

Методы оптимизации

Лекция 6: Подходы к построению солверов для решения задач оптимизации

Александр Катруца

Физтех-школа прикладной математики и информатики
Московский физико-технический институт



12 октября 2020 г.

На прошлой лекции

- ▶ Двойственность для задач с обобщёнными неравенствами

На прошлой лекции

- ▶ Двойственность для задач с обобщёнными неравенствами
- ▶ Коническая двойственность

На прошлой лекции

- ▶ Двойственность для задач с обобщёнными неравенствами
- ▶ Коническая двойственность
- ▶ Двойственная задача для SDP

На прошлой лекции

- ▶ Двойственность для задач с обобщёнными неравенствами
- ▶ Коническая двойственность
- ▶ Двойственная задача для SDP
- ▶ Условие Слейтера для SDP

Задача оптимизации

$$\begin{aligned} & \min_{\mathbf{x} \in \mathbb{R}^n} f_0(\mathbf{x}) \\ \text{s.t. } & f_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \\ & h_j(\mathbf{x}) = 0, \quad j = 1, \dots, p \end{aligned}$$

- Возможность эффективного решения сильно зависит от свойств f_0, f_i, h_j

Задача оптимизации

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} f_0(\mathbf{x}) \\ \text{s.t. } f_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \\ h_j(\mathbf{x}) = 0, \quad j = 1, \dots, p \end{aligned}$$

- ▶ Возможность эффективного решения сильно зависит от свойств f_0, f_i, h_j
- ▶ Если f_0, f_i, h_j аффинны, то это задача линейного программирования (LP), которая может быть решена крайне быстро

Задача оптимизации

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} f_0(\mathbf{x}) \\ \text{s.t. } f_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \\ h_j(\mathbf{x}) = 0, \quad j = 1, \dots, p \end{aligned}$$

- ▶ Возможность эффективного решения сильно зависит от свойств f_0, f_i, h_j
- ▶ Если f_0, f_i, h_j аффинны, то это задача линейного программирования (LP), которая может быть решена крайне быстро
- ▶ Простые на первый взгляд задачи с нелинейными f_i, h_j могут быть очень сложными для решения

Задача выпуклой оптимизации

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} f_0(\mathbf{x}) \\ \text{s.t. } f_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \\ \mathbf{Ax} = \mathbf{b} \end{aligned}$$

- ▶ f_0, f_i выпуклые функции: для всех \mathbf{x}, \mathbf{y} и $\alpha \in [0, 1]$

$$f(\alpha \mathbf{x} + (1 - \alpha) \mathbf{y}) \leq \alpha f(\mathbf{x}) + (1 - \alpha) f(\mathbf{y})$$

- ▶ Ограничения типа равенств аффинны

Свойства задач выпуклой оптимизации

- ▶ Подмножество задач оптимизации: LP – частный случай

Свойства задач выпуклой оптимизации

- ▶ Подмножество задач оптимизации: LP – частный случай
- ▶ Могут выглядеть очень сложно, однако решаются также эффективно как и задача LP

Свойства задач выпуклой оптимизации

- ▶ Подмножество задач оптимизации: LP – частный случай
- ▶ Могут выглядеть очень сложно, однако решаются также эффективно как и задача LP
- ▶ Встречаются гораздо чаще, чем можно было бы подумать

Свойства задач выпуклой оптимизации

- ▶ Подмножество задач оптимизации: LP – частный случай
- ▶ Могут выглядеть очень сложно, однако решаются также эффективно как и задача LP
- ▶ Встречаются гораздо чаще, чем можно было бы подумать
- ▶ Очень много приложений

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы
 - Просто для пользователя

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы
 - Просто для пользователя
 - Теряется часть преимуществ выпуклых задач

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы
 - Просто для пользователя
 - Теряется часть преимуществ выпуклых задач
- ▶ Проверка выпуклости задачи перед решением

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы
 - Просто для пользователя
 - Теряется часть преимуществ выпуклых задач
- ▶ Проверка выпуклости задачи перед решением
 - в общем случае может быть затруднительна

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы
 - Просто для пользователя
 - Теряется часть преимуществ выпуклых задач
- ▶ Проверка выпуклости задачи перед решением
 - в общем случае может быть затруднительна
- ▶ Построение выпуклой задачи из элементарных блоков

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы
 - Просто для пользователя
 - Теряется часть преимуществ выпуклых задач
- ▶ Проверка выпуклости задачи перед решением
 - в общем случае может быть затруднительна
- ▶ Построение выпуклой задачи из элементарных блоков
 - пользователь следует фиксированному набору правил при определении f_i

Общие подходы к использованию выпуклости

- ▶ Надеяться/предполагать/делать вид, что f_i выпуклы
 - Просто для пользователя
 - Теряется часть преимуществ выпуклых задач
- ▶ Проверка выпуклости задачи перед решением
 - в общем случае может быть затруднительна
- ▶ Построение выпуклой задачи из элементарных блоков
 - пользователь следует фиксированному набору правил при определении f_i
 - выпуклость проверяется автоматически

Как проверить выпуклость?

- ▶ Определение, критерии первого или второго порядка, например $f''(\mathbf{x}) \succeq 0$

Как проверить выпуклость?

- ▶ Определение, критерии первого или второго порядка, например $f''(\mathbf{x}) \succeq 0$
- ▶ Исчисление выпуклых функций: построение f специальным образом

Как проверить выпуклость?

- ▶ Определение, критерии первого или второго порядка, например $f''(x) \succeq 0$
- ▶ Исчисление выпуклых функций: построение f специальным образом
 - Дан набор простых функций, выпуклость которых известна

Как проверить выпуклость?

- ▶ Определение, критерии первого или второго порядка, например $f''(x) \succeq 0$
- ▶ Исчисление выпуклых функций: построение f специальным образом
 - Дан набор простых функций, выпуклость которых известна
 - Даны сочетания и преобразования, не меняющие выпуклость

Примеры простых выпуклых функций

- ▶ При $x > 0$: x^p для $p < 0$, $p \geq 1$ и x^{-p} для $p \in [0, 1]$

Примеры простых выпуклых функций

- ▶ При $x > 0$: x^p для $p < 0$, $p \geq 1$ и x^{-p} для $p \in [0, 1]$
- ▶ e^x , $-\log x$, $x \log x$

Примеры простых выпуклых функций

- ▶ При $x > 0$: x^p для $p < 0$, $p \geq 1$ и x^{-p} для $p \in [0, 1]$
- ▶ e^x , $-\log x$, $x \log x$
- ▶ $\langle \mathbf{a}, \mathbf{x} \rangle + b$

Примеры простых выпуклых функций

- ▶ При $x > 0$: x^p для $p < 0$, $p \geq 1$ и x^{-p} для $p \in [0, 1]$
- ▶ e^x , $-\log x$, $x \log x$
- ▶ $\langle \mathbf{a}, \mathbf{x} \rangle + b$
- ▶ $\|\mathbf{x}\|$ – любая норма

Примеры простых выпуклых функций

- ▶ При $x > 0$: x^p для $p < 0$, $p \geq 1$ и x^{-p} для $p \in [0, 1]$
- ▶ e^x , $-\log x$, $x \log x$
- ▶ $\langle \mathbf{a}, \mathbf{x} \rangle + b$
- ▶ $\|\mathbf{x}\|$ – любая норма
- ▶ $\max\{x_1, \dots, x_n\}$ и $\log(e^{x_1} + \dots + e^{x_n})$

Примеры простых выпуклых функций

- ▶ При $x > 0$: x^p для $p < 0$, $p \geq 1$ и x^{-p} для $p \in [0, 1]$
- ▶ e^x , $-\log x$, $x \log x$
- ▶ $\langle \mathbf{a}, \mathbf{x} \rangle + b$
- ▶ $\|\mathbf{x}\|$ – любая норма
- ▶ $\max\{x_1, \dots, x_n\}$ и $\log(e^{x_1} + \dots + e^{x_n})$
- ▶ $\log \det \mathbf{X}^{-1}$ для $\mathbf{X} \in \mathbb{S}_+^n$

Правила исчисления выпуклых функций

- ▶ Умножение на неотрицательную константу: f выпукла и $\alpha > 0$, тогда αf выпукла

Правила исчисления выпуклых функций

- ▶ Умножение на неотрицательную константу: f выпукла и $\alpha > 0$, тогда αf выпукла
- ▶ Сложение: f, g выпуклы, тогда $f + g$ выпукла

Правила исчисления выпуклых функций

- ▶ Умножение на неотрицательную константу: f выпукла и $\alpha > 0$, тогда αf выпукла
- ▶ Сложение: f, g выпуклы, тогда $f + g$ выпукла
- ▶ Композиция с аффинной функцией: f выпукла, тогда $f(\mathbf{Ax} + \mathbf{b})$ также выпукла

Правила исчисления выпуклых функций

- ▶ Умножение на неотрицательную константу: f выпукла и $\alpha > 0$, тогда αf выпукла
- ▶ Сложение: f, g выпуклы, тогда $f + g$ выпукла
- ▶ Композиция с аффинной функцией: f выпукла, тогда $f(\mathbf{Ax} + \mathbf{b})$ также выпукла
- ▶ Взятие максимума: f_1, \dots, f_m выпуклы, тогда $\max_{i=1, \dots, m} \{f_i(\mathbf{x})\}$ выпукла

Правила исчисления выпуклых функций

- ▶ Умножение на неотрицательную константу: f выпукла и $\alpha > 0$, тогда αf выпукла
- ▶ Сложение: f, g выпуклы, тогда $f + g$ выпукла
- ▶ Композиция с аффинной функцией: f выпукла, тогда $f(\mathbf{Ax} + \mathbf{b})$ также выпукла
- ▶ Взятие максимума: f_1, \dots, f_m выпуклы, тогда $\max_{i=1, \dots, m} \{f_i(\mathbf{x})\}$ выпукла
- ▶ Композиция: если h выпукла и возрастает, f выпукла, тогда $g(\mathbf{x}) = h(f(\mathbf{x}))$ выпукла

Правила исчисления выпуклых функций

- ▶ Умножение на неотрицательную константу: f выпукла и $\alpha > 0$, тогда αf выпукла
- ▶ Сложение: f, g выпуклы, тогда $f + g$ выпукла
- ▶ Композиция с аффинной функцией: f выпукла, тогда $f(\mathbf{Ax} + \mathbf{b})$ также выпукла
- ▶ Взятие максимума: f_1, \dots, f_m выпуклы, тогда $\max_{i=1, \dots, m} \{f_i(\mathbf{x})\}$ выпукла
- ▶ Композиция: если h выпукла и возрастает, f выпукла, тогда $g(\mathbf{x}) = h(f(\mathbf{x}))$ выпукла
- ▶ И многие другие...

Примеры

- ▶ $f(\mathbf{x}) = \max_{i=1,\dots,m} (\langle \mathbf{a}_i, \mathbf{x} \rangle + b_i)$

Примеры

- ▶ $f(\mathbf{x}) = \max_{i=1,\dots,m} (\langle \mathbf{a}_i, \mathbf{x} \rangle + b_i)$
- ▶ ℓ_1 регуляризация задачи наименьших квадратов

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

Примеры

- ▶ $f(\mathbf{x}) = \max_{i=1,\dots,m} (\langle \mathbf{a}_i, \mathbf{x} \rangle + b_i)$

- ▶ ℓ_1 регуляризация задачи наименьших квадратов

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Логарифмический барьер

$$-\sum_{i=1}^m \log(-f_i(\mathbf{x}))$$

при $\{\mathbf{x} \mid f_i(\mathbf{x}) < 0\}$ и выпуклых $f_i(\mathbf{x})$

Примеры

- ▶ $f(\mathbf{x}) = \max_{i=1,\dots,m} (\langle \mathbf{a}_i, \mathbf{x} \rangle + b_i)$

- ▶ ℓ_1 регуляризация задачи наименьших квадратов

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Логарифмический барьер

$$-\sum_{i=1}^m \log(-f_i(\mathbf{x}))$$

при $\{\mathbf{x} \mid f_i(\mathbf{x}) < 0\}$ и выпуклых $f_i(\mathbf{x})$

- ▶ Максимальное собственное значение $\mathbf{A} \in \mathbb{S}^n$:

$$\lambda_{\max}(\mathbf{A}) = \sup_{\|\mathbf{x}\|_2=1} (\mathbf{x}^\top \mathbf{Ax})$$

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера
 - сложность разработки компенсируется количеством пользователей

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера
 - сложность разработки компенсируется количеством пользователей
- ▶ Придумать и/или реализовать метод самостоятельно

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера
 - сложность разработки компенсируется количеством пользователей
- ▶ Придумать и/или реализовать метод самостоятельно
 - Трудоёмко

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера
 - сложность разработки компенсируется количеством пользователей
- ▶ Придумать и/или реализовать метод самостоятельно
 - Трудоёмко
 - Может быть эффективнее для конкретной задачи

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера
 - сложность разработки компенсируется количеством пользователей
- ▶ Придумать и/или реализовать метод самостоятельно
 - Трудоёмко
 - Может быть эффективнее для конкретной задачи
- ▶ Преобразовать задачу к стандартному виду и использовать стандартный солвер

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера
 - сложность разработки компенсируется количеством пользователей
- ▶ Придумать и/или реализовать метод самостоятельно
 - Трудоёмко
 - Может быть эффективнее для конкретной задачи
- ▶ Преобразовать задачу к стандартному виду и использовать стандартный солвер
 - Расширяет множество задач, подходящих для решения

Как решать задачу выпуклой оптимизации?

- ▶ Использовать «стандартный» солвер (для LP, QP, SDP...)
 - лёгкий путь
 - задача **должна быть** в стандартной форме для выбранного солвера
 - сложность разработки компенсируется количеством пользователей
- ▶ Придумать и/или реализовать метод самостоятельно
 - Трудоёмко
 - Может быть эффективнее для конкретной задачи
- ▶ Преобразовать задачу к стандартному виду и использовать стандартный солвер
 - Расширяет множество задач, подходящих для решения
 - Преобразование может быть громоздким

Общие методы решения задач выпуклой оптимизации

Субградиентный метод, метод эллипсоидов, проксимальный метод и их вариации

Общие методы решения задач выпуклой оптимизации

Субградиентный метод, метод эллипсоидов, проксимальный метод и их вариации

- ▶ В основном разработаны в СССР в 1960-1970-ых годах, подробнее см. [заметки Б.Т. Поляка](#)

Общие методы решения задач выпуклой оптимизации

Субградиентный метод, метод эллипсоидов, проксимальный метод и их вариации

- ▶ В основном разработаны в СССР в 1960-1970-ых годах, подробнее см. [заметки Б.Т. Поляка](#)
- ▶ Универсальные методы решения задач выпуклой оптимизации, даже для недифференцируемых f_i

Общие методы решения задач выпуклой оптимизации

Субградиентный метод, метод эллипсоидов, проксимальный метод и их вариации

- ▶ В основном разработаны в СССР в 1960-1970-ых годах, подробнее см. [заметки Б.Т. Поляка](#)
- ▶ Универсальные методы решения задач выпуклой оптимизации, даже для недифференцируемых f_i
- ▶ Метод эллипсоидов эффективен в теории (полиномиален)

Общие методы решения задач выпуклой оптимизации

Субградиентный метод, метод эллипсоидов, проксимальный метод и их вариации

- ▶ В основном разработаны в СССР в 1960-1970-ых годах, подробнее см. [заметки Б.Т. Поляка](#)
- ▶ Универсальные методы решения задач выпуклой оптимизации, даже для недифференцируемых f_i
- ▶ Метод эллипсоидов эффективен в теории (полиномиален)
- ▶ На практике такие методы могут быть медленными

Методы внутренней точки (IPM) для выпуклых задач

- ▶ Interior-Point Polynomial Algorithms in Convex Programming,
Y. Nesterov, A. Nemirovskii, 1994

Методы внутренней точки (IPM) для выпуклых задач

- ▶ Interior-Point Polynomial Algorithms in Convex Programming, Y. Nesterov, A. Nemirovskii, 1994
- ▶ Обзор про IPM см. [тут](#)

Методы внутренней точки (IPM) для выпуклых задач

- ▶ Interior-Point Polynomial Algorithms in Convex Programming, Y. Nesterov, A. Nemirovskii, 1994
- ▶ Обзор про IPM см. [тут](#)
- ▶ Применим для **гладких** f_i и задач в конической форме (LP, SOCP, SDP)

Методы внутренней точки (IPM) для выпуклых задач

- ▶ Interior-Point Polynomial Algorithms in Convex Programming, Y. Nesterov, A. Nemirovskii, 1994
- ▶ Обзор про IPM см. [тут](#)
- ▶ Применим для **гладких** f_i и задач в конической форме (LP, SOCP, SDP)
- ▶ Чрезвычайно эффективный метод: необходимо сделать несколько десятков итераций, независимо от размерности задачи

Методы внутренней точки (IPM) для выпуклых задач

- ▶ Interior-Point Polynomial Algorithms in Convex Programming, Y. Nesterov, A. Nemirovskii, 1994
- ▶ Обзор про IPM см. [тут](#)
- ▶ Применим для **гладких** f_i и задач в конической форме (LP, SOCP, SDP)
- ▶ Чрезвычайно эффективный метод: необходимо сделать несколько десятков итераций, независимо от размерности задачи
- ▶ На каждой итерации надо решить некоторую линейную систему

А если IPM нельзя применить к задаче?

А если IPM нельзя применить к задаче?

- ▶ Пример: ℓ_1 регуляризация задачи наименьших квадратов

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

А если IPM нельзя применить к задаче?

- ▶ Пример: ℓ_1 регуляризация задачи наименьших квадратов

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Задача выпуклая, но f негладкая!

А если IPM нельзя применить к задаче?

- ▶ Пример: ℓ_1 регуляризация задачи наименьших квадратов

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Задача выпуклая, но f негладкая!
- ▶ **Основная идея:** изменить задачу так, чтобы IPM можно было применять

А если IPM нельзя применить к задаче?

- ▶ Пример: ℓ_1 регуляризация задачи наименьших квадратов

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Задача выпуклая, но f негладкая!
- ▶ **Основная идея:** изменить задачу так, чтобы IPM можно было применять
- ▶ Даже если в новой задаче будет больше переменных и ограничений, она может быть эффективно решена с помощью IPM

Пример

- ▶ Исходная задача: n переменных, нет ограничений

$$\min_x \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

Пример

- ▶ Исходная задача: n переменных, нет ограничений

$$\min_x \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Введём новую переменную $\mathbf{t} \in \mathbb{R}^n$ и новые ограничения $|x_i| \leq t_i$:

$$\begin{aligned} \min_{(\mathbf{x}, \mathbf{t})} \quad & \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \mathbf{1}^\top \mathbf{t} \\ \text{s.t.} \quad & -\mathbf{t} \leq \mathbf{x} \leq \mathbf{t} \end{aligned}$$

Пример

- ▶ Исходная задача: n переменных, нет ограничений

$$\min_x \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Введём новую переменную $\mathbf{t} \in \mathbb{R}^n$ и новые ограничения $|x_i| \leq t_i$:

$$\begin{aligned} \min_{(\mathbf{x}, \mathbf{t})} \quad & \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \mathbf{1}^\top \mathbf{t} \\ \text{s.t.} \quad & -\mathbf{t} \leq \mathbf{x} \leq \mathbf{t} \end{aligned}$$

- ▶ В новой задаче $2n$ переменных и $2n$ ограничений, но она гладкая!

Пример

- ▶ Исходная задача: n переменных, нет ограничений

$$\min_x \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad \lambda > 0$$

- ▶ Введём новую переменную $\mathbf{t} \in \mathbb{R}^n$ и новые ограничения $|x_i| \leq t_i$:

$$\begin{aligned} \min_{(\mathbf{x}, \mathbf{t})} \quad & \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \mathbf{1}^\top \mathbf{t} \\ \text{s.t.} \quad & -\mathbf{t} \leq \mathbf{x} \leq \mathbf{t} \end{aligned}$$

- ▶ В новой задаче $2n$ переменных и $2n$ ограничений, но она гладкая!
- ▶ Важно: задачи эквивалентны! Решив одну, получаем решение другой и наоборот

Преобразование задачи и эффективность решения

- ▶ Дана выпуклая задача P_0

Преобразование задачи и эффективность решения

- ▶ Дана выпуклая задача P_0
- ▶ Выполняются последовательные эквивалентные преобразования

$$P_0 \rightarrow P_1 \rightarrow \dots \rightarrow P_K,$$

где P_K – задача, которую можно решать ИРМ

Преобразование задачи и эффективность решения

- ▶ Дана выпуклая задача P_0
- ▶ Выполняются последовательные эквивалентные преобразования

$$P_0 \rightarrow P_1 \rightarrow \dots \rightarrow P_K,$$

где P_K – задача, которую можно решать ИРМ

- ▶ Эффективное решение P_K

Преобразование задачи и эффективность решения

- ▶ Дана выпуклая задача P_0
- ▶ Выполняются последовательные эквивалентные преобразования

$$P_0 \rightarrow P_1 \rightarrow \dots \rightarrow P_K,$$

где P_K – задача, которую можно решать ИРМ

- ▶ Эффективное решение P_K
- ▶ Обратное преобразование решения P_K в решение P_0

Преобразование задачи и эффективность решения

- ▶ Дана выпуклая задача P_0
- ▶ Выполняются последовательные эквивалентные преобразования

$$P_0 \rightarrow P_1 \rightarrow \dots \rightarrow P_K,$$

где P_K – задача, которую можно решать ИРМ

- ▶ Эффективное решение P_K
- ▶ Обратное преобразование решения P_K в решение P_0
- ▶ P_K может иметь больше ограничений и/или переменных, но наличие определённой структуры и высокая эффективность ИРМ компенсируют это

Примеры преобразований задач

- ▶ Правила преобразования выпуклых функций порождают преобразования задач

Примеры преобразований задач

- ▶ Правила преобразования выпуклых функций порождают преобразования задач
- ▶ $\max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$

Примеры преобразований задач

- ▶ Правила преобразования выпуклых функций порождают преобразования задач
- ▶ $\max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Вводим новую переменную $t = \max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$

Примеры преобразований задач

- ▶ Правила преобразования выпуклых функций порождают преобразования задач
- ▶ $\max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Вводим новую переменную $t = \max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Добавляем ограничения $f_1(\mathbf{x}) \leq t, f_2(\mathbf{x}) \leq t$

Примеры преобразований задач

- ▶ Правила преобразования выпуклых функций порождают преобразования задач
- ▶ $\max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Вводим новую переменную $t = \max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Добавляем ограничения $f_1(\mathbf{x}) \leq t, f_2(\mathbf{x}) \leq t$
- ▶ $h(f(\mathbf{x}))$

Примеры преобразований задач

- ▶ Правила преобразования выпуклых функций порождают преобразования задач
- ▶ $\max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Вводим новую переменную $t = \max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Добавляем ограничения $f_1(\mathbf{x}) \leq t, f_2(\mathbf{x}) \leq t$
- ▶ $h(f(\mathbf{x}))$
 - Вводим новую переменную $t = f(\mathbf{x})$

Примеры преобразований задач

- ▶ Правила преобразования выпуклых функций порождают преобразования задач
- ▶ $\max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Вводим новую переменную $t = \max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$
 - Добавляем ограничения $f_1(\mathbf{x}) \leq t, f_2(\mathbf{x}) \leq t$
- ▶ $h(f(\mathbf{x}))$
 - Вводим новую переменную $t = f(\mathbf{x})$
 - Добавляем ограничение $f(\mathbf{x}) \leq t$

От доказательства выпуклости к применимости IPM

$$\begin{aligned} & \min_{\mathbf{x} \in \mathbb{R}^n} f_0(\mathbf{x}) \\ & \text{s.t. } f_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \\ & \quad \mathbf{Ax} = \mathbf{b} \end{aligned}$$

- Построение f_i из элементарных функций и правил преобразований даёт доказательство выпуклости

От доказательства выпуклости к применимости ИРМ

$$\begin{aligned} & \min_{\mathbf{x} \in \mathbb{R}^n} f_0(\mathbf{x}) \\ & \text{s.t. } f_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \\ & \quad \mathbf{Ax} = \mathbf{b} \end{aligned}$$

- ▶ Построение f_i из элементарных функций и правил преобразований даёт доказательство выпуклости
- ▶ Аналогичный разбор даёт преобразование задачи к форме, состоящей из элементарных функций и аффинных равенств

От доказательства выпуклости к применимости IPM

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} f_0(\mathbf{x}) \\ \text{s.t. } f_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \\ \mathbf{Ax} = \mathbf{b} \end{aligned}$$

- ▶ Построение f_i из элементарных функций и правил преобразований даёт доказательство выпуклости
- ▶ Аналогичный разбор даёт преобразование задачи к форме, состоящей из элементарных функций и аффинных равенств
- ▶ Если элементарные функции подходят для IPM, преобразование автоматически даёт форму задачи, которая может быть решена IPM

Disciplined convex programming (DCP)

- ▶ Задаются искомые переменные и фиксированные параметры

Disciplined convex programming (DCP)

- ▶ Задаются искомые переменные и фиксированные параметры
- ▶ Целевая функция и ограничения строятся из элементарных функций с помощью правил композиций и сочетаний

Disciplined convex programming (DCP)

- ▶ Задаются искомые переменные и фиксированные параметры
- ▶ Целевая функция и ограничения строятся из элементарных функций с помощью правил композиций и сочетаний
- ▶ Задача выпукла по построению

Disciplined convex programming (DCP)

- ▶ Задаются искомые переменные и фиксированные параметры
- ▶ Целевая функция и ограничения строятся из элементарных функций с помощью правил композиций и сочетаний
- ▶ Задача выпукла по построению
- ▶ Автоматически разбирается на элементы

Disciplined convex programming (DCP)

- ▶ Задаются искомые переменные и фиксированные параметры
- ▶ Целевая функция и ограничения строятся из элементарных функций с помощью правил композиций и сочетаний
- ▶ Задача выпукла по построению
- ▶ Автоматически разбирается на элементы
- ▶ Приводится к форме для запуска IPM

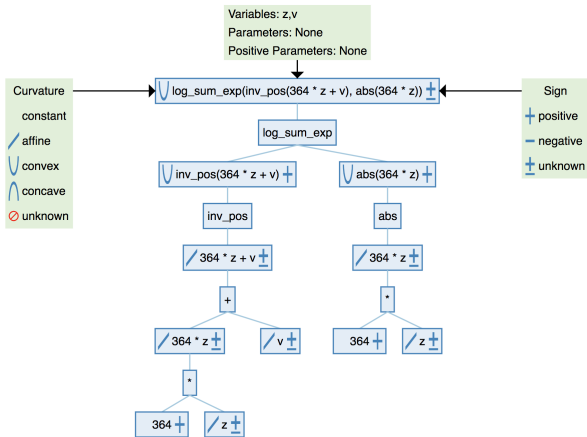
Disciplined convex programming (DCP)

- ▶ Задаются искомые переменные и фиксированные параметры
- ▶ Целевая функция и ограничения строятся из элементарных функций с помощью правил композиций и сочетаний
- ▶ Задача выпукла по построению
- ▶ Автоматически разбирается на элементы
- ▶ Приводится к форме для запуска IPM
- ▶ Решается некоторым стандартным пакетом для IPM

Disciplined convex programming (DCP)

- ▶ Задаются искомые переменные и фиксированные параметры
- ▶ Целевая функция и ограничения строятся из элементарных функций с помощью правил композиций и сочетаний
- ▶ Задача выпукла по построению
- ▶ Автоматически разбирается на элементы
- ▶ Приводится к форме для запуска IPM
- ▶ Решается некоторым стандартным пакетом для IPM
- ▶ Восстанавливается решение исходной задачи

Пример разбора выражения и проверка его на выпуклость



Больше примеров можно найти на [сайте](#)

История

- ▶ Системы **AMPL**, **GAMS** – 1970-ые
- ▶ Пакеты для задач SDP/LMI: `sdpsol` (Wu, Boyd), `lmilab` (Gahinet, Nemirovsky), `lmitool` (El Ghaoui) – 1990-ые
- ▶ `yalmip` (Löfberg 2000–)
- ▶ automated convexity checking (Crusius PhD thesis 2002)
- ▶ disciplined convex programming (DCP) (Grant, Boyd, Ye 2004)
- ▶ `cvx` (Grant, Boyd, Ye 2005) для MATLAB
- ▶ `cvxopt` (Dahl, Vandenberghe 2005)
- ▶ `Convex.jl` (M. Udell, et. al. 2014) для Julia
- ▶ `cvxpy` (Diamond, Boyd 2016) для Python

Главное по DCP

Pro:

- ▶ Проверка выпуклости и генерация преобразования задачи для IPM

Главное по DCP

Pro:

- ▶ Проверка выпуклости и генерация преобразования задачи для IPM
- ▶ Построение задачи: элементарные выпуклые функции + правила композиций и преобразований

Contra:

Главное по DCP

Pro:

- ▶ Проверка выпуклости и генерация преобразования задачи для IPM
- ▶ Построение задачи: элементарные выпуклые функции + правила композиций и преобразований
- ▶ Очень похоже на математическую нотацию

Contra:

Главное по DCP

Pro:

- ▶ Проверка выпуклости и генерация преобразования задачи для IPM
- ▶ Построение задачи: элементарные выпуклые функции + правила композиций и преобразований
- ▶ Очень похоже на математическую нотацию

Contra:

- ▶ Не про «plug & play» или «try my code»

Главное по DCP

Pro:

- ▶ Проверка выпуклости и генерация преобразования задачи для IPM
- ▶ Построение задачи: элементарные выпуклые функции + правила композиций и преобразований
- ▶ Очень похоже на математическую нотацию

Contra:

- ▶ Не про «plug & play» или «try my code»
- ▶ Нельзя записать произвольную задачу и надеяться, что она будет выпукла

Солверы для решения общих задач оптимизации

- ▶ ipopt
- ▶ Pyomo
- ▶ Gurobi

- ▶ Правила построения выпуклых функций

- ▶ Правила построения выпуклых функций
- ▶ Сведение задач к стандартной форме

Главное

- ▶ Правила построения выпуклых функций
- ▶ Сведение задач к стандартной форме
- ▶ Disciplined convex programming

Главное

- ▶ Правила построения выпуклых функций
- ▶ Сведение задач к стандартной форме
- ▶ Disciplined convex programming
- ▶ Примеры

- ▶ Правила построения выпуклых функций
- ▶ Сведение задач к стандартной форме
- ▶ Disciplined convex programming
- ▶ Примеры
- ▶ Солверы для решения задач оптимизации