Федеральное государственное автономное
образовательное учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»
Физтех-школа прикладной математики и информатики

ЛЕКЦИИ И СЕМИНАРЫ ПО МЕТОДАМ ОПТИМИЗАЦИИ

Авторы:
А. Н. Безносиков, Е. Д. Бородич, Д. М. Двинских, Г. В. Кормаков,
Н. М. Корнилов, И. А. Курузов, А. А. Лобанов, Д. С. Метелев,
А. А. Моложавенко, С. А. Чежегов, Ю. И. Шароватова, Н. Е. Юдин,
Д. В. Ярмошик, А. В. Андреев, А. И. Богданов, Д. А. Былинкин,
Е. В. Рябинин, С. С. Ткаченко, А. А. Шестаков

МОСКВА
ФПМИ МФТИ
2023

Учебное пособие содержит конспект лекций и материалы семинарских занятий по курсу «Методы оптимизации». В таком виде курс был прочитан осенью 2023–2024 учебного года студентам 3-го курса физтех-школы прикладной математики и информатики МФТИ. Важно отметить, что материалы курсы опираются на различные курсы методов оптимизации, прочитанные в России и за рубежом. А именно, авторы данного пособия хотели бы поблагодарить А. Бен-Тала, А. Г. Бирюкова, А. В. Гасникова, В. Я. Жадана, А. М. Катруцу, Д. А. Кропотова, А. С. Немировского, Ю. Е. Нестерова, Б. Т. Поляка, Ф. С. Стонякина, М. Такача, Р. Хильдебранда, М. Шмидта, М. Ягги за неоценимый вклад в преподавание и популяризацию численных методов оптимизации во всем мире.

РАБОТА НАД ПОСОБИЕМ ВЕДЕТСЯ В ДАННЫЙ МОМЕНТ! АВТОРЫ ПРИЗНАТЕЛЬНЫ ЗА ЛЮБЫЕ КОММЕНТАРИИ ПО ЕГО УЛУЧШЕНИЮ!

Содержание

1. Обозначения и классические факты

Обозначения
$\mathbb{N}$ – натуральные числа
$\mathbb{Z}$ – целые числа
$\mathbb{R}$ – вещественные числа
$\mathbb{R}_+$ – неотрицательные вещественные числа
$\mathbb{R}_{++}$ – положительные вещественные числа

$\forall$ – квантор «для всех».
$\exists$ – квантор «существует».

Для вектора $x \in \mathbb{R}^d$ мы обозначаем i-ю координату через x_i.
Скалярное произведение в $\mathbb{R}^d$ в стандартном базисе считается по формуле $\langle x, y\rangle := \sum_{i=1}^d x_i y_i = \langle y, x\rangle$.
Евклидова норма на $\mathbb{R}^d$ индуцируется скалярным произведением $\|x\|_2 := \sqrt{\langle x, x\rangle}$, где $x \in \mathbb{R}^d$.
p-норма $\|\cdot\|_p$ на $\mathbb{R}^d$ для $1 \le p < \infty$ определяется как $\|x\|_p := \left(\sum_{i=1}^d |x_i|^p\right)^{\frac{1}{p}}$.
∞-норма $\|\cdot\|_\infty$ на $\mathbb{R}^d$ определяется как $\|x\|_\infty := \max_{i=1,\dots,d} |x_i|$.

$A \in \mathbb{S}^m \iff A = A^\top$.
$A \in \mathbb{S}^m_+ \iff A \in \mathbb{S}^n; \quad \forall x: \quad x^\top A x \ge 0$.
$A \in \mathbb{S}^m_{++} \iff A \in \mathbb{S}^n; \quad \forall x \ne 0: \quad x^\top A x > 0$.

В случае матриц из $\mathbb{R}^{m\times k}$ скалярное произведение в стандартном базисе определено как $\langle X, Y\rangle := \mathrm{Tr}(X^\top Y) = \sum_{i=1}^m \sum_{j=1}^k X_{ij} Y_{ij} = \langle Y, X\rangle$.
Поэлементное умножение одинаковых по размерностям матриц обозначается как $\odot$: $(A \odot B)_{ij} = A_{ij} \cdot B_{ij}$.
Индуцированная векторной нормой $\|\cdot\|$ матричная норма для $A \in \mathbb{R}^{m\times k}$ определятся как $\|A\| := \sup_{\|x\|=1} Ax$.
В частности, можно привести такие замкнутые формы для классических норм:
1) $\|A\|_\infty = \max_{1\le i\le m} \sum_{j=1}^k |a_{ij}|$,
2) $\|A\|_1 = \max_{1\le j\le k} \sum_{i=1}^m |a_{ij}|$,
3) $\|A\|_2 = \sup_{\langle x,x\rangle=1} \sqrt{\langle Ax, Ax\rangle} = \sqrt{\lambda_{\max}(A^\top A)}$.
Норма Фробениуса для матрицы $A \in \mathbb{R}^{m\times k}$ и $q = \min(m, k)$ определяется как $||A||_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^k A_{ij}^2} = \sqrt{\mathrm{Tr}(A^\top A)} = \left(\sum_{i=1}^q \lambda_i(A^\top A)\right)^{\frac{1}{2}}$.

Полезные неравенства

- Неравенство Йенсена:

  Пусть функция $f(x)$ является выпуклой, тогда для m чисел $\alpha_i \geq 0 : \sum_{i=1}^{m} \alpha_i = 1$ и m векторов x_i выполняется неравенство

$$f\left(\sum_{i=1}^{m} \alpha_i x_i\right) \leq \sum_{i=1}^{m} \alpha_i f(x_i).$$

- Неравенство Коши-Буняковского-Шварца:

  Для векторов $x, y \in \mathbb{R}^d$ и чисел $p \geq 1, \frac{1}{q} + \frac{1}{p} = 1$ выполняется неравенство

$$|\langle x, y \rangle| \leq \|x\|_p \|y\|_q.$$

2. Полезные факты из линейной алгебры

Дмитрий Былинкин

2.1. Базовые факты

Определение 2.1. Столбцовым рангом матрицы $A \in \mathbb{R}^{m\times k}$ называется число $Rg_c(A)$, такое, что в A существует линейно независимая система из $Rg_c(A)$ столбцов и нет линейно независимой системы из большего числа столбцов. Аналогично вводится строчный ранг.

Предложение 2.2. Столбцовый и строчный ранг матрицы $A \in \mathbb{R}^{m\times k}$ равны.

Доказательство. □ Рассмотрим систему из $Rg_s(A)$ линейно независимых (базисных) строк в A. Очевидно, по ним раскладываются остальные строки, если они есть. Вычтем из каждой небазисной строки подходящую линейную комбинацию базисных, которая обратит ее в нулевую. Получили матрицу $A' \in \mathbb{R}^{m\times k} : Rg_s(A') = Rg_s(A)$. При этом очевидно, что при таком преобразовании $Rg_c(A') = Rg_c(A)$ и подматрица $\widehat{A} \in \mathbb{R}^{Rg_s(A)\times Rg_c(A)}$, стоящая в пересечении базисных строк и столбцов не изменилась.
Рассмотрим $\widehat{A}$. Если $Rg_c(A) > Rg_s(A)$, то строки $\widehat{A}$ линейно зависимы, поскольку любые $n+1$ строк длины n линейно зависимы. Тогда $Rg_c(A) \le Rg_s(A)$. Аналогично получим $Rg_c(A) \ge Rg_s(A)$. Тогда $Rg_c(A) = Rg_s(A)$. ■

Определение 2.3. $A \in \mathbb{R}^{m\times k}$ имеет полный ранг, если

$$Rg(A) = \min\{m, k\}.$$

Определение 2.4. $A \in \mathbb{R}^{m\times m}, h \in \mathbb{R}^m$. Если $Ah = \lambda h$, то h называется собственным вектором матрицы A, а скаляр λ - соответствующим ему собственным числом.

Замечание 2.5.

$$\operatorname{Tr} A = \sum_{i=1}^{m} \lambda_i$$

$$\det A = \prod_{i=1}^{m} \lambda_i$$

Определение 2.6. Рассмотрим $A \in \mathbb{R}^{m\times m}$. Для $n \in \{1, ..., m\}$ главным минором будем называть определитель

$$M_n = \begin{vmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{vmatrix}.$$

Определение 2.7. Матрица $A \in \mathbb{R}^{m\times m}$ называется положительно определенной (обозначается $A \succ 0$), если $\forall x \in \mathbb{R}^m \to x^T Ax > 0$. Аналогично вводится отрицательная определенность. В определении полуопределенности неравенства нестрогие.

Теорема 2.8. (Критерий Сильвестра):

$$A \in \mathbb{R}^{m\times m} : A \succ 0 \Leftrightarrow M_n > 0, \forall n \in \{1, ..., m\}.$$

2.2. Системы линейных уравнений

Рассмотрим систему линейных уравнений (СЛУ):

$$Ax = b, A \in \mathbb{R}^{m\times k}.$$

Если $m > k$, то система называется переопределенной и в общем случае не имеет решений. Если же $m < k$, то система недоопределенная и ее решение неединственно.
Далее в этом разделе будем рассматривать СЛУ с квадратной матрицей $A \in \mathbb{R}^{m\times m}$, к переопределенным системам вернемся позже.

Предложение 2.9. $\exists! x \in \mathbb{R}^m : Ax = b \Leftrightarrow \det A \neq 0.$

Предложение 2.10. $\exists! x \in \mathbb{R}^m : Ax = b \Leftrightarrow Rg(A) = m.$

Для произвольной матрицы $A \in \mathbb{R}^{m\times k}$ обращение требует $O(n^3)$ итераций, поэтому решать СЛУ дорого, однако существуют особые случаи, когда СЛУ может быть легко решена:

- $O(n)$ операций для диагональной матрицы,
- $O(n^2)$ для нижне- и верхнетругольной матрицы,
- $O(n^2)$ для ортогональной матрицы ($A^T A = I$),
- $O(n \log n)$ - например, если матрица циркулянтная.

Напрашивается идея единственный раз разложить матрицу в произведение "простых", а далее, имея готовое разложение, решать несколько СЛУ вместо исходной. Существует достаточно много способов разложить матрицу, далее мы разберем некоторые наиболее распространенные.

2.2.1. PLU-разложение

Рассматриваемое разложение можно вычислить за $\frac{2n^3}{3}$ операций методом Гаусса. Чтобы работать с произвольной матрицей, нам понадобятся вспомогательные утверждения:

Предложение 2.11. P - матрица перестановок строк A,

$\det A \neq 0, A \in \mathbb{R}^{m\times m} \Rightarrow \exists P \in \mathbb{R}^{m\times m} : M_n(PA) \neq 0, \forall n \in \{1, ..., m\}$.

Предложение 2.12. Матрицы перестановок образуют группу по умножению с I в качестве единичного элемента.

Теорема 2.13. $M_n(A) \neq 0, A \in \mathbb{R}^{m\times m} \Rightarrow \exists! L, U : A = LU$, где L - нижнетреугольная матрица, U - верхнетреугольная матрица с диагональными элементами, равными единице.

Доказательство. □ Существование разложения легко проверяется по индукции. Покажем единственность. Пусть есть два разложения: $LU = \widehat{L}\widehat{U} \Rightarrow \widehat{L}^{-1}L = \widehat{U}U^{-1}$. Левая часть равенства - нижнетреугольная матрица, правая - верхнетреугольная с единичной диагональю. Таким образом, обе матрицы единичные: $\widehat{L}^{-1}L = \widehat{U}U^{-1} = I \Rightarrow L = \widehat{L}, U = \widehat{U}$. ■

Из теоремы и предложений следует, что для любой невырожденной матрицы существует единственное разложение $A = PLU$.

2.2.2. Разложение Холецкого

Теорема 2.14. $A = A^T, A \succ 0 \Rightarrow \exists L : A = LL^T$, где L - нижнетреугольная матрица.

Доказательство. □ **База:** для $m = 1$ разложение очевидно.
Предположение: предположим, что утверждение доказано для матриц до порядка $(m-1)$.
Шаг:

$$A_m = \left(\begin{array}{c:c} A_{m-1} & b \\ \hdashline b^T & a_{m,m} \end{array}\right)$$

$A_m \succ 0 \Rightarrow A_{m-1} \succ 0$ по критерию Сильвестра. Тогда:
$\exists L_{m-1} : A_{m-1} = L_{m-1}L_{m-1}^T$. $(\det L_{m-1})^2 = \det A_{m-1} \neq 0 \Rightarrow \exists c : L_{m-1}c = b$
Введем x соотношением $c^T c + x^2 = a_{m,m}$. Тогда получим:

$$\left(\begin{array}{c:c} L_{m-1} & 0 \\ \hdashline c^T & x \end{array}\right) \left(\begin{array}{c:c} L_{m-1}^T & c \\ \hdashline 0 & x \end{array}\right) = A_m$$

Имеем $(\det L_{m-1})^2 x^2 = \det A_m$, где правая часть положительна, т.к. матрица A_m положительно определена. Таким образом, $x \in \mathbb{R}$ и мы построили разложение Холецкого. ■

Замечание 2.15. Разложение Холецкого может быть вычислено за $\frac{n^3}{3}$ операций.

Замечание 2.16. Условие симметричности матрицы A существенно.

Указание: Рассмотреть $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

2.2.3. Тонкости вычисления LU-разложения

Рассмотрим матрицу $A = \begin{pmatrix} \varepsilon & 1 \\ 1 & 1 \end{pmatrix}$. При уменьшении параметра мы столкнемся с неустойчивостью, в то время как вычисление разложения Холецкого всегда устойчиво. Далее нам потребуется еще одно вспомогательное утверждение для линейного оператора на **банаховом** пространстве X:

Предложение 2.17. (Ряд Неймана):

$$A \in \mathcal{L}(X) : \rho(A) < 1 \Rightarrow (I - A)^{-1} = \sum_{i=0}^{\infty} A^i,$$

где $\rho(A) = \lim_{k\to\infty} \|A^k\|^{\frac{1}{k}}$ - спектральный радиус.

Замечание 2.18. Поскольку в конечномерном случае $\rho(A) = \lambda_{max}(A)$, можем переписать:

$$A \in \mathbb{R}^{m\times m} : \lambda_{max}(A) < 1 \Rightarrow (I - A)^{-1} = \sum_{i=0}^{\infty} A^i.$$

В частности, можно показать $\|\sum_{i=1}^{\infty} A^i\| \le \frac{\|I\|}{1-\|A\|}$.

Предложение 2.19. Рассмотрим малое возмущение $E \in \mathbb{R}^{m\times m}$: $\lambda_{max}(A^{-1}E) < 1$. Тогда, пользуясь утвержд. 2.17:

$$(A + E)^{-1} = (A(I + A^{-1}E))^{-1} = \left(\sum_{i=0}^{\infty}(-A^{-1}E)^i\right) A^{-1}.$$

Более того, $\frac{\|(A+E)^{-1}-A^{-1}\|}{\|A^{-1}\|} \le \frac{\|A^{-1}\|\|E\|\|I\|}{1-\|A^{-1}E\|}$ по любой операторной норме.

Получили полезную оценку на относительное отклонение матрицы, обратной к возмущенной.
Продолжим сравнивать возмущенную СЛУ с исходной:

$$\begin{cases} (A + \Delta A)y = b + \Delta b \\ Ax = b \end{cases}$$

Определение 2.20. Назовем $\mu(A) = \|A\|\|A^{-1}\|$ числом обусловленности матрицы A.

Замечание 2.21. Если λ - собственное число матрицы A, то $\frac{1}{\lambda}$ - собственное число матрицы A^{-1}. Отсюда следует, что по второй норме $\mu_2(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}$.

Предложение 2.22. $\frac{\|y-x\|}{\|x\|} \le \frac{\mu(A)}{1-\mu(A)\frac{\|\Delta A\|}{\|A\|}} \left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|}\right)$.

Доказательство. □

$$\begin{aligned} y - x =& (A + \Delta A)^{-1}(b + \Delta b) - A^{-1}b \\ =& \left(\sum_{i=0}^{\infty} (-A^{-1}\Delta A)^i - I \right) A^{-1}b + \left(-A^{-1}\Delta A)^i\right) A^{-1}\Delta b \\ =& \left(\sum_{i=1}^{\infty} (-A^{-1}\Delta A)^i \right) A^{-1}b + \left(-A^{-1}\Delta A)^i\right) A^{-1}\Delta b \\ \frac{\|y - x\|}{\|x\|} \leq& \frac{1}{\|A^{-1}b\|} \left(\frac{\|A^{-1}\|\|\Delta A\|}{1 - \|A^{-1}\Delta A\|} \|A^{-1}b\| + \frac{1}{1 - \|A^{-1}\Delta A\|} \|A^{-1}\Delta b\| \right) \\ \leq& \frac{\|A\|\|A^{-1}\|}{1 - \|A^{-1}\Delta A\|} \frac{\Delta A}{A} + \frac{\|A^{-1}\|}{1 - \|A^{-1}\Delta A\|} \frac{\Delta b}{A^{-1}b} \end{aligned}$$

Из свойств нормы очевидно следует $\|A^{-1}b\| \geq \frac{\|b\|}{\|A\|}$. Подставим это в цепочку неравенств:

$$\frac{\|y - x\|}{\|x\|} \leq \frac{\|A\|\|A^{-1}\|}{1 - \|A^{-1}\|\|\Delta A\|} \left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|} \right).$$

Добавляя умную единицу в знаменатель и пользуясь определен. 2.20, получаем требуемое. ■

2.2.4. Переопределенные СЛУ

Определение 2.23. $A \in \mathbb{C}^{m \times k}$ называется эрмитовой, если

$$A^* = A^{-1},$$

где $*$ - последовательное применение комплексного сопряжения и транспонирования.

Для переопределенной системы стоит задача о минимизации невязки, то есть

$$\min_{x \in \mathbb{R}^k} \left[f(x) = \|Ax - b\|_2^2 \right].$$

Вычислим градиент целевой функции, чтобы применить необходимое условие экстремума:

$$\nabla f(x) = 2(A^*Ax - A^*b) = 0.$$

Таким образом, надо решать СЛУ с квадратной матрицей $A^*Ax = A^*b$ - оно называется нормальным уравнением, а матрица - матрицей Грама. По второй норме $\|A\| = \|A^*\|$, откуда легко следует, что $\mu(A^*A) =$

$\mu(A)^2$. Решать систему в таком виде - не самая лучшая идея. Ниже рассматривается ряд более подходящих методов.

2.2.5. Псевдообратная матрица

Определение 2.24. Матрица

$$A^\dagger = \lim_{\alpha \to 0} (\alpha I + A^* A)^{-1} A^*$$

называется псевдообратной матрицей Мура-Пенроуза.

Перечислим некоторые основные свойства псевдообратной матрицы:

- Если A имеет полный ранг, то A^*A, очевидно, невырождена, и $A^\dagger = (A^*A)^{-1}A^*$,
- Для квадратной невырожденной матрицы псевдообратная матрица совпадает с обратной,
- Если столбцы A линейно зависимы, то $A^\dagger b$ дает решение минимальной евклидовой нормы.

2.2.6. QR-разложение

Любую матрицу за $\frac{4n^3}{3}$ операций можно представить в виде $A = QR$, где Q - унитарная, а R - верхнетреугольная.

Предложение 2.25. Для матрицы полного ранга задача поиска оптимального x эквивалентна решению СЛУ с квадратной матрицей $RX = Q^*b$, где R, Q находятся из QR-разложения.

Доказательство. □ Для начала воспользуемся псевдообратной матрицей, учитывая, что A имеет полный ранг:

$$x = A^\dagger b = (A^*A)^{-1}A^*b.$$

Дальше остается подставить разложение и воспользоваться унитарностью Q

$$x = (R^*Q^*QR)^{-1}R^*Q^*b = R^{-1}R^{-*}R^*Q^*b = R^{-1}Q^*b.$$

■

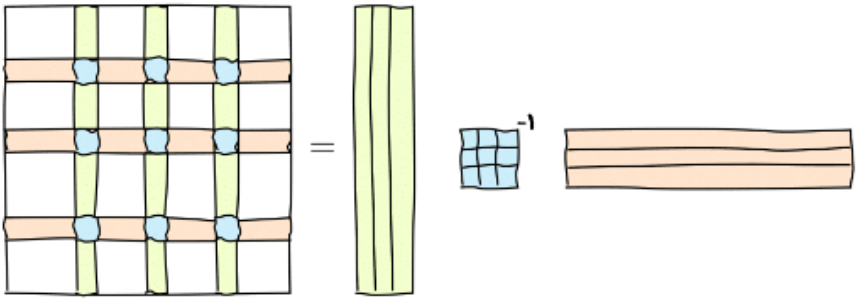
Как упоминалось ранее, СЛУ с треугольной матрицей можно решить за $O(n^2)$. Кроме того, такой подход более устойчивый, чем прямое использование пседообратной матрицы.

2.2.7. Малоранговое приближение

Рассмотрим $A \in \mathbb{R}^{m \times k} : Rg(A) = r$. Определим три матрицы:

- $C \in \mathbb{R}^{m \times r}$ - матрица, образованная некоторыми r столбцами A,
- $R \in \mathbb{R}^{r \times k}$ - матрица, образованная некоторыми r строками A,
- $\widehat{A}$ - невырожденная подматрица, образованная пересечением упомянутых выше строк и столбов.

Подматрица невырожденная, поэтому столбцы C линейно независимы. Выберем произвольный столбец a_i из A. Из сказанного выше очевидно, что $a_i = Cx$. Это m уравнений. Из них выберем r соответствующих строкам $\widehat{A}$. Получим $\widehat{r} = \widehat{A}x \Rightarrow x = \widehat{A}^{-1}\widehat{r}$. То есть $a_i = C\widehat{A}^{-1}\widehat{r}$.
Таким образом, получили $A = C\widehat{A}^{-1}R$. Такое разложение матрицы A называется **скелетным разложением**. Оно объясняет, почему и как матрицы малого ранга могут быть сжаты.



Определим $U = C\widehat{A}^{-1}$, $V = R$.

Замечание 2.26. $A = UV$ - стандартная форма записи скелетного разложения.

Таким образом, для матрицы ранга r можно выделить несколько хороших свойств:

- Достаточно хранить $(m+k)r$ ее элементов,
- Умножение $y = Vx$ и $y = Ux$ стоят $O(kr)$ и $O(mr)$ операций соответственно.

Последнее свойство можно использовать для вычисления результатов применения других операций

2.2.8. Сингулярное разложение

Существует множество способов найти малоранговое приближение заданной матрицы (например, PCA). Эта задача может быть решена с помощью сингулярного разложения.

Теорема 2.27. Любая матрица $A \in \mathbb{C}^{m\times k}$ представима в виде $A = U\Sigma V^*$, где U, V - унитарные, Σ - диагональная. При этом диагональные элементы Σ после номера $Rg(A)$ нулевые.

Доказательство. $\square$ A^*A неотрицательно определена, поэтому ее собственные числа неотрицательны. Кроме того она эрмитова. Отсюда следует существование унитарной матрицы $V : V^*A^*AV = diag(\sigma_1^2, ..., \sigma_m^2)$, где $\sigma_1^2 \geq ... \geq \sigma_m^2 \geq 0$.
Пусть $\sigma_i = 0, i > r, r \in \mathbb{Z}$. Тогда обозначим $V_r = (v_1, ..., v_r), \Sigma_r = diag(\sigma_1, ..., \sigma_r)$. Тогда имеем $V_r^*A^*AV_r = \Sigma_r^2 \Rightarrow (\Sigma_r^{-1}V_r^*A^*)(AV_r\Sigma_r^{-1}) = I$
Обозначим $U_r = AV_r\Sigma_r^{-1}$. Из написанного выше следует $U_r^*U_r = I$, то есть столбцы этой матрицы ортогональны. Присоединим к ней произвольные ортогональные столбцы, которые также ортогональны к уже имеющимся, и получим новую матрицу U. Теперь мы можем написать:

$$AV = U\left(\begin{array}{c|c} \Sigma_r & 0 \\ \hline 0 & 0 \end{array}\right)$$

Действительно, разложили матрицу A. Стоит отметить, что умножение на невырожденные матрицы U^*, V не меняет ранга, а потому $Rg(A) = r$. ■

2.2.9. Вычисление псевдообратной матрицы

Предложение 2.28. Пусть $A = U\Sigma V^*$ - SVD-разложение матрицы A. Тогда $A^\dagger = U\Sigma^\dagger V^*$.

Доказательство. □

$$\begin{aligned} A^\dagger &= \lim_{\alpha\to 0}(\alpha VV^* + V\Sigma^2V^*)^{-1}V\Sigma U^* \\ &= \lim_{\alpha\to 0}(V(\alpha I + \Sigma^2)V^*)^{-1}V\Sigma U^* \\ &= V \lim_{\alpha\to 0}(\alpha I + \Sigma^2)\Sigma U^* \\ &= V\Sigma^\dagger U^* \end{aligned}$$

■

Замечание 2.29. $\Sigma^\dagger$ состоит из обращенных ненулевых собственных чисел.

3. Семинар 1. Матрично-векторное дифференцирование. Теория

Никита Корнилов

3.1. Определения и полезные факты

3.1.1. Первая производная

Пусть U, V - конечномерные ЛНП. Рассмотрим отображение

$$f : X \to V, X \subset U.$$

Определение 3.1. f дифференцируемо в $x \in intX$, если

$$\exists L : U \to V : f(x+h) = f(x) + Lh + o(\|h\|), \|h\| \to 0.$$

Линейный оператор L называется производной f в точке x.

Замечание 3.2. Выбор нормы в определении 3.1 не имеет значения в силу топологической эквивалентности всех норм в конечномерных пространствах.

Замечание 3.3. Из определения 3.1 очевидно следует, что производная единственна, если определена. В дальнейшем условимся обозначать ее $df(x) \equiv Df(x)[h]$ - зависит от точки и приращения. Таким образом, дифференциал f - отображение $df : X \times U \to V$, линейное по второму аргументу.

Еще одним важным понятием является производная функции по направлению.

Определение 3.4. Производной по направлению называется

$$\frac{\partial f}{\partial h} \coloneqq \lim_{t \to +0} \frac{f(x+th) - f(x)}{t}.$$

Замечание 3.5. Подставив $h = th$ в определение 3.1 и воспользовавшись линейностью оператора производной, получим $Df(x)[h] = \frac{\partial f}{\partial h}$. Таким образом, производная по направлению - есть результат применения оператора производной к направлению.

Замечание 3.6. В случае $\mathbb{R}^m$, можем рассматривать производные по стандартному базису. Такие производные по направлениям называются частными.

Производную по направлению можно использовать для проверки на дифференцируемость.

Пример 3.7. Пусть $f(x) = \|x\|_2$.

$$Df(0)[h] = \frac{\partial f}{\partial h}(0) = \lim_{t \to +0} \frac{t\|h\|_2}{t} = \|h\|_2,$$

но вторая норма нелинейная, что противоречит определению оператора производной. Таким образом, вторая норма не дифференцируема в нуле.

3.1.2. Градиент, матрица Якоби

В этом параграфе мы введём основные понятия для производных в стандартных пространствах U, V.

- В случае $f : \mathbb{R}^n \to \mathbb{R}$ линейную функцию $Df(x)[h]$ можно представить в виде

  $$Df(x)[h] = \langle a_x, h \rangle, \quad \text{где } a_x \in \mathbb{R}^n \text{ зависит от } x.$$

  Вектор a_x называется `градиентом` $f(x)$ в точке x и обозначается $\nabla f(x)$.

  Подставив в качестве направлений $h = e_i$, получим из замечания 3.5 явное значение градиента в стандартном базисе со стандартным скалярным произведением

  $$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x) \right)^\top \in \mathbb{R}^n. \tag{1}$$

- В случае $f : \mathbb{R}^{n \times m} \to \mathbb{R}$ линейную функцию $Df(X)[H]$ можно представить в виде

  $$Df(X)[H] = \langle A_X, H \rangle, \quad \text{где } A_X \in \mathbb{R}^{n \times m} \text{ зависит от } X.$$

  Матрица A_x также называется `градиентом` $f(X)$ в точке X и обозначается $\nabla f(X)$.

  Аналогично подставив в качестве направлений $h = e_{ij}$, получим явное значение матрицы градиента в стандартном базисе со стандартным скалярным произведением

  $$\nabla f(X) = \left(\frac{\partial f}{\partial x_{ij}}(X) \right)_{i,j} \in \mathbb{R}^{n \times m}. \tag{2}$$

- В случае $f : \mathbb{R}^n \to \mathbb{R}^m$ линейный оператор $Df(x)[h]$ можно представить в виде

$$Df(x)[h] = J_f(x)h, \quad \text{где } J_f(x) \in \mathbb{R}^{n\times m} \text{ зависит от } x.$$

Матрица $J_x(x)$ называется `матрицей Якоби` $f(x)$ в точке x.

Аналогично подставив в качестве направлений $h = e_i$, получим явное значение матрицы Якоби в стандартном базисе со стандартным скалярным произведением

$$J_f(x) \equiv \frac{\partial f}{\partial x} := \left(\frac{\partial f_i}{\partial x_j}(x)\right)_{i,j} \in \mathbb{R}^{n\times m}. \tag{3}$$

- Во всех остальных случаях для построения производной достаточно найти все частные производные в виде тензора

$$\frac{\partial f_{ij}}{\partial x_{kl}}(x).$$

Следует помнить, что из существования частных производных ещё не следует дифференцируемость. Однако на практике чаще всего все эти частные производные непрерывны, и, как следствие, функция дифференцируема.

Финальная таблица с каноническими видами

Вход \ Выход	Скаляр $\mathbb{R}$	Вектор $\mathbb{R}^n$
Скаляр $\mathbb{R}$	$df(x) = f'(x)dx$ $f'(x)$ скаляр, dx скаляр.	-
Вектор $\mathbb{R}^m$	$df(x) = \langle \nabla f(x), dx\rangle$ $f(x)$ вектор, dx вектор	$df(x) = J_x dx$ J_x матрица, dx вектор
Матрица $\mathbb{R}^{n'\times m'}$	$df(X) = \langle \nabla f(X), dX\rangle$ $\nabla f(X)$ матрица, dX матрица	-

Стоит отметить, что данная таблица верна и для произвольных скалярных произведений, а не только для стандартного.

3.1.3. Вторая производная

Определение 3.8. Фиксируем приращение $h_1 \in U$ и рассмотрим $g(x) = Df(x)[h_1]$. Второй производной f в точке $x \in X$ называется $D^2f(x)[h_1, h_2] \coloneqq Dg(x)[h_2]$ - билинейная функция по h_1 и h_2.

Замечание 3.9. Вернемся к случаю $U = \mathbb{R}^m, V = R$. Теперь

$$D^2 f(x)[h] = \langle \nabla^2 f(x) h_1, h_2 \rangle$$

$\nabla^2 f(x)$ называется гессианом функции f.

В стандартном базисе гессиан имеет вид

$$\left(\nabla^2 f(x)\right)_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}(x).$$

Напомним, что для дважды непрерывно дифференцируемой функции гессиан - симметричная матрица.

Предложение 3.10. Из формулы градиента верно, что

$$d(\nabla f(x)) = (\nabla^2 f)^\top dx \Leftrightarrow \nabla^2 f(x) = (J_{\nabla f})^\top.$$

3.1.4. Теория дифференциалов

Напомним некоторые важные факты из курса математического анализа

Предложение 3.11. Пусть U, V — линейные пространства, $X \subset U, x \in X$ - внутренняя точка. Тогда справедливы следующие свойства:

- Линейность. Пусть $f : X \to V$ и $g : X \to V$. Если f, g дифференцируемы в точке x, при этом $c_1, c_2 \in \mathbb{R}$ числа, то $c_1 f + c_2 g$ дифференцируема в x и

$$d(c_1 f + c_2 g) = c_1 df + c_2 dg. \tag{4}$$

- Правило произведения. Пусть $\alpha : X \to \mathbb{R}$ и $f : X \to V$ функции. Если α, f дифференцируемы в точке x, то αf дифференцируема в точке x и

$$D(\alpha f)(x)[h] = (D\alpha(x)[h]) f(x) + \alpha(x)(Df(x)[h]) \tag{5}$$

  для любых приращений h.

- Правило композиции. Пусть Y - подмножество V, $f : X \to Y$ - функция. Также пусть W линейное пространство, $g : Y \to W$ - функция. Если f дифференцируема в точке x, g

дифференцируема в точке $f(x)$, то их композиция $(g\circ f)(x) \equiv g(f(x))$ дифференцируема в точке x и

$$D(g \circ f)(x) = Dg(f(x))[df] \Longleftrightarrow Dg(f(x))[Df(x)[h]]. \quad (6)$$

- Правило частного. Пусть $\alpha : X \to \mathbb{R}$ и $f : X \to V$ - функции. Если α, f дифференцируемы в x и не обращается в 0 на X, то $(1/\alpha)f$ дифференцируема в x и

$$D\left(\frac{f}{\alpha}\right)(x)[h] = \frac{\alpha(x)(Df(x)[h]) - (D\alpha(x)[h])f(x)}{\alpha(x)^2}. \quad (7)$$

- Правило произведения для матрично-значных функций. Пусть $f : X \to \mathbb{R}^{m\times n}$ и $g : X \to \mathbb{R}^{n\times k}$ - матрично-значные функции. Если f, g дифференцируемы в точке x, то fg дифференцируема в x и

$$D(\alpha f)(x)[h] = (D\alpha(x)[h])f(x) + \alpha(x)(Df(x)[h]). \quad (8)$$

Здесь подразумевается матричное умножение.

Следствие 3.12. • Для дифференцируемых в точке x векторнозначных функций $f : X \to \mathbb{R}^n$ и $g : X \to \mathbb{R}^n$ функция $\langle f, g\rangle$ дифференцируема в x и

$$d(\langle f, g\rangle) = \langle df, g\rangle + \langle f, dg\rangle. \quad (9)$$

- Для дифференцируемой в точке x векторно-значной функции $f : X \to \mathbb{R}^n$ и линейного отображения $L : \mathbb{R}^n \to \mathbb{R}^m$ дифференциал и L перестановочны:

$$D(L \circ f)(x)[h] = L[Df(x)[h]]. \quad (10)$$

- Матрица Якоби сложной функции $f(g(x))$ равна произведению матриц Якоби композитов

$$J_{g(f(x))} = J_g J_f.$$

3.2. Примеры и задачи

3.2.1. Вычисление по определению

Пример 3.13. Табличные производные.

а) Для $f(x) = \langle c, x\rangle, x \in \mathbb{R}^n$ и приращения h считаем

$$\begin{aligned} f(x+h) - f(x) =& \langle c, x+h\rangle - \langle c, x\rangle \\ =& \langle c, h\rangle. \end{aligned}$$

Отображение $h \to \langle c, h\rangle$ является линейным, поэтому его можно принять за производную по определению

$$Df(x)[h] = \langle c, h\rangle.$$

б) Для $f(x) = \langle Ax, x\rangle, x \in \mathbb{R}^n, A \in \mathbb{R}^{n\times n}$ и приращения h считаем

$$\begin{aligned} f(x+h) - f(x) =& \langle Ax + Ah, x+h\rangle - \langle Ax, x\rangle \\ =& \langle (A + A^\top)x, h\rangle + \langle Ah, h\rangle. \end{aligned}$$

Заметим, что

$$\|Ah, h\| \le \|Ah\|\|h\| \le \|A\|\|h\|^2 = o(\|h\|),$$

где первое неравенство следует из Коши-Буняковского, а второе из согласованности матричной нормы. При этом $h \to \langle (A + A^\top)x, h\rangle$ линейный оператор.

Получается, что по определению

$$Df(x)[h] = \langle (A + A^\top)x, h\rangle.$$

в) Пусть $S := \{X \in \mathbb{R}^{n\times n} : \det(X) \neq 0\}$ и функция $f : S \to S$ обращает матрицу $f(X) = X^{-1}$. Для произвольного малого приращения H посчитаем

$$\begin{aligned} f(X+H) - f(X) =& (X+H)^{-1} - X^{-1} \\ =& (X(I_n + X^{-1}H))^{-1} - X^{-1} \\ =& ((I_n + X^{-1}H)^{-1} - I_n)X^{-1}. \end{aligned}$$

Отдельно оценим $(I_n + X^{-1}H)^{-1}$, для чего вспомним ряд Неймана. В нашем случае мы можем применить ряд Неймана в силу малости H

$$(I_n + X^{-1}H)^{-1} = I_n - X^{-1}H + \sum_{k=2}^{\infty}(-X^{-1}H)^k.$$

Оценим норму последнего слагаемого

$$\begin{aligned}\left\|\sum_{k=2}^{\infty}(-X^{-1}H)^k\right\| \leq &\sum_{k=2}^{\infty}\|(-X^{-1}H)^k\| \\ \leq &\sum_{k=2}^{\infty}\|-X^{-1}\|^k\|H\|^k \\ = &\frac{\|X^{-1}\|^2\|H\|^2}{1-\|X^{-1}\|\|H\|} \\ = &o(\|H\|),\end{aligned}$$

где первое неравенство получается из неравенства треугольника в пределе, второе – из свойств матричной нормы, а третье равенство – это сумма геометрической прогрессии.

В итоге получаем разность

$$f(X+H)-f(X) = -X^{-1}HX^{-1}+o(\|H\|),$$

при этом отображение $H \to -X^{-1}HX^{-1}$ линейно. То есть по определению

$$Df(X)[H] = -X^{-1}HX^{-1}.$$

г) Пусть $S := \{X \in \mathbb{R}^{n\times n} : \det(X) \neq 0\}$ и функция $f : S \to \mathbb{R}$ считает определитель $f(X) = \det(X)$.

Для малого приращения H посчитаем приращение функции

$$\begin{aligned}f(X+H)-f(X) = &\det(X+H)-\det(X) \\ = &\det(X(I_n+X^{-1}H))-\det(X) \\ = &\det(X)(\det(I_n+X^{-1}H)-1).\end{aligned}$$

Отдельно оценим $\det(I_n+X^{-1}H)$. Пусть $\lambda_i(X^{-1}H)$ – собственные числа матрицы $X^{-1}H$ (в произвольном порядке и, возможно, комплексные). Тогда собственными числами матрицы $I_n + X^{-1}H$ будут $1 + \lambda_i(X^{-1}H)$. Поэтому

$$\begin{aligned}\det(I_n+X^{-1}H) = &\prod_{i=1}^{n}[1+\lambda_i(X^{-1}H)] \\ = &1+\sum_{i=1}^{n}\lambda_i(X^{-1}H)+\left(\sum_{1\leq i\leq j\leq n}\lambda_i(X^{-1}H)\lambda_j(X^{-1}H)+\dots\right),\end{aligned}$$

где ... обозначает всевозможные тройки, четверки и т.д. из $\lambda_i(X^{-1}H)$.

Для произвольной матрицы $A \in \mathbb{R}^{n\times n}$ все её собственные числа по модулю не превосходят её нормы. Действительно, для собственного числа $\lambda \in \mathbb{C}$ и единичного по норме собственного вектора $x \in \mathbb{R}^n$ верно

$$Ax = \lambda x \Rightarrow \|\lambda x\| = \|Ax\| \leq \|A\|\|x\|.$$

Следовательно, всё выражение в скобках будет $o(\|H\|)$, поскольку в каждом слагаемом больше одного собственного числа. Таким образом,

$$\begin{aligned}\det(I_n + X^{-1}H) =& 1 + \sum_{i=1}^{n} \lambda_i(X^{-1}H) + o(\|H\|)\\ =& 1 + \operatorname{Tr}(X^{-1}H) + o(\|H\|).\end{aligned}$$

Подставив это выражение для приращения функции, получим

$$f(X+H) - f(X) = \det(X)\operatorname{Tr}(X^{-1}H) + o(\|H\|).$$

Далее мы будем работать со стандартным скалярным произведением. Мы доказали формулу $d(\det(X)) = \det(X)\langle X^{-\top}, dX\rangle$ только для обратимых X. Однако формулу для дифференциала $d(\det(X))$ можно получить и для произвольной матрицы из $\mathbb{R}^{n\times n}$. Эта формула называется `формулой Якоби` и записывается как

$$d(\det(X)) = \langle \operatorname{Adj}(X)^{\top}, dx\rangle, \tag{11}$$

где $\operatorname{Adj}(X)$ - присоединённая матрица к X.

Присоединённая матрица определяется как $\operatorname{Adj}(X)_{ji} = (-1)^{(i+j)}M_{ij}$, где M_{ij} - дополнительный минор, определитель матрицы, получившийся вычеркиванием i-ой строки и j-ого столбца из X. В случае невырожденной X выполнено $\operatorname{Adj}(X) = \det(X)X^{-1}$ и формулы переходят друг в друга.

Вспомним, формулу вычисления определителя через дополнительные миноры по i-ой строке

$$\det(X) = \sum_k x_{ik}\cdot(-1)^{(i+k)}M_{ik}.$$

Тогда градиент равен

$$\frac{\partial f}{\partial x_{ij}} = (-1)^{(i+j)}M_{ij} = \operatorname{Adj}(X)_{ji}.$$

Финальная табличка с правилами преобразования и табличными значениями выглядит так

Правила преобразования
$d(\alpha X) = \alpha dX$
$d(AXB) = AdXB$
$d(X+Y) = dX + dY$
$d(X^T) = (dX)^T$
$d(XY) = (dX)Y + X(dY)$
$d\langle X, Y\rangle = \langle dX, Y\rangle + \langle X, dY\rangle$
$d\left(\frac{X}{\phi}\right) = \frac{\phi dX - (d\phi)X}{\phi^2}$
$d(g(f(x))) = g'(f)df(x)$
$J_{g(f)} = J_g J_f \iff \frac{\partial g}{\partial x} = \frac{\partial g}{\partial f}\frac{\partial f}{\partial x}$

Таблица стандартных производных
$dA = 0$
$\langle A, X\rangle = \langle A, dX\rangle$
$d\langle Ax, x\rangle = \langle (A + A^\top)x, dx\rangle$
$d\,\mathrm{Tr}(X) = \mathrm{Tr}(dX)$
$d(\det(X)) = \det(X)\,\mathrm{Tr}(X^{-1}dX)$
$d(X^{-1}) = -X^{-1}(dX)X^{-1}$

Стоит отметить, что данные таблицы верны и для произвольных скалярных произведений, а не только для стандартного.

Hint. Для запоминания формулы $d(X^{-1})$ через произведение X и X^{-1}

$$
\begin{aligned}
&I = XX^{-1},\\
&dI = 0 = d(XX^{-1}) = (dX)X^{-1} + Xd(X^{-1}),\\
&d(X^{-1}) = -X^{-1}(dX)X^{-1}.
\end{aligned}
$$

Однако это не является доказательством `существования` дифференциала.

3.2.2. Дифференцирование по вектору

Для начала попрактикуемся в подсчёте градиентов и вторых производных для функций вида $f(x) : \mathbb{R}^n \to \mathbb{R}$.

Начнём с самой простой и часто встречаемой квадратичной функции. Попробуем посчитать её производные прямым и дифференциальным метод и сравним их.

Пример 3.14. Квадратичная функция. Найдите первый и второй дифференциал $df(x), d^2f(x)$, а также градиент $\nabla f(x)$ и гессиан $\nabla^2 f(x)$ функции

$$f(x) = \frac{1}{2}\langle Ax, x\rangle + \langle b, x\rangle + c,$$

где $A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n, c \in \mathbb{R}$.

Решение. Попробуем применить оба подхода для решения данной задачи.

- Сначала используем прямой метод и выпишем явную скалярную зависимость $f(x_1, \dots, x_n)$

$$\begin{aligned} f(x_1, \dots, x_n) =& \frac{1}{2} \sum_{i=1}^{n} x_i \sum_{j=1}^{n} A_{ij} x_j + \sum_{i=1}^{n} x_i b_i + c \\ =& \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j + \sum_{i=1}^{n} x_i b_i + c. \end{aligned}$$

Найдём частную производную по x_k

$$\begin{aligned} f(x_1, \dots, x_n) =& \frac{1}{2} A_{kk} x_k^2 + \frac{1}{2} \sum_{i \neq k} A_{ik} x_i x_k + \frac{1}{2} \sum_{j \neq k} A_{kj} x_k x_j \\ &+ x_k b_k + \left(\frac{1}{2} \sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j + \sum_{i \neq k} x_i b_i + c \right). \end{aligned}$$

Взяв частную производную, получим

$$\begin{aligned} \frac{\partial f}{\partial x_k} =& \frac{1}{2} \cdot 2 A_{kk} x_k + \frac{1}{2} \sum_{i \neq k} A_{ik} x_i + \frac{1}{2} \sum_{j \neq k} A_{kj} x_j + b_k \\ =& \frac{1}{2} (Ax)_k + \frac{1}{2} (A^\top x)_k + b_k. \end{aligned}$$

Подставив координатно, посчитаем градиент

$$\nabla f(x) = \frac{1}{2} (A + A^\top) x + b.$$

Для подсчёта гессиана найдём двойную частную производную по x_k, x_l

$$\begin{aligned} \frac{\partial^2 f}{\partial x_l \partial x_k} =& \frac{\partial \frac{1}{2} \sum\limits_{i=1}^{n} A_{ik} x_i + \frac{1}{2} \sum\limits_{j=1}^{n} A_{kj} x_j + b_k}{\partial x_l} \\ =& \frac{1}{2} A_{lk} + \frac{1}{2} A_{kl} \\ =& \frac{1}{2} (A + A^\top)_{kl}. \end{aligned}$$

Следовательно, гессиан равен

$$\nabla^2 f(x) = \frac{1}{2} (A + A^\top).$$

- Теперь используем дифференциальное исчисление

$$\begin{aligned} df(x) = & \quad d\left(\tfrac{1}{2}\langle Ax, x\rangle + \langle b, x\rangle + c\right) \\ = & \quad \tfrac{1}{2}\langle (A + A^\top)x, dx\rangle + \langle b, dx\rangle + 0 \\ = & \quad \left\langle \tfrac{1}{2}(A + A^\top)x + b, dx\right\rangle. \end{aligned}$$

Следовательно, приведя к стандартному виду $df = \langle \nabla f(x), dx\rangle$, получаем градиент

$$\nabla f(x) = \frac{1}{2}(A + A^\top)x + b.$$

Далее для гессиана фиксируем первое приращение dx_1 у первого дифференциала и берём уже от него ещё один дифференциал

$$\begin{aligned} d^2 f =& d(df) \\ =& d\left\langle \frac{1}{2}(A + A^\top)x + b, dx_1\right\rangle \\ =& \left\langle d\left(\frac{1}{2}(A + A^\top)x + b\right), dx_1\right\rangle + \left\langle \frac{1}{2}(A + A^\top)x + b, \underbrace{d(dx_1)}_{=0}\right\rangle \\ =& \left\langle \frac{1}{2}(A + A^\top)dx, dx_1\right\rangle. \end{aligned}$$

Переносим и транспонируем матрицу в скалярном произведении, но поскольку $A + A^\top$ симметричная, она не меняется.

$$d^2 f = \left\langle dx, \frac{1}{2}(A + A^\top)^\top dx_1\right\rangle = \left\langle \frac{1}{2}(A + A^\top)dx_1, dx\right\rangle.$$

Следовательно, приведя к стандартному виду $d^2 f = \langle \nabla^2 f(x) \cdot dx_1, dx\rangle$, получаем гессиан

$$\nabla^2 f(x) = \frac{1}{2}(A + A^\top).$$

Заметим, что в случае если A симметричная, то

$$\nabla f(x) = Ax + b, \quad \nabla^2 f(x) = A.$$

С помощью примера 3.14 можно найти также градиент функции невязки $f(x) = \frac{1}{2}\|Ax - b\|^2, \quad x \in \mathbb{R}^n$.

Минимум квадрата невязки позволяет найти решение системы линейных уравнений с минимальной ошибкой по норме.

В Машинном обучении этот метод известен под названием Метод Наименьших Квадратов(МНК). По матрице признаков A и вектору параметров x мы будем пытаться линейно приближать целевой вектор b.

Далее посмотрим на пример посложнее на применение правила дифференцирования сложной функции.

Пример 3.15. Найдите первый и второй дифференциал $df(x), d^2f(x)$, а также градиент $\nabla f(x)$ и гессиан $\nabla^2 f(x)$ функции

$$f(x) = \ln\langle Ax, x\rangle$$

где $x \in \mathbb{R}^n, A \in \mathbb{S}^n_{++}$.

Решение. Найдём первый дифференциал

$$df = d\ln\langle Ax, x\rangle = \frac{1}{\langle Ax, x\rangle} d\langle Ax, x\rangle = \frac{2\langle Ax, dx\rangle}{\langle Ax, x\rangle} = \left\langle \frac{2Ax}{\langle Ax, x\rangle}, dx \right\rangle.$$

Теперь найдём дифференциал градиента

$$\begin{aligned} d\left(\frac{2Ax}{\langle Ax, x\rangle}\right) &= \frac{d(2Ax)\langle Ax, x\rangle - (2Ax)d\langle Ax, x\rangle}{\langle Ax, x\rangle^2} \\ &= \frac{2\langle Ax, x\rangle Adx - 4Ax\langle Ax, dx\rangle}{\langle Ax, x\rangle^2} \\ &= \left(\frac{2A}{\langle Ax, x\rangle} - \frac{4Axx^\top A}{\langle Ax, x\rangle^2}\right) dx = J_{\nabla f} dx. \end{aligned}$$

Поскольку $\nabla^2 f = (J_{\nabla f})^\top$, а гессиан симметричен из-за непрерывности, то

$$\nabla^2 f = \frac{2A}{\langle Ax, x\rangle} - \frac{4Axx^\top A}{\langle Ax, x\rangle^2}.$$

Пример 3.16. Евклидова норма. Найдите первый и второй дифференциал $df(x), d^2f(x)$, а также градиент $\nabla f(x)$ и гессиан $\nabla^2 f(x)$ функции

$$f(x) = \|x\|_2, \quad x \in \mathbb{R}^n \setminus \{0\}.$$

Решение. Найдём первый дифференциал

$$\begin{aligned} df(x) =& d(\langle x, x\rangle^{\frac{1}{2}}) \\ =& \left\{ dy^{\frac{1}{2}} = \frac{1}{2y^{\frac{1}{2}}} dy \right\} \\ =& \frac{d(\langle x, x\rangle)}{2\langle x, x\rangle^{\frac{1}{2}}} \\ =& \left\langle \frac{2x}{2\langle x, x\rangle^{\frac{1}{2}}}, dx \right\rangle \\ =& \left\langle \frac{x}{\|x\|}, dx \right\rangle. \end{aligned}$$

После этого приведём df к стандартному виду $df = \langle \nabla f, dx\rangle$ и получим градиент

$$\nabla f(x) = \frac{x}{\|x\|}.$$

Теперь посчитаем второй дифференциал, зафиксировав приращение dx_1 первого

$$\begin{aligned} df^2(x) =& d\left(\left\langle \frac{x}{\|x\|}, dx_1 \right\rangle \right) \\ =& \left\langle d\left(\frac{x}{\|x\|} \right), dx_1 \right\rangle = \textit{Правило частного} \\ =& \left\langle \frac{dx\|x\| - xd(\|x\|)}{\|x\|^2}, dx_1 \right\rangle \\ =& \left\langle \frac{dx\|x\| - x\left\langle \frac{x}{\|x\|}, dx \right\rangle}{\|x\|^2}, dx_1 \right\rangle \\ =& \left\langle \frac{I_n\|x\| - \frac{xx^\top}{\|x\|}}{\|x\|^2} dx, dx_1 \right\rangle \\ =& \left\langle \left(\frac{I_n\|x\| - \frac{xx^\top}{\|x\|}}{\|x\|^2} \right)^\top dx_1, dx \right\rangle. \end{aligned}$$

Представив $d^2 f$ в стандартной форме $\langle \nabla^2 f(x) \cdot dx_1, dx\rangle$, получим

$$\nabla^2 f(x) = \frac{I_n}{\|x\|} - \frac{xx^\top}{\|x\|^3}.$$

Заметим, что в точке $x = 0$ функция не является дифференцируемой. НО при этом мы можем посчитать производную по любому направлению h:

$$\frac{\partial f}{\partial h}(0) = \lim_{t \to 0} \frac{f(0 + th) - f(0)}{t} = \lim_{t \to +0} \frac{||th||}{t} = ||h||.$$

Если бы функция была дифференцируем, то

$$df(x)[h] = ||h||,$$

а это НЕлинейная функция от h.

Пример 3.17. Куб Нормы. Найдите первый и второй дифференциал $df(x), d^2f(x)$, а также градиент $\nabla f(x)$ и гессиан $\nabla^2 f(x)$ функции

$$f(x) = \frac{1}{3}\|x\|_2^3, \quad x \in \mathbb{R}^n.$$

Решение. Найдём первый дифференциал

$$\begin{aligned} df(x) =& \frac{1}{3} d\langle x, x\rangle^{3/2} \\ =& \frac{1}{3} \cdot \frac{3}{2} \langle x, x\rangle^{1/2} d\langle x, x\rangle \\ =& \frac{1}{2} \langle x, x\rangle^{1/2} \cdot 2\langle x, dx\rangle \\ =& \langle x\|x\|, dx\rangle. \end{aligned}$$

Приведя к стандартному виду $df = \langle \nabla f(x), dx\rangle$, получаем

$$\nabla f(x) = \|x\| x.$$

Найдём второй дифференциал

$$\begin{aligned} d^2 f(x) =& d(\|x\|\langle x, dx_1\rangle) \\ =& \underbrace{d(\|x\|)}_{= d\langle x,x\rangle^{1/2}} \langle x, dx_1\rangle + \|x\| d(\langle x, dx_1\rangle) \\ =& \left(\frac{1}{2}\langle x, x\rangle^{-1/2} 2\langle x, dx\rangle\right) \langle x, dx_1\rangle + \|x\|\langle dx, dx_1\rangle \\ =& \frac{1}{\|x\|}\langle x, dx\rangle\langle x, dx_1\rangle + \|x\|\langle dx, dx_1\rangle \\ =& \left\langle dx, \left(\frac{xx^\top}{\|x\|} + I_n\|x\|\right) dx_1 \right\rangle. \end{aligned}$$

Представив $d^2 f$ в стандартной форме $\langle \nabla^2 f(x) \cdot dx_1, dx \rangle$, получим

$$\nabla^2 f(x) = \frac{xx^\top}{\|x\|} + I_n \|x\|$$

Заметим, что в данная формула не определена в точке $x = 0$, поскольку ранее мы пользовались правилом дифференцирования функции $\sqrt{x}$, а её производная не определена в 0.

Однако можно найти вторую производную в $x = 0$ по определению. Зафиксируем приращение h_1 и рассмотрим

$$\begin{aligned}
&\lim_{h_2 \to 0} \frac{\|(Df[h_1])(0+h_2) - (Df[h_1])(0)\|}{\|h_2\|} \\
&= \lim_{h_2 \to 0} \frac{|\langle (0+h_2)\|0+h_2\| - 0\|0\|, h_1 \rangle|}{\|h_2\|} \\
&= \lim_{h_2 \to 0} \frac{\|h_2\| |\langle h_2, h_1 \rangle|}{\|h_2\|} \\
&= \lim_{h_2 \to 0} |\langle h_2, h_1 \rangle| = 0.
\end{aligned}$$

Следовательно, по определению вторая производная в точке $x = 0$ равна 0. Можно даже сказать, что функция дважды непрерывно дифференцируема, потому что $\lim\limits_{x \to 0} \left(\frac{xx^\top}{\|x\|} + I_n \|x\| \right) = 0$.

Рассмотрим часто встречающую в Deep Learning и нейронных сетях функцию softmax, которая позволяет вектор из n координат в распределение вероятностей на n исходах. Например, многоклассовой классификации тем самым мы получаем вектор вероятностей принадлежности объекта каждому из n классов.

Пример 3.18. Softmax. Найдите матрицу Якоби функции $s(x) = \mathrm{softmax}(x)$

$$\mathrm{softmax}(x) := \left(\frac{\exp(x_1)}{\sum_{i=1}^n \exp(x_i)}, \ldots, \frac{\exp(x_n)}{\sum_{i=1}^n \exp(x_i)} \right)^\top.$$

Решение. Считаем частные производные по определению

а) при $k \neq j$

$$\begin{aligned}\frac{\partial s_k}{\partial x_j} =& \frac{\partial}{\partial x_j}\frac{\exp(x_k)}{\sum_{i=1}^n \exp(x_i)}\\ =& \exp(x_k)\frac{\partial}{\partial x_j}\frac{1}{\sum_{i=1}^n \exp(x_i)}\\ =& \exp(x_k)\frac{-1}{(\sum_{i=1}^n \exp(x_i))^2}\frac{\partial}{\partial x_j}\left(\sum_{i=1}^n \exp(x_i)\right)\\ =& -\frac{\exp(x_k)\exp(x_j)}{(\sum_{i=1}^n \exp(x_i))^2}\\ =& -s_k \cdot s_j,\end{aligned}$$

б) при $k = j$

$$\begin{aligned}\frac{\partial s_j}{\partial x_j} =& \frac{\partial}{\partial x_j}\frac{\exp(x_j)}{\sum_{i=1}^n \exp(x_i)}\\ =& \frac{\exp(x_j)(\sum_{i=1}^n \exp(x_i)) - \exp(x_j)\frac{\partial}{\partial x_j}(\sum_{i=1}^n \exp(x_i))}{(\sum_{i=1}^n \exp(x_i))^2}\\ =& \frac{\exp(x_j)}{\sum_{i=1}^n \exp(x_i)} - \frac{\exp(x_j)\exp(x_j)}{(\sum_{i=1}^n \exp(x_i))^2}\\ =& s_j(1-s_j).\end{aligned}$$

Итого,

$$J_{k,j} = \begin{cases} -s_k \cdot s_j, & k \neq j \\ s_j(1-s_j), & k = j. \end{cases}$$

Не менее часто в DL встречаются покоординатные функции, которые применяются на выходе очередного слоя для каждого отдельного нейрона. Посмотрим, как через них считать градиент.

Пример 3.19. Покоординатные операции. Найдите градиент и гессиан функции $f(x) = h(g(x))$, где

$$g(x) = \sin(x) \text{ поэлементно},$$

$$h(u) = \sum_{i=1}^n u_i.$$

Решение. В этом примере нам в любом случае нужно будет применять именно первый подход для подсчёта матриц Якоби и градиентов. Действительно, входящие функции не являются стандартными,

но являются достаточно легкими, чтобы считать частные производные напрямую.

Также полезно вспомнить правило матрицы Якоби сложной функции

$$J_f = J_{h(g)} J_g,$$

оно же с градиентами имеет вид

$$\nabla f = J_g^\top \nabla h.$$

Далее посчитаем матрицу Якоби покоординатной функции вида $g(x) = \begin{pmatrix} g(x_1) \\ \vdots \\ g(x_n) \end{pmatrix}$

$$J_g = \operatorname{diag}(g'(x_1), \ldots, g'(x_n)) = \operatorname{diag}(g'(x)) = J_g^\top.$$

При умножении J_g на вектор удобно пользоваться поэлементным умножением матриц, обозначающимся $\odot$

$$(A \odot B)_{ij} = A_{ij} * B_{ij}.$$

Результат умножения J_g на вектор y равен

$$J_g y = \begin{pmatrix} g'(x_1) \\ \vdots \\ g'(x_n) \end{pmatrix} \odot y = g'(x) \odot y.$$

Заметим, что эта операция является довольно быстро вычислимой и легко поддаётся параллелизации.

Теперь приступим к непосредственному примеру

$$\begin{aligned} &J_g = \operatorname{diag}(\cos(x_1), \ldots, \cos(x_n)) = \operatorname{diag}(\cos(x)), \\ &\{\nabla h(u)\}_j = \frac{\partial(\sum_{i=1}^n u_i)}{\partial u_j} = 1 \quad \to \quad \nabla h(u) = \mathbf{1}, \\ &\nabla f = J_g^\top \nabla h = \cos(x) \odot \mathbf{1} = \cos(x). \end{aligned}$$

Теперь гессиан функции, который считается по формуле

$$\nabla^2 f(x) = J_{\nabla f}^\top = \operatorname{diag}(-\sin(x)).$$

Логистическая регрессия – модель машинного обучения для двухклассовой классификации. Более подробно про саму модель и интуицию за ней можно почитать здесь. Её обучение может быть сведено к оптимизации функции, представленной в примере ниже.

Пример 3.20. Логистическая регрессия. Найдите первый и второй дифференциал $df(x), d^2f(x)$, а также градиент $\nabla f(x)$ и гессиан $\nabla^2 f(x)$ функции

$$f(x) = \ln(1 + \exp(\langle a, x\rangle)),$$

где $a \in \mathbb{R}^n$.

Решение. Найдём первый дифференциал

$$\begin{aligned}
d(\ln(1 + \exp(\langle a, x\rangle))) =& \{d \ln y = \frac{1}{y} dy\} \\
=& \frac{1}{1 + \exp(\langle a, x\rangle)} d(1 + \exp(\langle a, x\rangle)) \\
=& \{d \exp(y) = \exp(y) dy\} \\
=& \frac{1}{1 + \exp(\langle a, x\rangle)} \exp(\langle a, x\rangle) d(\langle a, x\rangle) \\
=& \left\langle \frac{\exp(\langle a, x\rangle)}{1 + \exp(\langle a, x\rangle)} a, dx \right\rangle.
\end{aligned}$$

Для удобства введём функцию сигмоиды $\sigma(x) := \frac{1}{1+\exp(-x)}$. При этом заметим, что $\sigma(-x) = 1 - \sigma(x)$ и $\sigma'(x) = \sigma(x)(1 - \sigma(x))$. После этого приведём df к стандартному виду $df = \langle \nabla f, dx\rangle$ и получим градиент

$$\nabla f(x) = \sigma(\langle a, x\rangle) a.$$

Таким образом, градиент $\nabla f(x)$ - вектор коллинеарный вектору a с коэффициентом $\sigma(\langle a, x\rangle) \in (0, 1)$. В зависимости от точки x меняет лишь длина градиента, но не направление.

Теперь посчитаем второй дифференциал, зафиксировав приращение dx_1 первого

$$\begin{aligned}
d(df) =& d(\langle \sigma(\langle a, x\rangle) a, dx_1\rangle) \\
=& \langle d(\sigma(\langle a, x\rangle)) a, dx_1\rangle \\
=& \langle \sigma'(\langle a, x\rangle) d(\langle a, x\rangle) a, dx_1\rangle \\
=& \langle \sigma(\langle a, x\rangle)(1 - \sigma(\langle a, x\rangle)) \langle a, dx\rangle a, dx_1\rangle \\
=& \sigma(\langle a, x\rangle)(1 - \sigma(\langle a, x\rangle)) \langle \langle dx, a\rangle a, dx_1\rangle \\
=& \sigma(\langle a, x\rangle)(1 - \sigma(\langle a, x\rangle))(dx^\top a a^\top dx_1) \\
=& \sigma(\langle a, x\rangle)(1 - \sigma(\langle a, x\rangle)) \langle a a^\top dx_1, dx\rangle.
\end{aligned}$$

Представив $d^2 f$ в стандартной форме $\langle \nabla^2 f(x) \cdot dx_1, dx\rangle$, получим

$$\nabla^2 f(x) = \sigma(\langle a, x\rangle)(1 - \sigma(\langle a, x\rangle)) a a^\top.$$

Заметим, что $\nabla^2 f$ - одноранговая матрица, пропорциональная $aa^\top$ с коэффициентом $\sigma(\langle a, x\rangle)(1-\sigma(\langle a, x\rangle)) \in (0, 0.25)$. Точка x влияет лишь на коэффициент.

Полезно понимать, как работать не только с векторным входом, но и со скалярным.

Пример 3.21. Дифференциал скаляра. Рассмотрим функцию скалярного аргумента α

$$\phi(\alpha) := f(x + \alpha p), \quad \alpha \in \mathbb{R},$$

$x, p \in \mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}$ -дважды непрерывно дифференцируемая функция. Найдите первую и вторую производные $\phi'(\alpha), \phi''(\alpha)$ и выразите их через $\nabla f, \nabla^2 f$.

Решение. Важно помнить, что дифференцирование происходит не по стандартному вектору x, а по скаляру α со всеми вытекающими свойствами

$$\begin{aligned} d\phi =& \{df = \langle \nabla f(y), dy\rangle\} \\ =& \langle \nabla f(x + \alpha p), d(x + \alpha p)\rangle \\ =& \langle \nabla f(x + \alpha p), d(\alpha)p\rangle \\ =& \langle \nabla f(x + \alpha p), p\rangle d(\alpha). \end{aligned}$$

Заметим, что мы привили дифференциал к стандартному виду $d\phi = \phi'(\alpha) \cdot d\alpha$, то есть множитель перед $d\alpha$ – это производная

$$\phi'(\alpha) = \langle \nabla f(x + \alpha p), p\rangle.$$

Теперь вторая производная

$$\begin{aligned} d(\phi'(\alpha)) =& d\langle \nabla f(x + \alpha p), p\rangle \\ =& \{d(\nabla f(y)) = (\nabla^2 f(y))^\top dy\} \\ =& \langle (\nabla^2 f(x + \alpha p))^\top d(x + p\alpha), p\rangle \\ =& \langle (\nabla^2 f(x + \alpha p))^\top p d\alpha, p\rangle = \{\nabla^2 f(y) \\ =& (\nabla^2 f(y))^\top\}] \\ =& \langle \nabla^2 f(x + \alpha p)p, p\rangle d\alpha. \end{aligned}$$

Получается, что

$$\phi''(\alpha) = \langle \nabla^2 f(x + \alpha p)p, p\rangle.$$

3.2.3. Дифференцирование по матрице

Далее будем считать градиенты по матрицы для функций вида $f(X) : \mathbb{R}^{n\times m} \to \mathbb{R}$. В них активно применяются производные от таких матричных функций, как $\det, \mathrm{Tr}, X^{-1}$.

Пример 3.22. Фробениусова норма. Найти градиент $\nabla f(X)$ и дифференциал $df(X)$ функции $f(X)$

$$f(X) = \|AX - B\|_F, \quad X \in \mathbb{R}^{k\times n},$$

где $A \in \mathbb{R}^{m\times k}, B \in \mathbb{R}^{m\times n}$.

Решение. Вычислим отдельно $d(\|X\|)$

$$\begin{aligned} d(\|X\|) =& d(\langle X, X\rangle^{\frac{1}{2}}) \\ =& \left\{ dy^{\frac{1}{2}} = \frac{1}{2y^{\frac{1}{2}}} dy \right\} \\ =& \frac{d(\langle X, X\rangle)}{2\langle X, X\rangle^{\frac{1}{2}}} \\ =& \left\langle \frac{2X}{2\langle X, X\rangle^{\frac{1}{2}}}, dX \right\rangle \\ =& \left\langle \frac{X}{\|X\|}, dX \right\rangle. \end{aligned}$$

Тогда первый дифференциал равен

$$\begin{aligned} df(X) =& d(\|AX - B\|_F) \\ =& \left\langle \frac{AX - B}{\|AX - B\|}, d(AX - B) \right\rangle \\ =& \left\langle \frac{AX - B}{\|AX - B\|}, AdX \right\rangle \\ =& \mathrm{Tr}\left(\frac{(AX - B)^\top}{\|AX - B\|} AdX \right) \\ =& \mathrm{Tr}\left(\left(\frac{A^\top (AX - B)}{\|AX - B\|} \right)^\top dX \right) \\ =& \left\langle \frac{A^\top (AX - B)}{\|AX - B\|}, dX \right\rangle. \end{aligned}$$

Приведя к стандартному виду $df(X) = \langle \nabla f(X), dX\rangle$, получим

$$\nabla f(X) = \frac{A^\top (AX - B)}{\|AX - B\|}.$$

Пример 3.23. Найти градиент $\nabla f(X)$ и дифференциал $df(X)$ функции $f(X)$

$$f(X) = \mathrm{Tr}(AXBX^{-1}), \quad X \in \mathbb{R}^{n\times n}, \det(X) \neq 0,$$

где $A, B \in \mathbb{R}^{n\times n}$.

Решение. Перепишем след через скалярное произведение

$$f(X) = \langle I_n, AXBX^{-1}\rangle.$$

Найдём первый дифференциал

$$\begin{aligned} df(X) =& \langle I_n, d(AXBX^{-1})\rangle = \langle I_n, Ad(XBX^{-1})\rangle \\ =& \langle I_n, A(dX)BX^{-1} + AXd(BX^{-1})\rangle \\ =& \langle I_n, A(dX)BX^{-1} + AXB\cdot(-X^{-1}(dX)X^{-1})\rangle \\ =& \mathrm{Tr}(A(dX)BX^{-1}) - \mathrm{Tr}(AXBX^{-1}(dX)X^{-1}) \\ =& \mathrm{Tr}(BX^{-1}A(dX)) - \mathrm{Tr}(X^{-1}AXBX^{-1}(dX)) \\ =& \langle A^\top X^{-\top}B^\top - X^{-\top}B^\top X^\top A^\top X^{-\top}, dX\rangle. \end{aligned}$$

При работе со скалярными произведениями можно переходить к следу и обратно, для применения его полезных циклических и не только свойств. Главное, не забывать о транспонировании.

Приведя к стандартному виду $df(X) = \langle \nabla f(X), dX\rangle$, получим

$$\nabla f(X) = A^\top X^{-\top}B^\top - X^{-\top}B^\top X^\top A^\top X^{-\top}.$$

Пример 3.24. Найти градиент $\nabla f(X)$ и дифференциал $df(X)$ функции $f(X)$

$$f(X) = \mathrm{Tr}(AX^\top X).$$

Решение. Перепишем след через скалярное произведение для удобства

$$f(X) = \langle I, AX^\top X\rangle.$$

Найдём первый дифференциал

$$\begin{aligned}
df(X) =& d\langle I, AX^\top X\rangle = \langle I, Ad(X^\top X)\rangle \\
=& \langle I, Ad(X^\top)X\rangle + \langle I, AX^\top dX\rangle \\
=& \langle I, A(dX)^\top X\rangle + \langle (AX^\top)^\top I, dX\rangle \\
=& \operatorname{Tr}(I^\top A(dX)^\top X) + \langle XA^\top, dX\rangle \\
=& \{\operatorname{Tr}(Y) = \operatorname{Tr}(Y^\top)\} = \operatorname{Tr}(X^\top dXA^\top) + \langle XA^\top, dX\rangle \\
=& \operatorname{Tr}(A^\top X^\top dX) + \langle XA^\top, dX\rangle \\
=& \langle XA, dX\rangle + \langle XA^\top, dX\rangle \\
=& \langle XA + XA^\top, dX\rangle.
\end{aligned}$$

Приведя к стандартному виду $df(X) = \langle \nabla f(X), dX\rangle$, получим

$$\nabla f(X) = XA + XA^\top.$$

Пример 3.25. Логарифм определителя. Найдите первый и второй дифференциалы $df(X)$ и $d^2f(X)$, а также градиент $\nabla f(X)$ функции $f(X)$

$$f(X) = \ln(\det(X))$$

заданной на множестве $X \in \mathbb{S}^n_{++}$ в пространстве $\mathbb{S}^n$.

Решение. Заметим, что из положительной определённости следует $\det(X) > 0$, поэтому $f(X)$ определена корректно в каждой точке.

Найдём первый дифференциал

$$\begin{aligned}
df(X) =& d(\ln\det(X)) \\
=& \frac{d(\det(X))}{\det(X)} \\
=& \frac{\det(X)\langle X^{-\top}, dX\rangle}{\det(X)} \overset{X\in\mathbb{S}^n_{++}}{=} \langle X^{-1}, dX\rangle.
\end{aligned}$$

Приведя к стандартному виду $df(X) = \langle \nabla f(X), dX\rangle$, получим

$$\nabla f(X) = X^{-1}.$$

Теперь найдём второй дифференциал от первого с фиксированным приращением dX_1

$$d^2f(X) = \langle d(X^{-1}), dX_1\rangle = -\langle X^{-1}(dX)X^{-1}, dX_1\rangle.$$

Мы получили билинейную форму от dX, dX_1, выписать тензор производных в явном виде мы не будем.

Посмотрим, является ли эта форма отрицательно полуопределённой при фиксированной $X \in \mathbb{S}^n_{++}$. Для этого возьмём $H \in \mathbb{S}^n$ из исходного пространства.

Поскольку $X \in \mathbb{S}^n_{++}$, то $X^{-1} \in \mathbb{S}^n_{++}$. X^{-1}можно разложить на произведение двух одинаковых матриц, обозначим их $X^{-1/2}$, т.е. $X^{-1} = X^{-1/2}X^{-1/2}$. Такое разложение можно получить, прейдя в базис из собственных векторов, который всегда существует для симметричных матриц, при этом все собственные значения будут положительными

$$X^{-1} = S^{-1}\Lambda S \quad \Rightarrow X^{-1/2} = S^{-1}\sqrt{\Lambda}S.$$

Тогда

$$\begin{aligned} d^2 f(X)[H, H] &= -\langle X^{-1}HX^{-1}, H\rangle \\ &= -\operatorname{Tr}(X^{-1}HX^{-1}H) \\ &= -\operatorname{Tr}(X^{-1/2}X^{-1/2}HX^{-1/2}X^{-1/2}H) \\ &= -\operatorname{Tr}(X^{-1/2}HX^{-1/2} \cdot X^{-1/2}HX^{-1/2}) \\ &= -\langle X^{-1/2}HX^{-1/2}, X^{-1/2}HX^{-1/2}\rangle \\ &= -\|X^{-1/2}HX^{-1/2}\|_F^2 \leq 0. \end{aligned}$$

По одному из критериев выпуклости, который мы узнаем на следующих семинарах, можно сказать по полученному неравенству, что $f(X) = \ln\det(X)$ является вогнутой на $\mathbb{S}^n_{++}$.

Пример 3.26. Найдите первый дифференциал $df(X)$ и градиент $\nabla f(X)$ функции $f(X)$

$$f(X) = \det(AX^{-1}B),$$

где A, X, B – такие матрицы с нужными размерностями, что $AX^{-1}B$ обратима.

Решение. Найдём первый дифференциал

$$\begin{aligned} df(X) =& d(\det(AX^{-1}B)) = \{d\det(Y) = \det(Y)\langle Y^{-\top}, dY\rangle\} \\ =& \det(AX^{-1}B)\langle (AX^{-1}B)^{-\top}, d(AX^{-1}B)\rangle \\ =& \det(AX^{-1}B)\langle (AX^{-1}B)^{-\top}, Ad(X^{-1})B\rangle \\ =& \{d(Y^{-1}) = -Y^{-1}(dY)Y^{-1}\} \\ =& -\det(AX^{-1}B)\langle (AX^{-1}B)^{-\top}, AX^{-1}(dX)X^{-1}B\rangle \\ =& -\det(AX^{-1}B)\operatorname{Tr}((AX^{-1}B)^{-1}AX^{-1}(dX)X^{-1}B) \\ =& -\det(AX^{-1}B)\operatorname{Tr}(X^{-1}B(AX^{-1}B)^{-1}AX^{-1}(dX)) \\ =& -\det(AX^{-1}B)\langle (X^{-1}B(AX^{-1}B)^{-1}AX^{-1})^{\top}, dX\rangle. \end{aligned}$$

Приведя к стандартному виду $df(X) = \langle \nabla f(X), dX\rangle$, получим

$$\nabla f(X) = -\det(AX^{-1}B)X^{-\top}A^{\top}(AX^{-1}B)^{-\top}B^{\top}X^{-\top}.$$

Теперь посмотрим на случай, когда функция переводит матрица в матрицу, и записать производные в компактном виде через матрицу или вектор не получается.

Пример 3.27. Тензор производных. Найдите первый дифференциал и производную функции $f(A)$

$$f(A) = Ax,$$

где $A \in \mathbb{R}^{n\times m}$ – переменная, а $x \in \mathbb{R}^m$ - фиксированный вектор.

Решение. Дифференциал найти достаточно просто

$$df(A) = (dA)x.$$

Однако записать производную в виде вектора или матрицы уже не выйдет нужен трёхмерный тензор вида $\frac{\partial f_k}{\partial A_{ij}}(A)$ размерности $n\times n\times m$. В этом случае может быть полезен прямой подход и тензорное исчисление. Для этого выразим скалярную зависимость функции $f(A)$

$$f_k(A) = \sum_{l=1}^{m} A_{kl}x_l.$$

Теперь возьмём производную $\frac{\partial}{\partial A_{ij}}$

$$\begin{aligned}\frac{\partial f_k}{\partial A_{ij}}(A) =& \sum_{l=1}^{m} \frac{\partial (A_{kl} x_l)}{\partial A_{ij}} \\ =& \sum_{l=1}^{m} \delta(i, j = k, l) x_l \\ =& \sum_{l=1}^{m} \delta(i = k) \delta(j = l) x_l \\ =& \delta(i = k) x_j.\end{aligned}$$

3.3. Автоматическое дифференцирование

3.3.1. Граф вычислений

Настало время поговорить о том, как происходит подсчёт градиентов в реальной жизни. Чаще всего функции, с которыми приходится иметь дело на практике представляют собой последовательность (дифференцируемых) параметрических преобразований. Таким образом, их можно представить в виде вычислительного графа (computational graph), где промежуточным вершинам соответствуют преобразования, входящим стрелкам – входные переменные, а выходным стрелкам – результат преобразования. Это граф должен быть ациклическим, то есть DAG.

На рисунке 1 приведён вычислительный граф для логистической регрессии $f(x, \omega, y) = -\log(1 + \exp(-y\omega^\top x))$ – сплошные линии.

Вычисление значения функции по заданному входу часто называют прямым проходом, или же forward propagation (forward pass). На этом этапе происходит преобразование исходного вектора в целевой и последовательно строятся промежуточные значения — результаты применения преобразований к предыдущим значениям слева направо. Именно поэтому проход называют прямым. В случае линейного графа можно записать его формулой

$$f(x) = u_m(u_{m-1}(\dots u_1(x)\dots)). \tag{12}$$

3.3.2. Backpropagation

Но как же считать производные в таких графах? Ответ может дать правило вычисления дифференциала сложной функции. Рассмотрим

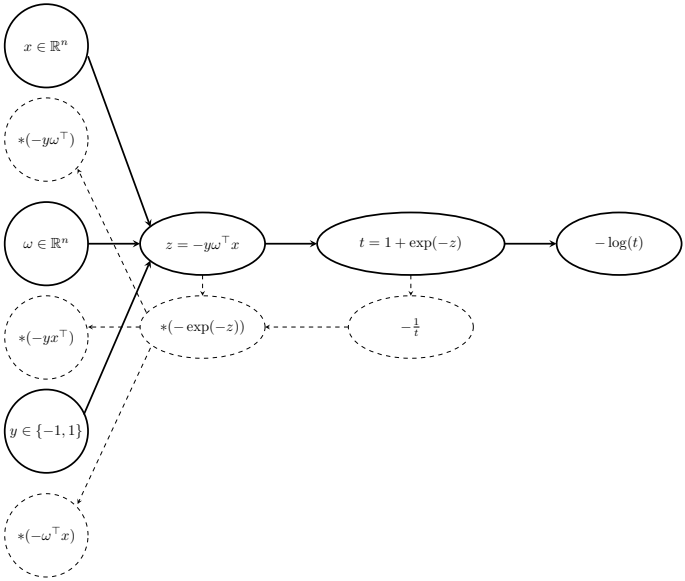


Рис. 1. Граф логистической регрессии

одну конкретную вершину в графе вида $u(x_1, \ldots, x_d)$ и её дифференциал

$$du = \sum_{i=1}^{d} \frac{\partial u}{\partial x_i}(x_i) dx_i,$$

где $\frac{\partial u}{\partial x_i}(x_i)$ – производная по переменным x_i. Причём каждый следующий дифференциал dx_i можно расписать через предыдущую вершину(если, конечно, x_i не являются искомыми переменными), двигаясь по рекурсии в графе от детей к родителям. В случае линейного графа (12) получим итоговую формулу

$$\frac{\partial f}{\partial x} = \underbrace{\frac{\partial u_m}{\partial u_{m-1}}(u_{m-1})}_{\frac{\partial f}{\partial u_{m-1}}} \cdot \frac{\partial u_{m-1}}{\partial u_{m-2}}(u_{m-2}) \cdot \ldots \cdot \frac{\partial u_1}{\partial x}(x). \tag{13}$$

Основная идея backward pass или backpropagation заключается в подсчёте формулы (13) **слева направо**.

В общем случае, пусть $u_1, \ldots, u_m$ – вершины графа вычислений в топологическом порядке(т.е. родители идут перед детьми). Обозначим

производную функции f по вершине u_i как

$$\overline{u_i} = \frac{\partial f}{\partial u_i}.$$

Общий алгоритм действий выглядит так

а) Произвести forward pass и сохранить все значения u_i как функции от их родителей.

б) Положить $\overline{u_m} = 1$ и для всех $i = m-1, \ldots, 1$ посчитать

$$\overline{u_i} = \sum_{j \in \text{потомки}(u_i)} \overline{u_j} \frac{\partial u_j}{\partial u_i}.$$

Вычисление backpropagation на рисунке 1 показано пунктирными линиями.

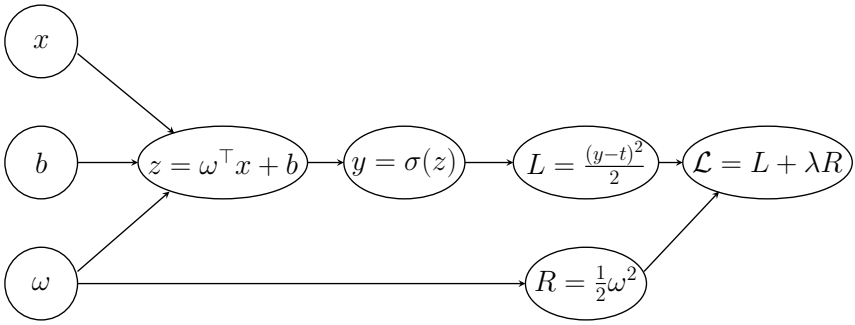


Рис. 2. Граф Вычислений

Пример 3.28. Посчитаем шаг backpropagation в графе вычислений 2, где параметры x, ω - векторы, b - скаляр, а λ, t – заранее фиксированные константы.

Forward pass

$$
\begin{aligned}
z =& \omega^\top x + b, \\
y =& \sigma(z) := \frac{1}{1+\exp(-z)}, \\
L =& \frac{1}{2}(y-t)^2, \\
R =& \frac{1}{2}\omega^2, \\
\mathcal{L} =& L + \lambda R.
\end{aligned}
$$

Backward pass

$$
\begin{aligned}
\overline{\mathcal{L}} =& 1, \\
\overline{R} =& \overline{\mathcal{L}}\frac{d\mathcal{L}}{dR} = \overline{\mathcal{L}} \cdot \lambda = \lambda, \\
\overline{L} =& \overline{\mathcal{L}}\frac{d\mathcal{L}}{dL} = \overline{\mathcal{L}} \cdot 1 = 1, \\
\overline{y} =& \overline{L}\frac{dL}{dy} = (y-t), \\
\overline{z} =& \overline{y}\frac{dy}{dz} = (y-t) \cdot \sigma'(z), \\
\overline{\omega} =& \overline{z}\frac{dz}{d\omega} + \overline{R}\frac{dR}{d\omega} = \overline{z}x^\top + \lambda\omega^\top, \\
\overline{b} =& \overline{z}\frac{dz}{db} = \overline{z}, \\
\overline{x} =& \overline{z}\frac{dz}{dx} = \overline{z}\omega^\top.
\end{aligned}
$$

3.3.3. Обсуждение backpropagation

- Для подсчёта backpropagation необходимо хранить ВСЕ промежуточные значения u_i на всех итерациях алгоритма. Это может быть существенным требованием к памяти, например, в больших нейронных сетях.

- Заметим, что вовсе необязательно для шага 2 полностью считать производную $\frac{\partial u_j}{\partial u_i}$, важно уметь быстро превращать градиент по выходу в градиент по входу (умножать якобиан на строку, см.

главу 3.3.5). Например, можно вспомнить Пример 3.19 с покоординатными функциями, где результат считается просто покоординатным умножением.

- Причина, по которой на практике backpropagation работает так быстро, заключается в том, что вычисления с якобианами преобразований уже эффективно разработаны в рамках библиотек автоматического дифференцирования. Обычно мы даже не создаем и не сохраняем полный якобиан, выполняя matvec напрямую. См. главу 3.3.5.

- Отдельно стоит отметить нейронные сети, которые как раз и состоят из таких блоков. Отдельному блоку совершенно не надо знать, что происходит вокруг. То есть блок действительно может быть запрограммирован как отдельная сущность, умеющая внутри себя делать forward pass и backward pass, после чего блоки механически, как кубики в конструкторе, собираются в большую сеть, которая сможет работать как одно целое.

- Вычисление производной можно представить через ещё один граф вычислений, тем самым, сделав уже backward pass по графу производной, можно считать гессианы и производные высших порядков.

3.3.4. Forward proragation

Может возникнуть желание посчитать формулу (13) не слева направо, а справа налево, распространяя производную в графе в направлении forward pass от родителей к детям. Так производная по параметрам x будет считаться как

$$\frac{\partial u_i}{\partial x} = \sum_{u_j \in \text{родители } u_i} \frac{\partial u_i}{\partial u_j} \frac{\partial u_j}{\partial x}.$$

При этом можно совершать проход вместе с подсчётом u_i и нужно будет хранить только один слой графа.

Однако у этого алгоритма есть один нюанс: в forward pass нужно хранить производные $\frac{\partial u_i}{\partial x}$, а в backward pass $\frac{\partial f}{\partial u_i}$. Если размерность входа x намного больше, чем размерность выхода $f(x)$, то на каждом шаге forward pass нужно будет хранить и обсчитывать настолько же больше данных, чем в backward pass. Это характерно и для нейронных сетей, и для скалярнозначимых функций от тензорного входа в обычной оптимизации. При этом проблема с хранением всех значений

u_i в backpropagation просто нивелируется. Если наоборот размерность выхода функции $f(x)$ намного больше чем размерность входа x, то выгоднее использовать forward pass. В Примере 3.28 мы вплоть до вершины z хранили только скалярные производные, а градиенты появились только в конце.

Аналогично backward pass подсчёт производной в forward pass можно представить в виде графа вычислений и получать производные высших порядков.

3.3.5. Умножение гессиана на вектор и якобиана на строку

Пусть дана дважды непрерывно дифференцируемая функция $f : \mathbb{R}^n \to \mathbb{R}$ с симметричным гессианом $\nabla^2 f(x)$. Покажем, как можно эффективно считать

$$\nabla^2 f(x) \cdot u$$

для любого вектора $u \in \mathbb{R}^n$.

Считать полный гессиан и умножать его на вектор неэффективно особенно при большой размерности n. Но заметим, что

$$d(\langle \nabla f(x), u \rangle) = d(\nabla f)^\top u = dx^\top \nabla^2 f(x) \cdot u = dx^\top \nabla g(x),$$

где $g(x) = \langle \nabla f(x), u \rangle$. Таким образом вместо полного гессиана можно найти градиент функции вида g, с которой могут справиться библиотеки автоматического дифференцирования jax/autograd/pytorch/tensorflow (для градиента функции тоже можно построить граф вычислений). Особенно отметим jax, который эффективен для функций вида $g : \mathbb{R}^n \to \mathbb{R}$.

Аналогично для функции вида $f : \mathbb{R}^n \to \mathbb{R}^m$ и вектора $v \in \mathbb{R}^m$ считается значение $v^\top J_f(x)$ в точке x, а именно

$$d\langle u, f(x) \rangle = \langle u, J_f dx \rangle = \langle J_f^\top u, dx \rangle = \langle \nabla g(x), dx \rangle,$$

где $g(x) = \langle f(x), u \rangle$.

Именно поэтому вычисления из backpropagation можно эффективно реализовать на практике.

4. Лекция 1. Введение

Александр Безносиков, Александр Богданов

4.1. Задача оптимизации

Задачу оптимизации, которая будет рассматриваться в рамках данного пособия, можно сформулировать следующим образом:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \mathcal{X} \subseteq \mathbb{R}^d \\ & g_i(x) = 0, \quad i = 1, \dots l \\ & h_j(x) \leq 0, \quad j = 1, \dots m. \end{aligned} \tag{14}$$

Получается, что варьируя вектор x, нам необходимо найти значение f^* – минимум функции f такой, что («s.t.» — «such that») дополнительно выполнен ряд условий/ограничений. Также часто будет встречаться постановка вида $\arg\min$ — в данном случае уже необходимо найти не минимальное значение функции f^*, а значение вектора x^*, на котором это значение достигается $f^* = f(x^*)$. Понятно, что в общем случае уникальность, да и вообще существование x^* и f^* никто не гарантирует (рассмотрите, например, задачу $\min_{x \in \mathbb{R}} x$). Отметим, что задача (14) имеет общий вид, и мы часто будем ее упрощать. Например, можно рассматривать задачу безусловной оптимизации, т.е. убрать всем ограничения вида равенств и неравенств и положить $\mathcal{X} = \mathbb{R}^d$.

Более подробно остановимся на объектах из формулировки (14):

- $f : \mathcal{X} \to \mathbb{R}$ — некоторая функция, заданная на множестве $\mathcal{X}$. Данная функция является целевой для задачи оптимизации (14). Мы будем накладывать дополнительные свойства на f: Липшецевость, гладкость, выпуклость. Связано это с тем, что уже в рамках этого параграфа станет понятно, что без предположений на f не получится построить оптимистичную теорию поиска решения (14).

- $\mathcal{X} \subseteq \mathbb{R}^d$ — подмножество d-мерного пространства. В общем случае $\mathcal{X}$ может быть любым множеством, но часто мы будем предполагать, что $\mathcal{X}$ является «простым» (суть «простоты» в данном случае неоднозначна и будет раскрыта далее, в момент изучения методов оптимизации на «простых множествах»). Например, в качестве $\mathcal{X}$ может выступать шар радиуса R с центром в точке a в p-норме $B_p^d(R, a) = \{x \in \mathbb{R}^d \mid \|x - a\|_p \leq R\}$, вероятностный

симплекс $\triangle^d = \{x \in \mathbb{R}^d \mid x_i \geq 0,\ i = 1, \dots d,\ \sum_{i=1}^d x_i = 1\}$ или положительный ортант $\perp^d = \{x \in \mathbb{R}^d \mid x_i \geq 0\}$.

- Также в задаче (14) присутствуют функциональные ограничения вида равенств и неравенств. $g_i(x) : \mathcal{X} \to \mathbb{R}$, $i = 1, \dots, l$ и $h_j(x) : \mathcal{X} \to \mathbb{R}$, $i = 1, \dots, m$ — функции, задающие ограничения. Мы также будем в дальнейшем предполагать, что данные функции являются достаточно «хорошими». Отметим, что формально все функциональные ограничения можно было просто спрятать в $\mathcal{X}$ (часто возможно и обратно — $\mathcal{X}$ записывается в виде набора функциональных ограничений), главное проблемой при таком действии может стать потеря «простоты» $\mathcal{X}$.

4.2. Общая схема методов оптимизации

Жизненный опыт подсказывает, что даже в одномерном случае задача оптимизации может не иметь аналитического решения (если вообще имеет). Если же задача оптимизации имеет решение, то на практике её обычно решают, вообще говоря, приближённо, итеративно приближаясь к решению. Для этого применяются специальные алгоритмы, которые и называют *методами оптимизации*.

При создании методов оптимизации хочется добиться унифицированности. Потому что нет смысла искать лучший метод для решения конкретной задачи. Например, лучший метод для решений задачи $\min_{x \in \mathbb{R}} x^2$ сходится за 1 итерацию: этот метод просто всегда выдаёт ответ $x^* = 0$. Очевидно, что для других задач такой метод не пригоден. Метод должен быть пригоден для целого класса однотипных или похожих задач.

Так как метод разрабатывается для целого класса задач, то метод не может иметь с самого начала полной информации о задаче (14). Вместо этого метод использует модель задачи, например, формулировку задачи, описание функциональных компонент, множества, на котором происходит оптимизация и т.д.. Предполагается, что численный метод может накапливать специфическую информацию о задаче при помощи некоторого *оракула*. Под оракулом можно понимать некоторое устройство (программу, процедуру), которое отвечает на последовательные вопросы метода оптимизации о свойствах функции. В идеальном мире, наверное, можно было спросить у оракула, где всю информацию о решении задачи (14), но очевидно, что такого рода оракулы неинтересны для практики. Мы будем работать с более приземленными вариантами оракулов, которые могут возвращать локальную информацию о целевой функции.

Определение 4.1. Оракулы имеют разный порядок в зависимости от степени подробности, возвращаемой информации:

- *Оракул нулевого порядка* в запрашиваемой точке x возвращает значение целевой функции $f(x)$.

- *Оракул первого порядка* в запрашиваемой точке возвращает значение функции $f(x)$ и её градиент в данной точке $\nabla f(x) = \left(\frac{\partial f(x)}{\partial x_1}, \ldots, \frac{\partial f(x)}{\partial x_n}\right)$.

- *Оракул второго порядка* в запрашиваемой точке возвращает значение и градиент функции $f(x), \nabla f(x)$, а также её гессиан в данной точке $\left(\nabla f^2(x)\right)_{ij} = \frac{\partial^2 f(x)}{\partial x_j \partial x_j}$.

Можно продолжать и до производных более высоких порядков.

Алгоритм 1 Общая итеративная схема метода оптимизации $\mathcal{M}$

Вход: начальная точка x^0 (0 – верхний индекс), требуемая точность решения задачи $\varepsilon > 0$

Настройка: Задать $k = 0$ (счётчик итераций) и $I_{-1} = \varnothing$ (накапливаемая информационная модель решаемой задачи)

1: **while** не выполнен критерий остановки $\mathcal{T}_\varepsilon$ **do**
2: Задать вопрос к оракулу $\mathcal{O}$ в точке x^k
3: Пересчитать информационную модель: $I_k = I_{k-1} \cup (x^k, \mathcal{O}(x^k))$
4: Применить правило метода $\mathcal{M}$ для получения новой точки x^{k+1} по модели I_k
5: Положить $k := k + 1$
6: **end while**

Замечание 4.2. В Алгоритме 1 и далее мы будем использовать верхний индекс x^k для обозначения переменной x на k итерации.

Пример 4.3. Рассмотрим задачу оптимизации

$$\min_{x \in \mathbb{R}^d} f(x),$$

где функция $f(x)$ дифференцируема. Предположим, что в любой точке мы можем посчитать её градиент. Тогда можно воспользоваться следующим методом:

Алгоритм 2 Градиентный спуск с постоянным размером шага

Вход: размер шага $\gamma > 0$, стартовая точка $x^0 \in \mathbb{R}^d$, количество итераций K

1: **for** $k = 0, 1, \ldots, K-1$ **do**
2: Вычислить $\nabla f(x^k)$
3: $x^{k+1} = x^k - \gamma \nabla f(x^k)$
4: **end for**

Выход: x^K

С градиентным спуском мы будем подробнее знакомиться уже в следующих параграфах. Сейчас мы хотим лишь понять, как он ложится в общую модель. В данном случае оракул $\mathcal{O}$ – оракул первого порядка, который возвращает градиенты ∇f в запрашиваемых точках. При этом в информационную модель мы каждую итерацию кладем точки $(x^k, \mathcal{O}(x^k)) = (x^k, \nabla f(x^k))$. Сам же метод градиентного спуска $\mathcal{M}$ используют только последнюю пару точке из информационной модели и по правилу: $x^{k+1} = x^k - \gamma \nabla f(x^k)$.

Замечание 4.4. В Алгоритме 1 информационная модель растет линейно с номером итерации k. Пример 4.3 показывает, что для практических методов вовсе не обязательно хранить всю информационную модель — градиентному спуска нужна только текущая точка x^k и $\mathcal{O}(x^k) = \nabla f(x^k)$.

Внимательный читатель мог заметить, что Пример 4.3 не полностью соответствует Алгоритму 1, а именно градиентный спуск останавливается после K итераций, где число K наперед задано. Схема из Алгоритма 1 использует критерий остановки $\mathcal{T}_\varepsilon$, который завязан на точность ε. Приведем наиболее популярные варианты критериев.

Определение 4.5. Под утверждением, что задача решена с точностью ε (найдено ε-решение /выполнен критерий остановки), можно понимать следующее:

- по аргументу: $\|x^k - x^*\|_2 \leq \varepsilon$,
- по значению функции: $f(x^k) - f^* \leq \varepsilon$,
- по норме градиента: $\|\nabla f(x^k)\|_2 \leq \varepsilon$.

Отметим, что данные критерии являются теоретическими. Мы их будем использовать при доказательстве сходимости методов и оценке их скорости работы. На практике же сложно, например, оценить $\|x^k - x^*\|$, так как мы не знаем расположение точки x^*. При этом

на практике может оказаться полезным критерии сходимости вида: $\|x^{k+1} - x^k\| \leq \varepsilon$. Из такого критерия не следуют какие-либо гарантии на $\|x^k - x^*\|_2$ — можно привести массу очевидно плохих методов, которые стоят на месте вдали от решения. Но верно обратное, а именно, если $\|x^{k+1} - x^*\|_2 \leq \|x^k - x^*\|_2 \leq \varepsilon/2$, то по неравенству треугольника:

$$\|x^{k+1} - x^k\|_2 \leq \|x^{k+1} - x^*\|_2 + \|x^k - x^*\|_2 \leq \varepsilon.$$

Поэтому при имеющейся теории или интуиции о том, что $\|x^k - x^*\|_2 \to 0$, критерий на основе $\|x^{k+1} - x^k\|$ может быть более чем полезен.

Похожая ситуация с критерием на основе $f(x^k) - f^*$ с той лишь разнице, что для некоторых задач мы можем знать значение f^* (например для $\min_{x \in \mathbb{R}^d} \|Ax - b\|_2^2$).

Следить за поведением $\|\nabla f(x^k)\|_2$ кажется более практичным. Это частично является правдой, но до того момента, пока мы рассматриваем безусловные варианты задачи (14) (без ограничений и с $\mathcal{X} = \mathbb{R}^d$). В общем же случае, такого рода критерий может ничего не сказать — достаточно рассмотреть задачу $\min_{x \in [1;2]} x^2$.

4.3. Сложность методов оптимизации. Верхние и нижние оценки

Для понимания того, насколько эффективно/быстро/дешево работают методы оптимизации, как их сравнивать между собой, необходимо ввести формальные понятия для описания *сложности* методов оптимизации:

Определение 4.6.

- *Аналитическая/Оракульная сложность* — число обращений метода к оракулу, необходимое для решения задачи с точностью ε (см. Определение 4.5).
- *Арифметическая/Временная сложность* — общее число вычислений (включая работу оракула), необходимых для решения задачи с точностью ε.
- *Итерационная сложность* — общее число итераций метода, необходимых для решения задачи с точностью ε.

Арифметическая сложность дает полное количество атомарных операций (сложений/умножений двух чисел), которое выполнил метод для решения задачи оптимизации. Время, которое затратит алгоритм, можно считать пропорциональным арифметическая сложность.

Оракульная сложность учитывает только количество вызовов оракула, что пропорционально общему количеству атомарных операций, которые выполнил оракул в ходе работы метода. На самом деле часто оракульной сложности бывает достаточно для оценки времени работы метода, так как вычисления оракула являются наиболее дорогой операцией во всем методе. Рассмотрим, например, Алгоритм из Примера 4.3 для целевой функции f из Примера 3.14. Вычисления ∇f требуют умножения матрицы на вектор, а все остальные операции Алгоритма производятся с вектором.

Итерационная сложность несет меньше всего информации о времени работы метода, так как разные методы могут иметь абсолютно разную арифметическую сложность одного вызова оракула. Поэтому из того, что один метод достигает ε-решения за меньшее число итераций, вообще говоря, не следует, что он работает быстрее по времени.

Теперь попробуем понять, а какие гарантии на сложность мы вообще можем дать на поиск ε-решения (14). Для этого рассмотреть следующую, на первый взгляд, довольно простую задачу оптимизации:

$$\min_{x \in C^d} f(x), \tag{15}$$

где $C^d = \{x \in \mathbb{R}^d \mid 0 \leq x_i \leq 1,\ i = 1, \ldots, d\}$ —«кубик». При этом мы дополнительно предположим, что функция $f(x)$ является M-липшицевой на C^d относительно ℓ_∞-нормы, т.е. для любых $x, y \in C^d$ справедливо

$$|f(x) - f(y)| \leq M\|x - y\|_\infty = M \max_{i=1,\ldots,d} |x_i - y_i|. \tag{16}$$

Замечание 4.7. Множество C^d является ограниченным и замкнутым, т.е. компактом, а из липшицевости функции f следует и её непрерывность, поэтому задача (15) имеет решение, так как непрерывная на компакте функция достигает своих минимального и максимального значений.

В поисках решения давайте ограничимся методами нулевого порядка (т.е. методы, которые могут вызывать оракул, возврвщающий значения целевой функции). Более формально:

- **Цель:** найти $\bar{x} \in C^d$: $f(\bar{x}) - f^* \leq \varepsilon$, где $f^* = \min_{x \in C^d} f(x)$.
- **Класс методов:** методы нулевого порядка.

Рассмотрим следующий довольно незатейливый алгоритм для решения поставленной цели.

Алгоритм 3 Метод равномерного перебора

Вход: целочисленный параметр перебора $p \geq 1$

1: Сформировать $(p+1)^d$ точек вида $x_{(i_1,\ldots,i_d)} = \left(\frac{i_1}{p}, \frac{i_2}{p}, \ldots, \frac{i_d}{p}\right)^\top$, где $(i_1, \ldots, i_d) \in \{0, 1, \ldots, p\}^n$

2: Среди точек $x_{(i_1,\ldots,i_d)}$ найти точку $\bar{x}$ с наименьшим значением целевой функции f.

Выход: $\bar{x}, f(\bar{x})$

Попробуем получить какие-нибудь гарантии на решение, которое находит данный алгоритм.

Теорема 4.8. [Теорема 1.1.1. из [1]] Алгоритм 3 с параметром p возвращает такую точку $\bar{x}$, что

$$f(\bar{x}) - f^* \leq \frac{M}{2p},$$

откуда следует, что методу равномерного перебора нужно в худшем случае

$$\left(\left\lfloor \frac{M}{2\varepsilon} \right\rfloor + 2\right)^d \tag{17}$$

обращений к оракулу, чтобы гарантировать $f(\bar{x}) - f^* \leq \varepsilon$.

Доказательство. $\square$ Пусть x^* – решение задачи (возможно не единственная точка глобального минимума функции f). Тогда в построенной «сетке» из точек найдётся такая точка $x_{(i_1,\ldots,i_d)}$, что $x := x_{(i_1,\ldots,i_d)} \preceq x^* \preceq x_{(i_1+1,\ldots,i_d+1)} =: y$, где знак «$\preceq$» применяется покомпонентно. Точки x и y определяют сторону «кубика сетки», в котором лежит x^*, т.е. $x_i^* \in [x_i, y_i]$ для всех $i = 1, \ldots, d$. Отметим, что сторона данного «кубика» $y_i - x_i = \frac{1}{p}$. Кроме того, рассмотрим точки $\hat{x}$ и $\tilde{x}$ такие, что $\hat{x} = \frac{x+y}{2}$ и

$$\tilde{x}_i = \begin{cases} y_i, & \text{если } x_i^* \geq \hat{x}_i, \\ x_i, & \text{иначе.} \end{cases}$$

$\hat{x}$ – центр «кубика», а $\tilde{x}$ определяет ближайший к x^* «уголок кубика». Заметим, что $\tilde{x}$ принадлежит «сетке» и $|\tilde{x}_i - x_i^*| \leq \frac{1}{2p}$, так как в худшем случае x^* расположен в $\hat{x}$, а значит равноудален от всех «уголков». Тогда $\|\tilde{x} - x^*\|_\infty = \max_{i \in 1,\ldots,d} |\tilde{x}_i - x_i^*| \leq \frac{1}{2p}$. Поскольку $f(\bar{x}) \leq f(\tilde{x})$

(так как $f(\bar{x})$ – минимум по всем точкам сетки), получаем

$$f(\bar{x}) - f^* \leq f(\tilde{x}) - f^* \leq M\|\tilde{x} - x^*\|_\infty \leq \frac{M}{2p}.$$

Выписанная выше оценка достигается методом равномерного перебора за $(p+1)^d$ обращений к оракулу. Следовательно, чтобы гарантировать $f(\bar{x}) - f^* \leq \varepsilon$, необходимо взять $p = \left\lfloor \frac{M}{2\varepsilon} \right\rfloor + 1$, т.е. метод сделает $\left(\left\lfloor \frac{M}{2\varepsilon} \right\rfloor + 2\right)^d$ обращений к оракулу нулевого порядка. ■

Интересно оценить порядок величин, которые мы получили:

- Предположим $M = 2$, $d = 13$ И $\varepsilon = 0.01$, то есть размерность задачи сравнительно небольшая (можно сказать, что никакая для задач оптимизации в современных приложениях) и точность решения задачи не слишком высокая (явно не для того, чтобы ракеты в космос запускать).
- Необходимое число обращений к оракулу: $\left(\left\lfloor \frac{M}{2\varepsilon} \right\rfloor + 2\right)^d = 102^{13} > 10^{26}$.
- Сложность одного вызова оракула не менее 1, но если потребовать, чтобы он обязательно считал переданную ему точки, то сложность не менее d операции.
- Производительность компьютера: 10^{13} арифметических операций в секунду.
- Общее время: хотя бы 10^{13} секунд, что займет порядка миллиона лет.

Выводы получаются не очень оптимистичные. Но есть надежда, что наш теоретический анализ из Теоремы 4.8 плох, либо метод не самый лучший, и существует какой-нибудь зубодробительный алгоритм, который будет давать куда более приятные гарантии на число оракульных вызовов.

В Теореме 4.8 мы получили так называемые верхние оценки на гарантии нахождения решения, но существует и обратное понятие — нижние оценки.

Определение 4.9. Пусть нам дан некоторый класс методов, а также некоторый класс задач оптимизации.

- *Верхняя оценка* – гарантия, что для *рассматриваемый* метод из класса методов на *любой* задаче из класса решаемых задач имеет

оракульную/арифметическую/итерационную сложность не хуже, чем утверждает верхняя оценка.

- *Нижняя оценка* – гарантия, что для *любого* метода из класса методов *существует* «плохая» задача из класса решаемых задач такая, что метод имеет оракульную/арифметическую/итерационную сложность не лучше, чем утверждает нижняя оценка.

Напомним, что в этом параграфе мы рассматриваем класс методов нулевого порядка, т.е. все методы, которые каким-либо образом используют информацию о значениях функции (но не производные). Также класс задач, которые мы сейчас рассматриваем описывается с помощью (15) и (16).

Нижние оценки нужны, чтобы понять, насколько полученные верхние оценки хороши. В частности, если верхние и нижние оценки совпали, это означает оптимальность предложенного метода для данного класса задач. Но если нижние оценки не совпали с верхними, то это может ничего не значить — возможно, в нижних оценках подобрана недостаточно «плохая» функция, а возможно, при получении верхних оценок пришлось загрубить теоретические выкладки. Нижние оценки можно получать не только для класса методов, но и для определенного метода, чтобы доказать оптимальность и неулучшаемость теоретического анализа и полученной верхней оценки.

Вернемся к нашей задач: (15)+(16) и методам, оперирующим информацией нулевого порядка. Следующая теорема представит нижние оценки на оракульную сложность.

Теорема 4.10. [Теорема 1.1.2. из [1]] Пусть $\varepsilon < \frac{M}{2}$. Тогда аналитическая сложность описанного класса задач, т.е. аналитическая сложность метода на «худшей» для него задаче из данного класса, составляет по крайней мере

$$\left(\left\lfloor \frac{M}{2\varepsilon} \right\rfloor\right)^d - 1 \quad \text{вызовов оракула,} \tag{18}$$

чтобы гарантировать $f(\bar{x}) - f^* \leq \varepsilon$.

Доказательство. $\square$ Пусть $p = \left\lfloor \frac{M}{2\varepsilon} \right\rfloor$. Доказываем от противного: предположим, что существует такой метод, который решает задачу за $N < (p^d - 1)$ обращений к оракулу, чтобы решить задачу с точностью ε (по функции). Построим такую функцию, на которой метод не сможет найти ε-решение, при помощи сопротивляющегося оракула: пусть из-

начально наша целевая функция $f(x)$ всюду равна 0. Запустим метод, он запросит значение f в N точках, везде получит 0 и выдаст какую-то точку (возможно, отличную от всех предыдущих N, как ответ). В итоге мы в ходе работы алгоритма заглянули в $N+1 < p^d$ точку. Тогда по принципу Дирихле найдётся такой «кубик» $C = \{x \mid \hat{x} \preceq x \preceq \hat{x} + \frac{1}{p}e\}$ (где $\hat{x}$ и $\hat{x} + \frac{1}{p}e$ — точки из «сетки» с шагом p, e — вектор из единиц, а знак «$\preceq$» применяется покомпонентно), который не содержит ни одной из $N+1$ точки (в том числе и выхода метода). Пусть x^* — это центр «кубика» C, т.е. $x^* = \hat{x} + \frac{1}{2p}e$. Немного модифицируем функцию $\bar{f}(x) = \min\{0, M\|x - x^*\|_\infty - \varepsilon\}$. Функция $\bar{f}(x)$ липшицева с константой M относительно ℓ_∞-нормы и принимает своё минимальное значение $-\varepsilon$ в точке x^*. Более того, функция $\bar{f}(x)$ отлична от нуля только внутри куба $C' = \{x \mid \|x - x^*\| \leq \frac{\varepsilon}{M}\}$, который лежит внутри куба C, т.к. $2p \leq \frac{M}{\varepsilon}$. Следовательно, рассмотренный метод на данной функции не может найти ε-решение. Противоречие. ■

Получается, что нижняя оценка из Теоремы 4.10 не особо лучше Теоремы 4.8. Зависимость $\left(\frac{M}{\varepsilon}\right)^d$ присутствует в обоих результатах. А это значит, что для класса липшецевых функций в общем случае все довольно печально. Нужны дополнительные предположения на задачу (14), чтобы гарантировать более оптимистичные гарантии. Но это вопрос уже следующих параграфов.

4.4. Скорость сходимости методов

Поймем, с какого характера верхними оценки на скорость сходимости/скорость приближения к решению мы хотим и будем иметь дело в дальнейшем.

Определение 4.11. Выделим четыре основных типа скоростей сходимости:

- Сублинейная: $\|x^k - x^*\|_2 \leq \dfrac{C}{k^\alpha},\ C > 0,\ \alpha > 0.$
- Линейная: $\|x^k - x^*\|_2 \leq Cq^k,\ C > 0,\ 0 < q < 1.$
- Сверхлинейная: $\|x^k - x^*\|_2 \leq Cq^{k^p},\ C > 0,\ 0 < q < 1,\ p > 1.$
- Квадратичная: $\|x^k - x^*\|_2 \leq Cq^{2^k},\ C > 0,\ 0 < q < 1.$

Из определения очевидно, что типы следуют в порядке увеличения скорости сходимости. Приведенные в Определении 4.11 типы скоро-

стей могут относится, как к арифметической, оракульной так и итерационной сложности. В частности, на Рисунке 3 мы рассматриваем итерационной сложности. Можно оценить, как данные виды сходимости визуально соотносятся между собой. Также из рисунка становится понятно, почему линейная скорость получали такое название.

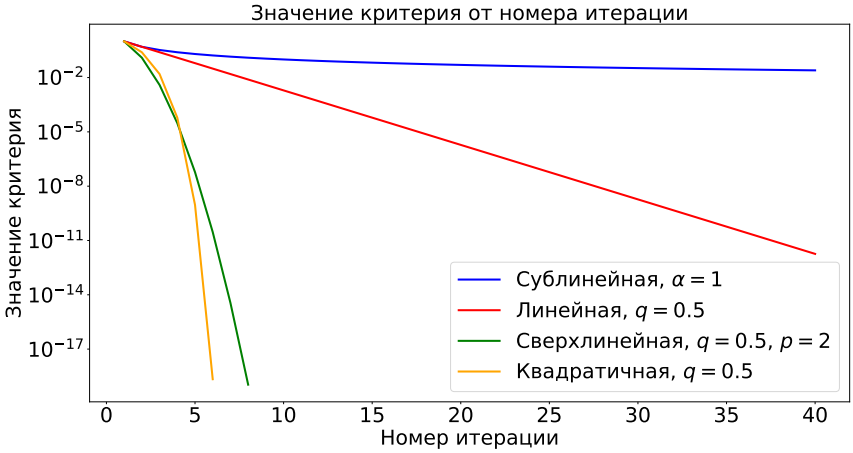


Рис. 3. Сравнение и характерный вид основных типов скоростей сходимости

Замечание 4.12. Из Определения 4.11, где даны скорости приближения к решению через k арифметических операций/оракульных вызовов/итераций, можно легко получить оценки на соотвествующие сложности.

Пусть для некоторого методы мы доказали, следующую оценку на сходимость: $\|x^k - x^*\|_2 \leq \dfrac{C}{k^\alpha}$, где k — номер арифметических операций-/оракульного вызова/итерации. Тогда, чтобы гарантировать точность ε ($\|x^k - x^*\|_2 \leq \varepsilon$), нам необходимо сделать $k \geq \left(\frac{C}{\varepsilon}\right)^{1/\alpha}$:

$$\|x^k - x^*\|_2 \leq \frac{C}{k^\alpha} \leq \varepsilon \;\Rightarrow\; k^\alpha \geq \frac{C}{\varepsilon}.$$

Аналогичные манипуляции можно проделать со всеми типа сходимости из Определения 4.11.

Литература

1. Нестеров Ю.Е. Методы выпуклой оптимизации. Москва : Изд-во МЦНМО, 2010. 281 с.