МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра методов оптимального управления

Селях Никита Евгеньевич

СИНТЕЗ ОПТИМАЛЬНОЙ СИСТЕМЫ НА ОСНОВЕ МЕТОДОВ ПАРАМЕТРИЧЕСКОГО ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Курсовой проект

Научный руководитель
канд. физ.-мат. наук
доцент Н.М. Дмитрук

Допущена к защите

«_____» ____________________ 2020 г.
Зав. кафедрой методов оптимального управления
канд. физ.-мат. наук, доцент Н.М. Дмитрук

Минск 2020

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Существуют два взгляда на теорию оптимального управления. Согласно одному из них, возникшему после открытия принципа максимума Понтрягина, теория оптимального управления — раздел современного (неклассического) вариационного исчисления. Другой взгляд трактует теорию оптимального управления как раздел современной теории управления, представляющей естественное развитие классической теории управления. В соответствии с этим в первом случае под управлениями понимаются элементы функциональных пространств, по которым ищется экстремум выбранного функционала качества, и основным вопросом теории считается анализ решения экстремальной задачи (существование, единственность, непрерывная зависимость решений, необходимые и достаточные условия оптимальности и т.п.).

Во втором случае управление — это процесс, в котором для достижения нужного поведения объекта управления в каждый текущий момент времени создаются целенаправленные (управляющие) воздействия на объект управления в зависимости от доступной к этому моменту информации о поведении объекта и действующих на него возмущениях. Управление называется программным, если (программные) управляющие воздействия (программы) планируются по априорной информации до начала процесса управления и не корректируются в процессе управления. При позиционном управлении (позиционные) управляющие воздействия создаются в процессе управления по текущим позициям, которые аккумулируют информацию, доступную к текущему моменту. Построение оптимальных позиционных управляющих воздействий называется синтезом оптимальных систем управления и является основным вопросом теории оптимального управления во втором случае.

В данной работе будет рассмотрено применение методов параметрического программирования для задачи синтеза оптимальной системы второго порядка.

ГЛАВА 1
ОБЗОР ЛИТЕРАТУРЫ

В настоящей главе приводится обзор основных результатов теории оптимального управления. Сначала обсуждаются формулировки задач, возникающих при оптимизации динамических систем управления. Затем приводятся основные результаты качественной теории — принцип максимума Л.С. Понтрягина и динамическое программирование Р. Беллмана.

В данной главе рассматриваются основные результаты теории оптимального управления, классический подход к посмтроению обратных связей, а также оптимальное управление в реальном времени и реализация оптимальной обратной связи

Кроме того, обсуждаются современные методы управления в реальном времени: реализация оптимальных обратных связей в задачах оптимального управления с конечным горизонтом и теория управления по прогнозирующей модели для решения классической проблемы теории управления — стабилизации нелинейных динамических систем при наличии ограничений.

1.1 Основные результаты теории оптимального управления

Постановка любой конкретной задачи оптимального управления включает в себя 5 необходимых элементов: промежуток управления, математическую модель управляемого объекта, класс управлений и ограничений на них, ограничения на фазовую траекторию, критерий качества. Рассмотрим их подробнее.

1) Промежуток управления. Прежде всего задачи оптимального управления разделяются на непрерывные, рассматриваемые на некотором промежутке времени $T = [t_0, t_f]$, и дискретные, в которых динамический процесс рассматривается в дискретные моменты времени $k = 0, 1, ...N$, где N — натуральное число.

По продолжительности процесса различаются задачи с фиксированным и нефиксированным временем окончания процесса. Выделяются также задачи на бесконечном интервале.

2) Математическая модель. Динамика изучаемого процесса моделирует-

ся, как правило, дифференциальными (для непрерывных систем)

$$\dot{x}(t) = f(x(t), u(t), t), t \in [t_0, t_f], \tag{1.1}$$

или разностными уравнениями (для дискретных систем)

$$x(k+1) = f(x(k), u(k), k), k = 0, 1, ...,$$

где n-вектор x называется состоянием системы, r-вектор u называется управлением, функция $f : R \times \ R^r \times R \to R^n$ задана.

Число переменных состояния n называется порядком системы управления, число r — числом входов.

Далее будем рассматривать непрерывные системы вида (1.1).

3) Класс управлений и ограничения на них. Для непрерывного процесса управления указывается класс функций, из которого выбираются управления. Это могут быть: измеримые, дискретные, кусочно-непрерывные, гладкие, импульсные функции и т.д.

Кроме класса доступных управлений задается множество $U \subset \ R$ — множество допустимых значений управления. Как правило, U — компакт в R^r.

Далее будем рассматривать управления из класса кусочно-непрерывных функций.

Определение 1.1 Кусочно-непрерывная функция $u(\cdot) = (u(t), t \in [t_0, t_f])$ называется доступным управлением, если $u(t) \in U, t \in [t_0, t_f]$.

4) Ограничения на фазовую траекторию. Ограничения на переменные состояния могут накладываться:

- в начальный момент времени t_0:

$$x(t_0) \in X_0;$$

- в конечный момент времени t_f — такие ограничения называются терминальными:

$$x(t_f) \in X_f;$$

- в изолированные моменты $t_i \in [t_0, t_f], i = \overline{1, m}$, из промежутка управления — промежуточные фазовые ограничения:

$$X(t_i) \in X_i, i = \overline{1, m},$$

- на всем промежутке управления — фазовые ограничения:

$$x(t) \in X(t), t \in [t_0, t_f],$$

где $X_0, X_f, X_i, i = \overline{1, m}, X(t), t \in [t_0, t_f],$ — заданные подмножества пространства состояний.

Задача управления с $x(t_f) \in X_f$ называется:

- задачей со свободным правым концом траектории, если $X_f = R^n$,
- задачей с закрепленным правым концом траектории, если $X_f = \{x_f\}$,
- задачей с подвижным правым концом траектории, если X_f содержит более одной точки и не совпадает с R^n.

Аналогичная классификация имеет место для задач с ограничениями на левый конец траектории $x_0 \in X_0$.

Выделяют также смешанные ограничения, учитывающие связи между переменными состояния и переменными управления:

$$(u(t), x(t)) \in S \subseteq R^r \times R^n, t \in [t_0, t_f[.$$

Определение 1.2 Доступное управление $u(\cdot)$ называется допустимым (или, программой), если оно порождает траекторию $x(\cdot) = (x(t), t \in [t_0, t_f],$ удовлетворяющую всем заданным ограничениям задачи.

5) Критерий качества. Качество допустимого управления оценивается так называемым критерием качества

Существуют четыре типа критерия качества:

i) критерий качества Майера (терминальный критерий)

$$J(u) = \varphi(x(t_f)),$$

ii) критерий качества Лагранжа (интегральный критерий)

$$J(u) = \int_{t_0}^{t_f} f_0(x(t), u(t), t)dt,$$

iii) критерий качества Больца

$$J(u) = \varphi(x(t_f)) + \int_{t_0}^{t_f} f_0(x(t), u(t), t)dt,$$

iv) задачи быстродействия (являются задачами с нефиксированной продолжительностью процесса).

$$J(u) = t_f - t_0 \to \min .$$

Определение 1.3 Допустимое управление $u^0(\cdot)$ называется оптимальным управлением (оптимальной программой), если на нем критерий качества достигает экстремального значения (min или max):

$$J(u^0) = extr J(u),$$

где минимум (максимум) берется по всем допустимым управлениям.

1.2 Принцип максимума и динамическое программирование

В теории оптимального управления существует два фундаментальных результата: принцип максимума Л.С. Понтрягина [2] и динамическое программирование Р. Беллмана. [1] Приведем эти результаты на примере простейшей задачи оптимального управления.

$$\begin{gathered} J(u) = \varphi(x(t_f)) + \int_{t_0}^{t_f} f_0(x(t), u(t), t)dt \to \min, \\ \dot{x}(t) = f(x(t), u(t), t), x(t_0) = x_0, \\ u(t) \in U, t \in [t_0, t_f]. \end{gathered} \tag{1.2}$$

1.2.1 Принцип максимума Понтрягина

Принципом максимума называется основное необходимое условие оптимальности в задачах оптимального управления, связанное с максимизацией гамильтониана:

$$H(x, \psi, u, t) = \psi' f(x, u, t) - f_0(x, u, t) = \sum_{j=1}^{n} \psi_j f_j(x, u, t) - f_0(x, u, t).$$

Здесь $\psi = \psi(t) \in R^n$ — сопряженная переменная.

Теорема 1.1 Пусть $u^0(\cdot), x^0(\cdot)$ — оптимальное управление и траектория в задаче (1.2), $\psi^0(\cdot)$ — соответствующее решение сопряженной системы

$$\dot{\psi}^0(t) = -\frac{\partial H}{\partial x}(x^0(t), \psi^0(t), u^0(t), t),$$

с начальным условием

$$\psi^0(t_f) = -\frac{\partial \varphi}{\partial x}(x^0(t_f)).$$

Тогда для любого $t \in [t_0, t_f]$, управление $u^0(t)$ удовлетворяет условию:

$$H(x^0(t), \psi^0(t), u^0(t), t) = \max_{v \in U} H(x^0(t), \psi^0(t), v, t), t \in [t_0, t_f].$$

Для того чтобы решить задачу с помощью принципа максимума обычно поступают следующим образом. Функцию $H(x, \psi, u, t)$ рассматривают как функцию r переменных $u = (u_1, ..., u_r)$. Далее проводят поточечную оптимизацию для каждого фиксированного набора (x, ψ, t)

$$u(x, \psi, t) = \arg\max_{v \in U} H(x, \psi, v, t). \tag{1.3}$$

Если исходная задача (1.2) имеет решение, функция (1.3) определена на непустом множестве значений (x, ψ, t).

Пусть u в виде (1.3) найдена, тогда можно рассмотреть следующую систему с граничными условиями:

$$\dot{x} = \frac{\partial H}{\partial \psi}(x, \psi, u(x, \psi, t), t) = f(x, \psi, u(x, \psi, t), t), \ x(t_0) = x_0,$$

$$\dot{\psi} = -\frac{\partial H}{\partial x}(x, \psi, u(x, \psi, t), t), \ \psi(t_f) = -\frac{\partial \varphi(x(t_f))}{\partial x}.$$

Таким образом получена специальная краевая задача, которая называется краевой задачей принципа максимума.

Можно ожидать, что имеются лишь отдельные изолированные пары функций $x(\cdot), \psi(\cdot)$, удовлетворяющие краевой задаче принципа максимума. Подставив одну такую пару в (1.3), получим:

$$u(t) = u(x(t), \psi(t), t), t \in [t_0, t_f], \tag{1.4}$$

которая удовлетворяет принципу максимума и, значит, может претендовать на роль оптимального управления, а функция $x(t) = x(t \mid t_0, x_0, u(\cdot)), t \in [t_0, t_f]$, — на роль оптимальной траектории в задаче.

Отметим, что принцип максимума в задаче (1.2) является лишь необходимым условием оптимальности, поэтому построенное управление не может быть оптимальным. Построенная функция называется экстремалью Понтрягина (1.4).

1.2.2 Динамическое программирование

Рассмотрим задачу (1.2) и предположим, что она имеет решение. Следуя динамическому программированию [1], погрузим задачу (1.2) в семейство задач

$$
\begin{aligned}
J_{\tau,z}(u) = \varphi(x(t_f)) + \int_{\tau}^{t_f} f_0(x(t), u(t), t)dt \to \min,\\
\dot{x}(t) = f(x,u),\ x(\tau) = z,\\
u(t) \in U,\ t \in T = [\tau, t_f],
\end{aligned}
\tag{1.5}
$$

зависящих от скаляра $\tau \in T$ и n-вектора z.

Пару (τ, z) назовем позицией в задаче (1.2). Обозначим через

$$
B(\tau, z) = \min J_{\tau,z}(u)
$$

минимальное значение критерия качества в задаче (1.5) для позиции (τ, z). Если для позиции (τ, z) задача (1.5) не имеет решения, положим $B(\tau, z) = +\infty$. Пусть

$$
X_\tau = \{z \in R : B(\tau, z) < +\infty\}.
$$

Функцию

$$
B(\tau, z), z \in X_\tau, \tau \in T, \tag{1.6}
$$

называют функцией Беллмана.

Уравнение в частных производных, которому удовлетворяет функция (1.6), называют уравнением Беллмана

$$
-\frac{\partial B(\tau, z)}{\partial \tau} = \min_{v \in U} \left\{ \frac{\partial B'(\tau, z)}{\partial z} f(z, v) + f_0(z, v) \right\}, z \in X_\tau, \tau \in T. \tag{1.7}
$$

Выделяя из семейства (1.5) задачу с $\tau = t_f$, находим граничное условие для уравнения Беллмана

$$
B(t_f, z) = \begin{cases} \varphi(z), z \in X, \\ +\infty, z \notin X. \end{cases} \tag{1.8}
$$

Схема применения динамического программирования состоит в следующем: для задач составляется уравнение Беллмана. По решению уравнения строиться позиционное решение.

1.3 Классический подход к построению оптимальных обратных связей

Рассматривается задача оптимального управления динамическими системами в условиях неопределенности. В отличие от классического подхода, в котором оптимальные обратные связи строятся по детерминированным моделям, в работе используются математические модели с множественной неопределенностью. Допустимые и оптимальные управления гарантируют определенный результат

$$J(u) = \varphi(x(t_f)) \to \min, \tag{1.9}$$

$$\dot{x} = f(x, u, t),\ x(t_0) = x_0 \tag{1.10}$$

$$x(t) \in X \in \mathbb{R}^n \tag{1.11}$$

$$u(t) \in U \in \mathbb{R}^r,\ t \in [t_0, t_f] \tag{1.12}$$

В поставленной задаче нужно минимаизировать критерий качества (1.9) на траекториях системы (1.10), которые в каждый момент времени лежат в заданном множестве X (1.11) с помощью ограниченных управляющих воздействий (1.12). (1.9) - (1.12): t_0, t_f — заданы,
$x = x(t) \in \mathbb{R}^n$ — состояния системы управления в t,
$u = u(t) \in \mathbb{R}^r$ — значения управляющего воздействия в t.
$f : \mathbb{R}^n \times \mathbb{R}^r \times \mathbb{R}^n$ обеспечивает существование и продолжимость решения уравнения (1.10) на промежутке времени $T = [t_0; t_f]$.

Задачу (1.9) - (1.12) будем рассматривать в классе кусочно-непрерывных управляющих воздействий u. Допустимое программное управление (программа) — кусочно-непрерывная функция $u(\cdot) \in U$, если она порождает траекторию x системы (1.10), удовлетворяющей (1.11). Допустимое программное управление - оптимальное (оптимальная программа), если на нем критерий качества (1.9) достигает оптимального значения. $J(u^0) = minJ(u)$, где минимум берется по всем программам.
Для введения понятия классической оптимальной обратной связи погрузим задачу (1.9) - (1.12) в следующее семейство задач:

$$\varphi(x(t_f)) \to \min, \dot{x} = f(x,u,t),\ x(z) = zx(t) \in Xu(t) \in U,\ t \in [\tau, t_f] \quad (1.13)$$

Пусть $u^0(t|\tau, z),\ t \in T_\tau$ — оптимальная программа задачи (1.13) для позиции (τ, z)
X_τ — множество состояний z таких, что для позиции (τ, z) существуют программные решения задачи (1.13)
Функция

$$u^0(\tau, z) = u^0(t|\tau, z) \quad (1.14)$$

— оптимальная обратная связь.

Построение (1.14) — синтез оптимальной системы управления.
Подстановка функции (1.14) в уравнение (1.11) — замыкание системы управления

$$\dot{x} = f(x, u^0(t,x), t),\ x(t_0) = x_0 \quad (1.15)$$

Полученное уравнение — математическая модель оптимальной автоматической системы управления.
Для любого состояния x_0 решение уравнения (1.15) с начальным состоянием $x(t_0) = x_0$ — оптимальная траектория для задачи (1.9) - (1.12).
Отметим, что если оптимальная обратная связь построена в классе кусочно-непрерывных управлений, то во многих задачах уравнение (1.15) представляет собой дифференциальное уравнение с разрывной правой частью и, вообще говоря, не имеет классического решения. В этих случаях используют обобщенное решение.

Чтобы избежать аналитических трудностей, таких как определение решения замкнутой системы управления, в задаче оптимального управления (1.9) - (1.12) перейдем от класса кусочно-непрерывных доступных управлений к дискретным управлениям.
Разобьем T на $n \in N$ частей $h = \frac{t_f - t_0}{N}$

$$u(t) = u(s),\ t \in [s, s+h],\ s \in T_h = \{t_0, t_0 + h, ..., t_f - h\} \quad (1.16)$$

Понятие программы и оптимальной программы в классе дискретных управлений аналогично определению в классе кусочно-непрерывных управлений.
Семейство, в которое прогружается задача (1.9) - (1.12) будет выглядеть сле-

дующим образом:

$$\varphi(x(t_f)) \to \min, \dot{x} = f(x, u, t),\ x(z) = z,\ u(t) \in U,\ t \in T_h. \tag{1.17}$$

Семейство зависит от $(\tau, z), z \in \mathbb{X},\ \tau \in T_h,$
Оптимальная дискретная обратная связь — функция

$$u^0(\tau, z) = u^0(t|\tau, z),\ z \in X_\tau,\ \tau \in T_h, \tag{1.18}$$

1.4 Оптимальное управление в реальном времени и реализация оптимальной обратной связи

Проанализируем, как используется $u^0(\tau, z) = u^0(t|\tau, z),\ \tau \in T_h,\ z \in \mathbb{X}_\tau$. Продолжая процесс таким образом получим управляющее воздействие $u^*(t), t \in T$ и траекторию $x^*(t), t \in T$ удовлетворяющее тождеству $x^*(t) \equiv f(x^*(t), u^*(t), t) + w^*(t), t \in T, x^*(t_0) = x_0$

$$u^*(t), t \in T : u^*(t) = u^0(\tau, x^*(z)), t \in [\tau, \tau + h], \tau \in T_h. \tag{1.19}$$

— реализация оптимальной обратной связи (1.9) в конкретном процессе управления.
Отсюда видно, что в конкретном процессе управления оптимальная обратная связь (1.9) не используется целиком на всей области ее определения. Нужны лишь ее значения вдоль одной последовательности измеряемых состояний физического объекта $x^*(\tau), \tau \in T_h$, и достаточно уметь для каждой текущей позиции $(\tau, x^*(\tau)), \tau \in T_h$, вычислять значение реализации $u^*(t) = u^0(\tau, x^*(\tau)),\ t \in [\tau, \tau + h]$ оптимальной обратной связи за время $s(\tau) < h$
Будем говорить, что процесс управления осуществляется в реальном времени, если для каждого текущего момента $\tau \in T_h$ зачение $u^*(\tau)$ вычисляется за время $s(\tau) < h$, т.е. до получения следующего измерения $x^*(\tau + h)$. Устройство, способное реализовать оптимальную обратную связь в реальном времени будем называть оптимальным регулятором.
Таким образом, с использованием принципа управления в реальном времени задача синтеза ОУ типа ОС сводится к построению алгоритма работы оптимального регулятора.

1.5 Теория управлнеия по прогнозирующей модели MPC

Постановка задачи

$$\begin{cases} \dot{x} = f(x, u),\ u(t) \in U, \\ x(0) = x_0,\ x(\tau) \in X. \end{cases}$$

$$\min_u J = \int_t^{\inf} F(x(\tau), u(\tau)) d\tau$$

$\varphi(t)$ — устойчива, если $\forall \varepsilon > 0\ \exists \delta(\varepsilon) > 0\ \|x(0) - \varphi(0)\| \leq \delta \Rightarrow \|x(t) - \varphi(t)\| \leq \varepsilon$

Ассимптотическая устойчивость:

1. $\varphi(t)$ — устойчива
2. $\exists \delta\ \|x(0) - \varphi(0)\| \leq \delta\ \lim \lim_{t \to \infty} \|x(t) - \varphi(t)\| = 0$

Динамика:

$$\begin{cases} \dot{x} = f(x, u),\ x \in \mathbb{R}^n, \\ x(0) = x_0,\ u \in \mathbb{R}^m. \end{cases}$$

Ограничения:

$$x(t) \in X,\ u(t) \in U\ \forall t \geq 0.$$

Задача MPC: В момент времени t, зная начальное состояние $x(t)$

$$\min_{\overline{u}(\cdot, t)} J(x(t), \overline{u}(\cdot, t))$$

$\overline{u}$ — предсказанное
$u(t, \tau)$ — $u(t)$ в предсказанный момент времени τ

$$J(x(t), \overline{u}(\cdot, t)) = \int_t^{t+T} L(\overline{x}(\tau, t), \overline{u}(\tau, t)) d\tau,$$

$$\dot{\overline{x}} = f(\overline{x}, \overline{u}),$$

$$\overline{x}(t,t) = x(t),$$

$$\overline{u}(\tau, t) \in U,$$

$$\overline{x}(\tau, t) \in X,\ \forall \tau \in [t, t+T].$$

L — стоимость этапа, δ — шаг дискретизации.

ГЛАВА 2

ПАРАМЕТРИЧЕСКОЕ ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

Задачи математического программирования, такие как линейное, выпуклое, нелинейное программирование, помимо переменных оптимизации содержат коэффициенты — параметры — задающие, например, целевые функции или функциональные ограничения. Изучением влияния *малых возмущений* этих параметров на решение задачи (оптимальный план, оптимальное значение целевой функции) посвящен анализ чувствительности. С другой стороны, существует ряд прикладных задач, в которых интерес представляет характеризация решения для полного диапазона значений параметров. Раздел теории математического программирования, занимающийся исследованием таких задач называется *параметрическим программирование*. При этом термин параметрическое программирование относится к задачам, в которых исследуется влияние скалярного параметра, в то время как изучением задач с векторными параметрами занимается многопараметрическое программирование.

Как отмечалось во введении, многопараметрическое программирование — основной инструмент построения оптимальных и стабилизирующих обратных связей в данной дипломной работе. В связи с этим, в настоящей главе приводятся основные определения, теоремы и алгоритмы вычисления решений в задачах многопараметрического программирования.

2.1 Общие положения

Первый метод решения параметрических задач линейного программирования был предложен в работе [?] в 1955 г., а затем развит в работах [?] и [?]. Интерес к многопараметрическому программированию возрос с развитием теории систем и оптимального управления. Это связано с тем, что для динамических систем с дискретным временем задачи оптимального управления с ограничениями на управления и состояния и конечным горизонтом управления могут быть сформулированы как задачи математического программирования. Это будут задачи линейного или выпуклого программирования для линейных систем с линейными или выпуклыми функционалами и ограничениями или задачи нелинейного программирования. Как обсуждалось в главе 1, интерес в таких задачах представляет построение оптимальной обратной связи, как функции позиции — пары, в общем случае, состоящей из момента времени и состояния системы в этот момент. Тогда параметром, от которого зависит критерий качества и ограничения, является позиция динамического процесса.

Используя многопараметрическое программирование, можно охарактеризовать и вычислить решение задачи оптимального управления в явном виде как функцию позиции на некотором ограниченного множестве допустимых позиций.

В теории управления по прогнозирующей модели, как обсуждалось в разд. **??**, требуется, чтобы прогнозирующая задача оптимального управления решалась в реальном времени для вычисления текущего управляющего воздействия. Прогнозирующая задача зависит от текущего состояния объекта управления, который берется в качестве начального состояния в прогнозирующей задаче и выступает в качестве параметра. С помощью многопараметрического программирования решение в реальном времени (онлайн) может быть перенесено оффлайн, до начала процесса управления. Тогда в реальном времени все вычисления сводятся к вычислению решения задачи многопараметрического программирования для конкретного значения параметра, что значительно проще решения прогнозирующей задачи оптимального управления.

Многопараметрический анализ использует понятие критической области — подмножества параметров задачи параметрического программирования, в котором локальные условия оптимальности остаются неизменными. В этой главе сначала приведем основные результаты нелинейного многопараметрического программирования из работы [**?**], в частности, результаты, касающиеся непрерывной зависимости решений от параметров, определения невырожденности задач, понятия критической области. Затем опишем алгоритмы решения многопараметрических задач линейного программирования (mp-LP) и многопараметрических задач квадратичного программирования (mp-QP).

Основная идея алгоритмов, представленных в этой главе, состоит в том, чтобы построить критическую область в окрестности заданного значения параметра, используя необходимые и достаточные условия оптимальности, а затем рекурсивно расширить пространство параметров за пределы этой области. Все алгоритмы чрезвычайно просты в реализации, если доступны программные реализации численных методов решения задач нелинейного программирования.

2.2 Непрерывная зависимость решений от параметров в общих многопараметрических задачах

Рассмотрим задачу нелинейного программирования, зависящую от параметра x, который входит в целевую функцию и в ограничения:

$$\begin{aligned} &J^0(x) = \min_z f(z,x), \\ &g(z,x) \leq 0, \end{aligned} \tag{2.1}$$

где $z \in M \subseteq \mathbb{R}^s$ — переменная оптимизации, $x \in X \subseteq \mathbb{R}^n$ — параметр, $f : \mathbb{R}^s \times \mathbb{R}^n \to \mathbb{R}$ — целевая функция, а функция $g : \mathbb{R}^s \times \mathbb{R}^n \to R^{n_g}$ задает ограничения, $g_i(z,x)$ — i-ая компонента вектор-функции $g(x,z)$.

Малые вариации параметра x в задаче математического программирования (2.1) в зависимости от свойств функций f и g может приводить к малым вариациям оптимального решения как функции параметра x, но может и давать его значительные вариации. Цель последующего изложения — установить эти вариации.

При заданном значении параметра x обозначим:

- $R(x)$ — допустимое множество переменных z (множество планов) задачи (2.1) при фиксированном значении параметра:

$$R(x) = \{z \in M \mid g(z,x) \leq 0\}; \tag{2.2}$$

- K^* — множество допустимых значение параметра, т. е.

$$K^* = \{x \in \mathbb{R}^n \mid R(x) \neq \varnothing\}; \tag{2.3}$$

- $J^0(x)$ — оптимальное значение задачи (2.1):

$$J^0(x) = \inf_z \{f(z,x) \mid z \in R(x)\}; \tag{2.4}$$

- $Z^0(x)$ — множество оптимальных решений (оптимальных планов) задачи (2.1):

$$Z^0(x) = \{z \in R(x) \mid f(z,x) = J^0(x)\}. \tag{2.5}$$

Множество $Z^0(x)$ для краткости будем называть оптимальным множеством. Если $Z^0(x)$ является одноточечным множеством для всех x (задача имеет единственное решение), то $z^0(x) \triangleq Z^0(x)$ будем называть оптимальным планом.

Будем предполагать, что K^* замкнуто и $J^0(x)$ конечно при всех $x \in K^*$.

Для того, чтобы функция $J^0(x)$ и оптимальное множество $Z^0(x)$ обладали хорошими свойствами, важным будет условие непрерывности отображения $R : x \mapsto R(x) \subseteq M$. Дадим ряд определений.

Определение 2.1 Отображение $R(x)$ открыто в точке $\overline{x} \in K^*$, если для последовательности $\{x^k\} \subset K^*,\ x^k \to \overline{x}$ и точки $\overline{z} \in R(\overline{x})$ существуют целое число m и последовательность $\{z^k \in M\}$ такие, что $z^k \in R(x^k)$ при $k \geq m$ и $z^k \to \overline{z}$.

Определение 2.2 Отображение $R(x)$ замкнуто в точке $\overline{x} \in K^*$, если $\{x^k\} \subset K^*,\ x^k \to \overline{x},\ z^k \in R(x^k)$, и $z^k \to \overline{z}$ влечет $\overline{z} \in R(\overline{x})$.

Определение 2.3 Отображение $R(x)$ непрерывно в точке $\overline{x} \in K^*$, если оно открыто и замкнуто в точке $\overline{x}$. R непрерывно в K^*, если R непрерывно для любого $x \in K^*$.

На основе приведенных определений проверка непрерывности отображения представляется неосуществимой. Однако существуют достаточные условия непрерывности отображения R на основе свойств функций, задающих ограничения задачи (2.1).

Теорема 2.1 [] Пусть выполняются следующие условия: 1) M выпукло; 2) все функции $g_i(z, x)$ непрерывны на $M \times X$ и выпуклы по z для каждого фиксированного $x \in X$; 3) существует такое $\overline{z}$, что $g(\overline{z}, \overline{x}) < 0$. Тогда $R(x)$ — непрерывное отображение.

Теорема 2.2 [] Пусть выполняются следующие условия: 1) M выпукло; 2) все функции $g_i(z, x)$ непрерывны на $M \times X$ и выпуклы по z и x. Тогда $R(x)$ — непрерывное отображение.

Следующие две теоремы дают достаточные условия непрерывности оптимального значения значения и оптимального плана в задаче (2.1).

Теорема 2.3 Рассмотрим задачу (2.1). Если $R(x)$ — непрерывное отображение и функция $f(z, x)$ непрерывна, то $J^0(x)$, $x \in K^*$ непрерывна.

Теорема 2.4 Рассмотрим задачу (2.1). Пусть выполняются условия: 1) $R(x)$ — непрерывное отображение; 2) множество $R(x)$ является выпуклым для каждого $x \in K^*$, 2) функция $f(z, x)$ непрерывна и строго квазивыпукла по переменной z для каждого x. Тогда $J^0(x)$ и $z^0(x)$ — непрерывные функции.

Теоремы 2.1 и 2.2 можно объединить с теоремами 2.3 и 2.4, чтобы получить следующие следствия:

Следствие 2.1 Рассмотрим многопараметрическую задачу нелинейного программирования (2.1). Предположим, что

1) M — компактное выпуклое множество в R^s;

2) f и g непрерывны на $M \times \mathbb{R}^n$;

3) каждая компонента g выпукла на $M \times K^*$.

Тогда функция $J^0(x)$, $x \in K^*$ непрерывна.

Следствие 2.2 Рассмотрим многопараметрическую задачу нелинейного программирования (2.1). Предположим, что

1) M — компактное выпуклое множество в R^s;

2) f и g непрерывны на $M \times \mathbb{R}^n$;

3) каждая компонента g выпукла на M для каждого $x \in K^*$.

Тогда $J^0(x)$ непрерывна в точке x, если существует точка $\overline{z}$ такая, что $g(\overline{z}, x) < 0$.

2.3 Критические области в задаче многопараметрического линейного программирования

Рассмотрим многопараметрическую задачу линейного программирования (mp-LP)

$$\begin{aligned} &J^0(x) = \min_z J(z, x) = c'z, \\ &Gz \leq W + Sx, \end{aligned} \tag{2.6}$$

где $z \in \mathbb{R}^s$ — переменные оптимизации, $x \in \mathbb{R}^n$ — вектор параметров, $J(z, x) \in \mathbb{R}$ — целевая функция и $G \in \mathbb{R}^{m\times s}$, $c \in \mathbb{R}^s$, $W \in \mathbb{R}^m$, и $S \in \mathbb{R}^{m\times n}$.

Для заданного многогранного множества параметров $K \subset R^n$

$$K \triangleq \{x \in \mathbb{R}^n : Tx \leq Z\} \tag{2.7}$$

обозначим через $K^* \subseteq K$ — множество параметров $x \in K$, при которых задача (2.6) допустима (имеет решение).

Цель последующего изложения — определить допустимую область параметров $K^* \subseteq K$, выражения для оптимального значения целевой функции и для одного из оптимальных планов $z^0(x) \in Z^0(x)$.

Дадим следующие определения прямой и двойственной невырожденности задачи (2.6). Определения приводятся в терминах активных ограничений на некотором плане $z(x)$. Напомним, что ограничение-неравенство $g_i(z,x)$ активно, если оно обращается в равенство: $g_i(z(x),x) = 0$.

Определение 2.4 Для любого заданного $x \in K^*$ задача (2.6) называется прямо невырожденной, если существует $z^0(x) \in Z^0(x)$ такое, что число активных ограничений на плане $z^0(x)$ больше, чем число переменных s.

Определение 2.5 Для любого данного $x \in K^*$ задача (2.6) называется двойственно невырожденной, если двойственная ей задача прямо невырождена.

Многопараметрический анализ основан на понятии критической области (critical region — CR).

В [4] критическая область определяется как подмножество пространства параметров, на котором некоторый фиксированный базис задачи линейного программирования является оптимальным. Алгоритм, предложенный в [4] для решения mp-LP, генерирует непересекающиеся критические области посредством построения и исследования графа базисов. На графе базисов вершины представляют собой оптимальные базисы данной многопараметрической задачи, и две вершины соединены ребром, если можно перейти от одного базиса к другому за одну итерацию (в этом случае базисы называются соседними).

В [?] определение критических областей связано не с базисами, а с набором активных ограничений (см. также [8]).

Пусть $J \triangleq \{1, ..., m\}$ — множество индексов ограничений. Для любого $A \subseteq J$ обозначим через G_A и S_A — подматрицы матриц G и S, соответственно, содержащие строки с индексами из A, через G_j, S_j и W_j — j-ую строку G, S и W, соответственно.

Определение 2.6 Оптимальным разбиением J, связанным с параметром x, назовем пару $(A(x), NA(x))$, где

$$A(x) \triangleq \{j \in J : G_j z^0(x) - S_j x = W_j \ \forall z^0(x) \in Z^0(x)\},$$

$$NA(x) \triangleq \{j \in J : \exists z^0(x) \in Z^0(x),\ G_j z^0(x) - S_j x < W_j\}.$$

Понятно, что множества $A(x), NA(x)$ не пересекаются и их объединением является множество всех индексов строк J.

Для заданного $x^* \in K^*$ обозначим $(A, NA) \triangleq (A(x^*), NA(x^*))$, и далее определим множества

$$\begin{aligned} CR_A &\triangleq \{x \in K : A(x) = A\}, \\ \overline{CR}_A &\triangleq \{x \in K : A(x) \supseteq A\}. \end{aligned} \tag{2.8}$$

Множество CR_A является критической областью, связанной с некоторым фиксированным множеством активных ограничений A, т. е. это множество всех параметров x, таких что ограничения $i \in A$, активны на оптимальном решении $z^0(x)$ задачи (2.6). Ясно, что $\overline{CR}_A \supseteq CR_A$.

Теорема 2.5 Пусть $(A, NA) \triangleq (A(x^*), NA(x^*))$ для некоторого $x^* \in K$, и пусть $d = \dim(\text{range} G_A \bigcap \text{range} S_A)$. Если $d = 0$, то $CR_A = \{x^0\}$. Если $d > 0$, то

i) CR_A — открытый многогранник размерности d;

ii) $\overline{CR}_A$ — замыкание CR_A;

iii) каждая грань $\overline{CR}_A$ есть множество $CR_{A'}$ для некоторой подматрицы $A' \supseteq A$.

Замечание 2.1 Открытый многогранник — множество вида $\{x : Px < q\}$, его замыкание — $\{x : Px \leq q\}$.

Из теоремы 2.5 и определения критических областей (2.8) следует, что множество K^* всегда *разбивается единственным образом*. Отметим, что это не так в подходе [5], где критическая область определяется на основе базиса решения задачи линейного программирования. Цель дальнейшего изложения — построение всех полноразмерных критических областей, содержащихся в K^*, в соответствии с определением (2.8).

Напомним некоторые свойства функции $J^0(x) : \mathbb{R}^n \to \mathbb{R}$ и множества K^* из монографии [5].

Теорема 2.6 ([5], стр. 178) *Пусть для фиксированного $x_0 \in K$ существует оптимальное решение $z^0(x_0)$ задачи (2.6). Тогда для всех $x \in K$ задача (2.6) имеет либо имеет решение, либо имеет пустое множество планов.*

Сформулированная теорема утверждает, что в задаче линейного программирования(2.6) либо нет решения (несовместны ограничения), либо решение существует. Третья ситуация — неограниченность целевой функции снизу — невозможна.

Теорема 2.7 ([5], стр. 179) *Пусть $K^* \subseteq K$ — множество всех параметров x, таких что задача линейного программирования (2.6) имеет оптимальное решение $z^0(x)$. Тогда K^* — замкнутый многогранник в $\mathbb{R}^n$.*

Следующая теорема формулирует свойства, которыми обладает решение в задаче многопараметрического линейного программирования.

Теорема 2.8 ([5], стр. 180) *Функция $J^0(x)$, $x \in K^*$, является выпуклой и кусочно-аффинной на K^* (в частности, аффинной в каждой критической области CR_{A_i}).*

Если для каждого $x \in K^*$ оптимальное решение $z^0(x)$ единственно, то функция $z^0(x)$, $x \in K^*$, является непрерывной и кусочно-аффинной. В противном случае (в случае неединственного решения) всегда можно определить непрерывную и кусочно-аффинную функцию решения $z^0(x) \in Z^0(x)$, $x \in K^*$.

В следующем разделе опишем алгоритм для определения множества K^*, его разбиения на полноразмерные критические области CR_{A_i}, построения кусочно-аффинных функций оптимального значения $J^0(x)$, $x \in K^*$, и оптимального решения $z^0(x)$, $x \in K^*$.

2.4 Алгоритм построения многопараметрического решения

Алгоритм состоит из двух основных этапов:

- Определить размерность $n' \leq n$ наименьшего аффинного подпространства $\mathcal{K}$, содержащего K^*. Если $n' < n$, найти уравнения по x, которые определяют $\mathcal{K}$.
- Определить разбиение K^* на критические области CR_{A_i} и найти функции оптимального значения $J^0(x)$, $x \in K^*$, и оптимального решения $z^0(x)$, $x \in K^*$.

Первый этап — предварительный, его цель состоит в том, чтобы уменьшить количество параметров и получить полномерную допустимую область параметров. Это облегчает второй этап, который вычисляет многопараметрическое решение и является основой алгоритма mp-LP.

2.4.1 Определение аффинного подпространства $\mathcal{K}$

Для того, чтобы этап 2 работал с минимальной размерностью вектора параметров, первый этап алгоритма нацелен на поиск подпространства $\mathcal{K} \subseteq \mathbb{R}^n$, содержащего параметры x, при которых задача (2.6) допустима (имеет решение).

Первое простое, но важное наблюдение касается числа линейно независимых столбцов матрицы S (обозначим его r_S). Понятно, что если $r_S < n$, то $n - r_S$ параметров можно исключить. Поэтому далее без потери общности считаем, что столбцы матрицы S линейно независимы.

Существует еще один случай, когда размерность вектора параметров может быть понижена. Таким образом, прежде чем решать многопараметрическую задачу, нужен тест для проверки размерности n' наименьшего аффинного подпространства $\mathcal{K}$, содержащего K^*. Более того, результатом теста будет $n^{'} < n$, нужно получить уравнения, описывающие $\mathcal{K}$ в $\mathbb{R}^n$. Эти уравнения используются для изменения координат, чтобы уменьшить число параметров с n до $n^{'}$ и получить многогранник K^*, который имеет полную размерность в $\mathbb{R}^{n^{'}}$.

Вспомним задачу (2.6) и рассмотрим ее как задачу линейного программирования в пространстве $\mathbb{R}^{s+n}$

$$\begin{aligned} &\min_z J(z,x) = c'z, \\ &Gz - Sx \le W. \end{aligned} \tag{2.9}$$

Понятно, что ограничения в (2.9) определяют многогранник $\mathcal{P}$ в $\mathbb{R}^{s+n}$. Следующая лемма утверждает, что проекция $\Pi_{\mathbb{R}^n}(\mathcal{P})$ многогранника $\mathcal{P}$ на пространство параметров $\mathbb{R}^n$ есть K^*.

Лемма 2.1 $x^* \in K^* \Longleftrightarrow \exists z : Gz - Sx^* \le W \Longleftrightarrow x^* \in \Pi_{\mathbb{R}^n}(\mathcal{P})$.

Как следствие, размерность $n^{'}$ наименьшего аффинного подпространства, содержащего K^*, может быть определена путем вычисления размерности проекции $\Pi_{\mathbb{R}^n}(\mathcal{P})$.

Пусть задано следующее H-представление многогранника: $\mathcal{C} = \{\xi \in \mathbb{R}^s : B\xi \le v\}$, т.е. многогранник $\mathcal{C}$, определяется набором полупространств $B_i\xi \le v_i$.

Определение 2.7 Действительным неравенством многогранника $\mathcal{C}$ назовем такое неравенство $B_i\xi \leq v_i$, что $\exists\overline{\xi} \in \mathcal{C} : B_i\overline{\xi} < v_i$.

Учитывая H-представление многогранника $\mathcal{C}$ следующая простая процедура определяет множество I всех недействительных неравенств $\mathcal{C}$.

Algorithm 1:

Input : многогранник $\mathcal{C}$

Output: Множество I всех недействительных неравенств $\mathcal{C}$

$I \leftarrow \emptyset$
$M \leftarrow \{1, ..., m\}$
while $M \neq \emptyset$ **do**
    $j \leftarrow$ first element of $\mathcal{M}$
    $\mathcal{M} \leftarrow \mathcal{M} \backslash \{j\}$
    решаем следующую задачу линейного программирования:
    $B_i\xi \leq v_i, \forall i \in \mathcal{M}$
    if $B_i\xi = v_i$ **then**
        $I \leftarrow I \cup \{j\}$
    end
    for $h \in \mathcal{M}$ **do**
        if $B_h x^* < v_h$ **then**
            $\mathcal{M} \leftarrow \mathcal{M} \backslash \{h\}$
        end
    end
end

Следующий алгоритм описывает стандартную процедуру для определения размерности $n^{'} \leq n$ наименьшего аффинного подпространства $\mathcal{K}$, содержащего K^*, и, когда $n^{'} \leq n$, он находит уравнения, определяющие $\mathcal{K}$.

Algorithm 2:

Input : матрицы G, S, W

Output: Размерность $n^{'} \leq n$ наименьшего аффинного подпространства $\mathcal{K}$, которое содержит k^{*}, если $n^{'} \leq n$, такая что $\mathcal{K} = \{x \in \mathbb{R}^n \; Tx = Z\}$

Отбрасываем истинные неравенства из ограничений 12

if *никакого неравнества не осталось* **then**

  $\mathcal{K} \leftarrow \mathbb{R}^n$

else

  Разрешим $\mathcal{P}_a \triangleq \{(z,x) : G_a z - S_a x = W_a\}$ быть полученным аффинным подпространством, полученным путем сбора не истинных неравенств

  $\{u_1, ..., u_{k'}\}$ - базис ядра $G_a^{'}$

  if $k^{'} = 0$ **then**

    $\Pi_{\mathbb{R}^n}(\mathcal{P}_a)$ и K^* полноразмерны

    $\mathcal{K} \leftarrow \mathbb{R}^n$

  else

    $\mathcal{K} \leftarrow \{x \mid Tx = Z\}$, где

$$T = -\begin{bmatrix} u_1^{'} \\ \vdots \\ u_{k'}^{'} \end{bmatrix} S_a, \; Z = \begin{bmatrix} u_1^{'} \\ \vdots \\ u_{k'}^{'} \end{bmatrix} W_a$$

  end

end

С учетом приведенных алгоритмов, в дальнейшем, не теряя общности, будем предполагать, что множество K имеет размерность n в $\mathbb{R}^n$.

2.4.2 Определение критических областей

В этом разделе приведем детали алгоритма mp-LP, а именно процедуру построения критических областей CR_{A_i} в заданном многогранном множестве K. Далее будем считать, что все задачи линейного программирования являются прямо и двойственно невырожденными. Описание вырожденных случаев можно найти в работе [**?**].

В силу невырожденности задач, для каждого значения параметра $x \in K^*$ имеется единственное оптимальное решение $z^0(x)$, т.е. $Z^0(x) = \{z^0(x)\}$, $x \in K^*$.

В алгоритме используются:

- прямая допустимость для получения H-представления полиэдральных критических областей,

- условия дополняющей нежесткости для вычисления функции оптимального решения $z^0(x)$, $x \in K^*$,

- двойственная задача для получения функции оптимального значения $J^0(x)$, $x \in K^*$.

Двойственная задача для (2.6) имеет вид

$$\begin{aligned} &\max_y (W + Sx)'y, \\ &G'y = c, \\ &y \le 0. \end{aligned} \tag{2.10}$$

Прямая и двойственная допустимость и условия дополняющей нежесткости для задач (2.6), (2.10) имеют вид

$$\begin{aligned} \text{прямая допустимость: } & Gz \le W + Sx, \\ \text{двойственная допустимость: } & G'y = c, \; y \le 0, \\ \text{условия доп. нежесткости: } & (G_j z - W_j - S_j x) y_i = 0, \; j \in J. \end{aligned} \tag{2.11}$$

Выберем произвольный вектор параметров $x_0 \in \mathcal{K}$ и решим прямую и двойственную задачи (2.6), (2.10) для $x = x_0$. Пусть z_0^0 и y_0^0 — решения прямой и двойственной задач, соответственно. Значение z_0^0 определяет следующее оптимальное разбиение

$$\begin{aligned} A(x_0) &\triangleq \{j \in J : G_j z_0^0 - S_j x_0 - W_j = 0\} \\ NA(x_0) &\triangleq \{j \in J : G_j z_0^0 - S_j x_0 - W_j < 0\} \end{aligned} \tag{2.12}$$

и, следовательно, критическую область $CR_{A(x_0)}$.

По предположению решение y_0^0 единственно, а по определению критической области значение y_0^0 остается оптимальным для всех $x \in CR_{A(x_0)}$. Тогда, согласно теореме двойственности, функция оптимального значения в $CR_{A(x_0)}$ определяется как

$$J^0(x) = (W + Sx)'y_0^0, \; x \in CR_{A(x_0)}, \tag{2.13}$$

и является аффинной функцией x на $CR_{A(x_0)}$, как указано в теореме 2.8.

Более того, для оптимального разбиения (2.12) условие прямой допустимости можно переписать в виде:

$$G_A z^0(x) = W_A + S_A x, \tag{2.14}$$

$$G_{NA} z^0(x) < W_N A + S_{NA} x. \tag{2.15}$$

В отсутствие двойственной невырожденности прямое решение единственно, и система (2.14) может быть решена, чтобы получить решение $z^0(x)$. Фактически, уравнения (2.14) образуют систему из l равенств, где в отсутствие прямой невырожденности $l = s$ — число активных ограничений. Из (2.14) следует, что

$$z^0(x) = -G_A^{-1} S_A x + G_A^{-1} W_A = Ex + Q, \tag{2.16}$$

откуда следует линейность z^0 по x.

Из условия прямой допустимости (2.15) получаем следующее представление критической области $CR_{A(x_0)}$

$$G_{NA}(Ex + Q) < W_{NA} + S_{NA} x. \tag{2.17}$$

Ее замыкание $\overline{CR}_{A(x_0)}$ имеет вид $G_{NA}(Ex + Q) \le W_{NA} + S_{NA} x$.

После определения критической области $\overline{CR}_{A(x_0)}$ необходимо исследовать оставшуюся часть пространства $R^{\text{rest}} = K \setminus \overline{CR}_{A(x_0)}$ и создать новые критические области.

Эффективный подход к разбиению остального пространства был предложен в [9] и формально доказан в [10]. Далее сформулируем теорему, которая обосновывает процедуру формирования остальных критических областей.

Теорема 2.9 Пусть $Y \subseteq \mathbb{R}^n$ — многогранник, $R_0 \triangleq \{x \in Y : Ax \le b\}$ — полиэдральное подмножество Y, где $b \in R^{m\times 1}$, $R_0 \neq \emptyset$. Пусть

$$R_i = \left\{ x \in Y : \begin{array}{l} A^i x > b^i \\ A^j x \le b^j, \ j < i \end{array} \right\}, \quad i = 1, ..., m,$$

где $b \in \mathbb{R}^{m\times 1}$ и пусть $R^{\text{rest}} \triangleq \bigcup_{i=1}^m R_i$.

Тогда

i) $R^{\text{rest}} \cup R_0 = Y$,

ii) $R_0 \cap R_i = \emptyset, \ R_i \cap R_j = \emptyset, \ \forall i \neq j$,

т.е. $\{R_0, R_1, \dots, R_m\}$ — разбиение Y.

Доказательство. i) Нужно доказать, что точка $x \in Y$ либо принадлежит R_0, либо принадлежит R_i для некоторого i. Если $x \in R_0$, то результат очевиден. В противном случае, существует такой индекс i, что $A^i x > b^i$. Пусть $i^* = \min_{i \leq m}\{i : A^i x > b^i\}$. Тогда $x \in R_{i^*}$, поскольку $A^{i^*} x > b^{i^*}$ и $A^j x \leq b^j$, $\forall j < i^*$, по определению i^*.

ii) Пусть $x \in R_0$. Тогда не существует ни одного i такого, что $A^i x > b^i$, из чего следует, что $x \notin R_i$, $\forall i \leq m$. Пусть $x \in R_i$ и возьмем $i > j$. Поскольку $x \in R_i$, по определению $R_i(i > j)$ имеем $A^j x \leq b^j$, из чего следует, что $x \notin R_j$.

2.5 Multi-Parametric Toolbox

Multi-Parametric Toolbox (или MPT) - это набор инструментов на основе Matlab с открытым исходным кодом для параметрической оптимизации, вычислительной геометрии. Инструментарий предлагает широкий спектр алгоритмов, скомпилированных в удобном и доступном для пользователя формате. Полученные в результате законы оптимального управления могут быть встроены в приложения в виде кода на C или развернуты на целевых платформах с использованием Real Time Workshop.

Существуют наборы инструментов, которые предлагают операции, связанные исключительно с вычислительной геометрией, то есть наборы инструментов GEOMETRY, CGLAB и Ellipsoidal Toolbox. Другие наборы инструментов помимо геометрических предлагают также алгоритмы для вычисления и реализации процедур управления, например набор инструментов Hybrid, MOBY-DIC, RACT, PnPMPC и RoMulOC. MPT также является одним из инструментов, который сочетает вычислительную геометрию с управляющими программами. Многие из этих наборов инструментов, включая MPT, используют YALMIP, который обеспечивает язык высокого уровня для моделирования и формулирования задач оптимизации. Содержание MPT

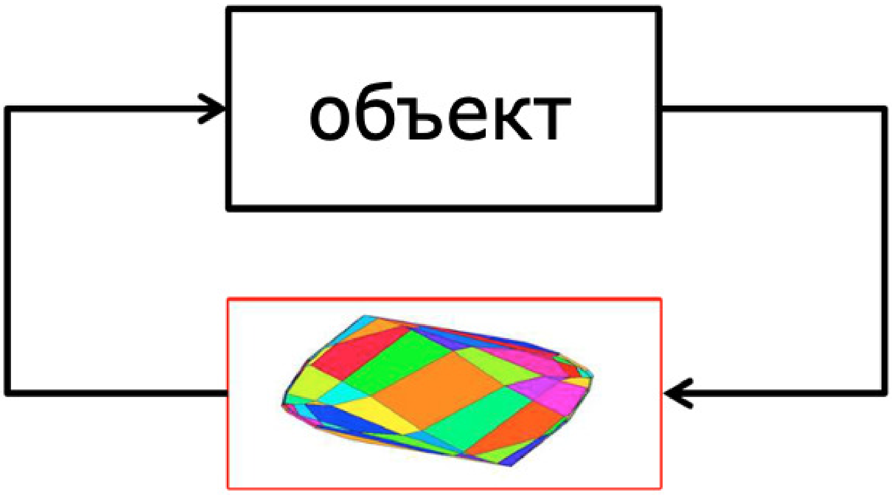
можно разделить на четыре модуля:

- Моделирование динамических систем
- MPC на основе синтеза
- Анализ решения с обратной связью
- Развертывание регуляторов MPC на аппаратном уровне

Каждая часть представляет собой один этап в разработке и реализации явного MPC. Модуль моделирования MPT позволяет описывать системы с дискретным временем с линейной динамикой. Последние могут быть напрямую импортированы из среды HYSDEL. Модуль управления позволяет формулировать и решать ограниченные задачи оптимального управления для линейных систем. Модуль анализа предоставляет методы для исследования поведения и производительности замкнутого контура. Кроме того, он также содержит методы, позволяющие уменьшить сложность явных обратных связей MPC.

Рассмотрим самые важные возможности, которые предоставляет MPT:

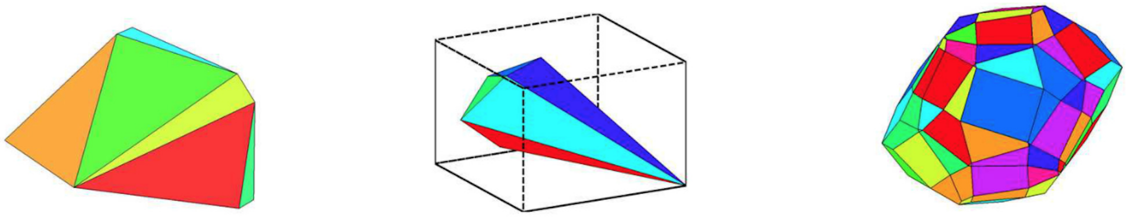
1. Набор инструментов Matlab для применения явного MPC
    - Реализация MPC в режиме реального времени
2. Подходы
    - Автономно: решить задачу оптимального управления параметрически
    - Онлайн: оценить полученную обратную связь PWA



3. Расширенная поддержка многогранников

   неограниченные, низкоразмерные множества
   P = Polyhedron('A, A, 'b', b, 'Ae', Ae, 'be', be)
   Q = Polyhedron('V', V, 'R', R)

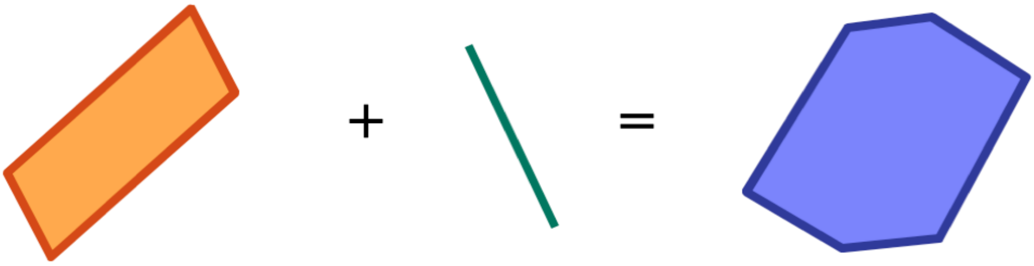
объединение многогранников с определенными свойствами
U = PolyUnion('Set', P, 'Convex', 1, 'Overlaps', 0)



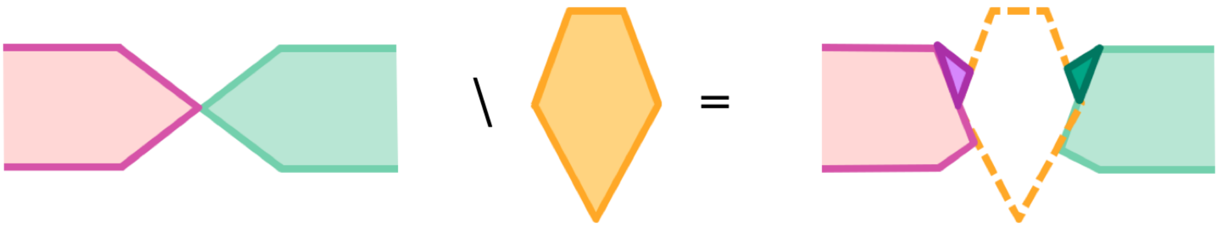
4. Поддерживаемые геометрические операции

   Сумма Минковского, разность Понтрягина, проекции, разности множеств и т.д.

   $S = P + Q$



   $T = U \backslash S$



5. Генерация явного решения

   - Построить объект регулятора MPC онлайн

$ctrl = MPCController(model, N)$
$u = ctrl.evaluate(x)$

- Настроить регулятор и закончить цикл
  $loop = ClosedLoop(ctrl, model)$
  $data = loop.simulate(x_0, Nsim)$
- Экспорт в явную форму
  $expl_{ctrl} = ctrl.toExplicit()$

Наглядно данный алгоритм проиллюстрирован на рисунке 2.1

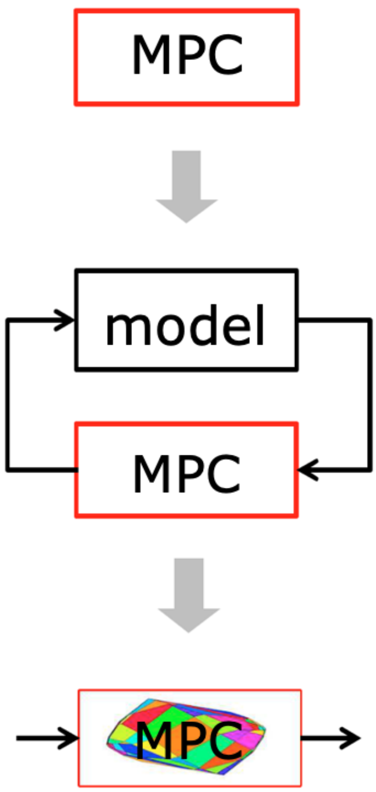


Рис. 2.1: Алгоритм

2D Пример осциллятора

- Функция стоимости
  $expl_{ctrl}.cost.fplot()$ (Рисунок 2.2)

- Закон обратной связи
  $expl_{ctrl}.feedback.fplot()$ (Рисунок 2.3)

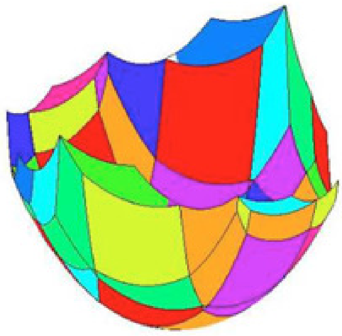


Рис. 2.2: Функция стоимости

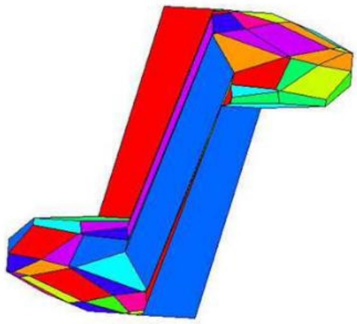


Рис. 2.3: Закон обратной связи

- Область
  $expl_{ctrl}.partition.plot()$ (Рисунок 2.4)

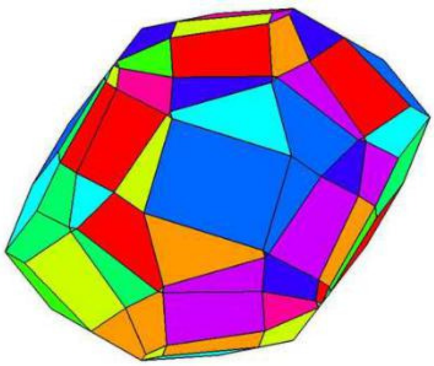


Рис. 2.4: Область

2.5.1 Единый "солвер"для параметрических задач

MPT может быть очень полезен в решении параметрических задач. Ниже приведен пример решения такой задачи.

$$\min 1/2u^T Hu + c^T u Au \leq b + E\xi,\ u \geq 0$$

$$\Downarrow$$

$$Hu + c + A^T\lambda - \upsilon = 0 Au \leq b + E\xi,\ u \geq 0 \lambda^T(Au - b - E\xi) = 0,\ \upsilon^T u = 0$$

$$\Downarrow$$

Найти w, z

$$w - Mz = q + G\xi w \geq 0,\ z \geq 0 w^T z = 0$$

$$\Downarrow$$

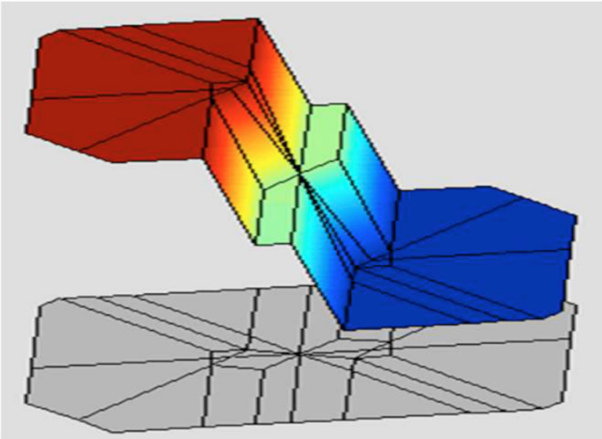


Рис. 2.5: Результат

Резултат выполнения программы показан на рисунке 2.5

Целью Multi-Parametric Toolbox (MPT) является предоставление эффективных вычислительных средства для получения регуляторов обратной связи для типов задач оптимального управления в среде программирования Matlab. Многопараметрическим программированием, линейным или задачи квадратичной оптимизации решается в автономном режиме. Соответствующее решение принимает форму обратной связи штата PWA. В частности, пространство состояний разбивается на многогранные множества и для каждого из этих наборов закон оптимального управления задается как одна аффинная функция. Для квадратичных задач, регулятор обратной связи может быть получен для ограниченных линейных системы с применением методов многопараметрического программирования. В настоящее время принято аппроксимировать оптимальное управление с ограниченным бесконечным временем (CITOC) Оптимальное управление кусочно-аффинными системами также вызвало большой интерес к исследованиям. сообщества, поскольку системы PWA представляют собой мощный инструмент для аппроксимации нелинейных систем и из-за их эквивалентности гибридным системам. Алгоритмы для вычислений регуляторы с обратной связью для систем PWA с ограничениями были представлены для квадратичного и линейного цели, а также включены в этот инструментарий. С помощью регуляторов обратной связи, которые минимизируют конечную стоимость затрат времени, также возможно получить оптимальное по бесконечному времени решение для систем PWA. Несмотря на то, что многопараметрические подходы основаны на автономном вычислении закона обратной связи, вычисление может быстро стать препятствующим для больших проблем. Это связано не только высокой сложности многопараметрических программ, но в основном из-за экспоненциального количества переходов между регионами, которые могут произойти, когда регулятор вычисляется в режиме динамического программирования.

Итак, Multi-Parametric Toolbox - это набор алгоритмов для моделирования, управления, анализа и развертывания ограниченных оптимальных регуляторов, разработанных в Matlab. Он обладает мощной геометрической библиотекой, которая расширяет возможности применения набора инструментов за пределы оптимального управления для различных задач, возникающих в вычислительной геометрии.

ГЛАВА 3

ПРИМЕНЕНИЕ ПАРАМЕТРИЧЕСКОГО ПРОГРАММИРОВАНИЯ В ЗАДАЧАХ СИНТЕЗА И MPC

Мы изучаем задачи оптимального управления и построение оптимальной обратной связи. В нашем подходе будет использоваться многопараметрическое программирование, и нашей главной целью будет получение решения с обратной связью по состоянию, а также получение алгоритмов для его эффективного вычисления.

В нашем фреймворке параметрическое программирование является основным методом, используемым для изучения и вычисления законов оптимального управления с обратной связью. Фактически, мы формулируем конечные задачи оптимального управления в виде математических программ, в которых входная последовательность является вектором оптимизации. В зависимости от динамической модели системы, характера ограничений и используемой функции, получается другая математическая программа. Текущее состояние динамической системы входит в функцию стоимости и ограничения в качестве параметра, который влияет на решение математической программы. Мы изучаем структуру решения при изменении этого параметра и описываем алгоритмы решения многопараметрических линейных, квадратичных и смешанных целочисленных программ. Они представляют собой основные инструменты для вычисления законов оптимального управления с обратной связью.

Мы показываем, что решение всех этих задач оптимального управления может быть выражено в виде кусочно-афинного закона обратной связи. Кроме того, закон оптимального управления непрерывен, а функция значений выпукла и непрерывна.

Оптимальное управление ограниченными линейно-кусочно-аффинными (PWA) системами получило отличное интерес к исследовательскому сообществу из-за легкости постановки сложных проблем и их решения.

3.1 Явный MPC

Явный MPC — очень популярный подход, который реализует идею, в которой вычисления переносятся со второго шага базового алгоритма MPC на подготовительный (еще до начала вычислений). Явные схемы на текущий момент времени могут быть использованы только для линейных динамических систем и линейных или квадратичных критериев качества.

В основе явных схем лежат следующие наблюдения: линейные или линейно-квадратичные задачи оптимального управления при численном решении сводятся к задачам линейного или квадратичного программирования, которые зависят от состояния $x = x(t)$, которое, в свою очередь, принимается как параметр. В итоге получаем задачу параметрического программирования, для которой к настоящему моменту были развиты специальные методы решения. Они позволяют нам получить явный закон управления $u(x)$.

Развитие подхода на нелинейные системы является не тривиальным, однако и в линейных случаях есть проблемы, касающиеся эффективности вычислений даже для задач средней размерности.

Ниже продемонстрируем основную идею явного MPC.

Решается задача следующего вида:

$$J(x) = \min_{u(\cdot|t)} \sum_{t}^{t+N-1} L(x(k(t), u(k(t))) + F(x(t+N)t)), \tag{3.1}$$

При условиях

$$x(k+1|\ t) = Ax(k \mid t) + Bu(k \mid t),$$

$$x(t \mid t) = x(t),$$

$$C_x x(k \mid t) \leq dx,\ t \leq k \leq t+N-1,$$

$$C_u u(k \mid t) \leq du,\ C_x x(t+N \mid t) = d_f,$$

$$C_f x(t+N|\ t) \leq d^f,$$

со следующей квадратичной функцией стоимости конкретного этапа и терминальной стоимостью:

$$L(x,u) = x^T Q x + u^T R u,$$
$$F(x) = x^T Q x + u^T R u,\ Q, R > 0,$$

где $C_x, C_u, C_f, d_x, d_u, d^f$ — ограничения на траекторию, управления и терминальное состояние.

Перепишем 3.1 в виде задачи квадратичного программирования:

$$X := [x^T(t+1|\ t),\ \ldots,\ x^T(t+N|\ t)]^T, \tag{3.2}$$

$$U := [u^T(t+1|\ t),\ \ldots,\ u^T(t+N|\ t)]^T, \tag{3.3}$$

$$J(x(t),\ U) = x^T(t)Qx(t) + X^T\tilde{Q}X + U^T\tilde{R}U, \tag{3.4}$$

Далее выражаем состояние системы в момент $t+k$ через $x(t)$ и u:

$$x(t+k \mid t) = A^k x(t) + \sum_{j=0}^{k-1} A^j Bu(t+k-j-1 \mid t),\ k = 1, \ldots, N$$

$$X = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} x(t) + \begin{bmatrix} B & 0 & \vdots & 0 \\ AB & B & \vdots & 0 \\ \cdots & \cdots & \vdots & \cdots \\ A^{N-1}B & A^{N-1}B & \vdots & B \end{bmatrix} U, \tag{3.5}$$

$$\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} \Rightarrow \Omega,\ \begin{bmatrix} B & 0 & \vdots & 0 \\ AB & B & \vdots & 0 \\ \cdots & \cdots & \vdots & \cdots \\ A^{N-1}B & A^{N-1}B & \vdots & B \end{bmatrix} \Rightarrow \Gamma$$

$$X = \Omega x(t) + \Gamma U$$

С помощью (3.5) переписываем (3.4):

$$J(x(t), u) = \frac{1}{2}x^T(t)Yx(t) + \frac{1}{2}U^T HU + x^T(t)FU_x,$$

где

$$Y = 2(Q + \Omega^T\tilde{Q}\Omega)$$

$$H = 2(\tilde{R} + \Gamma^T\tilde{Q}\Gamma)$$

$$F = 2\Omega^T\tilde{Q}\Gamma$$

$$
\begin{bmatrix} C_x & & \\ & \ddots & \\ & & C_x \end{bmatrix} (\Omega x(t) + \Gamma u) \leq \begin{bmatrix} dx \\ \vdots \\ dx \end{bmatrix}^T
$$

Аналагично для u и $x(t+N)$

$$
GU \leq W + Ex(t)
$$

Задача (3.1) примет вид

$$
\min_u \frac{1}{2} u^T HU + x^T FU = \frac{1}{2} x^T Y x,
$$

при условии

$$
GU \leq W + Ex.
$$

Сделаем замену переменных:

$$
z := U + H^{-1} F^T x,
$$

где H^{-1} — положительно определенная матрица.

$$
\overline{Y} = Y - FH^{-1}F^T,
$$

$$
\Rightarrow \min_z \frac{1}{2} z^T Hz + \frac{1}{2} x^T \tilde{Y} x, \tag{3.6}
$$

при условиях

$$
Gz \leq w + sx, \; S = E + GH^{-1},
$$

Задача (3.6) строго выпуклая с допустимым множеством с непустойвнутренней частью, поэтому для нее выполняются условия регулярности Слейтера. Существует единственное оптимальное решение и оно характеризуется условиями Каруша-Куна-Такера.

Явный MPC использует методы параметрического программирования для задачи (3.6) с параметром x.

$$
Hz + G^T \lambda = 0 \tag{3.7}
$$

$$
\lambda_i (G_i z - w_i - s_i x) = 0, \; i = \overline{1, q} \tag{3.8}
$$

$$
\lambda_i \geq 0 \tag{3.9}
$$

$$
Gz \leq w + sx \tag{3.10}
$$

$$
z^*(x) = \dots
$$

$A(x) = i = 1, \ldots, q, G_i z^0(x) = w_i + s_i x$ — оптимальное активное множество

G^A, W^A, S^A — сроки содержащие элементы активного множества

$$z^*(x) = -W^{-1}G^T\lambda = -W^{-1}(G^A)^T\lambda^A \tag{3.11}$$

Для всех активных компонент выполняется равенство:

$$\begin{aligned} &G^A z^*(x) = w(A) + s^A x, \\ &- G^A H^{-1}(G^A)^T\lambda^A = w^A + s^A x, \\ &\lambda^A = -(G^A H^{-1}(G^A)^T)^{-1}(W^A + s^A x). \end{aligned} \tag{3.12}$$

$$3.11,\, 3.12 \Rightarrow z^*(x) = H^{-1}(G^A)^T(G^A W^{-1}(G^A)^T)(W^A + s^A x) \tag{3.13}$$

$3.13 \Rightarrow 3.10$

$$GH^{-1}(G^A)^T(G^A H^{-1}(G^A)^T)^{-1}(W^A + s^A x) \leq w + sx \tag{3.14}$$

$3.14 \Rightarrow 3.9$

$$-(G^A H^{-1}(G^A)^T)^{-1}(W^A + s^A x) \geq 0 \tag{3.15}$$

Алгоритм:

1. $x_0 \in X$ (например $x_0 = 0$)
2. Решить задачу при $x = x_0 \Rightarrow z^*(x_0)$
3. Определить активные ограничения $\Rightarrow G^A, W^A, s^A$
4. Найти из 3.14, 3.15 CR^A — критическая область
5. Перейти через границу CR^A, взять новое x_0 и $\rightarrow 2$

Теорема 3.1 Пусть у нас имеется линейная система, линейные ограничения, функция стоимости квадратичная. Тогда полученный регулятор $u_{MPC}(x)$ является непрерывным и кусочно-линейным. Функция, предствляющая собой оптимальную стоимость $J^*(x)$ также является непрерывной выпуклой и кусочно-квадратичной

3.2 Синтез оптимальной системы

3.2.1 Успокоение материальной точки

$$\begin{cases} x_f \to \min, \\ \dot{x} = Ax + Bu,\ x(0) = x_0,\ x(t_f) = x_f, \\ \|u\|_\infty \le 1,\ t \in T = [0,\ t_f]. \end{cases}$$

Управление принимает лишь значения ± 1. Задача синтеза будет решена, если в пространстве $\mathbb{R}^n$ мы укажем области s_- и s_+ такие что, когда
$x \in s_- \Rightarrow u(t) = -1$
$x \in s_+ \Rightarrow u(t) = 1$
$s_- \cup s_+ = \mathbb{R}^n$
Границы этих областей — поверхности переключения. Рассмотрим задачу о наискорейшем успокоении с помощью граничных управлений материальной точки единичной массы движущейся по прямой из заданного начального положения $x(0) = x_{01}$ с заданной начальной скоростью $\dot{x}(0) = x_{02}$. Под наискорейшим успокоением понимается перевод в начало координат и остановка за минимальное время t_f.
$x(t_f) = 0\ \dot{x}(t_f) = 0$

$$t_f \to \min$$

$$\begin{cases} \dot{x}_1 = x_1 \\ \dot{x}_2 = u \end{cases}$$

$$x_1(0) = x_{01} \quad x_1(t_f) = 0 \quad x = x_1$$

$$x_2(0) = x_{02} \quad x_2(t_f) = 0 \quad \dot{x} = x_2$$

$$|u(t)| \le 1,\ t \in [0,\ t_f]$$

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} - b = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$x \in R^2,\ \dot{x} = Ax + Bu$$

$$H(x, \psi, u) = \psi^{'}(Ax + Bu) = \psi_1 x_2 + \psi_2 u$$

$$\begin{cases} \dot{\psi}_1 = 0 \\ \dot{\psi}_2 = -\psi_1 \end{cases}$$

$$\psi_1(t) = C_1,\ \psi_2(t) = -C_1 t + C_2,\ u(t) = sign(\psi_2(t)),\ t \geq 0$$

$\psi_2(t)$ линейна $\Rightarrow$ меняет знак не более одного раза

1.

$$u = 1 \Rightarrow \begin{cases} \dot{x_1} = x_2 \\ \dot{x_2} = 1 \end{cases}$$

$$x_2 = t + C_1$$

$$x_1 = t^2/2 + C_1 t + C_2$$

$$t = x_2 - C_1$$

$$x_1 = x_2^2/2 + \overline{C}$$

2.

$$u = -1 \Rightarrow \begin{cases} \dot{x_1} = x_2 \\ \dot{x_2} = -1 \end{cases}$$

$$x_2 = -t + C_1$$

$$x_1 = -t^2/2 + C_1 t + C_2$$

3.3 Динамическое программирование

Постановка задачи: пусть имеется некоторый объект, изменяющий свое состояние в дискретные моменты времени. В каждый момент времени $t = \overline{0,\ N}$ поведение объекта полностью описывается n-вектором
$x(t) = (x_1(t),\ \ldots,\ x_n(t))$
Динамика объекта описывается следующим рекурентным уравнением:
$x(t+1) = f(x(t),\ u(t),\ t),\ x(0) = x_0,\ t = \overline{0,\ N-1}$
$u(t) = (u_1(t),\ \ldots,\ u_r(t))$ — управляющее воздействие в момент времени t, $u(t) \in U(t) \in \mathbb{R}^r$, $f(\cdot)$ — вектор функция определенная в области значений своих аргументов, x_0 — начальное состояние объекта управления.
При заданном состоянии x_0 поведение объекта полнсотью определяется управляющим воздействием $u(t)$. Качество управления оценим следующим

критерием качества (типа Больца)

$$\begin{cases} J(u) = \phi(x(N)) + \sum_{t=0}^{N-1} f_0(x(t), u(t), t) \to \min \\ \phi(x),\ f_0(x,\ u,\ t) \text{ — функция определенная в области значений своих аргументо} \\ x(t+1) = f(x(t),\ u(t),\ t),\ x(0) = x_0,\ t = \overline{0, N-1},\ u(t) \in U(t) \end{cases} \tag{3.16}$$

3 этапа:

1. Инвариантное погружение, построение функции Беллмана. Исходная задача погружается в некое семейство зависящее от параметров
2. Составление управления Беллмана
3. Решение уравнения Беллмана и всей задачи в целом

Любая задача характеризуется набором своих параметров, таких как число переменных, момента начала и конца процесса. При инвариантном погружении отвлекаются от заданного значения некоторых параметров и считают их переменными велечинами.
Для нашей задачи 3.16 в качестве таких параметров выберем:

1. Момент начала управления вместо $t = 0$ возьмем $\tau = \overline{0, N-1}$
2. Начальное состояние процесса вместо x_0 состояние $z \in \mathbb{R}^n$ в момент времени τ

Таким образом семейство, в которое погружается задача 3.16 будет иметь следующий вид:

$$\begin{cases} J(u(\tau, z)) = \phi(x(N)) + \sum_{t=\tau}^{N-1} f_0(x(t),\ u(t),\ t) \to \min \\ x(t+1) = f(x(t), u(t), t),\ x(\tau) = z \\ u(t) \in U(t),\ t = \overline{\tau, N-1} \end{cases} \tag{3.17}$$

3.16, $\tau = 0,\ z = x_0$ — исходня задача.
Пара (τ, z) — позиция многошагового процесса. Минимальное значение критерия качества в задачах 3.17, как функция позиции (τ, z)

$$B(\tau, z) = \min_u J(u(\tau,\ z)) \text{ — функция Беллмана}$$

3.3.1 Динамическое программирование для задачи ОУ в классе кусочно-непрерывных управлений

$$\begin{cases} J(u) = \phi(x(t_f)) + \int_{t_0}^{t_f} f_0(x(t), u(t), t)dt \to \min \\ \dot{x} = f(x,\ u,\ t),\ x(t_0) = x_0 \\ u(t) \in U,\ t \in T = [t_0,\ t_f] \end{cases} \tag{3.18}$$

$u(t)$ — кусочно-непрерывная функция
Погрузим задачу 3.18 в следующее семейство задач:

$$\begin{cases} J(u(\tau,\ z)) = \phi(x(t_f)) + \int_{\tau}^{t_f} f_0(x(t), u(t), t)dt \to \min \\ \dot{x} = f(x, u, t),\ x(\tau) = z \\ u(t) \in U,\ t \in T(\tau) = [\tau, t_f] \end{cases} \tag{3.19}$$

— зависящее от $\tau \in T$ и $z \in \mathbb{R}^n$
$B(\tau,\ z) = \min_u J(u \mid \tau,\ z),\ \tau \in T,\ z \in \mathbb{R}^n$ — функция Беллмана После вывода уравнения Белламана получим решение уравнения:

$$\frac{\partial B(\tau,\ z)}{\partial z} f(z,\ u^0(\tau,\ z)) + f_0(z,\ u^0(\tau,\ z),\ z) = \min_{v \in U}\left(\frac{\partial B(\tau,\ z)'}{\partial z} f(z,\ v,\ \tau) + f_0(z,\ v,\ \tau)\right),$$

ГЛАВА 4

ЧИСЛЕННЫЕ ЭКСПЕРИМЕНТЫ

В настоящей главе исследуется применения параметрического программирования в задаче синтеза оптимальной линейной системы и MPC

4.1 Применение параметрического программирования в MPC

Для реализации решения задачи используется MPT3 Toolbox для Matlab, описанный в главе выше.

Первоначально нужно задать все начальные значения, чтобы передать их в $mptconstructMatrices$. $MptconstructMatrices$ представляет собой функцию, на вход которой приходят параметры $probStruct$ и $sysStruct$. sysStruct будет состоять из матриц A, B, C, D, ограничений на x ($x_m in$ и $x_m ax$) и ограничений на u ($u_m in$ и $u_m ax$). $probStruct$ в свою очередь будет состоять из нормы (norm), числа N, также Q,P_n,R и тд.

В качестве примера рассмотрим задачу со следующим критерием качества, который нужно минимизировать:

$$J(u) = \int_0^{t_f} |u(t)| dt \to \min,$$

на траекториях:

$$\dot{x} = Ax + Bu$$

Матрица А будет иметь следующий вид:

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Начальным параметрам присвоим значения: $N = 20, t_0 = 0, t_f = 10$. Вектор B будет иметь вид: $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Чтобы отобразить полную структуру нашей задачи в Matlab поступим так, как показано ниже:

```
sys = ss(A, B, [], []);
```

```
sysd = c2d(sys,tf/N,'zoh');
Ad = sysd.A;
Bd = sysd.B;

sysStruct.A= Ad;
sysStruct.B= Bd;
sysStruct.C= [0 0];
sysStruct.D= 0;

sysStruct.xmin = [-10; -10];
sysStruct.xmax = [10; 10];

sysStruct.umin = -1;
sysStruct.umax = 1;

nx = size(A, 2);
probStruct.norm=inf;
probStruct.Q=[0 0; 0 0];
probStruct.P_N=[0 0; 0 0];
probStruct.R=1;
probStruct.N=N;
probStruct.subopt_lev=0;
H = [eye(nx); -eye(nx)];
K = eps*ones(nx*2,1);
probStruct.Tconstraint=2;
probStruct.Tset = polytope(H, K);
```

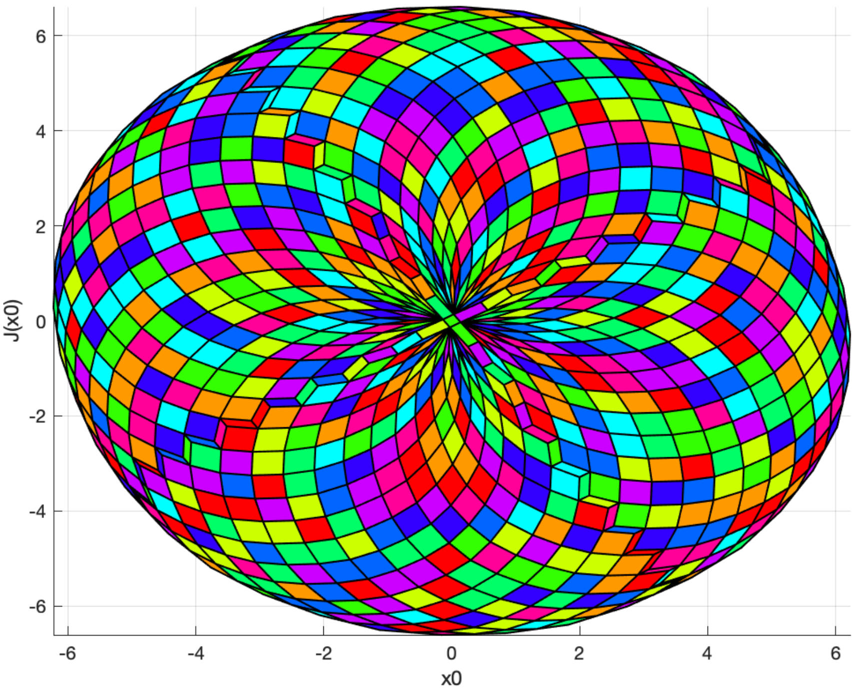
После этого мы будем иметь все начальные значения для нашей задачи сгенерированную структуру, которую можно решить, используя известные процедуры. Далее нужно на основании $mptconstructMatrices$ посмотреть класс Opt, который инкапсулирует информацию и решения для задач LP/QP/pLP/pQP/LCP/pLCP. Для конкретного момента времени высчитывается u на основании заданного x_τ и подается на систему. Затем на каждом отрезке из T пересчитвается заданное x_τ. Мы будем проедлывать описанную выше операцию в цикле на каждом шаге и на половине пути добавим возмущение w.

```
for tau = 0:NN-1
    probStruct.N=N;
```

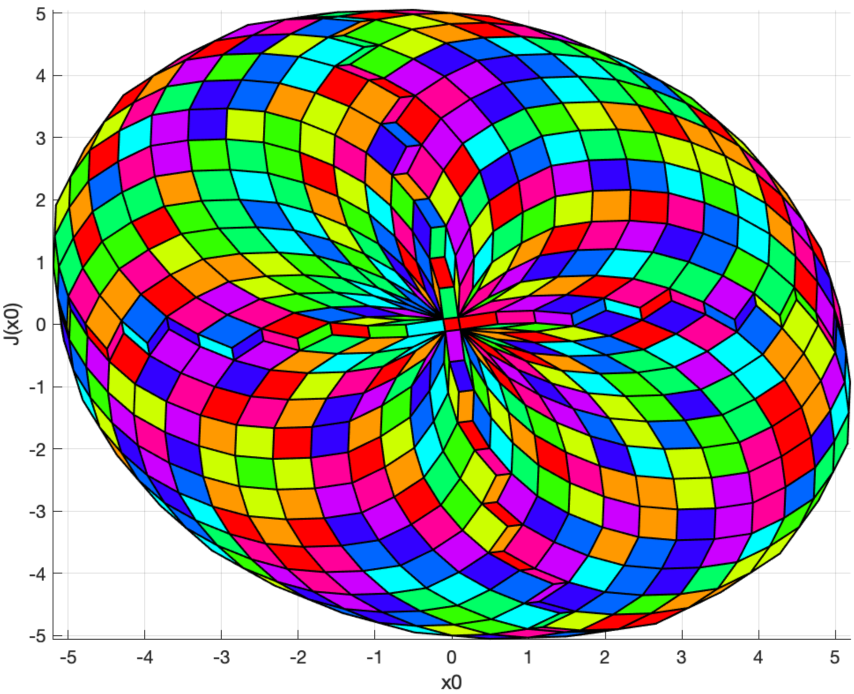
```
    Matrices = mpt_constructMatrices(sysStruct,probStruct);
    plp = Opt(Matrices);
    solution = plp.solve();
    u = solution.xopt.feval(xtau, 'primal');
    if N > NN/2
        xtau = Ad * xtau + Bd * u(1) + w;
    else
        xtau = Ad * xtau + Bd * u(1);
    end

    figure;
    solution.xopt.fplot('obj');
    xlabel('x0');
    ylabel('J(x0)');
    N = N-1;
    X = [X xtau];
    U = [U u(1)];
end
```

4.1.1 Результаты

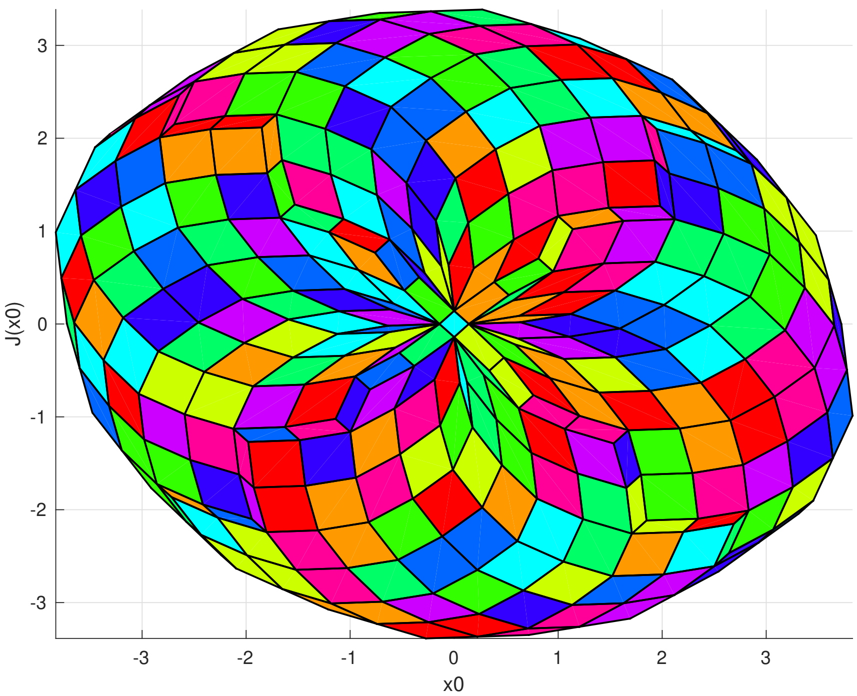
Далее проилюстриуем наглядно полученные результаты на каждом 5 шаге



N = 20

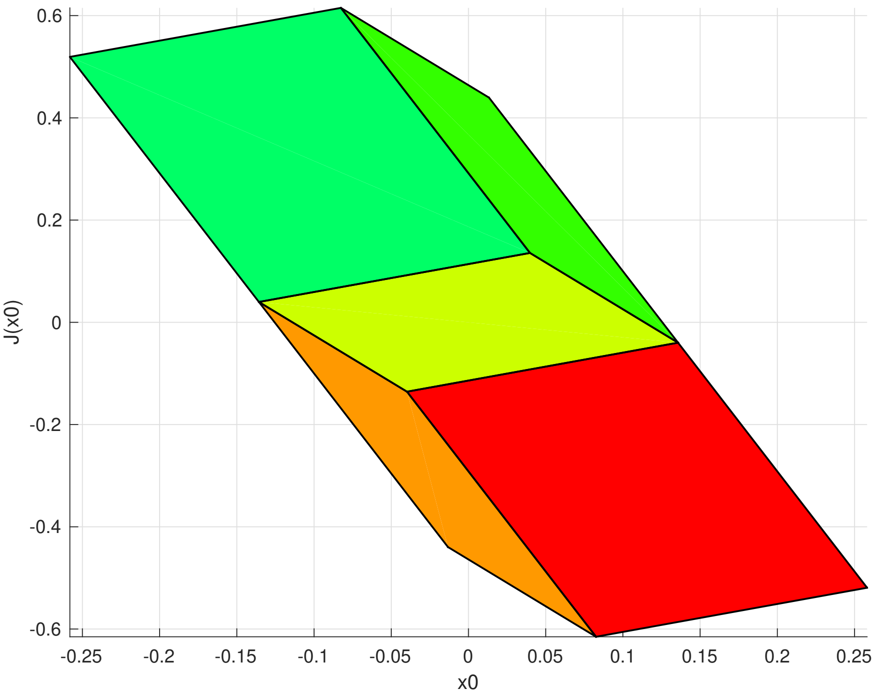


N = 15



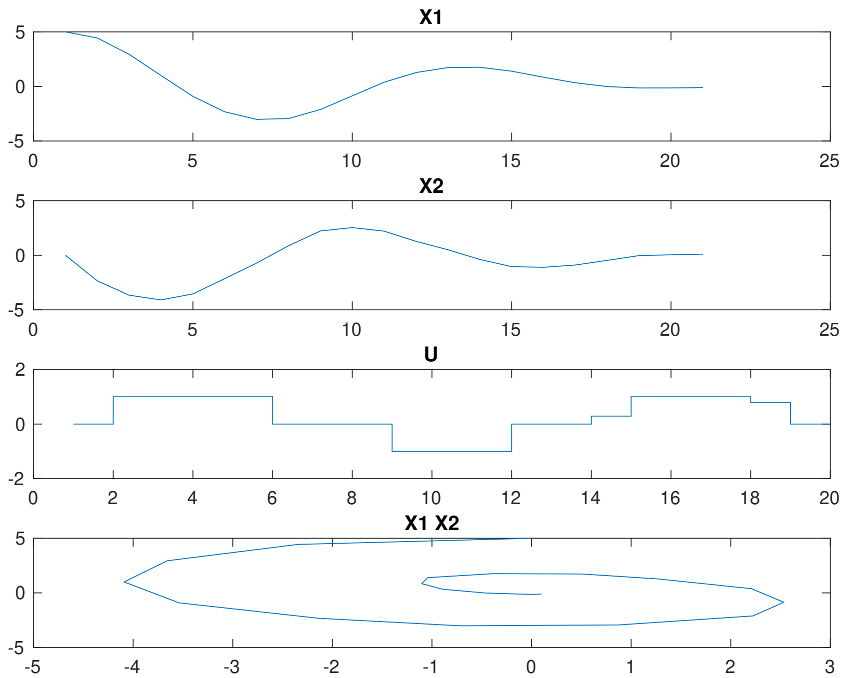
3
2
1
0
-1
-2
-3
J(x0)
-3
-2
-1
0
1
2
3
x0

N = 10



0.6
0.4
0.2
0
-0.2
-0.4
-0.6
J(x0)
-0.25
-0.2
-0.15
-0.1
-0.05
0
0.05
0.1
0.15
0.2
0.25
x0

N = 5 N = 1



X и U

4.2 Применение параметрического программирования в задаче синтеза оптимальной линейной системы

Ранее была рассмотрена теоретическая база по построению классической оптимальной обратной связи. В данной главе проилюстриуем, как параметрическое программирование применяется в задачах синтеза оптимальной линейной системы.

Рассмотрим следующую задачу:

$$\begin{aligned} &x_1(t_f) \to \min, \\ &x_2(t_f) = 0, \end{aligned}$$

на траекториях системы второго порядка

$$\begin{cases} \dot{x_1} = x_2, \\ \dot{x_2} = -x_1 + u, \end{cases} \quad x_1(t_0) = x_{10}, \quad x_2(t_0) = x_{20} \tag{4.1}$$

с ограничениями $-L \leq u \leq L,\ t \in T = [t_0,\ t_f]$

Матрица A имеет вид:

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Вектром b выглядит следующим образом:

$$b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Матрица H:

$$H = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Параметрам задачи присвоим значения: $t_0 = 0,\ t_f = 10$. Будем использовать дискретные управляющие воздействия с шагом $h = \frac{t_f - t_0}{100}$. Остальные параметры в программе зададим таким образом:

c_h, d_h на каждом шаге в matlab вычисляется процедурой integral в Matlab, которое высчитывается в цикле для каждого шага:

```
for i = 1:N
    ch(i) = integral(C,t0+h*(i-1),t0+h*i,'ArrayValued',true);
    dh(i) = integral(d,t0+h*(i-1),t0+h*i,'ArrayValued',true);
end
```

u_0, в свою очередь, вычисляется с помощью процедуры linprog:

```
u0 = linprog(-ch',[],[],dh,g0,-L*ones(N,1),L*ones(N,1))
```

Далее рассчитаем x:

```
x = zeros(2,N+1);
x(:,1) = x0;
for i = 1:N
    x(:,i+1) = F(h)*x(:,i)+integral(@(t)u0(i)*F(t0+(i+1)*h-t)*b,t0+i*h
end
x1 = zeros(2,N+1);
x1(:,1) = x0;
for i = 1:N
    x1(:,i+1) = F(h)*x1(:,i)+integral(@(t)u0(i)*F(t0+(i+1)*h-t)*b+w(t)
end
```

Затем на определенном промежутке времени добавим возмущение w и придет к ответу с помощью известных уже нам процедур:

```
    for j = 1:N
        j
        if j>90
```

```
        w = @(t)0*cos(3*t);
    end
    tau = t0 + (j-1)*h;
    g0 = g-H*F(tf-tau)*x2(:,j);
    u1 = linprog(-ch(j:N),[],[],dh(j:N),g0,-L*ones(N-j+1,1),L*ones
    u2(j) = u1(1);

    x2(:,j+1) = F(h)*x2(:,j)+integral(@(t)u2(j)*F(t0+(j+1)*h-t)*b+
    x(:,j+1)
    x2(:,j+1)
end
```

ЗАКЛЮЧЕНИЕ

В курсовой рассмотрена задача синтеза оптимальной системы второго порядка . Для исследуемой задачи предложены пути решения, включающие в себя использоваение MPT для Matlab. Проведен анализ методов параметрического программирования. Результаты проиллюстрированы на графиках, представленных выше. Также в работе были изучены основные результаты теории оптимального управления и подход к построению реализаций оптимальных обратных связей.

В дальнейшей перспективе возможна реализация управления по прогнозирующей модели, явные схемы MPC на основе параметрического линейного программирования.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 R. Bellman. The Dawn of Dynamic Programming. Dover Publications, reprint edition, 2003

2 Hazewinkel, Michiel. Pontryagin maximum principle. Encyclopedia of Mathematics, Springer Science+Business Media B.V. / Kluwer Academic Publishers, 2001.

3 W. M. Hogan. Point-to-set maps in mathematical programming. SIAM Review, 15(3):591–603, July 1973.

4 T. Gal and J. Nedoma. Multiparametric linear programming. Management Science, 18:406–442, 1972.

5 T. Gal. Postoptimal Analyses, Parametric Programming, and Related Topics. de Gruyter, Berlin, 2nd edition, 1995.

6 A.B. Berkelaar, K. Roos, and T. Terlaky. The optimal set and optimal partition approach to linear and quadratic programming. In T. Gal and H.J. Greenberg, editors, Advances in Sensitivity Analysis and Parametric Programming, volume 6 of International Series in Operations Research and Management Science, chapter 6. Kluwer Academic Publishers, 1997.

7 I. Adler and R.D.C. Monteiro. A geometric view of parametric linear programming. Algorithmica, 8(2):161–176, 1992.

8 C. Filippi. On the geometry of optimal partition sets in multiparametric linear programming. Technical Report 12, Department of Pure and Applied Mathematics, University of Padova, Italy, June 1997.

9 V. Dua and E.N. Pistikopoulos. An algorithm for the solution of multiparametric mixed integer linear programming problems. Annals of Operations Research, to appear.

10 A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. Automatica, 38(1):3–20, 2002.