Правительство Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования «Санкт-Петербургский государственный университет»
Математическое обеспечение и администрирование информационных систем

Кафедра информационно-аналитических систем

Новиков Георгий Сергеевич

Разработка алгоритма выпуклой оптимизации для минимизации всплесков в динамической системе управления

Бакалаврская работа

Научный руководитель:
д. ф.-м. н., профессор Граничин О. Н.

Рецензент:
к.ф.-м.н., доцент Михайлова Е.В.

Санкт-Петербург
2016

SAINT-PETERSBURG STATE UNIVERSITY

Sub-Department of Analytical Information Systems

Novikov Georgii

Development of convex optimisation algorithm for deviation minimisation in dynamical control system

Graduation Thesis

Scientific supervisor:
Prof. Oleg Granichin

Reviewer:
associate professor, PhD Elena Mikhailova

Saint-Petersburg
2016

Оглавление

1. Введение

1.1. Актуальность

Теория управления - молодая наука, находящаяся в процессе интенсивного развития. При этом существенно меняются взгляды на предмет и основные проблемы данной дисциплины, равно как и используемый математический аппарат. В XIX веке было введено понятие устойчивости регулируемого процесса и получены первые критерии устойчивости. В 30-е годы в основном рассматривались частотные методы и, соответственно, частотные критерии устойчивости. В 50-е годы произошло обновление в теории управления - появился новый аппарат описания систем - в пространстве состояний. Иначе говоря, движение системы подчиняется обыкновенному дифференциальному уравнению, в правой части которого стоит функция, которая может выбираться проектировщиком (управление). Появились различные требования к функции управления - возможность стабилизации системы при внешних возмущениях [1], робастная стабилизация и т.д., а также различные критерии оптимальности функции управления. Один из таких критериев - скорость сходимости системы к своей точке равновесия. Этот критерий не учитывает поведение решения на ранних этапах сходимости, а смотрит лишь на асимптотическое поведение уравнения. Оказывается, это не всегда допустимо - при определенных условиях, на начальных этапах решение может сильно отклоняться от точки сходимости, причем, как правило, чем больше скорость сходимости - тем больше отклонения. Минимизации таких отклонений и посвящена эта статья.

Основная техника исследования задач в этой статье связана с линейными матричными неравенствами (С. Бойд с соавторами [5], Д. В. Баландин и М. М. Коган [2]) и задачей полуопределенного программирования [11]. Для решения полученных задач разработаны мощные оптимизационные процедуры, основанные на методе внутренней точки [3], [6], [7]. В качестве вычислительного средства были использованы две платформы - система Matlab вместе с программным пакетом SeDuMi

[8] и пакетом cvx [9], [10]. А также система ipython notebook для языка python [13] с использованием программных пакетов numpy [14] и picos [15].

1.2. Предварительный пример

Рассмотрим систему управления

$$\dot{x} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -100 & -20000.02 & -1000040.0001 & -2000.02 \end{pmatrix} x \qquad (1)$$

Собственные числа матрицы этой системы равны $\lambda_1 = \lambda_2 = -1000$ и $\lambda_3 = \lambda_4 = -0.01$. Норма Евклида решения с начальным положением $(0, 0, 1, 0)$ показана на рисунке 1, и с начальным положением $(0, 1, 0, 0)$ на рисунке 2. Обратите внимание, что сильное отклонение происходит в разные моменты времени t. И величина отклонения, и момент времени, в который оно происходит, зависят от начального положения системы.

1.3. Структура работы

В этой работе в разделе (2) сформулированы ключевые понятия и теоремы. Затем в разделе (3) рассмотрен случай нулевого собственного числа матрицы и способы работы с такими матрицами. Далее в разделе (4) построен алгоритм сходимости в требуемую конфигурацию для агентов, закон движения которых описывается формулой $\dot{x} = v$. В разделе (5) описано программное обеспечение, разработанное в ходе выполнения работы. В разделе (6) находятся результаты моделирования для задачи о перемещении группы мобильных агентов в заданное расположение.

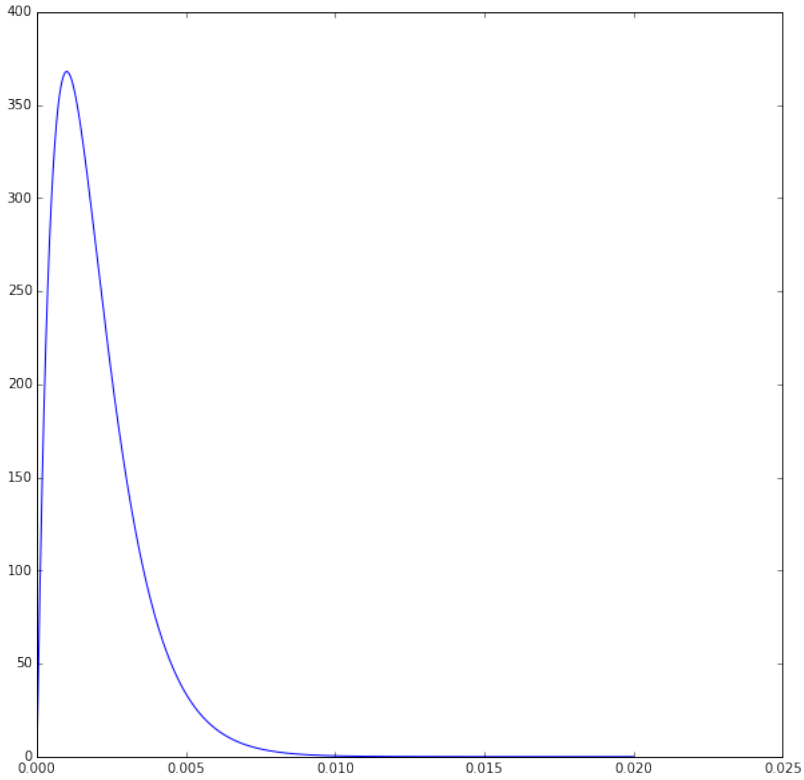

Рис. 1: Система (1), $x(0) = (0, 0, 1, 0)$

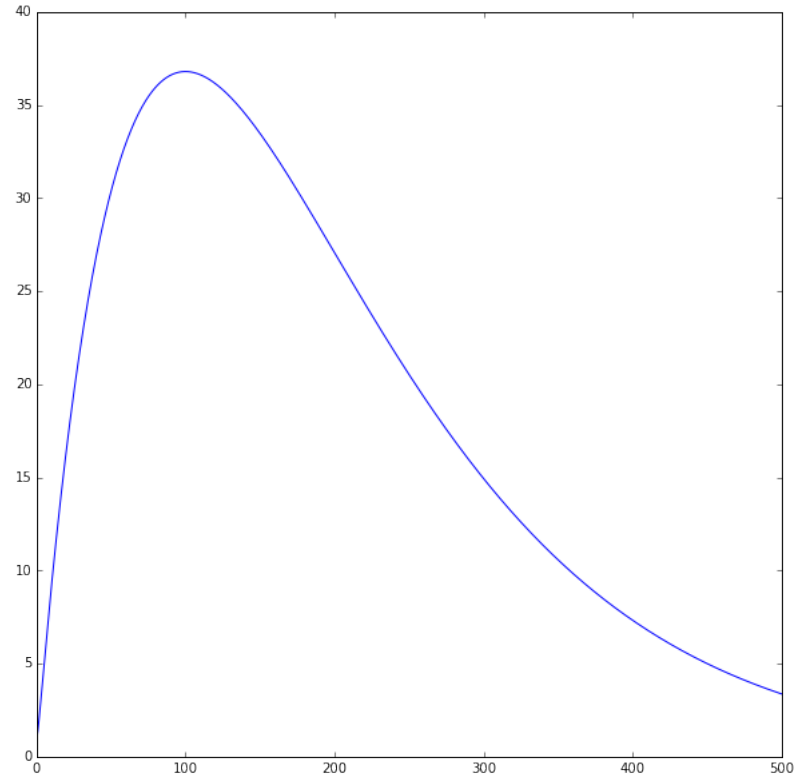

Рис. 2: Система (1), $x(0) = (0, 1, 0, 0)$

2. Задача о минимизации всплесков в системе управления

Рассмотрим линейную непрерывную стационарную систему управления с одномерным управлением

$$\dot{x} = Ax + bu, A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n, u \in \mathbb{R} \tag{2}$$

В случае, если вектора $A, Ab, \ldots, A^{n-1}b$ линейно независимы, то система (2) является управляемой и по теореме о размещении полюсов существует линейная обратная связь, задаваемая вектором K такая, что точка x = 0 является асимптотически устойчивой в уравнении

$$\dot{x} = Ax + bK^T x. \tag{3}$$

Более того, выбором вектора K можно получить систему с любым набором собственных чисел $\mathbb{L} \in \mathbb{C}$, тем самым получив любую наперед заданную скорость сходимости. Назовем величиной отклонения решения системы (3) от точки равновесия величину

$$\sup_{0\le t\le\infty} \sup_{|x_0|=1} |x(t, x_0)|. \tag{4}$$

Системы с большой скоростью сходимости, как правило, сильно отклоняются от точки сходимости на начальном этапе, что было показано Р. Измаиловым в [12]. А именно, им была доказана следующая теорема:

Theorem 1 $\exists\gamma = \gamma(A, b) > 0$ *такая, что если* $\{\lambda_1, \ldots, \lambda_n\}$ - *собственные значения системы (3) такие, что* $Re\lambda_i \le -\sigma < 0, i = \overline{1, n}$, *то верно неравенство*

$$\sup_{0\le t\le 1/\sigma, |x_0|=1} |x(t, x_0)| \ge \lambda\gamma^{n-1}. \tag{5}$$

Встает вопрос нахождения вектора K не просто с условием получения требуемой скорости сходимости (требуемых действительных частей собственных чисел матрицы), но также и минимизации величины возможных отклонений от точки сходимости. Данная задача исследована

в статье Б. Т. Поляка и Г. В. Смирнова [4]. В ней дан метод получения хорошего кандидата на вектор обратной связи. Его можно получить при помощи LMI подхода, решив задачу полуопределенного программирования.

Рассмотрим задачу минимизации

$$\min \alpha, \tag{6}$$

$$P(A + bK^T)^T + (A + bK^T)P \preceq -2\sigma P. \tag{7}$$

$$I \preceq P \preceq \alpha I. \tag{8}$$

Здесь $\alpha \geq 1, K \in \mathbb{R}^n, P = P^T \in \mathbb{R}^{n \times n}$ - переменные оптимизации. Неравенство Ляпунова (7) гарантирует, что действительные части всех собственных чисел системы (скорость сходимости всей системы) не превосходит σ и одновременно то, что эллипсоид $E = \{x \in \mathbb{R}^n : \langle x, P^{-1}x \leq 1\rangle\}$ является инвариантным множеством данной системы. Условие (8) гарантирует нам минимальность эксцентриситета эллипсоида E (а так как E инвариантное множество, то и минимальность величины отклонений, оцененных в таком виде). Заметим, что в таком виде задача минимизации не линейна. Это можно преодолеть произведя замену переменных $y = PK$. Тогда неравенство (7) приобретет вид

$$AP + PA^T + by^T + yb^T \leq -2\sigma P \tag{9}$$

В таком виде это линейная задача полуопределенного программирования, а ее решение является хорошим кандидатом в вектор обратной связи. Более того, в [4] показано, что это решение асимптотически (при больших σ) дает величину отклонений того же порядка, что и нижняя оценка Р. Измаилова (5).

3. Случай нулевого собственного числа

3.1. Поведение решения

Рассмотрим матрицу A с простым собственным числом, равным нулю, и соответствующим собственным вектором x^*. Для начала посмотрим, как будет вести себя решение с начальными данными x_0. Далее будем рассматривать разложение пространства в прямую сумму двух инвариантных подпространств матрицы A - собственного подпространства, соответствующего с.ч. 0 матрицы A и инвариантного подпространства, включающего в себя все остальные присоединенные вектора матрицы A. Назовем их нулевое и ненулевое инвариантные подпространства соответственно. Пусть x_0 раскладывается в сумму $x_0 = \alpha x^* + \beta x^\perp$, где $x^\perp$ принадлежит ненулевому инвариантному подпространству. Тогда

$$x(t, x_0) = e^{At}x_0 = \alpha e^{At}x^* + \beta e^{At}x^\perp \tag{10}$$

$$e^{At}x^* = \sum_{i=0}^{\infty} \frac{t^i A^i}{i!} x^* = x^*$$

то

$$x(t, x_0) = \alpha x^* + \beta e^{At}x^\perp \tag{11}$$

Рассмотрим, как действует матричная экспонента на вектор из ненулевого инвариантного подпространства. Для этого рассмотрим разложение матрицы A в жорданов базис $A = S^{-1}\begin{pmatrix} 0 & 0 \\ 0 & A_J \end{pmatrix} S$. Матрица S действует на $x^\perp$ следующим образом: $Sx^\perp = \begin{pmatrix} 0 \\ x \end{pmatrix}$ из чего следует:

$$e^{At}x^\perp = e^{S^{-1}\begin{pmatrix} 0 & 0 \\ 0 & A_J \end{pmatrix}S} x^\perp = S^{-1}\begin{pmatrix} 1 & 0 \\ 0 & e^{A_J t} \end{pmatrix} Sx^\perp = S^{-1}\begin{pmatrix} 0 \\ e^{A_j t}x \end{pmatrix} \tag{12}$$

Из этого следует, что решение из начальной точки x_0 не меняет своей составляющей x^*, а изменяет только свою составляющую в ненулевом инвариантном подпространстве, причем по закону $\dot{x} = A_j x$. Следующей задачей будет научиться работать со сходимостью в инвариантном подпространстве.

3.2. Постановка задачи

Дана система $\dot{x} = Ax + bK^T x$. Матрица A имеет простое собственное число, равное нулю, соответствующее собственному вектору x^*. Необходимо найти такой вектор K, что матрица $(A+bK^T)$ не поменяет нулевое и ненулевое инвариантные подпространства, собственные числа ненулевого инвариантного подпространства будут иметь вещественные части, меньшие $-\sigma$ для данного σ и величина отклонений системы от точки сходимости будет минимальной.

3.3. LMI для инвариантного подпространства

Мы хотим решить LMI задачу для матрицы A_j, получив после этого каким то образом вектор K. Наложим условия на вектора b, K, чтобы они не испортили нулевое собственное число и соответствующие два инвариантных подпространства:

$$A + bK^T = S^{-1}JS + bK^T = S^{-1}(\begin{pmatrix} 0 & 0 \\ 0 & A_J \end{pmatrix} + SbK^T S^{-1})S \tag{13}$$

Из этого следует, что:

$$\sum_{i=1}^{n} (S^{-1}b)_j K_i S_i^1 = 0, j = \overline{1,n} \tag{14}$$

$$\sum_{i=1}^{n} (S^{-1}b)_1 K_i S_i^j = 0, j = \overline{1,n} \tag{15}$$

Из (14) следует, что $K^T S_i^1 = 0$ (иначе $S^{-1}b \equiv 0 \Leftrightarrow b \equiv 0$), из (15) следует, что $(S^{-1}b)_1 = 0$ (иначе $K^T S \equiv 0 \Leftrightarrow K \equiv 0$). Вектор b является

входным параметром, поэтому условие на него должно быть соблюдено изначально. Пусть LMI метод как результат выдал нам вектор $\widehat{K}$ (вспомним, что он размера $n-1$, так как матрица $A_j \in \mathbb{R}^{n-1\times n-1}$, а также то, что он записан для нестандартного базиса матрицы A). Посмотрим на то, как должен выглядеть этот вектор в стандартном базисе с учетом условия (14):

$$\begin{cases} K^T S^1 = 0 \\ K^T S^2 = \widehat{K}_1 \\ \dots \\ K^T S^n = \widehat{K}_{n-1} \end{cases} \Leftrightarrow S^T K = \begin{pmatrix} 0 \\ \widehat{K} \end{pmatrix} \Leftrightarrow K = (S^T)^{-1} \begin{pmatrix} 0 \\ \widehat{K} \end{pmatrix} \tag{16}$$

В итоге получается, что нужно разложить матрицу A в базис, выделив нулевое собственное подпространство (например, подойдет жорданов базис). Подать на вход задаче LMI матрицу A_j, вектор $S^{-1}b$ без первого элемента (который по условию (15) должен быть равен нулю) и необходимую скорость сходимости σ, и по результирующему вектору $\widehat{K}$ получить вектор K по формуле $K = (S^T)^{-1} \begin{pmatrix} 0 \\ \widehat{K} \end{pmatrix}$.

4. Движение группы мобильных агентов

4.1. Задача о переводе в заданное расположение

Рассмотрим систему мобильных агентов, движение которых задается уравнениями

$$\begin{cases} \dot{x} = v \\ \dot{v} = q \end{cases} \tag{17}$$

$x \in \mathbb{R}^n$ - координаты системы, $v \in \mathbb{R}^n$ - скорости, $q \in \mathbb{R}^n$ - ускорения, которые являются нашим воздействием на эту систему. Я буду рассматривать случай, когда q линейно зависит от x и v. Процесс будет описываться следующей системой:

$$\begin{pmatrix} \dot{x} \\ v \end{pmatrix} = \begin{pmatrix} 0 & I \\ A_v & B_v \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} \tag{18}$$

Алгоритм состоит в следующем: наложим условия на A_v и B_v таким образом, что для наперед заданного вектора x^* вектор $\begin{pmatrix} x^* \\ 0 \end{pmatrix}$ является собственным вектором системы (18), соответствующим собственному числу ноль. x^* - это некоторая конфигурация, в которую система должна сойтись. Но в соответствии с (11) для начального положения $\begin{pmatrix} x_0 \\ v_0 \end{pmatrix}$ точкой сходимости является ее составляющая в нулевом инвариантном подпространстве, а не само $\begin{pmatrix} x^* \\ 0 \end{pmatrix}$. Поборем это следующим способом: добавим к системе новую размерность - управляющий параметр $u \in \mathbb{R}$, изменение которого также линейно зависит от полного состояния. Тогда система примет следующий вид:

$$\begin{pmatrix} \dot{x} \\ v \\ u \end{pmatrix} = \begin{pmatrix} 0 & I & 0 \\ A_v & B_v & c_v \\ a_u & b_u & c_u \end{pmatrix} \begin{pmatrix} x \\ v \\ u \end{pmatrix}, \tag{19}$$

где $a_u \in \mathbb{R}^n, b_u \in \mathbb{R}^n, c_u \in \mathbb{R}$. Использовать управляющий параметр мы

будем следующим образом: во-первых, вместо вектора $\begin{pmatrix} x^* \\ 0 \end{pmatrix}$ собственным вектором системы будет вектор $\begin{pmatrix} x^* \\ 0 \\ 1 \end{pmatrix}$. Во-вторых, по начальному положению $\begin{pmatrix} x_0 \\ v_0 \end{pmatrix}$ можно вычислить (см. (4.2)) такой управляющий параметр u_0, что $\begin{pmatrix} x_0 \\ v_0 \\ u_0 \end{pmatrix} = \begin{pmatrix} x^* \\ 0 \\ 1 \end{pmatrix} + \widehat{y}$, где $\widehat{y}$ принадлежит ненулевому инвариантному подпространству, а то-есть система асимптотически сойдется в состояние $\begin{pmatrix} x^* \\ 0 \\ 1 \end{pmatrix}$

4.2. Теоретические основы

Сначала сформулируем условия на $A_v, B_v, c_v, a_u, b_u, c_u$, чтобы получить необходимый собственный вектор:

$$\begin{pmatrix} 0 & I & 0 \\ A_v & B_v & c_v \\ a_u & b_u & c_u \end{pmatrix} \begin{pmatrix} x^* \\ 0 \\ 1 \end{pmatrix} = 0 \Leftrightarrow \begin{cases} A_v x^* + c_v = 0 \\ a_u x^* + c_v = 0 \end{cases} \tag{20}$$

Теперь по x_0 и v_0 найдем необходимое начальное значение управляющего параметра. Пусть матрица S - матрица перехода в базис, который разделяет два инвариантных подпространства (подпространство собственного числа 0 и остальных собственных чисел). Тогда

$$\begin{cases} S \begin{pmatrix} \alpha_1 \\ \dots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} x_0 \\ v_0 \\ u_0 \end{pmatrix} \\ \alpha_1 = 1 \end{cases} \Leftrightarrow S_1^{-1} \begin{pmatrix} x_0 \\ v_0 \\ u_0 \end{pmatrix} = 1 \tag{21}$$

4.3. Ход Алгоритма

В алгоритме можно выделить три части:

1. Построение матрицы
2. Подготовка начального положения
3. Процесс сходимости

Первый этап - построение матрицы с нужным собственным вектором и собственным числом и последующая оптимизация части матрицы, действующей в нужном инвариантном подпространстве, при помощи метода LMI. Это нужно сделать только один раз для конкретной конфигурации x^*. Второй этап - вычисление u_0 по заданным x_0 и v_0. Третий этап - процесс сходимости по закону (19). При построенной матрице на первом этапе, второй и третий этапы можно повторять сколько угодно раз, перестраивать матрицу каждый раз не нужно.

5. Программное обеспечение

В ходе работы было написано две версии вычислительной программы с идентичными интерфейсами - на языке MATLAB и на языке Python в среде разработки jupyter notebook. Всего можно выделить несколько базовых функций

1. Функция решения задачи полуопределенного программирования в форме (9). На вход она получает матрицу $A \in \mathbb{R}^{n\times n}$, вектор $b \in \mathbb{R}^n$ и число $\sigma \in \mathbb{R}_+$. Она приводит задачу к необходимому формату, после чего, используя внешний алгоритм минимизации, решает ее и возвращает искомый вектор $K \in \mathbb{R}^n$. Как расширение данная функция может принимать на вход вместо вектора b матрицу $B \in \mathbb{R}^{n\times m}$ и возвращать матрицу $K \in \mathbb{R}^{n\times k}$ решая тем самым более общую задачу поиска управления для системы $\dot{x} = Ax + BK^Tx$.

2. Функция, находящая базис, разделяющий нулевое и ненулевое инвариантные подпространства для данной матрицы A. В общем случае результатом является овеществленный жорданов базис матрицы A. В нашем же случае известно, что у матрицы A имеется простое собственное число 0, соответствующее собственному вектору x^*. Так что процедуру поиска можно сильно упростить. Заметим, что для любого $x = \alpha x^* + \beta\widehat{x}$, где $\widehat{x}$ принадлежит ненулевое инвариантное подпространство, $Ax = \beta A\widehat{x}$. Поэтому базис можно набрать из случайно сгенерированного набора векторов Ax.

3. Функция, моделирующая поведение системы мобильных агентов. На вход принимает матрицу $A \in \mathbb{R}^{2n*m+1\times 2n*m+1}$, собственный вектор $y \in \mathbb{R}^{2n*m+1}$, вектора $x_0 \in \mathbb{R}^{n*m}$ и $v_0 \in \mathbb{R}^{n*m}$ - начальное состояние системы и массив моментов времени t. n и m здесь - количество агентов и размерность пространства соответственно. Результатом является массив векторов состояний системы в данные моменты времени, полученный в соответствии с законом (19).

4. Функция, находящая по заданному набору собственных чисел для

системы (2) необходимый вектор K. Метод описан в статье Б. Поляка [4]

6. Пример использования

Рассмотрим задачу перевода системы мобильных агентов в заданное расположение и для заданной скорости сходимости σ сравним поведение двух систем - системы с одинаковыми собственными числами равными $-\sigma$ и системы, полученной из нее применением подхода LMI.

6.1. Случай одного мобильного агента

Рассмотрим случай одного мобильного агента на плоскости. Необходимое расположение задается вектором $x^* = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$. Матрица A сгенерирована случайно с точностью до удовлетворения условиям (20). Начальное состояние системы $x_0 = \begin{pmatrix} -5 \\ -5 \end{pmatrix}, v_0 = \begin{pmatrix} -7 \\ -3 \end{pmatrix}, \sigma = 0.1$. На рисунках 3 и 4 изображены траектории агента для системы без оптимизации и с оптимизацией соответственно. для. На рисунках 5 и 6 изображены графики зависимости расстояния состояния системы до точки сходимости от времени. Четко видно, что из-за недостаточной силы воздействия на агента в неоптимизированной системе, в процессе положение сильно отклоняется от точки сходимости. Тогда как в случае оптимизированной системы агент очень быстро с небольшими отклонениями сходится в нужную точку. Ненулевые собственные числа матрицы $A + bK^T$ примерно равны $-6.57, -0.36, -0.43 \pm 0.85i$.

6.2. Случай трех мобильных агентов

Более интересен случай трех и более мобильных агентов. Матрица системы (19) будет иметь размер хотя бы 12. В этом случае алгоритм построения матрицы с данным набором собственных чисел перестает работать - из-за возведения матрицы в большую степень погрешность вычислений превосходит необходимую точность. LMI подход имеет более хорошие показатели устойчивости, из-за чего может быть использован для построения матриц для более высокоразмерных задач. Рассмотрим

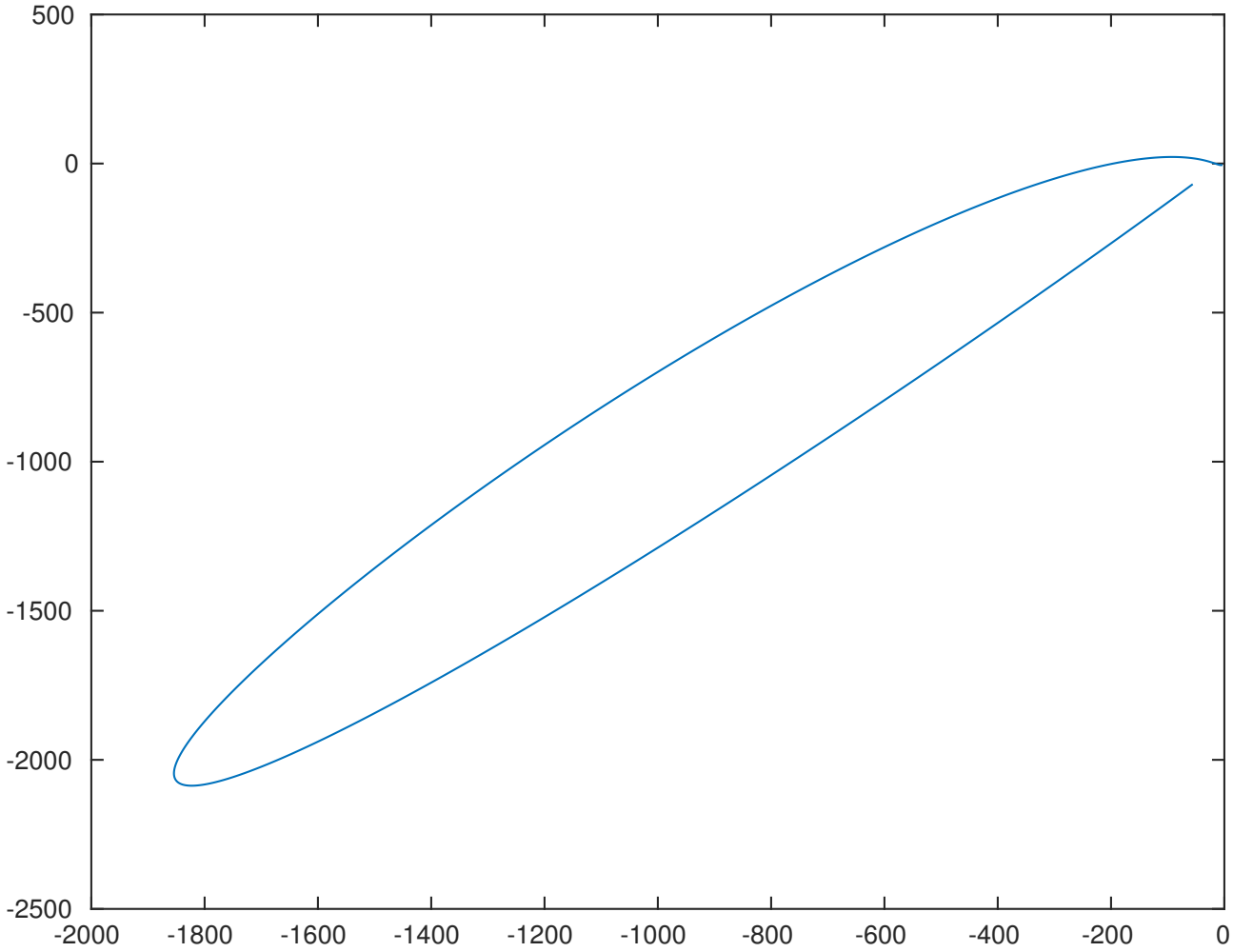

Рис. 3: Траектория движения агента в неоптимизированной системе

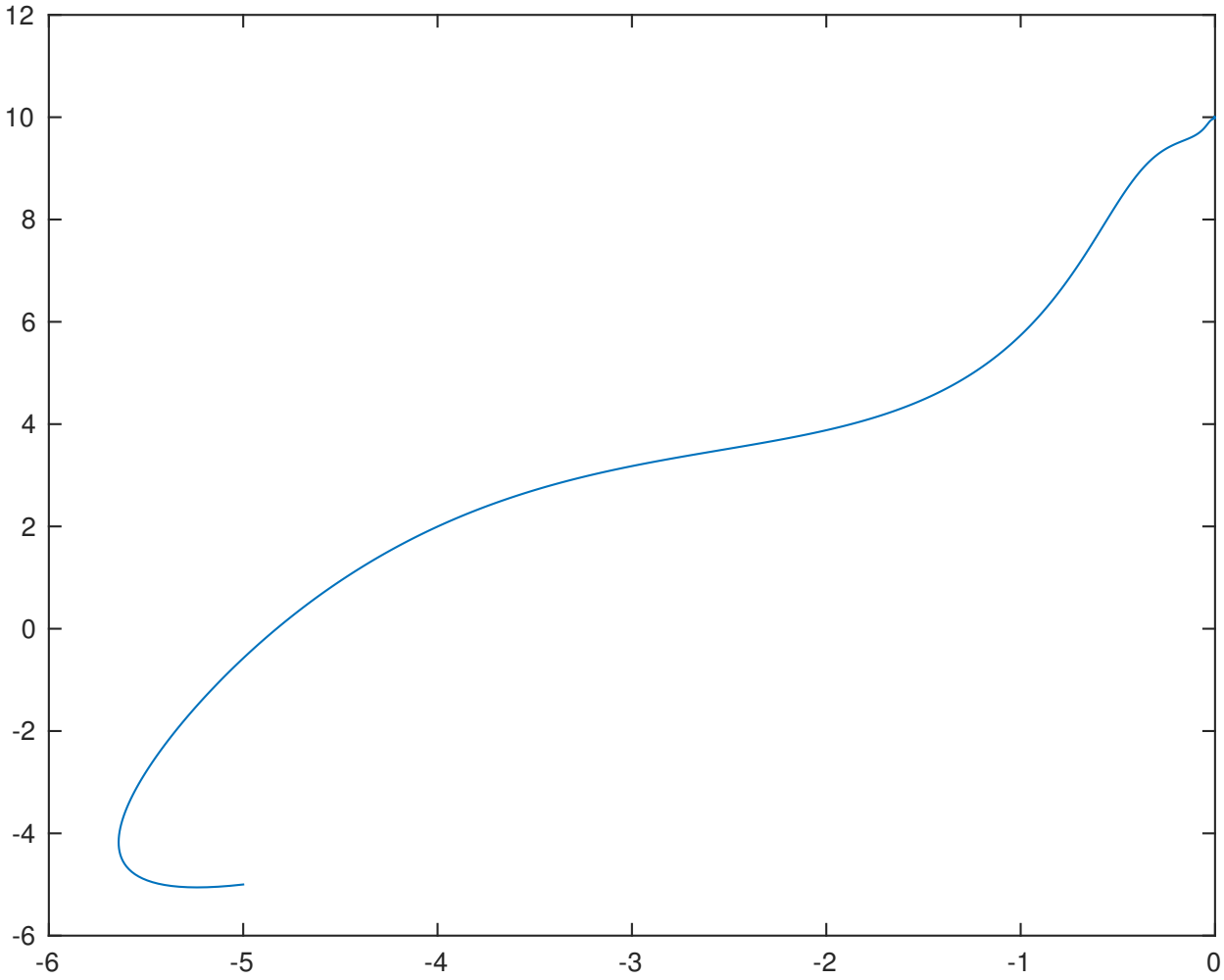

Рис. 4: Траектория движения агента в оптимизированной системе

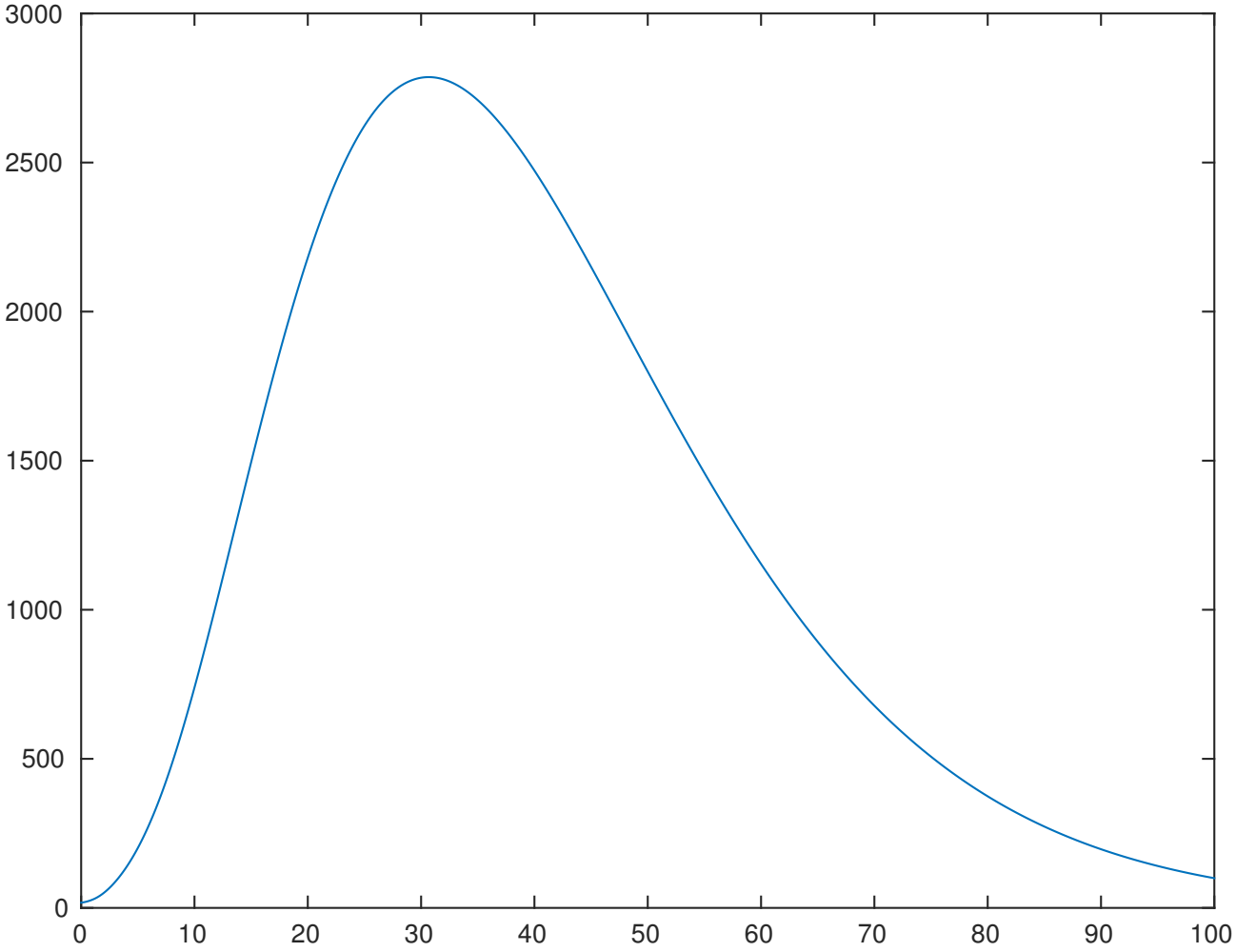

Рис. 5: Зависимость до точки сходимости от времени в неоптимизированной системе

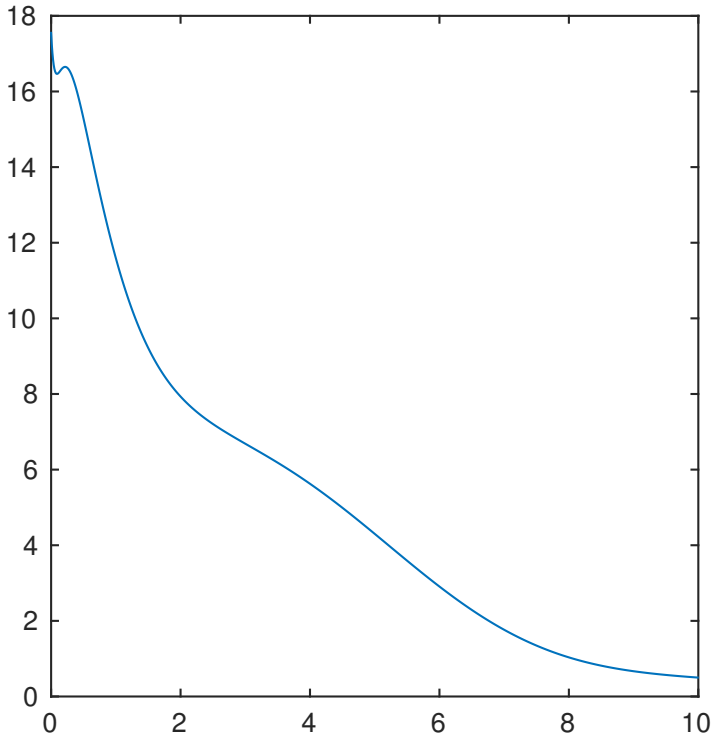

Рис. 6: Зависимость расстояния до точки сходимости от времени в оптимизированной системе

следующую систему: $x^* = \begin{pmatrix} 100 \\ 0 \\ 0 \\ 0 \\ -100 \\ 0 \end{pmatrix}, x_0 = \begin{pmatrix} -5 \\ -5 \\ 5 \\ -5 \\ 0 \\ 0 \end{pmatrix}, v_0 = \begin{pmatrix} -7 \\ -3 \\ 2 \\ 3 \\ 0 \\ 0 \end{pmatrix}, \sigma =$ 0.01. Это задача выстроить три мобильных агента в линию в точки с координатами (-100, 0), (0, 0) и (100, 0). На рисунке 7 изображены траектории агентов. По сравнению со случаем одного агента траектории сильно усложнились. На рисунке 8 изображена зависимость расстояния от состояния системы до точки сходимости от времени - отклонение не превышает 4 норм равновесного состояния.

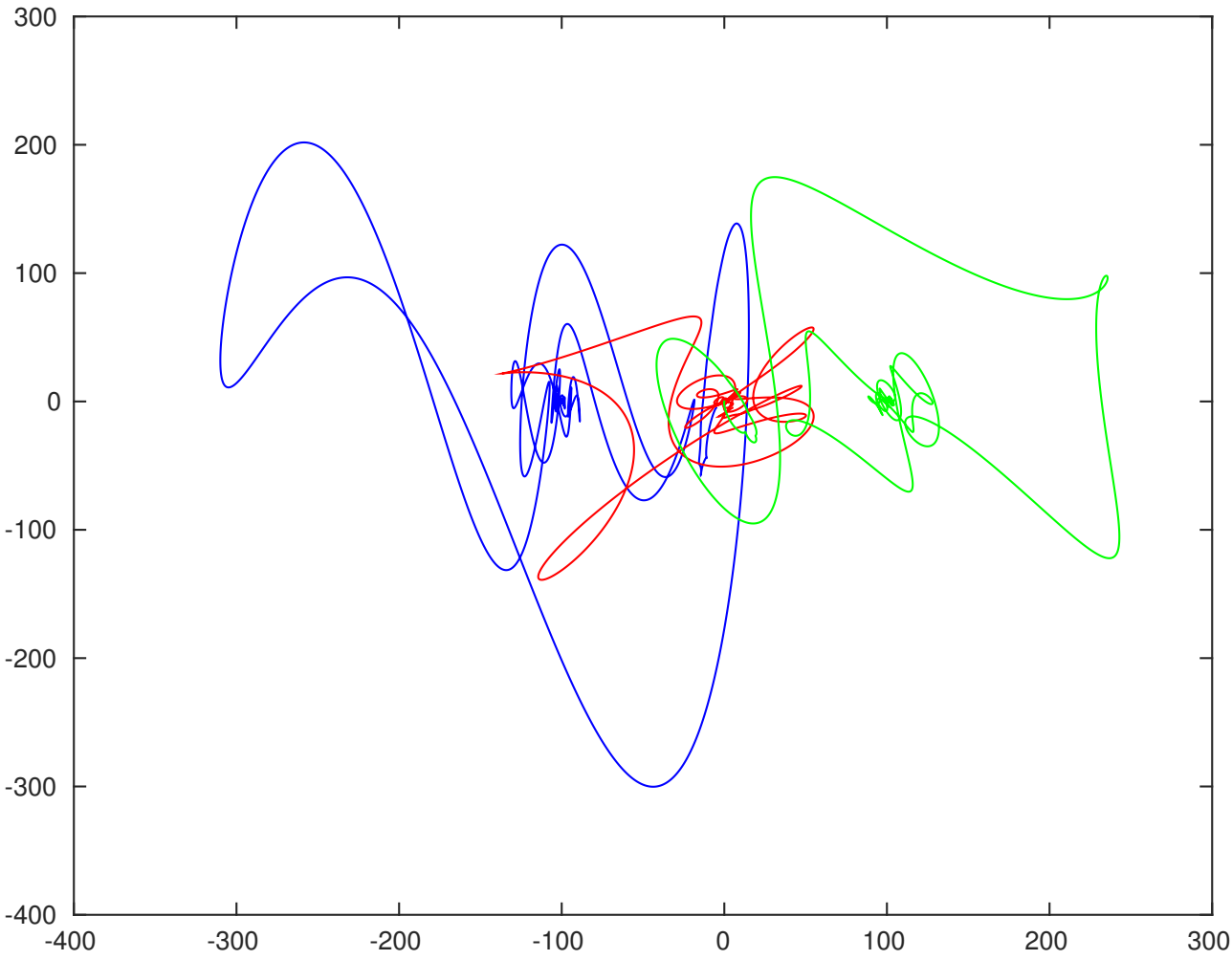

Рис. 7: Траектории агентов в системе из трех агентов

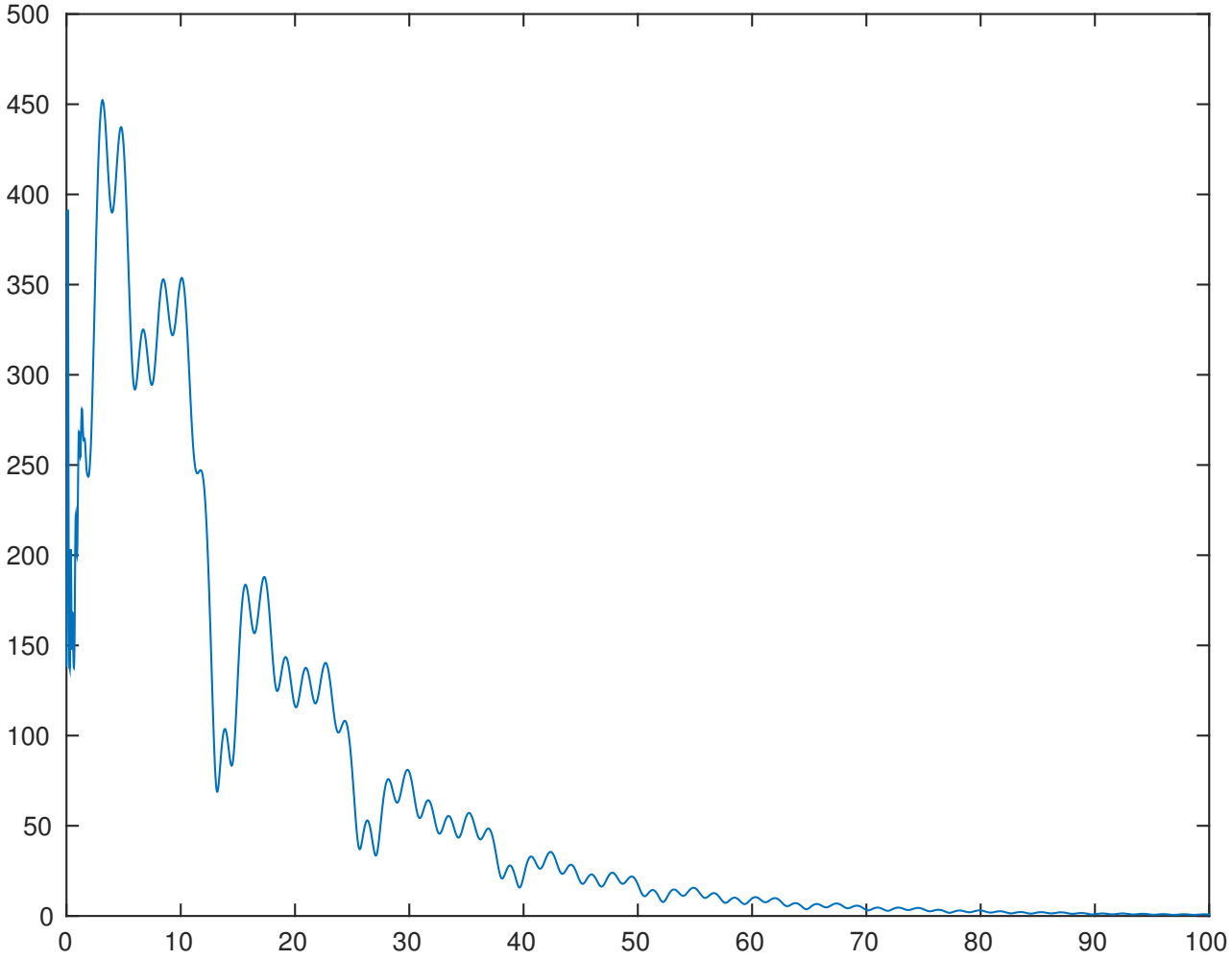

Рис. 8: Зависимость расстояния до точки сходимости от времени в системе из трех агентов

7. Заключение

В работе было проведено несколько исследований. Разработан способ применения метода поиска оптимального управления в смысле минимизации величины отклонения системы от точки равновесия к случаю матриц с простым нулевым собственным числом. Также разработана система, решающая задачу перемещения группы мобильных агентов в заданное расположение и применен к ней разработанный метод минимизации отклонений. Написал программа, производящая необходимые вычисления на языке MATLAB и языке python и произведен ряд экспериментов по сравнению движений агентов в соответствии с придуманной системой с и без применения метода минимизации отклонений. Получены улучшения для оптимизированных систем как по скорости сходимости, так и по величине всплесков.

Список литературы

[1] Б. Т. Поляк, М. В. Хлебников, and П. С. Щербаков. Управление линейными системами при внешних возмущениях: Техника линейных матричных неравенств. *М.: ЛЕНАНД*, 2014.

[2] Д. Баландин and М. Коган. *Синтез законов управления на основе линейных матричных неравенств*. ЛитРес, 2016.

[3] Нестеров Ю. Е. *Методы выпуклой оптимизации*. М.: МЦНМО, 2009.

[4] Boris T. Polyak and Georgi V. Smirnov. Large deviations in continuous-time linear single-input control systems. *{IFAC} Proceedings Volumes*, 47(3):5586 – 5591, 2014. 19th {IFAC} World Congress.

[5] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, June 1994.

[6] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[7] Yurii Nesterov, Arkadii Nemirovskii, and Yinyu Ye. *Interior-point polynomial algorithms in convex programming*, volume 13. SIAM, 1994.

[8] Jos F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones, 1998.

[9] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[10] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes

in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[11] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. MPS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, 2001.

[12] R. N. Izmailov. Peak effect in stationary linear-systems in scalar inputs and outputs. *Automation and Remote Control*, 48(8):1018–1024, 1987.

[13] Fernando Pérez and Brian E Granger. Ipython: a system for interactive scientific computing. *Computing in Science & Engineering*, 9(3):21–29, 2007.

[14] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[15] Guillaume Sagnol. Picos, a python interface to conic optimization solvers. Technical report, Technical Report 12-48, ZIB, 2012. http://picos. zib. de, 2012.