \mathbb{R}^{nN_{\min}}$  hold. Recall that the distance from a point to a set in the space  $\mathbb{R}^{nN_{\min}}$  is defined as

$$\text{dist}(z, Z) := \inf_{\zeta \in Z} \|z - \zeta\|, \quad z \in \mathbb{R}^{nN_{\min}}, \quad Z \subset \mathbb{R}^{nN_{\min}}.$$

**Theorem 5.** *Let  $N_{\min} < \infty$ . Then, for any value of  $\varepsilon > 0$ , the algorithm constructed above terminates its work after a finite number of iterations with an answer  $N_\varepsilon, u_\varepsilon$ . In this case,  $N_\varepsilon \leq N_{\min}$ , and if the number  $\varepsilon > 0$  is small enough, then  $N_\varepsilon = N_{\min}$ . Moreover, for such values of  $\varepsilon, u_\varepsilon \in \mathbb{R}^{nN_{\min}}$  holds and the convergence*

$$\text{dist}(u_\varepsilon, \mathcal{U}^*) \rightarrow 0, \quad \varepsilon \rightarrow 0$$

takes place.

Thus, the constructed algorithm allows us to approximately find solutions to the time-optimization problem for the system (1)–(2).

### 7. DISCUSSION AND COMMENTS

To implement the proposed algorithm, it is necessary to determine a method for solving two types of finite-dimensional convex optimization problems at steps 3 and 6.

The problems at step 3 are essentially problems of metric projection of given vectors onto given convex compact sets in  $\mathbb{R}^n$ . The latter are the sets  $\mathcal{A}_N(k + 1)U(k)$ , where all  $U(k)$  are known in advance, and the linear transformations  $\mathcal{A}_N(k + 1)$  are calculated from the known matrices of the system  $A(k)$  and depend on the current values of  $k$  and  $N$ . Thus, for any values of  $k, N$ , the set  $\mathcal{A}_N(k + 1)U(k)$  can be considered known in advance. In applications,  $U(k)$  are often linear transformations of some base set  $U$  (e.g.,  $U$  is the unit ball and  $U(k)$  is an ellipsoid;  $U$  is the unit cube and  $U(k)$  is a zonotope, etc.), and in this case  $\mathcal{A}_N(k + 1)U(k)$  have the same structure. For many such sets, the metric projection problem has been extensively studied, and various high-speed algorithms have been developed for its solution [16, 17]. It is also important to note that the computational errors associated with one or another method of approximate solution to the problems at step 3 do not accumulate in subsequent iterations, since each time a nonlocal improvement of the previously obtained (inaccurate) control program is performed anew. Note also that the optimization problems at step 3 can be rewritten in terms of trajectories. Namely, instead of performing steps 2 and 3 for each  $k \in \{0, \dots, N - 1\}$ , the following optimization problems can be solved sequentially:

$$x^{(l+1)}(k + 1) \in \text{Arg} \min_{x \in A(k)x^{(l+1)}(k) + U(k)} \|\mathcal{A}_N(k + 1)(x - x^{(l)}(k + 1)) + x^{(l)}(N)\|,$$

where for  $k = 0$  we have

$$x^{(l+1)}(0) = x_0.$$

In this case, the number of intermediate calculations required to construct the improved trajectory is somewhat reduced. In this case, the corresponding control values are determined by the equalities

$$u^{(l+1)}(k) = x^{(l+1)}(k + 1) - A(k)x^{(l+1)}(k), \quad k = 0, \dots, N - 1.$$

More important is the exact solving of the maximization problems at step 6. Calculating the estimate value  $E_N^{(l+1)}$  with the minimal possible error is important for correctly checking the condition  $E_N^{(l+1)} > 0$  at step 7, which guarantees that the current value of time  $N$  is strictly less than the optimal time  $N_{\min}$ . The presence of computational errors for  $E_N^{(l+1)}$  can lead to the issue that the optimal time is found incorrectly even in the case when the value  $\varepsilon > 0$  is sufficiently small. In this regard, it is helpful to use support functions of the sets  $U(k)$ , which can be determined in advance, since the structure of all  $U(k)$  is assumed to be known. With this approach, instead of solving problems of maximizing a linear function on a convex compact set, it will be sufficient to calculate the values of known support functions at given points. The latter allows increasing the accuracy of calculations and additionally reducing the overall running time.

The algorithm proposed in Section 6 can be implemented in any software environment in an arbitrary programming language. To demonstrate the results, an implementation was made in the freely distributed high-level language Python3. For the numerical solving of extremal problems at steps 3 and 6, we use a separate convex optimization library *cvxpy* and the methods *cvxpy.SCS* and *cvxpy.CLARABEL*. Note that in the case where the set  $U$  is a zonotope, it is advisable to use the *cvxpy.CLARABEL* method both in step 6 and in step 3. Detailed documentation and source codes of these methods can be found on the official website of the library <https://www.cvxpy.org/>.

The program listing is included in Appendix B. To feed the initial data from the following examples to the program input, one can use the json code provided in Appendix C.

### 8. EXAMPLES

*Example 1.* Consider a one-dimensional stationary system

$$x(k + 1) = x(k) + u(k), \quad k = 0, 1, \dots, \quad x(0) = x_0 \in (2; 3]$$

with a control constraint  $|u(k)| \leq 1$  and solve the time-optimization problem for this system using the algorithm from Section 6.

Since  $n = 1$ , all extremal problems at step 3 have unique solution, and  $\mathcal{A}_N(k) = 1$  for all  $N$  and  $k$ . We assume that  $\varepsilon > 0$  is so small that the complete stop condition at step 4 can be replaced by the condition  $|x^{(l+1)}(N)| = 0$ .

Let  $N = 1$  and  $u^{(0)}(0) = 0$ . At step 2 we have

$$\xi^{(0)}(0) = \xi^{(0)}(1) = 0.$$

At step 3 we find

$$u^{(1)}(0) = \arg \min_{u \in [-1; 1]} |u + x_0| = -1, \quad x^{(1)}(1) = x_0 - 1 \in (1; 2].$$

Checking at step 4 shows that  $|x^{(1)}(1)| > 0$ , so at step 5 we have

$$\psi^{(1)}(0) = \psi^{(1)}(1) = -2x^{(1)}(1) = 2 - 2x_0 < 0.$$

From here, in step 6 we calculate

$$E_1^{(1)} = |x^{(1)}(1)|^2 - \max_{u \in [-1; 1]} \psi^{(1)}(1)(u - u^{(1)}(0)) = |x^{(1)}(1)|^2 = (x_0 - 1)^2 > 0.$$

In step 7 we fix the found  $u^{(1)}(0) = -1$  and move on to the case  $N = 2$ .

For  $N = 2$  we have  $u^{(0)}(0) = -1, u^{(0)}(1) = 0$ . Then we sequentially obtain

$$\begin{aligned} \xi^{(0)}(1) &= \xi^{(0)}(2) = 0, \quad \xi^{(0)}(0) = 1; \\ u^{(1)}(0) &= \arg \min_{u \in [-1;1]} |u + x_0| = -1, \quad x^{(1)}(1) = x_0 - 1 \in (1; 2]; \\ u^{(1)}(1) &= \arg \min_{u \in [-1;1]} |u + x^{(1)}(1)| = -1, \quad x^{(1)}(2) = x_0 - 2 \in (0; 1]. \end{aligned}$$

Since  $|x^{(1)}(2)| > 0$ , we find

$$\begin{aligned} \psi^{(1)}(0) &= \psi^{(1)}(1) = \psi^{(1)}(2) = -2x^{(1)}(2) = 4 - 2x_0 < 0; \\ E_2^{(1)} &= |x^{(1)}(2)|^2 - \max_{u \in [-1;1]} \psi^{(1)}(1)(u - u^{(1)}(0)) - \max_{u \in [-1;1]} \psi^{(1)}(2)(u - u^{(1)}(1)) \\ &= |x^{(1)}(2)|^2 = (x_0 - 2)^2 > 0. \end{aligned}$$

We fix the found  $u^{(1)}(0) = u^{(1)}(1) = -1$  and finally move on to the case  $N = 3$ .

For  $N = 3$  we have  $u^{(0)}(0) = u^{(0)}(1) = -1, u^{(0)}(2) = 0$ . Here we get

$$\begin{aligned} \xi^{(0)}(2) &= \xi^{(0)}(3) = 0, \quad \xi^{(0)}(0) = \xi^{(0)}(1) = 1; \\ u^{(1)}(0) &= \arg \min_{u \in [-1;1]} |u + x_0 - 1| = -1, \quad x^{(1)}(1) = x_0 - 1 \in (1; 2]; \\ u^{(1)}(1) &= \arg \min_{u \in [-1;1]} |u + x^{(1)}(1)| = -1, \quad x^{(1)}(2) = x_0 - 2 \in (0; 1]. \\ u^{(1)}(2) &= \arg \min_{u \in [-1;1]} |u + x^{(1)}(2)| = -x^{(1)}(2) = 2 - x_0, \quad x^{(1)}(3) = 0. \end{aligned}$$

The check at step 4 shows that the algorithm is finished and a solution to the time-optimization problem has been found:  $N_{\min} = 3, u^*(0) = u^*(1) = -1, u^*(2) = 2 - x_0, x^*(1) = x_0 - 1, x^*(2) = x_0 - 2, x^*(3) = 0$ .

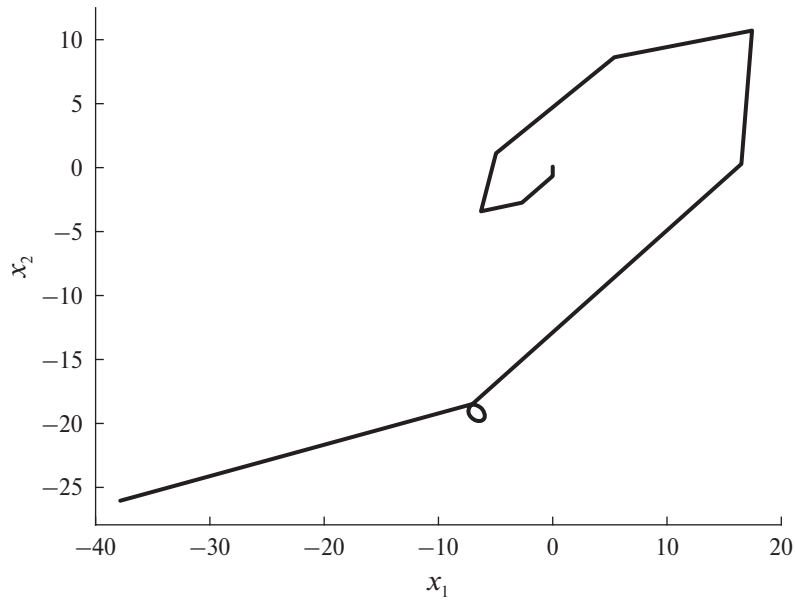
*Example 2.* Let in (1)–(2)  $n = 2$ , the system is stationary and it is known that

$$\begin{aligned} A(k) &\equiv \begin{pmatrix} \frac{4}{5}(\cos(1) + \sin(1)) & -\frac{8}{5}\sin(1) \\ \frac{4}{5}\sin(1) & \frac{4}{5}(\cos(1) - \sin(1)) \end{pmatrix}; \\ U(k) &\equiv \{v \in \mathbb{R}^2 \mid (2v_1 + v_2)v_1 + (v_1 + 3v_2)v_2 \leq 1\}; \\ x_0 &= (-37.79, -26.1); \quad \varepsilon = 0.0001. \end{aligned}$$

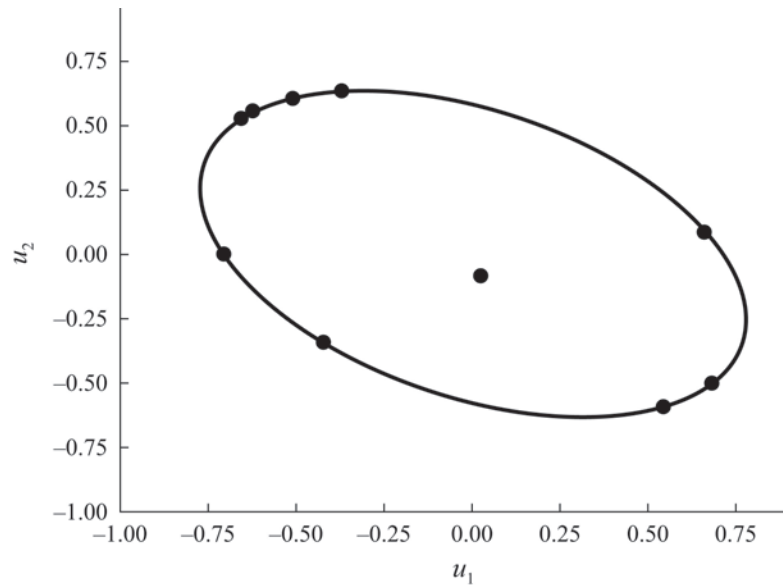
Applying the proposed algorithm, we obtain the following results:  $N_{\min} = 10$ , the time-optimal process has the form shown in Figs. 1 and 2. Table 1 provides detailed information on the conver-

**Table 1.** Convergence of the algorithm in Example 2

$N$	$l + 1$	$\ x(N)\ $	$\sqrt{E}$
1	1	19.72	19.72
2	1	16.24	16.24
3	1	20.4	20.4
4	1	8.91	8.9
5	1	4.58	4.54
6	1	7.07	7.07
7	1	2.52	2.51
8	3	0.55	0.25
9	1	0.22	0.19
10	1	$10^{-5}$	0



**Fig. 1.** Optimal trajectory in Example 2.



**Fig. 2.** Optimal control in Example 2.

gence of the algorithm. In this and subsequent tables,  $N$  and  $l$  correspond to the notations in the paper,  $\|x(N)\|$  should be read as  $\|x^{(l+1)}(N)\|$ , and  $\sqrt{E}$  as  $\sqrt{\max\{E_N^{(l+1)}, 0\}}$ .

*Example 3.* Let in the system (1)–(2)  $n = 2$  and it is known that

$$A(k) \equiv \begin{pmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \cos\left(\frac{\pi}{4}\right) \end{pmatrix};$$

$$x_0 = (9.33, 0.2); \quad \varepsilon = 0.0001.$$

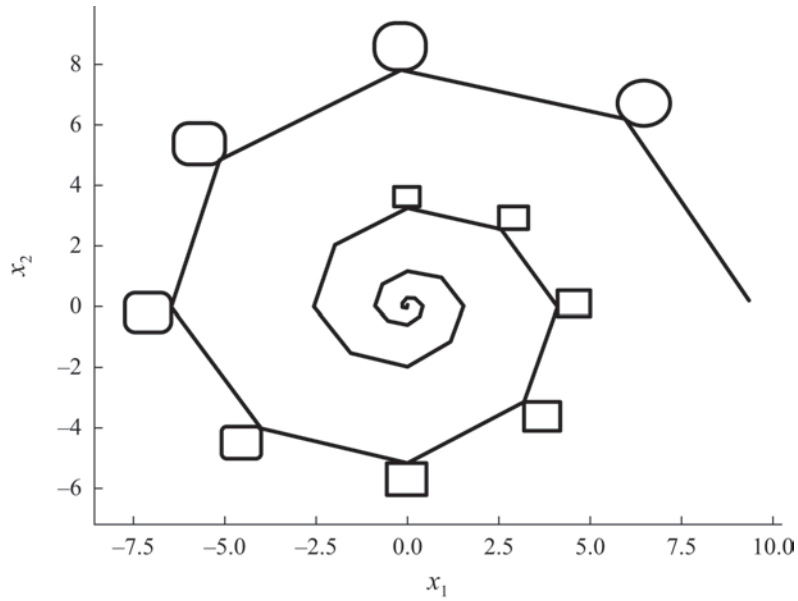


Fig. 3. Optimal trajectory in Example 3.

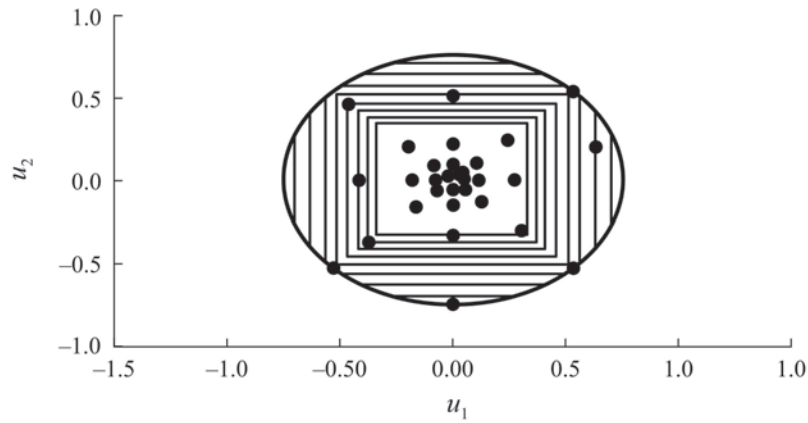


Fig. 4. Optimal control in Example 3.

Let us assume that the sets  $U(k)$  change according to the following rule:

$$U(k) = \left\{ v \in \mathbb{R}^2 \mid \max\{|v_1|, |v_2|\} \leq (0.9)^k \frac{\sqrt{3}}{2}, \|v\| \leq \frac{3}{4} \right\}.$$

Applying the proposed algorithm, we obtain the following results, presented in Table 2 and Figs. 3 and 4.

Table 2. Convergence of the algorithm in Example 3

$N$	$l + 1$	$\ x(N)\ $	$\sqrt{E}$
1	1	8.58	8.58
2	1	7.83	7.83
3	1	7.08	7.08
4	1	6.45	6.45
5	1	5.7	5.7
...	...	...	...
28	1	0.13	0.13
29	1	0.07	0.07
30	1	0.03	0.03
31	1	$10^{-6}$	0

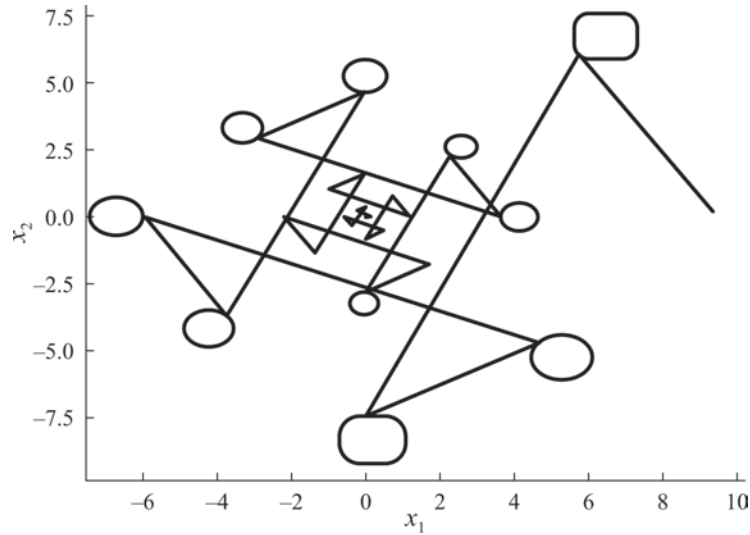


Fig. 5. Optimal trajectory in Example 4.

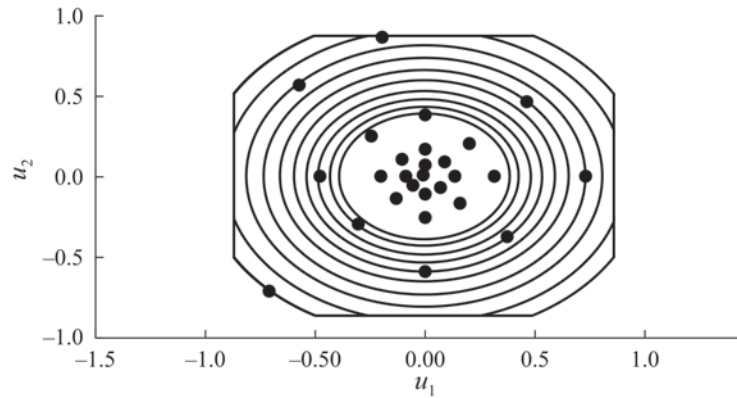


Fig. 6. Optimal control in Example 4.

Example 4. In the conditions of the previous example, suppose that both matrices  $A(k)$  and sets  $U(k)$  change according to the following rules:

$$A(k) = (-1)^k \begin{pmatrix} \cos(\frac{\pi}{4}) & -\sin(\frac{\pi}{4}) \\ \sin(\frac{\pi}{4}) & \cos(\frac{\pi}{4}) \end{pmatrix};$$

$$U(k) = \left\{ v \in \mathbb{R}^2 \mid \max\{|v_1|, |v_2|\} \leq \frac{\sqrt{3}}{2}, \|v\| \leq (0.9)^k \right\}.$$

We take the values of  $x_0$  and  $\varepsilon$  from Example 3. Applying the proposed algorithm, we obtain the following results, presented in Table 3 and Figs. 5 and 6.

Table 3. Convergence of the algorithm in Example 4

$N$	$l + 1$	$\ x(N)\ $	$\sqrt{E}$
1	1	8.33	8.33
2	1	7.46	7.46
3	1	6.65	6.65
4	1	5.92	5.92
5	1	5.27	5.27
...	...	...	...
24	1	0.16	0.16
25	1	0.08	0.08
26	1	0.1	0.1
27	1	$10^{-5}$	0

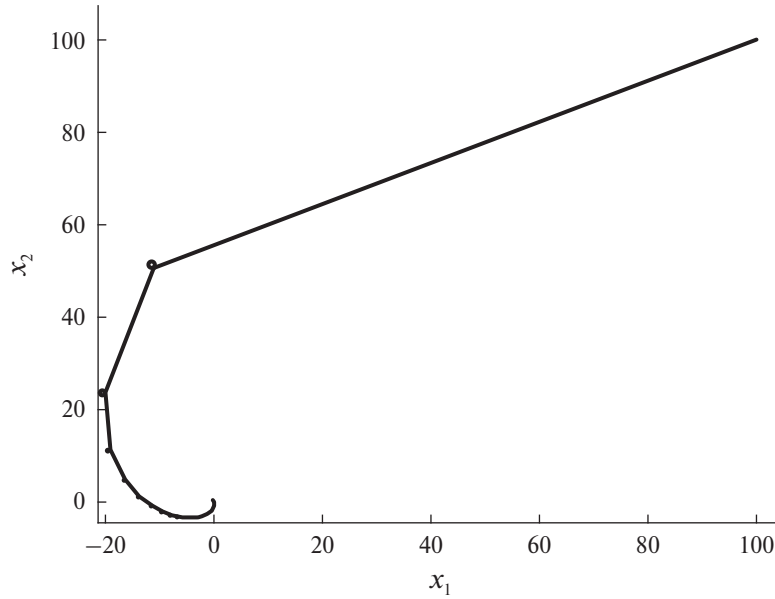


Fig. 7. Optimal trajectory in Example 5.

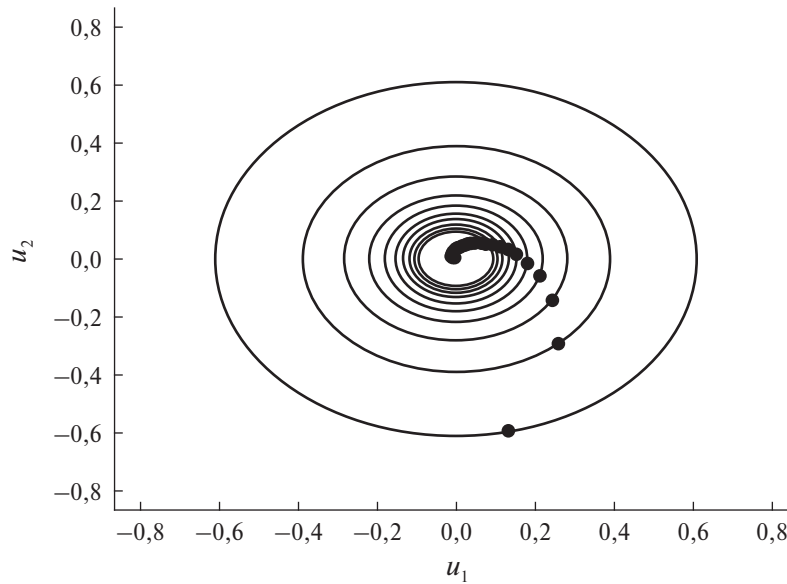


Fig. 8. Optimal control in Example 5.

Example 5. Consider the following example of a non-stationary system (1)–(2). Let  $n = 2$  and it is known that

$$A(k) = \begin{pmatrix} e^{-1/(k+1)} \cos(1/(k+1)) & -e^{-1/(k+1)} \sin(1/(k+1)) \\ e^{-1/(k+1)} \sin(1/(k+1)) & e^{-1/(k+1)} \cos(1/(k+1)) \end{pmatrix};$$

$$U(k) = \left\{ v \in \mathbb{R}^2 \mid 2v_1^2 + 2v_2^2 \leq 1 + e^{-2/(k+1)} - 2e^{-1/(k+1)} \cos(1/(k+1)) \right\};$$

$$x_0 = (100, 100); \quad \varepsilon = 0.0001.$$

Here the matrices  $A(k)$  and the sets  $U(k)$  correspond to a system of the form (1)–(2) obtained by discretizing a continuous-time system of the form

$$\dot{z}(t) = A_c z(t) + w(t), \quad z(0) = x_0,$$

where  $t \in [0; +\infty)$ ,

$$A_c = \begin{pmatrix} -1 & -1 \\ 1 & -1 \end{pmatrix},$$

the control function  $w(t)$  satisfies the geometric constraint  $\|w(t)\| \leq 1, t \geq 0$ , and is piecewise constant on time intervals  $[t_k; t_{k+1}), k = 0, 1, \dots$ , where  $t_0 = 0$  and the discretization step  $\Delta_k := t_{k+1} - t_k$  is not constant, but changes according to the rule  $\Delta_k = 1/(k + 1)$ .

Applying the proposed algorithm, we obtain the following results, presented in Table 4 and Figs. 7 and 8.

**Table 4.** Convergence of the algorithm in Example 5

$N$	$l + 1$	$\ x(N)\ $	$\sqrt{E}$
1	1	51.42	51.42
2	1	30.78	30.78
3	1	21.79	21.79
4	1	16.75	16.75
5	1	13.53	13.53
...	...	...	...
77	1	0.03	0.03
78	1	0.02	0.02
79	1	0.01	0.01
80	1	$10^{-5}$	0

### 9. CONCLUSION

The algorithm developed in this paper is a method for sequential global improvement of the control program in the time-optimization problem for a general linear discrete-time system. In the absence of additional assumptions about the problem and the properties of its solution, we establish the convergence to an optimal process. It should be emphasized that despite the presence of some enumeration procedures in the structure of the algorithm, the information obtained at earlier stages is not lost when moving to the next steps, but is used to construct a new approximation.

Examples show that there are problems in which this approach is more effective in comparison with the classical enumeration-optimization approach. However, at the moment it has not been proven that the algorithm will work as effectively in the general case. Moreover, computational practice shows that the convergence rate directly depends on both the method of software implementation and the initial data of the problem. At the same time, various additional information that may arise when solving specific applied problems (for example, any estimates of the optimal time and/or a known approximation to the optimal process) are naturally built into the structure of the algorithm and can be used to increase the rate of convergence to the solution. Theoretical study of the convergence properties of the developed iterative procedure seems to be a relevant direction for further research.

We also note that the results of this paper can be easily generalized to a more meaningful practical case of non-stationary linear systems of the form  $x(k + 1) = A(k)x(k) + B(k)u(k)$ , where the vector  $u(k)$  has dimension  $m$ , and, generally speaking,  $m \neq n$ , and the matrices  $B(k) \in \mathbb{R}^{n \times m}$  are considered to be known.

**Lemma 1.** Let  $\hat{u} \in \mathcal{U}_N$  be an arbitrary control and  $\hat{x} \in \mathcal{X}_N$  be found from (5). Let also  $\hat{\psi}, \hat{\xi} \in \mathcal{X}_N$  be defined by (6) and (9), respectively. Then for each  $k \in \{0, \dots, N\}$  the equality

$$\hat{\psi}(k) = 2\mathcal{A}_N(k)^T \left( \hat{\xi}(k) - \mathcal{A}_N(k)\hat{x}(k) \right) \quad (\text{A.1})$$

holds.

**Proof.** Indeed, for  $k = N$  the equality (A.1) follows from the initial conditions on the right-hand sides of the formulas (6) and (9), taking into account  $\mathcal{A}_N(N) = I$ . Suppose that (A.1) holds for some  $k \in \{1, \dots, N\}$ . Then

$$\begin{aligned} \hat{\psi}(k-1) &\stackrel{(6)}{=} A(k-1)^T \hat{\psi}(k) \stackrel{(\text{A.1})}{=} 2\mathcal{A}_N(k-1)^T \left( \hat{\xi}(k) - \mathcal{A}_N(k)\hat{x}(k) \right) \stackrel{(5)}{=} \\ &2\mathcal{A}_N(k-1)^T \left( \hat{\xi}(k) - \mathcal{A}_N(k-1)\hat{x}(k-1) - \mathcal{A}_N(k)\hat{u}(k-1) \right) \stackrel{(9)}{=} \\ &2\mathcal{A}_N(k-1)^T \left( \hat{\xi}(k-1) - \mathcal{A}_N(k-1)\hat{x}(k-1) \right). \end{aligned}$$

Consequently, (A.1) holds for all  $k \in \{0, \dots, N\}$ . Lemma 1 is proved.

**Proof of Theorem 1.** Since the relations (4)–(6) are a necessary and sufficient condition for optimality in the problem (1)–(3), it suffices to establish their equivalence to the relations (5), (7) and (9). The relation (5) is present in both sets. Moreover, for a fixed  $\hat{u} \in \mathcal{U}_N$ , it uniquely determines the trajectory  $\hat{x}$ , and the relations (6) and (9) uniquely determine the values of the dual variables  $\hat{\psi}(k)$  and  $\hat{\xi}(k)$ , which are related by the formula (A.1) by virtue of Lemma 1. Thus, it remains to check the equivalence of the conditions (4) and (7).

The condition (7) is equivalent to the condition

$$\hat{u}(k) \in \text{Arg} \min_{u \in U(k)} \left( \|\mathcal{A}_N(k+1)u\|^2 + 2\langle \mathcal{A}_N(k+1)u, \mathcal{A}_N(k)\hat{x}(k) - \hat{\xi}(k+1) \rangle \right).$$

Given (5) this is also equivalent to

$$\hat{u}(k) \in \text{Arg} \min_{u \in U(k)} \left( \|\mathcal{A}_N(k+1)u\|^2 + 2\langle \mathcal{A}_N(k+1)u, \mathcal{A}_N(k+1)(\hat{x}(k+1) - \hat{u}(k)) - \hat{\xi}(k+1) \rangle \right),$$

and by virtue of (A.1) it is equivalent to

$$\hat{u}(k) \in \text{Arg} \min_{u \in U(k)} \left( \|\mathcal{A}_N(k+1)u\|^2 - \langle u, \hat{\psi}(k+1) + 2\mathcal{A}_N(k+1)^T \mathcal{A}_N(k+1)\hat{u}(k) \rangle \right).$$

Therefore, the condition (7) is equivalent to the condition

$$\begin{aligned} &\langle \hat{\psi}(k+1), \hat{u}(k) \rangle + \|\mathcal{A}_N(k+1)\hat{u}(k)\|^2 \\ &\geq \langle \hat{\psi}(k+1), u \rangle + 2\langle \mathcal{A}_N(k+1)^T \mathcal{A}_N(k+1)\hat{u}(k), u \rangle - \|\mathcal{A}_N(k+1)u\|^2 \quad \forall u \in U(k) \end{aligned}$$

or, what is the same,

$$\langle \hat{\psi}(k+1), \hat{u}(k) - u \rangle + \|\mathcal{A}_N(k+1)(\hat{u}(k) - u)\|^2 \geq 0 \quad \forall u \in U(k).$$

Clearly, the latter holds if (4) holds. Suppose that (4) does not hold. Then there exists  $u' \in U(k)$  such that

$$\langle \hat{\psi}(k+1), \hat{u}(k) - u' \rangle < 0.$$

But then for  $v_\varepsilon = (1 - \varepsilon)\hat{u}(k) + \varepsilon u'$  with  $\varepsilon > 0$  we have

$$\langle \hat{\psi}(k + 1), \hat{u}(k) - v_\varepsilon \rangle = \varepsilon \langle \hat{\psi}(k + 1), \hat{u}(k) - u' \rangle < 0.$$

Moreover, for values  $\varepsilon \in (0; 1]$  we have  $v_\varepsilon \in U(k)$  since the set  $U(k)$  is convex. In addition,  $v_\varepsilon \rightarrow \hat{u}(k)$  for  $\varepsilon \rightarrow 0$ . Since

$$\begin{aligned} & \langle \hat{\psi}(k + 1), \hat{u}(k) - v_\varepsilon \rangle + \|\mathcal{A}_N(k + 1)(\hat{u}(k) - v_\varepsilon)\|^2 \\ &= \varepsilon \langle \hat{\psi}(k + 1), \hat{u}(k) - u' \rangle + \varepsilon^2 \|\mathcal{A}_N(k + 1)(\hat{u}(k) - u')\|^2 \\ &\leq \varepsilon \left( \langle \hat{\psi}(k + 1), \hat{u}(k) - u' \rangle + \varepsilon \|\mathcal{A}_N(k + 1)\|^2 \|\hat{u}(k) - u'\|^2 \right), \end{aligned}$$

then it follows that for a sufficiently small value of  $\varepsilon > 0$  it will be true that

$$\langle \hat{\psi}(k + 1), \hat{u}(k) - v_\varepsilon \rangle + \|\mathcal{A}_N(k + 1)(\hat{u}(k) - v_\varepsilon)\|^2 < 0,$$

where  $v_\varepsilon \in U(k)$ , so the condition (7) is also not satisfied in this case. We finally establish that the conditions (4) and (7) are equivalent. Theorem 1 is proved.

To prove Theorems 2, 3 and 5 we need the following constructions (see also [9, 10, 14]). Fix  $\hat{u} \in \mathcal{U}_N$ , determine the values of  $\hat{\xi}(k)$  from the equation (9) and set for arbitrary  $k \in \{0, \dots, N - 1\}$  and  $x, u \in \mathbb{R}^n$

$$\begin{aligned} \hat{\varphi}(k, x) &= 2\langle \mathcal{A}_N(k)^T \hat{\xi}(k), x \rangle - \|\mathcal{A}_N(k)x\|^2, \\ \hat{R}(k, x, u) &= \hat{\varphi}(k + 1, A(k)x + u) - \hat{\varphi}(k, x). \end{aligned}$$

**Lemma 2.** *Let  $\hat{u} \in \mathcal{U}_N$  be an arbitrary control, and  $\hat{x}, \hat{\xi} \in \mathcal{X}_N$  be the corresponding solutions to the equations (5) and (9). Then the condition (10) is satisfied if and only if*

$$\hat{R}(k, \tilde{x}(k), \tilde{u}(k)) = \max_{u \in U(k)} \hat{R}(k, \tilde{x}(k), u) \quad \forall k \in \{0, \dots, N - 1\}, \tag{A.2}$$

where  $\tilde{x}$  satisfies (11). In particular, the control  $\hat{u}$  is optimal exactly when

$$\hat{R}(k, \hat{x}(k), \hat{u}(k)) = \max_{u \in U(k)} \hat{R}(k, \hat{x}(k), u) \quad \forall k \in \{0, \dots, N - 1\}. \tag{A.3}$$

**Proof.** By definition we have

$$\begin{aligned} \hat{R}(k, \tilde{x}(k), u) &= \hat{\varphi}(k + 1, A(k)\tilde{x}(k) + u) - \hat{\varphi}(k, \tilde{x}(k)) \\ &= 2\langle \mathcal{A}_N(k + 1)^T \hat{\xi}(k + 1), A(k)\tilde{x}(k) + u \rangle - \|\mathcal{A}_N(k + 1)(A(k)\tilde{x}(k) + u)\|^2 - \hat{\varphi}(k, \tilde{x}(k)) \\ &= -\|\mathcal{A}_N(k)\tilde{x}(k) + \mathcal{A}_N(k + 1)u\|^2 + 2\langle \hat{\xi}(k + 1), \mathcal{A}_N(k)\tilde{x}(k) + \mathcal{A}_N(k + 1)u \rangle - \hat{\varphi}(k, \tilde{x}(k)) \\ &= -\|\mathcal{A}_N(k)\tilde{x}(k) + \mathcal{A}_N(k + 1)u - \hat{\xi}(k + 1)\|^2 + \|\hat{\xi}(k + 1)\|^2 - \hat{\varphi}(k, \tilde{x}(k)). \end{aligned}$$

Since the second and third terms obtained do not depend on  $u$ , we arrive at the first of the assertions to be proved. The second assertion follows directly from the first and Theorem 1. Lemma 2 is proved.

**Proof of Theorem 2.** By virtue of the introduced notation, we have

$$\begin{aligned} J_N(\tilde{u}) &= -\hat{\varphi}(N, \tilde{x}(N)) = -\hat{\varphi}(0, x_0) + \hat{\varphi}(0, x_0) - \hat{\varphi}(N, \tilde{x}(N)) \\ &= -\hat{\varphi}(0, x_0) - \sum_{k=0}^{N-1} \left( \hat{\varphi}(k + 1, \tilde{x}(k + 1)) - \hat{\varphi}(k, \tilde{x}(k)) \right) \\ &= -\hat{\varphi}(0, x_0) - \sum_{k=0}^{N-1} \hat{R}(k, \tilde{x}(k), \tilde{u}(k)) \end{aligned}$$

and, similarly,

$$J_N(\hat{u}) = -\hat{\varphi}(0, x_0) - \sum_{k=0}^{N-1} \hat{R}(k, \hat{x}(k), \hat{u}(k)).$$

Using that according to (9)

$$\begin{aligned} \hat{R}(k, x, \hat{u}(k)) &= \hat{\varphi}(k+1, A(k)x + \hat{u}(k)) - \hat{\varphi}(k, x) \\ &= 2\langle \mathcal{A}_N(k+1)^T \hat{\xi}(k+1), A(k)x + \hat{u}(k) \rangle - \|\mathcal{A}_N(k+1)(A(k)x + \hat{u}(k))\|^2 \\ &\quad - 2\langle \mathcal{A}_N(k)^T \hat{\xi}(k), x \rangle + \|\mathcal{A}_N(k)x\|^2 \equiv 2\langle \mathcal{A}_N(k+1)^T \hat{\xi}(k), \hat{u}(k) \rangle + \|\mathcal{A}_N(k+1)\hat{u}(k)\|^2 \end{aligned}$$

does not depend on  $x$ , then by (10) and Lemma 2 for each  $k \in \{0, \dots, N-1\}$  we have

$$\hat{R}(k, \tilde{x}(k), \tilde{u}(k)) \geq \hat{R}(k, \hat{x}(k), \hat{u}(k)) = \hat{R}(k, \hat{x}(k), \hat{u}(k)). \tag{A.4}$$

Therefore,  $J_N(\tilde{u}) \leq J_N(\hat{u})$ . Theorem 2 is proved.

Let us perform one more general construction, which will be used below in proving Theorems 3 and 5.

Consider the auxiliary system

$$y(k+1) = y(k) + v(k), \quad k = 0, \dots, N-1, \quad y(0) = y_0 := \mathcal{A}_N(0)x_0 \tag{A.5}$$

with geometric constraints  $v(k) \in V(k) := \mathcal{A}_N(k+1)U(k)$ . It is clear that all sets  $V(k)$  are convex, compact, and contain the origin. Let  $\hat{u} \in \mathcal{U}_N$  be given. In the system (A.5), we set  $v(k) = \hat{v}(k) = \mathcal{A}_N(k+1)\hat{u}(k)$ . Then, since  $\hat{u}(k) \in U(k)$ , we have  $\hat{v}(k) \in V(k)$  and the solution  $\hat{y}$  to the system (A.5) satisfies the relation

$$\hat{y}(k) = \mathcal{A}_N(k)\hat{x}(k), \quad k = 0, \dots, N.$$

We write constructions for the system (A.5) similar to those given above. Namely, let us set for  $k \in \{0, \dots, N-1\}$  and  $y, v \in \mathbb{R}^n$

$$\begin{aligned} \hat{\phi}(k, y) &= 2\langle \hat{\xi}(k), y \rangle - \|y\|^2, \\ \hat{\mathcal{R}}(k, y, v) &= \hat{\phi}(k+1, y+v) - \hat{\phi}(k, y), \end{aligned}$$

where the values of  $\hat{\xi}(k)$  are found from (9). Then the following relation holds:

$$\hat{\mathcal{R}}(k, \mathcal{A}_N(k)x, \mathcal{A}_N(k+1)u) = \hat{R}(k, x, u) \quad \forall x, u \in \mathbb{R}^n, \quad k = 0, \dots, N-1. \tag{A.6}$$

Indeed,

$$\begin{aligned} \hat{\mathcal{R}}(k, \mathcal{A}_N(k)x, \mathcal{A}_N(k+1)u) &= \hat{\phi}(k+1, \mathcal{A}_N(k)x + \mathcal{A}_N(k+1)u) - \hat{\phi}(k, \mathcal{A}_N(k)x) \\ &= 2\langle \hat{\xi}(k+1), \mathcal{A}_N(k)x + \mathcal{A}_N(k+1)u \rangle - \|(\mathcal{A}_N(k)x + \mathcal{A}_N(k+1)u)\|^2 \\ &\quad - 2\langle \hat{\xi}(k), \mathcal{A}_N(k)x \rangle + \|\mathcal{A}_N(k)x\|^2 = 2\langle \mathcal{A}_N(k+1)^T \hat{\xi}(k+1), A(k)x + u \rangle \\ &\quad - \|\mathcal{A}_N(k+1)(A(k)x + u)\|^2 - 2\langle \mathcal{A}_N(k)^T \hat{\xi}(k), x \rangle + \|\mathcal{A}_N(k)x\|^2 \\ &= \hat{\phi}(k+1, A(k)x + u) - \hat{\phi}(k, x) = \hat{R}(k, x, u). \end{aligned}$$

**Lemma 3.** *Let  $\hat{u} \in \mathcal{U}_N$  and the values of  $\hat{\xi}(k)$  be found from (9). Then there exists unique pair  $(\tilde{y}, \tilde{v})$  satisfying the conditions*

$$\hat{\mathcal{R}}(k, \tilde{y}(k), \tilde{v}(k)) = \max_{v \in V(k)} \hat{\mathcal{R}}(k, \tilde{y}(k), v), \quad k = 0, \dots, N-1, \tag{A.7}$$

$$\tilde{y}(k+1) = \tilde{y}(k) + \tilde{v}(k), \quad k = 0, \dots, N-1, \quad \tilde{y}(0) = y_0. \tag{A.8}$$

In this case, the condition (A.7) is equivalent to the condition

$$\tilde{v}(k) \in \text{Arg} \min_{v \in V(k)} \|v + \tilde{y}(k) - \hat{\xi}(k+1)\| \quad \forall k \in \{0, \dots, N-1\} \tag{A.9}$$

and for any pair  $(\tilde{x}, \tilde{u})$  satisfying the relations (10)–(11),

$$\tilde{y}(k) = \mathcal{A}_N(k)\tilde{x}(k), \quad k = 0, \dots, N, \quad \tilde{v}(k) = \mathcal{A}_N(k+1)\tilde{u}(k), \quad k = 0, \dots, N-1. \tag{A.10}$$

**Proof.** Let us consider the condition (A.7). By definition we have

$$\begin{aligned} \hat{\mathcal{R}}(k, \tilde{y}(k), v) &= \hat{\phi}(k+1, \tilde{y}(k) + v) - \hat{\phi}(k, \tilde{y}(k)) \\ &= 2\langle \hat{\xi}(k+1), \tilde{y}(k) + v \rangle - \|(\tilde{y}(k) + v)\|^2 - \hat{\phi}(k, \tilde{y}(k)) \\ &= -\|\tilde{y}(k) + v\|^2 + 2\langle \hat{\xi}(k+1), \tilde{y}(k) + v \rangle - \hat{\phi}(k, \tilde{y}(k)) \\ &= -\|\tilde{y}(k) + v - \hat{\xi}(k+1)\|^2 + \|\hat{\xi}(k+1)\|^2 - \hat{\phi}(k, \tilde{y}(k)). \end{aligned}$$

As in the proof of Lemma 2, we find that (A.7) is equivalent to (A.9). In the case  $k = 0$ , the condition (A.9) means that the value  $\tilde{v}(0)$  is found by solving the extremal problem

$$\|v + \mathcal{A}_N(0)x_0 - \hat{\xi}(1)\| \rightarrow \min_{v \in V(0)} .$$

It is clear that for a nonempty convex compact set  $V(0)$  the solution to this problem exists and is unique. Moreover, by (A.8) the value of  $\tilde{y}(1)$  is uniquely determined from here. At the same time, any  $\tilde{u}(0)$  satisfying (10) is one of the solutions to the problem

$$\|\mathcal{A}_N(1)u + \mathcal{A}_N(0)x_0 - \hat{\xi}(1)\| \rightarrow \min_{u \in U(0)} ,$$

whence  $\tilde{v}(0) = \mathcal{A}_N(1)\tilde{u}(0)$ , since the sets  $U(0)$  and  $V(0)$  are related by the equality  $V(0) = \mathcal{A}_N(1)U(0)$ . For  $\tilde{x}(1)$  from (11) we obtain  $\tilde{y}(1) = \mathcal{A}_N(1)\tilde{x}(1)$ , since  $y_0 = \mathcal{A}_N(0)x_0$ .

Let  $\tilde{v}(k-1)$  and  $\tilde{y}(k)$  be known for some  $k \in \{1, \dots, N-1\}$ . Then  $\tilde{v}(k)$  is defined as the unique solution to the problem

$$\|v + \tilde{y}(k) - \hat{\xi}(k+1)\| \rightarrow \min_{v \in V(k)} ,$$

where  $V(k)$  is nonempty, convex, and compact. For known  $\tilde{v}(k)$  and  $\tilde{y}(k)$ , the value  $\tilde{y}(k+1)$  is uniquely determined from (A.8). Thus, the pair  $(\tilde{y}, \tilde{v})$  satisfying conditions (A.7) and (A.8) is completely and uniquely determined.

Carrying out similar comparisons of the values of  $\tilde{v}(k)$  and  $\tilde{u}(k)$ ,  $\tilde{y}(k+1)$  and  $\tilde{x}(k+1)$  for  $k = 1, \dots, N-1$  and taking into account  $V(k) = \mathcal{A}_N(k+1)U(k)$ , we establish the validity of (A.10). Lemma 3 is proved.

**Proof of Theorem 3.** Suppose that the control  $\hat{u}$  is not optimal in problem (1)–(3). Then by Lemma 2 there exists  $r \in \{0, \dots, N-1\}$  such that

$$\hat{R}(r, \hat{x}(r), \hat{u}(r)) < \max_{u \in U(r)} \hat{R}(r, \hat{x}(r), u).$$

Take the smallest such  $r$ . Then by (A.6) for any  $k \in \{0, \dots, r-1\}$  we have

$$\hat{\mathcal{R}}(k, \hat{y}(k), \hat{v}(k)) = \hat{R}(k, \hat{x}(k), \hat{u}(k)) = \max_{u \in U(k)} \hat{R}(k, \hat{x}(k), u) = \max_{v \in V(k)} \hat{\mathcal{R}}(k, \hat{y}(k), v),$$

where  $\hat{y}(k) = \mathcal{A}_N(k)\hat{x}(k)$ ,  $\hat{v}(k) = \mathcal{A}_N(k+1)\hat{u}(k)$ . Since by Lemma 3 there exists unique pair  $(\tilde{y}, \tilde{v})$  satisfying conditions (A.7) and (A.8), we have

$$\tilde{v}(k) = \hat{v}(k), \quad k = 0, \dots, r-1,$$

and, consequently,  $\tilde{y}(r) = \hat{y}(r)$ . Hence, by (A.6) and (A.10), for any pair  $(\tilde{x}, \tilde{u})$  satisfying (10)–(11), we have

$$\begin{aligned} \hat{R}(r, \hat{x}(r), \hat{u}(r)) &< \max_{u \in U(r)} \hat{R}(r, \hat{x}(r), u) = \max_{v \in V(r)} \hat{\mathcal{R}}(r, \hat{y}(r), v) \\ &= \max_{v \in V(r)} \hat{\mathcal{R}}(r, \tilde{y}(r), v) = \max_{u \in U(r)} \hat{R}(r, \tilde{x}(r), u) = \hat{R}(r, \tilde{x}(r), \tilde{u}(r)). \end{aligned}$$

Returning now to the proof of Theorem 2, we find that for  $k = r$  the inequality in (A.4) is strict and therefore

$$J_N(\tilde{u}) < J_N(\hat{u}).$$

If the control  $\hat{u}$  is optimal in the problem (1)–(3), then according to Lemma 2 and (A.6) for all  $k \in \{0, \dots, N - 1\}$  we have

$$\hat{\mathcal{R}}(k, \hat{y}(k), \hat{v}(k)) = \hat{R}(k, \hat{x}(k), \hat{u}(k)) = \max_{u \in U(k)} \hat{R}(k, \hat{x}(k), u) = \max_{v \in V(k)} \hat{\mathcal{R}}(k, \hat{y}(k), v),$$

and hence the pair  $(\tilde{y}, \tilde{v}) = (\hat{y}, \hat{v})$  satisfies the conditions (A.7)–(A.8). Therefore, by Lemma 3 and (A.6), for an arbitrary pair  $(\tilde{x}, \tilde{u})$  satisfying (10)–(11), we have

$$\hat{R}(k, \tilde{x}(k), \tilde{u}(k)) = \hat{\mathcal{R}}(k, \tilde{y}(k), \tilde{v}(k)) = \hat{R}(k, \hat{x}(k), \hat{u}(k)) \quad \forall k \in \{0, \dots, N - 1\}.$$

But then, returning to the proof of Theorem 2, we find that all inequalities in (A.4) for values  $k = 0, \dots, N - 1$  are satisfied as equalities and, therefore,  $J_N(\tilde{u}) = J_N(\hat{u})$ . Theorem 3 is proved.

**Proof of Theorem 4.** Let  $\hat{u} \in \mathcal{U}_N$  be an arbitrary control, and let  $\hat{x}(k)$  and  $\hat{\psi}(k)$  be defined by (5) and (6). Consider a function  $\hat{L} : \mathbb{R}^n \times \mathcal{U}_N \rightarrow \mathbb{R}$  of the form

$$\hat{L}(x, u) = \|x\|^2 - 2\langle \hat{x}(N), x \rangle - \langle \hat{\psi}(0), x_0 \rangle - \sum_{k=0}^{N-1} \langle \hat{\psi}(k+1), u(k) \rangle.$$

We emphasize that in this notation  $x$  is a vector from  $\mathbb{R}^n$ , and  $u$  is an element of the set  $\mathcal{U}_N$ . We show that for the vector  $x(N)$  found from (1)–(2) for fixed  $u \in \mathcal{U}_N$ , it holds

$$\hat{L}(x(N), u) = \|x(N)\|^2 = J_N(u).$$

Indeed, due to (1) and (6) we obtain

$$\begin{aligned} \hat{L}(x(N), u) &= \|x(N)\|^2 - 2\langle \hat{x}(N), x(N) \rangle - \langle \hat{\psi}(0), x_0 \rangle - \sum_{k=0}^{N-1} \langle \hat{\psi}(k+1), u(k) \rangle \\ &= \|x(N)\|^2 + \langle \hat{\psi}(N), x(N) \rangle - \langle \hat{\psi}(0), x_0 \rangle \\ &\quad - \sum_{k=0}^{N-1} \left( \langle \hat{\psi}(k+1), A(k)x(k) + u(k) \rangle - \langle \hat{\psi}(k+1), A(k)x(k) \rangle \right) \\ &= \|x(N)\|^2 + \langle \hat{\psi}(N), x(N) \rangle - \langle \hat{\psi}(0), x_0 \rangle - \sum_{k=0}^{N-1} \left( \langle \hat{\psi}(k+1), x(k+1) \rangle - \langle \hat{\psi}(k), x(k) \rangle \right) = \|x(N)\|^2. \end{aligned}$$

In particular, for any optimal process  $(x^*, u^*)$  in problem (3) we have

$$\hat{L}(x^*(N), u^*) = J_N^* := J_N(u^*).$$

On the other hand, at the point of its global minimum, the function  $\hat{L}$  takes the value

$$\hat{L}^* = \min_{\substack{x \in \mathbb{R}^n \\ u \in \mathcal{U}_N}} \hat{L}(x, u) = -\|\hat{x}(N)\|^2 - \langle \hat{\psi}(0), x_0 \rangle - \sum_{k=0}^{N-1} \max_{u \in U(k)} \langle \hat{\psi}(k+1), u \rangle.$$

Consequently, due to (5) the inequality holds

$$\begin{aligned}
 J_N^* &= \hat{L}(x^*(N), u^*) \geq \hat{L}^* = -\|\hat{x}(N)\|^2 - \langle \hat{\psi}(0), x_0 \rangle - \sum_{k=0}^{N-1} \max_{u \in U(k)} \langle \hat{\psi}(k+1), u \rangle \\
 &= \|\hat{x}(N)\|^2 - 2\|\hat{x}(N)\|^2 - \langle \hat{\psi}(0), x_0 \rangle - \sum_{k=0}^{N-1} \max_{u \in U(k)} \left( \langle \hat{\psi}(k+1), A(k)\hat{x}(k) + u \rangle - \langle \hat{\psi}(k+1), A(k)\hat{x}(k) \rangle \right) \\
 &= \|\hat{x}(N)\|^2 + \langle \hat{\psi}(N), \hat{x}(N) \rangle - \langle \hat{\psi}(0), x_0 \rangle - \sum_{k=0}^{N-1} \max_{u \in U(k)} \left( \langle \hat{\psi}(k+1), A(k)\hat{x}(k) + u \rangle - \langle \hat{\psi}(k), \hat{x}(k) \rangle \right) \\
 &= \|\hat{x}(N)\|^2 - \sum_{k=0}^{N-1} \max_{u \in U(k)} \left( \langle \hat{\psi}(k+1), A(k)\hat{x}(k) + u \rangle - \langle \hat{\psi}(k+1), \hat{x}(k+1) \rangle \right) \\
 &= \|\hat{x}(N)\|^2 - \sum_{k=0}^{N-1} \max_{u \in U(k)} \langle \hat{\psi}(k+1), u - \hat{u}(k) \rangle.
 \end{aligned}$$

Theorem 4 is proved.

Let us make one more construction necessary for proving Theorem 5. Let  $N > 0$  be given. For an arbitrary  $\hat{u} \in \mathcal{U}_N$ , we define two numbers  $f_I(\hat{u})$  and  $f_E(\hat{u})$  as follows. Let

$$f_I(\hat{u}) = \|\tilde{y}(N)\|^2,$$

where  $\tilde{y}$  is determined by the conditions (A.9) and (A.8), in which  $\hat{\xi}$  is found from (9). In addition, we set

$$f_E(\hat{u}) = \|\hat{x}(N)\|^2 - \sum_{k=0}^{N-1} \max_{u \in U(k)} \langle \hat{\psi}(k+1), u - \hat{u}(k) \rangle,$$

where  $\hat{x}$  and  $\hat{\psi}$  are the solutions to the equations (5) and (6).

**Lemma 4.** *The functions  $f_I, f_E : \mathcal{U}_N \rightarrow \mathbb{R}$  are well defined and continuous. Moreover,*

$$f_I(\hat{u}) = J_N(\tilde{u})$$

for any  $\tilde{u}$  satisfying the conditions (10)–(11), and each of the two equalities

$$f_I(\hat{u}) = J_N(\hat{u}) = f_E(\hat{u})$$

holds exactly when  $\hat{u}$  is an optimal control in the problem (1)–(3).

**Proof.** The well-definedness of the function  $f_I$  follows from Lemma 3. The function  $f_E$  is well defined due to the compactness of all sets  $U(k)$ .

The solution  $\hat{\xi}$  to the equation (9) depends continuously on the parameters  $\hat{u}(k)$ , and the solution  $\tilde{y}$  to the equation (A.8) depends continuously on the parameters  $\tilde{v}(k)$ . In addition, for each  $k \in \{0, \dots, N-1\}$  the value  $\tilde{v}(k)$  is determined by the condition (A.9) as the metric projection of the point  $\hat{\xi}(k+1) - \tilde{y}(k)$  onto a nonempty convex compact set  $V(k)$  in the space  $\mathbb{R}^n$ . As is known, the operator of metric projection onto a convex and closed set in a finite-dimensional Euclidean space is well defined and continuous. Therefore, the function  $f_I$  is continuous. Let  $\tilde{u} \in \mathcal{U}_N$  and  $\tilde{x} \in \mathcal{X}_N$  satisfy the conditions (10)–(11). Then, by Lemma 3, we have

$$f_I(\hat{u}) = \|\tilde{y}(N)\|^2 = \|\mathcal{A}_N(N)\tilde{x}(N)\|^2 = \|\tilde{x}(N)\|^2 = J_N(\tilde{u}).$$

In particular, according to Theorem 3, the equality  $f_I(\hat{u}) = J_N(\hat{u})$  is satisfied if and only if  $\hat{u}$  is an optimal control in the problem (1)–(3).

The solution to the equation (5) depends continuously on the parameters  $\hat{u}(k)$ , the solution to the equation (6) depends continuously on the parameters  $\hat{x}(k)$ , and the quantity  $\max_{u \in U(k)} \langle \hat{\psi}(k+1), u \rangle$  is the value of the support function of the nonempty compact set  $U(k)$  at the point  $\hat{\psi}(k+1)$ , which is also continuous. Therefore, the function  $f_E$  is continuous. Moreover, if  $\hat{u}$  is an optimal control in the problem (1)–(3), then the maximum condition (4) is satisfied, whence

$$f_E(\hat{u}) = \|\hat{x}(N)\|^2 = J_N(\hat{u}).$$

Finally, by Theorem 4 for an arbitrary control  $\hat{u} \in \mathcal{U}_N$  the two-sided estimate

$$f_E(\hat{u}) \leq J_N^* \leq J_N(\hat{u})$$

holds, therefore, in the case of equality of the left and right parts, the control  $\hat{u}$  is optimal in the problem (1)–(3). Lemma 4 is proved.

**Proof of Theorem 5.** Since by the condition of the problem  $0 \in U(k)$  for all  $k$ , then for any fixed  $N > 0$  the control  $u^{(0)}$  constructed according to the rule

$$u^{(0)}(k) \in U(k), \quad k = 0, \dots, N-2, \quad u^{(0)}(N-1) = 0,$$

satisfies the geometric constraints, and step 1 of the algorithm is thus well defined. Steps 3 and 6 are well defined due to the compactness of all sets  $U(k)$ .

Let  $N > 0$  be arbitrary. Let  $J_N^*$  denote the optimal value of the function  $J_N$  in problem (1)–(3), and let  $\mathcal{U}_N^* \subset \mathcal{U}_N$  denote the set of all optimal controls in it. Let us consider the sequence  $\{u^{(l)}\} \subset \mathcal{U}_N$  constructed by the algorithm in steps 2–5. According to Theorem 2, the sequence of non-negative numbers  $J_N(u^{(l)})$  is monotonically non-increasing. Therefore, it has some limit  $J^0$ . Since the set  $\mathcal{U}_N$  is compact, there exists a subsequence  $\{u^{(l_m)}\}$  and an element  $u^* \in \mathcal{U}_N$  such that  $u^{(l_m)} \rightarrow u^*$  as  $m \rightarrow \infty$ . Let us show that  $u^* \in \mathcal{U}_N^*$ . Consider the function  $f_I$  constructed above. By Lemma 4, it satisfies the equality

$$f_I(u^{(l_m)}) = J_N(u^{(l_m+1)}),$$

where the function  $f_I$  is continuous. Passing to the limit with respect to  $m \rightarrow \infty$ , we obtain

$$f_I(u^*) = J^0.$$

On the other hand, the function  $J_N$  is also continuous due to the continuity of the squared norm function and the continuous dependence of the solution to the equations (1)–(2) on the parameters  $u(k)$ . Therefore, we have

$$J_N(u^*) = \lim_{m \rightarrow \infty} J_N(u^{(l_m)}) = J^0.$$

According to Lemma 4, the obtained equalities mean that  $u^*$  is an optimal control in the problem (1)–(3). In particular,  $J^0 = J_N(u^*) = J_N^*$ .

Let us now consider the function  $f_E$  constructed above, as well as the function  $\text{dist}(\cdot, \mathcal{U}_N^*)$ . Both of these functions are uniquely defined on  $\mathcal{U}_N$ . By Lemma 4, the function  $f_E$  is continuous and  $f_E(u^*) = J_N^*$  for any  $u^* \in \mathcal{U}_N^*$ . The function  $\text{dist}(\cdot, \mathcal{U}_N^*)$  is continuous by definition and  $\text{dist}(u^*, \mathcal{U}_N^*) = 0$  for any  $u^* \in \mathcal{U}_N^*$ . Let us show that the sequences of numbers  $f_E(u^{(l)})$  and  $\text{dist}(u^{(l)}, \mathcal{U}_N^*)$  have limits  $J_N^*$  and 0, respectively. Assume this is not true and, for definiteness, the sequence  $\{f_E(u^{(l)})\}$  does not tend to  $J_N^*$ . Then for some  $\delta > 0$  there exists a subsequence  $\{u^{(l_m)}\}$  such that  $|f_E(u^{(l_m)}) - J_N^*| > \delta$  for all  $m$ . Passing once again to a subsequence, in view of the compactness of the set  $\mathcal{U}_N \subset \mathbb{R}^{nN}$ , we can assume that there exists an element  $u^* \in \mathcal{U}_N$

such that  $u^{(l_m)} \rightarrow u^*$  for  $m \rightarrow \infty$ . By what was proved above, the inclusion  $u^* \in \mathcal{U}_N^*$  holds, i.e.  $f_E(u^*) = J_N^*$  by Lemma 4. But at the same time, the function  $f_E$  is continuous, and therefore  $|f_E(u^*) - J_N^*| \geq \delta > 0$ . The resulting contradiction shows that for the sequence of numbers  $E_N^{(l)} = f_E(u^{(l)})$  constructed in steps 2–5 of the algorithm, the following convergence holds:

$$E_N^{(l)} \rightarrow J_N^*, \quad l \rightarrow \infty.$$

Similar reasoning leads to the fact that

$$\text{dist}(u^{(l)}, \mathcal{U}_N^*) \rightarrow 0, \quad l \rightarrow \infty. \tag{A.11}$$

Since  $N > 0$  was chosen arbitrarily, the latter is also true for  $N = N_{\min}$ .

Thus, it is shown that for any  $N > 0$  the sequence of numbers  $J_N(u^{(l)})$  monotonically converges to  $J_N^*$  from above and, moreover, the sequence of numbers  $E_N^{(l)}$  converges to  $J_N^*$ . Moreover, by Theorem 4 we have  $E_N^{(l)} \leq J_N^*$  for all  $l$ . Thus, it is proved that

$$J_N(u^{(l)}) \downarrow J_N^* \quad \text{and} \quad E_N^{(l)} \uparrow J_N^* \quad \text{for} \quad l \rightarrow \infty. \tag{A.12}$$

Let  $\varepsilon > 0$  be given. For any  $N < N_{\min}$ , regardless of the value of  $\varepsilon$ , we have  $J_N^* > 0$ , and by virtue of (A.12) there exists an  $l' = l'(N)$  such that  $E_N^{(l')} > 0$ . Therefore, for any such  $N$ , after a finite number of internal iterations, the stopping condition at step 7 is satisfied (if the condition at step 4 was not satisfied before that). If  $N = N_{\min}$ , then  $J_N^* = 0$  and again by virtue of (A.12) there exists an  $l^*$  for which  $J_N(u^{(l^*)}) < \varepsilon^2$ , which means that after a finite number of iterations, the complete stopping condition checked at step 4 is satisfied. Consequently, the algorithm is guaranteed to complete its work in a finite total number of iterations.

Let the algorithm finish its work with an answer  $N_\varepsilon, u_\varepsilon$ . From the previous reasoning it follows that  $N_\varepsilon \leq N_{\min}$ . But if the number  $\varepsilon$  is chosen so small that

$$0 < \varepsilon < \varepsilon^* := \min_{0 < N < N_{\min}} J_N^*,$$

then the strict inequality  $N_\varepsilon < N_{\min}$  is impossible, and therefore  $N_\varepsilon = N_{\min}$ .

By virtue of (A.11), the sequence  $u^{(l)}$  constructed in steps 2–5 for  $N = N_{\min}$  converges to the set  $\mathcal{U}_{N_{\min}}^* = \mathcal{U}^*$  in the sense of the distance between a point and a set. Moreover, this sequence does not change depending on the choice of the value  $\varepsilon \in (0; \varepsilon^*)$ , since the internal stopping condition at step 7 does not depend on  $\varepsilon$ . In this case, the value of the number  $l^* = l^*(\varepsilon)$ , for which  $u_\varepsilon = u^{(l^*)}$  holds by algorithm result, does not decrease with decreasing  $\varepsilon$ . Assuming that there exists  $l_0$  such that  $l^*(\varepsilon) = l_0$  for all sufficiently small  $\varepsilon$ , we obtain  $J_{N_{\min}}(u^{(l_0)}) < \varepsilon^2$  for all small  $\varepsilon > 0$ , i.e.  $J_{N_{\min}}(u^{(l_0)}) = 0$  and  $u^{(l_0)} \in \mathcal{U}^*$ . Otherwise,  $l^*(\varepsilon) \rightarrow \infty$  as  $\varepsilon$  decreases. In each of these cases,  $\text{dist}(u_\varepsilon, \mathcal{U}^*) \rightarrow 0$  as  $\varepsilon \rightarrow 0$ . Theorem 5 is completely proved.

APPENDIX B

Source code of the executable program.

```
import numpy as np
import cvxpy as cp
from tkinter import import filedialog
import pathlib
import json
```

```

MAX_ITER_NUM = 200
MAX_NORM = 1e3

```

```

ex_file_path = filedialog.askopenfilename(
    initialdir = pathlib.Path(__file__).parent.resolve())

```

```

with open(ex_file_path, 'r') as file:
    ex_data = json.load(file)
n = ex_data['n']
A = ex_data['A']
x0 = np.array(ex_data['x0'])
epsilon = ex_data['epsilon']
U = ex_data['U']

```

```

Id = np.identity(n)
Zn = np.array([0] * n)

```

```

u = list()
A_ = list()
x = list()
xi = list()
psi = list()
v = cp.Variable(n)

```

```

N = 1
x.append(x0)
nrm = cp.norm(x0).value
E = 0
iter = 1

```

```

while nrm >= epsilon and nrm < MAX_NORM and iter < MAX_ITER_NUM:
    u.append(Zn)
    iter = 1

```

```

A_.clear()
A_.append(Id)
for k in range(N):
    A_.insert(0, A_[0]@np.array(eval(A.replace('k', str(N-k-1)))))

```

```

while iter < MAX_ITER_NUM:
    xi.clear()
    xi.append(Zn)
    for k in range(N):
        xi.insert(0, xi[0]-A_[N-k]@u[N-k-1])

```

```

x.clear()
x.append(x0)
for k in range(N):
    prob = cp.Problem(cp.Minimize(
        cp.norm(A_[k+1]@v+A_[k]@x[k]-xi[k+1])),
        [eval(constr.replace('k',str(k))) for constr in U])
    prob.solve(solver=cp.SCS)
    u[k] = v.value
    x.append(np.array(eval(A.replace('k',str(k))))@x[k]+u[k])

nrm = cp.norm(x[N]).value
if nrm < epsilon: break

    psi.clear()
    psi.append(-2*x[N])
    for k in range(N):
        psi.insert(0,np.array(eval(
            A.replace('k',str(N-k-1))))).T@psi[0])

E = nrm**2
for k in range(N):
    prob = cp.Problem(cp.Maximize(
        psi[k+1].T@(v-u[k])),
        [eval(constr.replace('k',str(k))) for constr in U])
    E -= prob.solve(solver=cp.CLARABEL)

print(N, iter , x[N] , nrm , np.sqrt(np.max([E,0])))

if E > 0:
    N += 1
    break
    iter += 1

if nrm >= MAXNORM:
    print("No convergence! Try another example.")
elif iter >= MAXITERNUM:
    print("Too many inner iterations! Break.")
else:
    print(N, x[N] , nrm)
    print("Nmin=", N)

ex_data['Nmin'] = N
ex_data['u_opt'] = [[u[k][j] for j in range(n)]
    for k in range(N)]
ex_data['x_opt'] = [[x[k+1][j] for j in range(n)]
    for k in range(N)]

with open(ex_file_path , 'w') as file:
    json.dump(ex_data , file , indent=4, sort_keys=True)

```

Initial data for Example 1.

```
{
  "n": 1,
  "A": "[[1]]",
  "U": ["cp.norm(v) <= 1"],
  "x0": [2.5],
  "epsilon": 0.0001
}
```

Initial data for Example 2.

```
{
  "n": 2,
  "A": "[[4/5*(np.cos(1)+np.sin(1)), -8/5*np.sin(1)],
        [4/5*np.sin(1), 4/5*(np.cos(1)-np.sin(1))]]",
  "U": [
    "cp.quad_form(v, np.array([[2, 1], [1, 3]])) <= 1"
  ],
  "x0": [-37.79, -26.1],
  "epsilon": 0.0001
}
```

Initial data for Example 3.

```
{
  "n": 2,
  "A": "[[np.cos(np.pi/4), -np.sin(np.pi/4)],
        [np.sin(np.pi/4), np.cos(np.pi/4)]]",
  "U": [
    "cp.norm(v, \"inf\") <= 0.9**k*cp.sqrt(3)/2",
    "cp.norm(v) <= 3/4"
  ],
  "x0": [9.33, 0.2],
  "epsilon": 0.0001
}
```

Initial data for Example 4.

```
{
  "n": 2,
  "A": "[[(-1)**k*np.cos(np.pi/4), -(-1)**k*np.sin(np.pi/4)],
        [(-1)**k*np.sin(np.pi/4), (-1)**k*np.cos(np.pi/4)]]",
  "U": [
    "cp.norm(v, \"inf\") <= cp.sqrt(3)/2",
    "cp.norm(v) <= (0.9)**k"
  ],
  "x0": [9.33, 0.2],
  "epsilon": 0.0001
}
```

Initial data for Example 5.

```
{
  "n": 2,
  "A": "[ [np.exp(-1/(k+1))*np.cos(1/(k+1)),
           -np.exp(-1/(k+1))*np.sin(1/(k+1))],
          [np.exp(-1/(k+1))*np.sin(1/(k+1)),
           np.exp(-1/(k+1))*np.cos(1/(k+1))] ]",
  "U": [
    "cp.norm(v) <= np.sqrt((1+np.exp(-2*1/(k+1))
    -2*np.exp(-1/(k+1))*np.cos(1/(k+1)))/2)"
  ],
  "x0": [100,100],
  "epsilon": 0.0001
}
```

### REFERENCES

1. Blanchini, F., Polyhedral Set Constrained Control for Discrete-Time Systems with Unknown Additive Disturbances, *IFAC Proc. Volumes*, 1991, vol. 24, no. 8, pp. 95–100.
2. Blanchini, F. and Ukovich, W., Linear Programming Approach to the Control of Discrete-Time Periodic Systems with Uncertain Inputs, *J. Opt. Theory Appl.*, 1993, vol. 78, no. 3, pp. 523–539.
3. Keerthi, S. and Gilbert, E., Computation of Minimum-Time Feedback Control Laws for Discrete-Time Systems with State-Control Constraints *IEEE Trans. Automatic Control*, 1987, vol. 32, no. 5, pp. 432–435.
4. Abdelhak, A. and Rachik, M., The Linear Quadratic Minimum-Time Problem for a Class of Discrete Systems, *J. Math. Program. Oper. Res.*, 2010, vol. 59, no. 4, pp. 575–587.
5. Amato, F., Cosentino, C., Tommasi, G.D., Pironti, A., and Romano, M., Input-Output Finite-Time Stabilization of Linear Time-Varying Discrete-Time Systems, *IEEE Trans. Automatic Control*, 2022, vol. 67, no. 9, pp. 4438–4450.
6. Chen, D., Bako, L., and Lecoecuche, S., The Minimum-Time Problem for Discrete-Time Linear Systems: A Non-Smooth Optimization Approach, *Proc. IEEE Intern. Conf. Control Appl.*, 2012, pp. 196–201.
7. Lee, J. and Haddad, W.M., Fixed Time Stability and Optimal Stabilisation of Discrete Autonomous Systems, *Intern. J. Control*, 2022, vol. 96, no. 9, pp. 2341–2355.
8. Yang, H., Xia, Y., and Geng, Q., Stabilization on Null Controllable Region, In: *Analysis and Synthesis of Delta Operator Systems with Actuator Saturation. Studies in Systems, Decision and Control*, 2019, vol. 193, pp. 39–65.
9. Krotov, V.F. and Gurman, V.I., *Metody i zadachi optimal'nogo upravleniya* (Methods and problems of optimal control), Moscow: Nauka, 1973.
10. Konnov, A.I. and Krotov, V.F., On Global Methods for the Successive Improvement of Control Processes, *Autom. Remote Control*, 1999, vol. 60, no. 10, pp. 1427–1436.
11. Propoi, A.I., *Elementy teorii optimal'nykh discretnykh protsessov* (Elements of the Theory of Optimal discrete Processes), Moscow: Nauka, 1973.
12. Gabasov, R.F., Kirillova, F.M., Tyatyushkin, A.I., et al., *Konstruktivnye metody optimizacii* (Constructive methods of optimization) (in 5 parts), Minsk: Universitetskoe izdanie, 1984–1998.
13. Srochko, V.A., *Iteracionnye metody resheniya zadach optimal'nogo upravleniya* (Iterative methods for solving optimal control problems), Moscow: Fismatlit, 2000.

14. Ibragimov, D.N. and Tsarkov, K.A., On an Approach to Solving the Time-Optimization Problem for Linear Discrete-Time Systems Based on Krotov Method, *Autom. Remote Control*, 2024, vol. 85, no. 11, pp. 1053–1078.
15. Gabasov, R.F. and Kirillova, F.M., *Kachestvennaya teoriya optimal'nyh processov* (Qualitative theory of optimal processes), Moscow: Nauka, 1981.
16. Jia, Z., Cai, X., and Han, D., Comparison of several fast algorithms for projection onto an ellipsoid, *J. Comp. Appl. Math.*, 2017, vol. 319, pp. 320–337.
17. Kitahara, T. and Sukegawa, N., A Simple Projection Algorithm for Linear Programming Problems, *Algorithmica*, 2018, vol. 81, pp. 167–178.

*This paper was recommended for publication by N.V. Kuznetsov, a member of the Editorial Board*

# PID Controller Design for Suppressing Bounded Exogenous Disturbances

**M. V. Khlebnikov**

*Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia*  
*e-mail: khlebnik@ipu.ru*

Received May 6, 2025

Revised July 23, 2025

Accepted August 15, 2025

**Abstract**—This paper proposes a novel approach to suppressing nonrandom bounded exogenous disturbances in linear control systems using a PID controller. The approach involves reducing the original problem to a nonconvex matrix optimization problem. A gradient method for finding the PID controller parameters is derived and justified. The recursive procedure proposed is simple to implement and yields controllers that are quite satisfactory in terms of engineering performance indices.

*Keywords:* linear system, exogenous disturbances, PID controller, optimization, Lyapunov equation, gradient method

**DOI:** 10.7868/S1608303225110021

## 1. INTRODUCTION

PID controllers are the most common type of automatic controllers; according to various estimates, they account for over 90% of controllers used today. In 2019, the IFAC Industry Committee conducted a survey [1] of its members to identify the control technologies mostly demanded by modern industry; multiple answers were allowed. PID control was ranked first with a large gap: by 91% of respondents. Furthermore, according to medium-term forecasts, this share will even remain exceptionally high, nearly 80%. The extensive application of PID controllers in industry is due to several circumstances. In addition to their suitability for solving most practical problems and low cost, a high demand for such controllers is associated with their simplicity: one needs to correctly choose only three coefficients (gains) to tune a PID controller.

However, the procedures for their practical tuning remain much heuristic: in real plants, PID controllers are often tuned manually, based on an intuitive understanding of the industrial process and the influence of separate PID control components on the latter. Known analytical approaches to PID controller design consider “fixed” models of plants, i.e., they are not universal; moreover, they often neglect the effect of uncertainties. Accordingly, the problem of developing regular approaches to PID controller tuning still retains its meaningfulness and topicality.

Among the publications devoted to this range of problems, note [2], where PID controllers were designed based on genetic algorithms, and [3], where evolutionary algorithms were used for this purpose. Also, we mention the works [4, 5], related to the so-called active disturbance rejection control (ADRC, belonging to model-free control), and [6, 7], with the magnitude optimum method applied for these purposes. The internal model method is also widespread for tuning PID controllers [8–10]. The interested reader will find an extensive bibliography in the recent review [11] as well.

The general trend of the latest decades is the transition to numerical PID controller design methods based on solving optimization problems. The optimization approach to the problem of

suppressing bounded exogenous disturbances, proposed in [12] and originating from [13], lies in this vein. Recall that the classical problem of suppressing nonrandom bounded exogenous disturbances is stated as follows. Consider a linear control system

$$\begin{aligned}\dot{x} &= Ax + Bu + Dw, & x(0) &= x_0, \\ y &= C_1x, \\ z &= C_2x + B_1u\end{aligned}$$

with the state vector  $x(t) \in \mathbb{R}^n$ , the measured output  $y(t) \in \mathbb{R}^\ell$ , the controlled output  $z(t) \in \mathbb{R}^r$ , the control input  $u(t) \in \mathbb{R}^p$ , and an exogenous disturbance  $w(t) \in \mathbb{R}^m$  such that  $\|w(t)\| \leq \bar{w}$  for all  $t \geq 0$ . The problem is to find a stabilizing feedback to reduce the “peak” of the output  $z$ , i.e., the value of  $\sup_{t \geq 0} \max_{\|w\| \leq \bar{w}} \|z(t)\|$ . In [12], this problem was reduced to a nonconvex matrix optimization problem, a gradient method for finding a static linear state-feedback ( $u = Kx$ ) or output-feedback ( $u = Ky$ ) control law was derived, and its justification was provided.

On the other hand, the optimization approach was applied to the PID controller design problem in [14]. More precisely, a regular approach to finding its parameters was proposed, involving the solution of a nonconvex matrix optimization problem. In this case, the controller’s performance was evaluated by a quadratic criterion of the system output: the controller was tuned against uncertainty in the initial conditions to make the system output uniformly small. The recursive procedure proposed therein proved to be very effective and yielded quite satisfactory controllers in terms of engineering performance indices. Later on, the optimization approach was adopted for suppressing bounded exogenous disturbances using a PI controller [15]. This paper continues and develops the above line of research: we design a PID controller for suppressing bounded exogenous disturbances. Note that the approach presented below can be extended to various robust statements of the problem.

From now on,  $\|\cdot\|$  is the Euclidean norm of a vector and the spectral norm of a matrix;  $\|\cdot\|_F$  indicates the Frobenius norm of a matrix;  $^T$  is the transpose symbol;  $\text{tr}$  stands for the trace of a matrix;  $I$  denotes an identity matrix of appropriate dimension;  $\lambda_i(A)$  are the eigenvalues of a matrix  $A$ ; finally,  $\sigma(A) \doteq -\max_i \text{Re}(\lambda_i(A)) > 0$  means the stability degree of a Hurwitz matrix  $A$ .

## 2. PROBLEM STATEMENT

Consider a linear SISO control system described by

$$\begin{aligned}\dot{x} &= Ax + bu + Dw, & x(0) &= x_0, \\ y &= c^T x, \\ z &= Cx,\end{aligned}\tag{1}$$

where  $A \in \mathbb{R}^{n \times n}$ ,  $b \in \mathbb{R}^n$ ,  $D \in \mathbb{R}^{n \times m}$ ,  $c \in \mathbb{R}^n$ ,  $C \in \mathbb{R}^{r \times n}$ , with the state vector  $x(t) \in \mathbb{R}^n$ , the control input  $u(t) \in \mathbb{R}$ , the measured output  $y(t) \in \mathbb{R}$ , the controlled output  $z(t) \in \mathbb{R}^r$ , and an exogenous disturbance  $w(t) \in \mathbb{R}^m$  satisfying the constraint

$$\|w(t)\| \leq \bar{w} \quad \text{for all } t \geq 0.\tag{2}$$

By assumption, the pair  $(A, D)$  is controllable, and the pair  $(A, C)$  is observable.

We will find the control input in the form of a PID controller

$$u(t) = -k_P y(t) - k_I \int_0^t y(\tau) d\tau - k_D \dot{y}(t)\tag{3}$$

that stabilizes the closed-loop system and suppresses the effect of exogenous disturbances  $w$ , minimizing the bounding ellipsoid for the output  $z$ .

Let us conceptually recall the method of invariant ellipsoids; more details can be found in [17]. Consider a linear time-invariant dynamical system described by

$$\begin{aligned} \dot{x} &= Ax + Dw, \quad x(0) = x_0, \\ z &= Cx \end{aligned} \tag{4}$$

with the state vector  $x(t) \in \mathbb{R}^n$ , the output  $z(t) \in \mathbb{R}^r$ , and a measurable exogenous disturbance  $w(t) \in \mathbb{R}^\ell$  that is bounded at each time instant:  $\|w(t)\| \leq 1$  for all  $t \geq 0$ . Assume that system (4) is stable (i.e., the matrix  $A$  is Hurwitz), and the pair  $(A, D)$  is controllable.

An ellipsoid centered at the origin is said to be *invariant* for the dynamical system (4) if any of its trajectories evolving from a point inside the ellipsoid will remain in this ellipsoid at any time instant under all admissible exogenous disturbances of the system.

When evaluating the effect of exogenous disturbances on the system output, it is natural to consider the minimal ellipsoids containing this output (in a certain sense). Obviously, if an ellipsoid

$$\mathcal{E} = \{x \in \mathbb{R}^n: \quad x^T P^{-1} x \leq 1\} \tag{5}$$

with a positive definite matrix  $P$  ( $P \succ 0$ ) is invariant, then the output of system (4) with  $x_0 \in \mathcal{E}$  belongs to the so-called *bounding* ellipsoid

$$\mathcal{E}_z = \{z \in \mathbb{R}^r: \quad z^T (CPC^T)^{-1} z \leq 1\}. \tag{6}$$

In the literature, the linear function  $f(P) = \text{tr} CPC^T$  (the sum of the squared semi-axes of the bounding ellipsoid) is often introduced as a minimality criterion.

An invariance criterion for ellipsoids in terms of linear matrix inequalities (LMIs) was established in the book [16]. Here, we formulate it as follows [17].

**Theorem 1.** *Assume that the matrix  $A$  is Hurwitz, the pair  $(A, D)$  be controllable, and the matrix  $P(\alpha) \succ 0$  satisfies the Lyapunov equation*

$$\left(A + \frac{\alpha}{2}I\right) P + P \left(A + \frac{\alpha}{2}I\right)^T + \frac{1}{\alpha} DD^T = 0$$

on the interval  $0 < \alpha < 2\sigma(A)$ .

Then the minimal bounding ellipsoid for system (4) is obtained by minimizing the function  $f(\alpha) = \text{tr} CP(\alpha)C^T$  on the interval  $0 < \alpha < 2\sigma(A)$ .

The strict convexity of the function  $f(\alpha)$  on the interval  $0 < \alpha < 2\sigma(A)$  was shown in [12] (under assumptions that can certainly be weakened). Also note that if  $\alpha^*$  is the minimum point in the above problem of Theorem 1 and  $x(0) = x_0$  satisfies the condition  $x_0^T P^{-1}(\alpha^*) x_0 \leq 1$ , then the uniform estimate

$$\|z(t)\| \leq \sqrt{\|CP(\alpha^*)C^T\|} \leq \sqrt{f(\alpha^*)}$$

obviously holds for all  $t \geq 0$ .

### 3. SOLUTION APPROACH

Let us introduce an auxiliary scalar variable  $\xi$  as follows:

$$\dot{\xi} = y, \quad \xi(0) = 0.$$

With the extended state vector

$$g = \begin{pmatrix} x \\ \xi \end{pmatrix} \in \mathbb{R}^{n+1},$$

system (1) can be written as

$$\begin{aligned} \dot{g} &= \begin{pmatrix} A & 0 \\ c^T & 0 \end{pmatrix} g + \begin{pmatrix} b \\ 0 \end{pmatrix} u + \begin{pmatrix} D \\ 0 \end{pmatrix} w, \quad g(0) = \begin{pmatrix} x_0 \\ 0 \end{pmatrix}, \\ y &= (c^T \ 0) g, \\ z &= (C \ 0) g. \end{aligned} \quad (7)$$

According to (1) and (3), we have

$$\begin{aligned} u &= -k_P y(t) - k_I \int_0^t y(\tau) d\tau - k_D \dot{y}(t) \\ &= -k_P c^T x - k_I \xi - k_D c^T \dot{x} = -k_P c^T x - k_I \xi - k_D c^T (Ax + bu + Dw) \\ &= -k_P (c^T \ 0) g - k_I (0 \ 1) g - k_D (c^T A \ 0) g - k_D c^T bu - k_D c^T Dw; \end{aligned}$$

consequently,

$$(1 + k_D c^T b) u = -k_P (c^T \ 0) g - k_I (0 \ 1) g - k_D (c^T A \ 0) g - k_D c^T Dw,$$

and

$$\begin{aligned} u &= -\frac{k_P}{1 + k_D c^T b} (c^T \ 0) g - \frac{k_I}{1 + k_D c^T b} (0 \ 1) g \\ &\quad - \frac{k_D}{1 + k_D c^T b} (c^T A \ 0) g - \frac{k_D}{1 + k_D c^T b} c^T Dw. \end{aligned} \quad (8)$$

With the new variables

$$k_1 = \frac{k_P}{1 + k_D c^T b}, \quad k_2 = \frac{k_I}{1 + k_D c^T b}, \quad k_3 = \frac{k_D}{1 + k_D c^T b},$$

the expression (8) takes the form

$$u = -\left(k_1 c^T + k_3 c^T A \quad k_2\right) g - k_3 c^T Dw, \quad (9)$$

and the original PID controller parameters are uniquely given by

$$k_P = \frac{k_1}{1 - k_3 c^T b}, \quad k_I = \frac{k_2}{1 - k_3 c^T b}, \quad k_D = \frac{k_3}{1 - k_3 c^T b}.$$

Thus, system (7) with the feedback control law (9) is described by

$$\begin{aligned} \dot{g} &= \begin{pmatrix} A - k_1 b c^T - k_3 b c^T A & -k_2 b \\ c^T & 0 \end{pmatrix} g + \begin{pmatrix} (I - k_3 b c^T) D \\ 0 \end{pmatrix} w, \quad g(0) = \begin{pmatrix} x_0 \\ 0 \end{pmatrix}, \\ z &= (C \ 0) g. \end{aligned}$$

It can be represented as

$$\begin{aligned} \dot{g} &= (\mathcal{A}_0 + k_1 \mathcal{A}_1 + k_2 \mathcal{A}_2 + k_3 \mathcal{A}_3) g + (\mathcal{D}_0 + k_3 \mathcal{D}_3) w, \quad g(0) = \begin{pmatrix} x_0 \\ 0 \end{pmatrix}, \\ z &= \mathcal{C} g, \end{aligned} \quad (10)$$

where

$$\mathcal{A}_0 = \begin{pmatrix} A & 0 \\ c^T & 0 \end{pmatrix}, \quad \mathcal{A}_1 = \begin{pmatrix} -bc^T & 0 \\ 0 & 0 \end{pmatrix}, \quad \mathcal{A}_2 = \begin{pmatrix} 0 & -b \\ 0 & 0 \end{pmatrix}, \quad \mathcal{A}_3 = \begin{pmatrix} -bc^T A & 0 \\ 0 & 0 \end{pmatrix},$$

$$\mathcal{D}_0 = \begin{pmatrix} D \\ 0 \end{pmatrix}, \quad \mathcal{D}_3 = \begin{pmatrix} -bc^T D \\ 0 \end{pmatrix}, \quad \mathcal{C} = (C \ 0).$$

*Remark 1.* Note that the pair  $(\mathcal{A}_0, \mathcal{D}_0)$  is controllable. Indeed, otherwise there would exist a vector  $0 \neq v \in \mathbb{C}^{n+1}$  such that, for some  $\lambda \in \mathbb{C}$ ,

$$v^* \begin{pmatrix} A & 0 \\ c^T & 0 \end{pmatrix} = \lambda v^*, \quad v^* \begin{pmatrix} D \\ 0 \end{pmatrix} = 0.$$

By writing the vector  $v$  as  $v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ ,  $v_1 \in \mathbb{C}^n$ ,  $v_2 \in \mathbb{C}$ , we obtain

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix}^* \begin{pmatrix} A & 0 \\ c^T & 0 \end{pmatrix} = \lambda \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}^*, \quad \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}^* \begin{pmatrix} D \\ 0 \end{pmatrix} = 0,$$

or  $v_1^* A = \lambda v_1^*$ ,  $v_1^* D = 0$ , which obviously contradicts the controllability of the pair  $(A, D)$ .

Thus, the “nominal” system

$$\begin{aligned} \dot{g} &= \mathcal{A}_0 g + \mathcal{D}_0 w, \\ z &= \mathcal{C} g \end{aligned} \tag{11}$$

is controllable.

Similarly, the pair  $(\mathcal{A}_0, \mathcal{C})$  is observable. Indeed, otherwise there would exist a vector  $0 \neq v \in \mathbb{C}^{n+1}$  such that, for some  $\lambda \in \mathbb{C}$ ,

$$\begin{pmatrix} A & 0 \\ c^T & 0 \end{pmatrix} v = \lambda v, \quad (C \ 0) v = 0.$$

Again, by representing the vector  $v$  as  $v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ ,  $v_1 \in \mathbb{C}^n$ ,  $v_2 \in \mathbb{C}$ , we arrive at

$$\begin{pmatrix} A & 0 \\ c^T & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \lambda \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}, \quad (C \ 0) \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = 0,$$

or  $Av_1 = \lambda v_1$ ,  $Cv_1 = 0$ , which contradicts the observability of the pair  $(A, C)$ . Thus, the “nominal” system (11) is observable as well.

Following the method of invariant ellipsoids, let the state  $g$  of system (10) belong to the invariant ellipsoid (5) generated by a matrix  $0 \prec P \in \mathbb{S}^{n+1}$ . We will minimize the size of the corresponding bounding ellipsoid (6) with respect to the output  $z = \mathcal{C}g$ .

In view of condition (2), according to Theorem 1, we appropriately scale the matrix  $D$  to arrive the problem of minimizing  $\text{tr } \mathcal{C}P\mathcal{C}^T$  subject to the constraint

$$\begin{aligned} &\left( \mathcal{A}_0 + k_1 \mathcal{A}_1 + k_2 \mathcal{A}_2 + k_3 \mathcal{A}_3 + \frac{\alpha}{2} I \right) P + P \left( \mathcal{A}_0 + k_1 \mathcal{A}_1 + k_2 \mathcal{A}_2 + k_3 \mathcal{A}_3 + \frac{\alpha}{2} I \right)^T \\ &\quad + \frac{\bar{w}^2}{\alpha} (\mathcal{D}_0 + k_3 \mathcal{D}_3) (\mathcal{D}_0 + k_3 \mathcal{D}_3)^T = 0. \end{aligned}$$

By introducing the designations

$$A_k \doteq \mathcal{A}_0 + \{\mathcal{A}, k\} \doteq \mathcal{A}_0 + k_1 \mathcal{A}_1 + k_2 \mathcal{A}_2 + k_3 \mathcal{A}_3,$$

$$D_k \doteq \begin{pmatrix} (I - k_3 b c^T) D \\ 0 \end{pmatrix}$$

for convenience, we write this constraint as

$$\left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2} I \right) P + P \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2} I \right)^T + \frac{\bar{w}^2}{\alpha} D_k D_k^T = 0. \quad (12)$$

Here, optimization is performed with respect to the matrix variable  $0 \prec P \in \mathbb{S}^{n+1}$ , the vector variable  $k \in \mathbb{R}^3$ , and the scalar parameter  $\alpha > 0$ .

Finally, as the performance criterion, we take the function

$$f(k, \alpha) = \text{tr } C P C^T + \rho \|k\|^2, \quad \rho > 0, \quad (13)$$

which includes a control penalty. (The coefficient  $\rho > 0$  adjusts its significance.)

Thus, the original problem (the design of a PID controller to suppress exogenous disturbances) has been reduced to the nonconvex matrix optimization problem (13)–(12). Note that for given  $k$  and  $\alpha$ , the matrix  $P$  is found from the Lyapunov equation (12); thus, the independent variables are  $k$  and  $\alpha$ .

Nonconvexity in this optimization problem is due to the presence of the bilinear terms  $A_i k_i P$ ,  $i = 1, \dots, 3$ , in the Lyapunov equation (12), which defines the constraint in the parameter space. Essentially, we are dealing with a bilinear matrix equation, and its solution (as in the case of bilinear matrix inequalities) is an NP-hard problem [18–20]. In some particular cases, bilinear matrix equations (or inequalities) can be linearized by special variable changes, e.g., the problem of stabilizing a linear time-invariant system of the form  $\dot{x} = Ax + Bu$  by a proportional (P) controller  $u = Kx$ . However, even in the problem of stabilizing this system by the output  $y = Cx$  (i.e., with a controller  $u = KCx$ ), such an approach becomes fundamentally impossible, and other solution methods are required; one alternative is to impose stronger conditions on the system, e.g., in the spirit of the passification conditions (see the feedback Kalman–Yakubovich–Popov lemma [19]). Despite the fundamental difference between stabilization and suppression of exogenous disturbances, similar challenges arise in such problems, which demonstrates the complexity of solving the problem stated above. In the next section, after studying some properties of the objective function, we will apply direct optimization in the parameter space to solve this problem.

#### 4. SOME PROPERTIES OF THE FUNCTION $f(k, \alpha)$

The objective function  $f(k, \alpha)$  can be minimized with respect to  $\alpha$ , e.g., using Newton's method, as proposed in [12]. Consider the problem

$$\min f(\alpha), \quad f(\alpha) = \text{tr } P C^T C,$$

subject to the constraint

$$\left( A + \frac{\alpha}{2} I \right) P + P \left( A + \frac{\alpha}{2} I \right)^T + \frac{1}{\alpha} D D^T = 0$$

with respect to the matrix variable  $P \in \mathbb{S}^n$  and the scalar parameter  $0 < \alpha < 2\sigma(A)$ ; the matrix  $A$  is assumed stable (Hurwitz).

Let us choose an initial approximation  $0 < \alpha_0 < 2\sigma(A)$  and apply the iterative process

$$\alpha_{j+1} = \alpha_j - \frac{f'(\alpha_j)}{f''(\alpha_j)},$$

where

$$f'(\alpha) = \text{tr} Y \left( P - \frac{1}{\alpha^2} DD^T \right), \quad f''(\alpha) = 2 \text{tr} Y \left( X + \frac{1}{\alpha^3} DD^T \right),$$

and the matrices  $Y$  and  $X$  are the solutions of the Lyapunov equations

$$\left( A + \frac{\alpha}{2} I \right)^T Y + Y \left( A + \frac{\alpha}{2} I \right) + C^T C = 0$$

and

$$\left( A + \frac{\alpha}{2} I \right) X + X \left( A + \frac{\alpha}{2} I \right)^T + P - \frac{1}{\alpha^2} DD^T = 0,$$

respectively. The method converges globally (faster than the geometric progression with a ratio of 1/2), with quadratic convergence in the neighborhood of the solution.

On the other hand, due to the convexity of the function  $f(\alpha)$  (see [12]), it can be effectively minimized on the interval  $(0, 2\sigma(A))$  by simpler techniques, e.g., the golden section method.

Next, we introduce the function

$$f(k) \doteq \min_{\alpha} f(k, \alpha).$$

Obviously,  $f(k)$  is well-defined and nonnegative on the set  $\mathcal{S}$  of all stabilizing controllers (their gains  $k$ ). Moreover, the set  $\mathcal{S}$  can be nonconvex and disconnected, and its boundaries can be nonsmooth.

*Assumption. Let*

$$k^{(0)} = \begin{pmatrix} k_1^{(0)} \\ k_2^{(0)} \\ k_3^{(0)} \end{pmatrix}$$

*be the gains of a known stabilizing PID controller, i.e., the matrix  $A_{k^{(0)}} = \mathcal{A}_0 + \{\mathcal{A}, k^{(0)}\}$  is Hurwitz.*

We proceed to some properties of the gradient of the objective function.

**Lemma 1.** *The function  $f(k, \alpha)$  is well-defined on the set of stabilizing gains  $k$  for  $0 < \alpha < 2\sigma(A_k)$ . It is differentiable on this admissible set, and the gradient is given by*

$$f'_\alpha(k, \alpha) = \text{tr} Y \left( P - \frac{\bar{w}^2}{\alpha^2} D_k D_k^T \right),$$

$$\frac{1}{2} \frac{\partial f(k, \alpha)}{\partial k_i} = \text{tr} P Y \mathcal{A}_i + \rho k_i - \delta_{i3} \frac{\bar{w}^2}{\alpha} \text{tr} Y D_k \begin{pmatrix} b c^T D \\ 0 \end{pmatrix}^T, \quad i = 1, \dots, 3, \tag{14}$$

where  $\delta_{ij}$  denotes the Kronecker delta, and the matrices  $P$  and  $Y$  are the solutions of equation (12) and the Lyapunov equation

$$\left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2} I \right)^T Y + Y \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2} I \right) + C^T C = 0, \tag{15}$$

respectively.

The function  $f(k, \alpha)$  achieves minimum at an inner point of the admissible set that is determined by the conditions

$$f'_\alpha(k, \alpha) = 0, \quad f'_{k_i}(k, \alpha) = 0, \quad i = 1, \dots, 3.$$

In addition,  $f(k, \alpha)$  as a function of  $\alpha$  is strictly convex on  $0 < \alpha < 2\sigma(A_k)$  and achieves minimum at an inner point of this interval.

The Jacobian of the function  $f(k)$  has the following properties.

**Lemma 2.** *The function  $f(k)$  is twice differentiable, and*

$$\frac{1}{2}(f''(k)v, v) = 2 \operatorname{tr} PY' \{ \mathcal{A}, v \} + \rho(v, v) - \frac{\bar{w}^2}{\alpha} v_3 \operatorname{tr} \left[ 2Y'D_k - v_3 Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T, \quad (16)$$

where  $v \in \mathbb{R}^3$  and the matrices  $P$ ,  $Y$ , and  $Y'$  are the solutions of equation (12), equation (15), and the Lyapunov equation

$$\left( \mathcal{A}_0 + \{ \mathcal{A}, k \} + \frac{\alpha}{2} I \right)^T Y' + Y' \left( \mathcal{A}_0 + \{ \mathcal{A}, k \} + \frac{\alpha}{2} I \right) + \{ \mathcal{A}, v \}^T Y + Y \{ \mathcal{A}, v \} = 0, \quad (17)$$

respectively.

The gradient of the function  $f(k)$  is not Lipschitz on the set  $\mathcal{S}$  of all stabilizing controllers, but it possesses this property on a subset  $\mathcal{S}_0 \subset \mathcal{S}$ . The corresponding result will be presented below.

For obtaining simple quantitative estimates in Lemmas 3 and 4 below, we incorporate the regularizing terms  $\varepsilon$  and  $\delta$  into the optimization problem (13), (12) as follows:

$$\min f(k, \alpha), \quad f(k, \alpha) = \operatorname{tr} P(C^T C + \varepsilon I) + \rho \|k\|^2, \quad 0 < \varepsilon \ll 1,$$

subject to the constraint

$$\left( A_k + \frac{\alpha}{2} I \right) P + P \left( A_k + \frac{\alpha}{2} I \right)^T + \frac{\bar{w}^2}{\alpha} (D_k D_k^T + \delta I) = 0, \quad 0 < \delta \ll 1. \quad (18)$$

The requirement for their introduction can be significantly weakened, but the current aim is to obtain the simplest and most illustrative results.

**Lemma 3.** *The function  $f(k)$  is coercive on the set  $\mathcal{S}$  of all stabilizing controllers (i.e., tends to infinity on its boundary) and, moreover,*

$$f(k) \geq \frac{\bar{w}^2}{4\sigma(A_k)} \frac{\varepsilon}{\|A_k\| + \sigma(A_k)} \|D_k\|_F^2, \quad (19)$$

$$f(k) \geq \rho \|k\|^2.$$

Let us introduce the level set

$$\mathcal{S}_0 = \{k \in \mathcal{S}: f(k) \leq f(k^{(0)})\}.$$

Obviously, Corollary 1 is immediate from Lemma 3.

**Corollary 1.** *For any controller  $k^{(0)} \in \mathcal{S}$ , the set  $\mathcal{S}_0$  is bounded.*

On the other hand, the function  $f(k)$  has a minimum point on the set  $\mathcal{S}_0$  (as a continuous function on a compact set), but the set  $\mathcal{S}_0$  shares no points with the boundary of  $\mathcal{S}$  due to (19). According to the above considerations,  $f(k)$  is differentiable on  $\mathcal{S}_0$ . Consequently, Corollary 2 is true.

**Corollary 2.** *There exists a minimum point  $k_*$  on the set  $\mathcal{S}$ , and  $f'(k_*) = 0$ .*

**Lemma 4.** *On the set  $\mathcal{S}_0$ , the gradient of the function  $f(k)$  is Lipschitz with the constant*

$$\begin{aligned}
 L &= 2\sqrt{3(n+1)}\frac{f(k_0)}{\varepsilon}\frac{2(n+1)^{3/2}f(k_0)}{\bar{w}^2\varepsilon\delta} \\
 &\times \left[ \frac{4f^2(k_0)}{\bar{w}^4\delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho}f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] \\
 &\times \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho}f(k_0)} \right) \max_i \|\mathcal{A}_i\| + 2\rho \\
 &+ 2\bar{w}^2 \left( 2\frac{(n+1)f(k_0)}{\bar{w}^2\varepsilon\delta} \left[ \frac{4f^2(k_0)}{\bar{w}^4\delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho}f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] \right. \\
 &\quad \left. \times \left( 1 + \sqrt{\frac{f(k_0)}{\rho}} \|bc^T\| \right) \|D\|_F + \frac{f(k_0)}{\bar{w}^2\delta} \|bc^T D\|_F \right) \|bc^T D\|_F.
 \end{aligned}$$

These properties of the objective function and its derivatives allow constructing an optimization procedure and justifying its convergence.

### 5. OPTIMIZATION ALGORITHM

We propose an iterative approach to solve the problem posed. This approach is based on the application of the gradient method with respect to variable  $k$  and convex minimization with respect to  $\alpha$ . The corresponding algorithm includes several steps as follows.

- (1) Choose some values of the parameters  $\varepsilon > 0$ ,  $\gamma > 0$ , and  $0 < \tau < 1$  and an initial stabilizing approximation  $k^{(0)}$ . Compute

$$\alpha_0 = \sigma(\mathcal{A}_0 + \{\mathcal{A}, k^{(0)}\}).$$

- (2) On the  $j$ th iteration, the values of  $k^{(j)}$  and  $\alpha_j$  are given. Compute  $A_{k^{(j)}} = \mathcal{A}_0 + \{\mathcal{A}, k^{(j)}\}$  and solve equations (12) and (15) to find the matrices  $P$  and  $Y$ . Compute the gradient

$$H_j = \nabla_k f(k^{(j)}, \alpha_j)$$

from the relations (14). If  $\|H_j\| \leq \varepsilon$ , then take  $k^{(j)}$  as an approximate solution and terminate the algorithm.

- (3) Perform the gradient method step:

$$k^{(j+1)} = k^{(j)} - \gamma_j H_j.$$

Adjust the step length  $\gamma_j > 0$  by fractionating until the following conditions are satisfied:

- (a) The matrix  $\mathcal{A}_0 + \{\mathcal{A}, k^{(j+1)}\} + \frac{\alpha_j}{2}I$  is Hurwitz.
  - (b)  $f(k^{(j+1)}) \leq f(k^{(j)}) - \tau\gamma_j\|H_j\|^2$ .
- (4) Minimize  $f(k^{(j+1)}, \alpha)$  with respect to  $\alpha$  (see the beginning of Section 4) and find  $\alpha_{j+1}$ . Revert to Step 2.

This method converges in the following sense.

**Theorem 2.** *In Algorithm 1, only a finite number of fractions are realized for  $\gamma_j$  on each iteration, the function  $f(k^{(j)})$  is monotonically decreasing, and its gradient vanishes with an exponential rate (like a geometric progression):*

$$\lim_{j \rightarrow \infty} \|H_j\| = 0.$$

Indeed, Algorithm 1 is well-defined at the initial point since  $k^{(0)}$  is a stabilizing controller by the above assumption. For sufficiently small  $\gamma_j$ , the function  $f(k)$  monotonically decreases (moves in the direction of its antigradient); with this step adjustment, the values of  $k^{(j)}$  remain in the domain  $\mathcal{S}_0$ , where Lemma 4 ensures the Lipschitz property of the gradient. Thus, the gradient method for unconstrained minimization is convergent [21]. In particular, condition (b) at Step 3 of Algorithm 1 will be satisfied after a finite number of fractions, and the gradient method will demonstrate gradient convergence with a linear rate.

Naturally, it is difficult to expect convergence to a global minimum: the definitional domain of  $f(k)$  may even be disconnected.

Let us finally emphasize the following aspect.

*Remark 2.* With the Euclidean norm in the objective function (13) being replaced by the weighted

$$\|x\|_\rho \doteq \sqrt{\sum_i \rho_i x_i^2}, \quad \rho_i > 0,$$

one can tune the PID controller parameters more flexibly via assigning different weights.

## 6. EXAMPLES

Consider an illustrative example from the paper [22]. The transfer function has the form

$$G(s) = \frac{1}{(1+s)(1+\alpha s)(1+\alpha^2 s)(1+\alpha^3 s)}, \quad \alpha = 0.5.$$

MATLAB's procedure `tf2ss` gave the following matrices of system (4) in the state space:

$$A = \begin{pmatrix} -15 & -70 & -120 & -64 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad c = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 64 \end{pmatrix}.$$

Let us choose the matrix

$$D = \begin{pmatrix} 10 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

and the controlled output matrix

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

We assign  $\rho = 0.1$  and the stabilizing PID controller

$$k_0 = \begin{pmatrix} 0.6389 \\ 0.9853 \\ 0.3243 \end{pmatrix}$$

as an initial one.

The iterative process of the optimization algorithm terminated with the PID controller

$$k_* = \begin{pmatrix} 0.6670 \\ 0.5466 \\ 0.1081 \end{pmatrix}$$

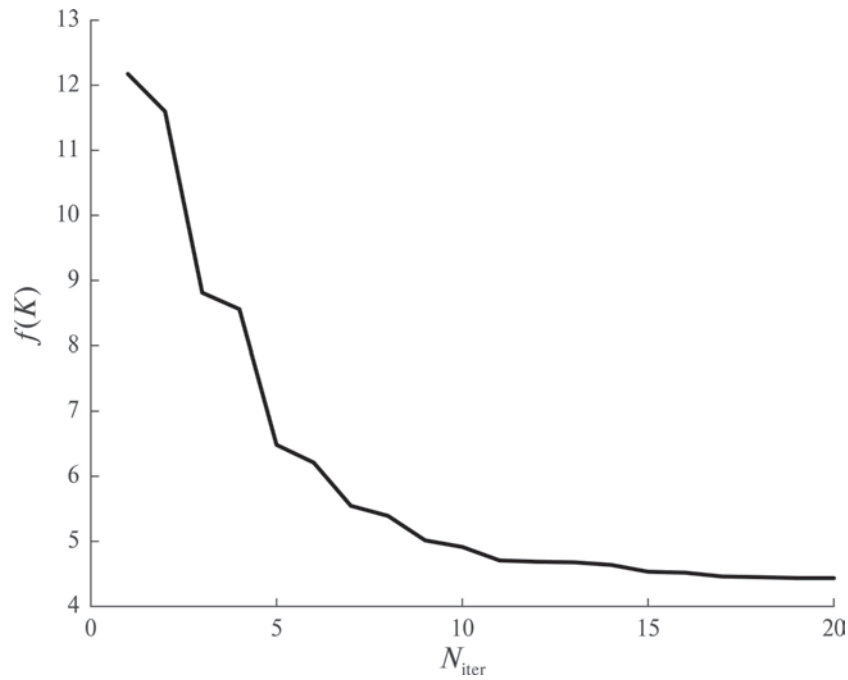


Fig. 1. Optimization procedure.

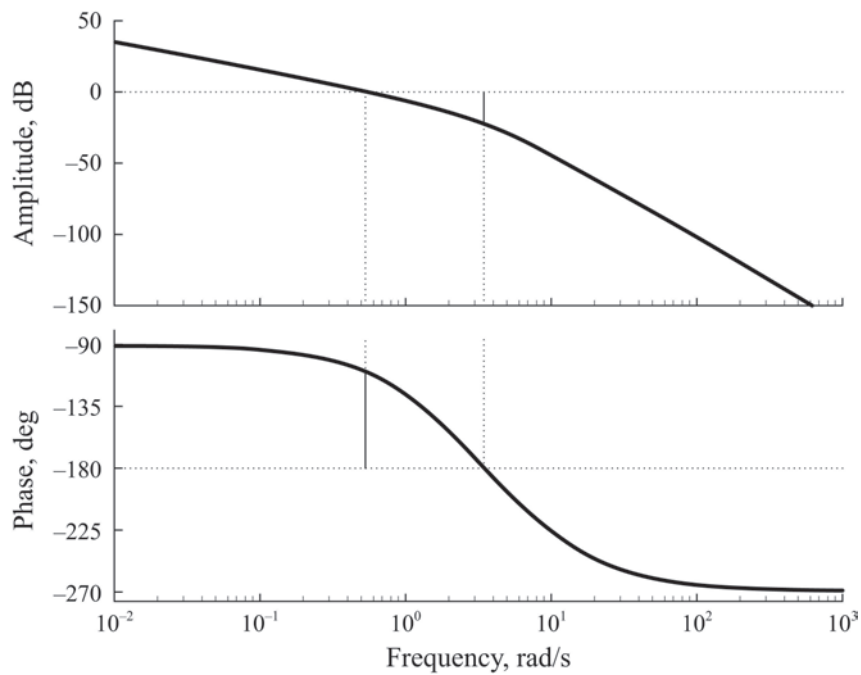


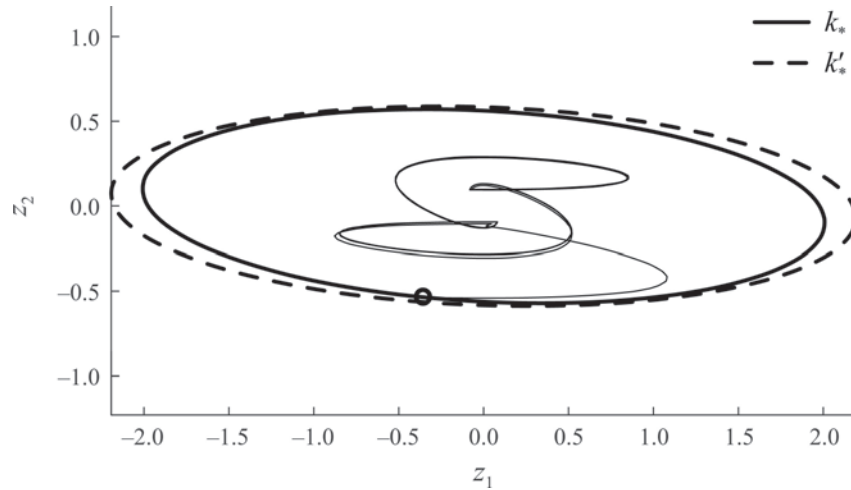
Fig. 2. Bode plots of the closed-loop system.

and the bounding ellipse matrix

$$P_* = \begin{pmatrix} 4.0336 & -0.1999 \\ -0.1999 & 0.3259 \end{pmatrix}, \quad \text{tr } P_* = 4.3595.$$

The dynamics of the criterion  $f(k)$  are shown in Fig. 1.

The closed-loop system with the PID controller  $k_*$  is stable by the Nyquist criterion; its minimal gain and phase margins are 23.3 dB and 70.3°, respectively (Fig. 2).



**Fig. 3.** Bounding ellipses.

By setting  $\rho = 10$ , we arrived at the PID controller

$$k'_* = \begin{pmatrix} 0.1232 \\ 0.2410 \\ -0.0710 \end{pmatrix}$$

and the bounding ellipse matrix

$$P'_* = \begin{pmatrix} 4.8090 & -0.1560 \\ -0.1560 & 0.3453 \end{pmatrix}, \quad \text{tr } P'_* = 5.15437.$$

Therefore, the norm of the vector of PID controller gains was reduced three times, at the “price” of a less than 20% increase in the size of the bounding ellipse.

Figure 3 shows the resulting bounding ellipses and the trajectory of the closed-loop system with the PID controller  $k_*$  under some admissible exogenous disturbance.

All computations were carried out in MATLAB using `cvx` [23].

In the future, we intend to conduct extensive numerical simulations and discuss all computational aspects thoroughly. In this paper, it is important to demonstrate the principal effectiveness of the novel approach in disturbance suppression.

## 7. CONCLUSIONS

This paper has proposed a novel and easily implementable approach to the design problem of PID controllers suppressing nonrandom bounded exogenous disturbances in linear control systems. The approach is based on reducing the original problem to a nonconvex matrix optimization problem, which is then solved by the gradient method. The corresponding algorithm has been constructed and justified.

Although only SISO systems have been considered, the approach is fully transferable to the multidimensional case; here, the constructs will become somewhat more cumbersome, while the conceptual side will change little.

An important direction for further research is to extend the above results to systems with uncertainties.

FUNDING

The research presented in Sections 4 and 5 was supported by the Russian Science Foundation, project no. 25-29-20062, <https://rscf.ru/en/project/25-29-20062/>.

APPENDIX

**Lemma A.1** [12]. *Let  $X$  and  $Y$  be the solutions of the dual Lyapunov equations with a Hurwitz matrix  $A$  :*

$$A^T X + X A + W = 0 \quad \text{and} \quad A Y + Y A^T + V = 0.$$

Then  $\text{tr}(XV) = \text{tr}(YW)$ .

**Lemma A.2** [24]. 1. *Real matrices  $A$  and  $B$  of compatible dimensions satisfy the relations*

$$\begin{aligned} \|AB\|_F &\leq \|A\|_F \|B\|, \\ |\text{tr} AB| &\leq \|A\|_F \|B\|_F, \\ \|A\| &\leq \|A\|_F, \\ AB + B^T A^T &\leq \varepsilon AA^T + \frac{1}{\varepsilon} B^T B \quad \text{for any } \varepsilon > 0. \end{aligned}$$

2. *Positive semidefinite matrices  $A$  and  $B$  satisfy the relations*

$$0 \leq \lambda_{\min}(A)\lambda_{\max}(B) \leq \lambda_{\min}(A) \text{tr} B \leq \text{tr} AB \leq \lambda_{\max}(A) \text{tr} B \leq \text{tr} A \text{tr} B.$$

**Proof of Lemma 1.** Differentiating equation (12) with respect to  $\alpha$  gives

$$\left(\mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I\right) \frac{\partial P}{\partial \alpha} + \frac{\partial P}{\partial \alpha} \left(\mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I\right)^T + P - \frac{\bar{w}^2}{\alpha^2} D_k D_k^T = 0. \tag{A.1}$$

With Lemma A.1 applied to the dual Lyapunov equations (A.1) and (15), we obtain

$$f'_\alpha(k, \alpha) = \text{tr} C \frac{\partial P}{\partial \alpha} C^T = \text{tr} \frac{\partial P}{\partial \alpha} C^T C = \text{tr} Y \left( P - \frac{\bar{w}^2}{\alpha^2} D_k D_k^T \right).$$

To differentiate with respect to  $k$ , we add the increment  $\Delta k$  and denote the corresponding increment of  $P$  by  $\Delta P$  :

$$\begin{aligned} &\left(\mathcal{A}_0 + \{\mathcal{A}, k + \Delta k\} + \frac{\alpha}{2}I\right) (P + \Delta P) \\ &+ (P + \Delta P) \left(\mathcal{A}_0 + \{\mathcal{A}, k + \Delta k\} + \frac{\alpha}{2}I\right)^T + \frac{\bar{w}^2}{\alpha} D_{k+\Delta k} D_{k+\Delta k}^T = 0, \end{aligned}$$

where

$$D_{k+\Delta k} = \begin{pmatrix} (I - (k_3 + \Delta k_3)bc^T)D \\ 0 \end{pmatrix} = D_k - \Delta k_3 \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}.$$

Let us apply linearization and subtract this and the previous equations to get

$$\begin{aligned} &\left(\mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I\right) \Delta P + \Delta P \left(\mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I\right)^T \\ &+ \{\mathcal{A}, \Delta k\} P + P \{\mathcal{A}, \Delta k\}^T - \Delta k_3 \frac{\bar{w}^2}{\alpha} \left[ D_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T + \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} D_k^T \right] = 0. \end{aligned} \tag{A.2}$$

The increment of  $f(k)$  is calculated by linearizing the corresponding terms:

$$\Delta f(k) = \text{tr} \mathcal{C}(P + \Delta P)\mathcal{C}^T + \rho\|k + \Delta k\|^2 - (\text{tr} \mathcal{C}P\mathcal{C}^T + \rho\|k\|^2) = \text{tr} \Delta P\mathcal{C}^T\mathcal{C} + 2\rho k^T \Delta k.$$

Due to Lemma A.1, for the dual Lyapunov equations (A.2) and (15), we have

$$\begin{aligned} \Delta f(k) &= 2 \text{tr} Y \left[ \{\mathcal{A}, \Delta k\}P - \Delta k_3 \frac{\bar{w}^2}{\alpha} D_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T \right] + 2\rho k^T \Delta k \\ &= 2 \text{tr} PY\{\mathcal{A}, \Delta k\} - 2\frac{\bar{w}^2}{\alpha} \text{tr} YD_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T \Delta k_3 + 2\rho k^T \Delta k. \end{aligned}$$

Thus,

$$df(k) = 2 \text{tr} PY \sum_{i=1}^3 \mathcal{A}_i dk_i - 2\frac{\bar{w}^2}{\alpha} \text{tr} YD_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T dk_3 + 2\rho \sum_{i=1}^3 k_i dk_i.$$

The proof of Lemma 1 is complete.

**Proof of Lemma 2.** Let  $v = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \in \mathbb{R}^3$ . The value of  $(f''(k)v, v)$  is obtained by differentiating  $f'(k)$  in the direction  $v$ . For this purpose, linearizing the corresponding terms and using the convenient designation

$$[\text{tr} PY\mathcal{A}] \doteq \begin{pmatrix} \text{tr} PY\mathcal{A}_1 \\ \text{tr} PY\mathcal{A}_2 \\ \text{tr} PY\mathcal{A}_3 \end{pmatrix},$$

we calculate the increment of  $f'(k)$  in the direction  $v$ :

$$\begin{aligned} \frac{1}{2}\Delta f'(k)v &= [\text{tr}(P + \Delta P)(Y + \Delta Y)\mathcal{A}] + \rho(k + \delta v) - \frac{\bar{w}^2}{\alpha} \text{tr}(Y + \Delta Y)D_{k+\delta v} \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T e_3 \\ &- \left( [\text{tr} PY\mathcal{A}] + \rho k - \frac{\bar{w}^2}{\alpha} \text{tr} YD_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T e_3 \right) = [\text{tr}(P + \delta P'(k)v)(Y + \delta Y'(k)v)\mathcal{A}] + \rho(k + \delta v) \\ &- \frac{\bar{w}^2}{\alpha} \text{tr}(Y + \delta Y'(k)v)D_{k+\delta v} \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T e_3 - \left( [\text{tr} PY\mathcal{A}] + \rho k - \frac{\bar{w}^2}{\alpha} \text{tr} YD_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T e_3 \right) \\ &= \delta[\text{tr}(PY'(k)v + P'(k)vY)\mathcal{A}] + \delta\rho v - \frac{\bar{w}^2}{\alpha} \text{tr} \left[ (Y + \delta Y') \left( D_k - \delta v_3 \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right) - YD_k \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T e_3 \\ &= \delta[\text{tr}(PY'(k)v + P'(k)vY)\mathcal{A}] + \delta\rho v - \delta\frac{\bar{w}^2}{\alpha} \text{tr} \left[ Y'D_k - v_3 Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T e_3, \end{aligned}$$

where  $e_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$  and

$$\begin{aligned} \Delta P &= P(k + \delta v) - P(k) = \delta P'(k)v, \\ \Delta Y &= Y(k + \delta v) - Y(k) = \delta Y'(k)v. \end{aligned}$$

Thus, with  $P' = P'(k)v$  and  $Y' = Y'(k)v$ , we have

$$\frac{1}{2}(f''(k)v, v) = \text{tr}(PY' + P'Y)\{\mathcal{A}, v\} + \rho(v, v) - \frac{\bar{w}^2}{\alpha} \text{tr} \left[ Y'D_k - v_3Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T v_3.$$

Furthermore,  $P = P(k)$  is the solution of equation (12). We write it in increments in the direction  $v$  :

$$\begin{aligned} & \left( \mathcal{A}_0 + \{\mathcal{A}, k + \delta v\} + \frac{\alpha}{2}I \right) (P + \delta P') \\ & + (P + \delta P') \left( \mathcal{A}_0 + \{\mathcal{A}, k + \delta v\} + \frac{\alpha}{2}I \right)^T + \frac{\bar{w}^2}{\alpha} D_{k+\delta v} D_{k+\delta v}^T = 0, \end{aligned}$$

or

$$\begin{aligned} & \left( \mathcal{A}_0 + \{\mathcal{A}, k + \delta v\} + \frac{\alpha}{2}I \right) (P + \delta P') + (P + \delta P') \left( \mathcal{A}_0 + \{\mathcal{A}, k + \delta v\} + \frac{\alpha}{2}I \right)^T \\ & + \frac{\bar{w}^2}{\alpha} \left[ D_k - \delta v_3 \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \left[ D_k - \delta v_3 \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right]^T = 0. \end{aligned}$$

After linearization,

$$\begin{aligned} & \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I \right) (P + \delta P') + (P + \delta P') \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I \right)^T \\ & + \delta \left( \{\mathcal{A}, v\}P + P\{\mathcal{A}, v\}^T \right) + \frac{\bar{w}^2}{\alpha} \left[ D_k D_k^T - \delta v_3 D_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T - \delta v_3 \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} D_k^T \right] = 0. \end{aligned}$$

Subtracting equation (12) termwise from this expression gives

$$\begin{aligned} & \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I \right) P' + P' \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I \right)^T \tag{A.3} \\ & + \{\mathcal{A}, v\}P + P\{\mathcal{A}, v\}^T - v_3 \frac{\bar{w}^2}{\alpha} \left[ D_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T + \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} D_k^T \right] = 0. \end{aligned}$$

Similarly,  $Y = Y(k)$  is the solution of the Lyapunov equation (15). We write it in increments in the direction  $v$  :

$$\left( \mathcal{A}_0 + \{\mathcal{A}, k + \delta v\} + \frac{\alpha}{2}I \right)^T (Y + \delta Y') + (Y + \delta Y') \left( \mathcal{A}_0 + \{\mathcal{A}, k + \delta v\} + \frac{\alpha}{2}I \right) + \mathcal{C}^T \mathcal{C} = 0,$$

or

$$\begin{aligned} & \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I \right)^T (Y + \delta Y') + (Y + \delta Y') \left( \mathcal{A}_0 + \{\mathcal{A}, k\} + \frac{\alpha}{2}I \right) \\ & + \delta(\{\mathcal{A}, v\}^T Y + Y\{\mathcal{A}, v\}) + \mathcal{C}^T \mathcal{C} = 0. \end{aligned}$$

Subtracting equation (15) termwise from this expression yields the relation (17).

From (A.3) and (17) it follows that

$$\text{tr} P'Y\{\mathcal{A}, v\} = \text{tr} Y' \left[ \{\mathcal{A}, v\}P - v_3 \frac{\bar{w}^2}{\alpha} D_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T \right] = \text{tr} PY'\{\mathcal{A}, v\} - v_3 \frac{\bar{w}^2}{\alpha} \text{tr} Y'D_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T,$$

so

$$\begin{aligned} \frac{1}{2}(f''(k)v, v) &= \text{tr}(PY' + P'Y)\{\mathcal{A}, v\} + \rho(v, v) - \frac{\bar{w}^2}{\alpha}v_3 \text{tr} \left[ Y'D_k - v_3Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T \\ &= 2 \text{tr} PY'\{\mathcal{A}, v\} + \rho(v, v) - v_3 \frac{\bar{w}^2}{\alpha} \text{tr} Y'D_k \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T - \frac{\bar{w}^2}{\alpha}v_3 \text{tr} \left[ Y'D_k - v_3Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T \\ &= 2 \text{tr} PY'\{\mathcal{A}, v\} + \rho(v, v) - \frac{\bar{w}^2}{\alpha}v_3 \text{tr} \left[ 2Y'D_k - v_3Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T. \end{aligned}$$

The proof of Lemma 2 is complete.

**Proof of Lemma 3.** Consider a sequence of stabilizing controllers  $\{k^{(j)}\} \in \mathcal{S}$  such that  $k^{(j)} \rightarrow k \in \partial\mathcal{S}$ , i.e.,  $\sigma(A_k) = 0$ . In other words, for any  $\epsilon > 0$  there exists a number  $N = N(\epsilon)$  such that

$$|\sigma(A_{k^{(j)}}) - \sigma(A_k)| = \sigma(A_{k^{(j)}}) < \epsilon$$

for all  $j \geq N(\epsilon)$ .

Let  $P_j$  be the solution of the Lyapunov equation (12) associated with the controller  $k^{(j)}$  :

$$\left( A_{k^{(j)}} + \frac{\alpha_j}{2}I \right) P_j + P_j \left( A_{k^{(j)}} + \frac{\alpha_j}{2}I \right)^T + \frac{\bar{w}^2}{\alpha_j} (D_{k^{(j)}} D_{k^{(j)}}^T + \delta I) = 0.$$

Also, let  $Y_j$  be the solution of the dual Lyapunov equation

$$\left( A_{k^{(j)}} + \frac{\alpha_j}{2}I \right)^T Y_j + Y_j \left( A_{k^{(j)}} + \frac{\alpha_j}{2}I \right) + C^T C + \epsilon I = 0.$$

Then

$$\begin{aligned} f(k^{(j)}) &= \text{tr} P_j (C^T C + \epsilon I) + \rho \|k^{(j)}\|^2 \\ &\geq \text{tr} P_j (C^T C + \epsilon I) = \text{tr} Y_j \frac{\bar{w}^2}{\alpha_j} (D_{k^{(j)}} D_{k^{(j)}}^T + \delta I) \\ &\geq \frac{\bar{w}^2}{\alpha_j} \lambda_{\min}(Y_j) \text{tr} (D_{k^{(j)}} D_{k^{(j)}}^T + \delta I) \geq \frac{\bar{w}^2}{\alpha_j} \frac{\lambda_{\min}(C^T C + \epsilon I)}{2 \|A_{k^{(j)}} + \frac{\alpha_j}{2}I\|} \|D_{k^{(j)}}\|_F^2 \\ &\geq \frac{\bar{w}^2}{4\sigma(A_{k^{(j)}})} \frac{\epsilon}{\|A_{k^{(j)}}\| + \sigma(A_{k^{(j)}})} \|D_{k^{(j)}}\|_F^2 \geq \frac{\bar{w}^2}{4\epsilon} \frac{\epsilon}{\|A_{k^{(j)}}\| + \epsilon} \|D_{k^{(j)}}\|_F^2 \xrightarrow{\epsilon \rightarrow 0} +\infty \end{aligned}$$

since

$$0 < \alpha_j < 2\sigma(A_{k^{(j)}})$$

and

$$\|A_{k^{(j)}} + \frac{\alpha_j}{2}I\| \leq \|A_{k^{(j)}}\| + \frac{\alpha_j}{2} \leq \|A_{k^{(j)}}\| + \sigma(A_{k^{(j)}}).$$

On the other hand,

$$f(k^{(j)}) = \text{tr} P_j (C^T C + \epsilon I) + \rho \|k^{(j)}\|^2 \geq \rho \|k^{(j)}\|^2 \xrightarrow{\|k^{(j)}\| \rightarrow +\infty} +\infty.$$

The proof of Lemma 3 is complete.

**Proof of Lemma 4.** We establish several estimates useful for further considerations. First of all,

$$\|\{\mathcal{A}, v\}\| = \left\| \sum_i \mathcal{A}_i v_i \right\| \leq \sum_i \|\mathcal{A}_i\| |v_i| \leq \max_i \|\mathcal{A}_i\| \|v\|_1 \leq \sqrt{3} \max_i \|\mathcal{A}_i\| \|v\| \tag{A.4}$$

since  $\|a\|_1 \leq \sqrt{n}\|a\|$  for  $a \in \mathbb{R}^n$ .

In view of (A.4),  $\alpha$  can be estimated as follows:

$$\begin{aligned} \alpha &< 2\sigma(\mathcal{A}_0 + \{\mathcal{A}, k\}) \leq 2\|\mathcal{A}_0 + \{\mathcal{A}, k\}\| \leq 2(\|\mathcal{A}_0\| + \|\{\mathcal{A}, k\}\|) \\ &\leq 2(\|\mathcal{A}_0\| + \sqrt{3} \max_i \|\mathcal{A}_i\| \|k\|) \leq 2\left(\|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)}\right) \end{aligned} \tag{A.5}$$

since

$$\|k\| \leq \sqrt{\frac{f(k)}{\rho}} \leq \sqrt{\frac{f(k_0)}{\rho}}.$$

Finally,

$$\begin{aligned} \|D_k\|_F &= \left\| \begin{pmatrix} (I - k_3 bc^T)D \\ 0 \end{pmatrix} \right\|_F = \|(I - k_3 bc^T)D\|_F \leq \|I - k_3 bc^T\| \|D\|_F \\ &\leq (1 + |k_3| \|bc^T\|) \|D\|_F \leq \left(1 + \sqrt{\frac{f(k_0)}{\rho}} \|bc^T\|\right) \|D\|_F. \end{aligned} \tag{A.6}$$

Applying Lemma A.2 to (16), we have

$$\begin{aligned} \frac{1}{2} \|f''(k)\| &= \frac{1}{2} \sup_{\|v\|=1} (f''(k)v, v) \leq 2 \sup_{\|v\|=1} |\text{tr} PY'\{\mathcal{A}, v\}| + \rho \sup_{\|v\|=1} |(v, v)| \\ &\quad + \frac{\bar{w}^2}{\alpha} \sup_{\|v\|=1} \left| v_3 \text{tr} \left[ 2Y'D_k - v_3 Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right] \begin{pmatrix} bc^T D \\ 0 \end{pmatrix}^T \right| \\ &\leq \|P\|_F \|Y'\|_F \sup_{\|v\|=1} \|\{\mathcal{A}, v\}\| + \rho + \bar{w}^2 \sup_{\|v\|=1} |v_3| \sup_{\|v\|=1} \left\| \frac{2}{\alpha} Y'D_k - \frac{v_3}{\alpha} Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right\|_F \left\| \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right\|_F \\ &\leq \sqrt{3} \|P\|_F \|Y'\|_F \max_i \|\mathcal{A}_i\| + \rho + \bar{w}^2 \left( 2 \frac{1}{\alpha} \|Y'D_k\|_F + \sup_{\|v\|=1} |v_3| \frac{1}{\alpha} \left\| Y \begin{pmatrix} bc^T D \\ 0 \end{pmatrix} \right\|_F \right) \|bc^T D\|_F \\ &\leq \underbrace{\sqrt{3} \|P\|_F}_{(A.8)} \underbrace{\|Y'\|_F}_{(A.12)} \max_i \|\mathcal{A}_i\| + \rho + \bar{w}^2 \left( \underbrace{2 \frac{1}{\alpha} \|Y'\|}_{(A.11)} \underbrace{\|D_k\|_F}_{(A.6)} + \underbrace{\frac{1}{\alpha} \|Y\| \|bc^T D\|_F}_{(A.9)} \right) \|bc^T D\|_F. \end{aligned} \tag{A.7}$$

The upper bound for  $\|P\|$  is derived as follows:

$$\varepsilon \|P\| \leq \lambda_{\min}(\mathcal{C}^T \mathcal{C} + \varepsilon I) \|P\| \leq \text{tr} P(\mathcal{C}^T \mathcal{C} + \varepsilon I) = f(k) - \rho \|k\|^2 \leq f(k) \leq f(k_0);$$

consequently,

$$\|P\| \leq \frac{f(k_0)}{\varepsilon}$$

and

$$\|P\|_F \leq \sqrt{n+1} \frac{f(k_0)}{\varepsilon}. \tag{A.8}$$

Next, by Lemma A.1, equations (18) and (15) imply

$$\text{tr} Y(D_k D_k^T + \delta I) = \text{tr} P(\mathcal{C}^T \mathcal{C} + \varepsilon I).$$

As a result,

$$\begin{aligned} \frac{\bar{w}^2 \delta}{\alpha} \|Y\| &\leq \frac{\bar{w}^2}{\alpha} \lambda_{\min}(D_k D_k^T + \delta I) \text{tr} Y \leq \text{tr} Y \frac{\bar{w}^2}{\alpha} (D_k D_k^T + \delta I) \\ &= \text{tr} P(\mathcal{C}^T \mathcal{C} + \varepsilon I) = f(k) - \rho \|k\|^2 \leq f(k) \leq f(k_0), \end{aligned}$$

which gives

$$\frac{1}{\alpha} \|Y\| \leq \frac{f(k_0)}{\bar{w}^2 \delta} \quad (\text{A.9})$$

and, due to (A.5),

$$\|Y\| \leq \frac{\alpha}{\bar{w}^2 \delta} f(k_0) \leq \frac{2f(k_0)}{\bar{w}^2 \delta} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right). \quad (\text{A.10})$$

Moreover, using Lemma A.2 together with the upper bounds (A.4) and (A.10), for  $\|v\| = 1$  we have

$$\begin{aligned} \lambda_{\max}(\{\mathcal{A}, v\}^T Y + Y \{\mathcal{A}, v\}) &= \|\{\mathcal{A}, v\}^T Y + Y \{\mathcal{A}, v\}\| \leq \|Y^2 + \{\mathcal{A}, v\}^T \{\mathcal{A}, v\}\| \\ &\leq \|Y\|^2 + \|\{\mathcal{A}, v\}\|^2 \leq \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2. \end{aligned}$$

By Lemma A.1, equations (18) and (17) lead to

$$\text{tr} Y' \frac{\bar{w}^2}{\alpha} (D_k D_k^T + \delta I) = \text{tr} P(\{\mathcal{A}, v\}^T Y + Y \{\mathcal{A}, v\}).$$

Therefore,

$$\begin{aligned} \frac{\bar{w}^2 \delta}{\alpha} \|Y'\| &\leq \frac{\bar{w}^2}{\alpha} \lambda_{\min}(D_k D_k^T + \delta I) \text{tr} Y' \leq \text{tr} Y' \frac{\bar{w}^2}{\alpha} (D_k D_k^T + \delta I) \\ &= \text{tr} P(\{\mathcal{A}, v\}^T Y + Y \{\mathcal{A}, v\}) \leq \lambda_{\max}(\{\mathcal{A}, v\}^T Y + Y \{\mathcal{A}, v\}) \text{tr} P \\ &\leq \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] (n+1) \|P\| \\ &\leq \frac{(n+1)f(k_0)}{\varepsilon} \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right], \end{aligned}$$

so

$$\frac{1}{\alpha} \|Y'\| \leq \frac{(n+1)f(k_0)}{\bar{w}^2 \varepsilon \delta} \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right]. \quad (\text{A.11})$$

Accordingly, in view of (A.5), we obtain

$$\begin{aligned} \|Y'\| &\leq \alpha \frac{(n+1)f(k_0)}{\bar{w}^2 \varepsilon \delta} \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] \\ &\leq \frac{2(n+1)f(k_0)}{\bar{w}^2 \varepsilon \delta} \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] \\ &\quad \times \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right) \end{aligned}$$

and

$$\begin{aligned} \|Y'\|_F &\leq \sqrt{n+1} \|Y'\| \\ &\leq \frac{2(n+1)^{3/2} f(k_0)}{\bar{w}^2 \varepsilon \delta} \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] \\ &\quad \times \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right). \end{aligned} \quad (\text{A.12})$$

Returning to (A.7), based on the upper bounds above, we finally arrive at the relation

$$\begin{aligned} \frac{1}{2} \|f''(k)\| &\leq \sqrt{3(n+1)} \frac{f(k_0)}{\varepsilon} \frac{2(n+1)^{3/2} f(k_0)}{\bar{w}^2 \varepsilon \delta} \\ &\times \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] \\ &\times \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right) \max_i \|\mathcal{A}_i\| + \rho \\ &+ \bar{w}^2 \left( 2 \frac{(n+1)f(k_0)}{\bar{w}^2 \varepsilon \delta} \left[ \frac{4f^2(k_0)}{\bar{w}^4 \delta^2} \left( \|\mathcal{A}_0\| + \max_i \|\mathcal{A}_i\| \sqrt{\frac{3}{\rho} f(k_0)} \right)^2 + 3 \max_i \|\mathcal{A}_i\|^2 \right] \right. \\ &\left. \times \left( 1 + \sqrt{\frac{f(k_0)}{\rho}} \|bc^T\| \right) \|D\|_F + \frac{f(k_0)}{\bar{w}^2 \delta} \|bc^T D\|_F \right) \|bc^T D\|_F. \end{aligned}$$

The proof of Lemma 4 is complete.

## REFERENCES

1. The IFAC Newsletter. No. 2. April, 2019.  
[https://www.ifac-control.org/newsletter\\_archive/IFAC\\_Newsletter\\_2019\\_2\\_April.pdf](https://www.ifac-control.org/newsletter_archive/IFAC_Newsletter_2019_2_April.pdf)
2. Krohling, R.A. and Rey, J.P., Design of Optimal Disturbance Rejection PID Controllers Using Genetic Algorithms, *IEEE Transactions on Evolutionary Computation*, 2001, vol. 5, no. 1, pp. 78–82.
3. Kim, D.H. and Cho, J.H., Robust Tuning for Disturbance Rejection of PID Controller Using Evolutionary Algorithm, *Proc. 2004 IEEE Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS'04)*, Banff, Canada, 2004, vol. 1, pp. 248–253.
4. Han, J., From PID to Active Disturbance Rejection Control, *IEEE Transactions on Industrial Electronics*, 2009, vol. 56, no. 3, pp. 900–906.
5. Teppa-Garran, P.A. and Garcia, G., Optimal Tuning of PI/PID/PID<sup>(n-1)</sup> Controllers in Active Disturbance Rejection Control, *J. Control Engineer. Appl. Inform.*, 2013, vol. 15, no. 4, pp. 26–36.
6. Vrančić, D., Strmčnik, S., and Juričić, D., A Magnitude Optimum Multiple Integration Tuning Method for Filtered PID Controller, *Automatica*, 2001, vol. 37, no. 9, pp. 1473–1479.
7. Vrančić, D., Strmčnik, S., Kocijan, J., and de Moura Oliveira, P.B., Improving Disturbance Rejection of PID Controllers by Means of the Magnitude Optimum Method, *ISA Transactions*, 2010, vol. 49, no. 1, pp. 47–56.
8. Rivera, D.E., Morari, M., and Skogestad, S., Internal Model Control: PID Controller Design, *Industrial and Engineering Chemistry Process Design and Development*, 1986, vol. 25, no. 1, pp. 252–265.
9. Saxena, S. and Hote, Y.V., Simple Approach to Design PID Controller via Internal Model Control, *Arab. J. Sci. Engineer.*, 2016, vol. 41, pp. 3473–3489.
10. Zhou, C. and Shen, Y., A PID Control Method Based on Internal Model Control to Suppress Vibration of the Transmission Chain of Wind Power Generation System, *Energies*, 2022, vol. 15, no. 16, art. no. 5919.
11. Borase, R.P., Maghade, D.K., Sondkar, S.Y., and Pawar, S.N., A Review of PID Control, Tuning Methods and Applications, *Int. J. Dynam. Control*, 2021, vol. 9, pp. 818–827.
12. Polyak, B.T. and Khlebnikov, M.V., Static Controller Synthesis for Peak-to-Peak Gain Minimization as an Optimization Problem, *Autom. Remote Control*, 2021, vol. 82, no. 9, pp. 1530–1553.
13. Fatkhullin, I. and Polyak, B., Optimizing Static Linear Feedback: Gradient Method, *SIAM J. Control Optimiz.*, 2021, vol. 59, no. 5, pp. 3887–3911.

14. Polyak, B.T. and Khlebnikov, M.V., New Criteria for Tuning PID Controllers, *Autom. Remote Control*, 2022, vol. 83, no. 11, pp. 1724–1741.
15. Khlebnikov, M.V., PI Controller Design for Suppressing Exogenous Disturbances, *Autom. Remote Control*, 2023, vol. 84, no. 8, pp. 901–917.
16. Boyd, S., El Ghaoui, L., Feron, E., and Balakrishnan, V., *Linear Matrix Inequalities in System and Control Theory*, Philadelphia: SIAM, 1994.
17. Polyak, B.T., Khlebnikov, M.V., and Shcherbakov, P.S., *Upravlenie lineinymi sistemami pri vneshnikh vozmushcheniyakh: Tekhnika lineinykh matrichnykh neravenstv* (Control of Linear Systems Subjected to Exogenous Disturbances: The Technique of Linear Matrix Inequalities), Moscow: LENAND, 2014.
18. Churilov, A.N. and Gessen, A.V., *Issledovanie lineinykh matrichnykh neravenstv. Putevoditel' po programmnykh paketam* (Investigation of Linear Matrix Inequalities. A Guide to Software Packages), St. Petersburg: St. Petersburg University, 2004.
19. Fradkov, A., Passification of Non-square Linear Systems and Feedback Yakubovich–Kalman–Popov Lemma, *Eur. J. Control*, 2003, vol. 9, no. 6, pp. 577–586.
20. Toker, O. and Ozbay, H., On the NP-hardness of Solving Bilinear Matrix Inequalities and Simultaneous Stabilization with Static Output Feedback, *Proc. 1995 American Control Conference (ACC'95)*, Seattle, USA, June 21–23, 1995, vol. 4, pp. 2525–2526.
21. Polyak, B.T., *Introduction to Optimization*, New York: Optimization Software, 1987.
22. Åström, K.J. and Hägglund, T., Benchmark Systems for PID Control, *IFAC Proceedings Volumes*, 2000, vol. 33, no. 4, pp. 165–166.
23. Grant, M. and Boyd, S., CVX: Matlab Software for Disciplined Convex Programming, ver. 2.1. <http://cvxr.com/cvx>
24. Horn, R. and Johnson, C., *Matrix Analysis*, Cambridge: Cambridge University Press, 1985.

*This paper was recommended for publication by A.L. Fradkov, a member of the Editorial Board*

# Relay Controller Design in Self-Oscillating Control Systems Based on Training Examples

**V. A. Mozzhechkov**

*Tula State University, Tula, Russia*

*e-mail: v.a.moz@yandex.ru*

Received March 13, 2024

Revised June 10, 2025

Accepted June 20, 2025

**Abstract**—The problem of designing relay controllers in a self-oscillating control system with a linear plant is considered. It is required to ensure self-oscillations in the system with given parameters and to make its behavior maximally close to the desired one, defined by a set of time-varying system output laws (training examples). Controller's structural constraints and the requirement for the degree of stability of self-oscillations are taken into account. Explicit-form relations are obtained and applied to develop an iterative method for solving the above problem. This method yields relay controllers having a simple structure. Some examples of implementing the method are provided.

*Keywords:* relay controller, self-oscillations, training examples

**DOI:** 10.7868/S1608303225110033

## 1. INTRODUCTION

Relay feedback control, as well as analysis and design of self-oscillating control systems based on its application, are classical themes of control theory [1–9].

The topicality of the problem of designing self-oscillating control systems with a relay controller is confirmed by numerous examples of their practical use for various purposes. In particular, we mention self-oscillating systems for regulating various physical quantities [1], servosystems [1, 2, 4, 5, 8], self-tuning and adaptive systems using self-oscillations in the controller auto-tuning mode [6–8], self-oscillating voltage converters and signal auto-generators [1, 10], vibratory gyroscopes and accelerometers [11, 12], sensors based on nanoresonators, resonistors and cantilevers [13, 14], vibration machines [15], control systems for underactuated multilink mechanisms and robots using self-oscillations [16], and many other devices. In these systems, self-oscillations are an operating mode where a controlled variable oscillates with a given frequency within an admissible range.

For a long time, the predominant approach to the analysis and design of self-oscillating systems was based on frequency-domain methods [1–3, 8]. Among them, the harmonic linearization method [2, 3] became the most widespread due to its simple application. A significantly more accurate and universal frequency-domain method for analyzing self-oscillations was developed by Ya.Z. Tsyppkin [1], with his original concept of the hodograph (also called locus or plot in the literature) of a relay system. The development of state-space analysis methods for relay systems ensured higher efficiency of their analysis and design procedures, as well as the deeper consideration of possible operating modes of relay systems compared to frequency-domain methods. A method for designing self-oscillating systems based on the concept of a phase plot was proposed in [4, 5]. A phase plot is a line in the state space of a plant in which each point  $x^*$  corresponds to relay switching from minus to plus for some self-oscillation period inherent to it. The phase plot remains unchanged when choosing feedback coefficients (gains), which significantly simplifies the system

design and allows achieving the assigned self-oscillation parameters with high accuracy. An analytical dependence of the vector  $x^*$  on the self-oscillation period and the matrices and vectors of parameters appearing in the state-space description of a linear plant was obtained, and algebraic conditions for the existence of self-oscillations of a given frequency, including the condition of their local stability, were presented in [6, 7].

For a control system with a linear plant, relay controllers ensuring self-oscillations of given frequency and amplitude and making its behavior maximally close to the desired one, defined by a given transfer function, were designed in [9]. The design method proposed therein is based on a polynomial representation of the harmonically linearized system and reduces the design procedure to solving systems of linear algebraic equations and inequalities. Due to the use of harmonic linearization in the method [9], in some cases, the self-oscillation frequency significantly deviates from its given value, sometimes leading to an appreciable difference between the behavior of the system designed and the desired one.

In this paper, to overcome the limitations of the method [9] when designing a relay self-oscillating system, we utilize the results of the studies [4–7], with their higher accuracy in predicting self-oscillation parameters compared to the harmonic linearization method. The design procedure proposed below is based on explicit-form relations obtained using the state-space description of a plant and is the result of developing and extending the algorithm [17] to the class of relay self-oscillating control systems. The desired behavior of the system is defined by a set of its time-varying output laws, acting as training examples, as well as by the requirement for its degree of stability. The controller implements static feedback with a relay output. We propose a method for finding an initial approximation of the desired gains and refining them iteratively. In the general case, when the plant output contains several measurable and linearly independent variables, controllers with a simple structure [18, 19] are designed. In such controllers, only those gains are non-zero that are necessary and sufficient to give the system the desired properties.

## 2. PROBLEM STATEMENT

Consider a plant described by the system of equations

$$\dot{x} = Ax + Bu, \quad (1)$$

$$y = Cx \quad (2)$$

with the following notation:  $x$  is the state vector;  $\dot{x} = dx/dt$ ;  $t$  indicates time;  $u$  is the scalar control input;  $y$  is the output vector, all its components are measurable and available for control; the matrices  $A$  and  $C$  and the vector  $B$  are given, their elements, as well as the values of  $u$  and all components of the vectors  $x$  and  $y$ , are real numbers. By assumption, the pairs  $(A, B)$  and  $(A, C)$  are controllable and observable, respectively.

The mathematical description of the controller has the form

$$\tilde{u} = K^\top y, \quad (3)$$

$$u = \text{sgn}(\tilde{u}), \quad (4)$$

where the scalar variable  $\tilde{u}$  is the output of the linear part (3) of the controller; the vector  $K$  of the gains (real numbers) has to be determined; the function  $\text{sgn}(\cdot)$  describes the relay forming the control input  $u$ , its value is  $-1$ ,  $+1$ , or a real number from the closed interval  $[-1, 1]$  for the negative, positive, zero value of its argument, respectively. The amplitude values of  $u$  being equal to 1 do not restrict the generality of the problem under consideration: the real amplitudes of the control input can be taken into account in (1) when determining the vector  $B$ .

Let us impose structural constraints on the linear part of the controller, which usually reduce [20] to the zero-value requirement for some components of the vector  $K = (k_i)$ . Therefore, we adopt the condition

$$k_i = 0, \forall i \notin S, \tag{5}$$

where  $S$  is the set of the serial numbers of those gains  $k_i$  that are not required to be zero. The set  $S$  in the controller description (3)–(5) defines the list of its feedback signals and, in view of (3)–(5), its structure. Assume that when solving this problem, the controller structure can change only as a result of changing the set  $S$ . From this point onwards, the choice of the controller structure is hence identified with the choice of the set  $S$ , and the terms “controller structure” and “set  $S$ ” are used as synonyms.

The solution of the discontinuous system (1)–(5) is understood in the Filippov sense [21].

In system (1)–(5), the steady-state operating mode should be an asymptotically orbitally stable periodic motion representing self-oscillations with a given frequency  $\omega_0$  and a given amplitude  $\tilde{U}$  of oscillations of the signal  $\tilde{u}$ . The self-oscillations are associated with an isolated closed trajectory in the phase space, i.e., a limit cycle. As a result of solving the controller design problem, the limit cycle  $L$  must be stable, symmetric (with each point  $x \in L$  the point  $-x \in L$  must be associated), and simple (the signal  $\tilde{u}$  must change its sign only twice per self-oscillation period).

We describe the stability margin requirement for self-oscillations by the condition

$$\eta(K) \geq \tilde{\eta}, \tag{6}$$

where  $\eta(K)$  is the stability margin of self-oscillations and  $\tilde{\eta}$  is a given lower limit of its admissible values. The mathematical description of the function  $\eta(K)$  will be presented in Section 3.

Let the desired behavior of system (1)–(5) be defined by specifying, for initial conditions  $x_0^\gamma$ ,  $\gamma \in \{\overline{1, q}\}$ , the corresponding desired trajectories of the output (2) of system (1)–(5), represented by the vectors  $Y_\gamma = (y_k^\gamma)$ ,  $k \in \{\overline{1, N}\}$ , of its values at discrete time instants  $t = k\Delta t$ , where  $\Delta t$  is a sampling step and  $k$  is a natural number. The values of  $N$  and  $\Delta t$  determine the time interval  $[0, N\Delta t]$  on which the desired trajectories are specified. The initial states  $x_0^\gamma$ ,  $\gamma \in \{\overline{1, q}\}$ , must be sufficiently distant from the steady-state trajectory so that the transient has a duration significantly exceeding the self-oscillation period. In addition, the initial states must differ significantly from each other. It is reasonable to assign  $q \in [1, n]$ . The set  $Q = \{(x_0^\gamma, Y_\gamma)\}$ ,  $\gamma \in \{\overline{1, q}\}$ , is a set of training examples that defines the desired behavior of the system designed during its transition from a disturbed to steady-state motion mode. The requirement for system (1)–(5) to match the desired behavior is written as

$$\varepsilon_k^{\gamma-} \leq y(x_0^\gamma, K)_k - y_k^\gamma \leq \varepsilon_k^{\gamma+}, \forall k \in \{\overline{1, N}\}, \forall \gamma \in \{\overline{1, q}\}, \tag{7}$$

where  $\varepsilon_k^{\gamma-}$  and  $\varepsilon_k^{\gamma+}$  are given constant vectors;  $y(x_0^\gamma, K)_k$  is the output of system (1)–(5) at the  $k$ th time instant that corresponds to the initial condition  $x(0) = x_0^\gamma$  and the chosen vector  $K$ ; when applied to vectors, the symbol  $\leq$  indicates their component-wise inequality. The vectors  $\varepsilon_k^{\gamma-}$  and  $\varepsilon_k^{\gamma+}$  can be specified by calculating their coordinates  $\varepsilon_{ki}^{\gamma-}$ ,  $\varepsilon_{ki}^{\gamma+}$ , e.g., using the formulas  $\varepsilon_{ki}^{\gamma-} = -\delta_i m_i^\gamma$  and  $\varepsilon_{ki}^{\gamma+} = +\delta_i m_i^\gamma$ , where  $\delta_i$  is an admissible value of the absolute relative error  $(y(x_0^\gamma, K)_{ki} - y_{ki}^\gamma)/m_i^\gamma$  of the reproducing the  $i$ th coordinate of the desired trajectory in the system;  $m_i^\gamma$  is the maximum absolute value of the  $i$ th coordinate of the desired trajectory  $y^\gamma$ . In a more general case, particular values of  $\delta_i$  for different time instants  $k\Delta t$  and trajectories  $y^\gamma$  can be used when calculating  $\varepsilon_{ki}^{\gamma-}$  and  $\varepsilon_{ki}^{\gamma+}$ . It is reasonable to assign  $\delta_i \in [1, 20]10^{-2}$ .

The vector  $K$  will be chosen appropriately for making the behavior of system (1)–(5) maximally close to the desired one in the sense of minimizing the Euclidean norm of the vector  $\Delta y(K)$

composed of the differences  $y(x_0^\gamma, K)_k - y_k^\gamma$ , i.e., from the condition

$$|\Delta y(K)| \rightarrow \min_K, \tag{8}$$

where  $|\cdot|$  denotes the Euclidean norm [22].

In the case of a given controller structure (a specified set  $S$ , defining together with (3)–(5) this structure), the problem is to find a vector  $K$  under which the control system (1)–(5) will fulfill the requirements (6)–(8) for the given values of  $\omega_0$  and  $\tilde{U}$ .

In the general case, when the set  $S$  is not specified and the output vector  $y$  contains several measurable linearly independent variables, we will solve the structural design problem, i.e., given  $\omega_0$  and  $\tilde{U}$ , determine the sets  $S$  and the corresponding vectors  $K$  for which conditions (6)–(8) are satisfied and the structure of the controller (3), (5) is simple. (According to [18, 19], simplicity means that only those gains  $k_i$  are non-zero that are necessary and sufficient to give system (1)–(5) the desired properties.) A structure  $S'$  is said to be simpler than another structure  $S$  if  $S' \subset S$ . Formally, the problem of obtaining the set  $\Omega$  of simple controller structures is to find admissible structures  $S \in \zeta$  for which no less complex admissible structure  $S' \in \zeta$  can be specified. In other words, it is required to find

$$\Omega = \{S \in \zeta \mid \{S' \in \zeta \mid S' \subset S\} = \emptyset\}, \tag{9}$$

where  $\zeta$  stands for the set of admissible structures, i.e., those for which there exists a vector  $K$  under which the control system (1)–(5) will fulfill the requirements (6)–(8). The formula  $\{S' \in \zeta \mid S' \subset S\} = \emptyset$  indicates the absence of an admissible structure  $S'$  that is simpler than a structure  $S \in \Omega$ .

### 3. PROBLEM ANALYSIS

The following conditions [5, 6] are necessary for system (1)–(5) to have a simple symmetric limit cycle with period  $2h = 2\pi/\omega_0$  :

$$x^* = -(I + e^{Ah})^{-1} \int_0^h e^{A(h-\tau)} B d\tau, \tag{10}$$

$$K^\top C x^* = 0, \tag{11}$$

$$K^\top C \left( e^{At} x^* + \int_0^t e^{A(t-\tau)} B d\tau \right) > 0, \quad 0 < t < h, \tag{12}$$

where  $x^*$  is the limit cycle point corresponding to the change of sign of  $\tilde{u}$  from minus to plus and, consequently, to the switching of the relay (4) from  $-1$  to  $+1$ , and  $I$  denotes an identity matrix of compatible dimensions.

A limit cycle is locally stable [5–7] if and only if the absolute values of all the eigenvalues of the matrix

$$W = (I - vK^\top C / (K^\top C v)) e^{Ah}, \tag{13}$$

$$v = -Ax^* + B, \tag{14}$$

are less than unity. This statement was proved in [6, 7] for systems of the form (1)–(5) using the classical approach to the stability analysis of periodic motions based on the Poincaré maps.

The stability margin requirement (6) for self-oscillations will be considered fulfilled if

$$\eta(K) = 1 - \rho(W(K)) \geq \tilde{\eta}, \tag{15}$$

where  $\rho(W(K))$  denotes the spectral radius of the matrix  $W(K)$ ; it is reasonable to assign  $\tilde{\eta} \in [1, 10]h/(N\Delta t)$ .

Processes in relay systems that evolve much more slowly compared to a self-oscillation process are conventionally called slow processes [1, 2, 5]. In the case where the signal range at the relay input slightly exceeds the amplitude  $\tilde{U}$ , such processes are described by an approximate model in which the relay (4) with a stable simple symmetric limit cycle is replaced by its linearized representation [1, 2, 5]

$$u = R\tilde{u}, \quad R = 4/(\pi\tilde{U}), \tag{16}$$

where  $R$  is the linearization coefficient. Equations (1)–(3), (5), and (16) will be used as a simplified model of slow processes in the self-oscillating system (1)–(5).

Assume that the set of training examples contains descriptions of processes that can be considered slow for given parameters  $\omega_0$  and  $\tilde{U}$ . Then the problem of training system (1)–(5) to the desired behavior can be replaced by that of training system (1)–(3), (5), and (16). To apply the training method [17], we replace system (1)–(3) and (16) with its discrete-time analog

$$x_{k+1} = \hat{A}x_k + \hat{B}u_k, \quad y_k = Cx_k, \quad u_k = RKy_k, \tag{17}$$

where  $k$  is discrete time (a natural number);  $t = k\Delta t$ ; the time discretization step  $\Delta t$  is chosen from the condition  $\Delta t \ll h$ ; the vectors  $x_k, y_k$ , and  $u_k$  are the discrete approximations of the vectors  $x, y$ , and  $u$ , respectively; finally,  $\hat{A} = e^{A\Delta t}$  and  $\hat{B} = \int_0^{\Delta t} e^{At} B dt$ .

According to [17], the conditions (7), (8) for making the behavior of system (17) maximally close to the desired one, defined by the set of training examples  $Q$ , are equivalent to

$$\sum_{\gamma=1}^q |G_\gamma(K)_S K_S - \hat{Y}_\gamma|^2 \rightarrow \min_K, \tag{18}$$

$$\hat{Y}_\gamma + \varepsilon_\gamma^- \leq G_\gamma(K)_S K_S \leq \hat{Y}_\gamma + \varepsilon_\gamma^+, \quad \gamma \in \{\overline{1, q}\}, \tag{19}$$

where  $\hat{Y}_\gamma = Y_\gamma - Y_{0\gamma}$ , with the columns  $Y_\gamma, Y_{0\gamma}$ , and  $G_\gamma(K)$  being composed of the blocks  $y_k^\gamma, C\hat{A}^k x_0^\gamma$ , and  $G_{k\gamma}(K) = C \sum_{i=0}^{k-1} (y(x_0^\gamma, K)_i^\top \otimes \hat{A}^{k-i-1} \hat{B}R)$ ,  $k \in \{\overline{1, N}\}$ , respectively;  $\otimes$  indicates the Kronecker product [23, p. 83]; finally, the matrix  $G_\gamma(K)_S$  and the vector  $K_S$  are composed of the columns of the matrix  $G_\gamma(K)$  and the coordinates of the vector  $K$ , respectively, with the numbers specified in the set  $S$ .

System (17) differs from the control system considered in [17] only by the factor  $R$  in the expression for  $u_k$ . This difference is taken into account in the above expression for  $G_{k\gamma}(K)$ .

In view of (5), condition (11) becomes the linear equation

$$(Cx^*)_S^\top K_S = 0, \tag{20}$$

where the vector  $(Cx^*)_S$  is composed of the coordinates of the vector  $Cx^*$  with the numbers specified in the set  $S$ .

Due to (5), the discrete analog of the requirement (12) is the system of linear inequalities

$$H_{kS} K_S > 0, \quad \forall k \in \{\overline{1, \hat{h}}\}, \tag{21}$$

where the matrix  $H_{kS}$  is composed of the columns of the matrix

$$H_k = \left( C e^{Ak\Delta t} x^* + C \int_0^{k\Delta t} e^{A(k\Delta t-\tau)} B d\tau \right)^\top$$

with the numbers specified in  $S$  and  $\hat{h} = \text{fix}(h/\Delta t - 1)$ , where  $\text{fix}(\cdot)$  means rounding to the nearest integer.

For the self-oscillation mode, the left-hand side of inequality (12) determines the values of the signal  $\tilde{u}$  within the half-wave of its oscillations, so the amplitude  $\tilde{U}$  of the signal  $\tilde{u}$  is close to the value of  $H_{kS}K_S$  when  $k$  lies at the middle of the closed interval  $[1, \hat{h}]$ . To realize the specified amplitude  $\tilde{U}$  of the signal  $\tilde{u}$  in the mode of simple symmetric self-oscillations, we impose the condition

$$H_{kS}K_S = \tilde{U}, \quad k = \text{fix}((\hat{h} + 1)/2). \tag{22}$$

Thus, the above problem of finding the vector  $K$  under which the control system (1)–(5) will fulfill the requirements (6)–(8) for the given values of  $\omega_0$  and  $\tilde{U}$  is reduced to problem (18)–(22), (15). Here, the existence conditions (20)–(22) of self-oscillations with given parameters and their stability condition (15) have been obtained from the nonlinear model of the relay system (1)–(5) without simplifying assumptions: only the conditions (18), (19) for making its behavior maximally close to the desired one are based on its simplified linearized model (17).

#### 4. SOLUTION METHOD

Let the controller structure be given, i.e., the set  $S$  is specified. As shown above, the desired vector  $K$  can be determined by solving problem (18)–(22) and (15), which is equivalent to the problem of training a static controller considered in [17]. For this problem, we apply the method proposed in [17]: at each iteration, it is necessary to solve a constrained linear least-squares (CLLS) problem [24, p. 225], belonging to the class of convex programming problems [25]. For such problems, effective optimization procedures have been developed that guarantee a solution or claim its absence. (In MATLAB, the lsqin function is intended for solving CLLS problems.) At each iteration of the method [17], when solving the original problem, the transition to the CLLS problem is based on replacing the matrices  $G_\gamma(K)_S$  in (18) and (19) with the matrix  $G_\gamma(\hat{K})_S$  (fixed during this iteration) that corresponds to the vector  $\hat{K}$  found at the previous iteration, as well as on replacing the function  $\rho(W(K))$  in (15) with its linear approximation  $r_0 + r_1\hat{K}_S$  in the neighborhood of  $\hat{K}$ , i.e., on replacing condition (15) with the linear inequality

$$r_0 + r_1\hat{K}_S \leq 1 - \tilde{\eta}, \tag{23}$$

where  $r_0$  and  $r_1$  are linearization coefficients.

To determine an initial approximation for the solution of the problem under consideration, we separate its important special case with conditions (18), (20), and (22) only. Here, we simplify condition (18) by assuming that all desired trajectories in the set  $Q$  belong to the solution set of the system designed. Then [17] the matrix  $G_\gamma(K)_S$  in (18) can be replaced by the matrix  $\bar{G}_{\gamma S}$ , independent of  $K$ , and condition (18) can be therefore written as

$$\sum_{\gamma=1}^q |\bar{G}_{\gamma S}K_S - \hat{Y}_\gamma|^2 \rightarrow \min_K, \tag{24}$$

where the matrix  $\bar{G}_{\gamma S}$  consists of the columns of the matrix  $\bar{G}_\gamma$  specified in  $S$ , and the latter matrix is the column of the blocks  $\bar{G}_{k\gamma} = C \sum_{i=0}^{k-1} (y_i^{\gamma\top} \otimes \hat{A}^{k-i-1} \hat{B} R)$ ,  $k \in \{1, \overline{N}\}$ .

As is well known [24], a quadratic programming problem with linear equality constraints (and problem (20), (22), and (24) as its particular case) is reduced to a system of linear equations. In the case under study, this system takes the form

$$\begin{pmatrix} F & L^\top \\ L & 0 \end{pmatrix} \begin{pmatrix} K_S \\ \lambda \end{pmatrix} = \begin{pmatrix} c \\ d \end{pmatrix}, \tag{25}$$

where  $F = \bar{G}_S^\top \bar{G}_S$ ,  $\bar{G}_S$  is the matrix of the system of equations  $\bar{G}_{\gamma S} K_S = \hat{Y}_\gamma, \forall \gamma \in \{\overline{1, q}\}$ ,  $L = ((Cx^*)^\top, H_{kS})^\top, c = \bar{G}_S^\top \hat{Y}_\gamma, d = (0, \tilde{U})^\top$ , and  $\lambda$  is the vector of Lagrange multipliers.

The vector  $K_S$  obtained by solving system (25) can serve as an initial approximation for the solution of problem (18), (22), and (15).

Suppose that with the initial approximation given by (25), problem (18)–(22), (15) has not been solved successfully. Then it is possible to use the initial approximation calculated under the requirements described by the system of equalities (20), (22), and (24) and inequalities (19), (21), and (23). In this case, replacing (19) with its approximate analog, i.e., the condition

$$\hat{Y}_\gamma + \varepsilon_\gamma^- \leq \bar{G}_{\gamma S} K_S \leq \hat{Y}_\gamma + \varepsilon_\gamma^+, \quad \gamma \in \{\overline{1, q}\}, \tag{26}$$

one finds the initial approximation by solving the CLLS problem (20)–(24) and (26).

The above initial approximation is refined at subsequent iterations of the method without assuming that the trajectories constituting the set  $Q$  belong to the solution set of the system designed.

The set  $\Omega$  of simple controller structures can be found by a method intended for designing simple structures of a general form [19]. In this method, the above procedure for solving problem (18)–(22) and (15) can be used to assess the admissibility of the controller structure and determine the corresponding vector  $K$ .

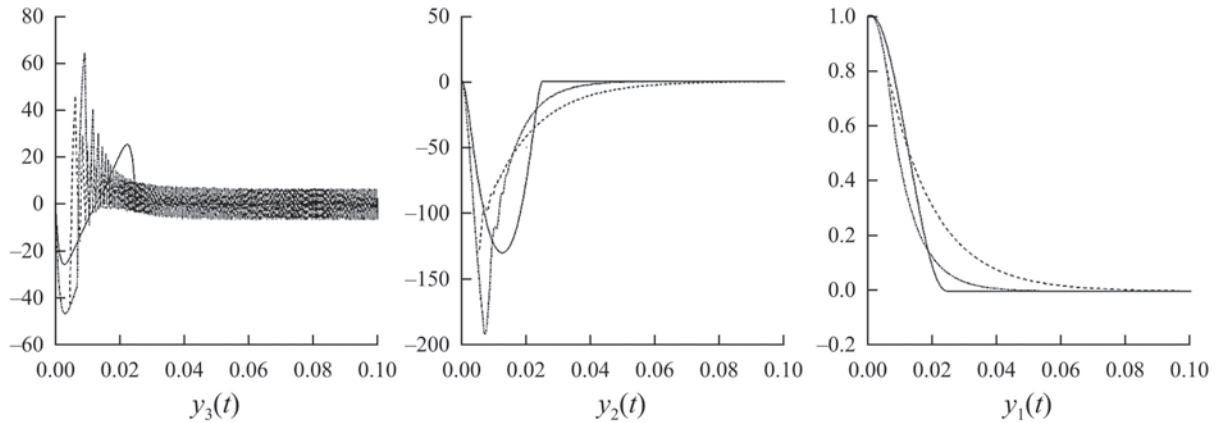
### 5. EXAMPLES

*Example 1.* Consider the problem of designing a relay controller for a self-oscillating control system of an electric drive. The plant consists of a DC motor with a constant excitation flux, a gearbox, and an inertial load. This plant is described by equations (1)–(2) with

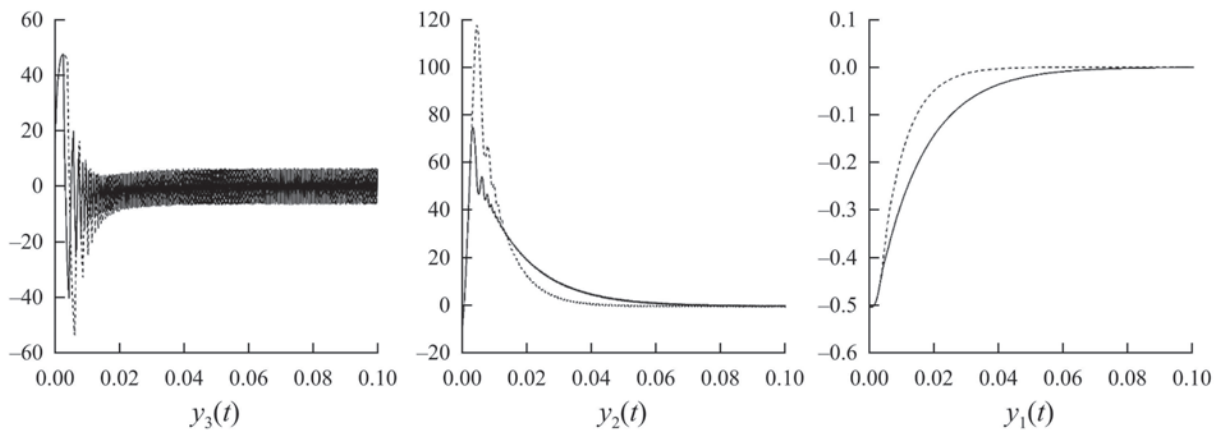
$$A = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & C_m/J_m \\ 0 & -C_e/L_e & -R_e/L_e \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 0 \\ U/L_e \end{pmatrix}, \quad C = \begin{pmatrix} 1/k_r & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

where  $R_e$  and  $L_e$  are the resistance and inductance of the motor armature winding;  $C_e$  and  $C_m$  are the back electromotive force (EMF) coefficient and the torque coefficient of the motor;  $U$  is the supply voltage;  $k_r$  is the gear ratio; finally,  $J_m$  is the moment of inertia of the moving parts reduced to the motor shaft. Let  $R_e = 0.475, L_e = 5.7 \times 10^{-4}, C_e = C_m = 6.83 \times 10^{-2}, U = 27, k_r = 2,$  and  $J_m = 9.43 \times 10^{-5}$  in the international system of units. Direct verification shows that the pairs  $(A, B)$  and  $(A, C)$  are controllable and observable, respectively. The state vector has the form  $x = (x_1, x_2, x_3)$ , where  $x_1$  and  $x_2$  are the angular position and velocity of the motor shaft, respectively, and  $x_3$  is the current in the armature winding. The output has the form  $y = (y_1, y_2, y_3) = (x_1/k_r, x_2, x_3)$ , where  $y_1$  is the controlled variable (the angular position of the output shaft of the electric drive), and  $S = \{1, 2, 3\}$ . Let us assign  $\tilde{U} = 0.01$  V and suppose that the amplitude of self-oscillations of the controlled variable  $y_1$  should not exceed 0.1 mrad. To fulfill this requirement, according to the analysis of the plant’s frequency response, the self-oscillation frequency must be at least 1820 Hz, so we set  $f = 2000$  Hz, with  $\omega_0 = 2\pi f = 12\,566$  rad/s.

It is required to find the gains constituting the desired vector  $K$ .



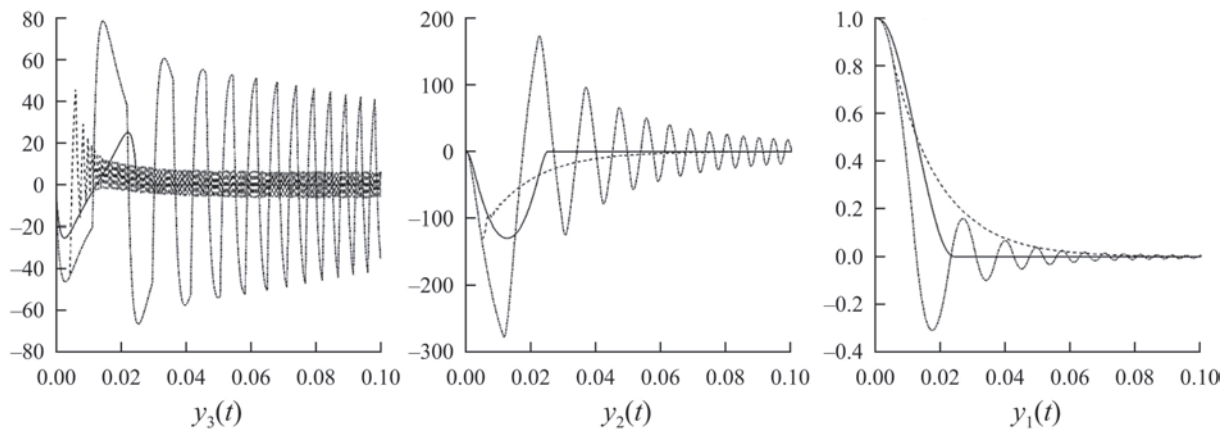
**Fig. 1.** System transients for  $x(0) = x_0^1$ . (The desired process and processes obtained in Examples 1 and 2 are indicated by solid, dotted, and dash-dotted lines, respectively.)



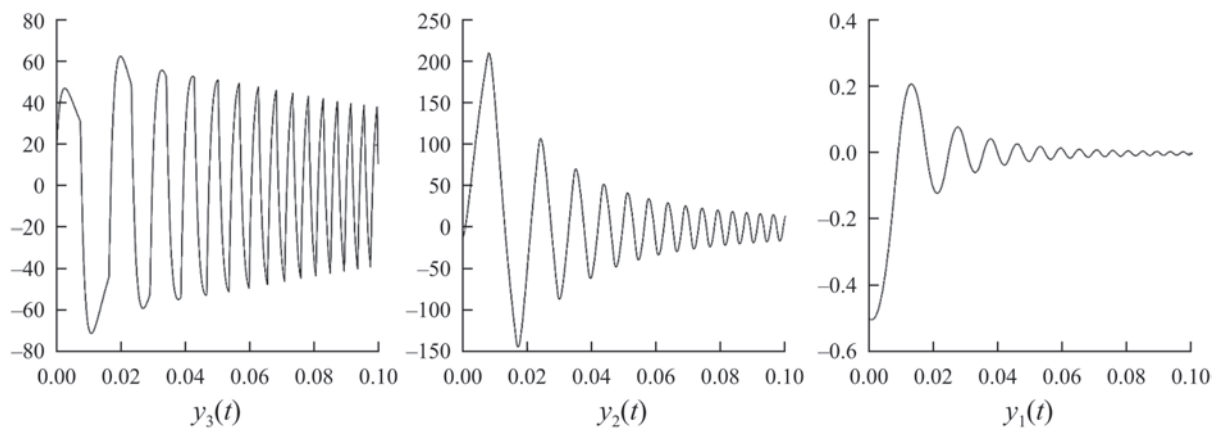
**Fig. 2.** System transients for  $x(0) = x_0^3$ . (The desired process and processes obtained in Examples 1 and 2 are indicated by solid and dotted lines, respectively.)

We form a set of training examples defining the desired behavior of the system designed during its transition from a disturbed to a steady-state motion mode. Following the above recommendation  $q \in [1, n]$  (see Section 2),  $q = 2$  training examples are taken. The desired system behavior is the trajectories of the plant (1) and (2) from the significantly different initial states  $x_0^1 = (1; 0; 0)$  and  $x_0^2 = (0; 10; -10)$  to the origin in a time of 0.025 s. These trajectories can be calculated using known dependencies [22, p. 128]. The specified trajectories  $Y_1$  and  $Y_2$ , together with the corresponding initial conditions on the time interval from 0 to 0.05 s, constitute the set of training examples  $Q = \{(x_0^1, Y_1), (x_0^2, Y_2)\}$ . When calculating the trajectories  $Y_1$  and  $Y_2$ , the time discretization step  $\Delta t = 2.5 \times 10^{-5} \ll h = 2.5 \times 10^{-4}$  is chosen according to the recommendation of Section 3; the corresponding value is  $N = 2000$ . The problem of calculating these trajectories has a solution due to the controllability and observability of the plant.

First, we solve the design problem without the constraints (6), (7), which is equivalent to problem (20), (22), and (24). Let the initial approximation be the solution of system (25). Three iterations of the method presented in subsection 4.1 yielded  $K = (-5.11; -3.73 \times 10^{-2}; 1.06 \times 10^{-4})$ , with the stability margin  $\eta(K) = 0.017$  and  $2.1 \times 10^{-6}$  as the corresponding value of the objective function (18). The resulting values of  $\tilde{U}$  and  $\omega_0$  were 0.010 V and 12 566 rad/s, respectively. The transients in system (1)–(5), corresponding to the found vector  $K$ , are presented in Figs. 1 and 2 for the initial states  $x(0) = x_0^1$  and  $x(0) = x_0^3 = (-0.5; -10; +10)$ , respectively. Figure 2 demonstrates the system behavior corresponding to the initial state  $x_0^3$ , which significantly differs from the initial states  $x_0^1$  and  $x_0^2$  specified in this example.



**Fig. 3.** System transients for  $x(0) = x_0^1$ . (The desired process and processes obtained in Examples 1 and 4 are indicated by solid, dotted, and dash-dotted lines, respectively.)



**Fig. 4.** System transients for  $x(0) = x_0^3$ , obtained in Example 4.

*Example 2.* Next, we solve the design problem of Example 1 with the following modifications: increase the stability margin  $\eta(K)$  to at least  $\tilde{\eta} = 0.030$  and reduce the admissible deviation of the output coordinates  $y_1$  and  $y_2$  from their desired values, assigning for them  $\delta_1 = 0.05$  and  $\delta_2 = 0.15$  on the time interval from 0.025 to 0.05 s and  $\delta_1 = 0.1$  and  $\delta_2 = 0.2$  on the time interval from 0 to 0.025 s. The vectors  $\varepsilon_k^{\gamma^-}$  and  $\varepsilon_k^{\gamma^+}$  corresponding to the above values of  $\delta_1$  and  $\delta_2$  are calculated according to the explanations to (7). The value  $\tilde{\eta} = 0.030$  is chosen according to the explanations to (15) by the formula  $\tilde{\eta} = 6h/(N\Delta t)$ , where  $h = 1/(2f) = 1/4000$  s and  $N\Delta t = 0.05$  s.

Under the specified requirements, we solve problem (18)–(22), and (15), with the solution of the CLLS problem (20)–(24), and (26) taken as the initial approximation.

Three iterations of the method from subsection 4.1 yielded

$$K = (-9.81; -3.76 \times 10^{-2}; 9.75 \times 10^{-5}).$$

In this case,  $\eta(K) = 0.0323$ , conditions (6) and (7) hold, and the value of the objective function (18) is  $4.1 \times 10^{-6}$ .

The transients corresponding to the found vector  $K$  are presented in Figs. 1 and 2 for the initial states  $x(0) = x_0^1$  and  $x(0) = x_0^3$ , respectively.

*Example 3.* Let us change the problem statement of Example 1, i.e., solve the structural design problem: given  $\omega_0$  and  $\tilde{U}$ , it is required to find the sets  $S$  and the corresponding vectors  $K$  under which the controller structure will be simple according to (9). The desired set  $\Omega$  can be obtained

by the design method of simple structures presented in [19, p. 19] for “general-form problems.” At the first step, it is necessary to check the admissibility of all structures  $S \subseteq \{1, 2, 3\}$ . Recall that for an admissible structure  $S$ , it is possible to find a vector  $K$  under which the control system (1)–(5) will fulfill the requirements (6)–(8). (See the explanations to (9).)

We check the existence of such a vector  $K$  using the solution procedure of problem (18)–(22), and (15), described in Section 4. According to the checking results, the structure  $S = \{1, 2, 3\}$  is admissible, whereas all structures  $S \subseteq \{1, 2, 3\}$  corresponding to two-component sets (i.e.,  $\{1, 2\}$ ,  $\{1, 3\}$ , and  $\{2, 3\}$ ) are inadmissible. As a consequence, the sets  $\{1\}$ ,  $\{2\}$ ,  $\{3\}$ , representing the subsets of the inadmissible sets  $\{1, 2\}$ ,  $\{1, 3\}$ , and  $\{2, 3\}$ , are surely inadmissible and do not require checking [19, Proposition 8]. Thus, in this example, the structure  $S = \{1, 2, 3\}$  is admissible and simple.

*Example 4.* Now we solve the problem of Example 3, with the lower limit of the stability margin reduced to  $\tilde{\eta} = 2.5 \times 10^{-3}$ . The resulting set  $\Omega$  contains the single set  $S = \{1, 2\}$ , which defines the controller structure using feedback signals for the angular position  $y_1$  and velocity  $y_2$  of the output shaft of the drive. In this case, the structure  $S = \{1, 2, 3\}$  is redundant, and the other structures are inadmissible. The found simple structure  $S = \{1, 2\}$  is associated with  $K = (-61.45; -3.73 \times 10^{-2}; 0)$ . The transients in system (1)–(5) corresponding to the found vector  $K$  are presented in Figs. 3 and 4 for the initial states  $x(0) = x_0^1$  and  $x(0) = x_0^3$ , respectively.

## 6. CONCLUSIONS

In this paper, we have posed and solved the problem of designing relay controllers for a self-oscillating system with a linear plant. The novelty of this problem statement lies in the joint fulfillment of existence conditions for asymptotically stable self-oscillations with given parameters in the system, controller’s structural constraints, and requirements for making the system behavior maximally close to a desired one. The desired system behavior has been specified by a set of time-varying system output laws (training examples). A method for solving the problem has been proposed, which further develops and extends the algorithm outlined in [17] to the class of relay self-oscillating control systems. The mathematical model of the controller uses the function of an ideal relay. The impact of the differences between the characteristics of real and ideal relays on the system behavior can be assessed via modeling by replacing the function of an ideal relay in (1)–(5) with a more complete and detailed mathematical description of a real relay.

## REFERENCES

1. Tsytkin, Ya.Z., *Relay Control Systems*, Cambridge: Cambridge University Press, 1985.
2. Popov, E.P., *Teoriya nelineinykh sistem avtomaticheskogo regulirovaniya i upravleniya* (Theory of Nonlinear Automatic Control and Regulation Systems), Moscow: Nauka, 1974.
3. Atherton, D.P., *Nonlinear Control Engineering – Describing Function Analysis and Design*, London: Van Nostrand Company Limited, 1975.
4. Faldin, N.V. and Rudnev, S.A., Synthesis of Relay Systems by the Phase Plot Method, *Izv. Vuzov. Priborostroenie*, 1982, no. 7, pp. 32–36.
5. Faldin, N.V., *Releinye sistemy avtomaticheskogo upravleniya. Matematicheskie modeli, dinamicheskie kharakteristiki i analiz sistem avtomaticheskogo upravleniya* (Relay Automatic Control Systems. Mathematical Models, Dynamic Characteristics and Analysis of Automatic Control Systems), Pupkov, K.A. and Egupov, N.D., Eds., Moscow: Bauman Moscow State Technical University, 2004.
6. Astrom, K.J., Oscillations in Systems with Relay Feedback, in *Adaptive Control, Filtering and Signal Processing*, Astrom, K.J., Goodwin, G.C., and Kumar, P.R., Eds., New York: Springer-Verlag, 1995.

7. Astrom, K.J. and Hagglund, T., Automatic Tuning of Simple Regulators, *Proceedings of the 9th IFAC World Congress*, Budapest, Hungary, 1984, pp. 267–272.
8. Boiko, I., *Discontinuous Control Systems*, Boston: Birkhauser, 2009.
9. Mozzhechkov, V.A., Synthesis of Simple Relay Controllers in Self-Oscillating Control Systems, *Autom. Remote Control*, 2022, vol. 83, no. 9, pp. 1393–1403.
10. Varigonda, S. and Georgiou, T.T., Dynamics of Relay Relaxation Oscillators, *IEEE Trans. Autom. Control*, 2001, vol. 46, no. 1, pp. 65–77.
11. Han, T., Wang, G., and Dong, C., A Self-Oscillating Driving Circuit for Low-Q MEMS Vibratory Gyroscopes, *Micromachines*, 2023, vol. 14, no. 5, pp. 1057–1072.
12. Maletin, A.N. and Khatanzeyskaya, M.A., Research of Filtering Properties of a Pendulum Accelerometer Functioning in the Auto-Oscillation Mode, *Aerospace Instrument-Making*, 2021, no. 10, pp. 3–12.
13. Indeitsev, D.A., Loboda, O.S., and Morozov, N.F., Self-Oscillation Mode of a Nanoresonator, *Fizicheskaya Mezomekhanika*, 2016, no. 5, pp. 23–28.
14. Urasaki, S. and Yabuno, H., Amplitude Control for Sensorless Self-Excited Oscillation of Cantilever Based on a Piezoelectric Device, *Nonlinear Dynamics*, 2022, vol. 108, pp. 15–25.
15. Krasovskii, A.A., Synthesis of Self-Oscillating Systems with Application to a New-Class Wind Power Plant, *Journal of Computer and Systems Sciences International*, 1996, vol. 34, no. 2, pp. 20–28.
16. Aguilar, L.T., Boiko, I., Fridman, L., and Iriarte, R., Generating Self-Excited Oscillations for Underactuated Mechanical Systems via Two-Relay Controller, *International Journal of Control*, 2009, vol. 82, no. 9, pp. 1678–1691.
17. Mozzhechkov, V.A., Static Feedback Design in Linear Discrete-Time Control Systems Based on Training Examples, *Autom. Remote Control*, 2023, vol. 84, no. 9, pp. 1065–1074.
18. Mozzhechkov, V.A., Design of Simple-Structure Linear Controllers, *Autom. Remote Control*, 2003, vol. 64, no. 1, pp. 23–36.
19. Mozzhechkov, V.A., Simple Structures in Control Problems: Formalization and Design, *Journal of Computer and Systems Sciences International*, 2022, vol. 61, no. 3, pp. 309–325.
20. Syrmos, V.L., Abdallah, C.T., Dorato, P., and Grigoriadis, K., Static Output Feedback – a Survey, *Automatica*, 1997, vol. 33, no. 2, pp. 125–137.
21. Filippov, A.F., *Differential Equations with Discontinuous Righthand Sides*, Dordrecht: Kluwer Academic Publishers, 1988.
22. Polyak, B.T., Khlebnikov, M.V., and Rapoport, L.B., *Matematicheskaya teoriya avtomaticheskogo upravleniya* (Mathematical Theory of Automatic Control), Moscow: Lenand, 2019.
23. Ikramov, Kh.D., *Chislennoe reshenie matrichnykh uravnenii* (Numerical Solution of Matrix Equations), Moscow: Nauka, 1984.
24. Gill, P.E., Murray, W.W., and Wright, M., *Practical Optimization*, Academic Press, 1981.
25. Polyak, B.T., *Introduction to Optimization*, New York: Optimization Software, 1987.

*This paper was recommended for publication by S.A. Krasnova, a member of the Editorial Board*

# Model for Servicing Multiservice Traffic in an Access Node of a Satellite Communications Network with a Dynamically Variable Service Delivery Rate

A. A. Maslov<sup>\*,a</sup>, G. V. Sebekin<sup>\*,b</sup>, M. S. Stepanov<sup>\*\*,c</sup>,  
S. N. Stepanov<sup>\*\*,d</sup>, and A. O. Shchurkov<sup>\*,e</sup>

<sup>\*</sup>Moscow Institute of Physics and Technology (National Research University), Moscow, Russia

<sup>\*\*</sup>Moscow Technical University of Communications and Informatics, Moscow, Russia

e-mail: <sup>a</sup>maslov.aa@mipt.ru, <sup>b</sup>sebekin.gv@mipt.ru, <sup>c</sup>m.s.stepanov@mtuci.ru,

<sup>d</sup>s.n.stepanov@mtuci.ru, <sup>e</sup>shchurkov.ao@mipt.ru

Received July 18, 2025

Revised September 19, 2025

Accepted September 22, 2025

**Abstract**—In multi-service satellite communication networks, as a rule, it is possible to provide a particular service with different quality, while using different traffic transfer rates. Accordingly, license agreements may stipulate that a particular service is provided at a certain speed for the majority of the time, and it is acceptable to reduce the speed to the maximum threshold in the remaining time. At the same time, network operators need a mathematical apparatus that allows them to evaluate the fulfillment of the requirements specified in the agreements in order to have an idea to what extent it is possible to expand the subscriber capacity of the network. The article develops a mathematical model of joint maintenance in access nodes of such networks of real-time service traffic and elastic data traffic based on the formalization of the network operation process using the apparatus of multidimensional stepwise Markov processes. Examples of solving the problems of determining the required resource at the network planning stage and evaluating the possibility of expanding the subscriber capacity of the network with the available resource are given.

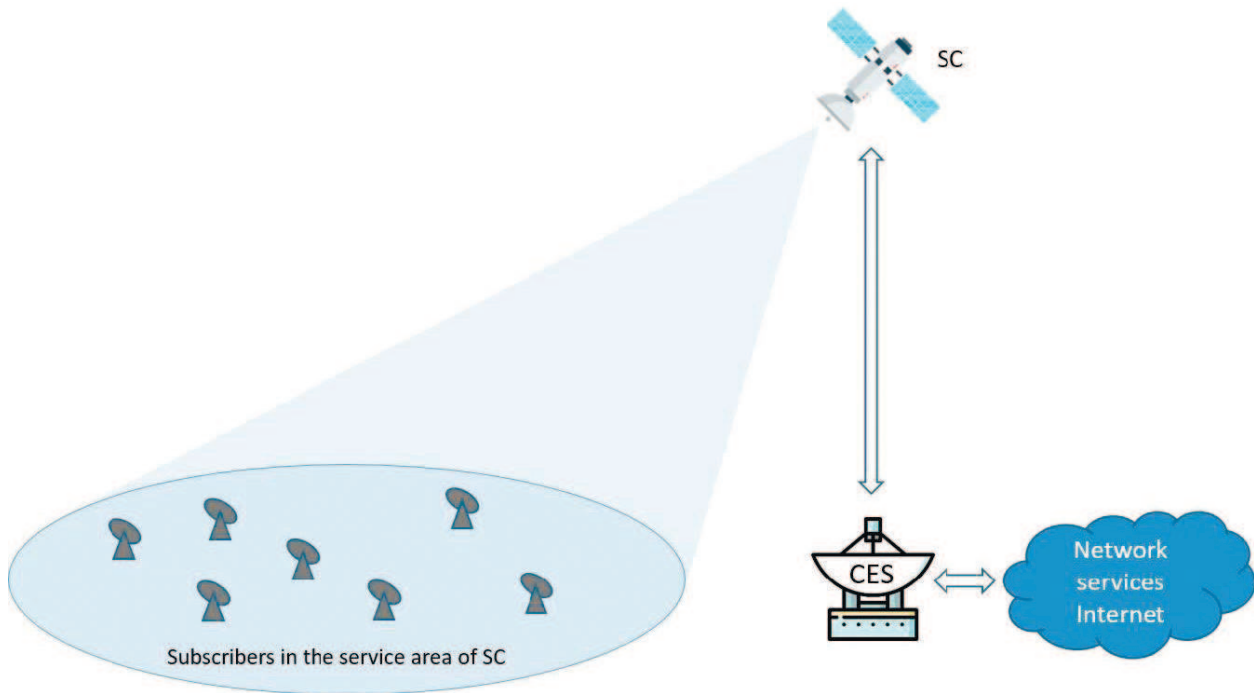
*Keywords:* spacecraft, channel resource, multi-service traffic, real-time traffic, elastic traffic, multidimensional stepwise Markov processes

**DOI:** 10.7868/S1608303225110041

## 1. INTRODUCTION

In multi-service satellite communication networks (Fig. 1), where the role of subscriber access nodes to terrestrial network and Internet services is played by central earth stations (CES) together with spacecraft (SC), there is typically the possibility of providing one or another service at different transmission rates. The higher the speed, the better the quality, for example, clearer video images, etc.

Since the bandwidth of satellite channels is limited, when there is a large number of simultaneously served subscribers, using the highest possible speeds becomes impossible, and for some subscribers, a reduction in speed is required. Accordingly, licensing agreements (Service Level Agreements, or SLA) may stipulate that a specific service is to be provided at high speed for the majority of the time, while a reduction to a minimum threshold is permitted for the remaining time. In such cases, network operators require a mathematical framework to evaluate compliance with the specified SLA requirements. This framework is essential for determining to what extent the subscriber capacity of the network can be expanded under the given constraints, or when further



**Fig. 1.** Simplified diagram of a multi-service satellite communication network.

expansion becomes impossible, necessitating network upgrades, adjustments to standards, changes in tariff policies, and so forth.

This article addresses this task by considering an access node of a multi-service network. The quality of service (QoS) in this network is assessed by the following indicators:

- The high transmission rates for each service's traffic, ensuring the best quality.
- The fractions of time during which the service traffic is transmitted at high rates.
- The minimum allowable transmission rates for each service's traffic for the remaining time.
- The fractions of lost requests of each type due to insufficient resources.
- The average file delivery time.

It is assumed that the network handles  $K$  types of real-time (RT) service traffic and elastic data service traffic. The resource for servicing requests is allocated in discrete units of speed quanta.

The quantum,  $\Delta v$ , is measured in bits/s. The network infrastructure within the Central Earth Station (CES) includes a connection management system. This system implements rules that uniquely determine how resources are assigned and redistributed upon the arrival and completion of service requests, taking into account the current network state. A service request is accepted only if the available resource is sufficient to service all current requests, at least at their minimum speeds. Any remaining resource is then used to assign high service speeds to some or all of the requests, based on the importance of the corresponding service types and the volume of the remaining resource. Finally, any leftover resource, if present, is allocated to increase the speed of elastic traffic. Consequently, for elastic traffic, the high and minimum speeds specified in the SLA are effectively guaranteed and can be exceeded. Furthermore, while the session duration for RT traffic is independent of its transmission rate, the file delivery time decreases as the rate increases.

The purpose of this article is to develop a mathematical model for a multi-service satellite network access node. This model accounts for the ability to change the transmission rates of individual services and enables the evaluation of the aforementioned quality of service indicators for each traffic type.

Modeling of multi-service networks is usually carried out under the assumption that the flows of requests in the network are Poisson, and the service durations are exponentially distributed. This allows building analytical models using the apparatus of multidimensional stepwise Markov processes. A wide range of works is devoted to the development of models for multi-service networks. Fundamental results are presented, for example, in [1–5]. A large number of results have been obtained for networks of various purposes [6–23], including multi-service satellite networks [16–23], including networks based on spacecraft in geostationary and highly elliptical orbits [16–19], [22], as well as in low circular orbits [20, 21]. Although the factor of transmission rate change in most works was taken into account only for elastic traffic, the possibilities of adjusting the rate in a certain range for real-time traffic were also considered, for example in [15]. However, different quality indicators from those mentioned above were taken into account. So in [15] the goal is to analyze the average data transmission rate of video conferences. Therefore, the task of modeling a network access node is relevant.

To achieve this goal, Section 2 describes the model, Section 3 formulates and solves the system of equilibrium equations of the Markov process describing the dynamics of network state changes, and obtains relations for calculating quality indicators. Section 4 provides a numerical analysis of the model characteristics.

## 2. MODEL DESCRIPTION

In the considered network, transmission of real-time service traffic (voice information, video conferencing data, etc.), depending on the service type, can be performed at high speeds  $V_k$ ,  $k = 1, \dots, K$ , provided in the SLA, and at certain time intervals with a large number of active subscribers at minimally allowable speeds  $V'_k$ . File transmission is carried out within elastic traffic and can be performed during the main part of the time at a speed not lower than the guaranteed high speed  $V_e$  and the rest of the time at a speed not lower than the minimally allowable  $V'_e$ .

Let  $b_k = \lceil V_k / \Delta v \rceil$ ,  $k = 1, \dots, K$  and  $b_e = \lceil V_e / \Delta v \rceil$  be the resources, expressed as an integer number of used speed quanta, required for transmitting RT and elastic traffic at high speeds  $V_k$  or  $V_e$ . Note that the resource for data transmission by one subscriber can be more than  $b_e$ . Let  $b'_k = \lceil V'_k / \Delta v \rceil$ ,  $k = 1, \dots, K$  and  $b'_e = \lceil V'_e / \Delta v \rceil$  be the resources corresponding to the minimum speeds  $V'_k$  or  $V'_e$ . Here also the resource for data transmission by one subscriber can be more than  $b'_e$ .

Service RT with number  $k = 1, \dots, K$  is received by  $N_k$  subscribers. The service time of a request for service is an exponentially distributed random variable with parameter  $\mu_k$ . After the service of a request is completed, a new request from this subscriber arises after a random time, which has an exponential distribution with parameter  $\beta_k$ . The corresponding parameters for elastic traffic are denoted by  $N_e$  and  $\beta_e$ , and  $\mu_e$  is the parameter of the exponential distribution of file transmission time when allocating resource  $b_e$ .

Let us introduce notations for quality indicators. Let  $P_k$ ,  $k = 1, \dots, K$ , and  $P_e$  be the fractions of time during which the traffic of each RT service is transmitted at high speed, and elastic data traffic at a speed not less than the guaranteed high speed. Let  $\pi_k$ ,  $k = 1, \dots, K$ , and  $\pi_e$  be the fractions of lost requests for service provision that were denied due to insufficient free resource upon their arrival,  $W$  be the average file delivery time, and  $v$  be the total network resource managed by the access node.

The dynamics of the network state change is described by the random process  $r(t) = (i_1(t), \dots, i_K(t), d(t))$ , where  $i_k(t)$  is the number of serviced requests for service  $k$  and  $d(t)$  is the number of serviced data requests at time  $t$ . The process is defined on the state space  $S$ , which includes states  $s = (i_1, \dots, i_K, d)$  with integer non-negative components, each of which does not

exceed the number of subscribers receiving the corresponding service. The space  $S$  can be written as

$$S = \left\{ (i_1, \dots, i_K, d) : N_k \geq i_k \geq 0, k = 1, \dots, K; N_e \geq d \geq 0; \sum_{k=1}^K i_k b'_k + db'_e \leq v \right\}. \quad (1)$$

In this space, one can distinguish subsets of states  $U_k, k = 1, \dots, K$ , and  $U_e$ , in which incoming requests for transmission of each type of real-time traffic and data traffic are denied due to insufficient resource for their service. These subsets are defined as follows:

$$U_k = \left\{ (i_1, \dots, i_K, d) : (i_1, \dots, i_K, d) \in S, \left( \sum_{k=1}^K i_k b'_k + db'_e > v - b'_k \right) \cup (i_k = N_k) \right\}, \quad (2)$$

$$U_e = \left\{ (i_1, \dots, i_K, d) : (i_1, \dots, i_K, d) \in S, \left( \sum_{k=1}^K i_k b'_k + db'_e > v - b'_e \right) \cup (d = N_e) \right\}. \quad (3)$$

To manage the network, the operator needs a rule  $f(s)$  that unambiguously determines for each network state  $s \in S$  the number of requests of each type  $i_k^{(h)}$  out of  $i_k, k = 1, \dots, K$ , and  $d^{(h)}$  out of  $d$ , that will be serviced at high speed.

It should be noted that different networks have different lists of provided services, and the importance of each service is determined by the operator taking into account the target tasks solved by the network. Therefore, the choice of the rule  $f(s)$  for resource allocation is qualitative and is not considered in the article. We will only assume that:

- interruption of request service is unacceptable;
- all requests of the same type have equal value;
- when applying the rule, the resource for traffic transmission at high speed is allocated for servicing requests in order of decreasing importance of their types, i.e., first for requests of the most important type, then for the next in importance, etc.

The rule  $f(s)$  is executed when the network state changes and includes three stages:

- at the first stage, it is determined whether the resource is sufficient to service all requests at least at minimum speeds;
- if the resource is sufficient, at the second stage, high speeds are allocated for servicing requests taking into account the amount of free resource and the importance of service types;
- if there are remnants of free resource, they are directed to increase the transmission speed of elastic traffic.

A high-level algorithm for applying the rule is illustrated in Fig. 2. Commenting on the algorithm, we note that if any requests remain being serviced at minimum rates after the second stage, it is impossible to increase their speed to the high rate. Furthermore, allocating additional resources to requests whose traffic is already being transmitted at high speed is not efficient. Therefore, any remaining resource from the first two stages is allocated in the third stage to increase the transmission speed of elastic traffic.

It should also be noted that when new requests for important traffic types arrive, resources are allocated to them as a priority. For less important traffic types, if resources are insufficient, their transmission speed may be downgraded from high to minimum. In other words, a redistribution of the resources used for servicing different traffic types will occur.

A possible example of such a rule is considered in Section 4 when conducting numerical analysis.

Let us introduce the vector  $s^{(h)} = (i_1^{(h)}, \dots, i_K^{(h)}, d^{(h)})$ . Then  $s^{(h)} = f(s)$ . Obviously, after assigning high service speeds to requests, the resource constraint must be satisfied, which is more

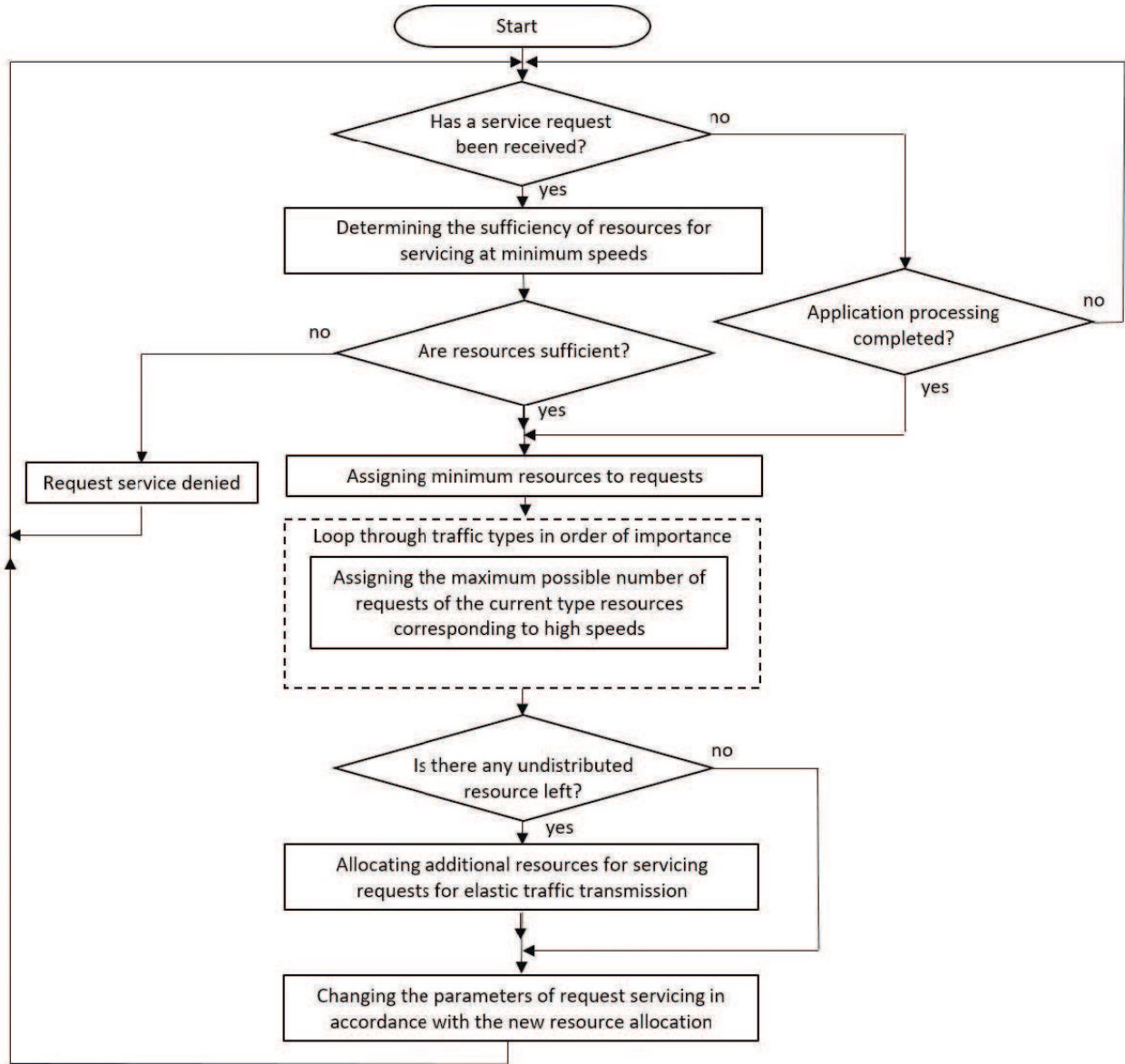


Fig. 2. High-level algorithm for applying the resource allocation rule.

stringent compared to the corresponding constraint in expression (1):

$$v_{\min} = \sum_{k=1}^K \left[ (i_k - i_k^{(h)}) b'_k + i_k^{(h)} b_k \right] + (d - d^{(h)}) b'_e + d^{(h)} b_e \leq v. \tag{4}$$

If the network is servicing data requests, i.e.,  $d > 0$ , and in expression (4)  $v_{\min} < v$ , then at the third stage of applying the rule  $f(s)$  the residual resource in the amount of  $v - v_{\min}$  is directed to service these requests. As a result, the total resource allocated for servicing data requests under the condition  $d > 0$  is

$$v_e(s, s^{(h)}) = v - v_{\min} + (d - d^{(h)}) b'_e + d^{(h)} b_e, \tag{5}$$

and the total service intensity of such requests is

$$\mu_{et}(s, s^{(h)}) = \frac{v_e(s, s^{(h)}) \mu_e}{b_e}. \tag{6}$$

When events occur that change the network state, for the new state  $s$ ,  $s^{(h)} = f(s)$  is computed and resource redistribution between requests is performed, and expression (6) determines the intensity  $\mu_{et}(s, s^{(h)})$  of leaving state  $s$  due to the event of file transmission completion.

3. SYSTEM OF EQUILIBRIUM EQUATIONS AND SERVICE PERFORMANCE MEASURES

Let  $p(s)$  denote the probability of the network being in state  $s \in S$ , and  $P(s)$  denote the unnormalized probability of that state. Unnormalized probabilities are used in iterative methods for solving systems of equilibrium equations (SEE). The connection between  $p(s)$  and  $P(s)$ , taking into account the normalization condition, is as follows:

$$p(s) = \frac{P(s)}{\sum_{s \in S} P(s)}. \tag{7}$$

The SEE has the form:

$$\begin{aligned} &P(i_1, \dots, i_K, d) \left[ \sum_{k=1}^K (\beta_k (N_k - i_k) I((i_1, \dots, i_K, d) \in S \setminus U_k) + i_k \mu_k) \right. \\ &\quad \left. + \beta_e (N_e - d) I((i_1, \dots, i_K, d) \in S \setminus U_e) + \mu_{et}(s, s^{(h)}) I(d > 0) \right] \\ &= \sum_{k=1}^K \left[ P(i_1, \dots, i_k - 1, \dots, i_K, d) \beta_k (N_k - i_k + 1) I(i_k > 0) \right. \\ &\quad \left. + P(i_1, \dots, i_k + 1, \dots, i_K, d) (i_k + 1) \mu_k I((i_1, \dots, i_K, d) \in S \setminus U_k) \right] \\ &\quad + I(d > 0) P(i_1, \dots, i_K, d - 1) \beta_e (N_e - d + 1) \\ &\quad + P(i_1, \dots, i_K, d + 1) \mu_{et}(s_d, s_d^{(h)}) I((i_1, \dots, i_K, d) \in S \setminus U_e), \quad (i_1, \dots, i_K, d) \in S. \end{aligned} \tag{8}$$

The system of equations (8) is homogeneous, and the uniqueness of the solution is ensured by adding the normalization condition (7) to it. Here  $s_d = (i_1, \dots, i_K, d + 1)$  and  $s_d^{(h)} = f(s_d)$ . The intensities  $\mu_{et}(s, s^{(h)})$  and  $\mu_{et}(s_d, s_d^{(h)})$  are computed according to relations (5) and (6). The indicator function  $I(\text{condition } A)$  is also used, equal to 1 if the condition is satisfied and 0 otherwise.

The system of equations (8) can be solved numerically. The Gauss-Seidel method, described in [1] and used in a number of works, for example in [16–20], has proven itself well for solving such SEEs. This method is also used to obtain numerical results in Section 4.

Next, we proceed to obtain relations for calculating the quality of service indicators, interpreting the state probabilities  $p(s)$ ,  $s \in S$ , as the fractions of time during which the network is in the corresponding states, i.e., over a sufficiently long time interval  $T$ , the network is in state  $s$  for approximately time  $Tp(s)$ . The total service time of  $i_k$ ,  $k = 1, \dots, K$ , requests for provision of the  $k$ th RT service in this state is  $i_k T p(s)$ . At the same time, the vector of the number of requests serviced at high speeds is determined as  $s^{(h)} = f(s)$ , and the total service time of  $i_k^{(h)} \leq i_k$  requests is  $i_k^{(h)} T p(s)$ . Consequently, the fractions of time during which the traffic of each service is transmitted at high speed are determined as

$$P_k = \frac{\sum_{s \in S} i_k^{(h)} p(s)}{\sum_{s \in S} i_k p(s)}, \quad k = 1, \dots, K. \tag{9}$$

Similarly for data requests:

$$P_e = \frac{\sum_{s \in S} d^{(h)} p(s)}{\sum_{s \in S} d p(s)}. \tag{10}$$

Since the input traffic in the considered network depends on its state  $s$ , the fraction of requests for transmission of each type of traffic lost due to lack of free channel resource should be estimated as the ratio of the intensity of lost requests of the corresponding flow to the intensity of arrived requests of this flow [1]. For requests for service of RT traffic of the  $k$ th type we obtain

$$\pi_k = \frac{\sum_{s \in U_k} (p(s) (N_k - i_k))}{\sum_{s \in S} (p(s) (N_k - i_k))}, \quad k = 1, \dots, K, \tag{11}$$

and for data transmission requests

$$\pi_e = \frac{\sum_{s \in U_e} (p(s) (N_e - d))}{\sum_{s \in S} (p(s) (N_e - d))}. \tag{12}$$

The average file delivery time  $W$  can be determined using Little's formula as the ratio of the average number of simultaneously serviced data requests in the network  $y_e$  to the intensity  $\lambda_e$  of the flow of such requests accepted for service, and is equal to

$$W = \frac{y_e}{\lambda_e}, \tag{13}$$

$$y_e = \sum_{s \in S} dp(s), \tag{14}$$

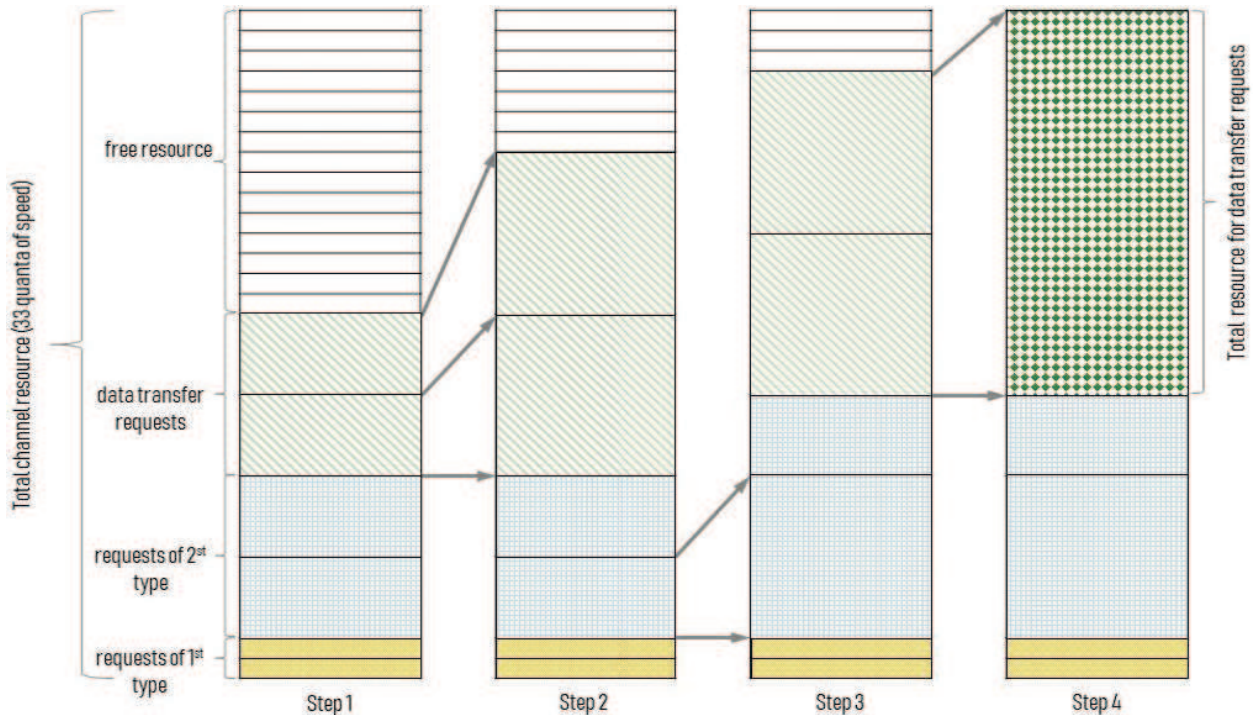
$$\lambda_e = \beta_e \sum_{s \in S \setminus U_e} p(s) (N_e - d). \tag{15}$$

#### 4. NUMERICAL ANALYSIS OF MODEL CHARACTERISTICS

As an example for numerical analysis, consider a three-service network of low-power mobile subscriber terminals based on high-throughput spacecraft [16, 17], serving two types of RT traffic and elastic data traffic. The traffic characteristics are presented in the table.

**Table.** System Parameters

Parameter	Value
Resource per request for RT service of the first type, $b_1/b'_1$	1/1
Resource per request for RT service of the second type, $b_2/b'_2$	8/4
Resource per data request, $b_e/b'_e$	8/4
Number of users of RT service of the first type, $N_1$	150
Number of users of RT service of the second type, $N_2$	60
Number of data service users, $N_e$	120
Service intensity of requests for RT service of the first type, $\mu_1$	0.3 min <sup>-1</sup>
Service intensity of requests for RT service of the second type, $\mu_2$	0.15 min <sup>-1</sup>
Service intensity of data requests when allocating resource $b_e$ , $\mu_e$	1.2 min <sup>-1</sup>
Parameter of exponential distribution of time until request arrival of the first type, $\beta_1$	0.06 min <sup>-1</sup>
Parameter of exponential distribution of time until request arrival of the second type, $\beta_2$	0.009 min <sup>-1</sup>
Parameter of exponential distribution of time until data request arrival, $\beta_e$	0.3 min <sup>-1</sup>
Fraction of time during which RT service traffic is transmitted at the specified high speed, $P_k$ , $k = 1, 2$ , not less than	0.9
Fraction of time during which data service traffic is transmitted at a speed not lower than the specified threshold, $P_e$ , not less than	0.9
Maximum allowable fractions of lost requests of each type due to insufficient resource	0.01
Average file delivery time, not more than	0.5 min



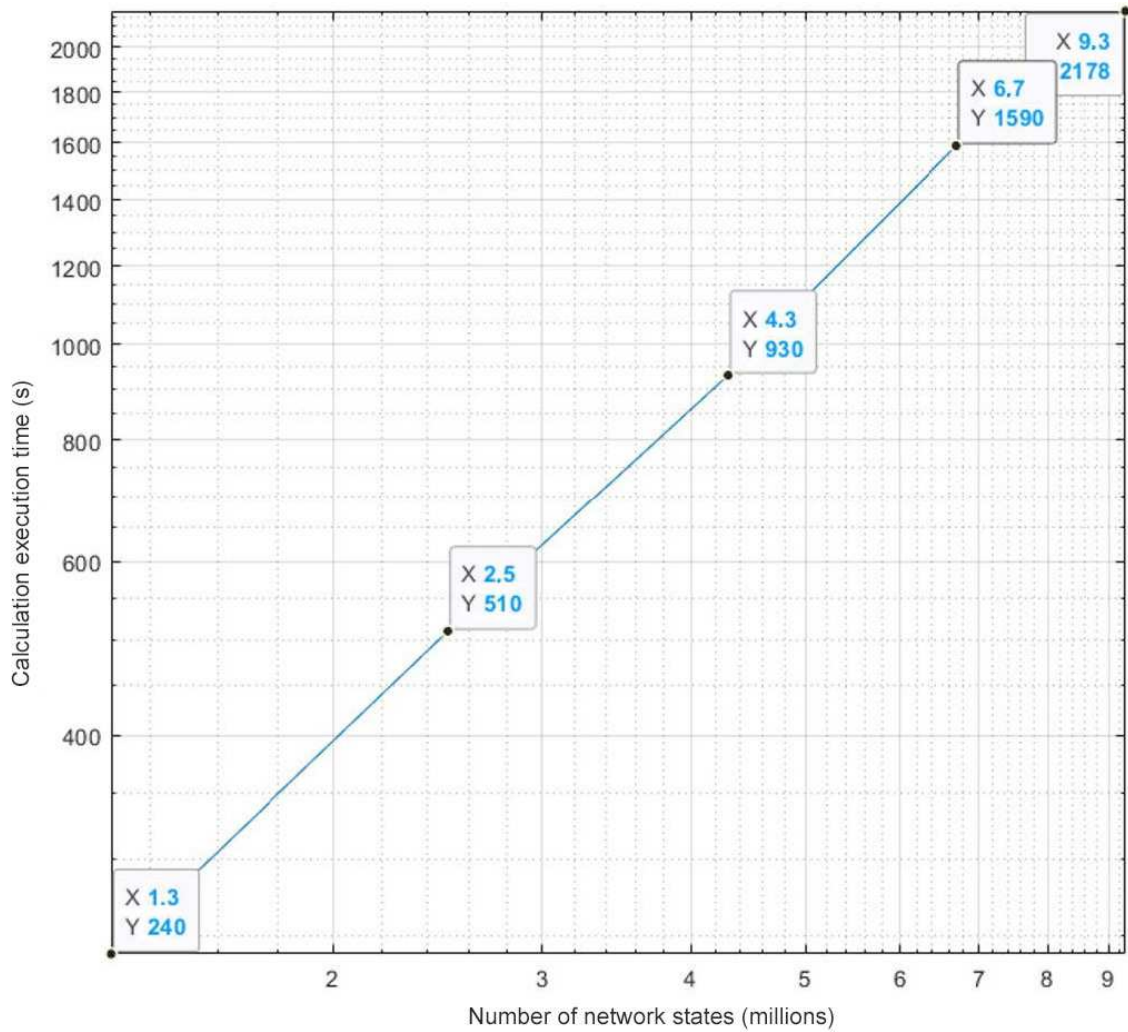
**Fig. 3.** Example of applying the resource allocation rule for a three-service network.

The rule  $f(s)$  for assigning high speeds is based on the fact that ensuring high speed when servicing data requests is more important than for RT traffic of the second type, and RT traffic of the first type does not need speed increase, hence  $b_1 = b'_1$ . At the same time,  $f(s)$  carries out the following procedure:

- Determination of whether the resource is sufficient to service all requests when allocating them resources  $b'_1$ ,  $b'_2$  and  $b'_e$ ;
- if the resource is sufficient and there is a remainder, allocation of resources  $b_e$  to as many data requests as possible;
- if all data requests are allocated resource  $b_e$  and there is a remainder, allocation of resources  $b_2$  to as many requests for RT traffic of the second type as possible;
- if there is again a remainder of free resource and there are data requests, allocation of the specified free resource to increase the service speed of these requests.

The use of the described rule  $f(s)$  for  $v = 33$  and  $s = (2, 2, 2)$  is illustrated in Fig. 3. At the first step, it is established that the resource is sufficient to service all requests at minimum speed and there is some residual resource, i.e.,  $2b'_1 + 2b'_2 + 2b'_e = 18$  and the residual resource is 15 quanta. The second step involves checking whether a speed increase for data requests is possible, and this possibility is confirmed. Each data request is allocated resource  $b_e$ . The residual resource decreases by 8 and now amounts to 7 quanta. At the third step, it is determined that speed increase is possible only for one request of the second type, and the residual resource is insufficient to increase the speed for the second request. Accordingly, the resource of one request of the second type increases to 8 quanta, and the residual resource decreases to 3 quanta. This remaining resource is given for servicing data requests at the fourth step. As a result, the total resource of data requests will be  $2b_e + 3 = 19$  speed quanta. For distributing the resource among these requests, known approaches described in [1] can be used, for example, the water-filling algorithm.

It should be noted that the proposed rule is only a particular example. Other rules are considered in some sources, for example in [24].



**Fig. 4.** Dependence of computation time on the number of network states  $N_{\text{states}}$ .

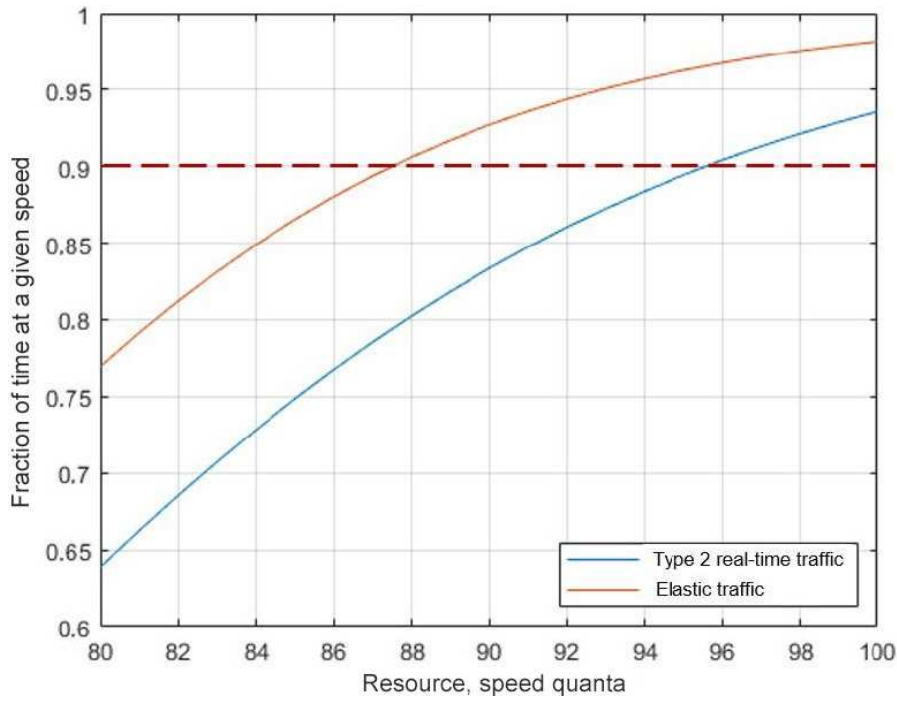
For the numerical analysis, we developed a MATLAB scenario to form the SEE from a set of network characteristics, solve it using the Gauss-Seidel method, and calculate the quality indicators. An assessment of the possibility of solving SEE of large dimension was carried out, illustrated in Fig. 4.

The dependence of the computation time on the number of states  $N_{\text{states}}$  is monotonically increasing and nearly linear. For  $N_{\text{states}} \approx 9.3$  million, the time was about 36 minutes.

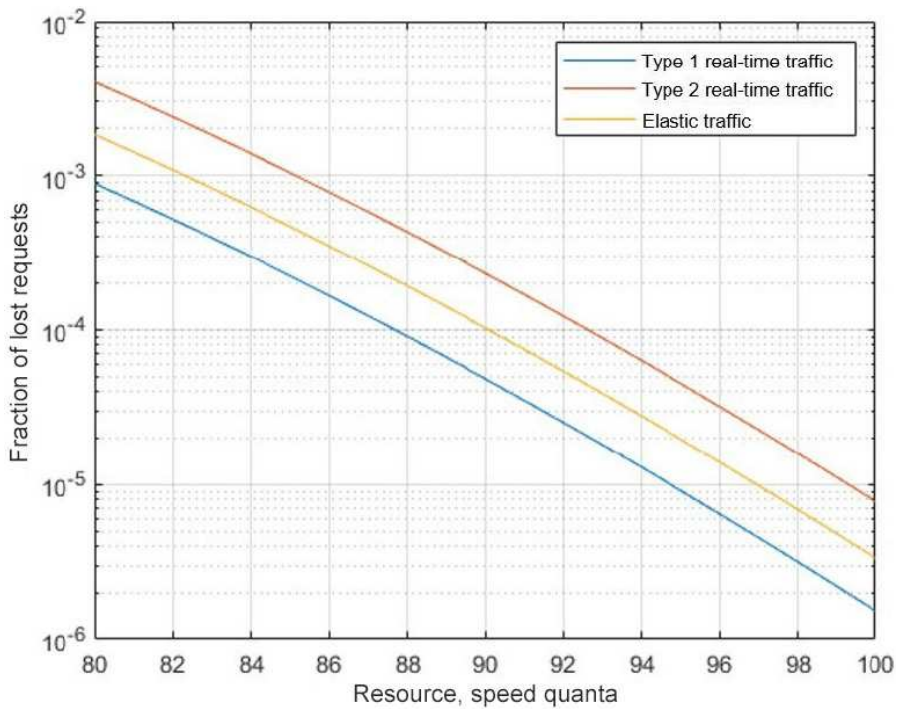
As noted in Section 1, the operator may be interested in both the task of determining the required resource at the network planning stage and the task of assessing the possibility of expanding the subscriber capacity of the network with the available resource.

When solving the first task, dependencies of traffic service quality indicators on the resource are constructed, illustrated in Figs. 5, 6, and 7.

Figure 5 shows that a value  $P_2$  no lower than 0.9 is ensured given a resource of  $v \geq 96$ , while for  $P_e$  the required resource is  $v \geq 88$ . For all values of  $v$  in the range from 80 to 100, as follows from Fig. 6, the constraint on the maximum fraction of lost requests is satisfied with a large margin. According to Fig. 7, to satisfy the constraint on the average file delivery time, a resource  $v \geq 87$  is required. Thus, all constraints are satisfied when the network resource  $v \geq 96$ . The minimum required volume of the network resource is 96 speed quanta.



**Fig. 5.** Dependencies on resource  $v$  of the fraction of time  $P_2$  during which RT traffic of the second type is transmitted at the specified high speed, and the fraction of time  $P_e$  during which data service traffic is transmitted at a speed not lower than the specified threshold.



**Fig. 6.** Dependencies on resource  $v$  of the fraction of lost requests of each type due to insufficient resource  $\pi_k$ ,  $k = 1, 2$  and  $\pi_e$ .

We will assess the possibility of network expansion using the example of increasing the number of consumers of the second RT service. Assume that the network has a resource  $v = 100$  and traffic parameters are according to the table with the difference that the possibility of increasing

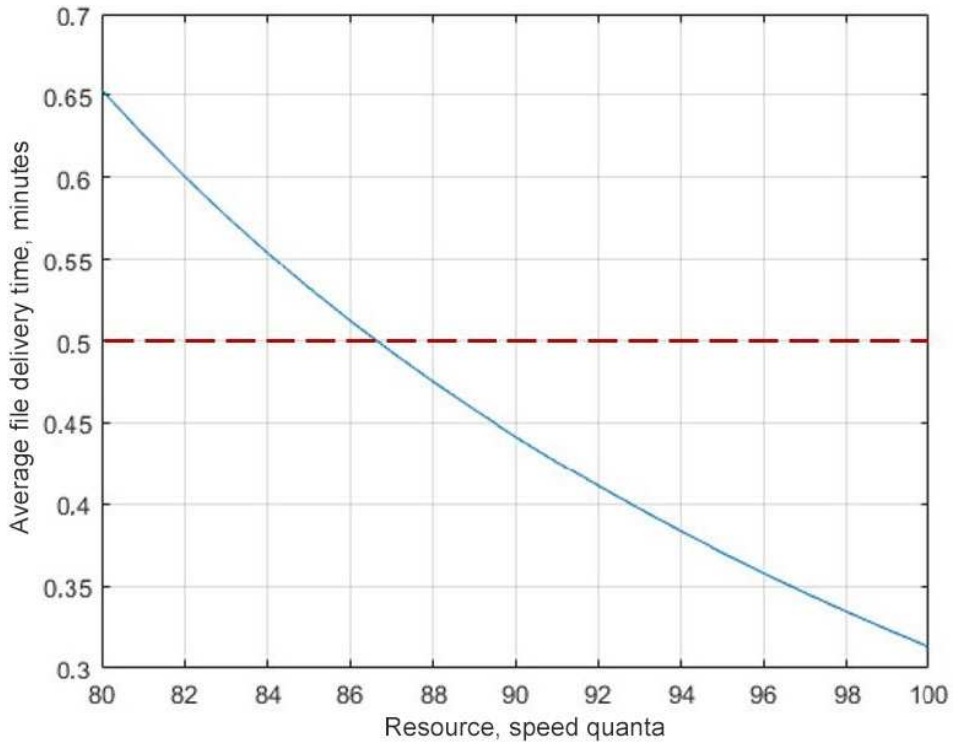


Fig. 7. Dependencies on resource  $v$  of the average file delivery time  $W$ .

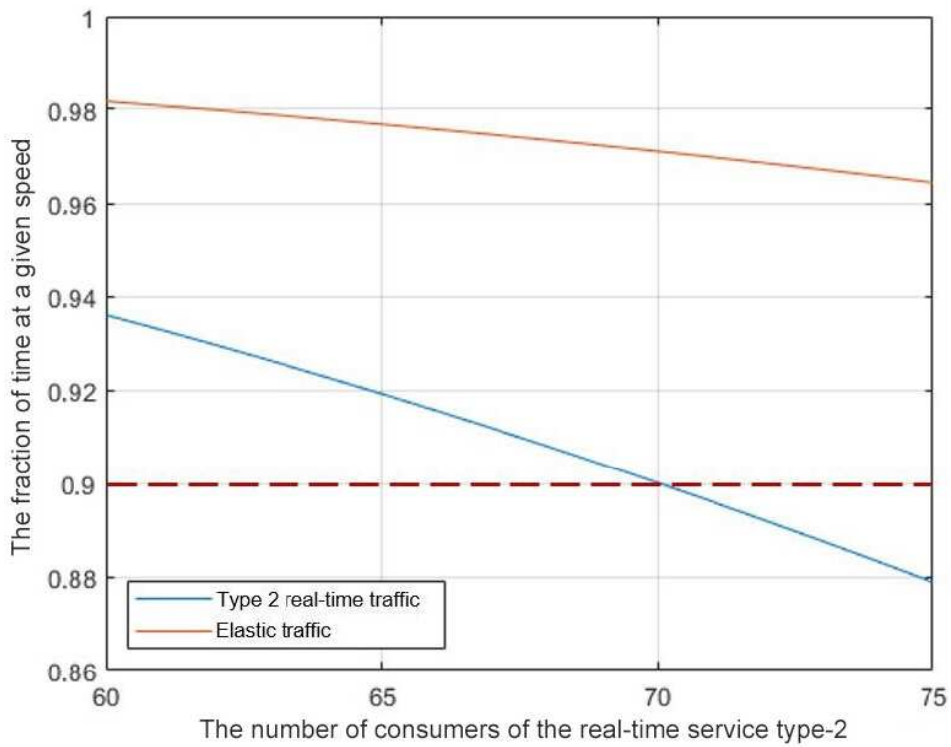


Fig. 8. Dependencies on the number of consumers of the second service  $N_2$  of the fraction of time  $P_2$  during which RT traffic of the second type is transmitted at the specified high speed, and the fraction of time  $P_e$  during which data service traffic is transmitted at a speed not lower than the specified threshold.

the parameter  $N_2$  is evaluated and it varies in the range from 60 to 75. The calculation results of indicators  $P_2$  and  $P_e$  are presented in Fig. 8. It can be seen that with an increase in  $N_2$ , the values of indicators  $P_2$  and  $P_e$  decrease. At the same time, the value  $P_e$  remains within the specified limits, and for  $P_2$  the constraint is satisfied only for  $N_2 \leq 70$ .

Thus, the network allows an increase in the number of consumers of the second RT service up to 70 without changing the resource volume. If the number of consumers exceeds 70, then to meet the specified constraints on the quality indicators, an increase in the network resource will be required.

## 5. CONCLUSION

This study builds a model for the joint servicing of real-time and elastic data traffic in a satellite multi-service network access node with a dynamically variable transmission rate. One of the key quality indicators is the fraction of time each service is provided at high speed. The model formalizes the network operation process using the apparatus of multidimensional stepwise Markov processes. Furthermore, relations for calculating QoS indicators have been derived. The software implementation of the model solves the systems of equilibrium equations for the Markov processes using the iterative Gauss-Seidel method. This approach enables the estimation of performance indicators for large-scale networks, a capability confirmed by numerical experiments involving up to ten million network states. Examples are provided to demonstrate the model's application in solving two key problems: 1) determining the required resource during the network planning stage, and 2) assessing the potential for expanding the network's subscriber capacity given the available resources. The model is suitable for integration into software for satellite network management systems. Furthermore, it can be used in operational contexts to justify the feasibility of measures aimed at enhancing network characteristics and modernization.

## REFERENCES

1. Stepanov, S.N., *Teletraffic Theory: Concepts, Models, Applications*, Moscow: Goryachaya liniya – Telekom, 2015, 868 p. (In Russ.)
2. Vishnevsky, V.M. and Efrosinin, D.V., *Queueing Theory and Machine Learning*, Moscow: INFRA-M, 2024, 370 p. (In Russ.)
3. Vishnevsky, V.M., Rykov, V.V., Kozyrev, D.V., et al., *Reliability Modeling of Tethered High-Altitude Unmanned Telecommunication Platforms*, Moscow: Tekhnosfera, 2022, 194 p. (In Russ.)
4. Vishnevsky, V.M., Dudin, A.N., and Klimenok, V.I., *Stochastic Systems with Correlated Flows: Theory and Applications in Telecommunication Networks*, Moscow: Tekhnosfera, 2018, 564 p. (In Russ.)
5. Vishnevsky, V.M., Lyakhov, A.I., Portnoy, S.L., et al., *Broadband Wireless Information Transmission Networks*, Moscow: Tekhnosfera, 2005, 529 p. (In Russ.)
6. Stepanov, M.S., Kanishcheva, M.G., Malikova, E.E., et al., The Development and Analysis of a Service Model for the Traffic of a Surveillance System Operator by a Dedicated Resource of an LTE Cell. *Proc. of 2022 Systems of Signals Generating and Processing in the Field of on Board Communications*, 2022, pp. 1–6.
7. Stepanov, M.S., Stepanov, S.N., and Kroshin, F.S., Effective Algorithm of Estimation the Performance Measures of Group of Servers with Dependence of Call Repetition on the Type of Call Blocking. *DCCN 2022 / Lecture Notes Computer Science*, Springer, Cham, 2022.
8. Gibadullina, E.E., Viskova, E.V., and Stepanov, S.N., Automated Service Configuration Management in IP/MPLS Networks. *4th International Science and Technology Conference "Modern Network Technologies – 2022"*, MoNeTec – 2022.
9. Andrabai, U.M., Kanishcheva, M., and Stepanov, S.N., Observation system resource planning in presence of access control based on volume of resource occupied by traffic flows. *T-Comm*, 2022, vol. 16, no. 8, pp. 54–62.

10. Dawood, T., Stepanov, M.S., Naoussi, C., et al., The Mathematical Model of the Internet of Things Traffic Servicing in Case of its Impulse Nature. *Proc. of 2023 Systems of Signals Generating and Processing in the Field of on Board Communications*, 2023, pp. 1–8.
11. Ndimumahoro, F., Stepanov, M.S., Muzata, A.R., et al., Using the Principles of Mobile Systems Modeling for LoRaWAN Characteristics Estimation. *Proc. of 2022 Systems of Signals Generating and Processing in the Field of on Board Communications*, 2022, pp. 1–8.
12. Stepanov, S.N. and Stepanov, M.S., Approximate Method for Evaluating Characteristics of Joint Service of Real-Time Traffic and Elastic Data Traffic in Multiservice Access Nodes. *Automation and Remote Control (ARC)* 2023, no. 11, pp. 93–114. (In Russ.)
13. Stepanov, S.N. and Stepanov, M.S., Methods for Estimating Required Resource Volume of Multiservice Access Nodes. *Automation and Remote Control (ARC)*, 2020, no. 12, pp. 129–152. (In Russ.)
14. Makeyeva, Ye.D., Kochetkova, I.A., and Shorgin, V.S., Model for Selecting eMBB Broadband Traffic Rates under URLLC Priority Transmission in 5G Network. *Systems and Means of Informatics*, 2023, vol. 33, no. 4, pp. 60–68. (In Russ.)
15. Borodakiy, V.Y., Samouylov, K.E., Gudkova, I.A., et al., Analyzing mean bit rate of multicast video conference in LTE network with adaptive radio admission control scheme. *Journal of Mathematical Sciences*, 2016, vol. 218, no. 3, pp. 257–268.
16. Maslov, A.A., Sebekin, G.V., Stepanov, S.N., et al., Model of processes for joint maintenance of real-time multiservice traffic and elastic data traffic in a network of low-power mobile subscriber terminals based on high-throughput satellites. *T-COMM*, 2024, vol. 18, no. 5, pp. 41–49.
17. Maslov, A.A., Sebekin, G.V., Stepanov, M.S., et al., Channel resource reservation model for heterogeneous traffic service in a network of low-power mobile subscriber terminals based on high-throughput satellites. *Information Processes*, 2024, vol. 24, no 1, pp. 1–15. (In Russ.)
18. Sebekin, G.V., Maslov, A.A., and Shchurkov, A.O., Modeling joint service of real-time multiservice traffic and elastic data traffic in networks based on high-throughput satellites. *Information-Measuring and Control Systems*, 2024, vol. 22, no. 2, pp. 11–22. (In Russ.)
19. Makov, S.V., Maslov, A.A., and Sebekin, G.V., Efficiency assessment of capacity resource usage for relay channels of HTS satellites in geostationary and highly elliptical orbits for organizing three-traffic-type transmission network. *Nanoindustry*, 2023, vol. 16, no. S9–2 (119), pp. 613–619.
20. Maslov, A.A., Sebekin, G.V., Stepanov, M.S., et al., Modeling subscriber service processes in a data transmission network based on low Earth orbit satellites. Part I. *Information Processes*, 2024, vol. 24, no. 4, pp. 335–349. (In Russ.)
21. Maslov, A.A., Sebekin, G.V., Stepanov, M.S., et al., Modeling subscriber service processes in a data transmission network based on low Earth orbit satellites. Part II. *Information Processes*, 2025, vol. 25, no. 2, pp. 151–168. (In Russ.)
22. Antonovich, P.I., Maslov, A.A., and Sebekin, G.V., Problem of minimizing satellite channel resource for organizing IoT networks operation. *Systems of Synchronization, Signal Generation and Processing*, 2021, vol. 12, no. 6, pp. 57–64. (In Russ.)
23. Sebekin, G.V., Shchurkov, A.O., Maslov, A.A., et al., Building a multiservice satellite communication platform based on LTE (3GPP) data network solutions. *Advances in Modern Radioelectronics*, 2024, vol. 78, no. 2, pp. 66–75. (In Russ.)
24. Stepanov, S.N., Model for servicing real-time services and data traffic with dynamically changing transmission rate. *Automation and Remote Control (ARD)*, 2010, no 1, pp. 18–33. (In Russ.)

*This paper was recommended for publication by V.M. Vishnevskii, a member of the Editorial Board*

# Trajectory Countermeasures Against a Linear Observer

A. Potapov<sup>\*,a</sup> and A. Galyaev<sup>\*,b</sup>

*\*Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia*

*<sup>a</sup>potapov@ipu.ru, <sup>b</sup>galyaev@ipu.ru*

Received April 17, 2025

Revised August 11 2025

Accepted August 18, 2025

**Abstract**—We consider the controlled dynamics of three objects in  $n$ -dimensional space: Attacker (A), Defender (D), and Target (T). Attacker estimates Target’s relative position using a Kalman–Bucy filter and constructs a collision trajectory based on this estimate. In response, the Target deploys the Defender, which disrupts the Attacker’s estimation process by interfering with its reception channel, thereby preventing interception. This leads to the formulation of a problem: designing an optimal trajectory for the Defender to maximize the time until the Attacker intercepts the Target. Numerical simulations of the dynamics of each object are conducted to evaluate the effectiveness of deploying the Defender.

*Keywords:* observations control, Kalman–Bucy filter, ADT-game, guidance countermeasures, trajectory optimisation

**DOI:** 10.7868/S1608303225110055

## 1. INTRODUCTION

Multi-player cooperative control problem are of interest to researchers both for practical reasons, such as the development of unmanned vehicle technologies, and for academic ones, as they provide an analytical framework for evaluating solutions and algorithms. Applied multi-player problems, as well as adversarial two-player game-theoretic problems, are divided into two subclasses with opposing objectives: interception [1] and evasion problems [2, 3] or countering interception [4, 5].

As the number of players on either side increases, so do their opportunities to achieve their goals. The one and simple expansion of two-player game (Attacker and Target) is achieved by adding the Defender. It’s called ADT (Attacker–Defender–Target) game [6]. However, before formalizing the problem statement, it is necessary to select the policy of each player. The closest to practical applications is the situation when the Attacker bases its decisions on its sensor data. To achieve this, it must estimate its own state and/or that of the target. This is accomplished using the theory of observation control [7], specifically the Kalman filtering method [8, 9], which has been actively developing for a long time [10], including the use of artificial intelligence technologies [11, 12]. Subsequently, in order to successfully intercept a Target, the Attacker needs to choose a guidance algorithm, the most famous of which is the law of proportional navigation [13].

The Defender’s objective meanwhile can be interception of the Attcker [14] (in this case it’s called “hard” counteraction) as well as influence on its reception channel [15, 16] (in this case it’s called “soft” counteraction). Furthermore both the distance between the Target and the Attacker at the terminal point in time and the time required to intercept the Target can be used as a problem criteria.

At the moment, there are a lot of works that provide an analytical solution to the problem of “hard” counteraction in one formalization or another [17]. However, Monte Carlo [18] or artificial intelligence [19] methods are most often used to solve problems of “soft” counteraction, which does

not allow to analyze the solution. At the same time, the theory of observational control and the theory of random processes make it possible to study complex dynamic systems, taking into account the decision-making methods used by objects.

Current work aims to formulate the problem of countering the simplest guidance algorithm and its analytical solution using the described mathematical apparatus. The specifics of the setup are such that, due to the use of the Defender, the equation of the observed process — based on which the Attacker builds an estimate of the Target's relative position — does not match the equation that this observed process actually follows.

The structure of the work is as follows. Section 2 formalizes the ADT-game with incomplete information, where every player has a linear dynamic, and the Attacker gets information about the outside world from the linear observation channel, depending on position of the Target and the Defender in the relative coordinate system associated with the Attacker. Acting in a coalition, by choosing their own direction of movement, the Target and Defender must delay or prevent the intersection of the Target, therefore, Section 3 is devoted to the formulation of the optimal control problem. Sections 4 and 5 provide a solution to this problem and numerical simulations showing the effectiveness of using a Defender. In conclusion, the directions of development of the considered task are proposed.

## 2. THE ADT GAME MODEL WITH INCOMPLETE INFORMATION

### 2.1. Description of the Studied System

Let's consider a linear system with 3 types of players — the Attacker (A), the Defender (D) and the Target (T). The equations of motion of such a system can be written in the form of Ito equations

$$\begin{cases} dx_A(t) = Fx_A(t)dt + B_Au(t)dt + \sigma_A dw_A(t), \\ dx_D(t) = Fx_D(t)dt + B_Dv(t)dt + \sigma_D dw_D(t), \\ dx_T(t) = Fx_T(t)dt + \sigma_T dw_T(t) \end{cases} \quad (1)$$

with some initial conditions at time zero.

Let's assume that the Attacker receives information about the system through the observation channel, which is described by the equation

$$dz(t) = \beta_e e(t)dt + \beta_\varepsilon \varepsilon(t)dt + \sigma_z dw_z(t), \quad (2)$$

where

$$\begin{aligned} x_A, x_D, x_T \in \mathbb{R}^n, \quad F \in \mathbb{R}^{n \times n}, \quad B_D \in \mathbb{R}^{m \times n}, \quad v \in \mathbb{R}^m, \\ B_A \in \mathbb{R}^{r \times n}, \quad u \in \mathbb{R}^r, \quad \beta_e, \beta_\varepsilon \in \mathbb{R}^{n \times c}, \\ e(t) = x_A(t) - x_T(t), \quad \varepsilon(t) = x_A(t) - x_D(t). \end{aligned}$$

In equations (1), (2)  $w_A, w_D, w_T, w_z$  — are standard  $n$ -dimensional Wiener processes,  $\sigma_A, \sigma_D, \sigma_T, \sigma_z \in \mathbb{R}^{n \times n}$ . Thus, the coordinates of all players have dimension  $n$ , the control vectors of the Attacker and Defender have dimensions  $r$  and  $m$ , respectively, and the measured value  $z$  has dimension  $c$ .

With this equation of the measurement channel, it is more convenient to switch from the equations (1) to the equations of relative coordinates  $e, \varepsilon$  dynamics. They have the form of

$$\begin{cases} d\varepsilon(t) = F\varepsilon(t)dt + B_Au(t)dt - B_Dv(t)dt + \sigma_\varepsilon dw_\varepsilon(t), \\ de(t) = Fe(t)dt + B_Au(t)dt + \sigma_e dw_e(t), \end{cases} \quad (3)$$

where

$$\sigma_\varepsilon = \sigma_A + \sigma_D, \quad \sigma_e = \sigma_A + \sigma_T.$$

Let the Attacker estimates the relative position of the Target. It's assumed that that it does not know about the existence of the Defender and uses the Kalman filter [9] as an optimal estimation algorithm. The equations of states and observations, based on which the Attacker makes an estimation, are written as

$$\begin{cases} de(t) = Fe(t)dt + B_A u(t)dt + \sigma_e dw_e(t), \\ dy(t) = \beta_e e(t)dt + \sigma_y dw_y(t). \end{cases} \tag{4}$$

**Proposition 1.** *To simplify the subsequent discussion, we assume that in system (4) pair  $(F, \sigma_e)$  is controllable, pair  $(F, \beta_e)$  is observable, and  $\det \sigma_q \neq 0$ , where  $q$  — is any of symbols  $A, D, T, z, y$ .*

*Remark 1.* The second equation of the (4) system does not match the observation equation (2). This is due to the fact that the Attacker is unaware of the Defender's existence when forming the estimation. It is this circumstance that is associated with the accumulation of errors in estimation the relative position of the Target and the subsequent miss of the Attacker.

For brevity of notation, time arguments will be omitted.

Let's assume that the Attacker uses the following control law to approach the Target (it is obtained when solving some linear-quadratic problems, see, for example, [20]):

$$u = \lambda \hat{e}, \quad \lambda \in \mathbb{R}^{n \times r},$$

where  $\hat{e}$  — vector  $e$  estimation.

In such case random vector  $e$  estimation is described in [7] by solving equations

$$d\hat{e} = F\hat{e} dt + B_A u dt + \gamma \beta_e^T (\sigma_y \sigma_y^T)^{-1} (\beta_e e dt + \beta_\varepsilon \varepsilon dt + \sigma_z dw_z - \beta_e \hat{e} dt), \tag{5}$$

$$\frac{d\gamma}{dt} = F\gamma + \gamma F^T + \sigma_e \sigma_e^T - \gamma \beta_e^T (\sigma_y \sigma_y^T)^{-1} \beta_e \gamma^T, \tag{6}$$

where  $\gamma$  — mean square filtering errors.

Then, denoting  $\varphi(\gamma) = \gamma \beta_e^T (\sigma_y \sigma_y^T)^{-1}$  and then substituting the expression for control  $u$  into the filtration equations (5), (6), we obtain the dynamic equation of vector  $\hat{e}$  in the following form

$$d\hat{e} = \hat{F}(\gamma)\hat{e} dt + \varphi(\gamma)\beta_\varepsilon \varepsilon dt + \varphi(\gamma)\beta_e e dt + \hat{\sigma}_e(\gamma)dw_z,$$

where

$$\begin{aligned} \hat{F}(\gamma) &= F + B_A \lambda - \varphi(\gamma)\beta_e, \\ \hat{\sigma}_e(\gamma) &= \varphi(\gamma)\sigma_z. \end{aligned}$$

As a result, the dynamics equations of all the players and the Kalman filter make up the system

$$\begin{cases} d\varepsilon = F\varepsilon dt + B_A \lambda \hat{e} dt - B_D v dt + \sigma_\varepsilon dw_\varepsilon, \\ de = Fe dt + B_A \lambda \hat{e} dt + \sigma_e dw_e, \\ d\hat{e} = \hat{F}(\gamma)\hat{e} dt + \varphi(\gamma)\beta_\varepsilon \varepsilon dt + \varphi(\gamma)\beta_e e dt + \hat{\sigma}_e(\gamma)dw_z, \\ \dot{\gamma} = F\gamma + \gamma F^T + \sigma_e \sigma_e^T - \gamma \beta_e^T (\sigma_y \sigma_y^T)^{-1} \beta_e \gamma^T. \end{cases} \tag{7}$$

## 2.2. Problem Statement

Let's assume that the mathematical expectations of the variables of the (7) system are known at the initial moment of time:

$$\mathbb{E} q(0) = \mu_q(0), \quad (8)$$

where  $q$  — is any of symbols  $\varepsilon$ ,  $e$ ,  $\hat{e}$ . The initial value of the covariance matrices of all vectors will, in turn, be considered unknown.

*Remark 2.* In fact, setting the initial conditions in this form means that the Target-Defender coalition knows the initial position of the Attacker with accuracy up to a certain measurement error for some reason (for example, according to its own measurements). We also assume that the coalition knows the Attacker's algorithm for estimating the position of the Target. This leads to the following formulation of the countermeasures problem.

*Problem 1.* For a system whose dynamics is described by the equations (7), the mathematical expectation of random processes at the initial moment of time is described by (8), the travel time is limited by  $t_*$ , it is necessary to find the control  $v(t)$ , limited with bounded absolute value  $|v| \leq \varkappa$ , which delivers the maximum to the criterion

$$J[e] = \mathbb{E} \left( e^T(t_*)e(t_*) \right). \quad (9)$$

*Remark 3.* At its core, the (9) criterion is a quadratic miss. Let's also pay attention to the type of initial conditions (8) — the variance of random vectors is considered unknown. As will be shown below, due to the type of criterion (9) and the control form, information about the variance will not be needed to solve the problem.

## 3. REDUCTION OF THE PROBLEM TO A DETERMINISTIC FORM

It is easy to see that the criterion (9) can be transformed to the form

$$J[e] = \mathbb{E} \left( e^T(t_*)e(t_*) \right) = \text{tr} (\Sigma_e(t_*)) + \mu_e^T(t_*)\mu_e(t_*), \quad (10)$$

where  $\Sigma_e(t) = \text{Var} e(t)$  — is a variance of a random vector  $e(t)$ .

It is also obvious that the dynamic equations of mathematical expectations have the form

$$\begin{cases} \dot{\mu}_\varepsilon = F\mu_\varepsilon + B_A\lambda\hat{\mu}_e - B_Dv, \\ \dot{\mu}_e = F\mu_e + B_A\lambda\hat{\mu}_e, \\ \dot{\hat{\mu}}_e = \hat{F}(\gamma)\hat{\mu}_e + \varphi(\gamma)\beta_\varepsilon\mu_\varepsilon + \varphi(\gamma)\beta_e\mu_e, \\ \dot{\gamma} = F\gamma + \gamma F^T + \sigma_e\sigma_e^T - \gamma\beta_e^T(\sigma_y\sigma_y^T)^{-1}\beta_e\gamma^T. \end{cases} \quad (11)$$

If we talk about variance dynamic equations, the following lemma is important, which is given without proof due to its well-known nature.

**Lemma 1.** *If a random process  $x(t)$  follows the Ito equation*

$$dx(t) = A(t)x(t)dt + B(t)v(t)dt + \sigma(t)dw(t)$$

*with deterministic matrices  $A(t)$ ,  $B(t)$ ,  $\sigma(t)$  and control  $v(t)$ , then the dynamics of its covariance matrix follows the Riccati equation*

$$\dot{\Sigma}(t) = A(t)\Sigma(t) + \Sigma(t)A^T(t) + \sigma(t)\sigma^T(t).$$

Let's apply Lemma 1 to the first three equations of (7). Then, the variance of the random vector  $x = (\varepsilon^T \ e^T \ \hat{e}^T)^T$  is described by the equation

$$\dot{\Sigma} = \Phi(\gamma)\Sigma + \Sigma\Phi^T(\gamma) + \sigma_{\text{total}}\sigma_{\text{total}}^T, \tag{12}$$

where

$$\Phi(\gamma) = \begin{pmatrix} F & 0 & B_A\lambda \\ 0 & F & B_A\lambda \\ \varphi(\gamma)\beta_\varepsilon & \varphi(\gamma)\beta_e & \hat{F}(\gamma) \end{pmatrix}, \quad \sigma_{\text{total}} = \text{diag}(\sigma_\varepsilon, \sigma_e, \hat{\sigma}(\gamma)).$$

**Lemma 2.**  $\text{tr}(\Sigma_e(t_*))$  does not depend on the choice of control  $v(t)$ .

**Proof.** It is easy to see that the equations of the (12) system, together with the last equation of the (7) system, form a closed system in which there is no control, which leads to the statement of the lemma. Thus, maximizing the criterion (10) by control is equivalent to maximizing only its second term.

*Remark 4.* In fact, the criterion (10) contains 2 terms responsible for the observer's "miss". The first of them characterizes the influence of the measurement channel noise of the criterion, which means the influence of the signal emitted by the Defender.

The second term characterizes the effect on the criterion of the Defender's own trajectories. Lemma 2 actually states that the inability to control the signal emitted by the Defender leads to the inability to influence the first term in the criterion (10). This term would be important if it were possible to control the emitted signal, namely the coefficient  $\beta_e$ . However, this is not possible in the formulation under study, and therefore the criterion will be maximized only using the second term responsible for the trajectory of the observer and Defender.

On the other hand, this form of criterion makes it possible to compare the contributions of each component to the overall task criterion at each step or during the evaluation of the entire mission. At the same time, considering each term separately, it is possible to establish the capabilities of the Defender to counteract both by maneuvering and by influencing the measuring channels.

Next, note that the Riccati equation in the (11) system is an independent differential equation. In this case, the remaining three equations of the system, in fact, constitute a linear non-autonomous system of differential equations. This, in turn, means that instead of the initial dynamic system (7), we can consider a system of mathematical expectations described by the equations

$$\begin{cases} \dot{\mu}_\varepsilon = F\mu_\varepsilon + B_A\lambda\hat{\mu}_e - B_Dv, \\ \dot{\mu}_e = F\mu_e + B_A\lambda\hat{\mu}_e, \\ \dot{\hat{\mu}}_e = \hat{F}(\gamma)\hat{\mu}_e + \varphi(\gamma)\beta_\varepsilon\mu_\varepsilon + \varphi(\gamma)\beta_e\mu_e, \end{cases}$$

which we will write down more briefly as

$$\dot{\mu} = \Phi(\gamma)\mu + Bv, \tag{13}$$

where

$$\mu = \begin{pmatrix} \mu_\varepsilon \\ \mu_e \\ \hat{\mu}_e \end{pmatrix}, \quad \Phi(\gamma) = \begin{pmatrix} F & 0 & B_A\lambda \\ 0 & F & B_A\lambda \\ \varphi(\gamma)\beta_\varepsilon & \varphi(\gamma)\beta_e & \hat{F}(\gamma) \end{pmatrix}, \quad B = \begin{pmatrix} -B_D \\ 0 \\ 0 \end{pmatrix},$$

and  $\gamma(t)$  — is a solution of a differential equation

$$\dot{\gamma} = F\gamma + \gamma F^T + \sigma_e\sigma_e^T - \gamma\beta_e^T(\sigma_y\sigma_y^T)^{-1}\beta_e\gamma^T$$

with an initial condition  $\gamma(0) = \gamma_0$ .

Instead of the (9) criterion, taking into account the 2 Lemma, we write a new integral criterion for the (13) system

$$J'[e] = -\mu_e^T(t_*)\mu_e(t_*) = -\frac{1}{2} \int_0^{t_*} \langle \dot{\mu}, Q\mu \rangle dt = -\frac{1}{2} \int_0^{t_*} \langle A\mu, Q\mu \rangle dt,$$

where

$$Q = \begin{pmatrix} 0 & 0 & 0 \\ 0 & E_n & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

and formulate the following problem.

*Problem 2.* For a system whose dynamics is described by the equations (13), the initial conditions are set by the equations (8), the travel time is limited by  $t_*$ , it is necessary to find the control  $v(t)$ , limited modulo  $|v| \leq \varkappa$ , which delivers minimum of the criterion

$$J'[\mu] = -\frac{1}{2} \int_0^{t_*} \langle A\mu, Q\mu \rangle dt. \quad (14)$$

**Proposition 2.** *The solution to the Problem 2 is the solution to the Problem 1.*

**Proof.** The proof, in fact, is the reasoning carried out above.

#### 4. SOLVING A DETERMINISTIC OPTIMAL CONTROL PROBLEM

To solve the Problem 2, let's use the Pontryagin maximum principle [21]. To do this, we will write down the Pontryagin function

$$H = \langle \psi, \Phi(\gamma)\mu + Bv \rangle + \frac{1}{2} \langle \Phi(\gamma)\mu, Q\mu \rangle.$$

The maximum condition

$$\langle \psi, Bv \rangle = \langle B^T \psi, v \rangle \longrightarrow \max_{|v| \leq \varkappa}$$

gives the following form of optimal control almost everywhere:

$$v^*(t) = -\varkappa \left| B_D^T \psi_1 \right|^{-1} B_D^T \psi_1, \quad (15)$$

where  $\psi_i$  — are components of the vector  $\psi = \left( \psi_1^T \quad \psi_2^T \quad \psi_3^T \right)^T$ .

The equations of the conjugate system are written as

$$\dot{\psi} = -\Phi^T(\gamma)\psi - \frac{1}{2} \left( \Phi^T(\gamma)Q + Q\Phi(\gamma) \right) \mu. \quad (16)$$

The initial conditions for the (13) system are set as (8). At the same time, for conjugate variables from the transversality conditions [21], values are known at the right end of the trajectory in the form of  $\psi(t_*) = 0$ .

The solution of such two-point problem is found by the sequential approximation method described in [22], having previously solved the Ricatti differential equation separately. Its essence is as follows: any valid control  $v_1(t)$  is chosen as an initial approximation. Next, at the  $k$ th iteration of the method, it is required:

- (1) to solve the Cauchy problem for the equation (13) with initial conditions (8) and control  $v_k(t)$ . Thus, we obtain the trajectory  $\mu_k(t)$  by  $[0, t_*]$ ;
- (2) to solve the conjugate system (16) with terminal conditions  $\psi(t_*) = 0$  from  $t_*$  to 0 for  $v(t) = v_k(t)$ ,  $\mu(t) = \mu_k(t)$ . Thus, we obtain the conjugate variables  $\psi_k(t)$  by  $[0, t_*]$ ;
- (3) to define the control of  $v_{k+1}(t)$  on  $[0, t_*]$ , according to the gather (15).

The algorithm stops when the value

$$\text{Err} = \sum_{t=0}^{t_*} |v_{k+1}(t) - v_k(t)| \tag{17}$$

becomes less than the pre-selected number  $\bar{\delta}$ :  $\text{Err} < \bar{\delta}$ . The amount in (17) is taken from all points of the time grid entered by the user on the segment  $[0, t_*]$ .

*Remark 5.* Solving such a boundary value problem can take a considerable amount of time. Namely, executing a function written in MATLAB to calculate the control in the Section 6 takes an average of 0.0967 s on an Apple M1 processor. In the case of high object speeds, such a time to solve the boundary value problem will not allow successful Target protection.

### 5. LINEARIZED SYSTEM RESEARCH

Note that for linear systems, it is known that in the case of observability and controllability of the system, the Kalman filter error  $\gamma$ , and therefore the feedback coefficient itself  $\varphi(\gamma)$  converge at  $t \rightarrow \infty$  for any initial matrix  $\gamma(0)$  [23]. In this case, given the terminal form of the (10) functional, it would be reasonable to consider the (13) system starting from some large point in time  $t$  with a constant matrix  $\gamma$ .

According to the Proposition 1, the conditions of Theorem 3.7 [23, p. 237] on the convergence of the solution of the Riccati differential equation are fulfilled. Therefore, the limiting value of the covariance matrix satisfies the algebraic Riccati equation

$$FP + PF^T + \sigma_e \sigma_e^T - P \beta_e^T (\sigma_y \sigma_y^T)^{-1} \beta_e P = 0.$$

The limiting value of the feedback coefficient is found according to the equation

$$\varphi = P \beta_e^T (\sigma_y \sigma_y^T)^{-1}.$$

In this case, for large  $t$ , instead of a dynamic system (13) with a nonlinear Riccati equation, we can consider a linear system of mathematical expectations described by the equations

$$\begin{cases} \dot{\mu}_\varepsilon = F \mu_\varepsilon + B_A \lambda \hat{\mu}_e - B_D v, \\ \dot{\mu}_e = F \mu_e + B_A \lambda \hat{\mu}_e, \\ \dot{\hat{\mu}}_e = \hat{F} \hat{\mu}_e + \varphi \beta_\varepsilon \mu_\varepsilon + \varphi \beta_e \mu_e, \end{cases}$$

which we will write down more briefly as

$$\dot{\mu} = A \mu + B v, \tag{18}$$

where

$$\mu = \begin{pmatrix} \mu_\varepsilon \\ \mu_e \\ \hat{\mu}_e \end{pmatrix}, \quad A = \begin{pmatrix} F & 0 & B_A \lambda \\ 0 & F & B_A \lambda \\ \varphi \beta_\varepsilon & \varphi \beta_e & \hat{F} \end{pmatrix}, \quad B = \begin{pmatrix} -B_D \\ 0 \\ 0 \end{pmatrix}.$$

The optimal control problem for the (18) system is formulated using the (14) criterion as follows:

*Problem 3.* For a system whose dynamics is described by the equations (18), the initial conditions are set by the equations (8), the travel time is limited by  $t_*$ , it is necessary to find the control  $v(t)$ , limited by module  $|v| \leq \varkappa$ , which delivers minimum to the criterion

$$J'[\mu] = -\frac{1}{2} \int_0^{t_*} \langle A\mu, Q\mu \rangle dt.$$

In the Problem 3 the law of optimal control coincides with (15), however, the equations of the conjugate system, unlike the equations of (16), are written as

$$\dot{\psi} = -A^T \psi - \frac{1}{2} (A^T Q + QA) \mu.$$

## 6. NUMERICAL SIMULATION

### 6.1. Simulation Parameters

Let's consider a system that is represented by a double integrator. The dynamics of such a system is described in simple movements. We will denote by the symbol  $E_k$  a unit matrix of size  $k \times k$ , and by the symbol  $0_k$  the matrix  $k \times k$ , each element of which is equal to 0. Next, let's assume that the matrices of the (1) system have the form

$$F = \begin{pmatrix} 0_2 & E_2 \\ 0_2 & 0_2 \end{pmatrix}, \quad B_A = B_D = \begin{pmatrix} 0_2 \\ E_2 \end{pmatrix}, \quad n = 4, \quad m = r = 2. \tag{19}$$

Other constants are set as follows:

$$\lambda = -6 \begin{pmatrix} E_2 & E_2 \end{pmatrix}, \quad \varkappa = 6 \times 10^{-3}, \quad \beta_e = \beta_\varepsilon = E_4, \quad t_* = 50, \tag{20}$$

$$\sigma_A = \sigma_D = \sigma_T = \sigma_z = \sigma_y = 10^{-3} \times E_4, \tag{21}$$

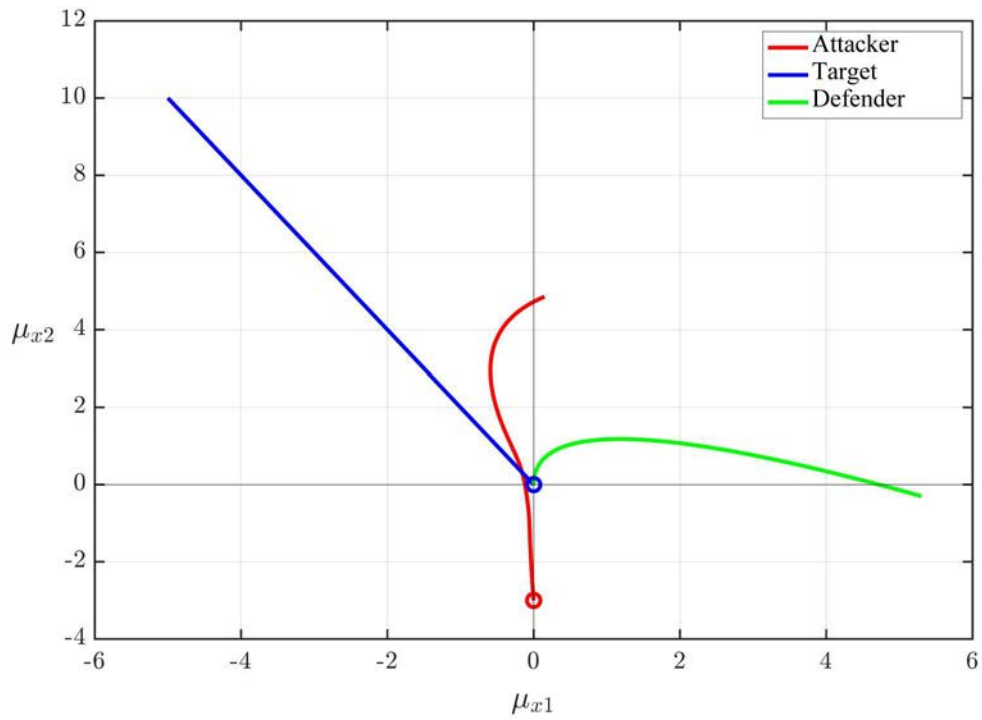
$$\mathbb{E} x_A(0) = x_A^0 = \begin{pmatrix} 0 \\ -3 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbb{E} x_T(0) = \mathbb{E} x_D(0) = x_D^0 = x_T^0 = \begin{pmatrix} 0 \\ 0 \\ -0.1 \\ 0.2 \end{pmatrix}. \tag{22}$$

*Remark 6.* For the (4) system with matrices selected according to (19)–(22), proposition 1 is fulfilled.

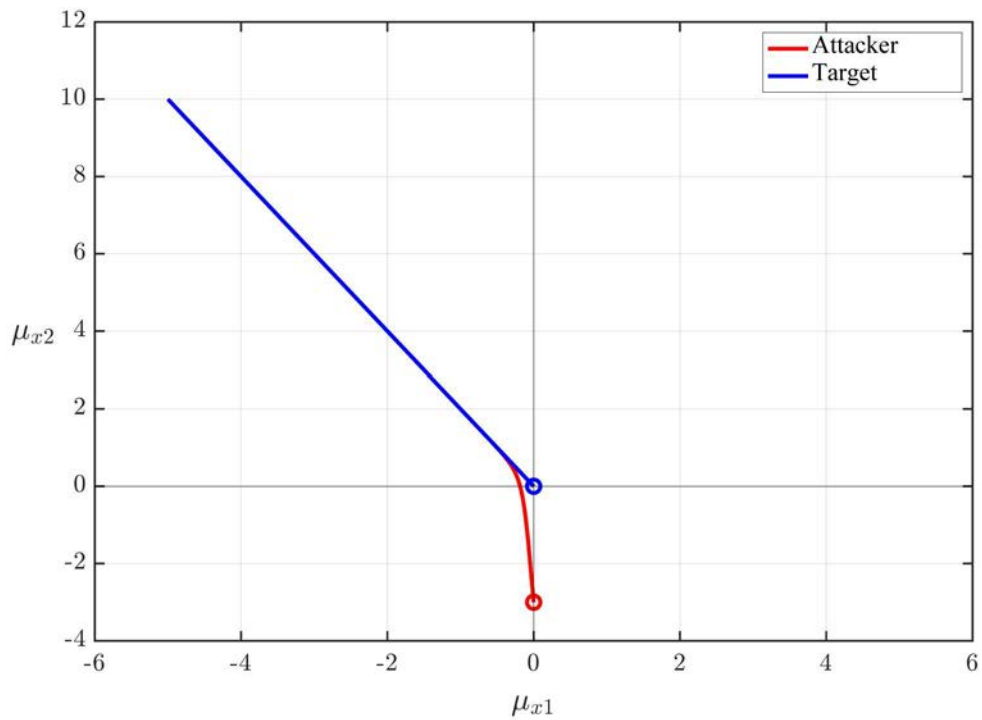
### 6.2. Simulation Results for Successful Defender Operation over the Entire Time Period

In the following we will talk about the mathematical expectations of the corresponding random processes only, since this characteristic fully reflects the essence of the problem.

Let's consider the case when the Defender has an expected effect on the receiving channel, affecting the trajectory of the Attacker as a whole. At the same time, up to the terminal moment of time  $t_*$ , we believe that the Attacker does not change his targeting tactics, even though the estimation in his receiving channel may deteriorate over time. The trajectories along which objects move with control calculated according to the equality (15) and the equations (18) in the plane of mathematical expectations of their coordinates are shown in Fig. 1 in the case of using a Defender, as well as in Fig. 2 without using a Defender.



**Fig. 1.** Object movement trajectories when using the Defender.



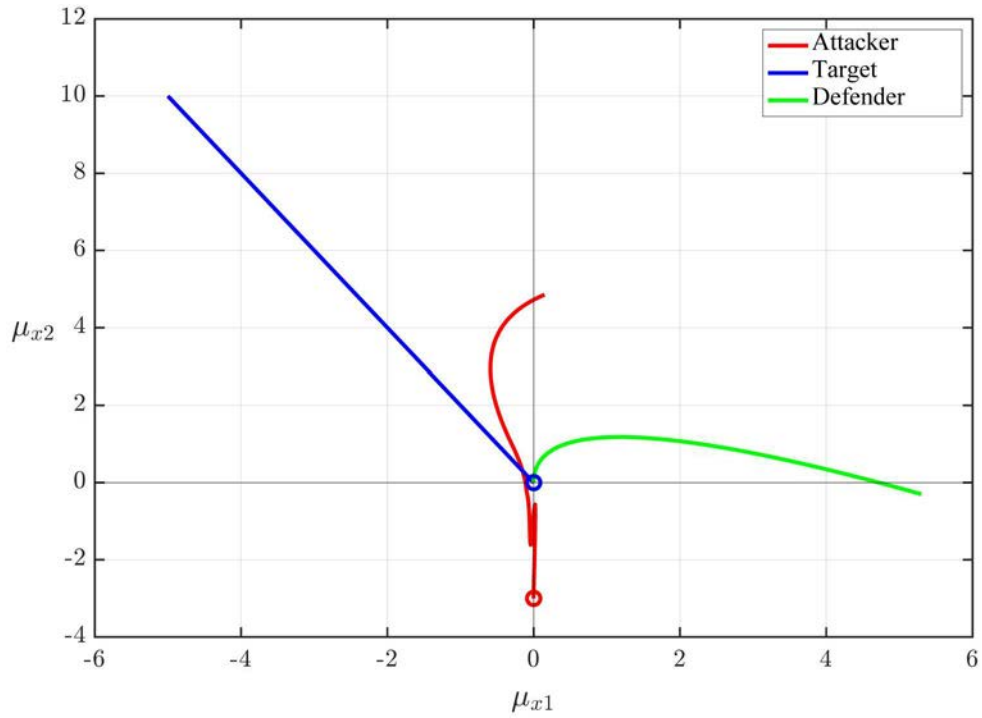
**Fig. 2.** Object movement trajectories without using a Defender.

In the first case, the criterion value is

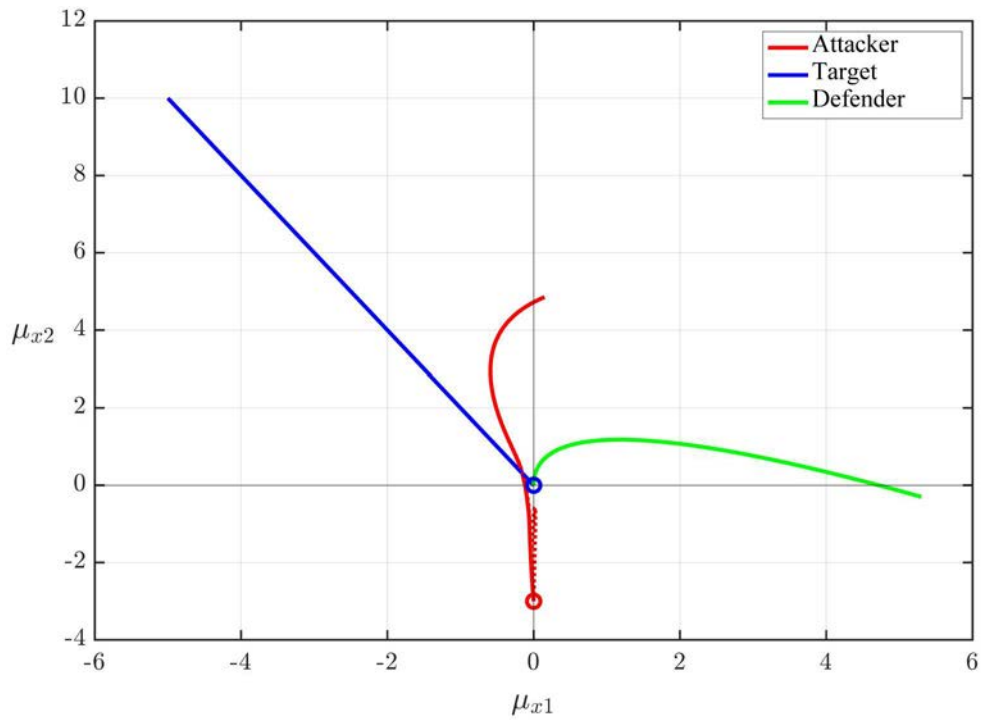
$$J'[\mu_1] = -53.06,$$

while in the second, it is

$$J'[\mu_2] \approx 0.$$

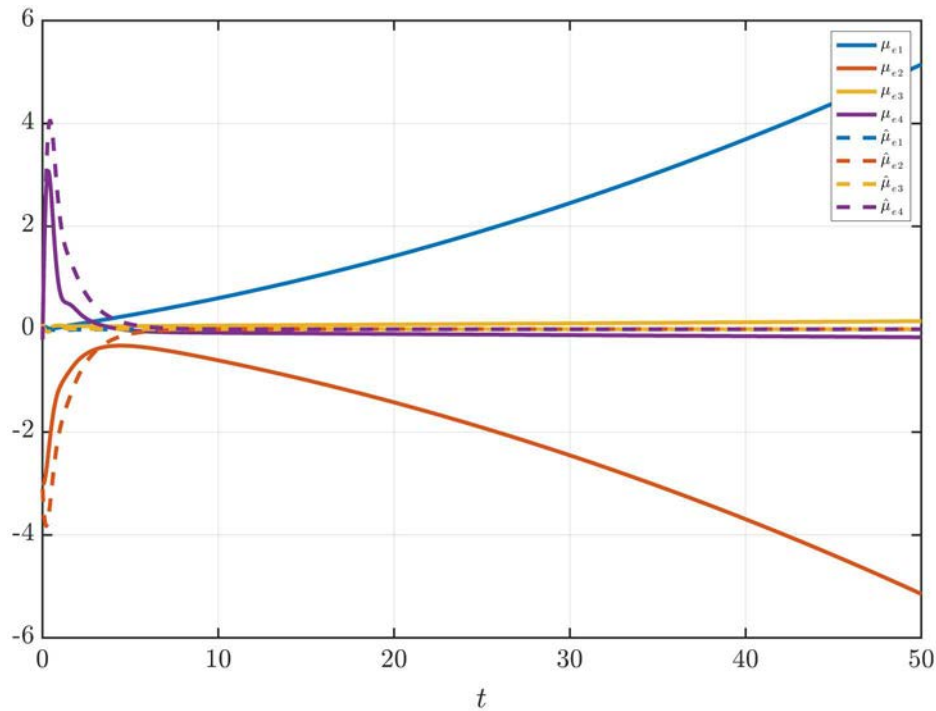


**Fig. 3.** Object movement trajectories when using the Defender, calculated according to (13).



**Fig. 4.** Object movement trajectories when using the Defender, calculated according to (18) and (13).

At the same time, the trajectories along which objects move with control calculated according to the equality (15) and the equations (13) in the plane of mathematical expectations of their coordinates are shown in Fig. 3 (for  $\gamma_0 = 4 \times 10^{-4} E_4$ ). As one can see, after some time, which is required for the convergence of the solution of the Riccati equation, the mathematical expectations of the coordinates of the objects practically do not differ. This is especially clearly seen in Fig. 4,



**Fig. 5.** Change of the vectors  $\mu_e$  and  $\hat{\mu}_e$  components over time.

on which solid lines represent the trajectories of objects calculated according to the equations (18), and dotted lines — according to (13).

We also show in Fig. 5 graphs of changes in the components of mathematical expectations of the vector of relative coordinates and its estimates.

*6.3. Simulation Results for the Successful Operation of the Defender in a Part of the Time Period*

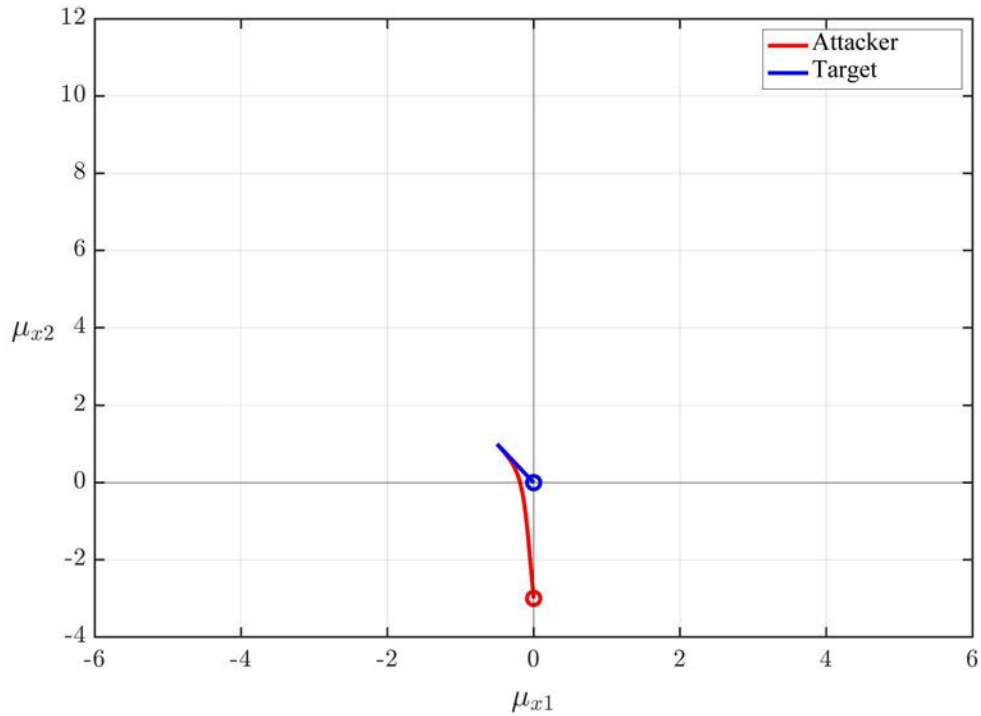
Now let's assume that at some point in time  $t = \tau$  the signal emitted by the Defender stopped influencing the choice of the Attacker's direction of movement. In practice, such a situation may be associated with a failure of the signal-generating element on board the Defender or a correction of the Attacker's guidance algorithms. To describe such a scenario, which we will call a scenario with correction of guidance algorithms, we introduce the concept of the moment (time) of interception of  $t^*$  according to the equality

$$t^* = \min\{t : |\mu_e(t)| < r_0\}.$$

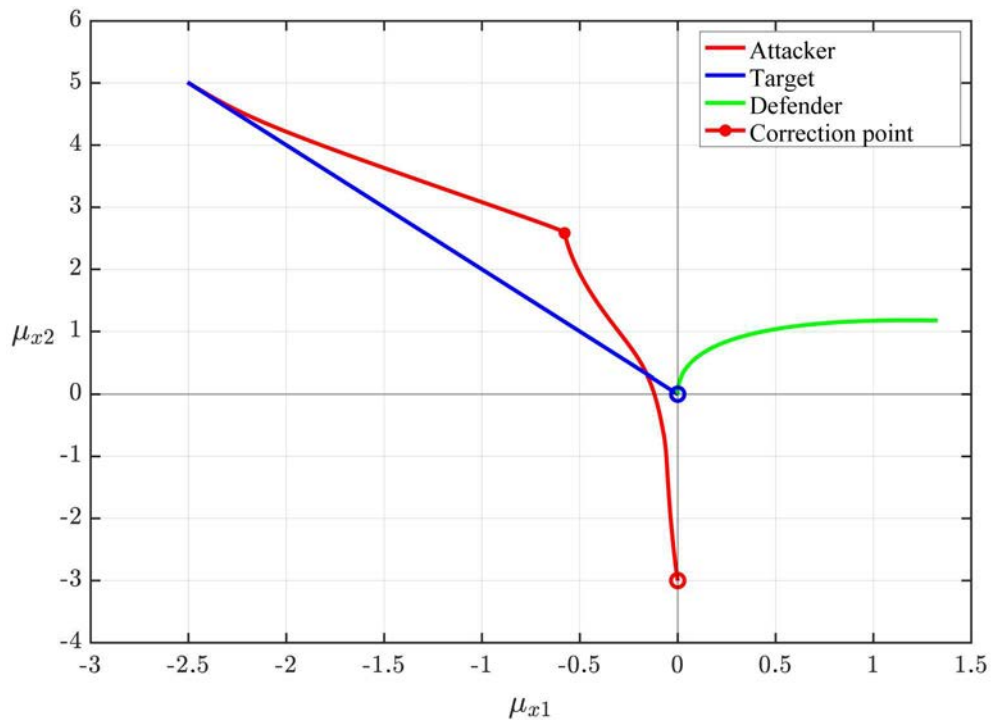
Let's choose  $r_0 = 10^{-4}$ . The trajectories of all objects in the plane of mathematical expectations up to the moment of interception, if the Defender is not in use, are shown in Fig. 6. The interception time is equal to  $t^* = 5.0$ .

Let's assume that when using the Defender, the correction of the guidance algorithms occurred at  $t = \tau = 20$ . The trajectories of objects moving in the plane of mathematical expectations up to the moment of interception in this case are shown in Fig. 7. The interception time is equal to  $t^* = 25.05$ .

A significant increase in interception time makes the use of even one Defender advisable if the Attacker's targeting algorithms involve correcting the work taking into account the use of the Defender by the Target. In practice, this allows you to provide the Target with a temporary reserve to perform an evasive maneuver, build a subsequent defense strategy, or release other Defenders.



**Fig. 6.** Object movement trajectories without using a Defender until the moment of interception.



**Fig. 7.** Object movement trajectories when using a Defender until the moment of interception. The moment of correction of the Attacker's targeting algorithms is indicated by a solid red dot.

*Remark 7.* The proposed method for constructing the Defender's trajectory also involves using it if the known equations are not the equations of motion of objects in the laboratory report system (i.e., the equations (1)), but the equations of motion of objects in the coordinate system associated with the Attacker (i.e., the equations (3)).

6.4. Comparison of Proposed and Alternative Strategies

Let's compare the value of the criterion on the optimal trajectory obtained in Section 6.2 and on the trajectories obtained using simpler heuristic strategies:

- (1) moving in the direction opposite to the Target;
- (2) maximum proximity to the Attacker.

The first strategy is described by the control law

$$\bar{v}(t) = -\kappa \left| \begin{pmatrix} x_T^0 \\ \end{pmatrix}_{3,4} \right|^{-1} \begin{pmatrix} x_T^0 \\ \end{pmatrix}_{3,4} = \text{const}, \tag{23}$$

and the second one, by –

$$\bar{v}(t) = -0.02 \left( (x_D(t))_{1,2} - (x_A(t))_{1,2} \right). \tag{24}$$

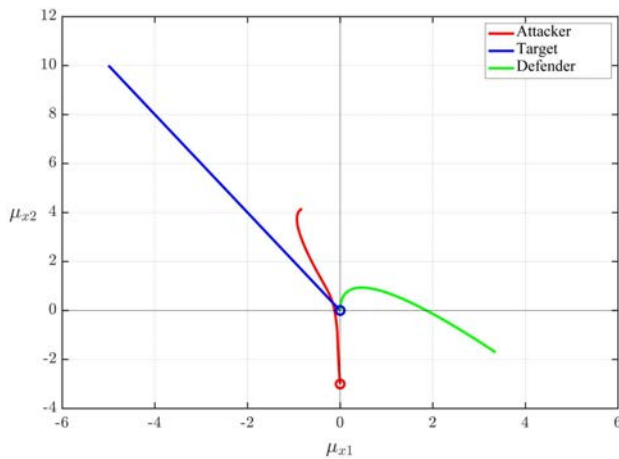
The trajectories of all objects in the plane of mathematical expectations when using the Defender of the law of control (23) are shown in Fig. 8. The value of the criterion  $J'$  in this case is

$$J'[\mu_3] = -51.6962 > J'[\mu_1].$$

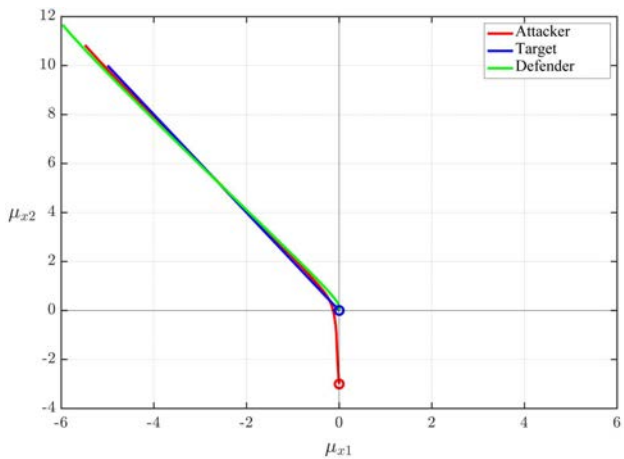
The trajectories of the movement of objects in the case of using the Defender of the law of control (24) are shown in Fig. 9. The value of the criterion  $J'$  in this case is

$$J'[\mu_4] = -0.93297 > J'[\mu_1].$$

Thus, the proposed optimal solution allows you to obtain a trajectory that significantly reduces the criterion compared to simpler strategies and, as a result, improves the tactical situation for the Target.



**Fig. 8.** Object movement trajectories when the Law of Control (23) is used by the Defender.



**Fig. 9.** Object movement trajectories when the Law of Control (24) is used by the Defender.

7. CONCLUSIONS

The work showed the effectiveness of using a Defender in the problem of distracting an Attacker from intercepting a target. The proposed model of using a Defender when influencing an Attacker's

receiving channel makes it possible to qualitatively model and predict changes in the Attacker's guidance algorithms, as well as formalize the formulation of problems for optimizing the Target's trajectory evasion from interception in a game with incomplete information of three players.

Further work will be aimed at formalizing and researching interception problems with more complex guidance algorithms, including those that take into account the possibility of using one or more Defenders and determining how to use them.

## 8. FUNDING

This work was supported in part by the Russian Science Foundation, project no. 23-19-00134.

## REFERENCES

1. Galyaev, A.A., Lysenko, P.V., and Rubinovich, E.Y., Optimal Stochastic Control in the Interception Problem of a Randomly Tacking Vehicle, *Mathematics*, 2021, vol. 19, no. 9, p. 2386.
2. Leitmann, G., A differential game of pursuit and evasion, *Int. J. Non-Linear Mech.*, 1969, vol. 4, no. 1, pp. 1–6.
3. Andreev, K.V. and Rubinovich, E.Ya., Moving observer trajectory control by angular measurements in tracking problem, *Autom. Remote Control*, 2016, vol. 77, no. 1, pp. 106—129.
4. Vassilyev, S.N., Galyaev, A.A., Zaletin, V.V., Kulakov, K.S., Silnikov, M.V., and Yakushenko, E.I., Joint Use of Mechatronic Systems to Organize Effective Counteraction to the Coordinated Action of Enemy Torpedoes, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2022, vol. 23, no. 4, pp. 197–208. <https://doi.org/10.17587/mau.23.197-208>
5. Buzikov, M.E., Vasiliev, S.N., Galyaev, A.A., et al., Model of group counteraction to homing system, *Proc. Conf. Control Mar. Syst. (UMS-2022)*, 2022, pp. 95–97.
6. Galyaev, A.A., Samokhin, A.S., and Samokhina, M.A., Modeling of the target's interception delay in an ADT game with one or two defenders, *Control Sci.*, 2024, no. 2, pp. 66–76.
7. Grigoriev, F.N., Kuznetsov, N.A., and Serebrovsky, A.P., *Upravlenie nablyudeniyami v avtomaticheskikh sistemakh* (Observation Control in Automatic Systems), Moscow: Nauka, 1986.
8. Kalman R.E., A New Approach to Linear Filtering and Prediction Problems, *Transactions of the ASME-Journal of Basic Engineering*, 1960, pp. 35–45.
9. Liptser, R.Sh. and Shiryaev, A.N., *Statistika sluchainykh protsessov* (Statistics of Random Processes), Moscow: Nauka, 1974.
10. Julier, S.J. and Uhlmann, J.K., Unscented Filtering and Nonlinear Estimation, *Proceedings of the IEEE*, 2004, vol. 92, no. 3, pp. 401–422.
11. Song, F., Li, Y., Cheng, W., et al., An Improved Kalman Filter Based on Long Short-Memory Recurrent Neural Network for Nonlinear Radar Target Tracking, *Wireless Communications and Mobile Computing*, 2022, pp. 10.
12. Coskun, H., Achilles, F., DiPietro, R., et al., Long Short-Term Memory Kalman Filters: Recurrent Neural Estimators for Pose Regularization, *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 5525–5533.
13. Girard, A.R. and Kabamba, P.T., Proportional Navigation: Optimal Homing and Optimal Evasion, *SIAM Review*, 2015, vol. 57, no. 4, pp. 611–624.
14. Potapov, A.P. and Rubinovich, E.Ya., Building a defender's 3D program path in an ADT game with incomplete a priori target information, *Control Sci.*, 2024, no. 5, pp. 52–62.
15. Potapov, A.P. and Galyaev, A.A., Countermeasures against the attacker's homing algorithm in a game of three players, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2024, vol. 25, no. 11, pp. 575–584. <https://doi.org/10.17587/mau.25.575-584>

16. Potapov, A.P. and Galyaev, A.A., Model of group counteraction to homing system, *Proc. 27th All-Russ. Sci.-Pract. Conf. Actual Probl. Prot. Secur.*, 2024, pp. 71–73.
17. Garcia Eloy, Casbeer David W., and Pachter Meir, The Complete Differential Game of Active Target Defense, *J.Optim. Theor. Appl.*, 2021, vol. 191, no. 2–3, pp. 675–699.
18. Akhil, K.R., Ghose, D., and Rao, S. Koteswara, Optimizing deployment of multiple decoys to enhance ship survivability, *2008 American Control Conference*, 2008.
19. Chen, Y.C. and Guo, Y.H., Optimal Combination Strategy for Two Swim-Out Acoustic Decoys to Countermeasure Acoustic Homing Torpedo, *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, 2017, pp. 1061–1065.
20. Polyak, B.T., Khlebnikov, M.V., and Rapoport, L.B., *Matematicheskaya teoriya avtomaticheskogo upravleniya* (Mathematical Theory of Automatic Control), Moscow: Lenand, 2019.
21. Ioffe, A.D. and Tikhomirov, V.M., *Teoriya ekstremal'nykh zadach* (Theory of Extremal Problems), Moscow: Nauka, 1974.
22. Chernousko, F.L. and Banichuk, N.V., *Variatsionnye zadachi mekhaniki i upravleniya. Chislennyye metody* (Variational Problems of Mechanics and Control. Numerical Methods), Moscow: Nauka, 1973.
23. Kwaternaak, H. and Sivan, R., *Linear optimal control system*, Wiley, 1979.

*This paper was recommended for publication by B.M. Miller, a member of the Editorial Board*

# A Distributed Graph Vertex Numbering Algorithm Combined with Breadth-First Search Tree Construction

O. P. Kuznetsov

*Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia*  
*e-mail: olpkuz@yandex.ru*

Received February 11, 2025

Revised May 29, 2025

Accepted September 12, 2025

**Abstract**—A new distributed algorithm for numbering the vertices of a rooted undirected graph is proposed. During the numbering process, it constructs a spanning tree that is also a breadth-first search tree. The complexity of this algorithm is estimated.

*Keywords:* undirected graph, distributed algorithm, vertex numbering, spanning tree, breadth-first search

**DOI:** 10.7868/S1608303225110061

## 1. INTRODUCTION

Problems of distributed computations on graphs have a long history. The first problem of this class was the firing squad synchronization problem, proposed by J. Myhill in 1957 and solved by V. Levenshtein [1] and F. Moore [2]. It considers a chain of identical automata that, after activating one of them, must pass to the same state in minimal time. Starting from the 1980s, many distributed algorithms for solving various graph problems emerged. Among them, note the problems of finding a minimum spanning tree (MST) [3–7] and a breadth-first search (BFS) tree [8–10]. Methods for solving these and many other problems were reviewed in [11–13].

The general scheme of distributed algorithms is as follows. Processors located at the vertices of a graph exchange messages in a synchronous or asynchronous mode. The computation process can be initiated in two possible ways:

- 1) Processors in all vertices start working simultaneously.
- 2) The algorithm starts with activating one vertex, called the graph root; the remaining vertices are activated after receiving messages.

Almost all these algorithms assume that the vertices are either numbered (see the surveys [11–13]) or have some unique identifiers [10]. A distributed vertex numbering algorithm based on the well-known Tarry’s algorithm for traversing graph edges [14] was described in [11].

This paper proposes a distributed algorithm for numbering the vertices of an undirected graph; during the numbering process, the algorithm constructs a spanning tree that is also a BFS tree and, accordingly, a tree of shortest paths from the root.

## 2. ALGORITHM DESCRIPTION

Consider a connected undirected graph with  $n$  vertices,  $m$  edges, and a specified initial vertex (root), denoted by  $v_0$ . Exactly such graphs, called rooted, will be studied below. Each vertex contains a processor that can execute one of the local algorithms, depending on the state of this vertex. All vertices can be in one of four states, indicated by colors: white, gray, black, and red.

A particular local algorithm corresponds to each state (color). From this point onwards, the vertex processor will be identified with the vertex itself: speaking of the actions of a vertex, we mean the actions of its processor.

Each vertex has an ordered list of incident edges. Edges have an attribute that takes one of the four values: incoming (upper), black (lower), dotted (chord), and red (see the details below).

Vertices can exchange messages of two types.

Type 1 has the form  $(i, c)$ , where  $i$  is a number, i.e.,  $i \in \{1, \dots, n-1\}$ , and  $c$  is a color, i.e.,  $c \in \{\text{black}, \text{red}\}$ . For brevity, messages of this type will be specified as “black (red)  $i$ ” or “black (red) number”. By default, we assume that the number is black, and the message is of type 1: the expression “send  $i$ ” means “send message  $(i, \text{black})$ ”.

Type 2 has the form  $(i, j)$ , where  $i$  and  $j$  are black numbers; this type of messages arises when a vertex, responding to an attempt to assign number  $i+1$  to it, reports its number  $j$ .

In both types of messages,  $i$  is the last number assigned. It will be called the current  $i$  or current number.

A vertex is white if:

- It is not numbered.
- It waits for messages to receive its number.

After that, a vertex becomes either gray (when, in addition to the incoming edge, it has other incident edges) or red (when it has no such edges, i.e., it is pendant).

A vertex is gray if:

- It is numbered but has unnumbered neighbors.
- It waits for a type 1 message to start numbering the adjacent vertices or for a type 2 message to label the edge via which this message has arrived as a chord.

A vertex is black if it is numbered, and all its neighbors are also numbered. Upon receiving the current number, a black vertex transmits it to the next level.

A vertex is red if the vertices reachable from it at the next levels are either absent or numbered. In both cases, this is reported upward: red  $i$  is sent via the incoming edge. The conditions of transition to the red color will be described in detail below.

The idea of coloring vertices white, gray, and black is borrowed from [15], albeit with the following modifications due to the distributed nature of computations: a particular algorithm corresponds to each color; moreover, the red color is added to report the end of the numbering process on a certain branch.

The vertex numbering algorithm proposed here is based on breadth-first search in a graph [15] and is sequential: only one vertex is active at any time instant. Sending a message essentially means transferring control: receiving a message activates the vertex (starts its algorithm, depending on the current color of the vertex), whereas sending a message terminates activity. Due to the sequential nature of the algorithm, during each activity period, a vertex can send only one message to one address.

As will be shown below, the algorithm constructs a spanning tree consisting of incoming edges; each vertex of this tree is at a minimum distance from the root. Therefore, the algorithm can be described using two concepts, namely, level and branch. A branch of a vertex is a subtree with this vertex as the root; the  $i$ th level is the set of vertices at distance  $i$  from the root of the original graph. Breadth-first traversal means that the vertices of subsequent levels are not numbered until the vertices of the current level are all numbered; thus, if  $i < j$ , the number of any vertex at the  $i$ th level will be less than the number of any vertex at the  $j$ th level.

Recall that a chord is an edge not contained in a spanning tree.

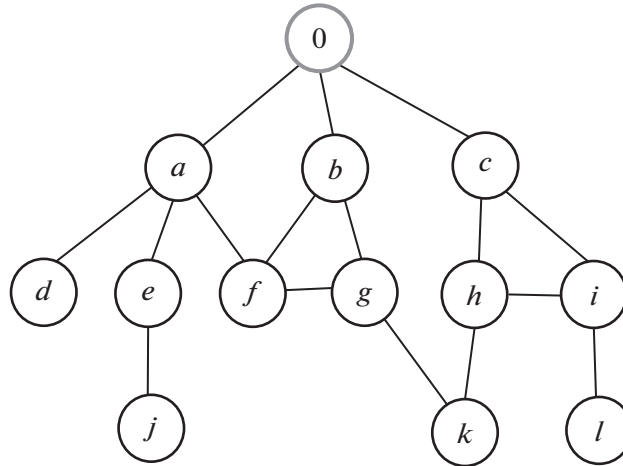


Fig. 1.

As an example, we will take the graph in Fig. 1 at various stages of the algorithm. The letters labeling the vertices are introduced for convenient description and reading. The letters of the vertices are unknown to their neighbors and to the root as well; therefore, they are not the identifiers of vertices in the conventional sense and are not used in the algorithm's operation.

In black-and-white graphics, vertex colors will be depicted as follows. White (unnumbered) vertices are those labeled by letters. Gray vertices have a regular (thin) contour. Black vertices are shaded. Red vertices and edges have a thicker contour. The root is depicted at the top, so the expression "send upward" means "send toward the root."

#### The scheme of the general vertex numbering algorithm

Initial state:

All vertices, except the root, are white; the root has no color; all edges are black. The set of black edges (SBE) is ordered, so it makes sense to speak of the first edge of the SBE.

Start:

The root assigns number 0 to itself.

The root forms a queue from the SBE and sends 0 via the first edge of the queue.

Subsequently, it executes the local algorithm of the root, which is to manage the numbering process of levels. The general process runs as follows.

In the first cycle, the root initiates the numbering process of level 1 vertices by sending 0 via the first edge of the queue. At the end of this cycle, the queue becomes empty; this means that the first level is numbered, and its vertices have become gray (and the pendant vertices of the first level have become red). The  $k$ th cycle ends with the following results: all vertices of the  $k$ th level have become gray or red, all vertices of the previous levels have become black or red, and a message with the current number has arrived at vertex 0 via the last edge of the queue.

In the  $(k + 1)$ th cycle, upon receiving this current number, the root re-forms the queue from the SBE and sends a message with the black number via the first edge of the queue. Upon passing through  $k - 1$  black vertices, this message arrives at a gray vertex  $v$  of the  $k$ th level; upon receiving this message, the latter vertex numbers its neighbors connected to  $v$  by black edges (included in the SBE); upon finishing the numbering process, vertex  $v$  sends the last number upward via the incoming edge  $(u, v)$  and becomes black. If a neighbor vertex  $w$  has been already numbered, edge  $(v, w)$  is labeled as a chord. If the SBE of vertex  $v$  becomes empty, this vertex becomes red and sends a red number upward to its black vertex  $u$ , which makes edge  $(u, v)$  red. As soon as all

lower edges of a black vertex have become red, this vertex becomes red itself and sends the red number upward. The algorithm ends when all edges incident to the root become red.

By default, the expression “send message” means the end of the algorithm and waiting for the next message.

Now we describe particular local algorithms: that of the root and those corresponding to different vertex states (colors).

**The local algorithm of the root**

- 1a) Assign number 0 to itself.
- 1b) Start the algorithm for numbering neighbors with current  $i$ .
- 1c) Form a queue from the SBE.
- 1d) Send  $i$  via the first edge of the queue; wait for a message.
- 2) If black  $i$  has been received via the current edge of the queue (the next level of this branch is numbered), then:
  - 2a) remove this edge from the queue;
  - 2b) if the remaining queue is non-empty, send  $i$  via the first edge of the queue; else (the next level of the graph is numbered):
  - 2c) form a queue from the black edges;
  - 2d) send  $i$  via the first edge of the queue.
- 3) Else (a red number  $i$  has been received via the current edge of the queue; this means the absence of unnumbered vertices on this branch):
  - 3a) make this edge red and remove it from the queue;
  - 3b) if the queue is non-empty, send  $i$  via the first edge;
  - 3c) if the queue is empty and the SBE is non-empty, go to Step 2c);
  - 3d) if the SBE is empty (all edges are red), **end of the general algorithm.**

At the first level, there are no chords, so the root receives only type 1 messages.

**The local algorithm of a white vertex**

A white vertex has no number. All its edges are black.

Upon receiving a type 1 message with number  $i$  :

- 1) Assign number  $i + 1$  to itself.
- 2) Label the edge via which  $i$  has arrived as incoming.
- 3) If the SBE is non-empty, then:
  - 3a) send black  $i + 1$  upward via the incoming edge;
  - 3b) become gray; end of the algorithm.
- 4) Else:
  - 4a) send red  $i + 1$  upward via the incoming edge;
  - 4b) become red; end of the algorithm.

Upon receiving its number and becoming gray, a white vertex “does not know” the status of the numbering process of its level (completed or not); therefore, it does not number its neighbors at the next level, but only reports its number upward via the incoming edge to its “upper” vertex so that the latter continues numbering neighbors. But a white vertex may be pendant, i.e., have only one neighbor connected to it by an incoming edge. In this case (see Step 4) of the algorithm), it skips the gray and black stages and immediately becomes red.

**The local algorithm of a gray vertex**

- 1) If a message  $(x, y)$  of type 1 has been received via the incoming edge, then:

- 1a) Form a queue from the SBE.
- 1b) Send  $i$  via the first edge of the queue.
- 1c) If black  $i + 1$  is returned via this edge, then:
  - 1c1) remove this edge from the queue;
  - 1c2) if the queue is non-empty, go to Step 1b) with  $i + 1$ ;  
else:
  - 1c3) send  $i + 1$  via the incoming edge;
  - 1c4) become black; end of the local algorithm.
- 1d) If red  $i + 1$  is returned via this edge, then:
  - 1d1) make this edge red;
  - 1d2) remove it from the queue;
  - 1d3) if the queue is non-empty, go to Step 1b) with  $i + 1$ ;  
else
  - 1d4) if the SBE is non-empty, go to Step 1c3);  
else (all edges are red):
  - 1d5) send red  $i$  upward;
  - 1d6) become red.
- 2) Else (a message  $(x, y)$  of type 1 has been received via a black edge):
  - 2a) make this edge dashed (this edge is a chord);
  - 2b) send via this edge a type 2 message  $(i, j)$ , where  $j$  is the number of this  
vertex;
  - 2c) if the SBE is non-empty, end of the algorithm;
  - 2d) else:
    - 2d1) send red  $i$  upward;
    - 2d2) become red; end of the algorithm.

Step 2) corresponds to the situation when the vertex is already numbered and is the end of a chord. As a result of Step 2b), the SBE may become empty, and then 2d) the vertex becomes red. Case 2d) violates the sequential nature of the algorithm (see Example 4 below). The arrival of a number via a black edge means that the numbering process runs on another branch  $q$ , and the emptying of the SBE causes the parallel transmission of a red number upward via this branch  $p$ , which may stop at any vertex of the branch. When the red number reaches the root (this will happen when all vertices of branch  $p$  become red), the root receives two current numbers: from branch  $p$ , numbered earlier, and from branch  $q$ , where the last numbering has occurred. In this case: (a) the incoming edge of branch  $p$  becomes red and is removed from the root's SBE; (b) the current number remains the one received from branch  $q$  since it cannot be less than the number received from branch  $p$ .

*Example 1* (Fig. 2). Vertex 1 finished numbering its lower neighbors (vertices 5 and 6 became gray and vertex 4 (pendant) red), sent the last number 6 upward, and became black. Upon receiving number 6, the root removes edge  $(0, 1)$  from the queue and sends number 6 to gray vertex 2.

*Example 2* (Fig. 3). Vertex 2 (gray) starts numbering neighbors and attempts to number gray vertex 6, which has already been numbered by vertex 1. Step 2) of the gray algorithm is triggered for vertex 6: it labels edge  $(2, 6)$  as dashed (chord) and sends a type 2 message  $(6, 6)$  via this edge to vertex 2. Vertex 2 also labels edge  $(2, 6)$  as dashed and sends 6 to the next vertex, which receives number 7 and sends it via the incoming edge to vertex 2. The numbering process of neighbors of vertex 2 is now complete (the queue is empty); it reports this upward and becomes black.

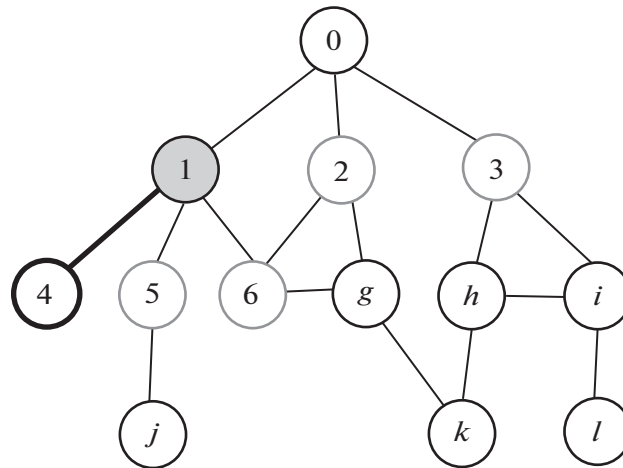


Fig. 2.

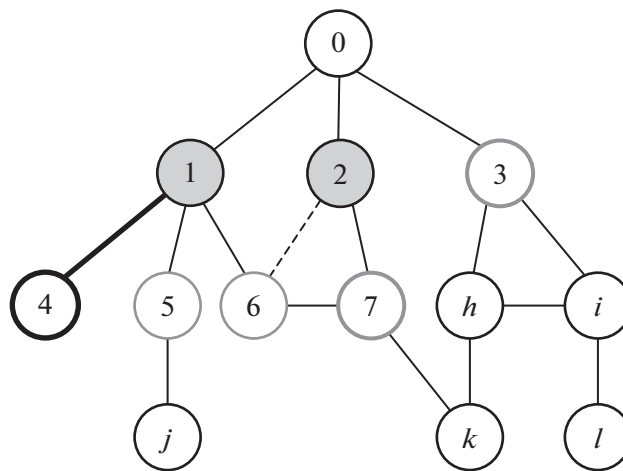


Fig. 3.

**The local algorithm of a black vertex**

- 1) If a black number  $i$  is received via the incoming edge, then:
  - 1a) form a queue from the SBE;
  - 1b) send  $i$  via the first edge of the queue.
- 2) If a black number  $i$  is received downward, then:
  - 2a) remove this edge from the queue;
  - 2b) if the queue is non-empty, go to Step 1b) with  $i$ .

Else send  $i$  via the incoming edge; end of the local algorithm.
- 3) If a red number  $i$  is received downward, then:
  - 3a) make this edge red and remove it from the queue;
  - 3b) if the queue is non-empty, send black  $i$  via the first edge;
  - 3c) if the queue is empty, then:
 

if all edges are red, become red and send red  $i$  upward; end of the local algorithm.

Else send black  $i$  upward; end of the local algorithm.

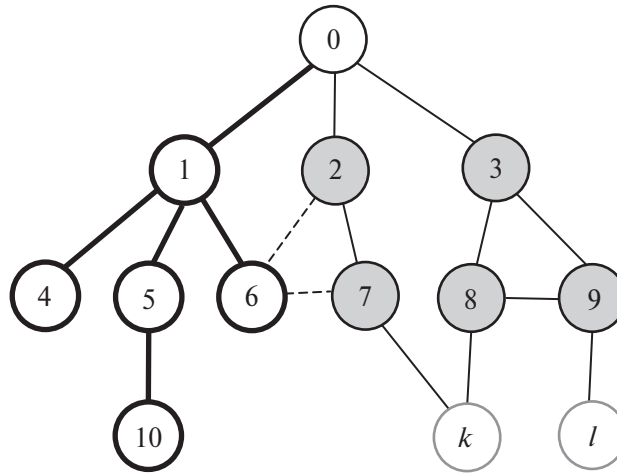


Fig. 4.

*Example 3.* Figure 4 shows a stage of the general algorithm where the numbering process of level 3 vertices of branch 2 is completed. Consider this process, which leads to the state demonstrated in the figure. The root sent the current number 9 to vertex 1. Vertex 1 and edge (1, 4) became red earlier, during the numbering process of level 2 (see Example 1), so edge (1, 4) is no longer in the SBE of vertex 1. Vertex 5 numbered vertex 10; the latter detected itself to be red and sent red number 10 to vertex 5, which also became red and sent red number 10 to vertex 1, making edge (1, 5) red. Then vertex 1 sends number 10 to vertex 6, which attempts to number vertex 7, receives a type 2 message from it, labels edge (6, 7) as a chord, becomes red (see Step 2d) of the gray algorithm), and sends red number 10 to vertex 1. Vertex 1 makes edge (1, 6) red, detects itself to have no black edges, becomes red, and sends red number 10 to the root. Edge (0,1) becomes red, i.e., is removed from the root's SBE, so branch 1 does not participate in further numbering cycles.

Note the following. The levels of the chord ends may either coincide (such chords will be called horizontal; see chord (6, 7) as an example) or differ at most by 1 (vertical chords; see chord (2, 6) as an example). Indeed, let vertex  $x$  on an edge  $(x, y)$  be numbered and be at level  $k$ , and let  $y$  be not numbered yet. Obviously, at this time instant,  $(x, y)$  is in the SBE of  $x$ . Therefore, during the numbering process of the  $(k + 1)$ th level, vertex  $x$  either numbers  $y$  or detects that  $y$  has already been numbered; in the latter case,  $(x, y)$  is labeled as a chord. In any case, as illustrated by the example of edge (6, 7), a horizontal chord of level  $k$  is detected as a chord only during the numbering process of the  $(k + 1)$ th level. As we will see below, this can affect the total numbering time.

*Example 4* (continuation of Example 3). Upon receiving number 10 from vertex 1, the root starts numbering branch 2, which ends with the arrival of black number 11 at the root, yielding no new red vertices: Step 3) is triggered in the algorithm of white vertex 11 (the SBE still contains edge (8, 11)), and it becomes not red but gray. After this, the numbering process of level 3 of branch 3 begins, during which vertex 8 detects chords (8, 9) and (8, 11). When vertex 8 attempts to number the already numbered (i.e., gray) vertex 11, Step 2) of the gray algorithm is triggered for this vertex, it becomes red and sends its red number upward. Thus, for some time, **two parallel processes** will run: along branch 2, red number 11 is transmitted to the root, and along branch 3, numbering continues. In general, two options are possible: a) the processes run along different branches; b) the processes run along different sub-branches of one branch. In the first case, they converge at the root; in the second, they arrive at some vertex of one branch, which may either change its color (if its SBE becomes empty) or not.

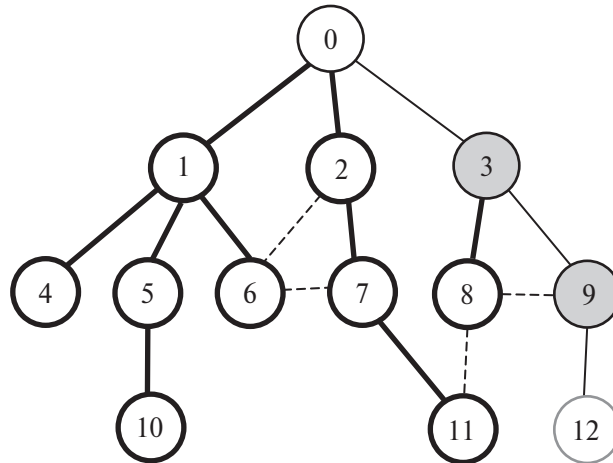


Fig. 5.

Figure 5 shows the time instant when the numbering process is complete. Vertex 12 has just received its number. At the next time instant, it will become red; and through vertices 9 and 3, red number 12 will arrive at the root, which ends the numbering process.

*Example 5.* Now suppose that a horizontal chord (11, 12) is added to the graph in Fig. 5. As noted at the end of Example 3, a horizontal chord of level  $k$  will be detected as a chord only during the numbering process of the  $k + 1$ th level. Therefore, the process will run not as described in Example 4: vertices 11 and 12 do not become red; after completing the numbering of level 3, black number 12 arrives at the root, and the root starts numbering level 4, unaware of its non-existence. Vertices 11 and 12 become red only in the fourth cycle, and the corresponding messages are sent in parallel along branches (11, 7, 2, 0) and (12, 9, 3, 0).

### 3. MAIN RESULTS. ALGORITHM COMPLEXITY ESTIMATION

**Theorem 1.** *The incoming edges of an original graph  $G$ , labeled by the algorithm, form a tree that is (a) spanning and (b) a tree of shortest paths from the root, i.e., a BFS tree.*

Consider the graph  $G^*$  formed by the incoming edges. Each vertex has only one incoming edge. This follows from the fact that an edge is labeled as incoming only when the vertex is white, which then becomes gray. Moreover, if an edge  $(u, v)$  is incoming for  $v$ , then the number of  $u$  is less than the number of  $v$ : vertex  $v$  has received its number via the edge  $(u, v)$ , and vertex  $u$  has already been numbered by that time. Therefore, on any path from  $v$  via incoming edges, the vertex numbers decrease. Hence, (1) a path of incoming edges from any vertex  $v$  cannot be a loop, and (2) it can end only at the root, because only the root has no incoming edge. Thus, the graph  $G^*$  is connected (any two vertices are either on the same path to the root or connected by two paths leading to the root) and contains no loops, thereby representing a tree. Since each vertex has an incoming edge, the tree  $G^*$  contains all vertices of the original graph, i.e., it is spanning.

Now we prove that for any vertex  $v$ , its path to the root  $v_0$  in the graph  $G^*$  is the shortest one in the graph  $G$ . The proof is by induction on the cycles of the numbering process.

The 1st cycle is to number the root's neighbors. Obviously, after its completion, all neighbors of the root become gray, they are numbered, the edges connecting them to the root are incoming, and their distance to the root is 1. Suppose that they form level 1. Clearly, there are no other vertices with unit distance to the root.

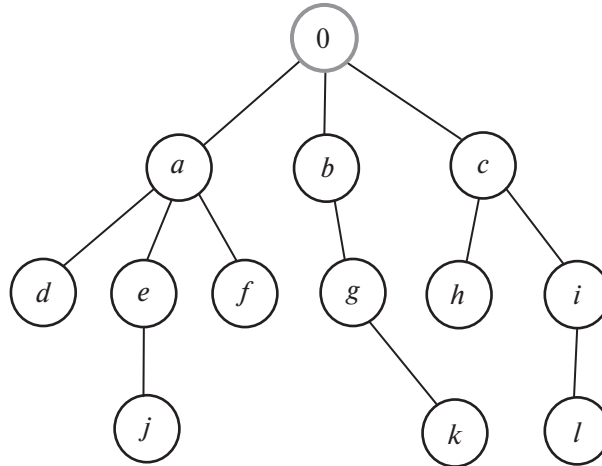


Fig. 6.

Next, assume that the  $k$ th numbering cycle has been completed, resulting in new gray vertices whose distance to the root is  $k$ , and there are no other vertices with distance  $k$  to the root. These vertices form level  $k$ . Then all vertices with distance  $k + 1$  to the root are neighbors of level  $k$  vertices; consequently, the former vertices will be numbered by the latter vertices in the  $k + 1$ th cycle, becoming gray and forming level  $k + 1$ . The edges via which they have been numbered will become incoming for them and enter the tree  $G^*$ . Thus, for any  $k$ , a vertex with distance  $k$  to the root will enter level  $k$  of the tree  $G^*$ , proving that the graph  $G^*$  composed of the incoming edges of the original graph  $G$  is a tree of shortest paths to the root, i.e., a BFS tree.

Thus, the algorithm produces the following results:

- (1) All vertices of the graph are numbered, and the total number of vertices is found.
- (2) A spanning tree of the graph is constructed from the incoming edges; the chords of the graph are found, thereby determining its cyclomatic number.
- (3) This spanning tree is a BFS tree.

Thus, the algorithm simultaneously solves two problems: distributed vertex numbering and distributed construction of a BFS tree. Therefore, this algorithm will be called the NBFS algorithm.

Note that the local parallelism, arising sometimes during the operation of the NBFS algorithm (see Example 4), does not require synchronization: the final state of the vertex, where two parallel messages arrive, is independent of the order of their arrival.

The spanning tree constructed is not unique and depends on the order of edges in the queues formed by the local algorithms.

For the current example, the BFS tree is shown in Fig. 6. If in the second cycle the queue at the root had the form  $bac$ , vertex  $f$  would receive number 4, and the edge  $(b, f)$  would not be a chord.

**Complexity.** Henceforth, by the complexity of the NBFS algorithm we mean its communication complexity, i.e., the number of messages generated during the operation of this algorithm. Let us introduce the following notation:

$C_{NBFS}(n)$  is the complexity of the NBFS algorithm for rooted graphs with  $n$  vertices;

$C_{NBFS}(G)$  is the complexity of the NBFS algorithm for a particular graph  $G$ ;

$e(G)$  is the eccentricity of the root of the graph  $G$ , i.e., the maximum of the distances from  $v_0$  to the other vertices;

$\gamma(G)$  is the cyclomatic number (and, accordingly, the number of chords) of the graph  $G$ ;

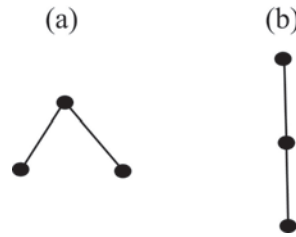


Fig. 7.

$G_{BFS}$  is the spanning tree of the graph  $G$ . From this point onwards, we will simply write  $e$  and  $\gamma$ .

Any numbering algorithm for the vertices of a rooted graph must sequentially traverse all vertices starting from the root. A distributed algorithm must traverse all edges: in the absence of initial information about global properties of the graph, there is no guarantee of visiting all vertices until visiting all edges. The well-known Tarry’s algorithm [14] for traversing all edges is based on splitting each edge into two “half-edges,” resulting in an Euler graph where each half-edge can be traversed once; this is equivalent to traversing the original edge twice in different directions, with the traversal necessarily ending at the root. Therefore, for the distributed Tarry algorithm [11], the communication complexity equals the number of visits to each edge, i.e.,  $2m$ . Obviously, due to the sequential nature of any numbering algorithm, the estimate  $O(m)$  cannot be lowered.

Note that there are  $e$  levels in  $G$  and  $G_{BFS}$ ; hence, they have  $e$  numbering cycles as well. Obviously, among all rooted undirected graphs with  $n$  vertices, the chain  $Ch_n$  has the greatest eccentricity, i.e., a connected graph with  $n$  vertices where two end vertices have degree 1, one representing the root, and all other vertices have degree 2. In this case,

$$e(Ch_n) = n - 1. \tag{1}$$

Let us estimate the complexity of the chain  $Ch_n$ .

**Lemma 1.**

$$C_{NBFS}(Ch_n) = (n - 1)n. \tag{2}$$

Due to (1), the chain  $Ch_n$  has  $n - 1$  levels; therefore, its numbering requires  $n - 1$  cycles. The  $k$ th cycle ( $k = 1, \dots, n - 1$ ) involves  $k + 1$  vertices (including the root). The messages they generate can be represented as a chain of length  $k + 1$  of the form  $12 \dots 21$ , where the  $i$ th element of the chain ( $i \leq k + 1$ ) corresponds to the number of messages generated by the level  $(i - 1)$  vertex. Indeed, the root and the last vertex of the chain generate one message each, and the remaining vertices generate two each: one to number the next vertex, and the other to report the current number towards the root. The total number of messages in one such chain is  $2k$ ; accordingly, the total number of messages generated by the numbering process of a chain of length  $n$  is the sum of messages over all  $n - 1$  cycles, i.e., twice the sum of the corresponding arithmetic progression:  $2 \sum_{k=1}^{n-1} k = (n - 1)n$ .

Since the  $k$ th numbering cycle involves all black edges of the first  $k$  levels in message exchange, obviously, eccentricity significantly affects the value of  $C_{NBFS}$ . A more precise assertion is as follows.

**Lemma 2.** *Among all rooted trees with  $n$  vertices, the tree with the greatest eccentricity—the chain  $Ch_n$ —has the highest NBFS complexity.*

We prove this result by induction.

The minimum  $n$  for which this statement makes sense is 3. For  $n = 3$ , two trees are possible (Fig. 7).

Tree 7a has one level and therefore requires one numbering cycle, which corresponds to the chain 211 (two messages from the root and one message from each of the other vertices), i.e., 4 messages. Tree 7b is a chain with two levels and two numbering cycles: the first cycle corresponds to chain 11 and the second cycle to chain 121; in total,  $2 + 4 = 6$  messages. Thus, the value of  $C_{NBFS}$  is greater for tree 7b than for tree 7a, so the lemma is valid for  $n = 3$ .

Assume now that the desired assertion is true for all trees with a number of vertices not exceeding  $k$ , i.e., the complexity  $C_{NBFS}(Ch_k) = (k - 1)k$  is maximum for all trees with  $k$  vertices. Based on this inductive hypothesis, we will establish the lemma's validity for  $k + 1$ : the chain's complexity  $C_{NBFS}(Ch_{k+1}) = (k + 1)k$  is maximum for all trees with  $k + 1$  vertices.

Any tree with  $k + 1$  vertices can be obtained from some tree  $G_k$  with  $k$  vertices by attaching an edge  $(x, y)$  so that vertex  $x$  is identified with some vertex of  $G_k$  and vertex  $y$  becomes pendant in the new tree  $G_{k+1}$ . Two options are possible here.

(a)  $G_k$  is a chain. Attaching an edge  $(x, y)$  to the chain's end yields the chain  $Ch_{k+1}$  with complexity  $(k + 1)k$ . If an edge  $(x, y)$  is attached to any other level  $i$  vertex of  $G_k$  ( $i < k$ ), the level of the resulting tree  $G_{k+1}$  will not change; hence, the  $(k + 1)$ th numbering cycle will be absent, and during the numbering of the  $(i + 1)$ th level, two new messages will appear, from vertex  $y$  and back; and  $y$  will become red and will not generate further messages. Thus,  $C_{NBFS}(G_{k+1}) = C_{NBFS}(Ch_k) + 2 < C_{NBFS}(Ch_{k+1})$  and, consequently, the chain  $Ch_{k+1}$  has the maximum complexity in the class of all trees with  $k + 1$  vertices obtained by attaching an edge to the chain  $Ch_k$ .

(b)  $G_k$  is not a chain. Then  $e(G_k) \leq k - 1$  and, by the inductive hypothesis,

$$C_{NBFS}(G_k) \leq (k - 1)k. \quad (3)$$

Let an edge  $(x, y)$  be attached to a pendant vertex  $v$  at level  $i \leq k - 1$ . Any pendant vertex is the end of some chain attached at the other end to a vertex with a degree above 2, and its length does not exceed  $k - 1$ . During the numbering process of the tree  $G_k$ , vertex  $v$  (and the entire chain) becomes red at the  $i$ th numbering cycle and does not participate in further cycles. In the new tree  $G_{k+1}$ , this chain is lengthened by one edge, and its former end  $v$  is replaced by vertex  $y$  at level  $i + 1 \leq k$ . Hence, this chain will participate in the  $(i + 1)$ th numbering cycle, where at most  $2(i + 1) \leq 2k$  messages will be added to the complexity  $C_{NBFS}(G_k)$ . Therefore,  $C_{NBFS}(G_{k+1}) \leq C_{NBFS}(G_k) + 2k$ , and, using (3), we have  $C_{NBFS}(G_{k+1}) \leq (k - 1)k + 2k = (k + 1)k = C_{NBFS}(Ch_{k+1})$ . Thus, in case (b) as well, the chain's complexity turns out to be maximum, which finally proves Lemma 2.

Consider now the NBFS complexity of an arbitrary rooted undirected graph  $G$  with  $n$  vertices and  $m$  edges. The cyclomatic number (equivalently, the number of chords) of such a graph is calculated by the well-known formula  $\gamma = m - n + 1$ .

**Theorem 2.**

$$C_{NBFS}(n) = \mathbf{O}(n^2 - n). \quad (4)$$

The complexity  $C_{NBFS}(G)$  of a particular graph  $G$  can be represented as the sum of two terms:

$$C_{NBFS}(G) = C_{NBFS}(G_{BFS}) + C_\gamma(G). \quad (5)$$

The first term is the complexity of the spanning tree of the graph  $G$ . It is determined by messages related to the numbering itself (assigning a number and reporting the current number back); they are transmitted only via incoming edges, i.e., through the future spanning tree. According to Lemmas 1 and 2,  $C_{NBFS}(G_{BFS}) \leq n^2 - n$ .

The second term is the number of messages generated when detecting chords. In general, detecting a chord  $(x, y)$  is associated with two messages: vertex  $x$  attempts to number vertex  $y$ ;

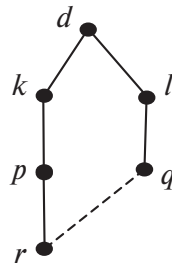


Fig. 8.

vertex  $y$  reports that it is already numbered. After this, the edge  $(x, y)$  is removed from the SBE of vertices  $x$  and  $y$ ; no subsequent messages are transmitted via it. However, a more complex situation is possible, see Example 4. We study it in detail.

Figure 8 shows a fragment of some graph where vertices  $p, q$  are at the  $i$ th level and the  $(i + 1)$ th numbering cycle is in progress. Along branch  $dkpr$  it has already been completed, vertex  $r$  has received its number and sent it back to vertex  $p$ ; meanwhile, the edge  $(r, q)$  remains in the SBE of vertex  $r$ . After its activation, vertex  $q$  attempts to number vertex  $r$  and receives a message from this vertex that the latter has already been numbered. Then, both vertices remove the edge  $(r, q)$  from their SBE. If the resulting SBE of vertex  $r$  is empty (possible only if this vertex is pendant in the future spanning tree), then it becomes red and sends another message to vertex  $p$ . Upon receiving this message, vertex  $p$  may also become red, which will generate a message from it to vertex  $k$ , etc. Thus, simultaneously with the numbering process in branch  $dlq$ , an additional chain of messages will arise in branch  $dkpr$ , contributing to the term  $C_\gamma(G)$ . All these messages make some edges red, causing their removal from the SBE; therefore, only one such message can be transmitted via each edge. Hence, their total number does not exceed  $m$  (the total number of edges in the graph); as is well known,  $m \leq \frac{n^2-n}{2}$ .

Another option in this situation has been described in Example 5, where a horizontal chord connects two pendant vertices of the tree  $G_{BFS}$ . In this case, an additional numbering cycle arises. The messages of this cycle will be transmitted, at most twice, via all incoming edges of the graph. In this case, the chord's contribution to  $C_\gamma(G)$  does not exceed  $2m \leq n^2 - n$ .

Thus, both terms are smaller than or equal to  $n^2 - n$ , which completes the proof of Theorem 2.

For different classes of graphs, the ratio between the two terms in (5) can vary significantly. In particular, in sparse graphs (graphs with a small number of chords), the main contribution to  $C_{NBFS}$  is made by the complexity of the spanning tree, which, in turn, is determined by the root's eccentricity. Therefore, even within the same graph, the complexity will change appreciably depending on the choice of the root; as naturally expected, it takes the minimum value when the root is the graph's center and the maximum value when the root is the end of one of the diameters. According to Lemmas 1 and 2, the upper bound is achieved when the graph is a chain with the root at one of its ends.

The other extreme case is the complete graph  $K_n$ , in which  $e = 1$  for any choice of root and there are  $m = \frac{n^2-n}{2}$  edges; of these,  $n - 1$  are incoming edges, and the rest are chords:  $\gamma = \frac{n^2-n}{2} - n + 1$ . From  $e = 1$  it follows that all chords are horizontal. Due to these formulas, the main contribution to  $C_{NBFS}(K_n)$  is made by the value of  $C_\gamma(G)$ , and the upper bound is reached by the number of chords.

The operation of the NBFS algorithm on the graph  $K_n$  consists of two numbering cycles. In the first cycle, all vertices receive their numbers, but owing to the horizontality of all chords, no vertex becomes red; therefore, the root does not know that the numbering is complete and starts the second cycle, in which all chords are detected.

## 4. CONCLUSIONS

The numbering problem has universal significance for distributed algorithms. Almost all distributed algorithms on graphs assume that vertex numbering has already been performed. Moreover, with the uniform numbering for all vertices, the root can request from each vertex the lists of incident edges and thus reconstruct a complete description of the graph.

The efficiency of the NBFS algorithm can be judged by comparing it with Tarry's algorithm. For "dense" graphs, close in the number of edges to  $K_n$ , the complexity estimates of both algorithms are of the same order, and consequently, the NBFS algorithm is preferable since it simultaneously constructs a BFS tree. Conversely, for sparse graphs, close to trees, the NBFS algorithm should not be used: firstly, for such graphs, it is much inferior to Tarry's algorithm in terms of complexity (obviously, a chain can be numbered in one traversal); and secondly, the main advantage of the NBFS algorithm (the simultaneous construction of a BFS tree during numbering) does not work here since any tree is itself a BFS tree.

Among potential applications of the NBFS algorithm, in addition to communication networks as a usual application of distributed algorithms, we mention swarm robotics. One possible scenario is as follows: a group of robots starts a patrolling mission in an area, having a complete graph of connections within the group. By the end of the mission, some connections are disrupted, and the leader needs to reconstruct the graph of remaining connections. The percentage of disrupted connections is small, and the graph of remaining connections is close to complete; therefore, the NBFS algorithm will be effective.

Another algorithm that constructs a BFS tree and numbers the vertices was described in [16]. It first constructs a BFS tree and then performs vertex numbering on this tree using a tree traversal similar to Tarry's algorithm, with the significant difference that  $m = n - 1$  for trees. Combined with the use of parallelism (in [16], vertices send messages to all their neighbors simultaneously), this gives a time complexity estimate of  $O(n)$ , which is better than the complexity of the NBFS algorithm. However, this improvement comes at the cost of introducing synchronization, which may either require additional hardware or simply be difficult to implement in real applications, e.g., the above swarm robotics scenario. Moreover, the numbering described in [16] appears irregular: the vertex number value says nothing about the vertex's proximity to the root (i.e., the numbers are merely unique identifiers). In contrast, the numbers in the NBFS algorithm possess the following property (see the beginning of Section 2): if  $i < j$ , any level  $i$  vertex will have a number smaller than any level  $j$  vertex. This property has been utilized in the proof of Theorem 1 and may be valuable in applications.

## REFERENCES

1. Levenshtein, V.I., On a Method of Solving the Problem of Synchronizing a Chain of Automata in Minimal Time, *Probl. Inf. Transm.*, 1965, vol. 1, no. 4, pp. 14–25.
2. Moore, F.R. and Langdon, G.G., A Generalized Firing Squad Problem, *Information and Control*, 1968, vol. 12, pp. 212–220.
3. Gallager, R.G., Humblet, P.A., and Spira, P.M., A Distributed Algorithm for Minimum-Weight Spanning Trees, *ACM Transactions on Programming Languages and Systems*, 1983, vol. 5, no. 1, pp. 66–77.
4. Peleg, D. and Rubinovich, V., A Near-Tight Lower Bound on the Time Complexity of Distributed MST Construction, *Proc. 40 IEEE Symp. on Found. of Comp. Sci. (FOCS)*, 1999, pp. 253–261.
5. Vyalyi, M.N. and Khuziev, I.M., Distributed Communication Complexity of Spanning Tree Construction, *Probl. Inf. Transm.*, 2015, vol. 51, no. 1, pp. 49–65.
6. Vyalyi, M.N. and Khuziev, I.M., Fast Protocols for Leader Election and Spanning Tree Construction in a Distributed Network, *Probl. Inf. Transm.*, 2017, vol. 53, no. 2, pp. 183–201.

7. Dinitz, M., Halldórsson, M., Izumi, T., and Newport, C., Distributed Minimum Degree Spanning Trees, *Proceedings 2019 ACM Symposium Principles Distributed Computing*, 2019, pp. 511–520.
8. Awerbuch, B. and Gallagher, R.G., Distributed BFS Algorithms, *Proc. 26 IEEE Symposium on Foundations Computer Science (FOCS)*, 1985, pp. 250–255.
9. Park, J., Tokura, N., Masuzawa, T., and Hagihara, K., An Efficient Distributed Algorithm for Constructing a Breadth-First Search Tree, *Systems and Computers in Japan*, 1989, vol. 20, pp. 15–30.
10. Makki, S.A.M., Efficient Distributed Breadth-First Algorithm, *Computer Communications*, 1996, vol. 19, no. 8, pp. 628–636.
11. Burdonov, I.B. and Kossatchev, A.S., A General Approach to Solving Problems on Graphs by Collective Automata, *Proceedings of the Institute for System Programming of the RAS*, 2017, vol. 29, no. 2, pp. 27–76.
12. Burdonov, I., Kossatchev, A., and Sortov, A., Distributed Algorithms on Rooted Undirected Graphs, *Proceedings of the Institute for System Programming of the RAS*, 2017, vol. 29, no. 5, pp. 283–310.
13. Ghaffari, M., Distributed Graph Algorithms, 2022.  
<https://people.csail.mit.edu/ghaffari/DA22/Notes/DGA.pdf>
14. Ore, O., *Theory of Graphs*, Providence: American Mathematical Society, 1962.
15. Cormen, T.H., Leiserson, C.E., Rivest, R.L., and Stein, C., *Introduction to Algorithms*, Cambridge: MIT Press, 2009.
16. Métivier, Y., Robson, J.M., and Zemmari, A., A Distributed Enumeration Algorithm and Applications to All Pairs Shortest Paths, Diameter., *Information and Computation*, 2016, vol. 247, pp. 141–151.

*This paper was recommended for publication by P.Yu. Chebotarev, a member of the Editorial Board*