DOI: https://doi.org/10.15407/kvt203.01.039

UDC 519.688

MISHCHENKO M.D. [1], PhD Student,
Department of Mathematical Methods in System Analysis
ORCID: 0000-0001-6135-2569
e-mail: mishenkomihailo@gmail.com
GUBAREV V.F. [2], DSc (Engineering),
Corresponding Member of NAS of Ukraine,
Head of the Dynamic Systems Control Department
ORCID: 0000-0001-6284-1866
e-mail: v.f.gubarev@gmail.com
[1] Institute for Applied Systems Analysis,
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"
37, Peremohy av., Kyiv, 03056, Ukraine
[2] Space Research Institute of the National Academy
of Sciences of Ukraine and the State Space Agency of Ukraine
40, Acad. Glushkova, Kyiv, 03187,Ukraine

HORIZON LENGTH TUNING FOR MODEL PREDICTIVE CONTROL IN LINEAR MULTI INPUT MULTI VARIABLE SYSTEMS

Introduction. *There is wide range of systems describable as linear Multi Input Multi Variable systems evolving in discrete time. This mathematical model is often used in engineering, but it can also be applied in many other fields. The problem of stabilization of this kind of system frequently arises. In this paper we consider the Model Predictive Control approach to this problem. Its main principle is to generate control signals by optimizing consequent system's future dynamics on limited prediction horizon. While it demonstrates some good results, in practice we are always limited in terms of computational resources. Thus, we can optimize outcomes of our future control sequence only for limited horizon lengths. That is why it is valuable to understand how this limit affects control quality.*

The purpose of the paper *is to propose a way to appraise drawbacks of limiting of the prediction horizon to certain length for a particular system, so that we can make informed choice of such limit and therefore choose controller's microprocessor with sufficient computing power.*

Methods. *Several indexes which characterize the stabilization process are defined. Their heatmaps built against system's initial state are used as a convenient visualization of how system's stabilization dynamics changes depending on its initial state and of drawbacks induced by prediction*

horizon length limiting. Such heatmaps were built for several prominent example systems with different structures by performing corresponding series of computational experiments.

Results. *Drawbacks of prediction horizon length limiting vary from severe to completely nonexistent depending on the system's structure and representation. These drawbacks relax with increase of this limit.*

Conclusions. *The stabilization dynamics depends largely on the system's structure. Therefore, it is advised to take it into account and build heatmaps of aforementioned indexes to decide on prediction horizon length limit. A good system's representation can improve stabilization time with limited prediction horizon length.*

Keywords: *MPC, MIMV, heatmap, control synthesis, discrete controllable system, linear system, moving horizon, stabilization.*

INTRODUCTION

The object of this study is the Model Predictive Control (MPC) based control process. The aim of the considered control process is to stabilize a system, e. g. for a system with non-zero state vector x at initial point of time to bring its state to zero with specially crafted sequence of control signals. In this paper we are working with discrete-time linear Multi Input Multi Variable (MIMV) systems with constrained input values, which can be described as

$$x_{k+1}=\boldsymbol{A}x_k+\boldsymbol{B}u_k,\ k \in \mathbb{Z}, \tag{1}$$

$$|u_k[j]| \le u_{max},\quad k \in \mathbb{Z},\quad j \in 1..r \tag{2}$$

where x is the system's state at a particular point of time represented as n-dimensional real-valued vector, u is the control applied to the system at a particular point of time, k is a sequential number of a related point of time, j is an index of a particular element of the control vector u, $u_{\max}$ is a positive value representing constraint on control signal values, $\boldsymbol{A}$ and $\boldsymbol{B}$ are real-valued matrices of corresponding dimensions.

The aforementioned formalization is applicable to wide variety of systems. First of all, the system dynamics equation (1) can be obtained as a result of discretization of a model of a continuous-time system by transforming its differential equation into a difference equation with certain sampling rate. This transformation is indispensable if we want to control such system with digital controller, which perceives continuously changing values through analog-to-digital converters. Usage of programmable controllers is essential in various fields where we need precise online control, i. e. for operating industrial plants or various mechanisms and engines.

There are also applications outside of traditional engineering scope. Various naturally-formed (e. g. not engineered by someone) complex systems can be represented in form of weighted digraphs, as discussed in subsection 4.3 of [1]. We can also think of them as of cognitive maps. The pulse process model described further in subsection 4.4 of [1] allows us to model dynamics of such systems. Coincidentally, dynamics of the so-called pulses (differences between current system state vector and previous) can also be described with equation (1). This gives us opportunity to control them in the same way as ordinary linear systems, as it is proposed in [2–3].

It is a common problem to stabilize the system by influencing it with control signals. In this context "to stabilize" means to make system's state x equal (or at least nearly equal) to zero at some point of time. This problem was thoroughly studied within the framework of the control theory. For textbook description of the control theory approach to this problem see, for example, [4–5]. But this conventional approach also have its drawbacks. One of them is that in real-life (not simulated) systems there is always a constraint on control signal values in one form or another, which can't be directly taken into account with the control theory. The other drawback is that control loops produced this way require fine-tuning to ensure its stability and satisfy other requirements at the same time, such as high response speed and satisfaction of the aforementioned control signal constraints.

To mitigate this and some other drawbacks the MPC approach to this problem was introduced in [6]. The MPC-based feedback loop is organized similarly to ones employed in the control theory. The main difference is that control signals are calculated not by multiplying the current system's state (or its estimation) by a matrix, as in the control theory, but by solving a mathematical optimization problem. The solution of this optimization problem is a sequence of future control signals, which would result in the best predicted future system's states in terms of employed objective function. This leads to another significant difference: this way we generate a whole sequence of control signals instead of just one next control signal. Thus, there are many possible ways to employ generated control sequences. For example, we can apply the whole control sequence as is, or just use the first control signal value from each generated control sequence. Different optimization problem variants, approaches to their solving and ways to integrate their solutions into the control loop were proposed in [6–7].

PROBLEM STATEMENT

While showing some promising results [6–7], this approach requires significant computational resources, especially in comparison with control theory based controllers. Moreover, the required computational resources rapidly grow with prediction horizon length increase. Considering the fact, that there is no such thing as infinite computational resources, the prediction horizon we can implement consequentially becomes limited by hardware we use for computations and by system's sampling rate (which enforces hard limit on time we have to compute next control signal value).

As actual prediction horizon length required to produce ideal (in terms of stabilization time) control sequence differs from one situation to another, it is valuable to understand what actual drawbacks to expect from its limiting. This would give us the required information to make informed choice of such limit and therefore to choose controller's microprocessor with sufficient computing power.

The purpose of this paper is to propose a way to represent drawbacks of prediction horizon limiting in a meaningful and understandable way. Since the stabilization dynamics is essentially different for systems with different structure, it is required to perform such investigation for any particular system we are considering. In other words, in general case we can only say that the bigger horizon length limit, the better. So, in this paper we propose a way to appraise this drawbacks for a particular system and analyze some example systems to explore the impact of system's structure.

The proposed method is essentially series of system stabilization computational experiments and a way to visualize their results. So, the rest of the paper is organized as follows. The Experiment Design section describes exactly what, how and for what purpose are we doing in the computational experiments. The Experiment Results section shows some examples of application of the proposed method to perform the discussed analysis for some example systems with essentially different structures. In the Analysis of Results section we discuss our findings derived from the experiment results. Lastly, the Conclusions section provide a brief summary of obtained results.

EXPERIMENT DESIGN

Our goal here is to evaluate effects of horizon length limiting alone, in isolation from other additional factors, which can influence the quality of control process for a given horizon length limit. That is why we implement the most basic feedback loop scheme and consider the system being deterministic, even though in most real-world cases systems suffer from random perturbations.

Assumptions about the system. We consider matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ having full ranks, because otherwise there is a roughly equivalent representation of the system with full-ranked matrices and reduced dimensions.

For the same reason we also consider that $r \leq n$, i. e. that number of control vector's dimensions is not greater than number system state vector dimensions.

As it was already mentioned, the system is deterministic, i. e. there are no noises affecting the system.

Structure of matrix $\boldsymbol{A}$. The linear system representation (1–2) is not invariant, since for every invertible linear transformation of the state space we have corresponding alternative representation in the same form, but with different matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. In most examples in the Experiment Results section we construct $\boldsymbol{A}$ matrices for subject systems in our computation experiments using real-valued modification of Jordan normal form (with rotation, rotation-shrinking or rotation-expanding block corresponding to complex eigenvalues) to be sure to examine most distinct variants of stabilization dynamics. But even though linear transformations of state-space do not affect stabilization process if the prediction horizon length is not limited, it may have its impact otherwise. Thus, we also added an example described in non-canonical form for comparison.

Heatmap experiment result representation. It is obvious, that for given system and controller the system's future depends on its initial state. For deterministic systems it is fully determined by its initial state. Thus, if there are some significant numeric characteristics (indicators) of the system's dynamics, it is valuable to plot them against system's initial state.

Of course, the system's state vector x may have many dimensions, but it is not very meaningful to try to analyze a heatmap built for more than two dimensions manually. It may be possible to do with machine learning techniques, but we will not touch this topic in this paper. Thus, we propose to plot heatmaps of indicators for initial system states residing on a certain 2-dimensional (hyper-) plane.

As it is obvious that the system model's equation (1) and control synthesis algorithm we use does not have any fractal properties, it is enough to calculate the indicators we consider (by performing corresponding computational experiments) only for a finite subset of possible initial states uniformly distributed on a (hyper-)plane patch we build heatmap for. This way if we have two direction vectors e_1 and e_2, and a grid step $g \in (0,+\infty]$, then the grid will consist of points $g\,(ie_1 + je_2)$, $i, j \in \mathbb{Z}$.

The last significant thing about the heatmap structure which should be mentioned is that all heatmaps we build here demonstrate point symmetry with its center in zero. It is due to properties of the considered system structure (1—2). That is why we plot heatmaps only for a half-plane to reduce computation time required to produce such plot and to fit them on page.

Control synthesis. We compute series of controls as a solution for an optimization problem

$$\min_{u_k, u_{k+1}, \ldots, u_{k+s-1}} \left\| x_{k+s}\left(x_k, u_k, u_{k+1}, \ldots, u_{k+s-1}\right)\right\| \tag{3}$$

$$\left|u_k[j]\right| \le u_{\max}, \quad i \in 1..s-1, \quad j \in 1..r \tag{4}$$

where s is the prediction horizon length. This is a convex quadratic program, which we solve with the CVXOPT coneqp solver [8]. We apply computed control vector sequences as is, without correcting them at each next point of time, which would be natural if we were working with non-deterministic systems, i. e. when there are random noises affecting the system. Such intermediate control sequence corrections would require more sophisticated optimization problem and/or structure of the feedback loop. Obviously, the combination of the optimization problem and the way its solutions are used for control is not the most optimal, but the most simplistic one. It is because one of our aims is to demonstrate a way to benchmark such combination in general. The analysis of this basic combination should also give some valuable insights which hopefully will help in development of better ones in future.

Indicators and corresponding control strategies (controllers). As we said, we will plot heatmaps for different indicators representing significant information about system's properties regarding the stabilization process. And to calculate them we need to simulate stabilization process with different control strategies.

Stabilization time with unlimited horizon length. Our goal is to explore negative effects of prediction horizon length limiting, so firstly we need to determine what actually happens if the horizon length is unlimited to have a basis to compare with. So, for each considered initial state x_0 we sequentially try to solve the problem (3–4) for $s = 1$, $s = 2$ and so on, until the corresponding optimal final state $x_s(x_0, u_0, \ldots, u_{s-1})$ is zero. This way we obtain the first index we will plot: minimal number of iterations needed to stabilize the system with corresponding initial state.

Of course, this is just a simplification. In practice, all computations are performed with floating point numbers, which by itself causes numerical errors and thus introduces otherwise negligible perturbations, which nevertheless make obtaining a state vector *exactly* equal to zero highly improbable. In addition, the CVXOPT solver computes a numeric estimation of problem's solution, rather than its precise value. That is why we follow a common practice to check if the

final state vector $x_s(x_0, u_0, \ldots, u_{s-1})$ resides in a certain small ε-neighborhood of zero instead of checking for strict equality.

The other peculiarity we must address here is that if the system is unstable, i. e. $\rho(\boldsymbol{A}) > 1$, then for such system there are initial states, for which stabilizing control does not exist. And, obviously, the aforementioned optimal control synthesis procedure will never meet its stop condition for such initial states. In addition, the solver faces computational difficulties when we try to find a solution of the problem (3–4) for really huge values of s. Thereby, we stop the procedure when the next horizon length s exceeds a certain huge value `huge_s`.

We will call the simulated model controller implementing aforementioned principles "the ascending controller" for short.

Stabilization time with limited horizon length. The next index we will plot as a heatmap is stabilization time obtainable with a controller, whose solver is unable (or prohibited) to solve problems with s higher than certain value s_{max}. We will call corresponding simulated model controller "the descending controller". This controller will generate control sequences of limited length and apply them to the simulated system as is in loop, until the ε-neighborhood of zero is reached.

It initially will try to solve the problem (3–4) for maximum allowed horizon length $s_{\max}$. But if the controller will just apply generated control sequences as is, the stabilization time that we would measure would be a multiple of $s_{\max}$. The value measured in this way would often overestimate the required stabilization time, because the last generated control sequence (which finally leads the system to the aimed ε-neighborhood of zero) could have been shorter. So, in this situation the descending controller will check if the next generated control sequence would finally stabilize the system. If it is the case, the controller will try to solve the problem (3–4) for $s = s_{\max} - 1$, $s = s_{\max} - 2$ and so on until corresponding $x_{k+s}(x_k, u_k, u_{k+1}, \ldots, u_{k+s-1})$ will fall out of the ε-neighborhood of zero, or until s = 1 inclusive. Hence the name of this controller: the descending controller. The shortest stabilizing sequence will be applied to the system. This way we will obtain more adequate measurement of required stabilization time.

Stabilization time loss for limited horizon length. To compare the stabilization time measured with the descending and ascending controllers, we will calculate their difference, which is another index we will plot heatmap for. It is expected that this difference will be nonnegative.

There are three implicit values which originate from the fact that this is a computational experiment and which affect this experiment's outcome. The first one was already mentioned. It is the radius of the zero's neighborhood we compare system's state x with: ε. The other two are related to the CVXOPT solver, namely `reltol` and `abstol`, which control relative and absolute accuracy of results returned by the solver. These three values need to be finely tuned to obtain meaningful results. Too small ε or too big `reltol` and/or `abstol` leads to overestimation of minimal stabilization time (measured for ascending controller). Too big ε leads to its underestimation. Too small `reltol` or `abstol` leads to computational difficulties in the solver.

Such overestimation becomes apparent if we see negative values on the stabilization time loss heatmap. This gives us required information to manually tune these implicit variables.

Distance loss for limited horizon length. There is one more index we measure. It has less obvious construction. It measures minimum distance from zero

achievable by a controller with solver with limited horizon length at the point of time, when the same system with the same initial state would have been stabilized by the ascending controller.

To measure this index we use another specific emulated controller. Let us call it "the distance controller". As in the descending controller, this controller will generate control sequences of limited length. The main difference is that it does not check whether the generated control sequence would stabilize the system (i. e. would lead the system's state to the ε-neighborhood of zero). Instead it calculates and applies control sequences of maximum allowed length until it can do so without exceeding number of system's iterations $s_{opt}(x_0)$ needed by the ascending controller to stabilize the same system from the same initial state. When such situation eventually occurs at certain point of time k, it solves the problem (3–4) for horizon length $s = s_{opt}(x_0) - k < s_{max}$ and applies generated control sequence to the system. This way this controller executes the same number of iterations, as it was done by the ascending controller for the same system with the same initial state x_0.

In the beginning the distance controller has the same behavior as the descending one, e. g. they both generate control sequences for horizon length $s = s_{max}$, so we reuse them to reduce computation time.

EXPERIMENT RESULTS

In this section indicator heatmaps we built for systems of various structures will be shown, and how their structure affects stabilization process will be discussed. As it can be seen from the Experiment Design section, the computational experiment get a number of input variables, which can be divided into four groups.

The first group describes the system itself: the $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices. The spectral radius of $\boldsymbol{A}$, which is denoted as $\rho(\boldsymbol{A})$, is also shown in tables next to them, because its value determines, whether the system is stable or not. The second group is constraints on control values and horizon length, denoted as u_{max} and s_{max}. The third group consists of plotting parameters, namely the direction vectors e_1 and e_2 of the heatmap grid and its grid step g. On heatmap plots e_1 and e_2 correspond to the horizontal and vertical axes. The last group consists of some experiment parameters concerning caveats of numeric computations: ε, `reltol`, `abstol` and `huge_s`. All following computational experiments were performed with ε = 1E-6, `reltol` = 1E-12, `abstol` = 1E-10.

Stable systems. In this subsection results obtained for stable systems, e. g. whose matrix $\boldsymbol{A}$ have spectral radius $\rho(\boldsymbol{A})$ less than 1, will be presented. In this case systems can stabilize themselves even without any control, so the purpose of the controller is to make the system stabilize faster.

Stable system with real-valued eigenvalues. Let us begin with the most obvious case demonstrated on Fig. 1. As can be seen from the sysem's structure on Table 1, dimensions of the state space do not influence each other and can be controlled separately. So, on the heatmaps of stabilization time the maximum of required times for both dimensions can be seen. The border on which the stabilization times of each dimension are equal can also be seen on the plots. In this particular case this lines are straight and form 45° and 135° angles with absciss. This is because the self-stabilization speeds (defined by eigenvalues of the matrix $\boldsymbol{A}$) and control impact magnitudes (defined by coefficients of the matrix $\boldsymbol{B}$) are equal for both dimensions. In other case this border would be curvilinear.

It is also worth to notice, that the time loss in this case is always nonexistent (equal to 0) and therefore the distance loss is always smaller than ε.

The horizon length limit s_{max} was set to 1, so, considering such good results it would be a waste to use bigger horizon length limit for such a trivial case.

Rotation-shrinking stable system. On heatmaps on Fig. 2 built for a stable system with complex-valued eigenvalues we can see the same results as for the one with real-valued eigenvalues. The only difference is the peculiar shape of them. The used $\boldsymbol{A}$ matrix represents combination of the following two operations: rotation by 45° counterclockwise and shrinking by a constant. If we use rotation by a degree, which is not a divisor of 360°, then the heatmap shape will become circular, rather than edged. We suppose that this edgedness is a consequence of the shape of the set of allowed control signals defined in (2).

As in the previous example, the system's structure and some other variables are presented on Table 2.

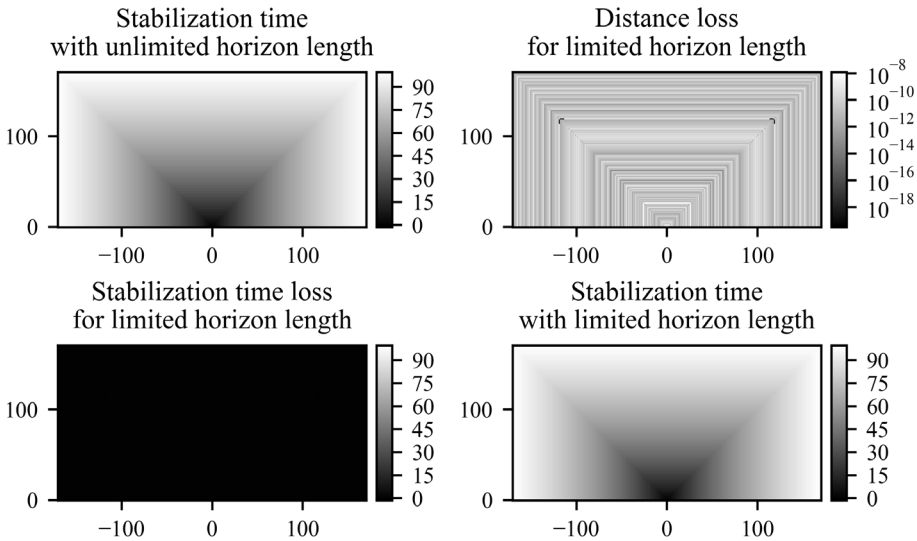

Fig. 1. Index heatmaps for a stable system with real eigenvalues

Table 1. **Computational experiment input values corresponding to heatmaps on Fig. 1**

Variable	Value	Variable	Value
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 0.99 & 0 \\ 0 & 0.99 \end{pmatrix}$	e_1	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	e_2	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\rho(\boldsymbol{A})$	0.99	grid step (g)	1.0

Constraints			
u_{max}	1.0	s_{max}	1

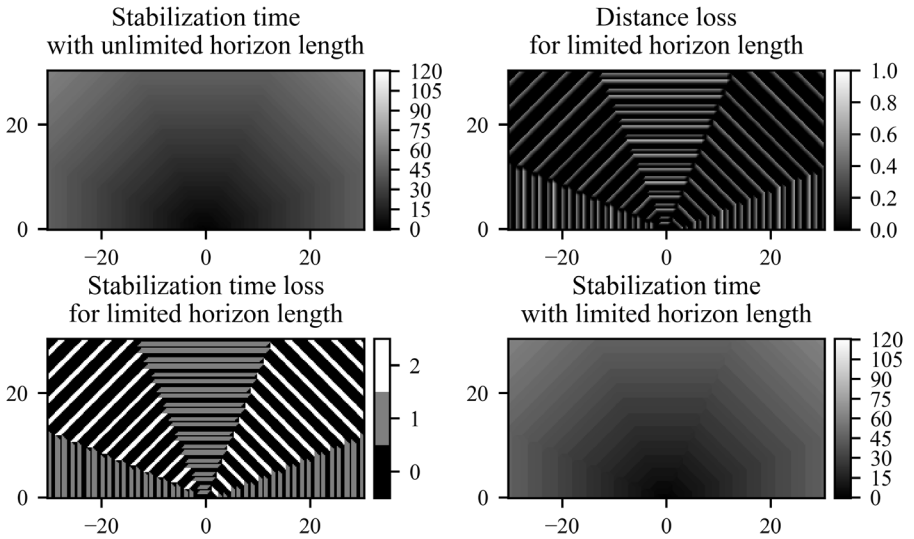

Fig. 2. Index heatmaps for a rotation-shrinking stable system

Table 2. **Computational experiment input values corresponding to heatmaps on Fig. 2**

Variable	Value	Variable	Value
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 0.7036 & -0.7036 \\ 0.7036 & 0.7036 \end{pmatrix}$	e_1	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	e_2	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\rho(\boldsymbol{A})$	~0.995	grid step (g)	1.0

Constraints			
$u_{\max}$	1.0	$s_{\max}$	1

Uncontrollable stable linear system. It is also interesting to see what happens if we can't control two different components of the state space separately. If these components correspond to two different real-valued eigenvalues, as in the system defined in Table 3, it is obvious that the system becomes uncontrollable. Nevertheless, on Fig. 3 we can see, that if the uncontrollable system is stable, then trying to control such system is not completely useless, because depending the initial state we still can make the stabilization process a bit faster. And for some particular initial states, whose set depends on structure of the matrix $\boldsymbol{B}$, we can make it stabilize significantly faster. But we had to use a system with significantly smaller $\rho(A)$ for this example to make stabilization time small enough to be able to compute it in reasonable time.

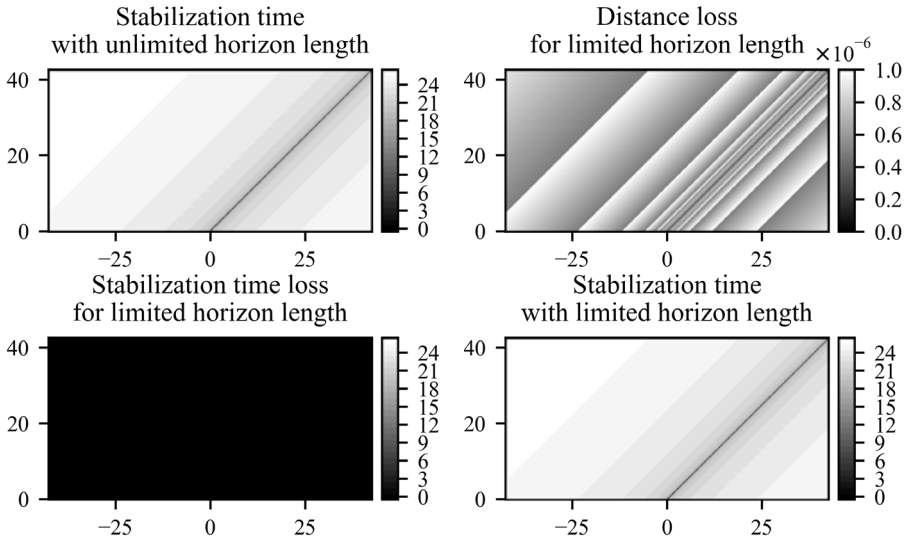

Fig. 3. Index heatmaps for uncontrollable stable linear system

Table 3. **Computational experiment input values corresponding to heatmaps on Fig. 3**

Variable	Value	Variable	Value
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}$	e_1	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	e_2	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\rho(\boldsymbol{A})$	0.5	grid step (g)	0.25

Constraints			
$u_{\max}$	1.0	$s_{\max}$	1

As in the previous examples, here the control with prediction horizon length equal to 1 has no drawbacks in comparison with control with unlimited horizon length.

Rotation-shrinking stable system with intertwined control. Significantly different picture can be seen if we can influence only one component of the state space among the two corresponding to the Jordan matrix cell for complex-valued pair of eigenvalues, as in Fig. 4, Table 4 and Fig. 5, Table 5. When we limit horizon length, the control synthesis algorithm becomes greedy by its nature, consequences of which can be clearly seen on the stabilization time and distance loss plots on Fig. 4 and Fig. 5.

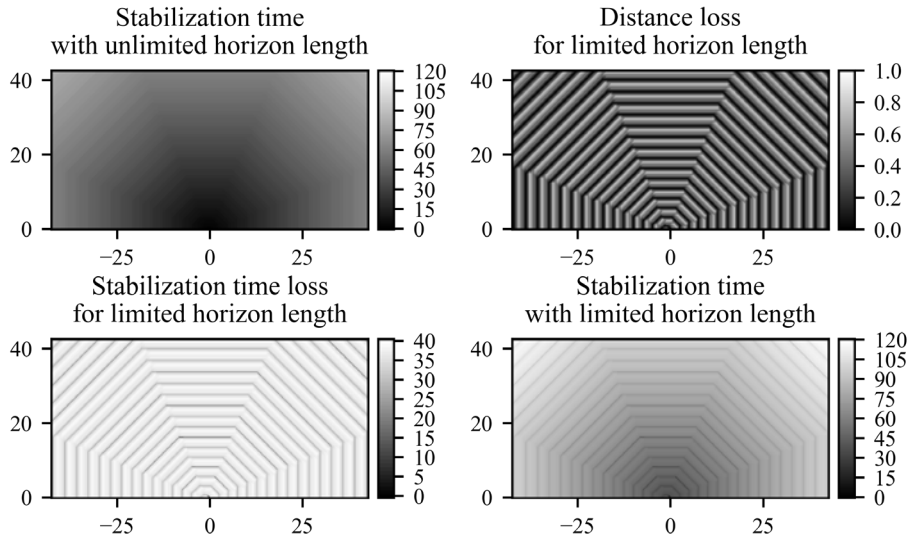

Fig. 4. Index heatmaps for rotation-shrinking stable system with intertwined control and smax = 1

Table 4. **Computational experiment input values corresponding to heatmaps on Fig. 4 and Fig. 6**

Variable	Value	Variable	Value
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 0.7036 & -0.7036 \\ 0.7036 & 0.7036 \end{pmatrix}$	e_1	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	e_2	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\rho(\boldsymbol{A})$	~0.995	grid step (g)	0.25

Constraints			
u_{max}	1.0	s_{max}	1

In this example matrix $\boldsymbol{B}$ does not allow to influence both components. This makes it similar to the previous example on Fig. 3, Table 3, where matrix $\boldsymbol{B}$ does not allow to control both components separately, because in both cases we can "push" the system only along one single direction, which intertwines control of two different state-space components. While it is so, the $\boldsymbol{A}$ is effectively a rotation-shrinking matrix. That is why influences on one of the components consequentially influence dynamics of both of them. In other words, the system does not become uncontrollable, like in the previous example. Instead, this peculiarity lets the greediness of the algorithm manifest.

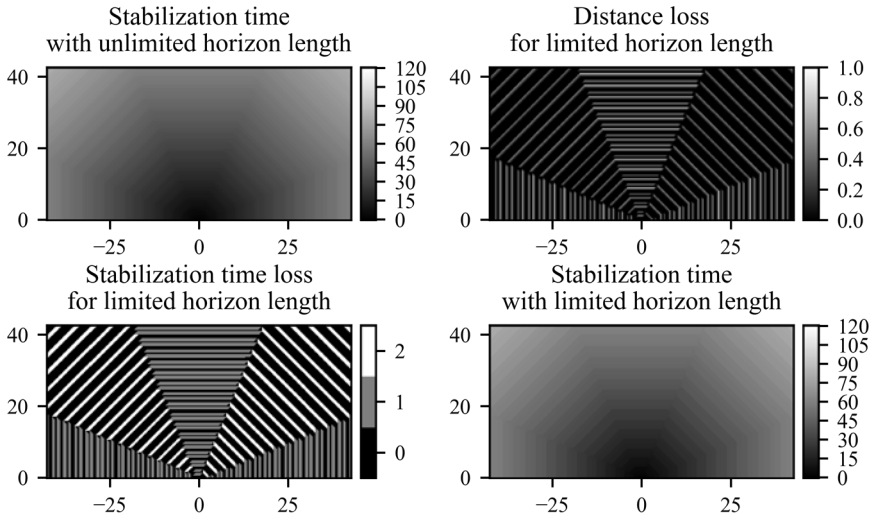

Fig. 5. Index heatmaps for rotation-shrinking stable system with intertwined control and $s_{max} = 2$

Table 5. **Computational experiment input values corresponding to heatmaps on Fig. 5**

Variable	Value	Variable	Value
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 0.7036 & -0.7036 \\ 0.7036 & 0.7036 \end{pmatrix}$	e_1	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$	e_2	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\rho(\boldsymbol{A})$	~0.995	grid step (g)	0.25

Constraints			
u_{max}	1.0	s_{max}	2

Even though there are visible losses from horizon length limiting, it is clearly seen that they are limited and do not worsen indefinitely for initial states further from zero, as we can see on Fig. 4 and Fig. 5. In addition, the time loss improves greatly when we increase horizon length limit from 1 to 2: its maximum drops from 39 to 2. At the same time, the distance loss does not improve with horizon length increase. We also tested other horizon lengths up to 8, but both loss indices did not improve further.

It is also interesting, that if we expand this heatmaps to another half-plane, we will see that both loss heatmaps have a shape of double spiral, which we can see on Fig. 6.

Rotation-shrinking stable system in sheared space. Until now we have seen systems with canonical blocks used as matrix $\boldsymbol{A}$. But, as it was already mentioned, the controller becomes greedy when the prediction horizon is limited. Thus, linear transformation of the state-space in form (5–6) can distract it into choosing not the best intermediate aims.

$$x'_k = Px_k, \quad k \in \mathbb{Z} \tag{5}$$

$$x'_{k+1} = PAP^{-1}x'_k + PBu_k, \quad k \in \mathbb{Z} \tag{6}$$

To demonstrate this effect we transformed the state-space for the system described in Table 2 with shearing matrix (7) and repeated the experiment. Full specification of it, as always, can be seen in Table 6.

$$P = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \tag{7}$$

As we can see on Fig. 7, after this transformation the duration loss is not limited, but instead it grows along certain directions, unlike what we see in the example on Fig. 2, Table 2.

Stable system with one real-valued eigenvalue and two generalized eigenvectors. The most interesting results we have when the matrix $\boldsymbol{A}$ is defective, as in example on Fig. 8, Table 7. On the heatmap for stabilization time with unlimited horizon length we can clearly see that initial positive values of the component corresponding to e_1 (which is the ordinary eigenvector of the matrix $\boldsymbol{A}$) compensate for initial negative values of the other component corresponding to e_2 (which is the generalized eigenvector of rank 2 of the matrix $\boldsymbol{A}$). Considering that the heatmaps are point-symmetric, we can also say, that negative values of the former also compensate for positive values of the latter.

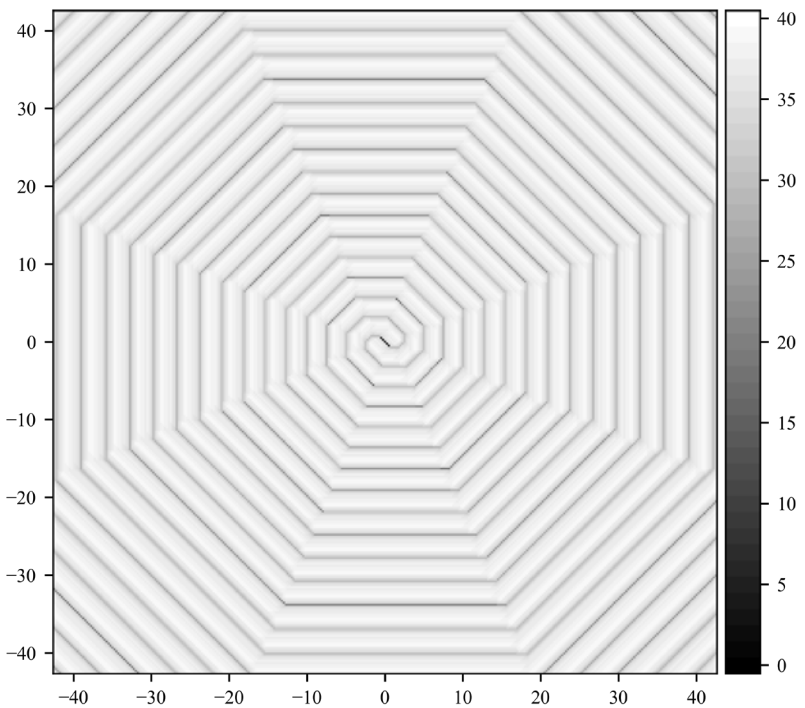

Fig. 6. Stabilization time loss for rotation-shrinking stable system with intertwined control and s_{max} =1, both half-planes

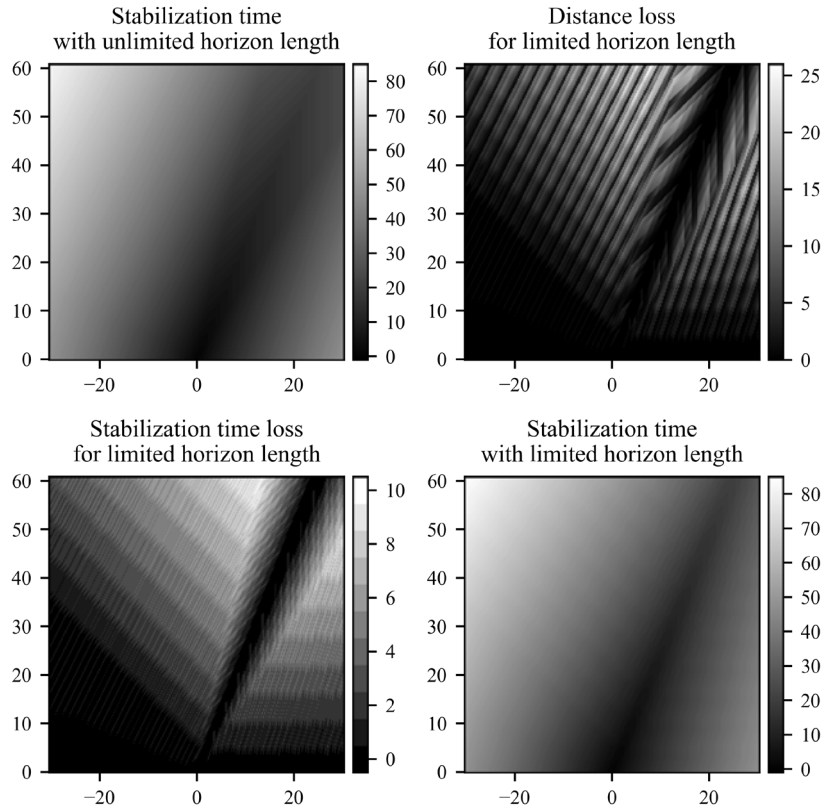

Fig. 7. Index heatmaps for rotation-shrinking stable system in sheared space

Table 6. **Computational experiment input values corresponding to heatmaps on Fig. 7**

Variable	**Value**	**Variable**	**Value**
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 2.1108 & -3.5180 \\ 0.7036 & -0.7036 \end{pmatrix}$	e_1	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$	e_2	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\rho(\boldsymbol{A})$	~0.9950	grid step (g)	1.0

Constraints			
$u_{\max}$	1.0	$s_{\max}$	1

Other significant difference is that, unlike in previous examples, the greediness of the algorithm for limited horizon lengths has significant impact, even though we can directly influence both components of the state-space. Even more, the losses are not limited - they increase indefinitely the further the initial state is from certain "free-fall trajectory", which is clearly visible on the corresponding heatmap, and along which the time loss is zero. Also, the stabilization time (either with or without horizon length limiting) grows the slowest along the same trajectory.

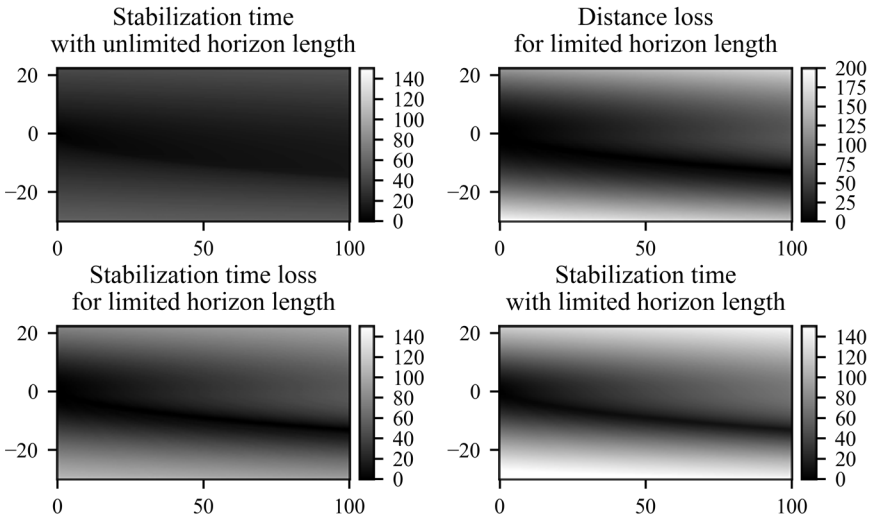

Fig. 8. Index heatmaps for stable system with defective matrix $\boldsymbol{A}$

Table 7. **Computational experiment input values corresponding to heatmaps on Fig. 8**

Variable	**Value**	**Variable**	**Value**
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 0.99 & 1 \\ 0 & 0.99 \end{pmatrix}$	e_1	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$	e_2	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\rho(\boldsymbol{A})$	0.99	grid step (g)	0.25

Constraints			
$u_{\max}$	1.0	$s_{\max}$	1

This heatmaps are built for $s_{max} = 1$, but the dynamics looks the same for its bigger values, even though the losses' growth becomes somewhat slower.

It is important to note, that in this example we can influence both components of the state-space separately, but this does not help much, like in some previous examples, even though the system is controllable.

Unstable systems. In this subsection unstable systems, e. g. whose matrix $\boldsymbol{A}$ has spectral radius $\rho(\boldsymbol{A})$ bigger than 1, will be discussed. In this case systems can not stabilize themselves.

Moreover, if they are left by themselves without any control, the slightest disturbance can make initially equal to zero system's state trend to infinity. This makes stabilizing control even more important for unstable systems, than for stable ones.

It is well-known, that if the control resources are limited (for example, as in (2)), then for an unstable system there are initial states, for which this system can not be stabilized, e. g. the stabilizing control sequence does not exist. Thus, it is expected that a certain border surface in state-space exist, inside of which stabilization is still possible, while outside of which it is not. These our expectations were corroborated with our

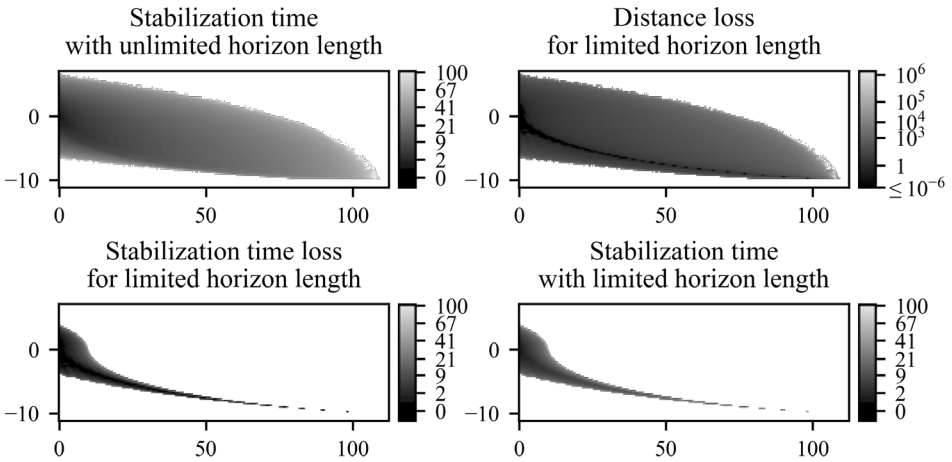

Fig. 9. Index heatmaps for unstable system with defective matrix $\boldsymbol{A}$

Table 8. **Computational experiment input values corresponding to heatmaps on Fig. 9**

Variable	**Value**	**Variable**	**Value**
System		**Plotting parameters**	
$\boldsymbol{A}$	$\begin{pmatrix} 1.1 & 1 \\ 0 & 1.1 \end{pmatrix}$	e_1	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\boldsymbol{B}$	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$	e_2	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\rho(\boldsymbol{A})$	1.05	grid step (g)	0.25

Constraints			
$u_{\max}$	1.0	$s_{\max}$	1

experiment results. The unstable systems demonstrate the same dynamics as equivalent stable ones, except the aforementioned border that can be clearly seen on plots.

When we talk about the border surface, it is natural to imagine it being bounded when projected on those state-space components, which correspond to unstable Jordan matrix's cells. According to results of our computational experiments, this intuition is indeed true, but only for systems with non-defective matrix $\boldsymbol{A}$. If this matrix is defective, then the "free-fall trajectory" similar to ones for stable systems with defective matrix $\boldsymbol{A}$ also exist. But, unlike in stable system, it does not continue indefinitely, which can be clearly seen on Fig. 9, Table 8.

It is also worth noticing that the stabilization time with unlimited horizon length near the boundary starts to increase so abruptly, that with the chosen grid step this growth was captured only by some of its cells near the border, even though not reaching the huge_s, which in this particular case was set to 130.

ANALYSIS OF RESULTS

As there is inexhaustible variety of possible linear discrete-time systems which can be defined in form (1–2), we limited the examples shown in the Experiment Result section to those displaying some prominent characteristics of the stabilization process. And now let us discuss our findings.

One of expected and rather obvious effects which was confirmed in this research is that stabilization time improves with increase of the prediction horizon limit if the objective function optimizes distance of the future system's state at the end of the horizon, as it was described in (3). The question was in what cases and to what degree it can be limited without significant losses, and how can the controller be modified to alleviate this losses.

As it was demonstrated in the examples from the Experiment Results section, for some systems losses do not grow indefinitely for initial states further from zero. In these cases a certain rather small horizon length limit can be safely assigned. At the same time, there are examples where this is not the case, and thus we need to consider the area in which initial states of the system are likely to occur. The most noticeable among them are the examples with defective matrix $\boldsymbol{A}$.

A valuable finding upon which we have stumbled is that a particular system's representation (among equivalent ones) has significant impact on efficiency of generated controls under limited prediction horizon. This leads us to a conclusion, that we should transform the system's representation (1) in some way, as it was described in equations (5–6), to get the most efficient controller and to minimize required computational resources at the same time.

As it was demonstrated in our experiments, in some trivial cases the prediction horizon can be limited to impressive one or two steps. Thus, it is tempting to just transform the matrix $\boldsymbol{A}$ into the real-valued modification of the Jordan normal form and transform matrix $\boldsymbol{B}$ accordingly. But the shape of possible controller's impacts on the system's state (the Bu_k part in (1)), which originates from values of matrix $\boldsymbol{B}$ and constraints (2), may (and in many cases — will) make it more optimal to sacrifice stabilization speed of some state's components to speed-up it for others in long term. So, even in this simplified representation the controller with limited prediction horizon may not catch the most fast stabilization trajectory. That is why it is still not obvious which transformation of the state-space would be the most efficient for a particular system.

For the same reason we can't neglect the structure of matrix $\boldsymbol{B}$ and decompose the system into independent subsystems by transforming matrix $\boldsymbol{A}$ into block-diagonal form and using each block as a subsystem's matrix $\boldsymbol{A}$ — this way we would reduce the aforementioned set of possible controller's impacts on the system's state, and thus we would not be able to use its full potential.

The other possible approach is to construct a different objective function, which does not blindly optimize the distance of predicted future system's state at some point of time. It leads us to a question, what the most efficient objective function looks like.

In this research we obtained an instrument to see the very limits of possible improvement of time required for stabilization process. Even though the most simplistic variant of controller was used, it still produces the most optimal stabilizing control sequence possible if we set the prediction horizon to appropriate length and have required computational resources to compute it. It gave us an

opportunity to see how exactly the minimum possible stabilization time change depending on initial system's state. It is valuable to see the limits of what is possible so that we will not waste time trying to do the impossible. It also gave us valuable hints about how to improve stabilization time while not allocating ridiculously big amount of computational resources.

From our observations in the Experiment Results section it becomes obvious, that the most efficient objective function is the one which optimizes the time which will be required to stabilize the system from next intermediate state. And this time is actually plotted on the "Stabilization time with unlimited horizon length" plots throughout the Experiment Results section. As we can see on the plots, this ideal objective function is not convex in most cases. This complicates controller development in various ways. Firstly, with such objective function we no longer can use convex optimization algorithms. Secondly, we need to somehow represent such rather intricate function to work with it. It would also be good to have a relatively fast way to calculate coefficients of this representation from matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, similarly to how we can do it for objective function (3).

Of course, this ideal objective function can be, for example, precomputed for some set of points in state-space and in order to interpolation, but this way the flexibility of digital controllers would be lost because this way the system's model no longer can be fixed on the fly. This is a significant drawback, because in some cases coefficients of the system drift and so we do need to fix the model used in controller.

Thus, the question about computationally-efficient controller which would give the best possible stabilization speed is still open. Thus, we propose to test new variants of controllers in a way described in this paper to have a comprehensive picture of their strong points and limitations. While examples in this paper are, in fact, two-dimensional, this approach can also be extended for systems with more dimensions. We can build heatmaps for different two-dimensional slices of state-space, as it was described in the Experiment Design section. This way we can see more comprehensive picture of stabilization dynamics, than a single slice can give us. Considering all the uncertainties mentioned above, it is advised to do so and not to approximate behavior of more complex systems with behavior of previously explored more simple ones.

CONCLUSIONS

In this paper we proposed a way to appraise and visualize negative effects of prediction horizon length limiting for particular system, objective function and scheme of feedback loop used in controller. We also demonstrated stabilization dynamics for some example systems with distinctive structures in case when the most basic variant of MPC-based feedback loop is used. These observations gave us some valuable insights about the stabilization dynamics with MPC-based controller in general.

It was shown that in many cases the losses from horizon length limiting can grow indefinitely for initial system's states further from zero if the used objective function optimize norm of predicted future state. Nevertheless, at least in some cases this growth can be prevented with certain good state- space transformation. It was also confirmed that for such objective function the losses decrease with increase of the prediction horizon length limit.

As a side result, we were able to plot minimum required stabilization time for various example systems, which can be used as the most efficient objective function if

found. This objective function was shown to have an intricate structure which largely depends on system's structure and representation. In our case the plots of this objective function were obtained with excessive amount of computations, so how to efficiently find and use it in practice remains an open question.

As there is an inexhaustible variety of possible modifications of the MPC-based feedback loop, it is important to have an instrument to analyze how a particular variant performs. The proposed method provides a convenient visualization for this purpose. This way we can compare the already existing stabilization methods and those to be developed with each other and with the very limits of possible performance.

This visualization also gives an opportunity to fine-tune a particular stabilization method and to determine amount of computational resources required for it to achieve required performance.

The proposed visualization also allowed to have a glimpse of how the best possible stabilization performance would look like. While in this work it required an amount of computations inadequate for usage in online controllers, we hope that the obtained results will help in future research in this direction to achieve best possible stabilization times with reasonable computational resources.

REFERENCES

1. Roberts F. Discrete Mathematical Models with Applications to Social, Biological, and Environmental Problems. Englewood Cliffs, Prentice-Hall, 1976. 559 p.
2. Romanenko, V. D., Milyavskiy, Yu. L. Ensuring the sustainability of pulse processes in cognitive maps on the basis of the models in the states space. *System research and information technologies*. 2014. № 1. P. 26–42. (In Russian)
3. Romanenko V.D., Milyavskiy Y.L. Impulse Processes Stabilization in Cognitive Maps of Complex Systems Based on Modal State Regulators. *Kibernetika i vyčislitelnaâ tehnika*. 2015, Iss. 179, pp 60–71. (In Russian)
4. Kailath, T. *Linear systems*. Englewood Cliffs, NJ: Prentice-Hall, 1980
5. Chen, C.-T. *Linear system theory and design.* NY: Oxford University Press. 1999/
6. Gubarev V.F., Mishchenko M.D., Snizhko B.M. Model Predictive Control for Discrete MIMO Linear Systems. In: Kondratenko Y., Chikrii A., Gubarev V., Kacprzyk J. (eds) Advanced Control Techniques in Complex Engineering Theory and Applications. Studies in Systems, Decision and Control, 2019 vol 203. pp 63–81 Springer, Cham. https://doi.org/10.1007/978-3-030-21927-7_4
7. Mishchenko M.D., Gubarev V.F. Methods of Model Predictive Control for Discrete Multi-Variable Systems with Input. *Cybernetics and Computer Engineering*, 2020, 1(199), pp 39–58.
8. Vandenberghe, L. The cvxopt linear and quadratic cone program solvers. March 2010 URL: http://www.ee.ucla.edu/~vandenbe/publications/coneprog.pdf, (Last accessed: 20.12.2020)

Received 24.12.2020

ЛІТЕРАТУРА

1. Roberts F. Discrete Mathematical Models with Applications to Social, Biological, and Environmental Problems. Englewood Cliffs, Prentice-Hall, 1976. 559 p.
2. Романенко В.Д., Милявский Ю.Л. Обеспечение устойчивости импульсных процессов в когнитивных картах на основе моделей в пространстве состояний. *Системні дослідження та інформаційні технології*. 2014. № 1. С. 26–42.
3. Романенко В.Д., Милявский Ю.Л. Стабилизация импульсных процессов в когнитивных картах сложных систем на основе модальных регуляторов состояния *Киб.и выч. техн.*. 2015, Вып. 179, С 60–71.

4. Kailath, T. *Linear systems*. Englewood Cliffs, NJ: Prentice-Hall, 1980
5. Chen, C.-T. *Linear system theory and design.* NY: Oxford University Press. 1999/
6. Gubarev V.F., Mishchenko M.D., Snizhko B.M. Model Predictive Control for Discrete MIMO Linear Systems. In: Kondratenko Y., Chikrii A., Gubarev V., Kacprzyk J. (eds) Advanced Control Techniques in Complex Engineering Systems: Theory and Applications. Studies in Systems, Decision and Control, 2019 vol 203. pp 63–81 S-pringer, Cham. https://doi.org/10.1007/978-3-030-21927-7_4
7. Mishchenko M.D., Gubarev V.F. Methods of Model Predictive Control for Discrete Multi-Variable Systems with Input. *Cybernetics and Computer Engineering*, 2020, 1(199), pp 39–58.
8. Vandenberghe, L. The cvxopt linear and quadratic cone program solvers. March 2010 http://www.ee.ucla.edu/~vandenbe/publications/coneprog.pdf, (Last accessed: 20.12.2020)

Отримано 24.12.2020

Міщенко М.Д.[1], аспірант,
кафедра математичних методів системного аналізу
ORCID: 0000-0001-6135-2569
e-mail: mishenkomihailo@gmail.com
Губарєв В.Ф.[2], д-р. техн. наук, чл.-кор. НАН України,
зав. відд. керування динамічними системами
ORCID: 0000-0001-6284-1866
e-mail: v.f.gubarev@gmail.com
[1] Інститут прикладного системного аналізу,
НТУУ «Київський політехничний інститут імені Ігоря Сікорського»
пр. Перемоги, 37, м. Київ, 03056, Україна,
[2] Інститут космічних досліджень НАН України та ДКА України,
пр. Акад. Глушкова 40, корп. 4/1, Київ, 03187, Україна

ВИБІР ДОВЖИНИ ГОРИЗОНТУ ДЛЯ КЕРУВАННЯ ЗА ПРОГНОЗНОЮ МОДЕЛЛЮ У ЛІНІЙНИХ СИСТЕМАХ З БАГАТЬМА ЗМІННИМИ ТА ВХОДАМИ

Вступ. Є широкий спектр систем, які можуть бути описані як лінійні системи з багатьма змінними та входами, що функціонують у дискретному часі. Ця математична модель часто застосовується в інженерії, але також може бути застосована у багатьох інших сферах. Завдання стабілізації систем такого типу є досить розповсюдженим. У статті розглядається підхід до керування за прогнозною моделлю у розв'язанні цієї задачі. Його головний принцип полягає у генеруванні керуючих сигналів шляхом оптимізації майбутніх станів, у які перейде система внаслідок цих керувань, на обмеженому прогнозному горизонті. Хоча цей підхід демонструє непогані результати, на практиці завжди є обмеження в обчислювальних ресурсах. Через це оптимізувати наслідки майбутньої послідовності керувань є можливим лише на горизонтах обмеженої довжини. Тому важливо розуміти, як це обмеження впливає на якість керування.

Метою статті є запропонувати спосіб оцінювання негативних впливів обмеження прогнозного горизонту до певної довжини для конкретної системи, аби можна було зробити поінформоване рішення щодо цієї максимальної довжини і таким чином вибирати для контролера мікропроцесор з достатньою обчислювальною потужністю.

Методи. Було задано декілька індексів, що характеризують процес стабілізації. Теплові карти їхньої залежності від початкового стану системи використовуються як зручна візуалізація змін динаміки стабілізації системи у залежності від початкового стану, а також негативних впливів, спричинених обмеженням довжини прогнозного горизонту. Такі теплові карти було побудовано для кількох визначних прикладів систем з різними структурами шляхом виконання відповідних обчислювальних експериментів.

Результати. Втрати від обмеження довжини прогнозного горизонту варіюються від значних до повної їх відсутності у залежності від структури системи і її подання. Ці втрати зменшуються, якщо збільшити межу довжини прогнозного горизонту. Проста цільова функція, що мінімізує норму майбутнього стану, дає найкращі результати для таких систем, матриця природнього відгуку яких є діагоналізовною над полем комплексних чисел і є поданою у дійсночисловій Жордановій формі. Інакше результати сильно погіршуються.

Висновки. Динаміка стабілізації суттєво залежить від структури системи. Тому варто брати це до уваги і будувати теплові карти індексів втрат для процесу стабілізації аби визначитись з обмеженням на довжину прогнозного горизонту. Вдале представлення системи може зменшити час стабілізації за умов обмеження на довжину прогнозного горизонту. Також, функція найменшого необхідного часу стабілізації для початкового стану може розглядатись як ідеальна цільова функція, але знаходження цієї функції для конкретної системи є проблематичним.

Ключові слова: *керування за прогнозною моделлю, система з багатьма змінними та входами, теплова карта, синтез керування, дискретна керована система, лінійна система, рухомий горизонт, стабілізація.*