

Лекция 1. Линейные матричные неравенства (Linear Matrix Inequalities — LMIs)

Общая характеристика

Метод линейных матричных неравенств (Linear Matrix Inequalities, LMI) — один из основных методов современной теории управления.

Используется в задачах устойчивости и стабилизации, качественной теории систем, робототехнике, обработке сигналов и многих других областях.

Существуют пакеты программ, эффективно реализующие этот метод в MATLAB и других средах моделирования.

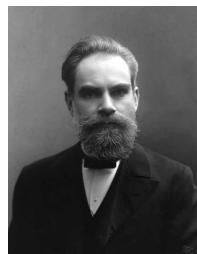
Задача считается решенной на теоретическом уровне, если она сведена к задаче исследования LMI.

Знакомство с LMI входит в необходимый минимум знаний специалиста по теории управления.

План изучения дисциплины

- Задачи, связанные с LMI. Примеры.
- Математическое обеспечение для исследования LMI. Знакомство с пакетами LMILAB, YALMIP, CVX, SOSTOOLS.
- Численные методы, лежащие в основе прикладных программ.
- Каждый студент получает индивидуальное задание: написать и отладить программы с использованием пакетов LMILAB, YALMIP, CVX.

История метода линейных матричных неравенств в теории управления



А. М. Ляпунов (1892), диссертация «Общая задача об устойчивости движения» — уравнение и неравенство Ляпунова.

А. И. Лурье (1951) — «разрешающие уравнения Лурье» (без аппарата теории матриц).

История метода линейных матричных неравенств в теории управления



В. А. Якубович (1962) — «метод матричных неравенств» в теории абсолютной устойчивости.

Е. С. Пятницкий (1980-е) — численное решение неравенств теории управления методом эллипсоидов.

История метода линейных матричных неравенств в теории управления



А. С. Немировский, Ю. Е. Нестеров (1990-е) — метод внутренней точки для решения LMIs.

Стивен Бойд (1990-е) — систематическое применение LMIs к задачам управления и других наук, первые программные пакеты.

Литература

Список литературы

- [1] Boyd S., El Ghaoui L., Feron E., Balakrishnan V. *Linear Matrix Inequalities in System and Control Theory*. Philadelphia: SIAM, 1994. <https://web.stanford.edu/~boyd/lmibook/>
- [2] Boyd S., Vandenberghe L. *Convex Optimization*. Cambridge University Press, 2004. <https://web.stanford.edu/~boyd/cvxbook/>
- [3] Чурилов А.Н., Гессен А.В. *Исследование линейных матричных неравенств. Путеводитель по программным пакетам*. СПб.: Изд-во СПбГУ, 2004.

Литература (приложения)

Список литературы

- [1] Поляк Б.Т., Щербаков П.С. *Робастная устойчивость и управление*. М.: Наука, 2002. <http://www.twirpx.com/files/special/tau/>
- [2] Баландин Д.В., Коган М.М. *Синтез законов управления на основе линейных матричных неравенств*. М.: Физматлит, 2007.
- [3] Поляк Б.Т., Хлебников М.В., Щербаков П.С. *Управление линейными системами при внешних возмущениях*. М.: URSS, 2014.

Программные пакеты

Список литературы

- [1] *MATLAB Robust Control Toolbox (раздел LMILAB)*. Документация: <https://se.mathworks.com/help/robust/linear-matrix-inequalities.html>
- [2] *YALMIP*. Пакет и документация: <https://yalmip.github.io/>
- [3] *SeDuMi*. Пакет: <http://sedumi.ie.lehigh.edu/>
- [4] *CVX*. Пакет и документация: <http://cvxr.com/>
- [5] *SOSTOOLS*. Пакет и документация: <http://www.cds.caltech.edu/sostools/>

Обозначения

Пусть $P = P^\top$, $Q = Q^\top$ — $n \times n$ -матрицы

$$\begin{aligned} P \geq 0 &\iff u^\top P u \geq 0, \quad \forall u \in \mathbb{R}^n \\ P > 0 &\iff u^\top P u > 0, \quad \forall u \in \mathbb{R}^n, u \neq 0 \\ P \leq 0 &\iff (-P) \geq 0 \\ P < 0 &\iff (-P) > 0 \\ P \leq Q &\iff P - Q \leq 0 \\ P < Q &\iff P - Q < 0 \end{aligned}$$

Простой пример

Пусть F — переменная матрица размера 2×2 , $F^\top = F$.

Параметризуем F :

$$F = \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix},$$

где x_1, x_2, x_3 — **скалярные** переменные.

Запишем F в виде

$$F = x_1 F_1 + x_2 F_2 + x_3 F_3,$$

где F_1, F_2, F_3 — постоянные коэффициенты:

$$F_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad F_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad F_3 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

Матричная переменная F свелась к трем скалярным переменным x_1, x_2, x_3

Простой пример (продолжение)

Неравенство $F > 0$ (положительная определенность) эквивалентно неравенству

$$x_1F_1 + x_2F_2 + x_3F_3 > 0$$

Неравенство $F > \varepsilon I$, где ε — постоянный скаляр, I — единичная матрица, эквивалентна неравенству:

$$F_0 + x_1F_1 + x_2F_2 + x_3F_3 > 0,$$

где $F_0 = -\varepsilon I$.

В более сложных неравенствах матрицы F_0, F_1, \dots имеют более сложную структуру.

Стандартная форма LMIs: общий вид

Пусть F_0, \dots, F_m — вещественные $n \times n$ -матрицы, $F_j = F_j^\top, j = 0, \dots, m$. Пусть

$$x = [x_1, \dots, x_m]^\top \in \mathbb{R}^m$$

Рассмотрим отображение $F : \mathbb{R}^m \rightarrow \mathbb{R}^{n \times n}$

$$F(x) = F_0 + x_1F_1 + \dots + x_mF_m$$

Очевидно $F(x)^\top = F(x)$.

Линейные матричные неравенства (LMI): строгие

$$F(x) > 0, \quad F(x) < 0$$

и нестрогие

$$F(x) \geq 0, \quad F(x) \leq 0$$

Все неравенства означают определенность квадратичных форм!

Стандартная форма LMIs: линейность и аффинность

$$F(x) = F_0 + L(x), \quad L(x) = \sum_{j=1}^m x_j F_j$$

Отображение $L(x)$ — линейное, $F(x)$ — аффинное. [Линейное отображение](#):

$$L(\alpha x + \beta y) = \alpha L(x) + \beta L(y), \quad \forall x, y \in \mathbb{R}^m, \forall \alpha, \beta \in \mathbb{R}$$

[Аффинное отображение](#):

$$F(\alpha x + (1 - \alpha)y) = \alpha F(x) + (1 - \alpha)F(y), \quad \forall x, y \in \mathbb{R}^m, \forall \alpha \in \mathbb{R}$$

Таким образом, «линейные матричные неравенства», строго говоря, — аффинные.

Стандартная форма LMIs: выпуклость

Здесь x_1, \dots, x_m — скалярные переменные. Множества:

$$\Omega^+ = \{ x \in \mathbb{R}^m : F(x) \geq 0 \}, \quad \Omega^- = \{ x \in \mathbb{R}^m : F(x) \leq 0 \}$$

Докажем, что Ω^- выпукло. Для этого убедимся:

$$x \in \Omega^-, \quad y \in \Omega^-, \quad 0 \leq \alpha \leq 1 \implies \alpha x + (1 - \alpha)y \in \Omega^-$$

т.е. Ω^- вместе с парой векторов x, y содержит все их выпуклые комбинации.

Из свойства аффинности

$$F(\alpha x + (1 - \alpha)y) = \alpha F(x) + (1 - \alpha)F(y) \leq 0$$

при $F(x) \leq 0, \quad F(y) \leq 0, \quad 0 \leq \alpha \leq 1.$

Отсюда Ω^- — выпукло, Ω^+ — аналогично.

Стандартная форма LMIs: случай нескольких неравенств

Система неравенств

$$F^{(1)}(x) > 0, \quad , F^{(2)}(x) > 0, \quad , \dots, \quad F^{(p)}(x) > 0$$

эквивалентна одному неравенству

$$\begin{bmatrix} F^{(1)}(x) \\ & \ddots \\ & & F^{(p)}(x) \end{bmatrix} > 0.$$

Иногда рассматривают дополнительные ограничения типа равенств:

$$F(x) > 0, \quad Ax = b$$

В последнем случае часть координат вектора x выражают через остальные координаты.

Основные задачи, связанные с LMIs

1-й тип задач: задача разрешимости (feasibility problem, FEASP)
feasible — допустимый

Задача FEASP

Существует ли вектор x , такой что $F(x) > 0$ ($F(x) \geq 0$) ? Если да, найти хотя бы один такой x .

Обычно численные алгоритмы итерационные, они не дают описание всех решений.

Основные задачи, связанные с LMIs

2-й тип задач: задача полуопределенного программирования (semi-definite programming, SDP)

Задача SDP

$$c^\top x \rightarrow \min, \quad F(x) \geq 0.$$

Варианты: $F(x) > 0$, дополнительно $Ax = b$.

Здесь $c \in \mathbb{R}^m$, $A \in \mathbb{R}^{k \times m}$, $b \in \mathbb{R}^k$

Целевая функция линейна, $F(x)$ — выпукла, т.е. это частный случай задачи [выпуклого программирования](#), обобщение [линейного программирования](#).

Скалярные и матричные переменные

Стандартная форма использует [скалярные переменные](#) x_1, \dots, x_m (decision variables).

В задачах теории управления обычно используют [матричные переменные](#) (matrix variables).

Простейшая ситуация: неравенство вида

$$L(H) < F_0,$$

где $L(H)$ — линейный оператор, матрицы F_0 , $L(H)$ — симметричны. H — неизвестная матричная переменная размера $n \times n$.

Если H — симметричная матрица, то она содержит $m = n(n + 1)/2$ неизвестных скалярных параметров.

Если H — произвольная матрица, то она содержит $m = n^2$ неизвестных скалярных параметров.

Если H — диагональная матрица, то она содержит $m = n$ неизвестных скалярных параметров.

LMIs — общая схема работы

Этапы:

1. Формулируем задачу управления в виде LMIs с матричными переменными.
2. Преобразуем задачу к стандартной форме (со скалярными переменными).
3. Решаем задачу в стандартной форме с помощью численных методов.
4. Преобразуем полученное решение к матричным переменным.

Этап 1 — человек-исследователь.

Этап 3 — программа-решатель (solver).

Этапы 2, 4 — интерфейсная программа (interface, parser, front-end).

BMIs — билинейные матричные неравенства

Bilinear Matrix Inequality: $F(x, y) > 0$, где

$$F(x, y) = F_0 + \sum_{i=1}^m x_i F_i + \sum_{j=1}^n y_j G_j + \sum_{i=1}^m \sum_{j=1}^n x_i y_j R_{ij}$$

Здесь $F(x, \cdot)$ линейна (аффинна) по x , $F(\cdot, y)$ линейна по y , но $F(x, y)$ нелинейна (квадратична) по совокупности переменных x, y .

В отличие от LMIs, алгоритмы решения задач BMIs имеют сложность выше полиномиальной (NP-hard). Практически их можно решать для задач малой размерности (имеются готовые программы).

Лекция 2. Примеры LMIs. Линейные системы

Неравенство Ляпунова

$$\frac{dx}{dt} = Ax, \quad (1)$$

$x(t)$ — n -вектор, A — $n \times n$, $A = \text{const}$, λ_j — собств. числа A

Эквивалентные утверждения:

- Система (1) глобально асимптотически устойчива.
- Матрица A гурвицева, т.е. $\text{Re } \lambda_j < 0$, $j = 1, \dots, n$.
- Существует функция Ляпунова вида $V(x) = x^T H x$, $H > 0$, такая, что при достаточно малом ε

$$\frac{dV}{dt} = x^T (HA + A^T H)x \leq -\varepsilon \|x\|^2.$$

Получаем LMIs (задача разрешимости): $\exists H, H = H^T$,

$$H > 0, \quad HA + A^T H < 0.$$

Неравенство Ляпунова — сведение к одному неравенству

Неравенство $HA + A^T H < 0$ называется [неравенством Ляпунова](#).

Систему неравенств

$$H > 0, \quad HA + A^T H < 0.$$

можно эквивалентно записать в виде одного неравенства с блочной матрицей:

$$\begin{bmatrix} HA + A^T H & 0 \\ 0 & -H \end{bmatrix} < 0$$

Эта система неравенств обычно имеет бесконечное множество решений.
Численный алгоритм находит одно из них.

Неравенство Ляпунова. Стандартная форма

$$n = 2, \quad A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad H = \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Неравенство $H > 0$:

$$x_1 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + x_3 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} > 0$$

Неравенство Ляпунова $H(-A) + (-A)^T H > 0$:

$$x_1 \begin{bmatrix} -2a_{11} & -a_{12} \\ -a_{12} & 0 \end{bmatrix} + x_2 \begin{bmatrix} -2a_{21} & -a_{11} - a_{22} \\ -a_{11} - a_{22} & -2a_{12} \end{bmatrix} + x_3 \begin{bmatrix} 0 & -a_{21} \\ -a_{21} & -2a_{22} \end{bmatrix} > 0$$

Пример. Неравенство Ляпунова для положительных систем

Положительные системы ДУ (биология, химия, экономика) — координаты решения всегда неотрицательны.

$$\frac{dx}{dt} = Ax \quad (1)$$

Система (1) наз. **положительной** (positive), если

$$x(0) \geq 0 \implies x(t) \geq 0, \forall t \geq 0$$

Когда линейная система положительна?

Матрица Мецлера: $A_{ij} \geq 0, \forall i \neq j$.

Утверждение

Система (1) положительна $\iff A$ — матрица Мецлера

Неравенство Ляпунова для положительных систем (продолжение)

Утверждение

Пусть A — матрица Мецлера. Тогда A — гурвицева тогда и только тогда, когда $\exists H$, H — **диагональная** и

$$H > 0, \quad HA + A^T H < 0.$$

Т.е. здесь число скалярных параметров не $m = n(n+1)/2$, как в общем случае, а $m = n$.

Дискретное неравенство Ляпунова

$$x(k+1) = Ax(k), \quad k = 0, 1, 2, \dots \quad (1)$$

$x(k)$ — n -вектор, A — $n \times n$ -матрица

Эквивалентные утверждения:

- Система (1) глобально асимптотически устойчива.
- Матрица A **устойчива по Шуру**, т.е. $|\lambda_j(A)| < 1, j = 1, \dots, n$.

- Существует функция Ляпунова вида $V(x) = x^T H x$, $H > 0$, такая что при малых ε

$$V(x(k+1)) - V(x(k)) \leq -\varepsilon \|x(k)\|^2.$$

Так как $V(x(k+1)) = x(k)^T A^T H A x(k)$, получаем LMIs:

$$H > 0, \quad A^T H A - H < 0.$$

Второе неравенство — [дискретное неравенство Ляпунова](#).

Линейная система: устойчивость с заданной скоростью затухания

Вновь рассмотрим

$$\frac{dx}{dt} = Ax, \quad (1)$$

$x(t)$ — n -вектор, $A = n \times n$, $A = \text{const}$.

Система (1) [устойчива со скоростью затухания \$\alpha > 0\$](#) , если существует $C = \text{const}$ такая, что

$$\|x(t)\| \leq C e^{-\alpha t} \|x(0)\|, \quad \forall t \geq 0. \quad (2)$$

По-английски α — decay rate.

Докажем, что для (2) достаточно существования функции Ляпунова $V(x) = x^T H x$, $H > 0$, такой, что

$$\dot{V}(x) + 2\alpha V(x) \leq 0, \quad \forall t \geq 0.$$

Устойчивость с заданной скоростью затухания: доказательство

Умножим неравенство

$$\dot{V}(x) + 2\alpha V(x) \leq 0 \quad (3)$$

слева и справа на $e^{2\alpha t}$:

$$e^{2\alpha t} \dot{V}(x) + 2\alpha e^{2\alpha t} V(x) \leq 0.$$

Эквивалентно:

$$\frac{d}{dt} [e^{2\alpha t} V(x(t))] = e^{2\alpha t} \dot{V}(x) + \frac{d}{dt} (e^{2\alpha t}) V(x) \leq 0.$$

Отсюда

$$e^{2\alpha t} V(x(t)) \leq V(x(0)), \quad \forall t \geq 0.$$

Так как $\lambda_{\min} \|x\|^2 \leq x^T H x \leq \lambda_{\max} \|x\|^2$, то

$$\|x(t)\| \leq C e^{-\alpha t} \|x(0)\|, \quad C = \sqrt{\lambda_{\max}/\lambda_{\min}}.$$

Устойчивость с заданной скоростью затухания: LMIs

Неравенство

$$\dot{V}(x) + 2\alpha V(x) \leq 0 \quad (3)$$

сводится к LMIs:

$$H > 0, \quad HA + A^T H + 2\alpha H \leq 0.$$

Здесь α — не переменная, а заданное число.

Эквивалентные неравенства:

$$H > 0, \quad H(A + \alpha I) + (A + \alpha I)^T H \leq 0, \quad (4)$$

где I — единичная матрица.

Замечание: Если α — не константа, а тоже переменная, то (4) будет уже не LMI, а BMI (билинейное по H, α).

Пример задачи BMI: $\alpha \rightarrow \max$ при условии (4).

Устойчивость дискретной системы с заданной скоростью затухания

$$x(k+1) = Ax(k), \quad k = 0, 1, 2, \dots$$

Система устойчива со скоростью затухания q , $0 < q < 1$, если существует $C = \text{const}$ такая, что

$$\|x(k)\| \leq Cq^k \|x(0)\|, \quad \forall k \geq 0.$$

При $V(x) = x^T H x$ задача сводится к неравенству

$$V(x(k+1)) \leq qV(x(k)), \quad \forall k \geq 0.$$

Получаем систему LMIs:

$$H > 0, \quad A^T H A - qH \leq 0.$$

Лекция 2а. Примеры LMIs. Задачи абсолютной устойчивости

S-процедура

Это основной прием, который используется в абсолютной устойчивости.

Рассмотрим функции $G(u), F_1(u), \dots, F_m(u)$, которые действуют $\mathbb{R}^n \rightarrow \mathbb{R}$.

Рассмотрим **условное** неравенство

$$G(u) \geq 0, \quad \forall u \in \Omega = \{v \in \mathbb{R}^n : F_i(v) \geq 0, i = 1, \dots, m\}$$

Достаточное условие выполнения неравенства (*S*-процедура):

$$\exists \tau_1 \geq 0, \dots, \exists \tau_m \geq 0 : G(u) - \sum_{i=1}^m \tau_i F_i(u) \geq 0, \quad \forall u \in \mathbb{R}^n$$

Эквивалентные ли эти две задачи? Вообще говоря, нет. Если эквивалентны, то *S*-процедура называется **неущербной**. Если *S*-процедура ущербна, ее все равно можно использовать (может быть, с потерей части решений).

Круговой критерий абсолютной устойчивости

$$\begin{aligned} \frac{dx}{dt} &= Ax + b\varphi(\sigma, t), & \sigma &= c^T x \\ \mu_1 &\leq \varphi(\sigma, t)/\sigma \leq \mu_2 \end{aligned}$$

Исследование задачи абсолютной устойчивости сводится к существованию функции Ляпунова вида

$$V(x) = x^T H x, \quad H > 0,$$

для которой выполнено (*S*-процедура, $m = 1$, $\tau > 0$ — переменная)

$$\dot{V}(x(t)) + \tau(\mu_2\sigma - \varphi)(\varphi - \mu_1\sigma) \leq -\varepsilon(\|x\|^2 + \varphi^2), \quad \forall t \geq 0$$

Последнее условие сводится к неравенству

$$2x^T H(Ax + bu) + \tau(\mu_2 c^T x - u)(u - \mu_1 c^T x) \leq -\varepsilon(\|x\|^2 + u^2), \quad \forall x, u$$

Круговой критерий абсолютной устойчивости (продолжение)

Переписываем неравенство:

$$\begin{aligned} \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + \\ + \tau \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} -\mu_1\mu_2 cc^T & \frac{1}{2}(\mu_1 + \mu_2)c \\ \frac{1}{2}(\mu_1 + \mu_2)c^T & -1 \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq -\varepsilon \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} x \\ u \end{bmatrix} \end{aligned}$$

Можно уменьшить число переменных: разделим неравенство на τ , в качестве новой H возьмем H/τ .

Получаем LMIs (ε можно взять сколь угодно малым): $\exists H = H^T$,

$$\begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} < \begin{bmatrix} \mu_1 \mu_2 c c^T & -\frac{1}{2}(\mu_1 + \mu_2)c \\ -\frac{1}{2}(\mu_1 + \mu_2)c^T & 1 \end{bmatrix}, \quad H > 0$$

Круговой критерий абсолютной устойчивости для дискретных систем

$$x(k+1) = Ax(k) + b\varphi(\sigma(k), k), \quad \sigma(k) = c^T x(k)$$

$$\mu_1 \leq \varphi(\sigma, k)/\sigma \leq \mu_2$$

Исследование задачи абсолютной устойчивости сводится к существованию функции Ляпунова вида

$$V(x) = x^T H x, \quad H > 0,$$

для которой должно быть $V(x(k+1)) < V(x(k))$, $\forall n \geq 0$. Получаем неравенство:

$$(Ax + bu)^T H (Ax + bu) - x^T H x +$$

$$+ (\mu_2 c^T x - u)(u - \mu_1 c^T x) \leq -\varepsilon(\|x\|^2 + u^2), \quad \forall x, u$$

Круговой критерий абсолютной устойчивости для дискретных систем (продолжение)

Получаем LMIs:

$$\begin{bmatrix} A^T H A - H & A^T H b \\ b^T H A & b^T H b \end{bmatrix} < \begin{bmatrix} \mu_1 \mu_2 c c^T & -\frac{1}{2}(\mu_1 + \mu_2)c \\ -\frac{1}{2}(\mu_1 + \mu_2)c^T & 1 \end{bmatrix}, \quad H > 0$$

Левая часть первого неравенства по-прежнему [линейна](#) относительно H , правая часть — постоянная матрица.

Критерий Попова абсолютной устойчивости

Стационарная система:

$$\frac{dx}{dt} = Ax + b\varphi(\sigma), \quad \sigma = c^T x$$

$$0 \leq \varphi(\sigma)/\sigma \leq \mu$$

Исследование задачи абсолютной устойчивости сводится к существованию функции Ляпунова типа Лурье—Постникова:

$$V(x) = x^T H x + \theta \int_0^{c^T x} \varphi(\sigma) d\sigma, \quad H > 0, \quad \theta \in \mathbb{R},$$

для которой $\dot{V}(x(t)) < 0$ при $t \geq 0$ и

$$2x^T H (Ax + bu) + \theta u c^T (Ax + bu) + (\mu c^T x - u)u \leq -\varepsilon(\|x\|^2 + u^2), \quad \forall x, u$$

Критерий Попова абсолютной устойчивости (продолжение)

Получаем LMIs:

$$\begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} + \begin{bmatrix} 0 & \frac{1}{2}\theta A^T c \\ \frac{1}{2}\theta c^T A & \theta c^T b \end{bmatrix} < \\ < \begin{bmatrix} 0 & -\frac{1}{2}\mu c \\ -\frac{1}{2}\mu c^T & 1 \end{bmatrix}, \quad H > 0.$$

Здесь неизвестные параметры: H, θ . Левая часть [линейна](#) относительно H, θ (в совокупности).

Лекция 2b. Примеры LMIs. Специальные преобразования

Цель преобразований

Требуется получить систему линейных матричных неравенств или задачу SDP из неравенств других видов.

Чаще всего используют такие приемы как:

- S-процедура
- Дополнения Шура
- Замена переменных
- Добавление вспомогательных переменных (в основном для задачи SDP)

Дополнение Шура

По-английски — Schur complement.

Теорема. Пусть задана симметричная блочная матрица

$$M = \begin{bmatrix} Q & P \\ P^T & R \end{bmatrix}.$$

Тогда условие $M > 0$ эквивалентно каждой из двух систем неравенств:

$$Q - PR^{-1}P^T > 0, \quad R > 0$$

или

$$R - P^TQ^{-1}P > 0, \quad Q > 0.$$

Алгебраическое неравенство Риккати

Квадратичное матричное неравенство относительно переменной $H = H^T$:

$$Q + HP + P^T H + HCR^{-1}C^T H < 0$$

Коэффициенты $Q = Q^T$, $R = R^T > 0$.

По формуле дополнения Шура это эквивалентно LMI (относительно H):

$$\begin{bmatrix} Q + HP + P^T H & HC \\ C^T H & -R \end{bmatrix} < 0.$$

Таким образом, квадратичное неравенство сводится к линейному.

Пример

Пусть C, D — заданные матрицы, $D = D^T$, H — неизвестная симметричная матрица. В силу дополнений Шура система неравенств

$$C^T H^{-1} C < D, \quad H > 0 \tag{1}$$

эквивалентно LMI относительно H :

$$\begin{bmatrix} H & C \\ C^T & D \end{bmatrix} > 0.$$

Неравенства (1) часто встречаются в составе более сложных условий.

Оценка спектральной нормы

Пусть $M - m \times n$ -матрица. Спектральная норма M :

$$\|M\|_2 = \sup_{x \neq 0} \frac{\|Mx\|}{\|x\|}, \quad \|\cdot\| - \text{векторная евклидова норма}$$

Пусть α — скаляр. Тогда

$$\|M\|_2 \leq \alpha \iff \|Mx\| \leq \alpha \|x\|, \forall x.$$

Эквивалентно, т.к. $\|x\| = \sqrt{x^T x}$, то

$$x^T M^T M x \leq \alpha^2 x^T x, \forall x \iff M^T M \leq \alpha^2 I.$$

Из дополнения Шура

$$\begin{bmatrix} \alpha^2 I & M^T \\ M & I \end{bmatrix} \geq 0.$$

Оценка спектральной нормы (продолжение)

Мы доказали: неравенство $\|M\|_2 \leq \alpha$ эквивалентно неравенству

$$\begin{bmatrix} \alpha^2 I & M^T \\ M & I \end{bmatrix} \geq 0. \quad (1)$$

Рассмотрим две задачи:

- Пусть α — заданное число, M — переменная матрица. Тогда (1) — LMI относительно M , которое задает условие $\|M\|_2 \leq \alpha$. Как правило, это условие используется в составе более сложной задачи FEASP или SDP.
- Пусть матрица M задана и требуется вычислить $\|M\|_2$. Введем новую переменную $t = \alpha^2$. Решим задачу SDP относительно неизвестной t :

$$t \rightarrow \inf, \quad tI \geq M^T M. \quad (2)$$

Тогда $\|M\|_2 = \sqrt{t}$, где t — решение задачи (2).

Новая переменная t нужна, чтобы обеспечить линейность.

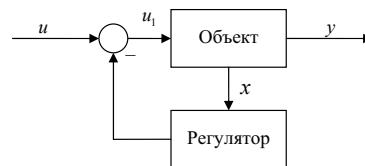
Стабилизация линейной системы обратной связью по состоянию

Задача

Дана линейная стационарная система

$$\frac{dx}{dt} = Ax + bu_1, \quad A \in \mathbb{R}^{n \times n}, \quad b \in \mathbb{R}^{n \times m}$$

Найти регулятор $u_1 = u - Kx$, $K \in \mathbb{R}^{m \times n}$, такой, что замкнутая система устойчива.



Стабилизация линейной системы обратной связью по состоянию (продолжение)

Уравнение замкнутой системы:

$$\frac{dx}{dt} = (A - bK)x + bu$$

Требуется: найти матрицу K , чтобы $A - bK$ была гурвицева.

Гурвицевость эквивалентна неравенству Ляпунова. Получаем систему относительно неизвестных H, K :

$$\exists H = H^T > 0, \quad \exists K : \quad H(A - bK) + (A - bK)^T H < 0.$$

Это BMI — билинейные матричные неравенства.

Преобразуем BMI к LMI. Умножим слева и справа на H^{-1} :

$$(A - bK)H^{-1} + H^{-1}(A - bK)^T < 0, \quad H^{-1} > 0$$

Стабилизация линейной системы обратной связью по состоянию (окончание)

$$AH^{-1} - bKH^{-1} + H^{-1}A^T - H^{-1}K^Tb^T < 0, \quad H^{-1} > 0$$

Сделаем замену переменных: $P = H^{-1}, Q = KH^{-1}$.

Получаем LMI:

$$AP + PA^T - bQ - Q^Tb^T < 0, \quad P > 0.$$

Если найти P, Q , то $K = QP^{-1}$.

Если нужен запас устойчивости: α — заданная константа, $\max_j \operatorname{Re} \lambda_j(A - bK) < -\alpha$. Получаем неравенства:

$$H(A - bK) + (A - bK)^T H + 2\alpha H < 0, \quad H > 0$$

Сводятся к LMI тем же приемом.

Лекция 3. Примеры LMIs. Разные задачи

Оценка эллипсоида

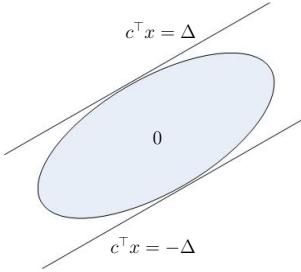
Даны эллипсоид (H — переменная)

$$\Omega = \{x \in \mathbb{R}^n : x^T H x \leq 1\}, \quad H = H^T > 0$$

и полоса, ограниченная гиперплоскостями (c, Δ — параметры)

$$\Pi = \{x \in \mathbb{R}^n : |c^T x| < \Delta\}, \quad c \in \mathbb{R}^n, \quad \Delta \in \mathbb{R}, \quad \Delta > 0.$$

Найти условия, линейные относительно H , при которых эллипсоид лежит в полосе ($\Omega \subset \Pi$).



Оценка эллипсоида (продолжение)

По-другому, должно быть выполнено:

$$x \in \Omega = \{x \in \mathbb{R}^n : x^T H x \leq 1\} \implies -\Delta < c^T x < \Delta$$

Получаем условие:

$$\max_{x \in \Omega} |c^T x| < \Delta \iff \max_{x \in \Omega} (c^T x)^2 < \Delta^2$$

Рассмотрим задачу [условной оптимизации](#):

$$(c^T x)^2 \rightarrow \max, \quad x^T H x \leq 1.$$

Все функции выпуклые, поэтому необходимые и достаточные условия получаем с помощью [функции Лагранжа](#):

$$L(x, \lambda) = (c^T x)^2 + \lambda(1 - x^T H x),$$

где $\lambda \geq 0$ — множитель Лагранжа. Ее частные производные по x , λ должны быть равны нулю.

Оценка эллипсоида (продолжение)

Рассмотрим функцию Лагранжа

$$L(x, \lambda) = (c^T x)^2 + \lambda(1 - x^T H x),$$

Условия экстремума:

$$\nabla_x L(x, \lambda) = 0, \quad L'_\lambda = 0.$$

Градиент по x :

$$\nabla_x L(x, \lambda) = \nabla_x [(c^T x)^2 - \lambda(x^T H x - 1)] = 2c^T x c - 2\lambda H x,$$

Получаем:

$$\begin{cases} L'_\lambda = 0, \\ \nabla_x L(x, \lambda) = 0, \end{cases} \iff \begin{cases} x^T H x = 1, \\ \lambda H x = c^T x c, \end{cases} \implies \lambda = (c^T x)^2$$

Наша задача — найти значение $c^T x$. Получаем:

$$\begin{cases} \lambda = (c^T x)^2, \\ \lambda H x = c^T x c, \end{cases} \implies x = \frac{1}{c^T x} H^{-1} c.$$

Оценка эллипсоида (продолжение)

$$\begin{cases} x = \frac{1}{c^T x} H^{-1} c, \\ x^T H x = 1, \end{cases} \implies (c^T x)^2 = c^T H^{-1} c.$$

В итоге:

$$\max_{x \in \Omega} (c^T x)^2 = c^T H^{-1} c.$$

Тогда

$$\Omega \subset \Pi \iff c^T H^{-1} c < \Delta^2.$$

Мы пришли к системе неравенств (нелинейных относительно H):

$$c^T H^{-1} c < \Delta^2, \quad H > 0$$

Сведем их к линейным, используя дополнение Шура:

$$\begin{bmatrix} H & c \\ c^T & \Delta^2 \end{bmatrix} > 0, \quad H > 0.$$

Оценка эллипсоида (второй способ)

Нам нужно, чтобы выполнялось:

$$x^T H x \leq 1 \implies \Delta^2 > (c^T x)^2$$

Применим S -процедуру:

$$\exists \tau > 0 : \Delta^2 - (c^T x)^2 - \tau(1 - x^T H x) > 0, \forall x$$

Эквивалентно:

$$\Delta^2 - \tau + x^T (\tau H - c c^T) x > 0, \forall x$$

Потребуем:

$$\Delta^2 - \tau > 0, \quad \tau H - c c^T \geq 0.$$

Возьмем $\tau = \Delta^2 - \varepsilon$, ε — малое. Потребуем:

$$\Delta^2 H - c c^T > 0 \iff H - c \Delta^{-2} c^T > 0.$$

Используем дополнение Шура:

$$\begin{bmatrix} H & c \\ c^T & \Delta^2 \end{bmatrix} > 0.$$

Еще один вид задач с LMI

Задача нахождения наибольшего определителя (determinant maximization, MAXDET)

$$\begin{aligned} c^T x + \ln \det G(x)^{-1} &\rightarrow \min, \\ F(x) \geq 0, \quad G(x) &> 0 \end{aligned}$$

Здесь $F(x)$, $G(x)$ — аффинные функции

$$\ln \det G(x)^{-1} = \ln(1/\det G(x)) = -\ln \det G(x)$$

Тогда

$$\ln \det G(x)^{-1} \rightarrow \min \iff \ln \det G(x) \rightarrow \max$$

Можно показать (см. Boyd), что функция $\ln \det G(x)^{-1}$ выпукла.

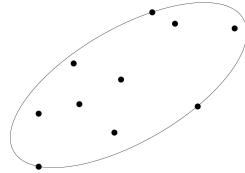
Пример. Нахождение эллипсоида наименьшего объема (S. Boyd)

Задача

Найти эллипсоид вида

$$\Omega = \{x : x^T H x \leq 1\}, \quad H > 0,$$

наименьшего объема, содержащий заданный набор векторов $s_i \in \mathbb{R}^n$, $i = 1, \dots, m$.



Очевидно

$$s_i \in \Omega \iff s_i^T H s_i \leq 1$$

Нахождение эллипсоида наименьшего объема (продолжение)

Объем шара в \mathbb{R}^n :

$$\begin{aligned} n = 1 &\implies V_1(R) = 2R, \\ n = 2 &\implies V_2(R) = \pi R^2, \\ n \geq 3 &\implies V_n(R) = \frac{2\pi R^2}{n} V_{n-2}(R) \end{aligned}$$

Объем эллипсоида в \mathbb{R}^n :

$$\text{Vol}(\Omega) = V_n(1) \sqrt{\det H^{-1}}$$

Получаем задачу MAXDET:

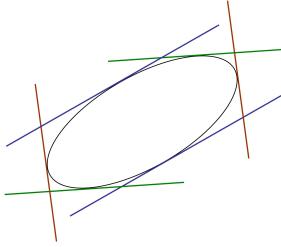
$$\ln \det H^{-1} \rightarrow \min, \quad s_i^T H s_i \leq 1, \quad i = 1, \dots, m, \quad H > 0$$

Пример. Нахождение эллипсоида наибольшего объема (S.Boyd)

Задача

Найти эллипсоид вида $\Omega = \{x : x^T H x \leq 1\}$, $H > 0$, наибольшего объема, который содержится в пересечении заданных полос

$$\Omega \subset \bigcap_{i=1}^m \{x : |c_i^T x| \leq 1\}$$



Нахождение эллипсоида наибольшего объема (продолжение)

Ранее показано:

$$\max_{x \in \Omega} |c_i^T x| = \sqrt{c_i^T H^{-1} c_i}$$

$$\text{Vol}(\Omega) = V_n(1) \sqrt{\det H^{-1}}$$

Получаем задачу:

$$\ln \det H^{-1} \rightarrow \max, \quad c_i^T H^{-1} c_i \leq 1, \quad i = 1, \dots, m, \quad H > 0$$

Обозначим $P = H^{-1}$. Получим задачу MAXDET:

$$\ln \det P^{-1} \rightarrow \min, \quad c_i^T P c_i \leq 1, \quad i = 1, \dots, m, \quad P > 0$$

Пример. Оценка H_2 -нормы

Рассмотрим линейную систему, для простоты SISO (один вход — один выход):

$$\begin{aligned} \frac{dx}{dt} &= Ax + Bu, & y &= Cx, \\ A \in \mathbb{R}^{n \times n}, \quad B \in \mathbb{R}^{n \times 1}, \quad C \in \mathbb{R}^{1 \times n}. \end{aligned}$$

Предположим, что A гурвицева (система устойчива).

H_2 -норму системы можно вычислить по формуле:

$$\|H\|_2 = \left[\int_0^{+\infty} (Ce^{At}B)^2 dt \right]^{1/2}.$$

Можно показать, что

$$\|H\|_2 = \left[\frac{1}{2\pi} \int_{-\infty}^{+\infty} |H(i\omega)|^2 d\omega \right]^{1/2}, \quad H(s) = C(sI_n - A)^{-1}B.$$

Оценка H_2 -нормы. Интегральное представление

Для нахождения H_2 -нормы вычислим интеграл:

$$\|H\|_2^2 = \int_0^{+\infty} (Ce^{At}B)^2 dt.$$

Так как $Ce^{At}B = B^T e^{A^T t} C^T$, то

$$(Ce^{At}B)^2 = B^T e^{A^T t} C^T C e^{At} B, \quad (Ce^{At}B)^2 = C e^{At} B B^T e^{A^T t} C^T.$$

Отсюда

$$\|H\|_2^2 = B^T \int_0^{+\infty} e^{A^T t} C^T C e^{At} dt B, \quad \|H\|_2^2 = C \int_0^{+\infty} e^{At} B B^T e^{A^T t} dt C^T.$$

Решение уравнения Ляпунова

Пусть A, C — матричные коэффициенты, матрица A — гурвицева.

Тогда матричное уравнение Ляпунова

$$PA + A^T P = -C^T C$$

имеет единственное решение, которое вычисляется по формуле

$$P = \int_0^{+\infty} e^{A^T t} C^T C e^{At} dt,$$

причем $P = P^T \geq 0$.

Доказательство: обозначим $Q(t) = e^{A^T t} C^T C e^{At}$.

Тогда $\dot{Q} = A^T Q + QA$ и

$$\int_0^{+\infty} \dot{Q}(t) dt = Q(+\infty) - Q(0) = 0 - C^T C.$$

Оценка H_2 -нормы. Уравнения Ляпунова

Если матрица A гурвицева, то уравнение Ляпунова имеет единственное решение $P^T = P$, $P \geq 0$:

$$PA + A^T P = -C^T C, \quad P = \int_0^{+\infty} e^{A^T t} C^T C e^{At} dt.$$

Таким образом,

$$\|H\|_2^2 = B^T \int_0^{+\infty} e^{A^T t} C^T C e^{At} dt B,$$

сводится к виду:

$$\|H\|_2^2 = B^T P B, \quad PA + A^T P = -C^T C, \quad P \geq 0.$$

Оценка H_2 -нормы. Уравнения Ляпунова

Аналогично:

$$\|H\|_2^2 = C \int_0^{+\infty} e^{At} BB^T e^{A^T t} dt C^T$$

сводится к виду

$$\|H\|_2^2 = CP_0C^T, \quad P_0A^T + AP_0 = -BB^T, \quad P_0 \geq 0.$$

Задача свелась к решению линейного матричного уравнения относительно P или P_0 . Однако с вычислительной точки зрения лучше иметь дело с неравенствами, а не с равенствами.

Оценка H_2 -нормы. Неравенства Ляпунова

Сравним решения уравнения и неравенства Ляпунова:

$$PA + A^T P = -C^T C, \quad \tilde{P}A + A^T \tilde{P} \leq -C^T C.$$

Тогда

$$(\tilde{P} - P)A + A^T(\tilde{P} - P) \leq 0.$$

Так как A гурвицева, то $\tilde{P} - P \geq 0$, откуда $B^T \tilde{P} B \geq B^T P B$.

Т.е. $B^T \tilde{P} B$ достигает минимума на решениях уравнения.

Сведем задачу

$$\|H\|_2^2 = B^T P B, \quad PA + A^T P = -C^T C, \quad P \geq 0.$$

к задаче:

$$B^T P B \rightarrow \inf, \quad PA + A^T P < -C^T C, \quad P > 0.$$

Оценка H_2 -нормы. Неравенства Ляпунова

Аналогично сведем задачу

$$\|H\|_2^2 = CP_0C^T, \quad P_0A^T + AP_0 = -BB^T, \quad P_0 \geq 0.$$

к задаче

$$CP_0C^T \rightarrow \inf, \quad P_0A^T + AP_0 < -BB^T, \quad P_0 > 0.$$

Оценка H_2 -нормы. LMIs

Преобразуем первую задачу

$$B^T P B \rightarrow \inf, \quad PA + A^T P < -C^T C, \quad P > 0$$

к виду: $\gamma \in \mathbb{R}$ — новая переменная, $\gamma > 0$,

$$\gamma \rightarrow \inf, \quad B^T P B < \gamma^2, \quad PA + A^T P < -C^T C, \quad P > 0.$$

Вместо P определим новую переменную $X = \gamma P^{-1}$, откуда $X^{-1} = \frac{1}{\gamma} P$ и

$$\gamma \rightarrow \inf, \quad B^T X^{-1} B < \gamma, \quad X^{-1} A + A^T X^{-1} < -\frac{1}{\gamma} C^T C, \quad \gamma > 0.$$

Эквивалентно:

$$\gamma \rightarrow \inf, \quad B^T X^{-1} B < \gamma, \quad AX + XA^T < -\frac{1}{\gamma} XC^T CX, \quad \gamma > 0.$$

Оценка H_2 -нормы. LMIs (окончание)

Используя формулы Шура, сведем задачу

$$\gamma \rightarrow \inf, \quad B^T X^{-1} B < \gamma, \quad AX + XA^T < -\frac{1}{\gamma} XC^T CX, \quad \gamma > 0$$

к эквивалентной задаче с LMIs: $\|H\|_2 = \gamma$, где

$$\gamma \rightarrow \inf, \quad \begin{bmatrix} X & B \\ B^T & \gamma \end{bmatrix} > 0, \quad \begin{bmatrix} AX + XA^T & XC^T \\ CX & -\gamma \end{bmatrix} < 0, \quad X^T = X.$$

Аналогично, для второй задачи получаем

$$\gamma \rightarrow \inf, \quad \begin{bmatrix} Y & C^T \\ C & \gamma \end{bmatrix} > 0, \quad \begin{bmatrix} YA + A^T Y & YB \\ B^T Y & -\gamma \end{bmatrix} < 0, \quad Y^T = Y.$$

Если не минимизировать γ , она дает оценку H_2 -нормы сверху.

Оценка L_2 -усиления

Вновь рассмотрим линейную SISO систему

$$\frac{dx}{dt} = Ax + Bu, \quad y = Cx,$$

$$A \in \mathbb{R}^{n \times n}, \quad B \in \mathbb{R}^{n \times 1}, \quad C \in \mathbb{R}^{1 \times n}.$$

Матрица A гурвицева (система устойчива).

Вновь рассмотрим L_2 -норму входного и выходного сигнала на $[0, +\infty)$:

$$\|u\|_2 = \left[\int_0^{+\infty} u^2(t) dt \right]^{1/2}, \quad \|y\|_2 = \left[\int_0^{+\infty} y^2(t) dt \right]^{1/2}.$$

H_∞ -норма

Рассмотрим L_2 -усиление (L_2 gain) системы:

$$\gamma_0 = \sup_{\|u\|_2 \neq 0} \frac{\|y\|_2}{\|u\|_2},$$

где супремум берется по всем решениям, таким что $x(0) = 0$.

Можно показать, что

$$\gamma_0 = \sup_{\omega} |H(i\omega)|.$$

где $H(s) = C(sI - A)^{-1}B$ — передаточная функция системы.

Величина $\sup_{\omega} |H(i\omega)|$ называется H_{∞} -нормой.

Таким образом, L_2 -усиление совпадает с H_{∞} -нормой:

$$\gamma_0 = \|H\|_{\infty}.$$

Оценка L_2 -усиления (продолжение)

Рассмотрим неравенство

$$\begin{bmatrix} HA + A^T H & HB \\ B^T H & 0 \end{bmatrix} + \begin{bmatrix} C^T C & 0 \\ 0 & -\gamma^2 I \end{bmatrix} \leq 0$$

относительно неизвестных матрицы $H = H^T$ и числа $\gamma > 0$. Тогда

$$2x^T H(Ax + Bu) + (Cx)^2 - \gamma^2 u^2 \leq 0, \quad \forall x, \forall u.$$

Очевидно $HA + A^T H \leq -C^T C$, поэтому $H \geq 0$.

Возьмем функцию $V(x) = x^T H x$. В силу системы

$$\frac{dV}{dt} + y^2 - \gamma^2 u^2 \leq 0.$$

Оценка L_2 -усиления (продолжение)

Проинтегрируем неравенство

$$\frac{dV}{dt} + y^2 - \gamma^2 u^2 \leq 0.$$

от 0 до произвольного $T > 0$

$$V(x(T)) - V(x(0)) + \int_0^T (y^2 - \gamma^2 u^2) dt \leq 0.$$

Так как $H \geq 0$ и $x(0) = 0$, то $V(x(T)) \geq 0$ и $V(x(0)) = 0$. Тогда

$$\int_0^T y^2(t) dt \leq \gamma^2 \int_0^T u^2(t) dt, \quad \forall T > 0.$$

Отсюда $\|y\|_2 \leq \gamma \|u\|_2$ и значит $\gamma_0 \leq \gamma$.

Оценка L_2 -усиления (окончание)

Чтобы свести задачу к LMI, введем переменную $t = \gamma^2$ и рассмотрим задачу

$$t \rightarrow \inf, \quad \begin{bmatrix} HA + A^T H & HB \\ B^T H & 0 \end{bmatrix} + \begin{bmatrix} C^T C & 0 \\ 0 & -tI \end{bmatrix} \leq 0$$

Тогда для L_2 -усиления получаем оценку: $\gamma_0 \leq \sqrt{t}$.

Лекция 4. Примеры LMIs. LMI-области

Понятие LMI-области

Определение

Пусть $A \in \mathbb{R}^{n \times n}$ и D — область на комплексной плоскости. Матрица называется D -устойчивой, если собственные числа матрицы A принадлежат D .

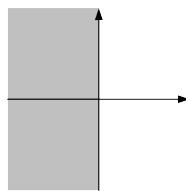
LMI-область: $D = \{z \in \mathbb{C} : f(z) < 0\}$, где

$$f(z) = L + zM + \bar{z}M^T, \quad L, M \in \mathbb{R}^{m \times m}, \quad L^T = L.$$

Неравенство понимается в смысле отрицательной определенности эрмитовой формы с матрицей $f(z) = f(z)^*$, $*$ — эрмитово сопряжение.

Частный случай: устойчивость по Гурвицу

A устойчива по Гурвицу $\iff D = \{z \in \mathbb{C} : \operatorname{Re} z < 0\}$

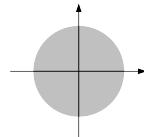


Можно переписать: $\operatorname{Re} z < 0 \iff f(z) = z + \bar{z} < 0$

Здесь $f(z) = L + zM + \bar{z}M^T$, $L = 0$, $M = 1$, $m = 1$.

Частный случай: устойчивость по Шуру

A устойчива по Шуру $\iff D = \{z \in \mathbb{C} : |z| < 1\}$



Можно переписать:

$$|z| < 1 \iff \det \begin{bmatrix} 1 & z \\ \bar{z} & 1 \end{bmatrix} = 1 - |z|^2 > 0 \iff f(z) = \begin{bmatrix} -1 & -z \\ -\bar{z} & -1 \end{bmatrix} < 0$$

$$f(z) = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} + z \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix} + \bar{z} \begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix},$$

$$L = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad M = \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix}$$

Свойства LMI-областей

$D = \{z \in \mathbb{C} : f(z) < 0\}$, где

$$f(z) = L + zM + \bar{z}M^T, \quad L, M \in \mathbb{R}^{m \times m}, \quad L^T = L.$$

- LMI-область выпукла;
- LMI-область симметрична относительно вещественной оси (т.к. $\overline{f(z)} = f(\bar{z}) = f(z)$);
- пересечение LMI-областей есть LMI-область.

Доказательство:

$$L_1 + zM_1 + \bar{z}M_1^T < 0, \quad L_2 + zM_2 + \bar{z}M_2^T < 0,$$

эквивалентно

$$\begin{bmatrix} L_1 & 0 \\ 0 & L_2 \end{bmatrix} + z \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} + \bar{z} \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix}^T < 0.$$

Произведение Кронекера

Пусть $A - m \times n$ -матрица, $B - p \times q$ -матрица,

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{bmatrix}, \quad B = \begin{bmatrix} b_{11} & \dots & b_{1q} \\ \vdots & \ddots & \vdots \\ b_{p1} & \dots & b_{pq} \end{bmatrix}$$

Кронекеровы произведения — блочные матрицы размера $mp \times nq$:

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix}, \quad B \otimes A = \begin{bmatrix} b_{11}A & \dots & b_{1q}A \\ \vdots & \ddots & \vdots \\ b_{p1}A & \dots & b_{pq}A \end{bmatrix}$$

LMI-области. Основное утверждение

Пусть $m \times m$ -матрицы $L = [L_{ij}]$, $M = [M_{ij}]$ определяют LMI-область D . На матрицах A , $H \in \mathbb{R}^{n \times n}$, $H^T = H$, определим симметричную $(mn) \times (mn)$ -матрицу-функцию, состоящую из блоков:

$$Q_{ij}(A, H) = L_{ij}H + M_{ij}HA + M_{ji}A^TH, \quad i, j = 1, \dots, m.$$

По-другому: $Q(A, H) = L \otimes H + M \otimes (HA) + M^T \otimes (A^TH)$, где \otimes — [кронекерово произведение](#).

Теорема

Два утверждения эквивалентны:

- Матрица A является D -устойчивой.
- Существует матрица $H > 0$ такая, что $Q(A, H) < 0$.

LMI-области. Замечание

Если область является пересечением LMI-областей

$$L_1 + zM_1 + \bar{z}M_1^T < 0, \quad L_2 + zM_2 + \bar{z}M_2^T < 0,$$

то

$$\begin{bmatrix} L_1 & 0 \\ 0 & L_2 \end{bmatrix} + z \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} + \bar{z} \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix}^T < 0.$$

и можно рассмотреть

$$H = \begin{bmatrix} H_1 & 0 \\ 0 & H_2 \end{bmatrix},$$

т.е

$$\begin{aligned} L_1 \otimes H_1 + M_1 \otimes (H_1 A_1) + M_1^T \otimes (A_1^T H_1) &< 0, \\ L_2 \otimes H_2 + M_2 \otimes (H_2 A_2) + M_2^T \otimes (A_2^T H_2) &< 0. \end{aligned}$$

LMI-области. Устойчивость по Гурвицу

Пусть

$$D = \{z \in \mathbb{C} : \operatorname{Re} z < 0\}, \quad L = 0, \quad M = 1$$

Получаем:

$$\begin{aligned} L \otimes H = 0, \quad M \otimes (HA) &= HA, \quad M^T \otimes (A^T H) = A^T H \\ Q(A, H) &= HA + A^T H < 0. \end{aligned}$$

Условия устойчивости:

$$\exists H : HA + A^T H < 0, \quad H > 0$$

LMI-области. Устойчивость по Шуру

$$D = \{z \in \mathbb{C} : |z| < 1\}, \quad L = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad M = \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix}$$

Получаем:

$$\begin{aligned} L \otimes H &= \begin{bmatrix} -H & 0 \\ 0 & -H \end{bmatrix}, \quad M \otimes (HA) = \begin{bmatrix} 0 & -HA \\ 0 & 0 \end{bmatrix} \\ M^T \otimes (A^T H) &= \begin{bmatrix} 0 & 0 \\ -A^T H & 0 \end{bmatrix}, \quad Q(A, H) = \begin{bmatrix} -H & -HA \\ -A^T H & -H \end{bmatrix} \end{aligned}$$

Используем дополнение Шура:

$$H - (A^T H)H^{-1}(HA) = H - A^T H A > 0.$$

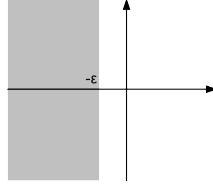
Условия устойчивости:

$$\exists H : A^T H A - H < 0, \quad H > 0$$

LMI-области. Вертикальная полуплоскость

Пусть

$$D = \{z \in \mathbb{C} : \operatorname{Re} z < -\varepsilon\}, \quad \varepsilon \in \mathbb{R}.$$



Так как $2\operatorname{Re} z = z + \bar{z}$, то: $z + \bar{z} < -2\varepsilon$. Здесь

$$f(z) = 2\varepsilon + z + \bar{z} < 0, \quad m = 1, \quad L = 2\varepsilon, \quad M = 1.$$

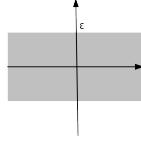
Получаем условие устойчивости:

$$\begin{aligned} Q(A, H) &= HA + A^T H + 2\varepsilon H = \\ &= H(A + \varepsilon I_n) + (A^T + \varepsilon I_n)H < 0, \quad H > 0. \end{aligned}$$

LMI-области. Горизонтальная полоса

Пусть

$$D = \{z \in \mathbb{C} : |\operatorname{Im} z| < \varepsilon\}, \quad \varepsilon > 0.$$



Так как $z - \bar{z} = 2i\operatorname{Im} z$, то: $|z - \bar{z}| < 2\varepsilon$. Можно взять:

$$f(z) = \begin{bmatrix} -2\varepsilon & z - \bar{z} \\ -(z - \bar{z}) & -2\varepsilon \end{bmatrix} < 0,$$

$$m = 2, \quad L = \begin{bmatrix} -2\varepsilon & 0 \\ 0 & -2\varepsilon \end{bmatrix}, \quad M = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

Получаем условие D -устойчивости:

$$Q(A, H) = \begin{bmatrix} -2\varepsilon H & HA - A^T H \\ -HA + A^T H & -2\varepsilon H \end{bmatrix} < 0, \quad H > 0.$$

LMI-области. Круг радиуса r с центром в нуле

Пусть

$$D = \{z \in \mathbb{C} : |z| < r\}, \quad r > 0.$$

$$f(z) = \begin{bmatrix} -r & -z \\ -\bar{z} & -r \end{bmatrix}, \quad m = 2, \quad L = \begin{bmatrix} -r & 0 \\ 0 & -r \end{bmatrix}, \quad M = \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix}.$$

Условие D -устойчивости:

$$Q(A, H) = \begin{bmatrix} -rH & -HA \\ -A^T H & -rH \end{bmatrix} < 0, \quad H > 0.$$

Используя дополнение Шура:

$$rH - A^T H \frac{1}{r} H^{-1} H A > 0.$$

Перемножим:

$$r^2 H - A^T H A > 0.$$

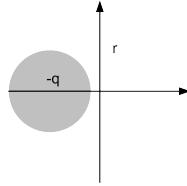
Единичный круг: $r = 1$. Тогда условие устойчивости

$$A^T H A - H < 0.$$

LMI-области. Круг с центром на вещественной оси

Пусть

$$D = \{z \in \mathbb{C} : |z + q| < r\}, \quad q \in \mathbb{R}, \quad r > 0.$$



$$f(z) = \begin{bmatrix} -r & z + q \\ \bar{z} + q & -r \end{bmatrix},$$

$$m = 2, \quad L = \begin{bmatrix} -r & q \\ q & -r \end{bmatrix}, \quad M = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

Условие D -устойчивости:

$$Q(A, H) = \begin{bmatrix} -rH & qH + HA \\ qH + A^T H & -rH \end{bmatrix} < 0, \quad H > 0.$$

LMI-области. Круг (продолжение)

Преобразуем неравенство

$$\begin{bmatrix} -rH & qH + HA \\ qH + A^T H & -rH \end{bmatrix} < 0, \quad H > 0,$$

используя дополнение Шура:

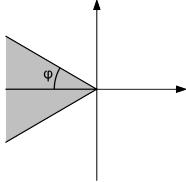
$$rH - (qH + A^T H) \frac{1}{r} H^{-1} (qH + HA) > 0.$$

Перемножим:

$$(r^2 - q^2)H - q(HA + A^T H) - A^T H A > 0.$$

LMI-области. Сектор с вершиной в нуле

Рассмотрим сектор ($0 < \varphi < \pi/2$):



Его можно описать неравенствами:

$$\left| \frac{\operatorname{Im} z}{\operatorname{Re} z} \right| < \operatorname{tg} \varphi, \quad \operatorname{Re} z < 0$$

или

$$|\operatorname{Im} z| \cos \varphi < (-\operatorname{Re} z) \sin \varphi.$$

LMI-области. Сектор с вершиной в нуле (продолжение)

Неравенство

$$|\operatorname{Im} z| \cos \varphi < (-\operatorname{Re} z) \sin \varphi.$$

можно переписать как

$$|z - \bar{z}| \cos \varphi < -(z + \bar{z}) \sin \varphi$$

или

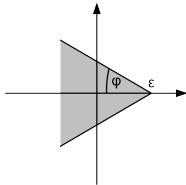
$$f(z) = \begin{bmatrix} (z + \bar{z}) \sin \varphi & (z - \bar{z}) \cos \varphi \\ -(z - \bar{z}) \cos \varphi & (z + \bar{z}) \sin \varphi \end{bmatrix} < 0.$$

Условие D -устойчивости:

$$\begin{bmatrix} (HA + A^T H) \sin \varphi & (HA - A^T H) \cos \varphi \\ -(HA - A^T H) \cos \varphi & (HA + A^T H) \sin \varphi \end{bmatrix} < 0.$$

LMI-области. Сектор с вершиной на вещественной оси

Рассмотрим более общий сектор ($0 < \varphi < \pi/2$):



Его можно описать неравенствами:

$$\left| \frac{\operatorname{Im} z}{\operatorname{Re} z - \varepsilon} \right| < \operatorname{tg} \varphi, \quad \operatorname{Re} z < \varepsilon.$$

или

$$|\operatorname{Im} z| \cos \varphi < -(\operatorname{Re} z - \varepsilon) \sin \varphi.$$

LMI-области. Сектор с вершиной на вещественной оси (продолжение)

Неравенство

$$|\operatorname{Im} z| \cos \varphi < -(\operatorname{Re} z - \varepsilon) \sin \varphi.$$

перепишем как

$$|z - \bar{z}| \cos \varphi < -(z + \bar{z} - 2\varepsilon) \sin \varphi$$

или

$$f(z) = \begin{bmatrix} (z + \bar{z} - 2\varepsilon) \sin \varphi & (z - \bar{z}) \cos \varphi \\ -(z - \bar{z}) \cos \varphi & (z + \bar{z} - 2\varepsilon) \sin \varphi \end{bmatrix} < 0.$$

Условие D -устойчивости:

$$\begin{bmatrix} (HA + A^T H - 2\varepsilon H) \sin \varphi & (HA - A^T H) \cos \varphi \\ -(HA - A^T H) \cos \varphi & (HA + A^T H - 2\varepsilon H) \sin \varphi \end{bmatrix} < 0, \quad H > 0.$$

LMI-области (заключение)

С помощью описанного метода можно рассмотреть пересечения указанных областей: полуплоскостей, полос, кругов, секторов.

Необходимое условие: они должны быть симметричны относительно вещественной оси.

Лекция 5. Программное обеспечение для исследования LMIs

Две формы задач SDP

- Исходная форма задач SDP — обычно матричная, например:

$$\begin{bmatrix} HA + A^T H & HB \\ B^T H & 0 \end{bmatrix} < F, \quad H > 0, \quad H \in \mathbb{S}^n$$

$H = H^T$ — **матричная переменная** (matrix variable), A, B, F — матричные коэффициенты.

Возникает из содержательных постановок задач.

- Стандартная форма задачи SDP, например:

$$c^T x \rightarrow \min, \quad F(x) \geq 0, \quad x \in \mathbb{R}^m,$$

x — **скалярные переменные** (decision variables).

Используются в численных методах.

Решатели и интерфейсы для задач SDP

Два вида программ:

- Решатели (solvers) — решают задачу в стандартной форме.
- Интерфейсы (interfaces, parcers) —
 - преобразуют исходную задачу в стандартную форму,
 - преобразуют решение в виде скалярных переменных к матричным переменным.

Этапы решения:

1. Преобразование исходной задачи к стандартной форме (со скалярными переменными) — interface.
2. Решение задачи в стандартной форме с помощью численных методов — solver.
3. Преобразование решения к матричным переменным — interface.

Решатели для задач SDP — программная реализация

- Написаны на универсальных языках высокого уровня, допускающих компиляцию в объектный модуль (обычно на C).
- Вызываются в какой-либо из стандартных сред моделирования (MATLAB, Scilab, PENOPT).
- Условия использования:

- полностью бесплатные,
- коммерческие.
- Исходный код обычно закрыт (только DLL библиотеки).
- Обычно реализуют численные методы внутренней точки, часто — одновременное решение прямой и двойственной задачи (primal-dual).

Интерфейсы для задач SDP — программная реализация

- Обычно пишут на языках сред моделирования (MATLAB, Scilab).
- Условия использования:
 - бесплатные,
 - коммерческие.
- Варианты интерфейса:
 - интерфейс рассчитан на работу с одним определенным решателем (часто в едином пакете — устаревшая схема);
 - интерфейс рассчитан на работу с несколькими решателями на выбор (предпочтительный вариант).
- Исходный код обычно открыт.

Пакет LMILAB

- Входит в стандартный пакет расширения MATLAB — Robust Control Toolbox. (До версии MATLAB 7 входил в LMI Control Toolbox.)
- Разработчики: P. Gahinet, M. Chilali (INRIA—The MathWorks), A. C. Немировский (Технион), A. Laub (Калифорнийский ун-т), 1995.
- Коммерческий, хорошо отлажен.
- Содержит интерфейс и решатель, возможна работа с GUI.
- Задачи: FEASP, SDP, GEVP. Ограничения: все неравенства должны быть строгими; не работает с комплексными данными, мало эффективен для задач большой размерности.
- Алгоритм: вариант метода внутренней точки (проективный алгоритм Немировского и Нестерова).
- Использует работу с функциями (процедурное программирование).

Пакет LMITOOL

- Входит в дистрибутив среды моделирования Scilab.
- Полностью бесплатный (freeware):
<http://www.scilab.org>
- Разработчики: R. Nikoukhah, F. Delebecque (INRIA) и L. El Ghaoui (ENSTA), середина 1990-х.
Основывается на пакете SP L. Vandenberghe (Калифорнийский ун-т) и S. Boyd (Стэнфордский ун-т).
- Содержит интерфейс и решатель, возможна работа с GUI.
- Задачи FEASP и SDP. Возможны ограничения типа нестрогих неравенств и равенств.
- Алгоритм: одновременное решение прямой и двойственной задачи (метод Нестерова—Тодда).
- Использует работу с функциями.

Некоторые популярные солверы

- SeDuMi (Self Dual Minimization)
 - Разработчик: Jos Sturm (Тилбургский ун-т, Нидерланды), 2001.

Йос Штурм (1971–2003)
 - Сопровождение с 2004 года: Лихайский ун-т (Пенсильвания), <http://sedumi.ie.lehigh.edu/>
 - Полностью бесплатный.
 - Алгоритм: одновременное решение прямой и двойственной задачи (модификация Нестерова—Тодда).
 - Преимущества: быстрота, надежность.
- SDPT3
 - Разработчики: Kim-Chuan Toh , Michael J. Todd, Reha H. Tutuncu (Национальный ун-т Сингапура),
 - Полностью бесплатный: <https://blog.nus.edu.sg/mattohkc/softwares/sdpt3/>
 - Алгоритм: решение прямой задачи.

Некоторые популярные солверы (замечание)

Решатели SeDuMi и SDPT3 входят в поставку интерфейсного пакета CVX. Они адаптированы для использования в этом пакете.

Некоторые популярные солверы (продолжение)

- CSDP
 - Разработчик: B. Borchers (New Mexico Tech)
 - Полностью бесплатный: <https://projects.coin-or.org/Csdp/>
 - Алгоритм: одновременное решение прямой и двойственной задачи (модификация Хельмберга и др. «предиктор—корректор» — специальный выбор барьерного параметра).
 - Эффективно работает с разреженными матрицами.
- SDPA
 - Разработчик: K. Fujisawa и др. (Tokyo Institute of Technology)
 - Полностью бесплатный: <http://sdpa.indsys.chuo-u.ac.jp/sdpa/>
 - Алгоритм: одновременное решение прямой и двойственной задачи (модификация Меротры «предиктор—корректор»).
 - Экономное использование оперативной памяти. Удобен для работы с большими, блочными и разреженными матрицами.

Некоторые популярные солверы (продолжение)

PENSDP

- Разработчики: Michal Koçvara (Чехия) и Michael Stingl (Германия)
- Коммерческий пакет. Часть программного комплекса PENOPT:
<http://www.penopt.com/>
- Предназначен для работы с матрицами большой размерности и разреженными матрицами.
- Может использоваться с внешним интерфейсом (YALMIP) и внутренним интерфейсом PEN среды PENOPT.
- Алгоритм: оригинальный вариант барьерного метода, основанный на диссертации М. Стингла.

Интерфейсный пакет YALMIP

Один из самых популярных интерфейсов для решения оптимизационных задач. Разработчик: Johan Löfberg (ныне — университет Линчёпинга,



Швеция).

- Для работы в MATLAB. ОС: Windows, Solaris и Linux.
- Полностью бесплатный: <http://yalmip.github.io>
- Интерфейс с более 40 решателями для разных оптимизационных задач (LP, QP, SDP и др.).

- Решатели SDP (бесплатные): SeDuMi, SDPT3, CSDP, DSDP, SDPA, SDPLR.
- Решатели SDP (коммерческие): LMILAB, PENSMP, PENBMI.
- Ограничения типа строгих и нестрогих неравенств, равенств; комплексные данные.
- Использует объектно-ориентированное программирование (работу с классами).

Интерфейсный пакет CVX

- Разработчики: Michael Grant, Stephen Boyd, Yinyu Ye (2005, Стенфордский университет, CVX Research).



- Для работы в MATLAB с Windows, Linux и Mac.
- Бесплатный (есть платная версия): <http://cvxr.com/cvx/>
- Работает в MATLAB (разрабатывается версия для Octave). Определяет новый, интуитивно понятный язык моделирования для решения задач выпуклого программирования.
- Следует книге: S. Boyd и L. Vandenberghe, *Convex Optimization*.
<http://stanford.edu/~boyd/cvxbook/>
- Интерфейс с решателями: SeDuMi, SDPT3, а также Gurobi и MOSEK (коммерч.).

Решение задач SOS

Пусть $p(\bar{x})$ — многочлен от n переменных ($\bar{x} \in \mathbb{R}^n$) степени m .

Многочлен $p(\bar{x})$ называется **SOS-многочленом** (Sum Of Squares polynomial), если существуют число k и многочлены $q_1(\bar{x}), \dots, q_k(\bar{x})$ такие, что

$$p(\bar{x}) = \sum_{i=1}^k [q_i(\bar{x})]^2, \quad \forall \bar{x} \in \mathbb{R}^n.$$

Работа с SOS-многочленами тесно связана с LMI. Можно использовать два вида пакетов:

- Пакет SOSTOOLS + MATLAB Symbolic Math Toolbox + SDP солвер (SeDuMi и пр.) Разработчики: Pablo A. Parrilo и др. (MIT–Caltech, Oxford Univ., Univ. Minnesota)
- Пакет SOSTOOLS + MATLAB Symbolic Math Toolbox + SDP солвер (SeDuMi и пр.) Разработчики: Pablo A. Parrilo и др. (MIT–Caltech, Oxford Univ., Univ. Minnesota)
- Пакет SOSTOOLS + MATLAB Symbolic Math Toolbox + SDP солвер (SeDuMi и пр.) Разработчики: Pablo A. Parrilo и др. (MIT–Caltech, Oxford Univ., Univ. Minnesota)
- SOS-модуль пакета YALMIP (входит в пакет).

Выводы

Пакеты имеют разные возможности. Желательно знакомство не с одним пакетом, а с несколькими.

Лекция 6. Пакет LMILAB (создание системы LMIs)

Общая характеристика

- Входит в стандартный пакет расширения MATLAB — Robust Control Toolbox.
- Хорошо отлажен и документирован.
- Содержит интерфейс и решатель, возможна работа с GUI.
- Задачи: FEASP, SDP, GEVP (нахождение обобщенных собственных чисел).
- Ограничения:
 - все неравенства должны быть строгими;
 - не работает с комплексными данными;
 - мало эффективен для задач большой размерности.
- Алгоритм: вариант метода внутренней точки (проективный алгоритм Немировского и Нестерова).

Функции пакета

- Описание исходной задачи
 - Описание матричных переменных
 - Описание линейных неравенств
 - Описание целевой функции (если нужно)
- Взаимные преобразования матричных и скалярных переменных.
- Работа с солверами для решения задач разных типов.
- Дополнительный сервис:
 - Получение информации о системе.
 - Оценивание неравенств в заданных точках.
 - Модификация системы неравенств.

Обычная последовательность действий

- Описание матричных переменных
- Описание линейных неравенств
- Описание целевой функции (если нужно)
- Вызов солвера для решения задачи нужного типа.
- Извлечение результата из выходных данных солвера.

Все данные о системе и результатах работы солвера хранятся в закодированном виде во вспомогательном одномерном массиве (будем называть его [внутренним представлением системы](#)). Массив формируется при описании системы.

Извлечь из него данные можно только с помощью специальных сервисных команд.

Создание системы LMIs

Пусть \mathbb{S}^n — линейное пространство симметричных $n \times n$ -матриц. Простейший вид отдельного неравенства:

$$L(x) < R(x), \quad L(x), R(x) : \mathbb{R}^m \mapsto \mathbb{S}^n,$$

$L(x)$, $R(x)$ аффинно зависят от x , x — *скалярные* переменные (decision variables).

Более общий вид неравенства:

$$M^T L(x) M < N^T R(x) N,$$

$M, N \in \mathbb{R}^{n \times k}$ — заданные постоянные матрицы (внешние множители).

Два способа описания:

- чисто программный,
- с помощью графического интерфейса пользователя (GUI) (здесь не рассматриваем).

Функция setlmis

Вызов функции `setlmis` начинает описание новой системы LMI.

Создание пустой (новой) системы:

```
setlmis([ ])
```

Инициализация ранее созданной системой с именем `lmisys`:

```
setlmis(lmisys)
```

Здесь `lmisys` содержит внутреннее представление LMI в виде одномерного массива.

Внутреннее представление получается по исходному с помощью функции `getlmis` (см. далее).

Функция lmivar

Служит для создания новой *матричной* переменной.

Простейший вызов:

```
X = lmivar(type,struct)
```

Здесь `X` — имя переменной, `type` и `struct` определяют ее структуру.

Параметр `type`:

- 1 — симметричная блочно-диагональная матрица,
- 2 — произвольная прямоугольная матрица,
- 3 — прямоугольная матрица, описываемая шаблоном.

Параметр `struct` содержит более детальную информацию.

Функция lmivar (продолжение)

Нумерация создаваемых переменных:

- Выходной параметр X равен номеру созданной **матричной переменной** (т.е. после первого вызова функции lmivar он равен 1, после второго — 2 и т.д.).
- **Скалярные переменные** нумеруются, начиная с единицы (сплошная нумерация по всем матричным переменным).

Число скалярных переменных зависит от симметричности и диагональности:

$$X = \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}, \quad X = \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix}, \quad X = \begin{bmatrix} x_1 & 0 \\ 0 & x_2 \end{bmatrix}$$

Две симметричные переменные, созданные последовательно:

$$X = \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix}, \quad Y = \begin{bmatrix} x_4 & x_5 \\ x_5 & x_6 \end{bmatrix}$$

Функция lmivar: type 1

Симметричная блочно-диагональная матрица из k блоков:

X = lmivar(1,struct)

$$X = \begin{bmatrix} X_{11} & 0 & \dots & 0 \\ 0 & X_{22} & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & X_{kk} \end{bmatrix}, \quad X_{ii} \text{ — блок } n_i \times n_i$$

$$\text{struct} = [n_1 \ s_1; \ n_2 \ s_2; \ \dots \ n_k \ s_k] \quad (k \times 2)$$

n_i — размер диагонального блока X_{ii} ; s_i — тип диагонального блока X_{ii} :

$s_i = 0$ в случае блока вида tI , t — скаляр, I — единичная матрица («скалярный» блок); $s_i = 1$ в случае произвольного симметричного блока; $s_i = -1$ в случае нулевого блока.

Функция lmivar: type 1 — примеры

1. Скалярная переменная eps:

eps = lmivar(1,[1 0]) (можно [1 1])

2. Переменная H представляет собой симметричную $n \times n$ -матрицу (один симметричный блок):

H = lmivar(1,[n 1])

3. Переменная H представляет собой диагональную $n \times n$ -матрицу. В struct задаем n строк вида [1 1]:

```
struct(1:n, 1:2) = 1;  
H = lmivar(1,struct)
```

т.е.

$$\text{struct} = \left\{ \begin{bmatrix} 1 & 1 \\ \dots & \dots \\ 1 & 1 \end{bmatrix} \right\} n$$

Функция lmivar: type 1 — примеры (продолжение)

Матрица S симметрична, имеет размеры 4×4 и структуру

$$S = \begin{bmatrix} s_{11} & 0 & 0 & 0 \\ 0 & s_{11} & 0 & 0 \\ 0 & 0 & s_{22} & s_{23} \\ 0 & 0 & s_{23} & s_{33} \end{bmatrix}$$

Имеем: диагональный блок 2×2 (тип блока 0) и симметричный блок 2×2 (тип блока 1).

```
S = lmivar(1,[2 0; 2 1])
```

Функция lmivar: type 2

Произвольная прямоугольная матрица

```
X = lmivar(2,struct)
```

struct = $[m, n]$, если X — произвольная матрица размера $m \times n$.

Пример: матрица X — произвольная, размера 5×6 :

```
X = lmivar(2,[5 6])
```

Функция lmivar: type 3

Прямоугольная матрица, описываемая шаблоном

```
X = lmivar(3,struct)
```

struct — матрица-шаблон того же размера, что и X, ее элементы равны:

$$\begin{aligned} 0, \quad &\Rightarrow X(i,j) = 0; \\ +k, \quad &\Rightarrow X(i,j) = x_k; \\ -k, \quad &\Rightarrow X(i,j) = -x_k. \end{aligned}$$

Пример:

```
X = lmivar(1,[2 1]) % симметричная матрица 2x2
D = lmivar(3,[1 0; 0 -3]) % матрица по шаблону 2x2
```

Получаем:

$$X = \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix}, \quad \text{шаблон} = \begin{bmatrix} 1 & 0 \\ 0 & -3 \end{bmatrix}, \quad D = \begin{bmatrix} x_1 & 0 \\ 0 & -x_3 \end{bmatrix}$$

Функция lmivar: дополнительные выходные параметры

Более общая форма вызова функции lmivar:

```
[X,ndec,xdec] = lmivar(type,struct)
```

Выходные параметры:

- **ndec** — общее число созданных к моменту после вызова функции скалярных переменных,
- **xdec** — структура созданной матричной переменной **X** (зависимость от скалярных переменных, как в матрице-шаблоне)

Пример: До вызова функции было уже создано 5 скалярных переменных. Функция **lmivar** создала еще 3 скалярных переменных. Выходной параметр **ndec** получит значение 8.

Функция lmivar (пример — шаг 1)

Рассмотрим последовательность команд:

```
>>setlmis([ ])
>>[a,n1,x1]=lmivar(1,[2 0])

a =
1

n1 =
1

x1 =
1     0
0     1
```

Сформированная переменная **a** получает номер 1. Она содержит одну скалярную переменную (ее номер также 1) и имеет структуру диагональной матрицы.

Функция lmivar (пример — шаг 2)

Продолжим создание переменных.

```
>> [b,n2,x2]=lmivar(1,[2 1])

b =
2

n2 =
4

x2 =
2     3
3     4
```

Вторая переменная **b** получила номер 2. Она представляет собой симметричную матрицу и содержит три новые скалярные переменные с номерами 2, 3, 4. Общее число созданных скалярных переменных равно четырем.

Функция lmivar (пример — шаг 3)

Создадим третью матричную переменную:

```
>> [c,n3,x3]=lmivar(3,[1 0; 0 -4])

c =
3

n3 =
4

x3 =
1     0
0    -4
```

Матрица c составлена из ранее созданных скалярных переменных (с номерами 1 и 4). Общее число скалярных переменных при этом не увеличилось (равно четырем).

Функция newlmi

Вызов `newlmi` начинает описание линейного матричного неравенства.

```
lmitag = newlmi
```

Выходной параметр `lmitag` (тег) — целое значение, равное номеру неравенства в системе.

Команда `newlmi` не обязательна — неравенству можно просто сопоставить порядковый номер в виде целого числа, но использование идентификаторов вместо номеров улучшает читаемость программы.

Функция lmitem

- Функция `lmitem` определяет структуру неравенств $L(x) < R(x)$ (т.е. матрицы $R(x)$ и $L(x)$).
- Матрицы $L(x)$ и $R(x)$ естественным образом разбиваются на блоки, каждый блок описывается своим вызовом функции `lmitem`.

Важное замечание

Из каждой пары симметричных блоков следует описывать только один. Например, если

$$R = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix},$$

где $R_{12} = R_{21}^T$, то следует описать либо R_{12} , либо R_{21} , но не оба вместе!

Не следует описывать блоки, состоящие из нулей (нули заносятся по умолчанию).

Функция `lmiterm` (продолжение)

Для описания блока функция `lmiterm` вызывается, по крайней мере, один раз. С помощью одного вызова можно выполнить одно из действий:

- занести в блок постоянную матрицу-коэффициент;
- занести в блок слагаемое вида AXB или $AX^T B$, где A, B — постоянные матрицы, X — матричная переменная;
- задать внешний множитель (он относится не к отдельному блоку, а ко всей матрице $L(x)$ или $R(x)$).

Функция `lmiterm` не имеет выходных параметров, ее вызов:

```
lmiterm(termid,A,B,flag)
```

Функция `lmiterm` (продолжение)

Параметр `termid` — вектор из четырех компонент.

- `termid(1) = +n` — работа с левой частью n -го неравенства, значение $-n$ — работа с его правой частью. (Считаем, что левая часть строго меньше правой. В качестве n обычно используют идентификатор неравенства, полученный с помощью `newlmi`.)
- Значение `termid(2 : 3) = [i j]` означает блок с координатами (i, j) . Для внешних множителей `termid(2 : 3) = [0 0]`.
 $\text{termid}(4) = 0$ — постоянное слагаемое,
• `termid(4)` — тип добавляемого слагаемого: $\text{termid}(4) = X$ — слагаемое вида AXB ,
 $\text{termid}(4) = -X$ — слагаемое вида $AX^T B$.
Здесь X — матричная переменная.

Функция `lmiterm` (продолжение)

- Параметр `A` задает значение либо внешнего множителя, либо постоянного слагаемого, либо матричного коэффициента в выражениях AXB или $AX^T B$.
- Параметр `B` определяет правый множитель в выражениях AXB или $AX^T B$. В остальных случаях (постоянного слагаемого и внешнего множителя) этот параметр не указывается.
- Параметр `flag` — необязательный параметр, который позволяет упростить задание выражения вида $AXB + B^T X^T A^T$ в диагональном блоке. При этом достаточно определить слагаемое AXB и придать параметру `flag` значение '`s`'.

Функция `lmiterm` (пример 1)

Пусть \mathbb{S}^n — линейное пространство симметричных $n \times n$ -матриц. Зададим неравенство $H > 0$, $H \in \mathbb{S}^n$:

```
H = lmivar(1,[n 1]);  
Hpos=newlmi;  
lmiterm([-Hpos 1 1 H],1,1); % 0 < H
```

С неравенством связан идентификатор `Hpos`.

Функция lmitem (пример 2)

Пусть A, b, F — заданные коэффициенты, $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^{n \times m}$, $F \in \mathbb{S}^{n+m}$. Составим неравенство относительно неизвестной $H \in \mathbb{S}^n$:

$$\begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} < F.$$

Фрагмент программы:

```
H = lmivar(1,[n 1]);
lmi1=newlmi;
lmitem([lmi1 1 1 H],1,A,'s'); % HA + A'H
lmitem([lmi1 1 2 H],1,b); % Hb
lmitem([-lmi1 1 1 0],F); % F = const
```

Неравенству сопоставлен идентификатор `lmi1`.

Функция lmitem (пример 3)

Пусть матрицы A, b определены как в предыдущем примере, $G \in \mathbb{S}^n$, $g \in \mathbb{R}^{n \times m}$, $\Gamma \in \mathbb{S}^m$. Запишем неравенство

$$\begin{bmatrix} A^T H A - H & A^T H b \\ b^T H A & b^T H b \end{bmatrix} < \begin{bmatrix} G & g \\ g^T & \Gamma \end{bmatrix},$$

Имеем:

```
H = lmivar(1,[n 1]);
lmi2=newlmi;
lmitem([lmi2 1 1 H],A',A); % A'HA
lmitem([lmi2 1 1 H],1,-1); % -H
lmitem([lmi2 1 2 H],A',b); % A'Hb
lmitem([lmi2 2 2 H],b',b); % b'Hb
lmitem([-lmi2 1 1 0],G); % G = const
lmitem([-lmi2 1 2 0],g); % g = const
lmitem([-lmi2 2 2 0],Gamma); % Gamma = const
```

Функция getlmis

Вызов `getlmis` завершает формирование системы линейных матричных неравенств. Он служит для получения внутреннего представления системы в виде одномерного массива специального вида.

```
lmisys = getlmis
```

Выходной параметр `lmisys` может быть передан информационным функциям или решающим функциям (`solvers`).

Общая схема формирования системы LMIs

- `setlmis`,
- `lmivar`,
- `newlmi`,

- `lmisTerm`,

- :

- `getLmis`.

Функции `setLmis` и `getLmis` вызываются только один раз — в начале и в конце формирования системы неравенств.

Остальные функции (включая `newLmi`) могут вызываться несколько раз.

Лекция 7. Пакет LMILAB: задача разрешимости

Решение задачи FEASP

- На предыдущей лекции мы разобрали вопрос описания переменных и линейных матричных неравенств. Этот этап заканчивается формированием внутреннего представления системы. Внутреннее представление — вектор, который в закодированном виде включает описания переменных и неравенств. (Я буду обозначать его `lmisys`.)
- Для работы с солверами нам дополнительно понадобятся некоторые сервисные и информационные функции.
- Начнем с задачи FEASP: определить, имеет ли решение система неравенств вида

$$L(x) < R(x),$$

где L, R — симметричные, аффинно зависят от вектора x .

Преобразование скалярных переменных в матричные

Функции-решатели (`solvers`) возвращают результат своей работы в виде массива скалярных переменных.

Функция `dec2mat` (`decision to matrix`) позволяет извлечь из него значение матричных переменных:

```
X = dec2mat(lmisys,xvars,k)
```

Здесь мы извлекаем из массива `xvars` матрицу `X`.

- `lmisys` содержит внутреннее представление системы неравенств (в частности, определяет структуру матриц),
- `xvars` — массив (вектор) скалярных переменных,
- `k` — номер извлекаемой матричной переменной (обычно используют идентификатор этой переменной). Массив `xvars` может содержать элементы нескольких матричных переменных.

Выходная переменная `X` — вещественная матрица MATLAB.

Функция `feasp`

Решает задачу feasibility problem — нахождения какого-либо допустимого решения x системы неравенств вида

$$L(x) < R(x)$$

Попутно функция проверяет, имеет ли система неравенств вообще какие-либо решения.

По исходному неравенству составляется вспомогательная задача:

$$t \rightarrow \inf \quad \text{при} \quad L(x) < R(x) + tI,$$

где t — скалярный параметр, I — единичная матрица.

Функция `feasp` формирует строго убывающую последовательность приближений $t_1 > t_2 > \dots$

Если для какого-то $t_k < 0$, вычисления прекращаются. Если за разумное число шагов получаем, что все $t_i \geq 0$, то считаем, что система не имеет решений.

Функция `feasp` (продолжение)

Вызов функции:

```
[tmin,xfeas] = feasp(lmisys,options,target)
```

Ее выходные значения:

- `tmin` — минимальное найденное значение t ,
- `xfeas` — допустимое значение вектора x .

Результат получается в виде вектора скалярных переменных. Чтобы получить из него матричные переменные, следует применить функцию `dec2mat`.

По умолчанию на экран выводится вся последовательность t_1, t_2, \dots

Входные параметры `options`, `target` можно не указывать — они получат значения по умолчанию.

Функция `feasp` (продолжение)

Входные параметры функции:

- `lmisys` содержит описание системы LMIs (внутреннее представление) в виде одномерного массива.
- `options` представляет собой одномерный массив настроек параметров, который может включать до пяти элементов.

Если `options(i) = 0`, используется значение по умолчанию. Правила задания элементов массива приведены в таблице (см. далее).

- `target` определяет условие окончания итерационного процесса. Вычисления прекращаются, когда $t < \text{target}$. По умолчанию `target = 0`. Для надежности, можно взять небольшое `target < 0`.

Параметр `options`

<code>options(1)</code>	Не используется.
<code>options(2)</code>	Максимальное число итераций (по умолчанию = 100).
<code>options(3)</code>	Вещественное значение R (f-radius). Предполагается, что во время вычислений $\ x\ < R$, где $\ \cdot\ $ — евклидова норма вектора. По умолчанию $R = 10^9$. Значение $R < 0$ означает отсутствие каких-либо ограничений.
<code>options(4)</code>	Целое число $L > 1$. Вычисления прекращаются, когда за L последних итераций величина t уменьшилась не более, чем на 1%. По умолчанию $L = 10$.
<code>options(5)</code>	Определяет подробность сообщений. Если придать этому параметру ненулевое значение, трассировка результатов промежуточных итераций отключается.

Пример: разрешимость неравенства Ляпунова

Пусть $A - n \times n$ -матрица. Матрица A гурвицева тогда и только тогда, когда существует $n \times n$ -матрица $H = H^T$ такая, что

$$HA + A^T H < 0, \quad H > 0.$$

Опишем функцию с заголовком:

```
function H = lyap1(A)
```

Здесь: H — решение (если найдено), $H = []$, если решения нет.

Функция: проверка разрешимости неравенства Ляпунова

```
function H0 = lyap1(A)
n = size(A,1); % число строк матрицы A
H0 = [];
setlmis([]);
H = lmivar(1,[n 1]);
MainLMI=newlmi; % неравенство Ляпунова
lmiterm([MainLMI 1 1 H],1,A,'s');
Hpos=newlmi; % неравенство H > 0
lmiterm([-Hpos 1 1 H],1,1);
lmisys=getlmis;
[tmin,x]=feasp(lmisys,[0 100 1e9 10 1]);
if tmin<0
    H0 = dec2mat(lmisys,x,H); % x -> H0
end
```

Разрешимость неравенства Ляпунова: вызов функции

```
function lyapunov()
clc
A=[-1 1; 2 -3];
H = lyap1(A);
```

```

if ~isempty(H)
    disp('Problem is feasible');
    disp(H);
else
    disp('Problem is not feasible');
end

```

Получаем:

```

Problem is feasible
1.3356 0.4373
0.4373 0.3474

```

Вызов функции при включенной трассировке

Включим трассировку (последняя опция 0):

```
[tmin,x]=feasp(lmisy, [0 100 1e9 10 0]);
```

Получаем:

```
The best value of t should be negative for feasibility
```

```

Iteration : Best value of t so far
1           -0.155973

Result: best value of t: -0.155973
f-radius saturation: 0.000% of R = 1.00e+09

Problem is feasible
1.3356 0.4373
0.4373 0.3474

```

Неравенство Ляпунова для положительной системы

Пусть A — $n \times n$ -матрица. Проверить, существует ли [диагональная](#) $n \times n$ -матрица H такая, что

$$HA + A^T H < 0, \quad H > 0.$$

В этом случае нужно по-другому определить переменную H :

```

n = size(A,1);
struct (1:n, 1:2) = 1; % n строк вида [1 1]
H = lmivar(1,struct);

```

Получаем:

```

Problem is feasible
0.4877      0
0      0.1686

```

Пример: круговой критерий абсолютной устойчивости

Рассмотрим нелинейную систему управления

$$\frac{dx}{dt} = Ax + b\xi, \quad \sigma = c^T x, \\ \xi = \varphi(\sigma, t).$$

Здесь $A \in \mathbb{R}^{m \times m}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^m$ — постоянные коэффициенты.

В терминах преобразований Лапласа $\tilde{\sigma}$, $\tilde{\xi}$ связаны соотношением

$$\tilde{\sigma} = -W(s)\tilde{\xi},$$

где $W(s) = c^T(A - sI_m)^{-1}b$ — передаточная функция линейной части системы от входа $-\xi$ к выходу σ .

Пример: круговой критерий (продолжение)

Нелинейная функция $\varphi(\sigma)$ принадлежит классу нелинейностей

$$\mathcal{S} = \left\{ \varphi : \mu_1 \leq \frac{\varphi(\sigma, t)}{\sigma} \leq \mu_2, \quad \forall \sigma, \forall t \right\},$$

где $\mu_1 \in \mathbb{R}$, $\mu_2 \in \mathbb{R}$ — заданные числа, $\mu_1 < \mu_2$. Очевидно $\varphi(0, t) \equiv 0$ и система имеет нулевое состояние равновесия $x(t) \equiv 0$.

Система называется абсолютно устойчивой в классе \mathcal{S} , если ее нулевое состояние равновесия равномерно экспоненциально устойчиво в целом: существуют постоянные $M > 0$, $\varepsilon > 0$ такие, что $\|x(t)\| \leq M\|x(t_0)\| \exp(-\varepsilon(t - t_0))$ при всех $t \geq t_0$, причем M и ε не зависят от функции $\varphi \in \mathcal{S}$.

Пример: круговой критерий (продолжение)

Достаточное условие абсолютної устойчивости:

Круговой критерий

Пусть существует число $\mu_0 \in [\mu_1, \mu_2]$ такое, что $A + \mu_0 bc^T$ гурвицева и

$$1 + (\mu_1 + \mu_2) \operatorname{Re} W(i\omega) + \mu_1 \mu_2 |W(i\omega)|^2 > 0, \quad \forall \omega \in \mathbb{R}.$$

Тогда система абсолютно устойчива в классе \mathcal{S} .

Пример: круговой критерий (продолжение)

Эквивалентное условие абсолютної устойчивости в виде LMIs:

Теорема. Пусть существует симметричная матрица $H > 0$, такая что

$$2x^T H(Ax + b\xi) + (\mu_2 c^T x - \xi)(\xi - \mu_1 c^T x) < 0 \\ \forall x \in \mathbb{R}^m, \forall \xi \in \mathbb{R}, \|x\| + |\xi| \neq 0.$$

Тогда система абсолютно устойчива в классе \mathcal{S} .

Пример: круговой критерий (продолжение)

Задача свелась к проверке разрешимости системы LMIs относительно неизвестной $H \in \mathbb{S}^m$:

$$\begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} < \begin{bmatrix} \mu_1 \mu_2 c c^T & -\frac{1}{2}(\mu_1 + \mu_2)c \\ -\frac{1}{2}(\mu_1 + \mu_2)c^T & 1 \end{bmatrix}, \quad H > 0.$$

Круговой критерий: функция circle

```
function [tmin,H0] = circle(A,b,c,mu1,mu2)
n = size(A,1);
H0 = [ ];
setlmis([ ]);
H=lmivar(1, [n 1]);
MLMI=newlmi; % основное неравенство
lmiterm([MLMI 1 1 H],1,A,'s');
lmiterm([MLMI 1 2 H],1,b);
lmiterm([-MLMI 1 1 0],mu1*mu2*c*c');
lmiterm([-MLMI 1 2 0],-0.5*(mu1+mu2)*c);
lmiterm([-MLMI 2 2 0],1);
Hpos=newlmi; % неравенство H > 0
lmiterm([-Hpos 1 1 H],1,1);
lmisys=getlmis;
[tmin,x0]=feasp(lmisys,[0 100 1e9 10 1]);
if tmin<0
    H0=dec2mat(lmisys,x0,H);
end
```

Пример: круговой критерий (продолжение)

Функцию можно поместить в файл `circle.m`.

Выходные параметры функции:

- $t_{\min} < 0$, если условия критерия выполнены, и ≥ 0 , если не выполнены,
- H_0 — значение матрицы H (если задача не имеет решения, этот параметр получает пустое значение).

Проверим условия кругового критерия при

$$W(s) = \frac{1}{s^2 + 2s + 0.5}, \quad \mu_2 = -\mu_1 = 0.1$$

Круговой критерий: вызов функции circle

```
function main_circle
clc;
mu1=-0.1;
mu2=0.1;
w=tf(1, [1 2 0.5]);
[A,b,c,d]=ssdata(w);
c=-c';
[tmin,H]=circle(A,b,c,mu1,mu2);
if tmin < 0
    disp('A feasible solution is');
    disp(H);
else
    disp('No feasible solutions');
end
```

Круговой критерий: входные параметры функции circle

Здесь:

- **tf** — получение передаточной функции по коэффициентам числителя и знаменателя,
- **ssdata** — вычисление минимальной реализации передаточной функции в пространстве состояний.

Замечание

Функция **ssdata** реализует передаточную функцию в виде

$$W(s) = c(sI - A)^{-1}b + d.$$

В данном примере получаем: $d = 0$,

$$A = \begin{bmatrix} -2.0000 & -0.1250 \\ 4.0000 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 0.5000 \\ 0 \end{bmatrix}, \quad c = [0 \quad 0.5000].$$

Для преобразования к нашим обозначениям используем оператор **c=-c'**.

Круговой критерий: работа функции circle

Результат работы тестового примера:

Problem is feasible

```
0.7240    0.2357  
0.2357    0.1671  
>>
```

Пример: критерий Попова

Сводится к разрешимости LMIs:

$$\begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} + \begin{bmatrix} 0 & \frac{1}{2}\theta A^T c \\ \frac{1}{2}\theta c^T A & \theta c^T b \end{bmatrix} < \begin{bmatrix} 0 & -\frac{1}{2}\mu c \\ -\frac{1}{2}\mu c^T & 1 \end{bmatrix}, \quad H > 0.$$

Здесь неизвестные переменные: H, θ .

Критерий Попова: функция

```
function [H0,theta0] = popov(A,b,c,mu)  
n = size(A,1); H0 = [] ; theta0 = [] ;  
setlmis([]);  
H=lmivar(1,[n 1]); % n x n - матрица  
theta=lmivar(1,[1 1]); % скаляр  
MLMI=newlmi; % основное неравенство  
lmiterm([MLMI 1 1 H],1,A,'s');  
lmiterm([MLMI 1 2 H],1,b);  
lmiterm([MLMI 1 2 theta],A',0.5*c);  
lmiterm([MLMI 2 2 theta],1,c'*b);
```

```
lmitemr([-MLMI 1 2 0],-0.5*mu*c);
lmitemr([-MLMI 2 2 0],1);
Hpos=newlmi; % неравенство H > 0
lmitemr([-Hpos 1 1 H],1,1);
lmisys=getlmis;
[tmin,x]=feasp(lmisys,[0 100 1e9 10 1]);
if tmin<0
    H0 = dec2mat(lmisys,x,H);
    theta0 = dec2mat(lmisys,x,theta);
end
```

Лекция 8. Пакет LMILAB: оптимизационная задача

Исследование системы линейных матричных неравенств

Функции этой группы — самые главные в пакете, собственно ради них пакет и разрабатывался. В качестве входного параметра они используют внутреннее представление системы линейных неравенств (в виде массива, который я обозначаю `lmisys`).

Задача SDP

Рассмотрим задачу SDP в стандартной форме:

$$\begin{aligned} c^T x &\rightarrow \min, \\ L(x) &< R(x), \end{aligned}$$

где x — вектор скалярных переменных, c — заданный вектор-столбец.

Мы уже знаем как определить неравенство $L(x) < R(x)$. Проблема: как задать вектор c , исходя из матричной формы задачи?

Нам понадобится еще несколько сервисных функций.

Примеры целевых функций

- Пусть t — переменная (скаляр),

$$t \rightarrow \min(\max)$$

- Пусть $H \in \mathbb{S}^n$ — матричная переменная, b — заданный n -мерный столбец,

$$b^T H b \rightarrow \min(\max)$$

- Пусть $H \in \mathbb{S}^n$ — матричная переменная, tr — след матрицы

$$\text{tr } H \rightarrow \min(\max)$$

- Пусть $H \in \mathbb{S}^n$ — матричная переменная, B — матричный коэффициент

$$\text{tr}(B^T H B) \rightarrow \min(\max)$$

Максимум и минимум

Очевидно случай

$$d^T x \rightarrow \max$$

сводится к случаю

$$c^T x \rightarrow \min$$

при $c = -d$. Поэтому достаточно иметь функцию для исследования на минимум.

Число скалярных переменных

Функция `decnbr`:

```
ndecv = decnbr(lmisys)
```

возвращает число неизвестных **скалярных** переменных (decision variables) в системе. Здесь `lmisys` — вектор внутреннего представления системы.

Функция `decinfo` — структура матричной переменной

```
decH = decinfo(lmisys,H)
```

Здесь `H` — имя матричной переменной, `decH` — матрица описания ее структуры (шаблон).

Например:

```
>> decH=decinfo(lmisys,H)
```

```
decH =
 3   4
 4   5
```

Означает, что H имеет структуру:

$$H = \begin{bmatrix} x_3 & x_4 \\ x_4 & x_5 \end{bmatrix}.$$

Функция `defcx`

Перед решением задачи SDP требуется представить целевую функцию в виде линейной формы $c^T x$, где x — вектор скалярных переменных. Функция `defcx` позволяет найти коэффициенты этой формы, т.е. найти вектор c .

Для этого матричная переменная H представляется в виде:

$$H = \sum_{j=1}^m x_j V_j,$$

где x_j — скалярные переменные, V_j — постоянные квадратные матрицы (некоторые V_j могут быть равны нулю).

Функция `defcx` позволяет вычислить матрицы V_j в этом представлении.

Функция `defcx` — пояснение

Пусть

$$x = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6]^T, \quad H = \begin{bmatrix} x_3 & x_4 \\ x_4 & x_5 \end{bmatrix}$$

Тогда H представляется в виде:

$$H = \sum_{j=1}^6 x_j V_j,$$

где

$$V_1 = V_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad V_3 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},$$

$$V_4 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad V_5 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad V_6 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Заголовок функции defcx

Пусть H — матричная переменная, x_1, \dots, x_m — скалярные переменные, $1 \leq k \leq m$.

$$V_k = \text{defcx}(\text{lmisys}, k, H)$$

Функция возвращает матрицу V_k в представлении

$$H = \sum_{j=1}^m x_j V_j,$$

Более общий вид:

$$[V_k^{(1)}, \dots, V_k^{(r)}] = \text{defcx}(\text{lmisys}, k, H_1, \dots, H_r)$$

Здесь H_1, \dots, H_r — матричные переменные,

$$H_i = \sum_{j=1}^m x_j V_j^{(i)}, \quad i = 1, \dots, r.$$

Вычисление вектора с: пример 1

Пусть требуется минимизировать значение неизвестной переменной t , которая описывает некоторое скалярное числовое значение.

Пусть \mathbf{x} — вектор скалярных переменных. Тогда $\exists k : x_k = t$, тогда

$$t = c^T x, \quad c = [0, \dots, 0, \underbrace{1}_k, 0, \dots, 0]^T;$$

Как вычислить c ? (k может быть неизвестно)

Вычисление вектора с: пример 1 (продолжение)

Пусть $t = x_k$, где k заранее неизвестно.

Способ 1:

```
m = decnbr(lmisys);
c = zeros(m,1); % preallocation
for j = 1:m
    c(j) = defcx(lmisys, j, t);
end
```

Здесь preallocation — предварительное выделение памяти для массива, что ускоряет работу цикла.

Способ 2:

```
m = decnbr(lmisys);
k = decinfo(lmisys, t);
c = zeros(m,1);
c(k) = 1;
```

Вычисление вектора c : пример 1 (продолжение)

Пусть переменная t создана [первой](#). Тогда $t = x_1$.

[Способ 3:](#)

```
m = decnbr(lmisys);
c = zeros(m,1);
c(1) = 1;
```

Пусть переменная t создана [последней](#). Тогда $t = x_m$, m — число скалярных переменных.

[Способ 4:](#)

```
m = decnbr(lmisys);
c = zeros(m,1);
c(m) = 1;
```

Вычисление вектора c : пример 2

Пусть требуется

$$b^T H b \rightarrow \min,$$

где b — заданный вектор, H — матричная переменная.

Представим

$$H = \sum_{j=1}^m x_j V_j,$$

где x_j — скалярные переменные, V_j — постоянные матрицы.

Тогда

$$b^T H b = \sum_{j=1}^m x_j b^T V_j b,$$

т.е. $b^T H b = c^T x$, где $c_j = b^T V_j b$.

Функция `defcx`. Пример 2

Представим в виде линейной формы вида $c^T x$ выражение $b^T H b$, где H — матричная переменная, b — заданный вектор. Коэффициенты вектора c могут быть найдены следующим образом:

```
m = decnbr(lmisys);
c = zeros(m,1); % preallocation
for j = 1:m
    Vj = defcx(lmisys,j,H);
    c(j) = b'*Vj*b;
end
```

Вычисление вектора c : пример 3

Пусть требуется

$$\text{tr}(B'HB) \rightarrow \min,$$

где B — заданная матрица.

Вновь

$$H = \sum_{j=1}^m x_j V_j,$$

В силу линейности следа матрицы,

$$\text{tr}(B^T HB) = \sum_{j=1}^m x_j \text{tr}(B^T V_j B),$$

т.е. $\text{tr}(B^T HB) = c^T x$, где $c_j = \text{tr}(B^T V_j B)$.

Функция `defcx`. Пример 3

Представим в виде линейной формы вида $c^T x$ выражение $\text{tr}(B^T HB)$, где H — матричная переменная, B — заданная матрица. Коэффициенты вектора c могут быть найдены следующим образом:

```
m = decnbr(lmisys);
c = zeros(m,1);
for j = 1:m
    Vj = defcx(lmisys,j,H);
    c(j) = trace(B'*Vj*B);
end
```

Вызов солвера для решения задачи SDP

Солвер задачи SDP вызывает функция `mincx` (вычисление минимума).

В пакете LMILAB нет опции выбора солвера — имеется единственный вариант.

Возможны два случая, когда задача не имеет решения:

- Множество допустимых векторов пусто.
- Множество допустимых векторов непусто, но целевая функция неограничена снизу.

Функция `mincx`

Вызов функции:

```
[copt,xopt] = mincx(lmisys,c,options,xinit,target)
```

- `lmisys` — внутреннее описание системы неравенств-ограничений.
- `c` — вектор-столбец того же размера, что и x . Для получения `c` обычно используется функция `defcx`.
- `options` (не обязательен) — одномерный массив, который может включать до пяти элементов (см. таблицу).

Параметр `options`

<code>options(1)</code>	Относительная точность вычисления оптимального значения $c^T x$ (по умолчанию — 0.01).
<code>options(2)</code>	Максимальное число итераций (по умолчанию — 100).
<code>options(3)</code>	Вещественное значение R . Предполагается, что присутствует ограничение $\ x\ < R$, где $\ \cdot\ $ — евклидова норма вектора. По умолчанию $R = 10^9$. Значение $R < 0$ означает отсутствие каких-либо ограничений.
<code>options(4)</code>	Целое число $L > 1$. Вычисления прекращаются, когда за L последних итераций величина $c^T x$ уменьшилась не более, чем на относительную величину, указанную в параметре <code>options(1)</code> . По умолчанию $L = 10$.
<code>options(5)</code>	Если придать этому параметру ненулевое значение, трассировка результатов промежуточных итераций отключается.

Функция `mincx` (продолжение)

- `xinit` задает начальное приближение оптимальной точки x . Если начальное приближение не задано, его можно положить пустым (т.е. указать пустые квадратные скобки). Если этот параметр не удовлетворяет системе ограничений, то он игнорируется.
- `target` — необязательный параметр, который определяет правило остановки вычислений. Итерации прекращаются, если найдено значение x , которое удовлетворяет ограничениям $L(x) < R(x)$ и такое, что $c^T x < \text{target}$. По умолчанию $\text{target} = -10^{20}$.
Служит для обнаружения случая, когда множество допустимых векторов непусто, но функция $c^T x$ на этом множестве неограничена снизу.

Функция `mincx` (продолжение)

Функция возвращает два значения:

- `copt` — глобальный минимум целевой функции $c^T x$
- `xopt` — оптимальное значение вектора скалярных переменных x . Для того, чтобы извлечь из него матричные переменные, требуется применить функцию `dec2mat`.

В случае, если система ограничений не имеет решения (множество допустимых переменных пусто), оба выходных параметра принимают пустое значение [].

Пример. Оценка, связанная с неравенством Ляпунова

Рассмотрим задачу:

$$\begin{aligned} \varepsilon &\rightarrow \max, \\ HA + A^T H &< -\varepsilon I, \quad 0 < H < I. \end{aligned}$$

Здесь A — заданная квадратная матрица, H — симметричная матричная переменная, ε — скалярный параметр.

Для того, чтобы свести задачу к задаче на минимум, введем новую переменную $\alpha = -\varepsilon$. Получаем:

$$\begin{aligned} \alpha &\rightarrow \min, \\ HA + A^T H &< \alpha I, \quad 0 < H < I. \end{aligned}$$

Пример (продолжение)

```
function [H0,eps] = lin1(A)
n=size(A,1);
H0 = [ ];
eps = [ ];
setlmis([ ]);
alpha = lmivar(1,[1 1]);
[H, ndec] = lmivar(1,[n 1]);
Hp = newlmi; % неравенство H > 0
lmiterm([-Hp 1 1 H],1,1);
Hn=newlmi; % неравенство H < I
lmiterm([Hn 1 1 H],1,1);
lmiterm([-Hn 1 1 0],eye(n));
LI=newlmi; % первое неравенство
lmiterm([LI 1 1 H],1,A,'s');
lmiterm([-LI 1 1 alpha],1,eye(n));
lmisys=getlmis;
```

Пример (продолжение функции)

```
c=zeros(ndec,1);
c(1)=1;
[alpha0,x]=mincx(lmisys,c,[0.01 100 1e9 10 0]);
if ~isempty(alpha0)
    H0 = dec2mat(lmisys,x,H);
    eps = -alpha0;
end
```

Вызовем эту функцию:

```
function main_lin
A=[-4 3; 1 -3];
[H,eps] = lin1(A);
H
eps
```

Пример: результаты работы

```
Solver for linear objective minimization under LMI constraints
Iterations : Best objective value so far
 1           -2.451855
 2           -2.846820
```

```
***          new lower bound: -3.743855
      3          -2.998704
***          new lower bound: -3.119452
      4          -3.029504
***          new lower bound: -3.058284
Result: feasible solution of required accuracy
best objective value: -3.029504
guaranteed relative accuracy: 9.50e-03
f-radius saturation: 0.000% of R = 1.00e+09
H =
  0.4751   0.2971
  0.2971   0.8316
eps =
  3.0295
```

Лекция 9. Пакет YALMIP: обзор и описание задачи

Общая характеристика пакета

Пакет YALMIP (ранее — Yet Another LMI Parser), разработан Юханом Лёффбергом, университет Линчёпинга, Швеция.

- интерфейсный пакет, поддерживает работу с ≈ 30 решателями (линейное программирование, выпуклое программирование, SDP и др.)
- представляет собой тулбокс для MATLAB
- freeware, загружается со страницы <http://yalmip.github.io>
- регулярно обновляется
- поддерживает бесплатные солверы SDP: CSDP, DSDP, SEDUMI, SDPT3, PENLAB, LOGDETTPA, SCS, SDPA, SDPLR, SDPNAL
- поддерживает коммерческие солверы SDP: LMILAB, PENBMI, PENSDP, MOSEK

Общая характеристика пакета (продолжение)

За редким исключением, YALMIP не содержит солверы — только интерфейс с ними. Солверы нужно скачивать и устанавливать отдельно. Не используйте с YALMIP солвер пакета LMILab! Хотя формально это возможно, но может привести к ошибкам.

Рекомендуемая конфигурация:

YALMIP + SeDuMi

SeDuMi (primal-dual solver): <http://sedumi.ie.lehigh.edu/>

Указание

Оба пакета следует разархивировать в свои папки и установить к ним пути. Эти папки нужно хранить отдельно от стандартных папок MATLAB.

Установка дополнительных пакетов в системе MATLAB

Для работы с нестандартными пакетами требуется установить пути к папкам пакета (кроме папок документации).

MATLAB хранит информацию о путях к установленным пакетам в файле

`.\toolbox\local\pathdef.m`

Модификацию этого файла можно выполнить интерактивно или программно (с помощью функции `addpath`).

Интерактивное указание путей

- Выберем пункт меню Home\SetPath. Появляется диалоговое окно.
- Для добавления нового пути следует нажать либо кнопку **Add Folder** (добавить папку) или **Add with Subfolders** (добавить папку вместе с вложенными в нее папками).
- Появляется новое диалоговое окно с заголовком «Browse for Folder», содержащее дерево папок. Выберем в нем папку, путь до которой мы хотим установить, и нажмем кнопку **OK**.
- Нажимаем кнопку **Save**.
- Для удаления путей из списка служит кнопка **Remove**.

Интерактивное указание путей: YALMIP

Для работы с YALMIP нужно добавить пути к папкам:

```
/YALMIP-master  
/YALMIP-master/extras  
/YALMIP-master/solvers  
/YALMIP-master/modules (с подпапками)  
/YALMIP-master/operators
```

Особенности установки солвера SDPT-3

Это один из самых популярных солверов, который также имеет смысл установить.

- Скачать архив с дистрибутивом со страницы разработчика:

```
https://blog.nus.edu.sg/mattohkc/softwares/sdpt3/
```

- Разархивировать дистрибутив в отдельную папку, войти в MATLAB и сделать эту папку текущей.
- Выполнить

```
>> Installmex(1)
```
- Добавить маршруты к папке в режиме Add with Subfolders.
- Не забудьте сохранить эти маршруты (кнопка Save).

Как проверить, какие солверы установлены в MATLAB?

Выполните команду:

```
>> yalmiptest
```

В списке должны появиться (с отметкой found) солверы LMILAB, SDPT3 (если установлен), SEDUMI (если установлен).

Нажатие следующей клавиши приведет к тестированию солверов.

Структура пакета YALMIP

YALMIP имеет объектную организацию.

Основной класс `sdpvar` включает 276 методов — см. папку `@sdpvar` (включая — перегруженные функции и операции MATLAB).

Дополнительные классы (папка `extra`): `blkvar`, `lmi`, `constraint`, `logdet` и др.

Имеется ряд внешних функций, которые не входят в какие-либо классы.

Описание системы LMIs

Как в случае LMILab, создание системы LMIs распадается на этапы:

- описание матричных переменных,
- задание ограничений (неравенств и равенств).

Описание матричных переменных

Матричные переменные — объекты класса `sdpvar`.

Создаются путем вызова конструктора класса:

```
X = sdpvar(m,n,matrixtype,datatype)
```

Здесь `X` — матрица $m \times n$.

Если `n` не указан, матрица считается квадратной ($n = m$).

Если `matrixtype` не указан, матрица считается симметричной при $m = n$.

Если `datatype` не указан, матрица считается вещественной.

Пример. Определим:

```
H = sdpvar(5); % симметричная веществ. 5x5
a = sdpvar(1); % веществ. скаляр
a = sdpvar; % то же
B = sdpvar(5,6); % веществ. 5x6
```

Описание матричных переменных: структура матрицы

Параметр `matrixtype` — структура матрицы, его значения:

- `'symmetric'` — симметричная (по умолчанию при $m = n$),
- `'full'` — произвольная (по умолчанию при $m \neq n$),
- `'hankel'` — ганкелева,
- `'toeplitz'` — тэплицева,
- `'skew'` — кососимметричная ($A^T = -A$),
- `'hermitian'` — эрмитова ($(\bar{A})^T = A$).

Сокращенные значения: `'sy'`, `'f'`, `'ha'`, `'t'`, `'sk'`, `'he'`.

Справка

Ганкелева матрица — симметричная $n \times n$ -матрица A , на диагоналях, перпендикулярных главной диагонали, все элементы одинаковы:

$$A = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_n \\ a_2 & a_3 & a_4 & \cdots & a_{n+1} \\ a_3 & a_4 & a_5 & \cdots & a_{n+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_n & a_{n+1} & a_{n+2} & \cdots & a_{2n-1} \end{bmatrix}$$

Тёплицева матрица — $n \times n$ -матрица A , на диагоналях, параллельных главной диагонали, элементы одинаковы:

$$A = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-n+1} \\ a_1 & a_0 & a_{-1} & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & & \ddots & a_1 & a_0 & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}$$

Пример

Обратите внимание:

```
M = sdpvar(3,3);
K = sdpvar(3,3,'full');
```

M — симметричная матрица 3×3 (6 скалярных переменных), K — произвольная матрица 3×3 (9 скалярных переменных)

Описание матричных переменных: тип данных

Параметр `datatype` — тип данных элементов матрицы, значения:

- `'real'` — вещественная (по умолчанию),
- `'complex'` — комплексная

Пример. Определим $K \in \mathbb{C}^{2 \times 3}$ произвольной структуры:

```
K = sdpvar(2, 3, 'full', 'complex');
```

или

```
K = sdpvar(2, 3, [ ], 'complex'); % 'full' по умолч.
```

Создаются 12 скалярные переменные (6 вещественных и 6 мнимых частей).

Описание матричных переменных (продолжение)

Кроме конструктора класс **sdpvar** имеет более 200 методов.

Их полный список можно получить с помощью команды

```
methods('sdpvar')
```

Они задают перегрузку стандартных функций и операций языка MATLAB, предназначенных для работы с матрицами.

В частности перегружены операции сложения, вычитания, умножения матриц, операции отношения, операции транспонирования и индексации, функции **trace**, **sum**, **diag**, **det** и др.

Таким образом, работа с матричными переменными, которые являются объектами класса **sdpvar**, почти не отличается от работы со стандартными матрицами MATLAB.

Как задать переменную в виде диагональной матрицы

Если требуется задать диагональную переменную, определите неизвестный вектор **D** подходящего размера (диагональ), например,

```
D = sdpvar(n,1); % столбец n x 1
```

а потом задайте

```
H = diag(D);
```

Функция blkvar

Служит для описания симметричной матрицы. Позволяет не описывать симметричные блоки (как в LMILab):

```
X = blkvar;
X(1,1) = M;
X(1,2) = N;
X(2,2) = zeros(m,m);
X = sdpvar(X);
```

Здесь **M**, **N** могут быть постоянными, переменными или выражениями.

Симметричные блоки заполняются автоматически:

$$X = \begin{bmatrix} M & N \\ N^T & 0 \end{bmatrix}$$

Описание матричных переменных (продолжение)

Если тип элементов матрицы отличен от вещественного или комплексного, вместо функции **sdpvar** используем другие функции:

- **intvar** — целые значения,
- **binvar** — значения 0 и 1.

Функция assign

Используется для присваивания значений матричным переменным. Ее вызов:

```
assign(X, val)
```

Здесь X — объект класса `sdpvar`, val — обычная матрица MATLAB.

Как правило `assign` используют перед вызовом решателя для задания начальных приближений.

Заметим, что, если скалярные переменные, входящие в X , входят и в другие матричные переменные, то эти матричные переменные также будут изменены.

Функция assign (пример)

Пример. Последовательность команд

```
P = sdpvar(3);
Q = sdpvar(2);
assign(P, eye(3));
assign(Q, 0);
```

создает матрицы $P \in \mathbb{S}^3$ и $Q \in \mathbb{S}^2$, инициализирует их единичной и нулевой матрицей.

Можно проверить:

```
value(P)
value(Q)
```

Функция `value` преобразует тип `sdpvar` в стандартный матричный тип MATLAB.

Функция fixsdpvar

Позволяет зафиксировать часть скалярных переменных, входящих в матричную переменную, задав их постоянные значения.

Вызов функции

```
Z = fixsdpvar(X, Y, W)
```

заменяет вхождения объекта Y (класса `sdpvar`) в объект X на постоянное значение W .

Пример. В результате последовательности команд

```
P = sdpvar(3);
P = fixsdpvar(P, diag(P), [1; 1; 1]);
```

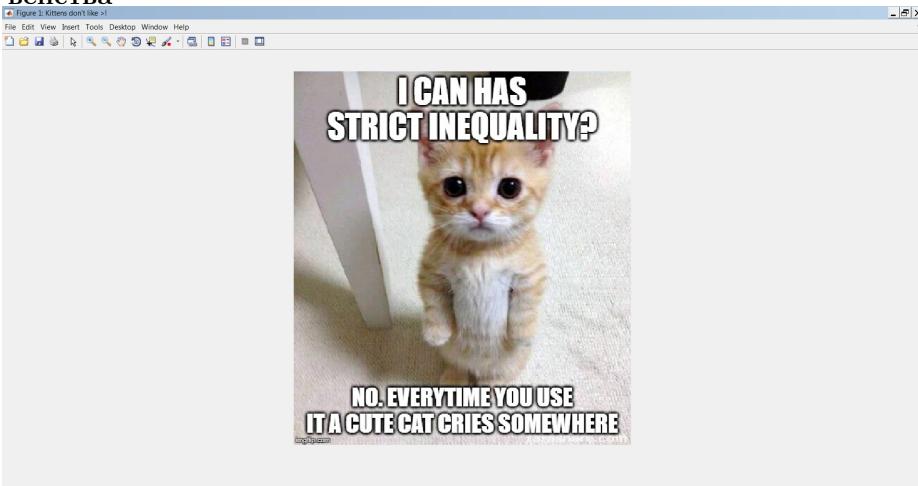
получаем

$$P = \begin{bmatrix} 1 & x_1 & x_2 \\ x_1 & 1 & x_3 \\ x_2 & x_3 & 1 \end{bmatrix}$$

Создание системы ограничений (общие сведения)

- Ограничения — объекты класса `lmi`, описываются с помощью конструктора `[]`.
- В YALMIP можно использоваться два вида ограничений: типа неравенств и типа равенств.
- YALMIP не делает различий между строгими и нестрогими неравенствами — все неравенства понимаются как [нестрогие](#).

Диагностика при попытке создать ограничение типа строгого неравенства



Как задать строгое неравенство (если очень нужно)?

Зададим $\varepsilon > 0$ — малое число. Например,

```
eps = 1E-5;
```

Пусть $F(x) \in \mathbb{S}^n$, I_n — единичная матрица (`eye(n)`). Тогда можно заменить:

$$\begin{aligned} F(x) \geq 0 &\implies F(x) \geq \varepsilon I_n, \\ F(x) \leq 0 &\implies F(x) \leq -\varepsilon I_n, \end{aligned}$$

Пусть $Q(x) \in \mathbb{R}^{m \times n}$, $U_{mn} — m \times n$ -матрица из единиц (`ones(m,n)`). Тогда можно заменить:

$$\begin{aligned} Q(x) \geq 0 &\implies Q(x) \geq \varepsilon U_{mn}, \\ Q(x) \leq 0 &\implies Q(x) \leq -\varepsilon U_{mn}, \end{aligned}$$

Задание ограничений

Оператором

`F = [];`

создается «пустая» задача, ограничения в которую могут быть добавлены позднее.

При описании ограничений можно использовать знаки операций отношений: `==`, `>=`, `<=`.

В случае сравнения симметричных и эрмитовых матриц операции `>=` и `<=` понимаются в смысле положительной (отрицательной) определенности, в остальных случаях — как поэлементное сравнение.

Задание ограничений (примеры)

Пример. Создадим переменную $H \in \mathbb{S}^n$:

`H = sdpvar(n);`

- Ограничение $H \geq 0$ (в смысле положительной определенности) задается:

`F = [H >= 0];`

- Ограничение $H > 0$ (гарантированно):

`eps = 1E-6;`
`F = [H >= eps * eye(n)];`

- Неотрицательность всех элементов матрицы H :

`F = [H(:) >= 0];`

Задание ограничений (примеры)

Пример. Определим прямоугольную (не квадратную) матрицу

`M = sdpvar(5,6);`

Тогда ограничение

`F = [M >= 0];`

будет означать уже не неотрицательную определенность, а неотрицательность всех элементов матрицы M .

Задание ограничений (примеры)

Пример. Ограничение $HA + A^T H \leq 0$, где A — квадратная матрица, можно задать в виде:

`F = [H * A + A' * H <= 0];`

Пример. Ограничение $Hb = c$, где b, c — матрицы, задается в виде

`F = [H * b == c];`

Задание системы ограничений

Пример. Пусть требуется задать систему ограничений вида

$$H \geq 0, \quad HA + A^T H \leq 0, \quad Hb = c.$$

Используем перегруженную операцию сложения:

1-й способ:

$$F = [H * A + A' * H \leq 0] + [H * b == c] + [H \geq 0];$$

2-й способ:

$$\begin{aligned} F &= [H * A + A' * H \leq 0]; \\ F &= F + [H * b == c]; \\ F &= F + [H \geq 0]; \end{aligned}$$

3-й способ:

$$F = [H * A + A' * H \leq 0, H * b == c, H \geq 0];$$

Задание системы ограничений: двойные неравенства

Для задания нескольких ограничений можно использовать двойные неравенства, например,

$$F = [0 \leq H \leq \text{eye}(3)];$$

означает, что $0 \leq H \leq I_3$.

Задание ограничений: замечание

Напомним особенности операций отношения в языке MATLAB. (Теми же свойствами обладают и перегруженные операции \geq , \leq , \neq для объектов класса `sdpvar`.)

- Сравнивать можно либо матрицы одного размера, либо матрицу и скалярное значение. В остальных случаях при несовпадении размеров матриц-операндов, выдается сообщение об ошибке.
- Если один из operandов скалярный, а другой — матричный, то перед сравнением скаляр распространяется до матрицы подходящего размера, все элементы которой одинаковы. Например:

$$\begin{aligned} H &= \text{sdpvar}(2); \quad \% \text{ симм. матрица } 2 \times 2 \\ F &= [H \leq 1]; \end{aligned}$$

означает, что $H \leq \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$.

Лекция 9а. Пакет YALMIP: решение задачи и примеры

Решение задачи SDP

Для решения SDP и ряда других задач используется функция `optimize`. Это внешняя функция, т.е. она не является методом какого-либо класса.

```
diagnostics = optimize(F,h,options)
```

Все три входные параметра необязательны.

Замечание. Если необязательный параметр является последним, он может быть опущен. Необязательный параметр, стоящий в начале или в середине списка параметров, может быть заменен пустыми квадратными скобками [].

Функция `optimize` (параметры)

- `F` — массив ограничений (объект класса `lmi`).
- `h` — объект класса `sdpvar` или `logdet`, описывающий целевую функцию. В процессе решения эта функция минимизируется.
Параметр `h` может быть опущен, если исследуется задача разрешимости системы ограничений (FEASP).
- `options` — структура, описывающая параметры оптимизации.

Функция `optimize` (`options`)

- Если положить

```
options = sdpsettings;
```

то будут использованы параметры, задаваемые по умолчанию.

- Для всех параметров, не заданных явно, сохраняются их стандартные значения.
- Для изменения стандартных параметров указывают список: имя параметра (`name`) в виде строки и его значение (`value`).

```
options = sdpsettings('name1',value1,  
                      'name2',value2,...);
```

Функция `optimize` (`options` — параметры)

Укажем наиболее важные из параметров.

- `'solver'` — имя программы-решателя. Для задач SDP:

```
'sdpt3', 'sedumi', 'sdpa',  
'pensdp', 'penbmi', 'mosek',  
'csdp', 'dsdp', 'maxdet',  
'lmilab', 'sdplr', 'kypd'.
```

Если решатель не указан явно, то функция `optimize` сама пытается подобрать подходящий решатель из числа, установленных в системе MATLAB.

- `'verbose'` — характеризует уровень подробности сообщений выдаваемых решателем. Может принимать целые неотрицательные значения (0 — минимальное число сообщений, 1 — стандартные сообщения, ≥ 2 — подробные сообщения). Значение по умолчанию равно 1. Содержание сообщений зависит от используемого солвера.

Функция `optimize (options — параметры)`

- `'radius'` — вещественное число, которое задает ограничения для вектора скалярных переменных (вида $\|x\| < R$). По умолчанию `Inf` (бесконечность в системе MATLAB).

Функция `optimize (options — параметры)`

- `'removenequalities'` — с помощью этого режима можно на этапе предварительной обработки (до вызова решателя) исключить ограничения типа равенств. Это особенно существенно для решателей, которые не допускают такой вид ограничений (например, `lmilab`). Параметр может принимать значения $-1, 0, 1, 2$ (по умолчанию 0).
 - Значение -1 предписывает заменить равенство на два неравенства (вместо $a = b$ получаем $a \geq b$ и $a \leq b$),
 - 0 — не выполнять никаких преобразований,
 - $1, 2$ — YALMIP устраниет равенства, уменьшая число скалярных переменных с помощью двух разных численных алгоритмов (один — более устойчив, другой использует разреженность).

Функция `optimize (options — параметры)`

- `'usex0'` — число 0 или 1 (0 по умолчанию). Единица означает, что заданные до вызова решателя значения переменных следует использовать в качестве начальных приближений. Для задания начальных значений переменных можно применить функцию `assign`.

Замечание. Не все солверы поддерживают выбор начальной точки. Солвер `SeDuMi` такую опцию не поддерживает, возникает ошибка.

Функция `optimize (выходные параметры)`

Выходной параметр `diagnostics` представляет собой структуру, содержащую диагностические сообщения. Она имеет поля:

- `yalmiptime` — время (в секундах), затраченное функциями пакета YALMIP на решение задачи (без учета времени работы внешнего решателя);
- `solvetime` — время работы решателя;

- `info` — формируемый функцией `yalmiperror` текст сообщения об ошибке (если ошибки нет, обычно выводится сообщение «`No problems detected`»);
- `problem` — код ошибки (целое число от -5 до 13); при нормальной работе имеет значение 0 (расшифровку кода можно найти в описании функции `yalmiperror`).

Функция `optimize` (получение результата)

Найденные значения целевой функции и матричных переменных могут быть извлечены из этих же переменных с помощью функции `value`:

```
X = value(P)
```

Здесь `P` — объект класса `sdpvar`.

Замечание для программиста

Каким образом результаты работы функции `optimize` хранятся в переменных, которые не являются выходными параметрами этой функции?

Конструктор класса `sdpvar` содержит локальные статические переменные, т.е. переменные, значения которых сохраняются между вызовами функции.

(В языке MATLAB локальные статические переменные описываются с помощью оператора `persistent`.)

Конструктор `sdpvar` часто вызывают не для создания новых объектов, а для чтения или изменения этих переменных.

Именно в статических переменных и хранятся результаты работы функции `optimize`.

Пример. Неравенство Ляпунова

Задача разрешимости относительно $H \in \mathbb{S}^n$:

$$HA + A^T H < 0, \quad H > 0.$$

При работе с YALMIP неравенства заменяются на:

$$HA + A^T H \leq 0, \quad H \geq 0.$$

Проблема: последняя система [всегда](#) имеет решение $H = 0$. Решение проблемы: рассмотреть систему

$$HA + A^T H \leq -\varepsilon I_n, \quad H \geq \varepsilon I_n,$$

где $\varepsilon > 0$ — некоторое число. (Поскольку система однородна относительно H , то даже нет необходимости выбирать ε малым.)

Неравенство Ляпунова: программа

```
function H0 = lyap(A)
n=size(A,1);
H=sdpvar(n);
H0 = [];
eps = 1e-5;
```

```

F = [H*A+A'*H <= -eps*eye(n)]+ [H >= eps*eye(n)];
options = sdpsettings('solver','sedumi','verbose',0);
diagn=optimize(F,[],options);
if diagn.problem == 0
    H0 = value(H);
end

```

Неравенство Ляпунова: вызов функции

```

function main_lyap
clc;
A=[-2 1; -2 -3];
H=lyap(A);
if ~isempty(H)
    H
else
    disp ('Problem is infeasible')
end

```

Результат:

```

H =
 0.4162   -0.0308
 -0.0308    0.2830

```

Неравенство Ляпунова: подробная диагностика

```

options = sdpsettings('solver','sedumi','verbose',1);

Получаем:

SeDuMi 1.3 by AdvOL, 2005-2008 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, theta = 0.250, beta = 0.500
eqs m = 3, order n = 5, dim = 9, blocks = 3
nnz(A) = 10 + 0, nnz(ADA) = 9, nnz(L) = 6
it :      b*y      gap      delta      rate      t/tP*      t/tD*      feas      cg      cg      prec
0 :          5.78E+01  0.000
1 :    0.00E+00  1.15E+01  0.000  0.1984  0.9000  0.9000    1.00    1    1    3.2E+00
2 :    0.00E+00  4.06E-01  0.000  0.0354  0.9900  0.9900    1.00    1    1    1.1E-01
3 :    0.00E+00  2.78E-05  0.000  0.0001  1.0000  1.0000    1.00    1    1    7.7E-06
4 :    0.00E+00  2.80E-12  0.000  0.0000  1.0000  1.0000    1.00    1    1    7.8E-13

iter seconds digits      c*x              b*y
        4       0.1    7.6 -2.3811743692e-18  0.0000000000e+00
|Ax-b| =  4.3e-13, [Ay-c]_+ =  0.0E+00, |x|=  1.3e-13, |y|=  5.0e-01

Detailed timing (sec)
      Pre           IPM           Post
1.600E-02    6.200E-02    0.000E+00
Max-norms: ||b||=0, ||c|| = 1.000000e-05,
Cholesky |add|=0, |skip| = 0, ||L.L|| = 1.
```

Неравенство Ляпунова: отсутствие решений

Если взять негурвицеву матрицу

```
A=[-2 1; -2 -3];
```

то получим Problem is infeasible.

Но, если положить $\text{eps} = 0$, то и в этом случае получим непустую матрицу H . При этом

```

eig(H):          eig(HA + A'H):
-0.0000          -2.6413
 0.5156          -0.0000

```

Пример. Круговой критерий абсолютной устойчивости

Нужно проверить разрешимости системы LMIs относительно неизвестной $H \in \mathbb{S}^n$:

$$\begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} < \begin{bmatrix} \mu_1 \mu_2 c c^T & -\frac{1}{2}(\mu_1 + \mu_2)c \\ -\frac{1}{2}(\mu_1 + \mu_2)c^T & 1 \end{bmatrix}, \quad H > 0.$$

Пример. Круговой критерий абсолютной устойчивости

```

function H0 = circle(A,b,c,mu1,mu2)
n = size(A,1);
H0 = [];
H = sdpvar(n);
R=[mu1*mu2*c*c' -0.5*(mu1+mu2)*c; -0.5*(mu1+mu2)*c' 1];
eps = 1E-6;
F=[[H*A+A'*H H*b; b'*H 0]<= R - eps*eye(n+1),...
     H >= eps*eye(n)];
options = sdpsettings('solver','sdpt3','verbose',0);
diagn = solvesdp(F,[],options);
if diagn.problem == 0
    H0 = value(H);
end

```

Круговой критерий абсолютной устойчивости (вариант кода)

```

H=sdpvar(n);
R = blkvar;
R(1,1) = mu1*mu2*c*c';
R(1,2) = -0.5*(mu1+mu2)*c;
R(2,2) = 1;
R = sdpvar(R);

L = blkvar;
L(1,1) = H*A+A'*H;
L(1,2) = H*b;
L(2,2) = 0; % необходимо
L = sdpvar(L);
F=[L <= R - eps*eye(n+1), H >= eps*eye(n)];

```

Круговой критерий абсолютной устойчивости: вызов функции

```

function main_circle
clc;
mu1=-0.1;

```

```

mu2=0.1;
w=tf(1,[1 2 3]);
[A,b,c]=ssdata(w);
c=-c';
H = circle(A,b,c,mu1,mu2);
if ~isempty(H)
    disp('Problem is feasible');
    disp(H);
else
    disp('Problem is not feasible');
end

```

Линейные целевые функции

- Требуется $t \rightarrow \min$, t — скаляр:

```
diagn=optimize(F,t,options)
```

- Требуется $b^T H b \rightarrow \max$:

```
diagn=optimize(F,-b'*H*b,options)
```

- Требуется $\text{tr } H \rightarrow \max$:

```
diagn = optimize(F,-trace(H),options)
```

- Требуется $\sum_{i,j} H_{ij} \rightarrow \min$:

```
diagn = optimize(F,sum(sum(H)),options)
```

Пример.

Пусть A, B, C — коэффициенты размеров $n \times n$, $n \times m$, $n \times l$. Переменные: $P \in \mathbb{S}^n$, $r \in \mathbb{R}$, $r > 0$. Решить задачу:

$$r \rightarrow \min, \quad \begin{bmatrix} PA + A^T P + CC^T & PB \\ B^T P & -r^2 I_m \end{bmatrix} < 0, \quad P > 0.$$

Введен новую переменную: $t = r^2$, $t \rightarrow \min$.

Пример: функция

```

function [r,H] = gain(A,B,C)
n = size(B,1);
m = size(B,2);
t = sdpvar(1);
P = sdpvar(n);
r = [];
H = [];

```

```

F = [[P*A+A'*P+C'*C  P*B; B'*P -t*eye(m)] <= 0];
F = F + [P >= 0];
options = sdpsettings('solver','sedumi','verbose',0);
diagn = optimize(F,t,options);
if diagn.problem == 0
    r = sqrt(value(t));
    H = value(P);
end

```

Пример: вызов функции

```

function main_gain
clc;
w = tf(1,[1 2 3]);
[A,B,C] = ssdata(w);
[r, P] = gain(A,B,C)

```

Результат:

```

r =
0.3536

P =
1.0000    0.5000
0.5000    0.7500

```

Лекция 9b. Пакет YALMIP: дополнительные разделы

Задача MAXDET

Пусть требуется решить задачу с целевым условием:

$$\det H \rightarrow \min (\max)$$

Целевая функция не является линейной и попытка вызвать

```
diagn=optimize(F,det(H),options)
```

приводит к ошибке.

Способ решения этой задачи зависит от используемого солвера.

Задача MAXDET (продолжение)

В YALMIP существуют два способа решения задачи MAXDET.

- Только для солвера SDPT3! При $H > 0$

$$\logdet(H) = \ln(\det H)$$

Тогда (для минимума $\det H$)

```
options = sdpsettings('solver','sdpt3')
diagn=optimize(F,logdet(H),options);
```

- Для SeDuMi и SDPT3. Использовать функцию `geomean`: для `sdpvar` это геометрическое среднее собственных чисел

$$\text{geomean}(H) = \sqrt[n]{\lambda_1 \cdots \lambda_n} = \sqrt[n]{\det H}$$

Вызов (для минимума $\det H$):

```
diagn=optimize(F,geomean(H),options)
```

Пример. Нахождение эллипсоида наименьшего объема

Задача 1

Найти эллипсоид вида

$$\Omega = \{x : x^T H x \leqslant 1\}, \quad H > 0,$$

наименьшего объема, содержащий заданный набор векторов $s_i \in \mathbb{R}^n$, $i = 1, \dots, m$.

Объем эллипсоида в \mathbb{R}^n (V_n — объем единичного шара):

$$\text{Vol}(\Omega) = V_n \sqrt{\det H^{-1}}$$

Получаем задачу MAXDET:

$$\det H \rightarrow \max, \quad s_i^T H s_i \leqslant 1, \quad i = 1, \dots, m, \quad H > 0$$

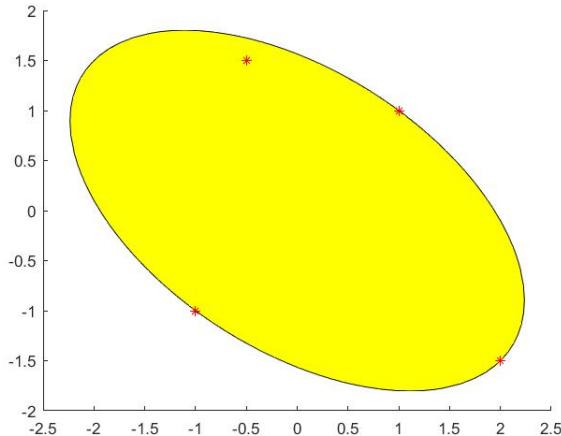
Целевую функцию можно записать в виде:

$$\begin{aligned} \logdet(H) &\rightarrow \max \\ \text{geomean}(H) &\rightarrow \max \end{aligned}$$

Программа к задаче 1

```
clc
s = [1 1; -0.5 1.5; 2 -1.5; -1 -1]';
H = sdpvar(2);
F = [H >= 0];
for i=1:4
    si = s(:,i);
    F = F + [si'*H*si <= 1];
end;
optimize(F,-logdet(H),sdpsettings('solver','sdpt3'));
H0 = value(H)
x = sdpvar(2,1);
plot(x'*H0*x <= 1,x,'y');
for i=1:4 % рисуем точки
    si = s(:,i); hold on;
    plot(si(1),si(2),'r*');
end
```

Программа к задаче 1: результаты счета



Программа к задаче 1 (второй вариант)

В той же программе запишем optimize как

```
optimize(F,-geomean(H),sdpsettings('solver','sdpt3'));
```

или

```
optimize(F,-geomean(H),sdpsettings('solver','sedumi'));
```

Результат будет такой же.

Пример. Нахождение эллипсоида наибольшего объема

Задача 2

Найти эллипсоид вида $\Omega = \{x : x^T H x \leq 1\}$, $H > 0$, наибольшего объема, который содержится в пересечении заданных полос

$$\Omega \subset \bigcap_{i=1}^m \{x : |c_i^T x| \leq 1\}$$

Обозначим $P = H^{-1}$. Получим задачу MAXDET:

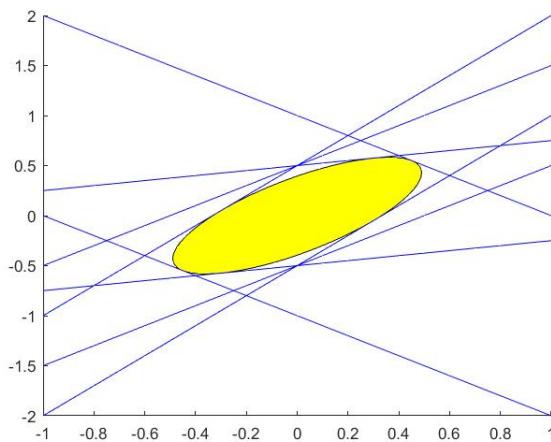
$$\det P \rightarrow \max, \quad c_i^T P c_i \leq 1, \quad i = 1, \dots, m, \quad P > 0$$

Математически, задача сводится к предыдущей.

Программа к задаче 2

```
c = [1 1; -0.5 2; 2 -2; -3 2]';
H = sdpvar(2);
F = [H >= 0];
for i=1:4
    ci = c(:,i);
    F = F + [ci'*H*ci <= 1];
end;
optimize(F,-geomean(H),sdpsettings('solver','sedumi'));
P = inv(value(H));
x = sdpvar(2,1);
plot(x'*P*x <= 1,x,'y'); hold on;
y = -1.:0.1:1.;
for i=1:4 % рисуем прямые
    ci = s(:,i);
    z1 = (1-ci(1)*y)/ci(2);
    z2 = (-1-ci(1)*y)/ci(2);
    plot(y,z1,'b'); hold on;
    plot(y,z2,'b'); hold on;
end
```

Программа к задаче 1: результаты счета



Проверка полученных значений

Чтобы оценить насколько хорошо полученное решение удовлетворяет заданным ограничениям, служит функция `checkset`.

Способы вызова функции:

- `checkset(F)`

Результаты работы функции выводятся на экран в виде таблицы.

- `[P,D] = checkset(F)`

Выходной параметр `P` характеризует ограничения прямой задачи, а выходной параметр `D` — ограничения двойственной задачи.

В качестве параметра функции `F` может использоваться либо массив ограничений, либо выражение вида `F(n)`, где `n` — номер ограничения.

Проверка полученных значений (продолжение)

Возвращаемые значения:

- Для ограничения прямой задачи вида $F(x) \geq 0$, где неравенство понимается в смысле квадратичных или эрмитовых форм, функция `checkset` возвращает минимальное собственное значение $F(x)$.
- Если неравенство $F(x) \geq 0$ понимается как поэлементное, она возвращает минимальный элемент матрицы $F(x)$.
- Для ограничения типа равенства $F(x) = 0$ функция возвращает максимальный из модулей элементов $F(x)$.

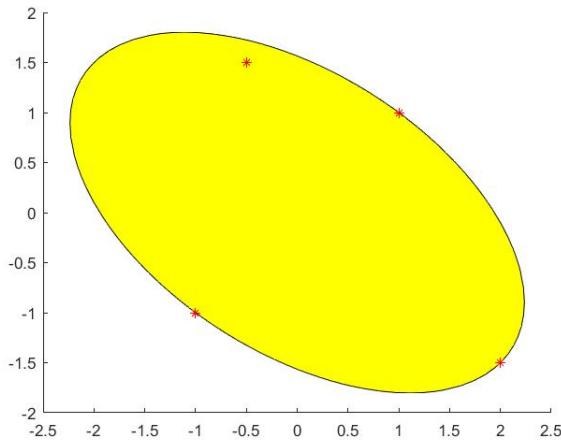
Аналогичная информация выдается для двойственных ограничений.

Проверка для задачи 1

```
+-----+-----+-----+-----+
| ID | Constraint | Primal residual | Dual residual |
+-----+-----+-----+-----+
| #1 | Matrix inequality | 0.07413 | 2.047e-11 |
| #2 | Elementwise inequality | 1.9032e-10 | 0.051641 |
| #3 | Elementwise inequality | 1.8772e-10 | 0.051638 |
| #4 | Elementwise inequality | 0.36 | 2.8145e-11 |
| #5 | Elementwise inequality | 1.0685e-10 | 0.09037 |
+-----+-----+-----+-----+
| A primal-dual optimal solution would show non-negative residuals. |
| In practice, many solvers converge to slightly infeasible          |
| solutions, which may cause some residuals to be negative.         |
| It is up to the user to judge the importance and impact of          |
| slightly negative residuals (i.e. infeasibilities)                  |
| https://yalmip.github.io/command/check/                            |
| https://yalmip.github.io/faq/solutionviolated/                      |
+-----+-----+-----+-----+
```

Ограничения: $H \geq 0$, $s_i^T H s_i \leq 1$, $i = 1, 2, 3, 4$. В столбце Primal residual выводятся наименьшее собственное число H и невязки $1 - s_i^T H s_i$, $i = 1, 2, 3, 4$.

Программа к задаче 1: результаты счета



Вспомогательные команды

- `yalmipdemo` — выводит на экран набор демонстрационных примеров.
- `yalmiptest` — тестирует MATLAB на наличие установленных решателей. После этого проверяет их работоспособность.
- `yalmip('clear')` — очищает рабочее пространство, удаляя из него ограничения и переменные.
- `yalmip('nvars')` — возвращает число скалярных переменных, присутствующих в рабочем пространстве.
- `yalmip('info')` — выводит информацию о переменных и ограничениях, присутствующих в рабочем пространстве.
- `yalmip('solver',s)` — устанавливает в качестве решателя по умолчанию решатель с именем `s` (`s` — строка, ее возможные значения перечислены при описании функции `sdpsettings`). Например:

```
yalmip('solver','sedumi');
```

Лекция 10. Пакет cvx; общие сведения

Общая характеристика пакета

Первая версия пакета предложена в 2005. Разработчики — Майкл Грант (Michael Grant), Стивен Бойд (Stephen Boyd) и Иньюй Е (Yinyu Ye), Стенфордский университет.

Пакет основан на книге:

Boyd S., Vandenberghe L. *Convex Optimization*. Cambridge, UK: Cambridge Univ. Press, 2004.

<http://www.stanford.edu/~boyd/cvxbook/>

cvx содержит набор конструкций специализированного языка моделирования, которые можно использовать наряду с командами MATLAB.

Имеются варианты для работы с языками Python, R, Julia.

Установка cvx

- Скачиваем дистрибутив cvx с сайта

<http://cvxr.com/cvx/download/>

Выбираем подходящий дистрибутив (например, Windows 32-bit). Есть варианты для Linux и Mac. Распаковываем в отдельную папку.

Не помещайте cvx в стандартную матлабовскую папку toolbox!

- Запускаем MATLAB и устанавливаем папку с cvx как текущую (справка Current Folder). Например:

`cd c:\MySoftware\cvx`

- В командной строке MATLAB:

`cvx_setup`

Получаем сообщение об успешной установке.

Установка cvx(продолжение)

- Сохраняем маршруты для последующих сеансов работы: меню

`File/Set Path.../Save`

Замечание

Дистрибутив cvx содержит дистрибутивы двух солверов — SeDuMi и SDPT3, адаптированных для cvx.

Они автоматически устанавливаются вместе с cvx.

При инсталляции вы получаете сообщение о том, что cvx не может установить солвер для линейного программирования GLPK (нужно скачивать отдельно) и коммерческие солверы MOSEK, Gurobi (нет лицензии).

Общая характеристика пакета

Пакет `cvx` (от слова «convex») предназначен для группы задач выпуклого программирования, которые могут быть описаны и решены единообразно. Сюда относятся:

- линейное программирование,
- квадратичное программирование,
- программирование с конусными ограничениями (SOCP),
- полуопределенное программирование ([SDP](#)),
- геометрическое программирование.

Нас будем интересовать часть пакета, относящаяся к SDP.

Disciplined Convex Programming

В пакете реализована концепция [Disciplined Convex Programming](#) (DCP, стандартное выпуклое программирование), предложенная группой С. Бойда.

Выражения, описывающие ограничения и целевую функцию, строятся с использованием

- аффинных, выпуклых или вогнутых функций
- набора преобразований, которые сохраняют аффинность, выпуклость или вогнутость.

Все функции, входящие в задачу, проверяются, удовлетворяют ли они условиям аффинности, выпуклости или вогнутости, согласно заданному набору правил DCP (DCP ruleset). Функции, не удовлетворяющие DCP ruleset, отвергаются.

Правила DCP обеспечивают достаточные (но не необходимые!) условия выпуклости или вогнутости.

Пример

Пример. Функция `x*sqrt(x)` выпукла, но будет отвергнута анализатором `cvx`.

Если записать ее в виде `x^(3/2)`, то она будет признана выпуклой.

Пример: правила DCP для проверки условия аффинности

Выражение считается аффинным, если оно представляет собой:

- константу;
- переменную;
- вызов стандартной функции, про которую известно, что она аффинна;
- сумму или разность аффинных выражений;
- произведение аффинного выражения на константу.

Похожая система правил имеется для проверки выпуклости и вогнутости.

Задачи DCP

DCP допускает три вида задач:

- минимизация выпуклой функции (с ограничениями или без);
- максимизация вогнутой функции (с ограничениями или без);
- проверка разрешимости системы ограничений (задаваемых аффинными, выпуклыми или вогнутыми функциями).

Правила DCP (продолжение)

DCP допускает три вида ограничений:

- ограничение типа равенств ($==$), левая и правая части — аффинные;
- ограничение типа неравенств (\leq), левая часть — выпуклая, правая часть — вогнутая;
- ограничение типа неравенств (\geq), левая часть — вогнутая, правая часть — выпуклая.

Для равенств допускаются комплексные числа.

Неравенства могут пониматься поэлементно или в смысле определенности квадратичных форм.

Предупреждение

Мы рассмотрим только подмножество `cvx`, относящееся к полуопределенному программированию.

Структура кода

Любая модель `cvx` описывается внутри окружения:

```
cvx_begin  
.....  
cvx_end
```

Конструкции внутри окружения — это операторы `cvx` а не операторы MATLAB!

Вне окружения действуют обычные грамматические правила MATLAB.

Структура кода (продолжение)

Мы будем рассматривать `cvx` в режиме SDP (с особыми правилами умолчания). Для работы в режиме SDP окружение имеет вид:

```
cvx_begin sdp  
.....  
cvx_end
```

Отличие режима по умолчанию и режима `sdp`:

для симметричных (в комплексном случае — эрмитовых) матриц в режиме по умолчанию неравенства понимаются как поэлементные, в режиме `sdp` — в смысле определенности форм.

Схема решения оптимизационной задачи

Как и для других программных пакетов нужно пройти следующие этапы:

- описание переменных,
- описание ограничений,
- описание целевой функции (если нужно),
- выбор и настройка солвера,
- вызов солвера,
- анализ успешности работы солвера,
- преобразование решения (если оно найдено) к исходной форме.

Описание переменных

Используются операторы `variable` или `variables`.

По умолчанию — вещественные матрицы. Примеры:

```
variable rho;      % число
variable x(10);    % вектор
variable A(5,6);   % матрица
variable C(5,6,2); % трехмерный массив
variables rho x(10) A(5,6);
```

Ограничение: с командой `variables` нельзя указывать явно тип или структуру матрицы (см. дальше).

Описание переменных (продолжение)

Для переменных можно явно указать тип данных или структуру матрицы, например

```
variable H(10,10) symmetric;
variable Z(5,5) complex hermitian;
```

Часто используемые типы переменных:

<code>symmetric</code>	симметричная
<code>diagonal</code>	диагональная
<code>complex</code>	комплексная
<code>hermitian</code>	эрмитова
<code>scaled_identity</code>	единичная матрица, умноженная на скалярную переменную

Редко используемые типы переменных

<code>lower_triangular</code>	нижняя треугольная
<code>upper_triangular</code>	верхняя треугольная
<code>skew_symmetric</code>	кососимметричная
<code>lower_bidiagonal</code>	нижняя двухдиагональная
<code>upper_bidiagonal</code>	верхняя двухдиагональная
<code>tridiagonal</code>	трехдиагональная
<code>toeplitz</code>	тёплещева (см. YALMIP)
<code>hankel</code>	ганкелева (см. YALMIP)
<code>upper_hankel</code>	ганкелева и верхняя треугольная
<code>lower_hessenberg</code>	верхняя Хессенберга
<code>upper_hessenberg</code>	нижняя Хессенберга
<code>banded(lb,ub)</code>	ленточная (<code>lb</code> и <code>ub</code> — ширины нижней и верхней лент)

Например: `banded(1,1)` — трехдиагональная

Пояснение

[Верхняя матрица Хессенберга](#) — все элементы ниже первой поддиагонали равны нулю.

[Нижняя матрица Хессенберга](#) — все элементы выше первой поддиагонали равны нулю.

Библиотека функций `cvx`

При описании задачи `cvx` можно использовать функции из библиотеки `cvx`. Они либо перегружают стандартные функции MATLAB, либо добавлены дополнительно.

[Чтобы оставаться в рамках SDP, в ограничениях не используйте функции, которые не допускает SDP !](#)

Примеры допустимых в SDP функций:

`sum`, `trace`, `diag`

Библиотека функций `cvx` (дополнительные функции)

Некоторые функции отсутствуют в MATLAB, но определены в `cvx`. Например:

`lambda_max` — наибольшее собственное значение симметричной или эрмитовой матрицы (выпуклая),

`lambda_min` — наименьшее собственное значение симметричной или эрмитовой матрицы (вогнутая),

`det_rootn` — $(\det X)^{1/n}$, где X — неотрицательно определенная матрица (вещественная симметричная или комплексная эрмитова). Вогнутая.

Задание целевой функции

Можно использовать операторы `minimize` или `maximize`.
(Эквивалентно: `minimise` или `maximise`).

Если целевая функция не задана, задача рассматривается как задача разрешимости (feasibility problem).

Даже в режиме SDP целевая функция не обязана быть линейной, а может быть выпуклой или вогнутой, достаточно общего вида. Линейную функцию можно как минимизировать, так и максимизировать. Выпуклую функцию можно только минимизировать, вогнутую — только максимизировать.

Задание ограничений

При задании ограничений допускаются отношения:

$=$ \leq \geq

Все неравенства считаются нестрогими! Строгие неравенства ($<$, $>$) формально допускаются, но понимаются как нестрогие.

Для наглядности перед перечнем ограничений можно поставить строку `subject to` — она не обязательна.

Ни в коем случае не путайте операции (=) и (==) !

Задание ограничений в режиме sdp

- Равенства понимаются как поэлементные.
 - Неравенства для скаляров и векторов понимаются как поэлементные.
 - Неравенства для неквадратных матриц запрещены. Если нужно поэлементное неравенство, матрицы следует векторизовать: вместо $X \leq Y$ пишем $X(:) \leq Y(:)$.
 - Неравенства для квадратных несимметричных матриц также запрещены.
 - Неравенства для квадратных вещественных (комплексных) матриц понимаются в смысле определенности квадратичных (эрмитовых) форм. При этом матрицы должны быть одного порядка.
- Скалярный ноль распространяется до матрицы, но другие скаляры — нет.

Задание ограничений в режиме sdp (продолжение)

Пример.

```
X >= 1 or 1 >= Y % неверно
X >= ones(n,n) or ones(n,n) >= Y % верно
X >= 0 or 0 >= Y % верно
```

Множества

CVX поддерживает описания стандартного набора выпуклых множеств (всего 14 типов). Укажем три типа множеств, которые могут быть полезны для SDP:

- Множество n -векторов с неотрицательными элементами `nonnegative(n)`:

$$\{x \in \mathbb{R}^n : x_i \geq 0, i = 1, \dots, n\}$$

- Множество неотрицательно определенных симметричных $n \times n$ -матриц `semidefinite(n)`:

$$\mathbb{S}_+^n = \{X \in \mathbb{R}^{n \times n} : X = X^T, \quad X \geq 0\}$$

- Множество неотрицательно определенных эрмитовых $n \times n$ -матриц `hermitian_semidefinite(n)`:

$$\{Z \in \mathbb{C}^{n \times n} : Z = Z^*, \quad Z \geq 0\}$$

Множества: описание неравенств

Множества дают альтернативный способ описания неравенств. Рассмотрим множество:

$$\mathbb{S}_+^n = \{X \in \mathbb{R}^{n \times n} : X = X^T, \quad X \geq 0\}$$

Тогда для матрицы $H = H^T$ эквивалентны утверждения:

$$H \geq 0 \iff H \in \mathbb{S}_+^n$$

Операцию принадлежности \in в `cvx` записываем как `==`

Условие $H \in \mathbb{S}_+^n$ можно записать как

```
H == semidefinite(n)      % в любом режиме
H >= 0                   % только в режиме sdp
```

Возможны более общие конструкции:

```
P == B'* semidefinite(n)*B
```

означает

$$\exists X \in \mathbb{S}_+^n : H = B^T X B$$

Выбор солвера

Для задач SDP `cvx` поддерживает бесплатные солверы SeDuMi и SDPT3. SeDuMi установлен по умолчанию.

Для явного выбора солвера используем команды

```
cvx_solver sdpt3
cvx_solver sedumi
```

Команда

```
cvx_solver
```

показывает какой солвер сейчас выбран.

Если команда смены солвера вызвана вне окружения `cvx`, она распространяется на все последующие окружения.

Если команда смены солвера вызвана внутри окружения `cvx`, она распространяется только на данное окружение.

Настройка точности решения задачи

Имеется три уровня относительной погрешности работы солвера:

$$\varepsilon_{\text{solver}} \leq \varepsilon_{\text{standard}} \leq \varepsilon_{\text{reduced}}$$

Здесь

- $\varepsilon_{\text{solver}}$ — точность солвера,
- $\varepsilon_{\text{standard}}$ — стандартная точность, при которой задача считается решенной,
- $\varepsilon_{\text{reduced}}$ — задача считается приближенно решенной (Inaccurate). Если она не достигается, задача считается нерешенной (Failed).

Стандартные значения $[\varepsilon^{1/2}, \varepsilon^{1/2}, \varepsilon^{1/4}]$, где $\varepsilon = 2.22 \times 10^{-16}$ — машинная точность

Настройка точности решения задачи (продолжение)

Точность регулируется командами:

<code>cvx_precision low;</code>	$[\varepsilon^{3/8}, \varepsilon^{1/4}, \varepsilon^{1/4}]$
<code>cvx_precision medium;</code>	$[\varepsilon^{1/2}, \varepsilon^{3/8}, \varepsilon^{1/4}]$
<code>cvx_precision default;</code>	$[\varepsilon^{1/2}, \varepsilon^{1/2}, \varepsilon^{1/4}]$
<code>cvx_precision high;</code>	$[\varepsilon^{3/4}, \varepsilon^{3/4}, \varepsilon^{3/8}]$
<code>cvx_precision best;</code>	$[0, \varepsilon^{1/2}, \varepsilon^{1/4}]$

Команда применяется глобально вне окружения и локально — внутри окружения.

Решение задачи cvx

Что делает `cvx` после `cvx_end`:

- Преобразует задачу к стандартной форме.
- Вызывает солвер.
- Переменные `cvx` преобразуются в обычные переменные MATLAB с числовыми значениями.
- Оптимальное значение (если оно есть) присваивается переменной `cvx_optval`.
- Статус завершения присваивается переменной `cvx_status`.

Проверка результатов вычисления

Статус завершения содержится в строковой переменной `cvx_status`. Возможные значения:

- Solved — решение найдено, оптимальное значение целевой функции записано в `cvx_optval` (0 для разрешимости).
- Unbounded — целевая функция неограничена, `cvx_optval` равно `-Inf` (задача на минимум) или `Inf` (задача на максимум или разрешимость).
- Infeasible — система ограничений несовместна, `cvx_optval` равно `Nan`.

Проверка результатов вычисления (продолжение)

- Ситуация неопределенности.
Inaccurate/Solved — система по-видимому имеет решение,
Inaccurate/Unbounded — целевая функция по-видимому неограничена,
Inaccurate/Infeasible — система ограничений по-видимому несовместна.
- Failed — решить задачу не удалось.
- Overdetermined — переопределенная система (число ограничений-равенств превышает число переменных).
Этот случай не обязательно влечет несовместность, но `cvx` выдает предупреждение:

```
Warning: Overdetermined equality constraints;
problem is likely infeasible.
```

Замечание

Если в процессе выполнения программы возникла ошибка, то прежде, чем запустить программу снова, рекомендуется выполнить команду

```
cvx_clear
```

чтобы удалить из памяти все активные данные.

Служебные команды

- `cvx_problem` — выдает подробную информацию о `cvx`-задаче,
- `cvx_clear` — удаляет из памяти все описания `cvx`-задач,
- `cvx_quiet(true)` — подавляет вывод на экран служебной информации солвера,
- `cvx_pause(true)` — приостанавливает работу программы до нажатия любой клавиши,
- `cvx_where` — выводит строку с маршрутом, по которому установлен пакет `cvx`,
- `cvx_version` — выводит данные о версии `cvx`.

Лекция 10а. Пакет cvx; примеры

Пример1. Неравенство Ляпунова

Требуется найти симметричную $n \times n$ -матрицу H :

$$HA + A^T H < 0, \quad H > 0.$$

Как и YALMIP, cvx не используются строгие неравенства. Поэтому приходится рассматривать задачу

$$HA + A^T H \leq -\varepsilon I_n, \quad H \geq \varepsilon I_n,$$

где $\varepsilon > 0$ — константа.

Неравенство Ляпунова: функция

```
function [H, status] = lyap(A)
    n = size(A,1);
    eps = 1E-5;
    cvx_begin sdp
        variable H(n,n) symmetric
        subject to
            H*A + A'*H <= -eps*eye(n)
            H >= eps*eye(n)
    cvx_end
    status = cvx_status;
```

Неравенство Ляпунова: вызов функции

```
function main_lyap
    clc
    A = [-5 3; 1 -6];
    cvx_solver sedumi
    [H,status] = lyap(A);
    if ~strcmp(status,'Solved')
        disp('Problem is infeasible');
        status
    else
        H
    end
```

Матричной переменной H отвечает три скалярных переменных. Но SeDuMi использует шесть переменных: три для прямой и три для двойственной задачи (вектора x и y).

Неравенство Ляпунова: результат работы

```
Calling SeDuMi 1.34: 6 variables, 3 equality constraints
-----
SeDuMi 1.3.4 by AdvOL, 2005-2008 and Jos F. Sturm, 1998-2003.
Alg = 2: xxz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
eqn m = 3, order n = 5, dim = 7, blocks = 3
nnz(A) = 14 + 0, nnz(ADA) = 9, nnz(L) = 6
it :      b*y      gap      delta      rate      t/tP*      t/tD*      feas      cg      cg      prec
 0 :          2.44E+02 0.000
 1 :   8.86E-06 2.43E+01 0.000 0.0994 0.9900 1.00 1 1 3.4E+00
```

```

2 : 3.49E-07 1.16E+00 0.000 0.0477 0.9900 0.9900 1.00 1 1 1.6E-01
3 : 4.00E-11 3.04E-04 0.000 0.0003 0.9999 0.9999 1.00 1 1 4.2E-05
4 : 4.01E-18 3.05E-11 0.000 0.0000 1.0000 1.0000 1.00 1 1 4.2E-12

iter seconds digits      c*x          b*y
4       0.2   Inf 0.0000000000e+00 4.0102899664e-18
|Ax-b| = 2.6e-12, |Ay-c|_+ = 3.0E-14, |x|= 2.6e+00, |y|= 2.8e-14

Detailed timing (sec)
Pre      IPM      Post
6.000E-02    1.200E-01    2.001E-02
Max-norms: ||b||=1.10000e-04, ||c|| = 0,
Cholesky |add|=0, |skip| = 0, ||L,L|| = 1.
-----
Status: Solved
Optimal value (cvx_optval): +0

H =
0.2803  0.0893
0.0893  0.2530

```

Неравенство Ляпунова: негурвицева матрица

Рассмотрим теперь случай негурвицевой матрицы A :

```

function main_lyap
    clc
    A = [-5 3; 1 6];
    cvx_solver sedumi
    cvx_quiet(true)      % без информации солвера
    [H,status] = lyap(A);
    if ~strcmp(status,'Solved')
        disp('Problem is infeasible');
        status
    else
        H
    end

```

Неравенство Ляпунова: неудачный вызов

Получаем (для SeDuMi):

```

Problem is infeasible
status =
'Inaccurate/Infeasible'

```

Если задать

```
cvx_solver sdpt3
```

получаем (для SDPT3):

```

status =
'Infeasible'

```

Неравенство Ляпунова: негурвицева матрица

Если для негурвицевой матрицы

```
A = [-5 3; 1 6];
```

взять

```
eps = 0;
```

то система разрешима с

```
H =
```

```
All zero sparse: 2x2
```

Пример 2. Круговой критерий

Задача сводится к проверке разрешимости системы LMIs относительно неизвестной $H \in \mathbb{S}^m$:

$$\begin{bmatrix} HA + A^T H & Hb \\ b^T H & 0 \end{bmatrix} < \begin{bmatrix} \mu_1 \mu_2 c c^T & -\frac{1}{2}(\mu_1 + \mu_2)c \\ -\frac{1}{2}(\mu_1 + \mu_2)c^T & 1 \end{bmatrix}, \quad H > 0.$$

Здесь A, b, c — матричные коэффициенты размеров $m \times m, m \times 1, m \times 1$, μ_1, μ_2 — скаляры.

Коэффициенты A, b, c вычисляем по передаточной функции $W(p) = -c^T(pI_m - A)^{-1}b$, I_m — единичная матрица.

Круговой критерий: функция

```
function [H,status] = circle(A,b,c,mu1,mu2)
n = size(A,1);
R=[mu1*mu2*c*c' -0.5*(mu1+mu2)*c; -0.5*(mu1+mu2)*c' 1];
cvx_begin sdp
    variable H(n,n) symmetric
    eps = 1E-6;
    subject to
    H >= eps*eye(n)
    [H*A+A'*H H*b; b'*H 0] <= R - eps*eye(n+1)
cvx_end
status = cvx_status;
```

Круговой критерий: вызов функции

```
function main_circle
clc
mu1 = -0.1;
mu2 = 0.1;
w = tf(1,[1 2 0.5]);
[A,b,c] = ssdata(w);
c = -c';
cvx_solver sedumi
cvx_quiet(true)
[H,status] = circle(A,b,c,mu1,mu2);
if ~strcmp(status,'Solved')
    disp('Problem is infeasible');
    status
else
    H
end
```

Круговой критерий: результат

```
H =
0.5348 0.3001
0.3001 0.7716
```

Круговой критерий: варианты кода

```
cvx_begin sdp
    variable H(n,n) symmetric
    eps = 1E-6;
    subject to
        H >= eps*eye(n);
        L11 = H*A+A'*H;
        L12 = H*b;
        [L11 L12; L12' 0] <= R - eps*eye(n+1)
cvx_end
```

Здесь L_{11} , L_{12} — аффинные выражения cvx.

Другой вариант кода:

```
R = R - eps*eye(n+1)
H -eps*eye(n) == semidefinite(n)
R - [L11 L12; L12' 0] == semidefinite(n+1)
```

Пример 3.

Пусть A, B, C — коэффициенты размеров $n \times n$, $n \times m$, $n \times l$. Переменные: $H \in \mathbb{S}^n$, $t \in \mathbb{R}$. Решить задачу:

$$t \rightarrow \min, \quad \begin{bmatrix} HA + A^T H + C^T C & HB \\ B^T H & -tI_m \end{bmatrix} < 0, \quad H > 0.$$

Пример 3. Функция

```
function [H,t,status] = gain_1(A,B,C)
n = size(B,1);
m = size(B,2);
cvx_begin sdp
    variable t
    variable H(n,n) symmetric
    minimize t
    subject to
        H >= 0
        [ A'*H + H*A + C'*C H*B; ...
        B'*H -t*eye(m)] <= 0
cvx_end
status = cvx_status;
```

Тестовая система

Для генерации тестовой системы удобно использовать функцию `rss`:

```
w = rss(n,l,m);
```

Случайным образом создает [устойчивую](#) минимальную (управляемую и наблюдаемую) систему порядка n с l выходами и m входами (random state space realization).

Тестовая система (продолжение)

Пример.

```
w = rss(3,4,2); % 4 выхода, 2 входа
[A,B,C] = ssdata(w)
```

Получаем:

```
A =
-2.2699    0.9215   -0.0422
 0.9215   -1.0180    0.2339
 -0.0422    0.2339   -0.8977

B =
 1.6555   -0.8655
 0.3075   -0.1765
 -1.2571        0

C =
      0    0.3914   -0.4762
 -2.3299    0.4517    0.8620
 -1.4491        0   -1.3617
      0        0    0.4550
```

Пример 3. Вызов функции

```
function main_gain
clc
n = 3; m = 2; l = 4;
w = rss(n,l,m);
[A,B,C] = ssdata(w)
cvx_solver sedumi
cvx_quiet(true)
[H,t,status] = gain_1(A,B,C);
if ~strcmp(status,'Solved')
    disp('Problem is infeasible');
    status
else
    H
    t
end
```

Пример 3. Возможный вариант вывода

```
H =
 4.2546   -0.0428    0.5930
 -0.0428    3.9141   2.2857
 0.5930    2.2857   6.5230

t =
 13.9268
```

Так как генерация матриц случайная, при некоторых запусках задача будет иметь решение, а при других — нет. Для отчета по заданию зафиксируйте значения коэффициентов, при которых решение получено!

Пример 4

Пусть A, C — коэффициенты размеров $n \times n, 1 \times n, H = H^T$ — переменная $n \times n$. Рассмотрим задачу;

$$\begin{aligned} \text{tr } H &\rightarrow \min, \\ HA + A^T H &< 0, \\ H &> C^T C. \end{aligned}$$

Очевидно система ограничений разрешима только когда A гурвицева.

Пример 4: функция

```
function [H,opt,status] = tr(A,C)
n = size(A,1);
cvx_begin sdp
variable H(n,n) symmetric
minimize trace(H)
subject to
H*A + A'*H <= 0
H >= C'*C
cvx_end
status = cvx_status;
opt = cvx_optval;
```

Пример 4: вызов функции

```
function main_tr
clc
cvx_solver sedumi
cvx_quiet(true)
n = 3;
w = rss(n,1,1);
[A,B,C] = ssdata(w)
[H,opt,status] = tr(A,C);
if ~strcmp(status,'Solved')
    disp('Problem is infeasible');
    status
else
    H
    opt
end
```

Пример 4. Возможный вариант вывода

```
A =
-0.9962    0.0503   -0.0138
 0.0503   -0.6336    0.1434
 -0.0138    0.1434   -0.4833
```

```

C =
0      0.2623   -0.8980

H =
0.0002   -0.0016   -0.0005
-0.0016    0.0863   -0.2304
-0.0005   -0.2304    0.8078

opt =
0.8943

```

Пример 5. Решение задачи MAXDET

В этом случае рассматривается функция: `det_rootn(X)` — вычисляет $(\det X)^{1/n}$, где X — неотрицательно определенная матрица (вещественная симметричная или комплексная эрмитова). Эта функция вогнутая.

Пример

Найти эллипсоид вида

$$\Omega = \{x : x^T H x \leq 1\}, \quad H > 0,$$

наименьшего объема, содержащий заданный набор векторов $s_i \in \mathbb{R}^n$, $i = 1, \dots, m$.

Получаем задачу MAXDET:

$$\det H \rightarrow \max, \quad s_i^T H s_i \leq 1, \quad i = 1, \dots, m, \quad H > 0$$

Решим задачу, максимизируя функцию `det_rootn(H)`

Решение задачи MAXDET: программа

```

clc; cvx_clear
cvx_solver sedumi
cvx_quiet(true)
s = [1 1; -0.5 1.5; 2 -1.5; -1 -1]';
n = 2;
cvx_begin sdp
variable H(n,n) symmetric
maximize det_rootn(H)
subject to
H >= 0
for i=1:4
    si = s(:,i);
    si'*H*si <= 1
end;
cvx_end
H
cvx_status
cvx_optval

```

Решение задачи MAXDET: результат

```
H =  
0.2653    0.1633  
0.1633    0.4082
```

```
cvx_status =  
'Solved'
```

```
cvx_optval =  
0.2857
```

Лекция 11. Пакет YALMIP: простейшая задача BMI

BMIs — билинейные матричные неравенства

Bilinear Matrix Inequalities: $F(x, y) > 0$, где

$$F(x, y) = F_0 + \sum_{i=1}^m x_i F_i + \sum_{j=1}^n y_j G_j + \sum_{i=1}^m \sum_{j=1}^n x_i y_j R_{ij}$$

Здесь $F(x, \cdot)$ линейна (аффинна) по x , $F(\cdot, y)$ линейна по y , но $F(x, y)$ нелинейна (квадратична) по совокупности переменных x, y .

Для общих задач с BMIs не существует эффективных методов решения (это задачи класса NP-hard). Мы рассмотрим частный вид задач BMIs, для которых решение может быть найдено относительно просто: путем решения последовательности задач LMIs.

Вычисление максимальной скорости затухания

Пусть $A — n \times n$ -матрица, t — скалярная переменная. Задача:

$$t \rightarrow \max, \quad \operatorname{Re} \lambda_j(A) \leq -t < 0, \forall t$$

Эквивалентно:

$$t \rightarrow \max, \quad A + tI_n — гурвицева.$$

Используем неравенство Ляпунова:

$$t \rightarrow \max, \quad H(A + tI_n) + (A + tI_n)^T H < 0, \quad H > 0.$$

Эквивалентно:

$$t \rightarrow \max, \quad HA + A^T H < -2tH \quad H > 0.$$

Эта система не линейна, но **билинейна, BMI** по совокупности переменных t, H . Поэтому непосредственно применить солвер SDP не получится.

Более общая постановка задачи

Более общая постановка задачи:

$$t \rightarrow \max, \quad F(t, x) \geq 0.$$

Предположения:

- При фиксированном t матрица-функция $F(t, x)$ линейна (аффинна) по x .
- При $t = 0$ неравенство $F(0, x)$ разрешимо относительно x .
- При фиксированном x функция $F(t, x)$ убывает по t : $F(\tau_1, x) \geq F(\tau_2, x)$ при $0 \leq \tau_1 \leq \tau_2$.

В этом случае задачу можно решить как последовательность задач SDP.

Замечание. Здесь для нас даже не существенно, что $F(x, t)$ линейна по t .

Метод деления пополам (бисекции)

Построим последовательность сужающихся отрезков $[t_k^-, t_k^+]$, $k = 0, 1, 2, \dots$, содержащих оптимальное t_{opt} .

1. Возьмем $t_0^- = 0$, t_0^+ — достаточно большое t , так что $F(t, x) \geq 0$ не разрешимо по x .
2. Вычислим $t_1 = \frac{1}{2}(t_0^- + t_0^+)$.
3. Если $F(t_1, x) \geq 0$ не разрешимо по x , возьмем $[t_1^-, t_1^+] = [t_0^-, t_1]$.
4. Если $F(t_1, x) \geq 0$ разрешимо по x , возьмем $[t_1^-, t_1^+] = [t_1, t_0^+]$.

И т. д. Процесс заканчивается, когда $t_k^+ - t_k^- < \delta$, δ — малое число.

Метод деления пополам: решение в YALMIP

Нам понадобятся два солвера:

- `bisection` — внутренний солвер YALMIP, реализующий деление пополам.
- Какой либо внешний SDP солвер (SeDuMi, SDPT3 и пр.) для решения задач разрешимости.

Указываем их в настройках:

```
options = sdpsettings('solver','bisection',...
    'bisection.solver','sedumi');
```

Способы вызова солверов (для максимизации t):

- `diagn = optimize(F, -t, options);`
- `diagn = bisection(F, -t, options);`

Метод деления пополам: функция

```
function [H0,t0] = maxdecay(A)
n = size(A,1);
H = sdpvar(n);
t = sdpvar(1);
H0 = [];
t0 = [];
eps0 = 1E-4;
F = [H>=eps0*eye(n), A'*H+H*A <= -2*t*H];
ops = sdpsettings('solver','bisection',...
    'bisection.solver','sedumi');
diagn = optimize(F, -t,ops);
if diagn.problem == 0
    H0 = value(H);
    t0 = value(t);
end
```

Метод деления пополам: вызов функции

```
function main_decay
clc
A = [-1 2;-3 -4];
[H,t] = maxdecay(A);
if isempty(t)
    disp('problem is not feasible');
else
    H
    t
end
```

Метод деления пополам: результаты счета

```
Selected solver: sedumi
Generating initial bound: 0 (ok), 1 (ok), 3 (fail).
Iteration  Lower bound   Test   Upper bound   Gap      Solver status at test
  1 :  1.00000E+00  2.00000E+00  3.00000E+00  2.00000E+00 Successfully solved
  2 :  2.00000E+00  2.50000E+00  3.00000E+00  1.00000E+00 Successfully solved
  3 :  2.50000E+00  2.75000E+00  3.00000E+00  5.00000E-01 Infeasible problem
  4 :  2.50000E+00  2.62500E+00  2.75000E+00  2.50000E-01 Infeasible problem
  5 :  2.50000E+00  2.56250E+00  2.62500E+00  1.25000E-01 Infeasible problem
  6 :  2.50000E+00  2.53125E+00  2.56250E+00  6.25000E-02 Infeasible problem
  7 :  2.50000E+00  2.51563E+00  2.53125E+00  3.12500E-02 Infeasible problem
  8 :  2.50000E+00  2.50781E+00  2.51563E+00  1.56250E-02 Infeasible problem
  9 :  2.50000E+00  2.50391E+00  2.50781E+00  7.81250E-03 Infeasible problem
 10 :  2.50000E+00  2.50195E+00  2.50391E+00  3.90625E-03 Infeasible problem
 11 :  2.50000E+00  2.50098E+00  2.50195E+00  1.95313E-03 Infeasible problem
 12 :  2.50000E+00  2.50049E+00  2.50098E+00  9.76563E-04 Infeasible problem
 13 :  2.50000E+00  2.50024E+00  2.50049E+00  4.88281E-04 Infeasible problem
 14 :  2.50000E+00  2.50012E+00  2.50024E+00  2.44141E-04 Infeasible problem
 15 :  2.50000E+00  2.50006E+00  2.50012E+00  1.22070E-04 Infeasible problem
 16 :  2.50000E+00  2.50003E+00  2.50006E+00  6.10352E-05 Infeasible problem
 17 :  2.50000E+00  2.50002E+00  2.50003E+00  3.05176E-05 Infeasible problem
 18 :  2.50000E+00  2.50001E+00  2.50002E+00  1.52588E-05 Infeasible problem
Bisection terminated successfully with objective 2.5
```

H =

1.2005	0.6002
0.6002	0.8003

t =

2.5000

Метод деления пополам: проверка решения Вызовем: checkset(F). Получим:

```
=====
| ID| Constraint| Primal residual| Dual residual|
=====
| #1| Matrix inequality| 0.36759| NaN|
| #2| Matrix inequality (bilinear)| -2.7889e-15| NaN|
=====
| A primal-dual optimal solution would show non-negative residuals. |
| In practice, many solvers converge to slightly infeasible          |
| solutions, which may cause some residuals to be negative.         |
| It is up to the user to judge the importance and impact of        |
| slightly negative residuals (i.e., infeasibilities)               |
| https://yalmip.github.io/command/check/                          |
| https://yalmip.github.io/faq/solutionviolated/                  |
=====
```

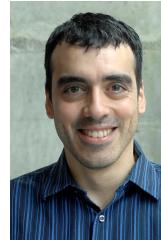
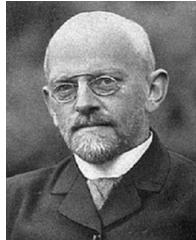
Два неравенства: $H > 0$, $HA + A^T H \leq -2tH$. Выводятся миним. собственные числа для невязок:

$$H, \quad -2tH - HA - A^T H.$$

Во втором случае число практически ноль (порядка 10^{-15}).

Лекция 12. Исследование сумм квадратов — задача SOS

Ключевые фигуры



Давид Гильберт (1862–1943)

Рудольф Калман (1930–2016)

Пабло А. Паррило, Массачусетский технологический институт

Неотрицательные многочлены и SOS-многочлены

Пусть $p(\bar{x})$ — многочлен от n переменных ($\bar{x} \in \mathbb{R}^n$) степени m .

Многочлен $p(\bar{x})$ называется **неотрицательным**, если $p(\bar{x}) \geq 0$ при всех $\bar{x} \in \mathbb{R}^n$.

Пример. Неотрицательный многочлен, $n = m = 2$,

$$p_0(x_1, x_2) = x_1^2 + x_1 x_2 + x_2^2 \geq 0, \quad \forall x_1, \forall x_2.$$

Многочлен $p(\bar{x})$ называется **SOS-многочленом** (Sum Of Squares polynomial), если существуют число k и многочлены $q_1(\bar{x}), \dots, q_k(\bar{x})$ такие, что

$$p(\bar{x}) = \sum_{i=1}^k [q_i(\bar{x})]^2, \quad \forall \bar{x} \in \mathbb{R}^n.$$

Очевидно **любой SOS-многочлен неотрицателен**.

SOS-многочлены: примеры

Часто положительный многочлен можно представить в виде суммы квадратов (обычно SOS-представление неоднозначно).

Пример 1. Многочлен

$$\begin{aligned} p_0(x_1, x_2) &= x_1^2 + x_1 x_2 + x_2^2 = \left(x_1 + \frac{1}{2}x_2\right)^2 + \left(\frac{\sqrt{3}}{2}x_2\right)^2 \\ p_0(x_1, x_2) &= \left(\frac{\sqrt{3}}{2}x_1\right)^2 + \left(\frac{1}{2}x_1 + x_2\right)^2 \end{aligned}$$

Пример 2. Многочлен 4-го порядка

$$p_1(x_1, x_2) = x_1^4 - x_1^2 x_2^2 + x_2^4 = \left(x_1^2 - \frac{1}{2}x_2^2\right)^2 + \left(\frac{\sqrt{3}}{2}x_2\right)^2.$$

По-другому:

$$p_1(x_1, x_2) = (x_1^2 - x_2^2)^2 + (x_1 x_2)^2.$$

Отсутствие необходимости SOS-представления

Является ли любой неотрицательный многочлен SOS-многочленом? В общем случае это не так.

Давид Гильберт доказал, что это выполнено в случаях (n — число переменных, m — степень):

- $n = 1, m$ — любое четное;
- n — любое, $m = 2$;
- $n = 2, m = 4$.

Примеры. Многочлены

$$p_2(x_1, x_2) = x_1^2 x_2^2 (x_1^2 + x_2^2 - 3) + 1,$$

$$p_3(x_1, x_2, x_3) = x_3^6 + x_1^4 x_2^2 + x_1^2 x_2^4 - 3x_1^2 x_2^2 x_3^2$$

неотрицательны, но не SOS.

Задачи SOS

Пусть $P(\bar{x})$ — полином (скалярный или с матричными симметричными коэффициентами).

- Задача разрешимости. Определить, выполнено ли

$$P(\bar{x}) \geq 0, \quad \forall \bar{x}.$$

Заменяется на задачу: имеет ли заданный $P(\bar{x})$ SOS-представление?

Наличие SOS-представления — достаточное (но не необходимое) условие положительности $p(\bar{x})$.

Тем не менее, для проверки положительности $p(\bar{x})$ часто используют задачу SOS.

- Задача оптимизации. Линейная целевая функция

$$c^T \bar{x} \rightarrow \min (\max)$$

при условии, что $p(\bar{x})$ имеет SOS-представление.

17-я проблема Гильберта

Формулировка проблемы

Пусть $p(\bar{x})$ — рациональная функция от n переменных с вещественными коэффициентами такая, что $p(\bar{x}) \geq 0$ при всех $\bar{x} \in \mathbb{R}^n$. Можно ли ее представить в виде суммы квадратов рациональных функций с вещественными коэффициентами?

Ответ на этот вопрос положительный (Эмиль Артин, 1927).

К сожалению, не существует эффективных алгоритмов для SOS-представления рациональных функций. Для полиномов такие алгоритмы есть и они сводятся к задачам SDP.

Не следует забывать, что существование SOS-представления — [достаточное](#) условие неотрицательности полинома.

SOS-многочлены и мономы (на примере)

Рассмотрим однородный многочлен (форму) степени $2m = 4$:

$$p_1(x_1, x_2) = x_1^4 - x_1^2 x_2^2 + x_2^4$$

Составим вектор Z из мономов (одночленов) степени $m = 2$:

$$Z = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}, \quad M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad L = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 2 & 0 \\ -1 & 0 & 0 \end{bmatrix}.$$

Так как $p_1(x_1, x_2) = Z^T M Z$ и $Z^T L Z = 0$, то

$$p_1(x_1, x_2) = Z^T (M + \alpha L) Z, \quad \forall \alpha \in \mathbb{R}.$$

Условие SOS-приводимости: $M + \alpha L \geq 0$ (LMI относительно α).

Неравенство разрешимо (по критерию Сильвестра) при $\alpha \geq 1$.

SOS-многочлены и мономы (продолжение)

Мы получили:

$$p_1(x_1, x_2) = Z^T (M + \alpha L) Z,$$

где $M + \alpha L \geq 0$ при $\alpha \geq 1$. Тогда существует $k \times 3$ -матрица C ($k \leq 3$) такая, что

$$M + \alpha L = C^T C \geq 0.$$

Пусть

$$C = \begin{bmatrix} C_1 \\ \vdots \\ C_k \end{bmatrix},$$

где C_j — матрица-строка. Тогда

$$p_1(x_1, x_2) = Z^T C^T C Z = \sum_{j=1}^k (C_j Z)^2,$$

т. е. $p_1(x_1, x_2)$ допускает SOS-представление.

Примеры содержательных задач SOS

Приведем несколько примеров, которые приводят к задачам SOS.

Пример. Круговой критерий

Пусть $W(s)$ — передаточная функция, μ_1, μ_2 — скаляры, $\mu_1 < \mu_2$. Неравенство кругового критерия:

$$1 + (\mu_1 + \mu_2) \operatorname{Re} W(i\omega) + \mu_1 \mu_2 |W(i\omega)|^2 \geq 0, \quad \forall \omega \in \mathbb{R}.$$

Так как комплексное сопряжение

$$\overline{W(i\omega)} = W(-i\omega),$$

то

$$\operatorname{Re} W(i\omega) = \frac{1}{2} (W(i\omega) + W(-i\omega)), \quad |W(i\omega)|^2 = W(-i\omega) W(i\omega).$$

Пример. Круговой критерий (продолжение)

Пусть $W(s) = M(s)/N(s)$, где $M(s), N(s)$ — многочлены. Тогда

$$\begin{aligned}\operatorname{Re} W(i\omega) &= \frac{1}{2} \left(\frac{M(i\omega)}{N(i\omega)} + \frac{M(-i\omega)}{N(-i\omega)} \right) = \\ &= \frac{1}{2} \frac{M(i\omega)N(-i\omega) + M(-i\omega)N(i\omega)}{N(i\omega)N(-i\omega)}, \\ |W(i\omega)|^2 &= \frac{M(i\omega)M(-i\omega)}{N(-i\omega)N(i\omega)}.\end{aligned}$$

Так как $N(-i\omega)N(i\omega) = |N(i\omega)|^2 \geq 0$, частотное неравенство принимает вид:

$$\begin{aligned}N(i\omega)N(-i\omega) + \frac{1}{2}(\mu_1 + \mu_2)(M(i\omega)N(-i\omega) + M(-i\omega)N(i\omega)) + \\ + \mu_1\mu_2 M(i\omega)M(-i\omega) \geq 0, \quad \forall \omega \in \mathbb{R}.\end{aligned}$$

Пример. Круговой критерий (продолжение)

Определим многочлены

$$\begin{aligned}P(s) &= N(s)N(-s) + \frac{1}{2}(\mu_1 + \mu_2)(M(s)N(-s) + M(-s)N(s)) + \\ &\quad + \mu_1\mu_2 M(s)M(-s), \\ Q(\omega) &= P(i\omega).\end{aligned}$$

Можно показать, что при вещественных коэффициентах $W(s)$ многочлен $Q(\omega)$ также имеет вещественные коэффициенты. Частотное неравенство принимает вид:

$$Q(\omega) \geq 0, \quad \forall \omega \in \mathbb{R},$$

т. е. нужно проверить, что многочлен $Q(\omega)$ положительный. Для этого достаточно проверить, что $Q(\omega)$ допускает SOS-представление.

Пример. Функция Ляпунова для системы с полиномиальной правой частью

Рассмотрим систему дифф. уравнений n -го порядка:

$$\frac{d\bar{x}}{dt} = Q(\bar{x}),$$

где \bar{x} — n -вектор, элементы вектор-функции в правой части $Q_k(\bar{x})$, $k = 1, \dots, n$, — полиномы. Обозначим квадрат нормы $N(\bar{x}) = x_1^2 + \dots + x_n^2$. Будем искать полиномиальную функцию Ляпунова $V = V(\bar{x})$. Тогда ее частные производные

$$\frac{\partial V}{\partial x_k} = \frac{\partial V}{\partial x_k}(\bar{x}), \quad k = 1, 2, 3,$$

тоже полиномы. На решениях системы ее производная

$$\frac{dV}{dt} = \sum_{k=1}^n \frac{\partial V}{\partial x_k} \frac{dx_k}{dt} = \sum_{k=1}^n \frac{\partial V}{\partial x_k}(\bar{x}) Q_k(\bar{x}).$$

Функция Ляпунова для системы с полиномиальной правой частью (продолжение)

Рассмотрим полином:

$$DV(\bar{x}) = \sum_{k=1}^n \frac{\partial V}{\partial x_k}(\bar{x}) Q_k(\bar{x}).$$

Пусть ε — некоторое малое число. Для функции Ляпунова получаем неравенства:

$$V(\bar{x}) \geq \varepsilon N(\bar{x}), \quad DV(\bar{x}) \leq -\varepsilon N(\bar{x}), \quad \forall \bar{x} \in \mathbb{R}^n$$

(Так как система однородна относительно V , то можно взять новую функцию $W(\bar{x}) = \frac{1}{\varepsilon} V(\bar{x})$, т. е. положить $\varepsilon = 1$.)

Получаем два полиномиальных условия: $V(\bar{x}) - \varepsilon N(\bar{x})$ является SOS, $-DV(\bar{x}) - \varepsilon N(\bar{x})$ является SOS.

Пример 3. Глобальный минимум полиномиальной функции

Пусть $f(\bar{x})$ — полином. Найдем глобальный минимум $f(\bar{x})$:

$$\gamma_0 = \inf_{\bar{x}} f(\bar{x})$$

Для этого решим оптимизационную задачу:

$$\gamma \rightarrow \max, \quad f(\bar{x}) \geq \gamma, \quad \forall \bar{x}.$$

Пусть \bar{x}, γ — переменные, $p(\bar{x}, \gamma) = f(\bar{x}) - \gamma$. Решим задачу SOS:

$$\gamma \rightarrow \max, \quad p(\bar{x}, \gamma) — SOS.$$

Тогда оптимальное γ даст γ_0 .

Пример 4. Минимум дробно-рациональной функции

Пусть $Q(x)$ — дробно-рациональная функция,

$$Q(x) = \frac{p(x)}{q(x)}, \quad p(x), q(x) \text{ полиномы,}$$

$q(x) > 0, \forall x$. Рассмотрим задачу:

$$\gamma \rightarrow \max, \quad p(x) \geq \gamma q(x), \quad \forall x$$

Тогда $Q_{\min} = \gamma_{\max}$.

Программное обеспечение

Существуют два различных пакета, оба для работы в системе MATLAB.

- Пакет **SOSTOOLS**. Отдельный пакет со своим интерфейсом. Работает совместно с MATLAB Symbolic Math Toolbox.
- Модуль SOS в составе пакета **YALMIP**.

В том и другом варианте дополнительно потребуется один из SDP солверов (например, SeDuMi или SDPT3).

Лекция 12а. Пакет SOSTOOLS. Работа с пакетом

Общая характеристика пакета SOSTOOLS

Пакет использует свой собственный интерфейс. (Позднее рассмотрим модуль SOS в составе YALMIP — это другой пакет.)

Первая версия пакета SOSTOOLS разработана в 2002, последняя — в сентябре 2021 (версия 4.0). Разработчики: A. Papachristodoulou, J. Anderson, G. Valmorbida, S. Prajna, P. Seiler, P. A. Parrilo (Oxford Univ., MIT, Univ. Minnesota)

Требования (к версии 2021 года):

- MATLAB R2009a или позднее
- MATLAB Symbolic Math Toolbox
- Какой-либо из SDP солверов: SeDuMi, SDPT3, CSDP, SDPA, SDPNAL, SDPNAL+

Скачиваем дистрибутив SOSTOOLS с сайта Калифорнийского технологического института (или зеркала страницы)

<https://www.cds.caltech.edu/sostools/>

Устанавливаем маршруты к пакету с подпапками (SetPath).

Скалярные многочлены в Math Symbolic Toolbox

```
syms x y; % скалярные символьные переменные
p = 2*x^2 + 3*x*y + 4*y^4; % символьный объект-полином
dpdx = diff(p,x) % частная производная p по x
dpdy = diff(p,y) % частная производная p по y
```

Получаем многочлены:

```
dpdx = 4*x + 3*y
dpdy = 3*x + 16*y
```

Вычисление второй производной от p по x:

```
d2pdx2 = diff(p,x,2) % 1-й способ
d2pdx2 = diff(p,x,x) % 2-й способ
d2pdx2 = diff(diff(p,x),x) % 3-й способ
```

Смешанная производная от p по x и y:

```
d2pdxdy = diff(p,x,y)
```

Матричные многочлены в Math Symbolic Toolbox

Аналогично работаем с матрицами.

Пример:

```

syms x y;
p = 2*x^2 + 3*x*y + 4*y^4;
d2pdx2 = diff(p,x,2);
d2pdx dy = diff(p,x,y);
d2pdy2 = diff(p,y,y);
J = [d2pdx2 d2pdx dy; d2pdx dy d2pdy2]

```

Получаем матрицу вторых производных (матрицу Гессе):

```
[ 4,      3]
[ 3, 48*y^2]
```

Вариант:

```
J1 = jacobian(jacobian(p, [x y]), [x y])
```

Multivariable Polynomial Toolbox

Программисты, не имеющие доступа к Math Symbolic Toolbox, могут альтернативно использовать бесплатный Multivariable Polynomial Toolbox (входит в пакет SOSTOOLS).

В нем вместо `syms` используется команда `pvar`.

Этапы решения задачи

Используется стандартная последовательность действий:

- Создание новой задачи задачи
- Описание переменных
- Описание ограничений
- Описание целевой функции (если нужно)
- Вызов солвера
- Получение решения

Независимые переменные и неизвестные переменные

Следует различать два вида переменных. Пусть ищется многочлен вида:

$$p(x, y) = ax^2 + bxy + cy^2.$$

Здесь x, y — независимые переменные (independent variables), a, b, c — неизвестные переменные (decision variables). И те, и другие в SOSTOOLS создаются на основе символьных переменных MATLAB (symbolic variables).

Таким образом, a, b, c выступают как неизвестные параметры полинома:

$$p(x, y) = p_{a,b,c}(x, y)$$

Создание новой SOS-задачи

Проще всего сначала создать «пустую» задачу, а потом добавить в нее переменные, ограничения и целевую функцию.

Новая задача создается командой **sosprogram**. При этом определяются **независимые** переменные (на основе символьных переменных).

```
syms x y z a b c
prog = sosprogram([x y z]);
```

Создана «пустая» SOS-задача **prog** с независимыми переменными **x**, **y**, **z**.

Далее будем дополнять описание задачи с помощью конструкции:

```
prog = команда(prog, параметры);
```

Описание неизвестных переменных (decision variables)

Пакет использует следующие виды **неизвестных** переменных:

- Скалярные неизвестные переменные (**sosdecvar**)
- Скалярные полиномиальные переменные (**sospolyvar**)
- Скалярные SOS-переменные (**sossosvar**)
- Матричные полиномиальные переменные (**sopolymatrixvar**)
- Матричные SOS-переменные (**sossosmatrixvar**)

Описание скалярных неизвестных переменных

Скалярные неизвестные создаются из символьных переменных с помощью команды **sosdecvar**.

Пример.

```
syms x y z a b
prog = sosprogram([x y z]);
prog = sosdecvar(prog, [a b]);
```

Здесь **x**, **y**, **z** — независимые переменные, **a**, **b** — скалярные неизвестные переменные.

Описание скалярных полиномиальных переменных

Полиномиальные скалярные переменные — полиномы с неизвестными коэффициентами.

1-й способ создания: описать скалярные независимые и неизвестные переменные и с их помощью создать полином.

```
syms x y a b c
prog = sosprogram([x y]);
prog = sosdecvar(prog, [a b c]);
v = a*x^2 + b*x*y + c*y^2;
```

2-й способ создания: создать полином из последовательности мономов с помощью функции [sospolyvar](#).

```
syms x y
prog = sosprogram([x y]);
[prog,v] = sospolyvar(prog,[x^2; x*y; y^2]);
```

Получаем:

```
v = coeff_1*x^2 + coeff_2*x*y + coeff_3*y^2
```

Описание полиномиальных переменных (продолжение)

После создания полиномиальной переменной с помощью [sospolyvar](#) (и далее — [sossosvar](#)) неизвестные коэффициенты полиномов получаю имена `coeff_n`, где n — номер переменной.

Это внутренние имена, которые нельзя использовать в командах MATLAB. Чтобы сделать их доступными в MATLAB, следует добавить еще один параметр типа строка:

```
[prog,v] = sospolyvar(prog,[x^2; x*y; y^2], 'wscoeff');
```

Описание полиномиальных переменных (продолжение)

Продолжим создание полиномиальных переменных:

```
[prog,u] = sospolyvar(prog,[1; x; y; x^2; x*y; y^2]);
```

Получаем:

```
u =
coeff_7*x^2 + coeff_8*x*y + coeff_5*x
+ coeff_9*y^2 + coeff_6*y + coeff_4
```

Описание скалярных SOS-переменных

SOS-переменные — это полиномы, представимые в виде

$$p(\bar{x}) = Z(\bar{x})^T Q Z(\bar{x}), \quad Q^T = Q \geq 0.$$

Здесь $Z(\bar{x})$ — вектор из мономов, Q — неизвестная матрица.

Для описания используется функция [sossosvar](#) аналогичная [sospolyvar](#).

```
syms x y
prog = sosprogram([x; y]);
[prog,u] = sossosvar(prog,[x y]);
```

Получаем:

```
u =
x*(coeff_1*x + coeff_2*y) + y*(coeff_3*x + coeff_4*y)
```

Описание скалярных SOS-переменных (продолжение)

Продолжим создание переменных:

```
[prog,w] = sossosvar(prog,[1; x; x^2]) % степени 0, 1, 2
```

Получим:

```
w =
coeff_5 + coeff_6*x + x*(coeff_10*x^2
+ coeff_9*x + coeff_8) + coeff_7*x^2
+ x^2*(coeff_13*x^2 + coeff_12*x + coeff_11)
```

Описание вектора мономов

Для генерации мономов можно использовать функцию [monomials](#) с двумя параметрами: вектор независимых переменных и степени мономов. Пример:

```
Z = monomials{[x y], [1 2 3]);
```

Здесь x, y — независимые переменные, генерируется вектор-столбец из всех мономов степени 1, 2, 3:

```
x; y; x^2; x*y; y^2; x^3; x^2*y; x*y^2; y^3
```

Вектор Z можно использовать при вызове функций [sospolyvar](#), [sossosvar](#):

```
[prog,u] = sospolyvar(prog,Z);
[prog,v] = sossosvar(prog,Z);
```

Матричные переменные

Для описания матричных переменных используют функции [sosmatrixvar](#), [sossosmatrixvar](#):

```
[prog,P] = sopolymatrixvar(prog,Z,[m n],matrixstr)
[prog,P] = sossosmatrixvar(prog,Z,[m n],matrixstr)
```

Здесь Z — вектор мономов, m, n — размеры матрицы, matrixstr — может принимать значение 'symmetric' (может отсутствовать).

P — матрица из неизвестных скалярных полиномов

Пример: матричные переменные степени 0

```
syms x y
X = [x y];
prog = sosprogram(X);
Z = monomials(X,0); % степень 0
[prog,P] = sopolymatrixvar(prog,Z,[2 2]);
[prog,Q] = sopolymatrixvar(prog,Z,[2 2],'symmetric');
P, Q
```

Получаем:

```

P =
[ coeff_1, coeff_2]
[ coeff_3, coeff_4]

Q =
[ coeff_5, coeff_6]
[ coeff_6, coeff_7]

```

Пример: матричные переменные степени 1

```

syms x y
X = [x y];
prog = sosprogram(X);
Z = monomials(X,1); % степень 1
[prog,P] = sopolymatrixvar(prog,Z,[2 2]);
[prog,Q] = sopolymatrixvar(prog,Z,[2 2], 'symmetric');
P, Q

```

Получаем:

```

P =
[ coeff_1*x + coeff_2*y, coeff_3*x + coeff_4*y]
[ coeff_5*x + coeff_6*y, coeff_7*x + coeff_8*y]

Q =
[ coeff_9*x + coeff_10*y, coeff_11*x + coeff_12*y]
[ coeff_11*x + coeff_12*y, coeff_13*x + coeff_14*y]

```

Пример: матричные переменные степени 2

```

syms x y
prog = sosprogram([x y]);
Z = monomials([x y],2);
[prog,Q] = sopolymatrixvar(prog,Z,[2 2], 'symmetric');
Q(1,1), Q(1,2), Q(2,1), Q(2,2)

```

Получаем:

```

ans =
coeff_1*x^2 + coeff_2*x*y + coeff_3*y^2
ans =
coeff_4*x^2 + coeff_5*x*y + coeff_6*y^2
ans =
coeff_4*x^2 + coeff_5*x*y + coeff_6*y^2
ans =
coeff_7*x^2 + coeff_8*x*y + coeff_9*y^2

```

Пример: матричные переменные, включающие мономы степеней 0,1,2

```

Z = monomials([x y], [0,1,2]);
[prog,Q] = sopolymatrixvar(prog,Z,[2 2], 'symmetric');

```

Получается 42 неизвестных коэффициента.

Виды ограничений

- Ограничения типа равенств (soseq). Могут быть для скалярных и матричных полиномов.
- Ограничения типа неравенств для скалярных полиномов (sosineq).
- Ограничения типа неравенств для матричных полиномов (sosmatrixineq). Только для квадратных симметричных матриц.

Ограничения типа равенств

Используется функция **soseq**.

Пример. Пусть $p(x, y)$, $q(x, y)$ — полиномы. Равенство

$$p(x, y) = q(x, y) \quad \forall x, \forall y$$

записывается как

```
prog = soseq(prog, p - q)
```

Система равенств

$$p(x, y) = x^2, \quad p(x, y)q(x, y) = xy \quad \forall x, \forall y$$

записывается как:

```
prog = soseq(prog, [p - x^2; p*q - x*y])
```

Ограничения типа неравенств для скалярных полиномов

Используется функция **sosineq**.

Неравенство

$$p(x, y) - x^2y^2 \geq 0 \quad \forall x, \forall y$$

записывается как

```
prog = sosineq(prog, p - x^2*y^2)
```

Точнее, выражение $p(x, y) - x^2y^2$ будет представлено как SOS.

Функцию **sosineq** следует использовать только для скалярных неравенств. Для векторных неравенств используйте **sosmatrixineq**.

Ограничения типа скалярных неравенств (продолжение)

Если $p = p(x)$, где x — скаляр, то для x можно указать диапазон:

```
prog = sosineq(prog, p - x^2, [-1 1])
```

означает

$$p(x) - x^2 \geq 0, \quad -1 \leq x \leq 1.$$

Можно использовать **-Inf**, **Inf**.

Если указать

```
prog = sosineq(prog, p - x^2, 'sparse')
```

то программа будет стремиться минимизировать число использованных мономов.

Ограничения типа неравенств для матричных полиномов

Используется функция [sosmatrixineq](#). В SOSTOOLS матричные неравенства можно задавать двумя разными способами. При этом ограничения будут иметь немного разный смысл.

Пусть $M(\bar{x})$ — квадратная симметричная полиномиальная матрица. Тогда неравенство $M(\bar{x}) \geq 0$ может пониматься так:

- 1-й способ: $y^T M(\bar{x}) y \geq 0$ при всех y и \bar{x} .
- 2-й способ: существует матрица $G(\bar{x})$ такая что, $M(\bar{x}) = G^T(\bar{x})G(\bar{x})$ при всех \bar{x} . (Это означает, что существует SOS-представление.)

Этим двум случаям отвечают разные третий параметры [sosmatrixineq](#):

- 1-й способ: 'quadraticMineq'
- 2-й способ: 'Mineq'

Пример: ограничения типа матричных неравенств

Пример. Неравенство

$$P(x, y) \geq 0 \quad \forall x, \forall y,$$

где P — 2×2 -матрица.

```
syms x y
prog = sosprogram([x y]);
Z = monomials([x y],0:2); % мономы степеней 0,1,2
[prog,P] = sopolymatrixvar(prog,Z,[2 2],'symmetric');
prog = sosmatrixineq(prog,P,'quadraticMineq')
```

Задание целевой функции

Для задачи разрешимости (feasibility problem) целевая функция не задается. Для задания целевой функции используется функция [sossetobj](#). Функция должна быть [линейной](#)!

Пусть требуется [минимизировать](#) целевую функцию, зависящую от независимых переменных x, y, z :

$$f(x, y, z) = 2x - y + 3z$$

Определяем

```
prog = sossetobj(prog,2*x-y+3*z);
```

В качестве переменной можно брать также параметры [coeff_n](#). (В последнем случае при создании полиномиальной переменной нужно указать параметр 'wscoeff')

Решение задачи: вызов солвера

Используем функцию `sosssolve`:

```
prog = sosssolve(prog);
```

Более сложный вызов:

```
prog = sosssolve(prog,options);
```

По умолчанию

```
options.solver = 'SeDuMi';
```

Можно поменять солвер, например:

```
options.solver = 'sdpt3';
```

С помощью `options` можно настроить точность вычислений.

Для получения дополнительной информации (разрешимость, время CPU и пр.):

```
[prog, info] = sosssolve(prog);
```

Как убедиться, что задача решена успешно?

Пример успешно решенной задачи:

```
info =
struct with fields:
    iter: 4
    feasratio: 1.0000
    pinf: 0
    dinf: 0
    numerr: 0
    timing: [0.3120 0.7650 0.0620]
    wallsec: 1.1390
    cpusec: 1.2168
```

Признаки успеха:

```
feasratio >= 1, pinf=dinf=0, numerr=0
```

Система ограничений несовместна: `feasratio=-1`, $(-1, 1)$ — сомнительный случай.

Получение решений задачи

С помощью функции `sosgetsol` можно извлечь значения переменных после окончания работы солвера.

Если `p` — имя переменной, то ее значение можно получить как

```
sol_p = sosgetsol(prog,p);
```

По умолчанию берется пять значащих цифр. Для увеличения точности представления до десяти цифр:

```
sol_p = sosgetsol(prog,p,10);
```

Стандартные функции

Кроме универсальной функции `sosssolve` в `SOSTOOLS` имеется три функции для решения важных частных задач:

- `findsos` — нахождение SOS-представления для скалярного полинома или полиномиальной матрицы.
- `findlyap` — нахождение функции Ляпунова для системы дифференциальных уравнений с полиномиальной правой частью. Функция ищется в виде скалярного полинома.
- `findbound` — нахождение глобальной нижней границы (глобального минимума) для скалярного полинома. Могут присутствовать полиномиальные ограничения типа неравенств или равенств (условная оптимизация).

Лекция 12b. Пакет SOSTOOLS. Примеры

Задача 1. Проверка неотрицательности многочлена

Проверить неотрицательность скалярного или матричного многочлена можно двумя способами:

- использовать специальную функцию findsos,
- использовать универсальную функцию sossolve.

Функция findsos: скалярный полином

Для заданного многочлена $p(\bar{x})$ проверить условие $p(\bar{x}) \geq 0$ при всех \bar{x} . Ищется представление вида

$$p(\bar{x}) = Z^T(\bar{x})QZ(\bar{x}) \quad \forall \bar{x},$$

где $Q^T = Q \geq 0$, $Z(\bar{x})$ — вектор из мономов (feasibility problem). Простейший вызов:

```
[Q,Z]=findsos(p)
```

Если $p(\bar{x})$ не является SOS-полиномом, то $Q = []$, $Z = []$.

Функция findsos: скалярный полином (продолжение)

Общий вид функции:

```
[Q,Z,F,dig]=findsos(p,flag,options)
```

Здесь F — вектор из полиномов,

$$p(\bar{x}) = \sum_k (F_k(\bar{x}))^2,$$

dig — точность представления. Если $flag = 'rational'$, то коэффициенты в F считаются рациональными числами. С помощью $options$ можно выполнить настройку. Например:

```
options.solver = 'sdpt3'; % 'SeDuMi' по умолчанию  
options.params.tol = 1E-9; % точность вычисления  
% (tolerance)
```

Скалярный полином: программа

$$p(x,y) = 4x^2y^2 + x^2 - xy + y^2$$

Ищется представление вида $p(\bar{x}) = Z^T(\bar{x})QZ(\bar{x})$, $Q \geq 0$.

```
syms x y;  
p = 4*x^2*y^2+x^2-x*y+y^2;  
[Q,Z]=findsos(p)
```

Получаем:

```
Q =  
1.0000 -0.5000 0.0000  
-0.5000 1.0000 -0.0000  
0.0000 -0.0000 4.0000  
Z =  
x  
y  
x*y
```

Функция findsos: матричный полином

Пусть $P(\bar{x})$ — полином, все значения которого — симметричные $m \times m$ -матрицы.

Ищется представление вида

$$P(\bar{x}) = (I_m \otimes Z)^T(\bar{x})Q(I_m \otimes Z(\bar{x})),$$

где $Q^T = Q \geq 0$, $Z(\bar{x})$ — вектор из мономов, \otimes — произведение Кронекера.
Например, при $m = 2$:

$$P(\bar{x}) = \begin{bmatrix} Z^T & 0 \\ 0 & Z^T \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12} & Q_{22} \end{bmatrix} \begin{bmatrix} Z & 0 \\ 0 & Z \end{bmatrix}$$

Функция в этом случае работает нестабильно!

Матричный полином: программа

```
P = [4*x^2*y^2+x^2-x*y+y^2      0; 0      x^2+y^2+x*y];
[Q,Z]=findsos(P)
```

Получаем матрицу 6×6 :

```
Q =
 1.0000   -0.5000    0.0000    0.0000   -0.0000    0.0000
 -0.5000    1.0000   -0.0000    0.0000   -0.0000   -0.0000
 0.0000   -0.0000    4.0000    0.0000    0.0000   -0.0000
 0.0000    0.0000    0.0000    1.0000    0.5000   -0.0000
 -0.0000   -0.0000    0.0000    0.5000    1.0000   -0.0000
 0.0000   -0.0000   -0.0000   -0.0000   -0.0000    0.0000

Zm =
 x
 y
 x*y
```

Проверка неотрицательности многочлена (способ 2)

Используем универсальную функцию sossolve.

```
syms x y;
prog = sosprogram([x y]);
p = 4*x^2*y^2+x^2-x*y+y^2;
% неизвестных здесь нет
prog = sosineq(prog,p); % неравенство p >= 0
[prog,info] = sossolve(prog);
test = info.feasratio
```

Получаем:

```
test = 1.0000
```

Значение поля `feasratio>=1` говорит, что задача разрешима (`feasible`),
`-1` — неразрешима (`infeasible`), промежуточные значения отвечают неясной
ситуации.

Проверка неотрицательности многочлена (способ 3)

Вновь используем `sosolve`, но по-другому. Определим неизвестный SOS-полином q ($q \geq 0$) и проверим неравенство $p \geq q$.

```
syms x y;
prog = sosprogram([x; y]);
p = 4*x^2*y^2+x^2-x*y+y^2;
[prog, q] = sososvar(prog, [x; y; x^2; y^2; x*y]);
prog = soseq(prog,p-q); % условие p = q
[prog,info] = sosolve(prog);
test = info.feasratio
```

Снова получаем:

```
test = 1.0000
```

Пример. Круговой критерий

Пусть $W(s)$ — передаточная функция, μ_1, μ_2 — скаляры, $\mu_1 < \mu_2$. Проверим неравенство

$$1 + (\mu_1 + \mu_2) \operatorname{Re} W(i\omega) + \mu_1 \mu_2 |W(i\omega)|^2 \geq 0, \quad -\infty < \omega < \omega.$$

Пусть $W(s) = M(s)/N(s)$, где $M(s), N(s)$ — многочлены. Определим многочлен

$$\begin{aligned} P(s) = N(s)N(-s) + \frac{1}{2}(\mu_1 + \mu_2) &\left(M(s)N(-s) + M(-s)N(s) \right) \\ &+ \mu_1 \mu_2 M(s)M(-s). \end{aligned}$$

Частотное неравенство принимает вид:

$$Q(\omega) = P(i\omega) \geq 0, \quad -\infty < \omega < \omega.$$

Достаточно проверить, что $Q(\omega)$ допускает SOS-представление.

Круговой критерий: программа

```
syms s omega
mu2 = 0.5; mu1 = - 0.4;
M = s + 1;
N = s^3 + s^2 + s + 2;
Mm = subs(M,s,-s); % M(-s)
Nm = subs(N,s,-s); % N(-s)
P = N*Nm + 0.5*(mu1+mu2)*(M*Nm+Mm*N);
P = simplify(P + mu1*mu2*M*Mm)
Q = simplify(subs(P,s,i*omega))
[M,Z]= findsos(Q)
eig(M)
```

Получаем:

```
P = - s^6 - (11*s^4)/10 + (16*s^2)/5 + 4
Q = omega^6 - (11*omega^4)/10 - (16*omega^2)/5 + 4
```

Круговой критерий: результат

Получаем $Q = Z^T M Z$,

```
M =
4.0000 -0.0000 -2.7095 0.0000
-0.0000 2.2190 -0.0000 -1.4868
-2.7095 -0.0000 1.8736 0.0000
0.0000 -1.4868 0.0000 1.0000

Z =
1
omega
omega^2
omega^3
```

Собственные значения M :

5.8474 0.0262 3.2164 0.0026

$(M > 0)$

Задача 2. Нахождение функции Ляпунова

Найти функцию Ляпунова для системы с полиномиальной правой частью можно двумя способами:

- использовать специальную функцию `findlyap`,
- использовать универсальную функцию `sosolve`.

Функция `findlyap`

Служит для вычисления полиномиальной функции Ляпунова системы с полиномиальными правыми частями:

`V = findlyap(f,X,deg,options)`

Здесь f — правая часть системы (вектор полиномов), X — вектор независимых переменных,

$$\dot{X} = f(X),$$

`deg` — степень функции Ляпунова (четное число), `options` — настроечные параметры (можно задать солвер). Функция $V(X)$ положительно определенная, неограничена при больших x , ее производная отрицательно определена вдоль решений системы. Если задача не имеет решения, то $V = []$.

Пример 2-1. Функция Ляпунова с помощью `findlyap`

Построим функцию Ляпунова для системы:

$$\dot{x}_1 = -x_1^3 + x_2, \quad \dot{x}_2 = -x_1 - x_2.$$

Код:

```
syms x1 x2;
f = [-x1^3+x2; -x1-x2];
V = findlyap(f,[x1; x2],2)
```

Получаем:

$$V = 1.195*x1^2 + 5.801e-5*x1*x2 + 1.195*x2^2$$

Пример 2-2. Проверка гурвицевости многочлена

Рассмотрим полином:

$$p(s) = s^n + p_1 s^{n-1} + \dots + p_{n-1} s + p_n.$$

Является ли полином гурвицевым? Корни $p(s)$ совпадают с собственными числами сопровождающей матрицы (матрицы Фробениуса) (команда `compan(p)` — companion matrix):

$$A = \begin{bmatrix} -p_n & -p_{n-1} & \cdots & -p_1 \\ 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

Построим квадратичную функцию Ляпунова для системы

$$\dot{X} = AX$$

Пример 2-2. Проверка гурвицевости многочлена (продолжение)

Построим квадратичную функцию Ляпунова для системы

$$\dot{X} = AX$$

Ищем полиномиальную функцию Ляпунова $V = V(X)$:

$$V \geq \varepsilon \|X\|^2, \quad \varepsilon > 0;$$
$$\dot{V} = \nabla V(X)^\top AX \leq -\varepsilon \|X\|^2$$

Выбирая $W = \frac{1}{\varepsilon}V$, можно взять $\varepsilon = 1$.

Пример 2-2. Проверка гурвицевости многочлена (продолжение)

```

syms x1 x2 x3;
X = [x1; x2; x3];
p = [1 6 11 6];
A =compan(p); f = A*X;
Z = monomials(X,2); eps = 1e-6;
prog = sosprogram(X);
[prog,V] = sospolyvar(prog,Z);
prog = sosineq(prog,V-eps*(x1^2+x2^2+x3^2));
expr = -diff(V,x1)*f(1)-diff(V,x2)*f(2)-diff(V,x3)*f(3);
expr = expr - eps*(x1^2+x2^2+x3^2);
prog = sosineq(prog,expr);
[prog,info] = sossolve(prog);
if info.feasratio >= 1
    VL = sosgetsol(prog,V)
else
    disp('No solution');
end

```

Пример 2-2. Проверка гурвицевости многочлена (продолжение)

Получаем:

```
VL =
0.08746*x1^2 + 0.141*x1*x2 + 0.109*x1*x3
+ 1.102*x2^2 + 0.7391*x2*x3 + 0.876*x3^2
```

Аналогично:

```
U = findlyap(f,X,2)
```

дает

```
U =
0.4067*x1^2 + 0.3874*x1*x2 + 0.4207*x1*x3
+ 4.564*x2^2 + 2.232*x2*x3 + 3.153*x3^2
```

Пример 2-3. Функция Ляпунова

Рассмотрим систему уравнений с рациональной правой частью

$$\begin{aligned}\dot{x}_1 &= -x_1^3 - x_1 x_3^2, \\ \dot{x}_2 &= -x_2 - x_1^2 x_2 \\ \dot{x}_3 &= -x_3 - \frac{3x_3}{x_3^2 + 1} + 3x_1^2 x_3.\end{aligned}$$

Будем искать полиномиальную функцию Ляпунова $V = V(x_1, x_2, x_3)$:

$$\begin{aligned}V &\geq \varepsilon (x_1^2 + x_2^2 + x_3^2), \quad \varepsilon > 0; \\ \dot{V} &= \frac{\partial V}{\partial x_1} \dot{x}_1 + \frac{\partial V}{\partial x_2} \dot{x}_2 + \frac{\partial V}{\partial x_3} \dot{x}_3 \leq 0.\end{aligned}$$

Заменим $V \mapsto V/\varepsilon$ и умножим на $x_3^2 + 1$:

$$\begin{aligned}V &\geq x_1^2 + x_2^2 + x_3^2; \\ -(x_3^2 + 1) \frac{\partial V}{\partial x_1} \dot{x}_1 - (x_3^2 + 1) \frac{\partial V}{\partial x_2} \dot{x}_2 - (x_3^2 + 1) \frac{\partial V}{\partial x_3} \dot{x}_3 &\geq 0.\end{aligned}$$

Пример 2-3. Функция Ляпунова (продолжение)

```
syms x1 x2 x3;
X = [x1; x2; x3];
f = [-x1^3-x1*x3^2; -x2-x1^2*x2;
      -x3+3*x1^2*x3-3*x3/(x3^2+1)];
f = f * (x3^2+1);
Z = [x1^2; x2^2; x3^2];
eps = 1e-6;
prog = sosprogram(X);
[prog,V] = sospolyvar(prog,Z); % полиномиальная переменная
prog = sosineq(prog,V-eps*(x1^2+x2^2+x3^2));
expr = -diff(V,x1)*f(1)-diff(V,x2)*f(2)-diff(V,x3)*f(3);
prog = sosineq(prog,expr);
prog = sossolve(prog);
solV = sosgetsol(prog,V)
```

Получаем:

```
solv = 0.6309*x1^2 + 0.5163*x2^2 + 0.2054*x3^2
```

Задача 3. Нахождение глобального минимума скалярного полинома

Пусть $p(\bar{x})$ — скалярный многочлен четной степени. Рассматривается задача:

$$\gamma \rightarrow \max, \quad p(\bar{x}) \geq \gamma, \quad \forall \bar{x},$$

на \bar{x} можно задать ограничения типа неравенств или равенств.

Найти минимум полинома (с ограничениями или без) можно найти двумя способами:

- использовать специальную функцию `findbound`,
- использовать универсальную функцию `sosolve`.

Функция `findbound`: глобальный минимум

Пусть $p(X)$ — скалярный полином четной степени, `gamma` — скалярная переменная.

Ищем максимальная `gamma`, `p-gamma` — SOS.

```
[gamma,vars,xopt] = findbound(p,options)
```

`xopt` — точка минимума, `vars` — переменные, такие что минимум `p` достигается при `vars = opt` (`vars` может включать не весь `X`). Если задача не имеет решения, `xopt = []`

Функция `findbound`: программа

```
clear all; clc
syms x1 x2
p = x1^2+x2^2-0.5*x1 +10;
fmesh(p); % строим график
[gamma,vars,xopt] = findbound(p)
```

Получаем:

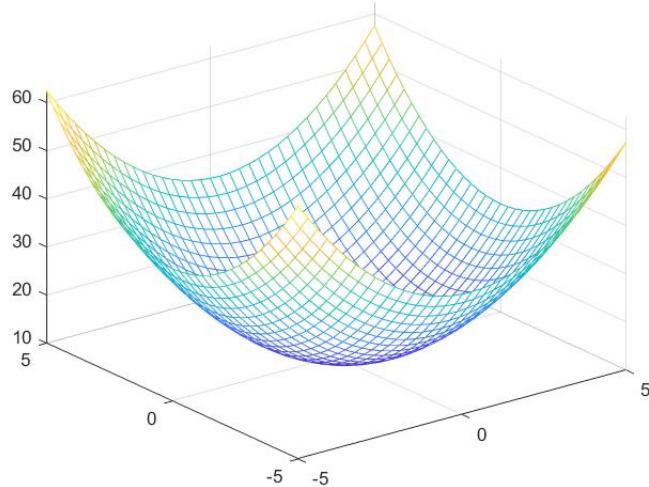
```
gamma =
9.9375

vars =
x1
x2

xopt =
0.2502
0.0000
```

График функции

Получаем `gamma = 9.9375`



Функция `findbound`: минимум при ограничениях

`[gamma, vars, xopt] = findbound(p, g, h, deg, options)`

Задача:

$$\gamma \rightarrow \max, \quad p(X) \geq \gamma, \quad g(X) \geq 0, \quad h(X) = 0$$

Ограничения g, h — массивы полиномиальных ограничений. Точнее: ищутся полиномиальные вектора $d_1(X), d_2(X)$, где элементы d_2 — суммы квадратов и

$$(p(X) - \gamma) - d_1^T(X)h(X) - d_2^T(X)g(X) \quad \text{— SOS}$$

Здесь \deg — степени $d_1(X), d_2(X)$. Чем выше степень, тем лучше оценка γ и тем больше объем вычислений.

Минимум при ограничениях: пример

Задача:

$$x_1 + x_2 \rightarrow \min, \quad x_1 \geq 0, \quad x_2 \geq 0.5, \quad x_1^2 + x_2^2 = 1, \quad x_2 = x_1^2 + 0.5$$

Программа:

```
clc; clear all
syms x1 x2;
degree = 4;
[gamma, vars, opt] = findbound(x1+x2, [x1, x2-0.5], ...
[x1^2+x2^2-1, x2-x1^2-0.5], degree)
```

Результат:

```
gamma =
1.3911
```

Использование функции `sosssolve`

Найдем **глобальный** минимум полиномиальной функции: $\gamma_0 = \inf_{\bar{x}} f(\bar{x}) = \sup\{\gamma : f(\bar{x}) \geq \gamma\}$.

```
syms x1 x2 gam;
prog = sosprogram([x1, x2]);
prog = sosdecvar(prog,gam);
f = x1^2+x2^2-0.5*x1 +10;
fmesh(f); % график функции
prog = sosineq(prog,f-gam); % f >= gam
prog = sossetobj(prog,-gam); % gam -> max
prog = sossolve(prog);
if info.feasratio >= 1
    minf = sosgetsol(prog,gam)
end
```

Получаем

```
minf = 9.938
```

Лекция 12с. SOS-модуль в пакете YALMIP

Модуль SOS в пакете YALMIP

YALMIP содержит свой модуль функций для задач SOS, отличный от `SOSTOOLS`. Возможности пакетов не совпадают.

Основные этапы:

- Независимые переменные описываются с помощью команд `sdpvar` (а не `sym`).
- Создание SOS-ограничения: $F = \text{sos}(p)$, где p — переменная типа `sdpvar`. (Это метод класса `sdpvar`.)
- Нахождение SOS-представления вида $p = h^T h$: `solvesos`
- Извлечение множителя h : `sosd`.
- Вывод полинома h на экран: `sdisplay`.
- Отбросить коэффициенты меньше eps : `clean(p, eps)`

Пример: задача разрешимости

Найдем SOS-представление многочлена:

$$p(x, y) = x^2 + y^2 + 2x - 2y + 2$$

```
clc; clear all
x = sdpvar(1); y = sdpvar(1);
p = x^2 + y^2 + 2*x - 2*y + 2;
F = sos(p);
options = sdpsettings('solver','sedumi');
sol = solvesos(F, [], options);
if sol.problem == 0
    h = sosd(F);
    h = clean(h, 1e-6);
    q1 = sdisplay(h)
    d = clean(p - h'*h, 1e-6); % проверка p=h'*h
    q2 = sdisplay(d)
end
```

SOS-представление (результат)

```
q1 =
3x1 cell array
'-1.41421356237-0.707106781187*x+0.707106781187*y'
'-0.707106781187*x-0.707106781187*y'
'0'

q2 =
cell
'0'
```

Здесь q_1, q_2 получаются как массивы ячеек, содержащие полиномы. Получаем: $q_2 = 0$,

$$\begin{aligned} q_1 &= \left(-\sqrt{2} - \frac{1}{\sqrt{2}}x + \frac{1}{\sqrt{2}}y \right)^2 + \left(-\frac{1}{\sqrt{2}}x - \frac{1}{\sqrt{2}}y \right)^2, \\ &= x^2 + y^2 + 2x - 2y + 2. \end{aligned}$$

Пример: задача разрешимости

Представим $p = v^T Q v$, v — вектор мономов, $Q^T = Q$.

```
x = sdpvar(1); y = sdpvar(1);
p = (1+x)^4 + (1-y)^2;
F = sos(p);
options = sdpsettings('solver','sedumi');
[sol,v,Q] = solvesos(F,[],options);
sdisplay(v{1})
value(Q{1})
clean(p-v{1}.*Q{1}.*v{1},1e-6) // проверка = 0
```

Получим:

```
v = '1'      Q = 2.0000  -1.0000  2.0000  0.9996
      'y'       -1.0000   1.0000  0.0000  0.0000
      'x'        2.0000   0.0000  4.0009  2.0000
      'x^2'     0.9996   0.0000  2.0000  1.0000
```

Пример: использование целевой функции

Глобальная нижняя оценка многочлена

```
sdpvar x y lower
p = (1+x*y)^2-x*y+(1-y)^2;
F = sos(p-lower); % p >= lower
options = sdpsettings('solver','sedumi');
solvesos(F,-lower,options);
value(lower)
```

Результат: 0.75

Второй параметр `solvesos` — целевая функция (минимум),

Лекция 13. Двойственность в задачах SDP: часть 1

Пространство \mathbb{S}^n

Рассмотрим \mathbb{S}^n — множество симметричных вещественных $n \times n$ -матриц:

$$\mathbb{S}^n = \{A \in \mathbb{R}^{n \times n} : A^T = A\}$$

Оно представляет собой линейное пространство:

- 1) $A \in \mathbb{S}^n, \alpha \in \mathbb{R} \implies \alpha A \in \mathbb{S}^n$.
- 2) $A \in \mathbb{S}^n, B \in \mathbb{S}^n \implies A + B \in \mathbb{S}^n$.

Превратим \mathbb{S}^n в евклидово пространство, т. е. в линейное пространство со скалярным произведением.

Основная теорема

Теорема. Функция $\langle A, B \rangle = \text{tr } AB$, т. е. след произведения AB , является скалярным произведением в \mathbb{S}^n .

След матрицы — сумма ее диагональных элементов.

Доказательство: свойства скалярного произведения

Скалярное произведение в линейном пространстве над \mathbb{R} определяется тремя свойствами:

1. Симметричность

$$\langle B, A \rangle = \langle A, B \rangle$$

2. Билинейность

$$\begin{aligned} \langle \alpha_1 A_1 + \alpha_2 A_2, B \rangle &= \alpha_1 \langle A_1, B \rangle + \alpha_2 \langle A_2, B \rangle, \\ \langle A, \alpha_1 B_1 + \alpha_2 B_2 \rangle &= \alpha_1 \langle A, B_1 \rangle + \alpha_2 \langle A, B_2 \rangle, \quad \forall \alpha_1, \alpha_2 \in \mathbb{R} \end{aligned}$$

3. Положительность

$$\langle A, A \rangle \geq 0; \quad \langle A, A \rangle = 0 \iff A = 0$$

Доказательство симметричности

Циклическое свойство следа матрицы: $A \in \mathbb{R}^{n \times k}, B \in \mathbb{R}^{k \times n} \implies \text{tr } AB = \text{tr } BA$.

Доказательство:

$$\begin{aligned} \text{tr } AB &= \sum_{i=1}^n (AB)_{ii} = \sum_{i=1}^n \sum_{j=1}^k A_{ij} B_{ji} = \sum_{j=1}^k \sum_{i=1}^n B_{ji} A_{ij} = \sum_{j=1}^k (BA)_{jj} = \\ &= \text{tr } BA \end{aligned}$$

Доказательство билинейности

Если α — скаляр, то

$$\operatorname{tr}(\alpha AB) = \sum_i (\alpha AB)_{ii} = \alpha \sum_i (AB)_{ii} = \alpha \operatorname{tr} AB,$$

$$\begin{aligned} \operatorname{tr}(A_1 + A_2)B &= \sum_i ((A_1 + A_2)B)_{ii} = \sum_i ((A_1 B)_{ii} + (A_2 B)_{ii}) = \\ &= \operatorname{tr} A_1 B + \operatorname{tr} A_2 B. \end{aligned}$$

Аналогично $\operatorname{tr} A(B_1 + B_2) = \operatorname{tr} AB_1 + \operatorname{tr} AB_2$.

Доказательство положительности

Положительность. Если $A \in \mathbb{S}^n$, то

$$\langle A, A \rangle \geq 0; \quad \langle A, A \rangle = 0 \iff A = 0$$

Доказательство: Пусть $A = [A_1, \dots, A_n]$, A_i — n -столбцы. Так как $A^T = A$, то

$$\begin{aligned} \operatorname{tr} A^2 &= \operatorname{tr} A^T A = \operatorname{tr} \begin{bmatrix} A_1^T \\ \vdots \\ A_n^T \end{bmatrix} [A_1, \dots, A_n] = \operatorname{tr} \begin{bmatrix} A_1^T A_1 & & \\ & \ddots & \\ & & A_n^T A_n \end{bmatrix} = \\ &= A_1^T A_1 + \dots + A_n^T A_n = \|A_1\|^2 + \dots + \|A_n\|^2, \end{aligned}$$

где $\|\cdot\|$ — векторная евклидова норма.

Значит $\operatorname{tr} A^2 = 0 \iff A_i = 0, \forall i \iff A = 0$.

Линейные функционалы на \mathbb{S}^n

Нас будут интересовать линейные функционалы, т. е. линейные отображения вида:

$$f : \mathbb{S}^n \mapsto \mathbb{R}$$

Как их можно определить?

Общий вид линейных функционалов

Пусть \mathbb{E} — произвольное евклидово пространство, $\langle \cdot, \cdot \rangle$ — скалярное произведение в \mathbb{E} , $f : \mathbb{E} \mapsto \mathbb{R}$ — линейный функционал.

Теорема Рисса:

для всякого линейного функционала $f(x)$ в \mathbb{E} существует и единственен $C \in \mathbb{E}$ такой, что $f(x) = \langle C, x \rangle, \forall x \in \mathbb{E}$.

Следствие для \mathbb{S}^n :

для всякого линейного функционала $f(X) : \mathbb{S}^n \mapsto \mathbb{R}$ существует и единственная матрица $C \in \mathbb{S}^n$ такая, что $f(X) = \operatorname{tr} CX = \operatorname{tr} XC, \forall X \in \mathbb{S}^n$.

Гиперплоскость в \mathbb{S}^n

Гиперплоскость в евклидовом пространстве \mathbb{E}^n :

$$\Pi = \{ X \in \mathbb{E}^n : \langle C, X \rangle = \alpha \}, \quad C \in \mathbb{E}^n, \quad C \neq 0, \quad \alpha \in \mathbb{R}$$

Гиперплоскость в \mathbb{S}^n :

$$\Pi = \{ X \in \mathbb{S}^n : \text{tr } CX = \alpha \}, \quad C \in \mathbb{S}^n, \quad C \neq 0, \quad \alpha \in \mathbb{R}$$

Множество неотрицательно определенных матриц \mathbb{S}_+^n

Определим подмножество \mathbb{S}^n :

$$\mathbb{S}_+^n = \{ A \in \mathbb{S}^n : A \geq 0 \} \subset \mathbb{S}^n$$

Очевидно \mathbb{S}_+^n — выпуклый конус в \mathbb{S}^n :

- 1) $A \in \mathbb{S}_+^n, \alpha \geq 0 \implies \alpha A \geq 0 \implies \alpha A \in \mathbb{S}_+^n$.
- 2) $A_1, A_2 \in \mathbb{S}_+^n, 0 \leq \alpha \leq 1 \implies \alpha A_1 + (1 - \alpha) A_2 \geq 0 \implies \alpha A_1 + (1 - \alpha) A_2 \in \mathbb{S}_+^n$.

Лемма 1 о свойствах \mathbb{S}_+^n

Лемма 1

Пусть $A \in \mathbb{S}_+^n$. Тогда: $\text{tr } A \geq 0$ и $\text{tr } A = 0 \iff A = 0$.

Доказательство. Представим: $A = XX^T$, $X = [X_1, \dots, X_k]$, X_i — n -мерные столбцы (см. приведение симм. матрицы к диаг. виду).

Тогда

$$\begin{aligned} \text{tr } A &= \text{tr } XX^T = \text{tr } X^T X = \\ &= \text{tr} \begin{bmatrix} X_1^T \\ \vdots \\ X_k^T \end{bmatrix} \begin{bmatrix} X_1 & \dots & X_k \end{bmatrix} = \text{tr} \begin{bmatrix} X_1^T X_1 & & \\ & \ddots & \\ & & X_k^T X_k \end{bmatrix} = \\ &= \|X_1\|^2 + \dots + \|X_k\|^2 \geq 0, \end{aligned}$$

Отсюда

$$\text{tr } A = 0 \iff X_i = 0, \forall i \iff X = 0 \implies A = 0.$$

Лемма 2 о свойствах \mathbb{S}_+^n

Лемма 2

Пусть $A \in \mathbb{S}_+^n$, $B \in \mathbb{S}_+^n$. Тогда: 1) $\text{tr } AB \geq 0$; 2) $\text{tr } AB = 0 \iff AB = 0$; 3) $\text{tr } AB = 0, B > 0 \implies A = 0$.

Доказательство. Представим: $A = XX^T$, $B = YY^T$. Тогда

$$\text{tr } AB = \text{tr } XX^T YY^T = \text{tr } Y^T XX^T Y = \text{tr } M^T M, \quad M = X^T Y$$

При этом $M^T M \geq 0$ и $AB = XMY^T$. Применим лемму 1 к $M^T M$: $\text{tr } M^T M \geq 0$,

$$\text{tr } M^T M = 0 \iff M^T M = 0.$$

Тогда $x^T M^T M x = 0, \forall x \implies \|Mx\|^2 = 0, \forall x \implies Mx = 0, \forall x \implies M = 0 \implies AB = 0$.

Лемма 3 о свойствах \mathbb{S}_+^n

Лемма 3

Пусть $A \in \mathbb{S}^n$. Пусть $\text{tr } AB \geq 0, \forall B \in \mathbb{S}_+^n$. Тогда $A \in \mathbb{S}_+^n$.

Доказательство. Доказательство от противного. Пусть $A \not\succeq 0$. Тогда $\exists q \in \mathbb{R}^n, q^T A q < 0$. Возьмем $B = qq^T \geq 0$. Тогда

$$\text{tr } AB = \text{tr } Aqq^T = q^T A q < 0.$$

Получено противоречие с предположением $A \not\succeq 0$.

Прямая задача SDP

Прямая задача SDP (primal problem):

$$c^T x \rightarrow \min, \quad F(x) = F_0 + \sum_{i=1}^m x_i F_i \geq 0, \quad F_i \in \mathbb{S}^n, \quad c \in \mathbb{R}^m, \quad x \in \mathbb{R}^m.$$

Пусть множество допустимых векторов $\Omega = \{x : F(x) \geq 0\}$ непусто.
Обозначим:

$$p^* = \inf_{F(x) \geq 0} c^T x \quad (\text{возможно } p^* = -\infty)$$

Задача FEASP

FEASP

Существует ли $x : F(x) \geq 0$?

Рассмотрим задачу SDP: пусть t — дополнительная скалярная переменная.

$$t \rightarrow \inf, \quad F(x) + tI_n \geq 0.$$

Множество допустимых точек (x, t) этой задачи **всегда непусто**: возьмем x — произвольно и t достаточно большим, $tI_n > -F(x)$.

Утверждение: Задача FEASP разрешима $\iff t^{\text{opt}} \leq 0$

Замечание. Если на каком-то шаге оптимизации получаем $t \leq 0$, то неравенство разрешимо и оптимум можно не искать.

Функция Лагранжа в классической постановке

Рассмотрим задачу условной оптимизации:

$$g(x) \rightarrow \min, \quad f_j(x) \geq 0, \quad j = 1, \dots, m, \quad x \in \mathbb{R}^n.$$

Пусть $g(x)$, $f_j(x)$ выпуклы, рассмотрим вектора

$$f(x) = [f_1(x) \dots f_m(x)]^T, \quad \lambda = [\lambda_1 \dots \lambda_m]^T,$$

λ — вектор множителей Лагранжа. Определим функцию Лагранжа:

$$L(x, \lambda) = g(x) - \sum_{j=1}^m \lambda_j f_j(x) = g(x) - \lambda^T f(x) = g(x) - \langle \lambda, f(x) \rangle.$$

Тогда экстремум x^0 удовлетворяет системе

$$\nabla_x L(x^0, \lambda) = \nabla g(x^0) - \sum_{j=1}^m \lambda_j \nabla f_j(x^0) = 0, \quad \lambda_j \geq 0, \quad j = 1, \dots, m.$$

Это условие является необходимым, но не достаточным!

Функция Лагранжа для прямой задачи SDP

Рассмотрим прямую задачу:

$$p^* = \inf_{F(x) \geq 0} c^T x.$$

Определим функцию Лагранжа

$$L(x, Z) = c^T x - \langle Z, F(x) \rangle = c^T x - \text{tr} Z F(x), \quad Z \in \mathbb{S}_+^n$$

Z — матрица множителей Лагранжа, аналог вектора λ в классическом случае. Условие $Z \in \mathbb{S}_+^n$ аналогично условию $\lambda_j \geq 0$, $j = 1, \dots, m$, в классическом случае.

Теорема 1 (теорема о минимаксе).

$$p^* = \inf_x \sup_{Z \geq 0} L(x, Z)$$

Доказательство теоремы о минимаксе

Нужно доказать:

$$p^* = \inf_{F(x) \geq 0} c^T x = \inf_x \sup_{Z \geq 0} L(x, Z), \quad L(x, Z) = c^T x - \text{tr} Z F(x)$$

Теорема вытекает из формулы:

$$\sup_{Z \geq 0} L(x, Z) = \begin{cases} c^T x & \text{при } x : F(x) \geq 0, \\ +\infty & \text{при других } x. \end{cases}$$

1) Пусть $F(x) \geq 0$. Так как $Z \geq 0$, то $\text{tr} Z F(x) \geq 0$, отсюда

$$L(x, Z) \leq c^T x \quad \text{при } F(x) \geq 0 \quad (\text{равенство достигается при } Z = 0)$$

2) Пусть $F(x) \not\geq 0$. Тогда $\exists z_0 \in \mathbb{R}^n$, $z_0^T F(x) z_0 < 0$. Положим

$$Z = \alpha z_0 z_0^T, \quad \alpha > 0 \quad \Rightarrow$$

$$\text{tr} Z F(x) = \alpha \text{tr} z_0 z_0^T F(x) = \alpha z_0^T F(x) z_0 \xrightarrow[\alpha \rightarrow +\infty]{} -\infty$$

Преобразование функции Лагранжа

Функцию Лагранжа

$$L(x, Z) = c^T x - \text{tr} Z F(x)$$

можно эквивалентно переписать в виде

$$L(x, Z) = -\text{tr} Z F_0 + \sum_{i=1}^m x_i (c_i - \text{tr} Z F_i),$$

Доказательство. Перепишем второе равенство:

$$\begin{aligned} L(x, Z) &= -\text{tr} Z F_0 + \sum_{i=1}^m x_i (c_i - \text{tr} Z F_i) = \\ &= \sum_{i=1}^m c_i x_i - \text{tr} Z \left(F_0 + \sum_{i=1}^m x_i F_i \right) = \\ &= c^T x - \text{tr} Z F(x). \end{aligned}$$

Двойственная задача

Теперь вместо минимакса рассмотрим [максимин](#):

$$d^* = \sup_{Z \geq 0} \inf_x L(x, Z)$$

Теорема 2 (теорема о максимине). $d^* = -\text{tr} F_0 Z^{opt}$ есть решение задачи:

$$-\text{tr} F_0 Z \rightarrow \sup, \quad Z \geq 0, \quad \text{tr} F_i Z = c_i, \quad i = 1, \dots, m$$

Эта задача называется [двойственной](#) (dual problem).

Функцию Лагранжа для этой задачи удобно записать в форме:

$$L(x, Z) = -\text{tr} Z F_0 + \sum_{i=1}^m x_i (c_i - \text{tr} Z F_i),$$

где x_i — множители Лагранжа (могут быть любого знака).

Пояснение

В классическом случае для функции Лагранжа

$$L(x, \lambda) = f_0(x) - \sum_{i=1}^n \lambda_i f_i(x)$$

в точке оптимума x^0 выполнено:

$$\nabla_x L(x^0, \lambda) = \nabla f_0(x^0) - \sum_{i=1}^n \lambda_i \nabla f_i(x^0) = 0.$$

В данном случае рассматривается функция Лагранжа:

$$L(x, Z) = -\text{tr} Z F_0 + \sum_{i=1}^m x_i (c_i - \text{tr} Z F_i),$$

поэтому

$$\nabla_x L(x, Z) = 0 \iff c_i = \text{tr} Z F_i, \quad i = 1, \dots, m.$$

Доказательство теоремы о максимине

Мы рассматриваем двойственную задачу:

$$-\operatorname{tr} F_0 Z \rightarrow \sup, \quad Z \geq 0, \quad \operatorname{tr} F_i Z = c_i, \quad i = 1, \dots, m$$

Определим множество $\Delta = \{Z : \operatorname{tr} F_i Z = c_i, i = 1, \dots, m\}$. Тогда двойственную задачу можно переписать в виде:

$$-\operatorname{tr} F_0 Z \rightarrow \sup, \quad Z \geq 0, \quad Z \in \Delta$$

По определению d^* :

$$d^* = \sup_{Z \geq 0} \inf_x L(x, Z).$$

Нужно доказать, что

$$\sup_{Z \geq 0} \inf_x L(x, Z) = \sup_{Z \geq 0, Z \in \Delta} (-\operatorname{tr} Z F_0),$$

Доказательство теоремы о максимине (продолжение)

$$L(x, Z) = -\operatorname{tr} Z F_0 + \sum_{i=1}^m x_i (c_i - \operatorname{tr} Z F_i),$$

$$\Delta = \{Z : \operatorname{tr} F_i Z = c_i, i = 1, \dots, m\}$$

Убедимся, что:

$$\inf_x L(x, Z) = \begin{cases} -\operatorname{tr} Z F_0 & \text{при } Z \in \Delta, \\ -\infty, & \text{при } Z \notin \Delta. \end{cases}$$

Действительно, если $Z \in \Delta$, то $L(x, Z) = -\operatorname{tr} Z F_0$. Если $Z \notin \Delta$, то $\exists k : c_k - \operatorname{tr} Z F_k \neq 0$. Выберем $x_k = -\alpha(c_k - \operatorname{tr} Z F_k)$, $\alpha \in \mathbb{R}$ и $x_i = 0, \forall i \neq k$. Тогда

$$x_k(c_k - \operatorname{tr} Z F_k) = -\alpha(c_k - \operatorname{tr} Z F_k)^2 \rightarrow -\infty \quad \text{при } \alpha \rightarrow +\infty.$$

Отсюда

$$d^* = \sup_{Z \geq 0} \inf_x L(x, Z) = \sup_{Z \geq 0, Z \in \Delta} (-\operatorname{tr} Z F_0),$$

Слабая и сильная двойственность

Мы рассмотрели две задачи оптимизации — прямую и двойственную:

$$p^* = \inf_x \sup_{Z \geq 0} L(x, Z) \quad d^* = \sup_{Z \geq 0} \inf_x L(x, Z)$$

Как соотносятся величины p^* и d^* ? Равны ли они? В общем случае ответ отрицательный, т. е. возможно $p^* \neq d^*$. Мы покажем, что всегда справедлива [слабая двойственность](#): $d^* \leq p^*$. [Сильная двойственность](#) $d^* = p^*$ требует дополнительных предположений (условий регулярности).

Большинство современных методов SDP-оптимизации требуют одновременного решения прямой и двойственной задачи и выполнения свойства сильной двойственности.

Лекция 13. Двойственность в задачах SDP: часть 2

Слабая и сильная двойственность

Мы рассмотрели две задачи оптимизации — прямую и двойственную:

$$p^* = \inf_x \sup_{Z \geq 0} L(x, Z) \quad d^* = \sup_{Z \geq 0} \inf_x L(x, Z)$$

Как соотносятся величины p^* и d^* ? Равны ли они? В общем случае ответ отрицательный. Мы покажем, что всегда справедлива **слабая двойственность**: $d^* \leq p^*$. **Сильная двойственность** $d^* = p^*$ требует дополнительных предположений (условий регулярности).

Большинство современных методов SDP-оптимизации требуют одновременного решения прямой и двойственной задачи и выполнения свойства сильной двойственности.

Слабая двойственность

Теорема 1 (о слабой двойственности).

$$d^* = \sup_{Z \geq 0} \inf_x L(x, Z) \leq p^* = \inf_x \sup_{Z \geq 0} L(x, Z)$$

Величина $p^* - d^*$ называется разрывом двойственности (duality gap).

Доказательство. (Не использует структуру $L(x, Z)$!)

$$\begin{aligned} \inf_x L(x, Z) &\leq L(x, Z), \quad \forall x, \quad \forall Z \quad \Rightarrow \\ \sup_{Z \geq 0} \inf_x L(x, Z) &\leq \sup_{Z \geq 0} L(x, Z), \quad \forall x \quad \Rightarrow \\ \sup_{Z \geq 0} \inf_x L(x, Z) &\leq \inf_x \sup_{Z \geq 0} L(x, Z) \end{aligned}$$

Сильная двойственность

Свойство сильной двойственности: $d^* = p^*$.

Теорема 2 (о сильной двойственности). Пусть выполнено условие Слейтера: $\exists x^0 : F(x^0) > 0$. Тогда $d^* = p^*$.

Замечание: возможно $d^* = p^* = -\infty$.

Доказательство основано на построении гиперплоскости, разделяющей выпуклые множества в пространстве $\mathbb{S}^n \times \mathbb{R}$.

Доказательство теоремы о сильной двойственности-1

Рассмотрим линейное пространство:

$$\mathbb{S}^n \times \mathbb{R} = \{(U, t) : U \in \mathbb{S}^n, t \in \mathbb{R}\}.$$

Превратим его в евклидово пространство, введя скалярное произведение:

$$\langle (U, s), (V, t) \rangle = \operatorname{tr} UV + st.$$

Гиперплоскость с направляющим вектором $(-D, \mu)$:

$$\langle (-D, \mu), (U, t) \rangle = \alpha, \quad D \in \mathbb{S}^n, \quad \mu, \alpha \in \mathbb{R},$$

По-другому — уравнение гиперплоскости:

$$-\operatorname{tr} DU + \mu t = \alpha, \quad \forall U \in \mathbb{S}^n, \quad \forall t \in \mathbb{R}$$

Доказательство теоремы о сильной двойственности-2

В силу слабой двойственности $d^* \leq p^*$, поэтому, если $p^* = -\infty$, то $d^* = -\infty$.

Пусть p^* конечно. Определим множества в $\mathbb{S}^n \times \mathbb{R}$:

$$\begin{aligned} \mathcal{A} &= \{(U, t) : \exists x : F(x) \geq U, c^T x \leq t\}, \\ \mathcal{B} &= \{(\mathbb{O}_{n \times n}, t) : t < p^*\} \end{aligned}$$

Докажем, что

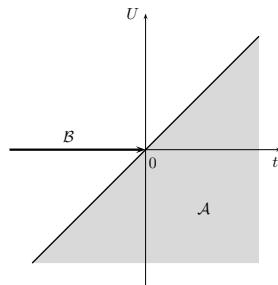
- Множества \mathcal{A}, \mathcal{B} выпуклы в $\mathbb{S}^n \times \mathbb{R}$.
- $\mathcal{A} \cap \mathcal{B} = \emptyset$.

Пример \mathcal{A}, \mathcal{B}

Рассмотрим одномерный случай: x, U — скаляры, $F(x) = x$, $c^T x = x$. Тогда

$$p^* = \inf_{x \geq 0} x = 0,$$

$$\begin{aligned} \mathcal{A} &= \{(U, t) : \exists x : x \geq U, x \leq t\} = \{(U, t) : U \leq t\}, \\ \mathcal{B} &= \{(U, t) : U = 0, t < 0\}. \end{aligned}$$



Доказательство выпуклости \mathcal{A}, \mathcal{B}

$$\begin{aligned}\mathcal{A} &= \{(U, t) : \exists x : F(x) \geq U, c^T x \leq t\}, \\ \mathcal{B} &= \{(\mathbb{O}_{n \times n}, t) : t < p^*\}\end{aligned}$$

Пусть $(U_1, t_1) \in \mathcal{A}, (U_2, t_2) \in \mathcal{A}$.

Покажем, что при $U = \alpha U_1 + (1 - \alpha)U_2, t = \alpha t_1 + (1 - \alpha)t_2, 0 \leq \alpha \leq 1$, выполнено $(U, t) \in \mathcal{A}$.

Для некоторых x_1, x_2 выполнено $F(x_1) \geq U_1, c^T x_1 \leq t_1$ и $F(x_2) \geq U_2, c^T x_2 \leq t_2$. Так как

$$\begin{aligned}F(\alpha x_1 + (1 - \alpha)x_2) &= \alpha F(x_1) + (1 - \alpha)F(x_2), \\ c^T(\alpha x_1 + (1 - \alpha)x_2) &= \alpha c^T x_1 + (1 - \alpha)c^T x_2,\end{aligned}$$

то при $x = \alpha x_1 + (1 - \alpha)x_2, 0 \leq \alpha \leq 1$ выполнено

$$F(x) \geq U, c^T x \leq t, \text{ т. е. } (U, t) \in \mathcal{A}.$$

Выпуклость \mathcal{B} доказывается аналогично.

Доказательство $\mathcal{A} \cap \mathcal{B} = \emptyset$

$$\begin{aligned}\mathcal{A} &= \{(U, t) : \exists x : F(x) \geq U, c^T x \leq t\}, \\ \mathcal{B} &= \{(\mathbb{O}_{n \times n}, t) : t < p^*\}\end{aligned}$$

Пусть $(U, t) \in \mathcal{A} \cap \mathcal{B}$. Тогда

$$U = 0, \exists x^0 : F(x^0) \geq 0, c^T x^0 \leq t < p^*.$$

Получаем

$$p^* = \inf_{F(x) \geq 0} c^T x \leq c^T x^0 < p^*.$$

Получено противоречие $p^* < p^*$, которое доказывает пустоту пересечения.

Доказательство теоремы о сильной двойственности-3

Множества \mathcal{A}, \mathcal{B} выпуклы и не пересекаются.

Значит для них существует разделяющая гиперплоскость с параметрами D, μ, α вида $-\operatorname{tr} UD + \mu t = \alpha$. Тогда

$$(U, t) \in \mathcal{A} \implies -\operatorname{tr} UD + \mu t \geq \alpha, \quad (1)$$

$$(U, t) \in \mathcal{B} \implies -\operatorname{tr} UD + \mu t \leq \alpha \quad (2)$$

Эти утверждения можно переписать в виде:

$$(1): \exists x : F(x) \geq U, c^T x \leq t \implies -\operatorname{tr} UD + \mu t \geq \alpha$$

$$(2): t < p^* \implies \mu t \leq \alpha$$

Докажем утверждение: $D \geq 0$ и $\mu > 0$.

Доказательство того, что $D \geq 0$

$$(1): \exists x : F(x) \geq U, c^T x \leq t \implies -\text{tr } UD + \mu t \geq \alpha$$

Докажем от противного: $D \not\geq 0 \iff \exists y : y^T Dy < 0$.

По условию Слейтера, $\exists x^0 : F(x^0) > 0$.

Возьмем $x = x^0$, $t = c^T x^0$, $U = -\gamma y y^T$, где $\gamma > 0$.

Тогда $U \leq 0$ и $F(x^0) > 0 \geq U$. Отсюда

$$\text{tr } UD = -\gamma \text{tr } D y y^T = -\gamma y^T D y.$$

Из (1) получаем

$$\gamma y^T D y + \mu t \geq \alpha.$$

Така как $y^T D y < 0$, то при $\gamma \rightarrow +\infty$ приходим к противоречию.

Доказательство того, что $\mu \geq 0$

$$(1): \exists x : F(x) \geq U, c^T x \leq t \implies -\text{tr } UD + \mu t \geq \alpha$$

По условию Слейтера. существует $F(x^0) > 0$.

Возьмем $x = x^0$, $U = 0$. Из (1) получаем:

$$t \geq c^T x^0 \implies \mu t \geq \alpha.$$

Если допустить, что $\mu < 0$, то при $t \rightarrow +\infty$ приходим к противоречию.

Преобразование условий (1), (2)

$$(1): \exists x : F(x) \geq U, c^T x \leq t \implies -\text{tr } UD + \mu t \geq \alpha$$

$$(2): t < p^* \implies \mu t \leq \alpha$$

Так как $\mu \geq 0$, то

$$\sup_{t < p^*} \mu t = \mu p^*$$

и условие (2) эквивалентно неравенству $\mu p^* \leq \alpha$.

Тогда из (1) и $\mu p^* \leq \alpha$ вытекает:

$$(3): \exists x : F(x) \geq U, c^T x \leq t \implies -\text{tr } UD + \mu t \geq \mu p^*.$$

Доказательство того, что $\mu \neq 0$

$$(3): \exists x : F(x) \geq U, c^T x \leq t \implies -\text{tr } UD + \mu t \geq \mu p^*.$$

Допустим, что $\mu = 0$. Тогда при $U = F(x)$ получим:

$$(3a): \text{tr } DF(x) \leq 0, \quad \forall x$$

По условию Слейтера, $\exists x^0 : F(x^0) > 0$.

Так как $D \geq 0$, $F(x^0) > 0$, то из леммы 2 о свойствах \mathbb{S}_+^n следует $\text{tr } DF(x^0) \geq 0$.

При $x = x^0$ из (3a) получаем противоположное неравенство: $\text{tr } DF(x^0) \leq 0$.

В результате: $\text{tr } DF(x^0) = 0$.

Так как $F(x^0) > 0$, то по лемме 2 о свойствах \mathbb{S}_+^n получаем $D = 0$.

Получено противоречие: $D = 0$, $\mu = 0$ — нет гиперплоскости.

Доказательство теоремы о сильной двойственности

$$(3): \exists x : F(x) \geq U, c^T x \leq t \implies -\text{tr } UD + \mu t \geq \mu p^*.$$

Возьмем $U = F(x)$, $t = c^T x$, тогда получим

$$\mu c^T x - \text{tr } DF(x) \geq \mu p^*, \quad \forall x$$

Мы показали, что $D \geq 0$, $\mu > 0$. Положим $Z^0 = D/\mu$ и получим:

$$c^T x - \text{tr } F(x)Z_0 \geq p^*, \quad \forall x.$$

Перепишем:

$$L(x, Z_0) \geq p^*, \quad \forall x, \implies \inf_x L(x, Z_0) \geq p^*.$$

Тогда по теореме о максимине

$$d^* = \sup_{Z \geq 0} \inf_x L(x, Z) \geq \inf_x L(x, Z_0) \geq p^*.$$

Значит $d^* \geq p^*$, откуда из слабой двойственности $d^* = p^*$.

Строгие и нестрогие неравенства

Рассмотрим задачу SDP:

$$c^T x \rightarrow \inf, \quad F(x) \geq 0.$$

Возможны два случая:

- $\exists x^0 : F(x^0) > 0$. Тогда

$$\{x : F(x) \geq 0\} = \overline{\{x : F(x) > 0\}}$$

и

$$\inf_{F(x) \geq 0} c^T x = \inf_{F(x) > 0} c^T x.$$

- Внутренность множества $\{x : F(x) \geq 0\}$ пуста, т. е. условие Слейтера не выполнено.

Строгие и нестрогие неравенства (продолжение)

Пусть

$$F(x) = F_0 + x_1 F_1 + \dots + x_m F_m, \quad F_i \in \mathbb{S}^n.$$

Утверждение (Boyd, с.19)

Пусть неравенство $F(x) \geq 0$ разрешимо, а неравенство $F(x) > 0$ — нет.

Тогда при некоторых $p \leq m$, $q \leq n$ существуют матрицы $A \in \mathbb{R}^{m \times p}$, $b \in \mathbb{R}^m$, $\tilde{F}_i \in \mathbb{S}^q$, $i = 1, \dots, p$, такие, что

$$F(x) \geq 0 \iff \begin{cases} \exists z \in \mathbb{R}^p : x = Az + b, \\ \tilde{F}(z) = \tilde{F}_0 + z_1 \tilde{F}_1 + \dots + z_p \tilde{F}_p \geq 0. \end{cases}$$

При этом $\exists z^0 : \tilde{F}(z^0) > 0$.

$$\text{Тогда } \inf_{F(x) \geq 0} c^T x = \inf_{\tilde{F}(z) > 0} c^T (Az + b) = c^T b + \inf_{\tilde{F}(z) > 0} c^T A z.$$

Регулярность двойственной задачи

Двойственная задача:

$$-\operatorname{tr} ZF_0 \rightarrow \sup, \quad Z \geq 0, \quad \operatorname{tr} ZF_i = c_i, \quad i = 1, \dots, m$$

Условие Слейтера (регулярности):

$$\exists Z^0 \in \mathbb{S}^n : Z^0 > 0, \quad \operatorname{tr} Z^0 F_i = c_i, \quad i = 1, \dots, m$$

Теорема 3. Пусть выполнено условие Слейтера для двойственной задачи. Тогда $d^* = p^*$.

Доказательство основано на работе с функцией Лагранжа

$$\tilde{L}(x, Z, \Lambda) = -\operatorname{tr} ZF_0 + \sum_{i=1}^m x_i(c_i - \operatorname{tr} ZF_i) + \operatorname{tr} Z\Lambda = L(x, Z) + \operatorname{tr} Z\Lambda.$$

Доказательство утверждения о максимине

В доказательстве теоремы о максимине было показано:

$$\inf_x L(x, Z) = \begin{cases} -\operatorname{tr} ZF_0 & \text{при } Z \in \Delta, \\ -\infty, & \text{при } Z \notin \Delta, \end{cases}$$

где $\Delta = \{Z : \operatorname{tr} F_i Z = c_i, i = 1, \dots, m\}$. Рассмотрим функцию Лагранжа $\tilde{L}(x, Z, \Lambda) = L(x, Z) + \operatorname{tr} Z\Lambda$.

Легко проверить:

$$\inf_{\Lambda \geq 0} \operatorname{tr} Z\Lambda = \begin{cases} 0, & Z \geq 0, \\ -\infty, & Z \not\geq 0, \end{cases}$$

откуда

$$\inf_{\Lambda \geq 0, x} \tilde{L}(x, Z, \Lambda) = \begin{cases} -\operatorname{tr} ZF_0, & Z \geq 0, Z \in \Delta, \\ -\infty, & \text{при остальных } Z. \end{cases}$$

Тогда по теореме о максимине

$$\sup_Z \inf_{\Lambda \geq 0, x} \tilde{L}(x, Z, \Lambda) = \sup_{Z \geq 0, Z \in \Delta} (-\operatorname{tr} F_0 Z) = d^*.$$

Доказательство утверждения о минимаксе

Определим

$$\begin{aligned} g(x, \Lambda) &= \sup_Z \tilde{L}(x, Z, \Lambda) = \sup_Z (c^T x - \operatorname{tr} ZF(x) + \operatorname{tr} Z\Lambda) = \\ &= \sup_Z (c^T x - \operatorname{tr} Z(F(x) - \Lambda)). \end{aligned}$$

Тогда

$$g(x, \Lambda) = \begin{cases} c^T x, & F(x) - \Lambda = 0, \\ +\infty, & \text{в остальных случаях} \end{cases}$$

По определению p^* :

$$\inf_{\Lambda \geq 0, x} g(x, \Lambda) = \inf_{F(x) = \Lambda, \Lambda \geq 0} c^T x = \inf_{F(x) \geq 0} c^T x = p^*.$$

Значит

$$\inf_{\Lambda \geq 0, x} \sup_Z \tilde{L}(x, Z, \Lambda) = p^*.$$

Доказательство: вспомогательные множества

Пусть d^* конечно. Определим два множества в $\mathbb{S}^n \times \mathbb{R}^m \times \mathbb{R}$:

$$\begin{aligned}\mathcal{A} &= \{(U, v, t) : \\ &\quad \exists Z \in \mathbb{S}^n : Z \geq U, \forall i, c_i - \text{tr } ZF_i = v_i, -\text{tr } F_0Z \geq t\}, \\ \mathcal{B} &= \{(\mathbb{O}_{n \times n}, \mathbb{O}_m, t) : t > d^*\}\end{aligned}$$

Очевидно, \mathcal{A}, \mathcal{B} выпуклы. Покажем, что и $\mathcal{A} \cap \mathcal{B} = \emptyset$. Пусть $(U, v, t) \in \mathcal{A} \cap \mathcal{B}$. Тогда $U = 0, v = 0$ и

$$t > d^*, \quad Z \geq 0, \quad Z \in \Delta, \quad -\text{tr } F_0Z \geq t.$$

Отсюда

$$Z \geq 0, \quad Z \in \Delta, \quad -\text{tr } F_0Z > d^*,$$

что противоречит теореме о максимине:

$$d^* = \sup_{Z \geq 0, Z \in \Delta} (-\text{tr } F_0Z).$$

Доказательство: разделяющая гиперплоскость

Существует разделяющая гиперплоскость:

$$\begin{aligned}(U, v, t) \in \mathcal{A} &\implies -\text{tr } UD + h^T v - \mu t \geq \alpha, \\ (U, v, t) \in \mathcal{B} &\implies -\text{tr } UD + h^T v - \mu t \leq \alpha,\end{aligned}$$

где $D \in \mathbb{S}^n, v \in \mathbb{R}^n, \mu \in \mathbb{R}, \alpha \in \mathbb{R}$.

Получаем: (1) $\exists Z : Z \geq U, c_i - \text{tr } ZF_i = v_i, \forall i, -\text{tr } F_0Z \geq t \implies -\text{tr } UD + h^T v - \mu t \geq \alpha$, (2) $t > d^* \implies -\mu t \leq \alpha$.

Как и в случае прямой задачи, покажем что $D \geq 0, \mu > 0$.

Доказательство: $\mu \geq 0$

Имеем: (1) $\exists Z : Z \geq U, c_i - \text{tr } ZF_i = v_i, \forall i, -\text{tr } F_0Z \geq t \implies -\text{tr } UD + h^T v - \mu t \geq \alpha$,

При фиксированных U, v первое неравенство может выполняться при $t \rightarrow -\infty$.

Тогда второе неравенство справедливо только при $\mu \geq 0$.

Так как $\mu \geq 0$, то условие (2) $t > d^* \implies -\mu t \leq \alpha$ эквивалентно условию $-\mu d^* \leq \alpha$.

Тогда из (1) следует:

$$(3) \exists Z : Z \geq U, c_i - \text{tr } ZF_i = v_i, \forall i, -\text{tr } F_0Z \geq t \implies -\text{tr } UD + h^T v - \mu t \geq -\mu d^*$$

Доказательство: $D \geq 0$

Имеем: (1) $\exists Z : Z \geq U, c_i - \text{tr} ZF_i = v_i, \forall i, -\text{tr} F_0 Z \geq t \implies -\text{tr} UD + h^T v - \mu t \geq \alpha$,

Условие регулярности:

$$\exists Z^0 : Z^0 > 0, \text{tr} Z^0 F_i = c_i, \forall i$$

Возьмем в (1) $Z = Z^0 > 0, v = 0, t = t^0 = -\text{tr} F_0 Z^0$. Тогда

$$U \leq 0 \implies -\text{tr} UD \geq \alpha + \mu t^0.$$

Обозначим $M = -U$. Как в доказательстве предыдущей теоремы,

$$\text{tr} MD \geq 0, \forall M \geq 0.$$

По лемме 3 о свойствах \mathbb{S}_+^n , получаем $D \geq 0$.

Доказательство: $\mu \neq 0$

(3) $\exists Z : Z \geq U, c_i - \text{tr} ZF_i = v_i, \forall i, -\text{tr} F_0 Z \geq t \implies -\text{tr} UD + h^T v - \mu t \geq -\mu d^*$.

Пусть $\mu = 0$. Тогда (3) примет вид:

$$(3) \exists Z : Z \geq U, c_i - \text{tr} ZF_i = v_i, \forall i \implies -\text{tr} UD + h^T v \geq 0.$$

Пусть $Z^0 > 0$ — из условия регулярности, $\text{tr} Z^0 F_i = c_i, \forall i$.

Возьмем $U = Z^0, v = 0$. Тогда $-\text{tr} Z^0 D \geq \alpha \geq 0$. откуда $\text{tr} Z^0 D \leq 0$. Так как $Z^0 > 0, D \geq 0$, то $\text{tr} Z^0 D \geq 0$.

Тогда $\text{tr} Z^0 D = 0$ и, по лемме 3, $D = 0$.

Получаем $D = 0, \mu = 0$ — нет гиперплоскости.

Доказательство сильной двойственности

(3) $\exists Z : Z \geq U, c_i - \text{tr} ZF_i = v_i, \forall i, -\text{tr} F_0 Z \geq t \implies -\text{tr} UD + h^T v - \mu t \geq -\mu d^*$.

Возьмем $U = Z, t = -\text{tr} F_0 Z$.

$$-\text{tr} ZD + \sum_{i=1}^m h_i(c_i - \text{tr} ZF_i) + \mu \text{tr} F_0 Z \geq -\mu d^*, \quad \forall Z \in \mathbb{S}^n.$$

Т.к. $\mu > 0$, положим $\Lambda^0 = D/\mu, x_i^0 = h_i/\mu$. Тогда

$$\begin{aligned} & -\text{tr} Z\Lambda^0 + \sum_{i=1}^m x_i^0(c_i - \text{tr} ZF_i) + \text{tr} ZF_0 \geq -d^*, \quad \forall Z \implies \\ & \tilde{L}(x^0, Z, \Lambda^0) \leq d^*, \quad \forall Z \implies \\ & \sup_Z \tilde{L}(x^0, Z, \Lambda^0) \leq d^* \implies \inf_{\Lambda \geq 0, x} \sup_Z \tilde{L}(x, Z, \Lambda) \leq d^* \implies \\ & p^* \leq d^* \implies p^* = d^* \end{aligned}$$

Одновременное решение прямой и двойственной задач

Прямая задача:

$$c^T x \rightarrow \min, \quad x \in \Omega_1 = \{x \in \mathbb{R}^m : F(x) \geq 0\}$$

Двойственная задача:

$$\begin{aligned} -\operatorname{tr} F_0 Z &\rightarrow \max, \\ Z \in \Omega_2 = \{Z \in \mathbb{S}^n : Z \geq 0, \operatorname{tr} F_i Z = c_i, i = 1, \dots, m\} \end{aligned}$$

Обозначим

$$p(x) = c^T x, \quad x \in \Omega_1, \quad d(Z) = -\operatorname{tr} F_0 Z, \quad Z \in \Omega_2$$

Величина

$$\eta(x, Z) = p(x) - d(Z) = c^T x + \operatorname{tr} F_0 Z$$

называется duality gap (разрыв двойственности).

Разрыв двойственности

Утверждение

$\eta(x, Z) = p(x) - d(Z) \geq 0$ при всех допустимых $x \in \Omega_1, Z \in \Omega_2$.

Если имеет место сильная двойственность и обе задачи имеют конечные решения x^*, Z^* , то $\eta(x^*, Z^*) = 0$.

Доказательство.

$$\begin{aligned} \forall x \in \Omega_1, \quad p(x) &= c^T x \geq p^* = \inf_{x \in \Omega_1} c^T x, \\ \forall Z \in \Omega_2, \quad d(Z) &= -\operatorname{tr} F_0 Z \leq d^* = \sup_{Z \in \Omega_2} (-\operatorname{tr} F_0 Z), \\ d^* &\leq p^* — слабая двойственность \end{aligned}$$

Отсюда $d(Z) \leq d^* \leq p^* \leq p(x)$, $\forall x \in \Omega_1, Z \in \Omega_2$. Если $p(x^*) = p^*$, $d(Z^*) = d^*$, $p^* = d^*$, то $\eta(x^*, Z^*) = 0$.

Разрывы двойственности (продолжение)

Утверждение

$\eta(x, Z) = p(x) - d(Z) = \operatorname{tr} F(x) Z$ при всех допустимых $x \in \Omega_1, Z \in \Omega_2$.

Доказательство.

Пусть $x \in \Omega_1, Z \in \Omega_2$.

Имеем: $F(x) \geq 0$, $Z \geq 0$, $\operatorname{tr} F_i Z = c_i$, $\forall i$. Тогда

$$\begin{aligned} \eta(x, Z) &= p(x) - d(Z) = c^T x + \operatorname{tr} F_0 Z = \sum_{i=1}^m c_i x_i + \operatorname{tr} F_0 Z = \\ &\{ \text{т.к. } \operatorname{tr} F_i Z = c_i \} = \sum_{i=1}^m x_i \operatorname{tr} F_i Z + \operatorname{tr} F_0 Z = \\ &= \operatorname{tr} \left[\left(F_0 + \sum_{i=1}^m x_i F_i \right) Z \right] = \operatorname{tr} F(x) Z. \end{aligned}$$

Условия дополняющей нежесткости

Условия дополняющей нежесткости (complementary slackness):

Пусть имеет место сильная двойственность и $p^* = d^*$ — конечно. Тогда для решений x^*, Z^* прямой и двойственной задач

$$F(x^*)Z^* = 0$$

Доказательство.

Было показано: $\eta(x, Z) = p(x) - d(Z) = \text{tr } F(x)Z$ при всех допустимых x, Z .

Получаем: если $p(x^*) = d(Z^*)$, то $\text{tr } F(x^*)Z^* = 0$, где $F(x^*) \geq 0, Z^* \geq 0$. По лемме 2 о свойствах \mathbb{S}_+^n получаем $F(x^*)Z^* = 0$.

Условия ККТ

Верно и обратное утверждение (без доказательства):

Теорема Каруша–Куна–Такера (Karush–Kuhn–Tucker):

Пусть $p^* = d^*$ — конечно. Тогда для того, чтобы x^*, Z^* были решениями прямой и двойственной задачи, необходимо и достаточно, чтобы

$$F(x^*) \geq 0, \quad Z^* \geq 0, \quad \text{tr } F_i Z^* = c_i, \quad \forall i \quad F(x^*)Z^* = 0$$

Смешанная задача

Рассмотрим смешанную задачу (primal-dual problem):

$$\begin{aligned} \eta(x, Z) &= c^T x + \text{tr}(F_0 Z) \rightarrow \min, \\ F(x) &\geq 0, \quad Z \geq 0, \\ \text{tr}(F_i Z) &= c_i, \quad i = 1, \dots, m. \end{aligned}$$

Тогда $\eta(x, Z) \geq 0$ и, если $\eta(x^*, Z^*) = 0$, по теореме ККТ получаем, что x^* — решение прямой задачи, Z^* — решение двойственной задачи.

На каждом шаге (из определения d^*, p^* и слабой двойственности):

$$d(Z) \leq d^* \leq p^* \leq p(x),$$

т.е. $p^* = d^* \in [d(Z), p(x)]$. Алгоритм строится так, что длина интервала $[d(Z), p(x)]$ стремится к нулю.

Лекция 14. Численные методы: метод Ньютона и метод барьерных функций

Введение

В этой лекции мы еще не будем рассматривать методы решения задач SDP. Рассмотрим их простейшие аналоги — работу со скалярными функциями одной или нескольких переменных. Различают два класса задач:

- Оптимизация без ограничений (безусловная оптимизация). Для этого класса задач вспомним метод Ньютона.
- Оптимизация с ограничениями (условная оптимизация). Для этого класса задач рассмотрим метод барьерных функций (метод внутренних штрафов).

Скалярная оптимизация без ограничений

Рассмотрим (для определенности) скалярную задачу минимизации без ограничений:

$$f(x) \rightarrow \min, \quad f : \mathbb{R} \mapsto \mathbb{R}$$

Если функция $f(x) \in C^2$ (имеет две непрерывные производные), то условие локального минимума:

$$f'(x^0) = 0, \quad f''(x^0) \geq 0.$$

Если $f''(x^0) > 0$ — строгий локальный минимум.

Многомерная оптимизация без ограничений

В многомерном случае $f : \mathbb{R} \mapsto \mathbb{R}^m$. Пусть $f(x) \in C^2$, $\nabla f(x)$ — градиент, $H_f(x)$ — матрица Гессе (гессиан):

$$\nabla f(x) = \left[\frac{\partial f}{\partial x_i}(x) \right]_{m \times 1}, \quad \nabla^2 f(x) = H_f(x) = \left[\frac{\partial^2 f}{\partial x_i \partial x_j}(x) \right]_{m \times m}$$

Матрица Гессе симметрична: $H_f^\top(x) = H_f(x)$, $\forall x$. Если $H_f(x) \geq 0$, $\forall x$, то функция $f(x)$ выпукла.

Условие локального минимума:

$$\nabla F(x^0) = 0, \quad H_f(x^0) \geq 0.$$

Если $H_f(x^0) > 0$ — строгий локальный минимум. Все неравенства понимаются в смысле положительной (неотрицательной) определенности.

Методы решения оптимизационных задач

Для гладких функций $f(x)$ существуют три класса методов оптимизации:

- Методы 0-го порядка (поисковые методы): используют только $f(x)$, не используют производные;
- Методы 1-го порядка (градиентные методы): используют $f(x)$ и $\nabla f(x)$;
- Методы 2-го порядка (ニュтоновские методы): используют $f(x)$, $\nabla f(x)$ и $H_f(x)$.

В задачах SDP используют методы 2-го порядка.

Решение нелинейного уравнения. Метод Ньютона

Рассмотрим скалярное нелинейное уравнение:

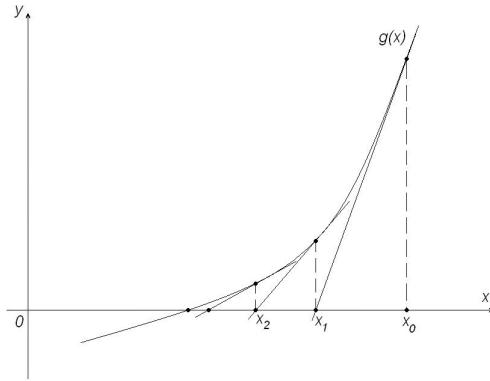
$$g(x) = 0,$$

где $g(x)$ — дифференцируемая функция. Для решения известен метод Ньютона (метод касательных). Корень вычисляется по итерационной формуле:

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)}.$$

При этом предполагается, что $g'(x) \neq 0$. Вычисления останавливаются либо когда $|g(x_k)| < \varepsilon_1$, либо когда $|x_{k+1} - x_k| < \varepsilon_2$.

Скалярный метод Ньютона: иллюстрация



Пример метода Ньютона

Рассмотрим задачу нахождения корня функции

$$g(x) = \frac{x}{\sqrt{1+x^2}}$$

Тогда $g'(x) = 1/(1+x^2)^{3/2}$ и

$$x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)} = -x_n^3.$$

Очевидно, если $|x_0| < 1$, то $x_n \rightarrow 0$ при $n \rightarrow +\infty$, причем очень быстро.

Если $|x_0| > 1$, то $|x_n| \rightarrow +\infty$ при $n \rightarrow +\infty$.

Скалярная задача оптимизации: метод Ньютона

Рассмотрим оптимизационную задачу

$$f(x) \rightarrow \min.$$

Применим метод Ньютона для решения уравнения экстремума

$$f'(x) = 0,$$

то-есть возьмем $g(x) = f'(x)$. При этом предполагаем, что $f''(x) \geq 0$, то-есть функция выпуклая.

Будем вычислять приближения по формуле

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}.$$

Вычисления прекращаются, когда либо $|f'(x_{k+1})| < \varepsilon_1$, либо $|x_{k+1} - x_k| < \varepsilon_2$, где $\varepsilon_1, \varepsilon_2$ — заданные малые числа (точность вычисления экстремума и невязки).

Особенности метода Ньютона

Метод имеет следующие преимущества и недостатки:

- Преимущество: высокая скорость сходимости.
- Недостатки: 1) требуется соблюдения условия $f''(x_k) \neq 0$; 2) жесткие условия сходимости — обычно начальное значение x_0 должно лежать в достаточно узкой окрестности начальной точки.

Многомерный метод Ньютона

Многомерный метод Ньютона:

$$x_{k+1} = x_k - H_f^{-1}(x_k) \nabla f(x_k).$$

Пусть x^0 — точка минимума. Существует окрестность x^0 такая, что, если начальная точка x_1 принадлежит этой окрестности, то $x_k \xrightarrow[k \rightarrow \infty]{} x^0$ и

$$\|x_{k+1} - x^0\| \leq C \|x_k - x^0\|^2,$$

где C — константа (скорость сходимости квадратична).

Сходимость метода Ньютона существенно зависит от выбора начальной точки.

Многомерный метод Ньютона (переменный шаг)

Метод Ньютона с регулировкой шага:

$$x_{k+1} = x_k - \mu_k H_f^{-1}(x_k) \nabla f(x_k),$$

μ_k — убывающая скалярная последовательность.

Сходимость методов минимизации определяется числом обусловленности (condition number) матрицы Гессе:

$$\lambda_c = \frac{\min_j |\lambda_j(H_f(x^0))|}{\max_j |\lambda_j(H_f(x^0))|}$$

$\lambda_c \ll 1$ — плохая обусловленность, $\lambda_c \approx 1$ — хорошая

Скалярная оптимизация с ограничениями. Метод штрафных функций

Другие названия: метод внутренних штрафов, метод барьерных функций (А. Фиакко и Г. П. Мак-Кормик, 1960-е)

Задача оптимизации с ограничениями

$$f(x) \rightarrow \min, \quad g_i(x) \geq 0, \quad i = 1, \dots, n$$

Здесь $x \in \mathbb{R}^m$, $f(x)$, $g_i(x)$ — выпуклые скалярные функции.

Условие Слейтера (Slater's condition):

$$\exists x_0 \in \mathbb{R}^m : g_i(x_0) > 0, \quad i = 1, \dots, n$$

Основная идея — попытаться заметить задачу оптимизации с ограничениями на задачу без ограничений.

Идея барьера

Множество строго допустимых точек

$$\Omega = \{ x \in \mathbb{R}^m : g_i(x) > 0, \quad i = 1, \dots, n \} \neq \emptyset$$

Формально рассмотрим индикаторную функцию

$$B_0(x) = \begin{cases} C_0, & x \in \Omega, \\ +\infty, & x \notin \Omega \end{cases}, \quad C_0 = \text{const}$$

Заменим исходную задачу на задачу минимизации без ограничений:

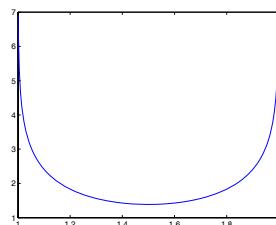
$$f(x) + \mu B_0(x) \rightarrow \min, \quad \mu > 0 — \text{параметр}$$

Идея — аппроксимировать индикаторную функцию гладкими функциями.

Барьерные функции

Функция $B(x)$ наз. *барьерной* для Ω , если она конечна при $x \in \Omega$ и

$$x_k \in \Omega, \quad x_k \rightarrow \partial\Omega \text{ при } k \rightarrow \infty \implies B(x_k) \rightarrow +\infty \text{ при } k \rightarrow \infty$$



Примеры барьеров

- Логарифмическая барьерная функция (Frisch barrier function):

$$B(x) = - \sum_{i=1}^n \ln(g_i(x)) = \sum_{i=1}^n \ln(g_i^{-1}(x))$$

Так как $(\ln x^{-1})'' = 1/x^2$, то функция $\ln x^{-1}$ выпукла. Тогда $B(x)$ гладкая и выпуклая при гладких и выпуклых $g_i(x)$.

- Степенная барьерная функция

$$B(x) = \sum_{i=1}^n g_i^{-p}(x), \quad p \geq 1.$$

- Экспоненциальная барьерная функция

$$B(x) = \sum_{i=1}^n \exp(g_i^{-1}(x)).$$

Метод барьерных функций

Заменим исходную задачу на выпуклую задачу без ограничений:

$$f(x) + \mu B(x) \rightarrow \min, \quad \mu > 0$$

μ — барьерный параметр

Решим ее (градиентный спуск, метод Ньютона): $x = x(\mu)$.

Пусть существует x^0 — решение исходной задачи.

При слабых предположениях

$$x(\mu) \rightarrow x^0 \quad \text{при } \mu \rightarrow +0$$

Для логарифм. барьера функцию $x(\mu)$ наз. центральным путем, а ее значения — центрами.

Метод барьерных функций (пример)

Задача:

$$x \rightarrow \min, \quad x \geq 1$$

Вспомогательная задача:

$$\varphi(x) = x - \mu \ln(x - 1) \rightarrow \min$$

Решение: Имеем

$$\varphi' = 1 - \frac{\mu}{x-1}, \quad \varphi'' = \frac{\mu}{(x-1)^2} > 0$$

$$\varphi' = 0 \implies x(\mu) = 1 + \mu \rightarrow 1 \quad \text{при } \mu \rightarrow +0$$

Точное решение: $x^0 = 1$ (принадлежит границе допустимого множества!)

Пример: продолжение

Теперь для минимизации $\varphi(x) = x - \mu \ln(x - 1)$ используем метод Ньютона:

$$x_{k+1} = x_k - \varphi'(x_k)/\varphi''(x_k)$$

Так как

$$\varphi' = 1 - \frac{\mu}{x-1}, \quad \varphi'' = \frac{\mu}{(x-1)^2} > 0,$$

то

$$\frac{\varphi'}{\varphi''} = \frac{1}{\mu}(x-1-\mu)(x-1).$$

Тогда

$$x_{k+1} = x_k - \frac{1}{\mu}(x_k - 1 - \mu)(x_k - 1)$$

Запишем уравнение в отклонениях: $y_k = x_k - 1 - \mu$:

$$y_{k+1} = -\frac{1}{\mu} y_k^2.$$

Отображение сжимающее, если

$$|y_k/\mu| < 1 \iff |y_k| < \mu.$$

Чем меньше μ , тем меньше область сходимости.

Метод барьерных функций: дробление шага

Если $x^0 \in \partial\Omega$, μ мало, матрица Гессе $H_\varphi(x)$ плохо обусловлена.

Алгоритм

1. Выбираем $x_1 \in \Omega$, μ_1
2. При заданных x_k , μ_k решаем задачу

$$f(x) + \mu_k B(x) \rightarrow \min, \quad x_k — начальная точка$$

Полагаем x_{k+1} — решение.

3. Если $\mu_k |B(x_{k+1})| < \varepsilon$, то останавливаемся. Иначе $\mu_{k+1} = \mu_k/2$ и к п. 2.

Метод барьерных функций (условия сходимости)

Теорема о сходимости (без доказательства). Пусть f , g_i непрерывны, выполнено условие Слейтера. Пусть исходная задача имеет решение x^0 , причем x^0 — предельная точка множества Ω . Тогда

$$f(x_k) + \mu_{k-1} B(x_k) \rightarrow f(x^0) \quad \text{при } k \rightarrow \infty$$

и предел любой сходящейся подпоследовательности $\{x_k\}$ есть x^0 .

Метод барьерных функций (недостатки и преимущества)

Недостатки:

1. Метод неприменим для ограничений типа равенств (обычно тогда $\Omega = \emptyset$).
2. Для нахождения начальной точки нужно решать еще одну задачу:

$$t \rightarrow \min, \quad g_i(x) + t \geq 0, \quad i = 1, \dots, n$$

3. Вблизи границы Ω есть риск «выскочить» за границу, если использовать дискретный шаг.

Преимущество:

все промежуточные точки допустимы и можно остановиться в любой момент.

Лекция 15. Метод внутренней точки

Логарифмический барьер для задач LMI

Рассмотрим аффинную функцию:

$$F(x) = F_0 + \sum_{i=1}^m x_i F_i, \quad x \in \mathbb{R}^m, \quad F_i = F_i^T \in \mathbb{R}^{n \times n}.$$

Предположения

- Матрицы F_1, \dots, F_m линейно независимы.
- Область

$$\Omega = \{x \in \mathbb{R}^m \mid F(x) > 0\}$$

непуста и ограничена.

Логарифмическая барьерная функция

$$x \in \Omega \iff \lambda_i(F(x)) > 0, \forall i.$$

На границе множества

$$x^0 \in \partial\Omega \implies \exists k : \lambda_k(F(x^0)) = 0 \implies \det F(x^0) = 0.$$

Определим барьерную функцию:

$$B(x) = \ln \det F(x)^{-1} = -\ln \det F(x), \quad x \in \Omega.$$

Свойство: если $x_k \in \Omega$, $x_k \xrightarrow{k \rightarrow \infty} \partial\Omega$, то $B(x_k) \xrightarrow{k \rightarrow \infty} +\infty$.

Т.к. Ω ограничена, то $\det F(x)$ ограничена, т.е. $0 < \det F(x) \leq C$, $\forall x \in \Omega$.
Тогда $B(x) \geq -\ln C$, $\forall x \in \Omega$.

Дифференцирование логарифмического барьера (S. Boyd, L. El Ghaoui)

Рассмотрим градиент и матрицу Гессе барьерной функции:

$$\nabla B(x) = \left[\frac{\partial B}{\partial x_i} \right]_{m \times 1}, \quad H_B(x) = \left[\frac{\partial^2 B}{\partial x_i \partial x_j} \right]_{m \times m}$$

Формулы для градиента и матрицы Гессе

$$[\nabla B(x)]_i = -\text{tr}(F(x)^{-1} F_i), \quad [H_B(x)]_{ij} = \text{tr}(F(x)^{-1} F_i F(x)^{-1} F_j).$$

Вычисление частных производных

Рассмотрим функцию $f : \mathbb{R}^m \rightarrow \mathbb{R}$ и вектора:

$$x = [x_1 \dots x_k \dots x_m]^T, \quad e_k = [0 \dots \underbrace{1}_k \dots 0]^T, \quad x + te_k = [x_1 \dots x_k + t \dots x_m]^T, \quad t \in \mathbb{R}.$$

Тогда

$$\frac{\partial f}{\partial x_k}(x) = \lim_{t \rightarrow 0} \frac{1}{t} (f(x + te_k) - f(x)),$$

$$\frac{\partial^2 f}{\partial x_i \partial x_k}(x) = \lim_{t \rightarrow 0} \frac{1}{t} \left(\frac{\partial f}{\partial x_k}(x + te_i) - \frac{\partial f}{\partial x_k}(x) \right).$$

Заметим:

$$F(x + te_k) = F_0 + \sum_{i \neq k} F_i x_i + F_k(x_k + t) = F(x) + tF_k.$$

Дифференцирование барьера: лемма

Лемма. Пусть $F_0, F_1 \in \mathbb{R}^{n \times n}$, $\det F_0 \neq 0$. Тогда: $\ln \det(F_0 + tF_1) - \ln \det F_0 = t \operatorname{tr} F_0^{-1} F_1 + o(t)$ при $t \rightarrow 0$,

Доказательство: Из формул Виета: $\det(\lambda I - A) = \lambda^n - \lambda^{n-1} \operatorname{tr} A + \dots$

Тогда

$$\begin{aligned} \det(I_n + tA) &= t^n \det\left(\frac{1}{t} I_n - (-A)\right) = \\ &= t^n \left(\frac{1}{t^n} + \frac{1}{t^{n-1}} \operatorname{tr} A + \dots\right) = \\ &= 1 + t \operatorname{tr} A + o(t) \end{aligned}$$

Дифференцирование барьера: лемма (продолжение)

Мы доказали: $\det(I_n + tA) = 1 + t \operatorname{tr} A + o(t)$. По условию $\det F_0 \neq 0$. Известно: $\ln(1 + tB) = tB + o(t)$. Тогда

$$\begin{aligned} \ln \det(F_0 + tF_1) &= \ln \det(F_0(I_n + tF_0^{-1}F_1)) = \\ &= \ln(\det(F_0) \det(I_n + tF_0^{-1}F_1)) = \\ &= \ln \det F_0 + \ln \det(I_n + tF_0^{-1}F_1) = \\ &= \ln \det F_0 + \ln(1 + t \operatorname{tr} F_0^{-1} F_1 + o(t)) = \\ &= \ln \det F_0 + t \operatorname{tr} F_0^{-1} F_1 + o(t) \end{aligned}$$

Отсюда

$$\ln \det(F_0 + tF_1) - \ln \det F_0 = t \operatorname{tr} F_0^{-1} F_1 + o(t)$$

Лемма доказана.

Дифференцирование барьера: градиент

Лемма: $\ln \det(F_0 + tF_1) - \ln \det F_0 = t \operatorname{tr} F_0^{-1} F_1 + o(t)$. Докажем формулу

$$[\nabla B(x)]_i = -\operatorname{tr} F(x)^{-1} F_i, \quad i = 1, \dots, m$$

Пусть $t \in \mathbb{R}$ — малое число. Напомним: $F(x + te_i) = F(x) + tF_i$. Т.к. $B(x) = -\ln \det F(x)$, получаем

$$\begin{aligned} B(x + te_i) - B(x) &= -\ln \det F(x + te_i) + \ln \det F(x) = \\ &= -\ln \det(F(x) + tF_i) + \ln \det F(x) = \\ &\stackrel{\text{лемма}}{=} -t \operatorname{tr} F(x)^{-1} F_i + o(t). \end{aligned}$$

$$\frac{\partial B}{\partial x_i} = \lim_{t \rightarrow 0} \frac{1}{t} (B(x + te_i) - B(x)) = -\operatorname{tr} F(x)^{-1} F_i.$$

Дифференцирование барьера: гессиан-1

Было доказано:

$$\frac{\partial B(x)}{\partial x_k} = -\operatorname{tr} F(x)^{-1} F_k, \quad F(x + te_i) = F(x) + tF_i$$

Тогда

$$H_{ik}(x) = \frac{\partial^2 B(x)}{\partial x_i \partial x_k} = \lim_{t \rightarrow 0} \frac{1}{t} \left(\frac{\partial B(x + te_i)}{\partial x_k} - \frac{\partial B(x)}{\partial x_k} \right).$$

$$\begin{aligned} \frac{\partial B(x + te_i)}{\partial x_k} - \frac{\partial B(x)}{\partial x_k} &= -\operatorname{tr} F(x + te_i)^{-1} F_k + \operatorname{tr} F(x)^{-1} F_k = \\ &= -\operatorname{tr}(F(x) + tF_i)^{-1} F_k + \operatorname{tr} F(x)^{-1} F_k \end{aligned}$$

Дифференцирование барьера: гессиан-2

$$\begin{aligned} \frac{\partial B(x + te_i)}{\partial x_k} - \frac{\partial B(x)}{\partial x_k} &= -\operatorname{tr}(F(x) + tF_i)^{-1} F_k + \operatorname{tr} F(x)^{-1} F_k \\ (I + tA)^{-1} &= I - tA + o(t) \quad \text{при } t \rightarrow 0 \end{aligned}$$

Тогда

$$\left. \begin{aligned} (F(x) + tF_i)^{-1} &= (I + tF(x)^{-1} F_i)^{-1} F(x)^{-1} = \\ &= (I - tF(x)^{-1} F_i + o(t))F(x)^{-1} \\ &= F(x)^{-1} - tF(x)^{-1} F_i F(x)^{-1} + o(t) \end{aligned} \right\} \implies$$

$$\begin{aligned} -(F(x) + tF_i)^{-1} F_k + F(x)^{-1} F_k &= tF(x)^{-1} F_i F(x)^{-1} F_k + o(t) \\ \frac{\partial B(x + te_i)}{\partial x_k} - \frac{\partial B(x)}{\partial x_k} &= t \operatorname{tr} F(x)^{-1} F_i F(x)^{-1} F_k + o(t) \end{aligned}$$

Выпуклость барьерной функции

Имеем:

$$\Omega = \{x \in \mathbb{R}^m \mid F(x) > 0\},$$

$$[H(x)]_{ij} = \text{tr}(F(x)^{-1} F_i F(x)^{-1} F_j).$$

Теорема. При линейно независимых F_1, \dots, F_m барьерная функция $B(x)$ строго выпукла, т.е. $B(\alpha x + (1-\alpha)y) < \alpha B(x) + (1-\alpha)B(y)$, $\forall x \in \Omega, \forall y \in \Omega, 0 < \alpha < 1$.

Для гладкой функции $B(x)$ строгая выпуклость эквивалентна: гессиан $H(x) > 0$, $\forall x \in \Omega$

Выпуклость барьерной функции: доказательство-1

$$\left. \begin{aligned} H_{ij}(x) &= \text{tr} F(x)^{-1} F_i F(x)^{-1} F_j = \\ &= \text{tr} F(x)^{-\frac{1}{2}} F(x)^{-\frac{1}{2}} F_i F(x)^{-\frac{1}{2}} F(x)^{-\frac{1}{2}} F_j = \\ &= \text{tr} F(x)^{-\frac{1}{2}} F_i F(x)^{-\frac{1}{2}} F(x)^{-\frac{1}{2}} F_j F(x)^{-\frac{1}{2}} \end{aligned} \right\} \implies$$

$$H_{ij}(x) = \text{tr } \tilde{F}_i(x) \tilde{F}_j(x), \quad \tilde{F}_i(x) = F(x)^{-\frac{1}{2}} F_i F(x)^{-\frac{1}{2}}$$

При $y \in \mathbb{R}^m$

$$\begin{aligned} y^T H(x) y &= \sum_{i,j=1}^m y_i y_j \text{tr } \tilde{F}_i(x) \tilde{F}_j(x) = \text{tr} \sum_{i,j=1}^m y_i \tilde{F}_i(x) y_j \tilde{F}_j(x) = \\ &= \text{tr} \left(\sum_{i=1}^m y_i \tilde{F}_i(x) \right)^2. \end{aligned}$$

Выпуклость барьерной функции: доказательство-2

Мы получили:

$$y^T H(x) y = \text{tr} \left(\sum_{i=1}^m y_i \tilde{F}_i(x) \right)^2 = \text{tr } M^2,$$

где

$$M = M^T = \sum_{i=1}^m y_i \tilde{F}_i(x) = F(x)^{-\frac{1}{2}} \left(\sum_{i=1}^m y_i F_i \right) F(x)^{-\frac{1}{2}}$$

Тогда $\text{tr } M^2 = \langle M, M \rangle \geq 0$ и, если

$$\langle M, M \rangle = 0 \iff M = 0 \iff \sum_{i=1}^m y_i F_i = 0$$

Из линейной независимости F_1, \dots, F_m следует $y = 0$.

Метод внутренней точки

Вместо задачи SDP

$$c^T x \rightarrow \min, \quad F(x) \geq 0.$$

рассмотрим задачу безусловной минимизации

$$\varphi_\mu(x) = c^T x + \mu B(x) \rightarrow \min, \quad \mu > 0 \quad \text{— барьерный параметр}$$

Очевидно

$$\nabla \varphi_\mu(x) = c + \mu \nabla B(x), \quad H_{\varphi_\mu}(x) = \mu H_B(x) > 0,$$

поэтому $\varphi_\mu(x)$ — строго выпуклая.

Если $\Omega = \{x \in \mathbb{R}^m \mid F(x) > 0\}$ ограничено снизу $\Rightarrow \varphi_\mu(x)$ имеет минимум при

$$x^*(\mu) = \operatorname{argmin} \varphi_\mu(x)$$

Метод внутренней точки (продолжение)

Для краткости обозначим:

$$H(x) = H_{\varphi_\mu}(x), \quad g(x) = \nabla \varphi_\mu(x)$$

Метод Ньютона с регулировкой шага:

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} H(x^{(k)})^{-1} g(x^{(k)})$$

Выбор шага (А. С. Немировский и Ю. Е. Нестеров):

$$\alpha^{(k)} = \begin{cases} 1 & \text{при } \delta(x^{(k)}) \leq 1/4, \\ 1/(1 + \delta(x^{(k)})) & \text{при } \delta(x^{(k)}) > 1/4, \end{cases}$$

$$\delta(x) = \sqrt{g(x)^T H(x)^{-1} g(x)}$$

Метод внутренней точки (продолжение)

Утверждения (Немировский и Нестеров):

- При данном выборе шага

$$F(x^{(k)}) > 0 \implies F(x^{(k+1)}) > 0$$

- Пусть ε — заданная точность. Тогда $\varphi(x^{(k)}) - \varphi(x^*) < \varepsilon$, если

$$k \geq C_1 + C_2 \ln \ln(1/\varepsilon) + C_3 [\varphi(x^{(0)}) - \varphi(x^*)],$$

C_1, C_2, C_3 — константы

Дробление барьера параметра

Начальные данные: x^0 (допустимый), μ^0
Параметры: $\varepsilon, \theta = 1/(4\sqrt{n} + 1)$

```

begin
     $x := x^0; \mu := \mu^0;$ 
    while  $n\mu > \varepsilon$  do
        решаем оптимиз. задачу: вычисляем  $x := x + \Delta$ 
         $\mu := (1 - \theta)\mu;$ 
    end
end

```

Немировский и Нестеров: число шагов $O(\sqrt{n} \ln(n\mu^0/\varepsilon))$

Другой вариант барьера

Заменим задачу

$$c^T x \rightarrow \min, \quad F(x) > 0$$

на задачу

$$\lambda \rightarrow \min, \quad F(x) > 0, \quad c^T x < \lambda$$

Пусть $p^* = \inf_{F(x)>0} c^T x$.

Зафиксируем λ как барьерный параметр, $\lambda > p^*$. Рассмотрим неравенство

$$\tilde{F}_\lambda(x) = \begin{bmatrix} \lambda - c^T x & 0 \\ 0 & F(x) \end{bmatrix} > 0.$$

Определим

$$\ln \det \tilde{F}_\lambda(x)^{-1} = \ln \frac{1}{\lambda - c^T x} + \ln \det F(x)^{-1}$$

Получаем задачу:

$$\ln \frac{1}{\lambda - c^T x} + \ln \det F(x)^{-1} \rightarrow \min, \quad \lambda \text{ убывает до } p^*$$

Смешанная задача (было показано)

Рассмотрим смешанную задачу (primal-dual problem):

$$\begin{aligned} \eta(x, Z) &= c^T x + \text{tr}(F_0 Z) \rightarrow \min, \\ F(x) &\geq 0, \quad Z \geq 0, \\ \text{tr}(F_i Z) &= c_i, \quad i = 1, \dots, m. \end{aligned}$$

Тогда $\eta(x, Z) \geq 0$ и, если $\eta(x^*, Z^*) = 0$, то x^* — решение прямой задачи, Z^* — решение двойственной задачи.

Возьмем $\eta > 0$ в качестве параметра:

$$\begin{aligned} c^T x + \text{tr}(F_0 Z) &\rightarrow \min, \\ F(x) &\geq 0, \quad Z \geq 0, \\ \text{tr}(F_i Z) &= c_i, \quad i = 1, \dots, m, \\ c^T x + \text{tr}(F_0 Z) &= \eta, \end{aligned}$$

При фиксированном значении η находим решение $x(\eta), Z(\eta)$.

Одновременное решение прямой и двойственной задачи

$$\begin{aligned} \ln \det F(x)^{-1} + \ln \det Z^{-1} &\rightarrow \min, \\ F(x) &> 0, \quad Z > 0, \\ \text{tr}(F_i Z) &= c_i, \quad i = 1, \dots, m, \\ c^T x + \text{tr}(F_0 Z) &= \eta. \end{aligned}$$

Здесь η — убывающий положительный параметр, $\eta \rightarrow +0$ (duality gap) — заменяет барьерный параметр.

При фиксированном значении η находим решение $x(\eta), Z(\eta)$.
Вычисления прекращается, когда $\eta < \varepsilon$.