UDC 519.853-049.8:[681.513.52:681.513.54]

M. Mishchenko

PRACTICAL ASPECTS OF MODEL PREDICTIVE CONTROL IN LINEAR SYSTEMS AND COGNITIVE MAPS

Mykhailo Mishchenko

National Technical University of Ukraine «Igor Sikorsky Kyiv Politechnic Institute», Institute for Applied Systems Analysis,
orcid: 0000-0001-6135-2569
mdmisch@firemail.cc, mdmisch@protonmail.com

The subject of this paper is peculiarities of model predictive control (MPC) application in linear discrete-time system stablization. While stabilization in linear systems is already a well-studied problem in control theory, the MPC approach gives opportunity to produce faster stabilization trajectories at cost of higher amount of computations required. Significant progress in capabilities of computers since emergence of the control theory makes the MPC approach feasible in modern times. The MPC approach gives an opportunity to achieve significantly better results, but its application requires great care. It is due to many unobvious and undesirable effects it leads to if used incorrectly. These effects are discussed, explained and demonstrated one by one on examples in this paper. Analysis of their causes reveals requirements for MPC-based stabilizing control algorithm which allow resulting controller to operate reliably. It also appears that in most cases an optimal stabilization trajectory is not unique, i.e. it is possible to choose between optimal trajectories to improve some kind of secondary objective. In addition, as an example which is valuable by itself, stabilization in linear cognitive maps is discussed separately. Being an example of discrete-time linear system, linear cognitive maps are susceptible of application of the same control strategies and algorithms to their impulses. But if nature of linear cognitive map is disregarded, their state starts to wander under pressure of external random perturbation (i.e. noise) even though stabilizing controller mitigates their influence on cognitive map's impulses. Ability of the MPC approach to consider secondary objectives allowed to mitigate this effect at least partially. In particular, it is achieved here by seeking a particular objective cognitive map state as a secondary objective in search for a stabilization trajectory. It is also demonstrated here, that only a certain hyperplane in cognitive map's state-space is reachable under assumption, that its impulse is zero at the end of trajectory.

Keywords: model predictive control (MPC), linear system, cognitive map, linear cognitive map, stabilization, aimed stabilization, optimization, quadratic cone programming.

Introduction

Linear discrete-time systems are ubiquitous in modern engineering. This general mathematical model appeared to be an adequate formal representation for wide range of systems appearing (or being constructed) in real world. At the same time, this model is useful for integration with various digital controllers. In fact, it first appeared as a discretization of continuous-time linear systems, which became necessary when analog-to-digital converters (ADC) started being used for their integration with digital controllers [1, p. 479-480].

Consequentially, methods and approaches of the control theory used in designing controllers for classic linear continuous-time systems were adapted for discrete-time systems. Even the core mathematical technique for continuous-time linear systems — the s-transform — was adapted, which gave birth to the z-transform intended for discrete-time linear systems.

For that time it was an adequate solution. Back then CPUs (central processor units, i.e. processors) used in digital controllers had extremely low computing speed, amount of memory and so on from today's point of view. The linear quadratic regulator and similar algorithms originating from the control theory were easy to implement with that day's scarce computational resources.

Today, when even commodity CPUs have exorbitant computation power, there is much more room for improvement. Modern controllers can run significantly more sophisticated algorithms in order to not only stabilize a system, but to do it in the shortest possible time.

The MPC approach, which is widely discussed in [2], gives an opportunity to design a better stabilization algorithm befitting powerfulness of modern computers. The core idea of MPC is that if system's behaviour is known and predictable, then it is possible to rank all achievable outcomes with some kind of objective function. Therefore, it is possible to synthesize control sequence corresponding to the best outcome by solving some kind of optimization problem.

This article brings altogether previous research made in [3–6] regarding linear discrete-time system stabilization with MPC approach. Family of control synthesis algorithms previously developed in these articles have promising properties, but they also show some peculiar quirks when applied in practice. The goal of this article is to demonstrate these quirks and to suggest how to construct an effective controller operating in a feedback loop using these algorithms.

In addition, an application of the proposed controller design to stabilization in impulse cognitive maps was explored. This example is valuable on its own. Cognitive maps in general (i.e. in various different formalizations) are widely used as a formalization for various convoluted and complex systems. This particular variant of cognitive maps considered in section 2 was first proposed in [7]. It combines concepts of cognitive maps and linear systems, brining together versatility of the former and well-known control algorithms developed for the latter. Thus, since then, impulse cognitive maps became a separate field of research.

State vector of a cognitive map represents state of a system modelled in this way. It would be natural to define bringing it to a desired value as an objective of its control. But this objective partially contradicts another one, which is mandatory to keep the system operational: extinguishing state impulses. Because of this contradiction aimed stabilization was not even considered. The MPC approach allows to combine multiple partially contradicting objectives in order of their importance in a single controller. It allowed to develop an effective and safe way of aimed stabilization as a by-product of this research.

1. Stabilizing a linear system

1.1. Problem statement. Let there be a discrete-time linear system

$$x_{k+1} = Ax_k + Bu_k, k \in \mathbb{Z} \tag{1}$$

and constraints on its controls

$$u_{\min} \preccurlyeq u_k \preccurlyeq u_{\max},\ k \in \mathbb{Z}, \tag{2}$$

were $x_k, x_{k+1} \in \mathbb{R}^n$ are consecutive system's states, $u_k \in \mathbb{R}^r$ is an applied control signal and $u_{\min}, u_{\max} \in \mathbb{R}^r$ are vectors of minimum and maximum values for scalar components of control signal. For such systems it is a common task to stabilize them, i.e. to bring their state vector x eventually to zero (and keep it there in case of external disturbance). This section is dedicated to solving this rather old problem with MPC approach.

1.2. Common notes about computational experiments. There are some common things throughout the article which should be made clear beforehand in order to avoid repetition. The material below is explained with results of computational experiments performed with two system models a and b. They both have 5-dimensional states x and 2-dimensional controls u. Their matrices and initial states are the following:

$$A_a = \left(\begin{array}{c|cc|cc} 1.1 & & & & \\ \hline & 1.1\cdot\sin(1.07) & -1.1\cdot\cos(1.07) & & \\ & 1.1\cdot\cos(1.07) & 1.1\cdot\sin(1.07) & & \\ \hline & & & 0.8\cdot\sin(0.57) & -0.8\cdot\cos(0.57) \\ & & & 0.8\cdot\cos(0.57) & 0.8\cdot\sin(0.57) \end{array}\right),$$

$$A_b = \begin{pmatrix} 2.06 & 0.69 & -0.87 & -2.37 & -0.12 \\ -1.05 & -0.58 & 1.24 & 2.76 & 1.02 \\ 6.05 & 3.77 & -4.15 & -8.03 & -3.75 \\ -1.41 & -0.51 & 1.1 & 2.04 & 0.93 \\ -3.86 & -1.76 & 2.98 & 6.27 & 1.91 \end{pmatrix},$$

$$B_a = B_b = \begin{pmatrix} 0.86 & 0.31 \\ -0.56 & -0.31 \\ 0.86 & 0.16 \\ 0.04 & 0.79 \\ -0.57 & 0.47 \end{pmatrix}, \quad x_{0\,a} = x_{0\,b} = \begin{pmatrix} 0.09 \\ -0.4 \\ -0.69 \\ -0.83 \\ -0.89 \end{pmatrix},$$

$$u_{\max\,a} = -u_{\min\,a} = \begin{pmatrix} 1.20 \\ 1.20 \end{pmatrix}, \quad u_{\max\,b} = -u_{\min\,b} = \begin{pmatrix} 0.34 \\ 0.34 \end{pmatrix}.$$

Eigenvalues of A_b are roughly $-0.139 \pm 1.102\mathrm{i}$, -0.454, $1.005 \pm 0.024\mathrm{i}$. Both these systems are unstable, i.e. those with $\max_i |\lambda_i(A)| > 1$, where $\lambda_i(A), i \in \{1, \ldots, n\}$ are eigenvalues of A in (1), are used here as examples in computational experiments. This choice was made due to the fact that stable systems are asymptotically stabilizing by themselves even without any controls (or, in other words, even when $\forall k \in \mathbb{Z}\ u_k = 0$).

All figures display how system's state 2-norms (on the ordinate) change with time (on the abscissa). For the sake of simplicity most experiments in this article will not include such common complications as state estimation from indirect measurements, effects originating from possible noises in state measurements and inside system's feedback loop. In cases where they are it will be told explicitly.

1.3. Basic trajectory choosing problem. The MPC approach suggests that information about a model of a system and its current state implicitly describes a set of possible future states for every future point of time, as well as corresponding control sequences. Thus, it is possible to just choose a desired future state from this set and get its control sequence as a by-product.

If the objective is linear system's stabilization, intuition suggests that the most desired future state is the closest one to 0. Together with constraints on control signal, it naturally defines an optimization problem

$$\begin{array}{ll} \text{minimize} & \|x_{k+s}(x_k, u_k, \ldots, u_{k+s-1})\|_2 \\ \text{considering} & u_{\min} \preccurlyeq u_{k+i} \preccurlyeq u_{\max},\ i \in \{0, \ldots, s-1\}, \end{array} \tag{3}$$

where $x_{k+s}(x_k, u_k, \ldots, u_{k+s-1})$ is a future state predicted from known current state x_k and variable future controls $u_k, \ldots, u_{k+s-1}$. Therefore, its solution gives (one of) the best possible control sequences.

While controls generated this way are the best ones from the formal point of view, their application in practice in feedback loop requires carefulness and being backed up by a somewhat sophisticated decision-making algorithm because of properties discussed further.

This optimization problem is an example of a quadratic cone program, which can be solved, for example, using the CVXOPT solver [8]. It was used in all computational experiments depicted in figures of this paper.

1.4. When prediction horizon is too long. If it is possible to stabilize the system in s steps, then it is also possible to stabilize it in $s + 1$ steps. This obvious statement has some consequences which should be made explicit. The only thing solutions of (3) guarantee is that norm of state at the end of chosen future horizon will be minimal. This strategy does not set any requirements on intermediate states. Thus, for example, if it is possible to stabilize a system in 5 steps and we choose a horizon length 10, a solution returned by a solver will almost certainly stabilize the system exactly in 10 steps, but not in 5. Fig. 1, *a*, *b* demonstrates this effect. It displays state 2-norms (on the ordinate) along stabilization trajectories produced by (3) for different prediction horizon lengths and two example systems *a* and *b*.

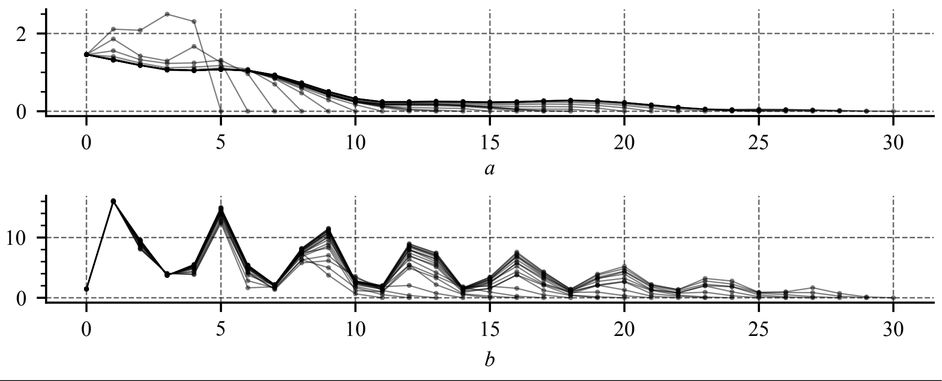


Fig. 1

1.5. Receding objective effect. The common system control workflow in algorithms based on the mathematical apparatus of the control theory (e.g. in the well-known discrete-time linear quadratic regulator) is to generate a single control signal and to apply it immediately at each step. This is convenient because in real life state measurements are not precise (and often are calculated from several previous indirect measurements), the state transition equation (1) is also not precise by itself, and the process it describes is affected by random noises. If control algorithm generates each next control signal value u considering current the most up-to-date information about system's state, it is a significant advantage.

At the same time, if we employ this straightforward approach while using the control synthesis problem (3), results would be less then perfect. If at each step only the first control from the sequence generated by (3) is used and the prediction horizon s remains the same, then the future point of time at which the control synthesis algorithm is asked to stabilize the system will recede. Considering the previously discussed effect caused by too-long prediction horizons, in this situation system stabilization would have asymptotic nature even though it is possible to finish it in a finite time.

This effect can be seen in Fig. 2, *a*, *b*. Like in the previous figure, the ordinate measures state's 2-norm. The bold line corresponds to system state's trajectory when it is controlled as described above. Here and in all following figures dashed lines correspond to trajectories assumed by the control synthesis algorithm at each step.

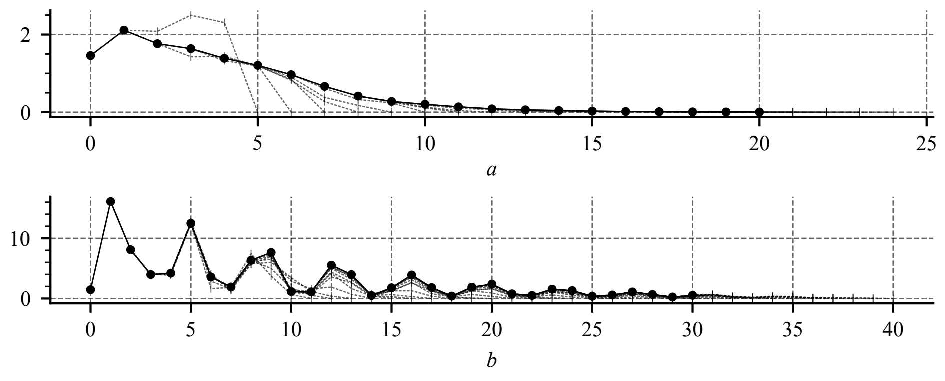


Fig. 2

1.6. Short horizon greediness. Reader may have noticed that demonstrations in Fig. 1 start with certain horizon length and asymptotic stabilization algorithm on Fig. 2 employs the same minimal horizon lengths. This is intentional. For system *a* and its initial state $x_{0\,a}$ it is required at least 5 steps to stabilize it completely, while system *b* with initial state $x_{0\,b}$ require 11 steps. When prediction horizon is too short, stabilization performance degrade in completely different way, which requires separate discussion.

Let us see what may happen on an example. Fig. 3, *a*, *b* shows the same stabilization strategy as on Fig. 2: at each step optimization problem (3) is solved for a fixed prediction horizon and the first control from generated sequence is applied to the system. But this time the chosen prediction horizon is too short: available control capacity is not enough to make it possible to reach zero state within its duration. In particular, here prediction horizon length for systems *a* and *b* are 3 and 4 correspondingly. In this situation the algorithm (3) becomes greedy. The system starts to wander in the wrong direction until it reaches a state at which the predefined horizon length becomes large enough for stabilization, as we can see in examples on Fig. 3. After this point system stabilization is asymptotic in the same way as discussed in section 1.5 and demonstrated in Fig. 2.

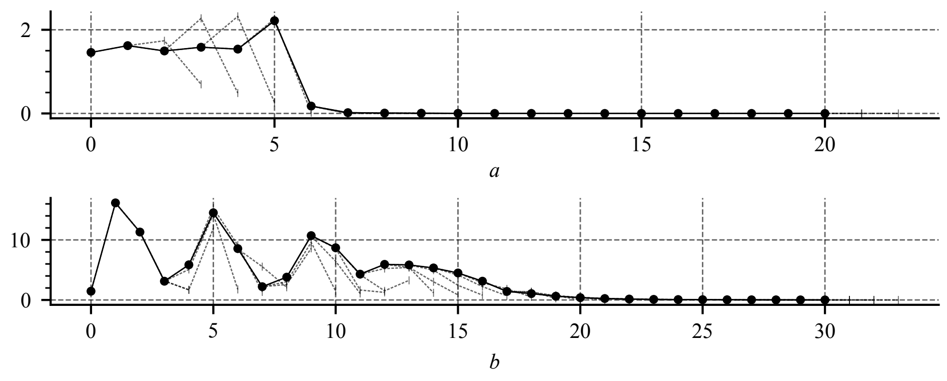


Fig. 3

It is also not guaranteed that the system will reach this point of asymptotic convergence. Under such greedy control the system's state may wander indefinitely without eventual stabilization (like on Fig. 4, *a*) or even may grow infinitely (like on Fig. 4, *b*), even though stabilization was apriori possible. Fig. 4, *a* and Fig. 4, *b* show the same control strategy, like the previous one, but for prediction horizons 2 and 1 correspondingly. Fig. 4, *b* has logarithmic ordinate for clarity.

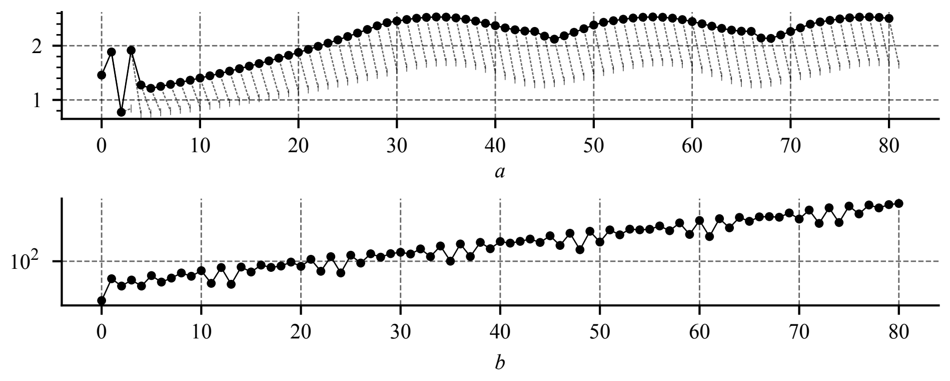


Fig. 4

1.7. In search for an optimal prediction horizon. Considering findings from above, it becomes apparent that this control strategy is not prefect when applied in feedback control loop and thus requires modification. A long prediction horizon gives more chances to eventually stabilize the system. A short one sometimes makes it stabilizing faster, but gives it more chances to not stabilize at all (even if it was apriori possible). Thus, to achieve the best results it is required to chose the right horizon length which is not too big and not too short for a particular considered initial state. But the whole point is that minimum number of steps required for stabilization is not known beforehand.

If there is enough computing power, the best strategy is to try every horizon length starting from 1 until there is a 0-state predicted at the end of the horizon. If it is expected that real system's trajectory for some reason diverges from the predicted one, we need to repeat this search at each discrete point of time to generate next best trajectory and corresponding next control signal.

An example of results obtained from this process can be seen in Fig. 5, *a*, *b*. In order to demonstrate trajectory adaptation a uniformly distributed noise was additively applied to system's state at each step. In particular, it was random vectors $\mathcal{U}^{\times n}_{[-0.03,\,0.03]}$ and $\mathcal{U}^{\times n}_{[-0.1,\,0.1]}$ for systems *a* and *b* correspondingly. These two examples on Fig. 5 demonstrate fast stabilization at the beginning of experiments. Rest of their duration demonstrate capability of this approach to keep the system stable despite random perturbations affecting the system.

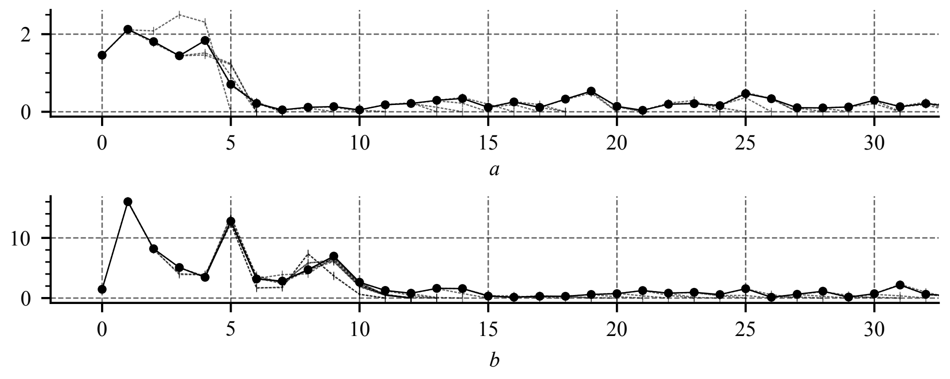


Fig. 5

Faster search strategies. Of course, the search process described above will not stop if it is not possible to stabilize the system at all, thus there should be a maximum length at which the algorithm will give up. It should be noted that computational complexity of solving (3) increases with horizon length, so in practice this maximum prediction horizon length is limited by capabilities of involved hardware. Also, this primitive search process can potentially be further enhanced with some clever search strategy to make it faster and thus to allow considering longer prediction horizons on the same hardware.

When the trajectory synthesis is performed for the first time, it is reasonable to use some modification of the binary search algorithm [9, section 6.2.1] instead of just trying each length starting from one. Considering different computation time of solving (3) for different prediction horizon lengths, it may be beneficial to have a custom specifically tuned strategy for choosing next horizon length to check instead of just checking the middle one like in the classic binary search algorithm. This strategy may be based on some game-theoretic representation of the search process and/or on some heuristic based on the current state.

At the subsequent points of time it is reasonable to consider that current state did not diverge too much from the last optimal trajectory computed due to some unknown noise impact. Thus, current optimal horizon length is probably equal to the previous one reduced by 1 or close to this value.

Precision tuning. Considering approximative nature of results returned by iterative solvers and floating point computations in general, it becomes apparent that calculated best end-state prediction for a given horizon will almost certainly be imprecise. In particular, computational results will almost certainly give a non-zero end-state even when the solution of (3) is actually 0. Thus, we are forced to chose some constant ε and to consider all such states x that $\|x\| < \varepsilon$ to be zero-states when searching for an optimal prediction horizon length with one of the procedures described above.

This well-known fact turns into an important implementation detail, because if this ε is too small (in reference to precision of the iterative solver being used), the algorithm will almost certainly miss the right horizon length and return a larger one. Thus, it is important to tune the solver's precision and ε used in the horizon search procedure.

Reusing previous optimization results. Another note should be taken about reusing trajectories generated by solving (3) at previous (discrete) points of time. In ideal situation with precise model and absence of noises it is the right choice to generate the best trajectory once and apply corresponding controls one after another. But, as it was already discussed, it is a common situation that real trajectory diverges from predicted one for various reasons and thus previously computed controls are no longer useful. At the same time recalculating optimal trajectory (with corresponding optimal controls) at each step from scratch even though real trajectory diverged not so much seems to be wasteful.

Nevertheless, previous optimization results can be used to speed-up subsequent computations. The remaining part of previously computed control sequence can be used as an initial point for an iterative solver (such as the interior-point one implemented in the CVXOPT library [8]), as well as some other its internal variables.

Similarly, optimization results for one horizon length can be reused as a first approximation when solving (3) for another, which can boost computation speed.

1.8. Fixed state-space. Algorithm's behaviour when optimization horizon is too short demonstrated in section 1.6 suggests, that state's norm is not a completely valid representation of work amount yet to be done in order to stabilize the system. Due to this the optimization problem (3) choses wrong intermediate objective states at the end of prediction horizon in this case.

As it was mentioned above, available computation resources may be extremely limited in some situations, which can prevent us from simply increasing the prediction horizon due to growing computation complexity of solving (3). In this case we are forced to use another approach: to alter the objective function of problem (3) in some way while keeping prediction horizon short.

One possible way to do it is to transform system's state-space in a way that would represent amount of stabilization work required for a state more explicitly. It is discussed in detail in [6]. In short, it suggests to use state-space transformation $x' = P^{-1}x$ and minimize a weighted norm of predicted transformed future state. P is proposed to be such that $A = PJP^{-1}$ where J is a real Jordan decomposition of A. Without loss of generality (and for more comprehensible weight choices), let us assume that 2-norm of each column of P is equal to 1.

The optimization problem (3) thus transforms into

$$\begin{aligned} &\text{minimize} && \|\Upsilon P^{-1} x_{k+s}(x_k, u_k, \ldots, u_{k+s-1})\|_2 \\ &\text{considering} && u_{\min} \preccurlyeq u_{k+i} \preccurlyeq u_{\max},\ i \in \{0, \ldots, s-1\}, \end{aligned} \tag{4}$$

where Υ is a diagonal matrix of (possibly hyperreal) weight coefficients.

This linear state-space transformation allows to decompose the system into multiple subsystems corresponding to different real Jordan blocks in J and thus having feedback loop independent from each other. It allows to explicitly assign weight coefficients according to optimization priority of each of these subsystems. While this approach can help, it always requires some experimentation regarding coefficients of Υ.

Fig. 6, *a*, *b* demonstrates examples of short-horizon stabilization with fixed state-space (black line). Trajectories from Fig. 4 are duplicated here for comparison (grey line). Prediction horizon limits are the same as on Fig. 4: 2 and 1 for systems *a* and *b* correspondingly.

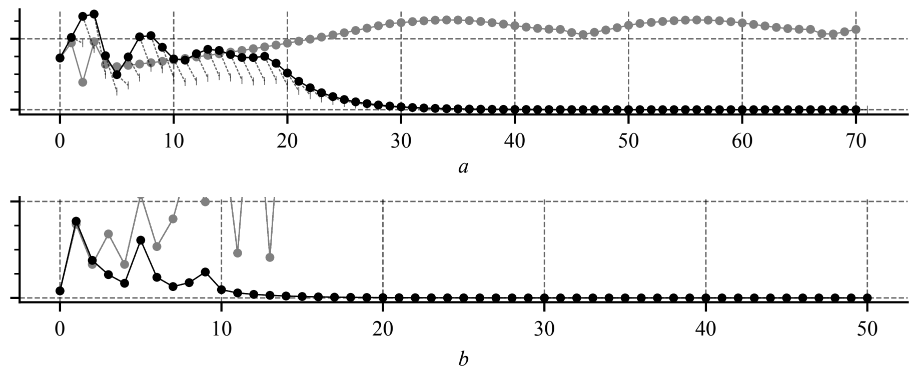


Fig. 6

Take note, that A_a is already in real Jordan normal form. Thus, on Fig. 6, *a* we also see a comparison of resulting trajectories obtained with different coefficients of Υ: when it is an identity matrix and when $\Upsilon = \operatorname{diag}[2.49, 2.49, 2.49, 0.32, 0.32]$ (considering that P_a is an identity matrix). As we can see, this choice of custom Υ helped to stabilize the system *a* event with this short horizon length.

Fig. 6, *b* demonstrates, that fixed state-space with right weight coefficients allows to stabilize the system even if the controller was originally not capable of this. Here Υ is an identity matrix.

This approach is also useful in less dramatic cases. Fixed state-space with good weights can also improve stabilization speed with short prediction horizon in cases, when controller is capable to stabilize the system even without this trick.

For some specific suggestions about how to choose coefficients see [6, Section 5]. In particular, it is advisable to choose the same coefficients of Υ corresponding to the same rotation real Jordan block in the decomposition. Exotic cases with A having generalized eigenvectors require usage of hyperreal-valued coefficients in Υ for best results. More specifically, it is required that if there is a sequence of generalized eigenvectors $v_1, \ldots, v_p$ such that $(A - I)v_1 = 0$ and $(A - I)v_{i+1} = v_i$ for $i = \{1, \ldots, p-1\}$, then corresponding weight coefficients $\upsilon_1, \ldots, \upsilon_p$ should be such that $\frac{\upsilon_{i+1}}{\upsilon_i}$ would be positive infinitesimal hyperreal numbers, i.e. such that $\forall a > 0, a \in \mathbb{R}\ 0 < \frac{\upsilon_{i+1}}{\upsilon_i} < a$. For instance, it can be done like this: $\upsilon_i = r_i d^{i-1}$. Here and further d is a positive infinitesimal hyperreal, i.e. such hyperreal number that $\forall a > 0, a \in \mathbb{R}\ 0 < d < a$. For complex generalized eigenvectors a combination of these two advises should be used.

1.9. Hyperreal-valued optimization problem decomposition. The section 1.8 casually suggested to formulate a convex optimization problem with hyperreal-valued objective function in some cases. Now it is important to describe how can it be solved in practice.

More detailed description of the hyperreal line can be found in [10–13]. Proof of solution existence can be found in [6, section 6]. General way of decomposition in such problems is described and proven in [6, section 7].

For our purposes here what is important is that an optimization problem with hyperreal-valued objective function and real-valued parameters

$$\begin{aligned} &\text{minimize} && \sum_{i=0}^{p-1} d^i f_i(x) \\ &\text{considering} && x \in \mathcal{C} \subset \mathbb{R}, \end{aligned} \tag{5}$$

where d is an infinitesimal and f_i are convex real-valued functions, can be decomposed into a sequence of real-valued optimization problems as follows.

Let there be a convex optimization problem $\mathcal{F}_0$ defined as

$$\begin{aligned} &\text{minimize} && f_0(x) \\ &\text{considering} && x \in \mathcal{C} \subset \mathbb{R}, \end{aligned}$$

whose solution is χ_0 and a recursive sequence of problems $\mathcal{F}_l$ defined for $l \in \{1,\ldots$ $\ldots,p-1\}$ as

$$\begin{aligned} &\text{minimize} && f_l(x) \\ &\text{considering} && x \in \mathcal{C} \subset \mathbb{R}, \\ & && f_i(x) = f_i(\chi_i) \text{ for } i \in \{0,\ldots,l-1\}, \end{aligned}$$

whose solution is χ_l. Then the solution χ_{p-1} of problem $\mathcal{F}_{p-1}$ is the solution of (5).

1.10. Secondary objectives and cascade optimization. There is a fact about solutions of problems (3) and (4) which was ignored until this point: their solutions are almost certainly not unique if $\dim(u)\cdot s > \dim(x)$. This gives us an opportunity to improve some additional objectives of secondary importance.

For example, if the system experiences some kind of increased deterioration when scalar elements of applied control vectors u reach their corresponding limits in $u_{\min}$ or $u_{\max}$, we can choose minimization of control signals as our second objective. This way (3) transforms into

$$\begin{aligned} &\text{minimize} && \|x_{k+s}(x_k, u_k, \ldots, u_{k+s-1}))\|_2 + d \sum_{i=k}^{k+s-1} \|u_i\|_2 \\ &\text{considering} && u_{\min} \preccurlyeq u_{k+i} \preccurlyeq u_{\max},\ i \in \{1,\ldots,s-1\}. \end{aligned}$$

Alternatively, if there is a high probability of some kind of crash when state's norm in the original state-space at some point of time is large, we can choose a more mild trajectory with

$$\begin{aligned} &\text{minimize} && \|\Upsilon P^{-1} x_{k+s}(x_k, u_k, \ldots, u_{k+s-1}))\|_2 + d\cdot\alpha \\ &\text{considering} && u_{\min} \preccurlyeq u_{k+i} \preccurlyeq u_{\max}, && i \in \{1,\ldots,s-1\}, \\ & && \|x_{k+p}(x_k, u_k, \ldots, u_{k+p-1}))\|_2 \le \alpha, && p \in \{1,\ldots,s-1\}, \end{aligned} \tag{6}$$

where α is an auxiliary optimization parameter.

This pattern of composite optimization problem construction can be used to implement a broad variety of additional rankings among already applicable stabilization trajectories.

But it should be taken with care: an ill-chosen secondary objective in combination with the same prediction horizon used at each step (like in experiment on Fig. 2) can produce a situation when controller prevents the system from blowing up, but never actually stabilize it. Nevertheless, it is not possible when the horizon search strategy discussed in section 1.7 is employed.

2. Controlling a cognitive map

Let us assume a more sophisticated example of linear system: an impulse cognitive map which consists of n vertices. Current state of the impulse cognitive map is represented by a real-valued n-dimensional vector x which evolves in discrete time according to the following equations:

$$\Delta x_{k+1} = A\Delta x_k + Bu_k, \quad k \in \mathbb{Z}, \tag{7}$$

$$\Delta x_k := x_k - x_{k-1}, \quad k \in \mathbb{Z}, \tag{8}$$

$$u_{\min} \preccurlyeq u_k \preccurlyeq u_{\max}, \quad k \in \mathbb{Z}, \tag{9}$$

where Δx_k, Δx_{k+1} are n-dimensional consecutive impulses of the cognitive map at consecutive points of time, x_{k-1} and x_k are n-dimensional consecutive states, u_k is an external control vector, A and B are $n \times n$ and $n \times r$ matrices. The task is the same as with ordinary linear system discussed above: to stabilize the cognitive map, i.e. to bring Δx to (the neighbourhood of) zero in finite time.

2.1. Wandering stability. If the only objective is to prevent the system from blowing up, it is enough to stabilize impulses Δx in the same way as it was previously done with (1), (2).

On Fig. 7, *a*, *b* we can see how $\|x\|_2$ changes in time when Δx is additively affected by a uniformly distributed noise ($\mathcal{U}^{\times n}_{[-0.03,\,0.03]}$ and $\mathcal{U}^{\times n}_{[-0.1,\,0.1]}$ for systems *a* and *b* correspondingly) and stabilized employing the horizon searching algorithm described in section 1.7. By the way, Fig. 5 displays $\|\Delta x\|_2$ at the beginning of the same pair of experiments.

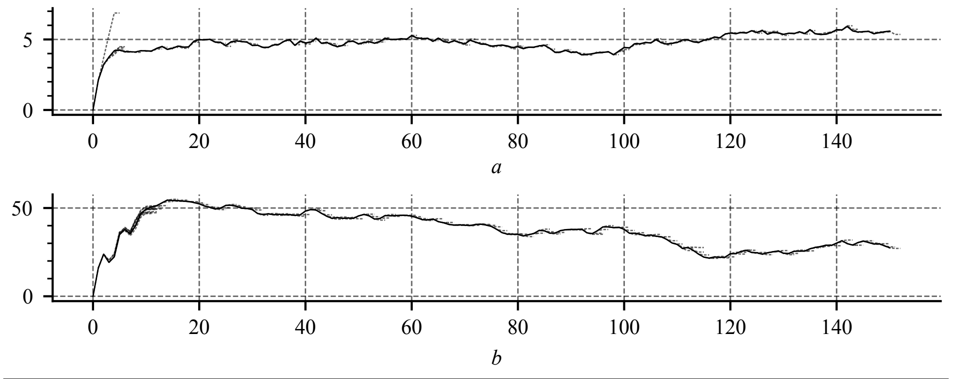


Fig. 7

As we can see, after initial stabilization behaviour of controlled cognitive map resembles the Wiener process a lot. It thus arises a question of whether it is possible to prevent such state wandering and to what degree.

2.2. Reachable stable states of a cognitive map. If the objective is to stabilize both the impulse Δx and the state x, it is natural to consider compound state $(\Delta x^{\mathsf{T}}, (x - x_{\text{aim}})^{\mathsf{T}})^{\mathsf{T}}$, where x_{aim} is the objective state. Without loss of generality, x_{aim} will be further considered to be equal to 0.

From this point of view, equations (7), (8) can be rewritten as

$$\begin{pmatrix} \Delta x_{k+1} \\ x_{k+1} \end{pmatrix} = \underbrace{\begin{pmatrix} A & 0 \\ I & I \end{pmatrix}}_{A_{\text{cogn.}}} \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} + \underbrace{\begin{pmatrix} B \\ B \end{pmatrix}}_{B_{\text{cogn.}}} u_k. \tag{10}$$

This representation brings hope that maybe it is possible to stabilize a cognitive map while bringing it to a predefined state. Let us explore when and to what degree is it possible.

Lemma 1. *For all* $k \in \mathbb{N}$ $\operatorname{Im}\left(A^{n-1}B \mid \cdots \mid B\right) = \operatorname{Im}\left(A^{k+n-1}B \mid \cdots \mid B\right)$.

Proof. Let $p_A(\lambda) = \lambda^n + \sum_{i=0}^{n-1} c_i\lambda^i$ be the characteristic polynomial of A. Then, from the Cayley–Hamilton theorem, $p_A(A) = 0$ or $A^n = -\sum_{i=0}^{n-1} c_iA^i$. With this fact we can write that for all $k \geq 0$

$$\left(A^{k+n}B \mid A^{k+n-1}B \mid \cdots \mid B\right) = \left(-\textstyle\sum_{i=0}^{n-1} c_iA^{k+i}B \mid A^{k+n-1}B \mid \cdots \mid B\right) =$$

$$= \left(A^{k+n-1}B \mid \cdots \mid B\right)\left(\begin{array}{c|ccc} -c_{n-1}I_r & I_r & & \\ \vdots & & \ddots & \\ -c_0I_r & & & \\ \hline 0 & & & \\ \vdots & & & \\ 0 & & & I_r \end{array}\right),$$

and this way $\operatorname{Im}\left(A^{n-1}B \mid \cdots \mid B\right) = \cdots = \operatorname{Im}\left(A^{k+n-1}B \mid \cdots \mid B\right)$. □

Lemma 2. *For all* $k \in \mathbb{N}$ $\operatorname{Im}\left(\sum_{i=0}^{k} A^iB \mid \cdots \mid B\right) = \operatorname{Im}\left(A^kB \mid \cdots \mid B\right)$.

Proof.

$$\left(\textstyle\sum_{i=0}^{k} A^iB \mid \cdots \mid B\right)\begin{pmatrix} I_r & & & & \\ -I_r & I_r & & & \\ & \ddots & \ddots & & \\ & & & I_r & \\ & & & -I_r & I_r \end{pmatrix} = \left(A^kB \mid \cdots \mid B\right).$$

□

Theorem 1 (Necessary conditions for cognitive map controllability)**.** *If linear system* (10) *is controllable (considering controls u to be unbounded), then linear system* (7) *is also controllable and* $\operatorname{rank} B = n$.

Proof. It is a well-known fact that a discrete-time linear system with unconstrained controls is controllable iff its controllability matrix has full rank. For system (10) it can be decomposed as follows:

$$\left(A_{\text{cogn.}}^{2n-1}B_{\text{cogn.}} \mid \cdots \mid B_{\text{cogn.}}\right) = \left(\begin{array}{c|c|c} A^{2n-1}B & \cdots & B \\ \hline \sum_{i=0}^{2n-1} A^iB & \cdots & B \end{array}\right) =$$

$$= \begin{pmatrix} I_n & 0 \\ -I_n & I_n \end{pmatrix}^{-1}\left(\begin{array}{c|c|c|c} A^{2n-1}B & \cdots & AB & B \\ \hline \sum_{i=0}^{2n-2} A^iB & \cdots & B & 0 \end{array}\right) =$$

$$= \begin{pmatrix} I_n & 0 \\ -I_n & I_n \end{pmatrix}^{-1}\begin{pmatrix} -I_n & A \\ 0 & I_n \end{pmatrix}^{-1}\left(\begin{array}{c|c|c|c|c} \sum_{i=1}^{2n-2} A^iB & \cdots & AB & 0 & -B \\ \hline \sum_{i=0}^{2n-2} A^iB & \cdots & \cdots & B & 0 \end{array}\right) =$$

$$= \begin{pmatrix} I_n & 0 \\ -I_n & I_n \end{pmatrix}^{-1}\begin{pmatrix} -I_n & A \\ 0 & I_n \end{pmatrix}^{-1}\begin{pmatrix} -I_n & I_n \\ 0 & I_n \end{pmatrix}^{-1}\left(\begin{array}{c|c|c|c|c} B & \cdots & B & 0 & B \\ \hline \sum_{i=0}^{2n-2} A^iB & \cdots & \cdots & B & 0 \end{array}\right).$$

where I_n is $n \times n$ identity matrix. It shows explicitly, that

$$\operatorname{rank}\left(A_{\text{cogn.}}^{2n-1}B_{\text{cogn.}} \mid \cdots \mid B_{\text{cogn.}}\right) = \operatorname{rank}\left(\begin{array}{c|c|c|c|c} B & \cdots & B & 0 & B \\ \hline \sum_{i=0}^{2n-2} A^iB & \cdots & \cdots & B & 0 \end{array}\right).$$

Thus, it becomes obvious that

$$\operatorname{rank}\left(A_{\text{cogn.}}^{2n-1}B_{\text{cogn.}} \mid \cdots \mid B_{\text{cogn.}}\right) \leq \operatorname{rank} B + \operatorname{rank}\left(\textstyle\sum_{i=0}^{2n-2} A^iB \mid \cdots \mid B\right).$$

From previous lemmas we can see that

$$\operatorname{rank}\left(\sum_{i=0}^{2n-2} A^i B \;\middle|\; \cdots \;\middle|\; B\right) = \operatorname{rank}\left(A^{2n-2}B \mid \cdots \mid B\right)$$

and

$$\operatorname{rank}\left(A^{2n-2}B \mid \cdots \mid B\right) = \operatorname{rank}\left(A^{n-1}B \mid \cdots \mid B\right).$$

Finally, $\operatorname{rank}\left(A_{\text{cogn.}}^{2n-1}B_{\text{cogn.}} \;\middle|\; \cdots \;\middle|\; B_{\text{cogn.}}\right) \le \operatorname{rank} B + \operatorname{rank}\left(A^{n-1}B \mid \cdots \mid B\right)$ and if linear system (10) is controllable, then $\operatorname{rank}\left(A_{\text{cogn.}}^{2n-1}B_{\text{cogn.}} \;\middle|\; \cdots \;\middle|\; B_{\text{cogn.}}\right) = 2n$, while $\operatorname{rank} B \le n$ and $\operatorname{rank}\left(A^{n-1}B \mid \cdots \mid B\right) \le n$. From all of this we make a conclusion that $\operatorname{rank} B = \operatorname{rank}\left(A^{n-1}B \mid \cdots \mid B\right) = n$. □

From this theorem we can see that in general case a cognitive map's state and impulse together are not necessarily controllable. And even worse, having $\operatorname{rank} B = n$ is a necessary condition for this joint controllability. It implies that it is required for the control vector u to be at least n-dimensional, which is hard to achieve in practice.

Knowing that not all stable states are reachable normally, it becomes important to know, which stable states still can be reached.

Theorem 2 (Reachable stable states of cognitive map with fully controllable impulses)**.** *If the impulse linear system* (7) *(disregarding control constraints) is fully controllable, then the set of reachable stable states of the cognitive map* (7), (8) *with unconstrained controls is a hyperplane described with predicate* $R_\infty(x) = \exists v \in \mathbb{R}^r : (I_n - A)x = A\Delta x_k + (I_n - A)x_k + Bv$.

Proof. From (7), (8) the formula for future state x_{k+s} can be derived as follows:

$$\Delta x_{k+i} = A^i \Delta x_k + \left(A^{i-1}B \mid \cdots \mid B\right)(u_k^\mathsf{T}, \ldots, u_{k+i-1}^\mathsf{T})^\mathsf{T},$$

$$x_{k+s} = x_k + \sum_{i=1}^{s} \Delta x_{k+i} = x_k + \sum_{i=1}^{s} A^i \Delta x_k + \left(\sum_{i=0}^{s-1} A^i B \;\middle|\; \cdots \;\middle|\; B\right)(u_k^\mathsf{T}, \ldots, u_{k+s-1}^\mathsf{T})^\mathsf{T}.$$

Thus, set of states reachable stable in s steps can be described with following predicate:

$$\begin{aligned} R_s(x_{k+s}) = {}& \exists u_k, \ldots, u_{k+s-1} \in \mathbb{R}^r: \\ & \left(\begin{pmatrix} 0 \\ x_{k+s} \end{pmatrix} = \begin{pmatrix} A^s & 0 \\ \sum_{i=1}^{s} A^i & I_n \end{pmatrix} \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} + \right. \\ & \left. + \left(\begin{array}{c|c|c} A^{s-1}B & \cdots & B \\ \hline \sum_{i=0}^{s-1} A^i B & \cdots & B \end{array} \right) (u_k^\mathsf{T}, \ldots, u_{k+s-1}^\mathsf{T})^\mathsf{T} \right). \end{aligned} \tag{11}$$

Repeating matrix transformations from proof of the Necessary conditions for cognitive map controllability theorem, the equation in (11) can be equivalently transformed by left-multiplying it with invertible matrix

$$\begin{pmatrix} -I_n & I_n \\ 0 & I_n \end{pmatrix} \begin{pmatrix} -I_n & A \\ 0 & I_n \end{pmatrix} \begin{pmatrix} I_n & 0 \\ -I_n & I_n \end{pmatrix} = \begin{pmatrix} A & I_n - A \\ -I_n & I_n \end{pmatrix}, \tag{12}$$

which gives as a result

$$\begin{aligned} R_s(x_{k+s}) = {}& \left(\exists u_k, \ldots, u_{k+s-1} \in \mathbb{R}^r: \right. \\ & \begin{pmatrix} I_n - A \\ I_n \end{pmatrix} x_{k+s} = \begin{pmatrix} A & I_n - A \\ \sum_{i=1}^{s-1} A^i & I_n \end{pmatrix} \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} + \\ & \left. + \left(\begin{array}{c|c|c|c|c} B & \cdots & B & 0 & B \\ \hline \sum_{i=0}^{s-2} A^i B & \cdots\cdots & & B & 0 \end{array} \right) \begin{pmatrix} u_k \\ \vdots \\ u_{k+s-1} \end{pmatrix} \right) = \end{aligned}$$

$$= \Bigg(\exists u_k, \ldots, u_{k+s-2} \in \mathbb{R}^r \colon$$

$$\Bigg(x_{k+s} = \left(\sum_{i=1}^{s-1} A^i \;\middle|\; I_n \right) \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} +$$

$$+ \left(\sum_{i=0}^{s-1} A^i B \;\middle|\; \cdots \;\middle|\; B \right) (u_k^\intercal, \ldots, u_{k+s-2}^\intercal)^\intercal \Bigg) \wedge$$

$$\wedge \Bigg(\exists u_{k+s-1} \in \mathbb{R}^r : (I_n - A) x_{k+s} = \left(A \;\middle|\; I_n - A \right) \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} +$$

$$+ B \Big(\sum_{i=0}^{s-3} u_{k+i} + u_{k+s-1} \Big) \Bigg) \Bigg).$$

With substitution $v = \sum_{i=0}^{s-3} u_{k+i} + u_{k+s-1}$ it transforms into

$$R_s(x_{k+s}) = \Bigg(\exists u_k, \ldots, u_{k+s-2} \in \mathbb{R}^r \colon$$

$$x_{k+s} = \left(\sum_{i=1}^{s-1} A^i \;\middle|\; I_n \right) \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} +$$

$$+ \left(\sum_{i=0}^{s-2} A^i B \;\middle|\; \cdots \;\middle|\; B \right) (u_k^\intercal, \ldots, u_{k+s-2}^\intercal)^\intercal \Bigg) \wedge$$

$$\wedge \left(\exists v \in \mathbb{R}^r : (I_n - A) x_{k+s} = \left(A \;\middle|\; I_n - A \right) \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} + Bv \right). \tag{13}$$

If the impulse linear system (7) is fully controllable, then $\left(A^{s-2} B \;\middle|\; \cdots \;\middle|\; B \right)$ has full rank at least for all $s \geq n+1$. From lemma 2, $\left(\sum_{i=0}^{s-2} A^i B \;\middle|\; \cdots \;\middle|\; B \right)$ thus also has full rank for the same values of s. This makes the first clause of (13) always true for all sufficiently big s. Because of this, set of all eventually-reachable stable states can be described with predicate

$$R_\infty(x) = \exists v \in \mathbb{R}^r : (I_n - A) x = \left(A \;\middle|\; I_n - A \right) \begin{pmatrix} \Delta x_k \\ x_k \end{pmatrix} + Bv. \tag{14}$$

□

Corollary. *If the impulse linear system* (7) *(disregarding control constraints) is fully controllable, then the set of reachable stable states of the cognitive map* (7), (8) *with unconstrained controls is* $(\operatorname{rank} B)$*-dimensional hyperplane.*

Proof. If matrix $(I_n - A)$ is invertible (i.e. A does not have eigenvalues equal to 1), then the set of reachable stable states is

$$R_\infty = \{ x_k + (I_n - A)^{-1} A \Delta x_k + (I_n - A)^{-1} B v \mid v \in \mathbb{R}^r \}, \tag{15}$$

which is obviously a $(\operatorname{rank} B)$-dimensional hyperplane. The other case when A has $l > 0$ eigenvalues equal to 1 will be discussed further.

If matrix $(I_n - A)$ is not invertible, then its image $\operatorname{Im}(I_n - A)$ is a linear subspace of $\mathbb{R}^n$ with dimensionality less than n. Let there be a singular value decomposition $(I_n - A) = UDV^\intercal$. Matrix $(I_n - A)$ is not invertible, so the diagonal matrix D from the decomposition has less dimensions than n. Then $(I_n - A)^+ := VD^{-1}U^\intercal$ is the Moore–Penrose inverse of $(I_n - A)$.

Let there also be a matrix $U_\complement$ such that $\left(U \;\middle|\; U_\complement \right)$ is an orthonormal $n \times n$ matrix. It makes $U_\complement$ an orthonormal basis of linear subspace $(\operatorname{Im}(I_n - A))^\perp$ orthogonal to $\operatorname{Im}(I_n - A)$. Then, $U_\complement^\intercal (I_n - A) = 0$ and $U_\complement^\intercal A = U_\complement^\intercal (I_n - (I_n - A)) = U_\complement^\intercal$.

This way (14) can be rewritten in the following way:

$$R_\infty(x) = \left(\exists v \in \mathbb{R}^r : \begin{pmatrix} U^\intercal \\ U_{\complement}^\intercal \end{pmatrix}(I_n - A)x = \begin{pmatrix} U^\intercal \\ U_{\complement}^\intercal \end{pmatrix}\big((I_n - A)x_k - A\Delta x_k + Bv\big)\right) =$$

$$= \exists v \in \mathbb{R}^r : \big(U^\intercal (I_n - A)x = U^\intercal \big((I_n - A)x_k - A\Delta x_k + Bv\big)\big) \wedge$$

$$\wedge \left(0 = U_{\complement}^\intercal \Delta x_k + U_{\complement}^\intercal Bv\right) =$$

$$= \exists v \in \mathbb{R}^r : \Big(\underbrace{VD^{-1}U^\intercal}_{(I_n-A)^+}(I_n - A)x = \underbrace{VD^{-1}U^\intercal}_{(I_n-A)^+}\big((I_n - A)x_k -$$

$$- A\Delta x_k + Bv\big)\Big) \wedge$$

$$\wedge \left(U_{\complement}^\intercal Bv = -U_{\complement}^\intercal \Delta x_k\right). \qquad (16)$$

The impulse linear system (7) is considered to be controllable. Thus, $\big(A^{n-1}B \mid \cdots \mid B\big)$ has full rank and because of that $\big((A^{n-1} - A^{n-2})B \mid \ldots \mid (A - I_n)B \mid B\big)$ has also full rank. This means that

$$\forall x^* \exists u_1, u_2 : -(I_n - A)\big(A^{n-2}B \mid \cdots \mid B\big)u_1 + Bu_2 = x^*$$

from which follows

$$\forall x^* \exists u_1, u_2 : \big(-\underbrace{U_{\complement}^\intercal (I_n - A)}_{0}\big(A^{n-2}B \mid \cdots \mid B\big)u_1 + U_{\complement}^\intercal Bu_2 = U_{\complement}^\intercal x^*\big) =$$

$$= \forall x^* \exists u_2 : \big(U_{\complement}^\intercal Bu_2 = U_{\complement}^\intercal x^*\big) = \forall x^{**} \exists u : \big(U_{\complement}^\intercal Bu = x^{**}\big).$$

In other words, matrix $U_{\complement}^\intercal B$ has rank equal to number of columns in $U_{\complement}$ (i.e. to l). This means, that there are such matrices P_1, P_2 (which can be straightforwardly calculated from matrix $U_{\complement}^\intercal B$) that statement $\big(U_{\complement}^\intercal Bv = -U_{\complement}^\intercal \Delta x_k\big)$ is equivalent to $\big(\exists w \in \mathbb{R}^{r-l} : v = P_1 w + P_2 U_{\complement}^\intercal \Delta x_k\big)$. Another subtle consequence of this fact is that $\operatorname{rank} B$ is required to be higher than number of 1-eigenvalues of A for impulse linear system (7) to be controllable.

This way, (16) transforms into

$$R_\infty(x) = \exists w \in \mathbb{R}^{r-l} : \Big(\underbrace{VV^\intercal}_{\Pr_{\operatorname{Im}(I_n-A)}} x = VV^\intercal x_k -$$

$$- (I_n - A)^+ (A + BP_2 U_{\complement}^\intercal)\Delta x_k + (I_n - A)^+ BP_1 w\Big).$$

Let there also be such matrix $V_{\complement}$ that $\big(V \mid V_{\complement}\big)$ is an orthonormal $n \times n$ matrix. Then, finally,

$$R_\infty(x) = \exists w \in \mathbb{R}^{r-l}, g \in \mathbb{R}^l : \Big(x = VV^\intercal x_k - (I_n - A)^+ (A + BP_2 U_{\complement}^\intercal)\Delta x_k +$$

$$+ (I_n - A)^+ BP_1 w + V_{\complement} g\Big) \qquad (17)$$

or, if written in more common form,

$$R_\infty = \{VV^\intercal x_k - (I_n - A)^+ (A + BP_2 U_{\complement}^\intercal)\Delta x_k + (I_n - A)^+ BP_1 w + V_{\complement} g$$

$$\mid w \in \mathbb{R}^{r-l}, g \in \mathbb{R}^l\}, \qquad (18)$$

and it is obviously a r-dimensional hyperplane. □

2.3. Aimed stabilization control synthesis in feedback loop. In total, cognitive map control has two objectives: to suppress impulses Δx and to bring system's state x as close as possible to some arbitrary state x_{aim}.

While it is not possible to actually reach an arbitrary stable state in general case, ability to reach projection of this arbitrary state is better than nothing. When the cognitive map is affected by external noise it allows to reduce state wandering at least partially. Considering that external random noises normally have more degrees of freedom than the control signal, they would change the closest reachable state at each step, so in this case it should be recalculated at each time step before synthesizing controls.

Pursuing both of the aforementioned objectives at the same time by stabilizing linear system (10) may introduce an undesirable effect: it would make a controller chose intermediate states x closer to x_{aim} at cost of corresponding Δx being further from 0. When $x_{\text{aim}} \notin R_\infty$ it would make vector $(\Delta x^\intercal, x^\intercal)^\intercal$ indefinitely wander around $(\overleftarrow{0}, x_{\text{aim}}^\intercal)^\intercal$ without actually stabilizing impulses (in the best case). It would also make prediction horizon choosing algorithm discussed in section 1.7 always choose the longest allowed one, because the objective can never be reached.

Even if objective is replaced with $(\overleftarrow{0}, (\Pr_{R_\infty} x_{\text{aim}})^\intercal)^\intercal$ the controller would require to consider larger prediction horizon than when stabilizing only Δx. The lack of prediction horizon length may produce unacceptable results like in Fig. 4.

Thus, in order to still produce adequate results on relatively limited prediction horizon, moving cognitive map's state to a predefined value should be made a secondary objective, like discussed in section 1.10. This approach is better for limited prediction horizon and will produce the same results when its length is enough for reaching both objectives at the same time.

Also, for best results the prediction horizon search discussed in section 1.7 should be applied with following slight modification: is should search for the smallest prediction horizon satisfying both primary objective $\|\Delta x\| \to 0$ and the secondary objective $\|x - \Pr_{R_\infty} x_{\text{aim}}\| \to 0$. It would produce longer prediction horizons than required just to stabilize impulse subsystem (7) in order to prevent a deadlock when impulse Δx is already equal to 0 while the state x did not reach $\Pr_{R_\infty} x_{\text{aim}}$ yet.

Having the set R_∞ represented in trivial form as a hyperplane like in (15) or (18) allows to calculate trivially the closest reachable stable state as a projection $\Pr_{R_\infty} x_{\text{aim}}$ by solving a linear least square problem. In fact, knowledge of $\Pr_{R_\infty} x_{\text{aim}}$ is only required for a stop-condition in the prediction horizon choosing algorithm. The problem $\|\Delta x_{k+s}\|_2 +$ $+ d\|x_{k+s} - \Pr_{R_\infty} x_{\text{aim}}\|_2 \to 0$ is equivalent to just $\|\Delta x_{k+s}\|_2 + d\|x_{k+s} - x_{\text{aim}}\|_2 \to 0$ under the same constraints on controls.

Fig. 8, *a*, *b* demonstrate aimed stabilization strategy discussed above in case when prediction horizon limit is enough for stabilization and impulses Δx are affected by uniformly distributed random noise at each step. Particular distributions are the same as used for experiments depicted in Fig. 5 and 7: $\mathcal{U}^{\times n}_{[-0.03,\,0.03]}$ and $\mathcal{U}^{\times n}_{[-0.1,\,0.1]}$ for systems *a* and *b* correspondingly. Objective state x_{aim} in both cases is chosen to be equal to 0.

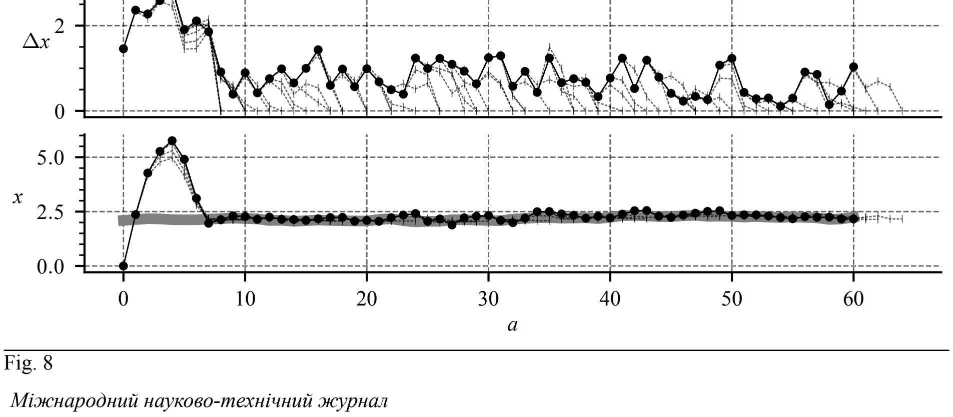
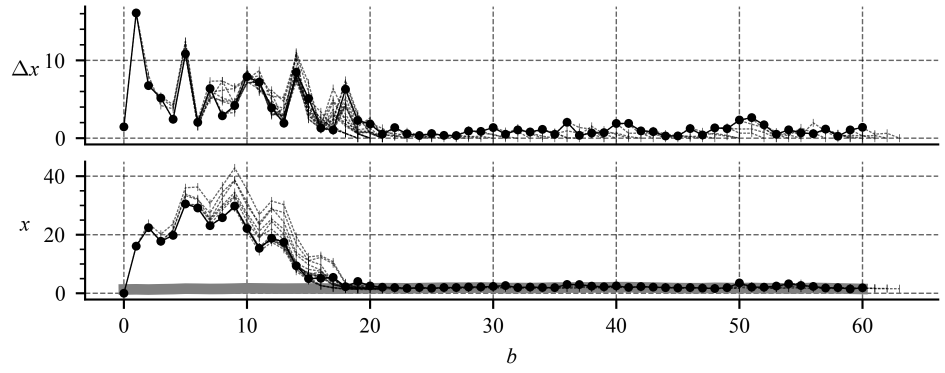


Fig. 8



A thick gray line depicts norm of $\mathrm{Pr}_{R_\infty} x_{\text{aim}}$. It can be seen that it changes slightly at each step because of random perturbations. If compared with trajectories of state x in Fig. 7, with aimed stabilization they demonstrate much less wandering while being affected by noises of the same magnitude. It is achieved at cost of less stability in impulse domain: Δx in Fig. 8 demonstrate larger distances from 0 on average than in Fig. 5. Also, as expected, initial stabilization considering both objectives required longer prediction horizon than in Fig. 5, where only impulses Δx are stabilized.

Fig. 9, *a*, *b* demonstrates the situation when prediction horizon limit (5 and 11 for systems *a* and *b* correspondingly) is enough for impulse stabilization, but is not enough to reach $\mathrm{Pr}_{R_\infty} x_{\text{aim}}$. It can be seen that under such constraints controller requires only slightly more time to reach it.

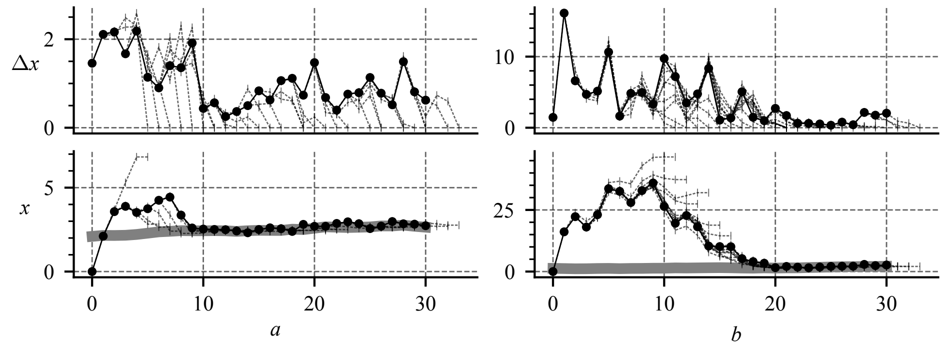


Fig. 9

Conclusion

Solutions of problem (3) produce best possible trajectories for corresponding prediction horizon length s, but adequate results can only be obtained with right horizon length: not too long and not too short. With too long prediction horizon controller becomes ”lazy” and underperforms. When it is too short, it appears that state’s 2-norm in general is not fully adequate representation for amount of ”stabilization work” yet to be done, and thus generated trajectory may lead the system in wrong direction. That is why, in order to generate a good trajectory the controller must find the shortest horizon length which is enough to be able to reach zero state at its end. This may be computationally hard, but there are a few programming tricks which can significantly increase search speed if compared with bruteforce search.

If the system is affected by random perturbations, it becomes important to update generated trajectory at each step according to next measured state. A new optimal trajectory may differ not only in corresponding values of generated control sequence, but also in its length. It may appear that after perturbation it is required slightly more or slightly less steps for stabilization. Thus, horizon length search should also be repeated. The fact that optimal horizon length normally changes only slightly is a significant help.

Cases when required prediction horizon length appears to be so big that there is not enough computational power to solve corresponding optimization problem are hard ones. If unattended, minimization of future state's 2-norm would most certainly not choose an appropriate intermediate state. For adequate results, it is required to transform the state-space in a way what would make state's 2-norm better represent amount of stabilization efforts required. While Jordan decomposition of the system matrix A in (1) gives a valuable hint for an engineer, it still requires their experience and intuition for choosing appropriate weight coefficients.

The control approach described above can be used as-is for impulse stabilization in cognitive maps if it is the only objective. If it is also important to reach certain cognitive map's state x_{aim} and remain there indefinitely, things become complicated. It appears that, counterintuitively, not all states can be reached with fully extinguished impulses at the same time if vector of control signals has less dimensions than the system's state. Nevertheless, if cognitive map's impulses are controllable, there exists a whole hyperplane of reachable stable states, which is defined by current state and impulse. Stabilization around a projection of x_{aim} onto this hyperplane can significantly decrease state wandering, which is a notorious effect appearing when only impulses are stabilized for a noised cognitive map.

М.Д. Міщенко

ПРАКТИЧНІ АСПЕКТИ КЕРУВАННЯ ЗА ПРОГНОЗНОЮ МОДЕЛЛЮ У ЛІНІЙНИХ СИСТЕМАХ ТА КОГНІТИВНИХ КАРТАХ

Міщенко Михайло Дмитрович

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Навчально-науковий інститут прикладного системного аналізу,
mdmisch@firemail.cc, mdmisch@protonmail.com

У роботі розглянуто особливості застосування підходу керування за прогнозною моделлю (model predictive control — MPC) для стабілізації лінійних систем у дискретному часі. Хоча стабілізація у лінійних системах уже добре досліджена в рамках теорії керування, більш новий підхід керування за прогнозною моделлю дозволяє отримувати швидші траєкторії стабілізації, але при більшій кількості необхідних обчислень. Завдяки суттєвому прогресові в обчислювальній потужності комп'ютерів з часів появи теорії керування, з'явилася практична можливість реалізації керування за прогнозною моделлю. Цей підхід дозволяє досягти значно кращих результатів, проте його застосування потребує значної уваги, адже його некоректне використання призводить до безлічі неочевидних і небажаних ефектів. У даній статті почергово обговорено, пояснено і на прикладах продемонстровано ці ефекти. Аналіз їхніх причин дозволив виявити вимоги до такого алгоритму керування за прогнозною моделлю, виконання яких забезпечить надійне функціонування контролера. Також з'ясувалося, що здебільшого оптимальна стабілізаційна траєкторія не є унікальною, тобто можливо обирати між оптимальними траєкторіями заради покращення якогось другорядного показника. На додачу, як приклад, який є цінним сам по собі, окремо розглянуто стабілізацію у лінійних когнітивних картах. Будучи прикладами лінійних систем у дискретному часі, лінійні когнітивні карти допускають застосування щодо їх імпульсів тих самих стратегій і алгоритмів керування. Але якщо опустити природу когнітивних карт, їх стан починає поступово змінюватися в непередбачуваному напрямку під тиском зовнішніх випадкових збурень (шуму) попри те, що стабілізуючий контролер придушує їх вплив на імпульси когнітивної карти. Здатність підходу керування за прогнозною моделлю враховувати другорядні цілі дозволила усунути цей ефект принаймні частково. У даній статті такою другорядною ціллю при пошуку стабілізаційної траєкторії було обрано досягнення когнітивною картою деякого цільового стану. Також продемонстровано, що з усього простору станів когнітивної карти лише деяка гіперплощина у ній є досяжною, якщо виходити з припущення, що її імпульс рівнятиметься нулю у кінці траєкторії.

Ключові слова: керування за прогнозною моделлю, лінійна система, когнітивна карта, лінійна когнітивна карта, стабілізація, цілеспрямована стабілізація, оптимізація, квадратичне програмування на конусі.

REFERENCES

1. Zadeh L.A., Desoer C.A. Linear system theory. New York : McGraw-Hill, 1979. ISBN: 0-88275-809-8. Reprint edition.
2. Camacho E.F., Bordons C. Model predictive control. Advanced textbooks in control and signal processing. 2 ed. London : Springer-Verlag, 2007. XXII, 405 p. ISBN: 978-1-85233-694-3. DOI: 10.1007/978-0-85729-398-5.
3. Gubarev V.F., Mishchenko M.D., Snizhko B.M. Model predictive control for discrete MIMO linear systems. *Advanced Control Techniques in Complex Engineering Systems: Theory and Applications: Dedicated to Professor Vsevolod M. Kuntsevich* / ed. by Kondratenko Y.P., Chikrii A.A., Gubarev V.F., Kacprzyk J. Cham : Springer International Publishing, 2019. P. 63–81. ISBN: 978-3-030-21927-7. DOI: 10.1007/978-3-030-21927-7_4.
4. Mishchenko M.D., Gubarev V.F. Horizon length tuning for model predictive control in linear multi input multi variable systems. *Cybernetics and Computer Engineering.* 2021. Vol. 2021, N 1 (203). P. 39–59. ISSN: 2663-2586. DOI: 10.15407/kvt203.01.039. URL: http://kvt-journal.org.ua/1634/.
5. Mishchenko M.D., Gubarev V.F. Methods of model predictive control for discrete multi-variable systems with input. *Cybernetics and Computer Engineering Journal.* 2020. Vol. 2020, N 1 (199). P. 39–58. ISSN: 2663-2586. DOI: 10.15407/kvt199.01.039. URL: http://kvt-journal.org.ua/1444/.
6. Mishchenko M. Structure-based optimization problem for model predictive control in linear multi-input multi-value systems. *International Scientific Technical Journal «Problems of Control and Informatics».* 2022. Vol. 67, N 3. P. 22–36. ISSN: 2786-6491. DOI: 10.34229/2786-6505-2022-3-2. URL: https://jais.net.ua/index.php/files/article/view/54.
7. 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. P. 26–42. ISSN: 1681-6048. URL: http://journal.iasa.kpi.ua/article/view/33504.
8. Andersen M., Dahl J., Vandenberghe L. CVXOPT: Python software for convex optimization. 2023. URL: https://cvxopt.org/ (online; accessed: 2023-07-25).
9. Knuth D.E. The art of computer programming, Volume 3: Sorting and searching. 2nd ed. Addison-Wesley Professional, 2016. P. 409–426. ISBN: 978-0-201-89685-5.
10. Goldblatt R. Lectures on the hyperreals: An introduction to nonstandard analysis. New York : Springer-Verlag, 1998. Vol. 188 of Graduate texts in mathematics. ISBN: 0-387-98464-X.
11. Henle J.M., Kleinberg E.M. Infinitesimal calculus. Cambridge, Mass. : MIT Press, 1979. ISBN: 9780486151014.
12. Keisler H.J. The hyperreal line. *Real numbers, generalizations of the reals, and theories of continua* / ed. by Ehrlich P. Kluwer Academic Publishers, 1994. P. 207–237.
13. Keisler H.J. Elementary calculus: An infinitesimal approach. Second ed. 2012. URL: https://people.math.wisc.edu/~keisler/calc.html (online; accessed: 2022-09-10).

Submitted 18.09.2023