

Марк Питер Дайзенрот,  
Альдо Фейзал, Чен Сунь Он

# Математика в машинном обучении

**Докопайся до сути**

Математические основы базовых концепций машинного обучения, позволяющие устраниить пробел между университетским курсом математики и уровнем, необходимым для изучения МО



Марк Питер Дайзенрот,  
А. Альдо Фейзал, Чен Сунь Он

# Математика в машинном обучении

Докопайся до сути



Санкт-Петербург · Москва · Минск

2024

ББК 32.813+22.1  
УДК 004.85+51  
Д14

**Дайзенрот Марк Питер, Альдо Фейзал А., Чен Сунь Он**

- Д14 Математика в машинном обучении. — СПб.: Питер, 2024. — 512 с.: ил. — (Серия «Для профессионалов»).  
ISBN 978-5-4461-1788-8

Фундаментальные математические дисциплины, необходимые для понимания машинного обучения, — это линейная алгебра, аналитическая геометрия, векторный анализ, оптимизация, теория вероятностей и статистика. Традиционно все эти темы размазаны по различным курсам, поэтому студентам, изучающим data science или computer science, а также профессионалам в МО, сложно выстроить знания в единую концепцию.

Эта книга самодостаточна: читатель знакомится с базовыми математическими концепциями, а затем переходит к четырем основным методам МО: линейной регрессии, методу главных компонент, гауссову моделированию и методу опорных векторов.

Тем, кто только начинает изучать математику, такой подход поможет развить интуицию и получить практический опыт в применении математических знаний, а для читателей с базовым математическим образованием книга послужит отправной точкой для более продвинутого знакомства с машинным обучением.

**16+** (В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ББК 32.813+22.1  
УДК 004.85+51

Права на издание получены по соглашению с Cambridge University Press. Все права защищены. Никакая часть данной книги не может быть воспроизведена в какой бы то ни было форме без письменного разрешения владельцев авторских прав.

Информация, содержащаяся в данной книге, получена из источников, рассматриваемых издательством как надежные. Тем не менее, имея в виду возможные человеческие или технические ошибки, издательство не может гарантировать абсолютную точность и полноту приводимых сведений и не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-1108455145 англ.  
ISBN 978-5-4461-1788-8

© Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong 2020  
© Перевод на русский язык ООО «Прогресс книга», 2023  
© Издание на русском языке, оформление ООО «Прогресс книга», 2023  
© Серия «Для профессионалов», 2023

# Краткое содержание

[https://t.me/it\\_boooks/2](https://t.me/it_boooks/2)

Математика в машинном обучении .....	15
Условные обозначения .....	17
Предисловие .....	20
Благодарности .....	24

## **ЧАСТЬ I Математические основы**

Глава 1. Введение и мотивация .....	28
Глава 2. Линейная алгебра .....	35
Глава 3. Аналитическая геометрия .....	100
Глава 4. Матричные разложения .....	134
Глава 5. Векторный анализ .....	185
Глава 6. Вероятность и распределения .....	225
Глава 7. Непрерывная оптимизация .....	287

## **ЧАСТЬ II Главные задачи машинного обучения**

Глава 8. О сочетании модели и данных .....	318
Глава 9. Линейная регрессия .....	366
Глава 10. Снижение размерности с помощью анализа главных компонент .....	400
Глава 11. Оценка плотности с помощью моделей гауссовой смеси .....	438
Глава 12. Классификация методом опорных векторов .....	464
Библиография .....	492

# Оглавление

От издательства .....	14
О научном редакторе русского издания .....	14
Математика в машинном обучении .....	15
Условные обозначения .....	17
Список аббревиатур и сокращений .....	19
Предисловие .....	20
Зачем нужна еще одна книга по машинному обучению? .....	21
Какова целевая аудитория книги? .....	22
Благодарности .....	24

## ЧАСТЬ I **Математические основы**

Глава 1. Введение и мотивация .....	28
1.1. Поиск интуитивно понятных формулировок .....	29
1.2. Два способа читать эту книгу .....	30
1.3. Упражнения и обратная связь .....	34
Глава 2. Линейная алгебра .....	35
2.1. Системы линейных уравнений .....	38
2.2. Матрицы .....	41
2.2.1. Сложение и перемножение матриц .....	42
2.2.2. Обращение и транспонирование .....	44
2.2.3. Умножение на скаляр .....	46
2.2.4. Компактное представление системы уравнений .....	47

2.3. Решение систем линейных уравнений . . . . .	47
2.3.1. Частное и общее решение . . . . .	48
2.3.2. Элементарные преобразования . . . . .	49
2.3.3. Прием с $-1$ . . . . .	54
2.3.4. Алгоритмы для решения системы линейных уравнений . . . . .	57
2.4. Векторные пространства . . . . .	58
2.4.1. Группы . . . . .	58
2.4.2. Векторные пространства . . . . .	60
2.4.3. Векторные подпространства . . . . .	62
2.5. Линейная независимость . . . . .	63
2.6. Базис и ранг . . . . .	68
2.6.1. Генерация множества и базиса . . . . .	69
2.6.2. Ранг . . . . .	72
2.7. Линейные отображения . . . . .	73
2.7.1. Матричное представление линейных отображений . . . . .	75
2.7.2. Изменение базиса . . . . .	79
2.7.3. Образ и ядро . . . . .	85
2.8. Аффинные пространства . . . . .	88
2.8.1. Аффинные подпространства . . . . .	89
2.8.2. Аффинные отображения . . . . .	90
2.9. Дополнительное чтение . . . . .	91
Упражнения . . . . .	92
 Глава 3. Аналитическая геометрия . . . . .	100
3.1. Нормы . . . . .	100
3.2. Внутренние произведения . . . . .	102
3.2.1. Скалярное произведение . . . . .	103
3.2.2. Общие внутренние произведения . . . . .	103
3.2.3. Симметричные положительно определенные матрицы . . . . .	104
3.3. Длины и расстояния . . . . .	106
3.4. Углы и ортогональность . . . . .	108
3.5. Ортонормированный базис . . . . .	111
3.6. Ортогональное дополнение . . . . .	112
3.7. Внутреннее произведение функций . . . . .	113

3.8. Ортогональные проекции . . . . .	114
3.8.1. Проекция на одномерные подпространства (прямые) . . . . .	116
3.8.2. Проекция на общие подпространства . . . . .	119
3.8.3. Ортогонализация Грама — Шмидта . . . . .	124
3.8.4. Проекция на аффинные подпространства . . . . .	125
3.9. Повороты . . . . .	126
3.9.1. Повороты в $\mathbb{R}^2$ . . . . .	127
3.9.2. Повороты в $\mathbb{R}^3$ . . . . .	128
3.9.3. Поворот в $n$ измерениях . . . . .	129
3.9.4. Свойства поворотов . . . . .	130
3.10. Дополнительное чтение . . . . .	130
Упражнения . . . . .	131
 Глава 4. Матричные разложения . . . . .	134
4.1. Детерминант и след . . . . .	136
4.2. Собственные значения и собственные векторы . . . . .	142
4.2.1. Графическая интуиция в двух измерениях . . . . .	147
4.3. Разложение Холецкого . . . . .	154
4.4. Собственное разложение и диагонализация . . . . .	156
4.4.1. Геометрическая интуиция для собственного разложения . . . . .	158
4.5. Разложение по сингулярным значениям . . . . .	160
4.5.1. Геометрические интуиции для SVD . . . . .	161
4.5.2. Построение SVD . . . . .	164
4.5.3. Разложение на собственные значения и разложение на сингулярные значения . . . . .	169
4.6. Матричное приближение . . . . .	173
4.7. Матричная филогения . . . . .	178
4.8. Дополнительное чтение . . . . .	180
Упражнения . . . . .	182
 Глава 5. Векторный анализ . . . . .	185
5.1. Дифференцирование функций одной переменной . . . . .	187
5.1.1. Ряд Тейлора . . . . .	189
5.1.2. Правила дифференцирования . . . . .	192

5.2. Частные производные и градиенты . . . . .	193
5.2.1. Основные правила взятия частных производных . . . . .	195
5.2.2. Цепное правило . . . . .	196
5.3. Градиенты векторнозначных функций . . . . .	197
5.4. Градиенты матриц . . . . .	204
5.5. Полезные тождества для вычисления градиентов . . . . .	208
5.6. Обратное распространение ошибки и автоматическое дифференцирование . . . . .	209
5.6.1. Градиенты в глубоких нейронных сетях . . . . .	210
5.6.2. Автоматическое дифференцирование . . . . .	211
5.7. Производные высших порядков . . . . .	215
5.8. Линеаризация и ряды Тейлора для нескольких переменных . . . . .	216
5.9. Для дальнейшего чтения . . . . .	222
Упражнения . . . . .	223
 Глава 6. Вероятность и распределения . . . . .	225
6.1. Построение вероятностного пространства . . . . .	225
6.1.1. Философские вопросы . . . . .	226
6.1.2. Вероятность и случайные величины . . . . .	228
6.1.3. Статистика . . . . .	231
6.2. Дискретные и непрерывные распределения . . . . .	232
6.2.1. Дискретные вероятности . . . . .	232
6.2.2. Непрерывные вероятности . . . . .	234
6.2.3. Различия дискретных и непрерывных распределений . . . . .	236
6.3. Правило суммы, правило произведения и теорема Байеса . . . . .	238
6.4. Обобщающие статистики и независимость . . . . .	241
6.4.1. Среднее и дисперсия . . . . .	242
6.4.2. Эмпирические среднее и дисперсия . . . . .	247
6.4.3. Три формулы дисперсии . . . . .	248
6.4.4. Суммы и преобразования случайных величин . . . . .	249
6.4.5. Статистическая независимость . . . . .	250
6.4.6. Скалярные произведения случайных величин . . . . .	251
6.5. Гауссово распределение . . . . .	254
6.5.1. Частные и условные распределения — тоже гауссианы . . . . .	255

6.5.2. Произведение гауссовых плотностей . . . . .	258
6.5.3. Суммы и линейные преобразования . . . . .	259
6.5.4. Семплирование из многомерного гауссова распределения . . . . .	262
6.6. Сопряженность и экспоненциальное семейство распределений . . . . .	263
6.6.1. Сопряженность . . . . .	266
6.6.2. Достаточные статистики . . . . .	269
6.6.3. Экспоненциальное семейство распределений . . . . .	270
6.7. Замена переменных / Обратное преобразование . . . . .	274
6.7.1. Метод функций распределения . . . . .	275
6.7.2. Замена переменных . . . . .	277
6.8. Для дальнейшего чтения . . . . .	282
Упражнения . . . . .	283
<b>Глава 7. Непрерывная оптимизация . . . . .</b>	<b>287</b>
7.1. Оптимизация с использованием градиентного спуска . . . . .	290
7.1.1. Размер шага . . . . .	293
7.1.2. Градиентный спуск с импульсом . . . . .	294
7.1.3. Стохастический градиентный спуск . . . . .	295
7.2. Ограниченнная оптимизация и множители Лагранжа . . . . .	297
7.3. Выпуклая оптимизация . . . . .	300
7.3.1. Линейное программирование . . . . .	305
7.3.2. Квадратичное программирование . . . . .	307
7.3.3. Преобразование Лежандра — Фенхеля и выпуклое сопряжение . .	308
7.4. Для дальнейшего чтения . . . . .	313
Упражнения . . . . .	314

## **ЧАСТЬ II**

### **Главные задачи машинного обучения**

<b>Глава 8. О сочетании модели и данных . . . . .</b>	<b>318</b>
8.1. Данные, модели и обучение . . . . .	318
8.1.1. Данные как векторы . . . . .	319
8.1.2. Модели как функции . . . . .	323
8.1.3. Модели как вероятностные распределения . . . . .	324
8.1.4. Обучение — это нахождение параметров . . . . .	325

8.2. Минимизация эмпирического риска .....	327
8.2.1. Гипотеза класса функций .....	328
8.2.2. Функция потерь для обучения.....	329
8.2.3. Регуляризация для борьбы с переобучением .....	331
8.2.4. Кросс-валидация для оценки производительности обобщения .....	333
8.2.5. Дальнейшее чтение .....	335
8.3. Оценка параметров .....	336
8.3.1. Метод максимального правдоподобия .....	336
8.3.2. Оценка апостериорного максимума.....	339
8.3.3. Обучение модели .....	341
8.3.4. Дополнительное чтение .....	344
8.4. Вероятностные модели и инференс .....	345
8.4.1. Вероятностные модели .....	345
8.4.2. Байесовский инференс .....	346
8.4.3. Модели латентных переменных .....	348
8.4.4. Дальнейшее чтение .....	350
8.5. Направленные графические модели .....	351
8.5.1. Семантика графов .....	352
8.5.2. Условная независимость и d-разбиение .....	355
8.5.3. Дальнейшее чтение .....	357
8.6. Выбор модели .....	358
8.6.1. Вложенная кросс-валидация.....	358
8.6.2. Выбор байесовской модели .....	359
8.6.3. Коэффициент Байеса для сравнения моделей .....	362
8.6.4. Дальнейшее чтение .....	364
<b>Глава 9. Линейная регрессия .....</b>	<b>366</b>
9.1. Постановка задачи .....	368
9.2. Оценка параметров .....	370
9.2.1. Оценка максимального правдоподобия .....	371
9.2.2. Переобучение при линейной регрессии .....	377
9.2.3. Оценка апостериорного максимума.....	380
9.2.4. MAP-оценивание как регуляризация .....	382

9.3. Байесовская линейная регрессия . . . . .	384
9.3.1. Модель . . . . .	384
9.3.2. Априорные предсказания . . . . .	385
9.3.3. Апостериорное распределение . . . . .	387
9.3.4. Апостериорные предсказания . . . . .	389
9.3.5. Вычисление маргинального правдоподобия . . . . .	393
9.4. Максимальное правдоподобие как ортогональная проекция . . . . .	395
9.5. Для дальнейшего чтения . . . . .	397
<b>Глава 10. Снижение размерности с помощью анализа главных компонент . . . . .</b>	<b>400</b>
10.1. Постановка проблемы . . . . .	401
10.2. Перспектива максимальной дисперсии . . . . .	404
10.2.1. Направление с максимальной дисперсией . . . . .	405
10.2.2. М-мерное подпространство с максимальной дисперсией . . . . .	407
10.3. Проекционная перспектива . . . . .	411
10.3.1. Настройка и цели . . . . .	411
10.3.2. Поиск оптимальных координат . . . . .	413
10.3.3. Нахождение базиса главного подпространства . . . . .	415
10.4. Вычисление собственного вектора и приближения низкого ранга . . . . .	419
10.4.1. PCA с использованием матричных приближений низкого ранга . .	420
10.4.2. Практические аспекты . . . . .	421
10.5. PCA в больших размерах . . . . .	422
10.6. Ключевые шаги PCA на практике . . . . .	424
10.7. Латентная переменная . . . . .	427
10.7.1. Генеративный процесс и вероятностная модель . . . . .	428
10.7.2. Правдоподобие и совместное распределение . . . . .	430
10.7.3. Апостериорное распределение . . . . .	431
10.8. Дополнительное чтение . . . . .	432
<b>Глава 11. Оценка плотности с помощью моделей гауссовой смеси . . . . .</b>	<b>438</b>
11.1. Модель гауссовой смеси . . . . .	440
11.2. Изучение параметров с помощью максимального правдоподобия . . . . .	440
11.2.1. Ответственность . . . . .	443
11.2.2. Обновление средних . . . . .	444

11.2.3. Обновление ковариаций .....	447
11.2.4. Обновление весов смеси .....	450
11.3. EM-алгоритм .....	453
11.4. Скрытая перспектива .....	456
11.4.1. Генеративный процесс и вероятностная модель .....	456
11.4.2. Правдоподобие .....	458
11.4.3. Апостериорное распределение.....	459
11.4.4. Расширение до полного набора данных .....	459
11.4.5. Еще раз про EM-алгоритм.....	461
11.5. Дополнительное чтение .....	461
 Глава 12. Классификация методом опорных векторов .....	464
12.1. Разделяющие гиперплоскости .....	466
12.2. Прямая задача метода опорных векторов .....	468
12.2.1. Понятие зазора .....	469
12.2.2. Нахождение зазора: традиционный способ.....	471
12.2.3. Почему можно взять зазор, равный 1 .....	473
12.2.4. SVM с мягким зазором: геометрический подход .....	474
12.2.5. SVM с мягким зазором: подход с использованием функции потерь ..	476
12.3. Двойственная задача SVM .....	479
12.3.1. Двойственность и множители Лагранжа .....	479
12.3.2. Двойственность и выпуклая оболочка .....	482
12.4. Ядра .....	484
12.5. Численное решение .....	487
12.6. Для дальнейшего чтения .....	489
 Библиография .....	492

## ОТ ИЗДАТЕЛЬСТВА

Ваши замечания, предложения, вопросы отправляйте по адресу [comp@piter.com](mailto:comp@piter.com) (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

На веб-сайте издательства [www.piter.com](http://www.piter.com) вы найдете подробную информацию о наших книгах.

Издательство выражает благодарность научному редактору Константину Кнопу за замечания, исправления и указания на неточности перевода, а также подробные консультации в процессе работы над книгой.

## О НАУЧНОМ РЕДАКТОРЕ РУССКОГО ИЗДАНИЯ

Константин Александрович Кноп — преподаватель математики, автор задач и головоломок, член методической комиссии и жюри Всероссийской олимпиады школьников по математике, автор нескольких математических книг. Любимое хобби Константина — игра «Что? Где? Когда?» Он не только играет в ее спортивных турнирах, но и активно участвует в их организации.

# Математика в машинном обучении

[https://t.me/it\\_booooks/2](https://t.me/it_booooks/2)

К фундаментальным математическим дисциплинам, необходимым для понимания машинного обучения (МО), относятся линейная алгебра, аналитическая геометрия, разложение матриц, векторный анализ, оптимизация, теория вероятностей и статистика. Традиционно все эти темы преподаются в различных курсах, и поэтому студентам, изучающим науку о данных (data science) или информатику (computer science), а также профессионалам в МО, сложно как следует осваивать математику.

Эта книга — самодостаточное руководство, она заполняет пробел между учебниками по математике и машинному обучению, знакомит читателя с математическими концепциями почти без предварительной подготовки. Далее на основе этих концепций выводятся четыре основных метода МО: линейная регрессия, метод главных компонент, гауссово моделирование смесей и метод опорных векторов. Для студентов и других читателей с базовым математическим образованием эти направления послужат отправной точкой для изучения руководств по машинному обучению. Тем, кто только начинает изучать математику, эти методы помогут наработать интуицию и практический опыт в применении математических концепций.

В каждой главе имеются рабочие примеры и упражнения, чтобы проверить, насколько усвоен материал. Руководства по программированию выложены на сайте, сопровождающем книгу<sup>1</sup>.

**Марк Питер Дайзенрот** — руководитель DeepMind в области искусственного интеллекта в Университетском колледже Лондона. До этого Марк работал штатным преподавателем в Имперском колледже Лондона. Сфера его научных интересов включает data-эффективное обучение<sup>2</sup>, вероятностное моделирование

---

<sup>1</sup> <https://mml-book.com> (на англ. языке. — Примеч. ред.).

<sup>2</sup> Эффективное обучение без необходимости использования большого количества данных. — Примеч. ред.

и автономное принятие решений. Его исследования удостаивались премии «За лучшую научную работу» на конференциях ICRA 2014 и ICCAS 2016. Марк удостоен Президентской премии для выдающихся молодых исследователей в Имперском колледже Лондона, корпоративной премии для сотрудников Google и гранта на соискание PhD от Microsoft.

**А. Альдо Фейзал** возглавляет лабораторию по изучению мозга и поведения в Имперском колледже Лондона, где наряду с тем является преподавателем на факультетах биоинженерии и вычислительной техники, а также членом института исследования данных. Он руководит Научно-исследовательским и инновационным центром Великобритании (UKRI) (организация с бюджетом 20 миллионов фунтов стерлингов), занимающимся подготовкой докторов наук в области искусственного интеллекта и здравоохранения. Он получил степень PhD в области вычислительной нейрофизиологии в Кембриджском университете и стал младшим научным сотрудником в лаборатории вычислительного и биологического обучения. Сфера его научных интересов располагается на стыке нейрофизиологии и машинного обучения; он стремится понять принципы поведения и работы мозга и выполнить их обратное проектирование.

**Чен Сунь Он** — главный научный сотрудник в Исследовательской группе по машинному обучению (Data61, CSIRO), а также ассоциированный адъюнкт-профессор в Австралийском национальном университете. Сфера его научных интересов связана с созданием условий для научных открытий путем расширенного использования статистических методов машинного обучения. Он получил степень PhD по информатике в Австралийском национальном университете в 2005 году, а также выступал с лекциями на факультете информатики в Швейцарской высшей технической школе Цюриха и в команде диагностической геномики в NICTA, Мельбурн.

# Условные обозначения

Символ	Типичное значение
$a, b, c, \alpha, \beta, \gamma$	Скаляры даются строчными буквами
$\mathbf{x}, \mathbf{y}, \mathbf{z}$	Векторы даются жирным шрифтом строчными буквами
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	Матрицы даются жирным шрифтом заглавными буквами
$\mathbf{x}^T, \mathbf{A}^T$	Вектор или матрица в транспонированном виде
$\mathbf{A}^{-1}$	Обратная матрица
$\langle \mathbf{x}, \mathbf{y} \rangle$	Внутреннее произведение $\mathbf{x}$ и $\mathbf{y}$
$\mathbf{x}^T \mathbf{y}$	Скалярное произведение $\mathbf{x}$ и $\mathbf{y}$
$B = (\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)$	(Упорядоченный) кортеж
$B = [\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3]$	Матрица вектор-столбцов, расположенных горизонтально
$B = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$	(Неупорядоченный) набор векторов
$\mathbb{Z}, \mathbb{N}$	Целые и натуральные числа соответственно
$\mathbb{R}, \mathbb{C}$	Вещественные (действительные) и комплексные числа соответственно
$\mathbb{R}^n$	$n$ -мерное векторное пространство вещественных (действительных) чисел
$\forall x$	Квантор всеобщности: для любого $x$
$\exists x$	Квантор существования: существует (значение) $x$
$a := b$	$a$ определяется как $b$
$a =: b$	$b$ определяется как $a$
$a \propto b$	$a$ пропорционально $b$ , то есть $a =$ константа $\cdot b$
$g \circ f$	Композиция функций: « $g$ после $f$ »
$\Leftrightarrow$	«Тогда и только тогда»
$\Rightarrow$	Импликация («Если..., то...»)
$\mathcal{A}, \mathcal{C}$	Множества
$a \in \mathcal{A}$	$a$ входит в множество $\mathcal{A}$
$\emptyset$	Пустое множество

$D$	Размерность; индексируется от $d = 1, \dots, D$
$N$	Количество точек данных; индексируется по $n = 1, \dots, N$
$I_m$	Единичная матрица размером $m \times m$
$0_{m,n}$	Матрица из нулей размером $m \times n$
$1_{m,n}$	Матрица из единиц размером $m \times n$
$e_i$	Стандартный/канонический вектор (где $i$ — индекс, который равен 1)
$\dim(V)$	Размерность векторного пространства $V$
$\text{rk}(\mathbf{A})$	Ранг матрицы $\mathbf{A}$
$\text{Im}(\Phi)$	Образ линейного отображения $\Phi$
$\ker(\Phi)$	Ядро (нулевое пространство) линейного отображения $\Phi$
$\text{span}[\mathbf{b}_1]$	Линейная оболочка (образующее множество) $\mathbf{b}_1$
$\text{tr}(\mathbf{A})$	След матрицы $\mathbf{A}$
$\det(\mathbf{A})$	Детерминант матрицы $\mathbf{A}$
$ \cdot $	Абсолютное значение или детерминант (в зависимости от контекста)
$\ \cdot\ $	Норма; евклидова, если не указано другое
$\lambda$	Собственное значение или множитель Лагранжа
$E_\lambda$	Собственное подпространство, соответствующее собственному значению $\lambda$
$\theta$	Вектор параметров
$\frac{\partial f}{\partial x}$	Частная производная от $f$ относительно $x$
$\frac{df}{dx}$	Полная производная от $f$ относительно $x$
$\nabla$	Градиент
$\mathcal{L}$	Лагранжиан
$\mathcal{L}$	Отрицательное логарифмическое правдоподобие
$\binom{n}{k}$	Биноминальный коэффициент из $n$ по $k$
$\mathbb{V}_x[\mathbf{x}]$	Дисперсия $\mathbf{x}$ относительно случайной величины $X$
$\mathbb{E}_x[\mathbf{x}]$	Математическое ожидание $\mathbf{x}$ относительно случайной величины $X$
$\text{Cov}_{x,y}[\mathbf{x}, \mathbf{y}]$	Ковариация между $\mathbf{x}$ и $\mathbf{y}$

$X \perp\!\!\!\perp Y   Z$	$X$ условно независим от $Y$ при условии $Z$
$X \sim p$	Задано распределение $p$ случайной величины $X$
$\mathcal{N}(\mu, \Sigma)$	Гауссово распределение со средним $\mu$ и ковариацией $\Sigma$
$Ber(\mu)$	Распределение Бернулли с параметром $\mu$
$Bin(N, \mu)$	Биномиальное распределение с параметрами $N, \mu$
$Beta(\alpha, \beta)$	Бета-распределение с параметрами $\alpha, \beta$

## СПИСОК АББРЕВИАТУР И СОКРАЩЕНИЙ

Сокращение	Значение
BLR	Bayesian linear regression (Байесовская линейная регрессия)
cdf	Cumulative distribution function (Функция распределения)
EM	Expectation maximization (Максимизация математического ожидания)
GMM	Gaussian mixture model (Модель гауссовой смеси)
i.i.d.	Independent, identically distributed (Независимые одинаково распределенные)
MAP	Maximum a posteriori (Оценка апостериорного максимума)
MCMC	Markov chain Monte Carlo (Метод Монте-Карло с использованием марковских цепей)
MDS	Multidimensional scaling (Многомерное масштабирование)
MLE	Maximum likelihood estimation/estimator (Оценка максимального правдоподобия)
ONB	Orthonormal basis (Ортонормированный базис, ОНБ)
PCA	Principal component analysis (Анализ главных компонент)
pdf	Probability density function (Плотность вероятности)
PPCA	Probabilistic principal component analysis (Вероятностный анализ главных компонент)
REF	Row-echelon form (Ступенчатый вид матрицы)
RMSE	Root mean square error (Среднеквадратичная ошибка)
SGD	Stochastic gradient descent (Стохастический градиентный спуск)
SPD	Symmetric, positive definite (Симметричная, положительно определенная (матрица))
SVD	Singular value decomposition (Сингулярное разложение)
SVM	Support vector machine (Метод опорных векторов)

# Предисловие

Машинное обучение — это наиболее свежая попытка в ряду многих, призванных перевести человеческие знания и суждения в форму, которая позволит конструировать машины и проектировать автоматизированные системы для обработки таких знаний и суждений. По мере того как машинное обучение распространяется все шире, а программные пакеты для его реализации становятся все проще в использовании, естественно и желательно абстрагироваться от низкоуровневых технических деталей, чтобы не отвлекать на них практикующего специалиста. Однако в таком случае возникает опасность, что практик перестанет ориентироваться в решениях, принимаемых системой, и этот фактор ограничивает применимость алгоритмов МО.

Практик-энтузиаст, нацеленный на то, чтобы подробнее изучить, казалось бы, магическую подоплеку успешных алгоритмов МО, сегодня вынужден усвоить ошеломительный объем знаний, прежде чем приступить к работе:

- Языки программирования и инструменты анализа данных.
- Крупномасштабные вычисления и связанные с ними фреймворки.
- Математику и статистику, чтобы понимать, как на их основе строится машинное обучение.

В вводных курсах по МО, преподаваемых в университетах, первая часть курса уходит на рассмотрение некоторых из этих предварительных тем. Исторически сложилось, что курсы по МО обычно преподаются на факультетах информатики, где студенты получают знания по двум первым группам дисциплин из вышеперечисленных, но не так глубоко изучают математику и статистику.

В современных учебниках по МО основное внимание уделяется алгоритмам и методологиям МО; предполагается, что читатель компетентен в математике и статистике. Таким образом, в подобных книгах одна-две главы отводятся на базовую математику: они либо располагаются в начале книги, либо выносятся в приложения. Мы обнаружили, что многим людям, желающим докопаться до

основ базовых методов машинного обучения, не хватает знаний в области математики, чтобы читать учебник по МО. Преподавая в университете курсы для студентов и аспирантов, мы пришли к выводу, что разрыв между университетским курсом математики и уровнем математики, который нужен для изучения стандартной книги по МО, для многих слишком велик.

В этой книге на передний план выводятся математические основы базовых концепций машинного обучения; в ней также собрана вся информация, необходимая, чтобы уменьшить или даже полностью устраниить этот пробел в знаниях.

## ЗАЧЕМ НУЖНА ЕЩЕ ОДНА КНИГА ПО МАШИННОМУ ОБУЧЕНИЮ?

Машинное обучение строится на математическом языке, позволяющем выразить концепции, которые интуитивно кажутся очевидными, но удивительно сложно поддаются формализации. Но стоит формализовать их как следует — и начинают просматриваться решения для стоящей перед нами задачи. Студенты-математики во всем мире жалуются, что темы из курса математики кажутся оторванными от практических проблем. Мы полагаем, что машинное обучение — явный и прямой стимул учить математику для всех, кто ей интересуется.

Эта книга призвана стать путеводителем по обширной математической литературе, образующей базис современного машинного обучения. Мы объясняем необходимость тех или иных математических концепций, прямо указывая на их пользу в контексте фундаментальных проблем МО. Стремясь добиться краткости книги, мы опустили множество деталей и сравнительно продвинутых концепций. Читатель, вооружившись изложенными здесь базовыми концепциями и понимая, как они вписываются в более широкий контекст машинного обучения, найдет многочисленные ресурсы для дальнейшего изучения; списки таких источников мы приводим в конце каждой главы. Для читателей, уже обладающих математическим бэкграундом, эта книга предоставит краткий, но четко сформулированный обзор МО. В отличие от других книг, где основное внимание уделяется методам и моделям МО (MacKay, 2003; Bishop, 2006; Alpaydin, 2010; Murphy 2012; Barber, 2012; Shalev-Shwartz and Ben-David, 2014; Rogers and Girolami, 2016) или программным аспектам МО (Muller and Guido, 2016; Raschka and Mirjalili, 2017; Chollet and Allaire, 2018)<sup>1</sup>, мы ограничились представлением лишь четырех репрезентативных примеров алгоритмов МО, сосредоточившись на математических концепциях, лежащих в основе моделей

<sup>1</sup> Список упоминаемых в тексте изданий вы найдете в конце книги в разделе «Библиография».

как таковых. Надеемся, что читатели смогут глубже разобраться в базовых вопросах машинного обучения и связать практические вопросы, возникающие при использовании МО, с фундаментальным выбором математической модели.

Мы не стремились написать классическую книгу по машинному обучению. Вместо этого мы намеревались предоставить математический бэкграунд, применимый для решения четырех центральных задач МО, чтобы читателю впоследствии было проще изучать другие книги по этой теме.

## КАКОВА ЦЕЛЕВАЯ АУДИТОРИЯ КНИГИ?

Поскольку машинное обучение распространяется в обществе все шире, мы считаем, что каждый должен иметь хотя бы некоторое представление о его базовых принципах. Эта книга выдержана в академическом математическом стиле, что позволяет нам в точности описывать концепции, на которых основана МО. Читателям, не знакомым с таким, казалось бы, сухим стилем, мы рекомендуем проявить выдержку и держать в уме цель каждой рассматриваемой темы. По всему тексту рассыпаны наши комментарии и ремарки. Мы надеемся, что они послужат полезными маячками и помогут сориентироваться в общей картине.

*Предполагается, что читатель обладает математическими знаниями, обычно преподаваемыми на уроках математики и физики в старших классах.* Например, читатель должен иметь представление о производных и интегралах, а также о геометрических двумерных и трехмерных векторах. Отталкиваясь от данного уровня, мы обобщаем эти концепции. Следовательно, к целевой аудитории данной книги относятся студенты университетов, учащиеся вечерних курсов, слушатели онлайн-курсов по машинному обучению.

Проводя аналогию с музыкой, можно сказать, что человек может взаимодействовать с МО тремя способами:

**Заинтересованный слушатель.** Демократизация МО, связанная с предоставлением программного обеспечения с открытым исходным кодом для решения задач из этой области, с появлением онлайн-курсов и облачных инструментов, позволяет пользователю не вдаваться в специфику работы пайплайнов МО. Пользователь может сосредоточиться на извлечении полезной информации из данных, оперируя готовыми инструментами. Поэтому МО может пригодиться экспертам из разных предметных областей, не слишком подкованным в его технической составляющей. Ситуация подобна прослушиванию музыки: пользователь может выбирать и различать те или иные типы МО и извлекать из них что-то ценное для себя. Более опытных пользователей можно сравнить с музыкальными критиками, умеющими задавать важные вопросы о применении МО,

касающиеся, например, этики, непредвзятости и неприкосновенности частной жизни индивида. Мы надеемся, что эта книга заложит основы, которые помогут задуматься о сертификации и управлении рисками при использовании систем МО и позволит экспертам из разных предметных областей, опираясь на профессиональный опыт, создавать качественные системы машинного обучения с учетом своей специализации.

**Опытный исполнитель.** Умелые специалисты, использующие машинное обучение на практике, могут подключать различные инструменты и библиотеки в аналитический пайплайн. Типичный практик такого рода — это дата-сайентист или инженер, понимающий интерфейсы МО, примеры их использования и умеющий выдавать великолепные прогнозы, опираясь на данные. Его можно сравнить с исполнителем-виртуозом: напрактиковавшись, такой музыкант оживляет инструмент, и его с удовольствием слушают. Воспользовавшись представленной здесь математикой как букварем, практик сможет понять пользу и ограничения любимого метода, а также расширить и обобщить существующие алгоритмы МО. Надеемся, что эта книга даст такому специалисту стимул придерживаться более строгой и последовательной разработки методов машинного обучения.

**Начинающий композитор.** По мере того как машинное обучение проникает в новые предметные области, разработчикам МО приходится изобретать новые методы и расширять существующие алгоритмы. Зачастую такой работой занимаются исследователи, которым необходимо понимать математический базис МО и вскрывать взаимосвязи между различными задачами. Эта работа напоминает творчество композитора, который, руководствуясь структурой и правилами теории музыки, создает новые удивительные пьесы. Надеемся, что эта книга послужит высокоуровневым обзором других технических книг для тех читателей, которые хотят заниматься МО, как композиторы — музыкой. Общество весьма нуждается в новых исследователях, способных предлагать и рассматривать новаторские подходы, позволяющие подступиться ко многим задачам, связанным с извлечением полезной информации из данных.

# Благодарности

Мы благодарны множеству коллег, которые ознакомились с первыми черновиками книги и смогли вынести выстраданное нами изложение концепций. Мы постарались вплести в текст все предложенные ими идеи, кроме тех, с которыми были категорически не согласны. Мы хотели бы выразить особую благодарность Кристфриду Веберсу (Christfried Webers) за то, как тщательно он вычитал многие разделы книги, и за его подробные рекомендации по структурированию и представлению материала. Многие наши друзья и коллеги также любезно уделили нам время и силы, помогая работать над разными версиями каждой главы. Мы были счастливы воспользоваться и великодушием онлайн-сообщества [github.com](https://github.com), где нам подсказывали, как улучшить книгу. Это весьма пошло ей на пользу.

Ниже перечислены люди, которые нашли в тексте ошибки, подсказали, как пояснить материал, и предложили нам релевантную литературу — либо через [github.com](https://github.com), либо при личном общении. Они перечислены в алфавитном порядке<sup>1</sup>.

Абдул-Ганий Усман (Abdul-Ganiy Usman)	Арег Сарвазян (Areg Sarvazyan)
Адам Гайер (Adam Gaier)	Артем Артемьев (Artem Artemev)
Адель Джексон (Adele Jackson)	Артем Степанов (Artyom Stepanov)
Адityа Менон (Aditya Menon)	Билл Кромидас (Bill Kromidas)
Аласдер Трэн (Alasdair Tran)	Боб Уильямсон (Bob Williamson)
Александар Крняич (Aleksandar Krnjaic)	Бун Пинь Лим (Boon Ping Lim)
Александр Макригиоргос (Alexander Makrigiorgos)	Чао Ку (Chao Qu)
Альфредо Канзиани (Alfredo Canziani)	Чень Ли (Cheng Li)
Али Шафти (Ali Shafti)	Крис Шерлок (Chris Sherlock)
Амр Халифа (Amr Khalifa)	Кристофер Грей (Christopher Gray)
Эндрю Тангтара (Andrew Tanggara)	Дэниел Макнамара (Daniel McNamara)
Ангус Грюн (Angus Gruen)	Дэниел Вуд (Daniel Wood)
Анталь А. Басс (Antal A. Buss)	Даррен Сиджел (Darren Siegel)
Антуан Туазул ле Канн (Antoine Toisoul Le Cann)	Дэвид Джонстон (David Johnston)
	Давей Чен (Dawei Chen)
	Эллен Броуд (Ellen Broad)

<sup>1</sup> Согласно английскому алфавиту. — Примеч. ред.

Фэнъкуангтян Чжу (Fengkuangtian Zhu)  
Фиона Кондон (Fiona Condon)  
Георгиос Теодоропу (Georgios Theodorou)  
Хэ Синь (He Xin)  
Ирен Раисса Камени (Irene Raissa Kameni)  
Якуб Набагло (Jakub Nabaglo)  
Джеймс Хенсман (James Hensman)  
Джейми Лю (Jamie Liu)  
Жан Каддур (Jean Kaddour)  
Жан-Поль Эбеджер (Jean-Paul Ebejer)  
Джерри Цян (Jerry Qiang)  
Джитеш Синхар (Jitesh Sindhare)  
Джон Ллойд (John Lloyd)  
Джонас Нгнаве (Jonas Nganawe)  
Джон Мартин (Jon Martin)  
Джастин Хси (Justin Hsi)  
Кай Арулкумаран (Kai Arulkumaran)  
Камиль Дрецковски (Kamil Dreczkowski)  
Лили Уонг (Lily Wang)  
Лионель Тонджий Нгоупей (Lionel Tondji Nguerpeou)  
Лидия Кнюфинг (Lydia Knüfing)  
Махмуд Аслан (Mahmoud Aslan)  
Марк Хартенстейн (Mark Hartenstein)  
Марк ван дер Вильк (Mark van der Wilk)  
Маркус Хегланд (Markus Hegland)  
Мартин Хевинг (Martin Hewing)  
Мэттью Элджер (Matthew Alger)  
Мэттью Ли (Matthew Lee)  
Максимус Макканн (Maximus McCann)  
Мэньянь Чжан (Mengyan Zhang)  
Майкл Беннетт (Michael Bennett)  
Майкл Педерсен (Michael Pedersen)  
Минджон Шин (Minjeong Shin)  
Мохаммад Малекзаде (Mohammad Malekzadeh)  
Навин Кумар (Naveen Kumar)  
Нико Монтали (Nico Montali)  
Оскар Армас (Oscar Armas)  
Патрик Хенриксен (Patrick Henriksen)  
Патрик Вишоллек (Patrick Wieschollek)  
Паттрават Чормай (Pattarawat Chormai)  
Пол Келли (Paul Kelly)  
Петрос Христодулу (Petros Christodoulou)

Петр Янушевски (Piotr Januszewski)  
Пранав Субрамани (Pranav Subramani)  
Цюйюй Конг (Quyu Kong)  
Рагиб Заман (Ragib Zaman)  
Руй Чжан (Rui Zhang)  
Райан-Риз Гриффитс (Ryan-Rhys Griffiths)  
Саломон Кабонго (Salomon Kabongo)  
Сэмюэл Огунмола (Samuel Ogunmola)  
Сандип Мавадья (Sandeep Mavadia)  
Сарвеш Никумбх (Sarvesh Nikumbh)  
Себастьян Рашка (Sebastian Raschka)  
Сенанаяк Сеш Кумар Карри (Senanayak Sesh Kumar Karri)  
Сеунг-Хеон Бек (Seung-Heon Baek)  
Шахбаз Чaudхари (Shahbaz Chaudhary)  
Шакир Мохамед (Shakir Mohamed)  
Шон Берри (Shawn Berry)  
Шейх Абдул Рахим Али (Sheikh Abdul Raheem Ali)  
Шэнь Сюэ (Sheng Xue)  
Сридхар Тяягараджан (Sridhar Thiagarajan)  
Сеид Нуман Хасани (Syed Nouman Hasany)  
Шимон Брых (Szymon Brych)  
Томас Бюлер (Thomas Bühler)  
Тимур Шарапов (Timur Sharapov)  
Том Меламед (Tom Melamed)  
Винсент Адам (Vincent Adam)  
Винсент Диотордуар (Vincent Dutordoir)  
Ву Минь (Vu Minh)  
Васим Афтаб (Wasim Aftab)  
Вэн Чжи (Wen Zhi)  
Войцех Стоковиц (Wojciech Stokowiec)  
Сяонан Чонг (Xiaonan Chong)  
Сяовэй Чжан (Xiaowei Zhang)  
Ячжу Хао (Yazhou Hao)  
Ичэн Люо (Yicheng Luo)  
Янг Ли (Young Lee)  
Ю Лю (Yu Lu)  
Юнь Чен (Yun Cheng)  
Сяоюй Хуань (Yuxiao Huang)  
Зак Кранко (Zac Cranko)  
Цзыцзян КАО (Zijian Cao)  
Зои Нолан (Zoe Nolan)

Контрибьюторы с Github, не указавшие в профиле своих реальных имен:

SamDataMad	cs-maillist
bumptiousmonkey	kudo23
idoamihai	empet
deepakiim	victorBigand
insad	17SKYE
HorizonP	jessjing1995

Мы также благодарны Парамесварану Раману (Parameswaran Raman) и множеству анонимных рецензентов, помогавших нам по просьбе издательства Cambridge University Press; они вычитали по одной или более глав, предоставили конструктивную критику, которая позволила нам значительно улучшить текст. Особого упоминания заслуживает Динеш Сингх Неги (Dinesh Singh Negi), помогавший нам в работе с LATEX, подробно и оперативно консультируя нас по всем вопросам, связанным с этим редактором. Наконец, но не в последнюю очередь, мы благодарим нашего редактора Лорен Коулз (Lauren Cowles), которая выступила терпеливой повитухой этой книги.

# ЧАСТЬ I

## Математические основы

# 1

## Введение и мотивация

[https://t.me/it\\_boooks/2](https://t.me/it_boooks/2)

Машинное обучение — это дисциплина, посвященная разработке алгоритмов, которые автоматически извлекают ценную информацию из имеющихся данных. В данном случае акцент следует сделать на «автоматически», то есть машинное обучение связано с методологиями общего характера, которые применимы к разнообразным наборам данных (датасетам) и позволяют извлечь из них нечто осмысленное. Суть машинного обучения можно описать в виде трех концепций: «данные», «модель» и «обучение».

Поскольку МО по сути своей ориентировано на работу с данными, *данные* являются основной из его ключевых составляющих. Цель МО — проектирование методологий общего характера, позволяющих извлекать ценные закономерности из данных; в идеале для этого почти не должен требоваться опыт работы в предметной области. Например, имея большой корпус документов (скажем, книг из множества библиотек) можно, воспользовавшись методами МО, автоматически найти релевантные темы, общие для множества документов (Hoffman et al., 2010). Для достижения этой цели проектируются *модели*, как правило, связанные с процессом генерации данных, аналогичных датасету, который мы получили. Например, при выполнении регрессии модель будет описывать функцию, которая сопоставляет входные данные с реальными выходными значениями. Перефразируя Митчелла (Mitchell, 1997): «Говорят, что модель обучается на данных, если ее производительность на определенной задаче увеличивается при учете вышеупомянутых данных». Цель — найти эффективные модели, хорошо обобщающие незнакомые данные, которые могут понадобиться нам в будущем. *Обучение* можно понимать как способ автоматического выявления закономерностей и структуры в данных путем оптимизации параметров модели.

Несмотря на то что известно множество историй успеха, связанных с МО, и можно без труда найти программное обеспечение, предназначенное для проектирования и обучения многофункциональных и гибких систем такого рода,

мы считаем, что математические основы МО важны для понимания фундаментальных принципов, на основе которых строятся более сложные системы. Понимание этих принципов способствует созданию новых решений для МО, пониманию и отладке имеющихся подходов, а также изучению неотъемлемых допущений и ограничений тех методологий, с которыми мы работаем.

## 1.1. ПОИСК ИНТУИТИВНО ПОНЯТНЫХ ФОРМУЛИРОВОК

В машинном обучении мы постоянно сталкиваемся с ситуациями, в которых смысл концепций и слов как бы ускользает, а конкретный компонент системы МО может быть абстрагирован до математических концепций, в которые вкладывается разный смысл. Например, в контексте МО есть два понимания термина «алгоритм». В первом смысле формулировка «алгоритм машинного обучения» означает систему, делающую прогнозы на основании входных данных. Такие алгоритмы называются *предикторами*. Во втором смысле ровно та же фраза, «алгоритм машинного обучения», означает систему, которая адаптирует некоторые внутренние параметры предиктора таким образом, чтобы он хорошо работал на еще не известных данных, которые поступят в будущем. Такая адаптация будет называться *обучением* системы.

Эта книга не решает проблем, связанных с подобной двойственностью, но мы хотим заранее подчеркнуть, что в зависимости от контекста одни и те же выражения могут означать разные вещи. Тем не менее, мы стараемся во всех случаях давать достаточно ясный контекст, чтобы по возможности устраниć неоднозначность.

Первая часть книги знакомит читателя с математическими концепциями и основами, необходимыми, чтобы говорить о трех основных компонентах системы машинного обучения: данных, моделях и обучении. Здесь мы кратко обрисуем эти концепции, а затем вновь обратимся к ним в главе 8, после того как будут рассмотрены необходимые математические понятия.

Хотя данные бывают не только числовыми, часто полезно трактовать их именно в числовом формате. В этой книге предполагается, что *данные* уже были требуемым образом преобразованы в числовые представления, подходящие для считывания компьютерной программой. Следовательно, мы понимаем данные как векторы. Еще один пример, иллюстрирующий ненадежность слов, заключается в том, что существует (как минимум) три разных понимания вектора: вектор как массив чисел (трактовка из информатики), вектор как стрелка, у которой есть направление и величина (трактовка из физики), и вектор как объект, поддающийся сложению и умножению (трактовка из математики).

*Модель*, как правило, используется для описания процесса генерации данных, подобных тем, что содержатся в имеющемся датасете. Следовательно, хорошие модели также можно трактовать как упрощенные версии реального (неизвестного) процесса генерации данных, схватывающие особенности, которые важны для моделирования данных и извлечения из них скрытых закономерностей. В дальнейшем хорошая модель может применяться для прогнозирования того, что произойдет в реальных условиях, без постановки соответствующих реальных экспериментов.

Теперь мы подходим к апогею нашей темы: собственно *обучению* в рамках МО. Допустим, у нас есть датасет и подходящая модель. *Обучение* модели означает использование доступных данных для оптимизации некоторых параметров модели с учетом функции полезности, оценивающей, насколько хорошо модель прогнозирует обучающие данные. Большинство методов обучения можно понимать как попытки взобраться на вершину холма. В данной аналогии вершина холма соответствует максимуму некоторого показателя желаемой производительности. Но на практике мы заинтересованы, чтобы модель хорошо работала на тех данных, которых ей еще не показывали. Хорошая производительность модели на уже показанных ей (обучающих) данных может означать лишь то, что мы нашли хороший способ запоминания этих данных. Однако может оказаться, что модель плохо обобщает заранее не известные данные, а на практике зачастую приходится задействовать нашу модель МО в таких ситуациях, с которыми она ранее не сталкивалась.

Итак, обобщим основные концепции машинного обучения, о которых пойдет речь в этой книге:

- Мы представляем данные в виде векторов.
- Мы выбираем подходящую модель, исходя из вероятностной или оптимизационной точки зрения.
- Мы учимся на доступных данных, используя методы численной оптимизации, и стремимся добиться того, чтобы модель хорошо работала на данных, не применявшимися для ее обучения.

## 1.2. ДВА СПОСОБА ЧИТАТЬ ЭТУ КНИГУ

Можно рассмотреть две стратегии, помогающие понять математику в рамках машинного обучения:

- **Восходящая:** опираясь на основополагающие концепции, изучаем все более продвинутые. Такой подход зачастую предпочтителен в точных науках, например в математике. Преимущество такой стратегии в том, что читатель в любой момент может обратиться к концепциям, изученным ранее. К со-

жалению, для практика основополагающие концепции не столь интересны сами по себе, и из-за отсутствия мотивации их изучать большинство определений таких базовых концепций быстро забываются.

- **Нисходящая:** конкретизация, сведение практических потребностей к более базовым требованиям. Такой целеориентированный подход хорош тем, что читателю в любой момент ясно, зачем нужно прорабатывать конкретную концепцию, и к нужным знаниям лежит ясный путь. Недостаток такой стратегии заключается в том, что основы таких знаний могут получиться шаткими, и читателю приходится запоминать набор слов, понять которые у него нет никакой возможности.

Мы решили написать эту книгу в виде системы модулей, чтобы отделить базовые (математические) концепции от прикладных, и текст можно было читать обоими вышеупомянутыми способами. Книга разделена на две части. В части I излагаются математические основы, а в части II концепции из части I применяются для решения набора фундаментальных задач машинного обучения, показанных на рис. 1.1: это регрессия, снижение размерности, оценка плотности и классификация. Последующие главы в части I в основном базируются на предыдущих, но при необходимости можно пропустить главу, прочитать следующую, а затем вернуться к пропущенной, если потребуется. Главы в части II связаны очень слабо, и их можно читать в любом порядке. В обоих частях книги расставлено множество отсылок на предыдущие и последующие главы, они связывают математические концепции с алгоритмами машинного обучения.

*Разумеется, найдутся и другие способы чтения этой книги, кроме двух вышеупомянутых.* Большинству читателей подойдет комбинация восходящего и нисходящего подхода. В некоторых случаях потребуется наработать базовые математические навыки, прежде чем подступаться к сложным концепциям, но также можно выбирать темы, отталкиваясь от возможностей применения машинного обучения.

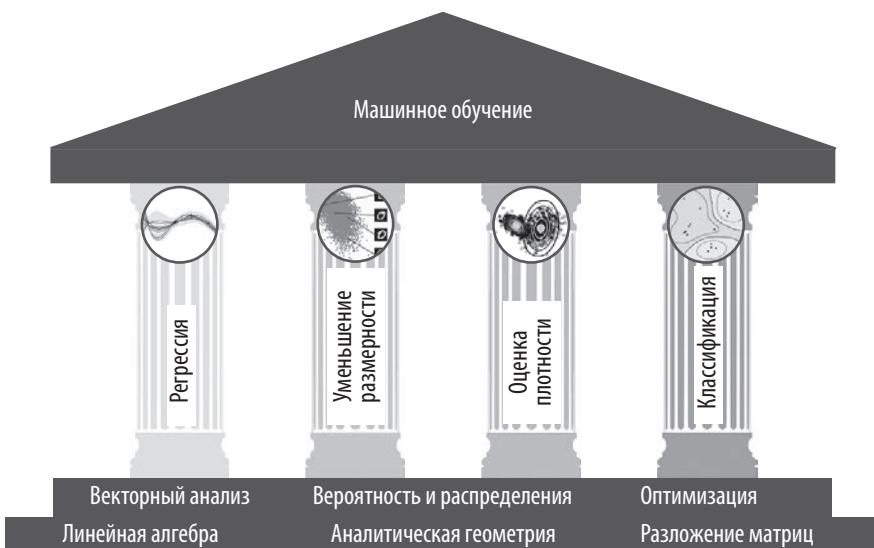
## Часть I — о математике

Четыре столпа машинного обучения, рассматриваемые в этой книге (рис 1.1), требуют основательного математического базиса, и этот базис изложен в части I.

Мы представляем числовые данные в векторном виде, а таблицу таких данных — как матрицу. Изучение векторов и матриц называется *линейной алгеброй*, с ней мы познакомимся в главе 2. Там же матрица описывается как совокупность векторов.

Имея два вектора, представляющих два реальных объекта, мы собираемся делать утверждения об их сходстве. Идея в том, что два схожих вектора должны давать

схожие выводы после обработки нашим алгоритмом МО (предиктором). Чтобы формализовать идею сходства векторов, необходимо ввести операции, принимающие два вектора в качестве ввода и возвращающие числовое значение, отражающее их сходство. Создание сходства и расстояний играет центральную роль в *аналитической геометрии*, рассматриваемой в главе 3.



**Рис. 1.1.** Основания и четыре столпа машинного обучения

В главе 4 вводятся некоторые фундаментальные концепции, касающиеся матриц и их *разложения*. Некоторые операции над матрицами исключительно полезны в МО и обеспечивают интуитивно понятную интерпретацию данных, а также более эффективное обучение.

Часто данные трактуются как зашумленные наблюдения некого истинного базового сигнала. Мы надеемся, что, применив МО, сможем вычленить сигнал из шума. Для этого нам нужен язык, на котором можно было бы количественно выразить, что такое «шум». Часто нам также хотелось бы иметь предикторы, которые позволили бы выразить некоторую неопределенность, например чтобы количественно охарактеризовать степень нашей уверенности в спрогнозированном значении в конкретной точке тестовых данных. *Квантификацией* (количественной оценкой) *неопределенности* занимается теория вероятности, ей посвящена глава 6.

Для обучения моделей обычно подыскиваются параметры, доводящие до максимума некоторую меру производительности. Многие оптимизационные при-

емы требуют понимания концепции градиента, который указывает, в каком направлении искать решение. Глава 5 посвящена *векторному анализу* и подробно описывает концепцию градиентов, которыми мы затем воспользуемся в главе 7; в ней же мы поговорим об *оптимизации* для поиска максимумов и минимумов функций.

## Часть II — о машинном обучении

Во второй части книги вы познакомитесь с *четырьмя столпами машинного обучения*, показанными на рис. 1.1. Мы проиллюстрируем, как математические концепции, объясненные в первой части книги, служат основаниями каждого из столпов. В широком смысле, главы упорядочены от простого к сложному.

В главе 8 мы освежим в памяти три компонента МО (данные, модели, оценка параметров), рассмотрев их с математической точки зрения. Кроме того, мы дадим некоторые рекомендации о том, как подбираются экспериментальные установки, помогающие перестраховаться от чрезмерно оптимистичной оценки систем МО. Как вы помните, наша цель — построить предиктор, хорошо работающий на ранее не известных данных.

В главе 9 мы подробно рассмотрим *линейную регрессию* и зададимся целью найти такие функции, которые сопоставляют входные данные  $\mathbf{x} \in \mathbb{R}^D$  с соответствующими наблюдаемыми значениями функции  $y \in \mathbb{R}$ , которые мы сможем интерпретировать как метки соответствующих входных данных. Мы обсудим классическую подгонку модели (оценку параметров) с применением методов максимального правдоподобия и максимальной апостериорной оценки, а также байесовскую линейную регрессию, где будем исключать параметры путем интегрирования, а не оптимизировать их.

В главе 10 основное внимание уделяется *снижению размерности*, второму столпу с рис. 1.1; для этого воспользуемся анализом главных компонент. Ключевая цель снижения размерности — найти компактное представление (с малым количеством измерений) для данных, описываемых большим количеством измерений  $\mathbf{x} \in \mathbb{R}^D$ ; такое компактное представление зачастую легче анализировать, нежели исходные данные. В отличие от регрессии, снижение размерности зависит только от моделирования данных — нет никаких меток, которые были бы связаны с точкой данных  $\mathbf{x}$ .

В главе 11 мы перейдем к нашему третьему столпу: *оценке плотности*. Назначение оценки плотности — найти вероятностное распределение, описывающее заданный датасет. Изучая эту тему, мы сосредоточимся на моделях гауссовых смесей и обсудим итеративную схему нахождения параметров такой модели. Как и в случае со снижением размерности, здесь нет никаких меток для точек данных  $\mathbf{x} \in \mathbb{R}^D$ . Однако мы не ищем такое представление данных, которое об-

ладало бы низкой размерностью. Нас скорее интересует плотностная модель, которая описывает эти данные.

Глава 12, завершающая книгу, содержит углубленное обсуждение четвертого столпа: *классификации*. Примерно как и при работе с регрессией (глава 9), у нас есть входные значения  $\mathbf{x}$  и соответствующие им метки  $y$ . Однако, в отличие от регрессии, при которой метки имеют вещественные значения, метки при классификации являются целочисленными, поэтому обращаться с ними нужно особенно осторожно.

### 1.3. УПРАЖНЕНИЯ И ОБРАТНАЯ СВЯЗЬ

Мы приводим некоторые упражнения в части I, и для выполнения большинства из них достаточно ручки и бумаги. К главе II мы подготовили руководства по программированию (это блокноты Jupyter), которые помогут вам исследовать некоторые свойства алгоритмов машинного обучения, рассматриваемых в этой книге.

Мы высоко ценим участие издательства Cambridge University Press, активно поддерживающего нас в нашем стремлении к демократизации образования. Издательство выложило эту книгу для свободного скачивания по адресу

<https://mml-book.com>,

где также находятся решения упражнений, списки найденных ошибок и дополнительные материалы. Сообщать об ошибках и оставлять отзывы можно по вышеупомянутой ссылке.

# 2

## Линейная алгебра

При формализации интуитивно понятных концепций принято подбирать набор объектов (символов) и давать набор правил по обращению с этими объектами. Такая наука называется *алгеброй*. Линейная алгебра — это наука о векторах и определенных правилах операций над ними. Векторы, которые многие помнят из школьного курса, называются «геометрическими» и обычно обозначаются стрелочкой над буквой, например,  $\vec{x}$  и  $\vec{y}$ . В этой книге речь пойдет о более обобщенном представлении векторов, и вектор будет обозначаться жирной латинской буквой, например  $\mathbf{x}$  и  $\mathbf{y}$ .

В принципе, векторы — это объекты особого рода, которые можно складывать друг с другом и умножать на скаляры, чтобы получить новый объект того же рода. С точки зрения абстрактной математики, любой объект, обладающий двумя этими свойствами, может считаться вектором. Вот несколько примеров таких векторных объектов:

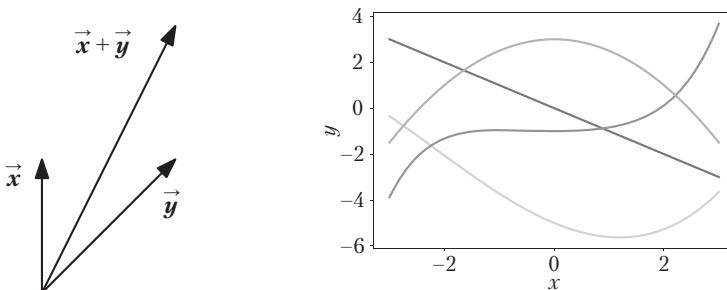
1. Геометрические векторы. Векторы такого рода изучаются в старших классах, в курсах математики и физики. Геометрические векторы — см. рис. 2.1(a) — это направленные отрезки, которые можно чертить (как минимум в двух измерениях). Два геометрических вектора  $\vec{x}$ ,  $\vec{y}$  можно сложить, так что их суммой  $\vec{x} + \vec{y} = \vec{z}$  будет третий геометрический вектор. Более того, умножение на скаляр,  $\lambda\vec{x}$ ,  $\lambda \in \mathbb{R}$ , также даст геометрический вектор. Фактически это исходный вектор, умноженный на  $\lambda$ . Следовательно, геометрические векторы — это примеры воплощения концепции вектора, с которой мы познакомились выше. Интерпретируя векторы как геометрические сущности, мы можем интуитивно судить об их направлении и величине, а также рассуждать о математических операциях над ними.
2. Многочлены — это тоже векторы; см. рис. 2.1(b): два многочлена можно сложить друг с другом, получив в результате третий многочлен; также их можно умножать на скаляр  $\lambda \in \mathbb{R}$ , и в результате тоже получится многочлен.

Следовательно, многочлены — это образцы векторов (пусть и довольно необычные). Обратите внимание на то, что многочлены очень отличаются от геометрических векторов. Тогда как геометрический вектор — это конкретный «рисунок», многочлен — это абстрактная концепция. Однако и те, и другие являются векторами в вышеизложенном смысле.

3. Аудиосигналы — это векторы. Аудиосигнал можно представить как последовательность чисел. Также можно складывать аудиосигналы друг с другом, и их суммой будет новый аудиосигнал. Если умножить аудиосигнал, также получится аудиосигнал. Следовательно, аудиосигналы также являются своеобразными векторами.
4. Элементы  $\mathbb{R}^n$  (кортежи из  $n$  вещественных чисел) — это векторы.  $\mathbb{R}^n$  более абстрактны, чем многочлены, и именно этой концепции уделяется особое внимание в данной книге. Так,

$$\mathbf{a} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \in \mathbb{R}^3 \quad (2.1)$$

— это пример тройки чисел. Покомпонентное сложение двух векторов  $\mathbf{a}$ ,  $\mathbf{b} \in \mathbb{R}^n$  дает еще один вектор:  $\mathbf{a} + \mathbf{b} = \mathbf{c} \in \mathbb{R}^n$ . Более того, при умножении  $\mathbf{a} \in \mathbb{R}^n$  на  $\lambda \in \mathbb{R}$  получается умноженный вектор  $\lambda\mathbf{a} \in \mathbb{R}^n$ . Рассматривать векторы как элементы  $\mathbb{R}^n$  удобно еще и потому, что в таком представлении они условно соответствуют массивам вещественных чисел с точки зрения компьютера<sup>1</sup>. Во многих языках программирования поддерживаются операции над массивами, благодаря чему удобно реализовывать алгоритмы, связанные с выполнением операций над векторами.

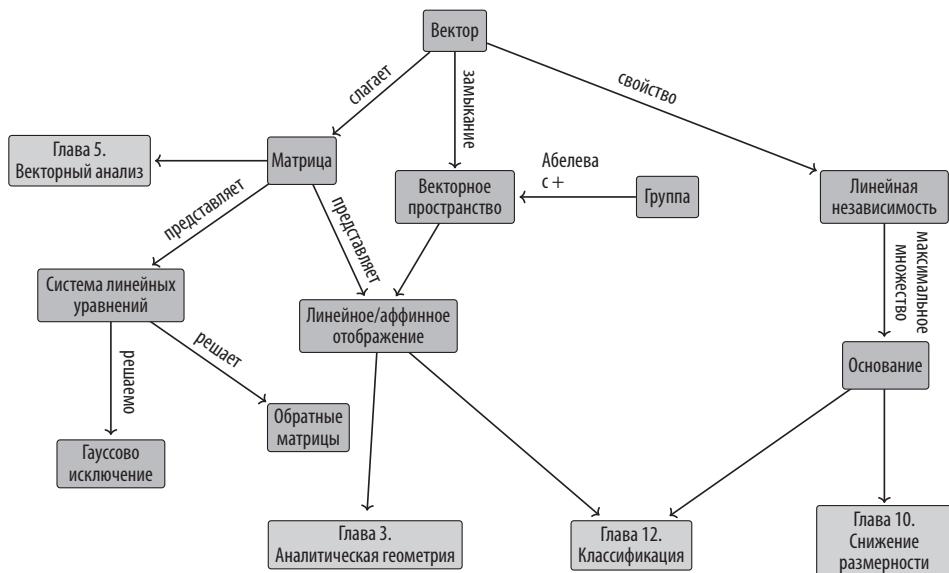


**Рис. 2.1.** Векторы разных типов. Векторы порой удивительны, и к ним относятся как (a) геометрические векторы, так и (b) многочлены

<sup>1</sup> Тщательно проверьте, на самом ли деле операции над массивами тождественны операциям над векторами, если реализовать их на компьютере.

В линейной алгебре особое внимание уделяется сходству между двумя этими векторными концепциями. Такие векторы можно складывать друг с другом и умножать на скаляры. Мы сосредоточимся преимущественно на векторах из  $\mathbb{R}^n$ , так как большинство алгоритмов линейной алгебры формулируются в  $\mathbb{R}^n$ . В главе 8 будет показано, что данные часто трактуются как векторы в  $\mathbb{R}^n$ . В этой книге мы сосредоточимся на изучении конечномерных векторных пространств, где существует однозначное соответствие между вектором любого рода и  $\mathbb{R}^n$ . Когда это будет удобно, мы будем использовать аналогии из области геометрических векторов, рассуждая об алгоритмах, основанных на массивах.

Одной из важнейших идей в математике является замыкание. Вот вопрос: каково множество всех результатов, которые могут быть получены при выполнении предлагаемых мной операций? В случае векторов формулировка такова: какое множество векторов можно получить, взяв за основу небольшой набор векторов, а затем складывая и нормируя их? В результате получится векторное пространство (раздел 2.4). На концепции векторного пространства и его свойствах во многом базируется машинное обучение. Концепции, введенные в этой главе, обобщены на рис. 2.2.



**Рис. 2.2.** Ассоциативная карта концепций, вводимых в этой главе, с указанием, в каких еще частях книги они фигурируют

Эта глава большей частью основана на конспектах и книгах Drumm and Weil (2001), Gilbert Strang (2003), Hogben (2013), Liesen and Mehrmann (2015), а так-

же на серии Павла Гринфельда (Pavel Grinfeld) «Линейная алгебра» (<http://tinyurl.com/nahclwm>). Другие отличные ресурсы — это курс Гилберта Стрэнга по линейной алгебре (MIT) (<http://tinyurl.com/29p5q8j>) и серия «Линейная алгебра» с 3Blue1Brown (<https://tinyurl.com/h5g4kps>).

Линейная алгебра играет важную роль в машинном обучении и в математике в целом. Концепции, включенные в эту главу, далее раскрываются в главе 3 с захватом геометрии. В главе 5 мы поговорим о векторном анализе, где важны хорошо усвоенные знания об операциях над матрицами. В главе 10 мы будем пользоваться проекциями (с ними вы познакомитесь в разделе 3.8) для снижения размерности с применением анализа главных компонент. В главе 9 мы поговорим о линейной регрессии, где линейная алгебра играет центральную роль в решении задач наименьших квадратов.

## 2.1. СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ

Системы линейных уравнений играют центральную роль в линейной алгебре. Многие задачи можно сформулировать в виде систем линейных уравнений, а линейная алгебра предоставляет инструментарий для их решения.

### Пример 2.1

Компания производит линейку продуктов  $N_1, \dots, N_n$ , для которых требуется ресурсы  $R_1, \dots, R_m$ . На производство единицы продукта  $N_j$  требуется  $a_{ij}$  единиц ресурса  $R_i$ , где  $i = 1, \dots, m$ , а  $j = 1, \dots, n$ .

Цель — подобрать оптимальный производственный план, то есть спланировать, сколько единиц  $x_j$  продукта  $N_j$  должно быть произведено, если доступно всего  $b_i$  единиц ресурса  $R_i$  и, в идеале, неизрасходованных ресурсов не остается.

Если мы произведем  $x_1, \dots, x_n$  единиц соответствующего продукта, то нам понадобится всего

$$a_{i1}x_1 + \dots + a_{in}x_n \quad (2.2)$$

единиц ресурса  $R_i$ . Следовательно, оптимальный производственный план  $(x_1, \dots, x_n) \in \mathbb{R}^n$  должен удовлетворять следующей системе уравнений:

$$\begin{aligned} a_{11}x_1 + \dots + a_{1n}x_n &= b_1 \\ &\vdots \\ a_{m1}x_1 + \dots + a_{mn}x_n &= b_m, \end{aligned} \quad (2.3)$$

где  $a_{ij} \in \mathbb{R}$  и  $b_i \in \mathbb{R}^n$ .

Уравнение (2.3) – это обобщенная форма *системы линейных уравнений*, а  $\mathbf{x}_1, \dots, \mathbf{x}_n$  – это *неизвестные* данной системы. Каждый  $n$ -кортеж  $(\mathbf{x}_1, \dots, \mathbf{x}_n) \in \mathbb{R}^n$ , удовлетворяющий (2.3), является *решением* системы линейных уравнений.

### Пример 2.2

Система линейных уравнений

$$\begin{aligned} x_1 + x_2 + x_3 &= 3 & (1) \\ x_1 - x_2 + 2x_3 &= 2 & (2) \\ 2x_1 + 3x_3 &= 1 & (3) \end{aligned} \quad (2.4)$$

*не имеет решения*. Сложив два первых уравнения, имеем  $2x_1 + 3x_3 = 5$ , что противоречит третьему уравнению (3).

Рассмотрим систему линейных уравнений

$$\begin{aligned} x_1 + x_2 + x_3 &= 3 & (1) \\ x_1 - x_2 + 2x_3 &= 2 & (2) \\ x_2 + 3x_3 &= 2 & (3). \end{aligned} \quad (2.5)$$

Из первого и третьего уравнения следует, что  $x_1 = 1$ . Из (1) + (2) имеем, что  $2x_1 + 3x_3 = 5$ , то есть  $x_3 = 1$ . Затем из (3) получаем, что  $x_2 = 1$ . Следовательно,  $(1, 1, 1)$  – это единственное возможное, *единственное решение* (чтобы убедиться, что  $(1, 1, 1)$  является решением, подставьте его в уравнение).

В качестве третьего примера рассмотрим:

$$\begin{aligned} x_1 + x_2 + x_3 &= 3 & (1) \\ x_1 - x_2 + 2x_3 &= 2 & (2) \\ 2x_1 + 3x_3 &= 5 & (3). \end{aligned} \quad (2.6)$$

Поскольку  $(1) + (2) = (3)$ , третье уравнение можно опустить (оно избыточно). Из (1) и (2) следует, что  $2x_1 = 5 - 3x_3$  и  $2x_2 = 1 + x_3$ . Мы определяем  $x_3 = a \in \mathbb{R}$  как свободную переменную, такую что любая тройка

$$\left( \frac{5}{2} - \frac{3}{2}a, \frac{1}{2} + \frac{1}{2}a, a \right), a \in \mathbb{R} \quad (2.7)$$

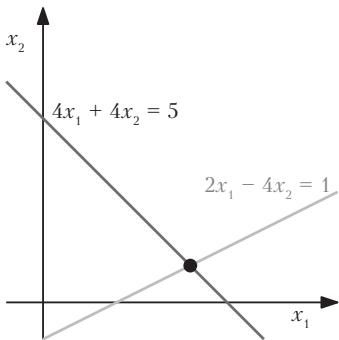
является решением системы линейных уравнений, то есть мы получаем множество, в котором содержится *бесконечное количество решений*.

В целом, в системах линейных уравнений, значения которых являются вещественными числами, имеется либо ноль, либо строго одно, либо бесконечно много решений. При помощи линейной регрессии (глава 9) решается такая версия примера 2.1, которую невозможно решить при помощи системы линейных уравнений.

**ПРИМЕЧАНИЕ** В системе линейных уравнений с двумя переменными  $x_1$ ,  $x_2$  каждое линейное уравнение определяет прямую линию на плоскости  $x_1x_2$ . Поскольку решение системы линейных уравнений должно удовлетворять всем решениям этой системы одновременно, данное решение является пересечением этих линий. Данное пересечение может представлять собой линию (если все линейные уравнения описывают одну и ту же прямую), или точку, или быть пустым (когда линии параллельны). На рис. 2.3 проиллюстрирована система уравнений

$$\begin{aligned} 4x_1 + 4x_2 &= 5 \\ 2x_1 - 4x_2 &= 1, \end{aligned} \tag{2.8}$$

где пространство решений — это точка  $(x_1, x_2) = (1, \frac{1}{4})$ . Аналогично, при трех переменных каждое линейное уравнение определяет плоскость в трехмерном пространстве. При пересечении этих плоскостей, то есть в ситуации, удовлетворяющей одновременно всем линейным уравнениям, можно получить решение, которое будет представлять собой плоскость, или прямую, или точку, или будет пустым (когда плоскости не пересекаются). ◆



**Рис. 2.3.** Пространство решений двух линейных уравнений с двумя переменными поддается геометрической интерпретации как пересечение двух прямых. Каждому линейному уравнению соответствует прямая

Для систематического подхода к решению систем линейных уравнений введем удобную компактную нотацию. Соберем коэффициенты  $a_{ij}$  в векторы, а век-

торы — в матрицы. Иными словами, запишем систему из (2.3) в следующей форме:

$$x_1 \begin{bmatrix} a_{11} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \quad (2.9)$$

$$\Leftrightarrow \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}. \quad (2.10)$$

Далее подробнее рассмотрим эти *матрицы* и определим правила их вычисления. Вернемся к решению линейных уравнений в разделе 2.3.

## 2.2. МАТРИЦЫ

Матрицам отводится центральная роль в линейной алгебре. Их можно использовать для компактного представления систем линейных уравнений, но также они представляют линейные функции (линейные отображения), как будет показано ниже, в разделе 2.7. Прежде чем перейти к обсуждению некоторых из этих интересных тем, давайте определим, что такое матрица и какие виды операций можно производить над матрицами. Подробнее о свойствах матриц мы поговорим в главе 4.

**Определение 2.1 (матрица).** При  $m, n \in \mathbb{R}$  матрица  $\mathbf{A}$  ( $m, n$ ), значениями которой являются вещественные числа, представляет собой  $m \cdot n$ -кортеж элементов  $a_{ij}$ ,  $i = 1, \dots, m, j = 1, \dots, n$ , упорядоченный в соответствии с прямоугольной структурой, в которой  $m$  строк и  $n$  столбцов:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, a_{ij} \in \mathbb{R} \quad (2.11)$$

Принято называть  $(1, n)$ -матрицы *строками*, а  $(m, 1)$ -матрицы *столбцами*. Такие особые матрицы также называются *векторами строк/столбцов*.

$\mathbb{R}^{m \times n}$  — это множество всех  $(m, n)$ -матриц, чьи значения являются вещественными числами.  $\mathbf{A} \in \mathbb{R}^{m \times n}$  можно эквивалентно представить как  $\mathbf{a} \in \mathbb{R}^{mn}$ , уложив все  $n$  столбцов матрицы в длинный вектор (рис. 2.4).



**Рис. 2.4.** Объединив столбцы матрицы  $A$ , можно представить ее как длинный вектор  $a$

### 2.2.1. Сложение и перемножение матриц

Сумма двух матриц<sup>1</sup>  $A \in \mathbb{R}^{m \times n}$ ,  $B \in \mathbb{R}^{m \times n}$  определяется как их покомпонентная сумма

$$A + B := \begin{bmatrix} a_{11} + b_{11} & \cdots & a_{1n} + b_{1n} \\ \vdots & & \vdots \\ a_{m1} + b_{m1} & \cdots & a_{mn} + b_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}. \quad (2.12)$$

Для матриц  $A \in \mathbb{R}^{m \times n}$ ,  $B \in \mathbb{R}^{n \times k}$  элементы  $c_{ij}$  произведения  $C = AB \in \mathbb{R}^{m \times k}$  вычисляются<sup>2</sup> как

$$c_{ij} = \sum_{l=1}^n a_{il} b_{lj}, \quad i = 1, \dots, m, j = 1, \dots, k. \quad (2.13)$$

Это означает, что для вычисления элемента  $c_{ij}$  мы умножаем элементы  $i$ -й строки  $A$  на  $j$ -й столбец  $B$  и суммируем их. Ниже, в разделе 3.2, мы назовем такой результат *скалярным произведением* соответствующих строки и столбца<sup>3</sup>. В случаях, когда нам потребуется явно указать, что мы выполняем умножение, мы обозначим его при помощи нотации  $A \cdot B$  (то есть покажем «·»).

**ПРИМЕЧАНИЕ** Матрицы поддаются перемножению, лишь если совпадают их «соседние» размерности. Например, матрицу  $A$  вида  $n \times k$  можно умножить на матрицу  $B$  вида  $k \times m$ , но только слева:

$$\underbrace{\begin{matrix} A \\ n \times k \end{matrix}}_{\text{столбцы}} \underbrace{\begin{matrix} B \\ k \times m \end{matrix}}_{\text{строки}} = \underbrace{\begin{matrix} C \\ m \times n \end{matrix}}_{\text{столбцы}}. \quad (2.14)$$

Произведение  $BA$  не определено, если  $m \neq n$ , так как соседние размерности не совпадают. ♦

<sup>1</sup> Обратите внимание на размер матриц.

<sup>2</sup>  $C = np.einsum('il, lj', A, B)$ .

<sup>3</sup> В  $A$   $n$  столбцов, а в  $B$   $n$  строк, так что можно вычислить  $a_{il}b_{lj}$  для  $l = 1, \dots, n$ . Обычно скалярное произведение двух векторов  $a$  и  $b$  обозначается как  $a^T b$  или  $\langle a, b \rangle$ .

**ПРИМЕЧАНИЕ** Умножение матриц *не* определяется как покомпонентная операция над элементами матрицы, то есть  $c_{ij} \neq a_{ij}b_{ij}$  (даже если размер  $\mathbf{AB}$  был подобран правильно). Подобное покомпонентное перемножение часто встречается в языках программирования, когда мы перемножаем друг с другом (многомерные) массивы, и называется *произведением Адамара*. ♦

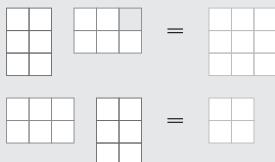
### Пример 2.3

Для  $A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} \in \mathbb{R}^{2 \times 3}$ ,  $B = \begin{bmatrix} 0 & 2 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{3 \times 2}$  мы получаем

$$\mathbf{AB} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ 2 & 5 \end{bmatrix} \in \mathbb{R}^{2 \times 2}, \quad (2.15)$$

$$\mathbf{BA} = \begin{bmatrix} 0 & 2 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 6 & 4 & 2 \\ -2 & 0 & 2 \\ 3 & 2 & 1 \end{bmatrix} \in \mathbb{R}^{3 \times 3}. \quad (2.16)$$

На этом примере уже можем убедиться, что перемножение матриц не-коммутативно, то есть  $\mathbf{AB} \neq \mathbf{BA}$ ; см. также рис. 2.5 в качестве иллюстрации.



**Рис. 2.5.** Даже если оба варианта умножения матриц  $\mathbf{AB}$  и  $\mathbf{BA}$  определены, размерности результатов могут отличаться

**Определение 2.2 (единичная матрица).** В  $\mathbb{R}^{n \times n}$  определяем *единичную матрицу*

$$\mathbf{I}_n := \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{n \times n} \quad (2.17)$$

как матрицу  $n \times n$ , содержащую 1 по диагонали и 0 во всех других направлениях.

Теперь, когда мы определили перемножение матриц, сложение матриц и разобрались, что такая единичная матрица, рассмотрим некоторые свойства матриц:

- Ассоциативность:

$$\forall A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times p}, C \in \mathbb{R}^{p \times q} : (AB)C = A(BC). \quad (2.18)$$

- Дистрибутивность:

$$\forall A, B \in \mathbb{R}^{m \times n}, C, D \in \mathbb{R}^{n \times p}, D \in \mathbb{R}^{p \times q} : (A + B)C = AC + BC. \quad (2.19a)$$

$$A(C + D) = AC + AD. \quad (2.19b)$$

- Умножение на единичную матрицу:

$$\forall A \in \mathbb{R}^{m \times n} : I_m A = A I_n = A. \quad (2.20)$$

Обратите внимание:  $I_m \neq I_n$  для  $m \neq n$ .

## 2.2.2. Обращение и транспонирование

**Определение 2.3 (обращение).** Рассмотрим квадратную матрицу<sup>1</sup>  $A \in \mathbb{R}^{n \times n}$ . Пусть матрица  $B \in \mathbb{R}^{n \times n}$  такова, что  $AB = I_n = BA$ .  $B$  называется *обратной*  $A$  и обозначается  $A^{-1}$ .

К сожалению, не у каждой матрицы  $A$  есть обратная  $A^{-1}$ . Если такая обратная матрица существует, то  $A$  называется *регулярной/обратимой/невырожденной* матрицей, а в противном случае – *вырожденной/необратимой*. Если обратная матрица существует, то она единственна. В разделе 2.3 мы обсудим общий способ расчета обратной матрицы путем решения системы линейных уравнений.

**ПРИМЕЧАНИЕ** Рассмотрим матрицу

$$A := \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \in \mathbb{R}^{2 \times 2}. \quad (2.21)$$

Если умножить  $A$  на

$$B := \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}, \quad (2.22)$$

получим

$$AB = \begin{bmatrix} a_{11}a_{22} - a_{12}a_{21} & 0 \\ 0 & a_{11}a_{22} - a_{12}a_{21} \end{bmatrix} = (a_{11}a_{22} - a_{12}a_{21})I. \quad (2.23)$$

---

<sup>1</sup> В квадратной матрице одинаковое количество строк и столбцов.

Следовательно,

$$A^{-1} = \frac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}, \quad (2.24)$$

тогда и только тогда, когда  $a_{11}a_{22} - a_{12}a_{21} \neq 0$ . В разделе 4.1 мы увидим, что  $a_{11}a_{22} - a_{12}a_{21}$  является определителем (детерминантом) матрицы  $2 \times 2$ . В дальнейшем можно использовать детерминант матрицы, чтобы проверить, является ли она обратимой. ♦

#### Пример 2.4 (обратная матрица)

Матрицы

$$A = \begin{bmatrix} 1 & 2 & 1 \\ 4 & 4 & 5 \\ 6 & 7 & 7 \end{bmatrix}, B = \begin{bmatrix} -7 & -7 & 6 \\ 2 & 1 & -1 \\ 4 & 5 & -4 \end{bmatrix} \quad (2.25)$$

являются обратными друг другу, поскольку  $AB = I = BA$ .

**Определение 2.4 (транспонирование).** Для  $A \in \mathbb{R}^{m \times n}$  матрица  $B \in \mathbb{R}^{n \times m}$  с  $b_{ij} = a_{ji}$  называется транспозицией  $A$ . Это записывается как  $B = A^T$ .

В общем случае  $A^T$  можно получить, записав столбцы  $A$  как строки  $A^T$ . Ниже показаны некоторые важные свойства обращения и транспозиции<sup>1</sup>:

$$AA^{-1} = I = A^{-1}A; \quad (2.26)$$

$$(AB)^{-1} = B^{-1}A^{-1}; \quad (2.27)$$

$$(A + B)^{-1} \neq A^{-1} + B^{-1}; \quad (2.28)$$

$$(A^T)^T = A; \quad (2.29)$$

$$(A + B)^T = A^T + B^T; \quad (2.30)$$

$$(AB)^T = B^TA^T. \quad (2.31)$$

---

<sup>1</sup> Главная диагональ (иногда именуемая «основной диагональю») матрицы  $A$  — это совокупность элементов  $A_{ij}$ , где  $i = j$ . Скалярное представление (2.28) записывается как

$$\frac{1}{2+4} = \frac{1}{6} \neq \frac{1}{2} + \frac{1}{4}.$$

**Определение 2.5 (симметричная матрица).** Матрица  $A \in \mathbb{R}^{n \times n}$  является *симметричной*, если  $A = A^T$ .

Обратите внимание: лишь  $(n, n)$ -матрицы могут быть симметричны. Обычно  $(n, n)$ -матрицы также именуются *квадратными*, поскольку число строк и столбцов в них одинаково. Более того, если  $A$  обратима, то и  $A^T$  обратима, а также  $(A^{-1})^T = (A^T)^{-1} =: A^{-T}$ .

**ПРИМЕЧАНИЕ** Сумма симметричных матриц  $A, B \in \mathbb{R}^{n \times n}$  всегда симметрична. Однако хотя их произведение обычно и является определенным, оно, как правило, несимметрично:

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}. \quad (2.32)$$



### 2.2.3. Умножение на скаляр

Рассмотрим, что происходит с матрицами, умножаемыми на скаляр  $\lambda \in \mathbb{R}$ . Предположим, что  $A \in \mathbb{R}^{m \times n}$  и  $\lambda \in \mathbb{R}$ . Тогда  $\lambda A = K$ ,  $K_{ij} = \lambda a_{ij}$ . Практически на  $\lambda$  умножается каждый компонент  $A$ . Для  $\lambda, \psi \in \mathbb{R}$  верно следующее:

- *Ассоциативность:*

$$(\lambda\psi)C = \lambda(\psi C), \quad C \in \mathbb{R}^{m \times n};$$

$$\lambda(BC) = (\lambda B)C = B(\lambda C) = (BC)\lambda, \quad B \in \mathbb{R}^{m \times n}, \quad C \in \mathbb{R}^{n \times k}.$$

Обратите внимание: это позволяет нам менять скалярные значения местами.

$$(\lambda C)^T = C^T \lambda^T = C^T \lambda = \lambda C^T, \text{ так как } \lambda = \lambda^T \text{ для всех } \lambda \in \mathbb{R}.$$

- *Дистрибутивность:*

$$(\lambda + \psi)C = \lambda C + \psi C, \quad C \in \mathbb{R}^{m \times n};$$

$$\lambda(B + C) = \lambda B + \lambda C, \quad B, C \in \mathbb{R}^{m \times n}.$$

#### Пример 2.5 (дистрибутивность)

Если определить

$$C := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad (2.33)$$

то для любых  $\lambda, \psi \in \mathbb{R}$ , имеем

$$(\lambda + \psi) \mathbf{C} = \begin{bmatrix} (\lambda + \psi)1 & (\lambda + \psi)2 \\ (\lambda + \psi)3 & (\lambda + \psi)4 \end{bmatrix} = \begin{bmatrix} \lambda + \psi & 2\lambda + 2\psi \\ 3\lambda + 3\psi & 4\lambda + 4\psi \end{bmatrix} = \quad (2.34a)$$

$$= \begin{bmatrix} \lambda & 2\lambda \\ 3\lambda & 4\lambda \end{bmatrix} + \begin{bmatrix} \psi & 2\psi \\ 3\psi & 4\psi \end{bmatrix} = \lambda \mathbf{C} + \psi \mathbf{C}. \quad (2.34b)$$

## 2.2.4. Компактное представление системы уравнений

Если рассмотреть систему линейных уравнений

$$\begin{aligned} 2x_1 + 3x_2 + 5x_3 &= 1 \\ 4x_1 - 2x_2 - 7x_3 &= 8 \\ 9x_1 + 5x_2 - 3x_3 &= 2 \end{aligned} \quad (2.35)$$

и воспользоваться правилами перемножения матриц, то данную систему уравнений можно записать в более компактной форме как

$$\begin{bmatrix} 2 & 3 & 5 \\ 4 & -2 & -7 \\ 9 & 5 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 8 \\ 2 \end{bmatrix}. \quad (2.36)$$

Обратите внимание: на  $x_1$  умножается первый столбец, на  $x_2$  — второй и на  $x_3$  — третий.

Как правило, систему линейных уравнений можно компактно представить в матричной форме как  $\mathbf{Ax} = \mathbf{b}$ , см. (2.3), а произведение  $\mathbf{Ax}$  — это (линейная) комбинация столбцов  $\mathbf{A}$ . Мы подробнее обсудим линейные комбинации в разделе 2.5.

## 2.3. РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ

В (2.3) мы в общем виде познакомили вас с системой уравнений, то есть

$$\begin{aligned} a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\ &\vdots \\ a_{m1}x_1 + \cdots + a_{mn}x_n &= b_m, \end{aligned} \quad (2.37),$$

где  $a_{ij} \in \mathbb{R}$  и  $b_i \in \mathbb{R}$  — это известные константы, а  $x_j$  — неизвестные,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$ . Как мы уже убедились, матрицы могут использоваться для компактной записи систем линейных уравнений, поэтому можем записать:  $\mathbf{Ax} = \mathbf{b}$  (2.10). Кроме того, мы определили базовые операции с матрицами, в частности научи-

лись их складывать и перемножать. Далее мы сосредоточимся на решении систем линейных уравнений и дадим алгоритм для нахождения обратной матрицы.

### 2.3.1. Частное и общее решение

Прежде чем перейти к рассмотрению общего случая решения систем линейных уравнений, обсудим пример. Рассмотрим систему уравнений:

$$\begin{bmatrix} 1 & 0 & 8 & -4 \\ 0 & 1 & 2 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 42 \\ 8 \end{bmatrix}. \quad (2.38)$$

В этой системе два уравнения и четыре неизвестных. Следовательно, как правило, в таком случае должно быть бесконечно много решений. Данная система уравнений особенно проста по форме, в ней первые два столбца состоят из 1 и 0. Как вы помните, нас интересуют скаляры  $x_1, \dots, x_4$ , такие что  $\sum_{i=1}^4 x_i \mathbf{c}_i = \mathbf{b}$ , где мы определяем  $\mathbf{c}_i$  как  $i$ -й столбец матрицы, а  $\mathbf{b}$  — как правую часть (2.38). Решение задачи (2.38) можно найти сразу, 42 раза взяв первый столбец и 8 раз второй столбец, так что

$$\mathbf{b} = \begin{bmatrix} 42 \\ 8 \end{bmatrix} = 42 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 8 \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \quad (2.39)$$

Следовательно, решение —  $[42, 8, 0, 0]^T$ . Это решение называется *частным* или *специальным*. Однако это не единственное решение системы линейных уравнений. Чтобы охватить все решения, нужно проявить креативность и нетривиальным образом генерировать 0, беря за основу столбцы матрицы. Если прибавить 0 к нашему специальному решению, то оно не изменится. Для этого выразим третий столбец, воспользовавшись двумя первыми столбцами (которые имеют очень простую форму):

$$\begin{bmatrix} 8 \\ 2 \end{bmatrix} = 8 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 2 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2.40)$$

так, чтобы  $0 = 8\mathbf{c}_1 + 2\mathbf{c}_2 - 1\mathbf{c}_3 + 0\mathbf{c}_4$  и  $(x_1, x_2, x_3, x_4) = (8, 2, -1, 0)$ . Фактически при любом умножении данного решения на  $\lambda_1 \in \mathbb{R}$  получается **0**-вектор, то есть

$$\begin{bmatrix} 1 & 0 & 8 & -4 \\ 0 & 1 & 2 & 12 \end{bmatrix} \begin{pmatrix} \lambda_1 \\ 8 \\ 2 \\ -1 \end{pmatrix} = \lambda_1 (8\mathbf{c}_1 + 2\mathbf{c}_2 - \mathbf{c}_3) = \mathbf{0}. \quad (2.41)$$

Следуя той же логике рассуждений, выразим четвертый столбец матрицы из (2.38) при помощи двух первых столбцов и сгенерируем еще одно нетривиальное выражение для  $\mathbf{0}$  вида

$$\begin{bmatrix} 1 & 0 & 8 & -4 \\ 0 & 1 & 2 & 12 \end{bmatrix} \begin{pmatrix} \lambda_2 \\ -4 \\ 12 \\ 0 \\ -1 \end{pmatrix} = \lambda_2(-4\mathbf{c}_1 + 12\mathbf{c}_2 - \mathbf{c}_4) = \mathbf{0} \quad (2.42)$$

для любой  $\lambda_2 \in \mathbb{R}$ . Собрав все вместе, мы получим полное множество решений для системы уравнений из (2.38), которое называется *общим решением* и представляет собой множество

$$\left\{ \mathbf{x} \in \mathbb{R}^4 : \mathbf{x} = \begin{bmatrix} 42 \\ 8 \\ 0 \\ 0 \end{bmatrix} + \lambda_1 \begin{bmatrix} 8 \\ 2 \\ -1 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} -4 \\ 12 \\ 0 \\ -1 \end{bmatrix}, \lambda_1, \lambda_2 \in \mathbb{R} \right\}. \quad (2.43)$$

**ПРИМЕЧАНИЕ** Общий подход, которого мы придерживались, включает три шага:

1. Найти частное решение для  $A\mathbf{x} = \mathbf{b}$ .
2. Найти все решения для  $A\mathbf{x} = \mathbf{0}$ .
3. Скомбинировать решения из шагов 1 и 2, чтобы найти общее решение.

Ни частное, ни общее решение не уникальны. ◆

Решить систему линейных уравнений из предыдущего примера было легко, так как матрица из примера (2.38) обладает именно такой удобной формой, которая позволяет нам тривиально найти частное и общее решение. Однако общие системы уравнений не столь просты по форме. К счастью, существует конструктивный алгоритмический способ преобразовать любую систему уравнений к такой особенно простой форме: это гауссово исключение. Ключевой составляющей гауссова исключения являются элементарные преобразования систем линейных уравнений, переводящие систему уравнений именно в такую простую форму. Далее к такой простой форме применимы те три шага, которые уже рассматривались в контексте примера (2.38).

### 2.3.2. Элементарные преобразования

Ключевым фактором решения систем линейных уравнений являются *элементарные преобразования*, при которых множество решений не меняется, но сама система уравнений преобразуется в более простую форму:

- Перестановка двух уравнений (строк в матрице, представляющей систему уравнений).
- Умножение уравнения (строки) на постоянную  $\lambda \in \mathbb{R} \setminus \{0\}$ .
- Сложение двух уравнений (строк).

### Пример 2.6

Для  $a \in \mathbb{R}$  найдем все решения следующей системы уравнений:

$$\begin{array}{rcl} -2x_1 + 4x_2 - 2x_3 - x_4 + 4x_5 = -3 \\ 4x_1 - 8x_2 + 3x_3 - 3x_4 + x_5 = 2 \\ x_1 - 2x_2 + x_3 - x_4 + x_5 = 0 \\ x_1 - 2x_2 - 3x_4 + 4x_5 = a \end{array} \quad (2.44)$$

Начнем с преобразования этой системы уравнений в компактную матричную нотацию  $A\mathbf{x} = \mathbf{b}$ . Мы более не будем явно упоминать переменные  $\mathbf{x}$  и построим *расширенную матрицу* (вида  $[A|b]$ ).

$$\left[ \begin{array}{ccccc|c} -2 & 4 & -2 & -1 & 4 & -3 \\ 4 & -8 & 3 & -3 & 4 & 2 \\ 1 & -2 & 1 & -1 & 1 & 0 \\ 1 & -2 & 0 & -3 & 4 & a \end{array} \right] \begin{array}{l} \text{обмен с } R_3 \\ \text{обмен с } R_1 \end{array}$$

где вертикальная линия служит для отделения левой части от правой (2.44)<sup>1</sup>. Символ  $\rightsquigarrow$  будет использован для указания преобразования расширенной матрицы с применением элементарных преобразований.

Поменяв местами строки 1 и 3, получим

$$\left[ \begin{array}{ccccc|c} 1 & -2 & 1 & -1 & 1 & 0 \\ 4 & -8 & 3 & -3 & 1 & 2 \\ -2 & 4 & -2 & -1 & 4 & -3 \\ 1 & -2 & 0 & -3 & 4 & a \end{array} \right] \begin{array}{l} -4R_1 \\ +2R_3 \\ -R_1 \end{array}$$

Теперь, применив указанные преобразования (то есть четырежды вычтя строку 1 из строки 2), получим

---

<sup>1</sup> Расширенная матрица  $[A|b]$  компактно представляет систему линейных уравнений  $A\mathbf{x} = \mathbf{b}$ .

$$\begin{array}{c}
 \left[ \begin{array}{ccccc|c} 1 & -2 & 1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & -3 & 2 \\ 0 & 0 & 0 & -3 & 6 & -3 \\ 0 & 0 & -1 & -2 & 3 & a \end{array} \right] \xrightarrow{-R_2 - R_3} \\
 \left[ \begin{array}{ccccc|c} 1 & -2 & 1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & -3 & 2 \\ 0 & 0 & 0 & -3 & 6 & -3 \\ 0 & 0 & 0 & 0 & 0 & a+1 \end{array} \right] \cdot(-1) \xrightarrow{\left( \begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \end{array} \right)} \\
 \left[ \begin{array}{ccccc|c} 1 & -2 & 1 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 3 & -2 \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & a+1 \end{array} \right].
 \end{array}$$

Расширенная матрица в данном случае приведена к удобному *ступенчатому виду* (REF). Снова обратив эту компактную нотацию к явной нотации с искомыми переменными, получим

$$\begin{aligned}
 x_1 - 2x_2 + x_3 - x_4 + x_5 &= 0 \\
 x_3 - x_4 + 3x_5 &= -2 \\
 x_4 - 2x_5 &= 1 \\
 0 &= a+1
 \end{aligned} \tag{2.45}$$

Данная система может быть решена только для  $a = -1$ . Частное решение —

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ -1 \\ 1 \\ 0 \end{bmatrix}. \tag{2.46}$$

Общее решение, охватывающее весь набор возможных решений, —

$$\left\{ \mathbf{x} \in \mathbb{R}^5 : \mathbf{x} = \begin{bmatrix} 2 \\ 0 \\ -1 \\ 1 \\ 0 \end{bmatrix} + \lambda_1 \begin{bmatrix} 2 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 2 \\ 0 \\ -1 \\ 2 \\ 1 \end{bmatrix}, \lambda_1, \lambda_2 \in \mathbb{R} \right\}. \tag{2.47}$$

В дальнейшем мы подробно опишем конструктивный способ получения частного и общего решения для системы линейных уравнений.

**ПРИМЕЧАНИЕ** Направляющий коэффициент строки (первое ненулевое значение слева) называется «ведущим» (pivot) и всегда находится строго справа от ведущего значения вышестоящего ряда. Следовательно, любая система уравнений, записанная в ступенчатом виде, структурно напоминает лестницу.



**Определение 2.6 (ступенчатая форма).** Матрица имеет *ступенчатую форму*, если

- все ее строки, содержащие только нули, расположены в самом низу матрицы; соответственно, все строки, в которых содержится как минимум один ненулевой элемент, расположены выше строк, содержащих только нули;
- при рассмотрении лишь ненулевых строк первое ненулевое значение слева (также именуемое *ведущим*, или *ведущим коэффициентом*<sup>1</sup>) всегда находится строго справа от ведущего значения, расположенного на ряд выше.

**ПРИМЕЧАНИЕ** Переменные, соответствующие ведущим коэффициентам при записи в ступенчатой форме, называются *базисными*, а другие переменные — *свободными*. Например, в (2.45)  $x_1, x_3, x_4$  — это базисные переменные, а  $x_2, x_5$  — свободные.



**ПРИМЕЧАНИЕ** Ступенчатая форма облегчает нам жизнь при нахождении частного решения. Для этого мы выражаем правую часть уравнения при помощи ведущих столбцов, так что  $\mathbf{b} = \sum_{i=1}^P \lambda_i \mathbf{p}_i$ , где  $\mathbf{p}_i, i = 1, \dots, P$ , это ведущие столбцы.  $\lambda_i$  проще всего определить, если начать с самого правого ведущего столбца и двигаться влево.

В предыдущем примере мы пытались найти  $\lambda_1, \lambda_2, \lambda_3$ , так чтобы

$$\lambda_1 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \lambda_3 \begin{bmatrix} -1 \\ -1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ 1 \\ 0 \end{bmatrix}. \quad (2.48)$$

Отсюда мы относительно легко находим, что  $\lambda_3 = 1, \lambda_2 = -1, \lambda_1 = 2$ . Собирая все вместе, следует не забывать и о неведущих столбцах, для каждого из которых подразумевается и ставится коэффициент 0. Следовательно, получаем частное решение  $\mathbf{x} = [2, 0, -1, 1, 0]^T$ .



<sup>1</sup> В других книгах иногда встречается требование, что ведущий коэффициент должен быть равен 1.

**ПРИМЕЧАНИЕ** Система уравнений находится в *приведенном ступенчатом виде* (также именуемом «каноническим»), если

- она находится в ступенчатом виде;
- все ведущие коэффициенты равны 1;
- ведущий коэффициент является единственным ненулевым в своем столбце.



Такой приведенный ступенчатый вид будет важен ниже, в разделе 2.3.3, поскольку позволяет легко определить общее решение для системы линейных уравнений.

**ПРИМЕЧАНИЕ** *Гауссово исключение* — это алгоритм, выполняющий элементарное преобразование, для того чтобы придать системе линейных уравнений приведенный ступенчатый вид.



### Пример 2.7 (приведенный ступенчатый вид)

Убедимся, что следующая матрица находится в приведенном ступенчатом виде (ведущие коэффициенты выделены **жирным**):

$$A = \begin{bmatrix} \mathbf{1} & 3 & 0 & 0 & 3 \\ 0 & 0 & \mathbf{1} & 0 & 9 \\ 0 & 0 & 0 & \mathbf{1} & -4 \end{bmatrix}. \quad (2.49)$$

Ключевая идея относительно нахождения решений  $A\mathbf{x} = \mathbf{0}$  — рассмотреть *неведущие столбцы*, которые нам понадобятся, чтобы выразить (линейную) комбинацию ведущих столбцов. В приведенном ступенчатом виде это относительно просто, и мы выражаем неведущие столбцы в терминах сумм и кратных тех ведущих столбцов, что расположены левее них. Второй столбец равен первому, умноженному втрое (можно игнорировать ведущие столбцы правее второго столбца). Следовательно, чтобы получить **0**, нужно вычесть второй столбец из первого, умноженного втрое. Далее рассмотрим пятый столбец, который является вторым из неведущих. Пятый столбец можно выразить как умноженный втрое первый ведущий столбец, умноженный вдвадцати втрой ведущий столбец и умноженный на  $-4$  третий ведущий столбец. Необходимо отслеживать все индексы ведущих столбцов и преобразовать имеющееся в умноженный втрое первый столбец, умноженный на 0 второй столбец (не являющийся ведущим), умноженный вдвадцати третий столбец (который является у нас вторым ведущим) и умноженный на  $-4$  четвертый столбец (который является третьим ведущим). Далее нам понадобится вычесть пятый стол-

бец, чтобы получить  $\mathbf{0}$ . В конце концов, мы все равно решаем однородную систему уравнений.

Резюмируя, отметим, что все решения  $A\mathbf{x} = \mathbf{0}, \mathbf{x} \in \mathbb{R}^5$  даются

$$\left\{ \mathbf{x} \in \mathbb{R}^5 : \mathbf{x} = \lambda_1 \begin{bmatrix} 3 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 3 \\ 0 \\ 9 \\ -4 \\ -1 \end{bmatrix}, \lambda_1, \lambda_2 \in \mathbb{R} \right\}. \quad (2.50)$$

### 2.3.3. Прием с $-1$

Далее расскажем о практическом приеме, помогающем считывать решения  $\mathbf{x}$  однородной системы линейных уравнений  $A\mathbf{x} = \mathbf{0}$ , где  $A \in \mathbb{R}^{k \times n}, \mathbf{x} \in \mathbb{R}^n$ .

Для начала предположим, что  $A$  находится в приведенном ступенчатом виде, без единого ряда, в котором содержались бы только нули, то есть

$$A = \begin{bmatrix} 0 & \dots & 0 & \mathbf{1} & * & \dots & * & 0 & * & \dots & * & 0 & * & \dots & * \\ \vdots & & \vdots & 0 & 0 & \dots & 0 & \mathbf{1} & * & \dots & * & \vdots & \vdots & \vdots & \vdots \\ \vdots & & \vdots & \vdots & \vdots & & \vdots & 0 & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & 0 & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & \mathbf{1} & * & \dots & * \end{bmatrix}, \quad (2.51)$$

где  $*$  может быть любым вещественным числом, с тем ограничением, что первая ненулевая запись в строке должна быть равна 1, а все остальные записи в соответствующем столбце должны быть равны 0. Столбцы  $j_1, \dots, j_k$  с ведущими (выделенными **жирным**) — это стандартные единичные векторы  $\mathbf{e}_1, \dots, \mathbf{e}_k \in \mathbb{R}^k$ . Эта матрица расширяется до  $n \times n$ -матрицы  $\tilde{A}$  путем добавления к ней  $n - k$  строк в форме

$$[0 \ \dots \ 0 \ -1 \ 0 \ \dots \ 0], \quad (2.52)$$

так что диагональ расширенной матрицы  $\tilde{A}$  содержит 1 или  $-1$ . Тогда столбцы  $\tilde{A}$ , содержащие  $-1$  в качестве ведущего, являются решениями однородной системы уравнений  $A\mathbf{x} = \mathbf{0}$ . Если быть точным, эти столбцы образуют базис (раздел 2.6.1) пространства решений  $A\mathbf{x} = \mathbf{0}$ , который мы далее будем называть **ядром матрицы**, или **нулевым пространством** (раздел 2.7.3).

**Пример 2.8 (прием с  $-1$ )**

Вернемся к матрице из примера (2.49), которая уже находится в ступенчатом виде:

$$A = \begin{bmatrix} 1 & 3 & 0 & 0 & 3 \\ 0 & 0 & 1 & 0 & 9 \\ 0 & 0 & 0 & 1 & -4 \end{bmatrix}. \quad (2.53)$$

Сейчас мы расширим эту матрицу до матрицы  $5 \times 5$ , добавив строки, как в примере (2.52), там где по диагонали отсутствуют ведущие коэффициенты, и получим

$$\tilde{A} = \begin{bmatrix} 1 & 3 & 0 & 0 & 3 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 9 \\ 0 & 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}. \quad (2.54)$$

Из этой формы мы можем сразу же считывать решения для  $A\mathbf{x} = \mathbf{0}$ , взяв те столбцы  $\tilde{A}$ , которые содержат  $-1$  по диагонали:

$$\left\{ \mathbf{x} \in \mathbb{R}^5 : \mathbf{x} = \lambda_1 \begin{bmatrix} 3 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 3 \\ 0 \\ 9 \\ -4 \\ -1 \end{bmatrix}, \lambda_1, \lambda_2 \in \mathbb{R} \right\}, \quad (2.55)$$

что идентично решению (2.50), на которое мы «вышли».

**Расчет обратной матрицы**

Чтобы вычислить матрицу  $A^{-1}$ , обратную  $A \in \mathbb{R}^{n \times n}$ , необходимо найти матрицу  $X$ , удовлетворяющую  $AX = I_n$ . Тогда  $X = A^{-1}$ . Можно записать это как множество одновременных линейных уравнений  $AX = I_n$ , где мы находим решение для  $X = [\mathbf{x}_1 | \dots | \mathbf{x}_n]$ . Мы используем нотацию расширенных матриц для представления данного множества систем линейных уравнений и получаем

$$[A | I_n] \rightsquigarrow \dots \rightsquigarrow [I_n | A^{-1}]. \quad (2.56)$$

Таким образом, если привести расширенную систему уравнений к приведенному ступенчатому виду, то можно получить обратное от правой части системы уравнений. Следовательно, определение обратной матрицы эквивалентно решению системы линейных уравнений.

**Пример 2.9 (вычисление обратной матрицы методом гауссова исключения)**

Чтобы определить матрицу, обратную

$$A = \begin{bmatrix} 1 & 0 & 2 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad (2.57)$$

запишем расширенную матрицу

$$\left[ \begin{array}{cccc|cccc} 1 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{array} \right]$$

и методом гауссова исключения преобразуем ее в приведенный ступенчатый вид

$$\left[ \begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & -1 & 2 & -2 & 2 \\ 0 & 1 & 0 & 0 & 1 & -1 & 2 & -2 \\ 0 & 0 & 1 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 1 & -1 & 0 & -1 & 2 \end{array} \right],$$

так что искомая обратная окажется в ее правой части.

$$A^{-1} = \begin{bmatrix} -1 & 2 & -2 & 2 \\ 1 & -1 & 2 & -2 \\ 1 & -1 & 1 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}. \quad (2.58)$$

Можно убедиться, что (2.58) действительно обратная, выполнив умножение  $AA^{-1}$  и убедившись, что в итоге получается  $I_4$ .

### 2.3.4. Алгоритмы для решения системы линейных уравнений

Далее кратко обсудим подходы к решению системы линейных уравнений вида  $\mathbf{A}\mathbf{x} = \mathbf{b}$ . Предположим, что решение существует. Если бы решения не существовало, нам следовало бы прибегнуть к приближенным решениям, которые мы не рассматриваем в этой главе. Один из способов решения приближенной задачи — это линейная регрессия, о которой мы подробно поговорим в главе 9.

В частных случаях мы можем быть в состоянии определить обратное  $\mathbf{A}^{-1}$ , такое что решение  $\mathbf{A}\mathbf{x} = \mathbf{b}$  будет выражаться как  $\mathbf{A}^{-1}\mathbf{b}$ . Однако это возможно лишь в случае, если матрица  $\mathbf{A}$  является квадратной и при этом необратима, что бывает нечасто. В противном случае, при довольно общих допущениях (согласно которым  $\mathbf{A}$  должна иметь линейно независимые столбцы) можно использовать преобразование

$$\mathbf{A}\mathbf{x} = \mathbf{b} \Leftrightarrow \mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b} \Leftrightarrow \mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \quad (2.59)$$

а также *псевдообратную матрицу Мура — Пенроуза*  $(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$  для определения решения, удовлетворяющего  $\mathbf{A}\mathbf{x} = \mathbf{b}$ , которое также соответствует минимальному решению наименьших квадратов. Недостаток этого подхода в том, что он требует множества вычислений для получения матрично-матричного произведения, а также расчета матрицы, обратной  $\mathbf{A}^T\mathbf{A}$ . Более того, по причинам, связанным с числовой точностью, обычно не рекомендуется вычислять обратную или псевдообратную. Поэтому далее мы кратко обсудим альтернативные подходы к решению систем линейных уравнений.

Метод гауссова исключения играет важную роль при вычислении детерминантов (раздел 4.1), проверке, является ли множество векторов линейно независимым (раздел 2.5), вычислении обратной матрицы (раздел 2.6.2) и определении базиса векторного пространства (раздел 2.6.1). Гауссово исключение — интуитивно понятный и конструктивный способ решения системы линейных уравнений с тысячами переменных. Однако для систем с миллионами переменных этот метод непрактичен, поскольку необходимое количество арифметических операций возрастает как третья степень от количества уравнений.

На практике системы, содержащие много линейных уравнений, решаются косвенно, при помощи либо стационарных итерационных методов, таких как метод Ричардсона, метод Якоби, метод Гаусса — Зейделя, и метода последовательной верхней релаксации, либо при помощи методов подпространств Крылова, таких как сопряженные градиенты, обобщенные минимальные невязки или бисопряженные градиенты. Подробнее эти темы рассмотрены в книгах Stoer and Burlirsch (2002), Strang (2003) и Liesen and Mehrmann (2015).

Пусть  $\mathbf{x}_.$  является решением  $A\mathbf{x} = \mathbf{b}$ . Ключевая идея этих итерационных методов заключается в выполнении итераций вида

$$\mathbf{x}^{(k+1)} = \mathbf{C}\mathbf{x}^{(k)} + \mathbf{d} \quad (2.60)$$

для подходящих  $\mathbf{C}$  и  $\mathbf{d}$ , чтобы на каждой итерации уменьшать остаточную ошибку  $\|\mathbf{x}^{(k+1)} - \mathbf{x}_.\|$  на каждой итерации, так чтобы решение сходилось к  $\mathbf{x}_.$ . В разделе 3.1 мы введем нормы  $\|\cdot\|$ , которые позволяют нам вычислять сходство между векторами.

## 2.4. ВЕКТОРНЫЕ ПРОСТРАНСТВА

Выше мы рассматривали системы линейных уравнений и способы их решений (раздел 2.3). Мы убедились, что системы линейных уравнений можно компактно представлять при помощи матрично-векторной нотации (2.10). Далее мы подробнее рассмотрим векторные пространства — так называется структурированное пространство, в котором располагаются векторы.

В начале этой главы мы упрощенно охарактеризовали векторы как объекты, поддающиеся сложению друг с другом и умножению на скаляр, причем они остаются объектами одного и того же типа. Теперь мы готовы это формализовать, и для начала введем концепцию группы, представляющей собой множество элементов и операцию, определенную для этих элементов, благодаря чему некоторая структура данного множества остается неприкосновенной.

### 2.4.1. Группы

Группы играют важную роль в информатике. Они не только закладывают основу для операций над множествами, но и активно используются в криптографии, теории кода и графике.

**Определение 2.7 (группа).** Рассмотрим множество  $\mathcal{G}$  и операцию  $\otimes: \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}$ , определенную для  $\mathcal{G}$ . Тогда  $G := (\mathcal{G}, \otimes)$  называется *группой*, если удовлетворяет следующим условиям:

1. *Замыкание*  $\mathcal{G}$  при  $\otimes$ :  $\forall x, y \in \mathcal{G}: x \otimes y \in \mathcal{G}$ .
2. *Ассоциативность*:  $\forall x, y, z \in \mathcal{G}: (x \otimes y) \otimes z = x \otimes (y \otimes z)$ .
3. *Нейтральный элемент*:  $\exists e \in \mathcal{G} \forall x \in \mathcal{G}: x \otimes e = x$  и  $e \otimes x = x$ .
4. *Обратный элемент*:  $\forall x \in \mathcal{G} \exists y \in \mathcal{G}: x \otimes y = e$  и  $y \otimes x = e$ . Часто пишут  $x^{-1}$ , обозначая элемент, обратный  $x$ .

**ПРИМЕЧАНИЕ** Обратный элемент определяется относительно операции  $\otimes$  и не обязательно означает  $\frac{1}{x}$ . ◆

Если к тому же  $\forall x, y \in \mathcal{G}: x \otimes y = y \otimes x$ , то  $G = (\mathcal{G}, \otimes)$  является *абелевой группой* (коммутативной).

### Пример 2.10 (группы)

Рассмотрим несколько примеров на множества и связанные с ними операции и разберемся, являются ли эти множества группами:

- $(\mathbb{Z}, +)$  это группа.
- $(\mathbb{N}_0, +)$  это не группа: хотя  $(\mathbb{N}_0, +)$  и содержит нейтральный элемент  $(0)$ , обратные элементы в ней отсутствуют<sup>1</sup>.
- $(\mathbb{Z}, \cdot)$  это не группа; хотя  $(\mathbb{Z}, \cdot)$  и содержит нейтральный элемент  $(1)$ , обратные элементы для любых  $z \in \mathbb{Z}, z \neq \pm 1$  отсутствуют.
- $(\mathbb{R}, \cdot)$  это не группа, поскольку у  $0$  не существует обратного элемента.
- $(\mathbb{R} \setminus \{0\}, \cdot)$  это абелева группа.
- $(\mathbb{R}^n, +), (\mathbb{Z}^n, +), n \in \mathbb{N}$  это абелева группа, если  $+$  определен покомпонентно, то есть

$$(x_1, \dots, x_n) + (y_1, \dots, y_n) = (x_1 + y_1, \dots, x_n + y_n). \quad (2.61)$$

Тогда  $(x_1, \dots, x_n)^{-1} := (-x_1, \dots, -x_n)$  — это обратный элемент, а  $e = (0, \dots, 0)$  — нейтральный элемент.

- $(\mathbb{R}^{m \times n}, +)$ , множество  $m \times n$  матриц — это абелева группа (с покомпонентным сложением, как определено в (2.61)).

Давайте подробнее остановимся на  $(\mathbb{R}^{n \times n}, \cdot)$ , то есть множестве  $n \times n$  матриц с умножением матриц, как определено в (2.13).

- Замыкание и ассоциативность прямо следуют из определения умножения матриц.
- Нейтральный элемент: единичная матрица  $I_n$  — это нейтральный элемент относительно умножения матриц « $\cdot$ » в  $(\mathbb{R}^{n \times n}, + \cdot)$ .
- Обратный элемент: если обратная существует ( $A$  обычная), то  $A^{-1}$  является обратным элементом  $A \in \mathbb{R}^{n \times n}$ , и именно в этом случае  $(\mathbb{R}^{n \times n}, + \cdot)$  является группой, называется при этом *полной линейной группой*.

<sup>1</sup>  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$ .

**Определение 2.8 (полная линейная группа).** Множество обычных (необратимых) матриц является группой  $A \in \mathbb{R}^{n \times n}$  относительно умножения матриц так, как оно определено в (2.13), и такая группа называется *полной линейной группой*  $GL(n, \mathbb{R})$ . Однако, поскольку умножение матриц некоммутативно, эта группа не является абелевой.

### 2.4.2. Векторные пространства

Обсуждая группы, мы рассматривали множества  $\mathcal{G}$  и внутренние операции над  $\mathcal{G}$ , то есть отображения  $\mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}$ , которые выполняются только над элементами  $\mathcal{G}$ . В дальнейшем мы рассмотрим множества, которые, кроме внутренней операции  $+$ , также содержат внешнюю операцию  $\cdot$ , умножение вектора  $x \in \mathcal{V}$  на скаляр  $\lambda \in \mathbb{R}$ . Можно трактовать внутреннюю операцию как форму сложения, а внешнюю операцию как форму умножения. Обратите внимание: внешние/внутренние операции никак не связаны с внешними/внутренними произведениями.

**Определение 2.9 (векторное пространство).** *Векторное пространство* с множеством вещественных значений  $V = (\mathcal{V}, +, \cdot)$  — это множество  $\mathcal{V}$  с двумя операциями

$$+ : \mathcal{V} \times \mathcal{V} \rightarrow \mathcal{V} \quad (2.62)$$

$$\cdot : \mathbb{R} \times \mathcal{V} \rightarrow \mathcal{V}, \quad (2.63)$$

где

1.  $(\mathcal{V}, +)$  — это абелева группа.
2. Дистрибутивность:
  - a.  $\forall \lambda \in \mathbb{R}, x, y \in \mathcal{V}: \lambda \cdot (x + y) = \lambda \cdot x + \lambda \cdot y$ .
  - b.  $\forall \lambda, \psi \in \mathbb{R}, x \in \mathcal{V}: (\lambda + \psi) \cdot x = \lambda \cdot x + \psi \cdot x$ .
3. Ассоциативность (внешняя операция):  $\forall \lambda, \psi \in \mathbb{R}, x \in \mathcal{V}: \lambda \cdot (\psi \cdot x) = (\lambda \psi) \cdot x$ .
4. Нейтральный элемент относительно внешней операции:  $\forall x \in \mathcal{V}: 1 \cdot x = x$ .

Элементы  $x \in V$  называются *векторами*. Нейтральный элемент  $(\mathcal{V}, +)$  — это нулевой вектор  $\mathbf{0} = [0, \dots, 0]^T$ , а внутренняя операция  $+$  называется *сложением векторов*. Элементы  $\psi \in \mathbb{R}$  называются *скалярами*, а внешняя операция  $\cdot$  — это *умножение на скаляр*. Обратите внимание: скалярное произведение — это иная сущность, и о ней мы поговорим в разделе 3.2.

**ПРИМЕЧАНИЕ** «Векторное умножение»  $ab, a, b \in \mathbb{R}^n$ , не определено. Теоретически можно было бы определить поэлементное умножение, такое что  $c = ab$  при  $c_j = a_j b_j$ . Такое «умножение матриц» распространено во многих языках про-

граммирования, но с математической точки зрения не слишком целесообразно, если пользоваться стандартными правилами перемножения матриц. Трактуя векторы как матрицы  $n \times 1$  (так обычно и делается), мы можем пользоваться перемножением матриц в том виде, как оно определено в (2.13). Но тогда размерности векторов не будут совпадать. Лишь следующие варианты умножения для векторов определены:  $\mathbf{ab}^T \in \mathbb{R}^{n \times n}$  (*внешнее произведение*),  $\mathbf{a}^T \mathbf{b} \in \mathbb{R}$  (*внутреннее/скалярное произведение*). ◆

### Пример 2.11 (векторные пространства)

Рассмотрим некоторые важные примеры:

- $\mathcal{V} = \mathbb{R}^n, n \in \mathbb{N}$  – это векторное пространство, операции в котором определяются следующим образом:
  - Сложение:  $\mathbf{x} + \mathbf{y} = (x_1, \dots, x_n) + (y_1, \dots, y_n) = (x_1 + y_1, \dots, x_n + y_n)$  для всех  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ .
  - Умножение на скаляр:  $\lambda \mathbf{x} = \lambda(x_1, \dots, x_n) = (\lambda x_1, \dots, \lambda x_n)$  для всех  $\lambda \in \mathbb{R}, \mathbf{x} \in \mathbb{R}^n$ .
- $\mathcal{V} = \mathbb{R}^{m \times n}, m, n \in \mathbb{N}$  – это векторное пространство, для которого верно следующее:
  - Сложение:

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} a_{11} + b_{11} & \cdots & a_{1n} + b_{1n} \\ \vdots & & \vdots \\ a_{m1} + b_{m1} & \cdots & a_{mn} + b_{mn} \end{bmatrix}$$

определяется поэлементно для всех  $\mathbf{A}, \mathbf{B} \in \mathcal{V}$ .

- Умножение на скаляр:

$$\lambda \mathbf{A} = \begin{bmatrix} \lambda a_{11} & \cdots & \lambda a_{1n} \\ \vdots & & \vdots \\ \lambda a_{m1} & \cdots & \lambda a_{mn} \end{bmatrix},$$

как определено в разделе 2.2. Не забывайте, что  $\mathbb{R}^{m \times n}$  эквивалентно  $\mathbb{R}^{mn}$ .

- $\mathcal{V} = \mathbb{C}$ , со стандартно определенным сложением комплексных чисел.

**ПРИМЕЧАНИЕ** В дальнейшем мы будем обозначать векторное пространство  $(\mathcal{V}, +, \cdot)$  через  $V$ , где  $+$  и  $\cdot$  – это стандартное сложение векторов и умножение на скаляр. Кроме того, для упрощения нотации мы будем обозначать векторы  $\mathcal{V}$  как  $x \in V$ . ◆

**ПРИМЕЧАНИЕ** Векторные пространства  $\mathbb{R}^n$ ,  $\mathbb{R}^{n \times 1}$ ,  $\mathbb{R}^{1 \times n}$  отличаются только способом записи векторов. В дальнейшем мы не будем делать разницы между  $\mathbb{R}^n$  и  $\mathbb{R}^{n \times 1}$ , что позволит нам записывать  $n$ -кортежи как *вектор-столбец*.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}. \quad (2.64)$$

Таким образом упрощается нотация, описывающая операции над векторными пространствами. Однако мы различаем  $\mathbb{R}^{n \times 1}$  и  $\mathbb{R}^{1 \times n}$  (векторы-строки), чтобы избежать путаницы с перемножением матриц. По умолчанию мы обозначаем вектор-столбец через  $\mathbf{x}$ , а вектор-строку через  $\mathbf{x}^\top$ , это *транспозиция*  $\mathbf{x}$ . ♦

### 2.4.3. Векторные подпространства

Далее мы познакомимся с векторными подпространствами. Интуитивно понятно, что это множества, содержащиеся в исходном векторном пространстве и обладающие таким свойством: при выполнении операций векторного пространства над элементами из этого подпространства мы никогда не покидаем это подпространство. В данном смысле векторные подпространства «замкнуты». Векторные подпространства — это ключевая идея в машинном обучении. Например, в главе 10 продемонстрировано, как использовать векторные подпространства для снижения размерности.

**Определение 2.10 (векторное подпространство).** Пусть  $V = (\mathcal{V}, +, \cdot)$  это векторное пространство, и  $\mathcal{U} \subseteq \mathcal{V}$ ,  $\mathcal{U} \neq \emptyset$ . Тогда  $U = (\mathcal{U}, +, \cdot)$  называется *векторным подпространством*  $V$  (или *линейным подпространством*), если  $U$  — это векторное пространство с операциями векторного пространства  $+$  и  $\cdot$ , ограниченное  $\mathcal{U} \times \mathcal{U}$  и  $\mathbb{R} \times \mathcal{U}$ . При помощи записи  $U \subseteq V$  мы обозначаем подпространство  $U$  от  $V$ .

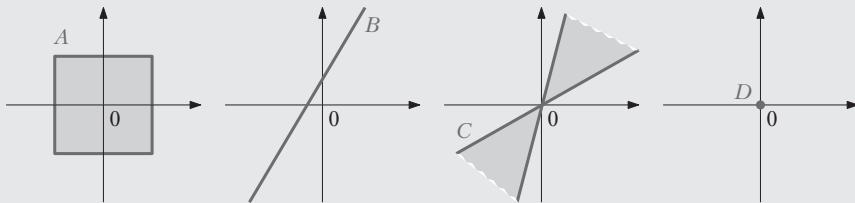
Если и  $\mathcal{U} \subseteq \mathcal{V}$ , а  $V$  — векторное пространство, то естественно, что  $U$  наследует некоторые свойства напрямую от  $V$ , поскольку они соблюдаются для всех  $\mathbf{x} \in \mathcal{V}$  и, в частности, для всех  $\mathbf{x} \in \mathcal{U} \subseteq \mathcal{V}$ . Сюда входят свойства абелевых групп, дистрибутивность, ассоциативность и нейтральный элемент. Чтобы определить, является ли  $(\mathcal{U}, +, \cdot)$  подпространством  $V$ , мы должны показать, что:

1.  $\mathcal{U} \neq \emptyset$ , в частности  $\mathbf{0} \in \mathcal{U}$ .
2. Замыкание  $U$ :
  - a. Относительно внешней операции:  $\forall \lambda \in \mathbb{R} \ \forall \mathbf{x} \in \mathcal{U} : \lambda \mathbf{x} \in \mathcal{U}$ .
  - b. Относительно внутренней операции:  $\forall \mathbf{x}, \mathbf{y} \in \mathcal{U} : \mathbf{x} + \mathbf{y} \in \mathcal{U}$ .

**Пример 2.12 (векторные подпространства)**

Рассмотрим несколько примеров:

- Для любого векторного пространства  $V$  тривиальными подпространствами являются само  $V$  и  $\{\mathbf{0}\}$ .
- Только пример  $D$  на рис. 2.6 является подпространством  $\mathbb{R}^2$  (с обычными внутренними и внешними операциями). В  $A$  и  $C$  свойство замыкания нарушается;  $B$  не содержит  $\mathbf{0}$ .
- Множество решений однородной системы уравнений  $A\mathbf{x} = \mathbf{0}$  с  $n$  неизвестных  $\mathbf{x} = [x_1, \dots, x_n]^T$  является подпространством  $\mathbb{R}^n$ .
- Решение неоднородной системы линейных уравнений  $A\mathbf{x} = \mathbf{b}, \mathbf{b} \neq \mathbf{0}$ , не является подпространством  $\mathbb{R}^n$ .
- Пересечение любого количества подпространств само является подпространством.



**Рис. 2.6.** Не все подмножества  $\mathbb{R}^2$  являются подпространствами. В  $A$  и  $C$  свойство замыкания нарушается;  $B$  не содержит  $\mathbf{0}$ . Только  $D$  является подпространством

**ПРИМЕЧАНИЕ** Каждое подпространство  $U \subseteq (\mathbb{R}^n, +, \cdot)$  является пространством решений однородной системы однородных линейных уравнений  $A\mathbf{x} = \mathbf{0}$  для  $\mathbf{x} \in \mathbb{R}^n$ . ♦

## 2.5. ЛИНЕЙНАЯ НЕЗАВИСИМОСТЬ

Далее подробно рассмотрим, что можно проделывать с векторами (элементами векторного пространства). В частности, векторы можно складывать друг с другом и умножать их на скаляры. Свойство замыкания гарантирует, что в итоге у нас получится другой вектор в том же векторном пространстве. Можно найти такое множество векторов, которое позволит представить любой вектор в векторном пространстве, складывая и умножая интересующие нас векторы. Это множество векторов называется *базис*, мы подробнее обсудим его в разделе 2.6.1.

До того нам понадобится познакомиться с концепциями линейных комбинаций и линейной независимости.

**Определение 2.11 (линейная комбинация).** Рассмотрим векторное пространство  $V$  и конечное множество векторов  $\mathbf{x}_1, \dots, \mathbf{x}_k \in V$ . Тогда любое  $\mathbf{v} \in V$  вида

$$\mathbf{v} = \lambda_1 \mathbf{x}_1 + \dots + \lambda_k \mathbf{x}_k = \sum_{i=1}^k \lambda_i \mathbf{x}_i \in V \quad (2.65)$$

при  $\lambda_1, \dots, \lambda_k \in \mathbb{R}$  является *линейной комбинацией* векторов  $x_1, \dots, x_k$ .

**0-вектор** всегда можно записать как линейную комбинацию  $k$  векторов  $x_1, \dots, x_k$ , поскольку  $\mathbf{0} = \sum_{i=1}^k 0 \mathbf{x}_i$  всегда верно. В дальнейшем нас будут интересовать нетривиальные линейные комбинации множества векторов для представления **0**, то есть линейные комбинации векторов  $x_1, \dots, x_k$ , где не все коэффициенты  $\lambda_i$  в (2.65) равны 0.

**Определение 2.12 (линейная (не)зависимость).** Рассмотрим векторное пространство  $V$  с  $k \in \mathbb{N}$  и  $\mathbf{x}_1, \dots, \mathbf{x}_k \in V$ . Если это нетривиальная линейная комбинация, такая что  $\mathbf{0} = \sum_{i=1}^k \lambda_i \mathbf{x}_i$ , при как минимум одном  $\lambda_i \neq 0$ , то векторы  $x_1, \dots, x_k$  *линейно зависимы*. Если существует только тривиальное решение, то есть  $\lambda_1 = \dots = \lambda_k = 0$ , то векторы  $\mathbf{x}_1, \dots, \mathbf{x}_k$  *линейно независимы*.

Линейная независимость — одна из важнейших концепций в линейной алгебре. Интуитивно понятно, что множество линейно независимых векторов состоит из векторов, не имеющих избыточности. То есть если удалить любые из этих векторов из множества, то какая-то информация будет утрачена. В следующих разделах мы точнее формализуем это интуитивное представление.

### Пример 2.13 (линейно зависимые векторы)

Чтобы прояснить концепцию линейной зависимости, приведем пример из географии. Житель Найроби (Кения), рассказывая, где находится город Кигали (Руанда), может сказать: «Чтобы добраться до Кигали, сначала нужно проехать на 506 километров к северо-западу в Кампалу (Уганда), а оттуда 374 километра на юго-запад». Этой информации достаточно, чтобы описать географическое местонахождение Кигали, так как систему географических координат можно считать двумерным векторным пространством (игнорируя высоту и кривизну земной поверхности). Еще кениец может добавить: «Это примерно в 751 километре к западу отсюда». Хотя это утверждение и верно, при наличии вышеприведенной информации оно не является необходимым, чтобы найти Кигали (в качестве иллюстрации см. рис. 2.7). В этом примере вектор «506 км

к северо-западу» и вектор «374 км к юго-западу» линейно независимы. Это означает, что юго-западный вектор нельзя описать в терминах северо-западного вектора и наоборот. Однако третий вектор «751 км к западу» является линейной комбинацией двух векторов, поэтому из-за него множество векторов становится линейно зависимым. Эквивалентно, имея «751 км к западу» и «374 км к юго-западу» можно линейно скомбинировать, чтобы получить «506 км к северо-западу».



**Рис. 2.7.** Географический пример (с грубыми приближениями к сторонам света), иллюстрирующий линейно зависимые векторы в двумерном пространстве (на плоскости)

**ПРИМЕЧАНИЕ** Следующие свойства пригодятся, если необходимо выяснить, являются ли векторы линейно независимыми.

- $k$ -векторы являются либо линейно зависимыми, либо линейно независимыми. Третьего не дано.
- Если как минимум один из векторов  $\mathbf{x}_1, \dots, \mathbf{x}_k$  равен  $\mathbf{0}$ , то они линейно зависимы. То же верно, если два вектора идентичны.
- Векторы  $\{\mathbf{x}_1, \dots, \mathbf{x}_k : \mathbf{x}_i \neq \mathbf{0}, i = 1, \dots, k\}$ ,  $k \geq 2$  линейно зависимы тогда и только тогда, если (как минимум) один из них является линейной комбинацией других. В частности, если один вектор является кратным другому вектору, то есть  $\mathbf{x}_i = \lambda \mathbf{x}_j$ ,  $\lambda \in \mathbb{R}$ , то множество  $\{\mathbf{x}_1, \dots, \mathbf{x}_k : \mathbf{x}_i \neq \mathbf{0}, i = 1, \dots, k\}$  является линейно зависимым.
- На практике, чтобы проверить, являются ли векторы  $\mathbf{x}_1, \dots, \mathbf{x}_k \in V$  линейно независимыми, удобно использовать гауссово исключение. Запишите все

векторы как столбцы матрицы  $A$  и выполняйте гауссово исключение до тех пор, пока матрица не примет ступенчатый вид (приведенный ступенчатый вид здесь не является необходимым):

- Ведущие столбцы указывают векторы, линейно независимые от векторов, расположенных слева от них. Обратите внимание: при построении матрицы соблюдается порядок векторов.
- Неведущие столбцы можно выразить как линейные комбинации ведущих столбцов, расположенных слева от них. Например, ступенчатый вид

$$\begin{bmatrix} 1 & 3 & 0 \\ 0 & 0 & 2 \end{bmatrix} \quad (2.66)$$

позволяет заключить, что первый и третий столбец являются ведущими. Второй столбец является неведущим, поскольку равен первому, умноженному на три. ♦

Все векторы-столбцы являются линейно независимыми тогда и только тогда, когда все столбцы являются ведущими. При наличии хотя бы одного неведущего столбца столбцы (и, следовательно, соответствующие им векторы) являются линейно зависимыми.

#### Пример 2.14

Рассмотрим  $\mathbb{R}^4$  с

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \\ -3 \\ 4 \end{bmatrix}, \mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 2 \end{bmatrix}, \mathbf{x}_3 = \begin{bmatrix} -1 \\ -2 \\ 1 \\ 1 \end{bmatrix}. \quad (2.67)$$

Чтобы проверить, являются ли они линейно зависимыми, воспользуемся обычным подходом и решим

$$\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2 + \lambda_3 \mathbf{x}_3 = \lambda_1 \begin{bmatrix} 1 \\ 2 \\ -3 \\ 4 \end{bmatrix} + \lambda_2 \begin{bmatrix} 1 \\ 1 \\ 0 \\ 2 \end{bmatrix} + \lambda_3 \begin{bmatrix} -1 \\ -2 \\ 1 \\ 1 \end{bmatrix} = \mathbf{0} \quad (2.68)$$

для  $\lambda_1 \dots \lambda_3$ . Запишем векторы  $\mathbf{x}_i, i = 1, 2, 3$  как столбцы матрицы и будем применять элементарные строковые операции, до тех пор пока не выявим ведущие столбцы:

$$\left[ \begin{array}{ccc} 1 & 1 & -1 \\ 2 & 1 & -2 \\ -3 & 0 & 1 \\ 4 & 2 & 1 \end{array} \right] \rightsquigarrow \dots \rightsquigarrow \left[ \begin{array}{ccc} 1 & 1 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{array} \right]. \quad (2.69)$$

Здесь каждый из столбцов матрицы является ведущим. Следовательно, нетривиального решения нет, и нам потребуется  $\lambda_1 = 0, \lambda_2 = 0, \lambda_3 = 0$ , чтобы решить систему уравнений. Следовательно, векторы  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$  линейно независимы.

**ПРИМЕЧАНИЕ** Рассмотрим векторное пространство  $V$  с  $k$  линейно независимыми векторами  $\mathbf{b}_1, \dots, \mathbf{b}_k$  и  $m$  линейных комбинаций

$$\begin{aligned} \mathbf{x}_1 &= \sum_{i=1}^k \lambda_{i1} \mathbf{b}_i, \\ &\vdots \\ \mathbf{x}_m &= \sum_{i=1}^k \lambda_{im} \mathbf{b}_i. \end{aligned} \quad (2.70)$$

Определяя  $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_k]$  как матрицу, чьи столбцы являются линейно независимыми векторами  $\mathbf{b}_1, \dots, \mathbf{b}_k$ , можно записать:

$$\mathbf{x}_j = \mathbf{B}\boldsymbol{\lambda}_j, \boldsymbol{\lambda}_j = \begin{bmatrix} \lambda_{1j} \\ \vdots \\ \lambda_{kj} \end{bmatrix}, j = 1, \dots, m, \quad (2.71)$$

в более компактной форме.

Мы хотим проверить, являются ли  $\mathbf{x}_1, \dots, \mathbf{x}_m$  линейно независимыми. Для этого воспользуемся обычным подходом и проверим, в самом ли деле  $\sum_{j=1}^m \psi_j \mathbf{x}_j = 0$ . При помощи (2.71) получим

$$\sum_{j=1}^m \psi_j \mathbf{x}_j = \sum_{j=1}^m \psi_j \mathbf{B}\boldsymbol{\lambda}_j = \mathbf{B} \sum_{j=1}^m \psi_j \boldsymbol{\lambda}_j. \quad (2.72)$$

Таким образом,  $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$  линейно независимы тогда и только тогда, когда векторы-столбцы  $\{\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_m\}$  линейно независимы. ◆

**ПРИМЕЧАНИЕ** В векторном пространстве  $V$ ,  $m$  линейные комбинации  $k$  векторов  $\mathbf{x}_1, \dots, \mathbf{x}_k$  линейно зависимы, если  $m > k$ . ◆

**Пример 2.15**

Рассмотрим множество линейно независимых векторов  $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4 \in \mathbb{R}^n$  и

$$\begin{aligned}\mathbf{x}_1 &= \mathbf{b}_1 - 2\mathbf{b}_2 + \mathbf{b}_3 - \mathbf{b}_4 \\ \mathbf{x}_2 &= -4\mathbf{b}_1 - 2\mathbf{b}_2 + 4\mathbf{b}_4 \\ \mathbf{x}_3 &= 2\mathbf{b}_1 + 3\mathbf{b}_2 - \mathbf{b}_3 - 3\mathbf{b}_4 \\ \mathbf{x}_4 &= 17\mathbf{b}_1 - 10\mathbf{b}_2 + 11\mathbf{b}_3 + \mathbf{b}_4\end{aligned}\quad (2.73)$$

Являются ли векторы  $\mathbf{x}_1, \dots, \mathbf{x}_4 \in \mathbb{R}^n$  линейно независимыми? Чтобы ответить на этот вопрос, проверим, являются ли векторы-столбцы

$$\left\{ \begin{bmatrix} 1 \\ -2 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} -4 \\ -2 \\ 0 \\ 4 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \\ -1 \\ -3 \end{bmatrix}, \begin{bmatrix} 17 \\ -10 \\ 11 \\ 1 \end{bmatrix} \right\} \quad (2.74)$$

линейно независимыми. Приведенный ступенчатый вид соответствующей линейной системы уравнений с матрицей коэффициентов

$$\mathbf{A} = \begin{bmatrix} 1 & -4 & 2 & 17 \\ -2 & -2 & 3 & -10 \\ 1 & 0 & -1 & 11 \\ -1 & 4 & -3 & 1 \end{bmatrix} \quad (2.75)$$

дается как

$$\begin{bmatrix} 1 & 0 & 0 & -7 \\ 0 & 1 & 0 & -15 \\ 0 & 0 & 1 & -18 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (2.76)$$

Как видим, соответствующая система линейных уравнений имеет нетривиальное решение: последний столбец не является ведущим, а  $\mathbf{x}_4 = -7\mathbf{x}_1 - 15\mathbf{x}_2 - 18\mathbf{x}_3$ . Следовательно,  $\mathbf{x}_1, \dots, \mathbf{x}_4$  линейно зависимы, поскольку  $\mathbf{x}_4$  может быть выражено как линейная комбинация  $\mathbf{x}_1, \dots, \mathbf{x}_3$ .

## 2.6. БАЗИС И РАНГ

В векторном пространстве  $V$  нас особенно интересуют множества векторов  $\mathcal{A}$ , таких что любой вектор  $\mathbf{v} \in V$  можно получить линейной комбинацией векторов из  $\mathcal{A}$ . Эти векторы являются особенными, и далее мы их охарактеризуем.

## 2.6.1. Генерация множества и базиса

**Определение 2.13 (генерация множества и оболочки).** Рассмотрим векторное пространство  $V = (\mathcal{V}, +, \cdot)$  и множество векторов  $\mathcal{A} = \{\mathbf{x}_1, \dots, \mathbf{x}_k\} \subseteq \mathcal{V}$ . Если каждый вектор  $v \in \mathcal{V}$  можно выразить как линейную комбинацию  $\mathbf{x}_1, \dots, \mathbf{x}_k$ , то  $\mathcal{A}$  называется *порождающим множеством*  $V$ . Множество всех линейных комбинаций векторов в  $\mathcal{A}$  называется *оболочкой*  $\mathcal{A}$ . Если  $\mathcal{A}$  охватывает векторное пространство  $V$ , то мы пишем  $V = \text{span}[\mathcal{A}]$  или  $V = \text{span}[\mathbf{x}_1, \dots, \mathbf{x}_k]$ .

Порождающие множества — это множества векторов, охватывающие (под)пространства векторов, то есть каждый вектор можно представить, как линейную комбинацию векторов из порождающего множества. Теперь углубимся в детали и опишем минимальное порождающее множество, охватывающее (под)пространство векторов.

**Определение 2.14 (базис).** Рассмотрим векторное пространство  $V = (\mathcal{V}, +, \cdot)$  и  $\mathcal{A} \subseteq \mathcal{V}$ . Порождающее множество  $\mathcal{A}$  от  $V$  называется *минимальным*, если не существует меньшего множества  $\tilde{\mathcal{A}} \subseteq \mathcal{A} \subseteq \mathcal{V}$ , которое охватывает  $V$ . Каждое линейно независимое порождающее множество  $V$  является минимальным и называется *базисом*<sup>1</sup>  $V$ .

Пусть  $V = (\mathcal{V}, +, \cdot)$  векторное пространство и  $\mathcal{B} \subseteq \mathcal{V}$ ,  $\mathcal{B} \neq \emptyset$ . Тогда следующие утверждения эквивалентны:

- $\mathcal{B}$  это базис  $V$ .
- $\mathcal{B}$  это минимальное порождающее множество.
- $\mathcal{B}$  это максимальное линейно независимое множество векторов в  $V$ , то есть при добавлении любого другого вектора в это множество оно станет линейно зависимым.
- Каждый вектор  $\mathbf{x} \in V$  является линейной комбинацией векторов из  $\mathcal{B}$ , и каждая линейная комбинация уникальна, то есть при

$$\mathbf{x} = \sum_{i=1}^k \lambda_i \mathbf{b}_i = \sum_{i=1}^k \psi_i \mathbf{b}_i \quad (2.77)$$

и  $\lambda_i, \psi_i \in \mathbb{R}$ ,  $\mathbf{b}_i \in \mathcal{B}$  следует, что  $\lambda_i = \psi_i$ ,  $i = 1, \dots, k$ .

---

<sup>1</sup> Базис — это минимальное порождающее множество и максимальное линейно независимое множество векторов.

**Пример 2.16**

- В  $\mathbb{R}^3$  канонический/стандартный базис — это

$$\mathcal{B} = \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\}. \quad (2.78)$$

- Другие базисы в  $\mathbb{R}^3$  — это

$$\mathcal{B}_1 = \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}, \quad \mathcal{B}_2 = \left\{ \begin{bmatrix} 0,5 \\ 0,8 \\ 0,4 \end{bmatrix}, \begin{bmatrix} 1,8 \\ 0,3 \\ 0,3 \end{bmatrix}, \begin{bmatrix} -2,2 \\ -1,3 \\ 3,5 \end{bmatrix} \right\}. \quad (2.79)$$

- Множество

$$\mathcal{A} = \left\{ \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}, \begin{bmatrix} 2 \\ -1 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ -4 \end{bmatrix} \right\} \quad (2.80)$$

является линейно независимым, но не порождающим (и не базисным) множеством  $\mathbb{R}^4$ : например, вектор  $[1, 0, 0, 0]^T$  невозможно получить линейной комбинацией элементов в  $\mathcal{A}$ .

**ПРИМЕЧАНИЕ** Каждое векторное пространство  $V$  обладает базисом  $\mathcal{B}$ . Как понятно из предыдущих примеров, у векторного пространства  $V$  может быть много базисов, то есть базис не является уникальным. Однако в каждом из базисов содержится одинаковое количество элементов — *базисных векторов*.

Мы рассматриваем только конечномерные векторные пространства  $V$ . В таком случае *размерность* пространства  $V$  — это количество базисных векторов в  $V$ , что записывается как  $\dim(V)$ <sup>1</sup>. Если  $U \subseteq V$  является подпространством  $V$ , то  $\dim(U) \leq \dim(V)$ , а  $\dim(U) = \dim(V)$  тогда и только тогда, когда  $U = V$ . Интуитивно размерность векторного пространства можно представить как количество независимых направлений в данном векторном пространстве.

---

<sup>1</sup> Размерность векторного пространства соответствует количеству его базисных векторов.

**ПРИМЕЧАНИЕ** Размерность векторного пространства не обязательно равна количеству элементов в векторе. Например, векторное пространство  $V = \text{span} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$  является одномерным, хотя в его базисном векторе два элемента.



**ПРИМЕЧАНИЕ** Найти базис подпространства  $U = \text{span} [\mathbf{x}_1, \dots, \mathbf{x}_m] \subseteq \mathbb{R}^n$  можно, выполнив следующие шаги:

1. Записать векторы оболочки как столбцы матрицы  $A$ .
2. Определить ступенчатый вид матрицы  $A$ .
3. Векторы оболочки, связанные с ведущими столбцами, будут базисом  $U$ .



### Пример 2.17 (определение базиса)

Для векторного подпространства  $U \subseteq \mathbb{R}^5$ , чьей оболочкой являются векторы

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \\ -1 \\ -1 \\ -1 \end{bmatrix}, \mathbf{x}_2 = \begin{bmatrix} 2 \\ -1 \\ 1 \\ 2 \\ -2 \end{bmatrix}, \mathbf{x}_3 = \begin{bmatrix} 3 \\ -4 \\ 3 \\ 5 \\ -3 \end{bmatrix}, \mathbf{x}_4 = \begin{bmatrix} -1 \\ 8 \\ -5 \\ -6 \\ 1 \end{bmatrix} \in \mathbb{R}^5, \quad (2.81)$$

мы хотим узнать, какие из векторов  $\mathbf{x}_1, \dots, \mathbf{x}_4$  являются базисом  $U$ . Для этого нужно проверить, являются ли  $\mathbf{x}_1, \dots, \mathbf{x}_4$  линейно независимыми. Следовательно, мы должны решить уравнение

$$\sum_{i=1}^4 \lambda_i \mathbf{x}_i = \mathbf{0}, \quad (2.82)$$

что даст нам однородную систему уравнений с матрицей

$$[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4] = \begin{bmatrix} 1 & 2 & 3 & -1 \\ 2 & -1 & -4 & 8 \\ -1 & 1 & 3 & -5 \\ -1 & 2 & 5 & -6 \\ -1 & -2 & -3 & 1 \end{bmatrix}. \quad (2.83)$$

Воспользовавшись базовыми правилами преобразования для систем линейных уравнений, получим ступенчатый вид

$$\left[ \begin{array}{cccc} 1 & 2 & 3 & -1 \\ 2 & -1 & -4 & 8 \\ -1 & 1 & 3 & -5 \\ -1 & 2 & 5 & -6 \\ -1 & -2 & -3 & 1 \end{array} \right] \rightsquigarrow \cdots \rightsquigarrow \left[ \begin{array}{cccc} 1 & 2 & 3 & -1 \\ 0 & 1 & 2 & -2 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right].$$

Поскольку по ведущим столбцам понятно, какие множества векторов являются линейно независимыми, мы заключаем по ступенчатому виду, что  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4$  линейно независимы (поскольку система линейных уравнений  $\lambda_1\mathbf{x}_1 + \lambda_2\mathbf{x}_2 + \lambda_4\mathbf{x}_4 = \mathbf{0}$  решаема только при  $\lambda_1 = \lambda_2 = \lambda_4 = 0$ ). Следовательно,  $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4\}$  является базисом  $U$ .

## 2.6.2. Ранг

Количество линейно независимых столбцов матрицы  $A \in \mathbb{R}^{m \times n}$  равно количеству линейно независимых строк и называется *рангом*  $A$ , обозначается  $\text{rk}(A)$ .

**ПРИМЕЧАНИЕ** У ранга матрицы есть ряд важных свойств:

- $\text{rk}(A) = \text{rk}(A^T)$ , то есть столбцовый ранг равен строчному.
- Столбцы  $A \in \mathbb{R}^{m \times n}$  охватывают подпространство  $U \subseteq \mathbb{R}^m$  с  $\dim(U) = \text{rk}(A)$ . Далее мы будем называть это подпространство *образом* или *диапазоном*. Базис  $U$  можно найти, применив гауссово исключение к  $A$ , чтобы выявить ведущие столбцы.
- Строки  $A \in \mathbb{R}^{m \times n}$  охватывают подпространство  $W \subseteq \mathbb{R}^n$  с  $\dim(W) = \text{rk}(A)$ . Базис  $W$  можно найти, применив гауссово исключение к  $A^T$ .
- Для всех  $A \in \mathbb{R}^{n \times n}$  соблюдается правило, что  $A$  регулярная (необратимая) тогда и только тогда, когда  $\text{rk}(A) = n$ .
- Для всех  $A \in \mathbb{R}^{m \times n}$  и  $\mathbf{b} \in \mathbb{R}^m$  линейная система уравнений  $A\mathbf{x} = \mathbf{b}$  может быть решена тогда и только тогда, когда  $\text{rk}(A) = \text{rk}(A | \mathbf{b})$ , где  $A | \mathbf{b}$  обозначает расширенную систему.
- Для  $A \in \mathbb{R}^{m \times n}$  подпространство решений  $A\mathbf{x} = \mathbf{0}$  обладает размерностью  $n - \text{rk}(A)$ . Далее мы будем называть это подпространство *ядром* или *нулевым пространством*.
- Матрица  $A \in \mathbb{R}^{m \times n}$  имеет *полный ранг*, если ее ранг равен максимальному возможному рангу матрицы с той же размерностью. Таким образом, ранг матрицы равен наименьшему количеству строк и столбцов, то есть  $\text{rk}(A) = \min(m, n)$ . Ранг матрицы называется *неполным*, если полным рангом она не обладает.



**Пример 2.18 (ранг)**

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

У  $A$  два линейно независимых столбца и две линейно независимые строки, так что  $\text{rk}(A) = 2$ .

$$A = \begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}.$$

Для определения ранга используем гауссово исключение:

$$\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix} \rightsquigarrow \dots \rightsquigarrow \begin{bmatrix} 1 & 2 & 1 \\ 0 & 1 & 3 \\ 0 & 0 & 0 \end{bmatrix}. \quad (2.84)$$

Здесь мы видим, что количество линейно независимых строк и столбцов равно 2, поэтому  $\text{rk}(A) = 2$ .

## 2.7. ЛИНЕЙНЫЕ ОТОБРАЖЕНИЯ

Далее мы изучим отображения на векторные пространства, позволяющие сохранять их структуру. Это позволит нам определить концепцию координат. В начале этой главы было сказано, что векторы — это объекты, которые можно складывать друг с другом и умножать на скаляр, а получаемый в результате объект все равно является вектором. Мы хотим сохранить это свойство и при применении отображений. Рассмотрим два вещественных векторных пространства  $V, W$ . Отображение  $\Phi : V \rightarrow W$  сохраняет структуру векторного пространства, если

$$\Phi(\mathbf{x} + \mathbf{y}) = \Phi(\mathbf{x}) + \Phi(\mathbf{y}) \quad (2.85)$$

$$\Phi(\lambda \mathbf{x}) = \lambda \Phi(\mathbf{x}) \quad (2.86)$$

для всех  $\mathbf{x}, \mathbf{y} \in V$  и  $\lambda \in \mathbb{R}$ . Данные отношения можно резюмировать в виде следующего определения:

**Определение 2.15 (линейное отображение).** Для векторных пространств  $V, W$  отображение  $\Phi : V \rightarrow W$  называется *линейным отображением* (или *гомоморфизмом векторного пространства / линейным преобразованием*), если

$$\forall \mathbf{x}, \mathbf{y} \in V \forall \lambda, \psi \in \mathbb{R} : \Phi(\lambda \mathbf{x} + \psi \mathbf{y}) = \lambda \Phi(\mathbf{x}) + \psi \Phi(\mathbf{y}). \quad (2.87)$$

Оказывается, что линейные отображения можно представлять как матрицы (раздел 2.7.1). Также напоминаем, что можно собрать множество векторов как столбцы матрицы. При работе с матрицами необходимо учитывать, что именно представляет матрица: линейное отображение или совокупность векторов. Подробнее о линейных отображениях мы поговорим в главе 4. Прежде чем продолжить эту тему, кратко познакомимся со специальными отображениями.

**Определение 2.16 (инъективные, сюръективные, биективные отображения).**

Рассмотрим отображение  $\Phi : \mathcal{V} \rightarrow \mathcal{W}$ , где  $\mathcal{V}, \mathcal{W}$  могут быть произвольными множествами. Тогда  $\Phi$  называется

- **инъективным**, если  $\forall \mathbf{x}, \mathbf{y} \in \mathcal{V}: \Phi(\mathbf{x}) = \Phi(\mathbf{y}) \Rightarrow \mathbf{x} = \mathbf{y}$ ;
- **сюръективным**, если  $\Phi: (\mathcal{V}) = \mathcal{W}$ ;
- **биективным**, если оно одновременно является инъективным и сюръективным.

Если  $\Phi$  сюръективно, то каждый элемент  $\mathcal{W}$  «достижим» из  $\mathcal{V}$  при помощи  $\Phi$ . Биективное  $\Phi$  можно «обратить», то есть существует отображение  $\Psi : \mathcal{W} \rightarrow \mathcal{V}$ , такое что  $\Psi \circ \Phi(\mathbf{x}) = \mathbf{x}$ . Такое отображение  $\Psi$  называется обратным  $\Phi$  и обычно обозначается  $\Phi^{-1}$ .

Опираясь на такие определения, введем следующие специальные случаи линейных отображений между векторными пространствами  $V$  и  $W$ :

- **Изоморфизм:**  $\Phi : V \rightarrow W$  линейное и биективное.
- **Эндоморфизм:**  $\Phi : V \rightarrow V$  линейное.
- **Автоморфизм:**  $\Phi : V \rightarrow V$  линейное и биективное.

Мы определяем  $\text{id}_V : V \rightarrow V, \mathbf{x} \mapsto \mathbf{x}$  как  *тождественное отображение* или *тождественный автоморфизм* в  $V$ .

**Пример 2.19 (гомоморфизм)**

Отображение  $\Phi : \mathbb{R}^2 \rightarrow \mathbb{C} \Phi(\mathbf{x}) = \mathbf{x} + ix_2$  является гомоморфизмом:

$$\begin{aligned} \Phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) &= (x_1 + y_1) + i(x_2 + y_2) = x_1 + ix_2 + y_1 + iy_2 = \\ &= \Phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) + \Phi\left(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right); \\ \Phi\left(\lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) &= \lambda x_1 + \lambda ix_2 = \lambda(x_1 + ix_2) = \lambda \Phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right). \end{aligned} \tag{2.88}$$

Это же объясняет, почему комплексные числа представимы в  $\mathbb{R}^2$  в качестве кортежей: существует биективное линейное отображение, которое преобразует поэлементное сложение кортежей в  $\mathbb{R}^2$  в множество комплексных чисел с соответствующим сложением. Обратите внимание: мы показали линейность, но не биективность.

**Теорема 2.17 (теорема 3.59 в Axler (2015)).** Конечномерные векторные пространства  $V$  и  $W$  являются изоморфными тогда и только тогда, если  $\dim(V) = \dim(W)$ .

Согласно теореме 2.17, существует линейное биективное отображение между двумя векторными пространствами одной и той же размерности. Интуитивно из этого следует, что векторные пространства одинаковой размерности — это вещи одного и того же рода, поскольку их можно преобразовывать друг в друга без каких-либо потерь.

Кроме того, теорема 2.17 позволяет трактовать  $\mathbb{R}^{m \times n}$  (векторное пространство из  $m \times n$  матриц) и  $\mathbb{R}^{mn}$  (векторное пространство из векторов длиной  $mn$ ) так же, как если бы их размерности были равны  $mn$  и существовало линейное биективное отображение, преобразующее одно в другое.

**ПРИМЕЧАНИЕ** Рассмотрим векторные пространства  $V, W, X$ . Тогда:

- Для линейных отображений  $\Phi : V \rightarrow W$  и  $\Psi : W \rightarrow X$  линейное отображение  $\Psi \circ \Phi : V \rightarrow X$  также линейное.
- Если  $\Phi : V \rightarrow W$  является изоморфизмом, то и  $\Phi^{-1} : W \rightarrow V$  является изоморфизмом.
- Если  $\Phi : V \rightarrow W, \Psi : W \rightarrow X$  линейны, то  $\Phi + \Psi$  и  $\lambda\Phi, \lambda \in \mathbb{R}$ , также линейны.



## 2.7.1. Матричное представление линейных отображений

Любое  $n$ -мерное векторное пространство изоморфно  $\mathbb{R}^n$  (теорема 2.17). Рассмотрим базис  $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$   $n$ -мерного векторного пространства  $V$ . В дальнейшем будет важен порядок базисных векторов. Следовательно, запишем

$$B = (\mathbf{b}_1, \dots, \mathbf{b}_n) \tag{2.89}$$

и назовем этот  $n$ -кортеж *упорядоченным базисом*  $V$ .

**ПРИМЕЧАНИЕ** Мы добрались до того места, где нотация становится несколько заковыристой. Поэтому обобщим здесь некоторые замечания о ней.

$B = (\mathbf{b}_1, \dots, \mathbf{b}_n)$  это упорядоченный базис,  $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$  это (неупорядоченный) базис, а  $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$  это матрица, чьими столбцами являются векторы  $\mathbf{b}_1, \dots, \mathbf{b}_n$ .



**Определение 2.18 (координаты).** Рассмотрим векторное пространство  $V$  и упорядоченный базис  $B = (\mathbf{b}_1, \dots, \mathbf{b}_n)$  из  $V$ . Для любого  $\mathbf{x} \in V$  мы получаем уникальное представление (линейную комбинацию)

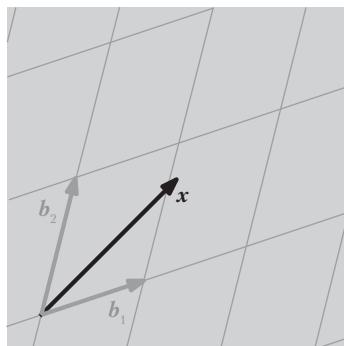
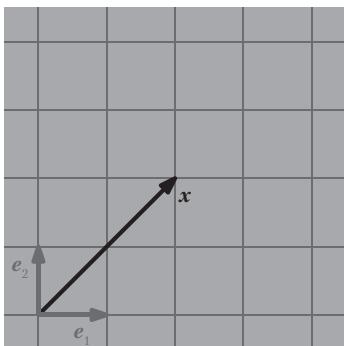
$$\mathbf{x} = \alpha_1 \mathbf{b}_1 + \dots + \alpha_n \mathbf{b}_n \quad (2.90)$$

$\mathbf{x}$  относительно  $B$ . Тогда  $\alpha_1, \dots, \alpha_n$  являются *координатами*  $\mathbf{x}$  относительно  $B$ , а вектор

$$\alpha = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} \in \mathbb{R}^n \quad (2.91)$$

— это *координатный вектор / координатное представление*  $\mathbf{x}$  относительно упорядоченного базиса  $B$ .

Базис фактически определяет координатную систему. Мы знакомы с системой декартовых координат в двух измерениях, которая охватывается каноническими базисными векторами  $\mathbf{e}_1, \mathbf{e}_2$ . В этой координатной системе вектор  $\mathbf{x} \in \mathbb{R}^2$  имеет представление, сообщающее нам, как линейно скомбинировать  $\mathbf{e}_1$  и  $\mathbf{e}_2$ , чтобы получить  $\mathbf{x}$ . Однако любой базис  $\mathbb{R}^2$  определяет действительную координатную систему, и тот же вектор  $\mathbf{x}$ , что показан выше, может иметь иное координатное представление при базисе  $(\mathbf{b}_1, \mathbf{b}_2)$ . На рис. 2.8 координаты  $\mathbf{x}$  относительно стандартного базиса  $(\mathbf{e}_1, \mathbf{e}_2)$  равны  $[2, 2]^T$ . Однако относительно базиса  $(\mathbf{b}_1, \mathbf{b}_2)$  тот

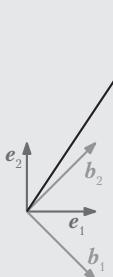


**Рис. 2.8.** Две различные координатные системы, определенные двумя множествами базисных векторов. У вектора  $\mathbf{x}$  разные координатные векторы, зависящие от выбранной координатной системы

же вектор  $\mathbf{x}$  имеет представление  $[1,09, 0,72]^\top$ , то есть  $\mathbf{x} = 1,09\mathbf{b}_1 + 0,72\mathbf{b}_2$ . В следующих разделах будет показано, как получить такое представление.

### Пример 2.20

Рассмотрим геометрический вектор  $\mathbf{x} \in \mathbb{R}^2$  с координатами  $[2, 3]^\top$  относительно стандартного базиса  $(\mathbf{e}_1, \mathbf{e}_2)$  от  $\mathbb{R}^2$ . Таким образом, можно записать  $\mathbf{x} = 2\mathbf{e}_1 + 3\mathbf{e}_2$ . Однако мы не обязаны брать стандартный базис для представления вектора. Если мы воспользуемся базисными векторами  $\mathbf{b}_1 = [1, -1]^\top$ ,  $\mathbf{b}_2 = [1, 1]^\top$ , то получим координаты  $\frac{1}{2}[-1, 5]^\top$ , позволяющие представить тот же самый вектор относительно  $(\mathbf{b}_1, \mathbf{b}_2)$  (рис. 2.9).



**Рис. 2.9.** Различные координатные представления вектора  $x$ , зависящие от выбранного базиса

**ПРИМЕЧАНИЕ** Для  $n$ -мерного векторного пространства  $V$  и упорядоченного базиса  $B$  от  $V$  отображение  $\Phi : \mathbb{R}^n \rightarrow V$ ,  $\Phi(\mathbf{e}_i) = \mathbf{b}_i$ ,  $i = 1 \dots, n$ , линейно (а по теореме 2.17 изоморфно), где  $(\mathbf{e}_1, \dots, \mathbf{e}_n)$  — это стандартный базис  $\mathbb{R}^n$ . ◆

Теперь мы готовы непосредственно связать матрицы и линейные отображения между конечномерными векторными пространствами.

**Определение 2.19 (матрица перехода).** Рассмотрим векторные пространства  $V$ ,  $W$  с соответствующими (упорядоченными) базисами  $B = (\mathbf{b}_1, \dots, \mathbf{b}_n)$  и  $C = (\mathbf{c}_1, \dots, \mathbf{c}_m)$ . Кроме того, рассмотрим линейное отображение  $\Phi : V \rightarrow W$ . Для  $j \in 1, \dots, n$

$$\Phi(\mathbf{b}_j) = \alpha_{1j}\mathbf{c}_1 + \dots + \alpha_{mj}\mathbf{c}_m = \sum_{i=1}^m \alpha_{ij}\mathbf{c}_i \quad (2.92)$$

это уникальное представление  $\Phi(\mathbf{b}_j)$  относительно  $C$ . Далее мы называем  $m \times n$ -матрицу  $A_\Phi$ , чьи элементы даются как

$$A_\Phi(i, j) = \alpha_{ij}, \quad (2.93)$$

матрицей перехода от  $\Phi$  (относительно упорядоченных базисов  $B$  от  $V$  и  $C$  от  $W$ ).

Координаты  $\Phi(\mathbf{b}_j)$  относительно упорядоченного базиса  $C$  от  $W$  – это  $j$ -й столбец  $A_\Phi$ . Рассмотрим (конечномерные) векторные пространства  $V, W$  с упорядоченными базисами  $B, C$  и линейное отображение  $\Phi : V \rightarrow W$  с матрицей перехода  $A_\Phi$ . Если  $\hat{\mathbf{x}}$  – координатный вектор  $\mathbf{x} \in V$  относительно  $B$ , а  $\hat{\mathbf{y}}$  – координатный вектор  $\mathbf{y} = \Phi(\mathbf{x}) \in W$  относительно  $C$ , то

$$\hat{\mathbf{y}} = A_\Phi \hat{\mathbf{x}}. \quad (2.94)$$

Это означает, что матрица перехода может использоваться для отображения координат относительно упорядоченного базиса в  $V$  на координаты относительно упорядоченного базиса  $W$ .

### Пример 2.21 (матрица перехода)

Рассмотрим гомоморфизм  $\Phi : V \rightarrow W$  и упорядоченные базисы  $B = (\mathbf{b}_1, \dots, \mathbf{b}_3)$  и  $C = (\mathbf{c}_1, \dots, \mathbf{c}_4)$  от  $W$ . При

$$\begin{aligned}\Phi(\mathbf{b}_1) &= \mathbf{c}_1 - \mathbf{c}_2 + 3\mathbf{c}_3 - \mathbf{c}_4 \\ \Phi(\mathbf{b}_2) &= 2\mathbf{c}_1 + \mathbf{c}_2 + 7\mathbf{c}_3 + 2\mathbf{c}_4 \\ \Phi(\mathbf{b}_3) &= 3\mathbf{c}_2 + \mathbf{c}_3 + 4\mathbf{c}_4\end{aligned} \quad (2.95)$$

матрица перехода  $A_\Phi$  относительно  $B$  и  $C$  удовлетворяет  $\Phi(\mathbf{b}_k) = \sum_{i=1}^4 a_{ik} \mathbf{c}_i$  для  $k = 1, \dots, 3$  и дается как

$$A_\Phi = [\alpha_1, \alpha_2, \alpha_3] = \begin{bmatrix} 1 & 2 & 0 \\ -1 & 1 & 3 \\ 3 & 7 & 1 \\ -1 & 2 & 4 \end{bmatrix}, \quad (2.96)$$

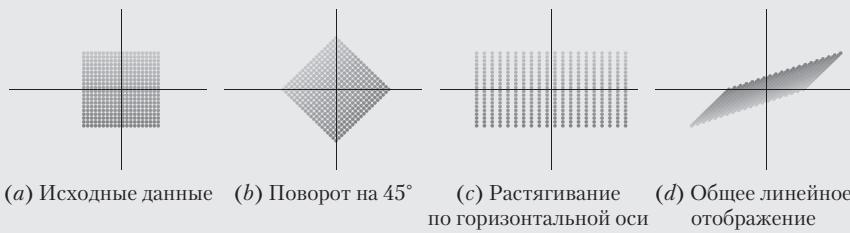
где  $\alpha_j, j = 1, 2, 3$  – это координатные векторы  $\Phi(\mathbf{b}_j)$  относительно  $C$ .

### Пример 2.22 (линейное преобразование векторов)

Рассмотрим три линейных преобразования множества векторов в  $\mathbb{R}^2$  с матрицами перехода

$$A_1 = \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \cos\left(\frac{\pi}{4}\right) \end{bmatrix}, \quad A_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \quad A_3 = \frac{1}{2} \begin{bmatrix} 3 & -1 \\ 1 & -1 \end{bmatrix}. \quad (2.97)$$

На рис. 2.10 приведено три примера линейных преобразований множества векторов. На рис. 2.10(a) показано 400 векторов из  $\mathbb{R}^2$ , каждый из которых представлен точкой с соответствующими координатами  $(x_1, x_2)$ . Векторы расположены квадратом. При использовании матрицы  $A_1$  в (2.97), чтобы линейно преобразовать каждый из этих векторов, мы получим повернутый квадрат, как на рис. 2.10(b). Если мы применим линейное отображение, представленное  $A_2$ , то получим прямоугольник как на рис. 2.10(c), где каждая координата  $x_1$  будет растянута вдвое. На рис. 2.10(d) показан исходный квадрат, тот же, что и на рис. 2.10(a), но линейно преобразованный с использованием  $A_3$ , что является комбинацией отражения, поворота и растягивания.



**Рис. 2.10.** Три примера линейного преобразования векторов, представленных в виде точек в (a): (b) — поворот на  $45^\circ$ , (c) растягивание в горизонтальных координатах в два раза и (d) комбинация отражения, поворота и растягивания

## 2.7.2. Изменение базиса

Далее мы подробнее рассмотрим, как меняются матрицы перехода для линейного отображения  $\Phi : V \rightarrow W$  при изменении базисов в  $V$  и  $W$ . Рассмотрим два упорядоченных базиса

$$B = (\mathbf{b}_1, \dots, \mathbf{b}_n), \tilde{B} = (\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_n) \quad (2.98)$$

из  $V$  и два упорядоченных базиса

$$C = (\mathbf{c}_1, \dots, \mathbf{c}_m), \tilde{C} = (\tilde{\mathbf{c}}_1, \dots, \tilde{\mathbf{c}}_m) \quad (2.99)$$

из  $W$ . Кроме того,  $A_\Phi \in \mathbb{R}^{m \times n}$  это матрица перехода линейного отображения  $\Phi : V \rightarrow W$  относительно базисов  $B$  и  $C$ , а  $\tilde{A}_\Phi \in \mathbb{R}^{m \times n}$  это соответствующее отображение перехода относительно  $\tilde{B}$  и  $\tilde{C}$ . Далее мы исследуем, как связаны  $A$  и  $\tilde{A}$ , то есть как (и можно ли) преобразовать  $A_\Phi$  в  $\tilde{A}_\Phi$ , если мы решим перейти от базисов  $B, C$  к  $\tilde{B}, \tilde{C}$ .

**ПРИМЕЧАНИЕ** Мы фактически получаем различные координатные представления тождественного отображения  $\text{id}_V$ . В контексте рис. 2.9 это означало бы отобразить координаты относительно  $(\mathbf{e}_1, \mathbf{e}_2)$  на координаты относительно  $(\mathbf{b}_1, \mathbf{b}_2)$ , не меняя вектор  $\mathbf{x}$ . Изменив базис и, соответственно, представление векторов, можно получить относительно этого нового базиса матрицу перехода в особенно простой форме, что располагает к прямолинейным вычислениям.



### Пример 2.23 (изменение базиса)

Рассмотрим матрицу перехода

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \quad (2.100)$$

относительно канонического базиса в  $\mathbb{R}^2$ . Если определить новый базис

$$\mathbf{B} = \left( \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \end{bmatrix} \right), \quad (2.101)$$

то получится диагональная матрица перехода

$$\tilde{\mathbf{A}} = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.102)$$

относительно  $\mathbf{B}$ , работать с которой проще, чем с  $\mathbf{A}$ .

В дальнейшем мы рассмотрим отображения, преобразующие координатные векторы, построенные относительно одного базиса, в координатные векторы, построенные относительно другого базиса. Сначала постулируем наш основной результат, а затем дадим ему объяснение.

**Теорема 2.20 (изменение базиса).** Для линейного отображения  $\Phi : V \rightarrow W$  упорядоченные базисы

$$\mathbf{B} = (\mathbf{b}_1, \dots, \mathbf{b}_n), \tilde{\mathbf{B}} = (\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_n) \quad (2.103)$$

от  $V$  и

$$\mathbf{C} = (\mathbf{c}_1, \dots, \mathbf{c}_m), \tilde{\mathbf{C}} = (\tilde{\mathbf{c}}_1, \dots, \tilde{\mathbf{c}}_m) \quad (2.104)$$

от  $W$ , а также матрицы перехода  $\mathbf{A}_\Phi$  от  $\Phi$  относительно  $\mathbf{B}$  и  $\mathbf{C}$ , соответствующая матрица перехода  $\tilde{\mathbf{A}}_\Phi$  относительно базисов  $\tilde{\mathbf{B}}$  и  $\tilde{\mathbf{C}}$ , дается как

$$\tilde{\mathbf{A}}_\Phi = \mathbf{T}^{-1} \mathbf{A}_\Phi \mathbf{S}. \quad (2.105)$$

Здесь  $S \in \mathbb{R}^{n \times n}$  это матрица перехода  $\text{id}_V$ , отображающая координаты относительно  $\tilde{B}$  на координаты относительно  $B$ , а  $T \in \mathbb{R}^{m \times m}$  это матрица перехода  $\text{id}_W$ , отображающая координаты относительно  $\tilde{C}$  на координаты относительно  $C$ .

*Доказательство.* Согласно Drummond and Weil (2001), можно записать векторы нового базиса  $\tilde{B}$  от  $V$  как линейную комбинацию базисных векторов  $B$ , так что

$$\tilde{b}_j = s_{1j} b_1 + \cdots + s_{nj} b_n = \sum_{i=1}^n s_{ij} b_i, j = 1, \dots, n. \quad (2.106)$$

Аналогично, мы записываем новые базисные векторы  $\tilde{C}$  от  $W$  как линейную комбинацию базисных векторов  $C$ , что дает нам

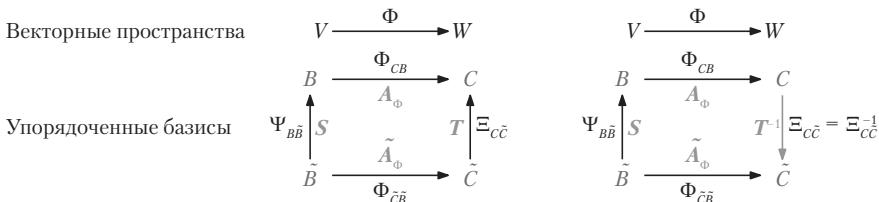
$$\tilde{c}_k = t_{1k} c_1 + \cdots + t_{mk} c_m = \sum_{l=1}^m t_{lk} c_l, k = 1, \dots, m. \quad (2.107)$$

Мы определяем  $S = ((s_{ij})) \in \mathbb{R}^{n \times n}$  как матрицу перехода, отображающую координаты относительно  $\tilde{B}$  на координаты относительно  $B$ , и  $T = ((t_{lk})) \in \mathbb{R}^{m \times m}$  как матрицу перехода, отображающую координаты относительно  $\tilde{C}$  на координаты относительно  $C$ . В частности,  $j$ -й столбец  $S$  – это координатное представление  $\tilde{b}_j$  относительно  $B$ , а  $k$ -й столбец  $T$  – это координатное представление  $\tilde{c}_k$  относительно  $C$ . Обратите внимание: как  $S$ , так и  $T$  являются регулярными.

Далее рассмотрим  $\Phi(\tilde{b}_j)$  с двух точек зрения. Сначала, применив отображение  $\Phi$ , получим, что для всех  $j \in 1, \dots, n$ ,

$$\Phi(\tilde{b}_j) = \sum_{k=1}^m \underbrace{\tilde{a}_{kj}}_{\in W} \tilde{c}_k \stackrel{(2.107)}{=} \sum_{k=1}^m \tilde{a}_{kj} \sum_{l=1}^m t_{lk} c_l = \sum_{l=1}^m \left( \sum_{k=1}^m t_{lk} \tilde{a}_{kj} \right) c_l, \quad (2.108)$$

где впервые выражаем новые базисные векторы  $\tilde{c}_k \in W$  как линейные комбинации базисных векторов  $c_l \in W$ , а затем меняем порядок суммирования.



**Рис. 2.11.** Для гомоморфизма  $\Phi: V \rightarrow W$  и упорядоченных базисов  $B, \tilde{B}$  от  $V$  и  $C, \tilde{C}$  от  $W$  можно выразить отображение  $\Phi_{\tilde{C}\tilde{B}}$  относительно  $\tilde{B}, \tilde{C}$  эквивалентно как композицию гомоморфизмов  $\Phi_{\tilde{C}\tilde{B}} = \Xi_{\tilde{C}C} \circ \Phi_{CB} \circ \Psi_{B\tilde{B}}$  относительно базисов, указанных в нижних индексах.

В альтернативном случае, выражая  $\tilde{\mathbf{b}}_j \in V$  как линейные комбинации  $\mathbf{b}_i \in V$ , получаем

$$\Phi(\tilde{\mathbf{b}}_j) \stackrel{(2.106)}{=} \Phi\left(\sum_{i=1}^n s_{ij} \mathbf{b}_i\right) = \sum_{i=1}^n s_{ij} \Phi(\mathbf{b}_i) = \sum_{i=1}^n s_{ij} \sum_{l=1}^m a_{li} c_l = \quad (2.109a)$$

$$= \sum_{l=1}^m \left( \sum_{i=1}^n a_{li} s_{ij} \right) = \mathbf{c}_l, \quad j = 1, \dots, n, \quad (2.109b)$$

здесь мы воспользовались линейностью  $\Phi$ . Из сравнения (2.108) и (2.109b) следует, что для всех  $j \in 1, \dots, n$  и  $l \in 1, \dots, m$  верно

$$\sum_{k=1}^m t_{lk} \tilde{a}_{kj} = \sum_{i=1}^n a_{li} s_{ij} \quad (2.110)$$

и, следовательно

$$\mathbf{T}\tilde{\mathbf{A}}_\Phi = \mathbf{A}_\Phi \mathbf{S} \in \mathbb{R}^{m \times n}, \quad (2.111),$$

так что

$$\tilde{\mathbf{A}}_\Phi = \mathbf{T}^{-1} \mathbf{A}_\Phi \mathbf{S}, \quad (2.112)$$

что является доказательством теоремы 2.20.  $\square$

Теорема 2.20 демонстрирует, что при изменении базиса в  $V$  ( $B$  меняется на  $\tilde{B}$ ) и  $W$  ( $C$  меняется на  $\tilde{C}$ ) матрица перехода  $\mathbf{A}_\Phi$  линейного отображения  $\Phi : V \rightarrow W$  заменяется эквивалентной матрицей  $\tilde{\mathbf{A}}_\Phi$ , такой что

$$\tilde{\mathbf{A}}_\Phi = \mathbf{T}^{-1} \mathbf{A}_\Phi \mathbf{S}. \quad (2.113)$$

Это отношение продемонстрировано на рис. 2.11. Рассмотрим гомоморфизм  $\Phi : V \rightarrow W$  и упорядоченные базисы  $B, \tilde{B}$  от  $V$  и  $C, \tilde{C}$  от  $W$ . Отображение  $\Phi_{CB}$  является примером  $\Phi$  и отображает базисные векторы  $B$  на линейные комбинации базисных векторов  $C$ . Предположим, что нам известна матрица перехода  $\mathbf{A}_\Phi$  от  $\Phi_{CB}$  относительно упорядоченных базисов  $B, C$ . Осуществляя изменение базиса  $B$  на  $\tilde{B}$  в  $V$  и  $C$  на  $\tilde{C}$  в  $W$ , мы можем определить соответствующую матрицу перехода  $\tilde{\mathbf{A}}_\Phi$  следующим образом: сначала находим матричное представление линейного отображения  $\Psi_{B\tilde{B}} : V \rightarrow V$ , отображающее координаты относительно нового базиса  $\tilde{B}$  на «универсальные» координаты, откладываемые относительно «старого» базиса  $B$  (в  $V$ ). Затем используем матрицу перехода  $\mathbf{A}_\Phi$  от  $\Phi_{CB}$ :  $V \rightarrow W$ , чтобы отобразить координаты относительно  $C$  на координаты относительно  $\tilde{C}$ . Следовательно, мы выражаем линейное

отображение  $\Phi_{\tilde{C}\tilde{B}}$  как составление линейных отображений, включающих «старый» базис:

$$\Phi_{\tilde{C}\tilde{B}} = \Xi_{\tilde{C}C} \circ \Phi_{CB} \circ \Psi_{B\tilde{B}} = \Xi_{CC}^{-1} \circ \Phi_{CB} \circ \Psi_{B\tilde{B}}. \quad (2.114)$$

Конкретно, мы используем  $\Psi_{B\tilde{B}} = \text{id}_V$  и  $\Xi_{CC} = \text{id}_W$ , то есть тождественные отображения, отображающие векторы на сами эти векторы, но относительно иного базиса.

**Определение 2.21 (эквивалентность).** Две матрицы  $A, \tilde{A} \in \mathbb{R}^{m \times n}$  *эквивалентны*, если существуют регулярные матрицы  $S \in \mathbb{R}^{n \times n}$  и  $T \in \mathbb{R}^{m \times m}$ , такие что  $\tilde{A} = T^{-1}AS$ .

**Определение 2.22 (подобие).** Две матрицы  $A, \tilde{A} \in \mathbb{R}^{n \times n}$  *подобны*, если существует регулярная матрица  $S \in \mathbb{R}^{n \times n}$  с  $\tilde{A} = S^{-1}AS$ .

**ПРИМЕЧАНИЕ** Подобные матрицы всегда эквивалентны. Но эквивалентные матрицы не обязательно подобны. ♦

**ПРИМЕЧАНИЕ** Рассмотрим векторные пространства  $V, W, X$ . Из примечания к теореме 2.17 нам уже известно, что для отображений  $\Phi: V \rightarrow W$  и  $\Psi: W \rightarrow X$  отображение  $\Psi \circ \Phi: V \rightarrow X$  также линейно. Имея матрицы перехода  $A_\Phi$  и  $A_\Psi$  соответствующих отображений, мы уже знаем, что общая матрица перехода будет иметь вид  $A_{\Psi \circ \Phi} = A_\Psi A_\Phi$ . ♦

В свете этого примечания можно рассмотреть изменения базиса с точки зрения составления линейных отображений:

- $A_\Phi$  это матрица перехода линейного отображения  $\Phi_{CB}: V \rightarrow W$  относительно базисов  $B, C$ .
- $\tilde{A}_\Phi$  это матрица перехода линейного отображения  $\Phi_{\tilde{C}\tilde{B}}: V \rightarrow W$  относительно базисов  $\tilde{B}, \tilde{C}$ .
- $S$  это матрица перехода линейного отображения  $\Psi_{B\tilde{B}}: V \rightarrow V$  (автоморфизм), представляющая  $\tilde{B}$  в терминах  $B$ . Как правило,  $\Psi = \text{id}_V$  это тождественное отображение в  $V$ .
- $T$  это матрица перехода линейного отображения  $\Xi_{CC}: W \rightarrow W$  (автоморфизм), представляющая  $\tilde{C}$  в терминах  $C$ . Как правило,  $\Xi_{CC} = \text{id}_W$  это тождественное отображение в  $W$ .

Если (нестрого) записать переход лишь в терминах базисов, то  $A_\Phi: B \rightarrow C$ ,  $\tilde{A}_\Phi: \tilde{B} \rightarrow \tilde{C}$ ,  $S: \tilde{B} \rightarrow B$ ,  $T: \tilde{C} \rightarrow C$  и

$$\tilde{B} \rightarrow \tilde{C} = \tilde{B} \rightarrow B \rightarrow C \rightarrow \tilde{C}; \quad (2.115)$$

$$\tilde{A}_\Phi = T^{-1}A_\Phi S. \quad (2.116)$$

Обратите внимание, что порядок выполнения в (2.116) – справа налево, так как векторы умножаются в правой части и, соответственно,  $\mathbf{x} \mapsto S\mathbf{x} \mapsto A_\Phi(S\mathbf{x}) \mapsto T^{-1}(A_\Phi(S\mathbf{x})) = \tilde{A}_\Phi \mathbf{x}$ .

### Пример 2.24 (изменение базиса)

Рассмотрим линейное отображение  $\Phi : \mathbb{R}^3 \rightarrow \mathbb{R}^4$ , чья матрица перехода

$$A_\Phi = \begin{bmatrix} 1 & 2 & 0 \\ -1 & 1 & 3 \\ 3 & 7 & 1 \\ -1 & 2 & 4 \end{bmatrix} \quad (2.117)$$

относительно стандартных базисов

$$B = \left( \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right), \quad C = \left( \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right). \quad (2.118)$$

Мы ищем матрицу перехода  $\tilde{A}_\Phi$  от  $\Phi$  относительно новых базисов

$$\tilde{B} = \left( \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \right) \in \mathbb{R}^3, \quad \tilde{C} = \left( \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right). \quad (2.119)$$

Тогда

$$S = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad T = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (2.120)$$

где  $i$ -й столбец  $S$  – это координатное представление  $\tilde{\mathbf{b}}_j$  относительно базисных векторов  $B$ . Поскольку  $B$  это стандартный базис, найти координатное представление легко. Для общего базиса  $B$  нам потребуется решить систему линейных уравнений, чтобы найти  $\lambda_i$ , такое что  $\sum_{i=1}^3 \lambda_i \mathbf{b}_i = \tilde{\mathbf{b}}_j$ ,  $j = 1, \dots, 3$ . Аналогично,  $j$ -й столбец  $T$  – это координатное представление  $\tilde{\mathbf{c}}_j$  относительно базисных векторов  $C$ .

Следовательно, получаем

$$\tilde{\mathbf{A}}_{\Phi} = \mathbf{T}^{-1} \mathbf{A}_{\Phi} \mathbf{S} = \frac{1}{2} \begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} 3 & 2 & 1 \\ 0 & 4 & 2 \\ 10 & 8 & 4 \\ 1 & 6 & 3 \end{bmatrix} = \quad (2.121a)$$

$$= \begin{bmatrix} -4 & -4 & -2 \\ 6 & 0 & 0 \\ 4 & 8 & 4 \\ 1 & 6 & 3 \end{bmatrix}. \quad (2.121b)$$

В главе 4 мы сможем воспользоваться концепцией изменения базиса, чтобы найти базис, относительно которого матрица перехода эндоморфизма имеет особенно простую (диагональную) форму. В главе 10 мы рассмотрим задачу сжатия данных и найдем удобный базис, на который сможем спроектировать данные, минимизировав при этом потери при сжатии.

### 2.7.3. Образ и ядро

Образ и ядро линейного отображения — это векторные подпространства, обладающие некоторыми важными свойствами. Далее мы охарактеризуем их более тщательно.

#### Определение 2.23 (образ и ядро)

Для  $\Phi : V \rightarrow W$  определим *ядро/нулевое пространство*

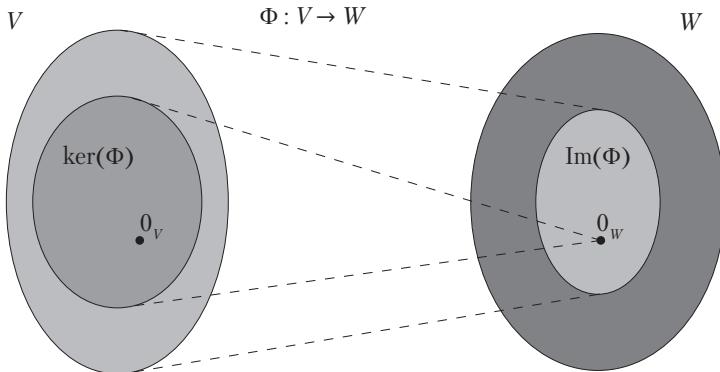
$$\ker(\Phi) := \Phi^{-1}(\mathbf{0}_W) = \{v \in V : \Phi(v) = \mathbf{0}_W\} \quad (2.122)$$

и *образ/диапазон*

$$\text{Im}(\Phi) := \Phi(V) = \{w \in W \mid \exists v \in V : \Phi(v) = w\}. \quad (2.123)$$

Также будем называть  $V$  и  $W$  *областью определения* и *областью значений*  $\Phi$  соответственно.

Интуитивно понятно, что ядро — это множество векторов в  $v \in V$ , отображаемое  $\Phi$  на нейтральный элемент  $\mathbf{0}_W \in W$ . Образ — это множество векторов  $w \in W$ , «достигшее» для  $\Phi$  из любого вектора в  $V$ . Это проиллюстрировано на рис. 2.12.



**Рис. 2.12.** Ядро и образ линейного отображения  $\Phi : V \rightarrow W$

**ПРИМЕЧАНИЕ** Рассмотрим линейное отображение  $\Phi : V \rightarrow W$ , где  $V, W$  – векторные пространства.

- Всегда верно, что  $\Phi(\mathbf{0}_V) = \mathbf{0}_W$ , и, следовательно,  $\mathbf{0}_V \in \ker(\Phi)$ . В частности, нулевое пространство не бывает пустым.
- $\text{Im}(\Phi) \subseteq W$  – это подпространство  $W$ , а  $\ker(\Phi) \subseteq V$  – это подпространство  $V$ .
- $\Phi$  инъектививно (один к одному) тогда и только тогда, когда  $\ker(\Phi) = \{\mathbf{0}\}$ .



**ПРИМЕЧАНИЕ** Рассмотрим  $A \in \mathbb{R}^{m \times n}$  и линейное отображение  $\Phi : \mathbb{R}^n \rightarrow \mathbb{R}^m$ ,  $x \mapsto Ax$ .

- Для  $A = [a_1, \dots, a_n]$ , где  $a_i$  – столбцы  $A$ , получим

$$\text{Im}(\Phi) = \{Ax : x \in \mathbb{R}^n\} = \left\{ \sum_{i=1}^n x_i a_i : x_1, \dots, x_n \in \mathbb{R} \right\} = \quad (2.124a)$$

$$= \text{span}[a_1, \dots, a_n] \subseteq \mathbb{R}^m, \quad (2.124b)$$

то есть образ – это оболочка столбцов  $A$ , также называемая *пространством столбцов*. Следовательно, пространство столбцов (образ) – это подпространство  $\mathbb{R}^m$ , где  $m$  – это «высота» матрицы.

- $\text{rk}(A) = \dim(\text{Im}(\Phi))$ .
- Ядро / нулевое пространство – это общее решение для однородной системы уравнений  $Ax = \mathbf{0}$ , охватывающее все возможные линейные комбинации элементов  $\mathbb{R}^n$ , дающие  $\mathbf{0} \in \mathbb{R}^m$ .

- Ядро — это подпространство  $\mathbb{R}^n$ , где  $n$  — это «ширина» матрицы.
- Суть ядра заключается в описании отношения между столбцами, и его можно использовать, чтобы определить, можно ли (и как) выразить столбец в качестве линейной комбинации других столбцов.



### Пример 2.25 (образ и ядро линейного отображения)

Отображение

$$\Phi: \mathbb{R}^4 \rightarrow \mathbb{R}^2, \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \mapsto \begin{bmatrix} 1 & 2 & -1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_1 + 2x_2 - x_3 \\ x_1 + x_4 \end{bmatrix} = \quad (2.125a)$$

$$= x_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + x_2 \begin{bmatrix} 2 \\ 0 \end{bmatrix} + x_3 \begin{bmatrix} -1 \\ 0 \end{bmatrix} + x_4 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2.125b)$$

линейное. Чтобы определить  $\text{Im}(\Phi)$ , можно взять оболочку столбцов матрицы перехода и получить

$$\text{Im}(\Phi) = \text{span} \left[ \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right]. \quad (2.126)$$

Чтобы вычислить ядро (нулевое пространство)  $\Phi$ , необходимо решить  $A\mathbf{x} = \mathbf{0}$ , то есть однородную систему уравнений. Для этого воспользуемся гауссовым исключением и преобразуем  $A$  в приведенный ступенчатый вид:

$$\begin{bmatrix} 1 & 2 & -1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \rightsquigarrow \dots \rightsquigarrow \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} & -\frac{1}{2} \end{bmatrix}. \quad (2.127)$$

Эта матрица находится в приведенном ступенчатом виде, поэтому можно воспользоваться приемом с минус единицей, чтобы вычислить базис ядра (раздел 2.3.3). В качестве альтернативы можно выразить неведущие столбцы (3 и 4) как линейные комбинации ведущих столбцов (1 и 2). Третий столбец  $\mathbf{a}_3$  эквивалентен произведению второго столбца  $\mathbf{a}_2$  на  $-\frac{1}{2}$ . Следовательно,  $\mathbf{0} = \mathbf{a}_3 + \frac{1}{2}\mathbf{a}_2$ . Аналогично можно убедиться, что  $\mathbf{a}_4 = \mathbf{a}_1 -$

$-\frac{1}{2}\mathbf{a}_2$  и, следовательно,  $\mathbf{0} = \mathbf{a}_1 - \frac{1}{2}\mathbf{a}_2 - \mathbf{a}_4$ . В итоге это дает нам следующее ядро (нулевое пространство):

$$\ker(\Phi) = \text{span} \left[ \begin{bmatrix} 0 \\ \frac{1}{2} \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ \frac{1}{2} \\ 0 \\ 1 \end{bmatrix} \right]. \quad (2.128)$$

**Теорема 2.24 (теорема о ранге и дефекте).** Для векторных пространств  $V, W$  и линейного отображения  $\Phi : V \rightarrow W$  верно, что

$$\dim(\ker(\Phi)) + \dim(\text{Im}(\Phi)) = \dim(V). \quad (2.129)$$

Теорема о ранге и дефекте также именуется *фундаментальной теоремой линейных отображений* (Axler, 2015, теорема 3.22). Нижесказанное прямо следует из теоремы 2.24:

- Если  $\dim(\text{Im}(\Phi)) < \dim(V)$ , то  $\ker(\Phi)$  нетривиально, то есть ядро содержит более чем  $\mathbf{0}_V$  и  $\dim(\ker(\Phi)) \geq 1$ .
- Если  $A_\Phi$  — это матрица перехода  $\Phi$  относительно упорядоченного базиса и  $\dim(\text{Im}(\Phi)) < \dim(V)$ , то система линейных уравнений  $A_\Phi \mathbf{x} = \mathbf{0}$  имеет бесконечно много решений.
- Если  $\dim(V) = \dim(W)$ , то соблюдается следующая тройная эквивалентность:
  - $\Phi$  инъективно;
  - $\Phi$  сюръективно;
  - $\Phi$  биективно.

Так как  $\text{Im}(\Phi) \subseteq W$ .

## 2.8. АФФИННЫЕ ПРОСТРАНСТВА

Далее мы подробнее рассмотрим такие пространства, которые расположены с отступом от начала координат, то есть уже не являющиеся векторными подпространствами. Кроме того, мы кратко обсудим свойства отображений между такими аффинными пространствами, напоминающие линейные отображения.

**ПРИМЕЧАНИЕ** В литературе по машинному обучению различие между линейными и аффинными иногда проводится не очень четко, поэтому можно

найти ссылки, где аффинные пространства/отображения называются линейными пространствами/отображениями.



## 2.8.1. Аффинные подпространства

**Определение 2.25 (аффинное подпространство).** Пусть  $V$  – это векторное пространство,  $\mathbf{x}_0 \in V$  и  $U \subseteq V$  это подпространство. Тогда подмножество

$$L = \mathbf{x}_0 + U := \{\mathbf{x}_0 + \mathbf{u} : \mathbf{u} \in U\} = \quad (2.130a)$$

$$= \{\mathbf{v} \in V \mid \exists \mathbf{u} \in U : \mathbf{v} = \mathbf{x}_0 + \mathbf{u}\} \subseteq V \quad (2.130b)$$

называется *аффинным подпространством* или *линейным многообразием* от  $V$ .  $U$  называется *направлением* или *пространством направлений*, а  $\mathbf{x}_0$  называется *точкой поддержки*. В главе 12 мы будем называть такое подпространство *гиперплоскостью*.

Обратите внимание: определение аффинного подпространства исключает  $\mathbf{0}$ , если  $\mathbf{x}_0 \notin U$ .

Примерами аффинных подпространств являются точки, прямые и плоскости в  $\mathbb{R}^3$ , которые не обязательно проходят через начало координат.

**ПРИМЕЧАНИЕ** Рассмотрим два аффинных подпространства  $L = \mathbf{x}_0 + U$  и  $\tilde{L} = \tilde{\mathbf{x}}_0 + \tilde{U}$  векторного пространства  $V$ . Тогда  $L \subseteq \tilde{L}$  тогда и только тогда, когда  $U \subseteq \tilde{U}$  и  $\mathbf{x}_0 - \tilde{\mathbf{x}}_0 \in \tilde{U}$ .

Аффинные подпространства часто описываются *параметрами*: рассмотрим  $k$ -мерное аффинное пространство  $L = \mathbf{x}_0 + U$  от  $V$ . Если  $(\mathbf{b}_1, \dots, \mathbf{b}_k)$  – это упорядоченный базис  $U$ , то любой элемент  $\mathbf{x} \in L$  можно описать как

$$\mathbf{x} = \mathbf{x}_0 + \lambda_1 \mathbf{b}_1 + \dots + \lambda_k \mathbf{b}_k, \quad (2.131)$$

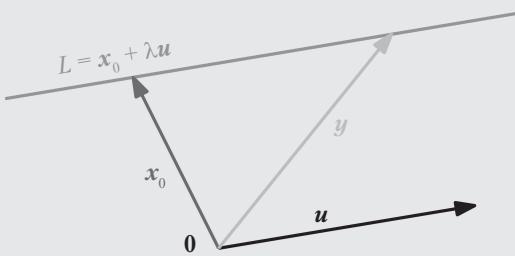
где  $\lambda_1, \dots, \lambda_k \in \mathbb{R}$ . Такое представление называется *параметрическим уравнением*  $L$  с векторами направления  $\mathbf{b}_1, \dots, \mathbf{b}_k$  и *параметрами*  $\lambda_1, \dots, \lambda_k$ .



### Пример 2.26 (аффинные подпространства)

- Одномерные аффинные подпространства называются *линиями* и могут быть записаны как  $\mathbf{y} = \mathbf{x}_0 + \lambda \mathbf{x}_1$ , где  $\lambda \in \mathbb{R}$ , где  $U = \text{span}[\mathbf{x}_1] \in \mathbb{R}^n$  – это одномерное подпространство  $\mathbb{R}^n$ . Это означает, что линия определяется точкой поддержки  $\mathbf{x}_0$  и вектором  $\mathbf{x}_1$ , задающим направление. См. рис. 2.13 в качестве иллюстрации.

- Двумерные аффинные подпространства  $\mathbb{R}^n$  называются *плоскостями*. Параметрическое уравнение для плоскостей записывается как  $\mathbf{y} = \mathbf{x}_0 + \lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2$ , где  $\lambda_1, \lambda_2 \in \mathbb{R}$  и  $U = [\mathbf{x}_1, \mathbf{x}_2] \subseteq \mathbb{R}^n$ . Это означает, что плоскость определяется точкой поддержки  $\mathbf{x}_0$  и двумя линейно независимыми векторами  $\mathbf{x}_1, \mathbf{x}_2$ , которые охватывают пространство направлений.
- В  $\mathbb{R}^n (n - 1)$ -мерные аффинные подпространства называются *гиперплоскостями*, и им соответствует параметрическое уравнение  $\mathbf{y} = \mathbf{x}_0 + \sum_{i=1}^{n-1} \lambda_i \mathbf{x}_i$ , где  $\mathbf{x}_1, \dots, \mathbf{x}_{n-1}$  образуют базис  $(n - 1)$ -мерного подпространства  $U$  от  $\mathbb{R}^n$ . Это означает, что гиперплоскость определяется точкой поддержки и  $(n - 1)$  линейно независимыми векторами  $\mathbf{x}_1, \dots, \mathbf{x}_{n-1}$ , охватывающими пространство направлений. В  $\mathbb{R}^2$  линия также является гиперплоскостью. В  $\mathbb{R}^3$  плоскость также является гиперплоскостью.



**Рис. 2.13.** Векторы  $\mathbf{y}$  на линии лежат в аффинном подпространстве  $L$  с точкой поддержки  $\mathbf{x}_0$  и направлением  $\mathbf{u}$

**ПРИМЕЧАНИЕ** Для  $A \in \mathbb{R}^{m \times n}$  и  $\mathbf{b} \in \mathbb{R}^m$  решением системы уравнений  $A\mathbf{x} = \mathbf{b}$  является либо пустое множество, либо аффинное подпространство  $\mathbb{R}^n$  с раз мерностью  $n - \text{rk}(A)$ . В частности, решение линейного уравнения  $\lambda_1 \mathbf{x}_1 + \dots + \lambda_n \mathbf{x}_n = \mathbf{b}$ , где  $(\lambda_1, \dots, \lambda_n) \neq (0, \dots, 0)$ , является гиперплоскостью в  $\mathbb{R}^n$ .

В  $\mathbb{R}^n$  любое  $k$ -мерное аффинное подпространство является решением линейной неоднородной системы уравнений  $A\mathbf{x} = \mathbf{b}$ , где  $A \in \mathbb{R}^{m \times n}$ ,  $\mathbf{b} \in \mathbb{R}^m$  и  $\text{rk}(A) = n - k$ . Как вы помните, для однородных систем уравнений  $A\mathbf{x} = \mathbf{0}$  решением было векторное подпространство, которое также можно считать особым аффинным пространством с точкой поддержки  $\mathbf{x}_0 = \mathbf{0}$ . ◆

## 2.8.2. Аффинные отображения

Подобно линейным отображениям между векторными пространствами, о чём мы говорили в разделе 2.7, можно определить аффинные отображения между двумя аффинными пространствами. Линейные и аффинные отображения тесно

связаны. Следовательно, многие свойства, уже известные нам по линейным отображениям, например что отображение, составленное из двух линейных отображений, само является линейным отображением, также соблюдаются и для аффинных отображений.

**Определение 2.26 (аффинное отображение).** Для двух векторных пространств  $V, W$  линейного отображения  $\Phi: V \rightarrow W$  и  $\mathbf{a} \in W$ , отображение

$$\varphi: V \rightarrow W \quad (2.132)$$

$$\mathbf{x} \mapsto \mathbf{a} + \Phi(\mathbf{x}) \quad (2.133)$$

является *аффинным отображением* с  $V$  на  $W$ . Вектор  $\mathbf{a}$  называется *вектором трансляции*  $\varphi$ .

- Каждое аффинное отображение  $\varphi: V \rightarrow W$  также является композицией линейного отображения  $\Phi: V \rightarrow W$  и трансляции  $\tau: W \rightarrow W$  в  $W$ , так что  $\varphi = \tau \circ \Phi$ . Отображения  $\Phi$  и  $\tau$  определяются уникально.
- Композиция  $\varphi' \circ \varphi$  аффинных отображений  $\varphi: V \rightarrow W$ ,  $\varphi': W \rightarrow X$  является аффинной.
- Геометрическая структура у аффинных отображений сохраняется инвариантной. Они также сохраняют размерность и параллелизм.

## 2.9. ДОПОЛНИТЕЛЬНОЕ ЧТЕНИЕ

Есть множество ресурсов для изучения линейной алгебры, в том числе учебники Стрэнга (Strang, 2003), Голана (Golan, 2007), Экслера (Axler, 2015), а также Лизена и Мерманна (Liesen and Mehrmann, 2015). Также есть несколько онлайн-ресурсов, которые мы упомянули во введении к этой главе. Здесь мы рассказали только о гауссовом исключении, но существует и много других способов решения систем линейных уравнений, и мы рекомендуем почитать многочисленные книги по линейной алгебре, в частности Stoer and Burlirsch (2002), Golub and Van Loan (2012) и Horn and Johnson (2013), где есть углубленное изучение этих тем.

В этой книге различаются темы, относящиеся к линейной алгебре (например, векторы, матрицы, линейная независимость, базис), и темы, связанные с геометрией векторного пространства. В главе 3 мы познакомимся с векторным произведением, а далее — с нормой. При помощи этих концепций мы сможем определять углы, длины и расстояния, которые понадобятся нам для ортогональных проекций. Проекции играют ключевую роль во многих алгоритмах машинного обучения, в частности для линейной регрессии и анализа главных компонент. Эти темы будут рассмотрены в главах 9 и 10 соответственно.

## УПРАЖНЕНИЯ

**2.1.** Рассмотрим  $(\mathbb{R} \setminus \{-1\}, *)$ , где

$$a * b := ab + a + b, \quad a, b \in \mathbb{R} \setminus \{-1\}. \quad (2.134)$$

- a. Покажите, что  $(\mathbb{R} \setminus \{-1\}, *)$  принадлежит абелевой группе.
- b. Решите

$$3 * x * x = 15$$

в абелевой группе  $(\mathbb{R} \setminus \{-1\}, *)$ , где  $*$  определено по (2.134).

**2.2.** Пусть  $n$  принадлежит  $\mathbb{N} \setminus \{0\}$ . Пусть  $k, x$  принадлежат  $\mathbb{Z}$ . Определим класс сравнений  $\bar{k}$  целого числа  $k$  как множество

$$\begin{aligned} \bar{k} &= \{x \in \mathbb{Z} \mid x - k = 0 \pmod{n}\} = \\ &= \{x \in \mathbb{Z} \mid (\exists a \in \mathbb{Z}) : (x - k = n \cdot a)\}. \end{aligned}$$

Определим  $\mathbb{Z}/n\mathbb{Z}$  (иногда обозначаемое как  $\mathbb{Z}_n$ ) как множество всех классов сравнения по модулю  $n$ . Евклидово деление означает, что это множество является конечным множеством, содержащим  $n$  элементов:

$$\mathbb{Z}_n = \{\bar{0}, \bar{1}, \dots, \bar{n-1}\}.$$

Для всех  $\bar{a}, \bar{b} \in \mathbb{Z}_n$  определим

$$\bar{a} \oplus \bar{b} := \overline{a + b}.$$

- a. Покажите, что  $(\mathbb{Z}_n, \oplus)$  является группой. Является ли данная группа абелевой?
- b. Определим другую операцию  $\otimes$  для всех  $\bar{a}$  и  $\bar{b}$  в  $\mathbb{Z}_n$  как

$$\bar{a} \otimes \bar{b} := \overline{a \times b}, \quad (2.135)$$

где  $a \times b$  представляет собой обычное умножение в  $\mathbb{Z}$ .

Пусть  $n = 5$ . Нарисуйте таблицу умножения элементов  $\mathbb{Z}_5 \setminus \{\bar{0}\}$  под  $\otimes$ , то есть вычислите произведение  $\bar{a} \otimes \bar{b}$  для всех  $\bar{a}$  и  $\bar{b}$  в  $\mathbb{Z}_5 \setminus \{\bar{0}\}$ .

Таким образом, покажем, что  $\mathbb{Z}_5 \setminus \{\bar{0}\}$  замкнуто относительно  $\otimes$  и содержит нейтральный элемент для  $\otimes$ . Отобразите обратное для всех элементов  $\mathbb{Z}_5 \setminus \{\bar{0}\}$  под  $\otimes$ . Сделайте вывод, что  $\mathbb{Z}_5 \setminus \{\bar{0}\}$  – абелева группа.

- c. Покажите, что  $(\mathbb{Z}_5 \setminus \{\bar{0}\}, \otimes)$  не является группой.

- d. Напомним, что теорема Безу утверждает, что два целых числа  $a$  и  $b$  взаимно просты (то есть  $\gcd(a, b) = 1$ ) тогда и только тогда, когда существуют два целых числа  $u$  и  $v$ , такие что  $au + bv = 1$ . Покажите, что  $(\mathbb{Z}_5 \setminus \{\bar{0}\}, \otimes)$  является группой тогда и только тогда, когда  $n \in \mathbb{N} \setminus \{\bar{0}\}$  простое число.

**2.3.** Рассмотрим множество  $\mathcal{G}$  матриц  $3 \times 3$ , определенное следующим образом:

$$\mathcal{G} = \left\{ \begin{bmatrix} 1 & x & z \\ 0 & 1 & y \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3 \times 3} \mid x, y, z \in \mathbb{R} \right\}. \quad (2.136)$$

Определим  $\cdot$  как стандартное матричное умножение.

Является ли  $(\mathcal{G}, \cdot)$  группой? Если да, то абелевой ли? Обоснуйте свой ответ.

**2.4.** Если возможно, вычислите следующие матричные произведения:

$$a. \quad \begin{bmatrix} 1 & 2 \\ 4 & 5 \\ 7 & 8 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

$$b. \quad \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

$$c. \quad \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$d. \quad \begin{bmatrix} 1 & 2 & 1 & 2 \\ 4 & 1 & -1 & -4 \end{bmatrix} \begin{bmatrix} 0 & 3 \\ 1 & -1 \\ 2 & 1 \\ 5 & 2 \end{bmatrix}$$

$$e. \quad \begin{bmatrix} 0 & 3 \\ 1 & -1 \\ 2 & 1 \\ 5 & 2 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 2 \\ 4 & 1 & -1 & -4 \end{bmatrix}$$

**2.5.** Найдите множество  $S$  всех решений по  $\mathbf{x}$  следующих неоднородных линейных систем  $A\mathbf{x} = \mathbf{b}$ , где  $A$  и  $\mathbf{b}$  определены следующим образом:

$$a. \quad A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix};$$

$$b. \quad A = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -3 & 0 \\ 2 & -1 & 0 & 1 & -1 \\ -1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 3 \\ 6 \\ 5 \\ -1 \end{bmatrix}.$$

**2.6.** Используя метод исключения Гаусса, найдите все решения системы неоднородных уравнений  $A\mathbf{x} = \mathbf{b}$  с:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}.$$

**2.7.** Найдите все решения в  $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \in \mathbb{R}^3$  системы уравнений  $A\mathbf{x} = 12\mathbf{x}$ , где:

$$A = \begin{bmatrix} 6 & 4 & 3 \\ 6 & 0 & 9 \\ 0 & 8 & 0 \end{bmatrix}$$

и  $\sum_{i=1}^3 x_i = 1$ .

**2.8.** Если возможно, выполните инвертирование следующих матриц:

$$a. \quad A = \begin{bmatrix} 2 & 3 & 4 \\ 3 & 4 & 5 \\ 4 & 5 & 6 \end{bmatrix};$$

$$b. \quad A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}.$$

**2.9.** Какие из следующих множеств являются подпространствами  $\mathbb{R}^3$ ?

a.  $A = \{(\lambda, \lambda + \mu^3, \lambda - \mu^3) \mid \lambda, \mu \in \mathbb{R}\}$ .

b.  $B = \{(\lambda^2, -\lambda^2, 0) \mid \lambda \in \mathbb{R}\}$ .

c. Пусть  $\gamma$  содержится в  $\mathbb{R}$ .

$$C = \{(\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3 \mid \xi_1 - 2\xi_2 + 3\xi_3 = \gamma\}.$$

d.  $D = \{(\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3 \mid \xi_2 \in \mathbb{Z}\}$ .

**2.10.** Являются ли следующие множества векторов линейно независимыми?

a.  $\mathbf{x}_1 = \begin{bmatrix} 2 \\ -1 \\ 3 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} 3 \\ -3 \\ 8 \end{bmatrix};$

b.  $\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}.$

**2.11.** Запишите

$$\mathbf{y} = \begin{bmatrix} 1 \\ -2 \\ 5 \end{bmatrix}$$

как линейную комбинацию

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}.$$

**2.12.** Рассмотрим два подпространства  $\mathbb{R}^4$ :

$$U_1 = \text{span} \left[ \begin{bmatrix} 1 \\ 1 \\ -3 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ -1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \end{bmatrix} \right], \quad U_2 = \text{span} \left[ \begin{bmatrix} -1 \\ -2 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ -2 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} -3 \\ 6 \\ -2 \\ -1 \end{bmatrix} \right].$$

Определите базис  $U_1 \cap U_2$ .

**2.13.** Рассмотрим два подпространства  $U_1$  и  $U_2$ , где  $U_1$  — пространство решений однородной системы уравнений  $\mathbf{A}_1 \mathbf{x} = \mathbf{0}$  и  $U_2$  — пространство решений однородной системы уравнений  $\mathbf{A}_2 \mathbf{x} = \mathbf{0}$  с

$$\mathbf{A}_1 = \begin{bmatrix} 1 & 0 & 1 \\ 1 & -2 & -1 \\ 2 & 1 & 3 \\ 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 3 & -3 & 0 \\ 1 & 2 & 3 \\ 7 & -5 & 2 \\ 3 & -1 & 2 \end{bmatrix}.$$

- a. Определите размер  $U_1$ ,  $U_2$ .
- b. Определите базисы  $U_1$  и  $U_2$ .
- c. Определите базис  $U_1 \cap U_2$ .

**2.14.** Рассмотрим два подпространства  $U_1$  и  $U_2$ , где  $U_1$  охватывает столбцы  $\mathbf{A}_1$ , а  $U_2$  охватывает столбцы  $\mathbf{A}_2$  с

$$\mathbf{A}_1 = \begin{bmatrix} 1 & 0 & 1 \\ 1 & -2 & -1 \\ 2 & 1 & 3 \\ 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 3 & -3 & 0 \\ 1 & 2 & 3 \\ 7 & -5 & 2 \\ 3 & -1 & 2 \end{bmatrix}.$$

- a. Определите размер  $U_1$ ,  $U_2$ .
- b. Определите базисы  $U_1$  и  $U_2$ .
- c. Определите базис  $U_1 \cap U_2$ .

**2.15.** Пусть  $F = \{(x, y, z) \in \mathbb{R}^3 \mid x + y - z = 0\}$  и  $G = \{(a - b, a + b, a - 3b) \mid a, b \in \mathbb{R}\}$ .

- a. Покажите, что  $F$  и  $G$  — подпространства в  $\mathbb{R}^3$ .
- b. Вычислите  $F \cap G$ , не прибегая к базисному вектору.
- c. Найдите один базис для  $F$  и один для  $G$ , вычислите  $F \cap G$ , используя ранее найденные базисные векторы, и проверьте свой результат с помощью предыдущего вопроса.

**2.16.** Являются ли следующие отображения линейными?

- a. Пусть  $a, b \in \mathbb{R}$ .

$$\Phi : L^1([a, b]) \rightarrow \mathbb{R}$$

$$f \mapsto \Phi(f) = \int_a^b f(x) dx,$$

где  $L^1([a, b])$  обозначает множество интегрируемых функций на  $[a, b]$ .

*b.*

$$\begin{aligned}\Phi : C^1 &\rightarrow C^0 \\ f &\mapsto \Phi(f) = f',\end{aligned}$$

где для  $k \geq 1$ ,  $C^k$  обозначает набор  $k$  раз непрерывно дифференцируемых функций, а  $C^0$  обозначает набор непрерывных функций.

*c.*

$$\begin{aligned}\Phi : \mathbb{R} &\rightarrow \mathbb{R} \\ x &\mapsto \Phi(x) = \cos(x).\end{aligned}$$

*d.*

$$\begin{aligned}\Phi : \mathbb{R}^3 &\rightarrow \mathbb{R}^2 \\ \mathbf{x} &\mapsto \begin{bmatrix} 1 & 2 & 3 \\ 1 & 4 & 3 \end{bmatrix} \mathbf{x}.\end{aligned}$$

*e.* Пусть  $\theta$  находится в  $[0, 2\pi]$ .

$$\begin{aligned}\Phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^2 \\ \mathbf{x} &\mapsto \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \mathbf{x}.\end{aligned}$$

**2.17.** Рассмотрим линейное отображение

$$\begin{aligned}\Phi : \mathbb{R}^3 &\rightarrow \mathbb{R}^4 \\ \Phi \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} &= \begin{pmatrix} 3x_1 + 2x_2 + x_3 \\ x_1 + x_2 + x_3 \\ x_1 - 3x_2 \\ 2x_1 + 3x_2 + x_3 \end{pmatrix}.\end{aligned}$$

- Найдите матрицу преобразования  $A_\Phi$ .
- Определите  $\text{rk}(A_\Phi)$ .
- Вычислите ядро и отображение  $\Phi$ . Чем являются  $\dim(\ker(\Phi))$  и  $\dim(\text{Im}(\Phi))$ ?

**2.18.** Пусть  $E$  — векторное пространство. Пусть  $f$  и  $g$  — два автоморфизма  $E$ , такие что  $f \circ g = \text{id}_E$  (то есть  $f \circ g$  — тождественное отображение  $\text{id}_E$ ). Докажите, что  $\ker(f) = \ker(g \circ f)$ ,  $\text{Im}(g) = \text{Im}(g \circ f)$  и  $\ker(f) \cap \text{Im}(g) = \{\mathbf{0}_E\}$ .

**2.19.** Рассмотрим эндоморфизм  $\Phi : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ , матрица преобразования которого (относительно стандартного базиса в  $\mathbb{R}^3$ ):

$$A_\Phi = \begin{bmatrix} 1 & 1 & 0 \\ 1 & -1 & 0 \\ 1 & 1 & 1 \end{bmatrix}.$$

- a. Определите  $\ker(\Phi)$  и  $\text{Im}(\Phi)$ .  
 b. Определите матрицу преобразований  $\tilde{A}_\Phi$  относительно базиса

$$B = \left( \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \right),$$

то есть выполните изменение базиса к новому базису  $B$ .

**2.20.** Рассмотрим  $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}'_1, \mathbf{b}'_2$  — четыре вектора  $\mathbb{R}^2$ , выраженные в стандартном базисе  $\mathbb{R}^2$  как:

$$\mathbf{b}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad \mathbf{b}_2 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \mathbf{b}'_1 = \begin{bmatrix} 2 \\ -2 \end{bmatrix}, \quad \mathbf{b}'_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

и определим два упорядоченных базиса  $B = (\mathbf{b}_1, \mathbf{b}_2)$  и  $B' = (\mathbf{b}'_1, \mathbf{b}'_2)$  из  $\mathbb{R}^2$ .

- a. Покажите, что  $B$  и  $B'$  являются двумя базисами  $\mathbb{R}^2$  и нарисуйте эти базисные векторы.  
 b. Вычислите матрицу  $P_1$ , которая выполняет изменение базиса с  $B'$  на  $B$ .  
 c. Рассмотрим  $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3$ , три вектора из  $\mathbb{R}^3$ , определенные в стандартном базисе  $\mathbb{R}$  как:

$$\mathbf{c}_1 = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}, \quad \mathbf{c}_2 = \begin{bmatrix} 0 \\ -1 \\ 2 \end{bmatrix}, \quad \mathbf{c}_3 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}.$$

Определим  $C = (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$ .

- Покажите, что  $C$  является базисом  $\mathbb{R}^3$ , например с помощью детерминантов (раздел 4.1).
  - Пусть  $C' = (\mathbf{C}'_1, \mathbf{C}'_2, \mathbf{C}'_3)$  — стандартный базис  $\mathbb{R}^3$ . Определите матрицу  $P_2$ , которая выполняет изменение базиса с  $C$  на  $C'$ .
- d. Рассмотрим гомоморфизм  $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ , такой что

$$\begin{aligned} \Phi(\mathbf{b}_1 + \mathbf{b}_2) &= \mathbf{c}_2 + \mathbf{c}_3; \\ \Phi(\mathbf{b}_1 - \mathbf{b}_2) &= 2\mathbf{c}_1 - \mathbf{c}_2 + 3\mathbf{c}_3. \end{aligned}$$

где  $B = (\mathbf{b}_1, \mathbf{b}_2)$  и  $C = (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$  — упорядоченные базисы  $\mathbb{R}^2$  и  $\mathbb{R}^3$  соответственно.

Определите матрицу преобразования  $A_\Phi$  матрицы  $\Phi$  относительно упорядоченных базисов  $\mathbf{B}$  и  $\mathbf{C}$ .

- e. Определите  $A'$ , матрицу преобразования  $\Phi$  относительно базисов  $\mathbf{B}'$  и  $\mathbf{C}'$ .
- f. Рассмотрим вектор  $\mathbf{x} \in \mathbb{R}^2$ , координаты которого в  $\mathbf{B}'$  —  $[2, 3]^\top$ . Другими словами,  $\mathbf{x} = 2\mathbf{b}'_1 + 3\mathbf{b}'_2$ .
  1. Вычислите координаты  $\mathbf{x}$  в  $B$ .
  2. Исходя из этого, вычислите координаты  $\Phi(\mathbf{x})$ , выраженные в  $C$ .
  3. Затем запишите  $\Phi(\mathbf{x})$  исходя из  $\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3$ .
  4. Используйте представление  $\mathbf{x}$  в  $\mathbf{B}'$  и матрицу  $A'$ , чтобы получить этот результат напрямую.

# 3

## Аналитическая геометрия

[https://t.me/it\\_boooks/2](https://t.me/it_boooks/2)

В главе 2 мы изучали векторы, векторные пространства и линейные отображения в общем, но абстрактном виде. В данной главе добавим к этим понятиям геометрическую интерпретацию и интуицию. В частности, рассмотрим геометрические векторы, научимся вычислять их длины, расстояния и углы между двумя векторами. Чтобы можно было это делать, определим на векторном пространстве внутреннее произведение, которое придает ему геометрические свойства. Внутренние произведения и соответствующие им нормы и метрики заключают в себе интуитивные представления о подобии и расстояниях, которыми мы воспользуемся в главе 12 при разработке машины опорных векторов. Затем нам понадобятся понятия длины вектора и угла между векторами, чтобы разобраться с ортогональными проекциями, которые будут в центре обсуждения анализа главных компонент в главе 10, а также регрессии и оценки максимального правдоподобия в главе 9. На рис. 3.1 показано, как связаны концепции из этой главы друг с другом и с материалом из других глав книги.

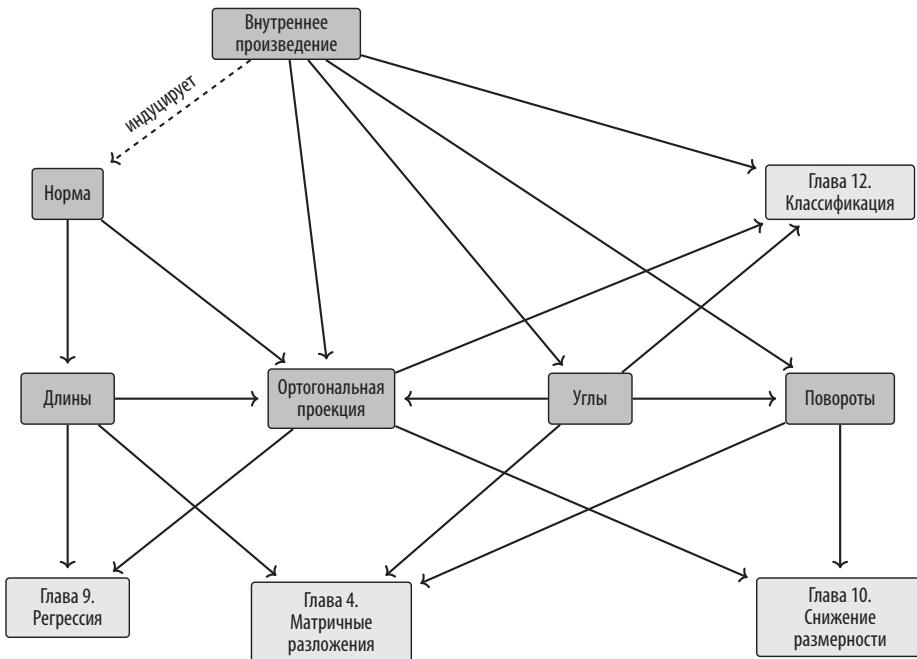
### 3.1. НОРМЫ

Рассуждая о геометрических векторах, то есть о направленных отрезках, начинаящихся в начале координат, мы интуитивно понимаем, что длина вектора — это расстояние от начала координат до «конца» этого отрезка. Далее мы будем обсуждать феномен длины вектора, пользуясь понятием нормы.

**Определение 3.1 (норма).** *Норма* векторного пространства  $V$  — это функция

$$\|\cdot\|: V \rightarrow \mathbb{R}, \quad (3.1)$$

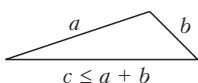
$$\boldsymbol{x} \mapsto \|\boldsymbol{x}\|, \quad (3.2)$$



**Рис. 3.1.** Ассоциативная карта концепций, вводимых в этой главе, и их связи с материалом из других частей книги

где каждому вектору  $\mathbf{x}$  присваивается значение *длины*  $\|\mathbf{x}\| \in \mathbb{R}$ , такое что для всех  $\lambda \in \mathbb{R}$  и  $\mathbf{x}, \mathbf{y} \in V$  соблюдается следующее:

- *Абсолютная однородность*:  $\|\lambda\mathbf{x}\| = |\lambda| \|\mathbf{x}\|$ .
- *Неравенство треугольника*:  $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ .
- *Положительная определенность*:  $\|\mathbf{x}\| \geq 0$  и  $\|\mathbf{x}\| = 0 \Leftrightarrow \mathbf{x} = \mathbf{0}$ .



**Рис. 3.2.** Неравенство треугольника

В геометрических терминах неравенство треугольника постулирует, что для любого треугольника сумма длин любых двух его сторон должна быть более или равна длине третьей стороны; см. рис. 3.2 в качестве иллюстрации. Определение 3.1 дано в терминах общего векторного пространства  $V$  (раздел 2.4), но в этой книге рассматривается только конечномерное векторное пространство  $\mathbb{R}^n$ . Пом-

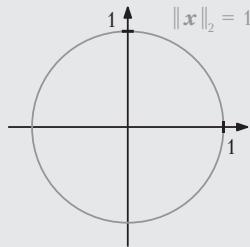
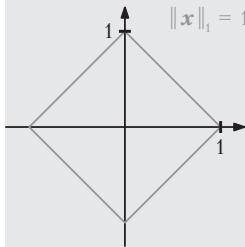
ните, что для вектора  $\mathbf{x} \in \mathbb{R}^n$  элементы вектора обозначаются с использованием нижнего индекса, то есть  $x_i$  — это  $i$ -й элемент вектора  $\mathbf{x}$ .

### Пример 3.1 (манхэттенская норма)

*Манхэттенская норма (манхэттенское расстояние)* для  $\mathbb{R}^n$  определяется для  $\mathbf{x} \in \mathbb{R}^n$  как

$$\|\mathbf{x}\|_1 := \sum_{i=1}^n |x_i|, \quad (3.3)$$

где  $|\cdot|$  — абсолютное значение. В левой панели на рис. 3.3 показаны все векторы  $\mathbf{x} \in \mathbb{R}^2$  при  $\|\mathbf{x}\|_1 = 1$ . Манхэттенская норма также называется *нормой  $\ell_1$* .



**Рис. 3.3.** Для различных норм линиями ограничены множества векторов с нормой 1. Слева: манхэттенская норма; справа: евклидово расстояние

### Пример 3.2 (евклидова норма)

Евклидова норма для  $\mathbf{x} \in \mathbb{R}^n$  определяется как

$$\|\mathbf{x}\|_2 := \sqrt{\sum_{i=1}^n x_i^2} = \sqrt{\mathbf{x}^\top \mathbf{x}} \quad (3.4)$$

и позволяет вычислить *евклидово расстояние* от  $\mathbf{x}$  до начала координат. В правой части рис. 3.3 показаны все векторы  $\mathbf{x} \in \mathbb{R}^2$  при  $\|\mathbf{x}\|_2 = 1$ . Евклидова норма также называется *нормой  $\ell_2$* .

**ПРИМЕЧАНИЕ** По умолчанию в этой книге используется евклидова норма (3.4), если не указано иное. ♦

## 3.2. ВНУТРЕННИЕ ПРОИЗВЕДЕНИЯ

При помощи внутренних произведений (внутренним называется произведение одного вектора на другой) удобно ввести интуитивно понятные геометрические

концепции, в частности длину вектора и угол или расстояние между двумя векторами. Главное назначение внутренних произведений — определять, ортогональны ли векторы друг другу.

### 3.2.1. Скалярное произведение

Возможно, вам уже известно об особом типе внутреннего произведения, *скалярном произведении*  $\mathbb{R}^n$ , которое записывается как

$$\mathbf{x}^T \mathbf{y} = \sum_{i=1}^n x_i y_i. \quad (3.5)$$

Такой частный случай внутреннего произведения мы будем называть в этой книге скалярным. Но концепция внутренних произведений более общая, и внутренние произведения обладают специфическими свойствами, о которых мы сейчас расскажем.

### 3.2.2. Общие внутренние произведения

Вспомните линейное отображение из раздела 2.7, где было показано, как можно переупорядочить отображение при сложении и умножении на скаляр. *Билинейное отображение*  $\Omega$  — это отображение с двумя аргументами, и в каждом аргументе оно линейно, то есть при рассмотрении векторного пространства  $V$  для всех  $\mathbf{x}, \mathbf{y}, \mathbf{z} \in V, \lambda, \psi \in \mathbb{R}$  верно, что

$$\Omega(\lambda \mathbf{x} + \psi \mathbf{y}, \mathbf{z}) = \lambda \Omega(\mathbf{x}, \mathbf{z}) + \psi \Omega(\mathbf{y}, \mathbf{z}); \quad (3.6)$$

$$\Omega(\mathbf{x}, \lambda \mathbf{y} + \psi \mathbf{z}) = \lambda \Omega(\mathbf{x}, \mathbf{y}) + \psi \Omega(\mathbf{x}, \mathbf{z}). \quad (3.7)$$

Здесь в (3.6) утверждается, что  $\Omega$  линейно в первом аргументе, а в (3.7) утверждается, что  $\Omega$  линейно во втором аргументе (см. также (2.87)).

**Определение 3.2.** Пусть  $V$  — векторное пространство, а  $\Omega : V \times V \rightarrow \mathbb{R}$  билинейное отображение, которое принимает два вектора и отображает их на вещественное число. Тогда

- $\Omega$  называется *симметричным*, если  $\Omega(\mathbf{x}, \mathbf{y}) = \Omega(\mathbf{y}, \mathbf{x})$  для всех  $\mathbf{x}, \mathbf{y} \in V$ , то есть порядок следования аргументов не важен.
- $\Omega$  называется *положительно определенным*, если

$$\forall \mathbf{x} \in V \setminus \{\mathbf{0}\} : \Omega(\mathbf{x}, \mathbf{x}) > 0, \Omega(\mathbf{0}, \mathbf{0}) = 0. \quad (3.8)$$

**Определение 3.3.** Пусть  $V$  — векторное пространство, а  $\Omega : V \times V \rightarrow \mathbb{R}$  билинейное отображение, которое принимает два вектора и отображает их на вещественное число. Тогда:

- Положительно определенное, симметричное билинейное отображение  $\Omega: V \times V \rightarrow \mathbb{R}$  называется *внутренним произведением*  $V$ . Как правило, пишут  $\langle \mathbf{x}, \mathbf{y} \rangle$ , а не  $\Omega(\mathbf{x}, \mathbf{y})$ .
- Пара  $(V, \langle \cdot, \cdot \rangle)$  называется *предгильбертовым пространством* или (вещественным) *векторным пространством с внутренним произведением*. Если воспользоваться скалярным произведением, определенным в (3.5), то  $(V, \langle \cdot, \cdot \rangle)$  называется *евклидовым векторным пространством*.

Такие пространства в этой книге будут называться предгильбертовыми пространствами.

**Пример 3.3 (внутреннее произведение, не являющееся скалярным)**

Рассмотрим  $V = \mathbb{R}^2$ . Если определить

$$\langle \mathbf{x}, \mathbf{y} \rangle := x_1 y_1 - (x_1 y_2 + x_2 y_1) + 2x_2 y_2, \quad (3.9)$$

то  $\langle \cdot, \cdot \rangle$  является внутренним произведением, но отличается от скалярного произведения. Докажите это в качестве упражнения.

### 3.2.3. Симметричные положительно определенные матрицы

Симметричные положительно определенные матрицы играют важную роль в машинном обучении. Они определяются при помощи внутреннего произведения. В разделе 4.3 мы вернемся к симметричным положительно определенным матрицам в контексте разложения матриц. Идея симметричных положительно полуопределеных матриц играет ключевую роль при определении ядер (раздел 12.4).

Рассмотрим  $n$ -мерное векторное пространство  $V$  с внутренним произведением  $\langle \cdot, \cdot \rangle: V \times V \rightarrow \mathbb{R}$  (см. определение 3.3) и упорядоченным базисом  $B = (\mathbf{b}_1, \dots, \mathbf{b}_n)$  от  $V$ . Как вы помните из раздела 2.6.1, любые векторы  $x, y \in V$  можно записать как линейную комбинацию базисных векторов, такую что  $\mathbf{x} = \sum_{i=1}^n \psi_i \mathbf{b}_i \in V$  и  $\mathbf{y} = \sum_{j=1}^n \lambda_j \mathbf{b}_j \in V$  для подходящих  $\psi_i, \lambda_j \in \mathbb{R}$ . Поскольку внутреннее произведение билинейно, для всех  $\mathbf{x}, \mathbf{y} \in V$  верно, что

$$\langle \mathbf{x}, \mathbf{y} \rangle = \left\langle \sum_{i=1}^n \psi_i \mathbf{b}_i, \sum_{j=1}^n \lambda_j \mathbf{b}_j \right\rangle = \sum_{i=1}^n \sum_{j=1}^n \psi_i \langle \mathbf{b}_i, \mathbf{b}_j \rangle \lambda_j = \hat{\mathbf{x}}^\top \mathbf{A} \hat{\mathbf{y}}, \quad (3.10)$$

где  $A_{ij} := \langle \mathbf{b}_i, \mathbf{b}_j \rangle$ , а  $\hat{\mathbf{x}}, \hat{\mathbf{y}}$  — это координаты  $x$  и  $y$  относительно базиса  $B$ . Это подразумевает, что внутреннее произведение  $\langle \cdot, \cdot \rangle$  уникально определяется через  $\mathbf{A}$ .

Симметрия внутреннего произведения также означает, что  $\mathbf{A}$  симметрично. Кроме того, положительная определенность внутреннего произведения подразумевает, что

$$\forall \mathbf{x} \in V \setminus \{\mathbf{0}\} : \mathbf{x}^T \mathbf{A} \mathbf{x} > 0. \quad (3.11)$$

**Определение 3.4 (симметричная, положительно определенная матрица).** Симметричная матрица  $\mathbf{A} \in \mathbb{R}^{n \times n}$ , удовлетворяющая (3.11), называется *симметричной, положительно определенной* или просто *положительно определенной*. Если для (3.11) верно только  $\geq$ , то  $\mathbf{A}$  называется *симметричной, положительно полуопределенной*.

#### Пример 3.4 (симметричные, положительно определенные матрицы)

Рассмотрим матрицы

$$\mathbf{A}_1 = \begin{bmatrix} 9 & 6 \\ 6 & 5 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 9 & 6 \\ 6 & 3 \end{bmatrix}. \quad (3.12)$$

$\mathbf{A}_1$  является положительно определенной, поскольку она симметрична и

$$\mathbf{x}^T \mathbf{A}_1 \mathbf{x} = [x_1 \ x_2] \begin{bmatrix} 9 & 6 \\ 6 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \quad (3.13a)$$

$$= 9x_1^2 + 12x_1x_2 + 5x_2^2 = (3x_1 + 2x_2)^2 + x_2^2 > 0 \quad (3.13b)$$

для всех  $\mathbf{x} \in V \setminus \{\mathbf{0}\}$ . Напротив,  $\mathbf{A}_2$  симметрична, но не положительно определена, так как  $\mathbf{x}^T \mathbf{A}_2 \mathbf{x} = 9x_1^2 + 12x_1x_2 + 3x_2^2 = (3x_1 + 2x_2)^2 - x_2^2$  может быть меньше 0, напр. для  $\mathbf{x} = [2, -3]^T$ .

Если  $\mathbf{A} \in \mathbb{R}^{n \times n}$  симметрична, положительно определена, то

$$\langle \mathbf{x}, \mathbf{y} \rangle = \hat{\mathbf{x}}^T \mathbf{A} \hat{\mathbf{y}} \quad (3.14)$$

определяет внутреннее произведение относительно упорядоченного базиса  $B$ , где  $\hat{\mathbf{x}}$  и  $\hat{\mathbf{y}}$  — координатные представления  $\mathbf{x}, \mathbf{y} \in V$  относительно  $B$ .

**Теорема 3.5.** Для вещественнозначного, конечномерного векторного пространства  $V$  и упорядоченного базиса  $B$  от  $V$  верно, что  $\langle \cdot, \cdot \rangle : V \times V \rightarrow \mathbb{R}$  является внутренним произведением тогда и только тогда, если существует симметричная, положительно определенная матрица  $\mathbf{A} \in \mathbb{R}^{n \times n}$  с

$$\langle \mathbf{x}, \mathbf{y} \rangle = \hat{\mathbf{x}}^T \mathbf{A} \hat{\mathbf{y}}. \quad (3.15)$$

Следующие свойства соблюдаются, если  $A \in \mathbb{R}^{n \times n}$  является симметричной и положительно определенной:

- Нулевое пространство (ядро)  $A$  состоит лишь из  $\mathbf{0}$ , поскольку  $\mathbf{x}^T A \mathbf{x} > 0$  для всех  $\mathbf{x} \neq \mathbf{0}$ . Это подразумевает, что  $A \mathbf{x} \neq \mathbf{0}$  и  $\mathbf{x} \neq \mathbf{0}$ .
- Диагональные элементы  $a_{ii}$  от  $A$  положительны, поскольку  $a_{ii} = \mathbf{e}_i^T A \mathbf{e}_i > 0$ , где  $\mathbf{e}_i$  это  $i$ -й вектор стандартного базиса в  $\mathbb{R}^n$ .

### 3.3. ДЛИНЫ И РАССТОЯНИЯ

В разделе 3.1 мы уже обсуждали нормы, которыми можно пользоваться для вычисления длины вектора. Внутренние произведения и нормы тесно связаны в том смысле, что любое внутреннее произведение вводит норму

$$\| \mathbf{x} \| := \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} \quad (3.16)$$

естественным образом, так что длины векторов можно вычислять при помощи внутреннего произведения. Но не всякая норма вводится внутренним произведением. Манхэттенская норма (3.3) — такая, у которой нет соответствующего внутреннего произведения. В дальнейшем мы сосредоточимся на нормах, вводимых внутренними произведениями, и познакомимся с геометрическими концепциями, в частности с длинами, расстояниями и углами.

**ПРИМЕЧАНИЕ** Для векторного пространства внутреннего произведения  $(V, \langle \cdot, \cdot \rangle)$  индуцированная норма  $\| \cdot \|$  удовлетворяет неравенству Коши — Шварца<sup>1</sup>

$$|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \| \mathbf{x} \| \| \mathbf{y} \| . \quad (3.17)$$



#### Пример 3.5 (длины векторов с использованием внутренних произведений)

В геометрии нас часто интересуют длины векторов. Теперь для их вычисления мы можем воспользоваться внутренним произведением (3.16). Предположим,  $\mathbf{x} = [1, 1]^T \in \mathbb{R}^2$ . Если воспользоваться скалярным произведением как внутренним, то, исходя из (3.16), получим

$$\| \mathbf{x} \| = \sqrt{\mathbf{x}^T \mathbf{x}} = \sqrt{1^2 + 1^2} = \sqrt{2} \quad (3.18)$$

<sup>1</sup> В отечественной литературе часто называют неравенством Коши — Буняковского — Шварца. — Примеч. науч. ред.

в качестве длины  $\mathbf{x}$ . Теперь выберем иное внутреннее произведение:

$$\langle \mathbf{x}, \mathbf{y} \rangle := \mathbf{x}^T \begin{bmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{bmatrix} \mathbf{y} = x_1 y_1 - \frac{1}{2}(x_1 y_2 + x_2 y_1) + x_2 y_2. \quad (3.19)$$

Если вычислить норму вектора, то это внутреннее произведение будет возвращать меньшие значения, чем скалярное произведение, в случае если у  $x_1$  и  $x_2$  одинаковый знак (а  $x_1 x_2 > 0$ ); в противном случае оно будет возвращать большие значения, чем скалярное произведение. С таким внутренним произведением имеем:

$$\langle \mathbf{x}, \mathbf{x} \rangle = x_1^2 - x_1 x_2 + x_2^2 = 1 - 1 + 1 = 1 \Rightarrow \|\mathbf{x}\| = \sqrt{1} = 1, \quad (3.20)$$

такое что  $\mathbf{x}$  «короче» с этим внутренним произведением, чем со скалярным произведением.

**Определение 3.6 (расстояние и метрика).** Рассмотрим пространство внутреннего произведения  $(V, \langle \cdot, \cdot \rangle)$ . Тогда

$$d(\mathbf{x}, \mathbf{y}) := \|\mathbf{x} - \mathbf{y}\| = \sqrt{\langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{y} \rangle} \quad (3.21)$$

называется *расстоянием* между  $\mathbf{x}$  и  $\mathbf{y}$  для  $\mathbf{x}, \mathbf{y} \in V$ . Если использовать скалярное произведение как внутреннее, то это расстояние называется *евклидовым расстоянием*. Отображение

$$d : V \times V \rightarrow \mathbb{R} \quad (3.22)$$

$$(\mathbf{x}, \mathbf{y}) \mapsto d(\mathbf{x}, \mathbf{y}) \quad (3.23)$$

называется *метрикой*.

**ПРИМЕЧАНИЕ** Подобно длине вектора, расстояние между векторами не требует внутреннего произведения; достаточно нормы. Если у вас есть норма, индуцированная внутренним произведением, то расстояние может отличаться в зависимости от выбранного внутреннего произведения. ♦

Метрика  $d$  удовлетворяет следующим условиям:

1.  $d$  является положительно определенной, то есть  $d(\mathbf{x}, \mathbf{y}) \geq 0$  для всех  $\mathbf{x}, \mathbf{y} \in V$  и  $d(\mathbf{x}, \mathbf{y}) = 0 \Leftrightarrow \mathbf{x} = \mathbf{y}$ .
2.  $d$  симметрично, то есть  $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$  для всех  $\mathbf{x}, \mathbf{y} \in V$ .
3. Неравенство треугольников:  $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$  для всех  $\mathbf{x}, \mathbf{y}, \mathbf{z} \in V$ .

**ПРИМЕЧАНИЕ** На первый взгляд, списки свойств внутренних произведений и метрик выглядят очень похоже. Однако если сравнить определение 3.3 с определением 3.6, то можно заменить, что  $\langle \mathbf{x}, \mathbf{y} \rangle$  и  $d(\mathbf{x}, \mathbf{y})$  действуют в противоположных направлениях. Очень схожие  $\mathbf{x}$  и  $\mathbf{y}$  будут давать большое значение для внутреннего произведения и малое значение для метрики. ♦

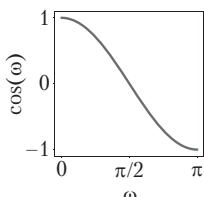
### 3.4. УГЛЫ И ОРТОГОНАЛЬНОСТЬ

Внутренние произведения позволяют не только определять длины векторов и расстояние между двумя векторами, но и схватывают геометрию векторного пространства, определяя угол  $\omega$  между двумя векторами. Мы воспользуемся неравенством Коши – Шварца (3.17) для определения углов  $\omega$  в пространствах внутренних произведений между двумя векторами  $\mathbf{x}, \mathbf{y}$ , и это определение совпадает с интуитивным пониманием  $\mathbb{R}^2$  и  $\mathbb{R}^3$ . Предположим, что  $\mathbf{x} \neq 0, \mathbf{y} \neq 0$ . Тогда

$$-1 \leq \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|} \leq 1. \quad (3.24)$$

Следовательно, существует уникальный  $\omega \in [0, \pi]$ , показанный на рис. 3.4, с

$$\cos \omega = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|}. \quad (3.25)$$



**Рис. 3.4.** В области  $[0, \pi]$   $f(\omega) = \cos(\omega)$  возвращает число в интервале  $[-1, 1]$

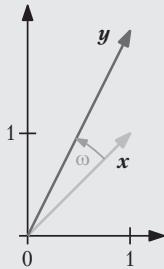
Число  $\omega$  — это угол между векторами  $\mathbf{x}$  и  $\mathbf{y}$ . Интуитивно понятно, что угол между двумя векторами позволяет судить, насколько схоже они ориентированы. Например, воспользовавшись скалярным произведением, найдем, что угол между  $\mathbf{x}$  и  $\mathbf{y} = 4\mathbf{x}$  (то есть  $\mathbf{y}$  — это  $\mathbf{x}$  с некоторым множителем) равен 0.

#### Пример 3.6 (угол между векторами)

Вычислим угол между  $\mathbf{x} = [1, 1]^T \in \mathbb{R}^2$  и  $\mathbf{y} = [1, 2]^T \in \mathbb{R}^2$ ; см. рис. 3.5, где мы используем внутреннее произведение как скалярное. Затем получаем

$$\cos \omega = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\sqrt{\langle \mathbf{x}, \mathbf{x} \rangle \langle \mathbf{y}, \mathbf{y} \rangle}} = \frac{\mathbf{x}^T \mathbf{y}}{\sqrt{\mathbf{x}^T \mathbf{x} \mathbf{y}^T \mathbf{y}}} = \frac{3}{\sqrt{10}}, \quad (3.26)$$

и угол между двумя векторами равен  $\arccos\left(\frac{3}{\sqrt{10}} \approx 0,32\right)$  радиана, что соответствует примерно  $18^\circ$ .



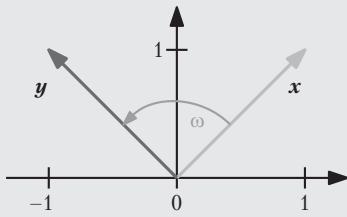
**Рис. 3.5.** Угол  $\omega$  между двумя векторами  $x$  и  $y$  вычисляется при помощи внутреннего произведения

**Определение 3.7 (ортогональность).** Два вектора  $x$  и  $y$  *ортогональны* тогда и только тогда, когда  $\langle x, y \rangle = 0$ , и мы пишем  $x \perp y$ . Если к тому же  $\|x\| = 1 = \|y\|$ , то есть векторы являются единичными, то  $x$  и  $y$  *ортонормированы*.

Из этого определения следует, что **0**-вектор ортогонален любому вектору в векторном пространстве.

**ПРИМЕЧАНИЕ** Ортогональность — это обобщение концепции перпендикулярности, применимой к билинейным формам, которые не обязательно дают скалярное произведение. В нашем контексте с геометрической точки зрения можно считать, что ортогональные векторы расположены под прямым углом относительно конкретного внутреннего произведения. ♦

#### Пример 3.7 (ортогональные векторы)



**Рис. 3.6.** Угол  $\omega$  между двумя векторами  $x$  и  $y$  может меняться в зависимости от внутреннего произведения

Рассмотрим два вектора  $x = [1, 1]^T$ ,  $y = [-1, 1]^T \in \mathbb{R}^2$ ; см. рис. 3.6. Нас интересует угол  $\omega$  между ними, который мы найдем, исходя из двух внутренних произведений. Используя скалярное произведение как внутрен-

нее, получим угол  $\omega$  между  $\mathbf{x}$  и  $\mathbf{y}$ , равный  $90^\circ$ , такой что  $\mathbf{x} \perp \mathbf{y}$ . Однако если выбрать внутреннее произведение

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{y}, \quad (3.27)$$

то получим, что угол  $\omega$  между  $\mathbf{x}$  и  $\mathbf{y}$  вычисляется как

$$\cos \omega = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|} = -\frac{1}{3} \Rightarrow \omega \approx 1,91 \text{ рад} \approx 109,5^\circ \quad (3.28)$$

и  $\mathbf{x}$  и  $\mathbf{y}$  не ортогональны. Следовательно, векторы, ортогональные относительно одного внутреннего произведения, не обязательно должны быть ортогональны относительно другого внутреннего произведения.

**Определение 3.8 (ортогональная матрица).** Квадратная матрица  $\mathbf{A} \in \mathbb{R}^{n \times n}$  является *ортогональной матрицей* тогда и только тогда, когда ее столбцы ортонормированы, так что

$$\mathbf{A}\mathbf{A}^T = \mathbf{I} = \mathbf{A}^T\mathbf{A}, \quad (3.29)$$

что подразумевает

$$\mathbf{A}^{-1} = \mathbf{A}^T, \quad (3.30)$$

то есть обратная матрица получается простым транспонированием матрицы.

Преобразования ортогональных матриц<sup>1</sup> представляют особенный случай, поскольку длина вектора  $\mathbf{x}$  не меняется, если преобразовывать его с применением ортогональной матрицы  $\mathbf{A}$ . Для скалярного произведения получаем

$$\|\mathbf{Ax}\|^2 = (\mathbf{Ax})^T (\mathbf{Ax}) = \mathbf{x}^T \mathbf{A}^T \mathbf{Ax} = \mathbf{x}^T \mathbf{Ix} = \mathbf{x}^T \mathbf{x} = \|\mathbf{x}\|^2. \quad (3.31)$$

Более того, угол между любыми двумя векторами  $\mathbf{x}$ ,  $\mathbf{y}$ , измеренный по их внутреннему произведению, также не изменится, если преобразовать оба эти вектора с применением ортогональной матрицы  $\mathbf{A}$ . Взяв скалярное произведение в качестве внутреннего, получим, что угол образов  $\mathbf{Ax}$  и  $\mathbf{Ay}$  дается как

---

<sup>1</sup> Принято называть такие матрицы ортогональными, но точнее было бы называть их ортонормированными. При преобразованиях ортогональных матриц сохраняются расстояния и углы.

$$\cos \omega \frac{(\mathbf{A}\mathbf{x})^T (\mathbf{A}\mathbf{y})}{\|\mathbf{A}\mathbf{x}\| \|\mathbf{A}\mathbf{y}\|} = \frac{\mathbf{x}^T \mathbf{A}^T \mathbf{A}\mathbf{y}}{\sqrt{\mathbf{x}^T \mathbf{A}^T \mathbf{A} \mathbf{x} \mathbf{y}^T \mathbf{A}^T \mathbf{A} \mathbf{y}}} = \frac{\mathbf{x}^T \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}, \quad (3.32)$$

то есть в точности равен углу между  $\mathbf{x}$  и  $\mathbf{y}$ . Это означает, что у ортогональных матриц  $\mathbf{A}$ , где  $\mathbf{A}^T = \mathbf{A}^{-1}$ , сохраняются как углы, так и расстояния. Оказывается, что ортогональные матрицы определяют преобразования, являющиеся поворотами (с возможностью переворота). В разделе 3.9 мы подробнее поговорим о поворотах.

### 3.5. ОРТОНОРМИРОВАННЫЙ БАЗИС

В разделе 2.6.1 мы охарактеризовали свойства базисных векторов и нашли, что в  $n$ -мерном векторном пространстве нам требуется  $n$  базисных векторов, то есть  $n$  векторов, которые линейно независимы. В разделах 3.3 и 3.4 мы пользовались внутренними произведениями для вычисления длины векторов и угла между векторами. Далее мы обсудим особый случай, в котором базисные векторы ортогональны друг другу и где длина каждого базисного вектора равна 1. Такой базис мы будем называть ортонормированным.

Дадим ему более строгое определение.

**Определение 3.9 (ортонормированный базис).** Рассмотрим  $n$ -мерное векторное пространство  $V$  и базис  $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$  от  $V$ . Если

$$\langle \mathbf{b}_i, \mathbf{b}_j \rangle = 0 \text{ для } i \neq j \quad (3.33)$$

$$\langle \mathbf{b}_i, \mathbf{b}_i \rangle = 1 \quad (3.34)$$

для всех  $i, j = 1, \dots, n$ , то базис называется *ортонормированным базисом (ОНБ)* (orthonormal basis, ONB). Если соблюдается лишь (3.3), то базис называется *ортогональным*. Обратите внимание: (3.34) предполагает, что длина/норма каждого базисного вектора равна 1.

Как вы помните из раздела 2.6.1, можно использовать гауссово исключение, чтобы найти базис векторного пространства, ограниченного множеством векторов. Допустим, у нас есть множество  $\{\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_n\}$  неортогональных ненормированных базисных векторов. Мы сцепляем их в матрицу  $\tilde{\mathbf{B}} = [\tilde{\mathbf{b}}_1 \ \dots \ \tilde{\mathbf{b}}_n]$  и применяем гауссово исключение к расширенной матрице (раздел 2.3.2)  $[\tilde{\mathbf{B}} \tilde{\mathbf{B}}^T \mid \tilde{\mathbf{B}}]$  для получения ортонормированного базиса. Такой конструктивный подход, позволяющий итеративно выстроить ортонормированный базис  $\{\mathbf{b}_1 \dots \mathbf{b}_n\}$ , называется «процесс Грама – Шмидта» (Strang 2003).

**Пример 3.8 (ортонормированный базис)**

Канонический/стандартный базис евклидова векторного пространства  $\mathbb{R}^n$  является ортонормированным базисом, внутреннее произведение которого — это скалярное произведение векторов.

В  $\mathbb{R}^2$  векторы

$$\mathbf{b}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \mathbf{b}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad (3.35)$$

образуют ортонормированный базис, поскольку  $\mathbf{b}_1^\top \mathbf{b}_2 = 0$  и  $\|\mathbf{b}_1\| = 1 = \|\mathbf{b}_2\|$ .

Мы воспользуемся концепцией ортонормированного базиса в главах 10 и 12, где обсудим машины опорных векторов и анализ главных компонент.

### 3.6. ОРТОГОНАЛЬНОЕ ДОПОЛНЕНИЕ

Определив ортогональность, давайте теперь рассмотрим векторные пространства, ортогональные друг другу. Эта концепция сыграет важную роль в главе 10, где мы обсудим линейное снижение размерности с геометрической точки зрения.

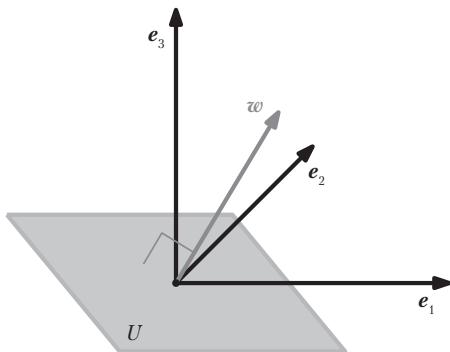
Рассмотрим  $D$ -мерное векторное пространство  $V$  и  $M$ -мерное подпространство, такое что  $U \subseteq V$ . Тогда его *ортогональное дополнение*  $U^\perp$  будет  $(D - M)$ -мерным подпространством от  $V$  и будет содержать все векторы из  $V$ , которые ортогональны каждому из векторов  $U$ . Кроме того,  $U \cap U^\perp = \{\mathbf{0}\}$ , поэтому каждый из векторов  $\mathbf{x} \in V$  поддается уникальному разложению в

$$\mathbf{x} = \sum_{m=1}^M \lambda_m \mathbf{b}_m + \sum_{j=1}^{D-M} \psi_j \mathbf{b}_j^\perp, \quad \lambda_m, \psi_j \in \mathbb{R}, \quad (3.36)$$

где  $(\mathbf{b}_1 \dots \mathbf{b}_M)$  это базис  $U$ , а  $(\mathbf{b}_1^\perp, \dots, \mathbf{b}_{D-M}^\perp)$  — это базис  $U^\perp$ .

Следовательно, ортогональное дополнение также может применяться для описания плоскости  $U$  (двумерного подпространства) в трехмерном векторном пространстве. Точнее говоря, вектор  $\mathbf{w}$  с  $\|\mathbf{w}\| = 1$ , ортогональный плоскости  $U$ , является базисным вектором  $U^\perp$ . Такая структура проиллюстрирована на рис. 3.7. Все векторы, ортогональные  $\mathbf{w}$ , должны (по определению) лежать в плоскости  $U$ . Вектор  $\mathbf{w}$  называется *нормальным вектором*  $U$ .

В принципе, ортогональные дополнения могут использоваться для описания гиперплоскостей в  $n$ -мерном векторном и аффинном пространствах.



**Рис. 3.7.** Плоскость  $U$  в трехмерном векторном пространстве можно описать ее нормальным вектором, который охватывает ее ортогональное дополнение  $U^\perp$

### 3.7. ВНУТРЕННЕЕ ПРОИЗВЕДЕНИЕ ФУНКЦИЙ

До сих пор мы рассматривали свойства внутренних произведений при вычислении длин, углов и расстояний. Мы уделяли особое внимание внутренним произведениям конечномерных векторов. Далее мы рассмотрим пример внутренних произведений векторов иного типа: поговорим о внутренних произведениях функций.

Внутренние произведения, которые мы обсуждали до сих пор, определялись для векторов с конечным числом включений. Можно считать вектор  $\mathbf{x} \in \mathbb{R}^n$  функцией с  $n$  значений функции. Концепцию внутреннего произведения можно обобщить до векторов с бесконечным числом включений (счетно бесконечных), а также до функций с непрерывными значениями (бесчетно бесконечных). Затем, просуммировав отдельные компоненты векторов — см., в частности, пример 3.5, — получаем интеграл.

Внутреннее произведение двух функций  $u: \mathbb{R} \rightarrow \mathbb{R}$  и  $v: \mathbb{R} \rightarrow \mathbb{R}$  можно определить как определенный интеграл

$$\langle u, v \rangle := \int_a^b u(x)v(x)dx \quad (3.37)$$

для нижнего и верхнего предела  $a, b < \infty$  соответственно. Как и в случае с обычным внутренним произведением, можно определять нормы и ортогональность, ориентируясь на внутреннее произведение. Если (3.37) результирует в 0, то функции  $u$  и  $v$  ортогональны. Чтобы добиться математической точности предыдущего внутреннего произведения, нужно уделить внимание мерам и определению интегралов, что приведет нас к определению гильбертова пространства. В дальнейшем, в отличие от внутренних произведений конечномерных векторов, внутренние произведения функций могут расходиться (иметь бесконечное значение). Все это требует вдаваться в тонкие детали вещественного и функционального анализа, но мы не рассматриваем данных тем в этой книге.

**Пример 3.9 (внутреннее произведение функций)**

Если взять  $u = \sin(x)$  и  $v = \cos(x)$ , то получится подынтегральная функция  $f(x) = u(x)v(x)$  от (3.37), показанная на рис. 3.8. Мы видим, что эта функция нечетная, то есть  $f(-x) = -f(x)$ . Следовательно, интеграл с пределами  $a = -\pi$ ,  $b = \pi$  от этого произведения равен 0. Следовательно, функции  $\sin$  и  $\cos$  ортогональны.

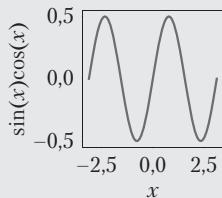


Рис. 3.8.  $f(x) = \sin(x)\cos(x)$

**ПРИМЕЧАНИЕ** Также верно, что набор функций

$$\{1, \cos(x), \cos(2x), \cos(3x), \dots\} \quad (3.38)$$

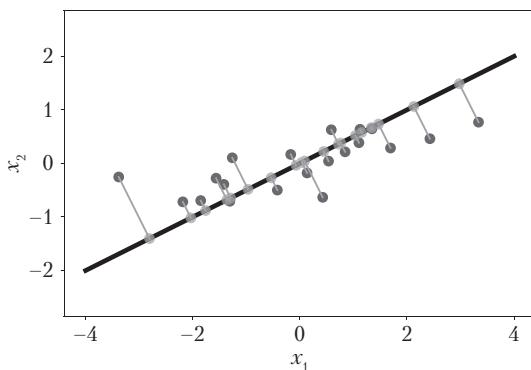
ортогонален, если интегрировать от  $-\pi$  до  $\pi$ , то есть в любой паре функций обе функции ортогональны друг другу. Набор функций в (3.38) охватывает большое подпространство функций, являющихся четными и периодическими на отрезке  $[-\pi, \pi]$ , и проектирование функций на это подпространство — фундаментальная идея, на которой основаны ряды Фурье. ♦

В разделе 6.4.6 мы рассмотрим второй тип необычных внутренних произведений: внутреннее произведение случайных переменных.

### 3.8. ОРТОГОНАЛЬНЫЕ ПРОЕКЦИИ

Проекции — это важный класс линейных преобразований (наряду с поворотами и отражениями). Они играют важную роль в графике, теории кодирования, статистике и машинном обучении. В машинном обучении часто приходится иметь дело с данными, обладающими высокой размерностью. Такие данные зачастую бывает сложно анализировать или визуализировать. Однако многомерным данным весьма часто присущее следующее свойство: основная часть информации содержится всего в нескольких измерениях, а большинство других измерений несущественны при описании ключевых свойств данных. При сжатии и визуализации данных, которые обладают большим количеством измерений, часть информации теряется. Чтобы минимизировать такие потери, приходящи-

ется на сжатие, в идеале нужно найти наиболее информативные измерения данных. Как говорилось в главе 1, данные можно представлять в векторном виде, и в этой главе мы обсудим некоторые инструменты, принципиально важные для сжатия данных. Точнее, мы рассмотрим, как можно проецировать исходные данные с высокой размерностью на признаковое<sup>1</sup> пространство с меньшей размерностью, чтобы извлечь больше информации обо всем множестве данных и извлечь из него важные паттерны. Например, в алгоритмах машинного обучения, таких как анализ главных компонент (PCA), рассмотренный в Pearson (1901) и Hotelling (1933), а также при работе с глубокими нейронными сетями (например, с глубокими автоэнкодерами) активно используется идея снижения размерности. Далее в этой главе мы сосредоточимся на ортогональных проекциях, которыми воспользуемся в главе 10 для линейного снижения размерности и в главе 12 — для классификации. Даже линейная регрессия, обсуждаемая в главе 9, поддается интерпретации при помощи ортогональных проекций. Для заданного пространства с малой размерностью ортогональные проекции данных, обладающих высокой размерностью, сохраняют максимум информации, насколько это возможно, а также минимизируют разницу/погрешность между исходными данными и соответствующей им проекцией. Такая ортогональная проекция проиллюстрирована на рис. 3.9. Прежде чем мы подробно обсудим, как получать такие проекции, давайте определим, что же представляет собой проекция.



**Рис. 3.9.** Ортогональная проекция (светлые точки) двумерного множества данных (темные точки) на одномерное подпространство (прямая линия)

**Определение 3.10 (проекция).** Пусть  $V$  — векторное пространство, а  $U \subseteq V$  — подпространство  $V$ . Линейное отображение  $\pi: V \rightarrow U$  называется *проекцией*, если  $\pi^2 = \pi \circ \pi = \pi$ .

<sup>1</sup> Термином «признак» часто обозначается представление данных.

Поскольку линейные отображения можно выражать при помощи матриц преобразований (см. раздел 2.7), предыдущее определение в равной степени применимо и к особому классу матриц преобразований — *проекционным матрицам*  $P_\pi$ . Такая матрица обладает свойством  $P_\pi^2 = P_\pi$ .

В дальнейшем мы будем выводить ортогональные проекции векторов в пространстве внутренних произведений ( $\mathbb{R}^n <\cdot, \cdot>$ ) на подпространства. Начнем с одномерных подпространств, которые также называются *линиями (прямыми)*. Если не указано иное, мы считаем  $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T \mathbf{y}$  внутренним произведением.

### 3.8.1. Проекция на одномерные подпространства (прямые)

Допустим, нам дана прямая (одномерное подпространство), проходящая через начало координат с базисным вектором  $\mathbf{b} \in \mathbb{R}^n$ . Прямая — это одномерное подпространство  $U \subseteq \mathbb{R}^n$ , чьей оболочкой является  $\mathbf{b}$ . Если спроектировать  $\mathbf{x} \in \mathbb{R}^n$  на  $U$ , то нужно найти вектор  $\pi_U(\mathbf{x}) \in U$ , ближайшему к  $\mathbf{x}$ . Воспользовавшись геометрическими аргументами, охарактеризуем некоторые свойства проекции  $\pi_U(\mathbf{x})$  (рис. 3.10(a) возьмем в качестве иллюстрации):

- Проекция  $\pi_U(\mathbf{x})$  является ближайшей к  $\mathbf{x}$ , где «ближайшая» подразумевает, что расстояние  $\|\mathbf{x} - \pi_U(\mathbf{x})\|$  является минимальным. Следовательно, отрезок  $\pi_U(\mathbf{x}) - \mathbf{x}$  от  $\pi_U(\mathbf{x})$  до  $\mathbf{x}$  ортогонален  $U$  и, следовательно, базисному вектору  $\mathbf{b}$  от  $U$ . Условие ортогональности дает  $\langle \pi_U(\mathbf{x}) - \mathbf{x}, \mathbf{b} \rangle = 0$ , поскольку углы между векторами определяются по внутреннему произведению.
- Проекция  $\pi_U(\mathbf{x})$  от  $\mathbf{x}$  на  $U$  должна быть элементом  $U$  и, следовательно, кратной базисному вектору  $\mathbf{b}$ , который охватывает  $U$ . Следовательно,  $\pi_U(\mathbf{x}) = \lambda \mathbf{b}$  для некоторого  $\lambda \in \mathbb{R}$ <sup>1</sup>.

На следующих трех этапах определяем координату  $\lambda$ , проекцию  $\pi_U(\mathbf{x}) \in U$ , и матрицу проекции  $P_\pi$ , отображающую любое на  $\mathbf{x} \in \mathbb{R}^n$  на  $U$ :

- Находим координату  $\lambda$ . Условие ортогональности дает

$$\langle \mathbf{x} - \pi_U(\mathbf{x}), \mathbf{b} \rangle = 0 \Leftrightarrow \langle \mathbf{x} - \lambda \mathbf{b}, \mathbf{b} \rangle = 0. \quad (3.39)$$

Теперь можно воспользоваться билинейностью внутреннего произведения и получить

$$\langle \mathbf{x}, \mathbf{b} \rangle - \lambda \langle \mathbf{b}, \mathbf{b} \rangle = 0 \Leftrightarrow \lambda = \frac{\langle \mathbf{x}, \mathbf{b} \rangle}{\langle \mathbf{b}, \mathbf{b} \rangle} = \frac{\langle \mathbf{b}, \mathbf{x} \rangle}{\|\mathbf{b}\|^2}. \quad (3.40)$$

---

<sup>1</sup> В таком случае  $\lambda$  является координатой  $\pi_U(\mathbf{x})$  относительно  $\mathbf{b}$ .

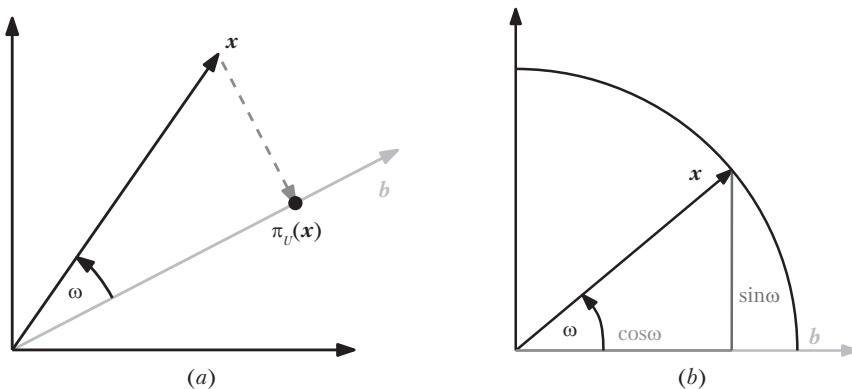


Рис. 3.10. Примеры проекций на одномерные подпространства

В последнем шаге мы воспользовались тем фактом, что внутренние произведения симметричны<sup>1</sup>. Если мы выберем  $\langle \cdot, \cdot \rangle$  в качестве скалярного произведения, то получим

$$\lambda = \frac{\mathbf{b}^T \mathbf{x}}{\mathbf{b}^T \mathbf{b}} = \frac{\mathbf{b}^T \mathbf{x}}{\|\mathbf{b}\|^2}. \quad (3.41)$$

Если  $\|\mathbf{b}\| = 1$ , то координата  $\lambda$  проекции будет описываться выражением  $\mathbf{b}^T \mathbf{x}$ .

- Находим точку проекции  $\pi_U(\mathbf{x}) \in U$ . Поскольку  $\pi_U(\mathbf{x}) = \lambda \mathbf{b}$ , мы, воспользовавшись (3.40), сразу же получаем, что

$$\pi_U(\mathbf{x}) = \lambda \mathbf{b} = \frac{\langle \mathbf{x}, \mathbf{b} \rangle}{\|\mathbf{b}\|^2} \mathbf{b} = \frac{\mathbf{b}^T \mathbf{x}}{\|\mathbf{b}\|^2} \mathbf{b}, \quad (3.42)$$

где последнее равенство сохраняется только для скалярного произведения. Мы также можем вычислить длину  $\pi_U(\mathbf{x})$  при помощи определения 3.1 как

$$\|\pi_U(\mathbf{x})\| = \|\lambda \mathbf{b}\| = |\lambda| \|\mathbf{b}\|. \quad (3.43)$$

Следовательно, наша проекция имеет длину  $|\lambda|$ , умноженного на длину  $\mathbf{b}$ . Это также интуитивно подсказывает, что  $\lambda$  является координатой  $\pi_U(\mathbf{x})$  относительно базисного вектора  $\mathbf{b}$ , охватывающего одномерное подпространство  $U$ .

<sup>1</sup> С общим внутренним произведением получаем  $\lambda = \langle \mathbf{x}, \mathbf{b} \rangle$ , если  $\|\mathbf{b}\| = 1$ .

Если мы воспользуемся скалярным произведением как внутренним, то получим

$$\|\pi_U(\mathbf{x})\| = \frac{|\mathbf{b}^T \mathbf{x}|}{\|\mathbf{b}\|^2} \|\mathbf{b}\| \stackrel{(3.25)}{=} |\cos \omega| \|\mathbf{x}\| \|\mathbf{b}\| \frac{\|\mathbf{b}\|}{\|\mathbf{b}\|^2} = |\cos \omega| \|\mathbf{x}\|. \quad (3.44)$$

Здесь  $\omega$  — это угол между  $\mathbf{x}$  и  $\mathbf{b}$ . Это уравнение должно быть знакомо вам из тригонометрии: если  $\|\mathbf{x}\| = 1$ , то  $\mathbf{x}$  располагается на единичной окружности. Отсюда следует, что проекция на горизонтальную ось<sup>1</sup>, охватываемую  $\mathbf{b}$ , составляет ровно  $\cos \omega$ , а длина соответствующего вектора  $\pi_U(\mathbf{x}) = |\cos \omega|$ . Это проиллюстрировано на рис. 3.10(b).

3. Нахождение проекционной матрицы  $\mathbf{P}_\pi$ . Мы знаем, что проекция является линейным отображением (см. определение 3.10). Следовательно, существует проекционная матрица  $\mathbf{P}_\pi$ , такая что  $\pi_U(\mathbf{x}) = \mathbf{P}_\pi \mathbf{x}$ . При использовании скалярного произведения в качестве внутреннего и

$$\pi_U(\mathbf{x}) = \lambda \mathbf{b} = \mathbf{b} \lambda = \mathbf{b} \frac{\mathbf{b}^T \mathbf{x}}{\|\mathbf{b}\|^2} = \frac{\mathbf{b} \mathbf{b}^T}{\|\mathbf{b}\|^2} \mathbf{x} \quad (3.45)$$

мы сразу видим, что

$$\mathbf{P}_\pi = \frac{\mathbf{b} \mathbf{b}^T}{\|\mathbf{b}\|^2}. \quad (3.46)$$

Отметим, что  $\mathbf{b} \mathbf{b}^T$  (и, следовательно,  $\mathbf{P}_\pi$ ) — это симметричная матрица<sup>2</sup> (с рангом 1), а  $\|\mathbf{b}\|^2 = \langle \mathbf{b}, \mathbf{b} \rangle$  — это скаляр.

Проекционная матрица  $\mathbf{P}_\pi$  проецирует любой вектор  $\mathbf{x} \in \mathbb{R}^n$  на линию, проходящую через начало координат в направлении  $\mathbf{b}$  (или, что эквивалентно, подпространство  $U$  охватывается  $\mathbf{b}$ ).

**ПРИМЕЧАНИЕ** Проекция  $\pi_U(\mathbf{x}) \in \mathbb{R}^n$  все равно является  $n$ -мерным вектором, а не скаляром. Однако координаты  $n$  нам больше не требуются для представления проекции, достаточно одной координаты, если мы хотим выразить ее относительно базисного вектора  $\mathbf{b}$ , являющегося оболочкой подпространства  $U: \lambda$ .



<sup>1</sup> Горизонтальная ось является одномерным подпространством.

<sup>2</sup> Проекционные матрицы всегда симметричны.

**Пример 3.10 (проекция на прямую)**

Найдем проекционную матрицу  $P_\pi$  на прямую, проходящую через начало координат, с оболочкой  $\mathbf{b} = [1 \ 2 \ 2]^\top$ .  $\mathbf{b}$  – это направление и базис одномерного подпространства (линии, проходящей через начало координат).

Опираясь на (3.46), получаем

$$P_\pi = \frac{\mathbf{b}\mathbf{b}^\top}{\mathbf{b}^\top\mathbf{b}} = \frac{1}{9} \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 2 & 2 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 & 2 & 2 \\ 2 & 4 & 4 \\ 2 & 4 & 4 \end{bmatrix}. \quad (3.47)$$

Выберем конкретный  $\mathbf{x}$  и посмотрим, находится ли он в подпространстве, охватываемом  $\mathbf{b}$ . Для  $\mathbf{x} = [1 \ 1 \ 1]^\top$  проекция равна

$$\pi_U(\mathbf{x}) = P_\pi \mathbf{x} = \frac{1}{9} \begin{bmatrix} 1 & 2 & 2 \\ 2 & 4 & 4 \\ 2 & 4 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 5 \\ 10 \\ 10 \end{bmatrix} \in \text{span} \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}. \quad (3.48)$$

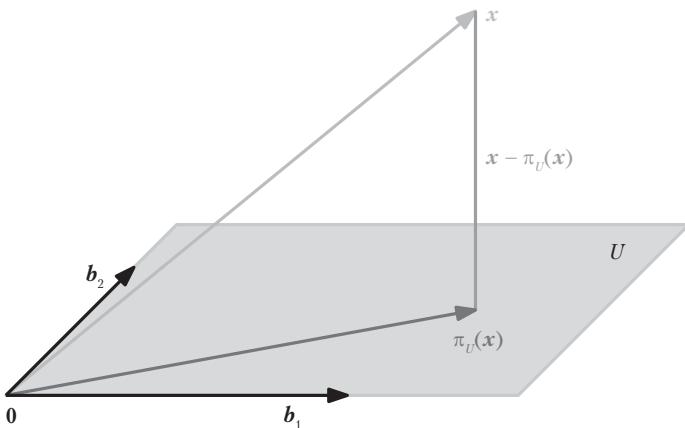
Обратите внимание: применение  $P_\pi$  к  $\pi_U(\mathbf{x})$  ничего не меняет, то есть  $P_\pi \pi_U(\mathbf{x}) = \pi_U(\mathbf{x})$ . Это ожидаемо, поскольку, согласно определению (3.10), мы знаем, что проекционная матрица  $P_\pi$  удовлетворяет  $P_\pi^2 \mathbf{x} = P_\pi \mathbf{x}$  для всех  $\mathbf{x}$ .

**ПРИМЕЧАНИЕ** По результатам главы 4 мы сможем показать, что  $\pi_U(\mathbf{x})$  является собственным вектором  $P_\pi$ , а соответствующее собственное значение равно 1. ♦

### 3.8.2. Проекция на общие подпространства

В дальнейшем рассмотрим ортогональные проекции векторов  $\mathbf{x} \in \mathbb{R}^n$  на пространства меньшей размерности  $U \subseteq \mathbb{R}^n$  с  $\dim(U) = m \geq 1$ . Это проиллюстрировано на рис. 3.11.

Предположим, что  $\mathbf{b}_1 \dots \mathbf{b}_m$  – упорядоченный базис  $U$ . Любая проекция  $\pi_U(\mathbf{x})$  на  $U$  обязательно является элементом  $U$ . Следовательно, они могут быть представлены как линейные комбинации базисных векторов  $\mathbf{b}_1 \dots, \mathbf{b}_m$  от  $U$ , такие что  $\pi_U(\mathbf{x}) = \sum_{i=1}^m \lambda_i \mathbf{b}_i$ .



**Рис. 3.11.** Проекция на двумерное подпространство  $U$  с базисом  $\mathbf{b}_1, \mathbf{b}_2$ . Проекция  $\pi_U(\mathbf{x})$  от  $\mathbf{x} \in \mathbb{R}^3$  на  $U$  может быть выражена как линейная комбинация  $\mathbf{b}_1, \mathbf{b}_2$ , и вектор смещения  $\mathbf{x} - \pi_U(\mathbf{x})$  ортогонален как  $\mathbf{b}_1$ , так и  $\mathbf{b}_2$

Как и в одномерном случае, мы выполняем трехэтапную процедуру, чтобы найти проекцию  $\pi_U(\mathbf{x})$  и проекционную матрицу  $\mathbf{P}_\pi$ .

1. Находим координаты  $\lambda_1, \dots, \lambda_m$  проекции (относительно базиса  $U$ ), такие что линейная комбинация

$$\pi_U(\mathbf{x}) = \sum_{i=1}^m \lambda_i \mathbf{b}_i = \mathbf{B}\boldsymbol{\lambda}, \quad (3.49)$$

$$\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_m] \in \mathbb{R}^{n \times m}, \quad \boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_m]^T \in \mathbb{R}^m, \quad (3.50)$$

является ближайшей к  $\mathbf{x} \in \mathbb{R}^n$ . Как и в одномерном случае, «ближайшая» означает «минимальное расстояние», что подразумевает следующее: вектор, соединяющий  $\pi_U(\mathbf{x}) \in U$  и  $\mathbf{x} \in \mathbb{R}^n$ , должен быть ортогонален всем базисным векторам  $U$ . Следовательно, мы получаем  $m$  одновременных условий (беря скалярное произведение в качестве внутреннего).

$$\begin{aligned} \langle \mathbf{b}_1, \mathbf{x} - \pi_U(\mathbf{x}) \rangle &= \mathbf{b}_1^T (\mathbf{x} - \pi_U(\mathbf{x})) = 0 \\ &\vdots \end{aligned} \quad (3.51)$$

$$\langle \mathbf{b}_m, \mathbf{x} - \pi_U(\mathbf{x}) \rangle = \mathbf{b}_m^T (\mathbf{x} - \pi_U(\mathbf{x})) = 0, \quad (3.52)$$

что, при  $\pi_U(\mathbf{x}) = \mathbf{B}\boldsymbol{\lambda}$ , можно записать как

$$\begin{aligned} \mathbf{b}_1^T (\mathbf{x} - \mathbf{B}\boldsymbol{\lambda}) &= 0 \\ &\vdots \end{aligned} \quad (3.53)$$

$$\mathbf{b}_m^T (\mathbf{x} - \mathbf{B}\boldsymbol{\lambda}) = 0, \quad (3.54)$$

так что мы получим однородную систему линейных уравнений

$$\begin{bmatrix} \mathbf{b}_1^T \\ \vdots \\ \mathbf{b}_m^T \end{bmatrix} \begin{bmatrix} \mathbf{x} - \mathbf{B}\lambda \end{bmatrix} = \mathbf{0} \Leftrightarrow \mathbf{B}^T (\mathbf{x} - \mathbf{B}\lambda) = \mathbf{0} \Leftrightarrow \quad (3.55)$$

$$\Leftrightarrow \mathbf{B}^T \mathbf{B} \lambda = \mathbf{B}^T \mathbf{x}. \quad (3.56)$$

Последнее выражение называется *нормальным уравнением*. Поскольку  $\mathbf{b}_1, \dots, \mathbf{b}_m$  являются базисом  $U$  и, следовательно, они линейно независимы,  $\mathbf{B}^T \mathbf{B} \in \mathbb{R}^{m \times m}$  регулярна и может быть обращена. Таким образом, мы можем решить ее с коэффициентами/координатами

$$\lambda = (\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{x}. \quad (3.57)$$

Матрица  $(\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T$  также называется *псевдообратной*  $\mathbf{B}$ , которую можно вычислять для неквадратных матриц  $\mathbf{B}$ . При этом требуется лишь, чтобы  $\mathbf{B}^T \mathbf{B}$  была положительно определенной, а ситуация такова, если у  $\mathbf{B}$  полный ранг. При применении на практике (например, с линейной регрессией) в  $\mathbf{B}^T \mathbf{B}$  часто добавляется «поправочный член» в  $\mathbf{I}$ , обеспечивающий повышенную численную устойчивость и положительную определенность. Этот «край» можно тщательно вычислить при помощи байесовского инференса. Подробнее об этом рассказано в главе 9.

- Находим проекцию  $\pi_U(\mathbf{x}) \in U$ . Мы уже установили, что  $\pi_U(\mathbf{x}) = \mathbf{B}\lambda$ . Следовательно, при (3.57)

$$\pi_U(\mathbf{x}) = \mathbf{B}(\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{x}. \quad (3.58)$$

- Находим проекционную матрицу  $\mathbf{P}_\pi$ . Из (3.58) видно, что проекционная матрица, решающая  $\mathbf{P}_\pi = \pi_U(\mathbf{x})$ , должна быть

$$\mathbf{P}_\pi = \mathbf{B}(\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T. \quad (3.59)$$

**ПРИМЕЧАНИЕ** Решение для проецирования на общие подпространства включает одномерный случай как специальный: если  $\dim(U) = 1$ , то  $\mathbf{B}^T \mathbf{B} \in \mathbb{R}$  это скаляр, и можно переписать проекционную матрицу  $\mathbf{P}_\pi = \mathbf{B}(\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T$  как  $\mathbf{P}_\pi = \frac{\mathbf{B} \mathbf{B}^T}{\mathbf{B}^T \mathbf{B}}$ , которая как раз является проекционной матрицей, приведенной в (3.46). ◆

**Пример 3.11 (проекция на двумерное подпространство)**

Для подпространства  $U = \text{span} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix} \subseteq \mathbb{R}^3$  и  $\mathbf{x} = \begin{bmatrix} 6 \\ 0 \\ 0 \end{bmatrix} \in \mathbb{R}^3$  найдем

координаты  $\lambda$  от  $\mathbf{x}$  в терминах подпространства  $U$ , проекционной точки  $\pi_U(\mathbf{x})$  и проекционной матрицы  $P_\pi$ .

Сначала обратим внимание, что порождающее множество  $U$  является базисом (линейная независимость) и запишем базисные векторы  $U$  в мат-

$$\text{рицу } \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix}.$$

Далее вычислим матрицу  $\mathbf{B}^T \mathbf{B}$  и вектор  $\mathbf{B}^T \mathbf{x}$  как

$$\mathbf{B}^T \mathbf{B} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 3 & 3 \\ 3 & 5 \end{bmatrix} = \begin{bmatrix} 6 \\ 6 \\ 6 \end{bmatrix}, \quad \mathbf{B}^T \mathbf{x} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 6 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \\ 0 \end{bmatrix}. \quad (3.60)$$

Затем решим нормальное уравнение  $\mathbf{B}^T \mathbf{B} \lambda = \mathbf{B}^T \mathbf{x}$ , чтобы найти  $\lambda$ :

$$\begin{bmatrix} 3 & 3 \\ 3 & 5 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \end{bmatrix} \Leftrightarrow \lambda = \begin{bmatrix} 5 \\ -3 \end{bmatrix}. \quad (3.61)$$

В-четвертых, проекцию  $\pi_U(\mathbf{x})$  от  $\mathbf{x}$  на  $U$ , то есть на пространство столбцов  $\mathbf{B}$ , можно напрямую вычислить при помощи

$$\pi_U(\mathbf{x}) = \mathbf{B} \lambda = \begin{bmatrix} 5 \\ 2 \\ -1 \end{bmatrix}. \quad (3.62)$$

Соответствующая *проекционная ошибка*<sup>1</sup> — это норма разностного вектора между исходным вектором и его проекцией на  $U$ , то есть

$$\|\mathbf{x} - \pi_U(\mathbf{x})\| = \| [1 \ -2 \ 1]^T \| = \sqrt{6}. \quad (3.63)$$

В-пятых, проекционная матрица (для любого  $\mathbf{x} \in \mathbb{R}^3$ ) дается как

$$\mathbf{P}_\pi = \mathbf{B} (\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T = \frac{1}{6} \begin{bmatrix} 5 & 2 & -1 \\ 2 & 2 & 2 \\ -1 & 2 & 5 \end{bmatrix}. \quad (3.64)$$

---

<sup>1</sup> Проекционную ошибку также называют ошибкой восстановления.

Чтобы проверить результаты, можно (a) проверить, ортогонален ли вектор смещения  $\pi_U(\mathbf{x}) - \mathbf{x}$  всем базисным векторам  $U$  и (b) убедиться, что  $\mathbf{P}_\pi = \mathbf{P}_\pi^2$  (определение 3.10). ◆

**ПРИМЕЧАНИЕ** Проекции  $\pi_U(\mathbf{x})$  все равно являются векторами в  $\mathbb{R}^n$ , хотя и находятся в  $m$ -мерном пространстве  $U \subseteq \mathbb{R}^n$ . Правда, чтобы представить спроцированный вектор, нам нужны только  $m$ -координаты  $\lambda_1, \dots, \lambda_m$  относительно базисных векторов  $\mathbf{b}_1, \dots, \mathbf{b}_m$  от  $U$ . ◆

**ПРИМЕЧАНИЕ** В векторных пространствах с общими внутренними произведениями требуется внимательно действовать при вычислении углов и расстояний, которые определяются при помощи внутреннего произведения. ◆

Проекции позволяют рассматривать такие ситуации, в которых у нас есть система  $A\mathbf{x} = \mathbf{b}$ , не имеющая решения<sup>1</sup>. Как вы помните, это означает, что  $\mathbf{b}$  не входит в оболочку  $A$ , то есть вектор  $\mathbf{b}$  не относится к подпространству, охватываемому столбцами  $A$ . Учитывая, что линейное уравнение не может быть решено точно, можно найти для него *приближенное решение*. Идея в том, чтобы найти в подпространстве, охватываемом столбцами  $A$ , такой вектор, который будет ближайшим к  $\mathbf{b}$ , то есть вычислить ортогональную проекцию  $\mathbf{b}$  на подпространство, охватываемое столбцами  $A$ . Такая проблема часто возникает на практике и решается *методом наименьших квадратов* (здесь предполагается, что скалярное произведение является внутренним) переопределенной системы уравнений. Эта проблема подробнее рассматривается в разделе 9.4. Использование ошибок реконструкции (3.63) — один из способов вывода анализа главных компонент (раздел 10.3).

**ПРИМЕЧАНИЕ** Мы только что рассмотрели проекции вектора  $\mathbf{x}$  на подпространство  $U$  с базисными векторами  $\{\mathbf{b}_1, \dots, \mathbf{b}_k\}$ . Если базис является ОНБ (ортонормированным), то есть (3.33) и (3.34) удовлетворяются, то проекционное уравнение значительно упрощается до

$$\pi_U(\mathbf{x}) = \mathbf{B}\mathbf{B}^\top \mathbf{x}, \quad (3.65)$$

поскольку  $\mathbf{B}^\top \mathbf{B} = \mathbf{I}$ , с координатами

$$\lambda = \mathbf{B}^\top \mathbf{x}. \quad (3.66)$$

Таким образом, нам больше не приходится рассчитывать обратную от (3.58), и мы экономим время на вычисления. ◆

<sup>1</sup> Для систем линейных уравнений, не имеющих решений, при помощи проекций можно находить приближенные решения.

### 3.8.3. Ортогонализация Грама — Шмидта

Проекции лежат в основе метода Грама — Шмидта, позволяющего конструктивно преобразовать любой базис  $(\mathbf{b}_1 \dots \mathbf{b}_n)$   $n$ -мерного векторного пространства  $V$  в ортогональный/ортонормированный базис  $(\mathbf{u}_1 \dots \mathbf{u}_n)$  от  $V$ . Такой базис всегда существует (Liesen and Mehrmann, 2015), и  $\text{span}[\mathbf{b}_1 \dots \mathbf{b}_n] = \text{span}[\mathbf{u}_1 \dots \mathbf{u}_n]$ . Метод ортогонализации Грама — Шмидта итеративно собирает ортогональный базис  $(\mathbf{u}_1, \dots, \mathbf{u}_n)$  из любого базиса  $(\mathbf{b}_1, \dots, \mathbf{b}_n)$  от  $V$  следующим образом:

$$\mathbf{u}_1 := \mathbf{b}_1; \quad (3.67)$$

$$\mathbf{u}_k := \mathbf{b}_k - \pi_{\text{span}[\mathbf{u}_1, \dots, \mathbf{u}_{k-1}]}(\mathbf{b}_k), k = 2, \dots, n. \quad (3.68)$$

В (3.68)  $k$ -й базисный вектор  $\mathbf{b}_k$  проецируется на подпространство, охватывающее первыми  $k - 1$  построенными ортогональными векторами  $\mathbf{u}_1 \dots \mathbf{u}_{k-1}$  (раздел 3.8.2.) Эта проекция затем вычитается из  $\mathbf{b}_k$  и дает вектор  $\mathbf{u}_k$ , ортогональный  $(k - 1)$ -мерному подпространству, охватываемому  $\mathbf{u}_1, \dots, \mathbf{u}_{k-1}$ . Если повторить эту процедуру для всех  $n$  базисных векторов  $\mathbf{b}_1, \dots, \mathbf{b}_n$ , то получается ортогональный базис  $(\mathbf{u}_1, \dots, \mathbf{u}_n)$  от  $V$ . Если нормировать  $\mathbf{u}_k$ , то получим ОНБ, где  $\|\mathbf{u}_k\| = 1$  для  $k = 1, \dots, n$ .



**Рис. 3.12.** Ортогонализация Грама — Шмидта. (a) Неортогональный базис  $(\mathbf{b}_1, \mathbf{b}_2)$  от  $\mathbb{R}^2$ . (b) Первый построенный базисный вектор  $\mathbf{u}_1$  и ортогональная проекция  $\mathbf{b}_2$  на  $\text{span}[\mathbf{u}_1]$ . (c) Ортогональный базис  $(\mathbf{u}_1, \mathbf{u}_2)$  от  $\mathbb{R}^2$

#### Пример 3.12 (ортогонализация Грама — Шмидта)

Рассмотрим базис  $(\mathbf{b}_1, \mathbf{b}_2)$  от  $\mathbb{R}^2$ , где

$$\mathbf{b}_1 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \quad \mathbf{b}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}; \quad (3.69);$$

см. также рис. 3.12(a). Воспользовавшись методом Грама — Шмидта, мы строим ортогональный базис  $(\mathbf{u}_1, \mathbf{u}_2)$  от  $\mathbb{R}^2$  следующим образом (считая, что скалярное произведение эквивалентно внутреннему произведению):

$$\mathbf{u}_1 := \mathbf{b}_1 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \quad (3.70),$$

$$\mathbf{u}_2 := \mathbf{b}_2 - \pi_{\text{span}[\mathbf{u}_1]}(\mathbf{b}_2) \stackrel{(3.45)}{=} \mathbf{b}_2 - \frac{\mathbf{u}_1 \mathbf{u}_1^\top}{\|\mathbf{u}_1\|^2} \mathbf{b}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \quad (3.71)$$

Эти шаги проиллюстрированы на рис. 3.12(b) и (c). Мы сразу видим, что  $\mathbf{u}_1$  и  $\mathbf{u}_2$  ортогональны, то есть что  $\mathbf{u}_1^\top \mathbf{u}_2 = 0$ .

### 3.8.4. Проекция на аффинные подпространства

До сих пор мы обсуждали, как спроектировать вектор на пространство с меньшим количеством измерений  $U$ . В дальнейшем будет показано, как спроектировать вектор на аффинное подпространство.

Рассмотрим ситуацию, показанную на рис. 3.13(a). Имеем аффинное подпространство  $L = \mathbf{x}_0 + U$ , где  $\mathbf{b}_1, \mathbf{b}_2$  — это базисные векторы  $U$ . Чтобы определить ортогональную проекцию  $\pi_L(\mathbf{x})$  от  $\mathbf{x}$  на  $L$ , переформулируем задачу в ту форму, в которой умеем ее решать: как проекцию на векторное подпространство. Чтобы прийти к этому, вычтем опорную точку  $\mathbf{x}_0$  из  $\mathbf{x}$  и из  $L$ , так чтобы  $L - \mathbf{x}_0 = U$  в точности равнялось векторному подпространству  $U$ . Теперь мы можем воспользоваться ортогональными проекциями на подпространство, которые обсуждались в разделе 3.8.2, и получить проекцию  $\pi_U(\mathbf{x} - \mathbf{x}_0)$ , показанную на рис. 3.13(b). Теперь эту проекцию можно транслировать обратно в  $L$ , сложив ее с  $\mathbf{x}_0$ , чтобы получить ортогональную проекцию на аффинное пространство  $L$  как

$$\pi_L(\mathbf{x}) = \mathbf{x}_0 + \pi_U(\mathbf{x} - \mathbf{x}_0), \quad (3.72)$$

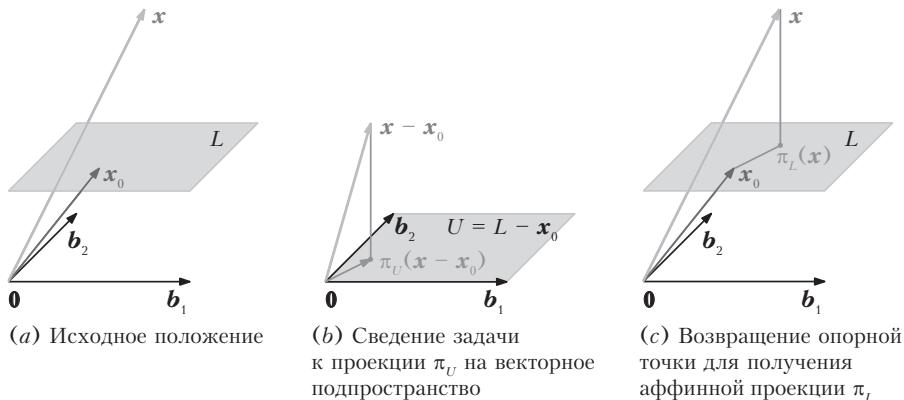
где  $\pi_U(\cdot)$  — ортогональная проекция на подпространство  $U$ , то есть пространство направлений от  $L$ ; см. рис. 3.13(c).

Из рис. 3.13 также очевидно, что расстояние  $\mathbf{x}$  от аффинного пространства  $L$  идентично расстоянию  $\mathbf{x} - \mathbf{x}_0$  от  $U$ , то есть

$$d(\mathbf{x}, L) = \|\mathbf{x} - \pi_L(\mathbf{x})\| = \|\mathbf{x} - (\mathbf{x}_0 + \pi_U(\mathbf{x} - \mathbf{x}_0))\| \quad (3.73a)$$

$$= d(\mathbf{x} - \mathbf{x}_0, \pi_U(\mathbf{x} - \mathbf{x}_0)). \quad (3.73b)$$

В разделе 12.1 мы воспользуемся проекциями на аффинное подпространство для вывода концепции разделяющей гиперплоскости.



**Рис. 3.13.** Проекция на аффинное пространство. (a) Исходное положение. (b) Положение, сдвинутое на  $-x_0$ , так чтобы  $x - x_0$  можно было спроектировать на пространство направлений  $U$ . (c) Проекция транслируется обратно на  $x_0 + \pi_U(x - x_0)$ , что дает нам окончательную ортогональную проекцию  $\pi_L(x)$

### 3.9. ПОВОРОТЫ

Сохранение длин и углов, о чём мы говорили в разделе 3.4, — это две характеристики линейных отображений при работе с ортогональными матрицами преобразований. Далее мы подробнее рассмотрим конкретные ортогональные матрицы преобразований, описывающие повороты.

*Поворот* — это линейное отображение (точнее, автоморфизм евклидова векторного пространства), поворачивающий плоскость на угол  $\theta$  относительно начала координат. То есть начало координат — это фиксированная точка. По общепринятому соглашению, для положительного угла с  $\theta > 0$  поворот производится против часовой стрелки. На рис. 3.14 показан пример, чья матрица преобразования

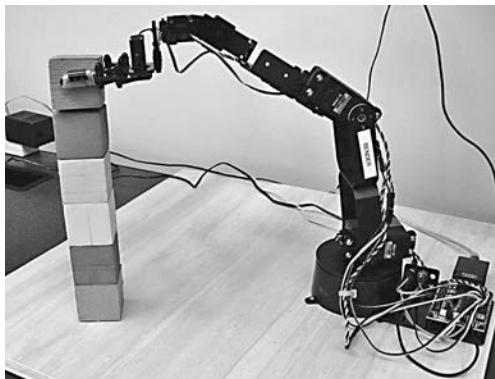
$$\mathbf{R} = \begin{bmatrix} -0,38 & -0,92 \\ 0,92 & -0,38 \end{bmatrix}. \quad (3.74)$$

К важным областям, в которых применяются повороты, относятся, в частности, компьютерная графика и робототехника. Например, в робототехнике часто

требуется знать, как поворачивать сочленения манипулятора, чтобы робот мог правильно подхватить и поставить предмет; см. рис. 3.15.



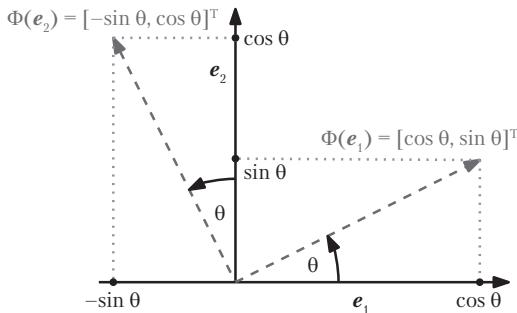
**Рис. 3.14.** При повороте объект в плоскости поворачивается относительно начала координат. Если угол поворота положительный, то поворот осуществляется против часовой стрелки



**Рис. 3.15.** Манипулятору робота необходимо поворачивать сочленения, чтобы подхватывать предметы или правильно их расставлять. Рисунок взят из работы Deisenroth et al. (2015)

### 3.9.1. Повороты в $\mathbb{R}^2$

Рассмотрим стандартный базис  $\left\{\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right\}$  от  $\mathbb{R}^2$ , определяющий стандартную систему координат в  $\mathbb{R}^2$ . Мы собираемся повернуть эту систему координат на угол  $\theta$ , как показано на рис. 3.16. Обратите внимание: после поворота векторы остались линейно независимыми и поэтому являются базисом  $\mathbb{R}^2$ . Таким образом, при повороте выполняется изменение базиса.



**Рис. 3.16.** Поворот стандартного базиса в  $\mathbb{R}^2$  на угол  $\theta$

Повороты  $\Phi$  являются линейными отображениями, поэтому их можно выразить при помощи *матрицы поворота*  $R(\theta)$ . Тригонометрия (см. рис. 3.16) позволяет определить координаты осей после поворота (образ  $\Phi$ ) относительно стандартного базиса в  $\mathbb{R}^2$ . Получаем

$$\Phi(e_1) = \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix}, \quad \Phi(e_2) = \begin{bmatrix} -\sin \theta \\ \cos \theta \end{bmatrix}. \quad (3.75)$$

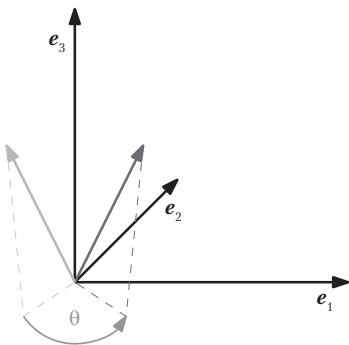
Соответственно, матрица поворота, осуществляющая изменение базиса в координаты после поворота  $R(\theta)$ , дается как

$$R(\theta) = [\Phi(e_1) \quad \Phi(e_2)] = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}. \quad (3.76)$$

### 3.9.2. Повороты в $\mathbb{R}^3$

В отличие от случая с  $\mathbb{R}^2$ , в  $\mathbb{R}^3$  можно поворачивать любую двумерную плоскость по одномерной оси. Чтобы указать общую матрицу поворота, прежде всего описать, как предполагается повернуть образы стандартного базиса ( $e_1, e_2, e_3$ ), а затем убедиться, что эти образы  $Re_1, Re_2, Re_3$  ортонормальны друг другу. Можно получить общую матрицу поворота  $R$ , скомбинировав образы стандартного базиса.

Чтобы получить понятный угол поворота, нужно определить, что такое «против часовой стрелки» при работе более чем в двух измерениях. Принято считать, что поворот «против часовой стрелки» (в плоскости) относится к повороту по оси «лицом вперед, от кончика к началу координат». Следовательно, в  $\mathbb{R}^3$  возможны три варианта поворота (в плоскости) по трем стандартным базисным векторам (рис. 3.17).



**Рис. 3.17.** Поворот вектора (серого) в  $\mathbb{R}^3$  на угол  $\theta$  по оси  $e_3$ . Повернутый вектор показан светлой стрелкой

- Поворот по оси  $e_1$ .

$$\mathbf{R}_1(\theta) = [\Phi(e_1) \quad \Phi(e_2) \quad \Phi(e_3)] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}. \quad (3.77)$$

Здесь координата  $e_1$  является фиксированной, а поворот против часовой стрелки выполняется в плоскости  $e_2, e_3$ .

- Поворот по оси  $e_2$ .

$$\mathbf{R}_2(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}. \quad (3.78)$$

Если повернуть плоскость  $e_1, e_3$  по оси  $e_2$ , то мы смотрим на ось  $e_2$  с ее «кончиком» по направлению к началу координат.

- Поворот по оси  $e_3$ .

$$\mathbf{R}_3(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (3.79)$$

Это проиллюстрировано на рис. 3.17.

### 3.9.3. Поворот в $n$ измерениях

Обобщение поворотов от 2D- и 3D- до  $n$ -мерных евклидовых векторных пространств можно интуитивно описать как фиксацию  $n - 2$  измерений и огра-

ничение вращения двумерной плоскостью в  $n$ -мерном пространстве. Как и в трехмерном случае, можно вращать любую плоскость (двумерное подпространство  $\mathbb{R}^n$ ).

**Определение 3.11 (поворот Гивенса).** Пусть  $V$  — это  $n$ -мерное евклидово векторное пространство, а  $\Phi : V \rightarrow V$  это автоморфизм с матрицей преобразования

$$\mathbf{R}_{ij}(\theta) := \begin{bmatrix} \mathbf{I}_{i-1} & \mathbf{0} & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cos\theta & \mathbf{0} & -\sin\theta & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{j-i-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \sin\theta & \mathbf{0} & \cos\theta & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \mathbf{0} & \mathbf{I}_{n-j} \end{bmatrix} \in \mathbb{R}^{n \times n}, \quad (3.80)$$

для  $1 \leq i < j \leq n$  и  $\theta \in \mathbb{R}$ . Тогда  $\mathbf{R}_{ij}(\theta)$  называется *поворотом Гивенса*. В сущности,  $\mathbf{R}_{ij}(\theta)$  — это единичная матрица  $\mathbf{I}_n$  с

$$r_{ii} = \cos\theta, \quad r_{ij} = -\sin\theta, \quad r_{ji} = \sin\theta, \quad r_{jj} = \cos\theta. \quad (3.81)$$

В двух измерениях (то есть  $n = 2$ ) мы получаем (3.76) в качестве частного случая.

### 3.9.4. Свойства поворотов

У поворотов обнаруживается ряд полезных свойств, которые можно вывести, если трактовать повороты как ортогональные матрицы (определение 3.8):

- При поворотах сохраняются расстояния, например  $\|\mathbf{x} - \mathbf{y}\| = \|\mathbf{R}_\theta(\mathbf{x}) - \mathbf{R}_\theta(\mathbf{y})\|$ . Иными словами, после такого преобразования, как поворот, расстояния между любыми двумя точками остаются неизменными.
- При поворотах сохраняются углы, то есть угол между  $\mathbf{R}_\theta(\mathbf{x})$  и  $\mathbf{R}_\theta(\mathbf{y})$  равен углу между  $\mathbf{x}$  и  $\mathbf{y}$ .
- Повороты в трех (или более) измерениях обычно некоммутативны. Следовательно, важен порядок, в котором применяются повороты, даже если это повороты вокруг одной и той же точки. Коммутативны только повороты векторов в двух измерениях, такие что  $\mathbf{R}(\phi)\mathbf{R}(\theta) = \mathbf{R}(\theta)\mathbf{R}(\phi)$  для всех  $\phi, \theta \in [0, 2\pi]$ . Они образуют абелеву группу (при умножении), только если поворачиваются вокруг одной и той же точки (например, начала координат).

## 3.10. ДОПОЛНИТЕЛЬНОЕ ЧТЕНИЕ

В этой главе был дан краткий обзор некоторых важнейших концепций аналитической геометрии, которыми мы воспользуемся в следующих главах книги. Для более широкого и углубленного обзора некоторых из представленных нами

концепций отсылаем вас к следующим отличным книгам: Axler (2015) и Boyd and Vandenberghe (2018).

Внутренние произведения позволяют определять конкретные базисы векторных (под)пространств, где каждый вектор ортогонален всем остальным (ортогональные базисы), используя при этом метод Грама — Шмидта. Эти базисы важны при оптимизации и в числовых алгоритмах для решения систем линейных уравнений. Например, методы подпространства Крылова, такие как сопряженные градиенты или обобщенный метод минимальных невязок (GMRES), минимизируют остаточные ошибки, ортогональные друг другу (Stoer and Burlirsch, 2002).

В машинном обучении внутренние произведения важны в контексте методов ядра (Schölkopf and Smola, 2002). Методы ядра опираются на тот факт, что многие линейные алгоритмы можно выразить чисто на уровне вычислений внутренних произведений. Затем «прием с ядром» позволяет неявно вычислить эти внутренние произведения в (потенциально бесконечномерном) признаковом пространстве, даже не обладая явной информацией об этом признаковом пространстве. Так обеспечивается «нелинеаризация» многих алгоритмов, применяемых в машинном обучении, в частности ядерного анализа главных компонент (Schölkopf et al., 1997) для снижения размерности. Гауссовые процессы (Rasmussen and Williams, 2006) также попадают в категорию методов ядра, являющихся стандартом качества в современной вероятностной регрессии (подгонка кривых под точки данных). Идея ядер более подробно исследована в главе 12.

Проекции часто используются в компьютерной графике, например для генерации теней. При оптимизации ортогональные проекции часто используются для (итеративной) минимизации остаточных ошибок. Это также находит применение в машинном обучении, например при линейной регрессии, когда требуется найти (линейную) функцию, минимизирующую остаточные ошибки, то есть длины ортогональных проекций данных на линейную функцию (Bishop, 2006). Эта тема будет подробнее рассмотрена в главе 9. При анализе главных компонент (Pearson, 1901; Hotelling, 1933) также используются проекции для сокращения размерности данных с исходно высокой размерностью. Об этом мы подробнее поговорим в главе 10.

## УПРАЖНЕНИЯ

**3.1.** Покажите, что  $\langle \cdot, \cdot \rangle$  определено для всех  $\mathbf{x} = [x_1, x_2]^T \in \mathbb{R}^2$  и  $\mathbf{y} = [y_1, y_2]^T \in \mathbb{R}^2$  на

$$\langle \mathbf{x}, \mathbf{y} \rangle := x_1 y_2 - (x_1 y_2 + x_2 y_1) + 2(x_2 y_2)$$

и является скалярным произведением.

**3.2.** Рассмотрим  $\mathbb{R}^2$  с  $\langle \cdot, \cdot \rangle$ , определенным для всех  $x$  и  $y$  в  $\mathbb{R}^2$  как

$$\langle x, y \rangle := x^T \underbrace{\begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix}}_{=:A} y.$$

Является ли  $\langle \cdot, \cdot \rangle$  скалярным произведением?

**3.3.** Вычислите расстояние между

$$x = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \quad y = \begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix},$$

используя

a.  $\langle x, y \rangle := x^T y.$

b.  $\langle x, y \rangle := x^T A y, \quad A := \begin{bmatrix} 2 & 1 & 0 \\ 1 & 3 & -1 \\ 0 & -1 & 2 \end{bmatrix}.$

**3.4.** Вычислите угол между

$$x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad y = \begin{bmatrix} -1 \\ -1 \end{bmatrix},$$

используя

a.  $\langle x, y \rangle := x^T y.$

b.  $\langle x, y \rangle := x^T B y, \quad B := \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}.$

**3.5.** Рассмотрим евклидово векторное пространство  $\mathbb{R}^5$  со скалярным произведением. Подпространство  $U \subseteq \mathbb{R}^5$  и  $x \in \mathbb{R}^5$  задаются формулами

$$U = \text{span} \left[ \begin{bmatrix} 0 \\ -1 \\ 2 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ -3 \\ 1 \\ -1 \\ 2 \end{bmatrix}, \begin{bmatrix} -3 \\ 4 \\ 1 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ -3 \\ 5 \\ 0 \\ 7 \end{bmatrix} \right], \quad x = \begin{bmatrix} -1 \\ -9 \\ -1 \\ 4 \\ 1 \end{bmatrix}.$$

a. Определите ортогональную проекцию  $\pi_U(x)$  точки  $x$  на  $U$ .

b. Определите расстояние  $d(x, U)$ .

**3.6.** Рассмотрим  $\mathbb{R}^3$  с внутренним произведением

$$\langle \mathbf{x}, \mathbf{y} \rangle := \mathbf{x}^T \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \mathbf{y}.$$

Помимо всего, мы определим  $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$  как стандартный/канонический базис в  $\mathbb{R}^3$ .

a. Определите ортогональную проекцию  $\pi_U(\mathbf{e}_2)$  точки  $\mathbf{e}_2$  на

$$U = \text{span}[\mathbf{e}_1, \mathbf{e}_3].$$

Подсказка: ортогональность определяется скалярным произведением.

b. Вычислите расстояние  $d(\mathbf{e}_2, U)$ .

c. Изобразите сценарий: стандартные базисные векторы и  $\pi_U(\mathbf{e}_2)$ .

**3.7.** Пусть  $V$  — векторное пространство  $\pi$ , эндоморфизм  $V$ .

a. Докажите, что  $\pi$  является проекцией тогда и только тогда, когда  $\text{id}_V - \pi$  является проекцией, где  $\text{id}_V$  — тождественный эндоморфизм на  $V$ .

b. Предположим теперь, что  $\pi$  является проекцией. Вычислите  $\text{Im}(\text{id}_V - \pi)$  и  $\ker(\text{id}_V - \pi)$  как функцию от  $\text{Im}(\pi)$  и  $\ker(\pi)$ .

**3.8.** Используя метод Грама — Шмидта, превратите базис  $B = (\mathbf{b}_1, \mathbf{b}_2)$  двумерного подпространства  $U \subseteq \mathbb{R}^3$  в ОНБ  $C = (\mathbf{c}_1, \mathbf{c}_2)$  пространства  $U$ , где:

$$\mathbf{b}_1 := \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{b}_2 := \begin{bmatrix} -1 \\ 2 \\ 0 \end{bmatrix}.$$

**3.9.** Пусть  $n \in \mathbb{N}^*$  и пусть  $x_1, \dots, x_n > 0$  будут  $n$  положительным действительными числами, такими что  $x_1 + \dots + x_n = 1$ . Воспользуйтесь неравенством Коши — Шварца и покажите, что:

$$a. \sum_{i=1}^n x_i^2 \geq \frac{1}{n}.$$

$$b. \sum_{i=1}^n \frac{1}{x_i} \geq n^2.$$

Подсказка: подумайте о скалярном произведении на  $\mathbb{R}^n$ . Затем выберите конкретные векторы  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$  и примените неравенство Коши — Шварца.

**3.10.** Поверните векторы на  $30^\circ$

$$\mathbf{x}_1 := \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \quad \mathbf{x}_2 := \begin{bmatrix} 0 \\ -1 \end{bmatrix}.$$

# 4

## Матричные разложения

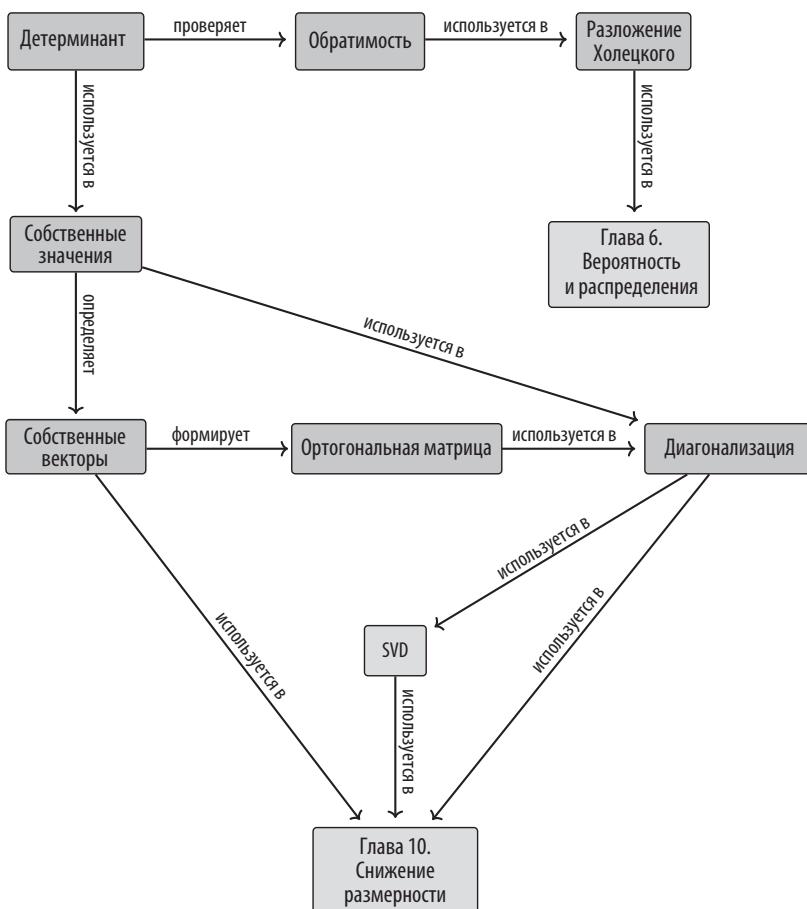
В главах 2 и 3 мы изучили способы работы с векторами и измерения векторов, проекции векторов и линейные отображения. Отображения и преобразования векторов можно удобно описать как операции, выполняемые с помощью матриц. Более того, данные часто также представлены в матричной форме, например где строки матрицы представляют разных людей, а столбцы описывают различные характеристики людей, такие как вес, рост и социально-экономический статус. В этой главе мы рассмотрим три аспекта работы с матрицами: как суммировать матрицы, как раскладывать матрицы и как эти разложения можно использовать для аппроксимации матриц.

Сначала рассмотрим методы, которые позволяют описывать матрицы всего несколькими числами, характеризующими общие свойства матриц. Мы сделаем это в разделах, посвященных детерминантам (раздел 4.1) и собственным значениям (раздел 4.2), для важного частного случая квадратных матриц. Эти характеристические числа имеют важные математические последствия и позволяют быстро понять, какими полезными свойствами обладает матрица. Отсюда мы перейдем к методам разложения матриц: аналогией разложения матриц является разложение чисел, например разложение 21 на простые числа  $7 \cdot 3$ . По этой причине разложение матриц также часто называют *факторизацией матриц*. Разложения матриц используются для описания матрицы посредством другого представления с использованием множителей интерпретируемых матриц.

Сначала мы рассмотрим операцию извлечения квадратного корня для симметричных положительно определенных матриц — разложение Холецкого (раздел 4.3). Отсюда мы рассмотрим два связанных метода факторизации матриц в канонические формы. Первый известен как матричная диагонализация (раздел 4.4), которая позволяет нам представить линейное отображение с помощью матрицы диагонального преобразования, если мы выберем подходящий базис.

Второй метод, сингулярное разложение (раздел 4.5), расширяет эту факторизацию на неквадратные матрицы и считается одним из фундаментальных понятий в линейной алгебре. Эти разложения полезны, поскольку матрицы, представляющие числовые данные, часто очень большие и их трудно анализировать. Мы завершим главу систематическим обзором типов матриц и характерных свойств, которые их различают, в форме матричной таксономии (раздел 4.7).

Методы, которые мы рассмотрим в этой главе, станут важными как в последующих математических главах, таких как глава 6, так и в прикладных главах, таких как уменьшение размерности в главе 10 или оценка плотности в главе 11. Общая структура этой главы изображена в диаграмме связей на рис. 4.1.



**Рис. 4.1.** Ассоциативная карта концепций, представленных в этой главе, а также их использования в других частях книги

## 4.1. ДЕТЕРМИНАНТ И СЛЕД

Детерминанты играют важную роль в линейной алгебре. Детерминант — это математический объект при анализе и решении систем линейных уравнений. Детерминанты определены только для квадратных матриц  $A \in \mathbb{R}^{n \times n}$ , то есть матриц с одинаковым числом строк и столбцов. В этой книге мы пишем детерминант как  $\det(A)$  или иногда как  $|A|^1$ , так что

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix}. \quad (4.1)$$

*Детерминант* квадратной матрицы  $A \in \mathbb{R}^{n \times n}$  это функция, которая отображает  $A$  на действительное число. Прежде чем дать определение детерминанта для общих матриц размера  $n \times n$ , давайте рассмотрим несколько примеров и определим детерминанты для некоторых специальных матриц.

### Пример 4.1 (проверка матрицы на обратимость)

Начнем с исследования, обратима ли квадратная матрица  $A$  (раздел 2.2.2). В самых узких случаях мы уже знаем, когда матрица обратима. Если  $A$  — матрица  $1 \times 1$ , то есть это скалярное число, то  $A = a \Rightarrow A^{-1} = \frac{1}{a}$ . Таким образом,  $a \frac{1}{a} = 1$  выполняется тогда и только тогда, когда  $a \neq 0$ .

Для матриц  $2 \times 2$  по определению обращения (определение 2.3) мы знаем, что  $AA^{-1} = I$ . Тогда с учетом (2.24) обратная к  $A$  есть

$$A^{-1} = \frac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}. \quad (4.2)$$

Следовательно,  $A$  обратима тогда и только тогда, когда

$$a_{11}a_{22} - a_{12}a_{21} \neq 0. \quad (4.3)$$

Эта величина является детерминантом  $A \in \mathbb{R}^{2 \times 2}$ , то есть

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}. \quad (4.4)$$

---

<sup>1</sup> Обозначение детерминанта  $|A|$  не следует путать с абсолютным значением.

Пример 4.1 уже указывает на связь между детерминантами и существованием обратных матриц. Следующая теорема утверждает тот же результат для матриц размера  $n \times n$ .

**Теорема 4.1.** Для любой квадратной матрицы  $\mathbf{A} \in \mathbb{R}^{n \times n}$  верно, что  $\mathbf{A}$  обратима тогда и только тогда, когда  $\det(\mathbf{A}) \neq 0$ .

Существуют явные (в замкнутой форме) выражения для детерминантов малых матриц через элементы матрицы. Для  $n = 1$

$$\det(\mathbf{A}) = \det(a_{11}) = a_{11}. \quad (4.5)$$

Для  $n = 2$

$$\det(\mathbf{A}) = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}, \quad (4.6)$$

что рассматривалось в предыдущем примере.

Для  $n = 3$  (известно как правило Сарруса)

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{21}a_{32}a_{13} + a_{31}a_{12}a_{23} - a_{31}a_{22}a_{13} - a_{11}a_{32}a_{23} - a_{21}a_{12}a_{33}. \quad (4.7)$$

Чтобы запомнить условия умножения в правиле Сарруса, попробуйте проследить за элементами тройных произведений в матрице.

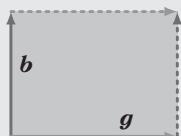
Квадратная матрица  $\mathbf{T}$  называется *верхнетреугольной матрицей*, если  $T_{ij} = 0$  для  $i > j$ , то есть матрица равна нулю ниже своей диагонали. Аналогичным образом определяется *нижнетреугольная* матрица с нулями над ее диагональю. Для треугольной матрицы  $\mathbf{T} \in \mathbb{R}^{n \times n}$  детерминант является произведением диагональных элементов, то есть

$$\det(\mathbf{T}) = \prod_{i=1}^n T_{ii}. \quad (4.8)$$

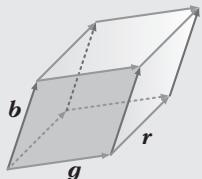
### Пример 4.2 (детерминанты как мера объема)

Понятие детерминанта естественно, когда мы рассматриваем его как отображение набора из  $n$  векторов, охватывающих объект в  $\mathbb{R}^n$ . Оказывается, детерминант  $\det(\mathbf{A})$  представляет собой ориентированный объем (объем со знаком)  $n$ -мерного параллелепипеда, образованного столбцами матрицы  $\mathbf{A}$ .

При  $n = 2$  столбцы матрицы образуют параллелограмм; см. рис. 4.2. По мере уменьшения угла между векторами уменьшается и площадь параллелограмма. Рассмотрим два вектора  $\mathbf{b}, \mathbf{g}$ , которые образуют столбцы матрицы  $\mathbf{A} = [\mathbf{b}, \mathbf{g}]$ . Тогда модуль детерминанта  $\mathbf{A}$  — это площадь параллелограмма с вершинами  $\mathbf{0}, \mathbf{b}, \mathbf{g}, \mathbf{b} + \mathbf{g}$ . В частности, если  $\mathbf{b}, \mathbf{g}$  линейно зависимы, так что  $\mathbf{b} = \lambda\mathbf{g}$  для некоторого  $\lambda \in \mathbb{R}$ , они больше не образуют двумерный параллелограмм. Следовательно, соответствующая площадь равна  $\mathbf{0}$ . Напротив, если  $\mathbf{b}, \mathbf{g}$  линейно независимы и кратны каноническим базисным векторам  $\mathbf{e}_1, \mathbf{e}_2$ , то их можно записать как  $\mathbf{b} = \begin{bmatrix} b \\ 0 \end{bmatrix}$  и  $\mathbf{g} = \begin{bmatrix} 0 \\ g \end{bmatrix}$ , и детерминант  $\begin{bmatrix} b & 0 \\ 0 & g \end{bmatrix} = bg - 0 = bg$ .



**Рис. 4.2.** Площадь параллелограмма (заштрихованная область), натянутого на векторы  $\mathbf{b}$  и  $\mathbf{g}$ , равна  $|\det([\mathbf{b}, \mathbf{g}])|$



**Рис. 4.3.** Объем параллелепипеда (заштрихованный объем), натянутого на векторы  $\mathbf{r}, \mathbf{b}, \mathbf{g}$ , равен  $|\det([\mathbf{r}, \mathbf{b}, \mathbf{g}])|$

Знак детерминанта указывает ориентацию остворных векторов  $\mathbf{b}, \mathbf{g}$  относительно стандартного базиса  $(\mathbf{e}_1, \mathbf{e}_2)$ . На нашем рисунке изменение порядка на  $\mathbf{g}, \mathbf{b}$  меняет местами столбцы  $\mathbf{A}$  и меняет ориентацию заштрихованной области. Получаем знакомую формулу: площадь = высота  $\times$  длина. Это верно и для более высоких размерностей. В  $\mathbb{R}^3$  мы рассматриваем три вектора  $\mathbf{r}, \mathbf{b}, \mathbf{g} \in \mathbb{R}^3$ , охватывающие ребра параллелепипеда, то есть твердое тело с гранями, которые являются параллелограммами (рис. 4.3). Абсолютное значение детерминанта матрицы  $3 \times 3 [\mathbf{r}, \mathbf{b}, \mathbf{g}]$  — это объем твердого тела. Таким образом, детерминант действует как функция, которая измеряет объем со знаком, образованный векторами — столбцами матрицы.

Рассмотрим три линейно независимых вектора  $\mathbf{r}, \mathbf{g}, \mathbf{b} \in \mathbb{R}^3$ , заданных как

$$\mathbf{r} = \begin{bmatrix} 2 \\ 0 \\ -8 \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} 6 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 4 \\ -1 \end{bmatrix}. \quad (4.9)$$

Запись этих векторов как столбцов матрицы

$$\mathbf{A} = [\mathbf{r}, \mathbf{g}, \mathbf{b}] = \begin{bmatrix} 2 & 6 & 1 \\ 0 & 1 & 4 \\ -8 & 0 & -1 \end{bmatrix} \quad (4.10)$$

позволяет нам вычислить объем как

$$V = |\det(\mathbf{A})| = 186. \quad (4.11)$$

Вычисление детерминанта матрицы размера  $n \times n$  требует общего алгоритма для решения случаев для  $n > 3$ , которые будут рассмотрены ниже. Теорема 4.2 ниже сводит проблему вычисления детерминанта матрицы размера  $n \times n$  к вычислению детерминанта матриц размера  $(n-1) \times (n-1)$ . Таким образом, рекурсивно применяя разложение Лапласа (теорема 4.2), можно вычислить детерминанты матриц размера  $n \times n$ , в конечном итоге вычислив детерминанты матриц  $2 \times 2$ .

**Теорема 4.2 (разложение Лапласа).** Рассмотрим матрицу  $\mathbf{A} \in \mathbb{R}^{n \times n}$ . Затем, для всех  $j = 1, \dots, n$ :

1. Разложение по столбцу  $j$

$$\det(\mathbf{A}) = \sum_{k=1}^n (-1)^{k+j} a_{kj} \det(\mathbf{A}_{k,j}). \quad (4.12)$$

2. Разложение по строке  $j$

$$\det(\mathbf{A}) = \sum_{k=1}^n (-1)^{k+j} a_{jk} \det(\mathbf{A}_{j,k}). \quad (4.13)$$

Здесь  $\mathbf{A}_{k,j} \in \mathbb{R}^{(n-1) \times (n-1)}$  — это подматрица матрицы  $\mathbf{A}$ , полученная при удалении строки  $k$  и столбца  $j$ <sup>1</sup>.

### Пример 4.3 (разложение Лапласа)

Вычислим детерминант

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}, \quad (4.14)$$

<sup>1</sup>  $\det(\mathbf{A}_{k,j})$  называется минором, а  $(-1)^{k+j} \det(\mathbf{A}_{k,j})$  — кофактором.

используя разложение Лапласа по первой строке. Применяя (4.13), получим

$$\begin{aligned} \left| \begin{array}{ccc} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 0 & 0 & 1 \end{array} \right| &= (-1)^{1+1} \cdot 1 \left| \begin{array}{cc} 1 & 2 \\ 0 & 1 \end{array} \right| + \\ &+ (-1)^{1+2} \cdot 2 \left| \begin{array}{cc} 3 & 2 \\ 0 & 1 \end{array} \right| + (-1)^{1+3} \cdot 3 \left| \begin{array}{cc} 3 & 1 \\ 0 & 0 \end{array} \right|. \end{aligned} \quad (4.15)$$

Воспользуемся (4.6) для вычисления детерминантов всех матриц  $2 \times 2$  и получим

$$\det(\mathbf{A}) = 1(1 - 0) - 2(3 - 0) + 3(0 - 0) = -5. \quad (4.16)$$

Для полной ясности мы можем сравнить этот результат с вычислением детерминанта по правилу Сарруса (4.7):

$$\begin{aligned} \det(\mathbf{A}) &= 1 \cdot 1 \cdot 1 + 3 \cdot 0 \cdot 3 + 0 \cdot 2 \cdot 2 - 0 \cdot 1 \cdot 3 - \\ &- 1 \cdot 0 \cdot 2 - 3 \cdot 2 \cdot 1 = 1 - 6 = -5. \end{aligned} \quad (4.17)$$

Для  $\mathbf{A} \in \mathbb{R}^{n \times n}$  детерминант проявляет следующие свойства:

- Детерминант произведения матриц равен произведению соответствующих детерминантов,  $\det(\mathbf{AB}) = \det(\mathbf{A})\det(\mathbf{B})$ .
- Детерминанты инвариантны к транспонированию, то есть  $\det(\mathbf{A}) = \det(\mathbf{A}^T)$ .
- Если  $\mathbf{A}$  регулярная (обратимая), то  $\det(\mathbf{A}^{-1}) = \frac{1}{\det(\mathbf{A})}$ .
- Подобные матрицы (определение 2.22) обладают одним и тем же детерминантом. Следовательно, для линейного отображения  $\Phi : V \rightarrow V$  все матрицы преобразования  $\mathbf{A}_\Phi$  отображения  $\Phi$  имеют один и тот же детерминант. Таким образом, детерминант инвариантен к выбору базиса линейного отображения.
- Добавление столбца/строки к другому столбцу/строке не изменяет  $\det(\mathbf{A})$ .
- Умножение столбца/строки с  $\lambda \in \mathbb{R}$  умножает  $\det(\mathbf{A})$  на  $\lambda$ . В частности,  $\det(\lambda\mathbf{A}) = \lambda^n \det(\mathbf{A})$ .
- Перестановка двух строк/столбцов меняет знак  $\det(\mathbf{A})$ .

Благодаря последним трем свойствам исключение Гаусса может быть использовано (раздел 2.1) для вычисления  $\det(\mathbf{A})$  путем приведения  $\mathbf{A}$  к приведенному ступенчатому виду. Можно остановить метод исключения Гаусса, когда  $\mathbf{A}$  имеет треугольную форму, где все элементы ниже диагонали равны 0. Напомним

из (4.8), что детерминант треугольной матрицы равен произведению ее диагональных элементов.

**Теорема 4.3.** Квадратная матрица  $A \in \mathbb{R}^{n \times n}$  имеет детерминант  $\det(A) \neq 0$  тогда и только тогда, когда  $\text{rk}(A) = n$ . Другими словами,  $A$  обратима тогда и только тогда, когда она имеет полный ранг.

Когда математические вычисления выполнялись в основном вручную, вычисление детерминанта считалось важным способом анализа обратимости матрицы. Однако современные подходы к машинному обучению используют прямые численные методы, которые вытеснили явное вычисление детерминанта. Например, в главе 2 мы узнали, что обратные матрицы могут быть вычислены методом исключения Гаусса. Таким образом, метод исключения Гаусса может использоваться для вычисления детерминанта матрицы. Детерминанты будут играть важную теоретическую роль в следующих разделах, особенно когда мы узнаем о собственных значениях и собственных векторах (раздел 4.2) через характеристический полином.

**Определение 4.4.** След квадратной матрицы  $A \in \mathbb{R}^{n \times n}$  определяется как

$$\text{tr}(A) := \sum_{i=1}^n a_{ii}, \quad (4.18)$$

то есть след — это сумма диагональных элементов  $A$ .

След удовлетворяет следующим свойствам:

- $\text{tr}(A + B) = \text{tr}(A) + \text{tr}(B)$  для  $A, B \in \mathbb{R}^{n \times n}$ .
- $\text{tr}(\alpha A) = \alpha \text{tr}(A)$ ,  $\alpha \in \mathbb{R}$  для  $A \in \mathbb{R}^{n \times n}$ .
- $\text{tr}(I_n) = n$ .
- $\text{tr}(AB) = \text{tr}(BA)$  для  $A \in \mathbb{R}^{n \times k}$ ,  $B \in \mathbb{R}^{k \times n}$ .

Можно показать, что существует единственная функция, удовлетворяющая этим четырем свойствам вместе — след (Gohberg et al., 2012). След произведения матриц обладает и более общими свойствами. В частности, след инвариантен относительно циклических перестановок, то есть

$$\text{tr}(AKL) = \text{tr}(KLA) \quad (4.19)$$

для матриц  $A \in \mathbb{R}^{a \times k}$ ,  $K \in \mathbb{R}^{k \times l}$ ,  $L \in \mathbb{R}^{l \times a}$ .

Это свойство распространяется на произведения произвольного числа матриц. Как частный случай (4.19) следует, что для двух векторов  $x, y \in \mathbb{R}^n$

$$\text{tr}(xy^T) = \text{tr}(y^Tx) = y^Tx \in \mathbb{R}. \quad (4.20)$$

Для линейного отображения  $\Phi : V \rightarrow V$ , где  $V$  – векторное пространство, мы определяем след этого отображения, используя след матричного представления  $\Phi$ . Для данного базиса  $V$  мы можем описать  $\Phi$  с помощью матрицы преобразования  $A$ . Тогда след матрицы  $\Phi$  является следом  $A$ . Для другого базиса  $V$  верно, что соответствующая матрица преобразования  $B$  матрицы  $\Phi$  может быть получена заменой базиса на  $S^{-1}AS$  для подходящего  $S$  (см. раздел 2.7.2). Для соответствующего следа  $\Phi$  это означает

$$\text{tr}(B) = \text{tr}(S^{-1}AS) \stackrel{(4.19)}{=} \text{tr}(ASS^{-1}) = \text{tr}(A). \quad (4.21)$$

Следовательно, хотя матричные представления линейных отображений зависят от базиса, след линейного отображения  $\Phi$  от базиса независим. В этом разделе мы рассмотрели детерминанты и следы как функции, характеризующие квадратную матрицу. Объединив понятия детерминантов и следов, теперь мы можем определить важное уравнение, описывающее матрицу  $A$  в терминах многочлена (полинома), которое мы будем широко использовать в следующих разделах.

**Определение 4.5 (характеристический многочлен).** Для  $\lambda \in \mathbb{R}$  и квадратной матрицы  $A \in \mathbb{R}^{n \times n}$

$$p_A(\lambda) := \det(A - \lambda I) = \quad (4.22a)$$

$$= c_0 + c_1\lambda + c_2\lambda^2 + \cdots + c_{n-1}\lambda^{n-1} + (-1)^n\lambda^n, \quad (4.22b)$$

$c_0, \dots, c_{n-1} \in \mathbb{R}$ , является характеристическим многочленом  $A$ . В частности,

$$c_0 = \det(A), \quad (4.23)$$

$$c_{n-1} = (-1)^{n-1}\text{tr}(A). \quad (4.24)$$

Характеристический полином (4.22a) позволит нам вычислить собственные значения и собственные векторы, которые будут рассмотрены в следующем разделе.

## 4.2. СОБСТВЕННЫЕ ЗНАЧЕНИЯ И СОБСТВЕННЫЕ ВЕКТОРЫ

Теперь мы познакомимся с новым способом характеристики матрицы и связанного с ней линейного отображения. Напомним из раздела 2.7.1, что каждое линейное отображение имеет уникальную матрицу преобразования с упорядоченным базисом. Мы можем интерпретировать линейные отображения и связанные с ними матрицы преобразования, выполнив «собственный» анализ. Как мы увидим, собственные значения линейного отображения расскажут нам, как

специальный набор векторов, собственные векторы, преобразуется линейным отображением.

**Определение 4.6.** Пусть  $A \in \mathbb{R}^{n \times n}$  квадратная матрица. Тогда  $\lambda \in \mathbb{R}$  — *собственное значение* оператора  $A$ , а  $x \in \mathbb{R}^n \setminus \{0\}$  — соответствующий *собственный вектор* оператора  $A$ , если

$$Ax = \lambda x. \quad (4.25)$$

Мы называем (4.25) *уравнением для собственных значений*.

**ПРИМЕЧАНИЕ** В литературе и программном обеспечении по линейной алгебре часто применяется соглашение, согласно которому собственные значения сортируются в порядке убывания, так что наибольшее собственное значение и связанный с ним собственный вектор называют первым собственным значением и связанным с ним собственным вектором, а вторые по величине называются вторым собственным значением и связанным с ним собственным вектором и т. д. Однако в учебниках и публикациях порядок упорядочения может отличаться или вообще отсутствовать. Мы не хотим предполагать упорядочение в этой книге, если это не указано явно. ♦

Следующие утверждения эквивалентны:

- $\lambda$  — собственное значение оператора  $A \in \mathbb{R}^{n \times n}$ .
- Существует  $x \in \mathbb{R}^n \setminus \{0\}$  с  $Ax = \lambda x$  или, что то же самое,  $(A - \lambda I_n)x = 0$ , что решается нетривиально, то есть  $x \neq 0$ .
- $\text{rk}(A - \lambda I_n) < n$ .
- $\det(A - \lambda I_n) = 0$ .

**Определение 4.7 (коллинеарность и сонаправленность).** Два вектора, указывающие в одном направлении, называются *сонаправленными*. Два вектора *коллинеарны*, если они указывают в одном или противоположном направлении.

**ПРИМЕЧАНИЕ** Если  $x$  — собственный вектор матрицы  $A$ , связанный с собственным значением  $\lambda$ , то для любого  $c \in \mathbb{R} \setminus \{0\}$  верно, что  $cx$  — собственный вектор матрицы  $A$  с тем же собственным значением, поскольку

$$A(cx) = cAx = c\lambda x = \lambda(cx). \quad (4.26)$$

Таким образом, все векторы, коллинеарные  $x$ , также являются собственными векторами  $A$ . ♦

**Теорема 4.8.**  $\lambda \in \mathbb{R}$  является собственным значением  $A \in \mathbb{R}^{n \times n}$  тогда и только тогда, когда  $\lambda$  является корнем характеристического многочлена  $p_A(\lambda)$  оператора  $A$ .

**Определение 4.9.** Пусть квадратная матрица  $A$  имеет собственное значение  $\lambda_i$ . Алгебраическая кратность  $\lambda_i$  — это количество вхождений этого корня в характеристический многочлен.

**Определение 4.10 (собственное подпространство и собственный спектр).** Для  $A \in \mathbb{R}^{n \times n}$  множество всех собственных векторов  $A$ , связанных с собственным значением  $\lambda$ , охватывает подпространство в  $\mathbb{R}^n$ , которое называется *собственным подпространством*  $A$  относительно  $\lambda$  и обозначается  $E_\lambda$ . Набор всех собственных значений оператора  $A$  называется *собственным спектром* или просто *спектром* оператора  $A$ .

Если  $\lambda$  является собственным значением оператора  $A \in \mathbb{R}^{n \times n}$ , то соответствующее собственное подпространство  $E_\lambda$  является пространством решений однородной системы линейных уравнений  $(A - \lambda I)x = 0$ . Геометрически собственный вектор, соответствующий ненулевому собственному значению, указывает в направлении, которое растягивается линейным отображением. Собственное значение — это коэффициент, на который оно растягивается. Если собственное значение отрицательное, направление растяжения меняется.

#### Пример 4.4 (случай единичной матрицы)

Единичная матрица  $I \in \mathbb{R}^{n \times n}$  имеет характеристический многочлен  $pI(\lambda) = \det(I - \lambda I) = (1 - \lambda)^n = 0$ , который имеет только одно собственное значение  $\lambda = 1$ , встречающееся  $n$  раз. Более того,  $Ix = \lambda x = x$  выполняется для всех векторов  $x \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ . Из-за этого единственное собственное подпространство  $E_1$  единичной матрицы охватывает  $n$  измерений, и все  $n$  стандартных базисных векторов  $\mathbb{R}^n$  являются собственными векторами  $I$ .

К полезным свойствам собственных значений и собственных векторов относятся следующие:

- Матрица  $A$  и ее транспонированная матрица  $A^T$  имеют одинаковые собственные значения, но не обязательно одинаковые собственные векторы.
- Собственное подпространство  $E_\lambda$  является нулевым пространством  $A - \lambda I$ , поскольку

$$Ax = \lambda x \Leftrightarrow Ax - \lambda x = \mathbf{0} \Leftrightarrow \quad (4.27a)$$

$$\Leftrightarrow (A - \lambda I)x = \mathbf{0} \Leftrightarrow x \in \ker(A - \lambda I). \quad (4.27b)$$

- Подобные матрицы (см. определение 2.22) имеют одинаковые собственные значения. Следовательно, линейное отображение  $\Phi$  имеет собственные значения, которые не зависят от выбора базиса его матрицы преобразования.

Это делает собственные значения, вместе с детерминантом и следом, ключевыми характеристическими параметрами линейного отображения, поскольку все они инвариантны при изменении базиса.

- Симметричные положительно определенные матрицы всегда имеют положительные действительные собственные значения.

**Пример 4.5 (вычисление собственных значений, собственных векторов и собственных подпространств)**

Найдем собственные значения и собственные векторы матрицы  $2 \times 2$

$$A = \begin{bmatrix} 4 & 2 \\ 1 & 3 \end{bmatrix}. \quad (4.28)$$

**Шаг 1: характеристический многочлен.** Из нашего определения собственного вектора  $\mathbf{x} \neq \mathbf{0}$  и собственного значения  $\lambda$  матрицы  $A$  найдется такой вектор, что  $A\mathbf{x} = \lambda\mathbf{x}$ , то есть  $(A - \lambda I)\mathbf{x} = \mathbf{0}$ . Поскольку  $\mathbf{x} \neq \mathbf{0}$ , для этого требуется, чтобы ядро (нулевое пространство) матрицы  $A - \lambda I$  содержало больше элементов, чем просто  $\mathbf{0}$ . Это означает, что  $A - \lambda I$  необратима, и, следовательно,  $\det(A - \lambda I) = 0$ . Следовательно, нам нужно вычислить корни характеристического многочлена (4.22a), чтобы найти собственные значения.

**Шаг 2: собственные значения.** Характеристический многочлен равен

$$p_A(\lambda) = \det(A - \lambda I) = \quad (4.29a)$$

$$= \det \left( \begin{bmatrix} 4 & 2 \\ 1 & 3 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right) = \begin{vmatrix} 4-\lambda & 2 \\ 1 & 3-\lambda \end{vmatrix} = \quad (4.29b)$$

$$= (4-\lambda)(3-\lambda) - 2 \cdot 1. \quad (4.29c)$$

Факторизуем характеристический многочлен и получаем

$$p(\lambda) = (4-\lambda)(3-\lambda) - 2 \cdot 1 = 10 - 7\lambda + \lambda^2 = (2-\lambda)(5-\lambda), \quad (4.30)$$

что дает корни  $\lambda_1 = 2$  и  $\lambda_2 = 5$ .

**Шаг 3: собственные векторы и собственные подпространства.** Мы находим собственные векторы, соответствующие этим собственным значениям, рассматривая векторы  $x$ , такие что

$$\begin{vmatrix} 4-\lambda & 2 \\ 1 & 3-\lambda \end{vmatrix} \mathbf{x} = \mathbf{0}. \quad (4.31)$$

При  $\lambda = 5$  получаем

$$\begin{vmatrix} 4-5 & 2 \\ 1 & 3-5 \end{vmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{vmatrix} -1 & 2 \\ 1 & -2 \end{vmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \mathbf{0}. \quad (4.32)$$

Решаем эту однородную систему и получаем пространство решений

$$E_5 = \text{span} \left[ \begin{bmatrix} 2 \\ 1 \end{bmatrix} \right]. \quad (4.33)$$

Это собственное подпространство одномерно, поскольку имеет единственный базисный вектор.

Аналогично находим собственный вектор при  $\lambda = 2$ , решая однородную систему уравнений

$$\begin{vmatrix} 4-2 & 2 \\ 1 & 3-2 \end{vmatrix} \mathbf{x} = \begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix} \mathbf{x} = \mathbf{0}. \quad (4.34)$$

Это означает, что любой вектор  $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ , где  $x_2 = -x_1$ , например  $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ , — собственный вектор с собственным значением 2. Соответствующее собственное подпространство имеет вид

$$E_2 = \text{span} \left[ \begin{bmatrix} 1 \\ -1 \end{bmatrix} \right]. \quad (4.35)$$

Два собственных подпространства  $E_5$  и  $E_2$  в примере 4.5 одномерны, поскольку каждое из них натянуто на один вектор. Однако в других случаях у нас может быть несколько идентичных собственных значений (см. определение 4.9), а собственное подпространство может иметь более одного измерения.

**Определение 4.11.** Пусть  $\lambda_i$  — собственное значение квадратной матрицы  $A$ . Тогда *геометрическая кратность*  $\lambda_i$  — это количество линейно независимых собственных векторов, связанных с  $\lambda_i$ . Другими словами, это размерность собственного подпространства, натянутого на собственные векторы, связанные с  $\lambda_i$ .

**ПРИМЕЧАНИЕ** Геометрическая кратность конкретного собственного значения должна быть не меньше единицы, потому что каждое собственное значение имеет как минимум один связанный собственный вектор. Геометрическая кратность собственного значения не может превышать его алгебраическую кратность, но может быть меньше. ♦

**Пример 4.6**

Матрица  $A = \begin{bmatrix} 2 & 1 \\ 0 & 2 \end{bmatrix}$  имеет два повторяющихся собственных значения  $\lambda_1 = \lambda_2 = 2$  и алгебраическую кратность 2. Однако собственное значение имеет только один отдельный единичный собственный вектор  $x_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$  и, следовательно, геометрическую кратность 1.

### 4.2.1. Графическая интуиция в двух измерениях

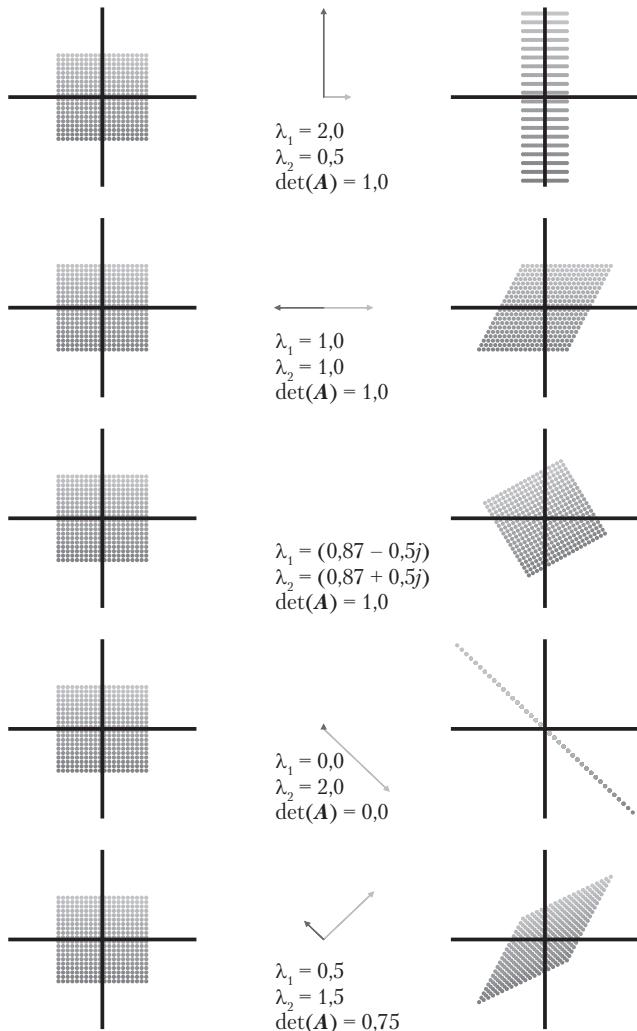
Давайте разберемся с примерами для детерминантов, собственных векторов и собственных значений, используя различные линейные отображения. На рис. 4.4 изображены пять матриц преобразования  $A_1, \dots, A_5$  и их влияние на квадратную сетку точек с центром в начале координат:

- $A_1 = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 2 \end{bmatrix}$ . Направление двух собственных векторов соответствует каноническим базисным векторам в  $\mathbb{R}^2$ , то есть двум координатным осям. Вертикальная ось растягивается в 2 раза (собственное значение  $\lambda_1 = 2$ ), а горизонтальная ось сжимается в  $\frac{1}{2}$  раза (собственное значение  $\lambda_2 = \frac{1}{2}$ ). Отображение сохраняет площадь<sup>1</sup> ( $\det(A_1) = 1 = 2 \cdot \frac{1}{2}$ ).
- $A_2 = \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & 1 \end{bmatrix}$  соответствует отображению сдвига, то есть сдвигает точки вдоль горизонтальной оси вправо, если они находятся на положительной половине вертикальной оси, и влево, если наоборот. Это отображение сохраняет площадь ( $\det(A_2) = 1$ ). Собственное значение  $\lambda_1 = 1 = \lambda_2$  повторяется, а собственные векторы коллинеарны (нарисованы здесь для выделения в двух противоположных направлениях). Это означает, что отображение действует только в одном направлении (горизонтальная ось).

- $A_3 = \begin{bmatrix} \cos\left(\frac{\pi}{6}\right) & -\sin\left(\frac{\pi}{6}\right) \\ \sin\left(\frac{\pi}{6}\right) & \cos\left(\frac{\pi}{6}\right) \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \sqrt{3} & -1 \\ 1 & \sqrt{3} \end{bmatrix}$ . Матрица  $A_3$  вращает точки на  $\frac{\pi}{6} = 30^\circ$  против часовой стрелки и имеет только комплексные собственные значения, отражая, что отображение является вращением (следовательно, собственные

<sup>1</sup> В геометрии свойство сохранения площади этого типа сдвига параллельно осям также известно как принцип Кавальери (Katz, 2004).

векторы не рисуются). Вращение должно сохранять объем, поэтому детерминант равен 1. Подробнее о поворотах см. в разделе 3.9.

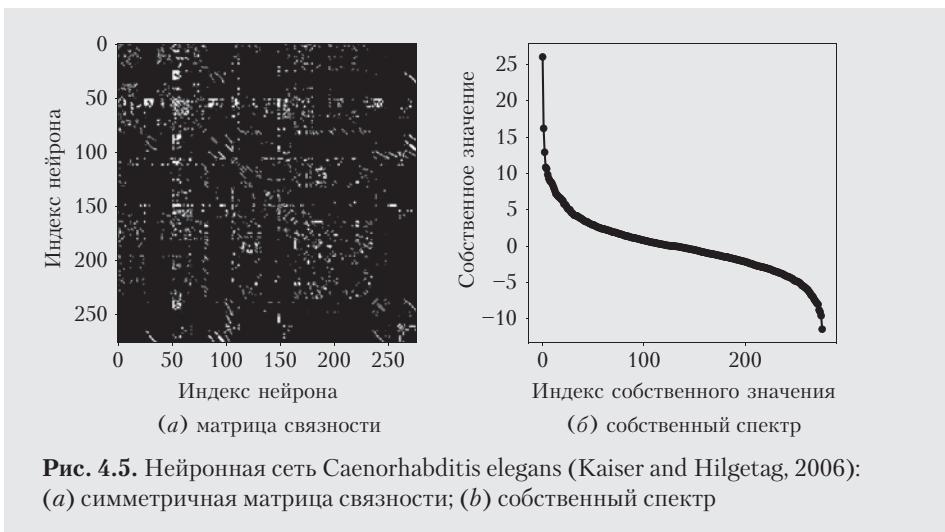


**Рис. 4.4.** Детерминанты и собственные подпространства. Обзор пяти линейных отображений и связанных с ними матриц преобразования  $A_i \in \mathbb{R}^{2 \times 2}$ , проецирующих 400 точек  $\mathbf{x} \in \mathbb{R}^2$  (левый столбец) на целевые точки  $A_i\mathbf{x}$  (правый столбец). В центральном столбце изображен первый собственный вектор, растянутый на соответствующее собственное значение  $\lambda_1$ , и второй собственный вектор — на собственное значение  $\lambda_2$ . Каждая строка показывает действие одной из пяти матриц  $A_i$  на стандартный базис

- $A_4 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$  представляет собой отображение в стандартном базисе, которое сворачивает двумерную область в одно измерение. Поскольку одно собственное значение равно 0, пространство в направлении собственного вектора, соответствующего  $\lambda_1 = 0$ , сжимается, в то время как ортогональный собственный вектор растягивает пространство в  $\lambda_2 = 2$  раза. Следовательно, площадь изображения равна 0.
- $A_5 = \begin{bmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \end{bmatrix}$  — отображение сдвига и растяжения, сжимающее пространство на 25%, поскольку  $|\det(A_5)| = \frac{3}{4}$ . Он растягивает пространство вдоль собственного вектора  $\lambda_2$  в 1,5 раза и сжимает его вдоль ортогонального собственного вектора в 0,5 раза.

#### Пример 4.7 (собственный спектр биологической нейронной сети)

Методы анализа и изучения сетевых данных являются важным компонентом методов машинного обучения. Ключ к пониманию сетей — это связь между сетевыми узлами, особенно если два узла подключены друг к другу или нет. В приложениях для обработки данных часто бывает полезно изучить матрицу, которая фиксирует эти данные о подключении. Мы строим матрицу связности/смежности  $A \in \mathbb{R}^{277 \times 277}$  полной нейронной сети червя *C.Elegans*. Каждая строка/столбец представляет один из 277 нейронов мозга этого червя. Матрица связности  $A$  имеет значение  $a_{ij} = 1$ , если нейрон  $i$  общается с нейроном  $j$  через синапс, и  $a_{ij} = 0$  в противном случае. Матрица связности не является симметричной, что означает, что собственные значения не могут быть действительными. Поэтому мы вычисляем симметризованную версию матрицы связности как  $A_{sym} := A + A^T$ . Эта новая матрица  $A_{sym}$  показана на рис. 4.5(a) и имеет не-нулевое значение  $a_{ij}$  тогда и только тогда, когда два нейрона соединены (белые пиксели), независимо от направления соединения. На рис. 4.5(b) мы показываем соответствующий собственный спектр  $A_{sym}$ . По горизонтальной оси отложены порядковые номера собственных значений, отсортированные по убыванию. Вертикальная ось показывает соответствующее собственное значение. S-образная форма этого собственного спектра типична для многих биологических нейронных сетей. Поиск основных механизмов, отвечающих за это, — тема актуальных исследований в нейробиологии.



**Рис. 4.5.** Нейронная сеть *Caenorhabditis elegans* (Kaiser and Hilgetag, 2006):  
(a) симметричная матрица связности; (b) собственный спектр

**Теорема 4.12.** Собственные векторы  $\mathbf{x}_1, \dots, \mathbf{x}_n$  матрицы  $\mathbf{A} \in \mathbb{R}^{n \times n}$  с  $n$  различными собственными значениями  $\lambda_1, \dots, \lambda_n$  линейно независимы.

Эта теорема утверждает, что собственные векторы матрицы с  $n$  различными собственными значениями образуют базис  $\mathbb{R}^n$ .

**Определение 4.13.** Квадратная матрица  $\mathbf{A} \in \mathbb{R}^{n \times n}$  является *дефектной*, если она имеет менее  $n$  линейно независимых собственных векторов. Дефектная матрица  $\mathbf{A} \in \mathbb{R}^{n \times n}$  не обязательно требует  $n$  различных собственных значений, но требует, чтобы собственные векторы составляли базис  $\mathbb{R}^n$ . Из рассмотрения собственных подпространств дефектной матрицы следует, что сумма размерностей собственных подпространств меньше  $n$ . В частности, дефектная матрица имеет по крайней мере одно собственное значение  $\lambda_i$  с алгебраической кратностью  $m > 1$  и геометрической кратностью меньше  $m$ .

**ПРИМЕЧАНИЕ** Дефектная матрица не может иметь  $n$  различных собственных значений, так как разные собственные значения имеют линейно независимые собственные векторы (теорема 4.12). ◆

**Теорема 4.14.** Для матрицы  $\mathbf{A} \in \mathbb{R}^{m \times n}$  мы всегда можем получить симметричную положительно полуопределенную матрицу  $\mathbf{S} \in \mathbb{R}^{n \times n}$ , задав

$$\mathbf{S} := \mathbf{A}^\top \mathbf{A}. \quad (4.36)$$

**ПРИМЕЧАНИЕ** Если  $\text{rk}(\mathbf{A}) = n$ , то  $\mathbf{S} := \mathbf{A}^\top \mathbf{A}$  симметрично, положительно определено. ◆

Понимание того, почему выполняется теорема 4.14, помогает понять, как мы можем использовать симметризованные матрицы: для симметрии требуется, чтобы  $S = S^T$ , и, вставив (4.36), мы получаем  $S = A^T A = A^T (A^T)^T = (A^T A)^T = S^T$ . Более того, положительная полуопределенность (раздел 3.2.3) требует, чтобы  $x^T S x \geq 0$ , и, подставляя (4.36), получаем  $x^T S x = x^T A^T A x = (x^T A^T)(A x) \geq (A x)^T (A x) \geq 0$ , потому что скалярное произведение вычисляет сумму квадратов (которые сами по себе неотрицательны).

**Теорема 4.15 (спектральная теорема).** *Если  $A \in \mathbb{R}^{n \times n}$  симметрично, существует ортонормированный базис соответствующего векторного пространства  $V$ , состоящий из собственных векторов  $A$ , и каждое собственное значение вещественно.*

Прямым следствием спектральной теоремы является то, что существует собственное разложение симметричной матрицы  $A$  (с действительными собственными значениями), и что мы можем найти ОНБ собственных векторов, так что  $A = P D P^T$ , где  $D$  диагональ, а столбцы  $P$  содержат собственные векторы.

### Пример 4.8

Рассмотрим матрицу

$$A = \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & 2 \\ 2 & 2 & 3 \end{bmatrix}. \quad (4.37)$$

Характеристический многочлен  $A$  равен

$$p_A(\lambda) = -(\lambda - 1)^2(\lambda - 7), \quad (4.38)$$

так что мы получаем собственные значения  $\lambda_1 = 1$  и  $\lambda_2 = 7$ , где  $\lambda_1$  — повторяющееся собственное значение. Следуя нашей стандартной процедуре вычисления собственных векторов, мы получаем собственные подпространства

$$E_1 = \text{span} \left[ \underbrace{\begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}}_{=: x_1}, \underbrace{\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}}_{=: x_2} \right], \quad E_7 = \text{span} \left[ \underbrace{\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}}_{=: x_3} \right]. \quad (4.39)$$

Мы видим, что  $x_3$  ортогонален как  $x_1$ , так и  $x_2$ . Однако поскольку  $x_1^T x_2 = 1 \neq 0$ , они не ортогональны. Спектральная теорема (теорема 4.15) утверждает, что существует ортогональный базис, но тот, который у нас есть, не ортогонален. Однако мы можем построить его. Чтобы построить такой базис,

мы используем тот факт, что  $\mathbf{x}_1, \mathbf{x}_2$  — собственные векторы, связанные с одним и тем же собственным значением  $\lambda$ . Следовательно, для любых  $\alpha, \beta \in \mathbb{R}$  выполняется

$$\mathbf{A}(\alpha\mathbf{x}_1 + \beta\mathbf{x}_2) = \mathbf{A}\mathbf{x}_1\alpha + \mathbf{A}\mathbf{x}_2\beta = \lambda(\alpha\mathbf{x}_1 + \beta\mathbf{x}_2), \quad (4.40)$$

то есть любая линейная комбинация  $\mathbf{x}_1$  и  $\mathbf{x}_2$  также является собственным вектором  $\mathbf{A}$ , связанным с  $\lambda$ . Алгоритм Грама — Шмидта (раздел 3.8.3) — это метод итеративного построения ортогонального/ортонормированного базиса из набора базисных векторов с использованием таких линейных комбинаций. Следовательно, даже если  $\mathbf{x}_1$  и  $\mathbf{x}_2$  не ортогональны, мы можем применить алгоритм Грама — Шмидта и найти собственные векторы, связанные с  $\lambda_1 = 1$ , которые ортогональны друг другу (и  $\mathbf{x}_3$ ). В нашем примере мы получим

$$\mathbf{x}'_1 = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{x}'_2 = \frac{1}{2} \begin{bmatrix} -1 \\ -1 \\ 2 \end{bmatrix}, \quad (4.41)$$

которые ортогональны друг другу, ортогональны  $\mathbf{x}_3$  и собственным векторам матрицы  $\mathbf{A}$ , связанным с  $\lambda_1 = 1$ .

Прежде чем мы закончим рассмотрение собственных значений и собственных векторов, полезно связать эти характеристики матрицы с понятиями детерминанта и следа.

**Теорема 4.16.** Детерминант матрицы  $\mathbf{A} \in \mathbb{R}^{n \times n}$  — это произведение ее собственных значений, то есть

$$\det(\mathbf{A}) = \prod_{i=1}^n \lambda_i, \quad (4.42)$$

где  $\lambda_i$  — (возможно, повторяющиеся) собственные значения  $\mathbf{A}$ .

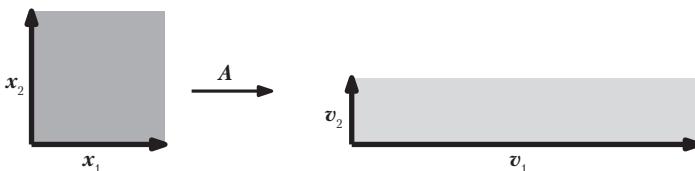
**Теорема 4.17.** След матрицы  $\mathbf{A} \in \mathbb{R}^{n \times n}$  — это сумма ее собственных значений, то есть

$$\text{tr}(\mathbf{A}) = \sum_{i=1}^n \lambda_i, \quad (4.43)$$

где  $\lambda_i$  — (возможно, повторяющиеся) собственные значения  $\mathbf{A}$ .

Дадим наглядную интерпретацию этим двум теоремам. Рассмотрим матрицу  $\mathbf{A} \in \mathbb{R}^{2 \times 2}$ , которая обладает двумя линейно независимыми собственными векто-

рами  $\mathbf{x}_1, \mathbf{x}_2$ . В этом примере мы предполагаем, что  $(\mathbf{x}_1, \mathbf{x}_2)$  являются ОНБ  $\mathbb{R}^2$ , так что они ортогональны и площадь квадрата, который они охватывают, равна 1 см (рис. 4.6). Из раздела 4.1 мы знаем, что детерминант вычисляет изменение площади единичного квадрата при преобразовании  $\mathbf{A}$ . В этом примере мы можем вычислить изменение площади явно: отображение собственных векторов с помощью  $\mathbf{A}$  дает нам векторы  $\mathbf{v}_1 = \mathbf{Ax}_1 = \lambda_1 \mathbf{x}_1$  и  $\mathbf{v}_2 = \mathbf{Ax}_2 = \lambda_2 \mathbf{x}_2$ , то есть новые векторы  $\mathbf{v}_i$  являются масштабированными версиями собственных векторов  $\mathbf{x}_i$ , а коэффициенты масштабирования — соответствующие собственные значения  $\lambda_i$ .  $\mathbf{v}_1, \mathbf{v}_2$  по-прежнему ортогональны, а площадь прямоугольника, который они охватывают, равна  $|\lambda_1 \lambda_2|$ . Учитывая, что  $x_1, x_2$  (в нашем примере) ортонормированы, мы можем напрямую вычислить длину периметра единичного квадрата как  $2(1 + 1)$ . Отображение собственных векторов с помощью  $\mathbf{A}$  создает прямоугольник с периметром  $2(|\lambda_1| + |\lambda_2|)$ . Следовательно, сумма абсолютных значений собственных значений говорит нам, как изменяется периметр единичного квадрата под действием матрицы преобразования  $\mathbf{A}$ .



**Рис. 4.6.** Геометрическая интерпретация собственных значений. Собственные векторы матрицы  $\mathbf{A}$  растягиваются на соответствующие собственные значения. Площадь единичного квадрата изменится на  $|\lambda_1 \lambda_2|$ , длина окружности изменится в 2 раза ( $|\lambda_1| + |\lambda_2|$ )

#### Пример 4.9 (Google's PageRank — веб-страницы как собственные векторы)

Google использует собственный вектор, соответствующий максимальному собственному значению матрицы  $\mathbf{A}$ , чтобы определить ранг страницы для поиска. Идея алгоритма PageRank, разработанного в Стэнфордском университете Ларри Пейджем и Сергеем Брином в 1996 году, заключалась в том, что важность любой веб-страницы можно приблизительно оценить по важности страниц, которые ссылаются на нее. Для этого они записывают все веб-сайты в виде огромного ориентированного графа, который показывает, какие страницы на какие ссылаются. PageRank вычисляет вес (важность)  $x_i \geq 0$  веб-сайта  $a_i$  путем подсчета количества страниц, указывающих на  $a_i$ . Более того, PageRank учитывает важность веб-сайтов, которые ссылаются на  $a_i$ . Затем навигационное поведение пользователя

моделируется с помощью матрицы переходов  $A$  этого графика, которая сообщает нам, с какой вероятностью (кликом) кто-то окажется на другом веб-сайте. Матрица  $A$  обладает тем свойством, что для любого начального вектора ранга/важности  $x$  веб-сайта последовательность  $x, Ax, A^2x, \dots$  сходится к вектору  $x^*$ . Этот вектор называется PageRank и удовлетворяет  $Ax^* = x^*$ , то есть это собственный вектор  $A$  (с соответствующим собственным значением 1). После нормализации  $x^*$ , такой что  $\|x^*\| = 1$ , мы можем интерпретировать элементы как вероятности. Более подробную информацию и различные точки зрения на PageRank можно найти в исходном техническом отчете (Page et al., 1999).

### 4.3. РАЗЛОЖЕНИЕ ХОЛЕЦКОГО

Есть много способов факторизовать специальные типы матриц, с которыми мы часто сталкиваемся в машинном обучении. В положительных действительных числах есть операция извлечения квадратного корня, которая дает разложение числа на идентичные компоненты, например  $9 = 3 \cdot 3$ . При работе с матрицами нужно тщательно следить за тем, чтобы операция квадратного корня вычислялась для положительных величин. Для симметричных положительно определенных матриц (см. раздел 3.2.3) мы можем выбирать из нескольких эквивалентных операций извлечения квадратного корня. Разложение Холецкого / факторизация Холецкого выполняет операцию над симметричными положительно определенными матрицами, эквивалентную квадратному корню.

**Теорема 4.18 (разложение Холецкого).** Симметричная положительно определенная матрица  $A$  может быть разложена на множители в произведение  $A = LL^T$ , где  $L$  – нижнетреугольная матрица с положительными диагональными элементами:

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ l_{n1} & \cdots & l_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & \cdots & l_{n1} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & l_{nn} \end{bmatrix}. \quad (4.44)$$

$L$  называется фактором Холецкого для  $A$ , и  $L$  единственна.

#### Пример 4.10 (факторизация Холецкого)

Рассмотрим симметричную положительно определенную матрицу  $A \in \mathbb{R}^{3 \times 3}$ . Нас интересует нахождение ее факторизации Холецкого  $A = LL^T$ , то есть

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \mathbf{LL}^T = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}. \quad (4.45)$$

Умножение правой части дает

$$\mathbf{A} = \begin{bmatrix} l_{11}^2 & l_{21}l_{11} & l_{31}l_{11} \\ l_{21}l_{11} & l_{21}^2 + l_{22}^2 & l_{31}l_{21} + l_{32}l_{22} \\ l_{31}l_{11} & l_{31}l_{21} + l_{32}l_{22} & l_{31}^2 + l_{32}^2 + l_{33}^2 \end{bmatrix}. \quad (4.46)$$

Сравнение левой части (4.45) и правой части (4.46) показывает, что в диагональных элементах  $l_{ii}$  имеется простая закономерность:

$$l_{11} = \sqrt{a_{11}}, \quad l_{22} = \sqrt{a_{22} - l_{21}^2}, \quad l_{33} = \sqrt{a_{33} - (l_{31}^2 + l_{32}^2)}. \quad (4.47)$$

Аналогично для элементов ниже диагонали ( $l_{ij}$ , где  $i > j$ ) также существует повторяющаяся закономерность:

$$l_{21} = \frac{1}{l_{11}} a_{21}, \quad l_{31} = \frac{1}{l_{11}} a_{31}, \quad l_{32} = \frac{1}{l_{22}} (a_{32} - l_{31}l_{21}). \quad (4.48)$$

Таким образом, мы построили разложение Холецкого для любой симметричной положительно определенной матрицы  $3 \times 3$ . Ключевая реализация состоит в том, что мы можем вычислить в обратном порядке, какими должны быть компоненты  $l_{ij}$  для  $\mathbf{L}$ , учитывая значения  $a_{ij}$  для  $\mathbf{A}$  и ранее вычисленные значения  $l_{ij}$ .

Разложение Холецкого — важный инструмент для выполнения вычислений, лежащих в основе машинного обучения. Здесь требуется часто иметь дело с симметричными положительно определенными матрицами. Например, ковариационная матрица многомерной гауссовой переменной (раздел 6.5) симметрична, положительно определена. Разложение Холецкого этой ковариационной матрицы позволяет нам генерировать выборки из распределения Гаусса. Она также позволяет нам выполнять линейное преобразование случайных величин, которое активно используется при вычислении градиентов в глубоких стохастических моделях, таких как вариационный автокодировщик (Jimenez Rezende et al., 2014; Kingma and Welling, 2014). Разложение Холецкого также позволяет очень эффективно вычислять детерминанты. Учитывая разложение Холецкого  $\mathbf{A} = \mathbf{LL}^T$ , мы знаем, что  $\det(\mathbf{A}) = \det(\mathbf{L}) \det(\mathbf{L}) = \det(\mathbf{L}^T)^2$ . Поскольку  $\mathbf{L}$  — треугольная матрица, детерминант — это просто произведение ее диагональных элементов, так что  $\det(\mathbf{A}) = \prod_i l_{ii}^2$ . Таким образом, многие пакеты для алгебраи-

ческих вычислений используют разложение Холецкого, чтобы сделать вычисления более эффективными.

## 4.4. СОБСТВЕННОЕ РАЗЛОЖЕНИЕ И ДИАГОНАЛИЗАЦИЯ

*Диагональная матрица* — это матрица, все недиагональные элементы которой равны нулю. То есть она имеет вид

$$\mathbf{D} = \begin{bmatrix} c_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & c_n \end{bmatrix}. \quad (4.49)$$

Такая форма матриц позволяет быстро вычислять детерминанты, степени и обратные величины. Детерминант — это произведение диагональных элементов матрицы, степень матрицы  $\mathbf{D}^k$  задается каждым диагональным элементом, возведенным в степень  $k$ , а обратная величина  $\mathbf{D}^{-1}$  — это величина, обратная произведению диагональных элементов матрицы, если все они не равны нулю.

В этом разделе мы обсудим, как преобразовать матрицы в диагональную форму. Это важное применение изменения базиса, которое мы обсуждали в разделе 2.7.2, и собственных значений из раздела 4.2.

Напомним, что две матрицы  $\mathbf{A}, \mathbf{D}$  подобны (определение 2.22), если существует обратимая матрица  $\mathbf{P}$ , такая что  $\mathbf{D} = \mathbf{P}^{-1}\mathbf{A}\mathbf{P}$ . Более конкретно, мы рассмотрим матрицы  $\mathbf{A}$ , которые похожи на диагональные матрицы  $\mathbf{D}$ , которые содержат собственные значения матрицы  $\mathbf{A}$  на диагонали.

**Определение 4.19 (диагонализуемость).** Матрица  $\mathbf{A} \in \mathbb{R}^{n \times n}$  диагонализуема, если она подобна диагональной матрице, то есть если существует обратимая матрица  $\mathbf{P} \in \mathbb{R}^{n \times n}$ , такая что  $\mathbf{D} = \mathbf{P}^{-1}\mathbf{A}\mathbf{P}$ .

Далее мы увидим, что диагонализация матрицы  $\mathbf{A} \in \mathbb{R}^{n \times n}$  — это способ выразить то же линейное отображение, но в другом базисе (раздел 2.6.1), который окажется базисом, состоящим из собственных векторов группы  $\mathbf{A}$ . Пусть  $\mathbf{A} \in \mathbb{R}^{n \times n}$ , пусть  $\lambda_1, \dots, \lambda_n$  — набор скаляров, и пусть  $\mathbf{p}_1, \dots, \mathbf{p}_n$  — множество векторов в  $\mathbb{R}^n$ . Определим  $\mathbf{P} := [\mathbf{p}_1, \dots, \mathbf{p}_n]$ , и пусть  $\mathbf{D} \in \mathbb{R}^{n \times n}$  — диагональная матрица с диагональными элементами  $\lambda_1, \dots, \lambda_n$ . Тогда мы можем показать, что

$$\mathbf{AP} = \mathbf{PD} \quad (4.50)$$

тогда и только тогда, когда  $\lambda_1, \dots, \lambda_n$  — собственные значения  $\mathbf{A}$  и  $\mathbf{p}_1, \dots, \mathbf{p}_n$  — соответствующие собственные векторы матрицы  $\mathbf{A}$ .

Мы видим, что это утверждение верно, потому что

$$\mathbf{A}\mathbf{P} = \mathbf{A}[\mathbf{p}_1, \dots, \mathbf{p}_n] = [\mathbf{A}\mathbf{p}_1, \dots, \mathbf{A}\mathbf{p}_n], \quad (4.51)$$

$$\mathbf{P}\mathbf{D} = [\mathbf{p}_1, \dots, \mathbf{p}_n] \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix} = [\lambda_1 \mathbf{p}_1, \dots, \lambda_n \mathbf{p}_n]. \quad (4.52)$$

Таким образом, из (4.50) следует, что

$$\begin{aligned} \mathbf{A}\mathbf{p}_1 &= \lambda_1 \mathbf{p}_1 \\ &\vdots \end{aligned} \quad (4.53)$$

$$\mathbf{A}\mathbf{p}_n = \lambda_n \mathbf{p}_n. \quad (4.54)$$

Следовательно, столбцы матрицы  $\mathbf{P}$  должны быть собственными векторами матрицы  $\mathbf{A}$ .

Наше определение диагонализации требует, чтобы  $\mathbf{P} \in \mathbb{R}^{n \times n}$  был обратимым, то есть  $\mathbf{P}$  имел полный ранг (теорема 4.3). Для этого необходимо иметь  $n$  линейно независимых собственных векторов  $\mathbf{p}_1, \dots, \mathbf{p}_n$ , то есть  $\mathbf{p}_i$  образуют базис  $\mathbb{R}^n$ .

**Теорема 4.20 (собственное разложение).** Квадратную матрицу  $\mathbf{A} \in \mathbb{R}^{n \times n}$  можно разложить на

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}, \quad (4.55)$$

где  $\mathbf{P} \in \mathbb{R}^{n \times n}$ , а  $\mathbf{D}$  – диагональная матрица, диагональные элементы которой являются собственными значениями матрицы  $\mathbf{A}$ , тогда и только тогда, когда собственные векторы матрицы  $\mathbf{A}$  образуют базис  $\mathbb{R}^n$ .

Из теоремы 4.20 следует, что только недефектные матрицы могут быть диагонализованы и что столбцы  $\mathbf{P}$  – это  $n$  собственных векторов  $\mathbf{A}$ . Для симметричных матриц мы можем получить еще более интересные результаты при разложении по собственным значениям.

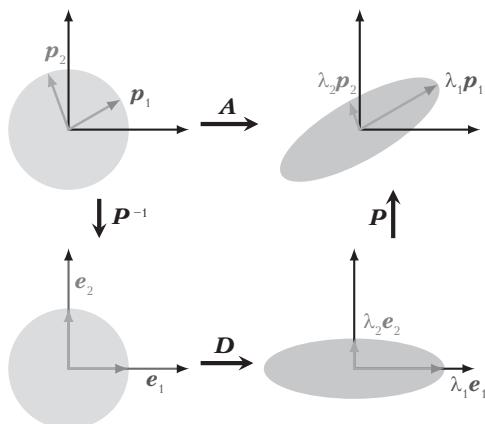
**Теорема 4.21.** Симметричную матрицу  $\mathbf{S} \in \mathbb{R}^{n \times n}$  всегда можно диагонализовать.

Теорема 4.21 непосредственно следует из спектральной теоремы 4.15. Более того, спектральная теорема утверждает, что мы можем найти ОНБ собственных векторов  $\mathbb{R}^n$ . Это делает  $\mathbf{P}$  ортогональной матрицей, так что  $\mathbf{D} = \mathbf{P}^T \mathbf{A} \mathbf{P}$ .

**ПРИМЕЧАНИЕ** Нормальная форма матрицы Жордана предлагает разложение, которое работает для дефектных матриц (Lang, 1987), но выходит за рамки этой книги. ◆

#### 4.4.1. Геометрическая интуиция для собственного разложения

Мы можем интерпретировать собственное разложение матрицы следующим образом (см. также рис. 4.7): пусть  $A$  будет матрицей преобразования линейного отображения относительно стандартного базиса.  $P^{-1}$  выполняет замену стандартного базиса на собственный. Это идентифицирует собственные векторы  $\mathbf{p}_i$  (светлые стрелки на рис. 4.7) на стандартных базисных векторах  $\mathbf{e}_i$ . Затем диагональ  $D$  масштабирует векторы вдоль этих осей на собственные значения  $\lambda_i$ . Наконец,  $P$  преобразует эти масштабированные векторы обратно в стандартные/канонические координаты, в результате чего получается  $\lambda_i \mathbf{p}_i$ .



**Рис. 4.7.** Наглядная иллюстрация собственного разложения как последовательных преобразований. От левого верхнего к левому нижнему изображению:  $P^{-1}$  выполняет изменение базиса (здесь нарисовано в  $\mathbb{R}^2$  и изображено как операция, подобная вращению), отображая собственные векторы в стандартный базис. От левого нижнего к правому нижнему:  $D$  выполняет масштабирование по перенаправленным ортогональным собственным векторам, изображенным здесь кругом, растянутым до эллипса. От правого нижнего к правому верхнему:  $P$  отменяет базовое изменение (изображенное как обратное вращение) и восстанавливает исходную систему координат

##### Пример 4.11 (собственное разложение)

Вычислим собственное разложение  $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$ .

**Шаг 1: вычислим собственные значения и собственные векторы.** Характеристический многочлен  $A$  равен

$$\det(A - \lambda I) = \det \begin{pmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{pmatrix} = \quad (4.56a)$$

$$= (2 - \lambda)^2 - 1 = \lambda^2 - 4\lambda + 3 = (\lambda - 3)(\lambda - 1). \quad (4.56b)$$

Следовательно, собственные значения матрицы  $A$  равны  $\lambda_1 = 1$  и  $\lambda_2 = 3$  (корни характеристического полинома), а соответствующие (нормированные) собственные векторы получаются с помощью

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} p_1 = 1 p_1, \quad \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} p_2 = 3 p_2. \quad (4.57)$$

Это дает

$$p_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad p_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \quad (4.58)$$

**Шаг 2: проверим наличие.** Собственные векторы  $p_1, p_2$  составляют базис  $\mathbb{R}^2$ . Следовательно,  $A$  можно диагонализовать.

**Шаг 3: построим матрицу  $P$  для диагонализации  $A$ .** Соберем собственные векторы матрицы  $A$  в  $P$ , так чтобы

$$P = [p_1, p_2] = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}. \quad (4.59)$$

Затем получим

$$P^{-1}AP = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} = D. \quad (4.60)$$

Аналогичным образом получаем (с учетом того, что  $P^{-1} = P^T$ , поскольку собственные векторы  $p_1$  и  $p_2$  в этом примере образуют ОНБ):

$$\underbrace{\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}}_A = \underbrace{\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}}_P \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}}_D \underbrace{\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}}_{P^T}. \quad (4.61)$$

- Диагональные матрицы  $D$  можно эффективно возвести в степень. Следовательно, мы можем найти степень матрицы для матрицы  $A \in \mathbb{R}^{n \times n}$  с помощью разложения по собственным значениям (если оно существует), так что

$$A^k = (PDP^{-1})^k = P D^k P^{-1}. \quad (4.62)$$

Вычисление  $\mathbf{D}^k$  эффективно, потому что мы применяем эту операцию индивидуально к любому диагональному элементу.

- Предположим, что собственное разложение  $\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$  существует. Тогда

$$\det(\mathbf{A}) = \det(\mathbf{P}\mathbf{D}\mathbf{P}^{-1}) = \det(\mathbf{P}) \det(\mathbf{D}) \det(\mathbf{P}^{-1}) = \quad (4.63a)$$

$$= \det(\mathbf{D}) = \prod_i d_{ii} \quad (4.63b)$$

позволяет эффективно вычислять детерминант  $\mathbf{A}$ .

Для разложения по собственным значениям требуются квадратные матрицы. Было бы полезно выполнить разложение матрицы общего вида. В следующем разделе мы представим метод разложения матриц общего вида — разложение по сингулярным числам.

## 4.5. РАЗЛОЖЕНИЕ ПО СИНГУЛЯРНЫМ ЗНАЧЕНИЯМ

Сингулярное разложение (singular value decomposition, SVD) матрицы — это центральный метод разложения матриц в линейной алгебре. Его называют «фундаментальной теоремой линейной алгебры» (Strang, 1993), потому что его можно применить ко всем матрицам, а не только к квадратным, и оно существует всегда. Более того, как мы исследуем ниже, SVD матрицы  $\mathbf{A}$ , которая представляет линейное отображение  $\Phi : V \rightarrow W$ , количественно определяет изменение базовой геометрии этих двух векторных пространств. Мы рекомендуем работы Калмана (1996) и Роя и Банерджи (2014) для более глубокого обзора математики SVD.

**Теорема 4.22 (теорема SVD).** Пусть  $\mathbf{A}^{m \times n}$  — прямоугольная матрица ранга  $r \in [0, \min(m, n)]$ . SVD оператора  $\mathbf{A}$  представляет собой разложение вида

$$m \begin{array}{|c|} \hline \mathbf{A} \\ \hline \end{array} = m \begin{array}{|c|} \hline \mathbf{U} \\ \hline \end{array} m \begin{array}{|c|} \hline \Sigma \\ \hline \end{array} \boxed{V^T} u \quad (4.64)$$

с ортогональной матрицей  $\mathbf{U} \in \mathbb{R}^{m \times m}$  с векторами-столбцами  $\mathbf{u}_i, i = 1, \dots, m$ , и ортогональной матрицей  $\mathbf{V} \in \mathbb{R}^{n \times n}$  с векторами-столбцами  $\mathbf{v}_j, j = 1, \dots, n$ . Более того,  $\Sigma$  является матрицей размера  $m \times n$  с  $\Sigma_{ii} = \sigma_i \geq 0$  и  $\Sigma_{ij} = 0, i \neq j$ .

Диагональные элементы  $\sigma_i, i = 1, \dots, r$ , вектора  $\Sigma$  называются *сингулярными значениями*,  $\mathbf{u}_i$  — *левосингулярными векторами*, а  $\mathbf{v}_j$  — *правосингулярными векторами*. По соглашению, особые значения упорядочены, то есть  $\sigma_1 \geq \sigma_2 \geq \sigma_r \geq 0$ .

*Матрица сингулярных значений*  $\Sigma$  уникальна, но требует некоторого внимания. Заметим, что  $\Sigma \in \mathbb{R}^{m \times n}$  прямоугольна. В частности,  $\Sigma$  имеет тот же размер, что и  $A$ . Это означает, что  $\Sigma$  имеет диагональную подматрицу, которая содержит сингулярные значения и требует дополнительного заполнения нулями. В частности, если  $m > n$ , то матрица  $\Sigma$  имеет диагональную структуру до строки  $n$  и затем состоит из  $\mathbf{0}^T$  векторов-строк от  $n + 1$  до  $m$  ниже, так что

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_n \\ 0 & \cdots & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix}. \quad (4.65)$$

Если  $m < n$ , матрица  $\Sigma$  имеет диагональную структуру до столбца  $m$  и столбцов, состоящих из 0, от  $m + 1$  до  $n$ :

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \ddots & 0 & 0 & & 0 \\ 0 & 0 & \sigma_n & 0 & \cdots & 0 \end{bmatrix}. \quad (4.66)$$

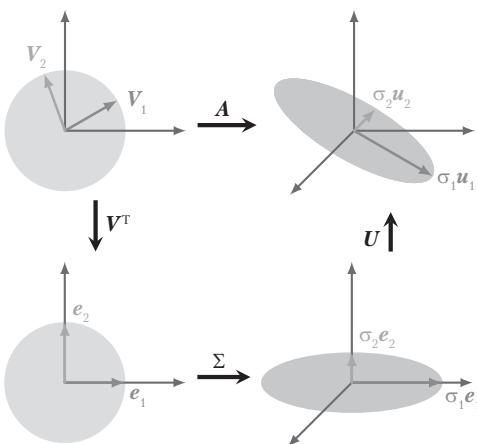
**ПРИМЕЧАНИЕ** SVD существует для любой матрицы  $A \in \mathbb{R}^{m \times n}$ .



### 4.5.1. Геометрические интуиции для SVD

SVD предлагает геометрическую интуицию для описания матрицы преобразования  $A$ . Далее мы обсудим SVD как последовательные линейные преобразования, выполняемые на базисах. В примере 4.12 мы затем применим матрицы преобразования SVD к набору векторов в  $\mathbb{R}^2$ , что позволит нам более четко визуализировать эффект каждого преобразования.

SVD матрицы можно интерпретировать как разложение соответствующего линейного отображения (вспомните раздел 2.7.1)  $\Phi : \mathbb{R}^n \rightarrow \mathbb{R}^m$  на три операции (рис. 4.8). Интуиция SVD следует внешне схожей структуре с нашей интуицией собственного разложения (см. рис. 4.7): в общих чертах, SVD выполняет изменение базиса через  $V^T$ , за которым следует масштабирование и увеличение (или уменьшение) размерности через матрицу сингулярных значений  $\Sigma$ . Наконец, оно выполняет вторую смену базиса через  $U$ . SVD влечет за собой ряд важных деталей и оговорок, поэтому мы рассмотрим нашу интуицию более подробно.



**Рис. 4.8.** Иллюстрация к представлению SVD матрицы  $A \in \mathbb{R}^{3 \times 2}$  в виде последовательных преобразований. От верхнего левого изображения к нижнему левому:  $V^T$  выполняет базисное изменение в  $\mathbb{R}^2$ . От нижнего левого к нижнему правому:  $\Sigma$  масштабируется и отображается от  $\mathbb{R}^2$  до  $\mathbb{R}^3$ . Эллипс в правом нижнем углу находится в  $\mathbb{R}^3$ . Третье измерение ортогонально поверхности эллиптического диска. От нижнего правого к верхнему правому:  $U$  выполняет базисное изменение в  $\mathbb{R}^3$

Предположим, нам дана матрица преобразования линейного отображения  $\Phi : \mathbb{R}^n \rightarrow \mathbb{R}^m$  относительно стандартных базисов  $B$  и  $C$   $\mathbb{R}^n$  и  $\mathbb{R}^m$  соответственно<sup>1</sup>. Кроме того, предположим, что есть второй базис  $\tilde{B}$  в  $\mathbb{R}^n$  и  $\tilde{C}$  в  $\mathbb{R}^m$ . Тогда:

1. Матрица  $V$  выполняет замену базиса в области  $\mathbb{R}^n$  от  $\tilde{B}$  (представленного векторами  $v_1$  и  $v_2$  в верхнем левом углу рис. 4.8) на стандартный базис  $B$ .  $V^T = V^{-1}$  выполняет изменение базиса с  $B$  на  $\tilde{B}$ . Векторы теперь выровнены по каноническому основанию в левом нижнем углу рис. 4.8.
2. Изменив систему координат на  $\tilde{B}$ ,  $\Sigma$  масштабирует новые координаты сингулярными значениями  $\sigma_i$  (и добавляет или удаляет измерения), то есть  $\Sigma$  является матрицей преобразования  $\Phi$  относительно  $\tilde{B}$  и  $\tilde{C}$ , представленной векторами, растянутыми и лежащими в плоскости  $e_1$ - $e_2$ , которая теперь встроена в третье измерение в правом нижнем углу рис. 4.8.
3.  $U$  выполняет замену базиса в области  $\mathbb{R}^m$  с  $\tilde{C}$  на канонический базис  $\mathbb{R}^m$ , представленный поворотом векторов из плоскости  $e_1$ - $e_2$ . Это показано в правом верхнем углу рис. 4.8.

SVD выражает изменение базиса как в домене, так и в кодомене. Это контрастирует с собственным разложением, которое работает в том же векторном пространстве, где применяется то же изменение базиса, а затем отменяется. Что делает SVD особенным, так это то, что эти два разных базиса одновременно связаны матрицей сингулярных значений  $\Sigma$ .

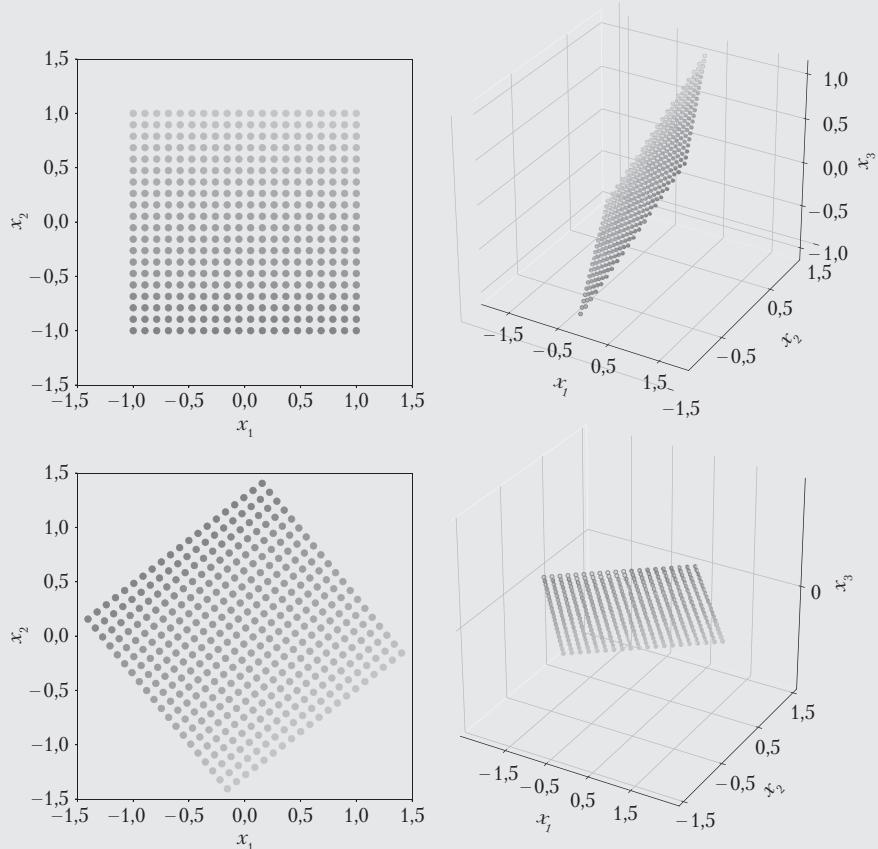
<sup>1</sup> Полезно освежить в памяти изменения базиса (раздел 2.7.2), ортогональные матрицы (определение 3.8) и ортонормированные базисы (раздел 3.5).

**Пример 4.12 (векторы и SVD)**

Рассмотрим отображение квадратной сетки векторов  $\mathcal{X} \in \mathbb{R}^2$ , которые помещаются в прямоугольник размером  $2 \times 2$  с центром в начале координат. Используя стандартный базис, сопоставим эти векторы с помощью

$$A = \begin{bmatrix} 1 & -0,8 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} = U \Sigma V^T = \quad (4.67a)$$

$$= \begin{bmatrix} -0,79 & 0 & -0,62 \\ 0,38 & -0,78 & -0,49 \\ -0,48 & -0,62 & 0,62 \end{bmatrix} \begin{bmatrix} 1,62 & 0 \\ 0 & 1,0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} -0,78 & 0,62 \\ -0,62 & -0,78 \end{bmatrix}. \quad (4.67b)$$



**Рис. 4.9.** SVD и отображение векторов (представленных точками). Части рисунка расположены в том же порядке против часовой стрелки, что и на рис. 4.8

Мы начинаем с набора векторов  $\mathcal{X}$  (светлые и темные точки; см. верхнее левое изображение на рис. 4.9), расположенных в виде сетки. Затем мы применяем  $\mathbf{V}^T \in \mathbb{R}^{2 \times 2}$ , который вращает  $\mathcal{X}$ . Повернутые векторы показаны на левом нижнем изображении рис. 4.9. Теперь мы отображаем эти векторы, используя матрицу сингулярных значений  $\Sigma$ , в область  $\mathbb{R}^3$  (см. нижнюю правую панель рис. 4.9). Обратите внимание, что все векторы лежат в плоскости  $x_1$ - $x_2$ . Третья координата всегда равна 0. Векторы в плоскости  $x_1$ - $x_2$  растянуты на сингулярные значения.

Прямое отображение векторов  $\mathcal{X}$  посредством  $\mathbf{A}$  в область  $\mathbb{R}^3$  равно преобразованию  $\mathcal{X}$  посредством  $\mathbf{U}\Sigma\mathbf{V}^T$ , где  $\mathbf{U}$  выполняет поворот внутри области  $\mathbb{R}^3$ , так что отображаемые векторы больше не ограничиваются плоскостью  $x_1$ - $x_2$ ; они все еще находятся в плоскости, как показано на правой верхней панели рис. 4.9.

### 4.5.2. Построение SVD

Далее мы обсудим, почему существует SVD, и подробно покажем, как его вычислить. SVD общей матрицы имеет некоторое сходство с собственным разложением квадратной матрицы.

**ПРИМЕЧАНИЕ** Сравните собственное разложение симметричной положительно определенной матрицы

$$\mathbf{S} = \mathbf{S}^T = \mathbf{P}\mathbf{D}\mathbf{P}^T \quad (4.68)$$

с соответствующим SVD

$$\mathbf{S} = \mathbf{U}\Sigma\mathbf{V}^T. \quad (4.69)$$

Если мы установим

$$\mathbf{U} = \mathbf{P} = \mathbf{V}, \quad \mathbf{D} = \Sigma, \quad (4.70)$$

мы видим, что SVD симметричных положительно определенных матриц является их собственным разложением. ♦

Далее мы исследуем, почему теорема 4.22 верна и как строится SVD. Вычисление SVD для  $\mathbf{A} \in \mathbb{R}^{m \times n}$  эквивалентно нахождению двух наборов ортонормированных базисов  $U = (\mathbf{u}_1, \dots, \mathbf{u}_m)$  и  $V = (\mathbf{v}_1, \dots, \mathbf{v}_n)$  области  $\mathbb{R}^m$  и области  $\mathbb{R}^n$  соответственно. По этим упорядоченным базисам построим матрицы  $\mathbf{U}$  и  $\mathbf{V}$ .

Наш план состоит в том, чтобы начать с построения ортонормированного набора правосингулярных векторов  $\mathbf{v}_1, \dots, \mathbf{v}_n \in \mathbb{R}^n$ . Затем мы строим ортонормированное множество левосингулярных векторов  $\mathbf{u}_1, \dots, \mathbf{u}_n \in \mathbb{R}^m$ . После этого мы свяжем эти два множества и потребуем, чтобы ортогональность  $\mathbf{v}_i$  сохранялась при преобразовании  $\mathbf{A}$ . Это важно, потому что мы знаем, что изображения  $\mathbf{Av}_i$  образуют набор ортогональных векторов. Затем мы нормализуем эти изображения с помощью скалярных множителей, которые окажутся сингулярными значениями.

Начнем с построения правосингулярных векторов. Спектральная теорема (теорема 4.15) говорит нам, что симметричная матрица обладает ОНБ собственных векторов, что также означает, что она может быть диагонализована. Более того, по теореме 4.14 мы всегда можем построить симметричную положительно полуопределенную матрицу  $\mathbf{A}^\top \mathbf{A} \in \mathbb{R}^{n \times n}$  из любой прямоугольной матрицы  $\mathbf{A} \in \mathbb{R}^{m \times n}$ . Таким образом, мы всегда можем диагонализовать  $\mathbf{A}^\top \mathbf{A}$  и получить

$$\mathbf{A}^\top \mathbf{A} = \mathbf{P} \mathbf{\Lambda} \mathbf{P}^\top = \mathbf{P} \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix} \mathbf{P}^\top, \quad (4.71)$$

где  $\mathbf{P}$  — ортогональная матрица, составленная из ортонормированного собственного базиса.  $\lambda_i \geq 0$  являются собственными значениями оператора  $\mathbf{A}^\top \mathbf{A}$ . Предположим, что SVD оператора  $\mathbf{A}$  существует, и подставим (4.64) в (4.71). Это дает

$$\mathbf{A}^\top \mathbf{A} = (\mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top)^\top (\mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top) = \mathbf{V} \mathbf{\Sigma}^\top \mathbf{U}^\top \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top, \quad (4.72)$$

где  $\mathbf{U}, \mathbf{V}$  — ортогональные матрицы. Следовательно, при  $\mathbf{U}^\top \mathbf{U} = I$  получаем

$$\mathbf{A}^\top \mathbf{A} = \mathbf{V} \mathbf{\Sigma}^\top \mathbf{\Sigma} \mathbf{V}^\top = \mathbf{V} \begin{bmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_n^2 \end{bmatrix} \mathbf{V}^\top. \quad (4.73)$$

Сравнивая теперь (4.71) и (4.73), отождествляем

$$\mathbf{V}^\top = \mathbf{P}^\top \quad (4.74)$$

$$\sigma_i^2 = \lambda_i. \quad (4.75)$$

Следовательно, собственные векторы матрицы  $\mathbf{A}^\top \mathbf{A}$ , составляющие  $\mathbf{P}$ , являются правыми сингулярными векторами  $\mathbf{V}$  матрицы  $\mathbf{A}$  (4.74). Собственные значения  $\mathbf{A}^\top \mathbf{A}$  — это квадраты сингулярных значений  $\Sigma$  (4.75).

Чтобы получить левосингулярные векторы  $\mathbf{U}$ , проделаем аналогичную процедуру. Начнем с вычисления SVD симметричной матрицы  $\mathbf{A}^T \mathbf{A} \in \mathbb{R}^{m \times m}$  (вместо предыдущей  $\mathbf{A}^T \mathbf{A} \in \mathbb{R}^{n \times n}$ ). SVD  $\mathbf{A}$  дает

$$\mathbf{A} \mathbf{A}^T = (\mathbf{U} \Sigma \mathbf{V}^T) (\mathbf{U} \Sigma \mathbf{V}^T)^T = \mathbf{U} \Sigma \mathbf{V}^T \mathbf{V} \Sigma^T \mathbf{U}^T = \quad (4.76a)$$

$$= \mathbf{U} \begin{bmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_m^2 \end{bmatrix} \mathbf{U}^T. \quad (4.76b)$$

Спектральная теорема говорит нам, что  $\mathbf{A} \mathbf{A}^T = \mathbf{S} \mathbf{D} \mathbf{S}^T$  можно диагонализовать, и мы можем найти ОНБ собственных векторов  $\mathbf{A} \mathbf{A}^T$ , которые собраны в  $\mathbf{S}$ . Ортонормированные собственные векторы  $\mathbf{A} \mathbf{A}^T$  являются левосингулярными векторами  $\mathbf{U}$  и образуют ортонормированный базис в кодомене SVD.

Остается вопрос о структуре матрицы  $\Sigma$ . Поскольку  $\mathbf{A} \mathbf{A}^T$  и  $\mathbf{A}^T \mathbf{A}$  имеют одинаковые ненулевые собственные значения (см. с. 144), ненулевые элементы  $\Sigma$ -матриц в SVD для обоих случаев должны быть одинаковыми.

Наконец, объединим вместе все, что мы уже сделали. У нас есть ортонормированный набор правосингулярных векторов в  $\mathbf{V}$ . Чтобы завершить построение SVD, мы соединяем их с ортонормированными векторами  $\mathbf{U}$ . Для достижения этой цели мы используем тот факт, что изображения  $\mathbf{v}_i$  под  $\mathbf{A}$  также должны быть ортогональными. Мы можем показать это, используя результаты из раздела 3.4. Мы требуем, чтобы скалярное произведение между  $\mathbf{A}\mathbf{v}_i$  и  $\mathbf{A}\mathbf{v}_j$  было равно  $\mathbf{0}$  для  $i \neq j$ . Для любых двух ортогональных собственных векторов  $\mathbf{v}_i, \mathbf{v}_j, i \neq j$ , справедливо

$$(\mathbf{A}\mathbf{v}_i)^T (\mathbf{A}\mathbf{v}_j) = \mathbf{v}_i^T (\mathbf{A}^T \mathbf{A}) \mathbf{v}_j = \mathbf{v}_i^T (\lambda_j \mathbf{v}_j) = \lambda_j \mathbf{v}_i^T \mathbf{v}_j = 0. \quad (4.77)$$

Для случая  $m \geq r$  справедливо, что  $\{\mathbf{A}\mathbf{v}_1, \dots, \mathbf{A}\mathbf{v}_r\}$  является базисом  $r$ -мерного подпространства в  $\mathbb{R}^m$ .

Для завершения построения SVD нам потребуются ортонормированные левосингулярные векторы: нормализуем образы правосингулярных векторов  $\mathbf{A}\mathbf{v}_i$  и получаем

$$\mathbf{u}_i := \frac{\mathbf{A}\mathbf{v}_i}{\|\mathbf{A}\mathbf{v}_i\|} = \frac{1}{\sqrt{\lambda_i}} \mathbf{A}\mathbf{v}_i = \frac{1}{\sigma_i} \mathbf{A}\mathbf{v}_i, \quad (4.78)$$

где последнее равенство было получено из (4.75) и (4.76b). Отсюда вытекает: собственные значения  $\mathbf{A} \mathbf{A}^T$  таковы, что  $\sigma_i^2 = \lambda_i$ . Следовательно, собственные векторы  $\mathbf{A}^T \mathbf{A}$ , которые, как мы знаем, являются правосингулярными векторами  $\mathbf{v}_i$ , и их нормализованные образы относительно  $\mathbf{A}$ , левосингулярные векторы  $\mathbf{u}_i$ ,

образуют два самосогласованных ОНБ, которые связаны посредством матрицы сингулярных значений  $\Sigma$ .

Перепишем (4.78), чтобы получить *сингулярное уравнение*

$$\mathbf{A}\mathbf{v}_i = \sigma_i \mathbf{u}_i, \quad i = 1, \dots, r. \quad (4.79)$$

Это уравнение очень похоже на уравнение для собственных значений (4.25), но векторы в левой и правой частях не совпадают.

При  $n > m$  (4.79) выполняется только для  $i \leq m$ , а (4.79) ничего не говорит о  $\mathbf{u}_i$  при  $i > m$ . Однако мы знаем по построению, что они ортонормированы. Наоборот, для  $m > n$  (4.79) выполняется только для  $i \leq n$ . Для  $i > n$  у нас  $\mathbf{A}\mathbf{v}_i = \mathbf{0}$ , и мы знаем, что  $\mathbf{v}_i$  образуют ортонормированное множество. Это означает, что SVD также предоставляет ортонормированный базис ядра (нулевое пространство)  $\mathbf{A}$ , набор векторов  $\mathbf{x}$  с  $\mathbf{A}\mathbf{x} = \mathbf{0}$  (см. раздел 2.7.3).

Более того, объединение  $\mathbf{v}_i$  как столбцов  $\mathbf{V}$  и  $\mathbf{u}_i$  как столбцов  $\mathbf{U}$  дает

$$\mathbf{A}\mathbf{V} = \mathbf{U}\Sigma, \quad (4.80)$$

где  $\Sigma$  имеет ту же размерность, что и  $\mathbf{A}$ , и диагональную структуру для строк  $1, \dots, r$ . Следовательно, умножение справа на  $\mathbf{V}^T$  дает  $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$ , которая является SVD  $\mathbf{A}$ .

### Пример 4.13 (расчет SVD)

Найдем сингулярное разложение

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix}. \quad (4.81)$$

SVD требует от нас вычисления правосингулярных векторов  $\mathbf{v}_j$ , сингулярных значений  $\sigma_k$  и левосингулярных векторов  $\mathbf{u}_i$ .

**Шаг 1: правосингулярные векторы как собственный базис  $\mathbf{A}^T\mathbf{A}$ .**

Начнем с вычисления

$$\mathbf{A}^T\mathbf{A} = \begin{bmatrix} 1 & -2 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 5 & -2 & 1 \\ -2 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}. \quad (4.82)$$

Мы вычисляем сингулярные значения и правосингулярные векторы  $\mathbf{v}_j$  с помощью разложения по собственным значениям матрицы  $\mathbf{A}^T\mathbf{A}$ , которое задается как

$$\mathbf{A}^T \mathbf{A} = \begin{bmatrix} \frac{5}{\sqrt{30}} & 0 & \frac{-1}{\sqrt{6}} \\ \frac{-2}{\sqrt{30}} & \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{30}} & \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{6}} \end{bmatrix} \begin{bmatrix} 6 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{5}{\sqrt{30}} & \frac{-2}{\sqrt{30}} & \frac{1}{\sqrt{30}} \\ 0 & \frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \\ \frac{-1}{\sqrt{6}} & \frac{-2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \end{bmatrix} = \mathbf{P} \mathbf{D} \mathbf{P}^T, \quad (4.83)$$

и мы получаем правосингулярные векторы как столбцы  $\mathbf{P}$ , так что

$$\mathbf{V} = \mathbf{P} = \begin{bmatrix} \frac{5}{\sqrt{30}} & 0 & \frac{-1}{\sqrt{6}} \\ \frac{-2}{\sqrt{30}} & \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{30}} & \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{6}} \end{bmatrix}. \quad (4.84)$$

### Шаг 2: матрица сингулярных чисел.

Так как сингулярные значения  $\sigma_i$  являются квадратными корнями из собственных значений оператора  $\mathbf{A}^T \mathbf{A}$ , мы получаем их прямо из  $\mathbf{D}$ . Поскольку  $\text{rk}(\mathbf{A}) = 2$ , есть только два ненулевых сингулярных значения:  $\sigma_1 = \sqrt{6}$  и  $\sigma_2 = 1$ . Матрица сингулярных значений должна быть того же размера, что и  $\mathbf{A}$ , и мы получаем

$$\Sigma = \begin{bmatrix} \sqrt{6} & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \quad (4.85)$$

### Шаг 3: левосингулярные векторы как нормализованное изображение правосингулярных векторов.

Находим левосингулярные векторы, вычисляя образ правосингулярных векторов под  $\mathbf{A}$  и нормализуя их, разделив на соответствующее сингулярное значение. Получаем:

$$\mathbf{u}_1 = \frac{1}{\sigma_1} \mathbf{A} \mathbf{v}_1 = \frac{1}{\sqrt{6}} \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{5}{\sqrt{30}} \\ \frac{-2}{\sqrt{30}} \\ \frac{1}{\sqrt{30}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{5}} \\ \frac{-2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \end{bmatrix}, \quad (4.86)$$

$$\mathbf{u}_2 = \frac{1}{\sigma_2} \mathbf{A} \mathbf{v}_2 = \frac{1}{1} \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ \frac{1}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \end{bmatrix} = \begin{bmatrix} \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \end{bmatrix}, \quad (4.87)$$

$$\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2] = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & 2 \\ -2 & 1 \end{bmatrix}. \quad (4.88)$$

Обратите внимание, что на компьютере подход, проиллюстрированный здесь, плохо реализуется, и SVD  $\mathbf{A}$  обычно вычисляется без использования разложения на собственные значения  $\mathbf{A}^T \mathbf{A}$ .

### 4.5.3. Разложение на собственные значения и разложение на сингулярные значения

Рассмотрим собственное разложение  $\mathbf{A} = \mathbf{P} \mathbf{D} \mathbf{P}^{-1}$  и SVD  $\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T$  и вспомним основные положения предыдущих разделов.

- SVD всегда существует для любой матрицы  $\mathbb{R}^{m \times n}$ . Собственное разложение определено только для квадратных матриц  $\mathbb{R}^{n \times n}$  и существует только в том случае, если мы можем найти базис из собственных векторов  $\mathbb{R}^n$ .
- Векторы в матрице собственного разложения  $\mathbf{P}$  не обязательно ортогональны, то есть смена базиса не является простым поворотом и масштабированием. А вот векторы в матрицах  $\mathbf{U}$  и  $\mathbf{V}$  в SVD ортонормированы, поэтому они соответствуют вращению.
- И собственное разложение, и SVD представляют собой композиции трех линейных отображений:
  1. Изменение базиса в домене.
  2. Независимое масштабирование каждого нового базисного вектора и отображение от домена к домену.
  3. Изменение базиса в кодомене.

Ключевое различие между собственным разложением и SVD состоит в том, что в SVD домен и кодомен могут быть векторными пространствами разных размеров.

- В SVD лево- и правосингулярные векторные матрицы  $\mathbf{U}$  и  $\mathbf{V}$  обычно не инверсны друг другу (они выполняют замену базиса в разных векторных про-

странствах). В собственном разложении матрицы замены базиса  $\mathbf{P}$  и  $\mathbf{P}^{-1}$  обратны друг другу.

- В SVD все элементы диагональной матрицы  $\Sigma$  являются действительными и неотрицательными, что обычно неверно для диагональной матрицы в собственном разложении.
- SVD и собственное разложение тесно связаны своими проекциями:
  - Левосингулярные векторы  $\mathbf{A}$  являются собственными векторами  $\mathbf{A}\mathbf{A}^T$ .
  - Правосингулярные векторы  $\mathbf{A}$  являются собственными векторами  $\mathbf{A}^T\mathbf{A}$ .
  - Ненулевые сингулярные значения  $\mathbf{A}$  являются квадратными корнями ненулевых собственных значений матрицы  $\mathbf{A}\mathbf{A}^T$  и равны ненулевым собственным значениям матрицы  $\mathbf{A}^T\mathbf{A}$ .
- Для симметричных матриц  $\mathbf{A} \in \mathbb{R}^{n \times n}$  разложение по собственным значениям и SVD — одно и то же, что следует из спектральной теоремы 4.15.

#### Пример 4.14 (определение структуры рейтингов фильмов и зрителей)

Давайте добавим практическую интерпретацию SVD, проанализировав данные о людях и их любимых фильмах. Рассмотрим трех зрителей (Али, Беатрикс и Чандру), оценивающих четыре разных фильма («Звездные войны», «Бегущий по лезвию», «Амели», «Деликатесы»). Их рейтинги представляют собой значения от 0 (наихудшее) до 5 (наилучшее) и за- кодированы в матрице данных  $\mathbf{A} \in \mathbb{R}^{4 \times 3}$ , как показано на рис. 4.10. Каждая строка представляет фильм, каждый столбец — пользователя. Таким образом, векторами-столбцами рейтингов фильмов, по одному для каждого зрителя, являются  $\mathbf{x}_{\text{Ali}}$ ,  $\mathbf{x}_{\text{Beatrix}}$ ,  $\mathbf{x}_{\text{Chandra}}$ .

Разлагая  $\mathbf{A}$  на множители с помощью SVD, можно проследить закономерности в том, как эти зрители оценивают фильмы, а также понять, существует ли структура, позволяющая увидеть, кому какие фильмы нравятся. Применяя SVD к нашей матрице данных  $\mathbf{A}$ , мы делаем ряд предположений:

1. Все зрители оценивают фильмы последовательно, используя одно и то же линейное отображение.
2. В рейтингах нет ошибок и шумов.
3. Мы интерпретируем левосингулярные векторы  $\mathbf{u}_i$  как стереотипные фильмы, а правосингулярные векторы  $\mathbf{v}_j$  как стереотипных зрителей.

Затем мы делаем допущение, что предпочтения любого зрителя в отношении фильма могут быть выражены как линейная комбинация  $\mathbf{v}_j$ . Ана-

логичным образом «предпочитаемость» любого фильма может быть выражена как линейная комбинация  $\mathbf{u}_i$ . Следовательно, вектор в области SVD можно интерпретировать как зрителя в «пространстве» стереотипных зрителей, а вектор в кодомене SVD, соответственно, как фильм в «пространстве» стереотипных фильмов<sup>1</sup>. Давайте проверим SVD нашей кино-пользовательской матрицы. Первый левосингулярный вектор  $\mathbf{u}_1$  имеет большие абсолютные значения для двух научно-фантастических фильмов и большое первое сингулярное значение (темная заливка на рис. 4.10). Таким образом, это группирует тип пользователей с определенным набором фильмов (тема научной фантастики). Точно так же первое правое единственное число  $\sigma_1$  показывает большие абсолютные значения для Али и Беатрикс, которые высоко оценивают научно-фантастические фильмы (светлая заливка на рис. 4.10). Это говорит о том, что  $\mathbf{v}_1$  отражает тип любителя научной фантастики.

$$\begin{array}{ccc}
 & \text{Али} & \\
 & 5 & 4 \\
 & 5 & 5 \\
 & 0 & 0 \\
 & 1 & 0
 \end{array}
 \begin{array}{ccc}
 & \text{Беатрикс} & \\
 & 1 & 0 \\
 & 0 & 5 \\
 & 4 & 0
 \end{array}
 \begin{array}{ccc}
 & \text{Чандра} & \\
 & 1 & 5 \\
 & 0 & 4
 \end{array}
 = \begin{bmatrix}
 -0,6710 & 0,0236 & 0,4647 & -0,5774 \\
 -0,7197 & 0,0254 & -0,4759 & 0,4619 \\
 -0,0939 & -0,7705 & -0,5268 & -0,3464 \\
 -0,1515 & -0,6030 & 0,5293 & -0,5774
 \end{bmatrix}
 \begin{bmatrix}
 9,6438 & 0 & 0 \\
 0 & 6,3639 & 0 \\
 0 & 0 & 0,7056 \\
 0 & 0 & 0
 \end{bmatrix}
 \begin{bmatrix}
 -0,7367 & -0,6515 & -0,1811 \\
 0,0852 & 0,1762 & -0,9807 \\
 0,6708 & -0,7379 & -0,7043
 \end{bmatrix}$$

**Рис. 4.10.** Оценки трех человек для четырех фильмов и их SVD-разложение

Точно так же  $\mathbf{u}_2$  отражает тему фильмов французского арт-хауса, а  $\mathbf{v}_2$  указывает на то, что Чандру можно считать почти идеализированной любительницей таких фильмов. Идеализированный любитель научной фантастики — пурист и не смотрит ничего, кроме научно-фантастических

<sup>1</sup> Эти два «пространства» будут значимыми для соответствующего зрителя и фильма, только если данные содержат достаточное разнообразие зрителей и фильмов.

фильмов, поэтому этот зритель  $v_1$  дает нулевую оценку всему, кроме тематики научной фантастики, — эта логика подразумевает диагональную подструктуру для матрицы сингулярных значений  $\Sigma$ . Таким образом, конкретный фильм представлен тем, как он (линейно) разлагается на стереотипные фильмы. Точно так же конкретного зрителя можно представить через тематическое распределение тех фильмов, что ему нравятся (здесь также применяется линейная комбинация).

Стоит кратко обсудить терминологию и условные обозначения SVD, поскольку в литературе используются разные версии. С математической точки зрения эти отличия инвариантны, но, вообще говоря, могут приводить к некоторой путанице.

- Для удобства в обозначениях и абстракции мы используем обозначение SVD, где SVD описывается как имеющее две квадратные лево- и правосингулярные векторные матрицы, но неквадратную матрицу сингулярных значений. Наше определение (4.64) для SVD иногда называют *полным SVD*.
- Некоторые авторы определяют SVD несколько иначе и сосредоточиваются на квадратных сингулярных матрицах. Тогда для  $A \in \mathbb{R}^{m \times n}$  и  $m \geq n$ ,

$$\underset{m \times n}{A} = \underset{m \times n}{U} \underset{n \times n}{\Sigma} \underset{n \times n}{V^T}. \quad (4.89)$$

Иногда эту формулировку называют *уменьшенным SVD* (например, Datta, 2010) или SVD (например, Press et al., 2007). Этот альтернативный формат меняет только способ построения матриц, но оставляет неизменной математическую структуру SVD. Удобство этой альтернативной формулировки состоит в том, что  $\Sigma$  диагональна, как в разложении на собственные значения.

- В разделе 4.6 мы узнаем о методах аппроксимации матриц с использованием SVD, который также называют *усеченным SVD*.
- Можно определить SVD матрицы  $A$  ранга  $r$  так, чтобы  $U$  была матрицей размера  $m \times r$ ,  $\Sigma$  — диагональной матрицей  $r \times r$ , а  $V$  — матрицей размера  $r \times n$ . Эта конструкция очень похожа на сформулированное нами определение и гарантирует, что диагональная матрица  $\Sigma$  содержит вдоль диагонали только ненулевые элементы. Основное удобство этой альтернативной записи заключается в том, что  $\Sigma$  диагональна, как в разложении по собственным значениям.
- Ограничение, состоящее в том, что SVD для  $A$  применяется только к матрицам  $m \times n$  с  $m > n$ , практически не требуется. Когда  $m < n$ , SVD-разложение

даст  $\Sigma$  с большим количеством нулевых столбцов, чем строк и, следовательно, сингулярными значениями  $\sigma_{m+1}, \dots, \sigma_n$ , равными 0.

SVD используется во множестве приложений машинного обучения, от метода наименьших квадратов при аппроксимации кривой до решения систем линейных уравнений. Эти приложения используют различные важные свойства SVD, его связь с рангом матрицы и его способность аппроксимировать матрицы заданного ранга матрицами более низкого ранга. Преимущество замены матрицы на ее SVD состоит в том, что вычисления становятся более устойчивыми к ошибкам численного округления. Как мы увидим в следующем разделе, способность SVD систематически аппроксимировать матрицы с помощью «более простых» матриц открывает возможности для приложений машинного обучения, начиная от уменьшения размерности и тематического моделирования до сжатия и кластеризации данных.

## 4.6. МАТРИЧНОЕ ПРИБЛИЖЕНИЕ

Мы рассматривали SVD как способ факторизовать  $A = U\Sigma V^T \in \mathbb{R}^{m \times n}$  в произведение трех матриц, где  $U \in \mathbb{R}^{m \times m}$  и  $V \in \mathbb{R}^{n \times n}$  ортогональны, а  $\Sigma$  содержит сингулярные значения на своей главной диагонали. Вместо того чтобы выполнять полную факторизацию SVD, мы теперь исследуем, как SVD позволяет нам представить матрицу  $A$  как сумму более простых (низкоранговых) матриц  $A_i$ , которая поддается схеме аппроксимации матрицы, которая дешевле в вычислении, чем полное SVD. Построим матрицу  $A_i \in \mathbb{R}^{m \times n}$  ранга 1 следующим образом:

$$A_i := u_i v_i^T, \quad (4.90)$$

который образован внешним произведением  $i$ -го ортогонального вектора-столбца  $U$  и  $V$ . На рис. 4.11 показано изображение Стоунхенджа, которое может быть представлено матрицей  $A \in \mathbb{R}^{1432 \times 1910}$  и некоторыми внешними произведениями  $A_i$ , как определено в (4.90). Матрица  $A \in \mathbb{R}^{m \times n}$  ранга  $r$  может быть записана как сумма матриц  $A_i$  ранга 1, так что

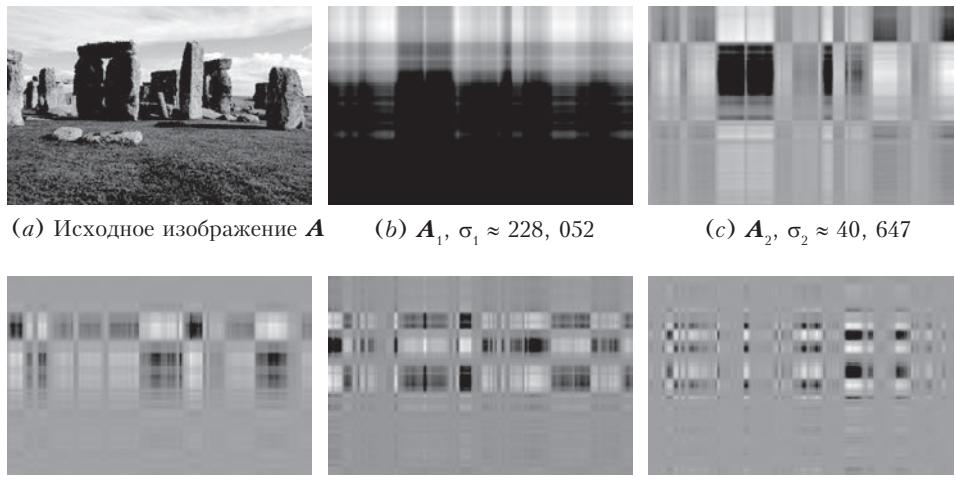
$$A = \sum_{i=1}^r \sigma_i u_i v_i^T = \sum_{i=1}^r \sigma_i A_i, \quad (4.91)$$

где матрицы внешнего произведения  $A_i$  взвешиваются  $i$ -м сингулярным значением  $\sigma_i$ . Мы можем видеть, почему выполняется (4.91): диагональная структура матрицы сингулярных значений  $\Sigma$  умножает только совпадающие левые и правосингулярные векторы  $u_i v_i^T$  и масштабирует их на соответствующее сингулярное значение  $\sigma_i$ . Все члены  $\sum_j u_i v_j^T$  обращаются в нуль при  $i \neq j$ , посколь-

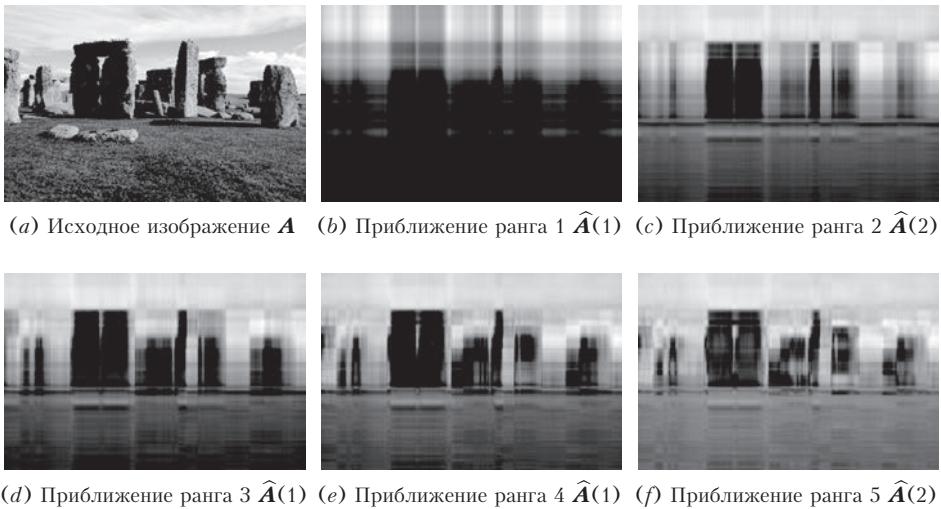
ку  $\Sigma$  — диагональная матрица. Любые члены  $i > r$  обращаются в нуль, поскольку соответствующие сингулярные значения равны 0. В (4.90) мы ввели матрицы  $A_i$  ранга 1. Мы просуммировали  $r$  отдельных матриц ранга 1, чтобы получить матрицу  $A$  ранга  $r$ ; см. (4.91). Если сумма не проходит по всем матрицам  $A_i$ ,  $i = 1, \dots, r$ , но только до промежуточного значения  $k < r$ , мы получаем приближение ранга  $k$

$$\hat{A}(k) := \sum_{i=1}^k \sigma_i u_i v_i^\top = \sum_{i=1}^k \sigma_i A_i \quad (4.92)$$

группы  $A$  с  $(\hat{A}(k)) = k$ . На рис. 4.12 показаны аппроксимации  $\hat{A}(k)$  низкого ранга исходного изображения  $A$  Стоунхенджа. Форма скал становится все более заметной и отчетливо узнаваемой в приближении 5-го ранга. В то время как исходное изображение требует  $1432 \cdot 1910 = 2\,735\,120$  чисел, приближение ранга 5 требует, чтобы мы сохраняли только пять сингулярных значений и пять левых и правых векторов ( $1432$ - и  $1910$ -размерный каждый), всего  $5 \cdot (1432 + 1910 + 1) = 16\,715$  чисел — чуть больше 0,6% от оригинала. Чтобы измерить разницу (ошибку) между  $A$  и его приближением  $\hat{A}(k)$  ранга  $k$ , нам понадобится понятие нормы. В разделе 3.1 мы уже применяли векторные нормы, чтобы измерить длину вектора. По аналогии мы также можем определить нормы матриц.



**Рис. 4.11.** Обработка изображений с помощью SVD. (a) Исходное изображение в градациях серого представляет собой матрицу размером  $1432 \times 1910$  значений от 0 (черный) до 1 (белый). (b)–(f) Матрицы  $A_1, \dots, A_5$  и соответствующие им особые значения  $\sigma_1, \dots, \sigma_5$ . Сеточноподобная структура каждой матрицы ранга 1 накладывается внешним произведением лево- и правосингулярных векторов



**Рис. 4.12.** Реконструкция изображения с помощью SVD. (a) Исходное изображение. (b)–(f) Восстановление изображения с использованием приближения низкого ранга SVD, где приближение ранга  $k$  задается как  $\hat{A}(k) = \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{v}_i^T$

**Определение 4.23 (спектральная норма матрицы).** Для  $\mathbf{x} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$  спектральная норма матрицы  $\mathbf{A} \in \mathbb{R}^{m \times n}$  определяется как

$$\|\mathbf{A}\|_2 := \max_{\mathbf{x}} \frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_2}. \quad (4.93)$$

Введем обозначение нижнего индекса в матричной норме (левая часть) аналогично евклидовой норме для векторов (правая часть), у которой есть нижний индекс 2. Спектральная норма (4.93) определяет, насколько длинным может стать любой вектор  $\mathbf{x}$  при умножении на  $\mathbf{A}$ .

**Теорема 4.24.** Спектральная норма матрицы  $\mathbf{A}$  — это наибольшее сингулярное значение  $\sigma_1$ .

Мы оставляем доказательство этой теоремы в качестве упражнения.

**Теорема 4.25 (теорема Эккарта – Юнга (Eckart and Young, 1936)).** Рассмотрим матрицу  $\mathbb{R}^{m \times n}$  ранга  $r$ , и пусть  $\mathbf{B} \in \mathbb{R}^{m \times n}$  — матрица ранга  $k$ . Для любого  $k \leq r$  с  $\hat{A}(k) = \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{v}_i^T$  выполняется

$$\hat{A}(k) = \arg \min_{\text{rk}(\mathbf{B})=k} \|\mathbf{A} - \mathbf{B}\|_2, \quad (4.94)$$

$$\|\mathbf{A} - \hat{A}(k)\|_2 = \sigma_{k+1}. \quad (4.95)$$

Теорема Эккарта – Юнга явно указывает, сколько ошибок мы вносим, аппроксимируя  $\mathbf{A}$  с использованием приближения ранга  $k$ . Мы можем интерпретировать приближение ранга  $k$ , полученное с помощью SVD, как проекцию матрицы  $\mathbf{A}$  полного ранга на пространство меньшей размерности матриц ранга не более  $k$ . Из всех возможных проекций SVD минимизирует ошибку (относительно спектральной нормы) между  $\mathbf{A}$  и любым приближением ранга  $k$ .

Можно повторно отследить некоторые шаги, чтобы понять, почему выражение (4.95) верно. Заметим, что разность между  $\mathbf{A}$  и  $\hat{\mathbf{A}}(k)$  представляет собой матрицу, содержащую сумму оставшихся матриц ранга 1:

$$\mathbf{A} - \hat{\mathbf{A}}(k) = \sum_{i=k+1}^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T. \quad (4.96)$$

По теореме 4.24 сразу получаем  $\sigma_{k+1}$  как спектральную норму разностной матрицы. Рассмотрим подробнее (4.94). Если предположить, что существует другая матрица  $\mathbf{B}$  с  $\text{rk}(\mathbf{B}) \leq k$ , такая что

$$\|\mathbf{A} - \mathbf{B}\|_2 < \|\mathbf{A} - \hat{\mathbf{A}}(k)\|_2, \quad (4.97)$$

то существует не менее чем  $(n - k)$ -мерное нулевое пространство  $Z \subseteq \mathbb{R}^n$ , такое что из  $\mathbf{x} \in Z$  следует, что  $\mathbf{Bx} = \mathbf{0}$ . Тогда следует, что

$$\|\mathbf{Ax}\|_2 = \|(\mathbf{A} - \mathbf{B})\mathbf{x}\|_2, \quad (4.98)$$

и, используя вариант неравенства Коши – Шварца (3.17), охватывающий нормы матриц, получаем

$$\|\mathbf{Ax}\|_2 \leq \|(\mathbf{A} - \mathbf{B})\|_2 \|\mathbf{x}\|_2 < \sigma_{k+1} \|\mathbf{x}\|_2. \quad (4.99)$$

Однако существует  $(k + 1)$ -мерное подпространство, где  $\|\mathbf{Ax}\|_2 \geq \sigma_{k+1} \|\mathbf{x}\|_2$ , которое натянуто на правосингулярные векторы  $\mathbf{v}_j, j \leq k + 1$  множества  $\mathbf{A}$ . Суммирование размерностей этих двух пространств дает число больше  $n$ , так как в обоих пространствах должен быть ненулевой вектор. Это противоречит теореме о ранге недействительности (теорема 2.24) из раздела 2.7.3.

Из теоремы Эккарта – Юнга следует, что мы можем использовать SVD для уменьшения матрицы  $\mathbf{A}$  ранга  $r$  до матрицы  $\hat{\mathbf{A}}$  ранга  $k$  принципиальным оптимальным (в смысле спектральной нормы) способом. Мы можем интерпретировать аппроксимацию  $\mathbf{A}$  матрицей ранга  $k$  как форму сжатия с потерями. Следовательно, низкоранговая аппроксимация матрицы появляется во многих приложениях машинного обучения, например при обработке изображений, фильтрации шума и регуляризации некорректно поставленных задач. Кроме того, она играет ключевую роль в уменьшении размерности и анализе главных компонентов, как мы увидим в главе 10.

**Пример 4.15 (поиск структуры в рейтингах фильмов и потребителях (продолжение))**

Возвращаясь к нашему примеру рейтингов фильмов, теперь мы можем применить концепцию аппроксимации низкого ранга для аппроксимации исходной матрицы данных. Напомним, что наша первая сингулярная величина отражает понятия темы научной фантастики в фильмах и любителя науки фантастики. Таким образом, используя только первый сингулярный член в разложении ранга 1 матрицы рейтингов фильмов, мы получаем прогнозируемые рейтинги

$$\mathbf{A}_1 = \mathbf{u}_1 \mathbf{v}_1^T = \begin{bmatrix} -0,6710 \\ -0,7197 \\ -0,0939 \\ -0,1515 \end{bmatrix} \begin{bmatrix} -0,7367 & -0,6515 & -0,1811 \end{bmatrix} = \quad (4.100a)$$

$$= \begin{bmatrix} 0,4943 & 0,4372 & 0,1215 \\ 0,5302 & 0,4689 & 0,1303 \\ 0,0692 & 0,0612 & 0,0170 \\ 0,1116 & 0,0987 & 0,0274 \end{bmatrix}. \quad (4.100b)$$

Это первое приближение ранга  $\mathbf{A}_1$  информативно: мы видим, что Али и Беатрикс любят научно-фантастические фильмы, такие как «Звездные войны» и «Бегущий по лезвию» (записи имеют значения  $> 4$ ), но не знаем, какие рейтинги дает другим фильмам Чандре. Это неудивительно, поскольку в первое сингулярное значение не входят фильмы, которые понравились бы Чандре. Второе сингулярное значение дает нам лучшее приближение ранга 1 для любителей этой тематики фильмов:

$$\mathbf{A}_2 = \mathbf{u}_2 \mathbf{v}_2^T = \begin{bmatrix} 0,0236 \\ 0,2054 \\ -0,7705 \\ -0,6030 \end{bmatrix} \begin{bmatrix} 0,0852 & 0,1762 & -0,9807 \end{bmatrix} = \quad (4.101a)$$

$$= \begin{bmatrix} -0,0154 & 0,0042 & -0,0174 \\ -0,1338 & 0,0362 & -0,1516 \\ 0,5019 & -0,1358 & 0,5686 \\ 0,3928 & -0,1063 & 0,445 \end{bmatrix}. \quad (4.101b)$$

Во втором приближении ранга 1  $\hat{\mathbf{A}}_2$  мы получаем рейтинги и тематики фильмов, которые любит Чандря, но сюда не попадают научно-фантастические фильмы. Поэтому следует рассмотреть приближение ранга 2  $\hat{\mathbf{A}}(2)$ , где мы объединяем первые два приближения ранга 1:

$$\hat{\mathbf{A}}(2) = \sigma_1 \mathbf{A}_1 + \sigma_2 \mathbf{A}_2 = \begin{bmatrix} 4,7801 & 4,2419 & 1,0244 \\ 5,2252 & 4,7522 & -0,0250 \\ 0,2493 & -0,2743 & 4,9724 \\ 0,7495 & 0,2756 & 4,0278 \end{bmatrix}. \quad (4.102)$$

$\hat{\mathbf{A}}(2)$  подобна исходной таблице рейтингов фильмов

$$\mathbf{A} = \begin{bmatrix} 5 & 4 & 1 \\ 5 & 5 & 0 \\ 0 & 0 & 5 \\ 1 & 0 & 4 \end{bmatrix}, \quad (4.103)$$

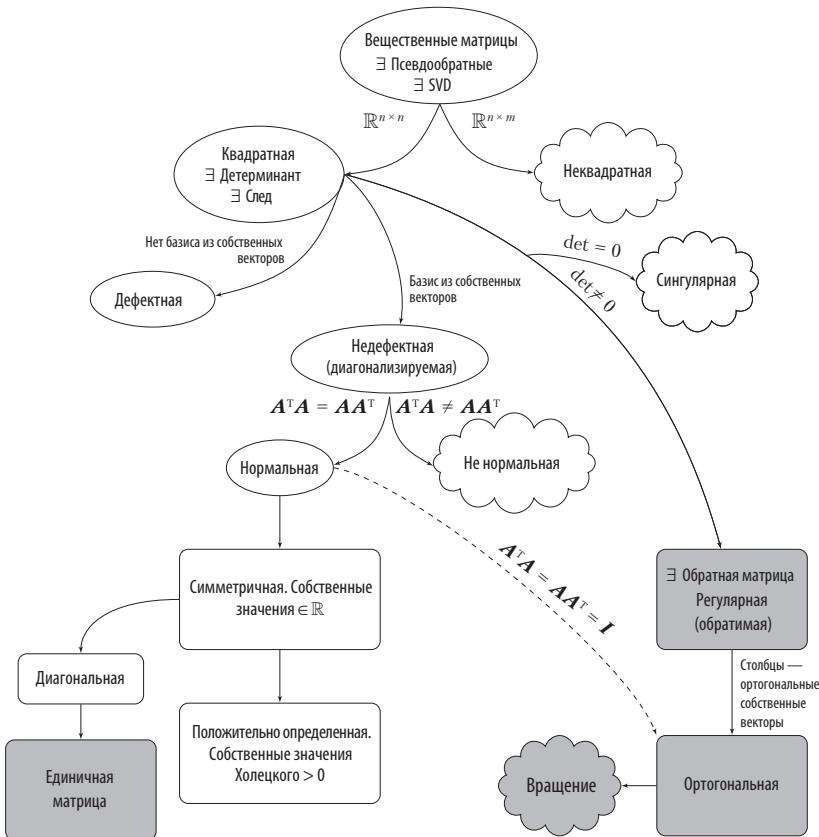
следовательно, мы можем игнорировать вклад  $\mathbf{A}_3$ . Это можно интерпретировать так, что в таблице нет данных по третьей категории тематики фильмов / любителей фильмов. Это также означает, что пространство тематики фильмов / любители фильмов в нашем примере — двухмерное, и содержит только тематику и любителей научной фантастики и французского арт-хауса.

## 4.7. МАТРИЧНАЯ ФИЛОГЕНИЯ

В главах 2 и 3 мы рассмотрели основы линейной алгебры и аналитической геометрии. В этой главе изучили фундаментальные характеристики матриц и линейных отображений. На рис. 4.13 изображено филогенетическое<sup>1</sup> дерево отношений между различными типами матриц (черные стрелки указывают «является подмножеством») и операциями, которые мы можем выполнять с ними. Рассмотрим все *вещественные матрицы*  $\mathbf{A} \in \mathbb{R}^{n \times m}$ . Для неквадратных матриц (где  $n \neq m$ ) SVD существует всегда, как мы видели в этой главе. Если сосредоточиться на *квадратных матрицах*  $\mathbf{A} \in \mathbb{R}^{n \times n}$ , то по их *детерминанту* можно определить, существует ли у квадратной матрицы обратная матрица, то есть принадлежит ли она к классу обратимых матриц. Если квадратная матрица размера  $n \times n$  имеет  $n$  линейно независимых собственных векторов,

<sup>1</sup> Слово «филогенетический» описывает, как мы фиксируем отношения между отдельными людьми или группами, и происходит от греческих слов «племя» и «источник».

то матрица не является *дефектной* и существует *собственное разложение* (теорема 4.12). Мы знаем, что повторяющиеся собственные значения могут привести к появлению дефектных матриц, которые не могут быть диагонализованы.



**Рис. 4.13.** Функциональная филогенетия матриц, встречающихся в машинном обучении

Неособые и недефектные матрицы — это не одно и то же. Например, матрица вращения будет обратимой (детерминант отличен от нуля), но не диагонализируемой в действительных числах (не гарантируется, что собственные значения будут действительными числами).

Давайте подробнее рассмотрим недефектные квадратные матрицы размера  $n \times n$ .  $A$  является нормальной, если выполняется условие  $A^T A = AA^T$ . Более того, если выполняется более жесткое условие  $A^T A = AA^T = I$ , то  $A$  называется ортогональной (см. определение 3.8). Множество ортогональных матриц яв-

ляется подмножеством регулярных (обратимых) матриц и удовлетворяет условию  $\mathbf{A}^T = \mathbf{A}^{-1}$ .

Нормальные матрицы имеют часто встречающееся подмножество симметричных матриц  $\mathbf{S} \in \mathbb{R}^{n \times n}$ , которые удовлетворяют условию  $\mathbf{S} = \mathbf{S}^T$ . Симметричные матрицы имеют только действительные собственные значения. Подмножество симметричных матриц состоит из положительно определенных матриц  $\mathbf{P}$ , которые удовлетворяют условию  $\mathbf{x}^T \mathbf{P} \mathbf{x} > 0$  для всех  $\mathbf{x} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ . В этом случае существует единственное *разложение Холецкого* (теорема 4.18). Положительно определенные матрицы имеют только положительные собственные значения и всегда обратимы (то есть имеют ненулевой детерминант).

Другое подмножество симметричных матриц состоит из *диагональных матриц*  $\mathbf{D}$ . Диагональные матрицы замкнуты при умножении и сложении, но не обязательно образуют группу (это происходит только в том случае, если все диагональные элементы не равны нулю, так что матрица обратима). Специальная диагональная матрица — это единичная матрица  $\mathbf{I}$ .

## 4.8. ДОПОЛНИТЕЛЬНОЕ ЧТЕНИЕ

Большая часть этой главы посвящена математическому аппарату и его связям с методами сопоставления (маппинга) данных. Эти методы лежат в основе программных решений машинного обучения. Характеристика матриц с использованием детерминантов, собственных спектров и собственных подпространств создает возможности и условия для категоризации и анализа матриц. Это распространяется на все формы представления маппинга данных, а также на оценку численной устойчивости вычислительных операций с такими матрицами (Press et al., 2007).

Детерминанты — это фундаментальные инструменты для обращения матриц и вычисления собственных значений «вручную». Однако почти во всех случаях численные вычисления с применением метода Гаусса более эффективны, чем детерминанты (Press et al., 2007). Тем не менее, детерминанты остаются мощной теоретической концепцией, например для получения интуитивного понимания ориентации базиса на основе знака детерминанты. Собственные векторы могут использоваться для выполнения базисных изменений для преобразования данных в координаты значимых ортогональных векторов признаков. Точно так же методы разложения матриц, такие как разложение Холецкого, часто возникают при вычислении или моделировании случайных событий (Rubinstein and Kroese, 2016). Таким образом, разложение Холецкого позволяет нам использовать *прием репараметризации*, когда мы хотим выполнить непрерывное дифференцирование по случайным величинам, на-

пример в вариационных автоэнкодерах (Jimenez Rezende et al., 2014; Kingma and Ba, 2014).

Фундаментальная ценность разложения матрицы на основе собственных векторов заключается в том, что оно позволяет извлекать значимую и интерпретируемую информацию, которая характеризует линейные отображения. Следовательно, собственное разложение лежит в основе общего класса алгоритмов машинного обучения, называемых *спектральными методами*, которые выполняют собственное разложение положительно определенного ядра. Эти методы спектральной декомпозиции включают классические подходы к статистическому анализу данных, такие как:

- *Анализ главных компонент* (principal component analysis — PCA (Pearson, 1901), см. также главу 10), в котором ищется низкоразмерное подпространство, которое объясняет большую часть изменчивости данных.
- *Дискриминантный анализ Фишера*, целью которого является определение разделяющей гиперплоскости для классификации данных (Mika et al., 1999).
- *Многомерное масштабирование* (multidimensional scaling — MDS) (Carroll and Chang, 1970).

Вычислительная эффективность этих методов обычно достигается за счет нахождения наилучшего приближения ранга  $k$  к симметричной положительно полуопределенной матрице. Более современные примеры спектральных методов имеют различное происхождение, но каждый из них требует вычисления собственных векторов и собственных значений положительно определенного ядра. Сюда входят *Isomap* (Tenenbaum et al., 2000), *лапласовские собственные карты* (Belkin and Niyogi, 2003), *собственные карты Гессе* (Donoho and Grimes, 2003) и *спектральная кластеризация* (Shi and Malik, 2000). Основные вычисления для них обычно подкрепляются методами аппроксимации матриц низкого ранга (Belabbas and Wolfe, 2009), с которыми мы встречались при изучении SVD.

SVD позволяет извлекать информацию того же типа, что и собственное разложение. Однако SVD более широко применяется к неквадратным матрицам и таблицам данных. Эти методы матричной факторизации становятся актуальными всякий раз, когда мы хотим идентифицировать неоднородность данных и планируем выполнить сжатие данных путем аппроксимации, например вместо хранения значений  $n \times m$  сохраняя  $k(n + m)$  значений. Другой вариант — когда хотим выполнить предварительную обработку данных, например для декорреляции переменных-предикторов матрицы плана (Ortmoneit et al., 2001). SVD работает с матрицами, которые можно интерпретировать как прямоугольные массивы с двумя индексами (строками и столбцами). Структуры, подобные матрицам, но в многомерном пространстве, называются тензорами. На самом

деле SVD – это частный случай более общего семейства декомпозиций, которые работают с такими тензорами (Kolda and Bader, 2009). Примеры SVD-подобных операций и низкоранговых аппроксимаций тензоров – *разложение Таккера* (Tucker, 1966) или *CP-разложение* (Carroll and Chang, 1970).

Приближение низкого ранга SVD часто используется в машинном обучении в силу его вычислительной эффективности. Это связано с тем, что оно уменьшает объем памяти и операций с ненулевым умножением, которые необходимо выполнять с потенциально очень большими матрицами данных (Trefethen and Bau III, 1997). Более того, аппроксимации низкого ранга используются для работы с матрицами, которые могут содержать пропущенные значения, а также в целях сжатия с потерями и уменьшения размерности (Moonen and De Moor, 1995; Markovsky, 2011).

## УПРАЖНЕНИЯ

**4.1.** Вычислите детерминант, используя разложение Лапласа (взьмите первую строку) и правило Сарруса для

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \\ 0 & 2 & 4 \end{bmatrix}.$$

**4.2.** Эффективно вычислите следующий детерминант:

$$\begin{bmatrix} 2 & 0 & 1 & 2 & 0 \\ 2 & -1 & 0 & 1 & 1 \\ 0 & 1 & 2 & 1 & 2 \\ -2 & 0 & 2 & -1 & 2 \\ 2 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

**4.3.** Вычислите собственные подпространства  $\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ ,  $\begin{bmatrix} -2 & 2 \\ 2 & 1 \end{bmatrix}$ .

**4.4.** Вычислите все собственные подпространства

$$A = \begin{bmatrix} 0 & -1 & 1 & 1 \\ -1 & 1 & -2 & 3 \\ 2 & -1 & 0 & 0 \\ 1 & -1 & 1 & 0 \end{bmatrix}.$$

**4.5.** Диагонализуемость матрицы не связана с ее обратимостью. Определите для следующих четырех матриц, являются ли они диагонализуемыми и/или обратимыми.

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

**4.6.** Вычислите собственные подпространства следующих матриц преобразования. Являются ли они диагонализуемыми?

a.  $\mathbf{A} = \begin{bmatrix} 2 & 3 & 0 \\ 1 & 4 & 3 \\ 0 & 0 & 1 \end{bmatrix}$

b.  $\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

**4.7.** Диагонализуемы ли следующие матрицы? Если да, определите их диагональную форму и базис, относительно которого матрицы преобразования являются диагональными. Если нет, укажите причины, по которым они не поддаются диагонализации.

a.  $\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -8 & 4 \end{bmatrix}$

b.  $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$

c.  $\mathbf{A} = \begin{bmatrix} 5 & 4 & 2 & 1 \\ 0 & 1 & -1 & -1 \\ -1 & -1 & 3 & 0 \\ 1 & 1 & -1 & 2 \end{bmatrix}$

d.  $\mathbf{A} = \begin{bmatrix} 5 & -6 & -6 \\ -1 & 4 & 2 \\ 3 & -6 & -4 \end{bmatrix}$

**4.8.** Вычислите сингулярное разложение матрицы:

$$\mathbf{A} = \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix}.$$

**4.9.** Вычислите сингулярное разложение

$$\mathbf{A} = \begin{bmatrix} 2 & 2 \\ -1 & 1 \end{bmatrix}.$$

**4.10.** Найдите наилучшее приближение ранга 1.

$$\mathbf{A} = \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix}.$$

**4.11.** Покажите, что для любого  $\mathbf{A} \in \mathbb{R}^{m \times n}$  матрицы  $\mathbf{A}^T \mathbf{A}$  и  $\mathbf{A} \mathbf{A}^T$  имеют одинаковые ненулевые собственные значения.

**4.12.** Покажите, что при  $\mathbf{x} \neq \mathbf{0}$  верна теорема 4.24, то есть покажите, что

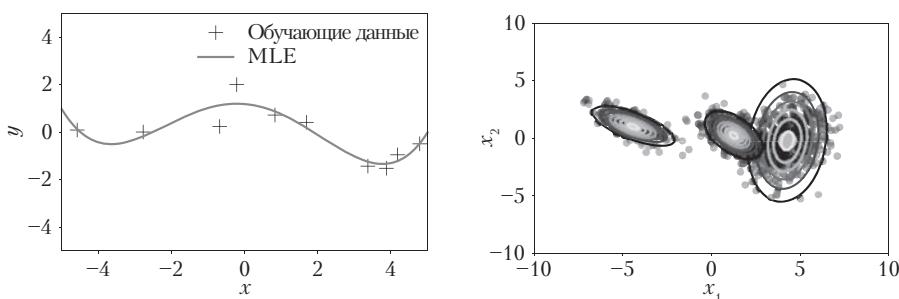
$$\max_{\mathbf{x}} \frac{\|\mathbf{Ax}\|_2}{\|\mathbf{x}\|_2} = \sigma_1,$$

где  $\sigma_1$  — наибольшее сингулярное значение  $\mathbf{A} \in \mathbb{R}^{m \times n}$ .

# 5

## Векторный анализ

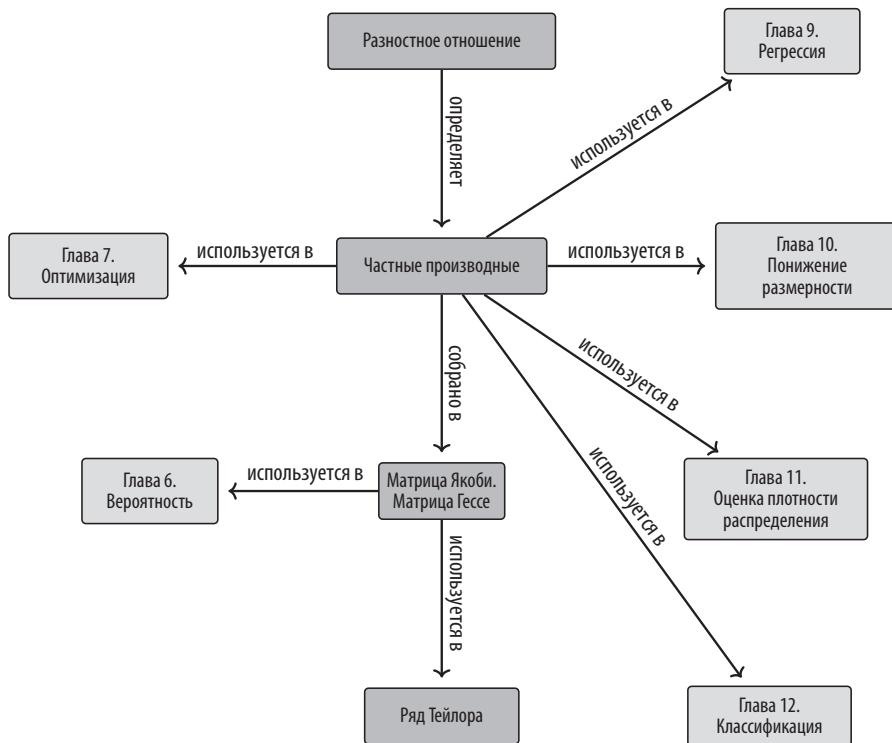
Многие алгоритмы машинного обучения оптимизируют целевую функцию относительно параметров модели, от которых зависит то, насколько хорошо модель описывает данные. Нахождение подходящих параметров можно сформулировать как задачу оптимизации (разделы 8.2 и 8.3). Примерами являются, в частности, (i) линейная регрессия (глава 9), используемая для задач приближения кривой, — в ней мы подбираем параметры (веса) так, чтобы максимизировать правдоподобие; (ii) нейронные сети-автокодировщики, применяемые для понижения размерности и сжатия данных, — параметрами будут веса и смещения на каждом из уровней, и мы минимизируем ошибку восстановления повторным применением цепного правила; и (iii) смешанные гауссовые модели (глава 11), используемые для моделирования распределений данных, — в них мы оптимизируем параметры (среднее и дисперсию) каждого из компонентов смеси, чтобы максимизировать правдоподобие модели. На рис. 5.1 показаны некоторые из



**Рис. 5.1.** Векторный анализ играет ключевую роль при (a) регрессии (приближение кривой) и (b) оценке плотностей (моделирование распределений данных).

(a) Задача регрессии: найти такие параметры, чтобы кривая хорошо соответствовала наблюдениям (обозначенным крестиками). (b) Оценка плотностей для смешанного гауссова распределения. Необходимо найти такие средние и дисперсии, которые соответствуют наблюдаемым данным (обозначенными точками)

этих задач, которые мы обычно решаем с помощью алгоритмов оптимизации, основанных на вычислении градиента (раздел 7.1). Рисунок 5.2 дает представление о том, как темы этой главы соотносятся между собой и с остальными главами книги.



**Рис. 5.2.** Ассоциативная карта концепций, вводимых в этой главе, с указанием, в каких еще частях книги они фигурируют

Основным в данной главе является понятие функции. Функция  $f$  — это правило, связывающее две величины: — в нашей книге переменную  $\mathbf{x} \in \mathbb{R}^D$  и значение функции  $f(\mathbf{x})$  (которое мы считаем, если не оговорено иное, вещественным). Здесь  $\mathbb{R}^D$  называется областью значений  $f$ , а множество всех значений  $f(\mathbf{x})$  — образом  $f$ .

В разделе 2.7.3 гораздо более подробно обсуждаются линейные функции. При задании функции часто пишут

$$f: \mathbb{R}^D \rightarrow \mathbb{R}; \quad (5.1a)$$

$$\mathbf{x} \mapsto f(\mathbf{x}), \quad (5.1b)$$

где (5.1a) означает, что  $f$  — отображение из  $\mathbb{R}^D$  в  $\mathbb{R}$ , а (5.1b) описывает правило, по которому входу (аргументу)  $\mathbf{x}$  сопоставляется значение  $f(\mathbf{x})$ . Функция  $f$  сопоставляет каждому  $\mathbf{x}$  ровно одно значение  $f(\mathbf{x})$ .

### Пример 5.1

Вспомним частный случай скалярного произведения — стандартное скалярное произведение (раздел 3.2.2). В представленных обозначениях  $f(\mathbf{x}) = \mathbf{x}^T \mathbf{x}$ ,  $\mathbf{x} \in \mathbb{R}^2$  будет выглядеть как

$$f: \mathbb{R}^D \rightarrow \mathbb{R}; \quad (5.2a)$$

$$\mathbf{x} \mapsto x_1^2 + x_2^2. \quad (5.2b)$$

В этой главе мы обсудим, как вычислять градиенты функций, что зачастую играет ключевую роль в процессе обучения моделей, так как направление градиента — это то направление, в котором функция быстрее всего возрастает. Поэтому векторный анализ является одним из главных математических инструментов машинного обучения. В данной книге мы будем предполагать, что все функции дифференцируемы. С некоторыми техническими дополнениями, которых мы касаться не будем, многие подходы можно обобщить до субдифференцируемых функций (непрерывных, но в некоторых точках не дифференцируемых). Обобщение для случая некоторых ограничений на функцию мы рассмотрим в главе 7.

## 5.1. ДИФФЕРЕНЦИРОВАНИЕ ФУНКЦИЙ ОДНОЙ ПЕРЕМЕННОЙ

В этом разделе мы кратко повторим дифференцирование функций одной переменной, знакомое, быть может, из старшей школы. Начнем с разностного отношения для функции одной переменной  $y = f(x)$ ,  $x, y \in \mathbb{R}$ , а затем определим производную.

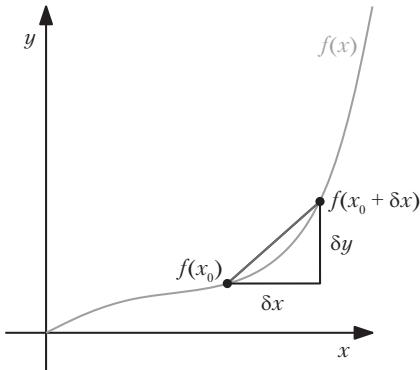
**Определение 5.1 (разностное отношение).** Разностное отношение

$$\frac{\delta y}{\delta x} := \frac{f(x + \delta x) - f(x)}{\delta x} \quad (5.3)$$

— это тангенс угла наклона секущей, проходящей через две точки на графике  $f$ . На рис. 5.3 это точки с  $x$ -координатами  $x_0$  и  $x_0 + \delta x$ .

Если  $f$  — линейная функция, то разностное отношение — это средний тангенс угла наклона секущей между  $x_0$  и  $x_0 + \delta x$ . В пределе, если  $f$  дифференцируема,

при  $\delta x \rightarrow 0$  получится тангенс угла наклона касательной к  $f$  в точке  $x$ . Он и будет значением производной  $f$  в точке  $x$ .



**Рис. 5.3.** Средний наклон графика функции  $f$  между точками  $x_0$  и  $x_0 + \delta x$  равен наклону секущей, проходящей через  $f(x_0)$  и  $f(x_0 + \delta x)$ , и задается формулой  $\delta y / \delta x$

**Определение 5.2 (производная).** Для  $h > 0$  производная  $f$  в точке  $x$  — это предел

$$\frac{df}{dx} := \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}, \quad (5.4)$$

и секущая на рис. 5.3 превращается в касательную.

### Пример 5.2 (производная многочлена)

Пусть нужно вычислить производную многочлена  $f(x) = x^n$ ,  $n \in \mathbb{N}$ . Возможно, вы уже знаете, что ответом будет  $nx^{n-1}$ , но давайте выведем его из определения производной как предела разностных отношений. По формуле (5.4) получаем

$$\frac{df}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h} = \quad (5.5a)$$

$$= \lim_{h \rightarrow 0} \frac{(x+h)^n - x^n}{h} = \quad (5.5b)$$

$$= \lim_{h \rightarrow 0} \frac{\sum_{i=0}^n \binom{n}{i} x^{n-i} h^i - x^n}{h}. \quad (5.5c)$$

Заметим, что  $x^n = \binom{n}{0} x^{n-0} h^0$ . Когда суммирование начинается с 1, слагаемое  $x^n$  уходит, и мы получаем

$$\frac{df}{dx} := \lim_{h \rightarrow 0} \frac{\sum_{i=1}^n \binom{n}{i} x^{n-i} h^i}{h} = \quad (5.6a)$$

$$= \lim_{h \rightarrow 0} \sum_{i=1}^n \binom{n}{i} x^{n-i} h^{i-1} = \quad (5.6b)$$

$$= \lim_{h \rightarrow 0} \binom{n}{i} x^{n-1} + \underbrace{\sum_{i=2}^n \binom{n}{i} x^{n-i} h^{i-1}}_{\rightarrow 0 \text{ при } h \rightarrow 0} = \quad (5.6c)$$

$$= \frac{n!}{1!(n-1)!} x^{n-1} = nx^{n-1}. \quad (5.6d)$$

### 5.1.1. Ряд Тейлора

Ряд Тейлора — это представление функции  $f$  в виде бесконечной суммы. Члены этой суммы определены через производные  $f$  в точке  $x_0$ .

**Определение 5.3 (многочлен Тейлора).** Многочлен Тейлора степени  $n$  функции  $f$  в точке  $x_0$  определяется как

$$T_n(x) := \sum_{k=0}^n \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k, \quad (5.7)$$

где  $f^{(k)}(x_0)$  —  $k$ -я производная функции  $f$  в точке  $x_0$  (мы предполагаем, что она существует), а  $\frac{f^{(k)}(x_0)}{k!}$  — коэффициенты многочлена<sup>1</sup>.

**Определение 5.4 (ряд Тейлора).** Для гладкой функции  $f \in C^\infty$  ( $f \in C^\infty$  означает, что  $f$  непрерывно дифференцируема бесконечно много раз),  $f: \mathbb{R} \rightarrow \mathbb{R}$ , ряд Тейлора функции  $f$  в точке  $x_0$  определяется как

$$T_\infty(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k. \quad (5.8)$$

<sup>1</sup> Мы определяем  $t^0 := 1$  для всех  $t \in \mathbb{R}$ .

Для  $x_0 = 0$  получаем частный случай ряда Тейлора — ряд Маклорена. Если  $f(x) = T_\infty C(x)$ ,  $f$  называют аналитической функцией.

**ПРИМЕЧАНИЕ** Многочлен Тейлора степени  $n$  — это аппроксимация функции  $f$ , вообще говоря, не полиномиальной. Его значения близки к значениям  $f$  в окрестности точки  $x_0$ . Однако если  $f$  — полиномиальная функция степени  $k \leq n$ , то многочлен Тейлора степени  $n$  совпадает с ней, поскольку все производные  $f^{(i)}$ ,  $i > k$ , обращаются в нуль. ♦

### Пример 5.3 (многочлен Тейлора)

Рассмотрим полиномиальную функцию

$$f(x) = x^4 \quad (5.9)$$

и построим многочлен Тейлора  $T_6$  в точке  $x_0 = 1$ . Сначала вычислим коэффициенты  $f^{(k)}(1)$  для  $k = 0, \dots, 6$ :

$$f(1) = 1; \quad (5.10)$$

$$f'(1) = 4; \quad (5.11)$$

$$f''(1) = 12; \quad (5.12)$$

$$f^{(3)}(1) = 24; \quad (5.13)$$

$$f^{(4)}(1) = 24; \quad (5.14)$$

$$f^{(5)}(1) = 0; \quad (5.15)$$

$$f^{(6)}(1) = 0. \quad (5.16)$$

Таким образом, искомый многочлен Тейлора равен

$$T_6(x) = \sum_{k=0}^6 \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k = \quad (5.17a)$$

$$= 1 + 4(x - 1) + 6(x - 1)^2 + 4(x - 1)^3 + (x - 1)^4 + 0. \quad (5.17b)$$

Раскрывая скобки и приводя подобные слагаемые, получим

$$\begin{aligned} T_6(x) &= (1 - 4 + 6 - 4 + 1) + x(4 - 12 + 12 - 4) + \\ &\quad + x^2(6 - 12 + 6) + x^3(4 - 4) + x^4 = \end{aligned} \quad (5.18a)$$

$$= x^4 = f(x), \quad (5.18b)$$

то есть в точности исходную функцию.

**Пример 5.4 (ряд Тейлора)**

Рассмотрим функцию с рис. 5.4, заданную формулой

$$f(x) = \sin(x) + \cos(x) \in C^\infty. \quad (5.19)$$

Найдем разложение  $f$  в ряд Тейлора в точке  $x_0 = 0$  (то есть разложение Маклорена). Производные выглядят так:

$$f(0) = \sin(0) + \cos(0) = 1; \quad (5.20)$$

$$f'(0) = \cos(0) - \sin(0) = 1; \quad (5.21)$$

$$f''(0) = -\sin(0) - \cos(0) = -1; \quad (5.22)$$

$$f^{(3)}(0) = -\cos(0) + \sin(0) = -1; \quad (5.23)$$

$$\begin{aligned} f^{(4)}(0) &= \sin(0) + \cos(0) = f(0) = 1. \\ &\vdots \end{aligned} \quad (5.24)$$

Видим закономерность: так как  $\sin(0) = 0$ , то коэффициенты нашего ряда Тейлора — это только  $\pm 1$ , повторяющиеся по два раза, и  $f^{(k+4)}(0) = f^{(k)}(0)$ . Таким образом, разложение  $f$  в ряд Тейлора в точке  $x_0 = 0$  выглядит как

$$T_\infty(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k = \quad (5.25a)$$

$$= 1 + x - \frac{1}{2!}x^2 - \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \frac{1}{5!}x^5 - \dots = \quad (5.25b)$$

$$= 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 \mp \dots + x - \frac{1}{3!}x^3 + \frac{1}{5!}x^5 \mp \dots = \quad (5.25c)$$

$$= \sum_{k=0}^{\infty} (-1)^k \frac{1}{(2k)!} x^{2k} + \sum_{k=0}^{\infty} (-1)^k \frac{1}{(2k+1)!} x^{2k+1} = \quad (5.25d)$$

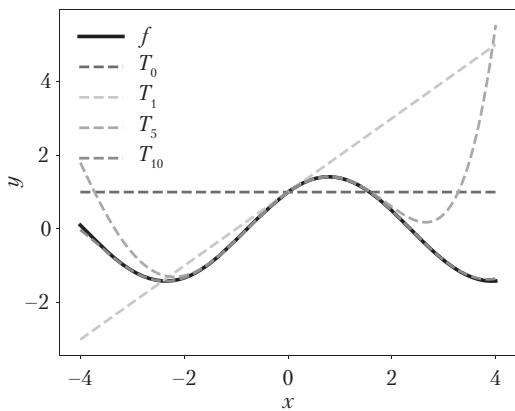
$$= \cos(x) + \sin(x), \quad (5.25e)$$

— последнее равенство получается из разложений в степенные ряды

$$\cos(x) = \sum_{k=0}^{\infty} (-1)^k \frac{1}{(2k)!} x^{2k}; \quad (5.26)$$

$$\sin(x) = \sum_{k=0}^{\infty} (-1)^k \frac{1}{(2k+1)!} x^{2k+1}. \quad (5.27)$$

На рис. 5.4 показаны многочлены Тейлора  $T_n$  для  $n = 0, 1, 5, 10$ .



**Рис. 5.4.** Многочлены Тейлора. Исходная функция  $f(x) = \sin(x) + \cos(x)$  (сплошная черная линия) приближается многочленом Тейлора (пунктирная линия) в окрестности точки  $x_0 = 0$ . Многочлены Тейлора более высоких порядков приближают  $f$  точнее и на большем участке.  $T_{10}$  близка к  $f$  на  $[-4, 4]$

**ПРИМЕЧАНИЕ** Ряд Тейлора — частный случай степенного ряда

$$f(x) = \sum_{k=0}^{\infty} a_k (x - c)^k, \quad (5.28)$$

где  $a_k$  — коэффициенты (имеющие в определении 5.4 специальный вид),  $c$  — константа. ♦

### 5.1.2. Правила дифференцирования

В этом разделе мы сформулируем основные правила дифференцирования. Производную функции  $f$  мы обозначаем  $f'$ .

Производная произведения:  $(f(x)g(x))' = f'(x)g(x) + f(x)g'(x)$ . (5.29)

Производная частного:  $\left(\frac{f(x)}{g(x)}\right)' = \frac{f'(x)g(x) - f(x)g'(x)}{(g(x))^2}$ . (5.30)

Производная суммы:  $(f(x) + g(x))' = f'(x) + g'(x)$ . (5.31)

Цепное правило  $(g(f(x)))' = (g \circ f)'(x) = g'(f(x))f'(x)$ . (5.32)  
(производная композиции):

Здесь  $g \circ f$  обозначает композицию функций  $x \mapsto f(x) \mapsto g(f(x))$ .

**Пример 5.5 (цепное правило)**

Используя цепное правило, вычислим производную функции  $h(x) = (2x + 1)^4$ . Так как

$$h(x) = (2x + 1)^4 = g(f(x)), \quad (5.33)$$

$$f(x) = 2x + 1, \quad (5.34)$$

$$g(f) = f^4, \quad (5.35)$$

и производные  $f$  и  $g$

$$f'(x) = 2, \quad (5.36)$$

$$g'(f) = 4f^3, \quad (5.37)$$

получаем производную  $h$ :

$$h'(x) = g'(f) f'(x) = (4f^3) \cdot 2 = 4(2x+1)^3 \cdot 2 = 8(2x+1)^3, \quad (5.38)$$

(мы применили правило (5.32) и подставили в  $g'(f)$  определение функции  $f$ ).

## 5.2. ЧАСТНЫЕ ПРОИЗВОДНЫЕ И ГРАДИЕНТЫ

В разделе 5.1 мы обсуждали дифференцирование функций одной скалярной переменной  $x \in \mathbb{R}$ . В данном разделе мы рассмотрим общий случай, в котором  $f$  зависит от одной или нескольких переменных  $\mathbf{x} \in \mathbb{R}^n$ , например  $f(\mathbf{x}) = f(x_1, x_2)$ . Обобщением производной для функций нескольких переменных является градиент.

Мы фиксируем значения всех переменных, кроме одной, относительно которой и вычисляем производную. Набор таких частных производных по всем переменным и будет называться градиентом.

**Определение 5.5 (частная производная).** Для функции  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $\mathbf{x} \mapsto f(\mathbf{x})$ ,  $\mathbf{x} \in \mathbb{R}^n$  от  $n$  переменных  $x_1, \dots, x_n$  частные производные определяются как

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= \lim_{h \rightarrow 0} \frac{f(x_1 + h, x_2, \dots, x_n) - f(\mathbf{x})}{h}, \\ &\vdots \\ \frac{\partial f}{\partial x_n} &= \lim_{h \rightarrow 0} \frac{f(x_1, \dots, x_{n-1}, x_n + h) - f(\mathbf{x})}{h}. \end{aligned} \quad (5.39)$$

Объединим их в вектор-строку

$$\nabla_{\mathbf{x}} f = \text{grad} f = \frac{df}{d\mathbf{x}} = \begin{bmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} & \frac{\partial f(\mathbf{x})}{\partial x_2} & \dots & \frac{\partial f(\mathbf{x})}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{1 \times n}, \quad (5.40)$$

где  $n$  — количество переменных, а 1 — размерность области значений  $f$ . Здесь мы рассматриваем вектор-столбец  $\mathbf{x} = [x_1, \dots, x_n]^T$ . Вектор-строка из (5.40) называется *градиентом* или *матрицей Якоби функции* (якобианом)  $f$  и обобщает понятие производной из раздела 5.1.

**ПРИМЕЧАНИЕ** Это определение якобиана является частным случаем общего определения якобиана для векторнозначных функций — как набора частных производных. К этому понятию мы вернемся в разделе 5.3. ♦

### Пример 5.6 (Вычисление частных производных по цепному правилу)

Для  $f(x, y) = (x + 2y^3)^2$  частные производные равны

$$\frac{\partial f(x, y)}{\partial x} = 2(x + 2y^3) \frac{\partial}{\partial x}(x + 2y^3) = 2(x + 2y^3); \quad (5.41)$$

$$\frac{\partial f(x, y)}{\partial y} = 2(x + 2y^3) \frac{\partial}{\partial y}(x + 2y^3) = 12(x + 2y^3)y^2, \quad (5.42)$$

— мы используем для их вычисления цепное правило (5.32)<sup>1</sup>.

**ПРИМЕЧАНИЕ** Нередко в литературе градиент определяется как вектор-столбец, в соответствии с наиболее общепринятым способом записи векторов. Мы определяем градиент как вектор-строку по двум причинам. Во-первых, это делает более логичным обобщение градиента на векторнозначные функции  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$  (тогда он становится матрицей). Во-вторых, тогда мы можем применять многомерное цепное правило вне зависимости от размерности градиента. Оба эти соображения мы обсудим в разделе 5.3. ♦

### Пример 5.7 (градиент)

Для функции  $f(x_1, x_2) = x_1^2 x_2 + x_1 x_2^3 \in \mathbb{R}$  частные производные (то есть производные  $f$  относительно  $x_1$  и  $x_2$ ) равны

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 2x_1 x_2 + x_2^3; \quad (5.43)$$

<sup>1</sup> Можно использовать правила для скалярного дифференцирования: каждая частная производная — это производная относительно скаляра.

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = x_1^2 + 3x_1x_2^2 \quad (5.44)$$

и градиент соответственно равен

$$\frac{\partial f}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f(x_1, x_2)}{\partial x_1} & \frac{\partial f(x_1, x_2)}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1x_2 + x_2^3 & x_1^2 + 3x_1x_2^2 \end{bmatrix} \in \mathbb{R}^{1 \times 2}. \quad (5.45)$$

### 5.2.1. Основные правила взятия частных производных

В случае функции нескольких переменных, когда  $\mathbf{x} \in \mathbb{R}^n$ , по-прежнему применимы основные правила дифференцирования (для произведения, для суммы, для композиции), известные нам со школы. Однако при вычислении производных относительно  $\mathbf{x} \in \mathbb{R}^n$  необходимо быть внимательными: градиенты теперь являются векторами или матрицами, а умножение матриц некоммутативно (раздел 2.2.1), то есть порядок множителей важен.

Вот как обобщаются правила вычисления производных произведения, суммы и композиции<sup>1</sup>:

$$\text{Производная произведения: } \frac{\partial}{\partial \mathbf{x}}(f(\mathbf{x})g(\mathbf{x})) = \frac{\partial f}{\partial \mathbf{x}}g(\mathbf{x}) + f(\mathbf{x})\frac{\partial g}{\partial \mathbf{x}}. \quad (5.46)$$

$$\text{Производная суммы: } \frac{\partial}{\partial \mathbf{x}}(f(\mathbf{x}) + g(\mathbf{x})) = \frac{\partial f}{\partial \mathbf{x}} + \frac{\partial g}{\partial \mathbf{x}}. \quad (5.47)$$

$$\text{Цепное правило (производная композиции): } \frac{\partial}{\partial \mathbf{x}}(g \circ f)(\mathbf{x}) = \frac{\partial}{\partial \mathbf{x}}(g(f(\mathbf{x}))) = \frac{\partial g}{\partial f} \frac{\partial f}{\partial \mathbf{x}}. \quad (5.48)$$

Рассмотрим цепное правило подробнее. Правило (5.48) несколько напоминает правило умножения матриц, где для существования произведения размеры матриц должны быть согласованы (раздел 2.2.1). Свойства цепного правила похожи: проходя по нему слева направо, мы встречаем  $\partial f$  в «знаменателе» первого сомножителя и в «числителе» второго<sup>2</sup>. Умножение определено:  $\partial f$  «сокращается», остается  $\partial g / \partial \mathbf{x}$ .

<sup>1</sup> Производная произведения:  $(fg)' = f'g + fg'$ , производная суммы:  $(f + g)' = f' + g'$ , производная композиции:  $(g(f))' = g'(f)f'$ .

<sup>2</sup> На самом деле такая аналогия некорректна: производная — это не дробь.

### 5.2.2. Цепное правило

Рассмотрим функцию  $f: \mathbb{R}^2 \rightarrow \mathbb{R}$  от двух переменных  $x_1, x_2$ , которые, в свою очередь, являются функциями от  $t: x_1(t), x_2(t)$ . Чтобы найти градиент  $f$  относительно  $t$ , мы применяем цепное правило (5.48) для функций нескольких переменных:

$$\frac{df}{dt} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1(t)}{\partial t} \\ \frac{\partial x_2(t)}{\partial t} \end{bmatrix} = \frac{\partial f}{\partial x_1} \frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2} \frac{\partial x_2}{\partial t}, \quad (5.49)$$

где  $d$  обозначает градиент, а  $\partial$  — частную производную.

#### Пример 5.8

Рассмотрим функцию  $f(x_1, x_2) = x_1^2 + 2x_2$ , где  $x_1 = \sin(t)$  и  $x_2 = \cos(t)$ . Тогда

$$\frac{df}{dt} = \frac{\partial f}{\partial x_1} \frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2} \frac{\partial x_2}{\partial t} = \quad (5.50a)$$

$$= 2 \sin t \frac{\partial \sin t}{\partial t} + 2 \frac{\partial \cos t}{\partial t} = \quad (5.50b)$$

$$= 2 \sin t \cos t - 2 \sin t = 2 \sin t (\cos t - 1) \quad (5.50c)$$

— производная  $f$  относительно  $t$ .

Если  $f(x_1, x_2)$  — функция от  $x_1$  и  $x_2$ , где  $x_1(s, t)$  и  $x_2(s, t)$  — функции двух переменных  $s$  и  $t$ , по цепному правилу вычисляем производные

$$\frac{\partial f}{\partial s} = \frac{\partial f}{\partial x_1} \frac{\partial x_1}{\partial s} + \frac{\partial f}{\partial x_2} \frac{\partial x_2}{\partial s}; \quad (5.51)$$

$$\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x_1} \frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2} \frac{\partial x_2}{\partial t}, \quad (5.52)$$

и градиент получается умножением матриц

$$\frac{df}{d(s, t)} = \frac{\partial f}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial (s, t)} = \underbrace{\begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{bmatrix}}_{\frac{\partial f}{\partial \mathbf{x}}} \underbrace{\begin{bmatrix} \frac{\partial x_1}{\partial s} & \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial s} & \frac{\partial x_2}{\partial t} \end{bmatrix}}_{\frac{\partial \mathbf{x}}{\partial (s, t)}}. \quad (5.53)$$

Этот краткий способ записи цепного правила как умножения матриц имеет смысл только в том случае, когда градиент записывается в виде строки. Иначе для согласованности размеров матриц нам пришлось бы транспонировать градиент. Пока градиент является вектором или матрицей, это просто, но когда градиент становится тензором (что мы обсудим в дальнейшем), транспонирование становится нетривиальной задачей.

**ПРИМЕЧАНИЕ** Определение частной производной как предела соответствующего разностного отношения (5.39) можно использовать для проверки, правильно ли реализовано нахождение градиентов в компьютерной программе. С помощью конечных разностей можно протестировать процедуру нахождения градиентов: взять маленькое  $h$  (например,  $h = 10^{-4}$ ) и сравнить приближение конечными разностями (5.39) с нашей аналитической реализацией градиента. Если ошибка мала, наша реализация градиента, скорее всего, верна. «Малой» ошибкой можно считать, например  $\sqrt{\frac{\sum_i (dh_i - df_i)^2}{\sum_i (dh_i + df_i)^2}} < 10^{-6}$ , где  $dh_i$  — приближение конечными разностями, а  $df_i$  — аналитические значения частной производной относительно переменной  $x_i$ . ◆

## 5.3. ГРАДИЕНТЫ ВЕКТОРНОЗНАЧНЫХ ФУНКЦИЙ

До сих пор мы обсуждали частные производные и градиенты функций  $f: \mathbb{R}^n \rightarrow \mathbb{R}$ , принимающих вещественные значения. В данном разделе мы обобщим понятие градиента на векторнозначные функции (векторные поля)  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$ , где  $n \geq 1$  и  $m > 1$ .

Функция  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$  и вектору  $\mathbf{x} = [x_1, \dots, x_n]^T \in \mathbb{R}^n$  соответствует вектор значений функции

$$f(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{bmatrix} \in \mathbb{R}^m. \quad (5.54)$$

Запись векторнозначной функции в таком виде позволяет нам рассматривать векторнозначную функцию  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$  как вектор  $[f_1, \dots, f_m]^T$  из функций  $f_i: \mathbb{R}^n \rightarrow \mathbb{R}$ , принимающих вещественные значения. Каждая из функций  $f_i$  дифференцируется по правилам из раздела 5.2.

Таким образом, частная производная векторнозначной функции  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$  относительно  $x_i \in \mathbb{R}$ ,  $i = 1, \dots, n$ , задается формулой

$$\frac{df}{dx_i} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} \\ \vdots \\ \frac{\partial f_m}{\partial x_i} \end{bmatrix} = \begin{bmatrix} \lim_{h \rightarrow 0} \frac{f_1(x_1, \dots, x_{i-1}, x_i + h, x_{i+1}, \dots, x_n) - f_1(x)}{h} \\ \vdots \\ \lim_{h \rightarrow 0} \frac{f_m(x_1, \dots, x_{i-1}, x_i + h, x_{i+1}, \dots, x_n) - f_m(x)}{h} \end{bmatrix} \in \mathbb{R}^m. \quad (5.55)$$

По формуле (5.40) градиент функции  $f$  относительно вектора — вектор-строка из частных производных. В формуле (5.55) каждая частная производная  $\partial f / \partial x_i$  является вектором-столбцом. Собрав вместе все эти частные производные, получим градиент  $\mathbf{f}: \mathbb{R}^n \rightarrow \mathbb{R}^m$  относительно  $\mathbf{x} \in \mathbb{R}^n$ :

$$\frac{d\mathbf{f}(\mathbf{x})}{d\mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_n} \end{bmatrix} = \quad (5.56a)$$

$$= \begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial f_m(\mathbf{x})}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m \times n}. \quad (5.56b)$$

**Определение 5.6 (матрица Якоби).** Набор всех частных производных первого порядка векторнозначной функции  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$  называется матрицей Якоби. Матрица Якоби  $J$  — это матрица размера  $m \times n$ , определяемая как

$$J = \nabla_x f = \frac{d\mathbf{f}(\mathbf{x})}{d\mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_n} \end{bmatrix} = \quad (5.57)$$

$$= \begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial f_m(\mathbf{x})}{\partial x_n} \end{bmatrix}, \quad (5.58)$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad J(i, j) = \frac{\partial f_i}{\partial x_j}. \quad (5.59)$$

В частном случае (5.58) функции  $f: \mathbb{R}^n \rightarrow \mathbb{R}^1$ , отображающей вектор  $\mathbf{x} \in \mathbb{R}^n$  в скаляр (например,  $f(\mathbf{x}) = \sum_{i=1}^n x_i$ ), по формуле (5.58) получаем вектор-строку (матрицу размера  $1 \times n$ ), см. (5.40).

**ПРИМЕЧАНИЕ** В нашей книге мы используем способ записи производной векторнозначной функции  $f$ , при котором компоненты  $f$  соответствуют строкам, а компоненты  $x$  — столбцам. Существует и другой способ, при котором мы получаем транспонированную матрицу по отношению к описанной. ♦

В разделе 6.7 мы увидим, как использовать матрицу Якоби в методе замены переменной для распределения вероятностей. Ее детерминант (якобиан) задает коэффициент, на который придется домножать при замене переменной.

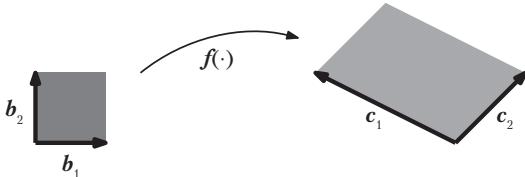
Из раздела 4.1 мы узнали, что с помощью детерминанта можно вычислить площадь параллелограмма. Если  $\mathbf{b}_1 = [1, 0]^T$ ,  $\mathbf{b}_2 = [0, 1]^T$  — стороны единичного квадрата (слева на рис. 5.5), площадь этого квадрата равна

$$\left| \det \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right| = 1. \quad (5.60)$$

Для параллелограмма  $\mathbf{c}_1 = [-2, 1]^T$ ,  $\mathbf{c}_2 = [1, 1]^T$  (справа на рис. 5.5) площадь равна абсолютной величине детерминанта (раздел 4.1):

$$\left| \det \begin{pmatrix} -2 & 1 \\ 1 & 1 \end{pmatrix} \right| = |-3| = 3, \quad (5.61)$$

то есть его площадь в три раза больше площади единичного квадрата. Этот множитель можно найти, рассмотрев отображение, переводящее единичный квадрат в параллелограмм. На языке линейной алгебры, мы переходим от базиса  $(\mathbf{b}_1, \mathbf{b}_2)$  к  $(\mathbf{c}_1, \mathbf{c}_2)$ . Замена переменных линейна, и абсолютная величина детерминанта отображения дает в точности искомый множитель.



**Рис. 5.5.** Якобиан  $f$  можно использовать для нахождения/отыскания, во сколько раз площадь фигуры справа больше, чем фигуры слева

Рассмотрим два подхода к поиску такого отображения. Первый подход использует его линейность, так что мы можем применять инструменты из главы 2. Второй, основанный на частных производных, использует материал из данной главы.

**Подход 1.** Подходя с точки зрения линейной алгебры, мы для начала отметим, что  $\{\mathbf{b}_1, \mathbf{b}_2\}$  и  $\{\mathbf{c}_1, \mathbf{c}_2\}$  – базисы  $\mathbb{R}^2$  (раздел 2.6.1). Фактически мы осуществляем замену базиса  $(\mathbf{b}_1, \mathbf{b}_2)$  на  $(\mathbf{c}_1, \mathbf{c}_2)$ , и нужно найти матрицу перехода. Результаты раздела 2.7.2 позволяют нам найти матрицу замены базиса:

$$\mathbf{J} = \begin{bmatrix} -2 & 1 \\ 1 & 1 \end{bmatrix}, \quad (5.62)$$

где  $\mathbf{J}\mathbf{b}_1 = \mathbf{c}_1$  и  $\mathbf{J}\mathbf{b}_2 = \mathbf{c}_2$ . Абсолютная величина детерминанта  $J$ , дающая искомый множитель, равна  $|\det(\mathbf{J})| = 3$ , то есть площадь параллелограмма со сторонами  $(\mathbf{c}_1, \mathbf{c}_2)$  втрое больше, чем квадрата со сторонами  $(\mathbf{b}_1, \mathbf{b}_2)$ .

**Подход 2.** Описанный ранее подход работает для линейных преобразований, для нелинейных же преобразований (которые понадобятся нам в разделе 6.7) будем следовать более общему методу, использующему частные производные. Рассмотрим функцию  $f: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ , осуществляющую замену переменных. В нашем примере  $f$  сопоставляет координатам вектора  $\mathbf{x} \in \mathbb{R}^2$  в базисе  $(\mathbf{b}_1, \mathbf{b}_2)$  его координаты в базисе. Нам хочется описать это отображение так, чтобы можно было вычислить изменение площади (или объема) при его применении. Для этого надо выяснить, как меняется  $f(\mathbf{x})$  при малом изменении  $x$ . На этот вопрос легко ответить, используя матрицу Якоби  $\frac{df}{dx} \in \mathbb{R}^{2 \times 2}$ . Так как можно записать

$$y_1 = -2x_1 + x_2; \quad (5.63)$$

$$y_2 = x_1 + x_2, \quad (5.64)$$

получаем функциональную зависимость между  $\mathbf{x}$  и  $\mathbf{y}$ , что позволяет нам вычислить частные производные

$$\frac{\partial y_1}{\partial x_1} = -2, \quad \frac{\partial y_1}{\partial x_2} = 1, \quad \frac{\partial y_2}{\partial x_1} = 1, \quad \frac{\partial y_2}{\partial x_2} = 1 \quad (5.65)$$

и записать матрицу Якоби

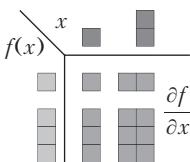
$$\mathbf{J} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 1 & 1 \end{bmatrix}. \quad (5.66)$$

Матрица Якоби описывает искомое преобразование координат. Оно является точным, если преобразование координат линейно (как в нашем случае), и (5.66) дает матрицу перехода от базиса к базису (5.62). Если же преобразование коор-

динат нелинейно, матрица Якоби локально приближает его линейным преобразованием. Абсолютная величина якобиана  $|\det(\mathbf{J})|$  будет множителем, на который изменится площадь или объем при замене координат. В нашем примере  $|\det(\mathbf{J})|=3$ .

Якобиан и замены переменных будут нужны нам в разделе 6.7 для работы со случайными величинами и распределениями вероятностей. Эти замены чрезвычайно важны для машинного обучения, а именно для глубоких нейросетей при проведении репараметризации.

В данной главе мы познакомились с производными функций. Рисунок 5.6 иллюстрирует их размерности.



**Рис. 5.6.** Размерности частных производных

Если  $f: \mathbb{R} \rightarrow \mathbb{R}$ , то градиент — это просто скаляр (на рисунке в левом верхнем углу). Для  $f: \mathbb{R}^D \rightarrow \mathbb{R}$  градиентом будет вектор-строка (матрица размера  $1 \times D$ ), для  $f: \mathbb{R} \rightarrow \mathbb{R}^E$  — вектор-столбец (матрица размера  $E \times 1$ ). Наконец, если  $f: \mathbb{R}^D \rightarrow \mathbb{R}^E$ , градиент будет матрицей размера  $E \times D$ .

### Пример 5.9 (градиент векторнозначной функции)

Пусть нам даны

$$\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x}, \mathbf{f}(\mathbf{x}) \in \mathbb{R}^M, \mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{x} \in \mathbb{R}^N.$$

Чтобы вычислить градиент  $d\mathbf{f}/d\mathbf{x}$ , сначала определим его размерность. Мы знаем, что  $\mathbf{f}: \mathbb{R}^N \rightarrow \mathbb{R}^M$ , значит  $d\mathbf{f}/d\mathbf{x} \in \mathbb{R}^{M \times N}$ . Затем найдем частные производные функции  $f$  относительно каждой из переменных  $x_i$ :

$$f_i(\mathbf{x}) = \sum_{j=1}^N A_{ij} x_j \Rightarrow \frac{\partial f_i}{\partial x_i} = A_{ii}. \quad (5.67)$$

Запишем частные производные в виде матрицы Якоби и получим градиент

$$\frac{d\mathbf{f}}{d\mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_N} \\ \vdots & & \vdots \\ \frac{\partial f_M}{\partial x_1} & \dots & \frac{\partial f_M}{\partial x_N} \end{bmatrix} = \begin{bmatrix} A_{11} & \dots & A_{1N} \\ \vdots & & \vdots \\ A_{M1} & \dots & A_{MN} \end{bmatrix} = \mathbf{A} \in \mathbb{R}^{M \times N}. \quad (5.68)$$

**Пример 5.10 (цепное правило)**

Рассмотрим функцию  $h: \mathbb{R} \rightarrow \mathbb{R}$ ,  $h(t) = (f \circ g)(t)$ , где

$$f: \mathbb{R}^2 \rightarrow \mathbb{R}; \quad (5.69)$$

$$g: \mathbb{R} \rightarrow \mathbb{R}^2; \quad (5.70)$$

$$f(\mathbf{x}) = \exp(x_1 x_2^2); \quad (5.71)$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = g(t) = \begin{bmatrix} t \cos t \\ t \sin t \end{bmatrix} \quad (5.72)$$

и найдем градиент  $f$  относительно  $t$ . Так как  $f: \mathbb{R}^2 \rightarrow \mathbb{R}$  и  $g: \mathbb{R} \rightarrow \mathbb{R}^2$ , то

$$\frac{\partial f}{\partial \mathbf{x}} \in \mathbb{R}^{1 \times 2}, \quad \frac{\partial g}{\partial t} \in \mathbb{R}^{2 \times 1}. \quad (5.73)$$

Получаем искомый градиент, применяя цепное правило:

$$\frac{dh}{dt} = \frac{\partial f}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial t} = \left[ \frac{\partial f}{\partial x_1} \quad \frac{\partial f}{\partial x_2} \right] \begin{bmatrix} \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial t} \end{bmatrix} = \quad (5.74a)$$

$$= \begin{bmatrix} \exp(x_1 x_2^2) x_2^2 & 2 \exp(x_1 x_2^2) x_1 x_2 \end{bmatrix} \begin{bmatrix} \cos t - t \sin t \\ \sin t + t \cos t \end{bmatrix} = \quad (5.74b)$$

$$= \exp(x_1 x_2^2) (x_2^2 (\cos t - t \sin t) + 2 x_1 x_2 (\sin t + t \cos t)), \quad (5.74c)$$

где  $x_1 = t \cos t$  и  $x_2 = t \sin t$  (5.72).

**Пример 5.11 (градиенты квадратичной функции потерь в линейной модели)**

Рассмотрим линейную модель<sup>1</sup>

$$\mathbf{y} = \Phi \theta, \quad (5.75)$$

где  $\theta \in \mathbb{R}^D$  — вектор параметров,  $\Phi \in \mathbb{R}^{N \times D}$  — признаки, а  $\mathbf{y} \in \mathbb{R}^N$  — соответствующие наблюдения. Определим функции

<sup>1</sup> Эту модель мы рассмотрим подробнее в главе 9, где нам понадобятся производные функции потерь относительно параметров.

$$L(\mathbf{e}) := \|\mathbf{e}\|^2; \quad (5.76)$$

$$\mathbf{e}(\theta) = \mathbf{y} - \Phi\theta. \quad (5.77)$$

Мы хотим найти  $\frac{\partial L}{\partial \theta}$  и будем использовать для этого цепное правило.  $L$  называется квадратичной функцией потерь.

Сначала определим размерность градиента:

$$\frac{\partial L}{\partial \theta} \in \mathbb{R}^{1 \times D}. \quad (5.78)$$

Цепное правило позволяет нам вычислить его по формуле

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \theta}, \quad (5.79)$$

где  $d$ -й компонент равен

$$\frac{\partial L}{\partial \theta}[1, d] = \sum_{n=1}^N \frac{\partial L}{\partial \mathbf{e}}[n] \frac{\partial \mathbf{e}}{\partial \theta}[n, d]. \quad (5.80)$$

Мы знаем, что  $\|\mathbf{e}\|^2 = \mathbf{e}^T \mathbf{e}$  (раздел 3.2), тогда

$$\frac{\partial L}{\partial \mathbf{e}} = 2\mathbf{e}^T \in \mathbb{R}^{1 \times N}. \quad (5.81)$$

Далее получаем

$$\frac{\partial \mathbf{e}}{\partial \theta} = -\Phi \in \mathbb{R}^{N \times D}, \quad (5.82)$$

и искомая производная равна

$$\frac{\partial L}{\partial \theta} = -2\mathbf{e}^T \Phi \stackrel{(5.77)}{=} -2 \underbrace{(\mathbf{y}^T - \theta^T \Phi^T)}_{1 \times N} \underbrace{\Phi}_{N \times D} \in \mathbb{R}^{1 \times D}. \quad (5.83)$$

**ПРИМЕЧАНИЕ** Тот же результат можно было получить и без применения цепного правила, сразу рассмотрев функцию

$$L_2(\theta) := \|\mathbf{y} - \Phi\theta\|^2 = (\mathbf{y} - \Phi\theta)^T(\mathbf{y} - \Phi\theta). \quad (5.84)$$

Этот подход удобен для простых функций вроде  $L_2$ , но плохо подходит для сложных композиций. ◆

## 5.4. ГРАДИЕНТЫ МАТРИЦ

Нам встретятся ситуации, в которых понадобится вычислять градиенты матриц относительно векторов или других матриц, получая в итоге тензоры. Такой тензор можно воспринимать как многомерный массив частных производных. Например, чтобы вычислить градиент  $m \times n$ -матрицы  $\mathbf{A}$  относительно  $p \times q$ -матрицы  $\mathbf{B}$ , матрица Якоби будет размера  $(m \times n) \times (p \times q)$ , то есть окажется четырехмерным тензором  $\mathbf{J}$ , элементы которого равны  $J_{ijkl} = \partial A_{ij} / \partial B_{kl}$ .

Матрицы соответствуют линейным отображениям. Можно воспользоваться изоморфизмом векторных пространств (то есть обратимым линейным отображением между ними) — пространства  $m \times n$ -матриц  $\mathbb{R}^{m \times n}$  и пространства  $mn$ -векторов  $\mathbb{R}^{mn}$ . Таким образом, наши матрицы превратятся в векторы длин  $mn$  и  $pq$  соответственно. Градиентом, таким образом, окажется матрица размера  $mn \times pq$ . На рис. 5.7 показаны оба подхода. На практике часто разумно превратить матрицу в вектор<sup>1</sup> и дальше работать с его матрицей Якоби. Цепное правило (5.48) тогда сводится к умножению матриц, в то время как при работе с тензорами Якоби понадобится внимательно следить за размерностями.

### Пример 5.12 (градиенты векторов относительно матриц)

Рассмотрим такой пример:

$$\mathbf{f} = \mathbf{Ax}, \mathbf{f} \in \mathbb{R}^M, \mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{x} \in \mathbb{R}^N. \quad (5.85)$$

Пусть мы хотим найти градиент  $d\mathbf{f}/d\mathbf{A}$ . Начнем с выяснения его размерности:

$$\frac{d\mathbf{f}}{d\mathbf{A}} \in \mathbb{R}^{M \times (M \times N)}. \quad (5.86)$$

По определению, градиент будет набором частных производных

$$\frac{d\mathbf{f}}{d\mathbf{A}} = \begin{bmatrix} \frac{\partial f_1}{\partial \mathbf{A}} \\ \vdots \\ \frac{\partial f_M}{\partial \mathbf{A}} \end{bmatrix}, \frac{df_i}{d\mathbf{A}} \in \mathbb{R}^{1 \times (M \times N)}. \quad (5.87)$$

<sup>1</sup> Матрицу можно превратить в вектор, записав ее столбцы друг под другом (векторизация матрицы).

Чтобы вычислить эти частные производные, вспомним формулу умножения матрицы на вектор:

$$f_i = \sum_{j=1}^N A_{ij} x_j, \quad i=1, \dots, M. \quad (5.88)$$

Тогда частные производные равны

$$\frac{\partial f_i}{\partial A_{iq}} = x_q. \quad (5.89)$$

Это позволит нам вычислить частные производные  $f_i$  относительно строки  $A$

$$\frac{\partial f_i}{\partial A_{i,:}} = \mathbf{x}^T \in \mathbb{R}^{1 \times 1 \times N}; \quad (5.90)$$

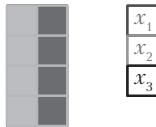
$$\frac{\partial f_i}{\partial A_{k \neq i,:}} = \mathbf{0}^T \in \mathbb{R}^{1 \times 1 \times N}; \quad (5.91)$$

здесь важно следить за размерностью! Так как  $f_i$  принимают значения в  $\mathbb{R}$ , а каждая строка матрицы  $A$  имеет размер  $1 \times N$ , частной производной  $f_i$  относительно строки  $A$  будет тензор размера  $1 \times 1 \times N$ .

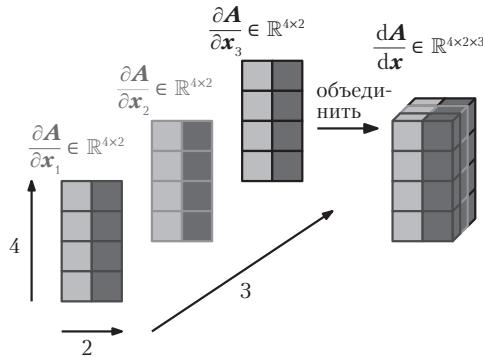
Соберем вместе частные производные из (5.91) и получим искомый градиент (5.87):

$$\frac{\partial f_i}{\partial A} = \begin{bmatrix} \mathbf{0}^T \\ \vdots \\ \mathbf{0}^T \\ \mathbf{x}^T \\ \mathbf{0}^T \\ \vdots \\ \mathbf{0}^T \end{bmatrix} \in \mathbb{R}^{1 \times (M \times N)}. \quad (5.92)$$

$$\mathbf{A} \in \mathbb{R}^{4 \times 2} \quad \mathbf{x} \in \mathbb{R}^3$$



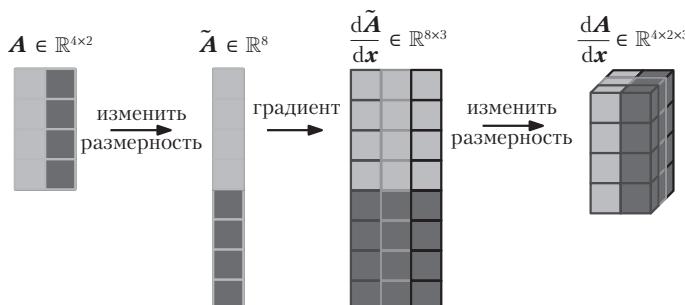
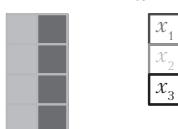
Частные производные:



(a) Подход 1: вычисляем частные производные  $\partial A / \partial x_1, \partial A / \partial x_2, \partial A / \partial x_3$ , каждая из которых будет матрицей  $4 \times 2$ , и объединяем их в тензор  $4 \times 2 \times 3$

**Рис. 5.7.** Вычисление градиента матрицы относительно вектора. Мы хотим вычислить градиент  $\mathbf{A} \in \mathbb{R}^{4 \times 2}$  относительно вектора  $\mathbf{x} \in \mathbb{R}^3$ . Мы знаем, что градиент  $\frac{d\mathbf{A}}{d\mathbf{x}} \in \mathbb{R}^{4 \times 2 \times 3}$ . Используем два эквивалентных подхода:  
(a) собираем частные производные в тензор Якоби;  
(b) преобразуем матрицу в вектор, ищем матрицу Якоби, преобразуем ее в тензор

$$\mathbf{A} \in \mathbb{R}^{4 \times 2} \quad \mathbf{x} \in \mathbb{R}^3$$



(b) Подход 2: превращаем  $A \in \mathbb{R}^{4 \times 2}$  в вектор  $\tilde{A} \in \mathbb{R}^8$ , вычисляем градиент  $\partial \tilde{A} / \partial x$  и делаем из него тензор

**Пример 5.13 (градиент матрицы относительно матрицы)**

Рассмотрим матрицу  $\mathbf{R} \in \mathbb{R}^{M \times N}$  и  $f: \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^{N \times N}$ , где

$$f(\mathbf{R}) = \mathbf{R}^T \mathbf{R} =: \mathbf{K} \in \mathbb{R}^{N \times N}, \quad (5.93)$$

и будем искать градиент  $d\mathbf{K} / d\mathbf{R}$ .

Приступая к этой непростой задаче, запишем, что мы знаем. Градиент имеет размерность

$$\frac{d\mathbf{K}}{d\mathbf{R}} \in \mathbb{R}^{(N \times N) \times (M \times N)}, \quad (5.94)$$

то есть является тензором. Далее,

$$\frac{dK_{pq}}{d\mathbf{R}} \in \mathbb{R}^{1 \times M \times N} \quad (5.95)$$

для  $p, q = 1, \dots, N$ , где  $K_{pq}$  — это  $(p, q)$ -й элемент  $\mathbf{K} = f(\mathbf{R})$ . Обозначим  $i$ -й столбец  $\mathbf{R}$  через  $\mathbf{r}_i$  и заметим, что каждый элемент  $\mathbf{K}$  — скалярное произведение двух столбцов  $\mathbf{R}$ , то есть

$$K_{pq} = \mathbf{r}_p^T \mathbf{r}_q = \sum_{m=1}^M R_{mp} R_{mq}. \quad (5.96)$$

Посчитав частную производную  $\frac{\partial K_{pq}}{\partial R_{ij}}$ , получим

$$\frac{\partial K_{pq}}{\partial R_{ij}} = \sum_{m=1}^M \frac{\partial}{\partial R_{ij}} R_{mp} R_{mq} = \partial_{pqij}, \quad (5.97)$$

$$\partial_{pqij} = \begin{cases} R_{iq} & \text{если } j = p, p \neq q \\ R_{ip} & \text{если } j = q, p \neq q \\ 2R_{iq} & \text{если } j = p, p \neq q \\ 0 & \text{в остальных случаях.} \end{cases} \quad (5.98)$$

Из формулы (5.94) знаем, что размерность искомого градиента —  $(N \times N) \times (M \times N)$ , а каждый элемент этого тензора равен  $\partial_{pqij}$  из (5.98), где  $p, q, j = 1, \dots, N$  и  $i = 1, \dots, M$ .

## 5.5. ПОЛЕЗНЫЕ ТОЖДЕСТВА ДЛЯ ВЫЧИСЛЕНИЯ ГРАДИЕНТОВ

В этом разделе мы перечислим некоторые полезные тождества, часто требующиеся в машинном обучении (Petersen and Pedersen, 2012). Здесь  $\text{tr}(\cdot)$  — след матрицы (см. определение 4.4),  $\det(\cdot)$  — ее детерминант (раздел 4.1) и  $f(\mathbf{X})^{-1}$  — обратная функция к  $f(\mathbf{X})$  (если она существует).

$$\frac{\partial}{\partial \mathbf{X}} f(\mathbf{X})^T = \left( \frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} \right)^T; \quad (5.99)$$

$$\frac{\partial}{\partial \mathbf{X}} \text{tr}(f(\mathbf{X})) = \text{tr}\left( \frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} \right); \quad (5.100)$$

$$\frac{\partial}{\partial \mathbf{X}} \det(f(\mathbf{X})) = \det(f(\mathbf{X})) \text{tr}\left( f(\mathbf{X})^{-1} \frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} \right); \quad (5.101)$$

$$\frac{\partial}{\partial \mathbf{X}} f(\mathbf{X})^{-1} = -f(\mathbf{X})^{-1} \frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} f(\mathbf{X})^{-1}; \quad (5.102)$$

$$\frac{\partial \mathbf{a}^T \mathbf{X}^{-1} \mathbf{b}}{\partial \mathbf{X}} = -(\mathbf{X}^{-1})^T \mathbf{a} \mathbf{b}^T (\mathbf{X}^{-1})^T; \quad (5.103)$$

$$\frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} = \mathbf{a}^T; \quad (5.104)$$

$$\frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a}^T; \quad (5.105)$$

$$\frac{\partial \mathbf{a}^T \mathbf{X} \mathbf{b}}{\partial \mathbf{X}} = \mathbf{a} \mathbf{b}^T; \quad (5.106)$$

$$\frac{\partial \mathbf{x}^T \mathbf{B} \mathbf{x}}{\partial \mathbf{x}} = \mathbf{x}^T (\mathbf{B} + \mathbf{B}^T); \quad (5.107)$$

$$\frac{\partial}{\partial \mathbf{s}} = (\mathbf{x} - \mathbf{A} \mathbf{s})^T \mathbf{W} (\mathbf{x} - \mathbf{A} \mathbf{s}) = -2(\mathbf{x} - \mathbf{A} \mathbf{s})^T \mathbf{W} \mathbf{A}. \quad (5.108)$$

для симметричной  $\mathbf{W}$

**ПРИМЕЧАНИЕ** В нашей книге мы обсуждали транспонирование и след только для матриц. Однако, как мы видим, производные могут быть тензорами большей размерности, так что обычные определения этих операций здесь не годятся. В этом случае след  $D \times D \times E \times F$  тензора будет  $E \times F$ -матрицей. Это частный случай тензорного сжатия. Аналогично при «транспонировании» тен-

зора мы меняем местами две первых размерности. В формулах (5.99)–(5.102), если нам нужно вычислять производные векторнозначных функций  $\mathbf{f}(\cdot)$  относительно матриц (не превращая их в векторы, как в разделе 5.4), нам нужны тензоры. ♦

## 5.6. ОБРАТНОЕ РАСПРОСТРАНЕНИЕ ОШИБКИ И АВТОМАТИЧЕСКОЕ ДИФФЕРЕНЦИРОВАНИЕ

Во многих задачах машинного обучения мы ищем подходящие параметры модели методом градиентного спуска (раздел 7.1), основанным на вычислении градиента целевой функции относительно параметров модели. Для заданной целевой функции вычислить этот градиент можно аналитическими методами, применяя цепное правило (раздел 5.2.2). Мы уже видели пример в разделе 5.3, когда рассматривали градиент квадратичной ошибки относительно параметров линейной регрессионной модели.

Рассмотрим функцию

$$f(x) = \sqrt{x^2 + \exp(x^2)} + \cos(x^2 + \exp(x^2)). \quad (5.109)$$

Применяя цепное правило и пользуясь линейностью дифференцирования, получим

$$\begin{aligned} \frac{df}{dx} &= \frac{2x + 2x \exp(x^2)}{2\sqrt{x^2 + \exp(x^2)}} - \sin(x^2 + \exp(x^2))(2x + 2x \exp(x^2)) = \\ &= 2x \left( \frac{1}{2\sqrt{x^2 + \exp(x^2)}} - \sin(x^2 + \exp(x^2)) \right) (1 + \exp(x^2)). \end{aligned} \quad (5.110)$$

Записывать градиент в таком явном виде не всегда разумно — выражение для производной будет слишком длинным. На практике из-за этого нахождение формулы для градиента может быть значительно сложнее, чем вычисление функции, что требует лишних ресурсов. Для глубоких нейронных сетей эффективным методом вычисления градиента функции потерь относительно параметров модели является алгоритм *обратного распространения ошибки*<sup>1</sup> (Kelley, 1960; Bryson, 1961; Dreyfus, 1962; Rumelhart et al., 1986).

---

<sup>1</sup> Подробное обсуждение алгоритма обратного распространения ошибки и цепного правила можно найти в блоге Тима Виеры <https://tinyurl.com/yckm2ytw>.

### 5.6.1. Градиенты в глубоких нейронных сетях

Глубокое обучение — область, где цепное правило используется на каждом шагу, поскольку в нем значение функции  $y$  вычисляется как композиция большого количества функций

$$\mathbf{y} = (f_K \circ f_{K-1} \circ \dots \circ f_1)(\mathbf{x}) = f_K(f_{K-1}(\dots(f_1(\mathbf{x}))\dots)), \quad (5.111)$$

где  $x$  — входные переменные (например, изображения),  $y$  — выходы (например, метки классов), и каждая из функций  $f_i$ ,  $i = 1, \dots, K$ , имеет свои собственные параметры. В нейронных сетях с большим числом уровней на  $i$ -м уровне находятся функции  $f_i(\mathbf{x}_{i-1}) = \sigma(\mathbf{A}_{i-1} + \mathbf{b}_{i-1})$ . Здесь  $\mathbf{x}_{i-1}$  — выход  $i-1$ -го слоя, а  $\sigma$  — функция активации<sup>1</sup>, например логистический сигмоид  $\frac{1}{1+e^{-x}}$ ,  $\tanh$  или ReLU. Для обучения таких моделей нам нужен градиент функции потерь  $L$  относительно всех параметров модели  $\mathbf{A}_j, \mathbf{b}_j, j = 1, \dots, K$ . Это потребует вычисления градиента  $L$  относительно входов каждого из слоев. Например, если  $\mathbf{x}$  — входы,  $\mathbf{y}$  — выходы, а структура сети задается формулами

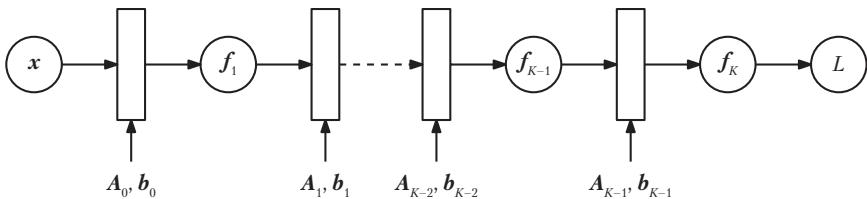
$$f_0 := \mathbf{x}; \quad (5.112)$$

$$f_i := \sigma_i(\mathbf{A}_{i-1} f_{i-1} + \mathbf{b}_{i-1}), i = 1, \dots, K, \quad (5.113)$$

(см. рис. 5.8), нам нужно найти такие  $\mathbf{A}_j, \mathbf{b}_j, j = 1, \dots, K$ , которые минимизируют квадратичную функцию потерь

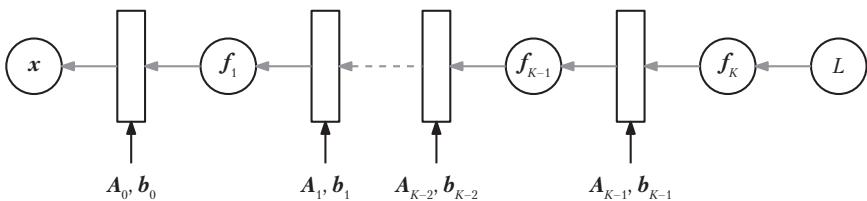
$$L(\theta) = \| \mathbf{y} - f_K(\theta, \mathbf{x}) \|^2, \quad (5.114)$$

где  $\theta = \{\mathbf{A}_0, \mathbf{b}_0, \dots, \mathbf{A}_{K-1}, \mathbf{b}_{K-1}\}$ .



**Рис. 5.8.** Прямой проход по глубокой нейронной сети, где ошибка  $L$  — функция входов  $\mathbf{x}$  и параметров  $\mathbf{A}_i, \mathbf{b}_i$

<sup>1</sup> Чтобы упростить обозначения, мы считаем функции активации в каждом слое одинаковыми.



**Рис. 5.9.** Обратный проход по глубокой нейронной сети для вычисления градиентов функции потерь

Чтобы найти градиент относительно набора параметров  $\theta$ , нужны частные производные функции  $L$  относительно параметров  $\theta_j = \{A_j, b_j\}$  для каждого уровня  $j = 0, \dots, K - 1$ . Мы находим их по цепному правилу<sup>1</sup>:

$$\frac{\partial L}{\partial \theta_{K-1}} = \frac{\partial L}{\partial f_K} \frac{\partial f_K}{\partial \theta_{K-1}}; \quad (5.115)$$

$$\frac{\partial L}{\partial \theta_{K-2}} = \frac{\partial L}{\partial f_K} \left[ \frac{\partial f_K}{\partial f_{K-1}} \frac{\partial f_{K-1}}{\partial \theta_{K-2}} \right]; \quad (5.116)$$

$$\frac{\partial L}{\partial \theta_{K-3}} = \frac{\partial L}{\partial f_K} \frac{\partial f_K}{\partial f_{K-1}} \left[ \frac{\partial f_{K-1}}{\partial f_{K-2}} \frac{\partial f_{K-2}}{\partial \theta_{K-3}} \right]; \quad (5.117)$$

$$\frac{\partial L}{\partial \theta_i} = \frac{\partial L}{\partial f_K} \frac{\partial f_K}{\partial f_{K-1}} \dots \left[ \frac{\partial f_{i+2}}{\partial f_{i+1}} \frac{\partial f_{i+1}}{\partial \theta_i} \right]. \quad (5.118)$$

Если мы нашли частные производные  $\partial L / \partial \theta_{i+1}$ , можно переиспользовать результаты предыдущих вычислений для нахождения  $\partial L / \partial \theta_i$ . Дополнительные слагаемые, которые нужно вычислить, обведены в рамку. На рис. 5.9 показано, как градиент движется обратно по сети.

## 5.6.2. Автоматическое дифференцирование

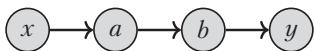
На самом деле, метод обратного распространения ошибки является частным случаем более общего численного метода, называемого *автоматическим дифференцированием*. Автоматическое дифференцирование можно считать набором приемов, позволяющих найти по возможности точное численное значение (а не символьную формулу) градиента или функции, применяя цепное правило. При автоматическом дифференцировании последовательно применяются элементарные арифметические операции (такие как сложение и умножение) и элементарные функции (например,  $\sin$ ,  $\cos$ ,  $\exp$ ,  $\log$ ). Применяя к этим операциям

<sup>1</sup> Более подробно градиенты в нейронных сетях обсуждаются в лекциях Дж. Домке: <https://tinyurl.com/yalcxg7v>.

цепное правило, можно автоматически вычислять градиенты весьма сложных функций. Автоматическое дифференцирование<sup>1</sup> реализуется в общем виде и может применяться как в прямом, так и в обратном порядке. В Baydin et al. (2018) дан замечательный обзор применений автоматического дифференцирования в машинном обучении.

На рис. 5.10 изображен график, показывающий, как мы приходим от входов  $x$  к выходам  $y$  через промежуточные значения  $a, b$ . Чтобы найти производную  $dy/dx$ , применим цепное правило и получим

$$\frac{dy}{dx} = \frac{dy}{db} \frac{db}{da} \frac{da}{dx}. \quad (5.119)$$



**Рис. 5.10.** Граф потока данных с входом  $x$ , выходом  $y$  и промежуточными значениями  $a, b$

Интуитивно понятно, что прямой и обратный порядок вычислений различаются порядком проведения умножения<sup>2</sup>. Благодаря ассоциативности умножения матриц можно выбирать между

$$\frac{dy}{dx} = \left( \frac{dy}{db} \frac{db}{da} \right) \frac{da}{dx} \text{ и} \quad (5.120)$$

$$\frac{dy}{dx} = \frac{dy}{db} \left( \frac{db}{da} \frac{da}{dx} \right). \quad (5.121)$$

Равенство (5.120) соответствует *обратному порядку*, так как градиенты распространяются по графу противоположно потоку данных. Равенство (5.121) описывает *прямой порядок*, при котором градиенты движутся по графу слева направо вместе с данными.

Далее мы сосредоточимся на автоматическом дифференцировании в обратном порядке, то есть на алгоритме обратного распространения ошибок. Для нейронных сетей, где размерность входа зачастую во много раз больше размерности выхода, обратный режим вычислительно много проще прямого.

Начнем с показательного примера.

<sup>1</sup> Автоматическое дифференцирование отличается от символьного дифференцирования и обычных численных методов нахождения градиента (например, через конечные разности).

<sup>2</sup> В общем случае матрица Якоби может быть вектором или, напротив, тензором Якоби.

**Пример 5.14**

Рассмотрим функцию

$$f(x) = \sqrt{x^2 + \exp(x^2)} + \cos(x^2 + \exp(x^2)) \quad (5.122)$$

из (5.109). Если нам придется реализовывать  $f$  программно, мы сможем ускорить вычисления, сохраняя *промежуточные значения*:

$$a = x^2; \quad (5.123)$$

$$b = \exp(a); \quad (5.124)$$

$$c = a + b; \quad (5.125)$$

$$d = \sqrt{c}; \quad (5.126)$$

$$e = \cos(c); \quad (5.127)$$

$$f = d + e. \quad (5.128)$$

Идея здесь такая же, как при использовании цепного правила. Заметим, что эти вычисления требуют меньше операций, чем реализация функции  $f(x)$  по определению (5.109). Соответствующий график вычислений на рис. 5.11 показывает поток данных и вычисления, благодаря которым мы получаем значения  $f$ .

Этот набор равенств, включающий промежуточные значения, можно рассмотреть как график вычислений. Такое представление часто используется в нейросетевых библиотеках. Производные промежуточных переменных относительно их входов можно вычислить напрямую по определениям производных элементарных функций. Получаем

$$\frac{\partial a}{\partial x} = 2x; \quad (5.129)$$

$$\frac{\partial b}{\partial a} = \exp(a); \quad (5.130)$$

$$\frac{\partial c}{\partial a} = 1 = \frac{\partial c}{\partial b}; \quad (5.131)$$

$$\frac{\partial d}{\partial c} = \frac{1}{2\sqrt{c}}; \quad (5.132)$$

$$\frac{\partial e}{\partial c} = -\sin(c); \quad (5.133)$$

$$\frac{\partial f}{\partial d} = 1 = \frac{\partial f}{\partial e}. \quad (5.134)$$

Рассмотрев граф вычислений с рис. 5.11, можно вычислить  $\partial f / \partial x$ , двигаясь от выхода назад:

$$\frac{\partial f}{\partial c} = \frac{\partial f}{\partial d} \frac{\partial d}{\partial c} + \frac{\partial f}{\partial e} \frac{\partial e}{\partial c}; \quad (5.135)$$

$$\frac{\partial f}{\partial b} = \frac{\partial f}{\partial c} \frac{\partial c}{\partial b}; \quad (5.136)$$

$$\frac{\partial f}{\partial a} = \frac{\partial f}{\partial b} \frac{\partial b}{\partial a} + \frac{\partial f}{\partial c} \frac{\partial c}{\partial a}; \quad (5.137)$$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial a} \frac{\partial a}{\partial x}. \quad (5.138)$$

Заметим, что для вычисления  $\partial f / \partial x$  мы неявно применяли цепное правило. Подставляя производные элементарных функций, получаем

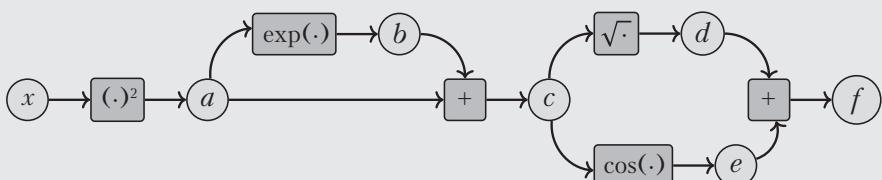
$$\frac{\partial f}{\partial c} = 1 \cdot \frac{1}{2\sqrt{c}} + 1 \cdot (-\sin(c)); \quad (5.139)$$

$$\frac{\partial f}{\partial b} = \frac{\partial f}{\partial c} \cdot 1; \quad (5.140)$$

$$\frac{\partial f}{\partial a} = \frac{\partial f}{\partial b} \exp(a) + \frac{\partial f}{\partial c} \cdot 1; \quad (5.141)$$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial a} \cdot 2x. \quad (5.142)$$

Принимая каждую из производных за новую переменную, видим, что вычисление производных имеет ту же сложность, что и всей функции. Это довольно контриинтуитивно, так как формула для производной  $\partial f / \partial x$  (5.110) выглядит существенно сложнее, чем для функции  $f(x)$  (5.109).



**Рис. 5.11.** Граф вычислений с входом  $x$ , выходом  $f$  и промежуточными значениями  $a, b, c, d, e$

Автоматическое дифференцирование формализует пример 5.14. Пусть  $x_1, \dots, x_d$  — входы функции,  $x_{d+1}, \dots, x_{D-1}$  — промежуточные переменные, а  $x_D$  — выход. Тогда график вычислений задается как

$$\text{для } i = d + 1, \dots, x_i = g_i(x_{\text{Pa}(x_i)}), \quad (5.143)$$

где  $g_i(\cdot)$  — элементарные функции, а  $x_{\text{Pa}(x_i)}$  — родительские узлы в графике для переменных  $x_i$ . Вычислить функцию, заданную таким образом, можно пошагово по цепному правилу. Так как по определению  $f = x_D$ , то

$$\frac{\partial f}{\partial x_D} = 1. \quad (5.144)$$

Применяем для других переменных  $x_i$  цепное правило:

$$\frac{\partial f}{\partial x_i} = \sum_{x_j: x_i \in \text{Pa}(x_j)} \frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial x_i} = \sum_{x_j: x_i \in \text{Pa}(x_j)} \frac{\partial f}{\partial x_j} \frac{\partial g_j}{\partial x_i}, \quad (5.145)$$

где  $\text{Pa}(x_j)$  — множество родительских узлов для  $x_j$  в графике вычислений. Уравнение (5.143) задает прямое распространение функции, а (5.145) — обратное распространение градиента по графу вычислений. При обучении нейронных сетей ошибка предсказания выхода распространяется обратно.

Автоматическое дифференцирование<sup>1</sup> применимо, когда функцию можно представить в виде графа вычислений, в котором элементарные функции дифференцируемы. На самом деле, функцией можно считать и компьютерную программу. Однако не все программы можно автоматически дифференцировать, например может не быть элементарных дифференцируемых функций. Такие конструкции программирования, как циклы и условные операторы, также требуют особого внимания.

## 5.7. ПРОИЗВОДНЫЕ ВЫСШИХ ПОРЯДКОВ

До сих пор мы обсуждали градиенты, то есть производные первого порядка. Иногда нам нужны производные высших порядков, например при использовании метода оптимизации Ньютона — второго порядка (Nocedal and Wright, 2006). В разделе 5.1.1 мы обсуждали приближение функций многочленами с помощью рядов Тейлора. То же самое можно сделать и в случае нескольких переменных. Начнем, однако, с обозначений.

---

<sup>1</sup> При автоматическом дифференцировании в обратном порядке необходимо строить дерево парсинга.

Рассмотрим функцию  $f: \mathbb{R}^2 \rightarrow \mathbb{R}$  от двух переменных  $x$  и  $y$ . Для частных производных и градиентов мы будем использовать следующие обозначения:

1.  $\frac{\partial^2 f}{\partial x^2}$  — вторая частная производная  $f$  относительно  $x$ .
2.  $\frac{\partial^n f}{\partial x^n}$  —  $n$ -я частная производная  $f$  относительно  $x$ .
3.  $\frac{\partial^2 f}{\partial y \partial x} = \frac{\partial}{\partial y} \left( \frac{\partial f}{\partial x} \right)$  — частная производная, полученная дифференцированием  $f$  сначала относительно  $x$ , а затем относительно  $y$ .
4.  $\frac{\partial^2 f}{\partial x \partial y}$  — частная производная, полученная дифференцированием  $f$  сначала относительно  $y$ , а затем относительно  $x$ .

Матрицей Гессе называется набор всех частных производных второго порядка. Если  $f(x, y)$  дважды непрерывно дифференцируема, то

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2 f}{\partial y \partial x}, \quad (5.146)$$

т. е. порядок взятия производных не важен, и соответствующая матрица Гессе

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \quad (5.147)$$

симметрична. Матрица Гессе обозначается  $\nabla_{x,y}^2 f(x, y)$ . Вообще говоря, для  $f: \mathbb{R}^n \rightarrow \mathbb{R}$  матрица Гессе имеет размер  $n \times n$ . Матрица Гессе характеризует выпуклость функции в окрестности  $(x, y)$ .

**ПРИМЕЧАНИЕ** Если  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$  — векторное поле, матрица Гессе будет тензором размера  $(m \times n \times n)$ . ◆

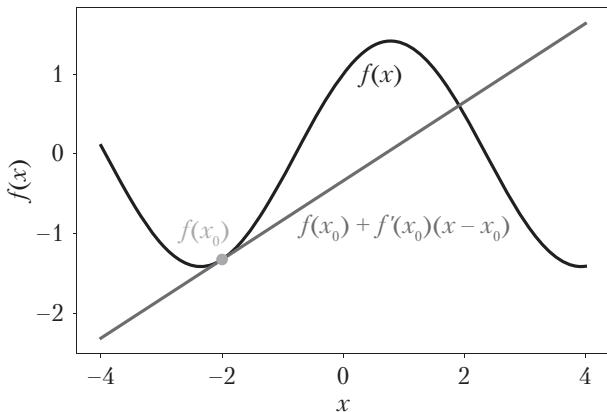
## 5.8. ЛИНЕАРИЗАЦИЯ И РЯДЫ ТЕЙЛОРА ДЛЯ НЕСКОЛЬКИХ ПЕРЕМЕННЫХ

Градиент  $\nabla f$  функции  $f$  часто используется для локально линейного приближения  $f$  в окрестности  $\mathbf{x}_0$ :

$$f(\mathbf{x}) \approx f(\mathbf{x}_0) + (\nabla_x f)(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0). \quad (5.148)$$

Здесь  $(\nabla_x f)(\mathbf{x}_0)$  — значение градиента функции  $f$  относительно  $\mathbf{x}$  в точке  $\mathbf{x}_0$ .

На рис. 5.12 изображено линейное приближение функции  $f$  в точке  $x_0$ .



**Рис. 5.12.** Линейное приближение функции. Исходная функция  $f$  аппроксимируется линейной в точке  $x_0 = -2$  использованием ряда Тейлора

Исходная функция аппроксимируется прямой линией. Это приближение локально довольно точно, но чем дальше мы удаляемся от  $\mathbf{x}_0$ , тем оно хуже. Равенство (5.148) является частным случаем разложения в ряд Тейлора для нескольких переменных, в котором мы рассматриваем только первые два слагаемых. Сейчас мы рассмотрим более общий случай, позволяющий достичь лучшего приближения.

**Определение 5.7 (ряд Тейлора от нескольких переменных).** Рассмотрим функцию

$$f: \mathbb{R}^D \rightarrow \mathbb{R}; \quad (5.149)$$

$$\mathbf{x} \mapsto f(\mathbf{x}), \mathbf{x} \in \mathbb{R}^D, \quad (5.150)$$

гладкую в точке  $\mathbf{x}_0$ . Если определить разностный вектор  $\delta = \mathbf{x} - \mathbf{x}_0$ , ряд Тейлора  $f$  в точке  $\mathbf{x}_0$  определяется как

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \frac{D_x^k f(\mathbf{x}_0)}{k!} \delta^k, \quad (5.151)$$

где  $D_x^k f(\mathbf{x}_0)$  — значение  $k$ -й полной производной функции  $f$  относительно  $\mathbf{x}$  в точке  $\mathbf{x}_0$ .

**Определение 5.8 (многочлен Тейлора).** Многочлен Тейлора степени  $n$  функции  $f$  в точке  $\mathbf{x}_0$  состоит из первых  $n + 1$  слагаемых ряда (5.151), то есть равен

$$T_n(\mathbf{x}) = \sum_{k=0}^n \frac{D_x^k f(\mathbf{x}_0)}{k!} \delta^k. \quad (5.152)$$

В формулах (5.151) и (5.152) мы использовали не вполне аккуратное обозначение  $\delta^k$ , не определявшееся для векторов  $\mathbf{x} \in \mathbb{R}^D$ ,  $D > 1$ ,  $k > 1$ . Заметим, что как  $D_x^k f$ , так и  $\delta^k$  — тензоры порядка  $k$ , то есть  $k$ -мерные массивы<sup>1</sup>. Тензор порядка  $k$   $\delta^k \in \mathbb{R}^{\overbrace{D \times D \times \dots \times D}^{k \text{ раз}}}$  равен  $k$ -й тензорной степени вектора  $x \in \mathbb{R}$ . Тензорное произведение обозначается  $\otimes$ . Например,

$$\delta^2 := \delta \otimes \delta = \delta \delta^T, \quad \delta^2[i, j] = \delta[i] \delta[j]; \quad (5.153)$$

$$\delta^3 := \delta \otimes \delta \otimes \delta, \quad \delta^3[i, j, k] = \delta[i] \delta[j] \delta[k]. \quad (5.154)$$

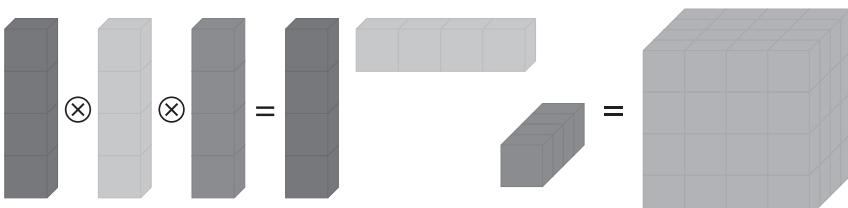
На рис. 5.13 изображены два таких тензорных произведения. В общем случае ряд Тейлора будет состоять из слагаемых

$$D_x^k f(\mathbf{x}_0) \delta^k = \sum_{i_1=1}^D \dots \sum_{i_k=1}^D D_x^k f(\mathbf{x}_0)[i_1, \dots, i_k] \delta[i_1] \dots \delta[i_k], \quad (5.155)$$

где  $D_x^k f(\mathbf{x}_0) \delta^k$  содержит многочлены степени  $k$ .



(a) Дан вектор  $\delta$ . Внешнее произведение  $\delta^2 := \delta \otimes \delta = \delta \delta^T$  будет матрицей



(b) Внешнее произведение  $\delta^3 := \delta \otimes \delta \otimes \delta$  будет тензором третьего порядка («трехмерной матрицей»), то есть массивом с тремя индексами

**Рис. 5.13.** Тензорные произведения. В тензорном произведении векторов каждый множитель увеличивает размерность на 1. (a) Тензорное произведение двух векторов будет матрицей. (b) Тензорное произведение трех векторов будет тензором порядка 3

<sup>1</sup> Вектор может быть реализован как одномерный массив, а матрица — как двумерный.

Теперь, определив ряды Тейлора для векторных полей, давайте выпишем в явном виде несколько первых слагаемых  $D_x^k f(\mathbf{x}_0) \delta^k$  ряда Тейлора для  $k = 0, \dots, 3$  и  $\delta := \mathbf{x} - \mathbf{x}_0$ :

$$k=0: D_x^0 f(\mathbf{x}_0) \delta^0 = f(\mathbf{x}_0) \in \mathbb{R}; \quad (5.156)$$

$$k=1: D_x^1 f(\mathbf{x}_0) \delta^1 = \underbrace{\nabla_{\mathbf{x}} f(\mathbf{x}_0)}_{1 \times D} \underbrace{\delta}_{D \times 1} = \sum_{i=1}^D \nabla_{\mathbf{x}} f(\mathbf{x}_0)[i] \delta[i] \in \mathbb{R}; \quad (5.157)$$

$$k=2: D_x^2 f(\mathbf{x}_0) \delta^2 = \text{tr} \left( \underbrace{\mathbf{H}(\mathbf{x}_0)}_{D \times D} \underbrace{\delta \delta^T}_{D \times 1 \times D} \right) = \delta^T \mathbf{H}(\mathbf{x}_0) \delta; \quad (5.158)$$

$$\sum_{i=1}^D \sum_{j=1}^D H[i, j] \delta[i] \delta[j] \in \mathbb{R}; \quad (5.159)$$

$$k=3: D_x^3 f(\mathbf{x}_0) \delta^3 = \sum_{i=1}^D \sum_{j=1}^D \sum_{k=1}^D D_x^3 f(\mathbf{x}_0)[i, j, k] \delta[i] \delta[j] \delta[k] \in \mathbb{R}. \quad (5.160)$$

Здесь  $\mathbf{H}(\mathbf{x}_0)$  — значение матрицы Гессе функции  $f$  в точке  $(\mathbf{x}_0)$ .

### Пример 5.15 (разложение в ряд Тейлора функции двух переменных)

Рассмотрим функцию

$$f(x, y) = x^2 + 2xy + y^3. \quad (5.161)$$

Мы хотим найти разложение  $f$  в ряд Тейлора в точке  $(x_0, y_0) = (1, 2)$ . Сначала поймем, что мы ожидаем получить. Функция, заданная формулой (5.161), — многочлен степени 3. Мы ищем разложение в ряд Тейлора, которое само является линейной комбинацией многочленов. Чтобы получился многочлен степени 3, ряд Тейлора не должен содержать члены четвертой степени и выше. Таким образом, чтобы точно представить (5.161), в (5.151) достаточно найти первые четыре слагаемых.

Начнем искать разложение в ряд Тейлора с вычисления свободного члена и первых производных:

$$f(1, 2) = 13; \quad (5.162)$$

$$\frac{\partial f}{\partial x} = 2x + 2y \Rightarrow \frac{\partial f}{\partial x}(1, 2) = 6; \quad (5.163)$$

$$\frac{\partial f}{\partial y} = 2x + 3y^2 \Rightarrow \frac{\partial f}{\partial y}(1, 2) = 14. \quad (5.164)$$

Таким образом получаем

$$D_{x,y}^1 f(1,2) = \nabla_{x,y} f(1,2) = \begin{bmatrix} \frac{\partial f}{\partial x}(1,2) & \frac{\partial f}{\partial y}(1,2) \end{bmatrix} = \begin{bmatrix} 6 & 14 \end{bmatrix} \in \mathbb{R}^{1 \times 2}, \quad (5.165)$$

так что

$$\frac{D_{x,y}^1 f(1,2)}{1!} \delta = \begin{bmatrix} 6 & 14 \end{bmatrix} \begin{bmatrix} x-1 \\ y-2 \end{bmatrix} = 6(x-1) + 14(y-2). \quad (5.166)$$

Заметим, что  $D_{x,y}^1 f(1,2) \delta$  содержит только линейные слагаемые, то есть многочлены степени 1.

Частные производные второго порядка равны

$$\frac{\partial^2 f}{\partial x^2} = 2 \Rightarrow \frac{\partial^2 f}{\partial x^2}(1,2) = 2; \quad (5.167)$$

$$\frac{\partial^2 f}{\partial y^2} = 6y \Rightarrow \frac{\partial^2 f}{\partial y^2}(1,2) = 12; \quad (5.168)$$

$$\frac{\partial^2 f}{\partial y \partial x} = 2 \Rightarrow \frac{\partial^2 f}{\partial y \partial x}(1,2) = 2; \quad (5.169)$$

$$\frac{\partial^2 f}{\partial x \partial y} = 2 \Rightarrow \frac{\partial^2 f}{\partial x \partial y}(1,2) = 2. \quad (5.170)$$

Объединим их в матрицу Гессе:

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} = \begin{bmatrix} 2 & 2 \\ 2 & 12 \end{bmatrix}. \quad (5.171)$$

Следовательно,

$$\mathbf{H}(1,2) = \begin{bmatrix} 2 & 2 \\ 2 & 12 \end{bmatrix} \in \mathbb{R}^{2 \times 2}. \quad (5.172)$$

Таким образом, следующий член ряда Тейлора равен

$$\frac{D_{x,y}^2 f(1,2)}{2!} \delta^2 = \frac{1}{2} \delta^T \mathbf{H}(1,2) \delta = \quad (5.173a)$$

$$= \frac{1}{2} [x-1 \quad y-2] \begin{bmatrix} 2 & 2 \\ 2 & 12 \end{bmatrix} \begin{bmatrix} x-1 \\ y-2 \end{bmatrix} = \quad (5.173b)$$

$$= (x-1)^2 + 2(x-1)(y-2) + 6(y-2)^2. \quad (5.173c)$$

Видим, что  $D_{x,y}^2 f(1, 2)\delta^2$  содержит только квадратичные слагаемые, то есть многочлены степени 2.

Производные третьего порядка равны

$$D_{x,y}^3 f = \begin{bmatrix} \frac{\partial \mathbf{H}}{\partial x} & \frac{\partial \mathbf{H}}{\partial y} \end{bmatrix} \in \mathbb{R}^{2 \times 2 \times 2}; \quad (5.174)$$

$$D_{x,y}^3 f[:, :, 1] = \frac{\partial \mathbf{H}}{\partial x} = \begin{bmatrix} \frac{\partial^3 f}{\partial x^3} & \frac{\partial^3 f}{\partial x^2 \partial y} \\ \frac{\partial^3 f}{\partial x \partial y \partial x} & \frac{\partial^3 f}{\partial x \partial y^2} \end{bmatrix}; \quad (5.175)$$

$$D_{x,y}^3 f[:, :, 2] = \frac{\partial \mathbf{H}}{\partial y} = \begin{bmatrix} \frac{\partial^3 f}{\partial y \partial x^3} & \frac{\partial^3 f}{\partial y \partial x \partial y} \\ \frac{\partial^3 f}{\partial y^2 \partial x} & \frac{\partial^3 f}{\partial y^3} \end{bmatrix}. \quad (5.176)$$

Так как все вторые производные из (5.171), кроме одной, являются константами, только одна из производных третьего порядка будет ненулевой:

$$\frac{\partial^3 f}{\partial y^3} = 6 \Rightarrow \frac{\partial^3 f}{\partial y^3}(1, 2) = 6. \quad (5.177)$$

Производные более высоких порядков, как и смешанные производные порядка 3 (например,  $\frac{\partial f^3}{\partial x^2 \partial y}$ ), обращаются в нуль, так что

$$D_{x,y}^3 f[:, :, 1] = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad D_{x,y}^3 f[:, :, 2] = \begin{bmatrix} 0 & 0 \\ 0 & 6 \end{bmatrix} \quad (5.178)$$

и

$$\frac{D_{x,y}^3 f(1, 2)}{3!} \delta^3 = (y-2)^3. \quad (5.179)$$

Мы нашли все кубические слагаемые. В итоге разложением  $f$  в ряд Тейлора в точке  $(x_0, y_0) = (1, 2)$  будет

$$f(x) = f(1, 2) + D_{x,y}^1 f(1, 2)\delta + \frac{D_{x,y}^2 f(1, 2)}{2!}\delta^2 + \frac{D_{x,y}^3 f(1, 2)}{3!}\delta^3 = \quad (5.180a)$$

$$\begin{aligned}
 &= f(1, 2) + \frac{\partial f(1, 2)}{\partial x}(x-1) + \frac{\partial f(1, 2)}{\partial y}(y-2) + \\
 &\quad + \frac{1}{2!} \left( \frac{\partial^2 f(1, 2)}{\partial x^2}(x-1)^2 + \frac{\partial^2 f(1, 2)}{\partial y^2}(y-2)^2 + \right. \\
 &\quad \left. + 2 \frac{\partial^2 f(1, 2)}{\partial x \partial y}(x-1)(y-2) \right) + \frac{1}{6} \frac{\partial^3 f(1, 2)}{\partial y^3}(y-2)^3 = \tag{5.180b} \\
 &= 13 + 6(x-1) + 14(y-2) + \\
 &\quad + (x-1)^2 + 6(y-2)^2 + 2(x-1)(y-2) + (y-2)^3. \tag{5.180c}
 \end{aligned}$$

В этом примере ряд Тейлора в точности равен многочлену из (5.161), то есть (5.180c) совпадает с (5.161). Здесь такой итог неудивителен, поскольку исходной функцией был многочлен степени 3, и мы в (5.180c) выражали его как линейную комбинацию констант и многочленов степеней 1, 2 и 3.

## 5.9. ДЛЯ ДАЛЬНЕЙШЕГО ЧТЕНИЯ

Больше материала про дифференцирование матриц вместе с кратким изложением необходимых сведений из линейной алгебры можно найти в книге Magnus and Neudecker (2007). Автоматическое дифференцирование также является темой с длинной историей, и заинтересованного читателя мы отсылаем к Griewank and Walther (2003, 2008) и Elliott (2009), а также их спискам литературы.

В машинном обучении и других областях нередко требуется вычислять математическое ожидание, имеющее вид интеграла

$$\mathbb{E}_x[f(\mathbf{x})] = \int f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}. \tag{5.181}$$

Даже при «хорошей» (например, гауссовой) функции  $p(\mathbf{x})$ , в общем случае этот интеграл не берется. Один из способов поиска приближенного решения использует разложение функции  $f$  в ряд Тейлора. Пусть  $p(\mathbf{x}) = \mathcal{N}(\mu, \Sigma)$  — плотность гауссова распределения. Приближение  $f$  многочленом Тейлора степени 1 в окрестности точки  $\mu$  локально приближает нелинейную функцию к линейной. Для линейных функций, если  $p(\mathbf{x})$  — плотность гауссового распределения, можно найти среднее и дисперсию (раздел 6.5). Это свойство играет важную роль в расширенном фильтре Калмана (Maybeck, 1979) для оценки состояния

нелинейных динамических систем (модель пространства состояний). Другие детерминированные способы приблизить интеграл из (5.181) — это «преобразование без запаха» (unscented transform) (Julier and Uhlmann, 1997), не требующее использования градиентов, и аппроксимация Лапласа (MacKay, 2003; Bishop, 2006; Murphy, 2012), использующая разложение в ряд Тейлора до квадратичных слагаемых (а значит, матрицу Гессе) для локального гауссова приближения  $p(\mathbf{x})$  в окрестности ее моды.

## УПРАЖНЕНИЯ

**5.1.** Вычислите производную  $f'(x)$  для

$$f(x) = \log(x^4) \sin(x^3).$$

**5.2.** Вычислите производную логистической сигмоидной функции

$$f(x) = \frac{1}{1 + \exp(-x)}.$$

**5.3.** Вычислите производную функции

$$f(x) = \exp\left(-\frac{1}{2\sigma^2}(x - \mu)^2\right),$$

где  $\mu, \sigma \in \mathbb{R}$  — константы.

**5.4.** Вычислите многочлены Тейлора  $T_n, n = 0, \dots, 5$  из  $f(x) = \sin(x) + \cos(x)$  в точке  $x_0 = 0$ .

**5.5.** Рассмотрим следующие функции:

$$f_1(\mathbf{x}) = \sin(x_1) \cos(x_2), \quad \mathbf{x} \in \mathbb{R}^2;$$

$$f_2(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^n;$$

$$f_3(\mathbf{x}) = \mathbf{x} \mathbf{x}^\top, \quad \mathbf{x} \in \mathbb{R}^n.$$

a. Каковы размеры  $\frac{\partial f_i}{\partial \mathbf{x}}$ ?

b. Вычислите якобианы.

**5.6.** Продифференцируйте  $f$  по  $\mathbf{t}$  и  $g$  по  $\mathbf{X}$ , где

$$f(\mathbf{t}) = \sin(\log(\mathbf{t}^\top \mathbf{t})), \quad \mathbf{t} \in \mathbb{R}^D;$$

$$g(\mathbf{X}) = \text{tr}(\mathbf{A} \mathbf{X} \mathbf{B}), \quad \mathbf{A} \in \mathbb{R}^{D \times E}, \mathbf{X} \in \mathbb{R}^{E \times F}, \mathbf{B} \in \mathbb{R}^{F \times D},$$

где  $\text{tr}$  обозначает след.

**5.7.** Вычислите производные  $df/d\mathbf{x}$  следующих функций, используя цепное правило. Укажите размеры каждой частной производной. Подробно опишите свои действия.

a.

$$f(z) = \log(1 + z), z = \mathbf{x}^T \mathbf{x}, \mathbf{x} \in \mathbb{R}^D.$$

b.

$$f(\mathbf{z}) = \sin(\mathbf{z}), \mathbf{z} = \mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{A} \in \mathbb{R}^{E \times D}, \mathbf{x} \in \mathbb{R}^D, \mathbf{b} \in \mathbb{R}^E,$$

где  $\sin(\cdot)$  применяется к каждому элементу  $\mathbf{z}$ .

**5.8.** Вычислите производные  $df/d\mathbf{x}$  следующих функций. Подробно опишите свои действия.

a. Используйте цепное правило. Укажите размеры каждой частной производной.

$$f(z) = \exp\left(-\frac{1}{2}z\right);$$

$$z = g(\mathbf{y}) = \mathbf{y}^T \mathbf{S}^{-1} \mathbf{y};$$

$$\mathbf{y} = h(\mathbf{x}) = \mathbf{x} - \boldsymbol{\mu},$$

где  $\mathbf{x}, \boldsymbol{\mu} \in \mathbb{R}^D, \mathbf{S} \in \mathbb{R}^{D \times D}$ .

b.  $f(\mathbf{x}) = \text{tr}(\mathbf{x}\mathbf{x}^T + \sigma^2 \mathbf{I}), \mathbf{x} \in \mathbb{R}^D$ .

Здесь  $\text{tr}(\mathbf{A})$  — след  $\mathbf{A}$ , то есть сумма диагональных элементов  $A_{ii}$ . Подсказка: явно выпишите внешний продукт.

c. Используйте цепное правило. Укажите размеры каждой частной производной. Явно вычислять произведение частных производных не нужно.

$$f = \tanh(\mathbf{z}) \in \mathbb{R}^M;$$

$$\mathbf{z} = \mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{x} \in \mathbb{R}^N, \mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{b} \in \mathbb{R}^M.$$

Здесь  $\tanh$  применяется к каждому компоненту  $\mathbf{z}$ .

**5.9.** Определим

$$g(\mathbf{z}, \mathbf{v}) := \log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}, \mathbf{v});$$

$$\mathbf{z} := t(\boldsymbol{\varepsilon}, \mathbf{v})$$

для дифференцируемых функций  $p, q, t$ . Используя цепное правило, вычислите градиент

$$\frac{d}{dv} g(\mathbf{z}, \mathbf{v}).$$

# 6

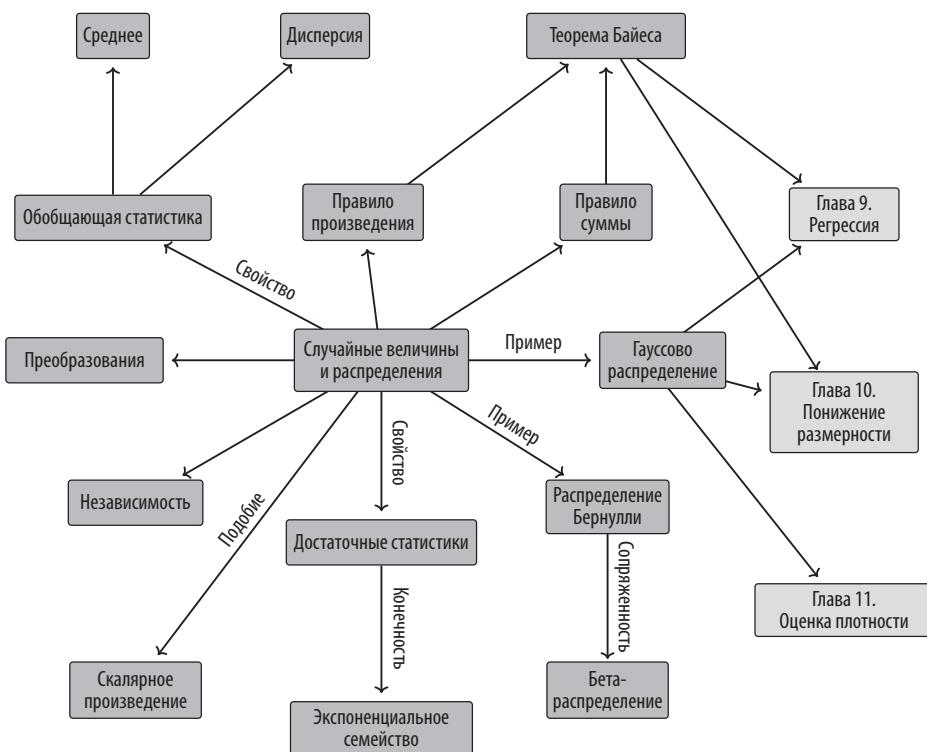
## Вероятность и распределения

Неформально говоря, теория вероятностей изучает неопределенность. Вероятность можно рассматривать как долю случаев, когда происходит данное событие, или как меру нашей уверенности в событии. Как упоминалось в главе 1, нам часто приходится оценивать неопределенность как в данных, так и в модели машинного обучения и ее прогнозах. Оценка неопределенности требует понятия *случайной величины* — функции, сопоставляющей исходам случайного эксперимента интересующие нас свойства. Со случайной величиной связана функция, измеряющая вероятность конкретного исхода (или множества исходов); ее называют *распределением вероятностей*.

Распределения вероятностей будут «кирпичиками», которые мы будем использовать в темах вероятностного моделирования (раздел 8.4), графических моделей (раздел 8.5) и выбора модели (раздел 8.6). В следующем разделе мы определим три составляющие, задающие вероятностное пространство (пространство исходов, события и вероятности событий), и их связь с понятием случайной величины. Мы сознательно выбрали не вполне формальный способ изложения, так как большая строгость помешала бы увидеть суть вводимых понятий. На рис. 6.1 показаны основные понятия данной главы.

### 6.1. ПОСТРОЕНИЕ ВЕРОЯТНОСТНОГО ПРОСТРАНСТВА

Цель теории вероятностей — построить математическую модель, позволяющую описывать случайные исходы экспериментов. Например, при подбрасывании монеты мы не можем предсказать результат, но, сделав большое число бросков, можем наблюдать закономерности. Используя математическую модель вероятности, мы хотим автоматически делать выводы — в этом смысле теория вероятностей обобщает логику (Jaynes, 2003).



### 6.1.1. Философские вопросы

При построении систем автоматического логического вывода классическая булева логика не позволяет нам выразить некоторые вполне разумные рассуждения. Рассмотрим такой сценарий. Мы обнаруживаем, что при ложности  $A$  реже случается  $B$ . Такой вывод нельзя сделать в классической логике. Если теперь мы узнаем, что  $B$  истинно, то  $A$  станет менее правдоподобным. Мы каждый день рассуждаем подобным образом. Например, пусть мы ждем подругу. Возможны три варианта развития событий:  $H_1$  — она придет вовремя,  $H_2$  — она застрянет в пробке и  $H_3$  — ее похитят инопланетяне. Заметив, что она опаздывает, мы логически отметим  $H_1$ . Мы также посчитаем  $H_2$  более правдоподобной версией, хотя строго говоря законы логики нас к этому не вынуждают. Мы можем не сбрасывать со счетов и  $H_3$ , но будем считать ее крайне малоправдоподобной. Как мы приходим к тому, что  $H_2$  наиболее разумна? С такой точки зрения, теорию вероятностей можно рассматривать как обобщение булевой логики. В машинном обучении она часто применяется при построении систем

автоматического логического вывода. Информацию об ее применении в системах логического вывода можно найти в Pearl (1988).

Философские основания теории вероятностей и ее связь с нашими предположениями об истинности изучались Коксом (Jaynes, 2003)<sup>1</sup>. Можно посмотреть на это и так: если мы подведем строгие основания под здравый смысл, в итоге мы построим понятие вероятности. Е. Т. Jaynes (1922–1998) перечислил три критерия, которые должны выполняться для любых рассуждений о вероятностях:

1. Вероятности должны представляться вещественными числами.
2. Эти числа должны быть согласованы со здравым смыслом.
3. Итоговые рассуждения должны быть корректными в трех смыслах:
  - (a) Непротиворечивость. При вычислении одной и той же вероятности разными способами должно получаться одно и то же значение.
  - (b) Полнота. Необходимо учесть все имеющиеся данные.
  - (c) Воспроизводимость. Если в двух задачах у нас имеется одна и та же информация, вероятности также должны совпадать.

По теореме Кокса – Джейнса, этих свойств достаточно для задания универсальных математических правил, касающихся вероятности  $p$ , с точностью до произвольного монотонного преобразования. Это и будут основные правила теории вероятностей.

**ПРИМЕЧАНИЕ** В машинном обучении и статистике существуют две основные интерпретации понятия вероятности: байесовская и частотная (Bishop, 2006; Efron and Hastie, 2016). Байесовская интерпретация вероятности – это степень нашей уверенности в некотором событии, ее также называют «субъективной вероятностью». При частотной интерпретации подсчитывается доля случаев, когда происходит интересующее нас событие, от общего количества экспериментов. Вероятность события определяется как предельное значение этой доли при стремящемся к бесконечности числе экспериментов. ◆

Часто тексты, посвященные вероятностным моделям в машинном обучении, используют нестрогие обозначения и жargon, порой затрудняющие понимание. Эта книга не будет исключением. Несколько разных понятий будут скрываться под названием «распределение вероятностей», и читателю часто придется понимать значение по контексту. Один из способов разобраться – подумать, моделируем ли мы разбиение на категории (дискретную случайную величину) или что-то, изменяющееся непрерывно (непрерывную случайную величину).

<sup>1</sup> «Чтобы рассуждать разумно, необходимо расширить дискретные понятия истины и лжи до непрерывных вероятностей» (Jaynes, 2003).

Вопросы, встающие перед нами в машинном обучении, часто тесно связаны с ответом на этот вопрос.

### 6.1.2. Вероятность и случайные величины

При обсуждении вероятности часто путают три разных понятия. Первое из них — понятие вероятностного пространства, позволяющее нам численно выразить вероятность. Обычно мы работаем с ним не напрямую, а через посредство случайных величин (второе понятие), которые «переносят» вероятность на более удобное пространство, зачастую числовое. Третье понятие — распределение вероятностей, закон, описывающий поведение случайной величины. Первые два понятия мы введем в данном разделе, а третье — в разделе 6.2.

Современная теория вероятностей основана на наборе аксиом, предложенных Колмогоровым (Grinstead and Snell, 1997; Jaynes, 2003), вводящим понятия пространства элементарных исходов, пространства событий и вероятностной меры. Вероятностное пространство описывает некоторый реальный процесс (который мы будем называть экспериментом) со случайным результатом.

#### Пространство элементарных исходов $\Omega$

*Пространство элементарных исходов* — это множество всех возможных исходов эксперимента, обычно обозначаемое как  $\Omega$ . Например, для двух последовательных бросков монеты пространство элементарных исходов — это  $\{\text{оо}, \text{пр}, \text{ор}, \text{ро}\}$ , где «о» — «орел», а «р» — «решка».

#### Пространство событий $\mathcal{A}$

*Пространство событий* — это пространство возможных результатов эксперимента. Подмножество  $A$  в пространстве элементарных исходов  $\Omega$  принадлежит пространству событий  $\mathcal{A}$ , если после эксперимента мы можем определить, принадлежит ли исход  $\omega \in \Omega$  подмножеству  $A$ . Пространство событий  $\mathcal{A}$  состоит из подмножеств  $\Omega$ , и для дискретных распределений (раздел 6.2.1) обычно совпадает с множеством всех подмножеств в  $\Omega$ .

#### Вероятность $P$

С каждым событием  $A \in \mathcal{A}$  мы свяжем число  $P(A)$ , измеряющее частоту события или нашу уверенность в нем.  $P(A)$  называют *вероятностью*  $A$ .

Вероятность любого события обязана лежать в интервале  $[0, 1]$ , а сумма вероятностей всех исходов в  $\Omega$  должна составлять 1, то есть  $P(\Omega) = 1$ .

Введя понятие вероятностного пространства  $(\Omega, \mathcal{A}, P)$ , мы хотим применить его к явлениям реального мира. В машинном обучении мы зачастую не ука-

зываем вероятностное пространство явно, а вместо этого говорим о вероятностях значений интересующей нас величины, множество которых обозначаем за  $\mathcal{T}$ . В этой книге мы будем называть  $\mathcal{T}$  фазовым пространством, а его элементы — состояниями. Введем функцию  $X: \Omega \rightarrow \mathcal{T}$ , принимающую элемент  $\Omega$  (элементарный исход) и возвращающую значение  $x$  интересующей нас величины, принадлежащее  $\mathcal{T}$ . Такое отображение из  $\Omega$  в  $\mathcal{T}$  называют случайной величиной<sup>1</sup>. Например, при подбрасывании двух монет и подсчете количества орлов случайная величина  $X$  сопоставляет элементарным исходам одно из трех значений:  $X(\text{oo}) = 2$ ,  $X(\text{op}) = 1$ ,  $X(\text{po}) = 1$  и  $X(\text{pp}) = 0$ . В этом случае  $\mathcal{T} = \{0, 1, 2\}$ , и нас интересуют вероятности элементов множества  $\mathcal{T}$ . Когда пространство элементарных исходов  $\Omega$  и фазовое пространство  $\mathcal{T}$  конечны, функцию, задающую случайную величину, можно представить в виде таблицы. С каждым подмножеством  $S \subseteq \mathcal{T}$  мы связываем вероятность  $P_X(S) \in [0, 1]$ , с которой происходит событие «значение  $X$  попадает в  $S$ ». Пример 6.1 иллюстрирует введенные сейчас термины.

**ПРИМЕЧАНИЕ** Пространство элементарных исходов  $\Omega$ , к сожалению, выступает в различных источниках под разными названиями. Его нередко называют «пространством состояний» (Jacod and Protter, 2004), хотя этот термин часто относится и к состояниям динамической системы (Hasselblatt and Katok, 2003).  $\Omega$  также порой выступает под именем «пространства исходов», «пространства возможностей» и «пространства событий». ♦

### Пример 6.1

Мы предполагаем, что читатель уже знаком с вычислением вероятностей пересечения и объединения событий. Более аккуратное введение в теорию вероятностей, с большим количеством примеров, можно найти в главе 2 Walpole et al. (2011).

Рассмотрим статистический эксперимент — игру, в которой мы вытаскиваем из мешка две монеты (с возвращением). В мешке есть американские монеты (обозначаемые как \$) и британские (обозначаемые как £), так что для двух вытаскиваний есть четыре возможных исхода. Таким образом, пространство элементарных исходов  $\Omega$  состоит из (\$, \$), (\$, £), (£, \$), (£, £). Допустим, что в мешке такое соотношение двух видов монет, что вероятность вытащить доллар равна 0,3<sup>2</sup>.

<sup>1</sup> Термин «случайная величина» вносит много путаницы — поскольку это не величина и она не случайна. На самом деле это функция.

<sup>2</sup> Этот игрушечный пример по сути аналогичен примеру с броском нечестной монетки.

Мы хотим узнать, сколько раз вытащим доллар. Обозначим через  $X$  случайную величину, отображающую исход из  $\Omega$  в элемент  $\mathcal{T}$ , обозначающий число вытащенных долларов.

Посмотрев на пространство элементарных исходов, мы понимаем, что можем получить 1, 2 или 3 доллара, так что  $\mathcal{T} = \{0, 1, 2\}$ . Случайная величина  $X$  (функция) может быть описана с помощью такой таблицы:

$$X((\$, \$)) = 2; \quad (6.1)$$

$$X((\$, £)) = 1; \quad (6.2)$$

$$X((£, \$)) = 1; \quad (6.3)$$

$$X((£, £)) = 0. \quad (6.4)$$

Так как перед тем, как вытащить вторую монету, мы возвращаем первую в мешок, то два вытаскивания независимы (что мы обсудим в разделе 6.4.5). Заметим, что двум элементарным исходам соответствует одно и то же событие из  $\mathcal{T}$ , когда вытащен только один доллар.

Таким образом, распределение вероятностей (раздел 6.2.1) для случайной величины  $X$  задано формулами

$$\begin{aligned} P(X = 2) &= P((\$, \$)) = \\ &= P(\$) \cdot P(\$) = \\ &= 0,3 \cdot 0,3 = 0,09; \end{aligned} \quad (6.5)$$

$$\begin{aligned} P(X = 1) &= P((\$, £) \cup (£, \$)) = \\ &= P((\$, £)) + P((£, \$)) = \\ &= 0,3 \cdot (1 - 0,3) + (1 - 0,3) \cdot 0,3 = 0,42; \end{aligned} \quad (6.6)$$

$$\begin{aligned} P(X = 0) &= P((£, £)) = \\ &= P(£) \cdot P(£) = \\ &= (1 - 0,3) \cdot (1 - 0,3) = 0,49. \end{aligned} \quad (6.7)$$

В предшествующих рассуждениях мы отождествляли две вероятности — вероятность определенного значения  $X$  и вероятность соответствующих исходов из  $\Omega$ . Так, в формуле (6.7) мы говорим, что  $P(X = 0) = P((£, £))$ . Рассмотрим случайную величину  $X : \Omega \rightarrow \mathcal{T}$  и подмножество  $S \subseteq \mathcal{T}$  (например, состоящее только из одного элемента  $\mathcal{T}$ , как выпадение одного орла при подбрасывании двух монет). Пусть  $X^{-1}(S)$  — прообраз  $S$  относительно  $X$ , то есть множество

элементов  $\Omega$ , которые переходят в элементы  $S$  при действии  $X$ ;  $\{\omega \in \Omega : X(\omega) \in S\}$ . Один из способов думать о таком переносе вероятностей с событий из  $\Omega$  на значения  $X$  — рассматривать вероятность прообраза  $S$  (Jacod and Protter, 2004). Для  $S \subseteq T$  это можно записать так:

$$P_X(S) = P(X \in S) = P(X^{-1}(S)) = P(\{\omega \in \Omega : X(\omega) \in S\}). \quad (6.8)$$

В левой части формулы (6.8) стоит вероятность множества значений случайной величины (например, того, что вытащен ровно один доллар), которые нас интересуют. Случайная величина  $X$  сопоставляет исходам из  $\Omega$  значения из  $T$ , так что в правой части (6.8) мы видим вероятность множества исходов (из  $\Omega$ ), обладающих нужным свойством (в данном случае,  $\$f$  и  $f\$$ ). Мы говорим, что задано распределение  $P_X$  случайной величины  $X$ , которое определяется соответствием между событиями и значением случайной величины. Иными словами, функция  $P_X$  (или, что то же самое,  $P \circ X^{-1}$ ) задает *закон распределения* случайной величины  $X$ .

**ПРИМЕЧАНИЕ** Фазовое пространство, то есть область значений  $T$  случайной величины  $X$ , характеризует тип случайной величины. Когда  $T$  конечно или счетно, случайная величина называется дискретной (раздел 6.2.1). Непрерывные случайные величины (раздел 6.2.2) мы рассматриваем только для  $T = \mathbb{R}$  или  $T = \mathbb{R}^D$ . ♦

### 6.1.3. Статистика

Теорию вероятностей и статистику часто упоминают вместе, но они касаются двух разных типов неопределенности. Их можно противопоставить по типу рассматриваемых задач. В теории вероятности мы строим модель некоторого процесса, выражаем имеющуюся неопределенность с помощью случайной величины и пытаемся вывести, что произойдет. В статистике мы уже знаем, что произошло, и пытаемся с помощью утверждений теории вероятностей понять свойства процесса. В этом смысле машинное обучение близко к статистике в своей цели построить модель, адекватно описывающую процесс, породивший данные. С помощью теории вероятностей мы можем подобрать наилучшую модель для имеющихся данных.

Другая важная черта машинного обучения — наша заинтересованность в обобщении (см. главу 8). Это означает, что нам на самом деле важны результаты нашей системы на объектах, которые мы еще не видели. Анализ будущих результатов основан на теории вероятностей и статистике, в основном выходящих за пределы содержания этой главы. Заинтересованному читателю советуем обратиться к книгам Boucheron et al. (2013) и Shalev-Shwartz and Ben-David (2014). Мы еще поговорим о статистике в главе 8.

## 6.2. ДИСКРЕТНЫЕ И НЕПРЕРЫВНЫЕ РАСПРЕДЕЛЕНИЯ

Сосредоточимся на способах описать вероятности событий, введенных в главе 6.1. В зависимости от того, дискретно или непрерывно фазовое пространство, естественно описывать распределения по-разному. Когда фазовое пространство  $T$  дискретно, можно найти вероятность, что случайная величина  $X$  принимает конкретное значение  $x \in T$ , то есть  $P(X=x)$ . Выражение  $P(X=x)$  для дискретной случайной величины  $X$  известно как распределение вероятности. Когда фазовое пространство  $T$  непрерывно, например это вещественная прямая  $\mathbb{R}$ , естественнее говорить о вероятностях того, что случайная величина  $X$  попадает в некоторый интервал, то есть  $P(a \leq X \leq b)$  для  $a < b$ . Нам удобно говорить о вероятности, что случайная величина  $X$  не превосходит некоторого  $x$ , то есть  $P(X \leq x)$ . Выражение  $P(X \leq x)$  для непрерывной случайной величины  $X$  известно как *функция распределения*. Непрерывные случайные величины мы обсудим в разделе 6.2.2. Терминологию и различия дискретных и непрерывных случайных величин мы обсудим в разделе 6.2.3.

**ПРИМЕЧАНИЕ** Термин «одномерное распределение» мы будем применять, говоря о распределениях одной случайной величины (значения которой обозначены обычной буквой  $x$ ). Распределения более чем одной случайной величины мы будем называть *многомерными* и рассматривать вектор случайных величин (обозначая состояния жирным  $\mathbf{x}$ ). ◆

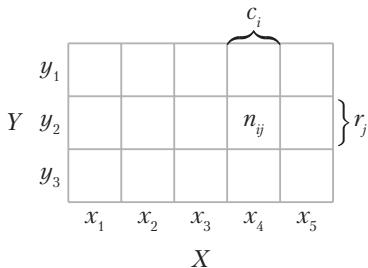
### 6.2.1. Дискретные вероятности

Когда фазовое пространство дискретно, можно представлять себе распределение многомерной случайной величины как многомерный массив чисел. На рис. 6.2 представлен пример. Фазовое пространство совместной вероятности представляет собой декартово произведение фазовых пространств для каждой из случайных величин. Совместную вероятность мы определяем как вероятность того, что обе случайные величины принимают заданные значения:

$$P(X=x_i, Y=y_j) = \frac{n_{ij}}{N}, \quad (6.9)$$

где  $n_{ij}$  — количество исходов, соответствующих значениям  $x_i$  и  $y_j$ , а  $N$  — общее число исходов. *Совместная вероятность* есть вероятность пересечения событий, то есть  $P(X=x_i, Y=y_j) = P(X=x_i \cap Y=y_j)$ . На рис. 6.2 показано дискретное распределение. Для двух случайных величин  $X$  и  $Y$  вероятность того, что  $X=x$  и  $Y=y$ , записывается как  $p(x, y)$  и называется совместной вероятностью. Такая запись обусловлена тем, что можно думать о ней как о функции, принимающей на вход значения  $x$  и  $y$  и возвращающей вещественное число. *Частная (маргинальная) вероятность* того, что  $X$  принимает значение  $x$  безотносительно значения  $Y$

записывается как  $p(x)$ . Чтобы показать, что случайной величине  $X$  соответствует распределение  $p(x)$ , мы пишем  $X \sim p(x)$ . Если мы рассматриваем только исходы, при которых  $X=x$ , доля исходов, при которых  $Y=y$  (*условная вероятность*), обозначается как  $p(y|x)$ .



**Рис. 6.2.** Изображение распределения для двух дискретных случайных величин  $X$  и  $Y$ .  
Диаграмма (с изменениями) взята из Bishop (2006)

### Пример 6.2

Рассмотрим две случайные величины  $X$  и  $Y$ , где  $X$  имеет пять возможных значений, а  $Y$  – три возможных значения, как показано на рис. 6.2. Количество исходов, при которых  $X=x_i$  и  $Y=y_j$ , мы обозначим как  $n_{ij}$ , а общее количество возможных исходов как  $N$ .  $c_i$  – сумма элементов  $i$ -го столбца, то есть  $c_i = \sum_{j=1}^3 n_{ij}$ . Аналогично,  $r_j$  – это суммы по строкам, то есть  $r_j = \sum_{i=1}^5 n_{ij}$ . С помощью этих обозначений можно полностью описать распределение  $X$  и  $Y$ . Частное распределение каждой из случайных величин выражается как сумма по строке или столбцу:

$$P(X=x_i) = \frac{c_i}{N} = \frac{\sum_{j=1}^3 n_{ij}}{N} \quad (6.10)$$

и

$$P(Y=y_j) = \frac{r_j}{N} = \frac{\sum_{i=1}^5 n_{ij}}{N}, \quad (6.11)$$

где  $c_i$  и  $r_j$  соответствуют  $i$ -му столбцу и  $j$ -й строке таблицы вероятностей. Мы договорились, что для дискретной случайной величины с конечным числом возможных значений сумма вероятностей всех значений равна единице:

$$\sum_{i=1}^5 P(X=x_i) = 1 \text{ и } \sum_{j=1}^3 P(Y=y_j) = 1. \quad (6.12)$$

Условная вероятность — это доля исходов из данной клетки от общего количества исходов в строке или столбце. Например, условная вероятность  $Y$  при заданном  $X$  равна

$$P(Y = y_j \mid X = x_i) = \frac{n_{ij}}{c_i}, \quad (6.13)$$

а условная вероятность  $X$  при заданном  $Y$

$$P(X = x_i \mid Y = y_j) = \frac{n_{ij}}{r_j}. \quad (6.14)$$

В машинном обучении дискретные распределения используют при моделировании *категориальных признаков*, то есть принимающих значения в конечном неупорядоченном множестве. Это могут быть признаки, подающиеся на вход алгоритму (например, образование при предсказании заработной платы) или метки классов (скажем, буквы при распознавании рукописных текстов). Дискретные распределения также часто используют для построения вероятностных моделей, комбинирующих конечное число непрерывных распределений (глава 11).

### 6.2.2. Непрерывные вероятности

В этом разделе мы рассматриваем вещественнозначные случайные величины, то есть фазовыми пространствами будут интервалы на вещественной оси. В данной книге мы делаем вид, что можем выполнять операции над вещественными случайными величинами так же, как в дискретном пространстве с конечным числом значений. Однако это упрощение неверно для двух ситуаций: повторения операции бесконечное число раз и случайного выбора точки на интервале. Первая ситуация возникает при обсуждении обобщающей способности (глава 8). Вторая — при рассмотрении непрерывных (например, гауссовых) распределений (раздел 6.5). Однако для наших целей допустимо подобное отступление от математической строгости ради более быстрого введения в тему.

**ПРИМЕЧАНИЕ** В непрерывных вероятностных пространствах возникают два дополнительных континтуитивных нюанса. Во-первых, множество всех подмножеств (которое было пространством событий  $\mathcal{A}$  в разделе 6.1) не обладает столь хорошими свойствами. Чтобы множества из  $\mathcal{A}$  «хорошо себя вели» при операциях дополнения, объединения и пересечения, необходимо брать не все подмножества. Во-вторых, количество элементов множества (которое в дискретных пространствах можно получить простым подсчетом элементов) оказы-

вается не столь простым понятием. «Размер» множества будет выражаться понятием *меры*. Например, мощность дискретных множеств, длина интервала в  $\mathbb{R}$  и объем области в  $\mathbb{R}^d$  — это меры. Множества, «хорошо себя ведущие» относительно операций над ним и вдобавок имеющие топологическую структуру, называются *борелевской  $\sigma$ -алгеброй*. Бетанкур подробно, но не увязая в ненужных деталях, пишет о построении вероятностных пространств на основе теории множеств, см. <https://tinyurl.com/yb3t6mfd>. Для более строгого изложения мы рекомендуем Billingsley (1995) и Jacod and Protter (2004).

В нашей книге мы рассматриваем вещественноизначные случайные величины с соответствующей борелевской  $\sigma$ -алгеброй. Мы рассматриваем случайные величины из  $\mathbb{R}^D$  как векторы из вещественных случайных величин. ♦

**Определение 6.1 (плотность вероятности).** Функция  $f: \mathbb{R}^D \rightarrow \mathbb{R}$  называется *плотностью вероятности* (probability density function, pdf), если

1.  $\forall \mathbf{x} \in \mathbb{R}^D: f(\mathbf{x}) \geq 0$ .
2. Она интегрируема и

$$\int_{\mathbb{R}^D} f(\mathbf{x}) d\mathbf{x} = 1. \quad (6.15)$$

Для распределений дискретной случайной величины аналогом интеграла (6.15) является сумма (6.12).

Заметим, что плотностью вероятности является любая неотрицательная функция  $f$ , интеграл от которой равен 1. С этой функцией мы связываем случайную величину  $X$ :

$$P(a \leq X \leq b) = \int_a^b f(x) dx, \quad (6.16)$$

где  $a, b \in \mathbb{R}$ , и  $x \in \mathbb{R}$  — значения непрерывной случайной величины  $X$ . Значения  $\mathbf{x} \in \mathbb{R}^D$  определяются аналогично, как вектор из  $x \in \mathbb{R}$ . Зависимость (6.16) называют *законом распределения* случайной величины  $X$ .

**ПРИМЕЧАНИЕ** В отличие от дискретных случайных величин, вероятность  $P(X = x)$  того, что непрерывная случайная величина  $X$  принимает конкретное значение, равна нулю. Представьте, что в формуле (6.16) указан интервал с  $a = b$ . ♦

**Определение 6.2 (функция распределения).** *Функция распределения* (cumulative distribution function, cdf) многомерной вещественноизначной случайной величины  $X$  со значениями  $\mathbf{x} \in \mathbb{R}^D$  задается формулой

$$F_X(\mathbf{x}) = P(X_1 \leq x_1, \dots, X_D \leq x_D), \quad (6.17)$$

где  $X = [X_1, \dots, X_D]^T$ ,  $\mathbf{x} = [x_1, \dots, x_D]^T$ , и в правой части стоят вероятности того, что  $X_i$  принимает значение, не превосходящее  $x_i$ .

Функцию распределения также можно представить как интеграл от плотности распределения<sup>1</sup>  $f(\mathbf{x})$ :

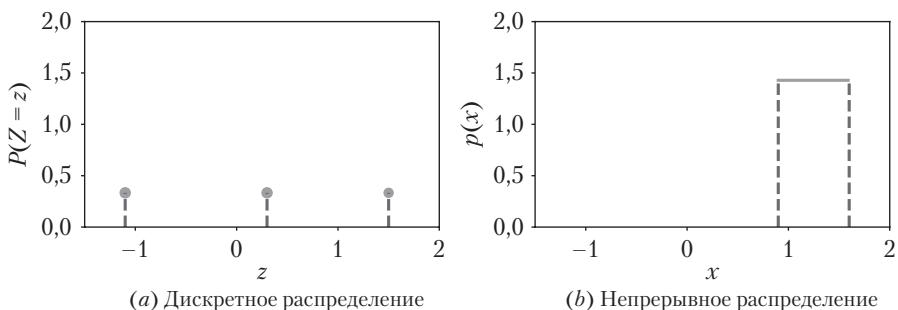
$$F_X(\mathbf{x}) = \int_{-\infty}^{x_1} \cdots \int_{-\infty}^{x_D} f(z_1, \dots, z_D) dz_1 \dots dz_D. \quad (6.18)$$

**ПРИМЕЧАНИЕ** Повторим, что, говоря о распределении, мы фактически имеем в виду два разных понятия. Первое — это плотность вероятности (обозначаемая  $f(x)$ ), неотрицательная функция с интегралом 1. Второе — закон распределения случайной величины  $X$ , то есть связь  $X$  с плотностью  $f(x)$ . ♦

В основном в этой книге мы будем использовать обозначения  $f(x)$  и  $F_x(x)$  для плотности распределения и функции распределения. В разделе 6.7 нам придется быть аккуратнее с этими понятиями.

### 6.2.3. Различия дискретных и непрерывных распределений

Из раздела 6.1.2 мы знаем, что вероятности неотрицательны и в сумме по всем исходам дают единицу. Для дискретных случайных величин (6.12) это означает, что вероятность любого значения лежит в интервале  $[0, 1]$ . Однако для непрерывных случайных величин нормализация (6.15) не подразумевает, что значение плотности во всех точках не превосходит 1. Это показано на рис. 6.3 на примере *равномерного распределения* для дискретной и непрерывной случайных величин.



**Рис. 6.3.** Примеры (a) дискретного и (b) непрерывного равномерного распределения. Подробности приведены в примере 6.3

<sup>1</sup> Для некоторых функций распределения не существует соответствующей плотности.

### Пример 6.3

Рассмотрим два примера равномерных распределений, в которых каждое значение равновероятно. Этот пример показывает некоторые различия между дискретными и непрерывными случайными величинами.

Пусть  $Z$  — дискретная равномерно распределенная случайная величина с тремя возможными значениями  $\{z = -1,1, z = 0,3, z = 1,5\}$ <sup>1</sup>. Распределение можно представить как таблицу значений вероятности

$z$	-1,1	0,3	1,5
$P(Z = z)$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$

Можно также представить его графически (рис. 6.4(a)), отмечая значения на оси  $x$ , а на оси  $y$  — вероятность каждого из значений. Мы специально удлинили ось  $y$  на рис. 6.4(a), чтобы она была такой же, как на рис. 6.4(b).

Пусть  $X$  — непрерывная случайная величина, принимающая значения на интервале  $0,9 \leq X \leq 1,6$ , как показано на рис. 6.4(b). Заметим, что плотность может подниматься выше 1. Однако должно выполняться условие

$$\int_{0,9}^{1,6} p(x) dx = 1. \quad (6.19)$$

**ПРИМЕЧАНИЕ** С дискретными распределениями связана еще одна дополнительная сложность. Значения  $z_1, \dots, z_d$  в общем случае не упорядочены, то есть их нельзя сравнивать, например  $z_1 = \text{красный}, z_2 = \text{зеленый}, z_3 = \text{синий}$ . Однако во многих задачах машинного обучения дискретные значения будут числовыми, например  $z_1 = -1,1, z_2 = 0,3, z_3 = 1,5$ , когда можно заметить, что  $z_1 < z_2 < z_3$ . Числовые дискретные значения особенно полезны, поскольку нас часто интересует математическое ожидание (раздел 6.4.1) случайных величин.

К сожалению, обозначения и терминология литературы по машинному обучению не показывают разницы между пространством элементарных исходов  $\Omega$ , фазовым пространством  $T$  и случайной величиной  $X$ . Для  $x$  из множества возможных значений случайной величины  $X$ , то есть  $x \in T, p(x)$  обозначает вероятность, что  $X$  принимает значение  $x$ <sup>2</sup>. Для дискретной случайной величины та же самая вероятность записывается как  $P(X = x)$  и известна как распределение случайной

<sup>1</sup> Конкретные значения не важны. Мы выбрали их так, чтобы подчеркнуть неважность порядка значений.

<sup>2</sup> Мы считаем исход  $x$  аргументом для вероятности  $p(x)$ .

величины. Для непрерывной случайной величины  $p(x)$  называют плотностью вероятности. Чтобы еще больше запутать дело,  $P(X \leq x)$  называют функцией распределения. В данной главе как одномерные, так и многомерные случайные величины мы будем обозначать буквой  $X$ , а их значения — соответственно  $x$  и  $\mathbf{x}$ . Терминологию мы собрали в табл. 6.1.

**ПРИМЕЧАНИЕ** Выражение «распределение вероятностей» мы будем использовать не только для дискретного распределения, но и для плотности вероятности в непрерывном случае, хотя в строгом смысле это некорректно. Как и в основной массе литературы по машинному обучению, мы опираемся на контекст при использовании слов «распределение вероятностей» в разных смыслах. ♦

**Таблица 6.1.** Терминология для распределений вероятностей

Тип	«Вероятность точки»	«Вероятность интервала»
Дискретная	$P(X = x)$ Распределение	Не применимо
Непрерывная	$p(x)$ Плотность вероятности	$P(X \leq x)$ Функция распределения

### 6.3. ПРАВИЛО СУММЫ, ПРАВИЛО ПРОИЗВЕДЕНИЯ И ТЕОРЕМА БАЙЕСА

Теорию вероятностей мы можем считать расширением логики. Как уже говорилось в разделе 6.1.1, все правила теории вероятностей следуют из выполнения некоторых естественных требований (Jaynes, 2003, глава 2). Вероятностное моделирование (раздел 8.4) является основой для построения моделей машинного обучения. Когда известны распределения вероятностей (раздел 6.2), соответствующие неопределенностям в данных и в задаче, остается применять лишь два фундаментальных правила: правило суммы и правило произведения.

Вспомним формулу (6.9), в ней  $p(\mathbf{x}, \mathbf{y})$  — совместное распределение двух случайных величин  $x$  и  $y$ . Соответствующие частные распределения обозначим как  $p(\mathbf{x})$  и  $p(\mathbf{y})$ , а  $p(\mathbf{y} | \mathbf{x})$  — условное распределение  $\mathbf{y}$  при заданном  $\mathbf{x}$ . Зная определения частной и условной вероятности для дискретных и непрерывных случайных величин из раздела 6.2, мы можем сформулировать два фундаментальных правила теории вероятностей<sup>1</sup>.

<sup>1</sup> Эти два правила естественным образом возникают из требований, которые мы обсуждали в разделе 6.1.1.

Первое из них, *правило суммы*, утверждает, что

$$p(\mathbf{x}) = \begin{cases} \sum_{y \in \mathcal{Y}} p(\mathbf{x}, y) & \text{если } y \text{ дискретна} \\ \int_y p(\mathbf{x}, y) dy & \text{если } y \text{ непрерывна} \end{cases}, \quad (6.20)$$

где  $\mathcal{Y}$  — фазовое пространство случайной величины  $Y$ . Это означает, что мы суммируем (или интегрируем) по всем возможным значениям  $Y$ . Правило суммы также известно как *правило маргинализации*, так как связывает совместное и частное (маргинальное) распределения. В общем случае, когда рассматривается совместное распределение более чем одной случайной величины, правило суммы можно применять к любому подмножеству случайных величин и получать их частное распределение. А именно, если  $\mathbf{x} = [x_1 \dots, x_D]^T$ , мы получаем частное распределение

$$p(x_i) = \int p(x_1, \dots, x_D) d\mathbf{x}_{\setminus i} \quad (6.21)$$

повторным применением правила суммы, складывая все случайные величины, кроме  $x_i$ , или интегрируя по ним (значок  $\setminus i$  означает «все, кроме  $i$ »).

**ПРИМЕЧАНИЕ** Многие вычислительные задачи вероятностного моделирования обусловлены применением правила суммы. Когда случайных величин или возможных значений дискретной случайной величины много, правило суммы сводится к вычислению многомерной суммы или интеграла, что вычислительно трудно (то есть нет известного полиномиального алгоритма, дающего точный результат). ◆

Второе правило, известное как *правило произведения*, связывает совместное распределение с условным распределением по формуле

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y} | \mathbf{x})p(\mathbf{x}). \quad (6.22)$$

Правило произведения можно истолковать так: любое совместное распределение двух случайных величин можно разложить в произведение двух других распределений. Этими двумя сомножителями будут частное распределение первой случайной величины  $p(\mathbf{x})$  и условное распределение второй случайной величины при заданной первой  $p(\mathbf{y} | \mathbf{x})$ . Так как порядок записи случайных величин в  $p(\mathbf{x}, \mathbf{y})$  произволен, из правила произведения также следует, что  $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x} | \mathbf{y})p(\mathbf{y})$ . Если быть точными, (6.22) относится к распределениям дискретных случайных величин. Для непрерывных случайных величин правило произведения записывается через плотности вероятности (раздел 6.2.3).

В машинном обучении и байесовской статистике нас часто интересует вывод значений ненаблюдаемых (скрытых) случайных величин, когда мы наблюдаем

другие случайные величины. Предположим, что у нас есть некоторое предварительное (априорное) знание  $p(\mathbf{x})$  о скрытой случайной величине  $\mathbf{x}$  и некоторая зависимость  $p(\mathbf{y} | \mathbf{x})$  между  $\mathbf{x}$  и второй случайной величиной  $\mathbf{y}$ , которую мы можем наблюдать. Наблюдая  $\mathbf{y}$ , мы можем с использованием теоремы Байеса делать выводы об  $\mathbf{x}$  при условии наблюдаемых значений  $\mathbf{y}$ . *Теорема (правило) Байеса*

$$\underbrace{p(\mathbf{x} | \mathbf{y})}_{\text{апостериор.}} = \frac{\overbrace{p(\mathbf{y} | \mathbf{x})}^{\text{правдоподобие}} \overbrace{p(\mathbf{x})}^{\text{априор.}}}{\underbrace{p(\mathbf{y})}_{\text{маргинальное правдоподобие}}} \quad (6.23)$$

напрямую следует из правила произведения (6.20), так как

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x} | \mathbf{y})p(\mathbf{y}) \quad (6.24)$$

и

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y} | \mathbf{x})p(\mathbf{x}), \quad (6.25)$$

так что

$$p(\mathbf{x} | \mathbf{y})p(\mathbf{y}) = p(\mathbf{y} | \mathbf{x})p(\mathbf{x}) \Leftrightarrow p(\mathbf{x} | \mathbf{y}) = \frac{p(\mathbf{y} | \mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}. \quad (6.26)$$

В формуле (6.23)  $p(\mathbf{x})$  — *априорное распределение*, характеризующее наше субъективное исходное знание о ненаблюданной (латентной) случайной величине  $\mathbf{x}$  до получения каких-либо данных. Мы можем выбрать любое осмысленное значение априорного распределения, но важно убедиться, что плотность вероятности (или вероятность) не равна нулю на всех возможных значениях  $\mathbf{x}$ , даже очень редких.

*Правдоподобие*  $p(\mathbf{y} | \mathbf{x})$  описывает, как связаны  $\mathbf{x}$  и  $\mathbf{y}$ , и в случае дискретного распределения это вероятность данных  $\mathbf{y}$  при известном значении скрытой переменной  $\mathbf{x}$ . Заметим, что правдоподобие не будет распределением случайной величины  $\mathbf{x}$ , а только  $\mathbf{y}$ . Мы говорим о  $p(\mathbf{y} | \mathbf{x})$  как о правдоподобии  $\mathbf{x}$  при заданном  $\mathbf{y}$  или вероятности  $\mathbf{y}$  при заданном  $\mathbf{x}$ , но не о правдоподобии  $\mathbf{y}$  (MacKay, 2003).

В байесовской статистике мы ищем *апостериорное распределение*  $p(\mathbf{x} | \mathbf{y})$ , так как оно в точности выражает интересующую нас величину, то есть то, что мы знаем об  $\mathbf{x}$ , узнав  $\mathbf{y}$ .

Выражение

$$p(\mathbf{y}) := \int p(\mathbf{y} | \mathbf{x})p(\mathbf{x}) d\mathbf{x} = \mathbb{E}_{\mathbf{x}}[p(\mathbf{y} | \mathbf{x})] \quad (6.27)$$

называется *маргинальным правдоподобием*. В правой части (6.27) стоит операция взятия математического ожидания, определенная в разделе 6.4.1. По определению, маргинальное правдоподобие — это интеграл от выражения (6.23) по скрытой переменной  $\mathbf{x}$ . Следовательно, маргинальное правдоподобие не зависит от  $\mathbf{x}$ , и оно гарантирует, что апостериорная вероятность  $p(\mathbf{x} | \mathbf{y})$  нормализована. Маргинальное правдоподобие также можно истолковать как ожидание правдоподобия относительно априорного распределения  $p(\mathbf{x})$ . Кроме нормализации апостериорного распределения, маргинальное правдоподобие также играет важную роль в байесовском выборе модели, как мы обсудим в разделе 8.6. Из-за интеграла в выражении (8.44) его зачастую непросто вычислить.

Теорема Байеса (6.23) позволяет нам «обратить» зависимость  $\mathbf{y}$  от  $\mathbf{x}$ , заданную функцией правдоподобия. Поэтому ее иногда называют теоремой об обращении вероятностей. Мы еще будем обсуждать теорему Байеса в разделе 8.4.

**ПРИМЕЧАНИЕ** В байесовской статистике нас интересует апостериорное распределение как учитывающее всю доступную информацию от априорного распределения и имеющихся данных. Вместо того чтобы продолжать работать с самим апостериорным распределением, можно сосредоточиться на некоторой его статистике, например его максимуме, что мы обсудим в разделе 8.3. Однако при фокусировке на некоторой статистике распределения теряется часть информации. Например, апостериорное распределение можно использовать в системах принятия решений. И знать его целиком может быть крайне полезно для принятия решений, устойчивых к ошибкам в данных. Например, в контексте моделей обучения с подкреплением Deisenroth et al. (2015) показывают, что знание апостериорного распределения подходящих функций перехода ведет к очень быстрому (эффективному в использовании данных) обучению, тогда как использование только максимума апостериорного распределения приводит к ошибкам. Таким образом, знание апостериорного распределения целиком может быть очень полезно. В главе 9 мы продолжим обсуждение этой темы в применении к линейной регрессии. ◆

## 6.4. ОБОБЩАЮЩИЕ СТАТИСТИКИ И НЕЗАВИСИМОСТЬ

Нас часто интересует некоторое обобщение набора случайных величин или сравнение двух случайных величин. Статистика случайной величины — детерминированная функция этой величины. Обобщающие статистики распределения помогают увидеть поведение случайной величины и обобщенно охарактеризовать распределение. Мы определим среднее и дисперсию, две известнейшие статистики. Затем мы обсудим два способа сравнивать пары случайных величин:

первый — определить, что они независимы; второй — найти их скалярное произведение.

### 6.4.1. Среднее и дисперсия

Среднее и дисперсия часто используются для описания свойств распределений вероятностей (как ожидаемое значение и мера разброса). В разделе 6.6 мы увидим, что существует важное семейство распределений (экспоненциальные распределения), для которых эти статистики несут всю возможную информацию.

Понятие математического ожидания является центральным для машинного обучения, и все основные понятия самой теории вероятностей можно вывести из его свойств (Whittle, 2000).

**Определение 6.3 (математическое ожидание).** Математическое ожидание для функции  $g: \mathbb{R} \rightarrow \mathbb{R}$  от одной непрерывной случайной величины  $X \sim p(x)$  задается формулой

$$\mathbb{E}_x[g(x)] = \int_{\mathcal{X}} g(x) p(x) dx. \quad (6.28)$$

Аналогично, математическое ожидание функции  $g$  от дискретной случайной величины  $X \sim p(x)$  задается формулой

$$\mathbb{E}_x[g(x)] = \sum_{x \in \mathcal{X}} g(x) p(x), \quad (6.29)$$

где  $\mathcal{X}$  — множество возможных значений (фазовое пространство) случайной величины  $X$ .

В этом разделе мы рассматриваем дискретные случайные величины, принимающие числовые значения (как можно заметить, функция  $g$  принимает вещественные аргументы).

**ПРИМЕЧАНИЕ** Мы рассматриваем многомерные случайные величины  $X$  как векторы из конечного числа одномерных:  $[X_1, \dots, X_n]^T$ . Для многомерных случайных величин мы определяем математическое ожидание покомпонентно:

$$\mathbb{E}_x[g(\boldsymbol{x})] = \begin{bmatrix} \mathbb{E}_{x_1}[g(x_1)] \\ \vdots \\ \mathbb{E}_{x_D}[g(x_D)] \end{bmatrix} \in \mathbb{R}^D, \quad (6.30)$$

где индекс  $X_D$  показывает, что мы берем математическое ожидание относительно  $d$ -й компоненты вектора  $\boldsymbol{x}$ . ◆

В определении 6.3  $\mathbb{E}_X$  представляет собой оператор взятия интеграла по плотности вероятности (для непрерывных распределений) или суммы по всем значениям (для дискретных распределений).

Определение среднего значения (определение 6.4) является частным случаем математического ожидания от функции случайной величины, где в качестве  $g$  берется тождественная функция.

**Определение 6.4 (среднее значение).** Среднее значение случайной величины  $X$  со значениями  $\mathbf{x} \in \mathbb{R}^D$  (математическое ожидание этой величины) определяется как

$$\mathbb{E}_X[\mathbf{x}] = \begin{bmatrix} \mathbb{E}_{X_1}[x_1] \\ \vdots \\ \mathbb{E}_{X_D}[x_D] \end{bmatrix} \in \mathbb{R}^D, \quad (6.31)$$

где

$$\mathbb{E}_{x_d}[x_d] := \begin{cases} \int_{\mathcal{X}} x_d p(x_d) dx_d, & \text{если } X \text{ — непрерывная случайная переменная;} \\ \sum_{x_i \in \mathcal{X}} x_i p(x_d = x_i), & \text{если } X \text{ — дискретная случайная переменная} \end{cases} \quad (6.32)$$

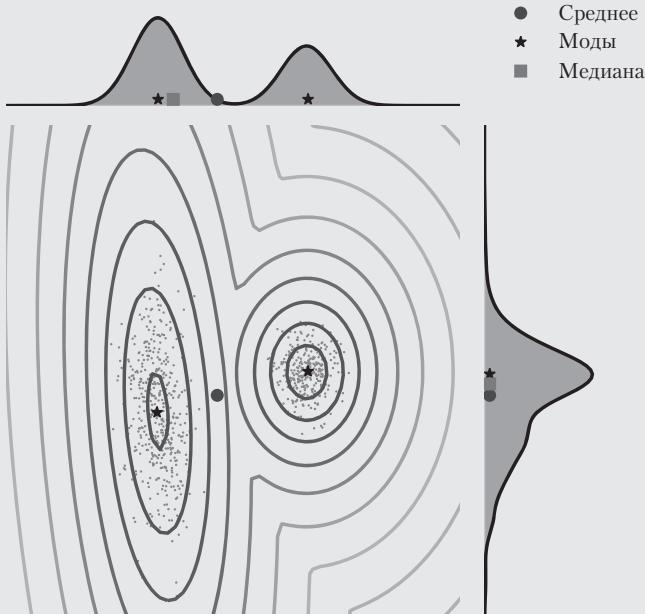
для  $d = 1, \dots, D$ , где индекс  $d$  указывает на компоненту вектора  $\mathbf{x}$ . Интеграл и сумма берутся по всем значениям  $\mathcal{X}$  фазового пространства случайной величины  $X$ .

В случае размерности 1 интуиция подсказывает еще два способа определить типичное значение величины, кроме среднего, — *медиану* и *моду*. Медиана — это «серединный» элемент среди отсортированных значений случайной величины, то есть 50% значений больше медианы, а 50% меньше медианы. Обобщить эту идею на непрерывные случайные величины можно, определив медиану как точку, в которой функция распределения (определение 6.2) равна 0,5. Для асимметричных или имеющих длинные хвосты распределений медиана оценивает типичное значение лучше (ближе к человеческому пониманию типичности), чем среднее значение. Кроме того, медиана устойчивее к выбросам, чем среднее. Обобщить медиану на многомерные распределения — непростая задача, так как отсутствует очевидный способ сортировки значений (Hallin et al., 2010; Kong and Mizera, 2012). *Мода* — это наиболее часто встречающееся значение. Для дискретной случайной величины мода определяется как значение  $x$ , встречающееся наибольшее количество раз. Для непрерывной случайной величины мода определяется как значение, соответствующее максимуму плотности  $p(\mathbf{x})$ . У функции плотности  $p(\mathbf{x})$  может быть более одной моды, и, более того, у многомерных распределений может быть очень много мод. Таким образом, нахождение всех мод распределения может быть весьма сложным вычислительно.

**Пример 6.4**

Рассмотрим два двумерных распределения, изображенных на рис. 6.4:

$$p(x) = 0,4\mathcal{N}\left(\boldsymbol{x} \left| \begin{bmatrix} 10 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right.\right) + 0,6\mathcal{N}\left(\boldsymbol{x} \left| \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 8,4 & 2,0 \\ 2,0 & 1,7 \end{bmatrix}\right.\right). \quad (6.33)$$



**Рис. 6.4.** Среднее, моды и медиана двумерного набора данных, а также частные плотности

В разделе 6.5 мы определим гауссово распределение  $\mathcal{N}(\mu, \sigma^2)$ . Также на рисунке показаны его частные распределения для каждой из компонент. Заметим, что это распределение бимодально (имеет две моды), но одно из частных распределений унимодально (имеет одну моду). Горизонтальное бимодальное одномерное распределение показывает, что среднее и медиана могут отличаться. Хотя и хочется определить двумерную медиану как пару медиан компонент, отсутствие порядка на точках в двумерном пространстве осложняет ситуацию. Говоря, что мы не можем ввести порядок, мы имеем в виду, что существует более одного способа определить отношение  $<$ , такое, что  $\begin{bmatrix} 3 \\ 0 \end{bmatrix} < \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ .

**ПРИМЕЧАНИЕ** Математическое ожидание (определение 6.3) является линейным оператором. Например, для вещественнонозначной функции  $f(\mathbf{x}) = ag(\mathbf{x}) + bh(\mathbf{x})$ , где  $a, b \in \mathbb{R}$  и  $\mathbf{x} \in \mathbb{R}^D$ , имеем

$$\mathbb{E}_{\mathbf{x}}[f(\mathbf{x})] = \int f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = \quad (6.34a)$$

$$= \int [ag(\mathbf{x}) + bh(\mathbf{x})] p(\mathbf{x}) d\mathbf{x} = \quad (6.34b)$$

$$= a \int g(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} + b \int h(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = \quad (6.34c)$$

$$= a\mathbb{E}_{\mathbf{x}}[g(\mathbf{x})] + b\mathbb{E}_{\mathbf{x}}[h(\mathbf{x})]. \quad (6.34d)$$



Иногда мы хотим описать связь между двумя случайными величинами. Ковариация характеризует, насколько сильно они зависят друг от друга.

**Определение 6.5 (ковариация (одномерная)).** Ковариация между двумя одномерными случайными величинами  $X, Y \in \mathbb{R}$  равна ожиданию произведения их отклонений от средних, то есть

$$\text{Cov}_{X,Y}[x, y] := \mathbb{E}_{X,Y}[(x - \mathbb{E}_X[x])(y - \mathbb{E}_Y[y])]. \quad (6.35)$$

**ПРИМЕЧАНИЕ** Когда ясно, какой случайной величине соответствует математическое ожидание или ковариация, индекс часто опускается (например,  $\mathbb{E}_X[x]$  записывается как  $\mathbb{E}[x]$ ). ◆

По линейности математического ожидания выражение из определения 6.5 можно переписать как матожидание произведения минус произведение матожиданий, то есть

$$\text{Cov}[x, y] = \mathbb{E}[xy] - \mathbb{E}[x]\mathbb{E}[y]. \quad (6.36)$$

Ковариация случайной величины с собой  $\text{Cov}[x, x]$  называется *дисперсией* и обозначается  $\mathbb{V}_x[x]$ . Квадратный корень из дисперсии называется *стандартным отклонением* и часто обозначается  $\sigma(x)$ . Понятие ковариации можно обобщить на многомерные случайные величины<sup>1</sup>.

**Определение 6.6 (ковариация (многомерная)).** Если  $X, Y$  – две многомерные случайные величины со значениями  $\mathbf{x} \in \mathbb{R}^D$  и  $\mathbf{y} \in \mathbb{R}^E$  соответственно, *ковариация*  $X$  и  $Y$  определяется как

$$\text{Cov}[\mathbf{x}, \mathbf{y}] = \mathbb{E}[\mathbf{x}\mathbf{y}^T] - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{y}]^T = \text{Cov}[\mathbf{y}, \mathbf{x}]^T \in \mathbb{R}^{D \times E}. \quad (6.37)$$

---

<sup>1</sup> Терминология: ковариацию многомерной случайной величины  $\text{Cov}[x, y]$  часто называют кросс-ковариацией, а ковариацией –  $\text{Cov}[x, x]$ .

Определение 6.6 можно применить и для случая, когда в роли обоих аргументов выступает одна и та же случайная величина, и результат характеризует разброс ее значений. Для многомерной случайной величины дисперсия описывает связь между ее отдельными компонентами.

**Определение 6.7 (дисперсия).** Дисперсия случайной величины  $X$  со значениями  $\mathbf{x} \in \mathbb{R}^D$  и средним  $\mu \in \mathbb{R}^D$  определяется как

$$\mathbb{V}_x[\mathbf{x}] = \text{Cov}_X[\mathbf{x}, \mathbf{x}] = \quad (6.38a)$$

$$= \mathbb{E}_x[(\mathbf{x} - \mu)(\mathbf{x} - \mu)^T] = \mathbb{E}_x[\mathbf{x}\mathbf{x}^T] - \mathbb{E}_x[\mathbf{x}]\mathbb{E}_x[\mathbf{x}]^T = \quad (6.38b)$$

$$= \begin{bmatrix} \text{Cov}[x_1, x_1] & \text{Cov}[x_1, x_2] & \dots & \text{Cov}[x_1, x_D] \\ \text{Cov}[x_2, x_1] & \text{Cov}[x_2, x_2] & \dots & \text{Cov}[x_2, x_D] \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}[x_D, x_1] & \dots & \dots & \text{Cov}[x_D, x_D] \end{bmatrix}. \quad (6.38c)$$

Матрица размера  $D \times D$  в формуле (6.38c) называется *ковариационной матрицей* многомерной случайной величины  $X$ . Ковариационная матрица симметрична и положительно определена, она несет информацию о разбросе в данных. На ее диагонали стоят дисперсии *частных распределений*

$$p(x_i) = \int p(x_1, \dots, x_D) dx_{\setminus i}, \quad (6.39)$$

где « $\setminus i$ » означает «все компоненты, кроме  $i$ ». Элементы вне диагонали — это *кросс-ковариации*  $\text{Cov}[x_i, x_j]$  для  $i, j = 1, \dots, D, i \neq j$ .

**ПРИМЕЧАНИЕ** В нашей книге мы для упрощения восприятия предполагаем, что ковариационная матрица всегда положительно определена, не рассматривая граничные случаи с положительно полуопределенной (неполного ранга) ковариационной матрицей. ◆

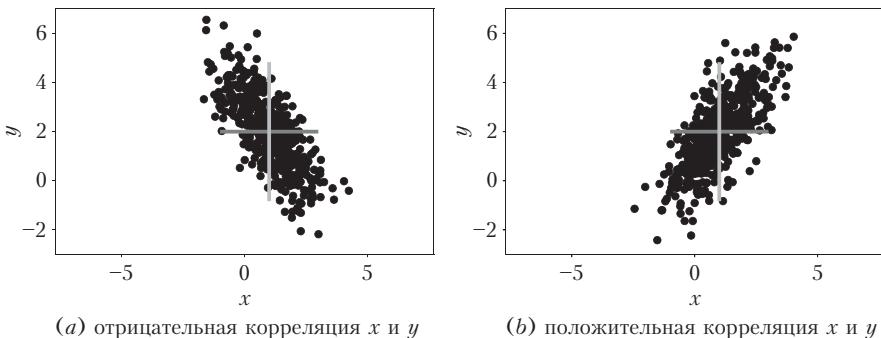
Когда мы хотим сравнить ковариации между различными парами случайных величин, оказывается, что на значение ковариации влияет дисперсия каждой из случайных величин. Нормализованная ковариация называется корреляцией.

**Определение 6.8 (корреляция).** Корреляция между двумя случайными величинами  $X, Y$  задается формулой

$$\text{corr}[x, y] = \frac{\text{Cov}[x, y]}{\sqrt{\mathbb{V}[x]\mathbb{V}[y]}} \in [-1, 1]. \quad (6.40)$$

Матрица корреляции — это ковариационная матрица стандартизованных случайных величин  $x/\sigma(x)$ . Иными словами, каждая случайная величина в корреляционной матрице делится на свое стандартное отклонение (квадратный корень из дисперсии).

Ковариация (и корреляция) показывают, насколько зависят друг от друга две случайные величины (рис. 6.5). Положительная корреляция  $\text{corr}[x, y]$  означает, что при росте  $x$  ожидается, что вырастет и  $y$ . Отрицательная корреляция показывает, что при увеличении  $x, y$  будет уменьшаться.



**Рис. 6.5.** Двумерные датасеты с идентичными средними значениями и дисперсиями вдоль обеих осей, но с разными ковариациями

### 6.4.2. Эмпирические среднее и дисперсия

Определения из раздела 6.4.1 часто также называют *средним и дисперсией генеральной совокупности*, так как они являются истинными статистиками по всей генеральной совокупности. В машинном обучении нам приходится обучать модели на эмпирических данных. Рассмотрим случайную величину  $X$ . Чтобы прийти от статистики по генеральной совокупности к эмпирической, надо прошить два шага. Сначала мы пользуемся конечностью выборки (размера  $N$ ), чтобы построить эмпирическую статистику как функцию от конечного числа одинаково распределенных случайных величин  $X_1, \dots, X_N$ . Затем применяем эту эмпирическую статистику к данным, то есть реализациям  $\mathbf{x}_1, \dots, \mathbf{x}_N$  наших случайных величин.

В частности, для среднего (определение 6.4) по выборке можно получить оценку, называемую *эмпирическим, или выборочным, средним*. То же верно и для выборочной дисперсии.

**Определение 6.9 (выборочные среднее и ковариация).** Вектор *выборочного среднего* состоит из средних арифметических наблюдений для каждой случайной величины и определяется как

$$\bar{\mathbf{x}} := \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n, \quad (6.41)$$

где  $\mathbf{x}_n \in \mathbb{R}^D$ .

Аналогично, *выборочная ковариационная матрица* — это матрица размера  $D \times D$ .

$$\Sigma := \frac{1}{N} \sum_{n=1}^N (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T. \quad (6.42)$$

Чтобы вычислить статистики для конкретной выборки, мы берем наблюдения  $\mathbf{x}_1, \dots, \mathbf{x}_N$  и подставляем в формулы (6.41) и (6.42). Выборочные ковариационные матрицы симметричны и положительно полуопределены (раздел 3.2.3)<sup>1</sup>.

### 6.4.3. Три формулы дисперсии

Сосредоточимся теперь на изучении одной случайной величины  $X$  с помощью представленных выше эмпирических формул выведем три возможные формулы для дисперсии. Аналогичные рассуждения будут верны и для дисперсии генеральной совокупности, разве что придется повозиться с интегралами<sup>2</sup>. Обычное определение дисперсии, следующее из определения ковариации (определение 6.5), — это математическое ожидание квадрата отклонения  $X$  от ее матожидания  $\mu$ , то есть

$$\mathbb{V}_x[x] := \mathbb{E}_x[(x - \mu)^2]. \quad (6.43)$$

Математическое ожидание из (6.43) и среднее  $\mu = \mathbb{E}_x(x)$  вычисляются по формулам (6.32), в зависимости от дискретности либо непрерывности  $X$ . Дисперсия по формуле (6.43) является средним для новой случайной величины  $Z := (X - \mu)^2$ .

Оценивая по выборке дисперсию с помощью формулы (6.43), мы должны дважды пройти по всей выборке: сначала вычислить среднее  $\mu$  по формуле (6.41), а затем, зная оценку  $\hat{\mu}$ , вычислить выборочную дисперсию. Оказывается, представляя слагаемые, мы можем избежать двух проходов. Формулу (6.43) можно преобразовать в такую формулу:

$$\mathbb{V}_x[x] := \mathbb{E}_x[x^2] - (\mathbb{E}_x[x])^2. \quad (6.44)$$

Выражение из (6.44) можно запомнить как «среднее от квадратов минус квадрат среднего». Его можно вычислить за один проход по выборке, так как мы можем одновременно прибавлять  $x_i$  (для вычисления среднего) и  $x_i^2$ , где  $x_i$  —  $i$ -е наблюдение. К сожалению, такая реализация может быть вычислительно неустойчивой<sup>3</sup>. Версия (6.44) может пригодиться в машинном обучении, на-

<sup>1</sup> В нашей книге мы используем выборочную ковариацию, являющуюся смешенной оценкой. Несмешенная оценка ковариации имеет в знаменателе  $N - 1$  вместо  $N$ .

<sup>2</sup> Доказательство оставлено в качестве упражнения.

<sup>3</sup> Если оба слагаемых в (6.44) большие и примерно равные, вычисления с плавающей запятой могут привести к потере точности.

пример при нахождении разложения на компоненты смещения и разброса (Bishop, 2006).

Третий способ — думать о дисперсии как о сумме попарных разностей между всеми парами наблюдений. Рассмотрим выборку  $x_1, \dots, x_N$  реализаций случайной величины  $X$  и вычислим квадраты разностей между  $x_i$  и  $x_j$ . Раскрывая скобки, можно заметить, что сумма  $N^2$  попарных разностей даст выборочную дисперсию:

$$\frac{1}{N^2} \sum_{i,j=1}^N (x_i - x_j)^2 = 2 \left[ \frac{1}{N} \sum_{i=1}^N x_i^2 - \left( \frac{1}{N} \sum_{i=1}^N x_i \right)^2 \right]. \quad (6.45)$$

Можно заметить, что (6.45) — это удвоенное выражение (6.44). Это означает, что сумму  $N^2$  попарных разностей можно представить как сумму  $N$  отклонений от среднего. Геометрически это означает, что между попарными расстояниями и расстояниями от центра множества точек есть эквивалентность. С вычислительной точки зрения это означает, что, вычисляя среднее ( $N$  слагаемых), а затем дисперсию (также  $N$  слагаемых), можно получить выражение из левой части (6.45), содержащее  $N^2$  слагаемых.

#### 6.4.4. Суммы и преобразования случайных величин

Иногда нам нужно моделировать явления, которые не описываются стандартными распределениями (некоторые из которых мы введем в разделах 6.5 и 6.6), так что приходится производить некоторые преобразования (например, складывать случайные величины).

Рассмотрим две случайные величины  $X, Y$  со значениями  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^D$ . Тогда

$$\mathbb{E}[\mathbf{x} + \mathbf{y}] = \mathbb{E}[\mathbf{x}] + \mathbb{E}[\mathbf{y}]; \quad (6.46)$$

$$\mathbb{E}[\mathbf{x} - \mathbf{y}] = \mathbb{E}[\mathbf{x}] - \mathbb{E}[\mathbf{y}]; \quad (6.47)$$

$$\mathbb{V}[\mathbf{x} + \mathbf{y}] = \mathbb{V}[\mathbf{x}] + \mathbb{V}[\mathbf{y}] + \text{Cov}[\mathbf{x}, \mathbf{y}] + \text{Cov}[\mathbf{y}, \mathbf{x}]; \quad (6.48)$$

$$\mathbb{V}[\mathbf{x} - \mathbf{y}] = \mathbb{V}[\mathbf{x}] + \mathbb{V}[\mathbf{y}] - \text{Cov}[\mathbf{x}, \mathbf{y}] - \text{Cov}[\mathbf{y}, \mathbf{x}]. \quad (6.49)$$

При аффинных преобразованиях среднее и дисперсия обладают некоторыми полезными свойствами. Рассмотрим случайную величину  $\mathbf{X}$  со средним  $\mu$  и ковариационной матрицей  $\Sigma$  и аффинное преобразование  $\mathbf{y} = A\mathbf{x} + \mathbf{b}$  значения  $\mathbf{x}$ . Тогда  $\mathbf{y}$  является случайной величиной со средним и ковариационной матрицей, заданными формулами

$$\mathbb{E}_Y[\mathbf{y}] = \mathbb{E}_X[A\mathbf{x} + \mathbf{b}] = A\mathbb{E}_X[\mathbf{x}] + \mathbf{b} = A\mu + \mathbf{b}; \quad (6.50)$$

$$\mathbb{V}_Y[\mathbf{y}] = \mathbb{V}_X[A\mathbf{x} + \mathbf{b}] = \mathbb{V}_X[A\mathbf{x}] = A\mathbb{V}_X[\mathbf{x}]A^T = A\Sigma A^T \quad (6.51)$$

соответственно. Далее,

$$\text{Cov}[\mathbf{x}, \mathbf{y}] = \mathbb{E}[\mathbf{x}(A\mathbf{x} + \mathbf{b})^T] - \mathbb{E}[\mathbf{x}]\mathbb{E}[A\mathbf{x} + \mathbf{b}]^T = \quad (6.52a)$$

$$= \mathbb{E}[\mathbf{x}]\mathbf{b}^T + \mathbb{E}[\mathbf{x}\mathbf{x}^T]A^T - \mu\mathbf{b}^T - \mu\mu^TA^T = \quad (6.52b)$$

$$= \mu\mathbf{b}^T - \mu\mathbf{b}^T + (\mathbb{E}[\mathbf{x}\mathbf{x}^T] - \mu\mu^T)A^T = \quad (6.52c)$$

$$\stackrel{(6.38b)}{=} \Sigma A^T, \quad (6.52d)$$

где  $\Sigma = \mathbb{E}[\mathbf{x}\mathbf{x}^T] - \mu\mu^T$  — ковариация  $X$ .

#### 6.4.5. Статистическая независимость

**Определение 6.10 (независимость).** Две случайные величины  $X, Y$  называются *статистически независимыми*, если и только если

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y}). \quad (6.53)$$

Неформально говоря, две случайные величины  $X$  и  $Y$  независимы, если значение  $\mathbf{y}$  не дает никакой дополнительной информации об  $\mathbf{x}$  (и наоборот).

Если  $X, Y$  статистически независимы, то

- $p(\mathbf{y} | \mathbf{x}) = p(\mathbf{y})$ ;
- $p(\mathbf{x} | \mathbf{y}) = p(\mathbf{x})$ ;
- $\mathbb{V}_{X,Y}[\mathbf{x} + \mathbf{y}] = \mathbb{V}_X[\mathbf{x}] + \mathbb{V}_Y[\mathbf{y}]$ ;
- $\text{Cov}_{X,Y}[\mathbf{x}, \mathbf{y}] = \mathbf{0}$ .

Обратное к последнему утверждению неверно, то есть две случайные величины могут не быть статистически независимыми, но иметь ковариацию 0. Чтобы понять причину, вспомним, что ковариация характеризует только линейную зависимость. Таким образом, случайные величины с нелинейной зависимостью могут иметь нулевую ковариацию.

#### Пример 6.5

Рассмотрим случайную величину  $X$  с нулевым средним ( $\mathbb{E}_x[x] = 0$ ), и пусть также  $\mathbb{E}_x[x^3] = 0$ . Положим  $y = x^2$  (таким образом,  $Y$  зависит от  $X$ ) и найдем ковариацию (6.36) между  $X$  и  $Y$ . Однако

$$\text{Cov}[x, y] = \mathbb{E}[xy] - \mathbb{E}[x]\mathbb{E}[y] = \mathbb{E}[x^3] = 0. \quad (6.54)$$

В машинном обучении часто встречаются задачи, которые можно моделировать с использованием *независимых одинаково распределенных* (*i.i.d.*) случайных величин  $X_1, \dots, X_N$ . Для более чем двух случайных величин термин «независимость» (определение 6.10) обычно относится к ситуации, когда все подмножества независимы (см. Pollard (2002, глава 4) и Jacod and Protter (2004, глава 3)). Понятие «одинаково распределенные» означает, что все эти случайные величины отвечают одному и тому же распределению.

Другим полезным для машинного обучения понятием будет условная независимость.

**Определение 6.11 (условная независимость).** Две случайные величины  $X$  и  $Y$  условно независимы при условии  $Z$ , если и только если

$$p(\mathbf{x}, \mathbf{y} | \mathbf{z}) = p(\mathbf{x} | \mathbf{z})p(\mathbf{y} | \mathbf{z}) \text{ для всех } \mathbf{z} \in \mathcal{Z}, \quad (6.55)$$

где  $\mathcal{Z}$  — множество значений случайной величины  $Z$ . Чтобы обозначить, что  $X$  и  $Y$  условно независимы при условии  $Z$ , мы пишем  $X \perp\!\!\!\perp Y | Z$ .

В определении 6.11 требуется выполнение равенства (6.55) для каждого из значений  $\mathbf{z}$ . Можно истолковать (6.55) так: «при заданном  $\mathbf{z}$  распределение  $\mathbf{x}$  и  $\mathbf{y}$  раскладывается на множители». Независимость можно рассмотреть как частный случай условной независимости:  $X \perp\!\!\!\perp Y | \emptyset$ . Используя правило произведения (6.22), можно переписать левую часть (6.55) и получить

$$p(\mathbf{x}, \mathbf{y} | \mathbf{z}) = p(\mathbf{x} | \mathbf{y}, \mathbf{z})p(\mathbf{y} | \mathbf{z}). \quad (6.56)$$

Сравнивая правую часть (6.55) с (6.56), замечаем, что  $p(\mathbf{y} | \mathbf{z})$  появляется в обеих, так что

$$p(\mathbf{x} | \mathbf{y}, \mathbf{z}) = p(\mathbf{x} | \mathbf{z}). \quad (6.57)$$

Равенство (6.57) дает нам альтернативное определение условной независимости, то есть  $X \perp\!\!\!\perp Y | Z$ . Такую запись можно понять как «если мы знаем  $\mathbf{z}$ , знание  $\mathbf{y}$  не дает нам информации об  $\mathbf{x}$ ».

#### 6.4.6. Скалярные произведения случайных величин

Вспомним определение скалярного произведения из раздела 3.2. В этой главе мы введем скалярное произведение двух случайных величин. Если у нас есть две не коррелирующие случайные величины  $X, Y$ , то

$$\mathbb{V}[x + y] = \mathbb{V}[x] + \mathbb{V}[y]. \quad (6.58)$$

Так как единица измерения дисперсии — это квадрат единицы, в которой измеряется случайная величина, выражение очень напоминает теорему Пифагора для прямоугольного треугольника  $c^2 = a^2 + b^2$ .

В дальнейшем мы увидим, как проинтерпретировать геометрически соотношение (6.58) для дисперсий коррелирующих случайных величин. Случайные величины можно рассматривать как элементы векторного пространства и, определив скалярные произведения, изучать геометрические свойства скалярных величин (Eaton, 2007). Определим скалярное произведение случайных величин<sup>1</sup>  $X$  и  $Y$  с нулевыми средними как

$$\langle X, Y \rangle := \text{Cov}[x, y]. \quad (6.59)$$

Заметим, что ковариация симметрична, положительно определена и линейна по каждому из аргументов. Нормой («длиной») случайной величины будет

$$\|X\| = \sqrt{\text{Cov}[x, x]} = \sqrt{\mathbb{V}[x]} = \sigma[x], \quad (6.60)$$

то есть ее стандартное отклонение<sup>2</sup>. Чем «длиннее» случайная величина, тем больше ее разброс (тем менее мы уверены в ее значении), а случайная величина с длиной 0 — это константа. Посмотрим на угол  $\theta$  между двумя случайными величинами  $X$  и  $Y$ :

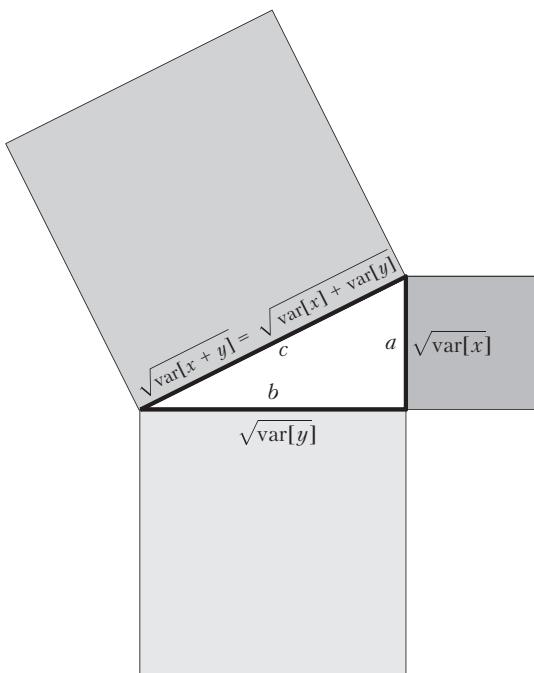
$$\cos \theta = \frac{\langle X, Y \rangle}{\|X\| \|Y\|} = \frac{\text{Cov}[x, y]}{\sqrt{\mathbb{V}[x] \mathbb{V}[y]}}, \quad (6.61)$$

Это просто корреляция (определение 6.8) между двумя случайными величинами. Это означает, что можно думать о корреляции как об угле между двумя рассматриваемыми геометрически случайными величинами. Из определения 3.7 мы знаем, что  $X \perp Y \Leftrightarrow \langle X, Y \rangle = 0$ . В нашем случае это означает, что  $X$  и  $Y$  ортогональны тогда и только тогда, когда  $\text{Cov}[x, y] = 0$ , то есть они не коррелируют. Эта связь показана на рис. 6.6.

---

<sup>1</sup> Так же можно определить скалярное произведение и для многомерных величин.

<sup>2</sup>  $\text{Cov}[x, x] = 0 \iff x = 0$ ,  $\text{Cov}[\alpha x + z, y] = \alpha \text{Cov}[x, y] + \text{Cov}[z, y]$  при  $\alpha \in \mathbb{R}$ .



**Рис. 6.6.** Геометрия случайных величин. Если случайные величины  $X$  и  $Y$  не коррелируют, их можно рассматривать как ортогональные векторы и применять теорему Пифагора

**ПРИМЕЧАНИЕ** Хотя и очень хочется для сравнения распределений вероятностей использовать евклидово расстояние (построенное на основе введенного ранее определения скалярного произведения), но это не лучший способ определить расстояние между распределениями. Вспомним, что вероятности (или плотность распределения) положительны и должны в сумме давать 1. Эти ограничения означают, что распределения «живут» на так называемом пространстве статистического многообразия. Изучение этого пространства распределений вероятностей называют информационной геометрией. Часто расстоянием между распределениями считают расхождение Кульбака – Лейблера, обобщающее те расстояния, которые задают свойства статистического многообразия. Подобно тому как евклидово расстояние является частным случаем метрики (раздел 3.3), расхождение Кульбака – Лейблера является частным случаем двух более общих классов: расхождения Брегмана и  $f$ -расхождения. Тема расхождений выходит за пределы содержания нашей книги, так что за подробностями отсылаем читателя к недавно вышедшей книге Amari (2016), одного из основоположников информационной геометрии. ◆

## 6.5. ГАУССОВО РАСПРЕДЕЛЕНИЕ

Гауссово распределение — самое изученное распределение вероятностей для непрерывных случайных величин<sup>1</sup>. Его также называют *нормальным распределением*. Оно столь важно из-за своих удобных для вычислений свойств, которые мы обсудим в дальнейшем. В частности, мы будем использовать его при определении правдоподобия и априорных вероятностей в задаче линейной регрессии (глава 9), а для оценки плотности рассматривать смесь гауссовых распределений (глава 11).

Гауссово распределение важно и для других областей машинного обучения, например для глубокого обучения, изучения гауссовых процессов и вариационного вывода. Оно широко используется и в других прикладных областях, таких как обработка сигналов (например, фильтр Калмана), управлении (например, линейно-квадратичный регулятор) и статистике (проверка гипотез).

Для одномерной случайной величины плотность гауссова распределения задается формулой

$$p(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right). \quad (6.62)$$

*Многомерное гауссово распределение* полностью определяется вектором средних  $\mu$  и ковариационной матрицей  $\Sigma$  по формуле

$$p(\mathbf{x} | \mu, \Sigma) = (2\pi)^{-\frac{D}{2}} |\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)\right), \quad (6.63)$$

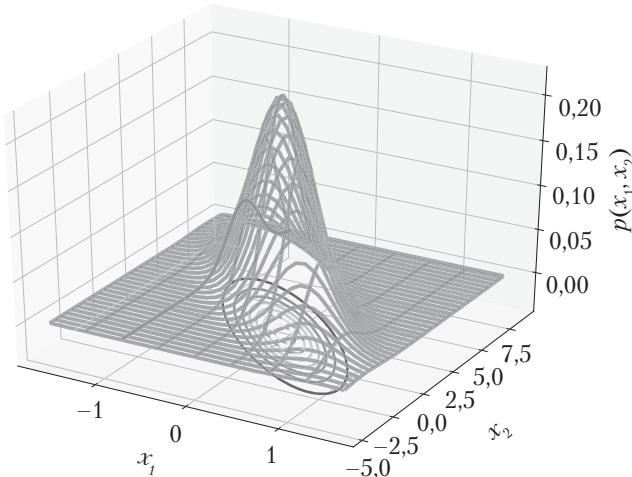
где  $\mathbf{x} \in \mathbb{R}^D$ . Мы записываем  $p(\mathbf{x}) = \mathcal{N}(\mathbf{x} | \mu, \Sigma)$  или  $X \sim \mathcal{N}(\mu, \Sigma)$ . На рис. 6.7 изображено гауссово распределение от двух переменных и соответствующий контурный график. На рис. 6.8 изображены гауссовые распределения — одномерное и двумерное — вместе с соответствующими выборками. Частный случай гауссова распределения с нулевым средним и единичной дисперсией известен как *стандартное нормальное распределение*.

Гауссианы широко применяются в статистическом оценивании и в машинном обучении, так как дают явные выражения для частных и условных распределений. В главе 9 мы будем активно использовать эти явные выражения применительно к линейной регрессии. Преимуществом моделирования с использованием гауссовых распределений является то, что зачастую не нужны преобразования переменных (раздел 6.7). Так как гауссово распределение

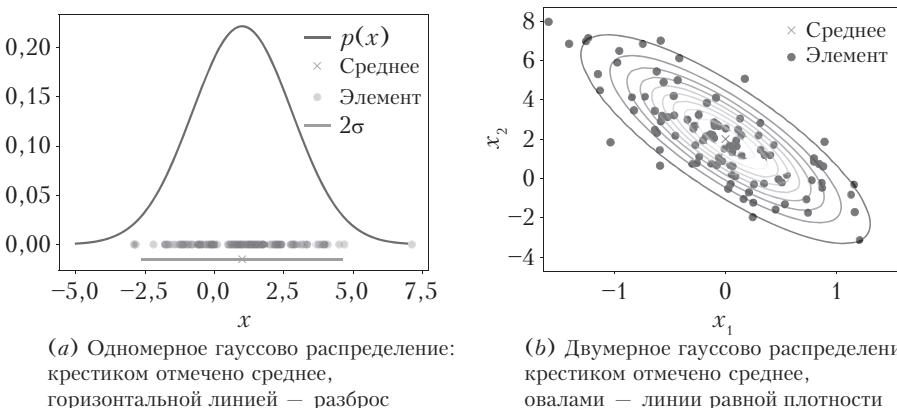
---

<sup>1</sup> Гауссово распределение естественным образом возникает при рассмотрении сумм независимых одинаково распределенных случайных величин. Это утверждение известно как центральная предельная теорема (Grinstead and Snell, 1997).

полностью задается своими средним и дисперсией, нам часто достаточно применить преобразования к среднему и ковариации исходной величины.



**Рис. 6.7.** Гауссово распределение от двух случайных величин  $x_1$  и  $x_2$



**Рис. 6.8.** Гауссовые распределения и выборка из 100 элементов:

(a) одномерный случай; (b) двумерный случай

### 6.5.1. Частные и условные распределения — тоже гауссианы

В этом разделе мы рассмотрим переход к частным и условным распределениям для многомерных случайных величин. Если при первом прочтении материал кажется слишком сложным, советуем рассмотреть вариант с двумя случайными

величинами. Пусть  $X$  и  $Y$  — две многомерные случайные величины, возможно, с разными распределениями. Чтобы увидеть эффект от применения правила суммы и перехода к условным вероятностям, запишем гауссово распределение для пар  $[\mathbf{x}, \mathbf{y}]^T$  в явном виде:

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{bmatrix}\right), \quad (6.64)$$

где  $\Sigma_{xx} = \text{Cov}[\mathbf{x}, \mathbf{x}]$  и  $\Sigma_{yy} = \text{Cov}[\mathbf{y}, \mathbf{y}]$  — частные ковариационные матрицы  $x$  и  $y$  соответственно, а  $\Sigma_{xy} = \text{Cov}[\mathbf{x}, \mathbf{y}]$  — матрица кросс-ковариации между  $\mathbf{x}$  и  $\mathbf{y}$ .

Условное распределение  $p(\mathbf{x} | \mathbf{y})$  также гауссово (показано на рис. 6.9(c)) и задается формулами (вывод которых можно увидеть в разделе 2.3 книги Bishop, 2006)

$$p(\mathbf{x} | \mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_{x|y}, \Sigma_{x|y}); \quad (6.65)$$

$$\boldsymbol{\mu}_{x|y} = \boldsymbol{\mu}_x + \Sigma_{xy} \Sigma_{yy}^{-1} (\mathbf{y} - \boldsymbol{\mu}_y); \quad (6.66)$$

$$\Sigma_{x|y} = \Sigma_{xx} - \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{yx}. \quad (6.67)$$

Заметим, что при вычислении среднего в (6.66), значение  $\mathbf{y}$  нам известно.

**ПРИМЕЧАНИЕ** Условное гауссово распределение часто появляется в задачах, где нас интересует апостериорная вероятность:

- Фильтр Калмана (Kalman, 1960), один из главных алгоритмов оценки состояний при обработке сигналов, всего лишь вычисляет гауссова условные вероятности совместных распределений (Deisenroth and Ohlsson, 2011; Särkkä, 2013).
- Гауссовые процессы (Rasmussen and Williams, 2006) реализуют на практике распределения на функциях. В гауссовом процессе мы предполагаем совместную гауссость случайных величин. Переходя к условным вероятностям, мы можем найти апостериорное распределение на функциях.
- Скрытые линейные гауссовые модели (Roweis and Ghahramani, 1999; Murphy, 2012), включая вероятностный анализ главных компонент (probabilistic principal component analysis, PPCA) (Tipping and Bishop, 1999). Мы подробнее обсудим PPCA в разделе 10.7.



Частное распределение  $p(\mathbf{x})$  для гауссова распределения  $p(\mathbf{x}, \mathbf{y})$  (6.64) само будет гауссовым, может быть вычислено по правилу суммы (6.20) и задается формулой

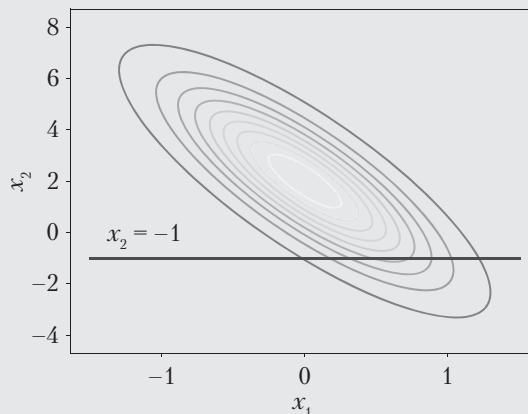
$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{y}) d\mathbf{y} = \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_x, \Sigma_{xx}). \quad (6.68)$$

Аналогичный результат верен для  $p(\mathbf{y})$ . Неформально говоря, мы смотрим на совместное распределение (6.64) и игнорируем все, что нас не интересует (производя интегрирование). Иллюстрацией служит рис. 6.9(б).

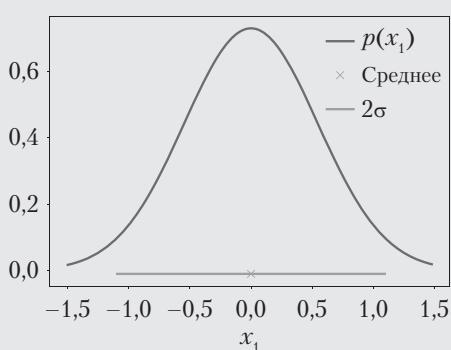
### Пример 6.6

Рассмотрим двумерное гауссово распределение (показанное на рис. 6.9):

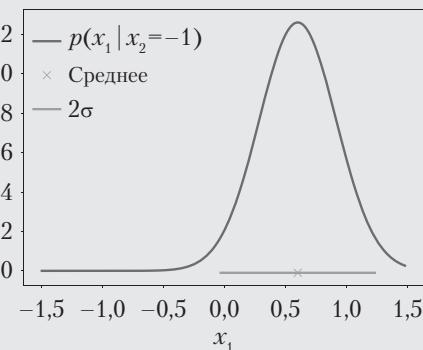
$$p(x_1, x_2) = \mathcal{N}\left(\begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 0,3 & -1 \\ -1 & 5 \end{bmatrix}\right). \quad (6.69)$$



(a) Двумерное гауссово распределение



(b) Частное распределение



(c) Условное распределение

**Рис. 6.9.** (a) Двумерное гауссово распределение. (b) Частное распределение для совместного гауссова распределения — гауссово. (c) Условное распределение также гауссово

Можно вычислить параметры одномерного гауссова распределения при условии  $x_2 = -1$ , применяя (6.66) и (6.67) для среднего и дисперсии соответственно. Мы получаем

$$\mu_{x_1|x_2=-1} = 0 + (-1) \cdot 0,2 \cdot (-1 - 2) = 0,6 \quad (6.70)$$

и

$$\sigma^2_{x_1|x_2=-1} = 0,3 - (-1) \cdot 0,2 \cdot (-1) = 0,1. \quad (6.71)$$

Таким образом, условное гауссово распределение выглядит как

$$p(x_1 | x_2 = -1) = \mathcal{N}(0,6, 0,1). \quad (6.72)$$

Частное распределение  $p(x_1)$ , напротив, можно получить по формуле (6.68), использующей значения среднего и дисперсии для  $x_1$ :

$$p(x_1) = \mathcal{N}(0, 0,3). \quad (6.73)$$

### 6.5.2. Произведение гауссовых плотностей

При построении модели линейной регрессии (глава 9) нам необходимо вычислить гауссово правдоподобие. Возможно, мы также хотим взять гауссово априорное распределение (раздел 9.3). Для вычисления апостериорного распределения мы применяем теорему Байеса, так что приходится умножать правдоподобие на априорную вероятность. Нам надо найти произведение двух гауссовых плотностей. Перемножив  $\mathcal{N}(\mathbf{x} | \mathbf{a}, \mathbf{A})$  и  $\mathcal{N}(\mathbf{x} | \mathbf{b}, \mathbf{B})$ , получим (с точностью до умножения на  $c \in \mathbb{R}$ ) гауссово распределение  $c\mathcal{N}(\mathbf{x} | \mathbf{c}, \mathbf{C})$ <sup>1</sup>, где

$$\mathbf{C} = (\mathbf{A}^{-1} + \mathbf{B}^{-1})^{-1}; \quad (6.74)$$

$$\mathbf{c} = \mathbf{C}(\mathbf{A}^{-1}\mathbf{a} + \mathbf{B}^{-1}\mathbf{b}); \quad (6.75)$$

$$c = (2\pi)^{-\frac{D}{2}} |\mathbf{A} + \mathbf{B}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{a} - \mathbf{b})^\top (\mathbf{A} + \mathbf{B})^{-1} (\mathbf{a} - \mathbf{b})\right). \quad (6.76)$$

Константу  $c$  также можно записать в виде гауссовой плотности от  $\mathbf{a}$  или  $\mathbf{b}$  с ковариационной матрицей  $\mathbf{A} + \mathbf{B}$ , то есть  $c = \mathcal{N}(\mathbf{a} | \mathbf{b}, \mathbf{A} + \mathbf{B}) = \mathcal{N}(\mathbf{b} | \mathbf{a}, \mathbf{A} + \mathbf{B})$ .

**ПРИМЕЧАНИЕ** Для удобства обозначений мы иногда будем использовать функциональную запись  $\mathcal{N}(\mathbf{x} | \mathbf{m}, \mathbf{S})$  даже в том случае, когда  $\mathbf{x}$  не является случайной величиной. Мы только что поступили именно так, записав

$$c = \mathcal{N}(\mathbf{a} | \mathbf{b}, \mathbf{A} + \mathbf{B}) = \mathcal{N}(\mathbf{b} | \mathbf{a}, \mathbf{A} + \mathbf{B}). \quad (6.77)$$

---

<sup>1</sup> Доказательство оставлено в качестве упражнения.

В этой формуле ни  $\mathbf{a}$ , ни  $\mathbf{b}$  не являются случайными величинами, но зато такая запись компактнее, чем (6.76). ◆

### 6.5.3. Суммы и линейные преобразования

Если  $X, Y$  — независимые гауссовые случайные величины (то есть совместное распределение задается как  $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$ , где  $p(\mathbf{x}) = \mathcal{N}(\mathbf{x} | \mu_x, \Sigma_x)$  и  $p(\mathbf{y}) = \mathcal{N}(\mathbf{y} | \mu_y, \Sigma_y)$ ), то  $\mathbf{x} + \mathbf{y}$  — также гауссова случайная величина и задается формулой

$$p(\mathbf{x} + \mathbf{y}) = \mathcal{N}(\mu_x + \mu_y, \Sigma_x + \Sigma_y). \quad (6.78)$$

Зная, что  $p(\mathbf{x} + \mathbf{y})$  — гауссова, мы можем сразу вычислить среднее и ковариационную матрицу по формулам (6.46)–(6.49). Это свойство будет важным, когда мы будем рассматривать независимые одинаково распределенные шумы, действующие на случайные величины, — например, в задаче линейной регрессии (глава 9).

#### Пример 6.7

Так как математическое ожидание является линейной операцией, можно рассмотреть взвешенную сумму независимых гауссовых случайных величин:

$$p(a\mathbf{x} + b\mathbf{y}) = \mathcal{N}(a\mu_x + b\mu_y, a^2\Sigma_x + b^2\Sigma_y). \quad (6.79)$$

**ПРИМЕЧАНИЕ** В главе 11 нам понадобится взвешенная сумма гауссовых плотностей. Это не то же самое, что взвешенная сумма гауссовых случайных величин. ◆

В теореме 6.12 случайная величина  $x$  принадлежит распределению, плотность которого является смесью плотностей  $p_1(x)$  и  $p_2(x)$ , с весом первого  $\alpha$ . Эту теорему можно обобщить на случай многомерных случайных величин, так как для них также выполняется свойство линейности математического ожидания. Однако возвведение в квадрат надо будет заменить на  $\mathbf{x}\mathbf{x}^\top$ .

**Теорема 6.12.** *Рассмотрим смесь двух одномерных гауссовых плотностей*

$$p(x) = \alpha p_1(x) + (1 - \alpha)p_2(x), \quad (6.80)$$

где константа  $0 < \alpha < 1$  — вес смеси, а  $p_1(x)$  и  $p_2(x)$  — одномерные гауссовые плотности распределения (6.62) с различными параметрами, то есть  $(\mu_1, \sigma_1^2) \neq (\mu_2, \sigma_2^2)$ .

Тогда среднее для смешанной плотности  $p(x)$  равно взвешенной сумме средних:

$$\mathbb{E}[x] = \alpha\mu_1 + (1 - \alpha)\mu_2. \quad (6.81)$$

Дисперсия для смешанной плотности  $p(x)$  задается формулой

$$\mathbb{V}[x] = [\alpha\sigma_1^2 + (1 - \alpha)\sigma_2^2] + \left( [\alpha\mu_1^2 + (1 - \alpha)\mu_2^2] - [\alpha\mu_1 + (1 - \alpha)\mu_2]^2 \right). \quad (6.82)$$

*Доказательство.* Среднее для смешанной плотности  $p(x)$  равно взвешенной сумме средних двух случайных величин. Применим определение среднего (определение 6.4), и подставим в формулу смеси (6.80), получив

$$\mathbb{E}[x] = \int_{-\infty}^{\infty} xp(x)dx = \quad (6.83a)$$

$$= \int_{-\infty}^{\infty} \alpha xp_1(x) + (1 - \alpha)xp_2(x)dx = \quad (6.83b)$$

$$= \alpha \int_{-\infty}^{\infty} xp_1(x)dx + (1 - \alpha) \int_{-\infty}^{\infty} xp_2(x)dx = \quad (6.83c)$$

$$= \alpha\mu_1 + (1 - \alpha)\mu_2. \quad (6.83d)$$

Для вычисления дисперсии мы можем использовать формулу (6.44), требующую найти матожидание квадрата случайной величины. Будем использовать определение матожидания функции (в данном случае квадрата) от случайной величины (определение 6.3):

$$\mathbb{E}[x^2] = \int_{-\infty}^{\infty} x^2 p(x)dx = \quad (6.84a)$$

$$= \int_{-\infty}^{\infty} \alpha x^2 p_1(x) + (1 - \alpha)x^2 p_2(x)dx = \quad (6.84b)$$

$$= \alpha \int_{-\infty}^{\infty} x^2 p_1(x)dx + (1 - \alpha) \int_{-\infty}^{\infty} x^2 p_2(x)dx = \quad (6.84c)$$

$$= \alpha(\mu_1^2 + \sigma_1^2) + (1 - \alpha)(\mu_2^2 + \sigma_2^2), \quad (6.84d)$$

где в последнем равенстве мы снова пользовались формулой (6.44), по которой  $\sigma^2 = \mathbb{E}[x^2] - \mu^2$ . Перенесем  $\mu^2$  в левую часть и получим, что математическое ожидание квадрата случайной величины равно сумме квадрата ее матожидания и ее дисперсии.

Таким образом, вычитаем (6.83d) из (6.84d) и получаем дисперсию

$$\mathbb{V}[x] = \mathbb{E}[x^2] - (\mathbb{E}[x])^2 = \quad (6.85a)$$

$$= \alpha(\mu_1^2 + \sigma_1^2) + (1-\alpha)(\mu_2^2 + \sigma_2^2) - (\alpha\mu_1 + (1-\alpha)\mu_2)^2 = \\ = [\alpha\sigma_1^2 + (1-\alpha)\sigma_2^2] +$$
(6.85b)

$$+ ([\alpha\mu_1^2 + (1-\alpha)\mu_2^2] - [\alpha\mu_1 + (1-\alpha)\mu_2]^2). \quad (6.85c)$$

□

**ПРИМЕЧАНИЕ** Вычисления выше верны для любой плотности, но так как гауссова плотность однозначно задается своими средним и дисперсией, плотность смеси можно записать в явном виде. ♦

Для смешанной плотности отдельные компоненты можно рассмотреть как условные распределения. Уравнение (6.85c) является примером формулы для условной дисперсии, известной также как *закон полной дисперсии*, которая в общем случае утверждает, что для двух случайных величин  $X$  и  $Y$  выполняется равенство  $\mathbb{V}_X[x] = \mathbb{E}_Y[\mathbb{V}_X[x|y]] + \mathbb{V}_Y[\mathbb{E}_X[x|y]]$ , то есть (полная) дисперсия  $X$  равна матожиданию условной дисперсии плюс дисперсии условного среднего.

В примере 6.17 мы рассматривали стандартную двумерную гауссову случайную величину  $X$  и применяли к ней линейное преобразование  $Ax$ . Результатом являлась гауссова случайная величина со средним, равным нулю, и ковариацией  $AA^T$ . Заметим, что прибавление константного вектора изменит среднее распределения, но не его дисперсию, то есть случайная величина  $x + \mu$  гауссова со средним  $\mu$  и единичной ковариационной матрицей. Таким образом, при любом линейном или аффинном преобразовании гауссова случайная величина остается гауссовой.

Рассмотрим гауссову случайную величину  $X \sim \mathcal{N}(\mu, \Sigma)$ . Для заданной матрицы  $A$  подходящего размера пусть  $Y$  — случайная величина, полученная преобразованием  $y = Ax$ . Среднее для  $y$  мы можем вычислить, пользуясь линейностью матожидания (6.50):

$$\mathbb{E}[y] = \mathbb{E}[Ax] = A\mathbb{E}[x] = A\mu. \quad (6.86)$$

Аналогично, дисперсию  $y$  можно найти по формуле (6.51):

$$\mathbb{V}[y] = \mathbb{V}[Ax] = A\mathbb{V}[x]A^T = A\Sigma A^T. \quad (6.87)$$

Таким образом, распределение случайной величины  $y$  задается как

$$p(y) = \mathcal{N}(y | A\mu, A\Sigma A^T). \quad (6.88)$$

Рассмотрим теперь обратное преобразование: пусть мы знаем, что среднее случайной величины задано линейным преобразованием другой случайной вели-

чины. Пусть для матрицы  $\mathbf{A} \in \mathbb{R}^{M \times N}$  полного ранга, где  $M \geq N$ ,  $\mathbf{y} \in \mathbb{R}^M$  — гауссова случайная величина со средним  $\mathbf{Ax}$ , то есть

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y} | \mathbf{Ax}, \Sigma). \quad (6.89)$$

Как выглядит соответствующее распределение  $p(\mathbf{x})$ ? Если матрица  $\mathbf{A}$  обратима, мы можем записать  $\mathbf{x} = \mathbf{A}^{-1}\mathbf{y}$  и применить преобразование из предыдущего параграфа. Однако в общем случае  $\mathbf{A}$  не обратима, и мы воспользуемся подходом, похожим на введение псевдообратных матриц (3.57). А именно, мы домножим обе части на  $\mathbf{A}^T$ , а затем найдем обратную к симметричной положительно определенной  $\mathbf{A}^T\mathbf{A}$  и получим

$$\mathbf{y} = \mathbf{Ax} \Leftrightarrow (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y} = \mathbf{x}. \quad (6.90)$$

Так как  $\mathbf{x}$  получена из  $\mathbf{y}$  линейным преобразованием, получаем

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x} | \mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}, (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\Sigma\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}. \quad (6.91)$$

#### 6.5.4. Семплирование из многомерного гауссова распределения

Мы не будем углубляться в тонкости компьютерного семплирования, а заинтересованный читатель может обратиться к книге Gentle (2004). В случае многомерного гауссова распределения процесс состоит из трех этапов: сначала с помощью псевдослучайного генератора получить элемент равномерного распределения на интервале  $[0,1]$ ; затем применить нелинейное преобразование, например преобразование Бокса — Мюллера (Devroye, 1986) и получить элемент из одномерного гауссова распределения, и, наконец, объединить результаты в вектор из многомерного стандартного нормального распределения  $\mathcal{N}(0, \mathbf{I})$ .

В общем случае для многомерного распределения Гаусса (когда среднее может быть отлично от нуля, а ковариационная матрица — от единичной) будем использовать свойства линейных преобразований гауссовых случайных величин. Пусть мы хотим сгенерировать выборку  $\mathbf{x}_i, i = 1, \dots, n$ , из многомерного гауссова распределения со средним  $\mu$  и ковариационной матрицей  $\Sigma$ . Мы сгенерируем выборку из многомерного стандартного нормального распределения  $\mathcal{N}(0, \mathbf{I})$ . Чтобы получить выборку из распределения  $\mathcal{N}(\mu, \Sigma)$ , применим свойства линейных преобразований гауссовых случайных величин. Если  $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$ , то  $\mathbf{y} = \mathbf{Ax} + \mu$ , где  $\mathbf{AA}^T = \Sigma$ , будет гауссовой случайной величиной со средним  $\mu$

и ковариационной матрицей  $\Sigma$ . Удобно, например, использовать  $A$  из разложения Холецкого (раздел 4.3) ковариационной матрицы  $\Sigma = AA^T$ . Преимуществом такого выбора будет то, что  $A$  из разложения Холецкого треугольна, что упрощает вычисления<sup>1</sup>.

## 6.6. СОПРЯЖЕННОСТЬ И ЭКСПОНЕНЦИАЛЬНОЕ СЕМЕЙСТВО РАСПРЕДЕЛЕНИЙ

Многие «именные» распределения вероятностей из учебников статистики были придуманы для моделирования конкретных типов явлений. Например, в разделе 6.5 мы встретились с гауссовым распределением. Распределения порой сложным образом связаны друг с другом (Leemis and McQueston, 2008). Для начинающего выбор подходящего распределения может быть чрезвычайно трудной задачей. Кроме того, многие распределения были придуманы во времена, когда вычисления делались вручную. Естественно задаться вопросом, какие понятия и идеи полезны в эру компьютеров (Efron and Hastie, 2016). В предыдущем разделе мы увидели, что многие вычисления удобно производить с гауссовыми распределениями. Здесь стоит напомнить наши пожелания к распределениям, используемым в машинном обучении:

1. При применении таких утверждений, как теорема Байеса, должно выполняться «свойство замкнутости» — применение некоторой операции должно давать распределение того же типа.
2. При получении большего количества данных нам не должны требоваться дополнительные параметры для описания распределения.
3. Так как нам интересно обучение на основе данных, мы хотим, чтобы хорошо работала оценка параметров.

Класс распределений, называемый *экспоненциальным семейством*, обеспечивает сочетание широкой применимости с удобством для вычислений и статистического вывода. Перед тем как мы познакомимся с экспоненциальным семейством, однако, рассмотрим еще три «именных» распределения вероятностей: распределение Бернулли (пример 6.8), биномиальное распределение (пример 6.9) и бета-распределение (пример 6.10).

---

<sup>1</sup> Чтобы вычислить разложение Холецкого, необходимо, чтобы матрица была симметрична и положительно определена. Ковариационные матрицы обладают этими свойствами.

**Пример 6.8**

*Распределение Бернулли* — это распределение случайной величины  $X$ , принимающей значения  $x \in \{0, 1\}$ . Оно задается одним параметром  $\mu \in [0, 1]$  — вероятностью того, что  $X = 1$ . Формулы для распределения Бернулли  $Ber(\mu)$ :

$$p(x | \mu) = \mu^x(1 - \mu)^{1-x}, x \in \{0, 1\}; \quad (6.92)$$

$$\mathbb{E}[x] = \mu; \quad (6.93)$$

$$\mathbb{V}[x] = \mu(1 - \mu), \quad (6.94)$$

где  $\mathbb{E}[x]$  и  $\mathbb{V}[x]$  — среднее и дисперсия случайной величины  $X$ .

Примером применения распределения Бернулли может служить моделирование бросания монетки (когда нас интересует, например, вероятность выпадения орла).

**ПРИМЕЧАНИЕ** Запись распределения Бернулли, в которой булевы случайные величины представлены как 0 и 1 и стоят в показателе степени, часто используется в книгах по машинному обучению. Аналогичная запись используется для мультиномиального распределения. ♦

**Пример 6.9 (биномиальное распределение)**

*Биномиальное распределение* является обобщением распределения Бернулли, но значения случайной величины уже будут целыми (рис. 6.10). В частности, биномиальным распределением можно описать вероятность  $m$  случаев с  $X = 1$  среди реализаций  $N$  бернуллиевской случайной величины с  $p(X = 1) = \mu \in [0, 1]$ . Биномиальное распределение  $Bin(N, \mu)$  задается формулами

$$p(m | N, \mu) = \binom{N}{m} \mu^m (1 - \mu)^{N-m}, \quad (6.95)$$

$$\mathbb{E}[m] = N\mu, \quad (6.96)$$

$$\mathbb{V}[m] = N\mu(1 - \mu), \quad (6.97)$$

где  $\mathbb{E}[m]$  и  $\mathbb{V}[m]$  — соответственно среднее и дисперсия  $m$ . Биномиальное распределение является обобщением распределения Бернулли, фазовым пространством для которого является множество целых чисел (оно показано на рис. 6.10).

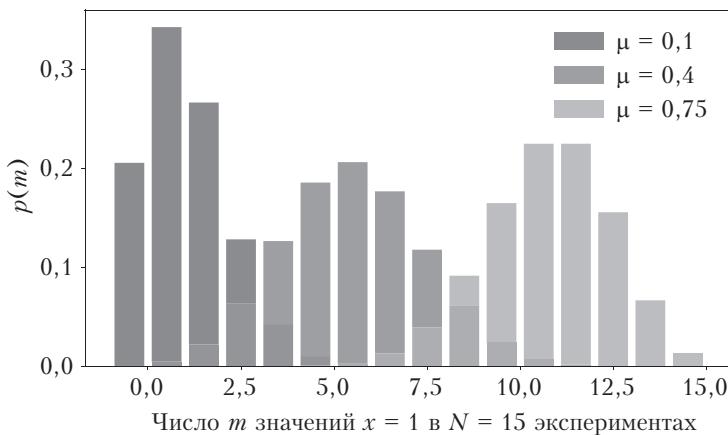


Рис. 6.10. Примеры биномиальных распределений для  $\mu \in \{0,1, 0,4, 0,75\}$  и  $N=15$

Пример использования биномиального распределения: пусть мы хотим описать вероятность получить  $m$  орлов в эксперименте из  $N$  подбрасываний монеты, если вероятность выпадения орла при одном броске равна  $\mu$ .

### Пример 6.10 (бета-распределение)

Иногда нам надо моделировать непрерывную случайную величину со значениями на конечном интервале.

*Бета-распределение* — это распределение непрерывной случайной величины  $\mu \in [0, 1]$ , которая зачастую используется для нахождения вероятности бинарного события (например, она может быть параметром распределения Бернулли). Само бета-распределение Beta( $\alpha, \beta$ ) (показано на рис. 6.11) задается двумя параметрами  $\alpha > 0, \beta > 0$  и определяется как

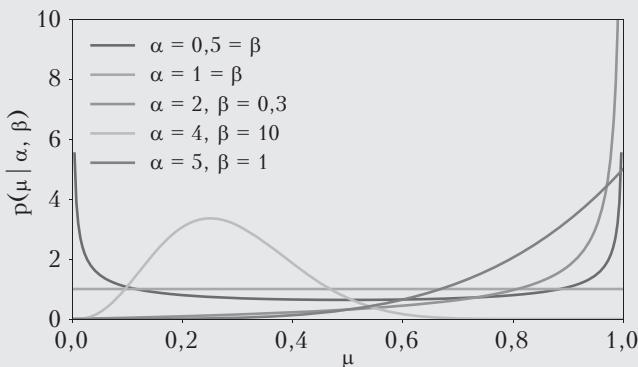
$$p(\mu | \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \mu^{\alpha-1} (1-\mu)^{\beta-1}; \quad (6.98)$$

$$\mathbb{E}[\mu] = \frac{\alpha}{\alpha + \beta}, \quad \mathbb{V}[\mu] = \frac{\alpha\beta}{(\alpha + \beta)^2 (\alpha + \beta + 1)}, \quad (6.99)$$

где  $\Gamma(\cdot)$  — гамма-функция, определяемая как

$$\Gamma(t) := \int_0^\infty x^{t-1} \exp(-x) dx, \quad t > 0. \quad (6.100)$$

$$\Gamma(t+1) = t\Gamma(t). \quad (6.101)$$



**Рис. 6.11.**  
Примеры  
бета-распределе-  
ний для различ-  
ных  $\alpha$  и  $\beta$

Заметим, что гамма-функции в знаменателе (6.98) нормируют бета-распределение.

Неформально говоря,  $\alpha$  сдвигает распределение к 1, а  $\beta$  – к 0. Частными случаями (Murphy, 2012) являются:

- При  $\alpha = 1 = \beta$  получаем равномерное распределение  $\mathcal{U}[0, 1]$ .
- При  $\alpha, \beta < 1$  получаем бимодальное распределение с пиками 0 и 1.
- При  $\alpha, \beta > 1$  распределение унимодально.
- При  $\alpha, \beta > 1$  и  $\alpha = \beta$  распределение унимодально, симметрично и центрировано на  $[0, 1]$ , то есть среднее совпадает с модой и равно  $1/2$ .

**ПРИМЕЧАНИЕ** Именные распределения – это целый зоопарк с многообразными связями между собой (Leemis and McQueston, 2008). Стоит иметь в виду, что каждое из именных распределений создавалось под определенную задачу, но может иметь и другие приложения. Знание, для каких задач было определено то или иное распределение, часто помогает на практике. Описанные три распределения мы ввели для того, чтобы на примерах продемонстрировать понятия сопряженности (раздел 6.6.1) и экспоненциальных семейств (раздел 6.6.3). ♦

### 6.6.1. Сопряженность

Согласно теореме Байеса (6.23), апостериорная вероятность пропорциональна произведению априорной вероятности на правдоподобие. Введение априорного распределения сложно по двум причинам. Во-первых, оно должно отражать наши знания о задаче, до того как мы получим данные. Описать их не всегда легко. Во-вторых, апостериорное распределение часто невозможно найти ана-

литически. Однако существуют удобные для вычислений априорные распределения — сопряженные.

**Определение 6.13 (сопряженное априорное распределение).** Априорное распределение является *сопряженным* к функции правдоподобия, если апостериорное распределение имеет тот же тип, что и априорное.

Сопряженность особенно удобна, поскольку мы можем найти апостериорное распределение, просто пересчитав параметры априорного.

**ПРИМЕЧАНИЕ** С геометрической точки зрения на распределение вероятностей при переходе от правдоподобия к сопряженному распределению сохраняются расстояния (Agarwal and Daumé III, 2010). ◆

В качестве конкретного примера сопряженного априорного распределения рассмотрим в примере 6.11 биномиальное распределение дискретной случайной величины и бета-распределение непрерывной случайной величины.

### Пример 6.11 (бета-биномиальная сопряженность)

Рассмотрим биномиальную случайную величину  $x \sim \text{Bin}(N, \mu)$ , где

$$p(x | N, \mu) = \binom{N}{x} \mu^x (1-\mu)^{N-x}, \quad x = 0, 1, \dots, N, \quad (6.102)$$

— вероятность выпадения  $x$  орлов при  $N$  бросках монеты ( $\mu$  — вероятность выпадения орла). Берем в качестве априорного распределения параметра  $\mu$  бета-распределение, то есть  $\mu \sim \text{Beta}(\alpha, \beta)$ , где

$$p(\mu | \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \mu^{\alpha-1} (1-\mu)^{\beta-1}. \quad (6.103)$$

Пусть мы наблюдаем значение  $x = h$ , то есть за  $N$  бросков выпало  $h$  орлов. Апостериорное распределение  $\mu$  будет выглядеть как

$$p(\mu | x = h, N, \alpha, \beta) \propto p(x | N, \mu) p(\mu | \alpha, \beta) \quad (6.104a)$$

$$\propto \mu^h (1-\mu)^{(N-h)} \mu^{\alpha-1} (1-\mu)^{\beta-1} \quad (6.104b)$$

$$= \mu^{h+\alpha-1} (1-\mu)^{(N-h)+\beta-1} \quad (6.104c)$$

$$\propto \text{Beta}(h + \alpha, N - h + \beta), \quad (6.104d)$$

то есть оно, как и априорное, будет бета-распределением. Таким образом, априорное бета-распределение для параметра  $\mu$  является сопряженным к биномиальной функции правдоподобия.

В следующем примере мы придем к похожему выводу — покажем, что бета-распределение является сопряженным априорным распределением для распределения Бернулли.

### Пример 6.12 (сопряженность бета — Бернулли)

Пусть  $x \in \{0, 1\}$  берется из распределения Бернулли с  $\theta \in [0, 1]$ , то есть  $p(x=1|\theta) = \theta$ . То же самое можно записать как  $p(x|\theta) = \theta^x(1-\theta)^{1-x}$ . Пусть  $\theta$  взята из бета-распределения с параметрами  $\alpha, \beta$ , то есть  $p(\theta|\alpha, \beta) \propto \theta^{\alpha-1}(1-\theta)^{\beta-1}$ .

Перемножая бета-распределение и распределение Бернулли, получим

$$p(\theta|x, \alpha, \beta) = p(x|\theta)p(\theta|\alpha, \beta) \quad (6.105a)$$

$$\propto \theta^x(1-\theta)^{1-x}\theta^{\alpha-1}(1-\theta)^{\beta-1} \quad (6.105b)$$

$$= \theta^{\alpha+x-1}(1-\theta)^{\beta+(1-x)-1} \quad (6.105c)$$

$$\propto p(\theta|\alpha+x, \beta+(1-x)). \quad (6.105d)$$

В последней строчке стоит бета-распределение с параметрами  $(\alpha + x, \beta + (1 - x))$ .

В табл. 6.2 приведены примеры сопряженных априорных вероятностей для параметров некоторых стандартных распределений, используемых в вероятностном моделировании. Такие распределения, как мультиномиальное, обратное гамма-распределение, обратное распределение Уишарта, встречаются в любом учебнике статистики и описаны, например, в Bishop (2006).

**Таблица 6.2.** Примеры сопряженных априорных распределений для распространенных функций правдоподобия

Правдоподобие	Сопряженное априорное распределение	Апостериорное распределение
Бернулли	Бета	Бета
Биномиальное	Бета	Бета
Гауссово	Гауссово / обратное гамма-распределение	Гауссово / обратное гамма-распределение
Гауссово	Гауссово / обратное распределение Уишарта	Гауссово / обратное распределение Уишарта
Мультиномиальное	Дирихле	Дирихле

Бета-распределение является сопряженным априорным распределением для параметра  $\mu$  как для биномиального, так и бернульевского правдоподобия. Когда функция правдоподобия гауссова, можно выбрать гауссово априорное распределение для среднего. Гауссово распределение указано в таблице дважды, для одномерного и многомерного случая. В одномерном случае сопряженным априорным распределением для дисперсии является обратное гамма-распределение. В многомерном случае сопряженным априорным распределением для ковариационной матрицы будет обратное распределение Уишарта. Распределение Дирихле является сопряженным априорным распределением для мультиномиальной функции правдоподобия<sup>1</sup>. Подробнее об этом можно почитать в Bishop (2006).

### 6.6.2. Достаточные статистики

Вспомним, что статистикой случайной величины называется детерминистическая функция от нее. Например, пусть  $\mathbf{x} = [x_1, \dots, x_N]^T$  — вектор одномерных случайных гауссовых величин, то есть  $x_i \sim \mathcal{N}(\mu, \sigma^2)$ . Тогда выборочное среднее  $\hat{\mu} = \frac{1}{N}(x_1 + \dots + x_N)$  будет статистикой. Сэр Рональд Фишер — автор понятия достаточной статистики. Идея состоит в том, что некоторые статистики содержат всю информацию о распределении, которую можно получить из данных. Иными словами, достаточные статистики содержат всю необходимую информацию, чтобы делать выводы о генеральной совокупности; их достаточно для описания распределения.

Пусть дано множество распределений, параметризованное  $\theta$ , и  $X$  — случайная величина с распределением  $p(x | \theta_0)$  для неизвестного  $\theta_0$ . Вектор статистик  $\varphi(x)$  называется *достаточной статистикой* для  $\theta_0$ , если он содержит всю возможную информацию о  $\theta_0$ . Говоря более строго, вероятность  $x$  при условии  $\theta$  можно разложить на множитель, не зависящий от  $\theta$ , и множитель, зависящий от  $\theta$  только через  $\varphi(x)$ . Формально это понятие описывается теоремой факторизации Фишера — Неймана, которую мы приводим под номером 6.14 без доказательства.

**Теорема 6.14 (Фишера — Неймана).** [Теорема 6.5 в Lehmann and Casella (1998)]  
*Пусть плотность вероятности  $X$  задается функцией  $p(x | \theta)$ . Тогда статистика  $\varphi(x)$  будет достаточной для  $\theta$ , если и только если  $p(x | \theta)$  может быть записана в виде*

$$p(x | \theta) = h(x)_{\theta_0}(\varphi(x)), \quad (6.106)$$

---

<sup>1</sup> Априорное гамма-распределение является сопряженным к обратной дисперсии для одномерного гауссова правдоподобия. Априорное распределение Уишарта является сопряженным к обратной для матрицы ковариации многомерного гауссова правдоподобия.

где  $h(x)$  – распределение, не зависящее от  $\theta$ , и вся зависимость от  $\theta$  содержит-  
ся в функции  $g_\theta$  от  $\phi(x)$ .

Если  $p(x|\theta)$  не зависит от  $\theta$ , ясно, что  $\phi(x)$  будет достаточной статистикой при любой функции  $\phi$ . Интересней случай, когда  $p(x|\theta)$  зависит только от  $\phi(x)$ , а не от самого  $x$ . В этом случае  $\phi(x)$  – достаточная статистика для  $\theta$ .

В машинном обучении мы имеем дело с конечными выборками. В случае простых распределений (например, бернульевского из примера 6.8) мы можем оценить параметры распределения по небольшой выборке. Рассмотрим обратную задачу: пусть у нас имеется набор данных (выборка из неизвестного распределения), какое распределение наилучшим образом его описывает? Естественно задаться вопросом, понадобится ли нам большее количество параметров  $\theta$  при увеличении количества данных? В общем случае – да, и этой темой занимается непараметрическая статистика (Wasserman, 2007). Можно, напротив, искать класс распределений, достаточные статистики для которых конечномерны (то есть количество необходимых для их описания параметров не может неогра-ниченно расти). Ответом будет экспоненциальное семейство распределений, которое мы изучим в следующем разделе.

### 6.6.3. Экспоненциальное семейство распределений

Рассматривать распределения дискретной или непрерывной случайной величины мы можем на трех уровнях абстракции. На первом, наименее абстрактном, мы имеем дело с конкретным именным распределением с фиксированными параметрами, например одномерным гауссовым  $\mathcal{N}(0, 1)$  с нулевым средним и единичной дисперсией. В машинном обучении часто применяется второй уровень абстракции, когда мы фиксируем вид распределения (одномерное гауссово) и находим значения параметров по данным. Например, мы предполагаем, что наше распределение – одномерное гауссово  $\mathcal{N}(\mu, \sigma^2)$  с неизвестными средним  $\mu$  и дисперсией  $\sigma^2$ , и методом максимального правдоподобия находим параметры  $(\mu, \sigma^2)$ . Пример мы увидим, рассматривая линейную ре-грессию в главе 9. Третьим уровнем абстракции является рассмотрение се-мейств распределений – в нашей книге мы познакомимся с экспоненциальным семейством. Одномерное гауссово распределение принадлежит этому семейству, как и многие другие широко используемые статистические модели (в том числе все «именные» из табл. 6.2). Все они могут быть описаны вместе (Brown, 1986).

**ПРИМЕЧАНИЕ** Краткая историческая справка. Как и многие другие идеи в математике и естественных науках, экспоненциальные семейства были одно-

время открыты несколькими исследователями. В 1935–1936 годах Эдвин Питман в Тасмании, Жорж Дармуа в Париже и Бернард Купман в Нью-Йорке независимо друг от друга показали, что экспоненциальные семейства распределений — единственные обладающие конечномерными достаточными статистиками при независимой генерации элементов выборки (Lehmann and Casella, 1998). ◆

*Экспоненциальным семейством* называется семейство распределений вероятностей с параметром  $\theta \in \mathbb{R}^D$  вида

$$p(\mathbf{x} | \theta) = h(\mathbf{x}) \exp(\langle \theta, \varphi(\mathbf{x}) \rangle - A(\theta)), \quad (6.107)$$

где  $\varphi(\mathbf{x})$  — векторная достаточная статистика. В общем случае в формуле (6.107) можно использовать любое скалярное произведение. Мы будем использовать стандартное скалярное произведение ( $\langle \theta, \varphi(\mathbf{x}) \rangle = \theta^T \varphi(\mathbf{x})$ ). Заметим, что вид экспоненциального семейства является частным случаем выражения  $g_\theta(\varphi(\mathbf{x}))$  из теоремы Фишера — Неймана (теорема 6.14).

Множитель  $h(\mathbf{x})$  можно внести под знак скалярного произведения, прибавив  $\log h(\mathbf{x})$  к вектору достаточной статистики  $\varphi(\mathbf{x})$  и задав значение соответствующего параметра  $\theta_0 = 1$ . Слагаемое  $A(\theta)$  — это нормировочная константа, обеспечивающая то, что сумма или интеграл от распределения равны 1. Можно получить общее представление об экспоненциальных семействах, игнорируя эти два слагаемых и рассматривая экспоненциальные семейства вида

$$p(\mathbf{x} | \theta) \propto \exp(\theta^T \varphi(\mathbf{x})). \quad (6.108)$$

При такой параметризации  $\theta$  называются *естественными параметрами*. На первый взгляд кажется, что экспоненциальные семейства — это неинтересное преобразование скалярного произведения. Однако то, что вся информация о данных будет содержаться в  $\varphi(\mathbf{x})$ , крайне полезно при построении модели и в вычислениях.

### Пример 6.13 (гауссианы как экспоненциальное семейство)

Рассмотрим одномерное гауссово распределение  $\mathcal{N}(\mu, \sigma^2)$ . Пусть

$$\varphi(\mathbf{x}) = \begin{bmatrix} x \\ x^2 \end{bmatrix}.$$

Тогда по определению экспоненциального семейства

$$p(x | \theta) \propto \exp(\theta_1 x + \theta_2 x^2). \quad (6.109)$$

Взяв

$$\theta = \begin{bmatrix} \mu \\ \theta^2 \end{bmatrix}, \quad -\frac{1}{2\sigma^2}$$
(6.110)

и подставив в (6.109), получим

$$p(x|\theta) \propto \exp\left(\frac{\mu x}{\sigma^2} - \frac{x^2}{2\sigma^2}\right) \propto \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right).$$
(6.111)

Таким образом, одномерное гауссово распределение принадлежит экспоненциальному семейству с достаточной статистикой  $\varphi(x) = \begin{bmatrix} x \\ x^2 \end{bmatrix}$  и естественными параметрами (6.110).

#### Пример 6.14 (распределения Бернулли как экспоненциальное семейство)

Вспомним распределение Бернулли из примера 6.8:

$$p(x|\mu) = \mu^x(1-\mu)^{1-x}, x \in \{0, 1\}.$$
(6.112)

Можно записать его как экспоненциальное семейство

$$p(x|\mu) = \exp [\log(\mu^x(1-\mu)^{1-x})] =$$
(6.113a)

$$= \exp [x \log \mu + (1-x) \log(1-\mu)] =$$
(6.113b)

$$= \exp [x \log \mu - x \log(1-\mu) + \log(1-\mu)] =$$
(6.113c)

$$= \exp \left[ x \log \frac{\mu}{1-\mu} + \log(1-\mu) \right].$$
(6.113d)

В последней строке (6.113d) можно узнать формулу для экспоненциальных семейств (6.107), заметив, что

$$h(x) = 1;$$
(6.114)

$$\theta = \log \frac{\mu}{1-\mu};$$
(6.115)

$$\varphi(x) = x;$$
(6.116)

$$A(\theta) = -\log(1-\mu) = \log(1+\exp(\theta)).$$
(6.117)

Зависимость между  $\theta$  и  $\mu$  обратима, так что

$$\mu = \frac{1}{1 + \exp(-\theta)}. \quad (6.118)$$

Соотношение (6.118) используется для получения второго из равенств в (6.117).

**ПРИМЕЧАНИЕ** Зависимость между параметром  $\mu$  исходного распределения Бернулли и естественным параметром  $\theta$  известна как *сигмоид*, или логистическая функция. Заметим, что  $\mu \in (0, 1)$ , а  $\theta \in \mathbb{R}$ , так что сигмоид «сжимает» вещественную ось в интервал  $(0, 1)$ . Это свойство важно для машинного обучения и используется, например, в логистической регрессии (Bishop, 2006, раздел 4.3.2) и в качестве нелинейной функции активации в нейронных сетях (Goodfellow et al., 2016, глава 6). ♦

Часто неясно, как найти параметрический вид сопряженного к заданному распределению (например, к распределениям из табл. 6.2). Экспоненциальные семейства позволяют искать сопряженные пары распределений. Рассмотрим случайную величину  $X$ , принадлежащую экспоненциальному семейству (6.107):

$$p(\mathbf{x} | \boldsymbol{\theta}) = h(\mathbf{x}) \exp\langle \boldsymbol{\theta}, \varphi(\mathbf{x}) \rangle - A(\boldsymbol{\theta}). \quad (6.119)$$

У любого элемента экспоненциального семейства имеется сопряженное априорное распределение (Brown, 1986)

$$p(\boldsymbol{\theta} | \boldsymbol{\gamma}) = h_c(\boldsymbol{\theta}) \exp\left(\left\langle \begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix}, \begin{bmatrix} \boldsymbol{\theta} \\ -A(\boldsymbol{\theta}) \end{bmatrix} \right\rangle - A_c(\boldsymbol{\gamma})\right), \quad (6.120)$$

где  $\boldsymbol{\gamma} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix}$  имеет размерность  $\dim(\boldsymbol{\theta}) + 1$ . Достаточными статистиками для сопряженного распределения будут  $\begin{bmatrix} \boldsymbol{\theta} \\ -A(\boldsymbol{\theta}) \end{bmatrix}$ . Зная общий вид сопряженных распределений для экспоненциальных семейств, мы можем найти сопряженные для конкретных распределений.

### Пример 6.15

Вспомним запись распределения Бернулли в виде экспоненциального семейства (6.113d):

$$p(x | \mu) = \exp\left[x \log \frac{\mu}{1-\mu} + \log(1-\mu)\right]. \quad (6.121)$$

Канонический вид сопряженного распределения:

$$p(\mu | \alpha, \beta) = \frac{\mu}{1-\mu} \exp \left[ \alpha \log \frac{\mu}{1-\mu} + (\beta + \alpha) \log(1-\mu) - A_c(\gamma) \right], \quad (6.122)$$

где мы определили  $\gamma := [\alpha, \beta + \alpha]^T$  и  $A_c(\mu) := \mu/(1-\mu)$ . Равенство (6.122) тогда упрощается до

$$p(\mu | \alpha, \beta) = \exp[(\alpha - 1) \log \mu + (\beta - 1) \log(1 - \mu) - A_c(\alpha, \beta)]. \quad (6.123)$$

Не в виде экспоненциального семейства это выглядит как

$$p(\mu | \alpha, \beta) \propto \mu^{\alpha-1} (1-\mu)^{\beta-1}, \quad (6.124)$$

и мы узнаем бета-распределение (6.98). В примере 6.12 мы предположили, что бета-распределение является сопряженным априорным распределением к распределению Бернулли, и показали, что это действительно так. В этом примере мы пришли к бета-распределению, записав канонический вид (как экспоненциального семейства) априорного сопряженного распределения к распределению Бернулли.

Как упоминалось в предыдущем разделе, главной мотивацией к использованию экспоненциальных семейств является наличие у них конечномерных достаточных статистик. Кроме того, легко записать сопряженное распределение, также принадлежащее экспоненциальному семейству. С точки зрения статистического вывода, оценка максимального правдоподобия хорошо себя ведет, так как оценки достаточных статистик по выборке являются оптимальными оценками этих статистик на генеральной совокупности (вспомним о среднем и ковариации для гауссова распределения). Так как функция логарифма правдоподобия вогнута, можно применить эффективные методы оптимизации (глава 7).

## 6.7. ЗАМЕНА ПЕРЕМЕННЫХ / ОБРАТНОЕ ПРЕОБРАЗОВАНИЕ

Может показаться, что известных распределений очень много, но на самом деле набор «именных» распределений довольно ограничен. Поэтому часто полезно понимать, каково распределение случайной величины после некоторого преобразования. Например, пусть  $X$  — случайная величина из одномерного нормального распределения  $\mathcal{N}(0, 1)$ . Как распределена  $X^2$ ? Другой пример, часто встречающийся в машинном обучении: пусть  $X_1$  и  $X_2$  — одномерные стандартные нормальные распределения. Как распределена  $(X_1 + X_2)/2$ ?

Одним из способов найти распределение  $(X_1 + X_2)/2$  будет вычислить среднее и дисперсию  $X_1$  и  $X_2$ , а затем работать с ними. Как мы увидели в разделе 6.4.4, мы можем вычислить среднее и дисперсию случайных величин, получающихся при аффинных преобразованиях. Однако мы не всегда можем описать получившееся распределение как функцию. Более того, порой нас интересуют нелинейные преобразования случайных величин, для которых не всегда есть явные формулы.

**ПРИМЕЧАНИЕ** В этом разделе мы говорим о случайных величинах и принимаемых ими значениях. Поэтому напомним, что для обозначения случайных величин мы используем заглавные буквы  $X, Y$ , а для принимаемых ими значений в фазовом пространстве  $\mathcal{T}$  — строчные буквы  $x, y$ . Мы явно записываем распределение дискретной случайной величины  $X$  как  $P(X = x)$ . Для непрерывной случайной величины  $X$  (раздел 6.2.2) мы обозначаем плотность распределения как  $f(x)$ , а функцию распределения — как  $F_x(x)$ . ◆

Рассмотрим два подхода к нахождению распределений случайных величин после преобразований: использующий определение функции распределения и метод замены переменной, использующий цепное правило (раздел 5.2.2). Метод замены переменной широко применяется, так как дает готовый «рецепт», как получить распределение после преобразования. Мы объясним оба подхода на примере одномерных случайных величин, а для многомерных сформулируем только результаты<sup>1</sup>.

Преобразования дискретных случайных величин можно описать напрямую. Пусть  $X$  — дискретная случайная величина с распределением  $P(X = x)$  (раздел 6.2.1), а  $U(x)$  — обратимая функция. Рассмотрим преобразованную случайную величину  $Y := U(X)$  с распределением  $P(Y = y)$ . Тогда

$$P(Y = y) = P(U(X) = y) \quad \text{интересующее нас преобразование} \quad (6.125a)$$

$$= P(X = U^{-1}(y)) \quad \text{обратное к нему}, \quad (6.125b)$$

и можно заметить, что  $x = U^{-1}(y)$ . Следовательно, для дискретных случайных величин преобразование работает на отдельных значениях (соответственно меняются и вероятности).

### 6.7.1. Метод функций распределения

Метод функций распределения отсылает нас к базовым понятиям и использует определение функции распределения как  $F_x(x) = P(X \leq x)$  и то, что ее производ-

---

<sup>1</sup> Производящие функции моментов также могут использоваться при изучении преобразований случайных величин (Casella and Berger, 2002, глава 2).

ная является плотностью распределения  $f(x)$  (Wasserman, 2004, глава 2). Для случайной величины  $X$  и функции  $U$  найдем плотность распределения случайной величины  $Y := U(X)$ .

1. Найдем функцию распределения:

$$F_Y(y) = P(Y \leq y). \quad (6.126)$$

2. Продифференцируем  $F_Y(y)$  и получим плотность распределения  $f(y)$ :

$$f(y) = \frac{d}{dy} F_Y(y). \quad (6.127)$$

Мы должны также помнить, что при преобразовании может измениться область значений случайной величины.

### Пример 6.16

Пусть  $X$  — непрерывная случайная величина с плотностью вероятности  $0 \leq x \leq 1$ ,

$$f(x) = 3x^2 \quad (6.128)$$

на интервале  $0 \leq x \leq 1$ . Найдем плотность вероятности  $Y = X^2$ .

$f$  — возрастающая функция от  $x$ , так что значения  $y$  лежат в интервале  $[0, 1]$ . Получаем

$$F_Y(y) = P(Y \leq y) \text{ определение функции распределения} \quad (6.129a)$$

$$= P(X^2 \leq y) \text{ преобразование} \quad (6.129b)$$

$$= P\left(X \leq y^{\frac{1}{2}}\right) \text{ обратное преобразование} \quad (6.129c)$$

$$= F_X\left(y^{\frac{1}{2}}\right) \text{ определение функции распределения} \quad (6.129d)$$

$$= \int_0^{y^{\frac{1}{2}}} 3t^2 dt \text{ функция распределения как определенный интеграл} \quad (6.129e)$$

$$= [t^3]_{t=0}^{t=y^{\frac{1}{2}}} \text{ результат интегрирования} \quad (6.129f)$$

$$= y^{\frac{3}{2}}, 0 \leq y \leq 1. \quad (6.129g)$$

Таким образом, функция распределения  $Y$

$$F_Y(y) = y^{\frac{3}{2}} \quad (6.130)$$

на интервале  $0 \leq y \leq 1$ . Чтобы получить плотность распределения, мы дифференцируем функцию распределения:

$$f(y) = \frac{d}{dy} F_Y(y) = \frac{3}{2} y^{\frac{1}{2}} \quad (6.131)$$

на интервале  $0 \leq y \leq 1$ .

В примере 6.16 мы рассматривали строго монотонную возрастающую функцию  $f(x) = 3x^2$ . Поэтому мы могли найти обратную функцию. В общем случае мы требуем, чтобы интересующая нас функция  $y = U(x)$  имела обратную <sup>1</sup>  $x = U^{-1}(y)$ . Можно рассмотреть функцию распределения  $F_X(x)$  случайной величины  $X$  и взять ее в качестве  $U(x)$ . Это приведет нас к следующей теореме.

**Теорема 6.15** [Теорема 2.1.10 в Casella and Berger (2002)]. *Пусть  $X$  – непрерывная случайная величина со строго монотонной функцией распределения  $F_X(x)$ . Тогда случайная величина  $Y$ , определенная как*

$$Y := F_X(x), \quad (6.132)$$

*имеет равномерное распределение.*

Теорема 6.15 известна как *интегральное преобразование вероятностей* и используется для создания алгоритмов семплинга (преобразуя результат семплинга из равномерного распределения) (Bishop, 2006). Алгоритм сначала порождает элемент равномерного распределения, а потом преобразует его с помощью обратной к функции распределения (если она существует), чтобы получить элемент из желаемого распределения. Интегральное преобразование вероятности также используется для проверки гипотез о происхождении выборки из данного распределения (Lehmann and Romano, 2005). Идея о принадлежности результата применения функции распределения равномерному распределению также лежит в основе теории коцул (Nelsen, 2006).

## 6.7.2. Замена переменных

Метод функций распределения, описанный в разделе 6.7.1, основан на базовых идеях: определении функции распределения, свойствах обратных функций, дифференцирования и интегрирования. Он использует два соображения:

1. Мы можем преобразовать функцию распределения  $Y$  в функцию распределения  $X$ .

---

<sup>1</sup> Функции, имеющие обратные, называются биективными (раздел 2.7).

2. Мы можем продифференцировать функцию распределения и получить плотность распределения.

Проведем доказательство пошагово. Это поможет нам понять более общий метод замены переменных, вводимый в теореме 6.16.

**ПРИМЕЧАНИЕ** Название происходит от метода замены переменной при взятии интегралов<sup>1</sup>. Для функций одной переменной она описывается формулой

$$\int f(g(x))g'(x)dx = \int f(u)du, \text{ где } u = g(x). \quad (6.133)$$

Вывод этой формулы основан на правиле дифференцирования композиции (5.32) и применении (дважды) основной теоремы анализа. Основная теорема анализа формализует тот факт, что дифференцирование и интегрирование «обратны» друг другу. Неформально ее можно объяснить, рассмотрев маленькие изменения (дифференциалы) уравнения  $u = g(x)$ , то есть  $\Delta u = g'(x)\Delta x$ . При подстановке  $u = g(x)$  выражение под интегралом в правой части (6.133) превращается в  $f(g(x))$ . Поверив, что  $du$  можно приблизить как  $du \approx \Delta u = g'(x)\Delta x$ , и что  $dx \approx \Delta x$ , получим (6.133). ◆

Рассмотрим одномерную случайную величину  $X$  и обратимую функцию  $U$ , дающую нам новую случайную величину  $Y = U(X)$ . Пусть случайная величина  $X$  принимает значения  $x \in [a, b]$ . По определению функции распределения

$$F_Y(y) = P(Y \leq y). \quad (6.134)$$

Нас интересует функция  $U$  от случайной величины

$$P(Y \leq y) = P(U(X) \leq y), \quad (6.135)$$

где мы предполагаем, что  $U$  обратима. Обратимая функция на интервале должна либо строго возрастать, либо строго убывать. Если  $U$  строго возрастает, обратная к ней  $U^{-1}$  также строго возрастает. Применяя обратную функцию  $U^{-1}$  к аргументу в  $P(U(X) \leq y)$ , получим

$$P(U(X) \leq y) = P(U^{-1}(U(X)) \leq U^{-1}(y)) = P(X \leq U^{-1}(y)). \quad (6.136)$$

Справа в (6.136) стоит функция распределения  $X$ . Вспомним ее определение через плотность распределения

$$P(X \leq U^{-1}(y)) = \int_a^{U^{-1}(y)} f(x)dx. \quad (6.137)$$

---

<sup>1</sup> Метод замены переменной в теории вероятностей основан на замене переменных из математического анализа (Tandra, 2014).

Теперь мы выразили функцию распределения  $Y$  через  $x$ :

$$F_Y(y) = \int_a^{U^{-1}(y)} f(x) dx. \quad (6.138)$$

Чтобы найти плотность распределения, продифференцируем (6.138) по  $y$ :

$$f(y) = \frac{d}{dy} F_y(y) = \frac{d}{dy} \int_a^{U^{-1}(y)} f(x) dx. \quad (6.139)$$

Заметим, что в правой части мы интегрируем по  $x$ , а нужен интеграл по  $y$  (поскольку дифференцировать будем по  $y$ ). Используя (6.133), получаем замену

$$\int f(U^{-1}(y)) U'^{-1}(y) dy = \int f(x) dx, \text{ где } x = U^{-1}(y). \quad (6.140)$$

Применяя формулу (6.140) к правой части (6.139), приходим к

$$f(y) = \frac{d}{dy} \int_a^{U^{-1}(y)} f_x(U^{-1}(y)) U'^{-1}(y) dy. \quad (6.141)$$

Теперь вспомним, что дифференцирование — линейный оператор. Чтобы не забыть, что  $f_x(U^{-1}(y))$  — функция от  $x$ , а не от  $y$ , используем индекс. Применим основную теорему анализа и получим

$$f(y) = f_x(U^{-1}(y)) \cdot \left( \frac{d}{dy} U^{-1}(y) \right). \quad (6.142)$$

Вспомним наше предположение, что  $U$  строго возрастает. Для убывающих функций при тех же рассуждениях мы получим знак «минус». Поэтому, чтобы выражение имело одинаковый вид как для возрастающих, так и для убывающих  $U$ , будем использовать абсолютное значение дифференциала:

$$f(y) = f_x(U^{-1}(y)) \cdot \left| \frac{d}{dy} U^{-1}(y) \right|. \quad (6.143)$$

Такой способ действий называется *методом замены переменной*. Множитель  $\left| \frac{d}{dy} U^{-1}(y) \right|$  показывает, как меняется единичный объем при применении  $U$  (см. определение якобиана в разделе 5.3).

**ПРИМЕЧАНИЕ** По сравнению с дискретным случаем в (6.125b), у нас появляется дополнительный множитель  $\left| \frac{d}{dy} U^{-1}(y) \right|$ . Непрерывный случай требует большей аккуратности, поскольку  $P(Y=y) = 0$  для всех  $y$ , и плотность вероятности  $f(y)$  не описывается как вероятность некоторого касающегося  $y$  события.



До сих пор в этом разделе мы изучали замену переменных в одномерном случае. Случай многомерной случайной величины аналогичен, но сложнее, из-за того что для многомерных функций вместо абсолютного значения производной надо использовать якобиан. Вспомним формулу (5.58): матрица Якоби — это матрица частных производных, и отличие ее определителя от нуля показывает, что она обратима. В разделе 4.1 обсуждалось, что детерминант в формуле появляется из-за того, что дифференциалы (считаем их кубами) матрица Якоби преобразует в параллелепипеды. Подведем итог всему сказанному выше в следующей теореме, дающей нам алгоритм для многомерной замены переменных.

**Теорема 6.16.** [Теорема 17.2 в Billingsley (1995)] Пусть  $f(\mathbf{x})$  — плотность распределения многомерной непрерывной случайной величины  $X$ . Если векторнозначная функция  $\mathbf{y} = U(\mathbf{x})$  дифференцируема и обратима для всех  $\mathbf{x}$  из области определения, то для соответствующих значений  $\mathbf{y}$  плотность вероятности случайной величины  $Y = U(X)$  задается формулой

$$f(\mathbf{y}) = f_{\mathbf{x}}(U^{-1}(\mathbf{y})) \cdot \left| \det \left( \frac{\partial}{\partial \mathbf{y}} U^{-1}(\mathbf{y}) \right) \right|. \quad (6.144)$$

На первый взгляд эта теорема выглядит пугающе, однако ее основная идея состоит в том, что замена переменных для многомерных случайных величин аналогична этой процедуре для одномерных. Сначала мы находим обратное преобразование, подставляем его в функцию плотности, потом вычисляем якобиан и перемножаем результаты. В примере ниже рассмотрен случай двумерной случайной величины.

### Пример 6.17

Рассмотрим двумерную случайную величину  $X$  со значениями  $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$  и плотностью распределения

$$f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \frac{1}{2\pi} \exp\left(-\frac{1}{2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right). \quad (6.145)$$

Используем метод замены переменной из теоремы 6.16, чтобы выяснить эффект от применения к случайной величине линейного преобразования (раздел 2.7). Возьмем матрицу  $\mathbf{A} \in \mathbb{R}^{2 \times 2}$ :

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}. \quad (6.146)$$

Мы хотим найти плотность вероятности для преобразованной случайной величины  $Y$ , принимающей значения  $\mathbf{y} = \mathbf{Ax}$ . При замене переменных нам требуется обратное преобразование, задающее  $\mathbf{x}$  как функцию от  $\mathbf{y}$ . Так

как мы имеем дело с линейными преобразованиями, обратное преобразование задается обратной матрицей (раздел 2.2.2). Для матриц размера  $2 \times 2$  можно явно записать обратную:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = A^{-1} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}. \quad (6.147)$$

Заметим, что  $ad - bc$  — это детерминант (раздел 4.1) матрицы  $A$ . Соответствующая плотность распределения задается формулой

$$f(\mathbf{x}) = f(A^{-1}\mathbf{y}) = \frac{1}{2\pi} \exp\left(-\frac{1}{2} \mathbf{y}^T A^{-T} A^{-1} \mathbf{y}\right). \quad (6.148)$$

Частная производная от произведения матрицы на вектор относительно вектора — это сама матрица (раздел 5.5), и, следовательно,

$$\frac{\partial}{\partial \mathbf{y}} A^{-1} \mathbf{y} = A^{-1}. \quad (6.149)$$

Вспомним (раздел 4.1), что детерминант обратной матрицы обратен к детерминанту исходной, так что якобиан равен

$$\det\left(\frac{\partial}{\partial \mathbf{y}} A^{-1} \mathbf{y}\right) = \frac{1}{ad - bc}. \quad (6.150)$$

Теперь мы можем применить формулу замены переменных из теоремы 6.16, перемножив (6.148) и (6.150) и получив

$$f(\mathbf{y}) = f(\mathbf{x}) \left| \det\left(\frac{\partial}{\partial \mathbf{y}} A^{-1} \mathbf{y}\right) \right| = \quad (6.151a)$$

$$= \frac{1}{2\pi} \exp\left(-\frac{1}{2} \mathbf{y}^T A^{-T} A^{-1} \mathbf{y}\right) |ad - bc|^{-1}. \quad (6.151b)$$

Хотя в примере 6.17 используется двумерная случайная величина, что позволяет нам легко вычислить обратную матрицу, соотношение выше выполняется и для больших размерностей.

**ПРИМЕЧАНИЕ** В разделе 6.5 мы видели, что плотность  $f(\mathbf{x})$  в (6.148) — стандартная гауссова, а  $f(\mathbf{y})$  после преобразования — гауссова плотность с ковариационной матрицей  $\Sigma = AA^T$ . ◆

Идеи этой главы будут использоваться в разделе 8.4 для описания вероятностного моделирования и в разделе 8.5 при введении графического языка. Их применение напрямую в машинном обучении будет рассмотрено в главах 9 и 11.

## 6.8. ДЛЯ ДАЛЬНЕЙШЕГО ЧТЕНИЯ

Изложение в этой главе довольно сжато. Более постепенное введение в тему, подходящее для самообразования, есть в книгах Grinstead and Snell (1997) и Walpole et al. (2011). Читатели, заинтересованные в более философских аспектах теории вероятностей, могут обратиться к Hacking (2001), тогда как подход Downey (2014) больше ориентирован на применение в программировании. Обзор экспоненциальных семейств можно найти в Barndorff-Nielsen (2014). В главе 8 мы больше узнаем о том, как использовать распределения вероятностей в задачах машинного обучения. Недавний вслеск интереса к нейронным сетям привлек внимание и к вероятностным моделям. Например, идея нормализации потоков (Jimenez Rezende and Mohamed, 2015) основана на замене переменных при преобразовании случайных величин. Обзор методов вариационного вывода в применении к нейронным сетям дан в главах 16–20 книги Goodfellow et al. (2016).

Мы обошли большую часть трудностей, касающихся непрерывных случайных величин, так как не касались вопросов теории меры (Billingsley, 1995; Pollard, 2002) и приняли без объяснений существование вещественных чисел, способы задания их подмножеств и определения вероятностей на этих подмножествах. Между тем эти подробности важны, например при определении условной вероятности  $p(y | x)$  для непрерывных случайных величин  $x, y$  (Proschan and Presnell, 1998). Не слишком подробная запись скрывает тот факт, что мы рассматриваем ситуацию  $X = x$  (нулевое множество). Далее, мы рассматриваем плотность вероятности для  $y$ . Более строгой записью было бы  $\mathbb{E}_y[f(y) | \sigma(x)]$ , где мы рассматриваем матожидание относительно  $y$  тестовой функции  $f$  при условии  $\sigma$ -алгебры, заданной  $x$ . Читатели с более высоким уровнем подготовки, интересующиеся теорией вероятностей, имеют широкий выбор: Jaynes, 2003; MacKay, 2003; Jacod and Protter, 2004; Grimmett and Welsh, 2014, в том числе среди очень продвинутых текстов — Shirayev, 1984; Lehmann and Casella, 1998; Dudley, 2002; Bickel and Doksum, 2006; Çinlar, 2011. Альтернативный подход к теории вероятностей — начать с понятия математического ожидания и двигаться «назад» к свойствам вероятностного пространства (Whittle, 2000). Так как машинное обучение позволяет нам моделировать более нетривиальные распределения на более сложных типах данных, разработчик вероятностных моделей обучения должен разбираться в этих тонкостях. К текстам о машинном обучении, фокусирующимся на вероятностном моделировании, относятся MacKay (2003); Bishop (2006); Rasmussen and Williams (2006); Barber (2012); Murphy (2012).

## УПРАЖНЕНИЯ

**6.1.** Рассмотрим следующее двумерное распределение  $p(x, y)$  двух дискретных случайных величин  $X$  и  $Y$ .

	$y_1$	0,01	0,02	0,03	0,1	0,1
$Y$	$y_1$	0,05	0,1	0,05	0,07	0,2
	$y_1$	0,1	0,05	0,03	0,05	0,04

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$X$				

Вычислите:

- a. Маргинальные распределения  $p(x)$  и  $p(y)$ .
- b. Условные распределения  $p(x \mid Y=y_1)$  и  $p(y \mid X=x_3)$ .

**6.2.** Рассмотрим смесь двух гауссовых распределений (показанную на рис. 6.4):

$$0,4\mathcal{N}\left(\begin{bmatrix} 10 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) + 0,6\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 8,4 & 2,0 \\ 2,0 & 1,7 \end{bmatrix}\right).$$

- a. Вычислите предельные распределения для каждого измерения.
- b. Вычислите среднее значение, моду и медианное значение для каждого предельного распределения.
- c. Вычислите среднее значение и моду для двумерного распределения.

**6.3.** Вы написали компьютерную программу, которая иногда компилируется, а иногда нет (код не меняется). Вы решаете смоделировать кажущуюся стохастичность (успех или отсутствие успеха)  $x$  компилятора, используя распределение Бернулли с параметром  $\mu$ :

$$p(x \mid \mu) = \mu^x(1 - \mu)^{1-x}, x \in \{0, 1\}.$$

Выберите сопряженное априорное распределение для вероятности Бернулли и вычислите апостериорное распределение  $p(\mu \mid x_1, \dots, x_N)$ .

**6.4.** Пусть есть две сумки. В первой сумке четыре манго и два яблока; во второй сумке четыре манго и четыре яблока.

Также есть несимметричная монета, которая показывает «орел» с вероятностью 0,6 и «решку» с вероятностью 0,4. Если на монете изображен «орел», наугад выбираем фрукт из мешка 1; в противном случае наугад выбираем фрукт из мешка 2.

Ваш друг подбрасывает монету (вы не видите результат), наугад выбирает фрукт из соответствующей сумки и преподносит вам манго. Какова вероятность того, что манго было взято из мешка 2?

*Подсказка:* используйте теорему Байеса.

### 6.5. Рассмотрим модель временных рядов

$$\begin{aligned}\mathbf{x}_{t+1} &= \mathbf{A}\mathbf{x}_t + \mathbf{w}, \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}); \\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t + \mathbf{v}, \quad \mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}),\end{aligned}$$

где  $\mathbf{w}, \mathbf{v}$  — независимые одинаково распределенные переменные гауссова шума. Далее, предположим, что  $p(\mathbf{x}_0) = \mathcal{N}(\mu_0, \Sigma_0)$ .

- a. Каков вид  $p(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T)$ ? Обоснуйте свой ответ (явно вычислять совместное распределение не нужно).
- b. Предположим, что  $p(\mathbf{x}_t | \mathbf{y}_1, \dots, \mathbf{y}_t) = \mathcal{N}(\mu_t, \Sigma_t)$ .
  - 1. Вычислите  $p(\mathbf{x}_{t+1} | \mathbf{y}_1, \dots, \mathbf{y}_t)$ .
  - 2. Вычислите  $p(\mathbf{x}_{t+1}, \mathbf{y}_{t+1} | \mathbf{y}_1, \dots, \mathbf{y}_t)$ .
  - 3. В момент времени  $t + 1$  наблюдается значение  $\mathbf{y}_{t+1} = \hat{\mathbf{y}}$ . Вычислите условное распределение  $p(\mathbf{x}_{t+1} | \mathbf{y}_1, \dots, \mathbf{y}_{t+1})$ .

**6.6.** Докажите связь в (6.44), которая связывает стандартное определение дисперсии с выражением необработанной оценки для дисперсии.

**6.7.** Докажите связь в (6.45), которая связывает попарную разницу между примерами в наборе данных с выражением необработанной оценки для дисперсии.

**6.8.** Выразите распределение Бернулли в форме естественного параметра экспоненциального семейства, см. (6.107).

**6.9.** Выразите биномиальное распределение в виде экспоненциального семейства распределения. Выразите также бета-распределение в виде экспоненциального семейства распределения. Покажите, что продукты бета-распределения и биномиального распределения также принадлежат к экспоненциальному семейству.

**6.10.** Выведите связь из раздела 6.5.2 двумя способами:

- a. Завершив квадрат.
- b. Выражая гауссиан в форме семейства экспоненциальных распределений.

Произведение двух гауссианов  $\mathcal{N}(\mathbf{x} | \mathbf{a}, \mathbf{A}) \mathcal{N}(\mathbf{x} | \mathbf{b}, \mathbf{B})$  является ненормализованным гауссовым распределением  $c \mathcal{N}(\mathbf{x} | \mathbf{c}, \mathbf{C})$  с

$$\begin{aligned}\mathbf{C} &= (\mathbf{A}^{-1} + \mathbf{B}^{-1})^{-1}; \\ \mathbf{c} &= \mathbf{C}(\mathbf{A}^{-1}\mathbf{a} + \mathbf{B}^{-1}\mathbf{b}); \\ c &= (2\pi)^{-\frac{D}{2}} |\mathbf{A} + \mathbf{B}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{a} - \mathbf{b})^\top (\mathbf{A} + \mathbf{B})^{-1}(\mathbf{a} - \mathbf{b})\right).\end{aligned}$$

Обратите внимание, что нормализующая константа  $c$  сама по себе может считаться (нормализованным) распределением Гаусса либо в  $\mathbf{a}$ , либо в  $\mathbf{b}$  с «занышенной» ковариационной матрицей  $\mathbf{A} + \mathbf{B}$ , то есть  $\mathcal{N}(\mathbf{a} | \mathbf{b}, \mathbf{A} + \mathbf{B}) = \mathcal{N}(\mathbf{b} | \mathbf{a}, \mathbf{A} + \mathbf{B})$ .

### 6.11. Повторяющиеся ожидания

Рассмотрим две случайные величины  $x, y$  с совместным распределением  $p(x, y)$ . Покажите, что

$$\mathbb{E}_x[x] = \mathbb{E}_y[\mathbb{E}_x[x | y]].$$

Здесь  $\mathbb{E}_x[x | y]$  обозначает математическое ожидание  $x$  при условном распределении  $p(x | y)$ .

### 6.12. Манипулирование гауссовыми случайными величинами

Рассмотрим гауссову случайную величину  $\mathbf{x} \sim \mathcal{N}(\mathbf{x} | \mu_x, \Sigma_x)$ , где  $\mathbf{x} \in \mathbb{R}^D$ . Кроме того, есть

$$\mathbf{y} = \mathbf{Ax} + \mathbf{b} + \mathbf{w},$$

где  $\mathbf{y} \in \mathbb{R}^E$ ,  $\mathbf{A} \in \mathbb{R}^{E \times D}$ ,  $\mathbf{b} \in \mathbb{R}^E$  и  $\mathbf{w} \sim \mathcal{N}(\mathbf{w} | \mathbf{0}, \mathbf{Q})$  является независимым гауссовым шумом. «Независимый» означает, что  $\mathbf{x}$  и  $\mathbf{w}$  — независимые случайные величины, а  $\mathbf{Q}$  диагональна.

- a. Запишите вероятность  $p(\mathbf{y} | \mathbf{x})$ .
- b. Распределение  $p(\mathbf{y}) = \int p(\mathbf{y} | \mathbf{x})p(\mathbf{x})d\mathbf{x}$  гауссово. Вычислите среднее  $\mu_y$  и ковариацию  $\Sigma_y$ . Получите подробный результат.
- c. Случайная величина  $\mathbf{y}$  преобразуется в соответствии с отображением измерений

$$\mathbf{z} = \mathbf{Cy} + \mathbf{v},$$

где  $\mathbf{z} \in \mathbb{R}^F$ ,  $\mathbf{C} \in \mathbb{R}^{F \times E}$ , и  $\mathbf{v} \sim \mathcal{N}(\mathbf{v} | \mathbf{0}, \mathbf{R})$  является независимым гауссовым (измерительным) шумом.

- Запишите  $p(\mathbf{z} | \mathbf{y})$ .
- Вычислите  $p(\mathbf{z})$ , то есть среднее значение  $\mu_z$  и ковариацию  $\Sigma_z$ . Получите подробный результат.

- d. Теперь измеряется значение  $\hat{y}$ . Вычислите апостериорное распределение  $p(\mathbf{x} | \hat{\mathbf{y}})$ .

*Подсказка:* эта апостериорная функция также является гауссовой, то есть нужно определить только ее среднее значение и матрицу ковариации. Начните с явного вычисления совместного гауссова  $p(\mathbf{x}, \mathbf{y})$ . Также требуются вычисления кросс-ковариаций  $\text{Cov}_{x,y}[\mathbf{x}, \mathbf{y}]$  и  $\text{Cov}_{y,x}[\mathbf{y}, \mathbf{x}]$ . Затем примените правила гауссовой обусловленности.

### 6.13. Интегральное преобразование вероятности

Учитывая непрерывную случайную величину  $x$ , с помощью кумулятивной функции распределения  $F_x(x)$  покажите, что случайная величина  $y = F_x(x)$  равномерно распределена.

# 7

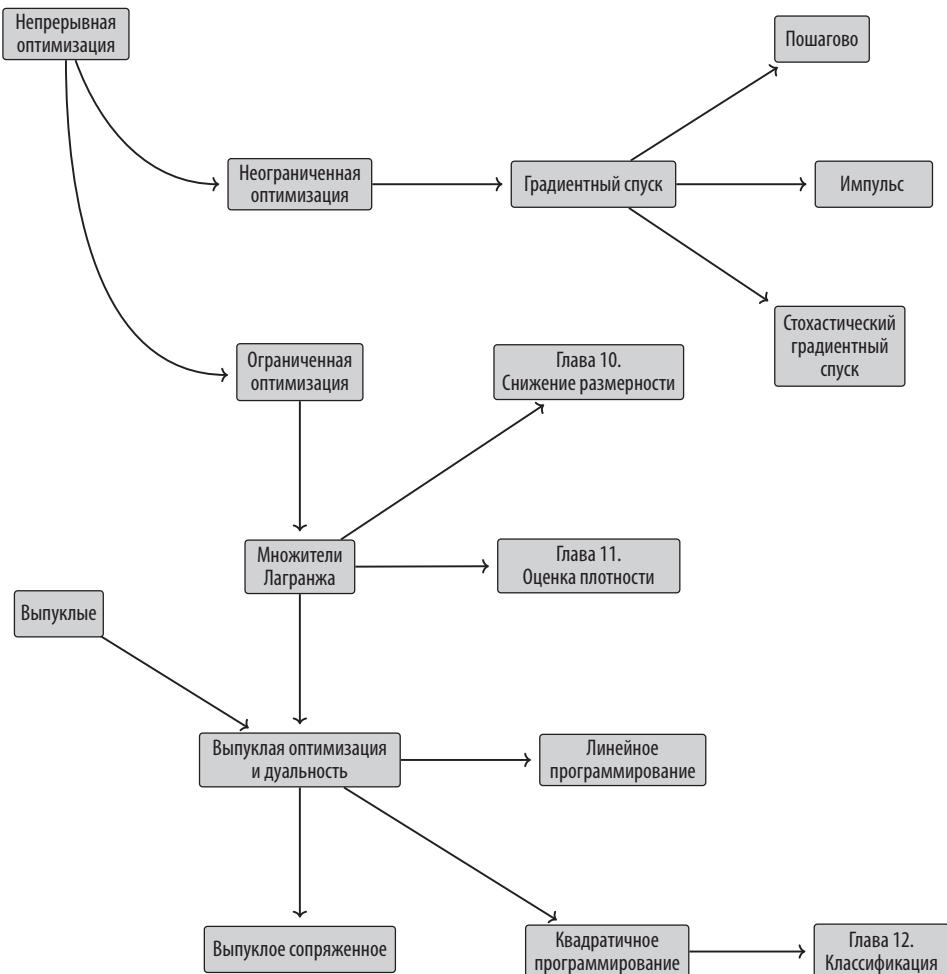
## Непрерывная оптимизация

Поскольку алгоритмы машинного обучения реализуются на компьютере, их математические основы выражаются как численные оптимизационные методы. В этой главе описаны базовые численные методы для обучения моделей МО. Зачастую обучение модели МО сводится к нахождению хорошего набора параметров. Что такое «хорошо» — зависит от целевой функции или вероятностной модели, примеры которых мы увидим во второй части этой книги. При наличии целевой функции находим наилучшее значение при помощи алгоритмов оптимизации.

В этой главе рассматриваются две основные ветви непрерывной оптимизации (рис. 7.1): неограниченная и ограниченная<sup>1</sup>. Далее мы предположим, что наша целевая функция является дифференцируемой (глава 5), поскольку нам доступен градиент в каждой точке пространства, что помогает нам найти оптимальное значение. По существующему соглашению, большинство целевых функций в машинном обучении требуется минимизировать, то есть наилучшим значением считается минимальное. Интуитивно нахождение наилучшего значения подобно нахождению минимумов функции, тогда как градиент соответствует движению вверх. Идея в том, чтобы двигаться вниз (по направлению, противоположному градиенту) и надеяться, что удастся найти самую глубокую точку. Для неограниченной оптимизации только эта концепция нам и требуется, но при проектировании есть несколько вариантов выбора, которые мы обсудим в разделе 7.1. Для ограниченной оптимизации потребуется ввести другие концепции, чтобы справиться с ограничениями (раздел 7.2). В этой главе мы познакомимся с особым классом задач (задачи выпуклой оптимизации в разделе 7.3), в рамках которых сможем высказывать суждения о достижении глобального оптимума.

---

<sup>1</sup> Поскольку мы рассматриваем данные и модели в  $\mathbb{R}^D$ , здесь мы имеем дело именно с задачами *непрерывной* оптимизации, которая противопоставляется *комбинаторной* оптимизации; задачи последней предназначены для работы с дискретными переменными.



**Рис. 7.1.** Ассоциативная карта концепций, связанных с оптимизацией, так, как они представлены в этой главе. Две основных идеи из этой карты — градиентный спуск и выпуклая оптимизация

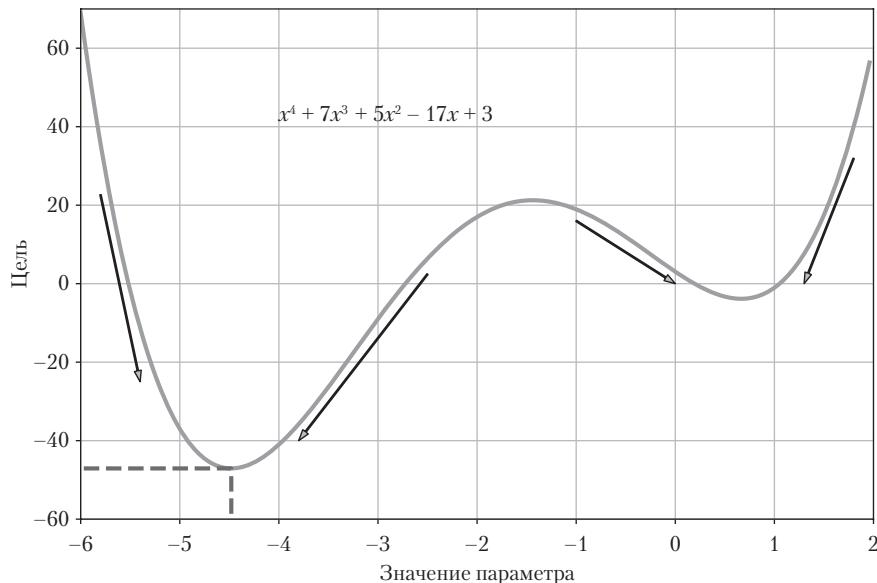
Рассмотрим функцию на рис. 7.2. У этой функции есть *глобальный минимум* в точке около  $x = -4,5$ , значение целевой функции для которого равно около  $-47$ . Поскольку эта функция гладкая, градиенты могут пригодиться при нахождении минимума и указывать, куда следует сделать шаг в его поисках — влево или вправо. При этом предполагается, что мы в правильной впадине функции, поскольку там есть еще один *локальный минимум* в районе  $x = 0,7$ .

Как вы помните, можно найти решения функции для всех ее стационарных точек<sup>1</sup>, рассчитав ее производную и установив ее равной 0. Для

$$\ell(x) = x^4 + 7x^3 + 5x^2 - 17x + 3 \quad (7.1)$$

имеем соответствующий градиент вида

$$\frac{d\ell(x)}{dx} = 4x^3 + 21x^2 + 10x - 17. \quad (7.2)$$



**Рис. 7.2.** Пример целевой функции. Градиенты обозначены стрелками, а глобальный минимум ограничен прерывистой линией

Поскольку это кубическое уравнение, у него, как правило, три решения при равенстве нулю. В данном примере два из них — это минимумы, а одно — максимум (около  $x = -1,4$ ). Чтобы проверить, является ли стационарная точка минимумом или максимумом, необходимо повторно взять производную и проверить, какова вторая производная в стационарной точке — положительная или отрицательная. В нашем случае вторая производная равна

$$\frac{d^2\ell(x)}{dx^2} = 12x^2 + 42x + 10. \quad (7.3)$$

<sup>1</sup> Стационарные точки — это вещественные корни производной, то есть точки с нулевым градиентом.

Подставив оцененные на глазок значения  $x = -4,5, -1,4, 0,7$ , мы, как и ожидалось, увидим, что средняя точка соответствует максимуму  $\left(\frac{d^2\ell(x)}{dx^2} < 0\right)$ , а две другие стационарные точки — минимумам.

Обратите внимание: в дискуссии выше мы избегали аналитического подхода к нахождению значений  $x$ , хотя такой подход можно применить с многочленами низшего порядка, такими как предыдущий. Как правило, мы не в состоянии найти аналитическое решение и поэтому должны начинать с какого-то значения, скажем,  $x_0 = -10$ , и следовать градиенту<sup>1</sup>. Градиент указывает, что мы должны двигаться вправо, но не конкретизирует, как далеко. Более того, если бы мы начали с правого края (например,  $x_0 = 0$ ), то градиент привел бы нас не к тому минимуму. На рис. 7.2 проиллюстрировано, что для  $x > -1$  градиент направлен к минимуму (в правой части рисунка), у которого большее целевое значение.

В разделе 7.3 мы познакомимся с классом выпуклых функций, для которых не характерна такая хитрая зависимость от начальной точки алгоритма. Для выпуклых функций все локальные минимумы равны глобальному минимуму. Оказывается, что многие целевые функции в области машинного обучения проектируются так, чтобы они получались выпуклыми, и пример этого будет показан в главе 12.

До сих пор речь в этой главе шла об одномерной функции, при работе с которой мы могли наглядно представить идеи градиентов, направлений спуска и оптимальных значений. В оставшейся части главы мы будем развивать те же идеи в более высоких измерениях. К сожалению, визуализировать концепции мы можем только в одномерном виде, но некоторые концепции не обобщаются непосредственно до высших измерений, поэтому читать этот материал нужно внимательно.

## 7.1. ОПТИМИЗАЦИЯ С ИСПОЛЬЗОВАНИЕМ ГРАДИЕНТНОГО СПУСКА

Рассмотрим, как найти решение для минимума вещественнозначной функции

$$\min_x f(\mathbf{x}), \quad (7.4)$$

где  $f: \mathbb{R}^d \rightarrow \mathbb{R}$  — это целевая функция, в которой заключена стоящая перед нами задача машинного обучения. Мы предполагаем, что функция  $f$  дифференцируема, и мы неспособны аналитически найти решение в замкнутой форме.

---

<sup>1</sup> Согласно теореме Абеля — Руффини, как правило, не существует алгебраического решения для многочленов степени 5 или выше (Abel, 1826).

Градиентный спуск — это оптимизационный алгоритм первого порядка. Чтобы найти локальный минимум функции при помощи градиентного спуска, нужно делать шаги, пропорциональные отрицательному значению, равному по модулю градиенту функции в данной точке. Как вы помните из раздела 5.1, градиент указывает направление самого крутого подъема<sup>1</sup>. Еще одно интуитивное понятие, которое полезно здесь учесть, это набор линий, вдоль которых функция сохраняет определенное значение ( $f(\mathbf{x}) = c$  для некоторого значения  $c \in \mathbb{R}$ ); эти линии также называются контурными. Мы хотим оптимизировать точки градиента в направлении, ортогональном контурным линиям.

Рассмотрим функции со многими переменными. Представим поверхность (описываемую функцией  $f(\mathbf{x})$ ), когда «шар» стартует в конкретной точке  $\mathbf{x}_0$ . Когда шар трогается с места, он начинает катиться вниз в направлении самого крутого спуска. При градиентном спуске используется тот факт, что  $f(\mathbf{x}_0)$  снижается быстрее всего при переходе от  $\mathbf{x}_0$  в направлении отрицательного градиента  $-((\nabla f)(\mathbf{x}_0))^T$  от  $f$  в  $\mathbf{x}_0$ . В этой книге предполагается, что функции дифференцируемы, и отсылаем читателя к более общим характеристикам, изложенным в разделе 7.4. Тогда, если

$$\mathbf{x}_1 = \mathbf{x}_0 - \gamma ((\nabla f)(\mathbf{x}_0))^T \quad (7.5)$$

для небольшого пошагового  $\gamma \geq 0$ , то для  $f(\mathbf{x}_1) \leq f(\mathbf{x}_0)$ . Обратите внимание, что здесь мы используем транспонирование для градиента, поскольку в противном случае измерения не складываются.

Данное наблюдение позволяет нам определить простой алгоритм для градиентного спуска: если мы хотим найти локальный оптимум  $f(\mathbf{x}_*)$  от функции  $f: \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $\mathbf{x} \mapsto f(\mathbf{x})$ , мы начинаем с исходной гипотезы  $\mathbf{x}_0$  о параметрах, которые хотим оптимизировать, а затем перебираем в соответствии с

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \gamma_i ((\nabla f)(\mathbf{x}_i))^T. \quad (7.6)$$

Для подходящего пошагового  $\gamma$ , последовательность  $f(\mathbf{x}_0) \geq f(\mathbf{x}_1) \geq \dots$  сходится к локальному минимуму.

### Пример 7.1

Рассмотрим квадратичную функцию в двух измерениях

$$f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \frac{1}{2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \begin{bmatrix} 2 & 1 \\ 1 & 20 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 5 \\ 3 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (7.7)$$

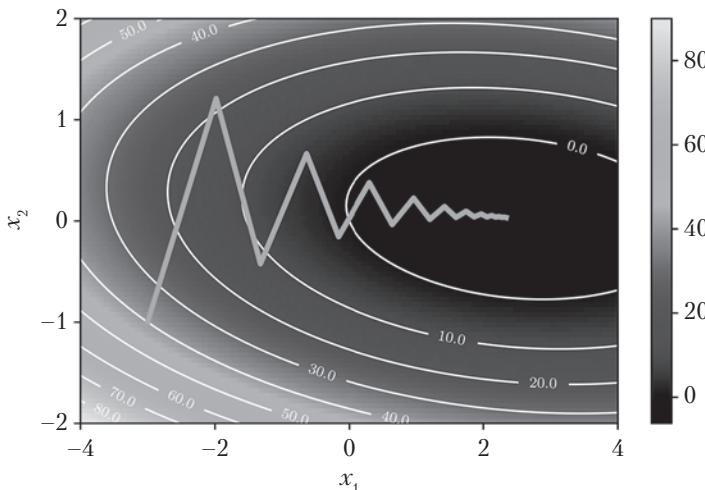
<sup>1</sup> Для градиентов используется то же соглашение, что действует для векторов.

с градиентом

$$\nabla f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \begin{bmatrix} 2 & 1 \\ 1 & 20 \end{bmatrix} - \begin{bmatrix} 5 \\ 3 \end{bmatrix}^T. \quad (7.8)$$

Начиная с исходного местоположения  $\mathbf{x}_0 = [-3, -1]^T$ , мы последовательно применяем (7.6) для получения последовательности оценок, которые сходятся к минимальному значению (проиллюстрировано на рис. 7.3). Видим (как на рисунке, так и подставив  $\mathbf{x}_0$  в (7.8) со значением  $\gamma = 0,085$ ), что градиент в  $\mathbf{x}_0$  указывает на северо-восток, приводя к  $\mathbf{x}_1 = [-1,98, 1,21]^T$ . Повторив этот аргумент, получим  $\mathbf{x}_2 = [-1,32, -0,42]^T$  и так далее.

**ПРИМЕЧАНИЕ** Поблизости от минимума градиентный спуск может быть относительно медленным: его асимптотический темп сходимости уступает многим другим методам. Взяв в качестве аналогии мяч, скатывающийся по склону холма, отметим, что если катится по узкой длинной канавке, то задача поставлена некорректно (Trefethen and Bau III, 1997). В случае некорректно поставленных выпуклых задач градиентный спуск все сильнее принимает форму зигзага, поскольку градиент направлен практически перпендикулярно кратчайшему направлению к точке минимума (рис. 7.3). ♦



**Рис. 7.3.** Градиентный спуск на двумерной квадратичной поверхности (показанной в виде тепловой карты). Описание см. в примере 7.1

### 7.1.1. Размер шага

Как упоминалось выше, при градиентном спуске важно правильно выбрать размер шага<sup>1</sup>. Если шаг слишком мал, то градиентный спуск может получиться медленным. Если выбранный размер шага слишком велик, то градиентный спуск может «зашкалить», не сойтись и даже разойтись. В следующем разделе мы поговорим об использовании импульса. Этот метод сглаживает эрратические явления при обновлениях градиента и гасит осцилляции.

Адаптивные градиентные методы позволяют пересматривать размер шага на каждой итерации, в зависимости от локальных свойств. Здесь есть две простые эвристики (Toissant, 2012):

- Когда после шага градиента значение функции увеличивается, это означает, что выбранный размер шага был слишком велик. Отмените шаг и уменьшите размер шага.
- Когда после шага значение функции уменьшается, это означает, что шаг мог бы быть и побольше. Попробуйте увеличить размер шага.

Хотя «отмена» шага кажется пустой тратой ресурсов, использование такой эвристики гарантирует монотонную сходимость.

#### Пример 7.2 (решение системы линейных уравнений)

При решении линейных уравнений вида  $\mathbf{A}\mathbf{x} = \mathbf{b}$ , на практике мы приблизительно решаем  $\mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0}$ , находя  $\mathbf{x}_*$ , минимизирующий квадратичную ошибку

$$\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 = (\mathbf{A}\mathbf{x} - \mathbf{b})^T(\mathbf{A}\mathbf{x} - \mathbf{b}) \quad (7.9)$$

при использовании евклидовой нормы. Градиент из (7.9) относительно  $\mathbf{x}$  равен

$$\nabla_{\mathbf{x}} = 2(\mathbf{A}\mathbf{x} - \mathbf{b})^T\mathbf{A}. \quad (7.10)$$

Этот градиент можно использовать напрямую в алгоритме градиентного спуска. Правда, в данном прикладном случае оказывается, что аналитическое решение существует, и его можно найти, установив градиент в ноль. Подробнее о решении задач со среднеквадратичной ошибкой мы поговорим в главе 9.

---

<sup>1</sup> Размер шага также называется скоростью обучения.

**ПРИМЕЧАНИЕ** При применении в ходе решения системы линейных уравнений  $\mathbf{A}\mathbf{x} = \mathbf{b}$  градиентный спуск может сходиться медленно. Скорость сходимости градиентного спуска зависит от *числа обусловленности*  $\kappa = \frac{\sigma(\mathbf{A})_{\max}}{\sigma(\mathbf{A})_{\min}}$ , которое является отношением максимального единичного значения к минимальному (раздел 4.5) у  $\mathbf{A}$ . В сущности, число обусловленности дает отношение наиболее искривленного варианта направления к наименее искривленному, что наглядно соответствует следующей картине: некорректно поставленные задачи похожи на узкие длинные канавки: графики очень искривлены в одном направлении, но плоские в другом. Вместо того чтобы непосредственно решать  $\mathbf{A}\mathbf{x} = \mathbf{b}$ , можно вместо этого решить  $\mathbf{P}^{-1}(\mathbf{A}\mathbf{x} - \mathbf{b}) = 0$ , где  $\mathbf{P}$  называется *предобусловливатель*. Цель — спроектировать  $\mathbf{P}^{-1}$ , так чтобы у  $\mathbf{P}^{-1}\mathbf{A}$  число обусловленности было лучше, но в то же время чтобы  $\mathbf{P}^{-1}$  было легко вычислять. Подробнее о градиентном спуске, предобусловливании и сходимости можно почитать в Boyd and Vandenberghe (2004, глава 9). ◆

### 7.1.2. Градиентный спуск с импульсом

Как показано на рис. 7.3, сходимость градиентного спуска может происходить очень медленно, если кривизна оптимизационной поверхности такова, что некоторые области на ней плохо шкалируются. Кривизна такова, что шаги градиентного спуска перекидываются между стенками долины, и приближение к оптимуму происходит малыми шагами. Предлагаемая коррекция, позволяющая улучшить сходимость, — вложить в градиентный спуск некоторую память.

Градиентный спуск с импульсом<sup>1</sup> (Rumelhart et al., 1986) — это метод, при котором дополнительный член используется в алгоритме для запоминания того, что произошло на предыдущей итерации. Такая память гасит осцилляции и сглаживает обновления градиента. Продолжая аналогию с шариком, можно сказать, что импульс в данной модели эмулирует тяжелый шар, который сложно сбить с курса. Идея реализовать обновление градиента с привлечением памяти связана с использованием скользящего среднего. Метод, основанный на использовании импульса, запоминает обновление  $\Delta\mathbf{x}_i$  на каждой итерации  $i$  и определяет следующее обновление как линейную комбинацию актуального и предыдущего градиентов

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \gamma_i((\nabla f)(\mathbf{x}_i))^T + \alpha\Delta\mathbf{x}_i; \quad (7.11)$$

$$\Delta\mathbf{x}_i = \mathbf{x}_i - \mathbf{x}_{i-1} = \alpha\Delta\mathbf{x}_{i-1} - \gamma_{i-1}((\nabla f)(\mathbf{x}_{i-1}))^T, \quad (7.12)$$

---

<sup>1</sup> Goh (2017) написал понятный пост о градиентном спуске с импульсом.

где  $\alpha \in [0, 1]$ . Иногда градиент будет известен нам лишь приблизительно. В таких случаях член импульса полезен, так как выравнивает различные зашумленные оценки градиента. Один очень дальний способ получить приблизительный градиент — использовать стохастическое приближение, о котором мы поговорим далее.

### 7.1.3. Стохастический градиентный спуск

На вычисление градиента может потребоваться значительное время. Однако часто удается найти «дешевое» приближение градиента. Приближение градиента так или иначе полезно, поскольку оно указывает примерно в том же направлении, что и истинный градиент.

*Стохастический градиентный спуск* (stochastic gradient descent, SGD) — это стохастическое приближение метода градиентного спуска для минимизации целевой функции, которая записывается как сумма дифференцируемых функций. Термин «стохастический» в данном случае означает признание того, что точный градиент нам не известен, но мы знаем несколько зашумленное приближение к нему. Ограничиваая вероятностное распределение приблизительных градиентов, мы все равно можем теоретически гарантировать сходимость SGD.

В машинном обучении, имея  $n = 1, \dots, N$  точек данных, мы часто рассматриваем целевые функции, являющиеся суммой потерь  $L_n$ , вызванных каждым примером  $n$ . В математической нотации это выражается как

$$L(\boldsymbol{\theta}) = \sum_{n=1}^N L_n(\boldsymbol{\theta}), \quad (7.13)$$

где  $\boldsymbol{\theta}$  — вектор интересующих нас параметров, то есть мы хотим найти  $\boldsymbol{\theta}$ , минимизирующий  $L$ . Пример, связанный с регрессией (глава 9), — отрицательное логарифмическое правдоподобие, выражаемое как сумма логарифмических правдоподобий отдельных примеров, так что

$$L(\boldsymbol{\theta}) = -\sum_{n=1}^N \log p(y_n | \mathbf{x}_n, \boldsymbol{\theta}), \quad (7.14)$$

где  $\mathbf{x}_n \in \mathbb{R}^D$  — это учебные входные данные,  $y_n$  — цели обучения, а  $\boldsymbol{\theta}$  — параметры регрессионной модели.

Стандартный градиентный спуск в том виде, как он представлен выше, — это метод «пакетной» оптимизации, то есть проводимой с применением всего учебного множества, путем обновления вектора параметров согласно

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \gamma_i (\nabla L(\boldsymbol{\theta}_i))^T = \boldsymbol{\theta}_i - \gamma_i \sum_{n=1}^N (\nabla L_n(\boldsymbol{\theta}_i))^T \quad (7.15)$$

для подходящего параметра ширины шага  $\gamma$ . Для получения результирующего градиента суммы могут потребоваться затратные расчеты градиентов от всех отдельных функций  $L_n$ . Когда учебное множество колоссально и/или существуют простые формулы, расчет сумм градиентов становится очень затратной операцией.

Рассмотрим член  $\sum_{n=1}^N (\nabla L_n(\theta_i))$  в (7.15). Здесь мы можем сократить объем вычислений, взяв сумму меньшего множества  $L_n$ . В отличие от пакетного градиентного спуска, использующего  $L_n$  для  $n = 1, \dots, N$ , мы случайным образом выберем подмножество  $L_n$  для градиентного спуска с мини-пакетами. В примере, доведенном до крайности, мы случайным образом выбираем всего одно значение  $L_n$  для оценки градиента. Ключевая причина, по которой действительно разумно брать подмножество данных, понятна, если учесть, что для сходимости градиентного спуска требуется лишь обеспечить, чтобы этот градиент был неискаженной оценкой истинного градиента. Фактически, член  $\sum_{n=1}^N (\nabla L_n(\theta_i))$  в (7.15) — это эмпирическая оценка ожидаемого значения градиента (раздел 6.4.1). Если размер мини-батча сравнительно велик, то оценить градиент можно сравнительно точно, уменьшив вариативность при обновлении параметров. Более того, крупные мини-батчи более выгодны благодаря сильной оптимизации матричных операций в векторизованных реализациях стоимости и градиента. Уменьшение вариативности приводит к более стабильной сходимости, но каждая операция расчета градиента будет обходиться дороже.

Напротив, мелкие мини-батчи поддаются сравнительно быстрой оценке. Если следить, чтобы мини-батчи оставались компактными, то зашумленность нашей оценки градиента позволит нам избавиться от некоторых плохих локальных оптимумов, с которыми в противном случае мы могли бы застрять. В машинном обучении методы оптимизации используются для обучения путем минимизации целевой функции для учебных данных, но стратегическая цель — улучшить обобщающую способность и производительность (глава 8). Поскольку цель машинного обучения не обязательно заключается в точной оценке минимума целевой функции, широко применяются приближенные градиенты с использованием мини-батчей. Стохастический градиентный спуск очень эффективен при решении крупномасштабных задач машинного обучения (Bottou et al., 2018), например при обучении глубоких нейронных сетей на миллионах изображений (Dean et al., 2012), работе с моделями топиков (Hoffman et al., 2013), обучении с подкреплением (Mnih et al., 2015) либо при обучении крупномасштабных моделей гауссовых процессов (Hensman et al., 2013; Gal et al., 2014).

## 7.2. ОГРАНИЧЕННАЯ ОПТИМИЗАЦИЯ И МНОЖИТЕЛИ ЛАГРАНЖА

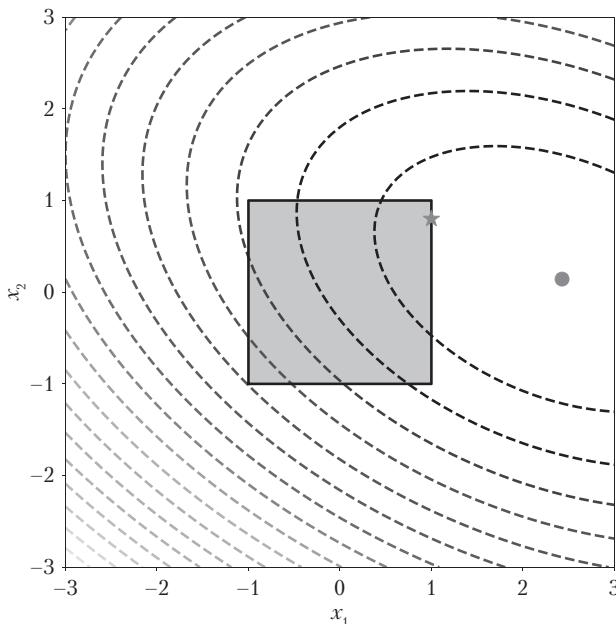
В предыдущем разделе мы решали задачу для минимума функции

$$\min_x f(\mathbf{x}), \quad (7.16)$$

где  $f: \mathbb{R}^D \rightarrow \mathbb{R}$ .

В этом разделе поработаем с дополнительными ограничениями. То есть для вещественнозначной функции  $g_i: \mathbb{R}^D \rightarrow \mathbb{R}$  при  $i = 1, \dots, m$  мы рассмотрим задачу ограниченной оптимизации (рис. 7.4)

$$\begin{aligned} & \min_x f(\mathbf{x}), \\ & \text{для которой справедливо } g_i(\mathbf{x}) \leq 0 \text{ для всех } i = 1, \dots, m. \end{aligned} \quad (7.17)$$



**Рис. 7.4.** Иллюстрация выпуклой оптимизации. Задача без ограничений (обозначенная контурными линиями) имеет минимум справа (обозначен кружком). Ограничения, обозначенные рамкой ( $-1 \leq x \leq 1$  и  $-1 \leq y \leq 1$ ), требуют, чтобы целевое решение находилось в пределах рамки, таким образом в результате получаем целевое значение, обозначенное звездочкой

Стоит отметить, что функции  $f$  и  $g_i$  могут быть в общем случае невыпуклыми, но в следующем разделе мы рассмотрим выпуклый случай.

Один очевидный, но не очень практический способ преобразовать ограниченную задачу (7.17) в неограниченную — использовать индикаторную функцию

$$J(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^m \mathbf{1}(g_i(\mathbf{x})), \quad (7.18)$$

где  $\mathbf{1}(z)$  — это бесконечная ступенчатая функция

$$\mathbf{1}(z) = \begin{cases} 0 & \text{если } z \leq 0 \\ \infty & \text{иначе} \end{cases}. \quad (7.19)$$

В результате, если ограничение не удовлетворяется, получаем бесконечный штраф и, соответственно, все то же решение. Но бесконечная ступенчатая функция столь же сложно поддается оптимизации. Чтобы справиться с этой трудностью, можно прибегнуть к *множителям Лагранжа*. Идея множителей Лагранжа в том, чтобы заменить ступенчатую функцию линейной.

С задачей (7.17) мы ассоциируем *лагранжиан*, вводя множители Лагранжа  $\lambda_i \geq 0$ , соответствующие каждому отдельному ограничению в неравенстве (Boyd and Vandenberghe, 2004, глава 4), так что

$$\mathfrak{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i=1}^m \lambda_i g_i(\mathbf{x}) = \quad (7.20a)$$

$$= f(\mathbf{x}) + \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{x}), \quad (7.20b)$$

где в последней строке мы склеили все ограничения  $g_i(\mathbf{x})$  в вектор  $\mathbf{g}(\mathbf{x})$ , а все множители Лагранжа — в вектор  $\boldsymbol{\lambda} \in \mathbb{R}^m$ .

Теперь познакомимся с идеей *лагранжевой двойственности*. В принципе, при оптимизации идея двойственности заключается в том, чтобы преобразовать задачу оптимизации, содержащую одно множество переменных  $\mathbf{x}$  (прямые переменные), в другую задачу оптимизации, содержащую другое множество переменных  $\boldsymbol{\lambda}$  (двойственные переменные). Мы обсудим два разных подхода к двойственности; в этом разделе поговорим о лагранжевой двойственности, а в разделе 7.3.3 речь пойдет о двойственности Лежандра — Фенхеля.

**Определение 7.1.** Задача в (7.17)

$$\min_{\mathbf{x}} \quad f(\mathbf{x}), \quad (7.21)$$

для которой справедливо  $g_i(\mathbf{x}) \leq 0$  для всех  $i = 1, \dots, m$ ,

известна под названием «*прямая задача*», и в ней мы оперируем прямыми переменными  $x$ . Связанная с ней *двойственная задача Лагранжа* задается как

$$\max_{\lambda \in \mathbb{R}^m} \mathfrak{D}(\lambda), \quad (7.22),$$

для которой справедливо  $\lambda \geq 0$ ,

где  $\lambda$  — это двойственные переменные, и  $\mathfrak{D}(\lambda) = \min_{x \in \mathbb{R}^d} \mathfrak{L}(x, \lambda)$ .

**ПРИМЕЧАНИЕ** Обсуждая определение 7.1, мы используем две концепции, которые также интересны сами по себе (Boyd and Vandenberghe, 2004).

Первая — это *неравенство минимакса*, согласно которому для любой функции с двумя аргументами  $\phi(x, y)$  максимин меньше минимакса, то есть

$$\max_y \min_x \phi(x, y) \leq \min_x \max_y \phi(x, y). \quad (7.23)$$

Это неравенство можно доказать, рассмотрев неравенство

$$\text{для всех } x, y \quad \min_x \phi(x, y) \leq \max_y \phi(x, y). \quad (7.24)$$

Обратите внимание: взяв максимум от  $y$  в левой части (7.24), неравенство сохраняется, поскольку оно верно для всех  $y$ . Аналогично, можно взять минимум от  $x$  в правой части (7.24), получив таким образом (7.23).

Вторая концепция — это *слабая двойственность*, использующая (7.23), чтобы продемонстрировать, что прямые значения всегда больше двойственных или равны им. Это подробнее описано в (7.27). ♦

Напомню, что разница между  $J(x)$  в (7.18) и лагранжианом в (7.20b) заключается в том, что мы ослабили индикаторную функцию, сделав ее линейной. Следовательно, при  $\lambda \geq 0$  лагранжиан  $\mathfrak{L}(x, \lambda)$  является нижним пределом  $J(x)$ . Таким образом, максимум  $\mathfrak{L}(x, \lambda)$  относительно  $\lambda$  равен

$$J(x) = \max_{\lambda \geq 0} \mathfrak{L}(x, \lambda). \quad (7.25)$$

Напомним, что исходная проблема сводилась к минимизации  $J(x)$ ,

$$\min_{x \in \mathbb{R}^d} \max_{\lambda \geq 0} \mathfrak{L}(x, \lambda). \quad (7.26)$$

Из неравенства минимакса (7.23) следует, что при перемене порядка минимальных и максимальных значений результирующее значение уменьшается.

$$\min_{x \in \mathbb{R}^d} \max_{\lambda \geq 0} \mathfrak{L}(x, \lambda) \geq \max_{\lambda \geq 0} \min_{x \in \mathbb{R}^d} \mathfrak{L}(x, \lambda). \quad (7.27)$$

Это явление также называется *слабая двойственность*. Обратите внимание: внутренняя составляющая правой части является двойственной целевой функцией  $\mathfrak{D}(\lambda)$ , определенной далее.

По сравнению с исходной задачей оптимизации, в которой есть ограничения,  $\min_{x \in \mathbb{R}^d} \mathfrak{L}(x, \lambda)$  — это задача оптимизации без ограничений для заданного значения  $\lambda$ . Если решить  $\min_{x \in \mathbb{R}^d} \mathfrak{L}(x, \lambda)$  легко, то легко решить и всю задачу. Причина в том, что внешняя задача (максимизация для  $\lambda$ ) дает максимум для множества аффинных функций и, следовательно, это вогнутая функция, пусть и  $f(\cdot)$ , и  $g_i(\cdot)$  могут быть невогнутыми. Максимум вогнутой функции эффективно поддается вычислению.

Предположив, что  $f(\cdot)$  и  $g_i(\cdot)$  являются дифференцируемыми, решаем задачу двойственности Лагранжа относительно  $x$ , устанавливая при этом дифференциал в 0 и решая задачу для целевого значения. Два конкретных примера мы обсудим в разделах 7.3.1 и 7.3.2, где  $f(\cdot)$  и  $g_i(\cdot)$  являются выпуклыми.

**ПРИМЕЧАНИЕ** Рассмотрим выражение (7.17) с дополнительными ограничениями равенства

$$\begin{aligned} \min_x \quad & f(x), \\ \text{для которого справедливо} \quad & g_i(x) \leq 0 \quad \text{для всех } i = 1, \dots, m; \\ \text{для которого справедливо} \quad & h_j(x) = 0 \quad \text{для всех } j = 1, \dots, n. \end{aligned} \tag{7.28}$$

Можно смоделировать ограничения равенства, заменив их двумя ограничениями неравенства. То есть для каждого ограничения равенства  $h_j(x) = 0$  справедливо, что его можно равнозначно заменить двумя ограничениями  $h_j(x) \leq 0$  и  $h_j(x) \geq 0$ . Оказывается, что получающиеся в результате множители Лагранжа не имеют ограничений.

Следовательно, мы ограничиваем множители Лагранжа так, чтобы ограничения из (7.28) оставались неотрицательными, и множители Лагранжа, соответствующие ограничениям равенства, мы оставляем неограниченными. ♦

## 7.3. ВЫПУКЛАЯ ОПТИМИЗАЦИЯ

Сосредоточимся на особенно полезном классе задач оптимизации, где мы можем гарантировать глобальную оптимальность. Если  $f(\cdot)$  — выпуклая функция, и ограничения  $g(\cdot)$  и  $h(\cdot)$  задают выпуклые множества, задача называется *задачей выпуклой оптимизации*. При такой постановке задачи существует *сильная двойственность*: оптимальное решение двойственной задачи совпада-

ет с оптимальным решением исходной задачи. Различие между выпуклыми функциями и выпуклыми множествами часто обходится стороной в литературе по машинному обучению, однако обычно можно понять значение этого слова из контекста.

**Определение 7.2.** Множество  $\mathcal{C}$  называется *выпуклым множеством*, если для любых  $x, y \in \mathcal{C}$  и любого скаляра  $\theta$  из интервала  $[0, 1]$  выполняется

$$\theta x + (1 - \theta)y \in \mathcal{C}. \quad (7.29)$$

Выпуклые множества — это такие множества, что отрезок, соединяющий любые две точки множества, лежит в нем целиком. Выпуклое и невыпуклое множество показаны на рис. 7.5 и 7.6 соответственно.

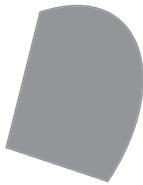


Рис. 7.5. Пример выпуклого множества



Рис. 7.6. Пример невыпуклого множества

Выпуклые функции — это такие функции, что отрезок, соединяющий любые две точки на их графике, лежит над графиком. На рис. 7.7 показана выпуклая функция.

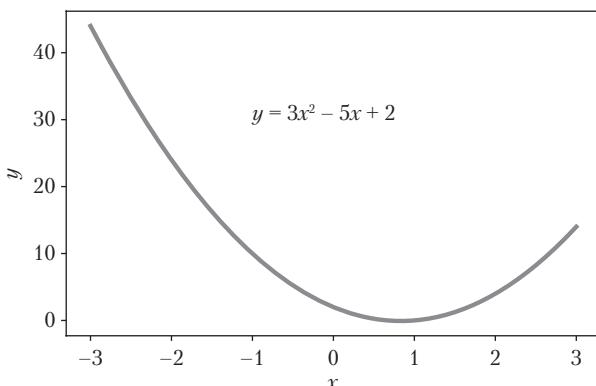


Рис. 7.7. Пример выпуклой функции

**Определение 7.3.** Пусть функция  $f: \mathbb{R}^D \rightarrow \mathbb{R}$  определена на выпуклом множестве. Функция  $f$  *выпукла*, если для любых  $\mathbf{x}, \mathbf{y}$  из области определения  $f$  и любого скаляра  $\theta$  из интервала  $[0, 1]$

$$f(\theta\mathbf{x} + (1 - \theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1 - \theta)f(\mathbf{y}). \quad (7.30)$$

**ПРИМЕЧАНИЕ** *Вогнутая функция* — противоположное к выпуклой понятие. ◆

Ограничения, заданные функциями  $g(\cdot)$  и  $h(\cdot)$  в (7.28), можно представить как некоторые множества. Еще одно соотношение между выпуклыми функциями и выпуклыми множествами мы получим из рассмотрения множества, «заполняющего» график выпуклой функции. Выпуклая функция похожа на чашку, и мы представляем, как наполняем ее водой. Получившееся множество, называемое *надграфиком* выпуклой функции, является выпуклым.

Если функция  $f: \mathbb{R}^n \rightarrow \mathbb{R}$  дифференцируема, выпуклость можно описать в терминах ее градиента  $\nabla_{\mathbf{x}} f(\mathbf{x})$  (раздел 5.2). Функция  $f(\mathbf{x})$  выпукла, если и только если для любых двух точек  $\mathbf{x}, \mathbf{y}$  выполняется неравенство

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla_{\mathbf{x}} f(\mathbf{x})^T (\mathbf{y} - \mathbf{x}). \quad (7.31)$$

Если, кроме того, мы знаем, что  $f(\mathbf{x})$  дважды дифференцируема, то есть для всех точек области определения существует матрица Гессе (5.147), то  $f(\mathbf{x})$  будет выпуклой в том и только том случае, когда  $\nabla_{\mathbf{x}}^2 f(\mathbf{x})$  положительно полуопределенна (Boyd and Vandenberghe, 2004).

### Пример 7.3

Отрицательная энтропия  $f(x) = x \log_2 x$  выпукла для  $x > 0$ . График этой функции изображен на рис. 7.8, и мы видим, что она выпукла. В качестве иллюстрации к предыдущим определениям выпуклости, проведем вычисления для точек  $x = 2$  и  $x = 4$ . Заметим, что для доказательства выпуклости  $f(x)$  мы должны были бы проверить это свойство для всех  $x \in \mathbb{R}$ .

Вспомним определение 7.3. Рассмотрим точку, лежащую ровно посередине между нашими двумя точками (то есть  $\theta = 0,5$ ); тогда в левой части стоит  $f(0,5 \cdot 2 + 0,5 \cdot 4) = 3 \log_2 3 \approx 4,75$ . Правая часть равна  $0,5(2\log_2 2) + 0,5(4\log_2 4) = 1 + 4 = 5$ . Таким образом, условие из определения выполнено.

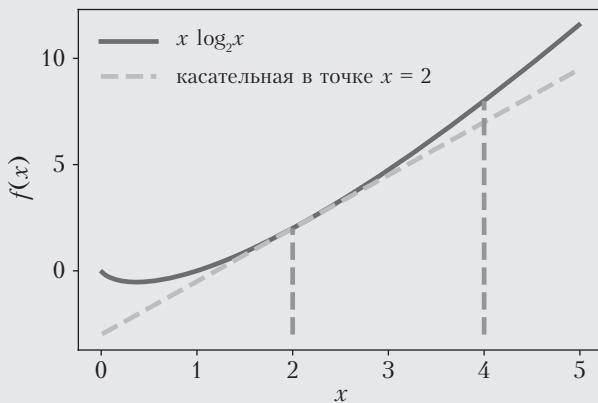
Так как  $f(x)$  дифференцируема, можно вместо этого использовать условие (7.31). Вычислив производную  $f'(x)$ , получаем

$$\nabla_x (x \log_2 x) = 1 \cdot \log_2 x + x \cdot \frac{1}{x \log_e 2} = \log_2 x + \frac{1}{\log_e 2}. \quad (7.32)$$

Взяв для проверки те же самые точки  $x = 2$  и  $x = 4$ , видим в левой части (7.31)  $f(4) = 8$ . Правая часть равна

$$f(x) + \nabla_x^T (y - x) = f(2) + \nabla f(2) \cdot (4 - 2) = \quad (7.33a)$$

$$= 2 + \left( 1 + \frac{1}{\log_e 2} \right) \cdot 2 \approx 6,9. \quad (7.33b)$$



**Рис. 7.8.** Функция отрицательной энтропии (выпуклая) и касательная к ней в точке  $x = 2$

Можно проверить выпуклость функции или множества по определению. На практике мы часто используем для этого то, что некоторые операции сохраняют выпуклость. По сути, это идея замкнутости, введенная в главе 2 для векторных пространств (хотя детали и сильно разнятся).

#### Пример 7.4

Взвешенная сумма выпуклых функций с неотрицательными коэффициентами выпукла. Заметим, что для выпуклой функции  $f$  и неотрицательного скаляра  $\alpha$  функция  $\alpha f$  будет выпуклой. Это легко увидеть, домножив

на  $\alpha$  обе части неравенства в определении 7.3 и учитывая, что умножение на неотрицательное число не меняет знак неравенства.

Если функции  $f_1$  и  $f_2$  выпуклы, то по определению

$$f_1(\theta \mathbf{x} + (1 - \theta) \mathbf{y}) \leq \theta f_1(\mathbf{x}) + (1 - \theta) f_1(\mathbf{y}); \quad (7.34)$$

$$f_2(\theta \mathbf{x} + (1 - \theta) \mathbf{y}) \leq \theta f_2(\mathbf{x}) + (1 - \theta) f_2(\mathbf{y}). \quad (7.35)$$

Складывая левые и правые части соответственно, получим

$$\begin{aligned} & f_1(\theta \mathbf{x} + (1 - \theta) \mathbf{y}) + f_2(\theta \mathbf{x} + (1 - \theta) \mathbf{y}) \leq \\ & \leq \theta f_1(\mathbf{x}) + (1 - \theta) f_1(\mathbf{y}) + \theta f_2(\mathbf{x}) + (1 - \theta) f_2(\mathbf{y}), \end{aligned} \quad (7.36)$$

где правую часть можно переписать как

$$\theta(f_1(\mathbf{x}) + f_2(\mathbf{x})) + (1 - \theta)(f_1(\mathbf{y}) + f_2(\mathbf{y})), \quad (7.37)$$

и мы доказали, что сумма выпуклых функций выпукла.

Объединяя два доказанных нами факта, видим, что  $\alpha f_1(\mathbf{x}) + \beta f_2(\mathbf{x})$  выпукла при  $\alpha, \beta \geq 0$ . Аналогичным образом можно обобщить это свойство замкнутости на взвешенные суммы более чем двух выпуклых функций с неотрицательными коэффициентами.

**ПРИМЕЧАНИЕ** Неравенство (7.30) иногда называют *неравенством Йенсена*. На самом деле, целый класс неравенств, касающихся взвешенных сумм выпуклых функций с неотрицательными коэффициентами, называют неравенствами Йенсена. ♦

В заключение сформулируем определение: задача оптимизации с ограничениями называется *задачей выпуклой оптимизации*, если

$$\begin{aligned} & \min_x f(\mathbf{x}) \\ & \text{при } g_i(\mathbf{x}) \leq 0 \quad \text{для всех } i = 1, \dots, m; \\ & \text{при } h_j(\mathbf{x}) = 0 \quad \text{для всех } j = 1, \dots, n, \end{aligned} \quad (7.38)$$

где  $f(\mathbf{x})$  и все  $g_i(\mathbf{x})$  — выпуклые функции и все  $h_j(\mathbf{x}) = 0$  — выпуклые множества. В дальнейшем мы опишем два глубоко изученных и широко применимых класса задач выпуклой оптимизации.

### 7.3.1. Линейное программирование

Рассмотрим частный случай задачи оптимизации, в котором все функции линейны, то есть

$$\begin{aligned} \min_{x \in \mathbb{R}^d} & \quad \mathbf{c}^T \mathbf{x} \\ \text{при } & \quad \mathbf{A}\mathbf{x} \leq \mathbf{b}, \end{aligned} \tag{7.39}$$

где  $\mathbf{A} \in \mathbb{R}^{m \times d}$  и  $\mathbf{b} \in \mathbb{R}^m$ . Такие задачи известны как *задачи линейного программирования*<sup>1</sup>. В нашей задаче  $d$  неизвестных и  $m$  линейных ограничений. Лагранжиан задается формулой

$$\mathfrak{L}(\mathbf{x}, \lambda) = \mathbf{c}^T \mathbf{x} + \lambda^T (\mathbf{A}\mathbf{x} - \mathbf{b}), \tag{7.40}$$

где  $\lambda \in \mathbb{R}^m$  — вектор неотрицательных множителей Лагранжа. Сгруппировав слагаемые, содержащие  $\mathbf{x}$ , получим

$$\mathfrak{L}(\mathbf{x}, \lambda) = (\mathbf{c} + \mathbf{A}^T \lambda)^T \mathbf{x} - \lambda^T \mathbf{b}. \tag{7.41}$$

Возьмем производную  $\mathfrak{L}(\mathbf{x}, \lambda)$  по  $\mathbf{x}$  и приравняем ее к нулю:

$$\mathbf{c} + \mathbf{A}^T \lambda = \mathbf{0}. \tag{7.42}$$

Таким образом, двойственный лагранжиан равен  $\mathfrak{D}(\lambda) = -\lambda^T \mathbf{b}$ . Вспомним, что мы хотим максимизировать  $\mathfrak{D}(\lambda)$ . Кроме ограничения, состоящего в равенстве  $\mathfrak{L}(\mathbf{x}, \lambda)$  нулю, мы также имеем ограничение  $\lambda \geq \mathbf{0}$ , и в итоге приходим к следующей двойственной задаче оптимизации<sup>2</sup>:

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^m} & \quad -\mathbf{b}^T \lambda \\ \text{при } & \quad \mathbf{c} + \mathbf{A}^T \lambda = \mathbf{0}, \\ & \quad \lambda \geq \mathbf{0}. \end{aligned} \tag{7.43}$$

Это снова задача линейного программирования, но уже от  $m$  переменных. Мы можем выбирать, решать ли исходную задачу (7.39) или двойственную (7.43) в зависимости от того,  $m$  или  $d$  больше. Напомним, что  $d$  — число переменных, а  $m$  — число ограничений в исходной задаче линейного программирования.

<sup>1</sup> Линейное программирование — один из самых часто применяемых на практике подходов.

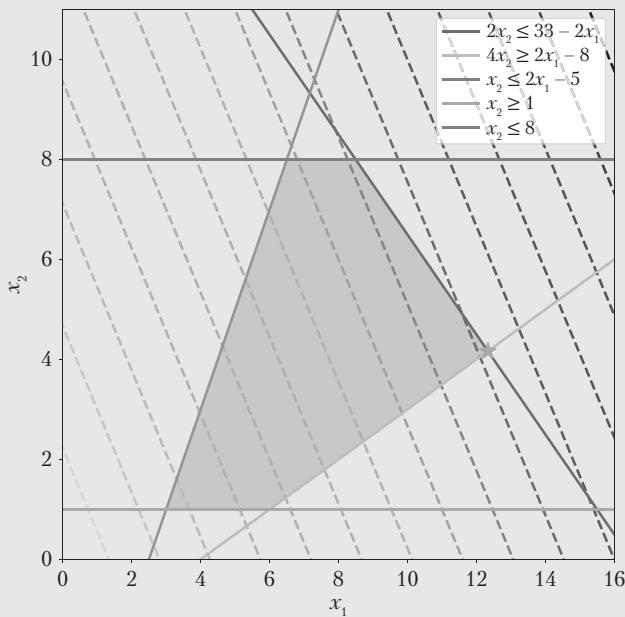
<sup>2</sup> Принято, что исходной задачей является задача минимизации, а двойственной — задача максимизации.

**Пример 7.5 (задача линейного программирования)**

Рассмотрим задачу с двумя переменными:

$$\begin{aligned} \min_{x \in \mathbb{R}^2} \quad & -\begin{bmatrix} 5 \\ 3 \end{bmatrix}^\top \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \text{при} \quad & \begin{bmatrix} 2 & 2 \\ 2 & -4 \\ -2 & 1 \\ 0 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 33 \\ 8 \\ 5 \\ -1 \\ 8 \end{bmatrix} \end{aligned} \quad (7.44)$$

Она изображена на рис. 7.9. Целевая функция линейна, так что линии ее равных значений — прямые. Стандартная запись ограничений содержитя в легенде. Оптимальное значение должно лежать в допустимой области (закрашена) и отмечено звездочкой.



**Рис. 7.9.** Задача линейного программирования. Оптимальное значение при данных ограничениях показано звездочкой

### 7.3.2. Квадратичное программирование

Рассмотрим случай выпуклой квадратичной целевой функции и аффинных ограничений, то есть

$$\min_{x \in \mathbb{R}^d} \frac{1}{2} x^T Q x + c^T x \quad (7.45)$$

при  $Ax \leq b$ ,

где  $A \in \mathbb{R}^{m \times d}$ ,  $b \in \mathbb{R}^m$  и  $c \in \mathbb{R}^d$ . Симметричная квадратная матрица  $Q \in \mathbb{R}^{d \times d}$  положительно определена, так что целевая функция выпукла. Такие задачи известны как задачи квадратичной оптимизации. Заметим, что в нашей задаче  $d$  переменных и  $m$  линейных ограничений.

#### Пример 7.6 (квадратичное программирование)

Рассмотрим задачу квадратичного программирования

$$\min_{x \in \mathbb{R}^2} \frac{1}{2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 5 \\ 3 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (7.46)$$

$$\text{при } \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad (7.47)$$

с двумя неизвестными. Задача показана также на рис. 7.4. Целевая функция квадратична, с симметричной положительно определенной матрицей  $Q$ . Линии равных значений имеют форму эллипсов. Оптимальное значение должно лежать в допустимой области (закрашена) и отмечено звездочкой.

Лагранжиан задается формулой

$$\mathcal{L}(x, \lambda) = \frac{1}{2} x^T Q x + c^T x + \lambda^T (Ax - b), \quad (7.48a)$$

$$= \frac{1}{2} x^T Q x + (c + A^T \lambda)^T x - \lambda^T b \quad (7.48b)$$

(мы снова перегруппировали слагаемые). Возьмем производную  $\mathcal{L}(x, \lambda)$  по  $x$  и приравняем ее нулю:

$$Qx + (c + A^T \lambda) = 0. \quad (7.49)$$

Если  $\mathbf{Q}$  обратима, получим

$$\mathbf{x} = -\mathbf{Q}^{-1}(\mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda}). \quad (7.50)$$

Подставив (7.50) в исходный лагранжиан  $\mathfrak{L}(\mathbf{x}, \boldsymbol{\lambda})$ , получим двойственный лагранжиан

$$\mathfrak{D}(\boldsymbol{\lambda}) = -\frac{1}{2}(\mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda})^T \mathbf{A} \mathbf{Q}^{-1}(\mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda}) - \boldsymbol{\lambda}^T \mathbf{b}. \quad (7.51)$$

В итоге, двойственная задача оптимизации выглядит как

$$\max_{\mathbf{x} \in \mathbb{R}^m} -\frac{1}{2}(\mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda})^T \mathbf{A} \mathbf{Q}^{-1}(\mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda}) - \boldsymbol{\lambda}^T \mathbf{b} \quad (7.52)$$

при  $\boldsymbol{\lambda} \geq \mathbf{0}$ .

С приложениями квадратичной оптимизации к машинному обучению мы познакомимся в главе 12.

### 7.3.3. Преобразование Лежандра — Фенхеля и выпуклое сопряжение

Вернемся к идее двойственности из раздела 7.2, пока не рассматривая ограничения. Важным фактом о выпуклых множествах является то, что их можно описать с помощью опорных гиперплоскостей. Гиперплоскость называется *опорной* для выпуклого множества, если она пересекается с этим множеством, и все множество лежит по одну сторону от нее. Напомним, что мы можем «заполнить» график выпуклой функции, чтобы получить надграфик, и она будет выпуклым множеством. Таким образом, мы можем описать и выпуклые функции через их опорные плоскости. Далее заметим, что опорная гиперплоскость пересекается с графиком только в одной точке, и на самом деле является касательной к нему. Вспомним теперь, что тангенс угла наклона касательной к графику функции  $f(\mathbf{x})$  в точке  $\mathbf{x}_0$  равен значению градиента в этой точке. Таким образом, так как выпуклые множества описываются с помощью опорных плоскостей, выпуклые функции описываются своей функцией градиента. *Преобразование Лежандра*<sup>1</sup> формализует эту идею.

Начнем с наиболее общего определения, к сожалению, не слишком понятного интуитивно, а затем рассмотрим частные случаи, чтобы связать определение с выработанным ранее интуитивным пониманием. Преобразование Лежандра — Фенхеля (слово «преобразование» здесь имеет тот же смысл, что и, например,

---

<sup>1</sup> Студенты-физики часто знакомятся с преобразованием Лежандра как с преобразованием в классической механике, связывающим лагранжиан и гамильтониан.

в словосочетании «преобразование Фурье») сопоставляет выпуклой дифференцируемой функции  $f(\mathbf{x})$  функцию, зависящую от касательных  $s(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x})$ . Стоит подчеркнуть, что мы преобразуем именно функцию  $f(\cdot)$ , а не переменную  $\mathbf{x}$  или значение в точке  $\mathbf{x}$ . *Преобразование Лежандра — Фенхеля* также известно как выпуклое сопряжение (причины почему мы скоро увидим) и тесно связано с двойственностью (Hiriart-Urruty and Lemaréchal, 2001, глава 5).

**Определение 7.4.** Выпукло сопряженная функция к функции  $f: \mathbb{R}^D \rightarrow \mathbb{R}$  — это функция  $f^*$ , заданная формулой

$$f^*(\mathbf{s}) = \sup_{\mathbf{x} \in \mathbb{R}^D} (\langle \mathbf{s}, \mathbf{x} \rangle - f(\mathbf{x})). \quad (7.53)$$

Заметим, что такое определение выпуклого сопряжения не требует от функции  $f$  быть ни выпуклой, ни дифференцируемой. В определении 7.4 мы использовали произвольное скалярное произведение (раздел 3.2), но далее в этом разделе мы для простоты рассматриваем стандартное скалярное произведение конечномерных векторов ( $\langle \mathbf{s}, \mathbf{x} \rangle = \mathbf{s}^T \mathbf{x}$ ).

Чтобы понять геометрический смысл определения 7.4, рассмотрим «хорошую» одномерную выпуклую дифференцируемую функцию<sup>1</sup>, например  $f(x) = x^2$ . Заметим, что так как мы рассматриваем одномерную задачу, гиперплоскости представляют собой прямые. Рассмотрим прямую  $y = sx + c$ . Мы знаем, что выпуклые функции можно описать с помощью опорных плоскостей, так что давайте попробуем описать нашу  $f(x)$  с помощью опорных прямых. Зафиксируем градиент прямой  $s \in \mathbb{R}$  и для каждой точки  $(x_0, f(x_0))$  на графике  $f$  найдем минимальное значение  $c$ , такое что прямая проходит через  $(x_0, f(x_0))$ . Заметим, что это минимальное значение  $c$  соответствует ситуации, когда прямая с фиксированным  $s$  является касательной к графику  $f(x) = x^2$ . Прямая, проходящая через точку  $(x_0, f(x_0))$  с градиентом  $s$ , задается формулой

$$y - f(x_0) = s(x - x_0). \quad (7.54)$$

$y$ -координаты точки на этой прямой задаются формулой  $-sc + f(x_0)$ . Тогда минимальное значение  $c$ , при котором  $y = sx + c$  пересекает график  $f$ , равно

$$\inf_{x_0} -sx_0 + f(x_0). \quad (7.55)$$

Выпукло сопряженная к данной функции по определению — выражение, противоположное к данному. Эти рассуждения не использовали выпуклость и дифференцируемость нашей функции (а также ее зависимость от только

---

<sup>1</sup> Эти рассуждения проще понять, если следить за ними по рисунку.

одной переменной), так что будут верны и для  $f : \mathbb{R}^D \rightarrow \mathbb{R}$ , невыпуклых и не-дифференцируемых<sup>1</sup>.

**ПРИМЕЧАНИЕ** Выпуклые дифференцируемые функции, такие как наш пример  $f(x) = x^2$ , — это «хороший» частный случай, где нет необходимости брать супремум и существует взаимно однозначное соответствие между функцией и ее преобразованием Лежандра. Выведем это из определений и простейших свойств. Для выпуклой дифференцируемой функции запишем уравнение касательной в точке  $f(x) = x^2$ :

$$f(x_0) = sx_0 + c. \quad (7.56)$$

Напомним, что мы хотели описать выпуклую функцию  $f(x)$  через ее градиент  $\nabla_x f(x)$ , и что  $s = \nabla_x f(x_0)$ . Выразим  $-c$ :

$$-c = sx_0 - f(x_0). \quad (7.57)$$

Заметим, что  $-c$  зависит от  $x_0$  и, следовательно, от  $s$ , поэтому мы можем рассматривать ее как функцию от  $s$ , которую мы обозначим

$$f^*(s) := sx_0 - f(x_0). \quad (7.58)$$

Сравнивая (7.58) с определением 7.4, видим, что (7.58) — частный случай определения (где нам не нужен супремум). ◆

Сопряженная функция обладает удобными нам свойствами, например для выпуклых функций повторное применение преобразования Лежандра возвращает нас к исходной функции. Аналогично тому, что наклон  $f(x)$  равен  $s$ , наклон  $f^*(s)$  равен  $x$ . Два следующих примера показывают частые применения выпукло-сопряженных функций в машинном обучении.

### Пример 7.7 (выпуклое сопряжение)

Покажем применение выпуклого сопряжения на примере квадратичной функции

$$f(\mathbf{y}) = \frac{\lambda}{2} \mathbf{y}^\top \mathbf{K}^{-1} \mathbf{y}, \quad (7.59)$$

заданной положительно определенной матрицей  $\mathbf{K} \in \mathbb{R}^{n \times n}$ . Обозначим переменную в исходной задаче как  $\mathbf{y} \in \mathbb{R}^n$ , а в двойственной — как  $\alpha \in \mathbb{R}^n$ .

<sup>1</sup> Классическое преобразование Лежандра определено для выпуклых дифференцируемых функций на  $\mathbb{R}^D$ .

Применив определение 7.4, получим функцию

$$f^*(\alpha) = \sup_{y \in \mathbb{R}^n} \langle y, \alpha \rangle = -\frac{\lambda}{2} y^T K^{-1} y. \quad (7.60)$$

Так как эта функция дифференцируема, мы можем найти максимум, взяв производную по  $y$  и приравняв ее нулю.

$$\frac{\partial \left[ \langle y, \alpha \rangle - \frac{\lambda}{2} y^T K^{-1} y \right]}{\partial y} = (\alpha - \lambda K^{-1} y)^T. \quad (7.61)$$

Таким образом, когда градиент равен нулю,  $y = (1/\lambda)K\alpha$ . Подставим это выражение в (7.60) и получим

$$f^*(\alpha) = \frac{1}{\lambda} \alpha^T K \alpha - \frac{\lambda}{2} \left( \frac{1}{\lambda} K \alpha \right)^T K^{-1} \left( \frac{1}{\lambda} K \alpha \right) = \frac{1}{2\lambda} \alpha^T K \alpha. \quad (7.62)$$

### Пример 7.8

В машинном обучении мы часто встречаемся с суммами функций, например целевая функция на обучающей выборке включает сумму потерь на каждом ее элементе. Сейчас мы найдем выпукло сопряженную функцию к сумме потерь  $\ell(t)$ , где  $\ell(t): \mathbb{R} \rightarrow \mathbb{R}$ . Этот пример также покажет применение сопряжения к многомерному случаю. Пусть  $\mathcal{L}(t) = \sum_{i=1}^n \ell_i(t_i)$ . Тогда

$$\mathcal{L}^*(z) = \sup_{t \in \mathbb{R}^n} \langle z, t \rangle - \sum_{i=1}^n \ell_i(t_i) = \quad (7.63a)$$

$$= \sup_{t \in \mathbb{R}^n} \sum_{i=1}^n z_i t_i - \ell_i(t_i) \text{ по определению скалярного произведения} \quad (7.63b)$$

$$= \sum_{i=1}^n \sup_{t \in \mathbb{R}^n} z_i t_i - \ell_i(t_i) = \quad (7.63c)$$

$$= \sum_{i=1}^n \ell_i^*(z_i) \text{ по определению сопряжения.} \quad (7.63d)$$

Вспомним, что в разделе 7.2 мы получили двойственную задачу оптимизации с помощью множителей Лагранжа. Далее, для задач выпуклой оптимизации имеется сильная двойственность, то есть решения исходной и двойственной задач равны. Описанное сейчас преобразование Лежандра — Фенхеля также можно использовать для получения двойственной задачи. Более того, если

функция выпукла и дифференцируема, супремум будет единственным. Чтобы подробнее изучить связь двух подходов, рассмотрим задачу выпуклой оптимизации с линейными ограничениями.

### Пример 7.9

Пусть  $f(\mathbf{y})$  и  $g(\mathbf{x})$  — выпуклые функции,  $\mathbf{A}$  — вещественная матрица подходящей размерности, такая что  $\mathbf{Ax} = \mathbf{y}$ . Тогда

$$\min_{\mathbf{x}} f(\mathbf{Ax}) + g(\mathbf{x}) = \min_{\mathbf{Ax} = \mathbf{y}} f(\mathbf{y}) + g(\mathbf{x}). \quad (7.64)$$

Вводим множитель Лагранжа  $\mathbf{u}$  для ограничений  $\mathbf{Ax} = \mathbf{y}$ ,

$$\min_{\mathbf{Ax} = \mathbf{y}} f(\mathbf{y}) + g(\mathbf{x}) = \min_{\mathbf{x}, \mathbf{y}} \max_{\mathbf{u}} f(\mathbf{y}) + g(\mathbf{x}) + (\mathbf{Ax} - \mathbf{y})^T \mathbf{u} = \quad (7.65a)$$

$$= \max_{\mathbf{u}} \min_{\mathbf{x}, \mathbf{y}} f(\mathbf{y}) + g(\mathbf{x}) + (\mathbf{Ax} - \mathbf{y})^T \mathbf{u}, \quad (7.65b)$$

где на последнем шаге мы можем поменять местами  $\max$  и  $\min$ , благодаря тому что  $f(\mathbf{y})$  и  $g(\mathbf{x})$  выпуклы. Перегруппировкой слагаемых получаем

$$\max_{\mathbf{u}} \min_{\mathbf{x}, \mathbf{y}} f(\mathbf{y}) + g(\mathbf{x}) + (\mathbf{Ax} - \mathbf{y})^T \mathbf{u} = \quad (7.66a)$$

$$= \max_{\mathbf{u}} \left[ \min_{\mathbf{y}} -\mathbf{y}^T \mathbf{u} + f(\mathbf{y}) \right] + \left[ \min_{\mathbf{x}} (\mathbf{Ax})^T \mathbf{u} + g(\mathbf{x}) \right] = \quad (7.66b)$$

$$= \max_{\mathbf{u}} \left[ \min_{\mathbf{y}} -\mathbf{y}^T \mathbf{u} + f(\mathbf{y}) \right] + \left[ \min_{\mathbf{x}} \mathbf{x}^T \mathbf{A}^T \mathbf{u} + g(\mathbf{x}) \right]. \quad (7.66c)$$

Вспомнив определение 7.4 выпуклого сопряжения и симметричность скалярного произведения<sup>1</sup>, получим

$$\max_{\mathbf{u}} \left[ \min_{\mathbf{y}} -\mathbf{y}^T \mathbf{u} + f(\mathbf{y}) \right] + \left[ \min_{\mathbf{x}} \mathbf{x}^T \mathbf{A}^T \mathbf{u} + g(\mathbf{x}) \right] = \quad (7.67a)$$

$$= \max_{\mathbf{u}} -f^*(\mathbf{u}) - g^*(-\mathbf{A}^T \mathbf{u}). \quad (7.67b)$$

Таким образом, мы показали, что

$$\min_{\mathbf{x}} f(\mathbf{Ax}) + g(\mathbf{x}) = \max_{\mathbf{u}} -f^*(\mathbf{u}) - g^*(-\mathbf{A}^T \mathbf{u}). \quad (7.68)$$

Сопряжение по Лежандру — Фенхелю оказывается весьма полезным для задач машинного обучения, которые могут быть выражены в виде задач выпуклой

<sup>1</sup> В общем случае для скалярного произведения  $\mathbf{A}^T$  заменяется на  $\mathbf{A}^*$ .

оптимизации. В частности, для выпуклых функций потерь, применяющихся по отдельности к каждому объекту, сопряженная функция потерь позволяет перейти к двойственной задаче.

## 7.4. ДЛЯ ДАЛЬНЕЙШЕГО ЧТЕНИЯ

Непрерывная оптимизация является активно развивающейся областью, и мы не претендуем на полное изложение недавних достижений.

У градиентного спуска есть два основных недостатка, каждому из которых посвящено некоторое количество литературы. Первая проблема состоит в том, что градиентный спуск — алгоритм первого порядка и не использует информацию о кривизне поверхности. В «оврагах» направление градиента перпендикулярно нужному. Идею моментов можно обобщить и получить ускоренные методы (Nesterov, 2018). Метод сопряженных градиентов избегает проблем обычного градиентного спуска, учитывая предшествующие направления (Shewchuk, 1994). Методы второго порядка, такие как метод Ньютона, используют матрицу Гессе для получения информации о кривизне. Рассмотрение кривизны поверхности приводит к разнообразным методам выбора шага и идеям, аналогичным моментам (Goh, 2017; Bottou et al., 2018). Квазиньютоновы методы, такие как L-BFGS, пытаются более эффективно с вычислительной точки зрения приближать матрицу Гессе (Nocedal and Wright, 2006). Недавно возник интерес к другому выбору метрик для вычисления направления градиента, приведя к таким подходам, как зеркальный спуск (Beck and Teboulle, 2003) и естественный градиент (Toussaint, 2012).

Второй проблемой является работа с недифференцируемыми функциями. Градиентные методы не всегда применимы, когда у функции имеются особенности. В этих случаях можно использовать субградиентные методы (Shor, 1985). Чтобы узнать больше об алгоритмах оптимизации недифференцируемых функций, советуем читать книгу Bertsekas (1999). Имеется огромный массив литературы о разных подходах к численному решению задач непрерывной оптимизации, в том числе об алгоритмах для задач с ограничениями. Хорошей отправной точкой могут служить книги Luenberger (1969) и Bonnans et al. (2006). Недавно обзор по непрерывной оптимизации был представлен в Bubeck (2015).

В современных приложениях машинного обучения размер входных данных часто не позволяет провести градиентный спуск по батчам, поэтому сейчас стандартным выбором для больших объемов данных является стохастический градиентный спуск. К недавним обзорам литературы относятся Hazan (2015) и Bottou et al. (2018).

По теме двойственности и выпуклой оптимизации советуем книгу Boyd and Vandenberghe (2004), а также прилагающиеся к ней лекции и слайды в интернете. Более математический подход можно найти в Bertsekas (2009), а недавно вышла книга Nesterov (2018), автор которой — один из самых выдающихся исследователей в области оптимизации. Выпуклая оптимизация основана на выпуклом анализе, и заинтересованному в математических основах читателю советуем обратиться к Rockafellar (1970), Hiriart-Urruty and Lemaréchal (2001), и Borwein and Lewis (2006). В данных книгах освещается и тема преобразований Лежандра — Фенхеля, но более подходящее для начинающего изложение представлено в Zia et al. (2009). Роль преобразований Лежандра — Фенхеля в анализе алгоритмов выпуклой оптимизации рассмотрена в Polyak (2016)<sup>1</sup>.

## УПРАЖНЕНИЯ

**7.1.** Рассмотрим одномерную функцию

$$f(x) = x^3 + 6x^2 - 3x - 5.$$

Найдите ее стационарные точки и укажите, являются ли они максимальной, минимальной или седловой точками.

**7.2.** Рассмотрим уравнение обновления для стохастического градиентного спуска (7.15). Запишите обновление, если мы используем размер мини-пакета, равный единице.

**7.3.** Верны ли следующие утверждения или нет:

- a. Пересечение любых двух выпуклых множеств выпукло.
- b. Объединение любых двух выпуклых множеств выпукло.
- c. Разница одного выпуклого множества  $A$  и второго выпуклого множества  $B$  есть выпуклое множество.

**7.4.** Верны ли следующие утверждения или нет:

- a. Сумма любых двух выпуклых функций выпукла.
- b. Разность любых двух выпуклых функций выпукла.
- c. Произведение любых двух выпуклых функций выпукло.
- d. Максимум любых двух выпуклых функций выпуклый.

---

<sup>1</sup> Блог Хьюго Гонсалвеса (Hugo Gonçalves) — также полезный ресурс для ознакомления с преобразованиями Лежандра — Фенхеля: <https://tinyurl.com/ydaal7hj>.

**7.5.** Выразите следующую задачу оптимизации в виде стандартной линейной программы в матричной форме

$$\max_{x \in \mathbb{R}^2, \xi \in \mathbb{R}} p^T x + \xi$$

с учетом ограничений  $\xi \geq 0$ ,  $x_0 \leq 0$  и  $x_1 \leq 3$ .

**7.6.** Рассмотрим линейную программу, показанную на рис. 7.9.

$$\begin{aligned} \min_{x \in \mathbb{R}^2} & -\begin{bmatrix} 5 \\ 3 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \text{при } & \begin{bmatrix} 2 & 2 \\ 2 & -4 \\ -2 & 1 \\ 0 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 33 \\ 8 \\ 5 \\ -1 \\ 8 \end{bmatrix}. \end{aligned}$$

Выполните двойственную линейную программу, используя двойственность Лагранжа.

**7.7.** Рассмотрим квадратичную программу, показанную на рис. 7.4,

$$\begin{aligned} \min_{x \in \mathbb{R}^2} & \frac{1}{2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 5 \\ 3 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \text{при } & \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}. \end{aligned}$$

Выполните двойственную квадратичную программу, используя двойственность Лагранжа.

**7.8.** Рассмотрим следующую задачу выпуклой оптимизации

$$\begin{aligned} \min_{w \in \mathbb{R}^D} & \frac{1}{2} w^T w \\ \text{при } & w^T x \geq 1. \end{aligned}$$

Выполните двойственный лагранжиан, введя множитель Лагранжа  $\lambda$ .

**7.9.** Рассмотрим отрицательную энтропию  $x \in \mathbb{R}^D$ ,

$$f(x) = \sum_{d=1}^D x_d \log x_d.$$

Получите выпуклую сопряженную функцию  $f^*(s)$ , предполагая стандартное скалярное произведение.

*Подсказка:* градиент соответствующей функции установите на ноль.

**7.10.** Рассмотрим функцию

$$f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^\top \mathbf{A} \mathbf{x} + \mathbf{b}^\top \mathbf{x} + c,$$

где  $\mathbf{A}$  строго положительно определена, что означает, что она обратима. Вывести выпуклое сопряжение  $f(\mathbf{x})$ .

*Подсказка:* градиент соответствующей функции установите на ноль.

**7.11.** Кусочно-линейная функция потерь (которая является потерей, используемой методом опорных векторов) определяется как

$$L(\alpha) = \max\{0, 1 - \alpha\}.$$

Если мы заинтересованы в применении градиентных методов, таких как L-BFGS, и не хотим прибегать к субградиентным методам, нам необходимо сгладить излом в кусочно-линейной функции потерь. Вычислите выпуклое сопряжение кусочно-линейной функции потерь  $L^*(\beta)$ , где  $\beta$  — двойственная переменная. Добавьте два проксимальных члена и вычислите сопряжение полученной функции

$$L^*(\beta) + \frac{\gamma}{2} \beta^2,$$

где  $\gamma$  — заданный гиперпараметр.

## ЧАСТЬ II

Главные задачи  
машинного обучения

# 8

## ○ сочетании модели и данных

В первой части книги мы познакомились с математикой, лежащей в основе многих методов машинного обучения. Надеемся, что читатель получил базовые представления о том математическом аппарате, которым мы будем пользоваться при описании и обсуждении МО. Во второй части книги мы познакомимся с четырьмя столпами машинного обучения:

- регрессия (глава 9);
- снижение размерности (глава 10);
- оценка плотности (глава 11);
- классификация (глава 12).

Основная цель этой части книги — проиллюстрировать, как математические концепции, показанные в первой части, могут использоваться при проектировании алгоритмов МО, которые, в свою очередь, применяются для решения задач в пределах заданной предметной области. Мы не собираемся рассказывать о продвинутых концепциях МО, а, напротив, предлагаем практические методы, которые помогут читателю воспользоваться знаниями, приобретенными в первой части книги. Также эти главы проложат путь к более обширной литературе по машинному обучению для тех читателей, что уже знакомы с математикой.

### 8.1. ДАННЫЕ, МОДЕЛИ И ОБУЧЕНИЕ

Здесь стоит приостановиться и задуматься о том, задачи какого рода должен решать алгоритм машинного обучения. Как говорилось в главе 1, в системе МО есть три основные составляющие: данные, модели и обучение. Основной вопрос МО: «Что мы понимаем под хорошими моделями?» В данном значении слово

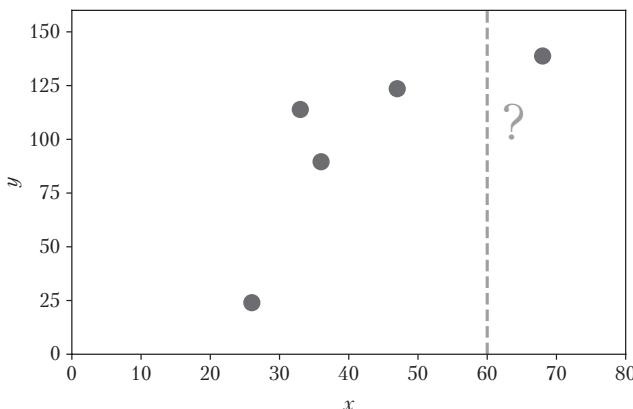
«модель» имеет много тонкостей, и мы неоднократно обратимся к ним в рамках этой главы. Также не вполне очевидно, как объективно оценить понятие «хороший». Один из руководящих принципов МО — в том, что хорошие модели должны успешно работать на еще не известных данных. Для этого требуется определить ряд метрик производительности, например точность (accuracy) или удаленность от истинного значения, а также найти способы достижения хороших результатов в соответствии с этими метриками. В этой главе рассмотрен ряд необходимых составляющих математического и статистического аппарата, которые обычно используются при рассуждении о моделях машинного обучения. Обсудив их, мы кратко очертиим современные наилучшие практики обучения модели, с опорой на которые предиктор должен показывать хорошие результаты на еще не известных данных.

Как упоминалось в главе 1, выражение «алгоритм машинного обучения» употребляется в контексте двух основных операций: обучение и прогнозирование. Мы опишем их в данной главе, а также расскажем, как выбирать модель. В разделе 8.2 мы познакомимся с фреймворком для минимизации эмпирического риска, в разделе 8.3 объясним принцип максимального правдоподобия, а в разделе 8.4 поговорим о вероятностных моделях. В разделе 8.5 мы кратко затронем графический язык для описания вероятностных моделей и, наконец, в разделе 8.6 поговорим о выборе модели. В оставшейся части этого раздела раскроем основные понятия МО: данные, модели и обучение.

### 8.1.1. Данные как векторы

Предполагается, что данные могут считываться компьютером и адекватно представляться в числовом формате. Также мы исходим из того, что данные хранятся в табличном виде (рис. 8.1), где каждая строка представляет конкретный экземпляр или пример, а каждый столбец — конкретный признак<sup>1</sup>. В последние годы МО применяется на данных самого разного типа, и не всегда очевидно, как представить некоторые из них в табличном числовом формате. Среди таких сложных примеров — геномные последовательности, текстовое и графическое содержимое веб-страницы, графики из социальных сетей. Мы не затрагиваем важные и нетривиальные вопросы выявления хороших признаков. Многие из этих задач зависят от опыта в предметной области и требуют тщательного проектирования; в последние годы они обобщаются под термином «наука о данных» (data science) (Stray, 2016; Adhikari and DeNero, 2018).

<sup>1</sup> Предполагается, что данные должны быть в аккуратном формате (Wickham, 2014; Codd, 1990).



**Рис. 8.1.** Простой пример данных для линейной регрессии. Обучающие данные в парах  $(x_n, y_n)$  из двух крайних правых столбцов в табл. 8.2. Нас интересует зарплата человека в возрасте 60 лет ( $x = 60$ ), показанная в виде вертикальной пунктирной линии красного цвета. Эти данные не входят в обучающий датасет

Хотя данные и представлены у нас в табличном формате, все равно остаются варианты, какое именно числовое представление для них выбрать. Например, в табл. 8.1 столбец «пол» (категориальная переменная) может быть преобразован в числа 0 (для «М») и 1 (для «Ж»). Другой вариант — выразить содержимое этого столбца при помощи чисел +1 и -1 соответственно (как показано в табл. 8.2). Кроме того, зачастую важно использовать при создании представления данные, относящиеся к предметной области: например, знать, что академические степени присваиваются в порядке «бакалавр — магистр — PhD» или что предоставленный почтовый код — это не просто последовательность символов, а обозначение одного из районов Лондона. В табл. 8.2 находятся данные из табл. 8.1, преобразованные в числовой формат. Каждый почтовый код представлен в виде двух числовых значений: широты и долготы. Любые числовые данные, которые потенциально поддаются непосредственному считыванию алгоритмом МО, нужно внимательно рассмотреть на предмет единиц измерения, масштаба и ограничений. Без дополнительной информации придется масштабировать все столбцы датасета таким образом, чтобы их эмпирическое среднее было равно 0, а эмпирическая дисперсия — 1. В контексте этой книги мы предполагаем, что эксперт уже правильно преобразовал данные, то есть что каждый ввод  $\mathbf{x}_n$  является  $D$ -мерным вектором вещественных чисел, которые называются *признаками*, *атрибутами* или *ковариантами*. Будем считать, что датасет представлен в такой форме, как в табл. 8.2. Обратите внимание: в новом числовом представлении мы отбросили столбец «Имя» из табл. 8.1. Это желательно по двум причинам: (1) мы не предполагаем, что идентификатор (Имя) будет информативен в контексте задачи машинного обучения,

и (2), возможно, мы захотим анонимизировать данные, чтобы обеспечить приватность информации о сотрудниках.

**Таблица 8.1.** Пример данных из фиктивной базы данных кадрового отдела, представленных не в числовом формате

Имя	Пол	Степень	Почтовый код	Возраст	Ежегодная зарплата
Адитья	М	Магистр	W21BG	36	89 563
Боб	М	PhD	EC1A1BA	47	123 543
Хлои	Ж	Бакалавр экономики	SW1A1BH	26	23 989
Дайсукэ	М	Бакалавр	SE207AT	68	138 769
Элизабет	Ж	МБА	SE10AA	33	113 888

**Таблица 8.2.** Пример данных из фиктивной базы данных кадрового отдела (см. табл. 8.1), преобразованных в числовой формат

ID пола	Степень	Широта (в десятичных градусах)	Долгота (в десятичных градусах)	Возраст	Ежегодная зарплата (в тысячах)
-1	2	51,5073	0.1290	36	89,563
-1	3	51,5074	0.1275	47	123,543
+1	1	51,5071	0.1278	26	23,989
-1	1	51,5075	0.1281	68	138,769
+1	2	51,5074	0.1278	33	113,888

В этой части книги мы обозначим через  $N$  количество примеров во множестве данных, а индексировать примеры будем строчной буквой  $n = 1, \dots, N$ . Предполагается, что дан набор числовых данных в виде массива векторов (табл. 8.2). Каждая строка соответствует конкретному отдельному  $\mathbf{x}_n$ , часто именуемому в машинном обучении *примером* или *точкой данных*. Нижний индекс  $n$  означает, что речь идет об  $n$ -м примере из общего количества  $N$  примеров в наборе данных. В каждом столбце представлен конкретный признак интересующего нас примера, и мы индексируем признаки как  $d = 1, \dots, D$ . Напоминаем, что данные представляются в векторном виде, и это означает, что каждый пример (каждая точка данных) является  $D$ -мерным вектором. Ориентация таблицы традиционная, принятая в сообществе специалистов по базам данных, но в не-

которых алгоритмах МО (например, из главы 10) удобнее представлять примеры в виде векторов-столбцов.

Рассмотрим задачу прогнозирования годовой зарплаты в зависимости от возраста сотрудника, основываясь на данных из табл. 8.2. Это пример задачи на обучение с учителем, где у нас есть *метка*  $y_n$  (зарплата), связанная с каждым из примеров  $\mathbf{x}_n$  (возраст). Метку, такую как  $y_n$ , могут называть по-разному, в частности целевой или зависимой переменной, а также аннотацией. Датасет записывается как набор пар «пример – метка»  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n), \dots, (\mathbf{x}_N, y_N)\}$ . Таблица примеров  $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  часто конкатенируется и записывается как  $\mathbf{X} \in \mathbb{R}^{N \times D}$ . На рис. 8.1 показан набор данных, составленный из двух крайних правых столбцов табл. 8.2, где  $x$  = возраст и  $y$  = зарплата.

Мы воспользуемся концепциями, представленными в первой части книги, для формализации задач МО таким образом, как это сделано в предыдущем абзаце. Представляя данные в виде векторов  $\mathbf{x}_n$ , можно пользоваться концепциями линейной алгебры (описанными в главе 2). Во многих алгоритмах МО требуется дополнительно иметь возможность сравнивать два вектора. Как будет показано в главах 9 и 12, при вычислении сходства или расстояния между двумя примерами можно формализовать интуитивную догадку о том, что у примеров со схожими признаками должны быть схожие метки. Для сравнения двух векторов требуется построить геометрию (подробнее в главе 3), что позволит нам оптимизировать результирующую задачу обучения, воспользовавшись приемами из главы 7.

Поскольку у нас есть векторные представления данных, их можно обработать так, чтобы найти потенциально более качественные представления. Мы обсудим два способа сделать это: поиск для исходного вектора признаков приближений более низкой размерности, а также использование нелинейных комбинаций исходного вектора признаков более высокой размерности, чем оригинал. В главе 10 рассмотрим пример приближения с малой размерностью для исходного пространства данных путем нахождения главных компонент. Поиск главных компонент тесно связаны с концепциями собственного значения и сингулярного разложения, с которыми мы познакомились в главе 4. Для представления в высокой размерности мы рассмотрим явную *признаковую карту*  $\phi(\cdot)$ , позволяющую представлять значения  $\mathbf{x}_n$  при помощи варианта с более высокой размерностью  $\phi(\mathbf{x}_n)$ . Основная причина, по которой целесообразно прибегать к представлениям с более высокой размерностью, в том, что можно составлять новые признаки как нелинейные комбинации исходных признаков, что, в свою очередь, упрощает задачу обучения. Мы поговорим о признаковой карте в разделе 9.2, и в разделе 12.4 покажем, как эта карта признаков ведет к *ядру*. В последние годы методы глубокого обучения (Goodfellow et al., 2016) показали

многообещающие результаты в плане использования данных для самостоятельного извлечения новых полезных признаков и позволили добиться больших успехов в таких областях, как компьютерное зрение, распознавание речи и обработка естественного языка. В этой части книги мы не будем затрагивать нейронные сети, но напоминаем читателю, что в разделе 5.6 дается математическое описание обратного распространения ошибки — ключевой концепции в обучении нейронных сетей.

### 8.1.2. Модели как функции

Имея данные в подходящем векторном представлении, мы можем пустить их в дело, сформировав с их помощью предиктивную функцию (так называемый *предиктор*). В главе 1 у нас еще не было языка для точного определения моделей. Но теперь, воспользовавшись концепциями из первой части книги, мы можем объяснить, что такое «модель». В этой книге представлено два основных подхода: предиктор как функция и предиктор как вероятностная модель. Первый подход мы опишем здесь, а второй — в следующем подразделе.

Предиктор — это функция, которая, получая на ввод конкретный пример (в нашем случае — вектор признаков), дает вывод. Пока давайте рассмотрим случай, в котором вывод — единственное число, то есть речь идет о вещественночисленном скалярном выводе. Это можно записать как

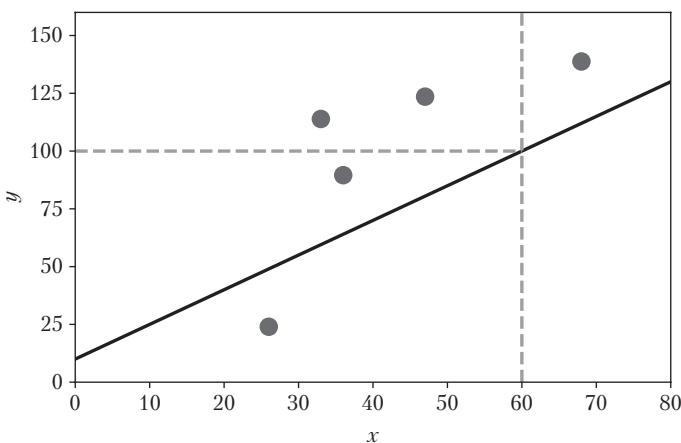
$$f: \mathbb{R}^D \rightarrow \mathbb{R}, \quad (8.1)$$

где входной вектор  $\mathbf{x}$  является  $D$ -мерным (имеет  $D$  признаков), а затем к нему применяется функция  $f$  (записывается как  $f(\mathbf{x})$ ), которая возвращает вещественное число. На рис. 8.2 проиллюстрирована возможная функция, позволяющая рассчитывать прогнозы для входных значений  $x$ .

В этой книге мы не рассматриваем общего случая для всех функций, для чего нам потребовался бы функциональный анализ. Вместо этого обсудим частный случай для линейных функций

$$f(\mathbf{x}) = \theta^T \mathbf{x} + \theta_0 \quad (8.2)$$

для неизвестных  $\theta$  и  $\theta_0$ . Такое ограничение означает, что материала глав 2 и 3 достаточно, чтобы точно сформулировать понятие предиктора для не вероятностного (в противовес вероятностному, описанному ниже) представления о машинном обучении. Линейные функции обеспечивают хороший баланс между универсальностью задач, решаемых с их помощью, и объемом математического бэкграунда, необходимого для работы с ними.



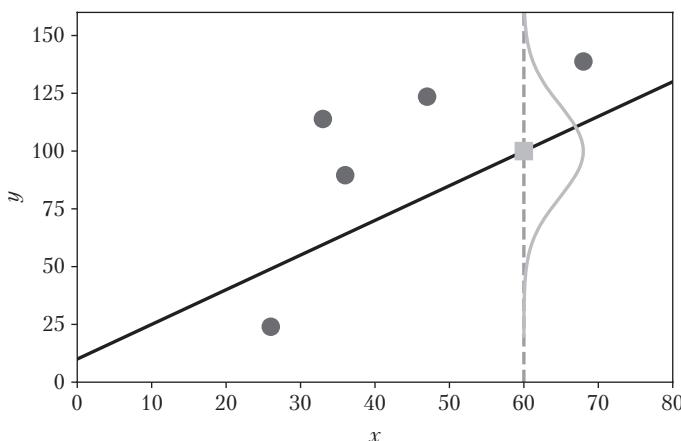
**Рис. 8.2.** Пример функции (черная сплошная диагональная прямая) и ее прогноз при  $x = 60$ , то есть  $f(60) = 100$

### 8.1.3. Модели как вероятностные распределения

Зачастую считается, что данные — это зашумленные проявления некоторого базового наблюдаемого эффекта. Мы надеемся, что, применив машинное обучение, сможем выделить сигнал из шума. Для этого нам требуется язык, который позволил бы количественно выразить эффект шума. Также мы обычно хотим иметь предикторы, при помощи которых выражается неопределенность того или иного рода, чтобы квантифицировать нашу уверенность в ценности прогноза или конкретной точки тестовых данных. Как было показано в главе 6, теория вероятностей обеспечивает язык для квантификации неопределенности. На рис. 8.3 прогностическая неопределенность функции показана в виде гауссова распределения.

Мы будем трактовать предиктор не как отдельную функцию, а как вероятностную модель, то есть модель, описывающую распределение возможных функций. В этой книге мы ограничимся рассмотрением частного случая распределений с конечномерными параметрами; это позволит нам описывать вероятностные модели без привлечения стохастических процессов и случайных мер. В данном частном случае можно считать вероятностные модели мультивариантными вероятностными распределениями, которые уже подходят для построения обширного класса моделей.

В разделе 8.4 мы расскажем, как использовать концепции из теории вероятностей (глава 6) для определения моделей МО, а в разделе 8.5 познакомим вас с графическим языком для компактного описания вероятностных моделей.



**Рис. 8.3.** Пример функции (черная сплошная диагональная прямая) и ее прогнозическая неопределенность при  $x = 60$  (показано как гауссиан)

#### 8.1.4. Обучение — это нахождение параметров

Цель обучения — найти модель и соответствующие ей параметры таким образом, чтобы результирующий предиктор хорошо работал на неизвестных данных. Существует три концептуально разных алгоритмических фазы, рассматриваемые при обсуждении МО:

1. Прогноз, или инференс.
2. Обучение, или оценка параметров.
3. Настройка гиперпараметров, или выбор модели.

Фаза прогноза имеет место, когда мы используем обученный предиктор на еще не известных тестовых данных. Иными словами, параметры и модель уже выбраны и зафиксированы, и предиктор применяется к новым векторам, представляющим новые входные точки данных. Как было показано в главе 1 и в разделе 8.1.3, в этой книге рассматриваются две школы машинного обучения, разница между которыми в том, чем является предиктор — функцией или вероятностной моделью. Когда у нас есть вероятностная модель (рассматриваемая далее в разделе 8.4), то фаза прогноза называется инференсом (inference).

**ПРИМЕЧАНИЕ** К сожалению, не существует общепринятых наименований для различных фаз алгоритмов. Слово «inference» иногда еще означает оценку параметров вероятностной модели, а также может означать прогнозирование для не вероятностных моделей. ♦

На этапе обучения или оценки параметров мы корректируем нашу предиктивную модель на основании обучающих данных, чтобы найти для них хорошие предикторы. Существует две основные стратегии, позволяющие это сделать: первая — нахождение наилучшего предиктора на основе некоторого качественного параметра (иногда этот процесс называется нахождением точечной оценки) и вторая — при помощи байесовского инференса. Нахождение точечной оценки может применяться с предикторами обоих типов, но для байесовского инференса требуются вероятностные модели.

Для не вероятностной модели мы придерживаемся принципа *минимизации эмпирического риска*, который будет описан в разделе 8.2. Минимизация эмпирического риска непосредственно подводит нас к задаче оптимизации для нахождения хороших параметров. В статистических моделях используется принцип *максимальной вероятности*, позволяющий найти хороший набор параметров (раздел 8.3). Дополнительно мы можем смоделировать неопределенность параметров при помощи вероятностной модели, которую подробнее рассмотрим в разделе 8.4.

Мы используем численные методы для нахождения параметров, которые «подходят» к данным, и большинство методов обучения можно рассматривать как методы поиска экстремума: ищется максимум для целевого параметра, например максимального правдоподобия. Для применения таких подходов, аналогичных поиску экстремума, используются градиенты, описанные в главе 5, и внедряются подходы с числовой оптимизацией<sup>1</sup>, описанные в главе 7.

Как упоминалось в главе 1, мы заинтересованы так обучить модель на основе имеющихся данных, чтобы в будущем она хорошо работала на новых. Недостаточно взять и подтянуть модель для хорошей работы на обучающем датасете; предиктор должен хорошо справляться с теми данными, которых ранее не видел. Симуляция поведения нашего предиктора на новых, будущих данных достигается при помощи *кросс-валидации*<sup>2</sup> (раздел 8.2.4). Как будет показано в этой главе, для достижения поставленной цели (хорошая работа на неизвестных данных) необходимо балансировать между хорошей подгонкой модели под обучающие данные и поиском «простых» объяснений феномена. Такой компромисс достигается при помощи регуляризации (раздел 8.2.3) или путем добавления априорного значения (раздел 8.3.2). В философии такой подход не считается ни индукцией, ни дедукцией, а называется *абдукцией*. Согласно Стэнфордской философской энциклопедии, абдукция — это процесс инференса наилучшего объяснения (Douven, 2017).

<sup>1</sup> Принято считать, что целью оптимизации является минимизация целевого параметра. Поэтому в задачах машинного обучения при определении целей зачастую ставится дополнительный знак «минус».

<sup>2</sup> Иначе говоря, перекрестной проверке. — Примеч. ред.

Часто приходится принимать при моделировании высокоуровневые решения о структуре предиктора, например сколько компонентов использовать или какой класс вероятностных распределений рассмотреть. Выбор количества компонентов — это пример *гиперпараметра*, и этот выбор может значительно повлиять на производительность модели. Проблема выбора одной из нескольких моделей называется *выбором модели* (model selection), о чём мы поговорим в разделе 8.6. В случае невероятностных моделей выбор зачастую делается при помощи *вложенной кросс-валидации*, которая описана в разделе 8.6.1, или при подборе гиперпараметров для нашей модели.

**ПРИМЕЧАНИЕ** Различие параметров и гиперпараметров является в некоторой степени произвольным, и в основном определяется разграничением между тем, что поддается численной оптимизации, и тем, что требует использования поисковых приемов. Другой подход к трактовке такой разницы — рассматривать параметры как явные показатели вероятностной модели, а гиперпараметры (параметры более высокого уровня) считать факторами, которые управляют распределением этих явных параметров. ◆

В следующих разделах мы рассмотрим три разновидности машинного обучения: минимизацию эмпирического риска (раздел 8.2), принцип максимального правдоподобия (раздел 8.3) и вероятностное моделирование (раздел 8.4).

## 8.2. МИНИМИЗАЦИЯ ЭМПИРИЧЕСКОГО РИСКА

Вооружившись всей необходимой математикой, мы в состоянии объяснить, что такое «учиться» с точки зрения машины. «Обучение» в составе машинного обучения сводится к оценке параметров на основании обучающего набора данных.

В этом разделе будет рассмотрен случай, в котором предиктор является функцией, а случай вероятностных моделей будет рассмотрен в разделе 8.3. Мы опишем идею минимизации эмпирического риска, которая изначально была популяризована при рассмотрении метода опорных векторов (о ней рассказано в главе 12). Однако общие принципы этой идеи широко применимы и позволяют поставить вопрос о сути обучения, не прибегая при этом к явному конструированию вероятностных моделей. Существует четыре основных варианта проектирования, которые мы подробно рассмотрим в следующих подразделах:

**Раздел 8.2.1.** Какое множество функций мы разрешаем принимать предиктору?

**Раздел 8.2.2.** Как мы измеряем эффективность работы предиктора на обучающих данных?

**Раздел 8.2.3.** Как конструировать предикторы на основании только обучающих данных, но таким образом, чтобы эти предикторы хорошо работали на новых тестовых данных?

**Раздел 8.2.4.** Какова процедура поиска в пространстве моделей?

### 8.2.1. Гипотеза класса функций

Допустим, нам дано  $N$  примеров  $\mathbf{x}_n \in \mathbb{R}^D$  и соответствующие скалярные метки  $y_n \in \mathbb{R}$ . Мы рассмотрим случай обучения с учителем, где получим пары  $(\mathbf{x}_1, y_1)$ , ...,  $(\mathbf{x}_N, y_N)$ . Имея эти данные, мы хотели бы оценить предиктор  $f(\cdot, \theta) : \mathbb{R}^D \rightarrow \mathbb{R}$ , параметризованный  $\theta$ . Мы надеемся, что удастся найти хороший параметр  $\theta^*$ , такой, что он позволит нам хорошо подогнать данные, чтобы

$$f(\mathbf{x}_n, \theta^*) \approx y_n \text{ для всех } n = 1, \dots, N. \quad (8.3)$$

В этом разделе мы используем нотацию  $\hat{y}_n = f(\mathbf{x}_n, \theta^*)$ , чтобы представить вывод предиктора.

**ПРИМЕЧАНИЕ** Для удобства представления мы опишем минимизацию эмпирического риска в терминах обучения с учителем (где у нас есть метки). Так становится проще определить класс гипотез и функцию потерь. В машинном обучении также распространена практика, при которой выбирается параметризованный класс функций, например аффинных функций. ♦

#### Пример 8.1

Введем задачу регрессионного анализа методом наименьших квадратов, чтобы проиллюстрировать минимизацию эмпирического риска. Более подробно о регрессии рассказано в главе 9. Когда метка  $y_n$  является вещественновозначной, для предикторов в качестве класса функций часто выбирают множество аффинных функций<sup>1</sup>. Мы выберем более компактную нотацию для аффинной функции, подцепив дополнительный единичный признак  $x^{(0)} = 1$  к  $\mathbf{x}_n$ , то есть  $\mathbf{x}_n = [1, x_n^{(1)}, x_n^{(2)}, \dots, x_n^{(D)}]^T$ . Соответственно, вектор параметров  $\theta = [\theta_0, \theta_1, \theta_2, \dots, \theta_D]^T$ , благодаря чему можно записать предиктор как линейную функцию

$$f(\mathbf{x}_n, \theta) = \theta^T \mathbf{x}_n. \quad (8.4)$$

---

<sup>1</sup> В машинном обучении аффинные функции, как правило, называют линейными.

Этот линейный предиктор эквивалентен аффинной модели

$$f(\mathbf{x}_n, \boldsymbol{\theta}) = \theta_0 + \sum_{d=1}^D \theta_d x_n^{(d)}. \quad (8.5)$$

Обратите внимание: предиктор принимает вектор признаков, представляющий одиночный пример  $\mathbf{x}_n$  как ввод и выдающий вещественночисленный вывод, то есть  $f : \mathbb{R}^{D+1} \rightarrow \mathbb{R}$ . На предыдущих рисунках в этой главе предиктор обозначался прямой линией, что означает: мы полагали его аффинной функцией.

Возможно, мы захотим рассмотреть в качестве предикторов нелинейные функции вместо линейных. Недавние достижения в области нейронных сетей обеспечивают эффективное вычисление более сложных классов нелинейных функций.

Имея класс функций, мы хотим поискать хороший предиктор. Теперь переходим ко второму компоненту минимизации эмпирического риска: измерим, насколько хорошо предиктор подходит под учебные данные.

### 8.2.2. Функция потерь для обучения

Рассмотрим метку  $y_n$  для конкретного примера и соответствующий ей прогноз  $\hat{y}_n$ , который мы делаем на основе  $\mathbf{x}_n$ . Чтобы определить, что такое «хорошо подстраиваться к данным», необходимо определить *функцию потерь*  $l(y_n, \hat{y}_n)$ , которая принимает в качестве ввода метку базовой истины и прогноз, а в ответ выдает неотрицательное число (именуемое «потерей»), отражающее, какова величина ошибки, допущенной нами при этом конкретном прогнозе. Наша цель при нахождении хорошего вектора параметров  $\boldsymbol{\theta}^*$  — минимизировать среднюю потерю<sup>1</sup> на множестве из  $N$  учебных примеров.

В машинном обучении часто делается допущение, что множество примеров  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$  *независимо и одинаково распределено*. Слово «независимо» (раздел 6.4.5) означает, что две точки данных  $(\mathbf{x}_i, y_i)$  и  $(\mathbf{x}_j, y_j)$  статистически не зависят друг от друга, то есть что их эмпирическое среднее — хорошая оценка математического ожидания (раздел 6.4.1). Так подразумевается, что эмпирическое среднее потери можно использовать с обучающими данными. Для заданного *обучающего датасета*  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$  вводится нотация матрицы-примера  $\mathbf{X} := [\mathbf{x}_1, \dots, \mathbf{x}_N]^T \in \mathbb{R}^{NxD}$  и вектор меток  $\mathbf{y} := [y_1, \dots, y_N]^T \in \mathbb{R}^N$ .

<sup>1</sup> Вместо термина «потеря» часто используется слово «ошибка».

При использовании этой матричной нотации средняя потеря определяется как

$$\mathbf{R}_{\text{emp}}(f, \mathbf{X}, \mathbf{y}) = \frac{1}{N} \sum_{n=1}^N \ell(y_n, \hat{y}_n), \quad (8.6)$$

где  $\hat{y}_n = f(\mathbf{x}_n, \theta^*)$ . Уравнение (8.6) называется *эмпирическим риском* и зависит от трех аргументов: предиктора  $f$  и данных  $\mathbf{X}, \mathbf{y}$ . Общая стратегия для обучения таким способом называется «*минимизация эмпирического риска*».

### Пример 8.2 (потеря наименьших квадратов)

Продолжая пример с регрессией наименьших квадратов, укажем, что измеряем цену ошибки при обучении при помощи квадратичной потери  $\ell(y_n, \hat{y}_n) = \ell(y_n - \hat{y}_n)^2$ . Мы хотим минимизировать эмпирический риск (8.6), представляющий собой среднее потерю в пересчете на все данные:

$$\min_{\theta \in \mathbb{R}^D} \frac{1}{N} \sum_{n=1}^N (y_n - f(\mathbf{x}_n, \theta))^2, \quad (8.7)$$

где мы подставили предиктор  $\hat{y}_n = f(\mathbf{x}_n, \theta)$ . Воспользовавшись выбранным нами линейным предиктором  $f(\mathbf{x}_n, \theta) = \theta^T \mathbf{x}_n$ , получаем оптимизированную задачу

$$\min_{\theta \in \mathbb{R}^D} \frac{1}{N} \sum_{n=1}^N (y_n - \theta^T \mathbf{x}_n)^2. \quad (8.8)$$

Это уравнение можно эквивалентно выразить в матричной форме:

$$\min_{\theta \in \mathbb{R}^D} \frac{1}{N} \| \mathbf{y} - \mathbf{X}\theta \|^2. \quad (8.9)$$

Такая ситуация известна под названием «*задача наименьших квадратов*». Для нее существует аналитическое решение в замкнутой форме, связанное с решением нормальных уравнений, — об этом речь пойдет в разделе 9.2.

Нас не интересует предиктор, который бы хорошо работал только на обучающих данных. Вместо этого нам нужен предиктор, который хорошо работает (демонстрирует низкий риск) на ранее не известных тестовых данных. Выражаясь более формально, нам нужно найти предиктор  $f$  (с фиксированными параметрами), который минимизирует *ожидаемый риск*

$$\mathbf{R}_{\text{true}}(f) = \mathbb{E}_{x,y} [\ell(y, f(\mathbf{x}))], \quad (8.10)$$

где  $y$  это метка, а  $f(\mathbf{x})$  — прогноз на основе примера  $\mathbf{x}$ . Нотация  $\mathbf{R}_{\text{true}}(f)$  означает, что именно таков истинный риск, если бы у нас был доступ к неограниченному объему данных. Ожидание касается (бесконечного) множества всех возможных данных и меток. Возникают два практических вопроса, связанных с нашим желанием минимизировать ожидаемый риск<sup>1</sup>, и их мы обсудим в двух следующих подразделах:

- Каким образом следует видоизменить нашу процедуру обучения, чтобы она хорошо обобщалась?
- Как оценивать ожидаемый риск от (конечных) данных?

**ПРИМЕЧАНИЕ** Многие задачи машинного обучения формулируются с сопутствующей им мерой точности, например с указанием точности прогноза или среднеквадратичной ошибкой. Мера, характеризующая производительность, может быть более сложной, чувствительной к ошибкам, либо охватывать подробности конкретного прикладного варианта. В принципе, проектирование функции потерь для минимизации эмпирического риска должно прямо соответствовать мере расчета производительности, которая указана для данной задачи МО. На практике зачастую возникает несоответствие между проектированием функции потерь и измерением производительности. Это может быть связано с такими факторами, как простота реализации или эффективность оптимизации. ◆

### 8.2.3. Регуляризация для борьбы с переобучением

В этом разделе описан материал, дополняющий минимизацию эмпирического риска и обеспечивающий его хорошее обобщение (путем приблизительной минимизации ожидаемого риска). Напомним, с какой целью обучается предиктор при машинном обучении: требуется, чтобы он хорошо работал на ранее не известных данных, то есть предиктор должен хорошо обобщаться. Мы имитируем такие новые данные, придерживаясь до поры до времени часть датасета. Такой придерживаемый набор данных называется *тестовым*. При наличии достаточно разнообразного класса функций для предиктора  $f$  мы, в сущности, можем запоминать обучающие данные для достижения нулевого эмпирического риска<sup>2</sup>. При том, насколько хорошо это помогает минимизировать потери (и, следовательно, риск) на обучающих данных, не приходится ожидать, что предиктор будет хорошо обобщаться на ранее не известных данных. На практике мы располагаем только конечным набором данных, поэтому разбиваем наше множество

<sup>1</sup> Ожидаемый риск также часто именуется «математическим ожиданием».

<sup>2</sup> Даже знание лишь о производительности предиктора на тестовом множестве — это утечка информации (Blum and Hardt, 2015).

данных на две выборки — обучающую и тестовую. Обучающий набор используется для коррекции модели, а тестовый (не виденный алгоритмом в ходе обучения) применяется для оценки качества обобщающей способности. Важно, чтобы пользователь не возвращался на новый круг обучения, после того как алгоритм проанализировал тестовый набор. При помощи нижних индексов *train* и *test* мы обозначаем обучающий и тестовый набор данных соответственно. В разделе 8.2.4 мы вернемся к этой идеи, воспользовавшись конечным датасетом для оценки ожидаемого риска.

Оказывается, что минимизация эмпирического риска может приводить к *переобучению*, то есть предиктор слишком сильно подстраивается под обучающие данные и плохо обобщает новые (Mitchell, 1997). Этот общий феномен, связанный с достижением очень низких средних потерь на обучающем наборе при больших средних потерях на тестовом, обычно случается, когда у нас мало данных, но сложный класс гипотез. Для конкретного предиктора  $f$  (с фиксированными параметрами) феномен переобучения возникает, когда оценка риска по обучающим данным  $\mathbf{R}_{\text{emp}}(f, \mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$  приводит к недооценке ожидаемого риска  $\mathbf{R}_{\text{true}}(f)$ . Поскольку мы оцениваем ожидаемый риск  $\mathbf{R}_{\text{true}}(f)$ , применяя эмпирический риск на тестовом наборе  $\mathbf{R}_{\text{emp}}(f, \mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}})$ , тот случай, в котором тестовый риск оказывается гораздо выше обучающего, явно свидетельствует о переобучении. Мы вернемся к этой идеи в разделе 8.3.3.

Следовательно, нам нужно каким-то образом сдвинуть поиск минимизатора эмпирического риска, введя штрафной показатель, который мешал бы оптимизатору возвращать чрезмерно гибкий предиктор. В машинном обучении такой штрафной показатель называется *регуляризацией*. Регуляризация — это своеобразный компромисс между точным решением при минимизации эмпирического риска и учетом размера или сложности решения.

### Пример 8.3 (регуляризация в задаче наименьших квадратов)

Регуляризация — это подход, предотвращающий сложные или экстремальные решения задачи оптимизации. Простейшая стратегия регуляризации — заменить задачу наименьших квадратов

$$\min_{\theta} \frac{1}{N} \|\mathbf{y} - \mathbf{X}\theta\|^2 \quad (8.11)$$

из предыдущего примера «регуляризованной» задачей, в которую добавлен штрафной показатель, включающий только  $\theta$ :

$$\min_{\theta} \frac{1}{N} \|\mathbf{y} - \mathbf{X}\theta\|^2 + \lambda \|\theta\|^2. \quad (8.12)$$

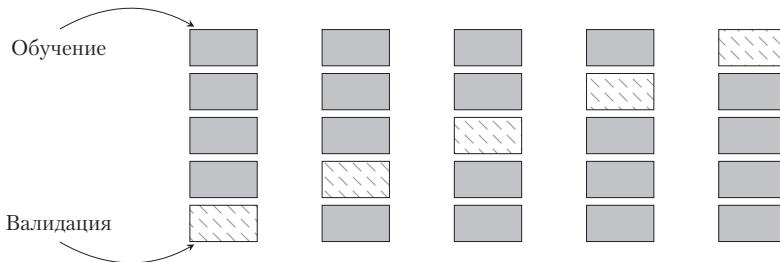
Дополнительный член  $\|\theta\|^2$  называется *регуляризатором*, а параметр  $\lambda$  — *параметром регуляризации*. Параметр регуляризации позволяет достичь компромисса при минимизации потерь на обучающем множестве и подборе выраженности параметров  $\theta$ . Часто бывает, что при переобучении получаемые значения параметров оказываются слишком велики (Bishop 2006).

Параметр регуляризации иногда называется *штрафным*; он сдвигает вектор  $\theta$  ближе к началу координат. Идея регуляризации также присутствует в вероятностных моделях как априорная вероятность параметров. Как вы помните из раздела 6.6, чтобы апостериорное распределение повторяло по форме априорное распределение, априорные данные и вероятность должны быть сопряжены. Мы вернемся к этой идеи в разделе 8.3.2. В главе 12 мы убедимся, что идея регуляризатора эквивалентна идеи широкого зазора.

### 8.2.4. Кросс-валидация для оценки производительности обобщения

В предыдущем разделе мы упомянули, что измеряем ошибку обобщения, оценивая ее путем применения предиктора к тестовым данным. Такие данные также иногда называются *валидационным набором*. Валидационный набор — это некое подмножество доступных обучающих данных, которое мы приберегаем до поры до времени. Практическая проблема, возникающая при таком подходе, заключается в ограниченности объема доступных данных, а в идеале нам нужно для обучения модели как можно больше данных. Для этого нам потребуется держать валидационный набор  $\mathcal{V}$  компактным, что приведет к зашумленной оценке (с большим разбросом) прогностической способности. Одно из решений для достижения этих противоречивых целей (большое обучающее множество, большое валидационное множество) — использовать *кросс-валидацию*.  $K$ -кратная кросс-валидация фактически сегментирует данные на  $K$  фрагментов,  $K - 1$  из которых образуют обучающее множество  $\mathcal{R}$ , а последний фрагмент служит валидационным множеством  $\mathcal{V}$  (в соответствии с идеей, изложенной выше). При кросс-валидации перебираются (в идеале) все комбинации присваивания фрагментов к  $\mathcal{R}$  и  $\mathcal{V}$ , см. рис. 8.4. Эта процедура повторяется для всех  $K$  вариантов валидационной выборки, и производительность модели на  $K$  прогонах усредняется.

Мы делим наш датасет на два множества  $\mathcal{D} = \mathcal{R} \cup \mathcal{V}$ , так чтобы они не пересекались ( $\mathcal{R} \cap \mathcal{V} = \emptyset$ ), где  $\mathcal{V}$  — валидационный набор, а обучаем мы нашу модель на  $\mathcal{R}$ . После обучения мы оцениваем производительность предиктора  $f$  на валидационном множестве  $\mathcal{V}$ , вычисляя, например, среднеквадратичную ошибку



**Рис. 8.4.**  $K$ -кратная кросс-валидация. Датасет делится на  $K = 5$  фрагментов,  $K - 1$  из которых служат обучающим набором (сплошная заливка), а один — валидационным (косая штриховка)

(RMSE, root mean square error) обученной модели на этой валидационной выборке. Точнее, для каждого сегмента  $k$  обучающие данные  $\mathcal{R}^{(k)}$  дают предиктор  $f^{(k)}$ , который затем применяется к валидационному множеству  $\mathcal{V}^{(k)}$  для вычисления эмпирического риска  $R(f^{(k)}, \mathcal{V}^{(k)})$ . Мы проходим через все возможные сегменты валидационных и обучающих выборок и вычисляем среднюю ошибку обобщения для предиктора. Кросс-валидация аппроксимирует ожидаемую ошибку обобщения.

$$\mathbb{E}_{\mathcal{V}}[R(f, \mathcal{V})] \approx \frac{1}{K} \sum_{k=1}^K R(f^{(k)}, \mathcal{V}^{(k)}), \quad (8.13)$$

где  $R(f^{(k)}, \mathcal{V}^{(k)})$  — это риск (например, RMSE) на валидационном множестве  $\mathcal{V}^{(k)}$  для предиктора  $f^{(k)}$ . Аппроксимация происходит по двум причинам: во-первых, из-за конечного размера обучающего набора данных, что дает неоптимальный вариант  $f^{(k)}$ ; во-вторых, из-за конечного размера валидационного набора, что приводит к неточной оценке риска  $R(f^{(k)}, \mathcal{V}^{(k)})$ . Потенциальный недостаток  $K$ -кратной кросс-валидации заключается в вычислительных расходах на обучение модели  $K$  раз, что может быть обременительно, если операция обучения ресурсозатратна. На практике зачастую недостаточно рассмотреть только лишь прямые параметры. Так, приходится исследовать параметры сложности (например, множество параметров регуляризации), которые могут не являться прямыми параметрами модели. Оценка качества модели в зависимости от этих гиперпараметров может выльиться в несколько обучающих прогонов, количество которых будет расти экспоненциально с ростом числа параметров модели. Для поиска хороших гиперпараметров можно использовать вложенную кросс-валидацию (раздел 8.6.1).

Но кросс-валидация является *чрезвычайно параллельной задачей*, то есть ценой минимальных усилий такую задачу можно распараллелить на отдель-

ные подзадачи. При наличии достаточных вычислительных ресурсов (например, оборудования для облачных вычислений, серверных ферм) кросс-валидация занимает не больше времени, чем однократная оценка производительности.

В этом разделе мы убедились, что минимизация эмпирического риска основана на следующих концепциях: класс функций, описывающих гипотезы, функция потерь, регуляризация. В разделе 8.3 будет рассмотрено, какой эффект достигается, если использовать вероятностное распределение вместо функций потерь и регуляризации.

### 8.2.5. Дальнейшее чтение

В силу того что исходные разработки минимизации эмпирического риска (Vapnik, 1998) излагались тяжеловесным теоретическим языком, многие последующие разработки были теоретическими. Эта область исследований называется *статистическая теория обучения* (Vapnik, 1999; Evgeniou et al., 2000; Hastie et al., 2001; von Luxburg and Schölkopf, 2011). Одну из свежих книг по машинному обучению, построенную на этой теоретической основе и посвященную разработке эффективных алгоритмов обучения, написали Шалев-Швartz и Бен-Дэвид (Shalev-Shwartz, Ben-David, 2014).

Концепция регуляризации коренится в решении плохо сформулированных обратных задач (Neumaier, 1998). Подход, представленный здесь, называется *метод регуляризации Тихонова*. Также существует весьма схожая ограниченная версия этого метода — регуляризация Иванова. Регуляризация Тихонова обладает глубокими взаимосвязями с компромиссной частотой исключений и подбором признаков (Bühlmann and Van De Geer, 2011). В качестве альтернативы кросс-валидации используется метод бутстрепа и складного ножа (Hall, 1992; Efron and Tibshirani, 1993; Davidson and Hinkley, 1997).

Неверно трактовать минимизацию эмпирического риска (раздел 8.2) как метод, полностью «лишенный вероятностей». Существует основополагающее неизвестное вероятностное распределение  $p(\mathbf{x}, y)$ , которое управляет генерацией данных. Однако при подходе с минимизацией эмпирического риска работа происходит без учета выбора такого распределения. Этим он отличается от стандартных статистических подходов, которые явно требуют знания  $p(\mathbf{x}, y)$ . Более того, поскольку распределение является совместным, сочетающим примеры  $\mathbf{x}$  и метки  $y$ , метки могут быть недетерминированными. В отличие от стандартной статистики, здесь не требуется указывать распределение шума для меток  $y$ .

## 8.3. ОЦЕНКА ПАРАМЕТРОВ

В разделе 8.2 мы не занимались явным моделированием нашей задачи с использованием вероятностных распределений. В этом разделе будет рассмотрено, как применять вероятностные распределения для моделирования нашей неопределенности, обусловленной процессом наблюдения и нашей неуверенностью в параметрах предикторов. В разделе 8.3.1 мы введем понятие правдоподобия, концептуально аналогичное функции потерь (раздел 8.2.2) при минимизации эмпирического риска. Концепция априорных вероятностей (раздел 8.3.2) аналогична концепции регуляризации (раздел 8.2.3).

### 8.3.1. Метод максимального правдоподобия

Идея в основе *метода максимального правдоподобия* (ММП) (maximum likelihood estimation) — определить функцию параметров, которая позволяет нам найти модель, хорошо обучающуюся на данных. Ключевой аспект в проблеме оценки — это функция *правдоподобия* или, точнее, ее отрицательный логарифм. Для данных, представленных случайной переменной  $\mathbf{x}$ , и для семейства плотностей вероятностей  $p(\mathbf{x} | \theta)$ , параметризованных  $\theta$ , *отрицательный логарифм правдоподобия* задается как

$$\mathcal{L}_x(\theta) = -\log p(\mathbf{x} | \theta). \quad (8.14)$$

Нотация  $\mathcal{L}_x(\theta)$  подчеркивает тот факт, что параметр  $\theta$  варьируется, а данные  $\mathbf{x}$  фиксированы. Часто мы опускаем ссылку на  $\mathbf{x}$  при записи отрицательного логарифма правдоподобия, поскольку в реальности он является функцией  $\theta$  и записывается как  $\mathcal{L}(\theta)$ , когда случайная переменная, представляющая неопределенность в данных, свободна от контекста.

Давайте интерпретируем, что представляет собой плотность вероятностей  $p(\mathbf{x} | \theta)$ , смоделировав ее для фиксированного значения  $\theta$ . Это распределение, моделирующее неопределенность данных. Иными словами, как только мы выберем тип функции, которую хотим видеть в качестве предиктора, правдоподобность дает нам вероятность, с которой можно наблюдать данные  $\mathbf{x}$ .

Дополнительно возможна такая перспектива: если рассматривать данные как фиксированные (поскольку мы их пронаблюдали) и варьировать параметры  $\theta$ , то о чём нам сообщит  $\mathcal{L}(\theta)$ ? О том, насколько вероятна конкретная конфигурация  $\theta$  для наблюдений  $\mathbf{x}$ . Основываясь на этой второй перспективе, средство оценки максимального правдоподобия дает нам наиболее вероятный параметр  $\theta$  для конкретного набора данных.

Мы рассматриваем конфигурацию обучения с учителем, где мы получаем пары  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$  с  $\mathbf{x}_n \in \mathbb{R}^D$  и метками  $y_n \in \mathbb{R}$ . Мы хотим сконструировать пре-

диктор, который бы принимал вектор признаков  $\mathbf{x}_n$  в качестве ввода и давал прогноз  $y_n$  (или какой-нибудь близкий к нему). То есть, имея вектор  $\mathbf{x}_n$ , мы хотим получить распределение вероятностей для метки  $y_n$ . Иными словами, мы указываем условное вероятностное распределение меток, имея примеры конкретной конфигурации параметра  $\theta$ .

### Пример 8.4

Первый часто применяемый пример — это указание, что условная вероятность меток в примерах описывается гауссовым распределением. Иными словами, мы предполагаем, что сможем объяснить наблюдаемую нами неопределенность независимым гауссовым шумом (раздел 6.5) с нулевым средним  $\varepsilon_n \sim \mathcal{N}(0, \sigma^2)$ . Далее мы предполагаем, что линейная модель  $\mathbf{x}_n^\top \boldsymbol{\theta}$  используется для прогноза. Таким образом, мы указываем гауссово правдоподобие для каждой используемой в качестве примеров пары  $\mathbf{x}_n, y_n$

$$p(y_n | \mathbf{x}_n, \boldsymbol{\theta}) = \mathcal{N}(y_n | \mathbf{x}_n^\top \boldsymbol{\theta}, \sigma^2). \quad (8.15)$$

Гауссово правдоподобие для заданного параметра  $\theta$  проиллюстрировано на рис. 8.3. В разделе 9.2 будет рассмотрено, как явно расширить предыдущее выражение в терминах гауссова распределения.

Мы предполагаем, что примеры во множестве  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$  являются *независимыми и одинаково распределенными*. Слово «независимый» (раздел 6.4.5) подразумевает, что правдоподобие всего набора данных ( $\mathcal{Y} = \{y_1, \dots, y_N\}$  и  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ) факторизуется в произведение правдоподобий каждого отдельного примера

$$p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}) = \prod_{n=1}^N p(y_n | \mathbf{x}_n, \boldsymbol{\theta}), \quad (8.16),$$

где  $p(y_n | \mathbf{x}_n, \boldsymbol{\theta})$  — это конкретное распределение (в примере 8.4 оно было гауссовым). Выражение «одинаково распределено» означает, что все члены в произведении (8.16) относятся к одному и тому же распределению, и все они используют общие параметры. С точки зрения оптимизации зачастую проще вычислять такие функции, которые можно разложить на суммы более простых функций. Следовательно, в машинном обучении часто рассматривается отрицательный логарифм правдоподобия<sup>1</sup>

$$\mathcal{L}(\boldsymbol{\theta}) = -\log p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}) = -\sum_{n=1}^N \log p(y_n | \mathbf{x}_n, \boldsymbol{\theta}). \quad (8.17)$$

---

<sup>1</sup> Как вы помните,  $\log(ab) = \log(a) + \log(b)$ .

Явно напрашивается такая интерпретация: поскольку  $\theta$  находится в правой части условия  $p(y_n | \mathbf{x}_n, \theta)$  (8.15), она должна трактоваться как наблюдаемая и фиксированная. Но эта интерпретация неверна. Отрицательный логарифм правдоподобия  $\mathcal{L}(\theta)$  – это функция  $\theta$ . Следовательно, чтобы найти хороший вектор параметров  $\theta$ , который качественно объясняет данные  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ , минимизируем отрицательный логарифм правдоподобия  $\mathcal{L}(\theta)$  относительно  $\theta$ .

**ПРИМЕЧАНИЕ** Знак «минус» в (8.17) – исторический рудимент, сохранившийся из-за соглашения, что правдоподобие требуется максимизировать; но в литературе о численной оптимизации обычно изучается минимизация функций. ♦

### Пример 8.5

Продолжая наш пример с гауссовыми правдоподобиями (8.15), отметим, что отрицательный логарифм правдоподобия можно переписать как

$$\mathcal{L}(\theta) = -\sum_{n=1}^N \log p(y_n | \mathbf{x}_n, \theta) = -\sum_{n=1}^N \log \mathcal{N}(y_n | \mathbf{x}_n^\top \theta, \sigma^2) = \quad (8.18a)$$

$$= -\sum_{n=1}^N \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_n - \mathbf{x}_n^\top \theta)^2}{2\sigma^2}\right) = \quad (8.18b)$$

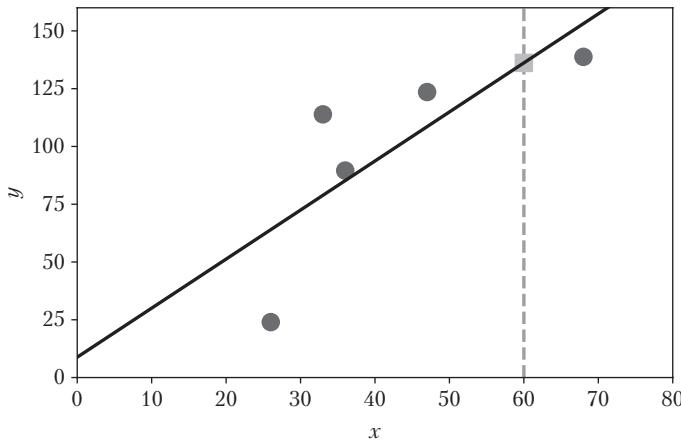
$$= -\sum_{n=1}^N \log \exp\left(-\frac{(y_n - \mathbf{x}_n^\top \theta)^2}{2\sigma^2}\right) - \sum_{n=1}^N \log \frac{1}{\sqrt{2\pi\sigma^2}} = \quad (8.18c)$$

$$= \frac{1}{2\sigma^2} \sum_{n=1}^N (y_n - \mathbf{x}_n^\top \theta)^2 - \sum_{n=1}^N \log \frac{1}{\sqrt{2\pi\sigma^2}}. \quad (8.18d)$$

Поскольку  $\sigma$  дано, второй член в (8.18d) является константой, и минимизация  $\mathcal{L}(\theta)$  соответствует решению задачи наименьших квадратов (сравните с (8.8)), выраженной в первом члене.

Оказывается, что для гауссовых правдоподобий результирующая задача оптимизации, соответствующая оценке максимального правдоподобия, имеет замкнутую форму решения. Подробнее мы поговорим об этом в главе 9. На рис. 8.5 показан регрессионный датасет, получаемый на основе параметров максимального правдоподобия. Оценка максимального правдоподобия может быть подвержена переобучению (раздел 8.3.3), аналогично нерегуляризованной минимизации эмпирического риска (раздел 9.2.3). Для других функций правдоподобия, например если мы моделируем наш шум при помощи негауссовых распределений,

оценка максимального правдоподобия может не иметь аналитического решения в замкнутой форме. В таком случае приходится прибегнуть к методам численной оптимизации, рассматриваемым в главе 7.



**Рис. 8.5.** Для заданных данных оценка максимального правдоподобия параметров дает сплошную диагональную прямую. Квадрат соответствует прогнозируемому значению максимального правдоподобия при  $x = 60$

### 8.3.2. Оценка апостериорного максимума

Если у нас есть априорные знания о распределении параметров  $\theta$ , то дополнительно к правдоподобию можно рассмотреть еще один член. Этот дополнительный член — априорное вероятностное распределение по параметрам  $p(\theta)$ . Как нам следует обновить распределение  $\theta$  для заданного априорного значения после наблюдения данных  $\mathbf{x}$ ? Иными словами, как нам представить тот факт, что мы обладаем более конкретными знаниями об  $\theta$ , когда пронаблюдаем  $\mathbf{x}$ ? Теорема Байеса, рассмотренная в разделе 6.3, дает нам научно-теоретический аппарат для уточнения наших вероятностных распределений случайных переменных. Она позволяет вычислять *апостериорное* распределение  $p(\theta | \mathbf{x})$  (более конкретное значение) о параметрах  $\theta$  по общим *априорным* утверждениям (априорному распределению)  $p(\theta)$  и функции  $p(\mathbf{x} | \theta)$ , связывающей параметры  $\theta$  и наблюдаемые данные  $\mathbf{x}$ , называемой *правдоподобием*.

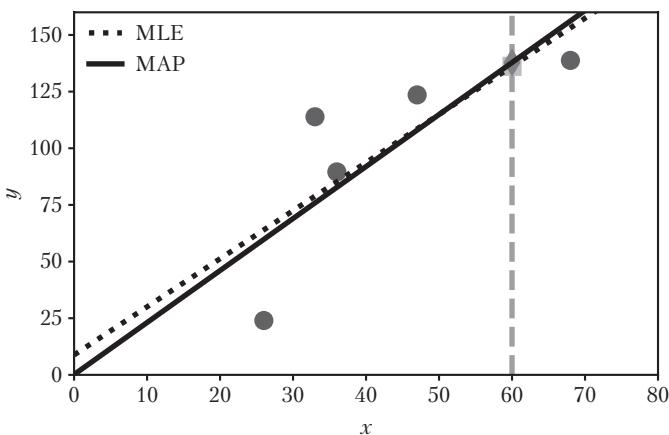
$$p(\theta | \mathbf{x}) = \frac{p(\mathbf{x} | \theta) p(\theta)}{p(\mathbf{x})}. \quad (8.19)$$

Напоминаем, что мы хотим найти такой параметр  $\theta$ , который максимизирует апостериорное значение. Поскольку распределение  $p(\mathbf{x})$  не зависит от  $\theta$ ,

мы в целях оптимизации можем игнорировать значение знаменателя и получить

$$p(\theta | \mathbf{x}) \propto p(\mathbf{x} | \theta)p(\theta). \quad (8.20)$$

Предыдущее пропорциональное отношение скрывает плотность данных  $p(\mathbf{x})$ , определить которую может быть сложно. Вместо оценки минимума отрицательного алгоритма правдоподобия мы теперь оцениваем минимум отрицательного логарифма плотности апостериорного распределения, что также называется *оценкой апостериорного максимума* (maximum a posteriori estimation, MAP). На рис. 8.6 показано, что будет, если добавить гауссово априорное распределение с нулевым средним.



**Рис. 8.6.** Сравнение прогнозов, при которых делается оценка максимального правдоподобия (MLE) и оценка MAP при  $x = 60$ . Априорное значение немного выравнивает крутизну кривой, а точка пересечения становится ближе к нулю. В данном случае фактор, сдвигающий точку пересечения ближе к нулю, на самом деле увеличивает крутизну

### Пример 8.6

Дополнительно к предположению о гауссовом правдоподобии из предыдущего примера, допустим, что вектор параметров распределен как многомерный гауссиан с нулевым средним, то есть  $p(\theta) = \mathcal{N}(\mathbf{0}, \Sigma)$ , где  $\Sigma$  — ковариационная матрица (раздел 6.5). Обратите внимание: сопряженное априорное распределение гауссiana также является гауссианом (раздел 6.6.1). Следовательно, ожидается, что апостериорное распределение также будет гауссовым. Подробнее максимум апостериорного распределения мы рассмотрим в главе 9.

Идея учета априорных знаний о том, где располагаются хорошие параметры, распространена в машинном обучении. В качестве альтернативной точки зрения, которую мы рассмотрим в разделе 8.2.3, предлагается регуляризация. Регуляризация вводит в выражение дополнительный член, сдвигающий регулятирующие параметры ближе к началу координат. Максимальная апостериорная оценка может считаться мостиком между не вероятностным и вероятностным миром, поскольку явно признает необходимость априорного распределения, но все равно обеспечивает лишь точечную оценку параметров.

**ПРИМЕЧАНИЕ** Оценка максимального правдоподобия  $\theta_{\text{ML}}$  обладает следующими свойствами (Lehmann and Casella, 1998; Efron and Hastie, 2016):

- Асимптотическая согласованность: оценка максимального правдоподобия сходится к истинному значению в пределе бесконечного множества наблюдений плюс случайная ошибка, которая приблизительно нормальна.
- Размер выборок, необходимых для достижения этих свойств, может быть весьма велик.
- Дисперсия ошибки снижается как  $1/N$ , где  $N$  — количество точек данных.
- Оценка максимального правдоподобия может приводить к *переобучению*, особенно в условиях «небольших» данных.



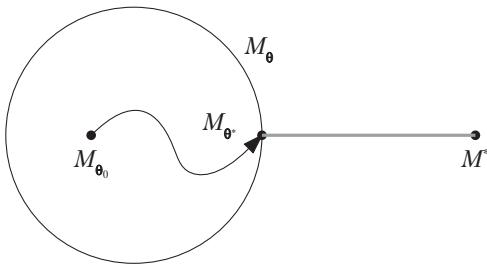
Принцип оценки максимального правдоподобия (и оценки апостериорного максимума) использует вероятностное моделирование при рассуждении о неопределенности параметров данных и модели. Однако мы здесь пока еще не используем вероятностное моделирование на полную мощность. В этом разделе регулятирующая процедура обучения по-прежнему дает точечную оценку предиктора, то есть в результате возвращается всего одно множество значений параметров, которое представляет наилучший предиктор. В разделе 8.4 мы исходим из того, что значения параметров также должны трактоваться как случайные переменные, и мы будем оценивать не «наилучшие» значения в данном распределении, а использовать при подготовке прогнозов все распределение параметров.

### 8.3.3. Обучение модели

Рассмотрим ситуацию, в которой нам дан датасет, и мы хотим обучить на нем параметризованную модель. Говоря об обучении (*fitting*), мы, как правило, имеем в виду оптимизацию/обучение модели таким образом, чтобы минимизи-

ровать некоторую функцию потерь, например отрицательный логарифм правдоподобия. Рассуждая о максимальном правдоподобии (раздел 8.3.1) и оценке апостериорного максимума (8.3.2), мы уже затронули два распространенных алгоритма, применяемых для приближения модели.

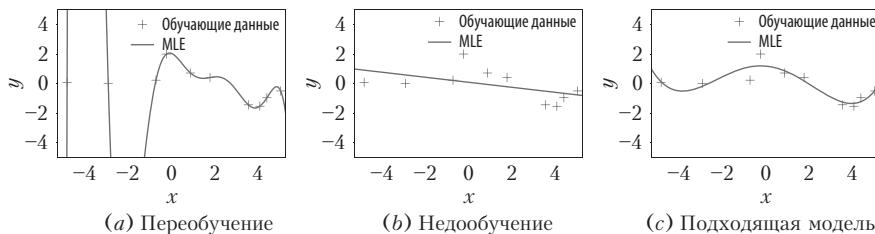
При параметризации модели определяется класс моделей  $M_\theta$ , которым мы можем оперировать. Например, в случае линейной регрессии можно определить отношение между входными точками  $x$  и (свободными от шума) наблюдениями  $y$ , такими чтобы  $y = ax + b$ , где  $\theta := \{a, b\}$  это параметры модели. В данном случае параметры модели  $\theta$  описывают семейство аффинных функций, то есть прямых линий с уклоном  $a$ , соответствующим смещению от 0 до  $b$ . Допустим, данные поступают от модели  $M^*$ , которая нам неизвестна. Для заданного обучающего набора мы оптимизируем  $\theta$  таким образом, чтобы  $M_\theta$  было как можно ближе к  $M^*$ , где «близость» определяется объективной функцией, которую мы оптимизируем (например, квадратичная функция потерь на обучающих данных). На рис. 8.7 проиллюстрирована ситуация, где у нас есть небольшой класс моделей (обозначенный кругом  $M_\theta$ ), а модель генерации данных  $M^*$  не входит во множество рассмотренных моделей. Поиск параметров мы начинаем с  $M_{\theta_0}$ . После оптимизации, то есть как только мы получим наилучшие возможные параметры  $\theta^*$ , возможны три случая: (i) переобучение, (ii) недообучение, (iii) подходящая модель. В общих чертах поясним, что означают эти концепции.



**Рис. 8.7.** Обучение модели. В параметризованном классе  $M_\theta$  моделей мы оптимизируем параметры модели  $\theta$ , так чтобы минимизировать расстояние до истинной (неизвестной) модели  $M^*$

Грубо говоря, под *переобучением* понимается ситуация, в которой параметризованный класс моделей слишком насыщен, чтобы смоделировать набор данных, сгенерированный  $M^*$ , то есть  $M_\theta$  могла бы моделировать гораздо более сложные датасеты. Например, если набор данных был сгенерирован линейной функцией, а мы определяем  $M_\theta$  как класс многочленов седьмого порядка, то можем в данном случае моделировать не только линейные функции, но и многочлены степени два, три и далее. Переобученные модели, как правило, содержат большое коли-

чество параметров. Часто приходится наблюдать, что чрезмерно гибкий класс моделей  $M_0$  использует всю свою силу моделирования, чтобы снизить ошибку при обучении<sup>1</sup>. Если обучающие данные зашумлены, то эта модель сможет вычленить в самом шуме какой-то полезный сигнал. Это спровоцирует колоссальные проблемы, если мы попытаемся делать прогнозы в отрыве от обучающих данных. На рис. 8.8(a) приведен пример переобучения в контексте регрессии, где параметры модели запоминаются средствами максимального правдоподобия (раздел 8.3.1). Подробнее о переобучении при регрессии мы поговорим в разделе 9.2.2.



**Рис. 8.8.** Обучение (метод максимального правдоподобия, MLE) различных классов моделей на регрессионном датасете

Сталкиваясь с *недообучением*, мы имеем дело с противоположной проблемой, где класс моделей  $M_0$  недостаточно насыщен. Например, если наше множество данных было сгенерировано синусоидальной функцией, но  $\theta$  параметризует лишь прямые, то наилучшая процедура оптимизации не приблизит нас к истинной модели. Тем не менее, мы все равно оптимизируем параметры и находим наилучшую прямую, которая моделирует этот набор данных. На рис. 8.8(b) приводится пример модели, которая недообучается — так как данная модель недостаточно гибкая. Как правило, у недообучающихся моделей мало параметров.

Третий случай возникает, когда параметризованный класс моделей почти правилен. В таком случае наша модель обеспечивает хорошее приближение, то есть не переобучается и не недообучается. Это означает, что наш класс моделей обладает вполне достаточной насыщенностью для описания того класса моделей, что нам дан. На рис. 8.8(c) показана модель, которая весьма хорошо описывает заданный датасет. В идеале именно с этим классом моделей нам хотелось бы работать, так как он обладает хорошими свойствами обобщения.

<sup>1</sup> Один из способов выявить переобучение на практике — заметить, что модель обладает низким риском на обучающем датасете, но высоким риском на тестовом датасете; это обнаруживается при кросс-валидации (раздел 8.2.4).

На практике зачастую определяются очень насыщенные классы моделей  $M_0$  со многими параметрами, например глубокие нейронные сети. Чтобы сгладить проблему переобучения, можно использовать регуляризацию (раздел 8.2.3) или априорные значения (раздел 8.3.2). В разделе 8.6 мы поговорим о том, как выбирать класс модели.

### 8.3.4. Дополнительное чтение

При рассмотрении вероятностных моделей принцип оценки максимального правдоподобия обобщает идею регрессии наименьших квадратов для линейных моделей, о чём мы подробно поговорим в главе 9. Ограничиваая предиктор только линейной формой путем применения к ее выводу дополнительной нелинейной функции  $\varphi$ , то есть

$$p(y_n | \mathbf{x}_n, \theta) = \varphi(\theta^T \mathbf{x}_n), \quad (8.21)$$

можно рассматривать другие модели для иных прогностических задач, например для бинарной классификации или моделирования счетных данных (McCullagh and Nelder, 1989). В качестве альтернативной трактовки можно рассмотреть правдоподобия из экспоненциального семейства (раздел 6.6). Класс моделей, в которых имеется линейная зависимость между параметрами и данными и имеется потенциально нелинейное преобразование  $\varphi$  (под названием *функция связи*), называется классом обобщенных линейных моделей (Agresti, 2002, глава 4).

У оценки максимального правдоподобия богатая история. Изначально этот метод был предложен сэром Рональдом Фишером (Ronald Fisher) в 1930-е. Идею вероятностных моделей мы подробнее рассмотрим в разделе 8.4. Исследователи вероятностных моделей, в частности, спорят, что лучше — фреквентистская статистика или вероятностные модели. Как упоминалось в разделе 6.1.1, эта дискуссия сводится к определению вероятности. Как вы помните из раздела 6.1, вероятность можно трактовать как обобщение (с допущением неопределенности) логического рассуждения (Cheeseman, 1985; Jaynes, 2003). Метод оценки максимального правдоподобия по природе своей фреквентистский, поэтому заинтересовавшиеся могут обратиться к Эфрону и Хасти (Efron and Hastie, 2016), чтобы составить взвешенное впечатление как о байесовской, так и о фреквентистской статистике.

Существуют такие вероятностные модели, где оценка максимального правдоподобия невозможна. Рекомендуем обратиться к более продвинутым книгам по статистике, например Казеллы и Бергера (Casella and Berger, 2002), где рассматриваются такие подходы, как метод импульсов,  $M$ -оценка и оценка уравнений.

## 8.4. ВЕРОЯТНОСТНЫЕ МОДЕЛИ И ИНФЕРЕНС

В машинном обучении нас часто интересует интерпретация и анализ данных, например для прогнозирования будущих событий и принятия решений. Чтобы упростить такую задачу, зачастую строятся модели, описывающие *генеративный процесс*, при котором порождаются наблюдаемые данные.

Например, можно описать результат эксперимента с подбрасыванием монетки («орел» или «решка») в два этапа. Сначала мы определяем параметр  $\mu$ , описывающий вероятность «орла» как параметр распределения Бернулли (глава 6); далее можно сделать выборку результатов  $x \in \{\text{орел, решка}\}$  из распределения Бернулли  $p(x|\mu) = \text{Ber}(\mu)$ . Параметр  $\mu$  порождает конкретный набор данных  $\mathcal{X}$  и зависит от используемой монетки. Поскольку  $\mu$  заранее не известен и не поддается непосредственному наблюдению, нужны механизмы, которые позволили бы что-то узнать о  $\mu$  при наличии наблюдаемых результатов в экспериментах с подбрасыванием монетки. В дальнейшем мы обсудим, как для этой цели может использоваться вероятностное моделирование.

### 8.4.1. Вероятностные модели

Вероятностные модели<sup>1</sup> представляют неопределенный характер эксперимента как распределения вероятностей. Польза от применения вероятностных моделей в том, что они предлагают единый и согласованный аппарат теории вероятностей (раздел 6) для моделирования, инференса, прогнозирования и выбора модели.

В вероятностном моделировании совместное распределение  $p(\mathbf{x}, \theta)$  наблюдаемых переменных  $\mathbf{x}$  и скрытых параметров  $\theta$  имеет первоочередное значение. В этом распределении заключена информация о:

- априорных значениях и правдоподобии (правило произведения, раздел 6.3);
- маргинальном правдоподобии  $p(\mathbf{x})$ , которое сыграет важную роль в выборе модели (раздел 8.6); оно может быть рассчитано так: берется совместное распределение, и из него интегрируются параметры (по правилу суммы, раздел 6.3);
- апостериорном значении, которое можно получить, разделив совместное правдоподобие на маргинальное.

Только совместное распределение обладает таким свойством. Следовательно, вероятностная модель описывается совместным распределением всех ее случайных переменных.

---

<sup>1</sup> Вероятностная модель описывается совместным распределением всех случайных переменных.

### 8.4.2. Байесовский инференс

Ключевая задача в машинном обучении — взять модель и данные, выявить значения скрытых переменных модели,  $\theta$ , имея наблюдаемые переменные  $x$ . В разделе 8.3.1 уже обсуждались два способа оценки параметров модели  $\theta$ : с помощью максимального правдоподобия или апостериорного максимума<sup>1</sup>. В обоих случаях мы получаем единственно лучшее значение  $\theta$ , так что ключевая алгоритмическая проблема оценки параметров сводится к решению задачи оптимизации. Когда нам известны эти точечные оценки  $\theta^*$ , мы пользуемся ими для прогнозирования. В частности, прогностическое распределение будет  $p(x | \theta^*)$ , где мы используем  $\theta^*$  в функции правдоподобия.

Как обсуждалось в разделе 6.3, сосредоточение на одной статистике апостериорного распределения (например, на параметре  $\theta^*$ , максимизирующем апостериорное значение) приводит к потере информации, которая может быть критически важна в системе, использующей прогноз  $p(x | \theta^*)$  для принятия решений. Как правило, такие системы принятия решений имеют иные объективные функции, нежели правдоподобие, среднеквадратичная потеря или ошибка классификации. Следовательно, иметь под рукой полное апостериорное распределение может быть исключительно полезно, это помогает принимать более надежные решения. *Байесовский инференс* заключается в нахождении такого апостериорного распределения (Gelman et al., 2004)<sup>2</sup>. Для набора данных  $X$  априорный параметр  $p(\theta)$  и функция правдоподобия

$$p(\theta | X) = \frac{p(X | \theta) p(\theta)}{p(X)}, \quad p(X) = \int p(X | \theta) p(\theta) d\theta \quad (8.22)$$

получаются путем применения теоремы Байеса. Ключевая идея, располагающая к применению теоремы Байеса, — инвертировать отношение между параметрами  $\theta$  и данными  $X$  (что определяется в зависимости от правдоподобия) для получения апостериорного распределения  $p(\theta | X)$ <sup>3</sup>.

Если у нас есть апостериорное распределение параметров, то из этого следует, что с его помощью можно распространить неопределенность с параметров на данные. В частности, при распределении  $p(\theta)$  для параметров наши прогнозы вычисляются как

$$p(x) = \int p(x | \theta) p(\theta) d\theta = \mathbb{E}_{\theta} [p(x | \theta)] \quad (8.23)$$

<sup>1</sup> Оценка параметров может быть сформулирована как задача оптимизации.

<sup>2</sup> Байесовский инференс связан с узнаванием распределения случайных переменных.

<sup>3</sup> Байесовский инференс инвертирует отношение между параметрами и данными.

и уже не будут зависеть от параметров модели ( $\theta$ ), которые мы смогли сделать несущественными / удалить интегрированием. Уравнение (8.23) показывает, что прогноз является средним среди всех вероятных значений параметра  $\theta$ , где такая вероятность выражена в виде распределения параметров  $p(\theta)$ .

Обсудив оценку параметров в разделе 8.3, а байесовский инференс здесь, давайте сравним два этих подхода к обучению. Оценка параметров методом оценки максимального правдоподобия (MLE) или апостериорного максимума (MAP) дает непротиворечивую точечную оценку параметров,  $\theta^*$ , а основная вычислительная задача, которую требуется при этом решить, — оптимизация. Напротив, байесовский инференс результирует в (апостериорное) распределение, и ключевая вычислительная задача, которую при этом приходится решить, — интегрирование. Прогнозы с использованием точечных оценок прямолинейны, тогда как при прогнозировании с помощью байесовского метода требуется дополнительно решать задачу по интегрированию; см. (8.23). Тем не менее байесовский инференс обеспечивает систематический способ учета априорных сведений, побочной информации, а также структурных знаний — а все это непросто сделать в контексте оценки параметров. Более того, распространение неопределенности параметров на прогноз может пригодиться в системах принятия решений для оценки рисков и исследовательской работы в контексте обучения с эффективным использованием данных (Deisenroth et al., 2015; Kamthe and Deisenroth, 2018).

Хотя байесовский инференс — это систематический математический фреймворк для получения информации о параметрах и для выполнения прогнозов, с ним сопряжены некоторые практические сложности; точнее, с задачами интеграции, которые нам придется решать, см. (8.22) и (8.23). Еще точнее, если не выбрать сопряженное априорное распределение для параметров (раздел 6.6.1), то интегралы в (8.22) и (8.23) окажутся не находимыми аналитически, и мы не сможем вычислить апостериорные значения, прогнозы или маргинальное правдоподобие в замкнутой форме. В таких случаях приходится прибегать к приближениям, например методу Монте-Карло с использованием марковских цепей (Markov chain Monte Carlo, MCMC) (Gilks et al., 1996) или детерминированным приближениям, например приближению Лапласа (Bishop, 2006; Barber, 2012; Murphy, 2012), вариационному инференсу (Jordan et al., 1999; Blei et al., 2017) или распространению ожидания (Minka, 2001).

Несмотря на все эти сложности, байесовский инференс успешно применяется для решения разнообразных задач, в том числе крупномасштабного тематического моделирования (Hoffman et al., 2013), прогнозирования кликабельности (Graepel et al., 2010), обучения с подкреплением в управляющих системах с эффективным использованием данных (Herbrich et al., 2007) и создания крупномасштабных рекомендательных систем. Существуют универсальные подходы, например байесовская оптимизация (Brochu et al., 2009; Snoek et al., 2012;

Shahriari et al., 2016), которые весьма полезны для эффективного поиска метапараметров моделей или алгоритмов.

**ПРИМЕЧАНИЕ** В литературе по машинному обучению наблюдается несколько произвольное разделение между (случайными) «переменными» и «параметрами». Тогда как параметры обычно рассчитываются (например, методом максимального правдоподобия), переменные обычно выводятся на периферию. В этой книге мы не так строго придерживаемся этого разделения, поскольку в принципе можем задать априорное значение для любого параметра, а потом убрать его методом интегрирования. В таком случае параметр превратится в случайную переменную, согласно вышеописанному разделению. ♦

### 8.4.3. Модели латентных переменных

На практике иногда бывает полезно иметь в составе модели дополнительные латентные переменные  $\mathbf{z}$  (кроме параметров модели  $\theta$ ) (Moustaki et al., 2015). Такие латентные переменные отличаются от параметров модели  $\theta$ , так как не параметризуют модель явно. Латентные переменные могут описывать процесс генерации данных, тем самым способствуя интерпретируемости модели. Также они зачастую помогают упростить структуру модели и позволяют определять более простые и насыщенные структуры моделей. Упрощение структуры модели зачастую сопровождается тем, что такая модель имеет сравнительно немногого параметров (Paquet, 2008; Murphy, 2012). Обучение на моделях с латентными переменными (хотя бы методом максимального правдоподобия) можно выполнять систематически при помощи алгоритма максимизации математического ожидания (expectation maximization, EM) (Dempster et al., 1977; Bishop, 2006). Пример, в котором могут пригодиться такие латентные переменные, — это, в частности, анализ главных компонент для снижения размерности (глава 10), модели смеси гауссовых распределений для оценки плотности (глава 11), скрытые марковские модели (Maybeck, 1979) или динамические системы (Ghahramani and Roweis, 1999; Ljung, 1999) для моделирования временных рядов, метаобучения и генерации задач (Hausman et al., 2018; Sæmundsson et al., 2018). Хотя введение таких переменных может упростить как структуру модели, так и процесс порождения, обучение на моделях с латентными переменными, как правило, протекает сложно, что будет показано в главе 11.

Модели с латентными переменными также позволяют определять процесс, генерирующий данные на основе параметров. Давайте его рассмотрим. Обозначив данные через  $\mathbf{x}$ , параметры модели через  $\theta$ , а латентные переменные через  $\mathbf{z}$ , получим условное распределение

$$p(\mathbf{x} | \theta, \mathbf{z}), \tag{8.24}$$

при помощи которого сможем сгенерировать данные для любых параметров модели и латентных переменных. Если  $\mathbf{z}$  — это латентные переменные, задаем для них априорное значение  $p(\mathbf{z})$ .

Как и в случае с моделями, которые мы обсуждали выше, модели с латентными переменными могут применяться для изучения параметров и для инференса в контекстах, рассмотренных в разделах 8.3 и 8.4.2. Чтобы способствовать обучению (например, методом оценки максимального правдоподобия или методом байесовского инференса), придерживаемся двухэтапной процедуры. Сперва вычисляем правдоподобие  $p(\mathbf{x}|\theta)$  модели, не зависящее от латентных переменных. Затем используем это правдоподобие для вычисления параметров или байесовского инференса, где применяем те же выражения, что и в разделах 8.3 и 8.4.2 соответственно.

Поскольку функция правдоподобия  $p(\mathbf{x}|\theta)$  является предиктивным распределением данных с учетом параметров модели, нам необходимо интегрировать наши латентные переменные, так чтобы

$$p(\mathbf{x}|\theta) = \int p(\mathbf{x}|\theta, \mathbf{z}) p(\mathbf{z}) d\mathbf{z}, \quad (8.25)$$

где  $p(\mathbf{x}|\mathbf{z}, \theta)$  дано в (8.24), а  $p(\mathbf{z})$  является априорным для латентных переменных. Обратите внимание, что правдоподобие не должно зависеть от латентных переменных  $\mathbf{z}$ , а является лишь функцией данных  $\mathbf{x}$  и параметров модели  $\theta$ .

Правдоподобие в (8.25) непосредственно располагает к оценке параметров методом максимального правдоподобия. МАР-оценка также легко выполняется с дополнительным априорным значением для параметров модели  $\theta$ , как рассматривалось в разделе 8.3.2. Более того, с учетом правдоподобия (8.25), байесовский инференс (раздел 8.4.2) в модели с латентными переменными работает как обычно: задаем априорное значение  $p(\theta)$  для параметров модели и при помощи теоремы Байеса получаем апостериорное распределение

$$p(\theta|\mathcal{X}) = \frac{p(\mathcal{X}|\theta)p(\theta)}{p(\mathcal{X})} \quad (8.26)$$

по параметрам модели, имея набор данных. Апостериорное в (8.26) может использоваться для прогнозов в системе байесовского инференса; см. (8.23).

Одна из трудностей, связанных с использованием такой модели латентных переменных, заключается в том, что правдоподобие  $p(\mathcal{X}|\theta)$  требует маргинализации латентных переменных согласно (8.25). Кроме случаев, когда мы выбираем априорное сопряжение  $p(\mathbf{z})$  для  $p(\mathbf{x}|\mathbf{z}, \theta)$ , маргинализация в (8.25) не является аналитически находимой, приходится довольствоваться приближениями (Bishop, 2006; Paquet, 2008; Murphy, 2012; Moustaki et al., 2015).

Аналогично апостериорному распределению параметров (8.26), можно вычислить распределение для латентных переменных согласно

$$p(\mathbf{z} | \mathcal{X}) = \frac{p(\mathcal{X} | \mathbf{z}) p(\mathbf{z})}{p(\mathcal{X})}, \quad p(\mathcal{X} | \mathbf{z}) = \int p(\mathcal{X} | \mathbf{z}, \boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}, \quad (8.27)$$

где  $p(\mathbf{z})$  является априорным распределением латентных переменных, а  $p(\mathcal{X} | \mathbf{z})$  требует исключить параметры модели  $\boldsymbol{\theta}$ .

Учитывая, насколько сложно аналитически вычислять интегралы, понятно, что одновременная маргинализация как латентных переменных, так и параметров модели вообще невозможна (Bishop, 2006; Murphy, 2012). Проще рассчитать апостериорное распределение латентных переменных, но эта величина обусловлена параметрами модели, то есть

$$p(\mathbf{z} | \mathcal{X}, \boldsymbol{\theta}) = \frac{p(\mathcal{X} | \mathbf{z}, \boldsymbol{\theta}) p(\mathbf{z})}{p(\mathcal{X} | \boldsymbol{\theta})}, \quad (8.28)$$

где  $p(\mathbf{z})$  является априорным распределением латентных переменных, а  $p(\mathcal{X} | \mathbf{z}, \boldsymbol{\theta})$  дано в (8.24).

В главах 10 и 11 мы выведем функции правдоподобия для анализа главных компонент и моделей смесей гауссовых распределений соответственно. Кроме того, мы вычислим апостериорные распределения (8.28) латентных переменных.

**ПРИМЕЧАНИЕ** Возможно, в следующих главах мы не будем проводить столь четкого разделения между латентными переменными  $\mathbf{z}$  и неопределенными параметрами модели  $\boldsymbol{\theta}$  и будем называть их как «латентными», так и «скрытыми» (поскольку они ненаблюдаются). В главах 10 и 11, где используются латентные переменные  $\mathbf{z}$ , мы все-таки обратим внимание на эту разницу, так как у нас будет два разных типа скрытых переменных: параметры модели  $\boldsymbol{\theta}$  и латентные переменные  $\mathbf{z}$ . ◆

Можно воспользоваться тем фактом, что все элементы вероятностной модели являются случайными переменными, и выработать унифицированный язык для их представления. В разделе 8.5 будет показан компактный графический язык для представления структуры вероятностных моделей. Мы воспользуемся им в следующих главах.

#### 8.4.4. Дальнейшее чтение

Вероятностные модели в машинном обучении (Bishop, 2006; Barber, 2012; Murphy, 2012) удобны для систематического описания неопределенности, присущей

данным и прогностическим моделям. У Гахрамани (Ghahramani, 2015) дан краткий обзор вероятностных моделей в машинном обучении. Возможно, нам удастся аналитически рассчитать интересующие параметры данной вероятностной модели, но в целом аналитические решения встречаются редко, и приходится использовать вычислительные методы, такие как формирование выборки (Gilks et al., 1996; Brooks et al., 2011) или вариационный инференс (Jordan et al., 1999; Blei et al., 2017). Мустаки и др. (Moustaki et al., 2015) и Паке (Paquet, 2008) дают хороший обзор байесовского инференса в моделях с латентными переменными.

В последние годы было предложено несколько языков программирования, цель которых — работа с переменными, определенными в программе как случайные величины, соответствующие вероятностным распределениям. Это нужно для написания сложных функций вероятностных распределений, пока под капотом компилятор автоматически обеспечивает выполнение правил байесовского инференса. Эта стремительно развивающаяся дисциплина называется *вероятностным программированием*.

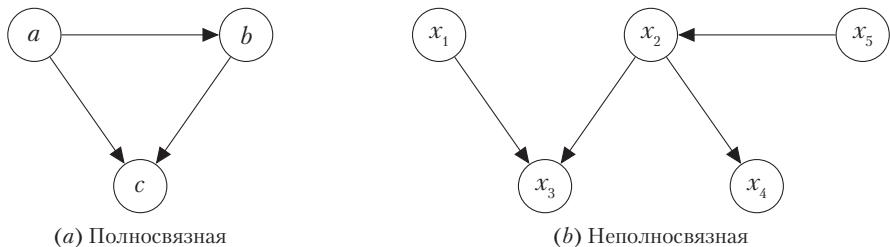
## 8.5. НАПРАВЛЕННЫЕ ГРАФИЧЕСКИЕ МОДЕЛИ

В этом разделе мы познакомимся с графическим языком для описания вероятностных моделей и получения так называемой *направленной графической модели*<sup>1</sup>. Этот язык обеспечивает лаконичный способ указания вероятностных моделей и позволяет читателю визуально разбирать зависимости между случайными переменными. Графическая модель наглядно показывает, как именно совместное распределение, охватывающее все случайные переменные, можно разложить на произведение множителей, зависящее только от подмножества этих переменных. В разделе 8.4 мы показали, что совместное распределение вероятностной модели — это основная интересующая нас величина, так как в нем содержится информация об априорных и апостериорных значениях, а также о правдоподобии. Тем не менее, совместное распределение само по себе может быть весьма сложным и ничего не говорит нам о структурных характеристиках вероятностной модели. Например, совместное распределение  $p(a, b, c)$  ничего не сообщает о независимости величин. Именно здесь в дело вступают графические модели. В этом разделе мы опираемся на концепции независимости и условной независимости, описанные в разделе 6.4.5.

В *графической модели* узлы являются случайными переменными. На рис. 8.9(а) все узлы соответствуют случайным переменным  $a, b, c$ . Ребра представляют

<sup>1</sup> Направленные графические модели также называются байесовскими сетями.

вероятностные отношения между переменными, например условные вероятности.



**Рис. 8.9.** Примеры направленных графических моделей

**ПРИМЕЧАНИЕ** Не всякое распределение можно представить в конкретной графической модели, выбранной нами. Этот вопрос подробно раскрывает Бишоп (Bishop, 2006). ◆

У вероятностных графических моделей есть некоторые удобные свойства:

- С их помощью просто визуализировать структуру вероятностной модели.
- Они позволяют проектировать или обосновывать новые виды статистических моделей.
- Достаточно рассмотреть только график, чтобы составить впечатление о свойствах модели, например ее условной независимости.
- Сложные вычисления для инференса и обучения в статистических моделях можно выразить в терминах графических манипуляций.

### 8.5.1. Семантика графов

*Направленные графические модели / байесовские сети* — это метод представления условных зависимостей в вероятностной модели. Они наглядно описывают условные вероятности и таким образом обеспечивают простой язык для представления сложных взаимосвязей. При модульном описании также снижается вычислительная сложность задачи. Направленные связи (стрелки) между двумя узлами (случайными переменными) указывают условные вероятности<sup>1</sup>. Например, стрелка между  $a$  и  $b$  на рис. 8.9(a) означает вероятность  $p(b | a)$  для  $b$  при условии события  $a$ .

<sup>1</sup> При дополнительных допущениях стрелки могут использоваться для указания причинных отношений (Pearl 2009).

Направленные графические модели можно вывести из совместных распределений, если нам что-либо известно об их факторизации.

### Пример 8.7

Рассмотрим совместное распределение

$$p(a, b, c) = p(c | a, b)p(b | a)p(a) \quad (8.29)$$

трех случайных переменных  $a, b, c$ . Из факторизации совместного распределения в (8.29) можно сделать некоторые выводы об отношении между случайными переменными:

- $c$  непосредственно зависит от  $a$  и  $b$ ;
- $b$  непосредственно зависит от  $a$ ;
- $a$  не зависит ни от  $b$ , ни от  $c$ .

Для факторизации в (8.29) мы получаем направленную графическую модель, показанную на рис. 8.9(а).

В принципе, можно построить соответствующую направленную графическую модель из факторизованного совместного распределения следующим образом:

1. Создать узел для всех случайных переменных.
2. Для каждого условного распределения добавить в граф прямую связь (стрелку) от узлов, соответствующих тем переменным, которыми обусловлено распределение.

Топология графа зависит от того, какой вариант факторизации был выбран для совместного распределения.

Мы обсуждали, как перейти от известной факторизации совместного распределения к соответствующей направленной графической модели. Теперь сделаем прямо противоположное — опишем, как извлечь совместное распределение множества случайных переменных из заданной графической модели.

### Пример 8.8

Рассмотрев графическую модель с рис. 8.9(б), воспользуемся двумя ее свойствами:

- Искомое нами совместное распределение  $p(x_1, \dots, x_5)$  получается из множества условий, по одному на каждый узел в графе. В данном конкретном примере нам понадобится пять условий.

- Каждое условие зависит только от узлов, родительских для конкретного узла в графе. Например,  $x_4$  будет обусловлено  $x_2$ .

Эти два свойства дают желаемую факторизацию совместного распределения

$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1)p(x_5)p(x_2 | x_5)p(x_3 | x_1, x_2)p(x_4 | x_2). \quad (8.30)$$

Вообще, совместное распределение  $p(\mathbf{x}) = p(x_1, \dots, x_K)$  задается как

$$p(\mathbf{x}) = \prod_{k=1}^K p(x_k | \text{Pa}_k), \quad (8.31)$$

где  $\text{Pa}_k$  означает «родительские узлы  $x_k$ ». Родительские узлы  $x_k$  — это узлы, стрелки от которых указывают на  $x_k$ .

Закончим этот подраздел конкретным экспериментом — «орел и решка». Рассмотрим схему Бернулли (пример 6.8), где вероятность, что результат  $x$  в данном случае будет решкой, равна

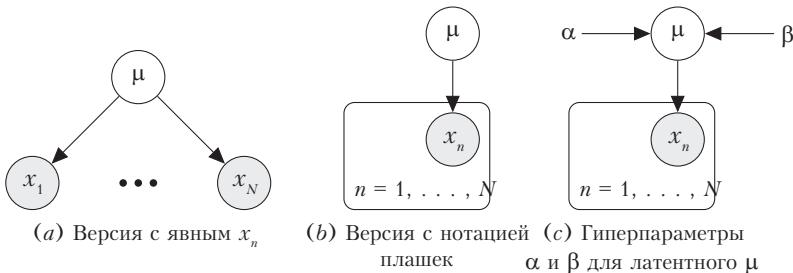
$$p(x | \mu) = \text{Ber}(\mu). \quad (8.32)$$

Теперь мы повторяем этот эксперимент  $N$  раз и наблюдаем его исходы  $\mathbf{x}_1, \dots, \mathbf{x}_N$ , так чтобы получить совместное распределение

$$p(x_1, \dots, x_N | \mu) = \prod_{n=1}^N p(x_n | \mu). \quad (8.33)$$

Выражение в правой части является произведением распределений Бернулли по каждому отдельному результату, поскольку эксперименты независимы. Как вы помните из раздела 6.4.5, статистическая независимость означает, что распределение факторизуется. Чтобы записать графическую модель для такой ситуации, нужно обозначить отличие между ненаблюдаемыми/латентными переменными и наблюдаемыми переменными. Графически наблюдаемые переменные обозначаются как затененные узлы, что позволяет нам получить такую графическую модель, как на рис. 8.10(a). Мы видим, что единственный параметр  $\mu$  одинаков для всех  $x_n, n = 1, \dots, N$ , поскольку результаты  $x_n$  распределены одинаково. Более компактная и при этом эквивалентная графическая модель для данной ситуации приведена на рис. 8.10(b), где используется нотация *плашек*. В плашке (рамке) все, что внутри (в данном случае, наблюдения  $x_n$ ), повторяется  $N$  раз. Следовательно, обе графические модели эквивалентны, но

нотация с плашками более компактна. Графические модели сразу же позволяют нам присвоить априорное распределение гиперпараметра для  $\mu$ . *Априорное распределение гиперпараметра* (hyperprior) — это второй уровень априорных распределений параметров над первым уровнем. На рис. 8.10(c) бета( $\alpha, \beta$ )-априорное распределение дается для латентной переменной  $\mu$ . Если считать  $\alpha$  и  $\beta$  детерминированными параметрами, а не случайными переменными, то обводить их в кружок не нужно.



**Рис. 8.10.** Графическое моделирование для серии испытаний Бернулли

### 8.5.2. Условная независимость и $d$ -разбиение

При помощи направленных графических моделей можно находить свойства условной независимости (раздел 6.4.5) для совместного распределения, просто посмотрев на график. Ключевую роль в данном случае играет концепция под названием *d-разбиение*.

Рассмотрим обычный направленный график, где  $\mathcal{A}, \mathcal{B}, \mathcal{C}$  — произвольные непересекающиеся множества узлов (объединение которых может быть меньше полного количества узлов в графике). Мы хотим убедиться, что утверждение о конкретном условном распределении « $\mathcal{A}$  условно независимо от  $\mathcal{B}$  при  $\mathcal{C}$ », обозначающееся как

$$\mathcal{A} \perp\!\!\!\perp \mathcal{B} \mid \mathcal{C}, \quad (8.34)$$

может быть представлено в виде направленного ациклического графа. Чтобы сделать это, рассмотрим все возможные пути (игнорирующие направление стрелок) от любого узла в  $\mathcal{A}$  к любым узлам в  $\mathcal{B}$ . Такой путь называют заблокированным, если он содержит хотя бы один узел, для которого верно одно из следующих утверждений:

- Стрелки на пути встречаются в узле либо голова в хвост, либо хвост в хвост, а узел относится к множеству  $\mathcal{C}$ .

- Стрелки встречаются в узле голова в голову, и ни этот узел, ни один из его потомков не относится к множеству  $\mathcal{C}$ .

Если все пути заблокированы, то говорят, что  $\mathcal{A}$   $d$ -отделено от  $\mathcal{B}$  из-за  $\mathcal{C}$ . В таком случае совместное распределение по всем переменным в графе будет удовлетворять  $\mathcal{A} \perp\!\!\!\perp \mathcal{B} \mid \mathcal{C}$ .

### Пример 8.9 (условная независимость)

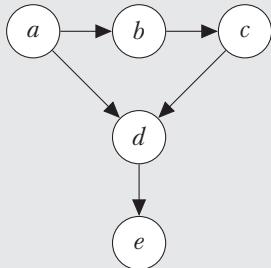
Рассмотрим графическую модель с рис. 8.11. Из рисунка видно, что

$$b \perp\!\!\!\perp d \mid a, c; \quad (8.35)$$

$$a \perp\!\!\!\perp c \mid b; \quad (8.36)$$

$$b \not\perp\!\!\!\perp d \mid c; \quad (8.37)$$

$$a \not\perp\!\!\!\perp c \mid b, e. \quad (8.38)$$



**Рис. 8.11.** Пример  $d$ -разделения

Направленные графические модели обеспечивают компактное представление вероятностных моделей. В главах 9–11 мы увидим примеры направленных графических моделей. Это представление, вместе с концепцией условной независимости, позволяет факторизовать соответствующие вероятностные модели в выражения, которые проще оптимизируются.

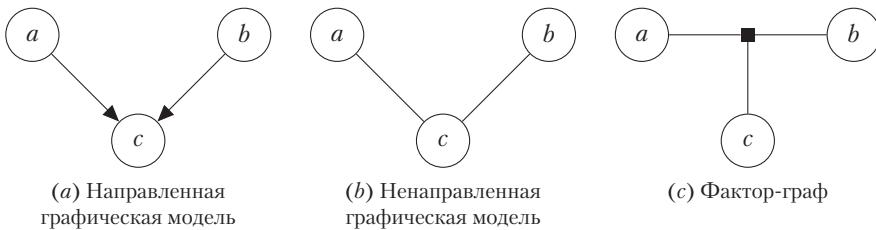
Графическое представление вероятностной модели позволяет наглядно показать влияние на структуру модели того или иного выбора при проектировании. Часто приходится делать обобщенные допущения о структуре модели. Эти допущения при моделировании (гиперпараметры) влияют на прогностическую производительность, но их нельзя выбрать напрямую при помощи тех подходов, что мы уже успели рассмотреть. Иные способы выбора структуры будут рассмотрены в разделе 8.6.

### 8.5.3. Дальнейшее чтение

Введение в вероятностные графические модели дает Бишоп (Bishop, 2006) в главе 8, а расширенное описание возможностей их применения и соответствующих алгоритмических следствий дается в книге Коллера и Фридмана (Koller and Friedman, 2009).

Существует три основных типа вероятностных графических моделей:

- *Направленные графические модели (байесовские сети)* (рис. 8.12(a)).
- *Ненаправленные графические модели (марковские случайные поля)* (рис. 8.12(b)).
- *Фактор-графы* (рис. 8.12(c)).



**Рис. 8.12.** Три типа графических моделей: (a) направленные графические модели (байесовские сети); (b) ненаправленные графические модели (марковские случайные поля); (c) фактор-графы

На основании графических моделей можно применять графовые алгоритмы для инференса и обучения, например путем локального обмена сообщениями. Области могут быть разными: построение рейтинга в онлайн-играх (Herbrich et al., 2007), компьютерное зрение (такие задачи, как сегментация изображений, семантическая разметка, слаживание изображений, восстановление изображений (Kittler and Föglein, 1984; Sucar and Gillies, 1994; Shotton et al., 2006; Szeliski et al., 2008)), теория кода (McEliece et al., 1998), решение систем линейных уравнений (Shental et al., 2008), итеративная байесовская оценка состояния при обработке сигналов (Bickson et al., 2007; Deisenroth and Mohamed, 2012).

Одна из тем, особенно важная в реальном применении, но не рассматриваемая в этой книге, — структурное прогнозирование (Bakir et al., 2007; Nowozin et al., 2014). Такие модели позволяют обрабатывать при помощи моделей машинного обучения структурные прогнозы, например последовательности, деревья, графы. Популярность моделей нейронных сетей располагает к использованию более гибких вероятностных моделей, благодаря чему находится множество вариантов полезного применения структурных моделей (Goodfellow et al., 2016, глава 16).

В последние годы вновь растет интерес к графическим моделям, поскольку они применимы для причинного вывода / инференса (Pearl, 2009; Imbens and Rubin, 2015; Peters et al., 2017; Rosenbaum, 2017).

## 8.6. ВЫБОР МОДЕЛИ

В машинном обучении часто приходится принимать высокоуровневые решения, которые критично влияют на производительность модели. Наш выбор (например, функциональная форма правдоподобия) влияет на количество и типы свободных параметров модели, а значит, также на гибкость и выразительность модели. Чем сложнее модель, тем более она гибкая в том смысле, что ее можно использовать для описания большего количества датасетов. Например, многочлен степени 1 (вида  $y = a_0 + a_1x$ ) может применяться лишь для описания линейных отношений между входными значениями  $x$  и наблюдениями  $y$ . Многочлен степени 2 также может описывать квадратичные отношения между исходными данными и наблюдениями<sup>1</sup>.

В таком случае можно подумать, что очень гибкие модели в целом предпочтительнее более простых моделей, поскольку гибкие модели выразительнее. Общая проблема в том, что во время обучения можно оценивать производительность модели и изучать ее параметры только на основании обучающего датасета. Однако производительность работы на этом наборе данных нас не очень интересует. В разделе 8.3 мы убедились, что оценка максимального правдоподобия может приводить к переобучению, особенно когда обучающая выборка небольшая. В идеале наша модель (также) хорошо работает на тестовом наборе (который в период обучения недоступен). Следовательно, нам нужны какие-то механизмы, которые помогут оценить, как модель *обобщает* неизвестные тестовые данные. *Выбор модели* нужен для решения именно этой проблемы.

### 8.6.1. Вложенная кросс-валидация

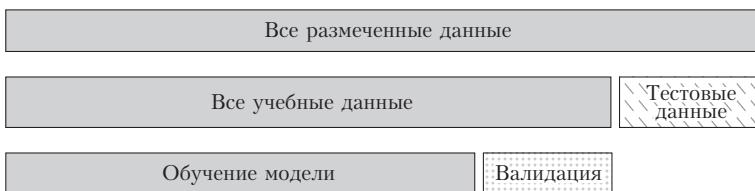
Мы уже рассматривали подход (кросс-валидация, раздел 8.2.4), который может использоваться для выбора модели. Как вы помните, кросс-валидация позволяет оценить ошибку обобщения, многократно разбивая датасет на два набора данных, обучающий и валидационный. Применим эту идею еще раз, то есть для каждого разделения снова будем проводить кросс-валидацию. Иногда такой подход называется *вложенной кросс-валидацией* (рис. 8.13). Внутренний уровень используется для оценки производительности конкретной выбранной модели или гиперпараметра внутреннего валидационного датасета. Внешний уровень

<sup>1</sup> Многочлен  $y = a_0 + a_1x + a_2x^2$  также может описывать линейные функции, если задать  $a_2 = 0$ , то есть он строго более выразителен, чем многочлен первого порядка.

используется для оценки обобщающей способности наилучшей модели, которая была выбрана во внутреннем цикле. Во внутреннем цикле можно тестировать различные варианты выбранных моделей и гиперпараметров. Для различения этих двух уровней набор, который используется для оценки производительности обобщения, часто называется *тестовым*, а тот, что используется для выбора наилучшей модели — *валидационным*. Внутренний цикл оценивает ожидаемое значение ошибки обобщения для данной модели (8.39), аппроксимируя погрешность путем применения эмпирической ошибки к валидационному набору, то есть

$$\mathbb{E}_{\mathcal{V}}[\mathbf{R}(\mathcal{V}|M)] \approx \frac{1}{K} \sum_{k=1}^K \mathbf{R}(\mathcal{V}^{(k)}, M), \quad (8.39)$$

где  $\mathbf{R}(\mathcal{V}|M)$  — это эмпирический риск (например, среднеквадратичная ошибка) валидационного набора  $\mathcal{V}$  для модели  $M$ . Эту процедуру повторим для всех моделей и выберем из них наиболее производительную. Обратите внимание: кросс-валидация дает нам не только ожидаемую ошибку обобщения, но и позволяет получить статистику высшего порядка, например стандартную ошибку<sup>1</sup> — примерный показатель того, насколько неопределенной является средняя оценка. Когда модель выбрана, можно оценить окончательную производительность тестового набора.



**Рис. 8.13.** Вложенная кросс-валидация. Выполняется два уровня  $K$ -кратной кросс-валидации

## 8.6.2. Выбор байесовской модели

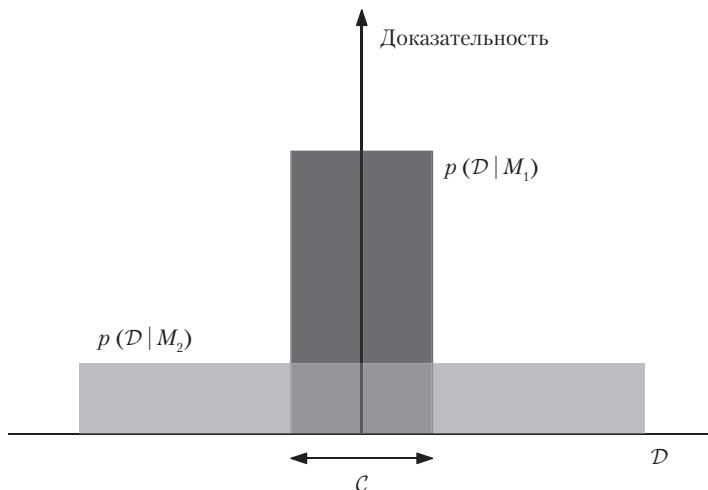
Существует множество подходов к выбору модели, некоторые из них рассмотрены в этом разделе. В целом, все они нацелены на достижение компромисса между сложностью модели и прогнозируемостью данных. Предполагается, что чем проще модель, тем менее она подвержена переобучению, поэтому цель вы-

<sup>1</sup> Стандартная ошибка определяется как  $\frac{\sigma}{\sqrt{K}}$ , где  $K$  — это количество экспериментов, а  $\sigma$  — стандартное отклонение риска для каждого эксперимента.

бора модели — найти простейшую модель, которая вполне хорошо описывает имеющиеся данные. Эта концепция также называется *бритва Оккама*.

**ПРИМЕЧАНИЕ** Если подойти к выбору модели как к задаче проверки гипотезы, то мы будем искать простейшую гипотезу, согласующуюся с данными (Murphy, 2012). ◆

Можно попробовать отдать приоритет тем моделям, которые тяготеют к простоте. Правда, делать это необязательно: «автоматическая бритва Оккама» количественно воплощена в применении байесовского вероятностного подхода (Smith and Spiegelhalter 1980; Jeffreys and Berger, 1992; MacKay, 1992). Рисунок 8.14, сделанный по образцу из книги МакКея (MacKay, 2003), позволяет интуитивно понять, почему сложные и очень выразительные модели — порой не самый лучший выбор для моделирования датасета  $\mathcal{D}$ . Давайте представим, что по горизонтальной оси представлено пространство всех возможных наборов данных  $\mathcal{D}$ . Если нас интересует апостериорная вероятность  $p(M_i | \mathcal{D})$  для моделей  $M_i$ , на основе данных  $\mathcal{D}$ , то можно воспользоваться теоремой Байеса. Предполагая, что мы используем однородное априорное  $p(M)$  для всех моделей, теорема Байеса присваивает моделям вознаграждение пропорционально тому, насколько хорошо они позволили спрогнозировать возникшие данные. Прогноз



**Рис. 8.14.** Байесовский инференс — это воплощение бритвы Оккама.

По горизонтальной оси представлено пространство всех возможных множеств данных  $\mathcal{D}$ . Доказательность (вертикальная ось) позволяет оценить, насколько хорошо модель прогнозирует доступные данные. Поскольку  $p(\mathcal{D} | M_i)$  должно интегрироваться до 1, мы выбираем модель с наибольшим значением по оси доказательности. Иллюстрация взята из книги МакКея (MacKay, 2003)

данных при наличии модели  $M_i$ ,  $p(\mathcal{D} | M_i)$ , называется *доказательностью*  $M_i$ . Простая модель  $M_1$  может прогнозировать лишь небольшое количество датасетов, что демонстрируется на примере  $p(\mathcal{D} | M_1)$ ; более производительная модель  $M_2$ , имеющая, например, больше свободных параметров, чем  $M_1$ , позволяет прогнозировать более разнообразные датасеты<sup>1</sup>. Однако это означает, что  $M_2$  не прогнозирует датасеты в области  $C$ , равно как и  $M_1$ . Предположим, что двум моделям присвоены одинаковые априорные вероятности. Затем, если датасет приходится на область  $C$ , менее мощная модель  $M_1$  дает более качественные вероятности.

Выше в этой главе утверждалось, что модели должны объяснять данные, то есть должен быть способ сгенерировать данные на основе имеющейся модели. Кроме того, если модель как следует обучилась на данных, то мы рассчитываем, что сгенерированные ею данные должны быть похожи на эмпирические. Для этого полезно сформулировать проблему выбора модели как задачу иерархического инференса, что позволяет нам вычислить апостериорное распределение для разных моделей.

Рассмотрим конечное количество моделей  $M = \{M_1, \dots, M_K\}$ , где каждая модель обладает параметрами  $\theta_k$ . При *байесовском выборе модели* априорное значение  $p(M)$  присваивается множеству моделей. Соответствующий *генеративный процесс*, позволяющий получить данные на основе этой модели, таков:

$$M_k \sim p(M); \quad (8.40)$$

$$\theta_k \sim p(\theta | M_k); \quad (8.41)$$

$$\mathcal{D} \sim p(\mathcal{D} | \theta_k), \quad (8.42)$$

как показано на рис. 8.15. Имея обучающий набор данных  $\mathcal{D}$ , мы применяем теорему Байеса и рассчитываем апостериорное распределение для моделей как

$$p(M_k | \mathcal{D}) \propto p(M_k) p(\mathcal{D} | M_k). \quad (8.43)$$

Обратите внимание: это апостериорное распределение более не зависит от параметров модели  $\theta_k$ , так как в байесовской постановке задачи они были сокращены методом интегрирования, поскольку

$$p(\mathcal{D} | M_k) = \int p(\mathcal{D} | \theta_k) p(\theta_k | M_k) d\theta_k, \quad (8.44)$$

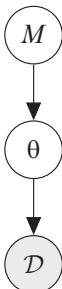
---

<sup>1</sup> Прогнозы количественно выражаются нормализованным распределением вероятностей  $\mathcal{D}$ , то есть результат должен интегрироваться/суммироваться до 1.

где  $p(\theta_k | M_k)$  – это априорное распределение параметров  $\theta_k$  модели  $M_k$ . Член (8.44) называется *доказательностью модели*, или *маргинальным правдоподобием*. Из аппостериорного значения в (8.43) мы определяем оценку МАР

$$M^* = \arg \max_{M_k} p(M_k | \mathcal{D}). \quad (8.45)$$

При однородном априорном  $p(M_k) = \frac{1}{K}$ , дающем всем моделям равную (априорную) вероятность, определение МАР сводится к выбору модели, обладающей максимальной доказательностью (8.44).



**Рис. 8.15.** Иллюстрация иерархического генеративного процесса при байесовском выборе модели. Устанавливаем априорное значение  $p(M)$  для множества моделей. Для каждой модели существует распределение  $p(\theta | M)$  для соответствующих параметров. Оно используется для генерации данных  $\mathcal{D}$

**ПРИМЕЧАНИЕ** Существует важное отличие между правдоподобием и маргинальным правдоподобием (доказательностью). Тогда как правдоподобие может страдать из-за переобучения, маргинального правдоподобия это обычно не касается, так как параметры модели были исключены (то есть нам больше не требуется подстраивать ее с учетом этих параметров). Более того, маргинальное правдоподобие автоматически воплощает компромисс между сложностью модели и прогнозируемостью данных (бритва Оккама). ◆

### 8.6.3. Коэффициент Байеса для сравнения моделей

Рассмотрим задачу сравнения двух вероятностных моделей  $M_1, M_2$  при наличии набора данных  $\mathcal{D}$ . Если вычислить апостериорные вероятности  $p(M_1 | \mathcal{D})$  и  $p(M_2 | \mathcal{D})$ , то можно рассчитать и соотношение апостериорных

$$\underbrace{\frac{p(M_1 | \mathcal{D}) p(M_1)}{p(M_2 | \mathcal{D})}}_{\text{Апостериорные шансы}} = \frac{\frac{p(\mathcal{D} | M_1) p(M_1)}{p(\mathcal{D})}}{\frac{p(\mathcal{D} | M_2) p(M_2)}{p(\mathcal{D})}} = \frac{\underbrace{p(M_1) p(\mathcal{D} | M_1)}_{\text{Априорные шансы}}}{\underbrace{p(M_2) p(\mathcal{D} | M_2)}_{\text{Априорные шансы}}} = \underbrace{\frac{p(M_1)}{p(M_2)}}_{\text{Коэффициент Байеса}}. \quad (8.46)$$

Отношение апостериорных вероятностей также называется *апостериорными шансами*. Первая дробь в правой части выражения (8.46), *априорные шансы*, позволяет измерить, насколько  $M_1$  предпочтительнее  $M_2$  согласно нашим априорным (исходным) представлениям. Отношение маргинальных правдоподобий (вторая дробь в правой части) называется *коэффициентом Байеса* и позволяет измерить, насколько успешнее  $M_1$  по сравнению с  $M_2$  прогнозирует  $\mathcal{D}$ .

**ПРИМЕЧАНИЕ** Согласно парадоксу Джеффриса — Линдли, «коэффициент Байеса всегда в пользу более простой модели, так как вероятность данных в сложной модели с диффузным априорным распределением будет очень невелика» (Murphy, 2012). Здесь диффузным называется такое априорное распределение, которое не отдает предпочтения конкретным моделям, то есть при таких априорных данных много моделей, которые кажутся вполне возможными.



Если выбрать однородное априорное для множества моделей, то член априорных шансов в (8.46) будет равен 1, то есть апостериорные шансы будут равны отношению маргинальных правдоподобий (коэффициенту Байеса)

$$\frac{p(M_1 | \mathcal{D})}{p(M_2 | \mathcal{D})}. \quad (8.47)$$

Если коэффициент Байеса больше 1, то мы выбираем модель  $M_1$ , в противном случае —  $M_2$ . По аналогии с частотной статистикой, здесь существуют рекомендации по величине отношения, которые нужно учесть, прежде чем судить о «значимости» результата (Jeffreys, 1961).

**ПРИМЕЧАНИЕ** Маргинальное правдоподобие играет важную роль при выборе модели. Необходимо вычислить коэффициенты Байеса (8.46) и апостериорные распределения для моделей (8.43).

К сожалению, для расчета маргинального правдоподобия требуется брать интеграл (8.44). Такая интеграция обычно аналитически невычислимая, поэтому приходится довольствоваться приемами приближения, например численным интегрированием (Stoer and Burlirsch, 2002), стохастическими приближениями с применением метода Монте-Карло (Murphy, 2012) или байесовскими методами Монте-Карло (O'Hagan, 1991; Rasmussen and Ghahramani, 2003).

Однако в некоторых частных случаях такие задачи решаемы. В разделе 6.6.1 мы обсуждали сопряженные модели. Если выбрать для сопряженных параметров априорное распределение  $p(\theta)$ , то можно вычислить маргинальное правдоподобие в замкнутой форме. В главе 9 именно это мы сделаем в контексте линейной регрессии.



В этой главе мы познакомились с базовыми концепциями машинного обучения. В оставшейся части книги мы рассмотрим, как три различные варианта обучения, представленные в разделах 8.2–8.4, применяются при работе с четырьмя столпами МО (регрессия, снижение размерности, оценка плотности и классификация).

### 8.6.4. Дальнейшее чтение

В начале раздела мы упоминали, что при высокоуровневом моделировании принимаются решения, выбор которых влияет на производительность модели. Среди примеров такого рода:

- степень многочлена при подготовке регрессии;
- количество компонентов в смешанной модели;
- архитектура (глубокой) нейронной сети;
- тип ядра в методе опорных векторов;
- размерность латентного пространства при анализе главных компонент;
- скорость (план) обучения в оптимизационном алгоритме.

Расмуссен и Гахрамани (Rasmussen and Ghahramani, 2001) показали, что необходимо добавлять ограничения (штраф) ряду параметров модели, однако с точки зрения сложности функций модель активна. Они также показали, что автоматическая бритва Оккама применима и для байесовских непараметрических моделей с множеством параметров, например для гауссовых процессов<sup>1</sup>.

Если сосредоточиться на оценке максимального правдоподобия, то можно использовать ряд эвристических приемов, действующих при подборе модели и снижающих вероятность переобучения. Они называются информационными критериями, и мы выбираем модель с наивысшим значением. Таков информационный критерий Акаике (Akaike information criterion, AIC) (Akaike, 1974), позволяющий скорректировать смещение оценки максимального правдоподобия:

$$\log p(\mathbf{x} | \theta) - M; \quad (8.48)$$

это делается путем добавления штрафного члена, позволяющего смягчить переобучение сравнительно сложных моделей с большим количеством параметров.

---

<sup>1</sup> В параметрических моделях количество параметров часто связано со сложностью класса моделей.

Здесь  $M$  — количество параметров модели. Критерий AIC оценивает относительную информацию, теряемую заданной моделью.

*Байесовский информационный критерий* (Bayesian information criterion, BIC) (Schwarz, 1978)

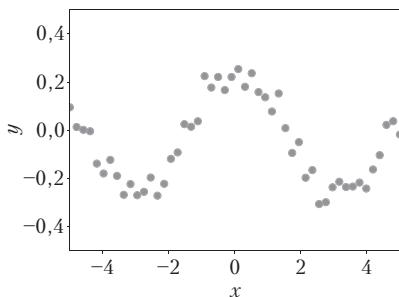
$$\log p(\mathbf{x}) = \log \int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta} \approx \log p(\mathbf{x}|\boldsymbol{\theta}) - \frac{1}{2}M \log N \quad (8.49)$$

может использоваться для экспоненциальных распределений семейств. Здесь  $N$  — это количество точек данных, а  $M$  — число параметров. BIC штрафует сложность моделей активнее, чем AIC.

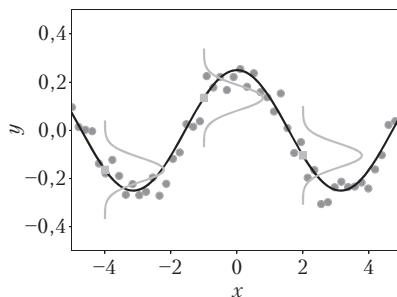
# 9

## Линейная регрессия

В данной главе мы применим математические понятия, изученные в главах 2 и 5–7, к решению задачи линейной регрессии (являющейся частным случаем задачи аппроксимации кривой). Цель задачи *регрессии* — найти функцию  $f$ , сопоставляющую входным точкам  $\mathbf{x} \in \mathbb{R}^D$  значения  $f(\mathbf{x}) \in \mathbb{R}$ . Предполагается, что нам дан обучающий набор входных значений  $x_n$  и соответствующих зашумленных наблюдений  $y_n = f(x_n) + \varepsilon$ , где  $\varepsilon$  — случайная величина, описывающая возникающий при наблюдениях/измерениях шум, а также какие-то не учтенные в модели процессы (мы не будем их рассматривать здесь). В данной главе мы предполагаем, что шум имеет гауссово распределение с нулевым средним. Нашей задачей будет найти функцию, которая не только хорошо моделирует обучающий набор данных, но и успешно предсказывает значения функции, когда входная точка не принадлежит обучающей выборке (см. главу 8). Пример задачи регрессии показан на рис. 9.1.



(a) Проблема регрессии: наблюдаемые зашумленные значения функции, из которых мы хотим вывести базовую функцию, породившую данные



(b) Решение регрессии: функция, которая могла бы сгенерировать данные (темная линия), с указанием шума измерения функции на соответствующих входных точках (светлые линии)

Рис. 9.1. (a) Набор данных; (b) возможное решение задачи регрессии

Типичная постановка задачи регрессии дана на рис. 9.2(a): для некоторых входных точек  $x_n$  мы наблюдаем (зашумленные) значения  $y_n = f(x_n) + \epsilon$ . Задача — восстановить функцию  $f$ , породившую данные и хорошо обобщающуюся на значения функции на новых входных данных. Возможное решение дано на рис. 9.2(b), где также показаны три распределения со средними в значениях  $f(x)$ , представляющие шум.



**Рис. 9.2.** Пример линейной регрессии. (a) Примеры линейных функций; (b) обучающая выборка; (c) оценка максимального правдоподобия

Регрессия играет важнейшую роль в машинном обучении, задачи регрессии появляются в самых разных теоретических и прикладных областях, таких как анализ временных рядов (например, системы распознавания), управление и робототехника (например, обучение с подкреплением), оптимизация (например, линейный поиск, глобальная оптимизация) и глубокое обучение (например, компьютерные игры, распознавание речи и изображений, автоматическая аннотация видео). Регрессия также является важным шагом в некоторых алгоритмах классификации. Нахождение функции регрессии требует решения многих задач, в частности:

- **Выбор модели (типа функции регрессии) и ее параметризации.** Какой класс функций (например, многочлены) подойдет для построения модели при имеющихся данных, и какие конкретные параметры (например, степень многочлена) выбрать? При выборе модели, как мы уже обсуждали в разделе 8.6, можно сравнить разные варианты и выбрать самый простой и достаточно хорошо объясняющий обучающие данные<sup>1</sup>.
- **Нахождение подходящих параметров.** Как, выбрав тип функции регрессии, найти хорошие параметры модели? Необходимо рассмотреть разные целевые функции / функции потерь (определяющие «хорошие» параметры) и алгоритмы оптимизации, используемые для минимизации потерь.

<sup>1</sup> Обычно при выборе модели задают и тип шума, но мы в этой главе считаем шум гауссовым.

- **Переобучение и выбор модели (model selection)**<sup>1</sup>. Переобучением называется ситуация, когда функция регрессии «слишком хорошо» запоминает обучающие данные, но не обобщается на незнакомые тестовые данные. Переобучение обычно происходит, если выбранная модель (или ее параметры) обладает излишней гибкостью и выразительностью (раздел 8.6). Мы рассмотрим причины этого и обсудим способы уменьшить эффект переобучения для линейной регрессии.
- **Связь между функцией потерь и априорными предположениями о параметрах модели.** Выбор функции потерь часто обусловлен некоторой вероятностной моделью. Мы рассмотрим, как функция потерь связана с нашими исходными предположениями.
- **Моделирование неопределенности.** В любой прикладной задаче нам доступен лишь ограниченный (хотя потенциально он может быть большим) набор обучающих данных для выбора модели и ее параметров. Если этот ограниченный набор не покрывает все возможные ситуации, мы можем задаться вопросом об описании неопределенности параметров (как меры уверенности в предсказаниях модели на тестовом множестве). Чем меньше обучающая выборка, тем важнее учитывать неопределенность. При правильном моделировании неопределенности предсказания модели снабжаются доверительными границами.

В данной главе мы применим математический инструментарий глав 3 и 5–7 к решению задач линейной регрессии. Мы обсудим использование оценок максимального правдоподобия (MLE) и апостериорного максимума (MAP) для нахождения оптимальных параметров модели. Используя эти оценки параметров, мы вкратце рассмотрим ошибки обобщения и переобучение. Ближе к концу главы мы обучим байесовскую линейную регрессию, которая позволяет нам рассуждать о параметрах модели на более высоком уровне, тем самым устранив некоторые проблемы оценки методом максимального правдоподобия и апостериорного максимума.

## 9.1. ПОСТАНОВКА ЗАДАЧИ

Из-за наличия в наблюдениях шума нам придется обратиться к вероятностному подходу и явно смоделировать шум с помощью функции правдоподобия. В частности, в этой главе мы рассмотрим задачу регрессии с функцией правдоподобия

$$p(y | \mathbf{x}) = \mathcal{N}(y | f(\mathbf{x}), \sigma^2). \quad (9.1)$$

---

<sup>1</sup> Это не то же самое, что «выбор модели» из первого пункта. Model selection — отдельная задача выбора модели из набора «моделей-кандидатов». — Примеч. ред.

Здесь  $\mathbf{x} \in \mathbb{R}^D$  — входные данные, а  $y \in \mathbb{R}$  — зашумленные значения функции. Как было сказано в 9.1, зависимость между  $x$  и  $y$  задается формулой

$$y = f(x) + \varepsilon, \quad (9.2)$$

где  $\varepsilon \sim \mathcal{N}(0, \sigma^2)$  — независимый одинаково распределенный гауссов шум измерений со средним 0 и дисперсией  $\sigma^2$ . Наша цель — найти функцию, которая близка к неизвестной функции  $f$ , породившей наши данные, и легко обобщается.

В данной главе мы сосредоточимся на параметрических моделях, то есть выберем параметризованную функцию и найдем параметры  $\theta$ , которые хорошо подходят для моделирования данных. Пока предположим, что дисперсия шума  $\sigma^2$  известна, и сконцентрируемся на нахождении параметров модели  $\theta$ . Линейная регрессия представляет собой частный случай, когда параметры  $\theta$  входят в формулу линейно. Примером линейной регрессии может служить

$$p(y | \mathbf{x}, \theta) = \mathcal{N}(y | \mathbf{x}^T \theta, \sigma^2) \Leftrightarrow \quad (9.3)$$

$$\Leftrightarrow y = \mathbf{x}^T \theta + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2), \quad (9.4)$$

где  $\theta \in \mathbb{R}^D$  — искомые параметры. Класс функций, заданных формулой (9.4), — это прямые, проходящие через начало координат. В формуле (9.4) мы берем параметризацию  $f(\mathbf{x}) = \mathbf{x}^T \theta$ .

*Правдоподобие* в (9.3) — это плотность распределения  $y$  при условии  $\mathbf{x}^T \theta$ . Заметим, что единственным источником неопределенности будет шум в наблюдениях (так как в (9.3) предполагается, что  $\mathbf{x}$  и  $\theta$  известны). Без шума зависимость между  $\mathbf{x}$  и  $y$  была бы детерминированной, и формула (9.3) задавала бы дельта-функцию Дирака<sup>1</sup>.

### Пример 9.1

При  $x, \theta \in \mathbb{R}$  линейная регрессионная модель (9.4) описывает прямые (линейные функции), а параметр  $\theta$  равен тангенсу угла наклона прямой. На рис. 9.2(a) показаны примеры функций, соответствующих различным значениям  $\theta$ .

<sup>1</sup> Дельта-функция Дирака равна нулю всюду, кроме одной точки. Ее интеграл равен нулю. Можно считать дельта-функцию гауссовой (предельный случай при стремящейся к нулю дисперсии).

Линейная регрессионная модель, заданная (9.3) и (9.4), линейна не только по параметрам<sup>1</sup>, но и по входным значениям  $\mathbf{x}$ . На рис. 9.2(a) показаны примеры таких функций. Далее мы поймем, что  $y = \phi^T(\mathbf{x})\theta$  для нелинейного преобразования  $\phi$  — на самом деле также линейная регрессионная модель, так как это понятие относится к моделям, линейным по параметрам, то есть задающимся линейными комбинациями входных признаков. Здесь роль признаков играют преобразования  $\phi(\mathbf{x})$  входных значений  $\mathbf{x}$ .

В дальнейшем мы более подробно обсудим, как находить подходящие параметры  $\theta$  и оценивать, насколько хорошо они подобраны. Пока мы будем считать, что дисперсия шума  $\sigma^2$  известна.

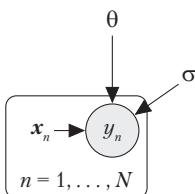
## 9.2. ОЦЕНКА ПАРАМЕТРОВ

Пусть нам поставлена задача линейной регрессии (9.4) и дана *обучающая выборка*  $\mathcal{D} := \{(\mathbf{x}^1, y^1), \dots, (\mathbf{x}_N, y_N)\}$ , состоящая из  $N$  входных значений  $\mathbf{x}_n \in \mathbb{R}^D$  и соответствующих наблюдений  $y_n \in \mathbb{R}$ ,  $n = 1, \dots, N$ . Графическая модель показана на рис. 9.3. Заметим, что  $y_i$  и  $y_j$  условно независимы относительно  $\mathbf{x}_i$ ,  $\mathbf{x}_j$ , так что правдоподобие выражается как

$$p(\mathcal{Y} | \mathcal{X}, \theta) = p(y_1, \dots, y_N | \mathbf{x}_1, \dots, \mathbf{x}_N, \theta) = \quad (9.5a)$$

$$= \prod_{n=1}^N p(y_n | \mathbf{x}_n, \theta) = \prod_{n=1}^N \mathcal{N}(y_n | \mathbf{x}_n^T \theta, \sigma^2), \quad (9.5b)$$

где  $\mathcal{X} := \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  и  $\mathcal{Y} := \{y_1, \dots, y_N\}$  — множество входных значений обучающего набора и соответствующих наблюдений. Правдоподобие, как и каждый из сомножителей  $p(y_n | \mathbf{x}_n, \theta)$ , из-за распределения шума будет гауссовым, см. (9.3).



**Рис. 9.3.** Графическая вероятностная модель для линейной регрессии. Наблюдаемые значения закрашены, детерминированные/известные величины не обведены кружком

Далее мы обсудим нахождение оптимальных параметров  $\theta_* \in \mathbb{R}^D$  для линейной регрессионной модели (9.4). Найдя параметры  $\theta^*$ , мы можем подставить эту

<sup>1</sup> Моделями линейной регрессии называют модели, линейные по параметрам.

оценку в (9.4) и предсказывать значения функции, так что на входном тестовом значении  $\mathbf{x}_*$  распределением  $y_*$  будет

$$p(y_* | \mathbf{x}_*, \theta^*) = \mathcal{N}(y_* | \mathbf{x}_*^T \theta, \sigma^2). \quad (9.6)$$

Далее мы рассмотрим оценку параметров методом максимального правдоподобия, эту тему мы уже немного затрагивали в разделе 8.3.

### 9.2.1. Оценка максимального правдоподобия

При нахождении оптимальных параметров  $\theta_{ML}$  часто используют *метод максимального правдоподобия*, при котором выбираются такие  $\theta_{ML}$ , которые максимизируют правдоподобие (9.5b). Можно понимать это как максимизацию вероятности обучающих данных при заданных параметрах модели. Параметры максимального правдоподобия ищутся как

$$\theta_{ML} = \arg \max_{\theta} p(\mathcal{Y} | \mathcal{X}, \theta). \quad (9.7)$$

**ПРИМЕЧАНИЕ** Правдоподобие  $p(y | \mathbf{x}, \theta)$  не является распределением  $\theta$ . Это функция от  $\theta$ , интеграл от которой не равен 1 (то есть не нормализованная) и может вообще не существовать. Однако правдоподобие (9.7) является нормализованным распределением  $\mathbf{y}$ . ◆

Чтобы найти искомые  $\theta_{ML}$ , максимизирующие правдоподобие, обычно применяют градиентный подъем (градиентный спуск относительно взятого с противоположным знаком правдоподобия). В рассматриваемом нами здесь случае линейной регрессии, однако, существует решение в явном виде, так что градиентный спуск не нужен. На практике вместо максимизации самого правдоподобия мы берем его логарифм<sup>1</sup> с противоположным знаком и ищем минимум получившейся функции.

**ПРИМЕЧАНИЕ** Поскольку правдоподобие в (9.5b) является произведением  $N$  гауссовых распределений, логарифмическое преобразование оказывается полезным, так как (a) не вызывает проблем потери значимости и (б) упрощает дифференцирование.

Потерей значимости называется ситуация, возникающая, например, при перемножении  $N$  вероятностей, где  $N$  – число точек с данными, поскольку невозможно представить очень маленькие числа, вроде  $10^{-256}$ . Логарифмическое преобразование также превращает произведение в сумму логарифмов вероят-

---

<sup>1</sup> Так как логарифм – монотонная строго возрастающая функция, оптимум  $f$  и оптимум  $\log f$  достигаются в одной точке.

ностей, так что градиент произведения  $N$  сомножителей раскладывается в сумму градиентов, и нет необходимости много раз применять правило произведения (5.46).  $\diamond$

Чтобы найти оптимальные параметры  $\theta_{ML}$  нашей задачи линейной регрессии, мы максимизируем взятый с противоположным знаком логарифм правдоподобия<sup>1</sup>

$$-\log p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}) = -\log \prod_{n=1}^N p(y_n | \mathbf{x}_n, \boldsymbol{\theta}) = -\sum_{n=1}^N \log p(y_n | \mathbf{x}_n, \boldsymbol{\theta}), \quad (9.8)$$

где воспользуемся тем, что правдоподобие (9.5b) благодаря независимости точек обучающей выборки раскладывается в произведение.

В модели линейной регрессии (9.4) правдоподобие будет гауссовым (из-за добавления гауссова шума), так что мы получаем

$$\log p(y_n | \mathbf{x}_n, \boldsymbol{\theta}) = -\frac{1}{2\sigma^2} (y_n - \mathbf{x}_n^\top \boldsymbol{\theta})^2 + \text{const}, \quad (9.9)$$

где константа содержит все слагаемые, не зависящие от  $\boldsymbol{\theta}$ . Подставив (9.9) в (9.8), получаем (проигнорировав константные слагаемые)

$$\mathcal{L}(\boldsymbol{\theta}) := \frac{1}{2\sigma^2} \sum_{n=1}^N (y_n - \mathbf{x}_n^\top \boldsymbol{\theta})^2 = \quad (9.10a)$$

$$= \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) = \frac{1}{2\sigma^2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|^2, \quad (9.10b)$$

где  $\mathbf{X} := [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times D}$  — матрица признаков для обучающего набора, а  $\mathbf{y} := [y_1, \dots, y_N]^\top \in \mathbb{R}^N$  — вектор целевых параметров для обучающей выборки. Заметим, что  $n$ -я строка в матрице признаков  $\mathbf{X}$  соответствует входной точке  $\mathbf{x}_n$ . В (9.10b) мы пользовались тем, что сумма квадратов ошибок<sup>2</sup>  $y_n$  и соответствующих предсказаний модели  $\mathbf{x}_n^\top \boldsymbol{\theta}$  равна квадрату расстояния между  $\mathbf{y}$  и  $\mathbf{X}\boldsymbol{\theta}$ .

Формула (9.10b) задает в явном виде функцию, которую мы хотим оптимизировать. Относительно  $\boldsymbol{\theta}$  (9.10b) квадратична. Это означает, что у  $\mathcal{L}$  существует единственный глобальный минимум  $\theta_{ML}$ . Мы можем найти его, взяв градиент  $\mathcal{L}$ , приравняв его нулю и решив полученное уравнение относительно  $\boldsymbol{\theta}$ .

<sup>1</sup> Отрицательное логарифмическое правдоподобие также называют функцией ошибок.

<sup>2</sup> Квадрат ошибки часто используется для измерения расстояния. В разделе 3.1 мы обсуждали, что  $\|\mathbf{x}\|^2 = \mathbf{x}^\top \mathbf{x}$  при выборе стандартного скалярного произведения.

Используя результаты главы 5, найдем градиент  $\mathcal{L}$  по параметрам:

$$\frac{d\mathcal{L}}{d\theta} = \frac{d}{d\theta} \left( \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}\theta)^T (\mathbf{y} - \mathbf{X}\theta) \right) = \quad (9.11a)$$

$$= \frac{1}{2\sigma^2} \frac{d}{d\theta} (\mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \mathbf{X}\theta + \theta^T \mathbf{X}^T \mathbf{X}\theta) = \quad (9.11b)$$

$$= \frac{1}{\sigma^2} (-\mathbf{y}^T \mathbf{X} + \theta^T \mathbf{X}^T \mathbf{X}) \in \mathbb{R}^{1 \times D}. \quad (9.11c)$$

Оценку максимального правдоподобия  $\theta_{ML}$  ищем как решение уравнения  $d\mathcal{L}/d\theta = \mathbf{0}^T$  (необходимое условие экстремума):

$$\frac{d\mathcal{L}}{d\theta} = \mathbf{0}^T \stackrel{(9.11c)}{\Leftrightarrow} \theta_{ML}^T \mathbf{X}^T \mathbf{X} = \mathbf{y}^T \mathbf{X} \Leftrightarrow \quad (9.12a)$$

$$\Leftrightarrow \theta_{ML}^T = \mathbf{y}^T \mathbf{X} (\mathbf{X}^T \mathbf{X})^{-1} \Leftrightarrow \quad (9.12b)$$

$$\Leftrightarrow \theta_{ML} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}. \quad (9.12c)$$

Первое уравнение можно домножать справа на  $(\mathbf{X}^T \mathbf{X})^{-1}$ , поскольку при  $\text{rk}(\mathbf{X}) = D$  (где  $\text{rk}(\mathbf{X})$  – ранг  $\mathbf{X}$ )  $\mathbf{X}^T \mathbf{X}$  положительно определена<sup>1</sup>.

**ПРИМЕЧАНИЕ** Равенство градиента нулю является необходимым и достаточным условием глобального минимума, поскольку матрица Гессе  $\nabla_\theta^2 \mathcal{L}(\theta) = \mathbf{X}^T \mathbf{X} \in \mathbb{R}^{D \times D}$  положительно определена. ◆

**ПРИМЕЧАНИЕ** Поиск параметров максимального правдоподобия из (9.12c) требует решения системы линейных уравнений  $\mathbf{A}\theta = \mathbf{b}$ , где  $\mathbf{A} = \mathbf{X}^T \mathbf{X}$  и  $\mathbf{b} = \mathbf{X}^T \mathbf{y}$ . ◆

### Пример 9.2 (проведение прямой)

Обратимся к рис. 9.2. Мы хотим методом максимального правдоподобия по имеющимся данным построить прямую  $f(x) = \theta x$ , где  $\theta$  – неизвестный нам тангенс угла наклона. Примеры функций из выбранного нами класса моделей (прямые) показаны на рис. 9.2(a). По данным с рис. 9.2(b) мы находим оценку максимального правдоподобия для параметра  $\theta$  с помощью (9.12c). Линейная функция максимального правдоподобия изображена на рис. 9.2(c).

<sup>1</sup> Мы считаем, что данные не повторяются и  $\text{rk}(\mathbf{X}) = D$  при  $N \geq D$ , то есть параметров не больше, чем точек с данными.

## Оценка максимального правдоподобия с преобразованием признаков

До сих пор мы рассматривали линейную регрессию, описанную в разделе 9.4, которая позволяла нам с помощью оценки максимального правдоподобия проводить прямую. Однако в случае более интересных данных оказывается, что прямые не очень выразительны. К счастью, линейная регрессия позволяет нам находить и нелинейные функции. Понятие «линейность» относится лишь к линейной зависимости параметров, тогда как над входными значениями  $\mathbf{x}$  можно делать произвольные нелинейные преобразования  $\phi(\mathbf{x})$ , а затем строить из них линейную комбинацию. Соответствующая модель выглядит так:

$$\begin{aligned} p(y|\mathbf{x}, \boldsymbol{\theta}) &= \mathcal{N}(y|\phi^T(\mathbf{x})\boldsymbol{\theta}, \sigma^2) \Leftrightarrow \\ \Leftrightarrow y &= \phi^T(\mathbf{x})\boldsymbol{\theta} + \varepsilon = \sum_{k=0}^{K-1} \theta_k \phi_k(\mathbf{x}) + \varepsilon, \end{aligned} \quad (9.13)$$

где  $\phi: \mathbb{R}^D \rightarrow \mathbb{R}^K$  — нелинейное преобразование входных значений  $\mathbf{x}$ , а  $\phi_k: \mathbb{R}^D \rightarrow \mathbb{R}^K$  —  $k$ -я компонента вектора преобразования  $\phi$ . Заметим, что параметры модели  $\boldsymbol{\theta}$  по-прежнему входят в выражение линейно.

### Пример 9.3 (полиномиальная регрессия)

Нас интересует задача регрессии  $y = \phi^T(\mathbf{x})\boldsymbol{\theta} + \varepsilon$ , где  $x \in \mathbb{R}$  и  $\boldsymbol{\theta} \in \mathbb{R}^K$ . В этой ситуации часто используют преобразование

$$\phi(x) = \begin{bmatrix} \phi_0(x) \\ \phi_1(x) \\ \vdots \\ \phi_{K-1}(x) \end{bmatrix} = \begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ \vdots \\ x^{K-1} \end{bmatrix} \in \mathbb{R}^K. \quad (9.14)$$

Таким образом, мы «поднимаем» входы из исходного одномерного пространства в  $K$ -мерное пространство признаков с базисом из всех одночленов  $x^k$  для  $k = 0, \dots, K - 1$ . Теперь мы, оставаясь в рамках метода линейной регрессии, можем брать в качестве модели любые многочлены степени не выше  $K - 1$ . Многочлен степени  $K - 1$  выглядит как

$$f(x) = \sum_{k=0}^{K-1} \theta_k x^k = \phi^T(\mathbf{x})\boldsymbol{\theta}, \quad (9.15)$$

где  $\phi$  определена в (9.14) и  $\boldsymbol{\theta} = [\theta_0, \dots, \theta_{K-1}]^T \in \mathbb{R}^K$  — вектор параметров  $\theta_k$ .

Рассмотрим теперь оценку максимального правдоподобия для параметров  $\theta$  в модели линейной регрессии (9.13). Пусть обучающая выборка состоит из входных значений  $\mathbf{x}_n \in \mathbb{R}^D$  и  $y_n \in \mathbb{R}$ ,  $n = 1, \dots, N$ . Определим *матрицу признаков* как

$$\Phi := \begin{bmatrix} \phi^T(\mathbf{x}_1) \\ \vdots \\ \phi^T(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \cdots & \phi_{K-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \cdots & \phi_{K-1}(\mathbf{x}_2) \\ \vdots & & \vdots \\ \phi_0(\mathbf{x}_N) & \cdots & \phi_{K-1}(\mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{N \times K}, \quad (9.16)$$

где  $\Phi_{ij} = \phi_j(\mathbf{x}_i)$  и  $\phi_j : \mathbb{R}^D \rightarrow \mathbb{R}$ .

#### Пример 9.4 (матрица признаков для многочленов степени 2)

Для многочленов степени 2 и обучающей выборки размера  $N$ , состоящей из точек  $x_n \in \mathbb{R}$ ,  $n = 1, \dots, N$ , матрицей признаков будет

$$\Phi := \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 \end{bmatrix}. \quad (9.17)$$

Для матрицы признаков  $\Phi$  из (9.16) взятый с противоположным знаком логарифм правдоподобия (9.13) можно записать как

$$-\log p(\mathcal{Y} | \mathcal{X}, \theta) = \frac{1}{2\sigma^2} (\mathbf{y} - \Phi\theta)^T (\mathbf{y} - \Phi\theta) + \text{const.} \quad (9.18)$$

Сравнив формулу (9.18) с (9.10b), где не происходило преобразования признаков, мы видим просто замену  $\mathbf{X}$  на  $\Phi$ . Так как и  $\mathbf{X}$ , и  $\Phi$  не зависят от параметров  $\theta$ , по которым мы оптимизируем, мы получаем *оценку максимального правдоподобия*

$$\theta_{\text{ML}} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{y} \quad (9.19)$$

для задачи линейной регрессии с преобразованием признаков (9.13).

**ПРИМЕЧАНИЕ** В отсутствие преобразования признаков мы требовали, чтобы  $\mathbf{X}^T \mathbf{X}$  была обратимой, то есть чтобы строки  $\mathbf{X}$  были линейно независимы. В (9.19) мы требуем обратимости  $\Phi^T \Phi \in \mathbb{R}^{K \times K}$ , что выполняется тогда и только тогда, когда  $\text{rk}(\Phi) = K$ . ◆

### Пример 9.5 (многочлен максимального правдоподобия)

Рассмотрим набор данных с рис. 9.4(a). Он состоит из  $N = 10$  пар  $(x_n, y_n)$ , где  $x_n \sim \mathcal{U}[-5, 5]$  и  $y_n = -\sin(x_n/5) + \cos(x_n) + \varepsilon$ , где  $\varepsilon \sim \mathcal{N}(0, 0, 2^2)$ .

Мы находим подходящий многочлен степени 4, используя метод максимального правдоподобия, то есть параметры  $\theta_{ML}$  задаются формулой (9.19). Оценка максимального правдоподобия дает нам значения функции  $\phi^T(x_*)\theta_{ML}$  на тестовых данных  $x_*$ . Результат показан на рис. 9.4(b).



**Рис. 9.4.** Полиномиальная регрессия: (a) обучающая выборка из пар  $(x_n, y_n)$ ,  $n = 1, \dots, 10$ ; (b) многочлен максимального правдоподобия степени 4

### Оценка дисперсии шума

До сих пор мы считали, что дисперсия шума  $\sigma^2$  известна. Однако мы можем применить метод максимального правдоподобия и для оценки этой дисперсии. Для этого мы последуем стандартной процедуре: запишем логарифм правдоподобия, найдем его производную по  $\sigma^2 > 0$ , приравняем ее к нулю и решим полученное уравнение. Логарифм правдоподобия выглядит как

$$\log p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}, \sigma^2) = \sum_{n=1}^N \log \mathcal{N}(y_n | \phi^T(\mathbf{x}_n)\boldsymbol{\theta}, \sigma^2) = \quad (9.20a)$$

$$= \sum_{n=1}^N \left( -\frac{1}{2} \log(2\pi) - \frac{1}{2} \log \sigma^2 - \frac{1}{2\sigma^2} (y_n - \phi^T(\mathbf{x}_n)\boldsymbol{\theta})^2 \right) = \quad (9.20b)$$

$$= -\frac{N}{2} \log \sigma^2 - \frac{1}{2\sigma^2} \underbrace{\sum_{n=1}^N (y_n - \phi^T(\mathbf{x}_n)\boldsymbol{\theta})^2}_{=: s} + \text{const.} \quad (9.20c)$$

Частная производная логарифма правдоподобия по  $\sigma^2$  равна

$$\frac{\partial \log p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}, \sigma^2)}{\partial \sigma^2} = -\frac{N}{2\sigma^2} + \frac{1}{2\sigma^4} s = 0 \Leftrightarrow \quad (9.21a)$$

$$\Leftrightarrow \frac{N}{2\sigma^2} = \frac{s}{2\sigma^4}, \quad (9.21b)$$

так что мы получаем

$$\sigma_{\text{ML}}^2 = \frac{s}{N} = \frac{1}{N} \sum_{n=1}^N (y_n - \phi^T(\mathbf{x}_n) \boldsymbol{\theta})^2. \quad (9.22)$$

Таким образом, оценка максимального правдоподобия для дисперсии шума — это эмпирическое среднее квадратов расстояний между значениями функции без учета шума  $\phi^T(\mathbf{x}_n) \boldsymbol{\theta}$  и соответствующими зашумленными наблюдениями  $y_n$  на входных точках  $\mathbf{x}_n$ .

### 9.2.2. Переобучение при линейной регрессии

Мы только что обсуждали, как использовать оценку максимального правдоподобия для подгонки линейной модели (например, многочленов) к данным. Качество модели можно оценить, вычислив функцию ошибок (потерь). Один из способов это сделать — взять логарифм правдоподобия с противоположным знаком (9.10b) и, минимизируя его, найти оценку максимального правдоподобия. Другой способ работает, если  $\sigma^2$  не является параметром модели. В этом случае можно проигнорировать множитель  $1/\sigma^2$  и прийти к квадратичной функции потерь  $\|\mathbf{y} - \Phi\boldsymbol{\theta}\|^2$ . Вместо нее часто используют *среднеквадратичное отклонение*

$$\sqrt{\|\mathbf{y} - \Phi\boldsymbol{\theta}\|^2 / N} = \sqrt{\frac{1}{N} \sum_{n=1}^N (y_n - \phi^T(\mathbf{x}_n) \boldsymbol{\theta})^2}, \quad (9.23)$$

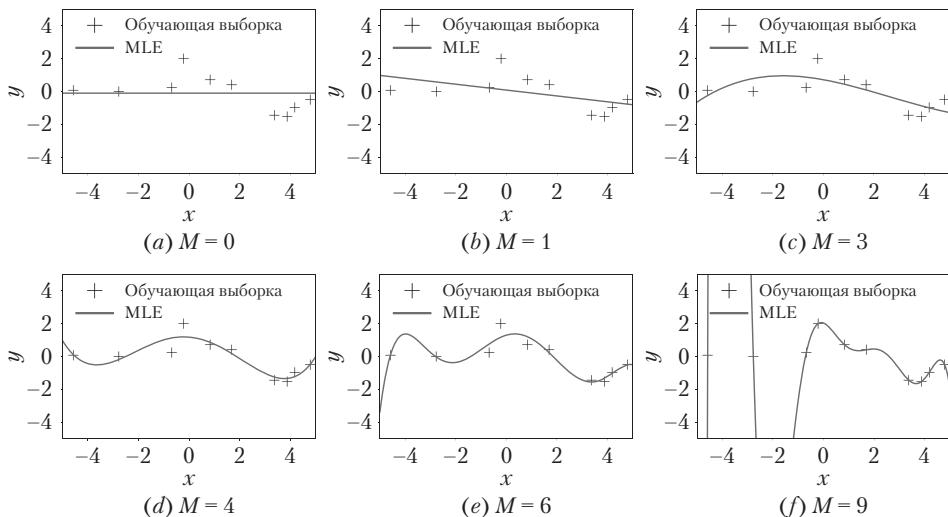
которое (a) позволяет сравнивать ошибки на наборах данных разной величины и (б) имеет тот же масштаб и те же единицы измерения, что и наблюдаемые значения функции  $y$ <sup>1</sup>. Например, если мы обучаем модель, сопоставляющую географические координаты ( $\mathbf{x}$  — широта и долгота) с ценой домов (значения  $y$  даны в евро), среднеквадратичная ошибка (RMSE) также измеряется в евро, тогда как квадрат ошибки — как «евро в квадрате». Если мы вернем множитель  $\sigma^2$  из исходной формулы (9.10b), то получим безразмерную целевую функцию, то есть больше не имеющую единицы измерения.

---

<sup>1</sup> Среднеквадратичное отклонение нормализовано.

При выборе модели (см. раздел 8.6) можно использовать среднеквадратичную ошибку (или логарифм правдоподобия с противоположным знаком<sup>1</sup>), чтобы определить оптимальную степень многочлена, то есть такую степень  $M$ , которая минимизирует целевую функцию. Так как степень многочлена — число натуральное, можно действовать перебором, проверяя все разумные значения  $M$ . Когда обучающая выборка имеет размер  $N$ , достаточно проверить значения  $M$  от 0 до  $N - 1$  включительно. При  $M < N$  оценка максимального правдоподобия будет единственной. При  $M \geq N$  параметров больше, чем точек с данными, так что приходится решать неопределенную систему линейных уравнений ( $\Phi^T \Phi$  из (9.19) больше не будет обратимой), и возможных оценок максимального правдоподобия будет бесконечно много.

На рис. 9.5 показан ряд попыток обучения многочленов, проведенных по набору данных размера  $N = 10$  с рис. 9.4(a) с помощью метода максимального правдоподобия. Заметим, что многочлены маленькой степени (например, константы ( $M = 0$ ) или линейные функции ( $M = 1$ )) плохо подгоняются к данным и, таким образом, плохо приближают реальную породившую их функцию. Для степеней  $M = 3, \dots, 5$  мы видим неплохое гладкое приближение. Переходя к многочленам более высоких степеней, мы видим, что они больше и больше приближаются к данным. В предельном случае  $M = N - 1 = 9$  график проходит через



**Рис. 9.5.** Решения максимального правдоподобия для различных степеней многочлена  $M$

<sup>1</sup> Отрицательный логарифм правдоподобия не имеет единицы измерения.

все заданные точки<sup>1</sup>. Однако многочлены высокой степени сильно колеблются и поэтому плохо представляют функцию, породившую данные, то есть мы имеем дело с переобучением.

Вспомним, что нашей целью является обобщение закономерностей и предсказание значений функции для новых (неизвестных) данных. То, как способность к обобщению зависит от степени многочлена  $M$ , можно увидеть, рассмотрев отдельную тестовую выборку из 200 точек, порожденных той же процедурой, что и обучающая выборка. В качестве тестовых входных данных мы возьмем 200 точек на интервале  $[-5, 5]$ , расположенных на одинаковых расстояниях друг от друга. Для каждого  $M$  мы вычисляем среднеквадратичную ошибку (9.23) на обучающей и тестовой выборках.

Посмотрев на величину ошибки на тестовой выборке, которая показывает обобщающую способность соответствующего многочлена, мы замечаем, что сначала при увеличении степени многочлена ошибка уменьшается (рис. 9.6, светлая линия). Для многочленов четвертой степени ошибка на тестовой выборке небольшая и остается примерно такой же для пятой степени. Однако начиная с шестой степени и далее, ошибка на тестовой выборке значительно возрастает, что говорит об очень плохой обобщающей способности многочленов высоких степеней. В нашем примере это видно из графиков многочленов максимального правдоподобия с рис. 9.5. Заметим, что ошибка на обучающей выборке (черная линия на рис. 9.6) не может увеличиться при увеличении степени многочлена. В нашем примере наилучшая обобщающая способность достигается при степени многочлена  $M = 4$  (когда ошибка на тестовой выборке минимальна).

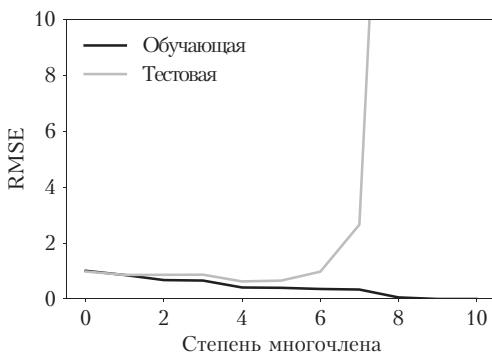


Рис. 9.6. Ошибки на обучающей и тестовой выборке

<sup>1</sup> Случай  $M=N-1$  является предельным в том смысле, что иначе пространство решений соответствующей однородной системы линейных уравнений будет ненулевым и задача линейной регрессии будет иметь бесконечно много оптимальных решений.

### 9.2.3. Оценка апостериорного максимума

Мы только что увидели, что метод максимального правдоподобия подвержен переобучению. Часто переобучение проявляется в увеличении абсолютных значений параметров (Bishop, 2006). Чтобы уменьшить этот эффект, можно установить априорное распределение вероятностей  $p(\theta)$  для параметров. Это априорное распределение явно задает, какие значения параметров мы считаем разумными еще до знакомства с данными. Например, априорное гауссово распределение  $p(\theta) = \mathcal{N}(0, 1)$  для единственного параметра  $\theta$  означает, что мы ожидаем значения параметра из интервала  $[-2, 2]$  (два стандартных отклонения от среднего). Когда у нас появляется набор данных  $\mathcal{X}, \mathcal{Y}$ , вместо максимизации правдоподобия мы максимизируем плотность апостериорной вероятности  $p(\theta | \mathcal{X}, \mathcal{Y})$ . Этот метод называется оценкой апостериорного максимума (MAP).

Плотность апостериорной вероятности значений  $\theta$  при обучающей выборке  $\mathcal{X}, \mathcal{Y}$  мы найдем по теореме Байеса (раздел 6.3):

$$p(\theta | \mathcal{X}, \mathcal{Y}) = \frac{p(\mathcal{Y} | \mathcal{X}, \theta) p(\theta)}{p(\mathcal{Y} | \mathcal{X})}. \quad (9.24)$$

Так как апостериорное распределение зависит от априорного распределения  $p(\theta)$ , априорное распределение влияет на то, какое значение параметра соответствует максимуму апостериорного распределения. Мы продемонстрируем это чуть позже. Вектор параметров  $\theta_{\text{MAP}}$ , максимизирующий апостериорную вероятность (9.24), будет оценкой апостериорного максимума.

Последовательность действий при нахождении MAP-оценки похожа на алгоритм, используемый в методе максимального правдоподобия. Начнем с взятия логарифма. Логарифм плотности апостериорной вероятности выглядит как

$$\log p(\theta | \mathcal{X}, \mathcal{Y}) = \log p(\mathcal{Y} | \mathcal{X}, \theta) + \log p(\theta) + \text{const}, \quad (9.25)$$

где константа объединяет все слагаемые, которые не зависят от  $\theta$ . Видно, что логарифм плотности апостериорной вероятности (9.25) является суммой логарифма правдоподобия  $p(\mathcal{Y} | \mathcal{X}, \theta)$  и логарифма плотности априорной вероятности  $\log p(\theta)$ , так что MAP-оценка будет своего рода «компромиссом» между нашими априорными предположениями о разумных значениях параметров (до знакомства с данными) и правдоподобием (зависящим от данных).

Чтобы найти MAP-оценку  $\theta_{\text{MAP}}$  мы минимизируем взятый с противоположным знаком логарифм плотности апостериорного распределения (относительно  $\theta$ ), то есть находим

$$\theta_{\text{MAP}} \in \arg \min_{\theta} \{-\log p(\mathcal{Y} | \mathcal{X}, \theta) - \log p(\theta)\}. \quad (9.26)$$

Градиент логарифма плотности апостериорного распределения по  $\theta$  равен

$$-\frac{d \log p(\theta | \mathcal{X}, \mathcal{Y})}{d\theta} = -\frac{d \log p(\mathcal{Y} | \mathcal{X}, \theta)}{d\theta} - \frac{d \log p(\theta)}{d\theta}, \quad (9.27)$$

где первое слагаемое в правой части представляет собой градиент взятого с противоположным знаком правдоподобия (9.11c).

Используя сопряженное априорное гауссово распределение  $p(\theta) = \mathcal{N}(\mathbf{0}, b^2 \mathbf{I})$  для параметров  $\theta$  из взятого с противоположным знаком логарифма плотности апостериорного распределения (9.13), получим

$$-\log p(\theta | \mathcal{X}, \mathcal{Y}) = \frac{1}{2\sigma^2} (\mathbf{y} - \Phi\theta)^T (\mathbf{y} - \Phi\theta) + \frac{1}{2b^2} \theta^T \theta + \text{const.} \quad (9.28)$$

Здесь первое слагаемое соответствует вкладу логарифма правдоподобия, а второе — логарифма априорного распределения. Градиент логарифма плотности апостериорного распределения по параметрам  $\theta$  тогда равен

$$-\frac{d \log p(\theta | \mathcal{X}, \mathcal{Y})}{d\theta} = \frac{1}{\sigma^2} (\theta^T \Phi^T \Phi - \mathbf{y}^T \Phi) + \frac{1}{b^2} \theta^T. \quad (9.29)$$

Найдем МАР-оценку  $\theta_{\text{MAP}}$ , приравнивая градиент к  $\mathbf{0}^T$  и решая уравнение. Получаем

$$\frac{1}{\sigma^2} (\theta^T \Phi^T \Phi - \mathbf{y}^T \Phi) + \frac{1}{b^2} \theta^T = \mathbf{0}^T \Leftrightarrow \quad (9.30a)$$

$$\Leftrightarrow \theta^T \left( \frac{1}{\sigma^2} \Phi^T \Phi + \frac{1}{b^2} \mathbf{I} \right) - \frac{1}{\sigma^2} \mathbf{y}^T \Phi = \mathbf{0}^T \Leftrightarrow \quad (9.30b)$$

$$\Leftrightarrow \theta^T \left( \Phi^T \Phi + \frac{\sigma^2}{b^2} \mathbf{I} \right) = \mathbf{y}^T \Phi \Leftrightarrow \quad (9.30c)$$

$$\Leftrightarrow \theta^T = \mathbf{y}^T \Phi \left( \Phi^T \Phi + \frac{\sigma^2}{b^2} \mathbf{I} \right)^{-1} \quad (9.30d)$$

и транспонируем обе части последнего равенства, так что МАР-оценка равна

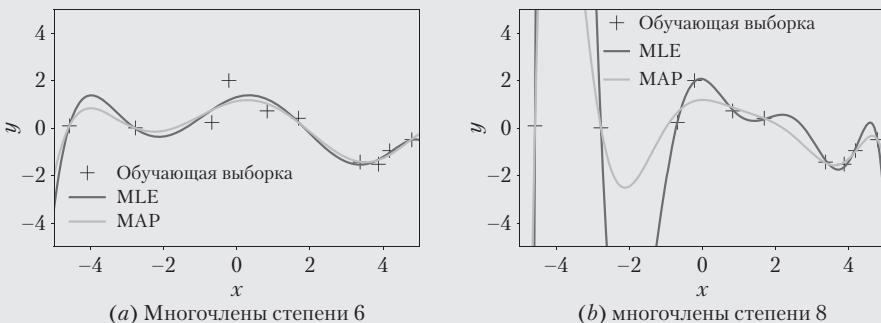
$$\theta_{\text{MAP}} = \left( \Phi^T \Phi + \frac{\sigma^2}{b^2} \mathbf{I} \right)^{-1} \Phi^T \mathbf{y}. \quad (9.31)$$

Сравнивая МАР-оценку (9.31) с оценкой максимального правдоподобия (9.19), мы видим, что разница между ними — дополнительное слагаемое  $(\sigma^2 / b^2) \mathbf{I}$  в обратной матрице. Оно гарантирует, что  $\Phi^T \Phi + (\sigma^2 / b^2) \mathbf{I}$  — симметричная и стро-

го положительно определенная матрица (так она обратима и МАР-оценка – это единственное решение системы линейных уравнений)<sup>1</sup>. Своим появлением это слагаемое обязано регуляризации.

### Пример 9.6 (МАР-оценка для полиномиальной регрессии)

В примере полиномиальной регрессии из раздела 9.2.1 мы выбрали априорное гауссово распределение  $p(\theta) = \mathcal{N}(\mathbf{0}, \mathbf{I})$  для параметров  $\theta$  и нашли МАР-оценку (9.31). На рис. 9.7 показаны оценка максимального правдоподобия и МАР-оценка для многочленов степени 6 (слева) и степени 8 (справа). Априорное распределение (регуляризатор) не сильно влияет на оценку для многочленов небольшой степени, однако для более высоких степеней предотвращает большие колебания. Хотя МАР-оценка может смягчить проблему переобучения, она, вообще говоря, не решает ее полностью, так что, чтобы справляться с переобучением, нам нужен другой подход.



**Рис. 9.7.** Полиномиальная регрессия: оценка максимального правдоподобия (MLE – maximum likelihood estimate) и МАР-оценка

#### 9.2.4. МАР-оценивание как регуляризация

Для смягчения эффекта переобучения вместо принятия априорного распределения параметров  $\theta$  можно ввести регуляризацию, штрафуя за слишком большие значения коэффициентов. Мы минимизируем относительно  $\theta$  (раздел 8.2.3) функцию потерь для метода наименьших квадратов с регуляризацией:

$$\| \mathbf{y} - \Phi \theta \|_2^2 + \lambda \| \theta \|_2^2. \quad (9.32)$$

<sup>1</sup>  $\Phi^T \Phi$  – симметричная положительно полуопределенная. Дополнительное слагаемое в (9.31) строго положительно определено, так что существует обратная матрица.

Здесь первое слагаемое, пропорциональное логарифму правдоподобия с противоположным знаком, см. (9.10b), отвечает за соответствие данным. Второе слагаемое называют регуляризатором, и параметр регуляризации  $\lambda \geq 0$  задает «жесткость» регуляризации.

**ПРИМЕЧАНИЕ** Вместо евклидовой нормы  $\|\cdot\|_2$ , в (9.32) можно взять произвольную  $p$ -норму  $\|\cdot\|_p$ . На практике меньшие значения  $p$  приводят к более разреженным решениям. Здесь разреженность означает, что многие из параметров  $\theta_d = 0$ , что удобно для выбора переменных. При  $p = 1$  регуляризатор был предложен Тибширани (Tibshirani, 1996) и назван LASSO (least absolute shrinkage and selection operator — оператор наименьшего абсолютного сжатия и выбора). ◆

Регуляризатор  $\lambda \|\theta\|_2^2$  в (9.32) можно интерпретировать как логарифм гауссова априорного распределения (с противоположным знаком), использованный нами ранее при МАР-оценивании, см. (9.26). А именно, из априорного гауссового распределения  $p(\theta) = \mathcal{N}(\mathbf{0}, b^2 \mathbf{I})$  получаем

$$-\log p(\theta) = \frac{1}{2b^2} \|\theta\|_2^2 + \text{const}, \quad (9.33)$$

так что при  $\lambda = 1 / (2b^2)$  слагаемое регуляризации и логарифм гауссового априорного распределения (с противоположным знаком) совпадают.

Так как функция потерь для метода наименьших квадратов с регуляризацией (regularized least-squares, RLS) (9.32) состоит из слагаемых, соответствующих логарифмам правдоподобия и априорного распределения, неудивительно, что ее минимизация дает нам решение, похожее на МАР-оценку (9.31), а именно

$$\boldsymbol{\theta}_{\text{RLS}} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda \mathbf{I})^{-1} \boldsymbol{\Phi}^T \mathbf{y}, \quad (9.34)$$

совпадающее с МАР-оценкой (9.31) при  $\lambda = \sigma^2 / b^2$ , где  $\sigma^2$  — дисперсия шума, а  $b^2$  — дисперсия априорного гауссового распределения  $p(\theta) = \mathcal{N}(\mathbf{0}, b^2 \mathbf{I})$ .

К этому моменту мы рассмотрели оценку параметров методами максимального правдоподобия и МАР, в которых находятся точечные оценки<sup>1</sup>  $\theta^*$ , оптимизирующие целевую функцию (правдоподобие или апостериорную вероятность). Мы видели, что оба метода могут приводить к переобучению. В следующем разделе мы обсудим байесовскую линейную регрессию, предсказания которой основаны на использовании байесовского инференса для нахождения апостериорного распределения неизвестных параметров. А именно, вместо точечной оценки происходит усреднение по всем возможным наборам параметров.

<sup>1</sup> Точечная оценка — это одно конкретное значение параметра, а не распределение вероятностей возможных значений.

## 9.3. БАЙЕСОВСКАЯ ЛИНЕЙНАЯ РЕГРЕССИЯ

До этого мы, рассматривая модели линейной регрессии, оценивали их параметры  $\theta$  методом максимального правдоподобия или МАР. Мы увидели, что метод наименьших квадратов может приводить к сильному переобучению, например при маленьком объеме данных. Метод МАР борется с этой проблемой, вводя априорное распределение параметров, выполняющее роль регуляризатора.

Метод байесовской линейной регрессии (Bayesian linear regression, BLR) представляет собой развитие идеи априорных предположений о параметрах. Он даже не пытается найти точечную оценку параметров при прогнозировании, а рассматривает апостериорное распределение параметров целиком. Таким образом, мы не подбираем параметры, а вычисляем среднее по всем их возможным значениям (в соответствии с апостериорным распределением).

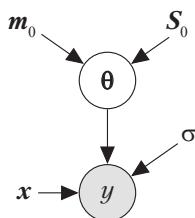
### 9.3.1. Модель

В задаче байесовской линейной регрессии модель выглядит как

$$\begin{array}{ll} \text{априорное распределение} & p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{S}_0), \\ \text{правдоподобие} & p(y | \mathbf{x}, \boldsymbol{\theta}) = \mathcal{N}(y | \phi^T(\mathbf{x})\boldsymbol{\theta}, \sigma^2), \end{array} \quad (9.35)$$

при этом мы в явном виде задаем априорное гауссово распределение  $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{S}_0)$  на  $\boldsymbol{\theta}$ , то есть рассматриваем вектор параметров как случайную величину. Это позволяет нам составить графическую модель (рис. 9.8), на которой явно показаны параметры априорного гауссового распределения для  $\boldsymbol{\theta}$ . Полная вероятностная модель, то есть совместное распределение наблюдаемых и ненаблюдаемых переменных,  $y$  и  $\boldsymbol{\theta}$  соответственно, имеет вид

$$p(y, \boldsymbol{\theta} | \mathbf{x}) = p(y | \mathbf{x}, \boldsymbol{\theta})p(\boldsymbol{\theta}). \quad (9.36)$$



**Рис. 9.8.** Графическая модель байесовской линейной регрессии

### 9.3.2. Априорные предсказания

На практике нас обычно мало интересуют сами по себе значения параметров  $\theta$ . Чаще нам важны предсказания, которые можно делать по этим значениям. При байесовской постановке задачи мы, зная распределение параметров, берем среднее по всем возможным значениям параметров. Это позволяет нам сделать прогноз. Чтобы предсказать результат для входного значения  $\mathbf{x}_*$ , мы интегрируем по  $\theta$  и получаем

$$p(y_* | \mathbf{x}_*) = \int p(y_* | \mathbf{x}_*, \theta) p(\theta) d\theta = \mathbb{E}_{\theta} [p(y_* | \mathbf{x}_*, \theta)], \quad (9.37)$$

что можно понимать как среднее значение предсказаний  $y_* | \mathbf{x}_*, \theta$  для всех возможных параметров  $\theta$  относительно априорного распределения  $p(\theta)$ . Заметим, что предсказания, использующие априорное распределение, не требуют от нас наличия обучающей выборки, а лишь входную точку  $\mathbf{x}_*$ .

В модели (9.35) мы берем сопряженное (гауссово) распределение для  $\theta$ , так что предсказываемое распределение также является гауссовым и может быть записано в явном виде. Для априорного распределения  $p(\theta) = \mathcal{N}(\mathbf{m}_0, \mathbf{S}_0)$  прогноз будет выглядеть как

$$p(y_* | \mathbf{x}_*) = \mathcal{N}(\varphi^T(\mathbf{x}_*) \mathbf{m}_0, \varphi^T(\mathbf{x}_*) \mathbf{S}_0 \varphi(\mathbf{x}_*) + \sigma^2), \quad (9.38)$$

где мы использовали (i) то, что предсказанное распределение тоже является гауссовым из-за сопряженности (раздел 6.6) и свойства маргинализации (раздел 6.5), (ii) независимость гауссова шума, из которой следует

$$\mathbb{V}[y_*] = \mathbb{V}_{\theta}[\varphi^T(\mathbf{x}_*) \theta] + \mathbb{V}_{\varepsilon}[\varepsilon], \quad (9.39)$$

и (iii) то, что  $y_*$  — линейное преобразование  $\theta$ , так что при вычислении среднего и ковариации предсказаний мы можем использовать (6.50) и (6.51) соответственно. В (9.38) слагаемое  $\varphi^T(\mathbf{x}_*) \mathbf{S}_0 \varphi(\mathbf{x}_*)$  в дисперсии предсказаний соответствует неопределенности в значениях  $\theta$ , в то время как  $\sigma^2$  — неопределенность, вызванная наличием шума.

Если нас интересуют не зашумленные  $y_*$ , а значения функции без влияния шума  $f(\mathbf{x}_*) = \varphi^T(\mathbf{x}_*) \theta$ , получим формулу

$$p(f(\mathbf{x}_*)) = \mathcal{N}(\varphi^T(\mathbf{x}_*) \mathbf{m}_0, \varphi^T(\mathbf{x}_*) \mathbf{S}_0 \varphi(\mathbf{x}_*)), \quad (9.40)$$

отличающуюся от (9.38) только отсутствием дисперсии шума  $\sigma^2$  в предсказанной дисперсии.

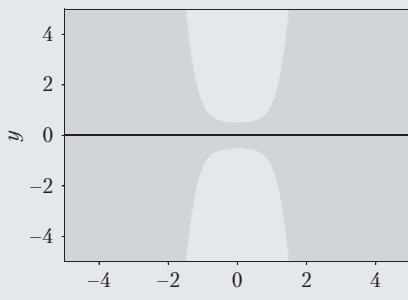
**ПРИМЕЧАНИЕ** Так как можно представить распределение  $p(\theta)$  с помощью множества случайных значений  $\theta_i$ , каждому из которых соответствует функция  $f_i(\cdot) = \theta_i^T \phi(\cdot)$ , то распределение параметров  $p(\theta)$  индуцирует распределение  $p(f(\cdot))$  на функциях. Здесь  $(\cdot)$  — обозначение для функциональной зависимости.



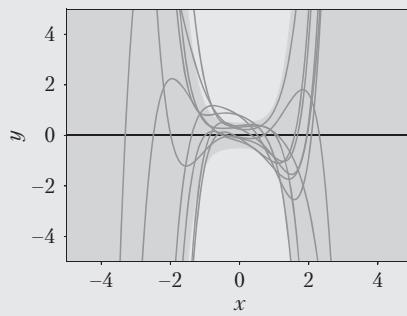
### Пример 9.7 (априорное распределение на функциях)

Рассмотрим задачу байесовской линейной регрессии для многочленов степени 5. Возьмем априорное распределение параметров  $p(\theta) = \mathcal{N}(0, (1/4)\mathbf{I})$ . На рис. 9.9 показаны индуцированное им априорное распределение на функциях (темно-серый цвет соответствует уровню доверия 67%, светло-серый — уровню доверия 95%) и некоторая выборка функций.

Выборка функций генерируется так: берется вектор значений параметров  $\theta_i \sim p(\theta)$  и вычисляются  $f_i(\cdot) = \theta_i^T \phi(\cdot)$ . Преобразование  $\varphi(\cdot)$  применялось к 200 входным значениям  $x_* \in [-5, 5]$ . Неопределенность (показанная на рис. 9.9 как закрашенная область) появляется лишь из-за неопределенности параметров, поскольку мы рассматриваем предсказания без учета шума (9.40).



(a) Априорное распределение на функциях



(b) Примеры функций из этого распределения

**Рис. 9.9.** Априорное распределение на функциях. (a) Распределение на функциях изображено как средняя функция (черная линия) и доверительные интервалы 67 и 95% (закрашены). (b) Примеры функций из этого распределения, сгенерированные по взятым параметрам из априорного распределения параметров

До сих пор мы рассматривали предсказания с использованием априорного распределения параметров  $p(\theta)$ . Однако если мы знаем апостериорное распределение параметров (то есть нам дана обучающая выборка  $\mathcal{X}, \mathcal{Y}$ ), верны те же

принципы предсказания и инференса (9.37) – достаточно заменить априорное распределение  $p(\theta)$  апостериорным  $p(\theta | \mathcal{X}, \mathcal{Y})$ . Далее мы приведем подробный вывод формулы апостериорного распределения.

### 9.3.3. Апостериорное распределение

Пусть нам дана обучающая выборка из точек  $\mathbf{x}_n \in \mathbb{R}^D$  и соответствующих наблюдений  $y_n \in \mathbb{R}$ ,  $n = 1, \dots, N$ . Вычислим апостериорное распределение параметров по теореме Байеса:

$$p(\theta | \mathcal{X}, \mathcal{Y}) = \frac{p(\mathcal{Y} | \mathcal{X}, \theta) p(\theta)}{p(\mathcal{Y} | \mathcal{X})}, \quad (9.41)$$

где  $\mathcal{X}$  – множество экземпляров из обучающей выборки, а  $\mathcal{Y}$  – множество соответствующих значений. Пусть  $p(\mathcal{Y} | \mathcal{X}, \theta)$  – правдоподобие,  $p(\theta)$  – априорное распределение параметров, а

$$p(\mathcal{Y} | \mathcal{X}) = \int p(\mathcal{Y} | \mathcal{X}, \theta) p(\theta) d\theta = \mathbb{E}_{\theta} [p(\mathcal{Y} | \mathcal{X}, \theta)] \quad (9.42)$$

– маргинальное правдоподобие, не зависящее от параметров  $\theta$  и обеспечивающее то, что апостериорное распределение нормализовано (то есть его интеграл равен 1). Можно думать о маргинальном правдоподобии как о среднем правдоподобии по всем возможным параметрам (относительно априорного распределения  $p(\theta)$ ).

**Теорема 9.1 (апостериорное распределение параметров).** В нашей модели (9.35) формула для апостериорного распределения параметров (9.41) будет выглядеть как

$$p(\theta | \mathcal{X}, \mathcal{Y}) = \mathcal{N}(\theta | \mathbf{m}_N, \mathbf{S}_N); \quad (9.43a)$$

$$\mathbf{S}_N = (\mathbf{S}_0^{-1} + \sigma^{-2} \Phi^T \Phi)^{-1}; \quad (9.43b)$$

$$\mathbf{m}_N = \mathbf{S}_N (\mathbf{S}_0^{-1} \mathbf{m}_0 + \sigma^{-2} \Phi^T \mathbf{y}), \quad (9.43c)$$

где  $N$  – размер обучающей выборки.

*Доказательство.* По теореме Байеса, плотность апостериорного распределения  $p(\theta | \mathcal{X}, \mathcal{Y})$  пропорциональна произведению правдоподобия  $p(\mathcal{Y} | \mathcal{X}, \theta)$ , и плотность априорного распределения  $p(\theta)$ :

$$\text{апостериорное распределение } p(\theta | \mathcal{X}, \mathcal{Y}) = \frac{p(\mathcal{Y} | \mathcal{X}, \theta) p(\theta)}{p(\mathcal{Y} | \mathcal{X})}; \quad (9.44a)$$

$$\text{правдоподобие } p(\mathcal{Y} | \mathcal{X}, \theta) = \mathcal{N}(\mathbf{y} | \Phi \theta, \sigma^2 \mathbf{I}); \quad (9.44b)$$

$$\text{априорное распределение } p(\theta) = \mathcal{N}(\theta | \mathbf{m}_0, \mathbf{S}_0). \quad (9.44c)$$

Вместо того чтобы работать с произведением, можно прологарифмировать выражение и найти среднее и дисперсию апостериорного распределения.

Сумма логарифмов плотности априорного распределения и правдоподобия равна

$$\log \mathcal{N}(\mathbf{y} | \Phi \boldsymbol{\theta}, \sigma^2 \mathbf{I}) + \log \mathcal{N}(\boldsymbol{\theta} | \mathbf{m}_0, \mathbf{S}_0) = \quad (9.45a)$$

$$= -\frac{1}{2} \left( (\sigma^{-2}(\mathbf{y} - \Phi \boldsymbol{\theta})^\top (\mathbf{y} - \Phi \boldsymbol{\theta}) + (\boldsymbol{\theta} - \mathbf{m}_0)^\top \mathbf{S}_0^{-1} (\boldsymbol{\theta} - \mathbf{m}_0)) \right) + \text{const}, \quad (9.45b)$$

где константа объединяет все слагаемые, не зависящие от  $\boldsymbol{\theta}$ . Далее мы не будем писать константу. Раскроем скобки в (9.45b) и получим

$$\begin{aligned} & -\frac{1}{2} (\sigma^{-2} \mathbf{y}^\top \mathbf{y} - 2\sigma^{-2} \mathbf{y}^\top \Phi \boldsymbol{\theta} + \boldsymbol{\theta}^\top \sigma^{-2} \Phi^\top \Phi \boldsymbol{\theta} + \boldsymbol{\theta}^\top \mathbf{S}_0^{-1} \boldsymbol{\theta} - \\ & - 2\mathbf{m}_0^\top \mathbf{S}_0^{-1} \boldsymbol{\theta} + \mathbf{m}_0^\top \mathbf{S}_0^{-1} \mathbf{m}_0) = \end{aligned} \quad (9.46a)$$

$$= -\frac{1}{2} \left( \boldsymbol{\theta}^\top (\sigma^{-2} \Phi^\top \Phi + \mathbf{S}_0^{-1}) \boldsymbol{\theta} - 2(\sigma^{-2} \Phi^\top \mathbf{y} + \mathbf{S}_0^{-1} \mathbf{m}_0)^\top \boldsymbol{\theta} \right) + \text{const}, \quad (9.46b)$$

где константа объединяет слагаемые в (9.46a), не зависящие от  $\boldsymbol{\theta}$  ( $-\frac{1}{2}(\sigma^{-2} \mathbf{y}^\top \mathbf{y} + \mathbf{m}_0^\top \mathbf{S}_0^{-1} \mathbf{m}_0)$ ). В последней формуле  $2(\sigma^{-2} \Phi^\top \mathbf{y} + \mathbf{S}_0^{-1} \mathbf{m}_0)^\top \boldsymbol{\theta}$  — слагаемые, линейные по  $\boldsymbol{\theta}$ , а  $\boldsymbol{\theta}^\top (\sigma^{-2} \Phi^\top \Phi + \mathbf{S}_0^{-1}) \boldsymbol{\theta}$  — квадратично зависящие от  $\boldsymbol{\theta}$ . Присмотревшись к (9.46b), мы видим, что это уравнение квадратично относительно  $\boldsymbol{\theta}$ . Так как логарифм ненормализованного апостериорного распределения — квадратичная функция, само это апостериорное распределение будет гауссовым:

$$p(\boldsymbol{\theta} | \mathcal{X}, \mathcal{Y}) = \exp(\log p(\boldsymbol{\theta} | \mathcal{X}, \mathcal{Y})) \propto \exp(\log p(\mathcal{Y} | \mathcal{X}, \boldsymbol{\theta}) + \log p(\boldsymbol{\theta})) \propto \quad (9.47a)$$

$$\propto \exp \left( -\frac{1}{2} \left( \boldsymbol{\theta}^\top (\sigma^{-2} \Phi^\top \Phi + \mathbf{S}_0^{-1}) \boldsymbol{\theta} - 2(\sigma^{-2} \Phi^\top \mathbf{y} + \mathbf{S}_0^{-1} \mathbf{m}_0)^\top \boldsymbol{\theta} \right) \right), \quad (9.47b)$$

где в последней строке мы воспользовались (9.46b).

Остается привести это ненормализованное гауссово распределение к виду, пропорциональному  $\mathcal{N}(\boldsymbol{\theta} | \mathbf{m}_N, \mathbf{S}_N)$ . Иными словами, нам нужно определить среднее  $\mathbf{m}_N$  и ковариационную матрицу  $\mathbf{S}_N$ . Для этого мы используем дополнение до полного квадрата. Логарифм искомого апостериорного распределения равен

$$\log \mathcal{N}(\boldsymbol{\theta} | \mathbf{m}_N, \mathbf{S}_N) = -\frac{1}{2} (\boldsymbol{\theta} - \mathbf{m}_N)^\top \mathbf{S}_N^{-1} (\boldsymbol{\theta} - \mathbf{m}_N) + \text{const} = \quad (9.48a)$$

$$= -\frac{1}{2} \left( \boldsymbol{\theta}^\top \mathbf{S}_N^{-1} \boldsymbol{\theta} - 2\mathbf{m}_N^\top \mathbf{S}_N^{-1} \boldsymbol{\theta} + \mathbf{m}_N^\top \mathbf{S}_N^{-1} \mathbf{m}_N \right). \quad (9.48b)$$

Здесь мы раскрыли скобки в выражении  $(\theta - \mathbf{m}_N)^T S_N^{-1} (\theta - \mathbf{m}_N)$  и получили квадратичное по  $\theta$  слагаемое  $(\theta^T S_N^{-1} \theta)$ , линейное по  $\theta$  слагаемое  $(2\mathbf{m}_N^T S_N^{-1} \theta)$  и константное слагаемое  $(\mathbf{m}_N^T S_N^{-1} \mathbf{m}_N)$ . Это позволяет нам определить  $S_N$  и  $\mathbf{m}_N$ , приравнивая друг другу слагаемые одного цвета в (9.46b) и (9.48b):

$$S_N^{-1} = \Phi^T \sigma^{-2} I \Phi + S_0^{-1} \Leftrightarrow \quad (9.49a)$$

$$\Leftrightarrow S_N = (\sigma^{-2} \Phi^T \Phi + S_0^{-1})^{-1} \quad (9.49b)$$

и

$$\mathbf{m}_N^T S_N^{-1} = (\sigma^{-2} \Phi^T \mathbf{y} + S_0^{-1} \mathbf{m}_0)^T \Leftrightarrow \quad (9.50a)$$

$$\Leftrightarrow \mathbf{m}_N = S_N (\sigma^{-2} \Phi^T \mathbf{y} + S_0^{-1} \mathbf{m}_0). \quad (9.50b)$$

□

**ПРИМЕЧАНИЕ** Если нам дано выражение

$$\mathbf{x}^T A \mathbf{x} - 2\mathbf{a}^T \mathbf{x} + \text{const}_1, \quad (9.51)$$

где  $A$  — симметричная положительно определенная матрица, и мы хотим привести его к виду

$$(\mathbf{x} - \mu)^T \Sigma (\mathbf{x} - \mu) + \text{const}_2, \quad (9.52)$$

это можно сделать, задав

$$\Sigma := A, \quad (9.53)$$

$$\mu := \Sigma^{-1} \mathbf{a} \quad (9.54)$$

и  $\text{const}_2 = \text{const}_1 - \mu^T \Sigma \mu$ .

◆

Можно заметить, что выражение под экспонентой в (9.47b) имеет вид (9.51) с

$$A := \sigma^{-2} \Phi^T \Phi + S_0^{-1}; \quad (9.55)$$

$$\mathbf{a} := \sigma^{-2} \Phi^T \mathbf{y} + S_0^{-1} \mathbf{m}_0. \quad (9.56)$$

Так как в выражениях вида (9.46a) бывает сложно определить  $A$ ,  $a$ , при поиске решения часто помогает привести их к виду (9.51), разделяющему квадратичные, линейные и константные слагаемые.

### 9.3.4. Апостериорные предсказания

В (9.37) было вычислено распределение  $y_*$  на тестовой выборке  $\mathbf{x}_*$  при априорном распределении параметров  $p(\theta)$ . Предсказание с помощью апостериорного рас-

пределения  $p(\theta | \mathcal{X}, \mathcal{Y})$  в целом мало от него отличается, так как оба распределения гауссовые (хотя и с разными средним и дисперсией). Таким образом, как и в нашем примере из раздела 9.3.2, получим апостериорное предсказанное распределение

$$p(y_* | \mathcal{X}, \mathcal{Y}, \mathbf{x}_*) = \int p(y_* | \mathbf{x}_*, \theta) p(\theta | \mathcal{X}, \mathcal{Y}) d\theta = \quad (9.57a)$$

$$= \int \mathcal{N}(y_* | \phi^T(\mathbf{x}_*)\theta, \sigma^2) \mathcal{N}(\theta | \mathbf{m}_N, \mathbf{S}_N) d\theta = \quad (9.57b)$$

$$= \mathcal{N}(y_* | \phi^T(\mathbf{x}_*)\mathbf{m}_N, \phi^T(\mathbf{x}_*)\mathbf{S}_N\phi(\mathbf{x}_*) + \sigma^2). \quad (9.57c)$$

Слагаемое  $\phi^T(\mathbf{x}_*)\mathbf{S}_N\phi(\mathbf{x}_*)$  отвечает за неопределенность параметров  $\theta$ . Заметим, что  $\mathbf{S}_N$  зависит от обучающей выборки через  $\Phi$ ; см. (9.43b). Предсказанное среднее  $\phi^T(\mathbf{x}_*)\mathbf{m}_N$  совпадает с МАР-оценкой.

**ПРИМЕЧАНИЕ** Вместо интеграла в (9.57a) предсказанное распределение можно записать как  $\mathbb{E}_{\theta}[\cdot | \mathcal{X}, \mathcal{Y}]$ , где математическое ожидание берется относительно распределения  $p(\theta | \mathcal{X}, \mathcal{Y})$ .

Запись предсказанного апостериорного распределения в таком виде подчеркивает сходство с маргинальным правдоподобием (9.42). Главные различия между маргинальным правдоподобием и предсказанным апостериорным распределением состоят в том, что (i) о маргинальном правдоподобии можно думать как о способе предсказать результат для обучающей выборки ( $y$ ), а не для тестовой ( $y_*$ ), и (ii) маргинальное правдоподобие усредняется относительно априорного, а не апостериорного распределения параметров. ◆

**ПРИМЕЧАНИЕ** Зачастую нам не очень интересно предсказанное распределение  $p(y_* | \mathcal{X}, \mathcal{Y}, \mathbf{x}_*)$  зашумленных наблюдений  $y_*$ . Вместо этого мы хотим узнать распределение значений функции без учета шума  $f(\mathbf{x}_*) = \phi^T(\mathbf{x}_*)\theta$ . Мы вычислим, когда он появляется, пользуясь свойствами среднего и дисперсии:

$$\begin{aligned} \mathbb{E}[f(\mathbf{x}_*) | \mathcal{X}, \mathcal{Y}] &= \mathbb{E}_{\theta}[\phi^T(\mathbf{x}_*)\theta | \mathcal{X}, \mathcal{Y}] = \phi^T(\mathbf{x}_*)\mathbb{E}_{\theta}[\theta | \mathcal{X}, \mathcal{Y}]; \\ \phi^T(\mathbf{x}_*)\mathbf{m}_N &= \mathbf{m}_N^T\phi(\mathbf{x}_*); \end{aligned} \quad (9.58)$$

$$\begin{aligned} \mathbb{V}_{\theta}[f(\mathbf{x}_*) | \mathcal{X}, \mathcal{Y}] &= \mathbb{V}_{\theta}[\phi^T(\mathbf{x}_*)\theta | \mathcal{X}, \mathcal{Y}]; \\ &= \phi^T(\mathbf{x}_*)\mathbb{V}_{\theta}[\theta | \mathcal{X}, \mathcal{Y}]\phi(\mathbf{x}_*) \\ &= \phi^T(\mathbf{x}_*)\mathbf{S}_N\phi(\mathbf{x}_*). \end{aligned} \quad (9.59)$$

Видно, что предсказанное среднее будет тем же самым, что и для зашумленных наблюдений (так как среднее значение шума равно 0, а предсказанный дисперсия отличается только на  $\sigma^2$ , то есть дисперсию шума). Предсказывая зашумленные значения функции, мы должны учитывать эту неопределенность и добавлять  $\sigma^2$ , а для значений без учета шума этого делать не надо — единственным источником неопределенности будут параметры. ◆

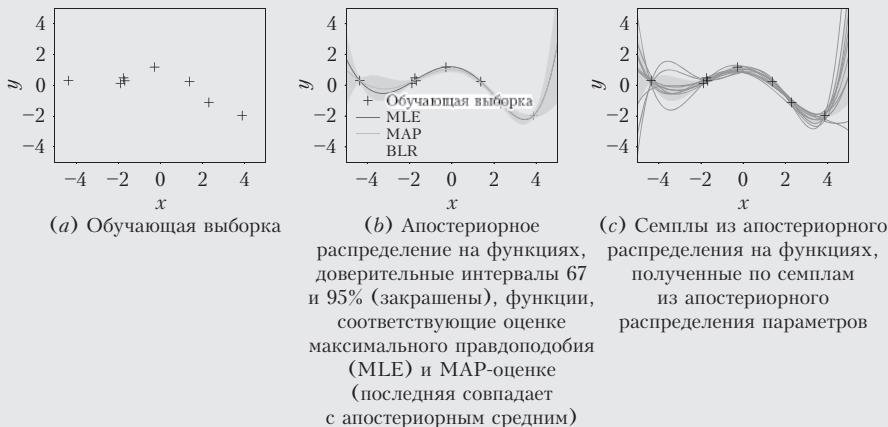
**ПРИМЕЧАНИЕ** Интегрирование по параметрам  $\theta$  задает распределение на функциях. Взяв случайные  $\theta_i \sim p(\theta | \mathcal{X}, \mathcal{Y})$  из апостериорного распределения параметров, получим  $\theta_i^T \phi(\cdot)$ .

Средняя функция, то есть все множество ожидаемых значений функции  $\mathbb{E}_\theta [f(\cdot) | \theta, \mathcal{X}, \mathcal{Y}]$ , для этого распределения задается формулой  $\mathbf{m}_N^T \phi(\cdot)$ . Маргинальная дисперсия, то есть дисперсия  $f(\cdot)$ , вычисляется как  $\phi(\cdot)^T \mathbf{S}_N \phi(\cdot)$ . ♦

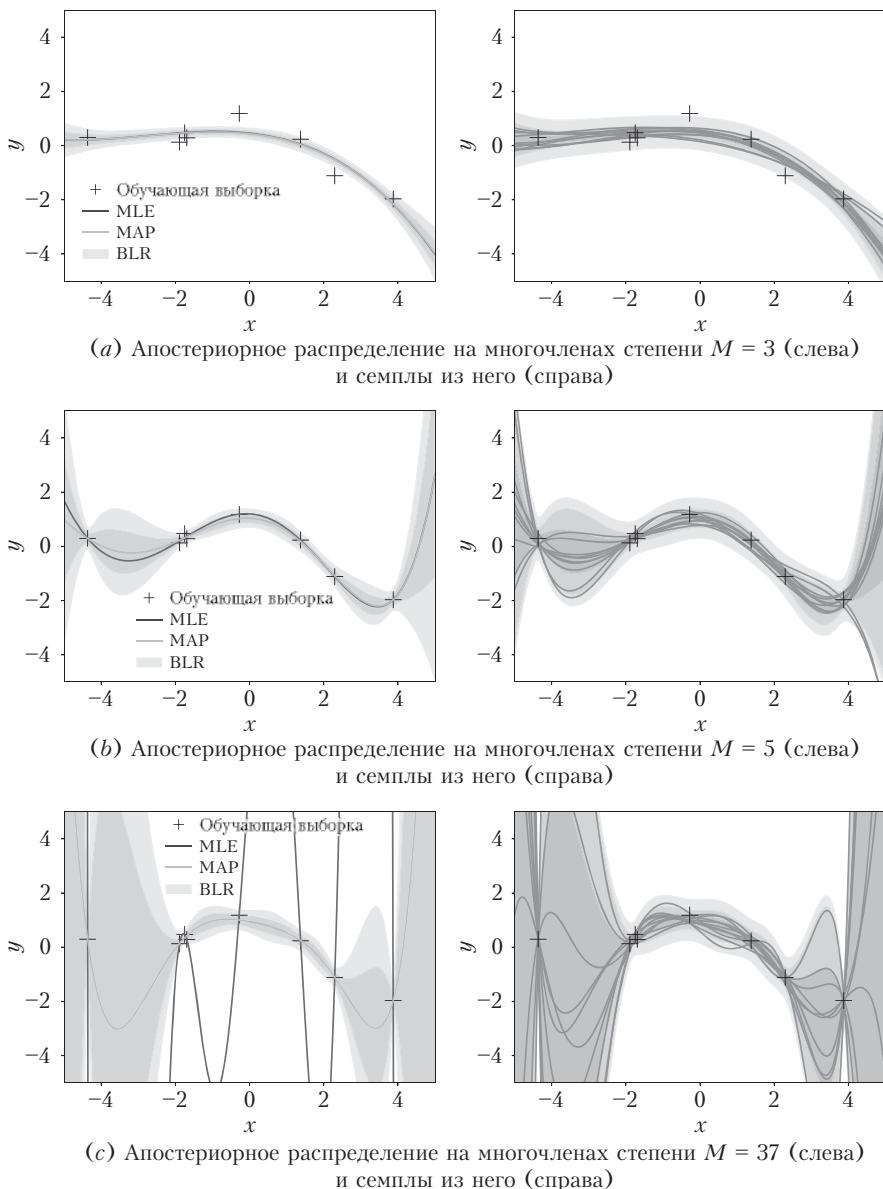
### Пример 9.8 (апостериорное распределение на функциях)

Вернемся к задаче байесовской линейной регрессии для многочленов степени 5. Возьмем априорное распределение параметров  $p(\theta) = \mathcal{N}(0, (1/4)\mathbf{I})$ . На рис. 9.9 показано априорное распределение на функциях, заданное априорным распределением параметров и случайно выбранными из этого распределения функциями.

На рис. 9.10 показано апостериорное распределение на функциях, полученное методом байесовской линейной регрессии (BLR). Обучающая выборка показана в части (a); часть (b) показывает апостериорное распределение на функциях, в том числе функции, которые мы бы получили методами максимального правдоподобия и MAP-оценивания. Функция, полученная методом MAP-оценивания, соответствует апостериорному среднему в задаче байесовской линейной регрессии. Часть (c) демонстрирует некоторые функции, взятые из найденного апостериорного распределения.



**Рис. 9.10.** Байесовская линейная регрессия и апостериорное распределение на функциях: (a) обучающая выборка; (b) апостериорное распределение на функциях; (c) функции (выборка) из апостериорного распределения



**Рис. 9.11.** Байесовская линейная регрессия. Слева: закрашены доверительные интервалы 67% (темно-серый) и 95% (светло-серый). Средняя функция для байесовской линейной регрессии совпадает с МАР-оценкой. Дисперсия равна сумме дисперсии шума и слагаемого, отвечающего за неопределенность параметров (зависит от тестового входа). Справа: образцы из апостериорного распределения на функциях

На рис. 9.11 показаны некоторые апостериорные распределения на функциях, заданные апостериорным распределением параметров. Для различных степеней многочлена  $M$  слева изображены функция максимального правдоподобия  $\theta_{ML}^T \phi(\cdot)$ , МАР-функция  $\theta_{MAP}^T \phi(\cdot)$  (совпадающая с апостериорным средним) и закрашены доверительные интервалы 67 и 95%, найденные методом байесовской линейной регрессии.

Справа изображены случайно выбранные функции из апостериорного распределения. Мы сформировали выборку параметров  $\theta_i$  из апостериорного распределения параметров и нашли функцию  $\phi^T(\mathbf{x}_*)\theta_i$ . Для многочленов небольшой степени апостериорное распределение не позволяет параметрам сильно меняться — случайные функции почти одинаковы. Увеличивая гибкость модели путем добавления большего количества параметров (то есть повышения степени многочленов), мы ослабляем ограничения на эти параметры, и случайно взятые функции могут сильно различаться. Можно заметить в левой части рисунка, как растет неопределенность, особенно на границах.

Хотя для многочленов седьмой степени МАР-оценка дает хорошее совпадение с данными, байесовская линейная регрессия говорит нам о большой апостериорной дисперсии. Это может быть важно, когда предсказания используются в системах принятия решений, где неудачные решения могут привести к серьезным последствиям (например, в обучении с подкреплением или в робототехнике).

### 9.3.5. Вычисление маргинального правдоподобия

В разделе 8.6.2 мы подчеркивали важность маргинального правдоподобия в байесовском выборе модели. Далее мы будем вычислять маргинальное правдоподобие для байесовской линейной регрессии с сопряженным гауссовым априорным распределением на параметрах, то есть в той постановке задачи, которая была дана в этой главе.

Вспомним, что мы рассматриваем следующие порождающие процессы:

$$\theta \sim \mathcal{N}(\mathbf{m}_0, \mathbf{S}_0); \quad (9.60a)$$

$$y_n | \mathbf{x}_n, \theta \sim \mathcal{N}(\mathbf{x}_n^T \theta, \sigma^2), \quad (9.60b)$$

$n = 1, \dots, N$ . Маргинальное правдоподобие вычисляется как

$$p(\mathcal{Y} | \mathcal{X}) = \int p(\mathcal{Y} | \mathcal{X}, \theta) p(\theta) d\theta = \quad (9.61a)$$

$$= \int \mathcal{N}(\mathbf{y} | \mathbf{X}\theta, \sigma^2 \mathbf{I}) \mathcal{N}(\theta | \mathbf{m}_0, \mathbf{S}_0) d\theta, \quad (9.61b)$$

где при интегрировании исчезает зависимость от параметров модели  $\theta$ . Маргинальное правдоподобие мы вычисляем в два шага: сначала показываем, что оно (как распределение  $y$ ) является гауссовым, потом находим среднее и ковариацию<sup>1</sup>.

1. Маргинальное правдоподобие гауссово. Из раздела 6.5.2 мы знаем, что (i) произведение двух гауссовых случайных величин будет (ненормализованной) гауссовой случайной величиной, и (ii) при линейном преобразовании гауссовой случайной величины получается гауссово распределение. В формуле (9.61b) мы с помощью линейных преобразований превращаем  $\mathcal{N}(y | X\theta, \sigma^2 I)$  в  $\mathcal{N}(\theta | \mu, \Sigma)$  для некоторых  $\mu, \Sigma$ . После этого можно вычислить интеграл и получить нормализующую константу для произведения двух гауссиан. Сама эта константа будет иметь гауссово распределение, см. (6.76).
2. Среднее и ковариация. Среднее и ковариационная матрица для маргинального правдоподобия вычисляются через стандартные свойства среднего и ковариации, касающихся аффинных преобразований случайных величин (раздел 6.4.4). Среднее для маргинального правдоподобия вычисляется как

$$\mathbb{E}_\theta[\mathcal{Y} | \mathcal{X}] = \mathbb{E}_\theta[X\theta + \varepsilon] = X\mathbb{E}_\theta[\theta] = X\mathbf{m}_0. \quad (9.62)$$

Заметим, что  $\varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$  — вектор из независимых одинаково распределенных случайных величин. Матрица ковариации вычисляется как

$$\text{Cov}_\theta[\mathcal{Y} | \mathcal{X}] = \text{Cov}[X\theta] + \sigma^2 I = X \text{Cov}_\theta[\theta] X^\top + \sigma^2 I = \quad (9.63a)$$

$$= X S_0 X^\top + \sigma^2 I. \quad (9.63b)$$

Значит, маргинальное правдоподобие задается формулой

$$p(\mathcal{Y} | \mathcal{X}) = (2\pi)^{-\frac{N}{2}} \det(X S_0 X^\top + \sigma^2 I)^{-\frac{1}{2}} \times \quad (9.64a)$$

$$\begin{aligned} & \times \exp\left(-\frac{1}{2}(\mathbf{y} - X\mathbf{m}_0)^\top (X S_0 X^\top + \sigma^2 I)^{-1} (\mathbf{y} - X\mathbf{m}_0)\right) = \\ & = \mathcal{N}(\mathbf{y} | X\mathbf{m}_0, X S_0 X^\top + \sigma^2 I). \end{aligned} \quad (9.64b)$$

Учитывая связь маргинального правдоподобия и предсказанного апостериорного распределения (см. примечание ранее в этом разделе), формула не должна вызывать удивления.

---

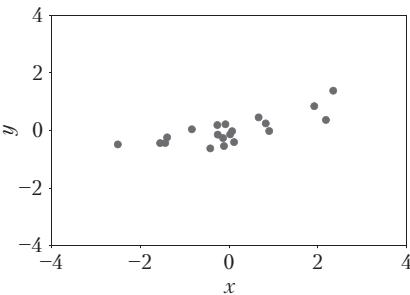
<sup>1</sup> Маргинальное правдоподобие можно рассматривать как математическое ожидание правдоподобия относительно априорного распределения параметров.

## 9.4. МАКСИМАЛЬНОЕ ПРАВДОПОДОБИЕ КАК ОРТОГОНАЛЬНАЯ ПРОЕКЦИЯ

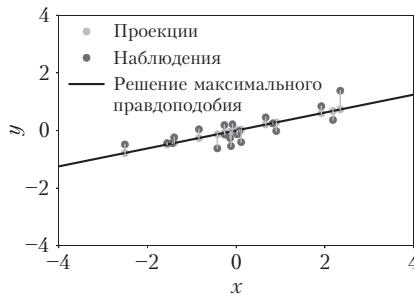
Продравшись через алгебраические дебри при выводе оценок максимального правдоподобия и МАР, мы предоставим геометрическую интерпретацию оценки максимального правдоподобия. Рассмотрим простую задачу линейной регрессии

$$y = x\theta + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2) \quad (9.65)$$

с линейными функциями  $f: \mathbb{R} \rightarrow \mathbb{R}$ , проходящими через начало координат (для простоты мы не используем преобразования). Параметр  $\theta$  задает угол наклона прямой. На рис. 9.12(a) показан одномерный датасет.



(a) Набор данных задачи регрессии:  
зашумленные наблюдения  $y_n$   
(показаны темным) значенияй  
функции  $f(x_n)$  на входах  $x_n$



(b) Светлые точки — проекции зашумленных наблюдений (темные точки) на прямую  $\theta_{ML}x$ . Решение максимального правдоподобия — это подпространство (прямая), сумма расстояний от всех наблюдений до которой минимальна

**Рис. 9.12.** Геометрическая интерпретация метода наименьших квадратов. (a) Данные; (b) решение максимального правдоподобия как проекция

Теперь, когда у нас есть обучающий набор данных  $\{(x_1, y_1), \dots, (x_N, y_N)\}$ , следует вспомнить результаты раздела 9.2.1 и найти оценку максимального правдоподобия для тангенса угла наклона:

$$\theta_{ML} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} = \frac{\mathbf{X}^T \mathbf{y}}{\mathbf{X}^T \mathbf{X}} \in \mathbb{R}, \quad (9.66)$$

где  $\mathbf{X} = [x_1, \dots, x_N]^T \in \mathbb{R}^N$ ,  $\mathbf{y} = [y_1, \dots, y_N]^T \in \mathbb{R}^N$ .

Таким образом, на входных точках  $\mathbf{X}$  обучающей выборки получаем оптимальные (максимально правдоподобные) значения ответов

$$\mathbf{X}\theta_{ML} = \mathbf{X} \frac{\mathbf{X}^T \mathbf{y}}{\mathbf{X}^T \mathbf{X}} = \frac{\mathbf{X} \mathbf{X}^T}{\mathbf{X}^T \mathbf{X}} \mathbf{y}, \quad (9.67)$$

то есть наилучшее (в смысле квадратичной функции ошибки) приближение  $\mathbf{X}\theta$  к  $\mathbf{y}$ .

Так как мы ищем такое  $\theta$ , что  $\mathbf{y} = \mathbf{X}\theta$ , можно рассматривать задачу линейной регрессии как решение систем линейных уравнений. Значит, мы можем обращаться к идеям из линейной алгебры и аналитической геометрии, которые мы обсуждали в главах 2 и 3. В частности, присмотревшись к формуле (9.67), мы видим, что оценка максимального правдоподобия  $\theta_{\text{ML}}$  в примере (9.65) фактически задает ортогональную проекцию  $\mathbf{y}$  на одномерное подпространство с системой образующих  $\mathbf{X}$ . Вспомнив результаты раздела 3.8, касающиеся ортогональных проекций, мы понимаем, что  $\frac{\mathbf{X}\mathbf{X}^T}{\mathbf{X}^T\mathbf{X}}$  — матрица проекции,  $\theta_{\text{ML}}$  — координаты проекции на одномерное подпространство в  $\mathbb{R}^N$ , натянутое на  $\mathbf{X}$ , а  $\mathbf{X}\theta_{\text{ML}}$  — ортогональная проекция  $\mathbf{y}$  на это подпространство.

Таким образом, метод максимального правдоподобия находит геометрически оптимальное решение, то есть ближайший к наблюдениям  $\mathbf{y}$  вектор из подпространства, натянутого на  $\mathbf{X}$ , где мерой близости является квадрат расстояния от  $y_n$  до  $x_n\theta$ . Этим ближайшим вектором будет ортогональная проекция. На рис. 9.12(b) показана проекция зашумленных наблюдений на подпространство, соответствующая минимуму квадрата расстояния от исходного набора данных до подпространства (заметьте, что  $x$ -координаты фиксированы). Это решение максимального правдоподобия.

В общем случае линейной регрессии, где

$$y = \varphi^T(\mathbf{x})\theta + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2) \quad (9.68)$$

с векторнозначными преобразованиями  $\varphi(\mathbf{x}) \in \mathbb{R}^K$ , также можно интерпретировать оценку максимального правдоподобия

$$\mathbf{y} \approx \Phi\theta_{\text{ML}}; \quad (9.69)$$

$$\theta_{\text{ML}} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y} \quad (9.70)$$

как проекцию на  $K$ -мерное подпространство в  $\mathbb{R}^N$ , натянутое на столбцы матрицы преобразования  $\Phi$  (раздел 3.8.2).

В частном случае, когда функции преобразования  $\varphi_k$ , используемые для построения  $\Phi$ , ортонормальны (раздел 3.7), то есть столбцы  $\Phi$  образуют ортонормированный базис (раздел 3.5), для которого  $\Phi^T\Phi = \mathbf{I}$ , проекция равна

$$\Phi(\Phi^T\Phi)^{-1}\Phi\mathbf{y} = \Phi\Phi^T\mathbf{y} = \left( \sum_{k=1}^K \phi_k\phi_k^T \right)\mathbf{y}, \quad (9.71)$$

то есть линейная зависимость разных преобразований исчезает, и проекция максимального правдоподобия будет просто суммой проекций  $\mathbf{y}$  на векторы базиса  $\phi_k$ , то есть столбцы  $\Phi$ . Многие популярные в обработке сигналов базисные функции (вейвлеты, базис Фурье) образуют ортогональные базисы. Базис, не являющийся ортогональным, можно привести к таковому методом Грама — Шмидта (Strang, 2003).

## 9.5. ДЛЯ ДАЛЬНЕЙШЕГО ЧТЕНИЯ

В данной главе мы обсуждали линейную регрессию для гауссовых функций правдоподобия и сопряженных априорных распределений параметров. Эти условия позволяли получить явные формулы для байесовского инференса. Однако в некоторых приложениях нам может понадобиться другая функция правдоподобия. Например, в задаче бинарной *классификации* возможных ответов только два, и гауссово распределение здесь не подходит. Вместо него мы можем взять правдоподобие Бернулли, дающее нам вероятность предсказания, которое говорит о принадлежности к классу (1 или 0). Советуем обратиться к книгам Бишопа (Bishop, 2006), Мерфи (Murphy, 2012) и Барбера (Barber, 2012) для подробного введения в задачи классификации. Другой пример не-гауссового правдоподобия возникает в ситуации, когда данные — это целые неотрицательные числа (например, когда речь идет о количестве единиц чего-либо). В этом случае биномиальное или пуассоновское правдоподобие будет более удачным выбором, чем гауссово. Все эти примеры относятся к классу *обобщенных линейных моделей*, представляющих собой гибкое обобщение линейной регрессии, которое позволяет ошибкам иметь отличное от гауссова распределение. Обобщенные линейные модели позволяют связывать линейную модель с наблюдаемыми значениями через гладкую обратимую (возможно, нелинейную) функцию  $\sigma(\cdot)$ , то есть  $y = \sigma(f(\mathbf{x}))$ , где  $f(\mathbf{x}) = \theta^T \phi(\mathbf{x})$  — модель линейной регрессии (9.13). Можно воспринимать обобщенную линейную модель как композицию функций  $y = \sigma \circ f$ , где  $f$  — модель линейной регрессии, а  $\sigma$  — функция активации. Заметим, что хотя модель и называется «обобщенной линейной», ее предсказания больше не будут линейными по параметрам  $\theta$ . В *логистической регрессии* в качестве функции активации берется *логистическая сигмоида*  $\sigma(f) = 1/(1 + \exp(-f)) \in [0, 1]$ , которую можно понимать как вероятность наблюдать значение  $y = 1$  бернуллиевской случайной величины  $y \in \{0, 1\}$ . Функцию  $\sigma(\cdot)$  называют *функцией активации*, а обратную к ней — *канонической функцией связи*. Таким образом, понятно, что обобщенные линейные модели являются «кирпичиками» для глубоких нейронных сетей прямого распространения. Фактически, обобщенная линейная модель  $\mathbf{y} = \sigma(\mathbf{A}\mathbf{x} + \mathbf{b})$ , где  $\mathbf{A}$  — матрица весов, а  $\mathbf{b}$  — вектор смещения, описывает одно-

слойную нейронную сеть с функцией активации<sup>1</sup>  $\sigma(\cdot)$ . Теперь можно построить композицию таких функций

$$\begin{aligned} \mathbf{x}_{k+1} &= f_k(\mathbf{x}_k); \\ f_k(\mathbf{x}_k) &= \sigma_k(\mathbf{A}_k \mathbf{x}_k + \mathbf{b}_k) \end{aligned} \quad (9.72)$$

для  $k = 0, \dots, K - 1$ , где  $\mathbf{x}_0$  — входные признаки,  $\mathbf{x}_K = \mathbf{y}$  — наблюдения, а  $f_{K-1} \circ \dots \circ f_0$  — нейронная сеть из  $K$  слоев. Таким образом, «кирпичиками» такой глубокой нейронной сети являются обобщенные линейные модели, определенные в (9.72)<sup>2</sup>. Нейронные сети (Bishop, 1995; Goodfellow et al., 2016) значительно превосходят по выразительности и гибкости модели линейной регрессии. Однако задача оценки максимального правдоподобия для параметров — невыпуклая задача оптимизации, а задача маргинализации параметров при полностью байесовской ее постановке не имеет аналитического решения.

Мы уже упоминали, что распределение параметров задает распределение функций регрессии. Гауссовые процессы (Rasmussen and Williams, 2006) являются моделями регрессии, в которых распределение функций играет ключевую роль. Вместо введения распределения параметров в гауссовых процессах мы сразу работаем с распределением функций. Для этого применяется «ядерный трюк» (kernel trick, Schölkopf and Smola, 2002), позволяющий вычислять скалярное произведение двух значений функции  $f(\mathbf{x}_i), f(\mathbf{x}_j)$  по входным  $\mathbf{x}_i, \mathbf{x}_j$ . Гауссовые процессы тесно связаны как с байесовской линейной регрессией, так и с методом опорных векторов, но их можно интерпретировать и как байесовские нейронные сети с одним скрытым слоем и стремящимся к бесконечности количеством узлов (Neal, 1996; Williams, 1997). Отличное введение в гауссовые процессы можно найти у МакКея (MacKay, 1998), а также у Расмуссена и Уильямса (Rasmussen and Williams, 2006).

В этой главе мы в основном уделяли внимание гауссовым априорным распределениям параметров, поскольку они позволяют вывести формулы для задач линейной регрессии в явном виде. Однако даже при гауссовом правдоподобии можно выбрать негауссово априорное распределение. Рассмотрим постановку задачи, в которой входные значения  $\mathbf{x} \in \mathbb{R}^D$ , а обучающая выборка маленькая — размера  $N \ll D$ . Это означает, что задача регрессии недоопределенна. В этом случае можно выбрать разреженное априорное распределение параметров, приравнивающее как можно большее число параметров к нулю (отбор признаков). Такое априорное распределение дает более сильную регу-

<sup>1</sup> Для обычной линейной регрессии функцией активации будет тождественная функция.

<sup>2</sup> На <https://tinyurl.com/glm-dnn> есть отличный пост о связи между обобщенными моделями и глубокими нейронными сетями.

ляризацию, чем гауссово априорное распределение, что часто помогает точности предсказаний и интерпретируемости модели. Одним из часто используемых априорных распределений является распределение Лапласа. Модель линейной регрессии с априорным распределением Лапласа эквивалентна линейной регрессии с L1-регуляризацией (LASSO) (Tibshirani, 1996). Распределение Лапласа имеет острый пик в нуле (первая производная имеет разрыв) и более сконцентрировано вблизи нуля, чем гауссово, то есть сильнее принуждает параметры быть равными нулю. Ненулевые параметры являются важными для задачи регрессии, поэтому мы часто говорим об «отборе признаков» («отборе переменных»).

# 10

## Снижение размерности с помощью анализа главных компонент

Непосредственная работа с многомерными данными, такими как изображения, сопряжена с некоторыми трудностями: их тяжело анализировать, сложно интерпретировать, визуализация практически невозможна, и (с практической точки зрения) хранение векторов данных может быть дорогостоящим. Однако многомерные данные часто обладают свойствами, которые мы можем использовать. Например, данные большой размерности часто являются избыточными, то есть многие измерения являются избыточными и могут быть объяснены комбинацией других измерений. Кроме того, измерения в данных высокой размерности часто коррелируют, так что данные обладают внутренней структурой с более низкой размерностью. Снижение размерности использует структуру и корреляцию и позволяет нам работать с более компактным представлением данных, в идеале без потери информации. Мы можем рассматривать уменьшение размерности как метод сжатия, подобный jpg или mp3, которые представляют собой алгоритмы сжатия изображений и музыки.

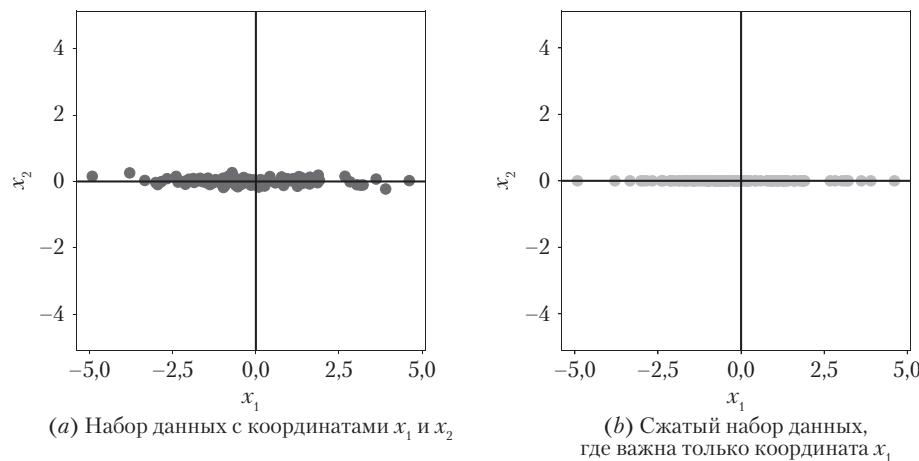
В этой главе мы обсудим *анализ главных компонент* (PCA – *principal component analysis*) и алгоритм *уменьшения линейной размерности*. PCA, предложенный Пирсоном (1901) и Хотеллингом (1933), существует уже более 100 лет и до сих пор остается одним из наиболее часто используемых методов сжатия и визуализации данных. Он также используется для идентификации простых закономерностей, скрытых факторов и структур многомерных данных. В сообществе

обработки сигналов РСА также известен как *преобразование Карунена – Лоэва*. В этой главе мы выводим РСА из первых принципов, опираясь на наше понимание базиса и изменения базиса (разделы 2.6.1 и 2.7.2), прогнозов (раздел 3.8), собственных значений (раздел 4.2), гауссовых распределений (раздел 6.5), и ограниченной оптимизации (раздел 7.2).

Снижение размерности обычно использует свойство данных с высокой размерностью (например, изображений), которое часто находится в подпространстве с низкой размерностью. На рис. 10.1 показан наглядный пример в двух измерениях. Хотя данные на рис. 10.1(a) не совсем лежат на прямой, данные не сильно различаются в направлении  $x_2$ , так что мы можем выразить их так, как если бы они были на прямой — почти без потерь; см. рис. 10.1(b). Для описания данных на рис. 10.1(b) требуется только координата  $x_1$ , а данные лежат в одномерном подпространстве  $\mathbb{R}^2$ .

## 10.1. ПОСТАНОВКА ПРОБЛЕМЫ

В РСА мы заинтересованы в том, чтобы найти проекции  $\hat{x}_n$  точек данных  $x_n$ , которые максимально похожи на исходные точки данных, но которые имеют значительно более низкую внутреннюю размерность. На рис. 10.1 показано, как это может выглядеть.



**Рис. 10.1.** Иллюстрация: уменьшение размерности. (a) Набор данных с координатами  $x_1$  и  $x_2$ . (b) Сжатый набор данных, где важна только координата  $x_1$ . (a) Исходный набор данных не сильно меняется в направлении  $x_2$ . (b) Данные из (a) могут быть представлены с использованием только координаты  $x_1$  почти без потерь

Более конкретно, мы рассматриваем независимый и одинаково распределенный набор данных  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N, \mathbf{x}_n\} \in \mathbb{R}^D$ , со средним значением 0, который обладает *ковариационной матрицей данных* (6.42)

$$\mathbf{S} = \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n \mathbf{x}_n^T. \quad (10.1)$$

Кроме того, мы предполагаем, что существует низкоразмерное сжатое представление (код)

$$\mathbf{z}_n = \mathbf{B}^T \mathbf{x}_n \in \mathbb{R}^M \quad (10.2)$$

для  $\mathbf{x}_n$ , где мы определяем проекционную матрицу

$$\mathbf{B} := [\mathbf{b}_1, \dots, \mathbf{b}_M] \in \mathbb{R}^{D \times M}. \quad (10.3)$$

Мы предполагаем, что столбцы  $\mathbf{B}$  ортонормированы (определение 3.7), так что  $\mathbf{b}_i^T \mathbf{b}_j = 0$  тогда и только тогда, когда  $i \neq j$  и  $\mathbf{b}_i^T \mathbf{b}_i = 1$ . Ищем  $M$ -мерное подпространство  $U \subseteq \mathbb{R}^D$ ,  $\dim(U) = M < D$ , на которое мы проецируем данные. Обозначим проецируемые данные через  $\tilde{\mathbf{x}}_n \in U$ , а их координаты (относительно базисных векторов  $\mathbf{b}_1, \dots, \mathbf{b}_M$  матрицы  $U$ ) через  $\mathbf{z}_n$ . Наша цель — найти проекции  $\tilde{\mathbf{x}}_n \subseteq \mathbb{R}^D$  (или, что эквивалентно, коды  $\mathbf{z}_n$  и базисные векторы  $\mathbf{b}_1, \dots, \mathbf{b}_M$ ), чтобы они были максимально похожи на исходные данные  $\mathbf{x}_n$  и минимизировали потери из-за сжатия<sup>1</sup>.

### Пример 10.1 (координатное представление / код)

Рассмотрим  $\mathbb{R}^2$  с каноническим базисом  $\mathbf{e}_1 = [1, 0]^T$ ,  $\mathbf{e}_2 = [0, 1]^T$ . Из главы 2 мы знаем, что  $\mathbf{x} \in \mathbb{R}^2$  можно представить как линейную комбинацию этих базисных векторов, например

$$\begin{bmatrix} 5 \\ 3 \end{bmatrix} = 5\mathbf{e}_1 + 3\mathbf{e}_2. \quad (10.4)$$

Однако, когда мы рассматриваем векторы вида

$$\tilde{\mathbf{x}} = \begin{bmatrix} 0 \\ z \end{bmatrix} \in \mathbb{R}^2, \quad z \in \mathbb{R}, \quad (10.5)$$

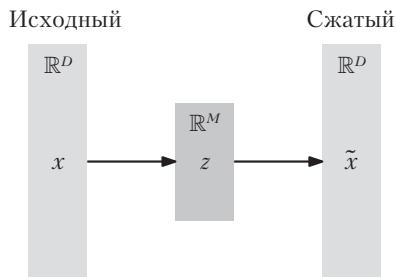
их всегда можно записать как  $0\mathbf{e}_1 + z\mathbf{e}_2$ . Чтобы представить эти векторы, достаточно запомнить/сохранить координату/код  $z$  вектора  $\tilde{\mathbf{x}}$  относи-

<sup>1</sup> Столбцы  $\mathbf{b}_1, \dots, \mathbf{b}_M$  пространства  $\mathbf{B}$  образуют базис  $M$ -мерного подпространства, в котором живут проецируемые данные  $\tilde{\mathbf{x}} = \mathbf{B}\mathbf{B}^T \mathbf{x} \in \mathbb{R}^D$ .

тельно вектора  $\mathbf{e}_2$ . Точнее, набор  $\tilde{\mathbf{x}}$  векторов (со стандартным векторным сложением и скалярным умножением) образует векторное подпространство  $U$  (раздел 2.4) с  $\dim(U) = 1$ , поскольку  $U = \text{span}[\mathbf{e}_2]$ <sup>1</sup>.

В разделе 10.2 мы найдем низкоразмерные представления, которые сохраняют как можно больше информации и минимизируют потери при сжатии. Альтернативный вывод РСА приведен в разделе 10.3, где мы рассмотрим минимизацию квадрата ошибки восстановления  $\|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$  между исходными данными  $\mathbf{x}_n$  и их проекцией  $\tilde{\mathbf{x}}_n$ .

Рисунок 10.2 иллюстрирует настройку, которую мы рассматриваем в РСА, где  $\mathbf{z}$  представляет низкоразмерное представление сжатых данных  $\tilde{\mathbf{x}}$  и играет роль узкого места, которое контролирует, сколько информации может передаваться между  $\mathbf{x}$  и  $\tilde{\mathbf{x}}$ . В РСА мы рассматриваем линейную зависимость между исходными данными  $\mathbf{x}$  и их низкоразмерным кодом  $\mathbf{z}$ , так что  $\mathbf{z} = \mathbf{B}^T \mathbf{x}$  и  $\tilde{\mathbf{x}} = \mathbf{B}\mathbf{z}$  для подходящей матрицы  $\mathbf{B}$ . Основываясь на мотивации представления РСА как метода сжатия данных, мы можем интерпретировать стрелки на рис. 10.2 как пару операций, представляющих кодеры и декодеры. Линейное отображение, представленное буквой  $\mathbf{B}$ , можно представить себе как декодер, который отображает низкоразмерный код  $\mathbf{z} \in \mathbb{R}^M$  обратно в исходное пространство данных  $\mathbb{R}^D$ . Точно так же  $\mathbf{B}^T$  можно представить себе кодировщиком, который кодирует исходные данные  $\mathbf{x}$  как низкоразмерный (сжатый) код  $\mathbf{z}$ .



**Рис. 10.2.** Графическое изображение РСА. В РСА мы находим сжатую версию  $\mathbf{z}$  исходных данных  $\mathbf{x}$ . Сжатые данные могут быть преобразованы в набор  $\tilde{\mathbf{x}}$ , который живет в исходном пространстве данных, но имеет внутреннее представление более низкой размерности, чем  $\mathbf{x}$

В этой главе мы будем использовать набор данных MNIST в качестве повторяющегося примера, который содержит 60 000 примеров рукописных цифр от 0 до 9. Каждая цифра представляет собой изображение в градациях серого размером  $28 \times 28$ , то есть содержит 784 пикселя, чтобы мы могли интерпретировать

<sup>1</sup> Размерность векторного пространства соответствует количеству его базисных векторов (раздел 2.6.1).

каждое изображение в этом наборе данных как вектор  $\mathbf{x} \in \mathbb{R}^{784}$ . Примеры этих цифр показаны на рис. 10.3.

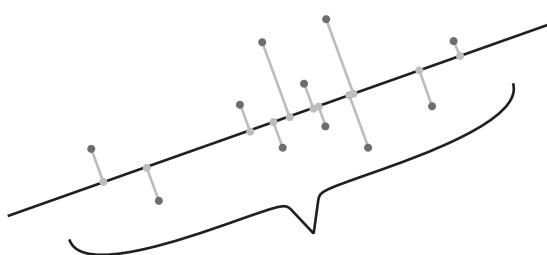


**Рис. 10.3.** Примеры рукописных цифр из набора данных MNIST.  
<http://yann.lecun.com/exdb/mnist/>

## 10.2. ПЕРСПЕКТИВА МАКСИМАЛЬНОЙ ДИСПЕРСИИ

На рис. 10.1 приведен пример того, как двумерный набор данных может быть представлен с использованием одной координаты. На рис. 10.1(b) мы решили игнорировать координату  $x_2$  данных, потому что она не добавляла слишком много информации, так что сжатые данные похожи на исходные данные на рис. 10.1(a). Мы могли бы игнорировать координату  $x_1$ , но тогда сжатые данные сильно отличались от исходных данных, и большая часть информации в данных была бы потеряна.

Если мы интерпретируем информационное содержание данных как «пространство, заполняющее» набор данных, то мы можем описать информацию, содержащуюся в данных, глядя на разброс данных. Из раздела 6.4.1 мы знаем, что дисперсия является индикатором распространения данных, и мы можем вывести РСА как алгоритм уменьшения размерности, который максимизирует дисперсию в низкоразмерном представлении данных, чтобы сохранить как можно больше информации. Рисунок 10.4 иллюстрирует это.



**Рис. 10.4.** РСА находит подпространство (линия) меньшей размерности, которое поддерживает как можно большую дисперсию (разброс данных), когда данные (черные точки) проецируются на это подпространство (светлые линии)

Учитывая настройку, обсуждаемую в разделе 10.1, наша цель — найти матрицу  $\mathbf{B}$  (раздел 10.3), которая сохраняет как можно больше информации при сжатии данных, проецируя ее на подпространство, охватываемое столбцами  $b_1, \dots, b_M$  матрицы  $\mathbf{B}$ . Сохранение большей части информации после сжатия данных эквивалентно улавливанию наибольшего количества дисперсии в коде низкой размерности (Hotelling, 1933).

**ПРИМЕЧАНИЕ** Для ковариационной матрицы данных в (10.1) мы приняли центрированные данные. Мы можем сделать это предположение без ограничения общности: предположим, что  $\mu$  — это среднее значение данных. Используя свойства дисперсии, которые мы обсуждали в разделе 6.4.4, получаем

$$\mathbb{V}_z[\mathbf{z}] = \mathbb{V}_x[\mathbf{B}^T(\mathbf{x} - \mu)] = \mathbb{V}_x[\mathbf{B}^T\mathbf{x} - \mathbf{B}^T\mu] = \mathbb{V}_x[\mathbf{B}^T\mathbf{x}], \quad (10.6)$$

то есть дисперсия низкоразмерного кода не зависит от среднего значения данных. Поэтому без ограничения общности мы предполагаем, что данные имеют среднее значение  $\mathbf{0}$  до конца этого раздела. С этим предположением среднее значение низкоразмерного кода также равно  $\mathbf{0}$ , поскольку  $\mathbb{E}_z[\mathbf{z}] = \mathbb{E}_x[\mathbf{B}^T\mathbf{x}] = \mathbf{B}^T\mathbb{E}_x[\mathbf{x}] = \mathbf{0}$ .  $\diamond$

### 10.2.1. Направление с максимальной дисперсией

Мы максимизируем дисперсию низкоразмерного кода, используя последовательный подход. Мы начинаем с поиска единственного вектора  $\mathbf{b}_1 \in \mathbb{R}^D$ , который максимизирует дисперсию проецируемых данных<sup>1</sup>, то есть мы стремимся максимизировать дисперсию первой координаты  $z_1$  элемента  $\mathbf{z} \in \mathbb{R}^M$ , так чтобы

$$V_1 := \mathbb{V}[z_1] = \frac{1}{N} \sum_{n=1}^N z_{1n}^2 \quad (10.7)$$

было максимизировано, где мы использовали предположение о независимости и одинаковом распределении данных и определили  $z_{1n}$  как первую координату низкоразмерного представления  $\mathbf{z}_n \in \mathbb{R}^M$  элемента  $\mathbf{x}_n \in \mathbb{R}^D$ . Обратите внимание, что первая компонента  $\mathbf{z}_n$  задается формулой

$$z_{1n} = \mathbf{b}_1^T \mathbf{x}_n, \quad (10.8)$$

то есть это координата ортогональной проекции  $\mathbf{x}_n$  на одномерное подпространство, натянутое на  $\mathbf{b}_1$  (раздел 3.8). Подставляем (10.8) в (10.7), что дает

$$V_1 = \frac{1}{N} \sum_{n=1}^N (\mathbf{b}_1^T \mathbf{x}_n)^2 = \frac{1}{N} \sum_{n=1}^N \mathbf{b}_1^T \mathbf{x}_n \mathbf{x}_n^T \mathbf{b}_1 = \quad (10.9a)$$

$$= \mathbf{b}_1^T \left( \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n \mathbf{x}_n^T \right) \mathbf{b}_1 = \mathbf{b}_1^T \mathbf{S} \mathbf{b}_1, \quad (10.9b)$$

---

<sup>1</sup> Вектор  $\mathbf{b}_1$  будет первым столбцом матрицы  $\mathbf{B}$  и, следовательно, первым из  $M$  ортонормированных базисных векторов, которые охватывают подпространство меньшей размерности.

где  $\mathbf{S}$  — ковариационная матрица данных, определенная в (10.1). В (10.9a) мы использовали тот факт, что скалярное произведение двух векторов симметрично по своим аргументам, то есть  $\mathbf{b}_1^T \mathbf{x}_n = \mathbf{x}_n^T \mathbf{b}_1$ .

Обратите внимание, что произвольное увеличение величины вектора  $\mathbf{b}_1$  увеличивает  $V_1$ , то есть вектор  $\mathbf{b}_1$ , который в два раза длиннее, может привести к  $V_1$ , который потенциально в четыре раза больше. Поэтому мы ограничиваем все решения до  $\|\mathbf{b}_1\|^2 = 1$ , что приводит к задаче ограниченной оптимизации, в которой мы ищем направление, в котором данные изменяются больше всего. При ограничении пространства решений единичными векторами вектор  $\mathbf{b}_1$ , который указывает в направлении максимальной дисперсии, может быть найден с помощью задачи оптимизации с ограничениями

$$\begin{aligned} & \max_{\mathbf{b}_1} \mathbf{b}_1^T \mathbf{S} \mathbf{b}_1 \\ & \text{при } \|\mathbf{b}_1\|^2 = 1. \end{aligned} \quad (10.10)$$

Следуя разделу 7.2, получаем лагранжиан

$$\mathcal{L}(\mathbf{b}_1, \lambda) = \mathbf{b}_1^T \mathbf{S} \mathbf{b}_1 + \lambda (1 - \mathbf{b}_1^T \mathbf{b}_1) \quad (10.11)$$

для решения этой задачи оптимизации с ограничениями. Частные производные  $\mathcal{L}$  по  $\mathbf{b}_1$  и  $\lambda$  равны

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}_1} = 2\mathbf{b}_1^T \mathbf{S} - 2\lambda \mathbf{b}_1^T, \quad \frac{\partial \mathcal{L}}{\partial \lambda} = 1 - \mathbf{b}_1^T \mathbf{b}_1 \quad (10.12)$$

соответственно. Установка этих частных производных равными  $\mathbf{0}$  дает нам соотношения

$$\mathbf{S} \mathbf{b}_1 = \lambda \mathbf{b}_1, \quad (10.13)$$

$$\mathbf{b}_1^T \mathbf{b}_1 = 1. \quad (10.14)$$

Сравнивая это с определением разложения по собственным значениям (раздел 4.4), мы видим, что  $\mathbf{b}_1$  является собственным вектором ковариационной матрицы данных  $\mathbf{S}$ , а множитель Лагранжа  $\lambda$  играет роль соответствующего собственного значения<sup>1</sup>. Это свойство собственного вектора (10.13) позволяет нам переписать нашу цель по дисперсии (10.10) как

$$V_1 = \mathbf{b}_1^T \mathbf{S} \mathbf{b}_1 = \lambda \mathbf{b}_1^T \mathbf{b}_1 = \lambda, \quad (10.15)$$

---

<sup>1</sup> Величина  $\sqrt{\lambda}$  также называется загрузкой единичного вектора  $\mathbf{b}_1$  и представляет собой стандартное отклонение данных, приходящихся на главный промежуток подпространства  $[\mathbf{b}_1]$ .

то есть дисперсия данных, проецируемых на одномерное подпространство, равна собственному значению, которое связано с базисным вектором  $b_1$ , охватывающим это подпространство. Следовательно, чтобы максимизировать дисперсию низкоразмерного кода, мы выбираем базисный вектор, связанный с наибольшим собственным значением матрицы ковариации данных. Этот собственный вектор называется первой *главной компонентой*. Мы можем определить влияние/вклад главной компоненты  $b_1$  в исходное пространство данных, отображая координату  $z_{1n}$  обратно в пространство данных, что дает нам прогнозируемую точку данных

$$\tilde{\mathbf{x}}_n = b_1 z_{1n} = b_1 b_1^T \mathbf{x}_n \in \mathbb{R}^D \quad (10.16)$$

в исходном пространстве данных.

**ПРИМЕЧАНИЕ** Хотя  $\tilde{\mathbf{x}}_n$  является  $D$ -мерным вектором, для его представления относительно базисного вектора  $b_1 \in \mathbb{R}^D$  требуется только одна координата  $z_{1n}$ .



## 10.2.2. М-мерное подпространство с максимальной дисперсией

Предположим, что мы нашли первые  $m - 1$  главных компонент как  $m - 1$  собственных векторов  $S$ , которые связаны с наибольшими  $m - 1$  собственными значениями. Поскольку  $S$  симметрично, спектральная теорема (теорема 4.15) утверждает, что мы можем использовать эти собственные векторы для построения ортонормированного собственного базиса  $(m - 1)$ -мерного подпространства в  $\mathbb{R}^D$ . Как правило,  $m$ -я главная компонента может быть найдена путем вычитания влияния первых  $m - 1$  главных компонент  $b_1, \dots, b_{m-1}$  из данных, тем самым помогая найти главные компоненты, которые сжимают оставшуюся информацию. Затем мы приходим к новой матрице данных

$$\hat{\mathbf{X}} := \mathbf{X} - \sum_{i=1}^{m-1} b_i b_i^T \mathbf{X} = \mathbf{X} - \mathbf{B}_{m-1} \mathbf{X}, \quad (10.17)$$

где  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{D \times N}$  содержит точки данных как векторы-столбцы<sup>1</sup>, а  $\mathbf{B}_{m-1} := \sum_{i=1}^{m-1} b_i b_i^T$  — матрица проекции, которая проецируется на подпространство, натянутое на  $b_1, \dots, b_{m-1}$ .

---

<sup>1</sup> Матрица  $\hat{\mathbf{X}} = [\hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_N] \in \mathbb{R}^{D \times N}$  в (10.17) содержит информацию в данных, которые еще не были сжаты.

**ПРИМЕЧАНИЕ** На протяжении всей этой главы мы не следуем соглашению о соборе данных  $\mathbf{x}_1, \dots, \mathbf{x}_N$  в качестве строк матрицы данных, но мы определяем их как столбцы  $\mathbf{X}$ . Это означает, что наша матрица данных  $\mathbf{X}$  является матрицей  $D \times N$  вместо обычной матрицы  $N \times D$ . Причина нашего выбора заключается в том, что алгебраические операции работают гладко, без необходимости транспонировать матрицу или переопределять векторы как векторы-строки, которые умножаются слева на матрицы. ♦

Чтобы найти  $m$ -ю главную компоненту, мы максимизируем дисперсию

$$V_m = \mathbb{V}[z_m] = \frac{1}{N} \sum_{n=1}^N z_{mn}^2 = \frac{1}{N} \sum_{n=1}^N (\mathbf{b}_m^\top \hat{\mathbf{x}}_n)^2 = \mathbf{b}_m^\top \hat{\mathbf{S}} \mathbf{b}_m \quad (10.18)$$

при условии  $\|\mathbf{b}_m\|^2 = 1$ , где мы проделали те же шаги, что и в (10.9b), и определили  $\hat{\mathbf{S}}$  как матрицу ковариации данных преобразованного набора данных  $\hat{\mathbf{X}} : \{\hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_N\}$ . Как и раньше, когда мы рассматривали только первую главную компоненту, мы решаем задачу оптимизации с ограничениями и обнаруживаем, что оптимальное решение  $\mathbf{b}_m$  является собственным вектором  $\hat{\mathbf{S}}$ , который связан с наибольшим собственным значением  $\hat{\mathbf{S}}$ .

Оказывается, что  $\mathbf{b}_m$  также является собственным вектором  $\mathbf{S}$ . В общем, наборы собственных векторов  $\mathbf{S}$  и  $\hat{\mathbf{S}}$  идентичны. Поскольку и  $\mathbf{S}$ , и  $\hat{\mathbf{S}}$  симметричны, мы можем найти ОНБ собственных векторов (спектральная теорема 4.15), то есть существует  $D$  различных собственных векторов как для  $\mathbf{S}$ , так и для  $\hat{\mathbf{S}}$ . Далее мы покажем, что каждый собственный вектор оператора  $\mathbf{S}$  является собственным вектором оператора  $\hat{\mathbf{S}}$ . Предположим, мы уже нашли собственные векторы  $\mathbf{b}_1, \dots, \mathbf{b}_{m-1}$  матрицы  $\hat{\mathbf{S}}$ . Рассмотрим собственный вектор  $\mathbf{b}_i$  оператора  $\mathbf{S}$ , то есть  $\mathbf{S}\mathbf{b}_i = \lambda_i \mathbf{b}_i$ . В общем,

$$\hat{\mathbf{S}}\mathbf{b}_i = \frac{1}{N} \hat{\mathbf{X}} \hat{\mathbf{X}}^\top \mathbf{b}_i = \frac{1}{N} (\mathbf{X} - \mathbf{B}_{m-1} \mathbf{X})(\mathbf{X} - \mathbf{B}_{m-1} \mathbf{X})^\top \mathbf{b}_i = \quad (10.19a)$$

$$= (\mathbf{S} - \mathbf{S}\mathbf{B}_{m-1} - \mathbf{B}_{m-1}\mathbf{S} + \mathbf{B}_{m-1}\mathbf{S}\mathbf{B}_{m-1})\mathbf{b}_i. \quad (10.19b)$$

Мы различаем два случая. Если  $i \geq m$ , то есть  $\mathbf{b}_i$  является собственным вектором, который не входит в число первых  $m - 1$  главных компонент, то  $\mathbf{b}_i$  ортогонален первым  $m - 1$  главным компонентам и  $\mathbf{B}_{m-1}\mathbf{b}_i = \mathbf{0}$ . Если  $i < m$ , то есть  $\mathbf{b}_i$  входит в число первых  $m - 1$  главных компонент, то  $\mathbf{b}_i$  является базисным вектором главного подпространства, на которое проецируется  $\mathbf{B}_{m-1}$ . Поскольку  $\mathbf{b}_1, \dots, \mathbf{b}_{m-1}$  являются ОНБ этого главного подпространства, получаем  $\mathbf{B}_{m-1}\mathbf{b}_i = \mathbf{b}_i$ . Эти два случая можно резюмировать следующим образом:

$$\mathbf{B}_{m-1}\mathbf{b}_i = \mathbf{b}_i, \text{ если } i < m; \mathbf{B}_{m-1}\mathbf{b}_i = \mathbf{0}, \text{ если } i \geq m. \quad (10.20)$$

В случае  $i \geq m$ , используя (10.20) в (10.19b), получаем  $\hat{\mathbf{S}}\mathbf{b}_i = (\mathbf{S} - \mathbf{B}_{m-1}\mathbf{S})\mathbf{b}_i = \mathbf{S}\mathbf{b}_i = \lambda_i \mathbf{b}_i$  то есть  $\mathbf{b}_i$  также является собственным вектором  $\hat{\mathbf{S}}$  с собственным значением  $\lambda_i$ . А именно:

$$\hat{\mathbf{S}}\mathbf{b}_m = \mathbf{S}\mathbf{b}_m = \lambda_m \mathbf{b}_m. \quad (10.21)$$

Уравнение (10.21) показывает, что  $\mathbf{b}_m$  является собственным вектором не только  $\mathbf{S}$ , но и  $\hat{\mathbf{S}}$ . В частности,  $\lambda_m$  — наибольшее собственное значение  $\hat{\mathbf{S}}$ , а  $\lambda_m$  —  $m$ -е наибольшее собственное значение  $\mathbf{S}$ , и оба имеют связанный собственный вектор  $\mathbf{b}_m$ .

В случае  $i < m$ , используя (10.20) в (10.19b), получаем

$$\hat{\mathbf{S}}\mathbf{b}_i = (\mathbf{S} - \mathbf{S}\mathbf{B}_{m-1} - \mathbf{B}_{m-1}\mathbf{S} + \mathbf{B}_{m-1}\mathbf{S}\mathbf{B}_{m-1})\mathbf{b}_i = \mathbf{0} = 0\mathbf{b}_i. \quad (10.22)$$

Это означает, что  $\mathbf{b}_1, \dots, \mathbf{b}_{m-1}$  также являются собственными векторами  $\hat{\mathbf{S}}$ , но они связаны с собственным значением 0, так что  $\mathbf{b}_1, \dots, \mathbf{b}_{m-1}$  покрывают нулевое пространство  $\hat{\mathbf{S}}$ .

В целом каждый собственный вектор  $\mathbf{S}$  также является собственным вектором  $\hat{\mathbf{S}}$ . Однако, если собственные векторы  $\mathbf{S}$  являются частью  $(m-1)$ -мерного главного подпространства, то соответствующее собственное значение  $\hat{\mathbf{S}}$  равно 0<sup>1</sup>. При соотношении (10.21) и  $\mathbf{b}_m^\top \mathbf{b}_m = 1$  дисперсия данных, проецируемых на  $m$ -ю главную компоненту, равна

$$V_m = \mathbf{b}_m^\top \mathbf{S} \mathbf{b}_m \stackrel{(10.21)}{=} \lambda_m \mathbf{b}_m^\top \mathbf{b}_m = \lambda_m. \quad (10.23)$$

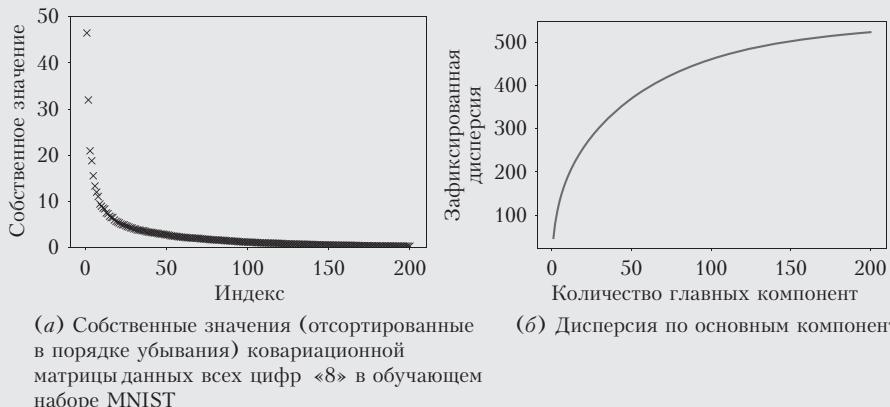
Это означает, что дисперсия данных при проецировании на  $M$ -мерное подпространство равна сумме собственных значений, которые связаны с соответствующими собственными векторами ковариационной матрицы данных.

### Пример 10.2 (собственные значения MNIST «8»)

Взяв все цифры «8» в обучающих данных MNIST, мы вычисляем собственные значения ковариационной матрицы данных. На рис. 10.5(a) показаны 200 наибольших собственных значений ковариационной матрицы данных. Мы видим, что только некоторые из них имеют значение, значительно отличающееся от 0. Следовательно, большая часть дисперсии при проецировании данных на подпространство, охватываемое соответ-

<sup>1</sup> Этот вывод показывает, что существует тесная связь между  $M$ -мерным подпространством с максимальной дисперсией и разложением по собственным значениям. Мы вернемся к этой связи в разделе 10.4.

ствующими собственными векторами, улавливается только несколькими главными компонентами, как показано на рис. 10.5(b).



**Рис. 10.5.** Свойства обучающих данных MNIST «8». (a) Собственные значения, отсортированные по убыванию; (b) дисперсия, зафиксированная главными компонентами, связанными с наибольшими собственными значениями

В целом, чтобы найти  $M$ -мерное подпространство  $\mathbb{R}^D$ , которое сохраняет как можно больше информации, PCA говорит нам выбрать столбцы матрицы  $\mathbf{B}$  в (10.3) в качестве  $M$  собственных векторов матрицы ковариации данных  $\mathbf{S}$ , которые связаны с  $M$  наибольшими собственными значениями. Максимальное количество дисперсии, которое PCA может зафиксировать с помощью первых  $M$  главных компонент, равно

$$V_M = \sum_{m=1}^M \lambda_m, \quad (10.24)$$

где  $\lambda_m$  — это  $M$  наибольших собственных значений ковариационной матрицы данных  $\mathbf{S}$ . Следовательно, дисперсия, теряемая при сжатии данных через PCA, составляет

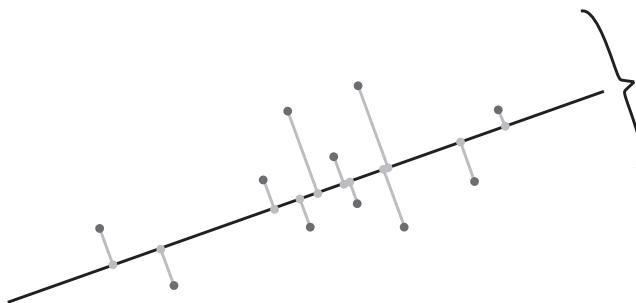
$$J_M := \sum_{j=M+1}^D \lambda_j = V_D - V_M. \quad (10.25)$$

Вместо этих абсолютных величин мы можем определить относительную дисперсию, захваченную как  $\frac{V_M}{V_D}$ , и относительную дисперсию, потерянную при сжатии, как  $1 - \frac{V_M}{V_D}$ .

## 10.3. ПРОЕКЦИОННАЯ ПЕРСПЕКТИВА

Далее мы выведем РСА как алгоритм, который напрямую минимизирует среднюю ошибку реконструкции. Эта перспектива позволяет нам интерпретировать РСА как реализацию оптимального линейного автокодировщика. Мы будем опираться на главы 2 и 3.

В предыдущем разделе мы вывели РСА, максимизируя дисперсию в проецируемом пространстве, чтобы сохранить как можно больше информации. Далее мы рассмотрим векторы разности между исходными данными  $\mathbf{x}_n$  и их реконструкцией  $\tilde{\mathbf{x}}_n$  и минимизируем это расстояние так, чтобы  $\mathbf{x}_n$  и  $\tilde{\mathbf{x}}_n$  были как можно ближе. Рисунок 10.6 иллюстрирует эту настройку.



**Рис. 10.6.** Иллюстрация подхода к проекции: найдите подпространство (прямую), которое минимизирует длину вектора разности между проецируемыми (светлые линии) и исходными (черные точки) данными

### 10.3.1. Настройка и цели

Предположим (упорядоченный) ортонормированный базис (ОНБ)  $B = (\mathbf{b}_1, \dots, \mathbf{b}_D)$   $\mathbb{R}^D$ , то есть  $\mathbf{b}_i^\top \mathbf{b}_j = 1$  тогда и только тогда, когда  $i = j$ , и 0 в противном случае.

Из раздела 2.5 мы знаем, что для базиса  $(\mathbf{b}_1, \dots, \mathbf{b}_D)$   $\mathbb{R}^D$  любое  $x \in \mathbb{R}^D$  можно записать как линейную комбинацию базисных векторов  $\mathbb{R}^D$ <sup>1</sup>, то есть

$$\mathbf{x} = \sum_{d=1}^D \zeta_d \mathbf{b}_d = \sum_{m=1}^M \zeta_m \mathbf{b}_m + \sum_{j=M+1}^D \zeta_j \mathbf{b}_j \quad (10.26)$$

для подходящих координат  $\zeta_d \in \mathbb{R}$ .

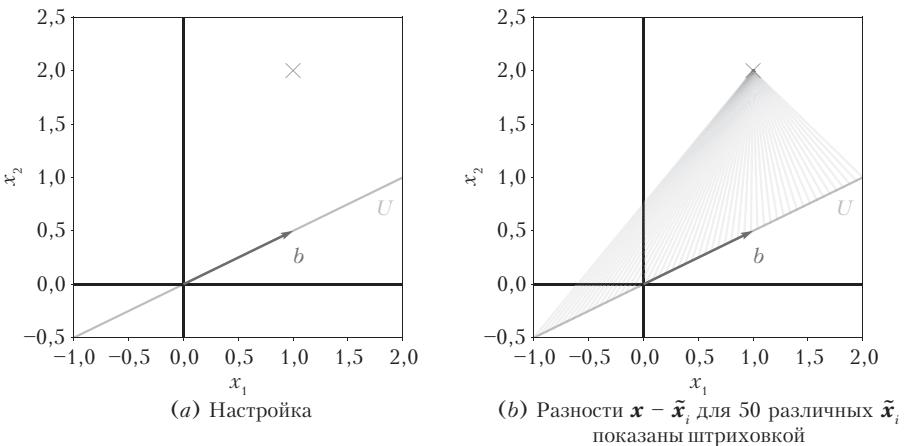
<sup>1</sup> Векторы  $\tilde{\mathbf{x}} \in U$  могут быть векторами на плоскости в  $\mathbb{R}^3$ . Размерность плоскости равна 2, но векторы по-прежнему имеют три координаты относительно стандартного базиса  $\mathbb{R}^3$ .

Нас интересует нахождение векторов  $\tilde{\mathbf{x}} \in \mathbb{R}^D$ , которые живут в подпространстве  $U \subseteq \mathbb{R}^D$  меньшей размерности,  $\dim(U) = M$ , так что

$$\tilde{\mathbf{x}} = \sum_{m=1}^M z_m \mathbf{b}_m \in U \subseteq \mathbb{R}^D \quad (10.27)$$

максимально похоже на  $\mathbf{x}$ . Отметим, что в этой точке нам нужно предположить, что координаты  $z_m$  точки  $\tilde{\mathbf{x}}$  и  $\zeta_m$  точки  $\mathbf{x}$  не идентичны.

В дальнейшем мы будем использовать именно такое представление  $\tilde{\mathbf{x}}$  для нахождения оптимальных координат  $z$  и базисных векторов  $\mathbf{b}_1, \dots, \mathbf{b}_M$ , таких что  $\tilde{\mathbf{x}}$  аналогичен исходной точке данных  $\mathbf{x}$ , то есть мы стремимся минимизировать (евклидово) расстояние  $\|\mathbf{x} - \tilde{\mathbf{x}}\|$ . Рисунок 10.7 иллюстрирует эту настройку.



**Рис. 10.7.** Упрощенная настройка проекции. (a) Вектор  $x \in \mathbb{R}^2$  (крестик) должен быть спроектирован на одномерное подпространство  $U \subseteq \mathbb{R}^2$ , натянутое на  $b$ . (b) Показаны разностные векторы между  $x$  и некоторыми кандидатами  $\tilde{x}$

Без ограничения общности мы предполагаем, что набор данных  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_n \in \mathbb{R}^D$ , с центром в  $\mathbf{0}$ , то есть  $\mathbb{E}[\mathcal{X}] = \mathbf{0}$ . Без предположения о нулевом среднем мы пришли бы к точно такому же решению, но обозначения были бы значительно более загроможденными.

Нас интересует поиск наилучшей линейной проекции на подпространство  $U$  меньшей размерности в  $\mathbb{R}^D$  с  $\dim(U) = M$  и ортонормированными базисными векторами  $\mathbf{b}_1, \dots, \mathbf{b}_M$ . Назовем это подпространство  $U$  главным подпространством. Проекции точек данных обозначены:

$$\tilde{\mathbf{x}}_n := \sum_{m=1}^M z_{mn} \mathbf{b}_m = \mathbf{B} \mathbf{z}_n \in \mathbb{R}^D, \quad (10.28)$$

где  $\mathbf{z}_n := [z_{1n}, \dots, z_{Mn}]^\top \in \mathbb{R}^M$  — вектор координат  $\tilde{\mathbf{x}}_n$  относительно базиса  $(\mathbf{b}_1, \dots, \mathbf{b}_M)$ . В частности, мы заинтересованы в том, чтобы  $\tilde{\mathbf{x}}_n$  было как можно более похожим на  $\mathbf{x}_n$ .

Мера подобия, которую мы используем ниже, — это квадрат евклидовой нормы  $\|\mathbf{x} - \tilde{\mathbf{x}}\|^2$  между  $\mathbf{x}$  и  $\tilde{\mathbf{x}}$ . Поэтому мы определяем нашу цель как минимизацию среднего квадрата евклидова расстояния (ошибка реконструкции) (Pearson, 1901)

$$J_M := \frac{1}{N} \sum_{n=1}^N \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2, \quad (10.29)$$

где мы явно указываем, что размерность подпространства, на которое мы проецируем данные, равна  $M$ . Чтобы найти эту оптимальную линейную проекцию, нам нужно найти ортонормированный базис главного подпространства и координаты  $\mathbf{z}_n \in \mathbb{R}^M$  проекций относительно этого базиса.

Чтобы найти координаты  $\mathbf{z}_n$  и ОНБ главного подпространства, мы следуем двухэтапному подходу. Сначала оптимизируем координаты  $\mathbf{z}_n$  для данного ОНБ  $(\mathbf{b}_1, \dots, \mathbf{b}_M)$ ; во-вторых, находим оптимальный ОНБ.

### 10.3.2. Поиск оптимальных координат

Начнем с поиска оптимальных координат  $z_{1n}, \dots, z_{Mn}$  проекций  $\tilde{\mathbf{x}}_n$  при  $n = 1, \dots, N$ . Рассмотрим рис. 10.8(b), где главное подпространство натянуто на один вектор  $\mathbf{b}$ .

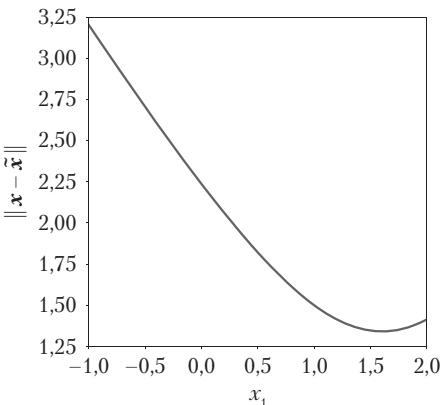
С геометрической точки зрения, нахождение оптимальных координат  $z$  соответствует нахождению представления линейной проекции  $\tilde{\mathbf{x}}$  относительно  $\mathbf{b}$ , которая минимизирует расстояние между  $\tilde{\mathbf{x}} - \mathbf{x}$ . Из рис. 10.8(b) ясно, что это будет ортогональная проекция, и ниже мы покажем именно это.

Предположим, что  $U \subseteq \mathbb{R}^D$  есть ОНБ  $(\mathbf{b}_1, \dots, \mathbf{b}_M)$ . Чтобы найти оптимальные координаты  $\mathbf{z}_m$  относительно этого базиса, нам потребуются частные производные

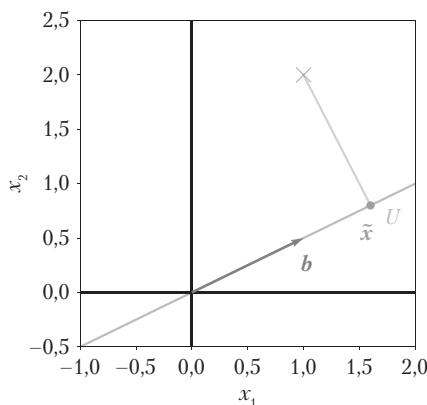
$$\frac{\partial J_M}{\partial z_{in}} = \frac{\partial J_M}{\partial \tilde{\mathbf{x}}_n} \frac{\partial \tilde{\mathbf{x}}_n}{\partial z_{in}}; \quad (10.30a)$$

$$\frac{\partial J_M}{\partial \tilde{\mathbf{x}}_n} = -\frac{2}{N} (\mathbf{x}_n - \tilde{\mathbf{x}}_n)^\top \in \mathbb{R}^{1 \times D}; \quad (10.30b)$$

$$\frac{\partial \tilde{\mathbf{x}}_n}{\partial z_{in}} \stackrel{(10.28)}{=} \frac{\partial}{\partial z_{in}} \left( \sum_{m=1}^M z_{mn} \mathbf{b}_m \right) = \mathbf{b}_i \quad (10.30c)$$



(a) Расстояния  $\|\mathbf{x} - \tilde{\mathbf{x}}\|$  для некоторого  $\tilde{\mathbf{x}} = z_1 \mathbf{b} \in U = \text{span}[\mathbf{b}]$ ; см. (b) для настройки



(b) Вектор  $\tilde{\mathbf{x}}$ , который минимизирует расстояние на (a), является его ортогональной проекцией на  $U$ . Координата проекции  $\tilde{\mathbf{x}}$  относительно базисного вектора  $\mathbf{b}$ , который охватывает  $U$ , является множителем, который нам нужен для масштабирования  $\mathbf{b}$ , чтобы «дотянуться» до  $\mathbf{x}$

**Рис. 10.8.** Оптимальная проекция вектора  $\mathbf{x} \in \mathbb{R}^2$  на одномерное подпространство (продолжение рис. 10.7). (a) Расстояния  $\|\mathbf{x} - \tilde{\mathbf{x}}\|$  для некоторого  $\tilde{\mathbf{x}} \in U$ .  
(b) Ортогональная проекция и оптимальные координаты

для  $i = 1, \dots, M$ , так что получаем

$$\frac{\partial J_M}{\partial z_{in}} \stackrel{(10.30b)}{=} -\frac{2}{N} (\mathbf{x}_n - \tilde{\mathbf{x}}_n)^T \mathbf{b}_i \stackrel{(10.28)}{=} -\frac{2}{N} \left( \mathbf{x}_n - \sum_{m=1}^M z_{mn} \mathbf{b}_m \right)^T \mathbf{b}_i = \quad (10.31a)$$

$$= -\frac{2}{N} (\mathbf{x}_n^T \mathbf{b}_i - z_{in} \mathbf{b}_i^T \mathbf{b}_i) = -\frac{2}{N} (\mathbf{x}_i^T \mathbf{b}_i - z_{in}), \quad (10.31b)$$

так как  $\mathbf{b}_i^T \mathbf{b}_i = 1$ . Установка этой частной производной на 0 сразу же дает оптимальные координаты

$$z_{in} = \mathbf{x}_n^T \mathbf{b}_i = \mathbf{b}_i^T \mathbf{x}_n \quad (10.32)$$

для  $i = 1, \dots, M$  и  $n = 1, \dots, N$ . Это означает, что оптимальные координаты  $z_{in}$  проекции  $\tilde{\mathbf{x}}_n$  являются координатами ортогональной проекции (раздел 3.8) исходной точки данных  $\mathbf{x}_n$  на одномерное подпространство, натянутое на  $\mathbf{b}_i$ . Следовательно:

- Оптимальная линейная проекция  $\tilde{\mathbf{x}}_n$  точки  $\mathbf{x}_n$  является ортогональной.

- Координаты  $\tilde{\mathbf{x}}_n$  относительно базиса  $(\mathbf{b}_1, \dots, \mathbf{b}_M)$  являются координатами ортогональной проекции  $\mathbf{x}_n$  на главное подпространство.
- Ортогональная проекция — лучшее линейное отображение с учетом цели (10.29).
- Координаты  $\zeta_m$  точки  $\mathbf{x}$  в (10.26) и координаты  $z_m$  точки  $\tilde{\mathbf{x}}$  в (10.27) должны быть идентичны для  $m = 1, \dots, M$ , поскольку  $U^\perp = \text{span} [\mathbf{b}_{M+1}, \dots, \mathbf{b}_D]$  является ортогональным дополнением (раздел 3.6) к  $U = \text{span} [\mathbf{b}_1, \dots, \mathbf{b}_M]$ .

**ПРИМЕЧАНИЕ** Кратко напомним ортогональные проекции из раздела 3.8. Если  $(\mathbf{b}_1, \dots, \mathbf{b}_D)$  — ортонормированный базис  $\mathbb{R}^D$ , то

$$\tilde{\mathbf{x}} = \mathbf{b}_j (\mathbf{b}_j^\top \mathbf{b}_j)^{-1} \mathbf{b}_j^\top \mathbf{x} = \mathbf{b}_j \mathbf{b}_j^\top \mathbf{x} \in \mathbb{R}^D \quad (10.33)$$

— ортогональная проекция  $\mathbf{x}$  на подпространство, натянутое на  $j$ -й базисный вектор,  $z_j = \mathbf{b}_j^\top \mathbf{x}$  — координата этой проекции относительно базисного вектора  $\mathbf{b}_j$ , который охватывает это подпространство<sup>1</sup>, поскольку  $z_j \mathbf{b}_j = \tilde{\mathbf{x}}$ . Рисунок 10.8(b) иллюстрирует эту настройку.

В более общем смысле, если мы стремимся проецировать на  $M$ -мерное подпространство  $\mathbb{R}^D$ , мы получаем ортогональную проекцию  $\mathbf{x}$  на  $M$ -мерное подпространство с ортонормированными базисными векторами  $\mathbf{b}_1, \dots, \mathbf{b}_M$  как

$$\tilde{\mathbf{x}} = \mathbf{B} (\underbrace{\mathbf{B}^\top \mathbf{B}}_{=I})^{-1} \mathbf{B}^\top \mathbf{x} = \mathbf{B} \mathbf{B}^\top \mathbf{x}, \quad (10.34)$$

где мы определили  $\mathbf{B} := [\mathbf{b}_1, \dots, \mathbf{b}_M] \in \mathbb{R}^{D \times M}$ . Координаты этой проекции относительно упорядоченного базиса  $(\mathbf{b}_1, \dots, \mathbf{b}_M)$  равны  $\mathbf{z} := \mathbf{B}^\top \mathbf{x}$ , как описано в разделе 3.8.

Мы можем думать о координатах как о представлении спроектированного вектора в новой системе координат, определяемой  $(\mathbf{b}_1, \dots, \mathbf{b}_M)$ . Заметим, что хотя  $\tilde{\mathbf{x}} \in \mathbb{R}^D$ , нам нужно только  $M$  координат  $z_1, \dots, z_M$  для представления этого вектора; остальные координаты  $D - M$  относительно базисных векторов  $(\mathbf{b}_{M+1}, \dots, \mathbf{b}_D)$  всегда равны 0. ♦

До сих пор мы показали, что для данного ОНБ мы можем найти оптимальные координаты  $\tilde{\mathbf{x}}$  с помощью ортогональной проекции на главное подпространство. Далее мы определим, какой базис является наилучшим.

### 10.3.3. Нахождение базиса главного подпространства

Для определения базисных векторов  $\mathbf{b}_1, \dots, \mathbf{b}_M$  главного подпространства мы перефразируем функцию потерь (10.29), используя полученные до сих пор

<sup>1</sup>  $\mathbf{b}_j^\top \mathbf{x}$  — координата ортогональной проекции  $\mathbf{x}$  на подпространство, натянутое на  $\mathbf{b}_j$ .

результаты. Это упростит поиск базисных векторов. Чтобы переформулировать функцию потерь, мы используем наши предыдущие результаты и получаем

$$\tilde{\mathbf{x}}_n = \sum_{m=1}^M z_{mn} \mathbf{b}_m \stackrel{(10.32)}{=} \sum_{m=1}^M (\mathbf{x}_n^\top \mathbf{b}_m) \mathbf{b}_m. \quad (10.35)$$

Теперь мы используем симметрию скалярного произведения, которая дает

$$\tilde{\mathbf{x}}_n = \left( \sum_{m=1}^M \mathbf{b}_m \mathbf{b}_m^\top \right) \mathbf{x}_n. \quad (10.36)$$

Поскольку в общем случае мы можем записать исходную точку данных  $\mathbf{x}_n$  как линейную комбинацию всех базисных векторов, то

$$\mathbf{x}_n = \sum_{d=1}^D z_{dn} \mathbf{b}_d \stackrel{(10.32)}{=} \sum_{d=1}^D (\mathbf{x}_n^\top \mathbf{b}_d) \mathbf{b}_d = \left( \sum_{d=1}^D \mathbf{b}_d \mathbf{b}_d^\top \right) \mathbf{x}_n = \quad (10.37a)$$

$$= \left( \sum_{m=1}^M \mathbf{b}_m \mathbf{b}_m^\top \right) \mathbf{x}_n + \left( \sum_{j=M+1}^D \mathbf{b}_j \mathbf{b}_j^\top \right) \mathbf{x}_n, \quad (10.37b)$$

где мы разбиваем сумму с  $D$  членами на сумму по  $M$  и сумму по  $D - M$  слагаемых. С этим результатом мы находим, что вектор смещения  $\mathbf{x}_n - \tilde{\mathbf{x}}_n$ , то есть вектор разности между исходной точкой данных и ее проекцией равен

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \left( \sum_{j=M+1}^D \mathbf{b}_j \mathbf{b}_j^\top \right) \mathbf{x}_n = \quad (10.38a)$$

$$= \sum_{j=M+1}^D (\mathbf{n}_n^\top \mathbf{b}_j) \mathbf{b}_j. \quad (10.38b)$$

Это означает, что разница — это в точности проекция точки данных на ортогональное дополнение главного подпространства: мы идентифицируем матрицу  $\sum_{j=M+1}^D \mathbf{b}_j \mathbf{b}_j^\top$  в (10.38a) как матрицу проекции, которая выполняет эту проекцию. Следовательно, вектор смещения  $\mathbf{x}_n - \tilde{\mathbf{x}}_n$  лежит в подпространстве, ортогональном главному подпространству, как показано на рис. 10.9.

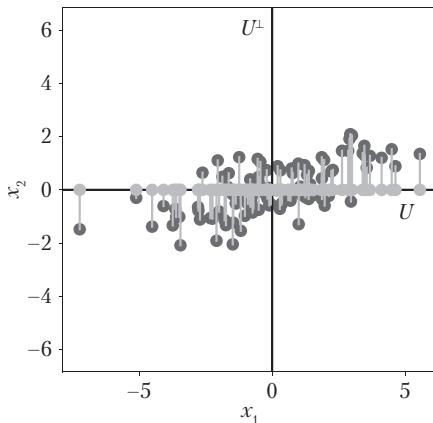
**ПРИМЕЧАНИЕ** В (10.38a) мы видели, что матрица проекции, проецирующая  $\mathbf{x}$  на  $\tilde{\mathbf{x}}$ , имеет вид

$$\sum_{m=1}^M \mathbf{b}_m \mathbf{b}_m^\top = \mathbf{B} \mathbf{B}^\top. \quad (10.39)$$

По построению это сумма матриц  $\mathbf{b}_m \mathbf{b}_m^\top$  первого ранга, и мы видим, что  $\mathbf{B} \mathbf{B}^\top$  симметрична и имеет ранг  $M$ . Следовательно, средний квадрат ошибки восстановления также может быть записан как

$$\frac{1}{N} \sum_{n=1}^N \| \mathbf{x}_n - \tilde{\mathbf{x}}_n \|^2 = \frac{1}{N} \sum_{n=1}^N \| \mathbf{x}_n - \mathbf{B}\mathbf{B}^T \mathbf{x}_n \|^2 = \quad (10.40a)$$

$$= \frac{1}{N} \sum_{n=1}^N \| (\mathbf{I} - \mathbf{B}\mathbf{B}^T) \mathbf{x}_n \|^2. \quad (10.40b)$$



**Рис. 10.9.** Ортогональные векторы проекции и смещения. При проецировании точек данных  $\mathbf{x}_n$  (чёрные кружки) на подпространство  $U_1$  мы получаем  $\tilde{\mathbf{x}}_n$  (светлым). Вектор смещения  $\tilde{\mathbf{x}}_n - \mathbf{x}_n$  полностью лежит в ортогональном дополнении  $U_2$  к  $U_1$

Нахождение ортонормированных базисных векторов  $\mathbf{b}_1, \dots, \mathbf{b}_M$ , которые минимизируют разницу между исходными данными  $\mathbf{x}_n$  и их проекциями  $\tilde{\mathbf{x}}_n$ , эквивалентно поиску наилучшего приближения  $\mathbf{B}\mathbf{B}^T$  ранга  $M$  единичной матрицы  $\mathbf{I}$  (раздел 4.6)<sup>1</sup>. ♦

Теперь у нас есть все инструменты для переформулирования функции потерь (10.29):

$$J_M = \frac{1}{N} \sum_{n=1}^N \| \mathbf{x}_n - \tilde{\mathbf{x}}_n \|^2 \stackrel{(10.38b)}{=} \frac{1}{N} \sum_{n=1}^N \left\| \sum_{j=M+1}^D (\mathbf{b}_j^T \mathbf{x}_n) \mathbf{b}_j \right\|^2. \quad (10.41)$$

Теперь мы явно вычисляем квадрат нормы и используем тот факт, что  $\mathbf{b}_j$  образуют ОНБ, что дает

$$J_M = \frac{1}{N} \sum_{n=1}^N \sum_{j=M+1}^D (\mathbf{b}_j^T \mathbf{x}_n)^2 = \frac{1}{N} \sum_{n=1}^N \sum_{j=M+1}^D \mathbf{b}_j^T \mathbf{x}_n \mathbf{b}_j^T \mathbf{x}_n = \quad (10.42a)$$

$$= \frac{1}{N} \sum_{n=1}^N \sum_{j=M+1}^D \mathbf{b}_j^T \mathbf{x}_n \mathbf{x}_n^T \mathbf{b}_j, \quad (10.42b)$$

<sup>1</sup> РСА находит наилучшее приближение ранга  $M$  единичной матрицы.

где мы использовали симметрию скалярного произведения на последнем шаге, чтобы записать  $\mathbf{b}_j^T \mathbf{x}_n = \mathbf{x}_n^T \mathbf{b}_j$ . Теперь меняем местами суммы и получаем

$$J_M = \sum_{j=M+1}^D \mathbf{b}_j^T \underbrace{\left( \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n \mathbf{x}_n^T \right)}_{=:S} \mathbf{b}_j = \sum_{j=M+1}^D \mathbf{b}_j^T \mathbf{S} \mathbf{b}_j = \quad (10.43a)$$

$$= \sum_{j=M+1}^D \text{tr}(\mathbf{b}_j^T \mathbf{S} \mathbf{b}_j) = \sum_{j=M+1}^D \text{tr}(\mathbf{S} \mathbf{b}_j \mathbf{b}_j^T) = \text{tr} \left( \underbrace{\left( \sum_{j=M+1}^D \mathbf{b}_j \mathbf{b}_j^T \right)}_{\text{матрица проекции}} \mathbf{S} \right), \quad (10.43b)$$

где мы воспользовались тем свойством, что оператор следа  $\text{tr}(\cdot)$  (4.18) является линейным и инвариантным по отношению к циклическим перестановкам его аргументов. Поскольку мы предположили, что наш набор данных центрирован, то есть  $\mathbb{E}[\mathcal{X}] = \mathbf{0}$ , мы идентифицируем  $\mathbf{S}$  как ковариационную матрицу данных. Поскольку матрица проекции в (10.43b) построена как сумма матриц  $\mathbf{b}_j \mathbf{b}_j^T$  первого ранга, она сама имеет ранг  $D - M$ .

Уравнение (10.43a) подразумевает, что мы можем сформулировать средний квадрат ошибки<sup>1</sup> восстановления эквивалентно как ковариационную матрицу данных, спроектированную на ортогональное дополнение главного подпространства. Таким образом, минимизация среднего квадрата ошибки восстановления эквивалентна минимизации дисперсии данных при проецировании на подпространство, которое мы игнорируем, то есть ортогональное дополнение к главному подпространству. Точно так же мы максимизируем дисперсию проекции, которую мы сохраняем в главном подпространстве, которое непосредственно связывает потерю проекции с формулировкой РСА с максимальной дисперсией, обсуждаемой в разделе 10.2. Но тогда это также означает, что мы получим то же решение, что и для перспективы максимальной дисперсии<sup>2</sup>. Поэтому мы опускаем вывод, который идентичен приведенному в разделе 10.2, и суммируем полученные ранее результаты в свете проекционной перспективы. Средний квадрат ошибки восстановления при проецировании на  $M$ -мерное главное подпространство составляет

$$J_M = \sum_{j=M+1}^D \lambda_j, \quad (10.44)$$

где  $\lambda_j$  — собственные значения ковариационной матрицы данных. Следовательно, чтобы минимизировать (10.44), нам нужно выбрать наименьшие собственные

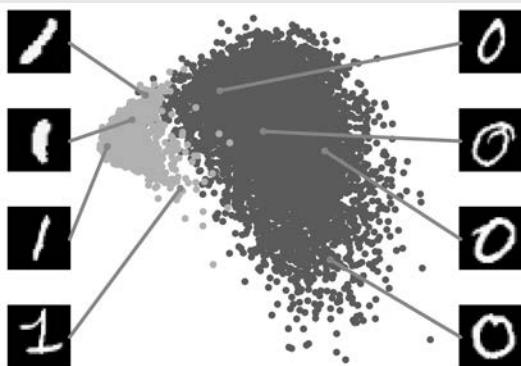
<sup>1</sup> Минимизация среднего квадрата ошибки восстановления эквивалентна минимизации проекции ковариационной матрицы данных на ортогональное дополнение главного подпространства.

<sup>2</sup> Минимизация средней квадратичной ошибки восстановления эквивалентна максимальному увеличению дисперсии прогнозируемых данных.

значения  $D - M$ , из чего следует, что соответствующие им собственные векторы являются базисом ортогонального дополнения главного подпространства. Следовательно, это означает, что в базис главного подпространства входят собственные векторы  $\mathbf{b}_1, \dots, \mathbf{b}_M$ , которые связаны с наибольшими  $M$  собственными значениями ковариационной матрицы данных.

### Пример 10.3 (встраивание цифр MNIST)

На рис. 10.10 показаны обучающие данные цифров «0» и «1» MNIST, встроенные в векторное подпространство, охватываемое первыми двумя главными компонентами. Мы наблюдаем относительно четкое разделение между «0» (темные точки) и «1» (светлые точки), и мы видим вариации внутри каждого отдельного кластера. Четыре вложения цифров «0» и «1» в главное подпространство показаны линиями вместе с соответствующими исходными цифрами. Рисунок показывает, что вариация в наборе «0» значительно больше, чем вариация в наборе «1».



**Рис. 10.10.** Встраивание цифр 0 (темным) и 1 (светлым) MNIST в двумерное главное подпространство с использованием РСА. Четыре вложения цифр «0» и «1» в главное подпространство показаны линиями вместе с соответствующими исходными цифрами

## 10.4. ВЫЧИСЛЕНИЕ СОБСТВЕННОГО ВЕКТОРА И ПРИБЛИЖЕНИЯ НИЗКОГО РАНГА

В предыдущих разделах мы получили базис главного подпространства как собственные векторы, которые связаны с наибольшими собственными значениями ковариационной матрицы данных

$$S = \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n \mathbf{x}_n^T = \frac{1}{N} \mathbf{X} \mathbf{X}^T; \quad (10.45)$$

$$\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{D \times N}. \quad (10.46)$$

Обратите внимание, что  $\mathbf{X}$  – это матрица  $D \times N$ , то есть это транспонированная матрица «типичных» данных (Bishop, 2006; Murphy, 2012). Чтобы получить собственные значения (и соответствующие собственные векторы<sup>1</sup>)  $\mathbf{S}$ , мы можем использовать два подхода:

- Мы выполняем собственное разложение (раздел 4.2) и вычисляем собственные значения и собственные векторы  $\mathbf{S}$  напрямую.
- Мы используем разложение по сингулярным числам (раздел 4.5). Поскольку  $\mathbf{S}$  симметрична и разлагается на  $\mathbf{X}\mathbf{X}^T$  (без учета множителя  $\frac{1}{N}$ ), собственные значения  $\mathbf{S}$  являются квадратами сингулярных значений  $\mathbf{X}$ .

Более конкретно, сингулярное разложение (SVD)  $\mathbf{X}$  определяется как

$$\underbrace{\mathbf{X}}_{D \times N} = \underbrace{\mathbf{U}}_{D \times D} \underbrace{\Sigma}_{D \times N} \underbrace{\mathbf{V}^T}_{N \times N}, \quad (10.47)$$

где  $\mathbf{U} \in \mathbb{R}^{D \times D}$  и  $\mathbf{V}^T \in \mathbb{R}^{N \times N}$  – ортогональные матрицы, а  $\Sigma \in \mathbb{R}^{D \times N}$  – матрица, единственными ненулевыми элементами которой являются сингулярные значения  $\sigma_{ii} \geq 0$ . Отсюда следует, что

$$\mathbf{S} = \frac{1}{N} \mathbf{X} \mathbf{X}^T = \frac{1}{N} \mathbf{U} \Sigma \underbrace{\mathbf{V}^T \mathbf{V}}_{=I_N} \Sigma^T \mathbf{U}^T = \frac{1}{N} \mathbf{U} \Sigma \Sigma^T \mathbf{U}^T. \quad (10.48)$$

Используя результаты раздела 4.5, мы получаем, что столбцы  $\mathbf{U}$  являются собственными векторами  $\mathbf{X}\mathbf{X}^T$  (и, следовательно,  $\mathbf{S}$ ). Кроме того, собственные значения  $\lambda_d$  матрицы  $\mathbf{S}$  связаны с сингулярными значениями  $\mathbf{X}$  через

$$\lambda_d = \frac{\sigma_d^2}{N}. \quad (10.49)$$

Эта связь между собственными значениями  $\mathbf{S}$  и сингулярными значениями  $\mathbf{X}$  обеспечивает связь между представлением максимальной дисперсии (раздел 10.2) и разложением по сингулярным значениям.

### 10.4.1. PCA с использованием матричных приближений низкого ранга

Чтобы максимизировать дисперсию проецируемых данных (или минимизировать среднюю квадратичную ошибку восстановления), PCA выбирает столбцы  $\mathbf{U}$  в (10.48) как собственные векторы, которые связаны с  $M$  наибольшими собственными значениями ковариационной матрицы данных  $\mathbf{S}$ , так что мы иден-

---

<sup>1</sup> Используйте собственное разложение либо SVD для вычисления собственных векторов.

тифицируем  $\mathbf{U}$  как матрицу проекции  $\mathbf{B}$  в (10.3), которая проецирует исходные данные на подпространство меньшей размерности — размерности  $M$ . Теорема Эккарта — Юнга (теорема 4.25 в разделе 4.6) предлагает прямой способ оценки низкоразмерного представления. Рассмотрим наилучшее приближение  $\tilde{\mathbf{X}}$  ранга  $M$

$$\tilde{\mathbf{X}}_M := \arg \min_{\text{rk}(\mathbf{A}) \leq M} \|\mathbf{X} - \mathbf{A}\|_2 \in \mathbb{R}^{D \times N}, \quad (10.50)$$

где  $\|\cdot\|_2$  — спектральная норма, определенная в (4.93). Теорема Эккарта — Юнга утверждает, что  $\tilde{\mathbf{X}}_M$  задается усечением SVD в сингулярном значении  $\text{top-}M$ . Другими словами, получаем

$$\tilde{\mathbf{X}}_M = \underbrace{\mathbf{U}_M}_{D \times M} \underbrace{\Sigma_M}_{M \times M} \underbrace{\mathbf{V}_M^\top}_{M \times N} \in \mathbb{R}^{D \times N} \quad (10.51)$$

с ортогональными матрицами  $\mathbf{U}_M := [\mathbf{u}_1, \dots, \mathbf{u}_M] \in \mathbb{R}^{D \times M}$  и  $\mathbf{V}_M := [\mathbf{v}_1, \dots, \mathbf{v}_M] \in \mathbb{R}^{N \times M}$  и диагональной матрицей  $\Sigma_M \in \mathbb{R}^{M \times M}$ , диагональные элементы которой являются  $M$  наибольшими сингулярными значениями матрицы  $\mathbf{X}$ .

## 10.4.2. Практические аспекты

Нахождение собственных значений и собственных векторов также важно в других фундаментальных методах машинного обучения, которые требуют разложения матриц. Теоретически, как мы обсуждали в разделе 4.2, мы можем найти собственные значения как корни характеристического полинома. Однако для матриц больше  $4 \times 4$  это невозможно, потому что нам нужно будет найти корни многочлена степени 5 или выше. Однако *теорема Абеля — Рuffини* (Ruffini, 1799; Abel, 1826) утверждает, что не существует алгебраического решения этой проблемы для многочленов степени 5 или выше. Поэтому на практике мы находим собственные значения или сингулярные значения, используя итерационные методы, которые реализованы во всех современных пакетах линейной алгебры<sup>1</sup>.

Во многих приложениях (например, в PCA, представленном в этой главе) нам требуется только несколько собственных векторов. Было бы расточительно вычислять полное разложение, а затем отбрасывать все собственные векторы с собственными значениями, превышающими несколько первых. Оказывается, что если нас интересуют только первые несколько собственных векторов (с наибольшими собственными значениями), то итерационные процессы, которые напрямую оптимизируют эти собственные векторы, в вычислительном отношении более эффективны, чем полное собственное разложение (или SVD). В крайнем случае, когда нужен только первый собственный вектор, очень эффективен простой метод, называемый методом *степенных итераций*. Метод степенных

---

<sup>1</sup> См. `np.linalg.eigh` или `np.linalg.svd`.

итераций выбирает случайный вектор  $\mathbf{x}_0$ , который не находится в нулевом пространстве  $\mathbf{S}$ , и следует за итерацией

$$\mathbf{x}_{k+1} = \frac{\mathbf{S}\mathbf{x}_k}{\|\mathbf{S}\mathbf{x}_k\|}, \quad k = 0, 1, \dots . \quad (10.52)$$

Это означает, что вектор  $\mathbf{x}_k$  умножается на  $\mathbf{S}$  на каждой итерации, а затем нормализуется, то есть мы всегда имеем  $\|\mathbf{x}_k\| = 1$ . Эта последовательность векторов сходится к собственному вектору, связанному с наибольшим собственным значением  $\mathbf{S}$ <sup>1</sup>. Исходный алгоритм Google PageRank (Page et al., 1999) использует такой алгоритм для ранжирования веб-страниц на основе их гиперссылок.

## 10.5. PCA В БОЛЬШИХ РАЗМЕРАХ

Чтобы выполнить PCA, нам нужно вычислить ковариационную матрицу данных. В размерностях  $D$  ковариационная матрица данных представляет собой матрицу  $D \times D$ . Вычисление собственных значений и собственных векторов этой матрицы требует больших затрат вычислительных ресурсов, поскольку масштабируется кубическим образом в  $D$ . Следовательно, PCA, как мы обсуждали ранее, будет невозможным в очень больших измерениях. Например, если наши  $\mathbf{x}_n$  — это изображения с 10 000 пикселей (например, изображения  $100 \times 100$  пикселей), нам нужно будет вычислить собственное разложение ковариационной матрицы  $10\,000 \times 10\,000$ . Далее мы предлагаем решение этой проблемы для случая, когда у нас значительно меньше точек данных, чем размеров, то есть  $N \ll D$ .

Предположим, у нас есть центрированный набор данных  $\mathbf{x}_1, \dots, \mathbf{x}_N, \mathbf{x}_n \in \mathbb{R}^D$ . Тогда ковариационная матрица данных имеет вид

$$\mathbf{S} = \frac{1}{N} \mathbf{X} \mathbf{X}^T \in \mathbb{R}^{D \times D}, \quad (10.53)$$

где  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$  — матрица размера  $D \times N$ , столбцы которой являются точками данных.

Теперь предположим, что  $N \ll D$ , то есть количество точек данных меньше размерности данных. Если нет повторяющихся точек данных, ранг ковариационной матрицы  $\mathbf{S}$  равен  $N$ , поэтому она имеет  $D - N + 1$  собственных значений, которые равны 0. Интуитивно это означает, что есть некоторые избыточности. Далее мы воспользуемся этим и превратим ковариационную матрицу  $D \times D$  в ковариационную матрицу  $N \times N$ , все собственные значения которой положительны.

---

<sup>1</sup> Если  $\mathbf{S}$  обратима, достаточно убедиться, что  $\mathbf{x}_0 \neq 0$ .

В PCA мы получили уравнение для собственных векторов

$$\mathbf{S}\mathbf{b}_m = \lambda_m \mathbf{b}_m, m = 1, \dots, M, \quad (10.54)$$

где  $\mathbf{b}_m$  – базисный вектор главного подпространства. Давайте немного перепишем это уравнение: с  $\mathbf{S}$ , определенным в (10.53), мы получаем

$$\mathbf{S}\mathbf{b}_m = \frac{1}{N} \mathbf{X}\mathbf{X}^\top \mathbf{b}_m = \lambda_m \mathbf{b}_m. \quad (10.55)$$

Умножим теперь  $\mathbf{X}^\top \in \mathbb{R}^{N \times D}$  из левой части, что дает

$$\frac{1}{N} \underbrace{\mathbf{X}^\top \mathbf{X}}_{N \times N}_{=: \mathbf{c}_m} \underbrace{\mathbf{X}^\top \mathbf{b}_m}_{=: \mathbf{c}_m} = \lambda_m \mathbf{X}^\top \mathbf{b}_m \Leftrightarrow \frac{1}{N} \mathbf{X}^\top \mathbf{X} \mathbf{c}_m = \lambda_m \mathbf{c}_m, \quad (10.56)$$

и мы получаем новое уравнение «собственный вектор / собственное значение»:  $\lambda_m$  остается собственным значением, что подтверждает наши результаты из раздела 4.5.3 о том, что ненулевые собственные значения  $\mathbf{X}\mathbf{X}^\top$  равны ненулевым собственным значениям  $\mathbf{X}^\top \mathbf{X}$ . Мы получаем собственный вектор матрицы  $\frac{1}{N} \mathbf{X}^\top \mathbf{X} \in \mathbb{R}^{N \times N}$ , ассоциированный с  $\lambda_m$  как  $\mathbf{c}_m := \mathbf{X}^\top \mathbf{b}_m$ . Предполагая, что у нас нет повторяющихся точек данных, эта матрица имеет ранг  $N$  и обратима. Это также означает, что  $\frac{1}{N} \mathbf{X}^\top \mathbf{X}$  имеет те же (ненулевые) собственные значения, что и ковариационная матрица данных  $\mathbf{S}$ . Но теперь это матрица размера  $N \times N$ , так что мы можем вычислять собственные значения и собственные векторы гораздо эффективнее, чем для исходной ковариационной матрицы данных  $D \times D$ .

Теперь, когда у нас есть собственные векторы  $\frac{1}{N} \mathbf{X}^\top \mathbf{X}$ , мы собираемся восстановить исходные собственные векторы, которые нам все еще нужны для PCA. В настоящее время мы знаем собственные векторы  $\frac{1}{N} \mathbf{X}^\top \mathbf{X}$ . Если мы умножим слева наше уравнение «собственное значение / собственный вектор» на  $\mathbf{X}$ , мы получим

$$\underbrace{\frac{1}{N} \mathbf{X}\mathbf{X}^\top}_{\mathbf{S}} \underbrace{\mathbf{X} \mathbf{c}_m}_{=: \mathbf{c}_m} = \lambda_m \mathbf{X} \mathbf{c}_m \quad (10.57)$$

и мы снова восстанавливаем матрицу ковариации данных. Теперь это также означает, что мы восстанавливаем  $\mathbf{X} \mathbf{c}_m$  как собственный вектор  $\mathbf{S}$ .

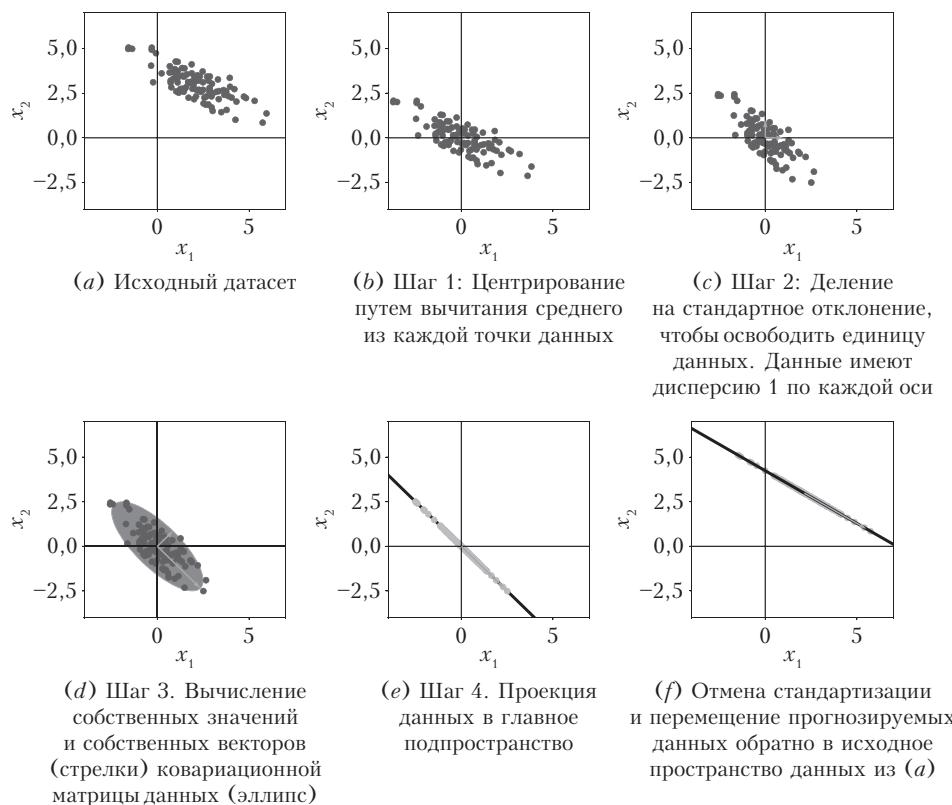
**ПРИМЕЧАНИЕ** Если мы хотим применить алгоритм PCA, который мы обсуждали в разделе 10.6, нам нужно нормализовать собственные векторы  $\mathbf{X} \mathbf{c}_m$  матрицы  $\mathbf{S}$  так, чтобы они имели норму 1. ♦

## 10.6. КЛЮЧЕВЫЕ ШАГИ РСА НА ПРАКТИКЕ

Далее мы рассмотрим отдельные шаги РСА на рабочем примере, который кратко представлен на рис. 10.11. Нам дан двумерный датасет (рис. 10.11(a)), и мы хотим использовать РСА, чтобы спроектировать его на одномерное подпространство.

### 1. Вычитание среднего значения

Мы начинаем с центрирования данных, вычисляя среднее значение  $\mu$  датасета и вычитая его из каждой отдельной точки данных. Это гарантирует, что датасет имеет среднее значение  $\mathbf{0}$  (рис. 10.11(b)). Вычитание среднего не обязательно, но снижает риск вычислительных проблем.



**Рис. 10.11.** Этапы РСА. (a) исходный датасет; (b) центрирование; (c) разделение на стандартное отклонение; (d) собственный состав; (e) проекция; (f) отображение обратно в исходное пространство данных

## 2. Стандартизация

Разделите точки данных на стандартное отклонение  $\sigma_d$  датасета для каждого измерения  $d = 1, \dots, D$ . Теперь данные не содержат единиц измерения и имеют отклонение 1 по каждой оси, что обозначено двумя стрелками на рис. 10.11(d). На этом этапе *стандартизация* данных завершается.

## 3. Собственное разложение ковариационной матрицы

Вычислите матрицу ковариации данных, ее собственные значения и соответствующие собственные векторы. Поскольку ковариационная матрица симметрична, спектральная теорема (теорема 4.15) утверждает, что мы можем найти ОНБ собственных векторов. На рис. 10.11(d) собственные векторы масштабированы по величине соответствующего собственного значения. Более длинный вектор охватывает главное подпространство, которое мы обозначим через  $U$ . Ковариационная матрица данных представлена эллипсом.

## 4. Прогноз

Мы можем спроектировать любую точку данных  $\mathbf{x}_* \in \mathbb{R}^D$  на главное подпространство: чтобы сделать это правильно, нам нужно стандартизировать  $\mathbf{x}_*$ , используя среднее значение  $\mu_d$  и стандартное отклонение  $\sigma_d$  обучающих данных в  $d$ -м измерении соответственно, так что

$$\mathbf{x}_*^{(d)} \leftarrow \frac{\mathbf{x}_*^{(d)} - \mu_d}{\sigma_d}, \quad d = 1, \dots, D, \quad (10.58)$$

где  $\mathbf{x}_*^{(d)}$  —  $d$ -я компонента  $\mathbf{x}_*$ . Получим проекцию как

$$\tilde{\mathbf{x}}_* = \mathbf{B}\mathbf{B}^T \mathbf{x}_* \quad (10.59)$$

с координатами

$$\mathbf{z}_* = \mathbf{B}^T \mathbf{x}_* \quad (10.60)$$

относительно базиса главного подпространства. Здесь  $\mathbf{B}$  — матрица, которая содержит собственные векторы, которые связаны с наибольшими собственными значениями ковариационной матрицы данных в виде столбцов. PCA возвращает координаты (10.60), а не проекции  $\mathbf{x}_*$ .

После стандартизации нашего набора данных (10.59) дает только прогнозы в контексте стандартизированного набора данных. Чтобы получить нашу проекцию в исходном пространстве данных (то есть до стандартизации), нам нужно отменить стандартизацию (10.58) и умножить на стандартное отклонение перед добавлением среднего, чтобы мы получили

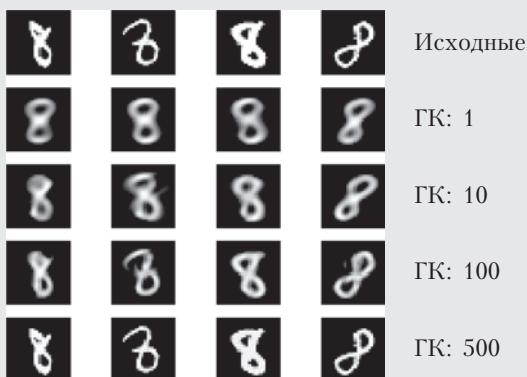
$$\tilde{\mathbf{x}}_*^{(d)} \leftarrow \tilde{\mathbf{x}}_*^{(d)} \sigma_d + \mu_d, \quad d = 1, \dots, D. \quad (10.61)$$

Рисунок 10.11(f) иллюстрирует проекцию в исходное пространство данных.

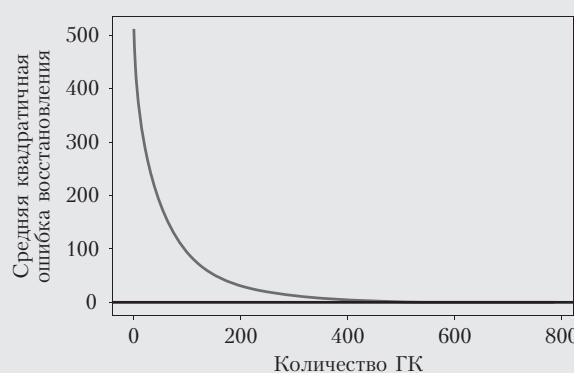
#### Пример 10.4 (цифры MNIST: реконструкция)

Далее мы применим РСА к набору данных цифр MNIST, который содержит 60 000 примеров рукописных цифр от 0 до 9. Каждая цифра представляет собой изображение размером  $28 \times 28$ , то есть содержит 784 пикселя, чтобы мы могли интерпретировать каждое изображение в этот датасет как вектор  $\mathbf{x} \in \mathbb{R}^{784}$ . Примеры этих цифр показаны на рис. 10.3.

В целях иллюстрации мы применяем РСА к подмножеству цифр MNIST и сосредоточиваемся на цифре «8». Мы использовали 5389 обучающих образов цифры «8» и определили главное подпространство, как подробно описано в этой главе. Затем мы использовали изученную матрицу проекций для восстановления набора тестовых изображений, который показан на рис. 10.12. В первой строке рис. 10.12 показан набор из четырех исходных цифр из тестового набора. В следующих строках показаны реконструкции именно этих цифр при использовании главного подпространства размерностей 1, 10, 100 и 500 соответственно. Мы видим, что даже с одномерным основным подпространством мы получаем довольно приличную реконструкцию исходных цифр, которая, однако, является размытой и общей. С увеличением числа главных компонент (ГК) реконструкции становятся более четкими, и учитывается все больше деталей. С 500 главными компонентами мы фактически получаем почти идеальную реконструкцию. Если бы мы выбрали 784 ГК, мы бы восстановили точную цифру без потери сжатия.



**Рис. 10.12.** Влияние увеличения количества главных компонент на реконструкцию



**Рис. 10.13.** Средняя квадратичная ошибка восстановления в зависимости от количества главных компонент. Средний квадрат ошибки восстановления — это сумма собственных значений в ортогональном дополнении к главному подпространству

На рис. 10.13 показан средний квадрат ошибки восстановления, равный

$$\frac{1}{N} \sum_{n=1}^N \| \mathbf{x}_n - \tilde{\mathbf{x}}_n \|^2 = \sum_{i=M+1}^D \lambda_i, \quad (10.62)$$

как функция количества главных компонент  $M$ . Мы видим, что важность главных компонент быстро падает, и лишь незначительный выигрыш может быть достигнут за счет добавления дополнительных ГК. Это в точности совпадает с нашим наблюдением на рис. 10.5, где мы обнаружили, что большая часть дисперсии прогнозируемых данных улавливается только несколькими главными компонентами. Имея около 550 ГК, мы можем практически полностью восстановить обучающие данные, содержащие цифру «8» (некоторые пиксели вокруг границ не показывают изменений в наборе данных, поскольку они всегда черные).

## 10.7. ЛАТЕНТНАЯ ПЕРЕМЕННАЯ

В предыдущих разделах мы вывели РСА без какого-либо понятия вероятностной модели, используя подход с максимальной дисперсией и проекцией. С одной стороны, этот подход может быть привлекательным, поскольку он позволяет нам обойти все математические трудности, связанные с теорией вероятностей, но с другой стороны, вероятностная модель предоставит нам больше гибкости и полезного понимания. В частности, вероятностная модель будет:

- включать в себя функцию правдоподобия, и мы можем явно иметь дело с зашумленными наблюдениями (которые мы даже не обсуждали ранее);
- позволять нам провести сравнение байесовских моделей с помощью предельного правдоподобия, как описано в разделе 8.6;
- рассматривать PCA как генеративную модель, которая позволяет нам моделировать новые данные;
- позволять нам напрямую подключаться к связанным алгоритмам;
- работать со случайно пропущенными измерениями данных, применяя теорему Байеса;
- давать нам представление о новизне новой точки данных;
- давать нам принципиальный способ расширить модель, например до смеси моделей PCA;
- использовать PCA, полученный нами в предыдущих разделах, как особый случай;
- обеспечивать полностью байесовский подход, исключив параметры модели.

Вводя латентную переменную  $\mathbf{z} \in \mathbb{R}^M$  с непрерывным знаком, можно сформулировать PCA как вероятностную модель латентных переменных. Типпинг и Бишоп (Tipping and Bishop, 1999) предложили эту модель латентных переменных как *вероятностный PCA* (PPCA). PPCA решает большинство из вышеупомянутых проблем, и решение PCA, которое мы получили путем максимизации дисперсии в проецируемом пространстве или минимизации ошибки восстановления, получено как частный случай оценки максимального правдоподобия в настройке без шума.

### 10.7.1. Генеративный процесс и вероятностная модель

В PPCA мы явно записываем вероятностную модель для уменьшения линейной размерности. Для этого мы предполагаем непрерывную латентную переменную  $\mathbf{z} \in \mathbb{R}^M$  со стандартным нормальным априорным  $p(\mathbf{z}) = \mathcal{N}(0, I)$  и линейной зависимостью между латентными переменными и наблюдаемыми данными  $\mathbf{x}$ , где

$$\mathbf{x} = \mathbf{B}\mathbf{z} + \mu + \varepsilon \in \mathbb{R}^D, \quad (10.63)$$

где  $\sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$  — гауссов шум наблюдения, а  $\mathbf{B} \in \mathbb{R}^{D \times M}$  и  $\mu \in \mathbb{R}^D$  описывают линейное/аффинное отображение латентных переменных в наблюдаемые. Таким образом, PPCA связывает латентные и наблюдаемые переменные через

$$p(\mathbf{x} | \mathbf{z}, \mathbf{B}, \mu, \sigma^2) = \mathcal{N}(\mathbf{x} | \mathbf{B}\mathbf{z} + \mu, \sigma^2 \mathbf{I}). \quad (10.64)$$

В целом PPCA вызывает следующий процесс генерации:

$$\mathbf{z}_n \sim \mathcal{N}(\mathbf{z} | \mathbf{0}, \mathbf{I}); \quad (10.65)$$

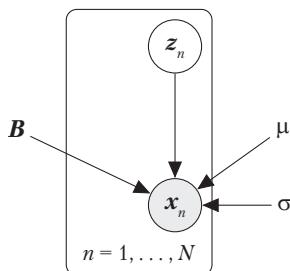
$$\mathbf{x}_n | \mathbf{z}_n \sim \mathcal{N}(\mathbf{x} | \mathbf{B}\mathbf{z}_n + \boldsymbol{\mu}, \sigma^2 \mathbf{I}). \quad (10.66)$$

Чтобы сгенерировать точку данных, которая типична для данных параметров модели, мы следуем схеме наследственного отбора: сначала мы выбираем латентную переменную  $\mathbf{z}_n$  из  $p(\mathbf{z})$ . Затем мы используем  $\mathbf{z}_n$  в (10.64) для выборки точки данных, обусловленной выбранным  $\mathbf{z}_n$ , то есть  $\mathbf{x}_n \sim p(\mathbf{x} | \mathbf{z}_n, \mathbf{B}, \boldsymbol{\mu}, \sigma^2)$ .

Этот процесс генерации позволяет нам записать вероятностную модель (то есть совместное распределение всех случайных величин (раздел 8.4)) как

$$p(\mathbf{x}, \mathbf{z} | \mathbf{B}, \boldsymbol{\mu}, \sigma^2) = p(\mathbf{x} | \mathbf{z}, \mathbf{B}, \boldsymbol{\mu}, \sigma^2)p(\mathbf{z}), \quad (10.67)$$

что сразу дает начало графической модели на рис. 10.14, использующей результаты из раздела 8.5.



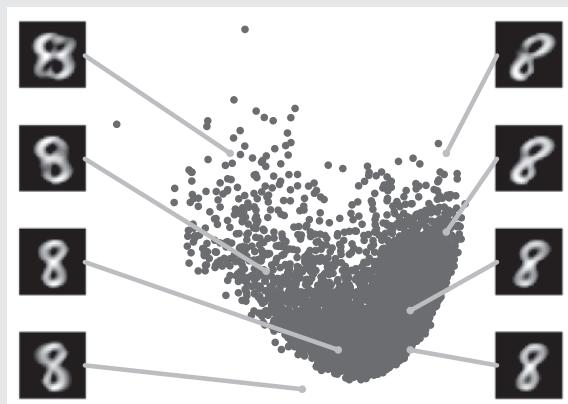
**Рис. 10.14.** Графическая модель для вероятностного РСА. Наблюдения  $\mathbf{x}_n$  явно зависят от соответствующих скрытых переменных  $\mathbf{z}_n \sim \mathcal{N}(0, \mathbf{I})$ . Параметры модели  $\mathbf{B}$ ,  $\boldsymbol{\mu}$  и параметр правдоподобия  $\sigma$  являются общими для всего набора данных

**ПРИМЕЧАНИЕ** Обратите внимание на направление стрелки, которая соединяет латентные переменные  $\mathbf{z}$  и наблюдаемые данные  $\mathbf{x}$ : стрелка указывает от  $\mathbf{z}$  к  $\mathbf{x}$ , что означает, что модель РРСА предполагает скрытую причину более низкой размерности  $\mathbf{z}$  для наблюдений  $\mathbf{x}$  высокой размерности. В конце концов, очевидно, заинтересованы в том, чтобы узнать что-нибудь о  $\mathbf{z}$  с учетом некоторых наблюдений. Для этого мы применим байесовский инференс, чтобы неявно «перевернуть» стрелку и перейти от наблюдений к латентным переменным.

### Пример 10.5 (генерация новых данных с использованием латентных переменных)

На рис. 10.15 показаны скрытые координаты цифр «8» MNIST, найденные РСА при использовании двумерного главного подпространства (точки на рисунке). Мы можем запросить любой вектор  $\mathbf{z}_*$  в этом скрытом пространстве и сгенерировать изображение  $\tilde{\mathbf{x}}_* = \mathbf{B}\mathbf{z}_*$ , напоминающее циф-

ру «8». Мы показываем восемь таких сгенерированных изображений с соответствующим им скрытым пространственным представлением. В зависимости от того, где мы запрашиваем скрытое пространство, сгенерированные изображения выглядят по-разному (форма, поворот, размер и т. д.). Если мы откажемся от обучающих данных, мы увидим все больше и больше артефактов, например верхние левые и верхние правые цифры. Обратите внимание, что внутренняя размерность этих сгенерированных изображений равна только двум.



**Рис. 10.15.** Создание новых цифр MNIST. Латентные переменные  $\mathbf{z}$  можно использовать для генерации новых данных  $\tilde{\mathbf{x}} = \mathbf{B}\mathbf{z}$ . Чем ближе мы приближаемся к обучающим данным, тем реалистичнее получаются данные

## 10.7.2. Правдоподобие и совместное распределение

Используя результаты главы 6, мы получаем правдоподобие<sup>1</sup> этой вероятностной модели путем интегрирования латентной переменной  $\mathbf{z}$  (раздел 8.4.3), так что

$$p(\mathbf{x} | \mathbf{B}, \mu, \sigma^2) = \int p(\mathbf{x} | \mathbf{z}, \mu, \sigma^2) p(\mathbf{z}) d\mathbf{z} = \quad (10.68a)$$

$$= \int \mathcal{N}(\mathbf{x} | \mathbf{B}\mathbf{z} + \mu, \sigma^2 \mathbf{I}) \mathcal{N}(\mathbf{z} | \mathbf{0}, \mathbf{I}) d\mathbf{z}. \quad (10.68b)$$

Из раздела 6.5 мы знаем, что решением этого интеграла является гауссово распределение со средним значением

$$\mathbb{E}_{\mathbf{x}}[\mathbf{x}] = \mathbb{E}_{\mathbf{z}}[\mathbf{B}\mathbf{z} + \mu] + \mathbb{E}_{\varepsilon}[\varepsilon] = \mu \quad (10.69)$$

<sup>1</sup> Правдоподобие не зависит от латентных переменных  $\mathbf{z}$ .

и с ковариационной матрицей

$$\mathbb{V}[\mathbf{x}] = \mathbb{V}_z[\mathbf{B}\mathbf{z} + \boldsymbol{\mu}] + \mathbb{V}_{\varepsilon}[\boldsymbol{\varepsilon}] = \mathbb{V}_z[\mathbf{B}\mathbf{z}] + \sigma^2 \mathbf{I} = \quad (10.70a)$$

$$= \mathbf{B}\mathbb{V}_z[\mathbf{z}] \mathbf{B}^T + \sigma^2 \mathbf{I} = \mathbf{B}\mathbf{B}^T + \sigma^2 \mathbf{I}. \quad (10.70b)$$

Правдоподобие в (10.68b) может использоваться для оценки максимального правдоподобия или MAP параметров модели.

**ПРИМЕЧАНИЕ** Мы не можем использовать условное распределение в (10.64) для оценки максимального правдоподобия, поскольку оно все еще зависит от латентных переменных. Функция правдоподобия, которая нам нужна для оценки максимального правдоподобия (или MAP), должна быть только функцией данных  $\mathbf{x}$  и параметров модели, но не должна зависеть от латентных переменных. ♦

Из раздела 6.5 мы знаем, что гауссова случайная величина  $\mathbf{z}$  и ее линейное/аффинное преобразование  $\mathbf{x} = \mathbf{B}\mathbf{z}$  совместно распределены по Гауссу. Мы уже знаем маргиналы  $p(\mathbf{z}) = \mathcal{N}(\mathbf{z} | \mathbf{0}, \mathbf{I})$  и  $p(\mathbf{x}) = \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \mathbf{B}\mathbf{B}^T + \sigma^2 \mathbf{I})$ . Отсутствующая кросс-ковариация выражается как

$$\text{Cov}[\mathbf{x}, \mathbf{z}] = \text{Cov}_z[\mathbf{B}\mathbf{z} + \boldsymbol{\mu}] = \mathbf{B}\text{Cov}_z[\mathbf{z}, \mathbf{z}] = \mathbf{B}. \quad (10.71)$$

Следовательно, вероятностная модель PPCA, то есть совместное распределение скрытых и наблюдаемых случайных величин, явно задается выражением

$$p(\mathbf{x}, \mathbf{z} | \mathbf{B}, \boldsymbol{\mu}, \sigma^2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{B}\mathbf{B}^T + \sigma^2 \mathbf{I} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{I} \end{bmatrix}\right), \quad (10.72)$$

со средним вектором длины  $D + M$  и ковариационной матрицей размера  $(D + M) \times (D + M)$ .

### 10.7.3. Апостериорное распределение

Совместное гауссово распределение  $p(\mathbf{x}, \mathbf{z} | \mathbf{B}, \boldsymbol{\mu}, \sigma^2)$  в (10.72) позволяет нам определить апостериорное распределение  $p(\mathbf{z} | \mathbf{x})$  немедленно, применяя правила гауссова процесса из раздела 6.5.1. Тогда апостериорное распределение латентной переменной с учетом наблюдения  $\mathbf{x}$  равно

$$p(\mathbf{z} | \mathbf{x}) = \mathcal{N}(\mathbf{z} | \mathbf{m}, \mathbf{C}); \quad (10.73)$$

$$\mathbf{m} = \mathbf{B}^T(\mathbf{B}\mathbf{B}^T + \sigma^2 \mathbf{I})^{-1}(\mathbf{x} - \boldsymbol{\mu}); \quad (10.74)$$

$$\mathbf{C} = \mathbf{I} - \mathbf{B}^T(\mathbf{B}\mathbf{B}^T + \sigma^2 \mathbf{I})^{-1}\mathbf{B}. \quad (10.75)$$

Обратите внимание, что апостериорная ковариация не зависит от наблюдаемых данных  $\mathbf{x}$ . Для нового наблюдения  $\mathbf{x}_*$  в пространстве данных мы используем (10.73) для определения апостериорного распределения соответствующей латентной переменной  $\mathbf{z}_*$ . Ковариационная матрица  $\mathbf{C}$  позволяет нам оценить, насколько надежно вложение. Ковариационная матрица  $\mathbf{C}$  с малым детерминантом (который измеряет объемы) говорит нам, что скрытое вложение  $\mathbf{z}_*$  достаточно надежно. Если мы получим апостериорное распределение  $p(\mathbf{z}_* | \mathbf{x}_*)$  с большой дисперсией, мы можем столкнуться с выбросом. Однако мы можем изучить это апостериорное распределение, чтобы понять, какие другие точки данных  $\mathbf{x}$  вероятны при этом апостериорном распределении. Для этого мы используем процесс генерации, лежащий в основе РРСА, который позволяет нам исследовать апостериорное распределение латентных переменных путем генерации новых данных, которые являются правдоподобными при следующем апостериорном распределении:

1. Выберите латентную переменную  $\mathbf{z}_* \sim p(\mathbf{z} | \mathbf{x}_*)$  из апостериорного распределения по латентным переменным (10.73).
2. Выберите восстановленный вектор  $\tilde{\mathbf{x}}_* \sim p(\mathbf{x} | \mathbf{z}_*, \mathbf{B}, \mu, \sigma^2)$  из (10.64).

Если мы повторим этот процесс много раз, мы сможем исследовать апостериорное распределение (10.73) латентных переменных  $\mathbf{z}_*$  и его влияние на наблюдаемые данные. Процесс выборки эффективно выдвигает гипотезу о данных, что является правдоподобным при апостериорном распределении.

## 10.8. ДОПОЛНИТЕЛЬНОЕ ЧТЕНИЕ

Мы получили РСА с двух точек зрения: (a) максимизация дисперсии в проецируемом пространстве; и (б) минимизация средней ошибки восстановления. Однако РСА также можно интерпретировать с разных точек зрения. Подведем итоги того, что мы сделали: мы взяли данные большой размерности  $\mathbf{x} \in \mathbb{R}^D$  и использовали матрицу  $\mathbf{B}^T$ , чтобы найти представление  $\mathbf{z} \in \mathbb{R}^M$  меньшей размерности. Столбцы  $\mathbf{B}$  являются собственными векторами ковариационной матрицы данных  $\mathbf{S}$ , которые связаны с наибольшими собственными значениями. Когда у нас есть низкоразмерное представление  $\mathbf{z}$ , мы можем получить его многомерную версию (в исходном пространстве данных) как  $\mathbf{x} \approx \tilde{\mathbf{x}}_* = \mathbf{B}\mathbf{z} = \mathbf{B}\mathbf{B}^T\mathbf{x} \in \mathbb{R}^D$ , где  $\mathbf{B}\mathbf{B}^T$  — матрица проекции.

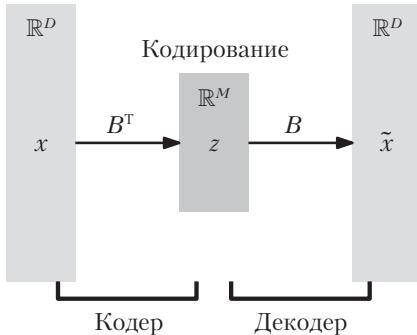
Мы также можем рассматривать РСА как линейный *автокодер*, как показано на рис. 10.16. Автокодер кодирует данные  $\mathbf{x}_n \in \mathbb{R}^D$  в код  $\mathbf{z}_n \in \mathbb{R}^M$  и декодирует их в  $\tilde{\mathbf{x}}_n$ , аналогично  $\mathbf{x}_n$ . Отображение данных в код называется *кодером*, а отображение кода обратно в исходное пространство данных называется *декодером*. Если мы рассмотрим линейные отображения, где код задается как  $\mathbf{z}_n = \mathbf{B}^T\mathbf{x}_n \in \mathbb{R}^M$ ,

и нас интересует минимизация средней квадратичной ошибки между данными  $\mathbf{x}_n$  и их реконструкцией  $\tilde{\mathbf{x}}_n = \mathbf{B}\mathbf{z}_n$ ,  $n = 1, \dots, N$ , мы получим

$$\frac{1}{N} \sum_{n=1}^N \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 = \frac{1}{N} \sum_{n=1}^N \|\mathbf{x}_n - \mathbf{B}^\top \mathbf{B} \mathbf{x}_n\|^2. \quad (10.76)$$

Это означает, что мы получаем ту же целевую функцию, что и в (10.29), которую мы обсуждали в разделе 10.3, так что мы получаем решение РСА, когда минимизируем возвещенные в квадрат потери автокодирования. Если мы заменим линейное отображение РСА на нелинейное отображение, мы получим нелинейный автокодер. Ярким примером этого является глубокий автокодер, в котором линейные функции заменены глубокими нейронными сетями. В этом контексте кодер также известен как *сеть распознавания*, или *сеть вывода*, тогда как декодер также называется *генератором*.

Исходный



**Рис. 10.16.** РСА можно рассматривать как линейный автокодер. Он кодирует данные  $\mathbf{x}$  большой размерности в представление (код) меньшей размерности  $\mathbf{z} \in \mathbb{R}^M$  и декодирует  $\mathbf{z}$  с помощью декодера. Декодированный вектор  $\tilde{\mathbf{x}}$  является ортогональной проекцией исходных данных  $\mathbf{x}$  на  $M$ -мерное главное подпространство

Другая интерпретация РСА связана с теорией информации. Мы можем рассматривать код как уменьшенную или сжатую версию исходной точки данных. Когда мы восстанавливаем наши исходные данные с помощью кода, мы получаем не точную точку данных, а ее слегка искаженную или зашумленную версию. Это означает, что наше сжатие происходит с потерями. Интуитивно мы хотим максимизировать корреляцию между исходными данными и низкоразмерным кодом<sup>1</sup>. Более формально это относится к взаимной информации. Затем мы получили бы то же решение для РСА, которое мы обсуждали в разделе 10.3, путем максимизации взаимной информации, ключевой концепции в теории информации (MacKay, 2003).

В нашем обсуждении РРСА мы предположили, что параметры модели, то есть  $\mathbf{B}$ ,  $\mu$  и параметр правдоподобия  $\sigma^2$ , известны. Типпинг и Бишоп (1999) описы-

<sup>1</sup> Код представляет собой сжатую версию исходных данных.

вают, как получить оценки максимального правдоподобия для этих параметров в настройке РРСА ( обратите внимание, что в этой главе мы используем другие обозначения). Параметры максимального правдоподобия при проецировании  $D$ -мерных данных на  $M$ -мерное подпространство равны

$$\boldsymbol{\mu}_{ML} = \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n; \quad (10.77)$$

$$\mathbf{B}_{ML} = \mathbf{T} (\Lambda - \sigma^2 \mathbf{I})^{\frac{1}{2}} \mathbf{R}; \quad (10.78)$$

$$\sigma_{ML}^2 = \frac{1}{D-M} \sum_{j=M+1}^D \lambda_j, \quad (10.79)$$

где  $\mathbf{T} \in \mathbb{R}^{D \times M}$  содержит  $M$  собственных векторов ковариационной матрицы данных,  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_M) \in \mathbb{R}^{M \times M}$  — диагональная матрица, собственные значения которой связаны с главными осями на ее диагонали, а  $\mathbf{R} \in \mathbb{R}^{M \times M}$  является произвольной ортогональной матрицей<sup>1</sup>. Решение максимального правдоподобия  $B_{ML}$  является уникальным с точностью до произвольного ортогонального преобразования, например мы можем умножить  $B_{ML}$  справа на любую матрицу вращения  $\mathbf{R}$ , так что (10.78) по существу является разложением по сингулярным значениям (раздел 4.5). Схема доказательства дана Типпингом и Бишопом (1999). Оценка максимального правдоподобия для  $\mu$ , приведенная в (10.77), является выборочным средним для данных.

Оценка максимального правдоподобия для дисперсии шума наблюдения  $\sigma^2$ , приведенная в (10.79), представляет собой среднюю дисперсию в ортогональном дополнении главного подпространства, то есть средняя оставшаяся дисперсия, которую мы не можем зафиксировать с помощью первых  $M$  главных компонент, рассматривается как шум наблюдения.

В пределе отсутствия шума, когда  $\sigma \rightarrow 0$ , РРСА и РСА предоставляют идентичные решения: поскольку матрица ковариации данных  $\mathbf{S}$  симметрична, ее можно диагонализовать (раздел 4.4), то есть существует матрица  $\mathbf{T}$  собственных векторов  $\mathbf{S}$ , поэтому

$$\mathbf{S} = \mathbf{T} \Lambda \mathbf{T}^{-1}. \quad (10.80)$$

В модели РРСА ковариационная матрица данных является ковариационной матрицей гауссова правдоподобия  $p(\mathbf{x} | \mathbf{B}, \mu, \sigma^2)$ , которая равна  $\mathbf{B}\mathbf{B}^T + \sigma^2 \mathbf{I}$ , см. (10.70b). При  $\sigma \rightarrow 0$  мы получаем  $\mathbf{B}\mathbf{B}^T$ , так что эта ковариация данных долж-

---

<sup>1</sup> Матрица  $\Lambda - \sigma^2 \mathbf{I}$  в (10.78) гарантированно будет положительно полуопределенной, поскольку наименьшее собственное значение матрицы ковариации данных ограничено снизу дисперсией шума  $\sigma^2$ .

на равняться ковариации данных РСА (и ее факторизации, заданной в (10.80)), так что

$$\text{Cov}[\mathcal{X}] = \mathbf{T}\Lambda\mathbf{T}^{-1} = \mathbf{B}\mathbf{B}^T \Leftrightarrow \mathbf{B} = \mathbf{T}\Lambda^{\frac{1}{2}}\mathbf{R}, \quad (10.81)$$

то есть мы получаем оценку максимального правдоподобия в (10.78) для  $\sigma = 0$ . Из (10.78) и (10.80) становится ясно, что (Р)РСА выполняет разложение ковариационной матрицы данных.

В настройках потоковой передачи, когда данные поступают последовательно, рекомендуется использовать алгоритм максимизации итеративного ожидания (EM) для оценки максимального правдоподобия (Roweis, 1998).

Чтобы определить размерность скрытых переменных (длина кода, размерность подпространства меньшей размерности, на которое мы проецируем данные), Гавиш и Донохо (Gavish and Donoho, 2014) предлагают такую эвристiku: если мы можем оценить дисперсию шума  $\sigma^2$  данных, мы должны отбросить все сингулярные значения, меньшие  $\frac{4\sigma\sqrt{D}}{\sqrt{3}}$ . В качестве альтернативы мы можем использовать (вложенную) перекрестную проверку (раздел 8.6.1) или критерии выбора байесовской модели (обсуждаемые в разделе 8.6.2), чтобы определить хорошую оценку внутренней размерности данных (Minka, 2001b).

Подобно нашему обсуждению линейной регрессии в главе 9, мы можем поместить априорное распределение для параметров модели и интегрировать их. Поступая таким образом, мы (a) избегаем точечных оценок параметров и проблем, связанных с этими точечными оценками (раздел 8.6), и (b) допускаем автоматический выбор подходящей размерности  $M$  скрытого пространства. В этом *байесовском РСА*, который был предложен Бишопом (1999), априорное значение  $p(\mu, \mathbf{B}, \sigma^2)$  помещается в параметры модели. Генеративный процесс позволяет нам интегрировать параметры модели, вместо того чтобы определять их, что решает проблемы переобучения. Поскольку эта интеграция аналитически неразрешима, Бишоп (1999) предлагает использовать приближенные методы вывода, такие как МCMC или вариационный вывод. Мысылаемся на работы Gilks et al. (1996) и Blei et al. (2017) для получения более подробной информации об этих приблизительных методах вывода.

В PPCA мы рассматривали линейную модель  $p(\mathbf{x}_n | \mathbf{z}_n) = \mathcal{N}(\mathbf{x}_n | \mathbf{B}\mathbf{z}_n + \mu, \sigma^2\mathbf{I})$ , с предшествующим  $p(\mathbf{z}_n) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ , где на все измерения наблюдения влияет одинаковое количество шума. Если мы позволим каждому измерению наблюдения  $d$  иметь различную дисперсию  $\sigma_d^2$ , мы получим *факторный анализ* (factor analysis – FA) (Spearman, 1904; Bartholomew et al., 2011). Это означает, что FA

дает вероятность большей гибкости<sup>1</sup>, чем PPCA, но по-прежнему заставляет данные объясняться параметрами модели  $\mathbf{B}$ ,  $\mu$ . Однако FA больше не допускает решения максимального правдоподобия в замкнутой форме, поэтому нам нужно использовать итерационную схему, такую как алгоритм максимизации математического ожидания, для оценки параметров модели. В то время как в PPCA все стационарные точки являются глобальными оптимумами, это больше не выполняется для FA. По сравнению с PPCA, FA не изменяется, если мы масштабируем данные, но он возвращает разные решения, если мы вращаем данные.

Алгоритм, который также тесно связан с PCA, представляет собой *анализ независимых компонент* (Independent component analysis, ICA (Hyvärinen et al., 2001)). Начиная снова с точки зрения скрытых переменных  $p(\mathbf{x}_n | \mathbf{z}_n) = \mathcal{N}(\mathbf{x}_n | \mathbf{B}\mathbf{z}_n + \mu, \sigma^2\mathbf{I})$ , мы теперь меняем априорность  $\mathbf{z}_n$  на негауссовые распределения. ICA можно использовать для *слепого разделения источников*. Представьте, что вы находитесь на оживленном вокзале, и много людей разговаривают. Ваши уши играют роль микрофонов, и они линейно смешивают различные речевые сигналы на вокзале. Цель разделения источников состоит в том, чтобы идентифицировать составные части смешанных сигналов. Как обсуждалось ранее в контексте оценки максимального правдоподобия для PPCA, исходное решение PCA инвариантно к любому вращению. Следовательно, PCA может идентифицировать лучшее подпространство более низкой размерности, в котором живут сигналы, но не сами сигналы (Murphy, 2012). ICA решает эту проблему, изменяя априорное распределение  $p(\mathbf{z})$  для скрытых источников, чтобы требовать негауссовых априорных значений  $p(\mathbf{z})$ . Мы ссылаемся на книги Hyvärinen et al. (2001) и Murphy (2012) для получения более подробной информации об ICA.

PCA, факторный анализ и ICA — три примера уменьшения размерности с помощью линейных моделей. Cunningham и Ghahramani (2015) предоставляют более широкий обзор снижения линейной размерности.

Модель (P)PCA, которую мы здесь обсуждали, допускает несколько важных расширений. В разделе 10.5 мы объяснили, как выполнить PCA, когда входная размерность  $D$  значительно превышает количество  $N$  точек данных. Используя понимание того, что PCA может быть реализовано путем вычисления (многих) внутренних продуктов, эта идея может быть доведена до крайности, рассматривая бесконечномерные функции. *Уловка с ядром* является основой *ядра PCA* и позволяет нам неявно вычислять внутренние продукты между бесконечномерными функциями (Schölkopf et al., 1998; Schölkopf and Smola, 2002).

---

<sup>1</sup> Чрезмерно гибкая вероятность могла бы объяснить больше, чем просто шум.

Существуют методы нелинейного уменьшения размерности, заимствованные из PCA (Burges, 2010, дает хороший обзор). Перспектива автокодировщика PCA, которую мы обсуждали ранее в этом разделе, может использоваться для рендеринга PCA как особого случая глубокого автокодировщика. В *глубоком автокодере* и кодер, и декодер представлены многослойными нейронными сетями с прямой связью, которые сами по себе являются нелинейными отображениями. Если мы установим функции активации в этих нейронных сетях как идентичные, модель станет эквивалентной PCA. Другой подход к уменьшению нелинейной размерности — это модель скрытых переменных гауссова процесса (*Gaussian process latent-variable model* — GP-LVM), предложенная Lawrence (2005). GP-LVM начинается с точки зрения скрытых переменных, которую мы использовали для получения PPCA, и заменяет линейную связь между скрытыми переменными  $\mathbf{z}$  и наблюдениями  $\mathbf{x}$  на гауссов процесс (*Gaussian process* — GP). Вместо оценки параметров отображения (как мы делаем в PPCA) GP-LVM маргинализирует параметры модели и делает точечные оценки скрытых переменных  $\mathbf{z}$ . Подобно байесовскому PCA, *байесовский GP-LVM*, предложенный Титсиасом и Лоуренсом (Titsias and Lawrence, 2010), поддерживает распределение скрытых переменных  $\mathbf{z}$  и также использует приближенный вывод для их интеграции.

# 11

## Оценка плотности с помощью моделей гауссовой смеси

В предыдущих главах мы уже рассмотрели две фундаментальные проблемы машинного обучения: регрессию (глава 9) и уменьшение размерности (глава 10). В этой главе мы рассмотрим третий столп машинного обучения: оценку плотности. По пути мы познакомим вас с важными концепциями, такими как алгоритм максимизации ожидания (expectation maximization – EM) и перспектива скрытых переменных для оценки плотности с помощью моделей смеси.

Когда мы применяем машинное обучение к данным, мы часто стремимся каким-то образом представить данные. Самый простой способ – взять сами точки данных как представление данных; см. рис. 11.1 для примера. Однако этот подход может оказаться бесполезным, если набор данных огромен или если мы заинтересованы в представлении характеристик данных. При оценке плотности мы представляем данные компактно, используя плотность из параметрического семейства, например гауссово или бета-распределение. Например, мы можем искать среднее значение и дисперсию набора данных, чтобы компактно представить данные с использованием распределения Гаусса. Среднее значение и дисперсию можно найти с помощью инструментов, которые мы обсуждали в разделе 8.3: максимальная вероятность или максимальная апостериорная оценка. Затем мы можем использовать среднее значение и дисперсию этого гауссиана для представления распределения, лежащего в основе данных, то есть мы думаем, что набор данных является типичной реализацией этого распределения, если бы мы брали из него выборку.

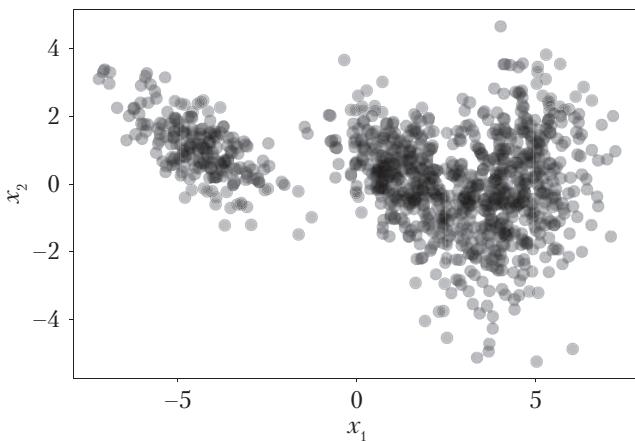
На практике гауссово распределение (или аналогично все другие распределения, с которыми мы встречались до сих пор) имеет ограниченные возможности моделирования. Например, гауссово приближение плотности, сгенерированное

данные на рис. 11.1, было бы плохим приближением. Далее мы рассмотрим более выразительное семейство распределений, которое мы можем использовать для оценки плотности: модели смеси. *Смесительные модели* могут использоваться для описания распределения  $p(\mathbf{x})$  выпуклой комбинацией  $K$  простых (базовых) распределений.

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k p_k(\mathbf{x}); \quad (11.1)$$

$$0 \leq \pi_k \leq 1; \quad \sum_{k=1}^K \pi_k = 1, \quad (11.2)$$

где компоненты  $p_k$  являются членами семейства основных распределений, например гауссианов, бернуlliевых или гамма-распределений, а  $\pi_k$  — *веса смеси*. Смесительные модели более выразительны, чем соответствующие базовые распределения, поскольку они допускают мультимодальные представления данных, то есть они могут описывать наборы данных с несколькими «кластерами», как в примере на рис. 11.1.



**Рис. 11.1.** Двумерный набор данных, который не может быть осмысленно представлен с помощью гауссиана

Мы сосредоточимся на моделях гауссовой смеси (Gaussian mixture models — GMM), где основные распределения являются гауссианами. Для данного набора данных мы стремимся максимизировать вероятность параметров модели для обучения GMM. Для этого воспользуемся результатами из глав 5, 6 и 7.2. Однако в отличие от других приложений, которые мы обсуждали ранее (линейная регрессия или PCA), мы не найдем решения максимального правдоподобия в замкнутой форме. Вместо этого мы придем к системе зависимых одновременных уравнений, которую мы можем решить только итеративно.

## 11.1. МОДЕЛЬ ГАУССОВОЙ СМЕСИ

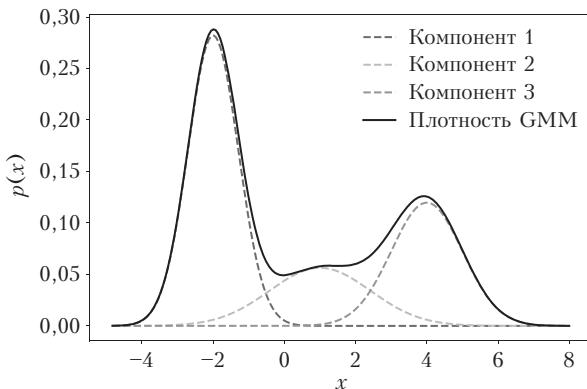
Модель гауссовой смеси (Gaussian mixture model, GMM) — это модель плотности, в которой мы комбинируем конечное число  $K$  гауссовых распределений  $\mathcal{N}(x | \mu_k, \Sigma_k)$ , так чтобы

$$p(x | \theta) = \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \Sigma_k); \quad (11.3)$$

$$0 \leq \pi_k \leq 1; \quad \sum_{k=1}^K \pi_k = 1, \quad (11.4)$$

где мы определили  $\theta := \{\mu_k, \Sigma_k, \pi_k : k = 1, \dots, K\}$  как совокупность всех параметров модели. Эта выпуклая комбинация гауссова распределения дает нам значительно большую гибкость для моделирования сложных плотностей, чем простое гауссово распределение (которое мы восстанавливаем из (11.3) для  $K = 1$ ). На рис. 11.2 представлена иллюстрация, на которой показаны взвешенные компоненты и плотность смеси, которая представлена как

$$p(x | \theta) = 0,5\mathcal{N}\left(x | -2, \frac{1}{2}\right) + 0,2\mathcal{N}\left(x | 1, 2\right) + 0,3\mathcal{N}\left(x | 4, 1\right). \quad (11.5)$$



**Рис.11.2.** Модель гауссовой смеси. Распределение GMM (чёрная линия) состоит из выпуклой комбинации распределений Гаусса и является более выразительным, чем любой отдельный компонент. Пунктирные линии представляют взвешенные гауссовые компоненты

## 11.2. ИЗУЧЕНИЕ ПАРАМЕТРОВ С ПОМОЩЬЮ МАКСИМАЛЬНОГО ПРАВДОПОДОБИЯ

Предположим, нам дан набор данных  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ , где  $\mathbf{x}_n, n = 1, \dots, N$ , рисуются независимыми и одинаково распределенными из неизвестного распределения  $p(\mathbf{x})$ . Наша цель — найти хорошее приближение/представление этого неизвестного распределения  $p(\mathbf{x})$  с помощью GMM с  $K$  компонентов смеси. Параметрами GMM являются  $K$ -средние  $\mu_k$ , ковариации  $\Sigma_k$  и веса смеси  $\pi_k$ . Суммируем все эти свободные параметры в  $\theta := \{\pi_k, \mu_k, \Sigma_k : k = 1, \dots, K\}$ .

### Пример 11.1 (первоначальная настройка)

В этой главе у нас будет простой рабочий пример, который поможет нам проиллюстрировать и визуализировать важные концепции.

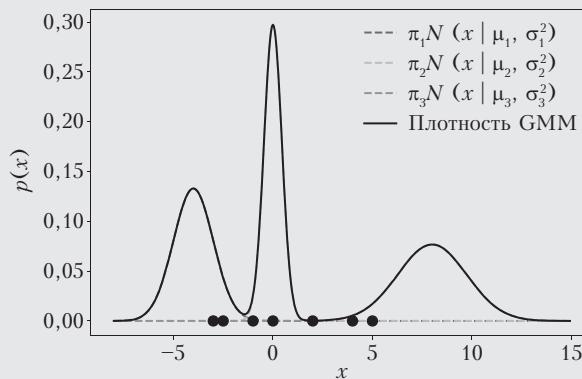
Мы рассматриваем одномерный набор данных  $\mathcal{X} = \{-3, -2,5, -1, 0, 2, 4, 5\}$ , состоящий из семи точек данных, и хотим найти GMM с  $K = 3$  компонентами, который моделирует плотность данных. Мы инициализируем компоненты смеси как

$$p_1(x) = \mathcal{N}(x | -4, 1); \quad (11.6)$$

$$p_2(x) = \mathcal{N}(x | 0, 0,2); \quad (11.7)$$

$$p_3(x) = \mathcal{N}(x | 8, 3) \quad (11.8)$$

и присваиваем им равные веса  $\pi_1 = \pi_2 = \pi_3 = \frac{1}{3}$ . Соответствующая модель (и точки данных) показаны на рис. 11.3.



**Рис. 11.3.** Начальная настройка: GMM (черная линия) со смесью трех компонентов смеси (пунктир) и семи точек данных (точки)

Далее мы подробно описываем, как получить оценку максимального правдоподобия  $\theta_{ML}$  параметров модели  $\theta$ . Мы начинаем с записи вероятности, то есть прогнозируемого распределения обучающих данных с учетом параметров. Мы используем наше независимое и одинаково распределенное предположение, которое приводит к факторизованной вероятности

$$p(\mathcal{X} | \theta) = \prod_{n=1}^N p(\mathbf{x}_n | \theta), p(\mathbf{x}_n | \theta) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k), \quad (11.9)$$

где каждый отдельный член вероятности  $p(\mathbf{x}_n | \theta)$  является плотностью гауссовой смеси. Тогда мы получаем логарифмическое правдоподобие как

$$\log p(\mathcal{X} | \boldsymbol{\theta}) = \sum_{n=1}^N \log p(\mathbf{x}_n | \boldsymbol{\theta}) = \sum_{n=1}^N \underbrace{\log \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}_{=: \mathcal{L}}. \quad (11.10)$$

Наша цель — найти параметры  $\boldsymbol{\theta}_{ML}^*$ , которые максимизируют логарифмическое правдоподобие  $\mathcal{L}$ , определенное в (11.10). Наша «нормальная» процедура заключалась бы в вычислении градиента  $d\mathcal{L}/d\boldsymbol{\theta}$  логарифма правдоподобия относительно параметров модели  $\boldsymbol{\theta}$ , установке его в 0 и решении относительно  $\boldsymbol{\theta}$ . Однако в отличие от наших предыдущих примеров оценки максимального правдоподобия (например, когда мы обсуждали линейную регрессию в разделе 9.2), мы не можем получить решение в замкнутой форме. Тем не менее, мы можем использовать итеративную схему, чтобы найти хорошие параметры модели  $\boldsymbol{\theta}_{ML}$ , которые, в конечном итоге, будут алгоритмом EM для GMM. Ключевая идея состоит в том, чтобы обновлять один параметр модели за один шаг итерации, оставляя другие неизменными.

**ПРИМЕЧАНИЕ** Если бы мы рассмотрели один гауссиан как желаемую плотность, сумма по  $k$  в (11.10) обращается в нуль, и  $\log$  можно применить непосредственно к гауссовой составляющей, так что мы получим

$$\log \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{D}{2} \log(2\pi) - \frac{1}{2} \log \det(\boldsymbol{\Sigma}) - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}). \quad (11.11)$$

Эта простая форма позволяет нам найти оценки максимального правдоподобия  $\boldsymbol{\mu}$  и  $\boldsymbol{\Sigma}$  в замкнутой форме, как обсуждалось в главе 8. В (11.10) мы не можем переместить  $\log$  в сумму по  $k$ , так что мы не можем получить простое решение с максимальным правдоподобием в замкнутой форме. ◆

Любой локальный оптимум функции обладает тем свойством, что его градиент по параметрам должен обращаться в нуль (необходимое условие); см. главу 7. В нашем случае мы получаем следующие необходимые условия, когда оптимизируем логарифмическое правдоподобие в (11.10) по параметрам GMM  $\boldsymbol{\mu}_k$ ,  $\boldsymbol{\Sigma}_k$ ,  $\pi_k$ :

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_k} = \mathbf{0}^T \Leftrightarrow \sum_{n=1}^N \frac{\partial \log p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \boldsymbol{\mu}_k} = \mathbf{0}^T; \quad (11.12)$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}_k} = \mathbf{0} \Leftrightarrow \sum_{n=1}^N \frac{\partial \log p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k} = \mathbf{0}; \quad (11.13)$$

$$\frac{\partial \mathcal{L}}{\partial \pi_k} = 0 \Leftrightarrow \sum_{n=1}^N \frac{\partial \log p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \pi_k} = 0. \quad (11.14)$$

Для всех трех необходимых условий, применяя цепное правило (раздел 5.2.2), нам потребуются частные производные вида

$$\frac{\partial \log p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{1}{p(\mathbf{x}_n | \boldsymbol{\theta})} \frac{\partial p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \quad (11.15)$$

где  $\boldsymbol{\theta} = \{\boldsymbol{\mu}_k, \Sigma_k, \pi_k, k = 1, \dots, K\}$  — параметры модели и

$$\frac{1}{p(\mathbf{x}_n | \boldsymbol{\theta})} = \frac{1}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)}. \quad (11.16)$$

Далее мы вычислим частные производные (11.12)–(11.14). Но прежде, чем мы это сделаем, мы представим величину, которая будет играть центральную роль в оставшейся части этой главы: ответственность.

### 11.2.1. Ответственность

Определяем количество

$$r_{nk} := \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)} \quad (11.17)$$

как *ответственность*  $k$ -го компонента смеси за  $n$ -ю точку данных. Ответственность  $r_{nk}$   $k$ -го компонента смеси для точки данных  $\mathbf{x}_n$  пропорциональна правдоподобию

$$p(\mathbf{x}_n | \pi_k, \boldsymbol{\mu}_k, \Sigma_k) = \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k) \quad (11.18)$$

компоненты смеси с учетом точки данных<sup>1</sup>. Следовательно, компоненты смеси несут большую ответственность за точку данных, когда точка данных может быть допустимой пробой из этого компонента смеси. Обратите внимание, что  $\mathbf{r}_n := [r_{n1}, \dots, r_{nK}]^\top \in \mathbb{R}^k$  является (нормированным) вектором вероятности, то есть  $\sum_k r_{nk} = 1 \geq 0$ . Этот вектор вероятности распределяет вероятностную массу между  $K$  компонентами смеси, и мы можем рассматривать  $\mathbf{r}_n$  как «мягкое присвоение»  $\mathbf{x}_n$  компонентам  $K$  смеси. Следовательно, ответственность  $r_{nk}$  из (11.17) представляет собой вероятность того, что  $\mathbf{x}_n$  был сгенерирован  $k$ -м компонентом смеси<sup>2</sup>.

<sup>1</sup>  $\mathbf{r}_n$  следует распределению Больцмана — Гиббса.

<sup>2</sup> Ответственность  $r_{nk}$  — это вероятность того, что  $k$ -й компонент смеси сгенерировал  $n$ -ю точку данных.

**Пример 11.2 (ответственности)**

В нашем примере из рис. 11.3 мы вычисляем ответственности  $r_{nk}$

$$\begin{bmatrix} 1,0 & 0,0 & 0,0 \\ 1,0 & 0,0 & 0,0 \\ 0,057 & 0,943 & 0,0 \\ 0,001 & 0,999 & 0,0 \\ 0,0 & 0,066 & 0,934 \\ 0,0 & 0,0 & 1,0 \\ 0,0 & 0,0 & 1,0 \end{bmatrix} \in \mathbb{R}^{N \times K}. \quad (11.19)$$

Здесь  $n$ -я строка сообщает нам ответственности всех компонентов смеси для  $x_n$ . Сумма всех  $K$  ответственностей для точки данных (сумма каждой строки) равна 1.  $k$ -й столбец дает нам обзор ответственности  $k$ -го компонента смеси. Мы видим, что третий компонент смеси (третий столбец) не отвечает ни за одну из первых четырех точек данных, но берет на себя большую ответственность за остальные точки данных. Сумма всех записей столбца дает нам значения  $N_k$ , то есть полную ответственность  $k$ -го компонента смеси. В нашем примере мы получаем  $N_1 = 2,058$ ,  $N_2 = 2,008$ ,  $N_3 = 2,934$ .

Далее мы определяем обновления параметров модели  $\mu_k$ ,  $\Sigma_k$ ,  $\pi_k$  для заданных ответственностей. Мы увидим, что все уравнения обновления зависят от ответственностей, что делает невозможным решение задачи оценки максимального правдоподобия в замкнутой форме. Однако для данных ответственностей мы будем обновлять один параметр модели за один шаг, оставляя другие неизменными. После этого мы пересчитаем ответственности. Итерация этих двух шагов в конечном итоге приведет к локальному оптимуму и является конкретным воплощением алгоритма EM. Мы обсудим это более подробно в разделе 11.3.

### 11.2.2. Обновление средних

**Теорема 11.1 (обновление средних GMM).** *Обновление средних параметров GMM  $\mu_k$ ,  $k = 1, \dots, K$  определяется выражением*

$$\mu_k^{new} = \frac{\sum_{n=1}^N r_{nk} \mathbf{x}_n}{\sum_{n=1}^N r_{nk}}, \quad (11.20)$$

где ответственности  $r_{nk}$  определены в (11.17).

**ПРИМЕЧАНИЕ** Обновление средних  $\mu_k$  отдельных компонентов смеси в (11.20) зависит от всех средних, ковариационных матриц  $\Sigma_k$  и весов смеси  $\pi_k$  через  $r_{nk}$ , указанные в (11.17). Следовательно, мы не можем получить решение в замкнутой форме для всех  $\mu_k$  сразу. ♦

*Доказательство.* Из (11.15) мы видим, что градиент логарифма правдоподобия относительно средних параметров  $\mu_k$ ,  $k = 1, \dots, K$ , требует, чтобы мы вычислили частную производную

$$\frac{\partial p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \mu_k} = \sum_{j=1}^K \pi_j \frac{\partial \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}{\partial \mu_k} = \pi_k \frac{\partial \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\partial \mu_k} = \quad (11.21a)$$

$$= \pi_k (\mathbf{x}_n - \mu_k)^T \sum_k^{-1} \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k), \quad (11.21b)$$

где мы использовали то, что только  $k$ -й компонент смеси зависит от  $\mu_k$ .

Мы используем наш результат из (11.21b) в (11.15) и собираем все вместе, так чтобы искомая частная производная  $\mathcal{L}$  по  $\mu_k$  была задана как

$$\frac{\partial \mathcal{L}}{\partial \mu_k} = \sum_{n=1}^N \frac{\partial \log p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \mu_k} = \sum_{n=1}^N \frac{1}{p(\mathbf{x}_n | \boldsymbol{\theta})} \frac{\partial p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \mu_k} = \quad (11.22a)$$

$$= \sum_{n=1}^N (\mathbf{x}_n - \mu_k)^T \sum_k^{-1} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}}_{=r_{nk}} = \quad (11.22b)$$

$$= \sum_{n=1}^N r_{nk} (\mathbf{x}_n - \mu_k)^T \Sigma_k^{-1}. \quad (11.22c)$$

Здесь мы использовали тождество из (11.16) и результат частной производной в (11.21b), чтобы получить (11.22b). Значения  $r_{nk}$  — это ответственности, которые мы определили в (11.17).

Решим теперь (11.22c) относительно  $\mu_k^{new}$ , так что  $\frac{\partial \mathcal{L}(\mu_k^{new})}{\partial \mu_k} = \mathbf{0}^T$ , и получим

$$\sum_{n=1}^N r_{nk} \mathbf{x}_n = \sum_{n=1}^N r_{nk} \mu_k^{new} \Leftrightarrow \mu_k^{new} = \frac{\sum_{n=1}^N r_{nk} \mathbf{x}_n}{\boxed{\sum_{n=1}^N r_{nk}}} = \frac{1}{\boxed{N_k}} \sum_{n=1}^N r_{nk} \mathbf{x}_n, \quad (11.23)$$

где мы определили

$$N_k := \sum_{n=1}^N r_{nk} \quad (11.24)$$

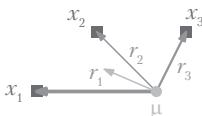
как полную ответственность  $k$ -го компонента смеси за весь набор данных. Это завершает доказательство теоремы 11.1.  $\square$

Интуитивно (11.20) можно интерпретировать как взвешенную по важности оценку Монте-Карло среднего, где веса важности точки данных  $\mathbf{x}_n$  — это ответственности  $r_{nk}$   $k$ -го кластера для  $\mathbf{x}_n$ ,  $k = 1, \dots, K$ . Следовательно, среднее значение  $\mu_k$  притягивается к точке данных  $\mathbf{x}_n$  с силой, заданной параметром  $r_{nk}$ . Средние значения тянутся сильнее к точкам данных, для которых соответствующий компонент смеси имеет высокую ответственность, то есть высокую вероятность. Рисунок 11.4 иллюстрирует это. Мы также можем интерпретировать среднее обновление в (11.20) как ожидаемое значение всех точек данных в соответствии с распределением, заданным формулой

$$\mathbf{r}_k := [r_{1k}, \dots, r_{Nk}]^T / N_k, \quad (11.25)$$

которое является нормализованным вектором вероятности, то есть

$$\mu_k \leftarrow \mathbb{E}_{\mathbf{r}_k} [\mathcal{X}]. \quad (11.26)$$



**Рис. 11.4.** Обновление среднего параметра компонента смеси в GMM. Среднее значение  $\mu$  приближается к отдельным точкам данных с весами, заданными соответствующими ответственностями

### Пример 11.3 (обновления средних)

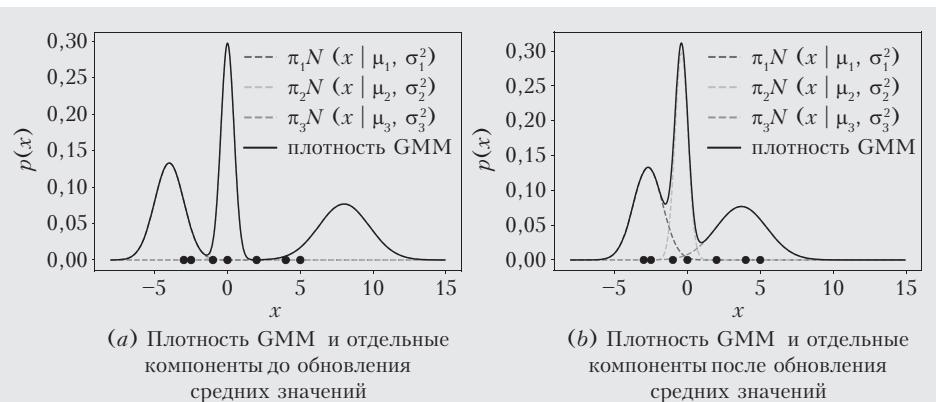
В нашем примере из рис. 11.3 средние значения обновляются следующим образом:

$$\mu_1 : -4 \rightarrow -2,7; \quad (11.27)$$

$$\mu_2 : 0 \rightarrow -0,4; \quad (11.28)$$

$$\mu_3 : 8 \rightarrow 3,7. \quad (11.29)$$

Здесь мы видим, что средние значения первого и третьего компонентов смеси смещаются в сторону режима данных, тогда как среднее значение второго компонента не меняется так резко. Рисунок 11.5 иллюстрирует это изменение, где рис. 11.5(a) показывает плотность GMM до обновления средних значений, а рис. 11.5(b) показывает плотность GMM после обновления средних значений  $\mu_k$ .



**Рис. 11.5.** Эффект обновления средних значений в GMM. (a) GMM перед обновлением средних значений; (b) GMM после обновления средних значений  $\mu_k$  с сохранением дисперсии и веса смеси

Обновление средних параметров в (11.20) выглядит довольно просто. Однако обратите внимание, что ответственности  $r_{nk}$  являются функцией  $\pi_j, \mu_j, \Sigma_j$  для всех  $j = 1, \dots, K$ , так что обновления в (11.20) зависят от всех параметров GMM, и решение в замкнутой форме, которое мы получили для линейной регрессии в разделе 9.2 или РСА в главе 10, получить невозможно.

### 11.2.3. Обновление ковариаций

**Теорема 11.2 (обновления ковариаций GMM).** *Обновление параметров ковариации  $\Sigma_k, k = 1, \dots, K$ , GMM определяется выражением*

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^T, \quad (11.30)$$

где  $r_{nk}$  и  $N_k$  определены в (11.17) и (11.24) соответственно.

**Доказательство.** Наш подход доказательства теоремы 11.2 состоит в том, чтобы вычислить частные производные логарифма правдоподобия  $\mathcal{L}$  по ковариациям  $\Sigma_k$ , установить их равными 0 и решить относительно  $\Sigma_k$ . Начнем с нашего общего подхода

$$\frac{\partial \mathcal{L}}{\partial \Sigma_k} = \sum_{n=1}^N \frac{\partial \log p(\mathbf{x}_n | \theta)}{\partial \Sigma_k} = \sum_{n=1}^N \frac{1}{p(\mathbf{x}_n | \theta)} \frac{\partial p(\mathbf{x}_n | \theta)}{\partial \Sigma_k}. \quad (11.31)$$

Мы уже знаем  $1/p(\mathbf{x}_n | \theta)$  из (11.16). Чтобы получить оставшуюся частную производную  $\partial p(\mathbf{x}_n | \theta) / \partial \Sigma_k$ , запишем определение гауссова распределения  $p(\mathbf{x}_n | \theta)$  (11.9) и отбросим все члены, кроме  $k$ -го. Тогда получаем

$$\frac{\partial p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \Sigma_k} = \quad (11.32a)$$

$$= \frac{\partial}{\partial \Sigma_k} \left( \pi_k (2\pi)^{-\frac{D}{2}} \det(\Sigma_k)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k)\right) \right) = \quad (11.32b)$$

$$= \pi_k (2\pi)^{-\frac{D}{2}} \left[ \frac{\partial}{\partial \Sigma_k} \det(\Sigma_k)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k)\right) + \right. \\ \left. + \det(\Sigma_k)^{-\frac{1}{2}} \frac{\partial}{\partial \Sigma_k} \exp\left(-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k)\right) \right]. \quad (11.32c)$$

Теперь мы используем тождества

$$\frac{\partial}{\partial \Sigma_k} \det(\Sigma_k)^{-\frac{1}{2}} \stackrel{(5.101)}{=} -\frac{1}{2} \det(\Sigma_k)^{-\frac{1}{2}} \Sigma_k^{-1}; \quad (11.33)$$

$$\frac{\partial}{\partial \Sigma_k} (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) \stackrel{(5.106)}{=} -\Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} \quad (11.34)$$

и получим (после некоторой перестановки) искомую частную производную, требуемую в (11.31), как

$$\frac{\partial p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \Sigma_k} = \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k) \times \\ \times \left[ -\frac{1}{2} (\Sigma_k^{-1} - \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1}) \right]. \quad (11.35)$$

Собирая все вместе, частная производная логарифма правдоподобия по  $\Sigma_k$  дается выражением

$$\frac{\partial \mathcal{L}}{\partial \Sigma_k} = \sum_{n=1}^N \frac{\partial \log p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \Sigma_k} = \sum_{n=1}^N \frac{1}{p(\mathbf{x}_n | \boldsymbol{\theta})} \frac{\partial p(\mathbf{x}_n | \boldsymbol{\theta})}{\partial \Sigma_k} = \quad (11.36a)$$

$$= \sum_{n=1}^N \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)}}_{= r_{nk}} \times \\ \times \left[ -\frac{1}{2} (\Sigma_k^{-1} - \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1}) \right] = \quad (11.36b)$$

$$= -\frac{1}{2} \sum_{n=1}^N r_{nk} (\Sigma_k^{-1} - \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1}) = \quad (11.36c)$$

$$= -\frac{1}{2} \Sigma_k^{-1} \underbrace{\sum_{n=1}^N r_{nk}}_{=N_k} + \frac{1}{2} \Sigma_k^{-1} \left( \sum_{n=1}^N r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \right) \Sigma_k^{-1}. \quad (11.36d)$$

Мы видим, что ответственности  $r_{nk}$  также присутствуют в этой частной производной. Приравнивая эту частную производную к  $\mathbf{0}$ , получаем необходимое условие оптимальности

$$N_k \Sigma_k^{-1} = \Sigma_k^{-1} \left( \sum_{n=1}^N r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \right) \Sigma_k^{-1} \Leftrightarrow \quad (11.37a)$$

$$\Leftrightarrow N_k \mathbf{I} = \left( \sum_{n=1}^N r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \right) \Sigma_k^{-1}. \quad (11.37b)$$

Решая относительно  $\Sigma_k$ , получаем

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T, \quad (11.38)$$

где  $\mathbf{r}_k$  — вектор вероятности, определенный в (11.25). Это дает нам простое правило обновления  $\Sigma_k$  для  $k = 1, \dots, K$  и доказывает теорему 11.2.  $\square$

Подобно обновлению  $\boldsymbol{\mu}_k$  в (11.20), мы можем интерпретировать обновление ковариации в (11.30) как взвешенное по важности ожидаемое значение квадрата центрированных данных  $\tilde{\mathcal{X}}_k := \{\boldsymbol{x}_1 - \boldsymbol{\mu}_k, \dots, \boldsymbol{x}_N - \boldsymbol{\mu}_k\}$ .

#### Пример 11.4 (обновления дисперсии)

В нашем примере из рис. 11.3 дисперсии обновляются следующим образом:

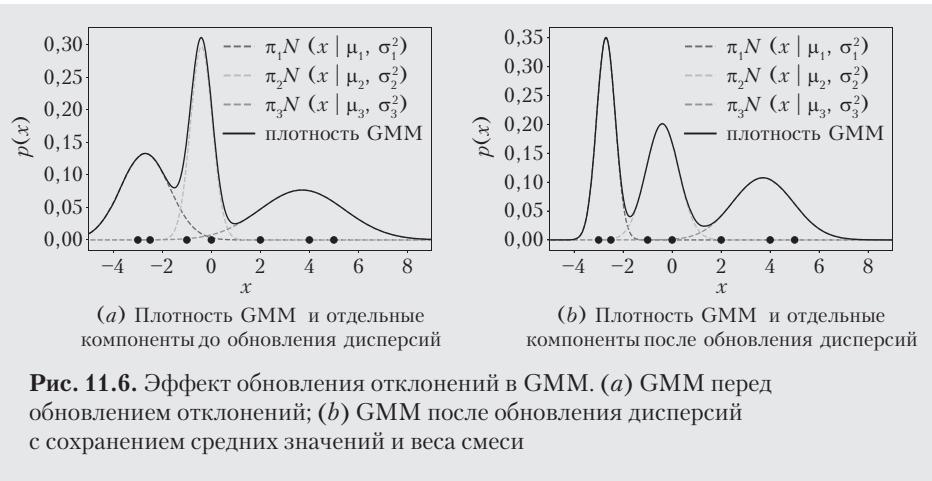
$$\sigma_1^2 : 1 \rightarrow 0,14; \quad (11.39)$$

$$\sigma_2^2 : 0,2 \rightarrow 0,44; \quad (11.40)$$

$$\sigma_3^2 : 3 \rightarrow 1,53. \quad (11.41)$$

Здесь мы видим, что дисперсия первого и третьего компонентов значительно сокращается, тогда как дисперсия второго компонента немножко увеличивается.

Рисунок 11.6 иллюстрирует эту настройку. Рисунок 11.6(a) идентичен рис. 11.5(b) (но увеличенному) и показывает плотность GMM и ее отдельные компоненты до обновления дисперсий. На рис. 11.6(b) показана плотность GMM после обновления дисперсий.



**Рис. 11.6.** Эффект обновления отклонений в GMM. (a) GMM перед обновлением отклонений; (b) GMM после обновления дисперсий с сохранением средних значений и веса смеси

Подобно обновлению средних параметров, мы можем интерпретировать (11.30) как оценку Монте-Карло взвешенной ковариации точек данных  $\mathbf{x}_n$ , связанных с  $k$ -м компонентом смеси, где веса — это ответственности  $r_{nk}$ . Как и обновления средних параметров, это обновление зависит от всех  $\pi_j, \mu_j, \Sigma_j, j = 1, \dots, K$ , через ответственности  $r_{nk}$ , запрещающие решение в замкнутой форме.

#### 11.2.4. Обновление весов смеси

**Теорема 11.3 (обновление весов смеси GMM).** Вес смеси GMM обновляется как

$$\pi_k^{new} = \frac{N_k}{N}, \quad k = 1, \dots, K, \quad (11.42)$$

где  $N$  — количество точек данных, а  $N_k$  определено в (11.24).

**Доказательство.** Чтобы найти частную производную логарифма правдоподобия по весовым параметрам  $\pi_k, k = 1, \dots, K$ , мы учитываем ограничение  $\sum_k \pi_k = 1$  с помощью множителей Лагранжа (раздел 7.2). Лагранжиан

$$\mathcal{L} = \mathcal{L} + \lambda \left( \sum_{k=1}^K \pi_k - 1 \right) = \quad (11.43a)$$

$$= \sum_{n=1}^N \log \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) + \lambda \left( \sum_{k=1}^K \pi_k - 1 \right), \quad (11.43b)$$

где  $\mathcal{L}$  — логарифм правдоподобия из (11.10), а второй член кодирует ограничение равенства, которое необходимо для суммирования всех весов смеси до 1. Мы получаем частную производную по  $\pi_k$  как

$$\frac{\partial \mathcal{L}}{\partial \pi_k} = \sum_{n=1}^N \frac{\mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)} + \lambda = \quad (11.44a)$$

$$= \frac{1}{\pi_k} \underbrace{\sum_{n=1}^N \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)}}_{= N_k} + \lambda = \frac{N_k}{\pi_k} + \lambda \quad (11.44b)$$

и частную производную по множителю Лагранжа  $\lambda$  как

$$\frac{\partial \mathcal{L}}{\partial \lambda} = \sum_{k=1}^K \pi_k - 1. \quad (11.45)$$

Приравнивая обе частные производные к  $0$  (необходимое условие оптимума), получаем систему уравнений

$$\pi_k = -\frac{N_k}{\lambda}; \quad (11.46)$$

$$1 = \sum_{k=1}^K \pi_k. \quad (11.47)$$

Используя (11.46) в (11.47) и решая относительно  $\pi_k$ , получаем

$$\sum_{k=1}^K \pi_k = 1 \Leftrightarrow -\sum_{k=1}^K \frac{N_k}{\lambda} = 1 \Leftrightarrow -\frac{N}{\lambda} = 1 \Leftrightarrow \lambda = -N. \quad (11.48)$$

$$\pi_k^{new} = \frac{N_k}{N}, \quad (11.49)$$

что дает нам обновление для весовых параметров  $\pi_k$  и доказывает теорему 11.3.

□

Мы можем определить вес смеси в (11.42) как отношение общей ответственности  $k$ -го кластера к количеству точек данных. Поскольку  $N = \sum_k N_k$ , количество точек данных также можно интерпретировать как общую ответственность всех компонентов смеси вместе, так что  $\pi_k$  — относительная важность  $k$ -го компонента смеси для набора данных.

**ПРИМЕЧАНИЕ** Поскольку  $N_k = \sum_{i=1}^N r_{ik}$ , уравнение обновления (11.42) выражает зависимости величин весов смеси  $\pi$  от всех величин  $\pi_j, \boldsymbol{\mu}_j, \Sigma_j, j = 1, \dots, K$  через коэффициенты ответственности  $r$ . ◆

### Пример 11.5 (обновления весовых параметров)

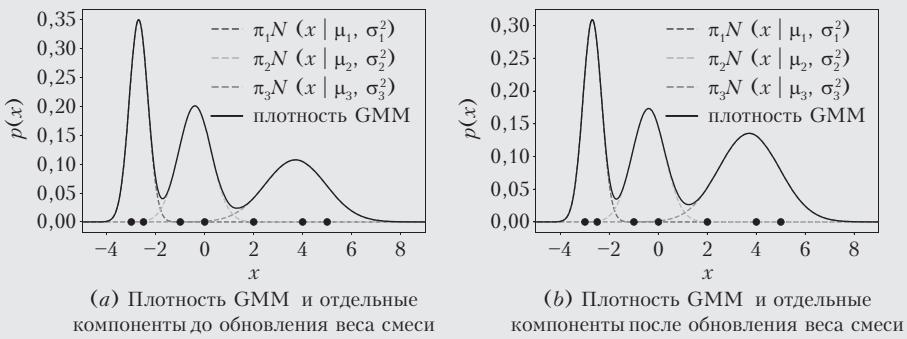
В нашем рабочем примере из рис. 11.3 веса смеси обновляются следующим образом:

$$\pi_1 : \frac{1}{3} \rightarrow 0,29; \quad (11.50)$$

$$\pi_2 : \frac{1}{3} \rightarrow 0,29; \quad (11.51)$$

$$\pi_3 : \frac{1}{3} \rightarrow 0,42. \quad (11.52)$$

Здесь мы видим, что третий компонент получает больший вес/важность, в то время как другие компоненты становятся немного менее важными. На рис. 11.7 показан эффект обновления весов смеси. Рисунок 11.7(a) идентичен рис. 11.6(b) и показывает плотность GMM и его отдельные компоненты до обновления весов смеси. На рис. 11.7(b) показана плотность GMM после обновления весов смеси.



**Рис. 11.7.** Эффект обновления весов смеси в GMM. (a) GMM перед обновлением весов смеси; (b) GMM после обновления весов смесей с сохранением средних значений и отклонений. Обратите внимание на разные масштабы вертикальных осей

В целом, обновив средние значения, дисперсию и веса один раз, мы получаем GMM, показанную на рис. 11.7(b). По сравнению с инициализацией, показанной на рис. 11.3, мы можем видеть, что обновления параметров заставили плотность GMM сместить часть своей массы в сторону точек данных.

После однократного обновления средних, дисперсий и весов соответствие GMM на рис. 11.7(b) уже заметно лучше, чем ее инициализация на рис. 11.3. Об этом также свидетельствуют значения логарифма правдоподобия, которые увеличились с 28,3 (инициализация) до 14,4 после одного полного цикла обновления.

### 11.3. EM-АЛГОРИТМ

К сожалению, обновления в (11.20), (11.30) и (11.42) не представляют собой решение в замкнутой форме для обновлений параметров  $\mu_k, \Sigma_k, \pi_k$  модели смеси, поскольку ответственности  $r_{nk}$  зависят от этих параметров сложным способом. Однако результаты предлагают простую *итерационную схему* для поиска решения задачи оценки параметров с помощью максимального правдоподобия. Алгоритм максимизации ожидания (алгоритм EM – expectation maximization) был предложен Dempster et al. (1977) и представляет собой общую итеративную схему для изучения параметров (максимального правдоподобия или MAP) в смешанных моделях и, в более общем смысле, в моделях со скрытыми переменными.

В нашем примере модели гауссовой смеси мы выбираем начальные значения для  $\mu_k, \Sigma_k, \pi_k$  и меняем их до сходимости между  $E$ -шагом и  $M$ -шагом:

- *E-шаг*: оцените ответственности  $r_{nk}$  (апостериорная вероятность того, что точка данных  $n$  принадлежит компоненту смеси  $k$ ).
- *M-шаг*: используйте обновленные ответственности для вычисления новой оценки параметров  $\mu_k, \Sigma_k, \pi_k$ .

Каждый шаг в алгоритме EM увеличивает логарифмическую функцию правдоподобия (Neal and Hinton, 1999). Для сходимости мы можем напрямую проверить логарифмическую вероятность или параметры. Конкретная реализация алгоритма EM для оценки параметров GMM выглядит следующим образом:

1. Инициализируйте  $\mu_k, \Sigma_k, \pi_k$ .

2. *E-шаг*: оцените ответственности  $r_{nk}$  для каждой точки данных  $x_n$ , используя текущие параметры  $\pi_k, \mu_k, \Sigma_k$ :

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_n | \mu_j, \Sigma_j)}. \quad (11.53)$$

1. *M-шаг*: заново оценить параметры  $\pi_k, \mu_k, \Sigma_k$ , используя текущие ответственности  $r_{nk}$  (из E-шага):

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n; \quad (11.54)$$

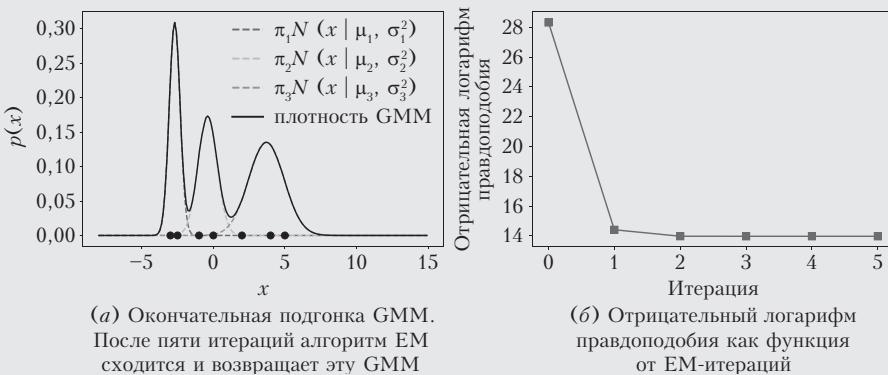
$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^T; \quad (11.55)$$

$$\pi_k = \frac{N_k}{N}. \quad (11.56)$$

### Пример 11.6 (GMM-соответствие)

Когда мы запускаем EM в нашем примере из рис. 11.3, мы получаем окончательный результат, показанный на рис. 11.8(a), после пяти итераций, а рис. 11.8(b) показывает, как отрицательное логарифмическое правдоподобие изменяется в зависимости от итераций EM. Окончательная GMM представлена как

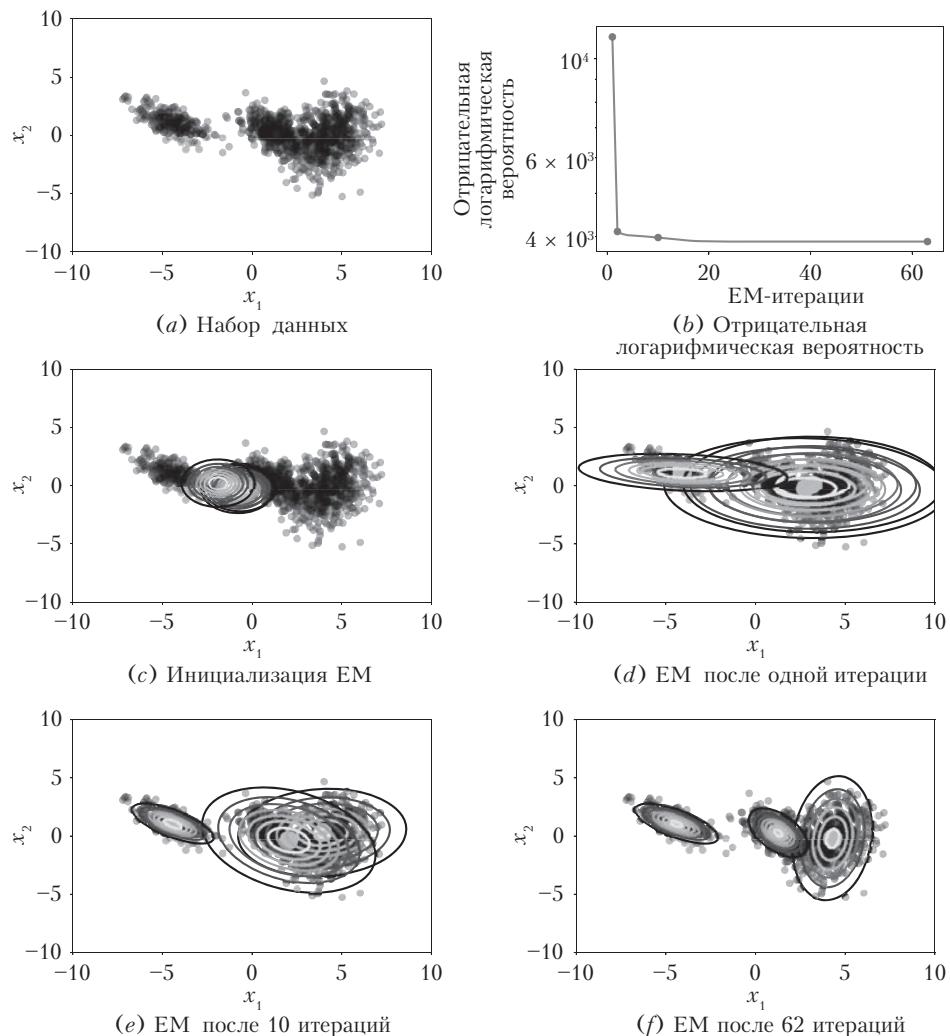
$$p(x) = 0,29\mathcal{N}(x | -2,75, 0,06) + 0,28\mathcal{N}(x | -0,50, 0,25) + \\ + 0,43\mathcal{N}(x | 3,64, 1,63). \quad (11.57)$$



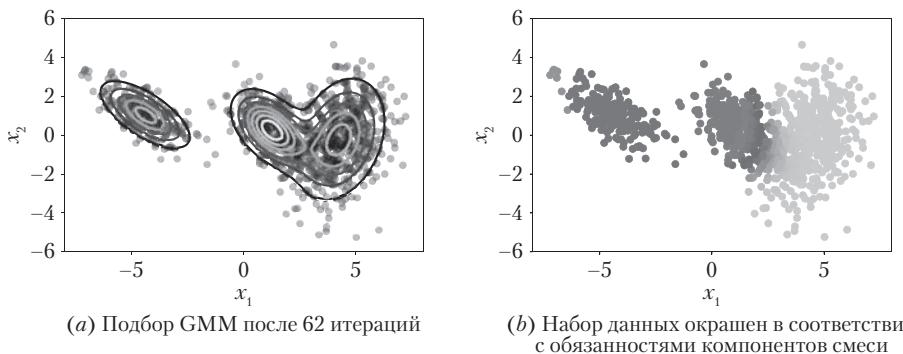
**Рис. 11.8.** EM-алгоритм, примененный к GMM из рис. 11.2.

(a) Окончательная подгонка GMM; (b) отрицательное логарифмическое правдоподобие как функция EM-итераций

Мы применили алгоритм EM к двумерному набору данных, показанному на рис. 11.1, с  $K = 3$  компонентами смеси. Рисунок 11.9 иллюстрирует некоторые шаги алгоритма EM и показывает отрицательную логарифмическую вероятность как функцию итерации EM (рис. 11.9(b)). На рис. 11.10(a) показана соответствующая окончательная подгонка GMM. Рисунок 11.10(b) визуализирует окончательные ответственности компонентов смеси для точек данных. Набор данных окрашен в соответствии с ответственностями компонентов смеси, когда EM сходится. Хотя один компонент смеси явно отвечает за данные слева, перекрытие двух кластеров данных справа могло быть создано двумя компонентами смеси. Становится ясно, что есть точки данных, которые не могут быть однозначно присвоены одному компоненту (синему или желтому), так что ответственность этих двух кластеров за эти точки составляет около 0,5.



**Рис. 11.9.** Иллюстрация алгоритма EM для подгонки модели гауссовой смеси с тремя компонентами к двумерному набору данных. (a) Набор данных; (b) отрицательная логарифмическая вероятность (чем ниже, тем лучше) как функция от итераций EM. Точки указывают итерации, для которых подходят компоненты смеси соответствующей GMM, показанные в пунктах (c)–(f). Светлые диски показывают средние значения компонентов гауссовой смеси. На рис. 11.11(a) показана окончательная подгонка GMM



**Рис. 11.10.** Соответствие GMM и его обязанности при сближении EM.

(a) GMM подходит, когда EM сходится. (b) Каждая точка данных окрашена в соответствии с обязанностями компонентов смеси

## 11.4. СКРЫТАЯ ПЕРСПЕКТИВА

Мы можем взглянуть на GMM с точки зрения модели дискретных скрытых переменных, то есть где скрытая переменная  $z$  может принимать только конечный набор значений. Это контрастирует с РСА, где скрытые переменные были числами с непрерывными значениями в  $\mathbb{R}^M$ .

Преимущества вероятностной перспективы заключаются в том, что (i) она оправдывает некоторые специальные решения, которые мы приняли в предыдущих разделах, (ii) она позволяет конкретную интерпретацию ответственности как апостериорные вероятности и (iii) итерационный алгоритм обновления параметров модели может быть выведен как алгоритм EM для оценки параметра максимального правдоподобия в моделях со скрытыми переменными.

### 11.4.1. Генеративный процесс и вероятностная модель

Чтобы вывести вероятностную модель для GMM, полезно подумать о генеративном процессе, то есть о процессе, который позволяет нам генерировать данные, используя вероятностную модель.

Мы предполагаем, что модель смеси — с  $K$  компонентами и что точка данных  $x$  может быть сгенерирована ровно одним компонентом смеси. Мы вводим двоичную индикаторную переменную  $z_k \in \{0, 1\}$  с двумя состояниями (раздел 6.2), которая указывает, сгенерировал ли  $k$ -й компонент смеси эту точку данных, так что

$$p(\mathbf{x} | z_k = 1) = \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k). \quad (11.58)$$

Определим  $\mathbf{z} := [z_1, \dots, z_K]^T \in \mathbb{R}^K$  как вектор вероятности, состоящий из  $K - 1$ , многих нулей и ровно одной 1. Например, для  $K = 3$  действительным  $\mathbf{z}$  будет  $\mathbf{z} = [z_1, z_2, z_3]^T = [0, 1, 0]^T$ , что выберет вторую компоненту смеси, так как  $z_2 = 1$ .

**ПРИМЕЧАНИЕ** Иногда такое распределение вероятностей называют «мультибулли» — обобщением распределения Бернулли на более чем два значения (Murphy, 2012). ♦

Свойства  $\mathbf{z}$  подразумевают, что  $\sum_{k=1}^K z_k = 1$ . Следовательно,  $\mathbf{z}$  — горячее кодирование (также: представление 1 из  $K$ ).

До сих пор мы предполагали, что индикаторные переменные  $z_k$  известны. Однако на практике это не так, и мы помещаем априорное распределение

$$p(\mathbf{z}) = \pi = [\pi_1, \dots, \pi_K]^T, \quad \sum_{k=1}^K \pi_k = 1, \quad (11.59)$$

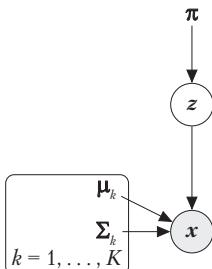
от скрытой переменной  $\mathbf{z}$ . Тогда  $k$ -я запись

$$\pi_k = p(z_k = 1) \quad (11.60)$$

этого вектора вероятности описывает вероятность того, что  $k$ -й компонент смеси сгенерировал точку данных  $\mathbf{x}$ .

**ПРИМЕЧАНИЕ** Построение этой модели скрытых переменных (см. соответствующую графическую модель на рис. 11.11) позволяет использовать очень простую процедуру выборки (генеративный процесс) для генерации данных:

1. Выборка  $z^{(i)} \sim p(\mathbf{z})$ .
2. Выборка  $\mathbf{x}^{(i)} \sim p(\mathbf{x} | z^{(i)} = 1)$ .



**Рис. 11.11.** Графическая модель для GMM с одной точкой данных

На первом этапе мы выбираем компонент смеси  $i$  (с помощью горячего кодирования  $\mathbf{z}$ ) случайным образом в соответствии с  $p(\mathbf{z}) = \pi$ ; на втором этапе мы берем

образец из соответствующего компонента смеси. Когда мы отбрасываем выборки скрытой переменной, так что у нас остается  $\mathbf{x}^{(i)}$ , у нас есть действительные выборки из GMM. Такой вид выборки, когда выборки случайных величин зависят от выборок от родителей переменной в графической модели, называется *наследственной выборкой*. ◆

Обычно вероятностная модель определяется совместным распределением данных и скрытых переменных (раздел 8.4). Используя априорную  $p(\mathbf{z})$ , определенную в (11.59) и (11.60), и условную  $p(\mathbf{x} | \mathbf{z})$  из (11.58), мы получаем все  $K$  компонент этого совместного распределения через

$$p(\mathbf{x}, z_k = 1) = p(\mathbf{x} | z_k = 1)p(z_k = 1) = \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \Sigma_k) \quad (11.61)$$

для  $k = 1, \dots, K$ , так что

$$p(\mathbf{x}, \mathbf{z}) = \begin{bmatrix} p(\mathbf{x}, z_1 = 1) \\ \vdots \\ p(\mathbf{x}, z_K = 1) \end{bmatrix} = \begin{bmatrix} \pi_1 \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1) \\ \vdots \\ \pi_K \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_K, \Sigma_K) \end{bmatrix}, \quad (11.62)$$

что полностью определяет вероятностную модель.

### 11.4.2. Правдоподобие

Чтобы получить вероятность  $p(\mathbf{x} | \theta)$  в модели со скрытыми переменными, нам нужно исключить скрытые переменные (раздел 8.4.3). В нашем случае это можно сделать, суммируя все скрытые переменные из соединения  $p(\mathbf{x}, \mathbf{z})$  в (11.62), так чтобы

$$p(\mathbf{x} | \theta) = \sum_z p(\mathbf{x} | \theta, \mathbf{z}) p(\mathbf{z} | \theta), \quad \theta := \{\boldsymbol{\mu}_k, \Sigma_k, \pi_k : k = 1, \dots, K\}. \quad (11.63)$$

Теперь мы явно обуславливаем параметры  $\theta$  вероятностной модели, которые мы ранее опускали. В (11.63) мы просуммируем по всем  $K$  возможным горячим кодировкам  $\mathbf{z}$ , которое обозначается  $\Sigma_z$ . Поскольку в каждом  $\mathbf{z}$  есть только одна ненулевая одиночная запись, существует только  $K$  возможных конфигураций/настроек  $\mathbf{z}$ . Например, если  $K = 3$ , то  $\mathbf{z}$  может иметь конфигурации

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \quad (11.64)$$

Суммирование по всем возможным конфигурациям  $\mathbf{z}$  в (11.63) эквивалентно просмотру ненулевого элемента  $\mathbf{z}$ -вектора и записи

$$p(\mathbf{x} | \theta) = \sum_z p(\mathbf{x} | \theta, \mathbf{z}) p(\mathbf{z} | \theta) = \quad (11.65a)$$

$$= \sum_{k=1}^K p(\mathbf{x} | \boldsymbol{\theta}, z_k = 1) p(z_k = 1 | \boldsymbol{\theta}) \quad (11.65b)$$

так что желаемое маржинальное распределение задается как

$$p(\mathbf{x} | \boldsymbol{\theta}) \stackrel{(11.65b)}{=} \sum_{k=1}^K p(\mathbf{x} | \boldsymbol{\theta}, z_k = 1) p(z_k = 1 | \boldsymbol{\theta}) = \quad (11.66a)$$

$$= \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad (11.66b)$$

которую мы идентифицируем как модель GMM из (11.3). Учитывая набор данных  $\mathcal{X}$ , мы сразу получаем вероятность

$$p(\mathcal{X} | \boldsymbol{\theta}) = \prod_{n=1}^N p(\mathbf{x}_n | \boldsymbol{\theta}) \stackrel{(11.66b)}{=} \prod_{n=1}^N \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad (11.67)$$

что и есть правдоподобие GMM из (11.9). Таким образом, модель со скрытыми переменными со скрытыми индикаторами  $z_k$  — это эквивалентный подход к модели гауссовой смеси.

### 11.4.3. Апостериорное распределение

Давайте кратко рассмотрим апостериорное распределение скрытой переменной  $\mathbf{z}$ . Согласно теореме Байеса, апостериорная часть  $k$ -го компонента, сгенерировавшая точку данных  $\mathbf{x}$ ,

$$p(z_k = 1 | \mathbf{x}) = \frac{p(z_k = 1) p(\mathbf{x} | z_k = 1)}{p(\mathbf{x})}, \quad (11.68)$$

где маргинал  $p(x)$  задан в (11.66b). Это дает апостериорное распределение для  $k$ -й индикаторной переменной  $z^k$

$$p(z_k = 1 | \mathbf{x}) = \frac{p(z_k = 1) p(\mathbf{x} | z_k = 1)}{\sum_{j=1}^K p(z_j = 1) p(\mathbf{x} | z_j = 1)} = \frac{\pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}, \quad (11.69)$$

которое мы определяем как ответственность  $k$ -го компонента смеси за точку данных  $\mathbf{x}$ . Отметим, что мы опустили явное обусловливание параметров GMM  $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ , где  $k = 1, \dots, K$ .

### 11.4.4. Расширение до полного набора данных

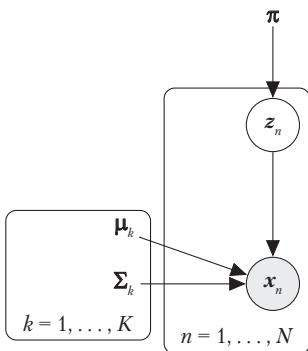
До сих пор мы обсуждали только случай, когда набор данных состоит только из одной точки данных  $\mathbf{x}$ . Однако концепции априорного и апостериорного могут быть непосредственно расширены на случай  $N$  точек данных  $\mathcal{X} := \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ .

В вероятностной интерпретации GMM каждая точка данных  $\mathbf{x}_n$  обладает собственной скрытой переменной

$$\mathbf{z}_n = [z_{n1}, \dots, z_{nK}]^\top \in \mathbb{R}^K. \quad (11.70)$$

Раньше (когда мы рассматривали только одну точку данных  $\mathbf{x}$ ) мы опускали индекс  $n$ , но теперь это становится важным.

Мы разделяем одно и то же априорное распределение  $\pi$  для всех скрытых переменных  $\mathbf{z}_n$ . Соответствующая графическая модель показана на рис. 11.12, где мы используем обозначения на табличке.



**Рис. 11.12.** Графическая модель GMM для  $N$  точек

Условное распределение  $p(\mathbf{x}_1, \dots, \mathbf{x}_N | \mathbf{z}_1, \dots, \mathbf{z}_N)$  факторизуется по точкам данных и задается как

$$p(\mathbf{x}_1, \dots, \mathbf{x}_N | \mathbf{z}_1, \dots, \mathbf{z}_N) = \prod_{n=1}^N p(\mathbf{x}_n | \mathbf{z}_n). \quad (11.71)$$

Чтобы получить апостериорное распределение  $p(z_{nk} = 1 | \mathbf{x}_n)$ , мы следуем тем же рассуждениям, что и в разделе 11.4.3, и применяем теорему Байеса для получения

$$p(z_{nk} = 1 | \mathbf{x}_n) = \frac{p(\mathbf{x}_n | z_{nk} = 1) p(z_{nk} = 1)}{\sum_{j=1}^K p(\mathbf{x}_n | z_{nj} = 1) p(z_{nj} = 1)} = \quad (11.72a)$$

$$= \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} = r_{nk}. \quad (11.72b)$$

Это означает, что  $p(z_k = 1 | \mathbf{x}_n)$  — это (апостериорная) вероятность того, что  $k$ -й компонент смеси сгенерировал точку данных  $\mathbf{x}_n$ , и соответствует ответствен-

ности  $r_{nk}$ , которую мы ввели в (11.17). Теперь обязанности также имеют не только интуитивно понятную, но и математически обоснованную интерпретацию как апостериорные вероятности.

### 11.4.5. Еще раз про EM-алгоритм

Алгоритм EM, который мы представили как итеративную схему для оценки максимального правдоподобия, может быть принципиально выведен с точки зрения скрытых переменных. Учитывая текущую настройку  $\theta^{(t)}$  параметров модели, E-шаг вычисляет ожидаемое логарифмическое правдоподобие

$$Q(\boldsymbol{\theta} | \boldsymbol{\theta}^{(t)}) = \mathbb{E}_{\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(t)}} [\log p(\mathbf{x}, \mathbf{z} | \boldsymbol{\theta})]; \quad (11.73a)$$

$$\int \log p(\mathbf{x}, \mathbf{z} | \boldsymbol{\theta}) p(\mathbf{z} | \mathbf{x}, \boldsymbol{\theta}^{(t)}) d\mathbf{z}, \quad (11.73b)$$

где математическое ожидание  $\log p(\mathbf{x}, \mathbf{z} | \boldsymbol{\theta})$  берется относительно апостериорного  $p(\mathbf{z} | \mathbf{x}, \boldsymbol{\theta}^{(t)})$  скрытых переменных. На  $M$ -шаге выбирается обновленный набор параметров модели  $\boldsymbol{\theta}^{(t+1)}$  путем максимизации (11.73b).

Хотя итерация EM увеличивает логарифмическую вероятность, нет никаких гарантий, что EM сходится к решению с максимальным правдоподобием. Возможно, что алгоритм EM сходится к локальному максимуму логарифма правдоподобия. Различные инициализации параметров  $\boldsymbol{\theta}$  могут использоваться в нескольких прогонах EM, чтобы снизить риск попадания в плохой локальный оптимум. Мы не будем вдаваться в подробности здесь, а обратимся к превосходным изложениям Rogers and Girolami (2016) и Bishop (2006).

## 11.5. ДОПОЛНИТЕЛЬНОЕ ЧТЕНИЕ

GMM можно рассматривать как генеративную модель в том смысле, что легко генерировать новые данные с использованием наследственной выборки (Bishop, 2006). Для заданных параметров GMM  $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$ , мы выбираем индекс  $k$  из вектора вероятности  $[\pi_1, \dots, \pi_K]$ , а затем выбираем точку данных  $\mathbf{x} \sim \mathcal{N}(\mu_k, \Sigma_k)$ . Если мы повторим это  $N$  раз, мы получим набор данных, созданный GMM. Рисунок 11.1 был создан с использованием этой процедуры.

На протяжении всей главы мы предполагали, что количество компонентов  $K$  известно. На практике часто бывает не так. Однако мы могли бы использовать вложенную перекрестную проверку, рассмотренную в разделе 8.6.1, чтобы найти хорошие модели.

Модели гауссовой смеси тесно связаны с алгоритмом кластеризации  $K$  средних. Алгоритм кластеризации  $K$  средних также использует алгоритм EM для назначения точек данных кластерам. Если мы рассматриваем средние в GMM как

центры кластеров и игнорируем ковариации (или устанавливаем их равными  $I$ ), мы приходим к алгоритму  $K$  средних. Как также хорошо описано MacKay (2003), алгоритм  $K$  средних выполняет «жесткое» назначение точек данных центрам кластера  $\mu_k$ , тогда как GMM выполняет «мягкое» назначение через обязанности.

Мы только коснулись перспективы GMM и алгоритма EM со скрытыми переменными. Обратите внимание, что EM может использоваться для обучения параметрам в общих моделях с латентными переменными, например в нелинейных моделях пространства состояний (Ghahramani and Roweis, 1999; Roweis and Ghahramani, 1999), а также для обучения с подкреплением, как обсуждалось в Barber (2012). Следовательно, перспектива скрытых переменных в GMM полезна для принципиального вывода соответствующего алгоритма EM (Bishop, 2006; Barber, 2012; Murphy, 2012).

Мы обсудили только оценку максимального правдоподобия (с помощью алгоритма EM) для нахождения параметров GMM. Здесь также применимы стандартные критические замечания максимальной вероятности:

- Как и в случае линейной регрессии, максимальная вероятность может пострадать от серьезного переобучения. В случае GMM это происходит, когда среднее значение компонента смеси идентично точке данных, а ковариация стремится к 0. Затем вероятность приближается к бесконечности. Bishop (2006) и Barber (2012) подробно обсуждают этот вопрос.
- Мы получаем точечную оценку параметров  $\pi_k, \mu_k, \Sigma_k$  только для  $k = 1, \dots, K$ , что не указывает на неопределенность значений параметров. Байесовский подход предусматривает априорность параметров, которую можно использовать для получения апостериорного распределения параметров. Эта апостериорная оценка позволяет нам вычислить свидетельство модели (предельное правдоподобие), которое можно использовать для сравнения моделей, что дает нам принципиальный способ определения количества компонентов смеси. К сожалению, вывод в замкнутой форме невозможен в этой настройке, потому что для этой модели нет сопряженного априорного значения. Однако приближения, такие как вариационный вывод, можно использовать для получения приблизительного апостериорного анализа (Bishop, 2006).

В этой главе мы обсудили модели смеси для оценки плотности. Существует множество доступных методов оценки плотности. На практике мы часто используем гистограммы и оценку плотности ядра.

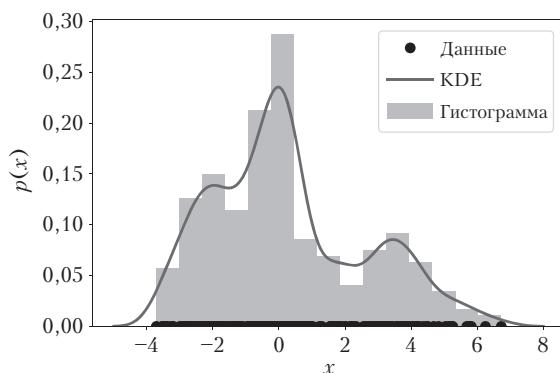
*Гистограммы* обеспечивают непараметрический способ представления непрерывных плотностей и были предложены Pearson (1895). Гистограмма строится путем «разделения» пространства данных и подсчета количества точек данных, попадающих в каждую ячейку. Затем в центре каждой ячейки рисуется полоса, высота которой пропорциональна количеству точек данных в этой ячейке. Раз-

мер ячейки является критическим гиперпараметром, и неправильный выбор может привести к переобучению или неполному подбору. Перекрестная проверка, как описано в разделе 8.2.4, может использоваться для определения подходящего размера ячейки.

*Оценка плотности ядра* (Kernel density estimation, KDE), независимо предложенная Rosenblatt (1956) и Parzen (1962), является непараметрическим способом оценки плотности. Учитывая  $N$  независимых и одинаково распределенных выборок, оценщик плотности ядра представляет лежащее в основе распределение как

$$p(\mathbf{x}) = \frac{1}{Nh} \sum_{n=1}^N k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right), \quad (11.74)$$

где  $k$  — функция ядра, то есть неотрицательная функция, которая интегрируется до 1, а  $h > 0$  — параметр сглаживания/полосы пропускания, который играет такую же роль, как размер ячейки в гистограммах. Обратите внимание, что мы помещаем ядро в каждую точку данных  $\mathbf{x}_n$  в наборе данных. Обычно используемые ядерные функции — это равномерное распределение и распределение Гаусса. Оценки плотности ядра тесно связаны с гистограммами, но, выбирая подходящее ядро, мы можем гарантировать гладкость оценки плотности. На рис. 11.13 показана разница между гистограммой и оценкой плотности ядра (с ядром в форме Гаусса) для данного набора данных из 250 точек данных.



**Рис. 11.13.** Гистограмма (столбцы с серой заливкой) и оценка плотности ядра (черная линия). Механизм оценки плотности ядра дает гладкую оценку отчетной плотности, в то время как гистограмма дает несглаженную счетную меру и позволяет измерить, сколько (черных) точек попадают в один интервал

# 12

## Классификация методом опорных векторов

Зачастую мы хотим, чтобы алгоритм машинного обучения в качестве предсказания выбирал один из дискретного множества классов. Например, почтовый клиент должен отличать спам от личного письма (два класса). При астрономических наблюдениях нужно идентифицировать небесный объект как галактику, звезду или планету. Возможных классов, как правило, немного, и, что важнее, у них обычно нет дополнительной структуры<sup>1</sup>. В данной главе мы рассмотрим задачу *бинарной классификации* — предсказание одного из двух возможных значений. Заметим, что постановка здесь отличается от задачи из главы 9 тем, что там мы предсказывали значения из непрерывного множества.

В задаче *бинарной классификации* множество возможных ответов (меток классов) состоит из двух элементов, которые мы обозначим как  $\{+1, -1\}$ . Иными словами, мы работаем с предсказаниями вида

$$f: \mathbb{R}^D \rightarrow \{+1, -1\}. \quad (12.1)$$

Вспомним, как в главе 8 мы записывали каждый пример (объект обучающей выборки)  $\mathbf{x}_n$  в виде вектора признаков, состоящего из  $D$  вещественных чисел<sup>2</sup>. Метки классов часто называют соответственно положительным и отрицательным классом. Заметим, что «положительность» класса  $+1$  вовсе не обязательно означает что-то хорошее. Например, в задаче диагностики рака, как правило, метка  $+1$  соответствует пациенту, у которого диагностировано это заболевание.

<sup>1</sup> Примером структуры является наличие порядка вариантов ответа: например, размеры футбольок упорядочены (S, M, L).

<sup>2</sup> Объекты обучающей выборки также называют точками данных, входными значениями или примерами.

В принципе, можно использовать в качестве меток любое множество из двух элементов, например, {истина, ложь}, {0, 1} или {красный, синий}<sup>1</sup>. Задача бинарной классификации хорошо изучена, и мы отложим обзор альтернативных подходов до раздела 12.6.

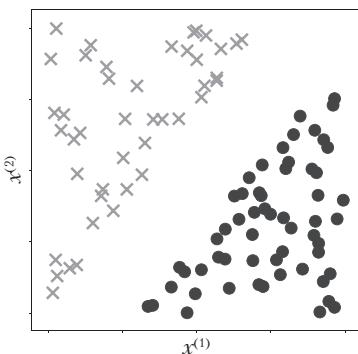
Сейчас для решения задачи бинарной классификации рассмотрим так называемый метод опорных векторов (support vector machine, SVM). Эта задача, как и задача регрессии, относится к машинному обучению с учителем. Имеется набор объектов  $\mathbf{x}_n \in \mathbb{R}^D$  с соответствующими бинарными метками  $y_n \in \{+1, -1\}$ . По обучающей выборке из пар «объект — метка»  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$  мы хотим найти параметры модели, дающие наименьшую ошибку классификации. Как и в главе 9, мы строим линейную модель, а всю нелинейность зашиваем в преобразование признаков  $\varphi$  (9.13). Мы еще вернемся к преобразованию  $\varphi$  в разделе 12.4.

Метод опорных векторов на данный момент является одним из лучших для многих приложений, а кроме того, для него есть теоретические гарантии качества результата (Steinwart and Christmann, 2008). Есть две главные причины тому, что мы решили рассказывать про бинарную классификацию, используя SVM. Во-первых, этот метод позволяет думать о задачах обучения с учителем геометрически. В то время как в главе 9 мы рассматривали задачу МО в терминах вероятностных моделей и решали ее с помощью оценок максимального правдоподобия и байесовского инференса, в данной главе мы рассмотрим другой подход, основанный на геометрических рассуждениях. Важнейшую роль в нем играют понятия, введенные в главе 3, например скалярные произведения и проекции. Во-вторых, задачи, в которых используется метод SVM, не имеют аналитического решения — так что нам остается пользоваться различными методами оптимизации из главы 7.

Метод опорных векторов по своим основным идеям отличается от подхода максимального правдоподобия из главы 9. В последнем модель, приводящая в итоге к задаче оптимизации, строится с помощью вероятностного подхода (рассматривается распределение данных). Метод опорных векторов строит функцию, оптимизируемую при обучении, используя геометрические соображения. Нечто подобное мы уже видели в главе 10, выводя из геометрических идей метод главных компонент. В случае SVM мы начинаем с построения цепевой функции, а потом минимизируем ее на обучающей выборке методами минимизации эмпирического риска (раздел 8.1). Можно трактовать этот подход как построение специфической функции потерь.

---

<sup>1</sup> В вероятностных задачах удобнее использовать бинарное представление {0, 1}, см. примечание после примера 6.12.



**Рис. 12.1.** Пример двумерных данных, на котором видно наличие линейного классификатора, способного разделить светлые крестики и темные кружочки

Давайте сформулируем задачу оптимизации, возникающую при использовании метода опорных векторов на обучающем множестве пар «объект — метка». На рис. 12.1 показаны данные, которые можно разделить на два класса гиперплоскостью. Каждый из объектов  $\mathbf{x}_n$  (векторов размерности 2) будет точкой в двумерном пространстве (с координатами  $x_n^{(1)}$  и  $x_n^{(2)}$ ), а соответствующая метка показана одним из двух символов — светлым крестиком или темным кружочком. Слово «гиперплоскость» часто употребляется в МО, и мы уже знакомились с гиперплоскостями в разделе 2.8. Гиперплоскостью называют аффинное подпространство размерности  $D - 1$  (в векторном пространстве размерности  $D$ ). Объекты делятся на два класса (соответствующие двум возможным меткам), при этом их признаки (координаты представляющих их векторов) должны позволять нам разделить классы прямой линией.

Далее мы формализуем идею «найти линейный разделитель классов». Введем идею отступа и расширим понятие линейного разделения, позволив примерам попадать на «неправильную» сторону, то есть допустив ошибки классификации. Покажем два способа формализации метода опорных векторов: геометрический (в разделе 12.2.4) и с помощью функции потерь (в разделе 12.2.5). Выведем двойственную задачу SVM, используя множители Лагранжа (раздел 7.2). Двойственная задача поможет нам увидеть третий способ смотреть на SVM: в терминах выпуклых оболочек для примеров каждого класса (раздел 12.3.2). В конце вкратце обсудим ядра и численное решение задачи SVM с нелинейным ядром.

## 12.1. РАЗДЕЛЯЮЩИЕ ГИПЕРПЛОСКОСТИ

Один из способов вычислить сходство между двумя примерами, заданными векторами  $\mathbf{x}_i$  и  $\mathbf{x}_j$ , — найти скалярное произведение  $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ . Из раздела 3.2 мы знаем, что скалярные произведения связаны с углом между двумя векторами. Значение скалярного произведения зависит от длины (нормы) каждого из векторов:

торов. Далее скалярные произведения позволяют нам строго определить такие геометрические понятия, как ортогональность и проекции.

Главная идея многих алгоритмов классификации — представить данные как векторы в  $\mathbb{R}^D$ , а затем разбить пространство на части, в идеале так, чтобы в каждой из частей находились примеры одного класса и только они. В случае бинарной классификации пространство будет разделено на две части, соответствующие положительному и отрицательному классам. Мы рассмотрим особенно удобный способ разделить пространство на две части — с помощью гиперплоскости.

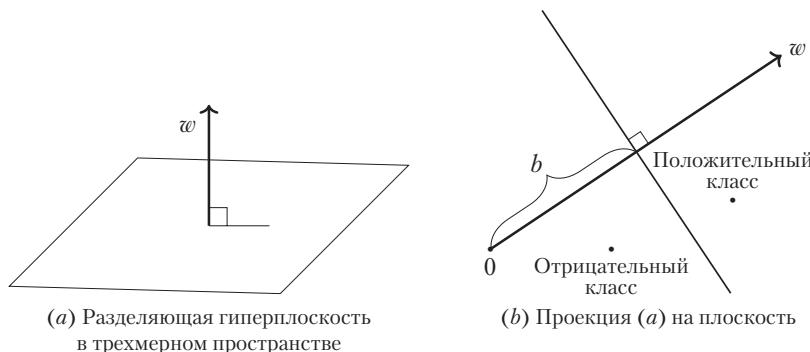
Рассмотрим пример  $\mathbf{x} \in \mathbb{R}^D$  из пространства данных. Возьмем функцию

$$f: \mathbb{R}^D \rightarrow \mathbb{R}; \quad (12.2a)$$

$$\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle + b \quad (12.2b)$$

с параметрами  $\mathbf{w} \in \mathbb{R}^D$  и  $b \in \mathbb{R}$ . Вспомним (раздел 2.8), что гиперплоскости являются аффинными подпространствами. Таким образом, гиперплоскость, разделяющую два класса в нашей задаче, можно задать как

$$\{\mathbf{x} \in \mathbb{R}^D : f(\mathbf{x}) = 0\}. \quad (12.3)$$



**Рис. 12.2.** Уравнение разделяющей гиперплоскости (12.3). (a) Стандартное представление в трехмерном пространстве. (b) Для удобства изображения мы смотрим на гиперплоскость «с ребра»

Гиперплоскость изображена на рис. 12.2, где вектор  $\mathbf{w}$  — нормаль к гиперплоскости, а  $b$  — смещение. Мы можем показать, что  $\mathbf{w}$  — нормаль к гиперплоскости (12.3), выбрав две произвольные точки  $\mathbf{x}_a$  и  $\mathbf{x}_b$  на гиперплоскости и показав, что соединяющий их вектор ортогонален  $\mathbf{w}$ . Это можно записать так:

$$f(\mathbf{x}_a) - f(\mathbf{x}_b) = \langle \mathbf{w}, \mathbf{x}_a \rangle + b - (\langle \mathbf{w}, \mathbf{x}_b \rangle + b) = \quad (12.4a)$$

$$= \langle \mathbf{w}, \mathbf{x}_a - \mathbf{x}_b \rangle, \quad (12.4b)$$

где вторая строка получена из линейности скалярного произведения (раздел 3.2). Так как мы выбрали  $\mathbf{x}_a$  и  $\mathbf{x}_b$  принадлежащими гиперплоскости, то  $f(\mathbf{x}_a) = 0$  и  $f(\mathbf{x}_b) = 0$ , значит,  $\langle \mathbf{w}, \mathbf{x}_a - \mathbf{x}_b \rangle = 0$ . Вспомним, что два вектора ортогональны, если их скалярное произведение равно нулю. Таким образом,  $\mathbf{w}$  ортогонален любому вектору из гиперплоскости.

**ПРИМЕЧАНИЕ** Из главы 2 мы знаем, что можно смотреть на векторы с разных точек зрения. В данной главе мы будем воспринимать вектор  $\mathbf{w}$  как «стрелку», указывающую направление, то есть как геометрический вектор. Напротив, о векторе  $\mathbf{x}$  мы будем думать как о точке, то есть как о наборе координат относительно стандартного базиса. ♦

Получив тестовый объект, мы классифицируем его как положительный или отрицательный, в зависимости от того, с какой стороны от гиперплоскости он оказался. Заметим, что формула (12.3) не только задает гиперплоскость, но и определяет направление, а значит, «положительную и отрицательную стороны» по отношению к гиперплоскости. Таким образом, чтобы классифицировать тестовый пример  $\mathbf{x}_{\text{test}}$ , мы должны вычислить значение функции  $f(\mathbf{x}_{\text{test}})$ , а затем присвоить примеру метку класса +1 при  $f(\mathbf{x}_{\text{test}}) \geq 0$  и метку класса -1 в противном случае.

С геометрической точки зрения, положительные примеры лежат «над» гиперплоскостью, а отрицательные — «под» ней. При обучении классификатора мы хотим добиться того, чтобы примеры с положительными метками находились с положительной стороны от гиперплоскости, то есть

$$\langle \mathbf{w}, \mathbf{x}_n \rangle + b \geq 0 \quad \text{при } y_n = +1, \quad (12.5)$$

а примеры с отрицательными метками — с отрицательной стороны, то есть

$$\langle \mathbf{w}, \mathbf{x}_n \rangle + b < 0 \quad \text{при } y_n = -1. \quad (12.6)$$

Помочь в геометрическом понимании классификации может рис. 12.2. Два приведенных выше условия часто объединяют формулой

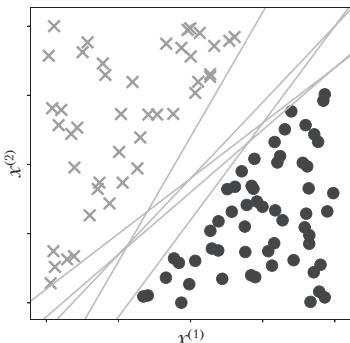
$$y_n (\langle \mathbf{w}, \mathbf{x}_n \rangle + b) \geq 0. \quad (12.7)$$

Уравнение (12.7) эквивалентно (12.5) и (12.6) благодаря умножению (12.5) и (12.6) соответственно на  $y_n = 1$  и  $y_n = -1$ .

## 12.2. ПРЯМАЯ ЗАДАЧА МЕТОДА ОПОРНЫХ ВЕКТОРОВ

Мы знакомы с понятием расстояния от точки до гиперплоскости и теперь готовы обсуждать метод опорных векторов. Для линейно разделимого набора данных

$\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$  существует бесконечное число подходящих гиперплоскостей (см. рис. 12.3), а значит, и классификаторов, решающих задачу без ошибок (на обучающей выборке). Один из вариантов, как прийти к единственному решению — выбрать гиперплоскость, максимизирующую отступ между классами. Другими словами, мы хотим, чтобы отступ между положительными и отрицательными примерами был как можно больше (раздел 12.2.1)<sup>1</sup>. Далее мы займемся вычислением расстояния от объекта до гиперплоскости. Вспомним, что ближайшая к данной точке  $\mathbf{x}_n$  точка на гиперплоскости получается с помощью ортогональной проекции (раздел 3.8).



**Рис. 12.3.** Возможные разделяющие гиперплоскости. Существует множество линейных классификаторов (сплошные линии), отделяющих светлые крестики от темных кружочек

## 12.2.1. Понятие зазора

Понятие *отступа* достаточно простое: это расстояние от разделяющей гиперплоскости до ближайшего объекта из выборки<sup>2</sup> (здесь мы предполагаем, что выборка линейно разделима). Однако при дальнейшей формализации возникает один технический нюанс: нам нужна шкала для измерения расстояний. Можно просто использовать ту же шкалу, в которой нам даны значения данных  $\mathbf{x}_n$ . Здесь возможна проблема: если мы изменим единицы измерения для  $\mathbf{x}_n$ , их значения также поменяются, а вместе с ними и расстояние до гиперплоскости. Как мы вскоре увидим, шкала измерения расстояний будет определена из самого уравнения гиперплоскости (12.3).

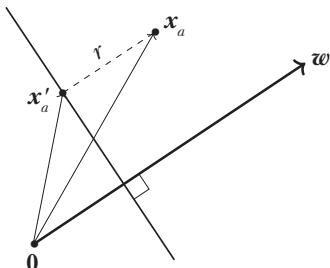
Рассмотрим гиперплоскость  $\langle \mathbf{w}, \mathbf{x} \rangle + b$  и пример  $\mathbf{x}_a$ , как показано на рис. 12.4. Не умоляя общности, мы можем считать, что объект  $\mathbf{x}_a$  расположен с положительной стороны от гиперплоскости, то есть  $\langle \mathbf{w}, \mathbf{x}_a \rangle + b > 0$ . Мы хотим вычислить расстояние  $r > 0$  от  $\mathbf{x}_a$  до гиперплоскости. Для этого мы находим ортогональную проекцию (раздел 3.8)  $\mathbf{x}_a$  на гиперплоскость, обозначаемую  $\mathbf{x}'_a$ . Так как  $\mathbf{w}$  орто-

<sup>1</sup> Классификатор с большим отступом обладает хорошей способностью к обобщению (Steinwart and Christmann, 2008).

<sup>2</sup> Ближайших объектов может быть несколько.

гонален гиперплоскости, расстояние  $r$  — это коэффициент, на который умножается  $\mathbf{w}$ . Если известна длина  $\mathbf{w}$ , можно найти коэффициент  $r$  и «настоящее» расстояние от  $\mathbf{x}_a$  до  $\mathbf{x}'_a$ . Для удобства мы возьмем вектор единичной длины, полученный делением  $\mathbf{w}$  на его норму  $\|\mathbf{w}\|$  (раздел 2.4), и заметим, что

$$\mathbf{x}_a = \mathbf{x}'_a + r \frac{\mathbf{w}}{\|\mathbf{w}\|}. \quad (12.8)$$



**Рис. 12.4.** Сложение векторов для нахождения расстояния до гиперплоскости:  $\mathbf{x}_a = \mathbf{x}'_a + r \frac{\mathbf{w}}{\|\mathbf{w}\|}$

Другой способ — воспринимать число  $r$  как координату  $\mathbf{x}_a$  в подпространстве, натянутом на  $\mathbf{w}/\|\mathbf{w}\|$ . Теперь  $r$  — расстояние от  $\mathbf{x}_a$  до гиперплоскости, и если  $\mathbf{x}_a$  — ближайшая к гиперплоскости точка данных, то  $r$  называют отступом.

Мы хотим, чтобы положительные примеры находились на расстоянии, не меньшем  $r$ , от гиперплоскости, а отрицательные примеры — также на расстоянии, не меньшем  $r$ , но с противоположной стороны. Аналогично тому, как мы получали формулу (12.7) из (12.5) и (12.6), можно переформулировать наши цели как

$$y_n (\langle \mathbf{w}, \mathbf{x}_n \rangle + b) \geq r. \quad (12.9)$$

Таким образом мы объединили требования к расстояниям между положительными примерами и гиперплоскостью и к расстояниям между отрицательными примерами и гиперплоскостью в одно неравенство.

Так как нас интересует только направление вектора параметров  $\mathbf{w}$ , мы добавляем предположение, что  $\mathbf{w}$  — вектор единичной длины,  $\|\mathbf{w}\| = 1$ , где  $\|\mathbf{w}\| = \sqrt{\mathbf{w}^T \mathbf{w}}$  — евклидова норма (раздел 3.1)<sup>1</sup>. Это предположение также делает более понятным определение расстояния  $r$  (12.8) как коэффициента при векторе единичной длины.

**ПРИМЕЧАНИЕ** Читатель, знакомый с другими определениями отступа, заметит, что наше предположение  $\|\mathbf{w}\| = 1$  отличается от стандартного изложения

<sup>1</sup> Использование других скалярных произведений (раздел 3.2) мы увидим в разделе 12.4.

метода опорных векторов, например у Шелькопфа и Смолы (Schölkopf and Smola, 2002). В разделе 12.2.3 мы покажем эквивалентность обоих подходов.



Объединяя три требования в одну задачу оптимизации с ограничениями (условной оптимизации), получим целевую функцию

$$\max_{\mathbf{w}, b, r} \underbrace{r}_{\text{отступ}}$$

при условиях  $\underbrace{y_n(\langle \mathbf{w}, \mathbf{x}_n \rangle + b) \geq r}_{\text{соответствие данным}}, \underbrace{\|\mathbf{w}\| = 1}_{\text{нормализация}}, r > 0,$

(12.10)

это означает, что мы хотим максимизировать отступ  $r$ , при этом данные должны лежать с правильной стороны от гиперплоскости.

**ПРИМЕЧАНИЕ** Понятие отступа активно проникает в разные области машинного обучения. Владимир Вапник и Алексей Червоненкис использовали его, чтобы показать, что при большом отступе «сложность» класса функций низкая и возможно обучение (Vapnik, 2000). Оно также оказалось полезным в различных методах теоретического анализа ошибок обобщения (Steinwart and Christmann, 2008; Shalev-Shwartz and Ben-David, 2014).



## 12.2.2. Нахождение зазора: традиционный способ

В предыдущем разделе мы вывели функцию (12.10), заметили, что нас интересует направление  $\mathbf{w}$ , но не его длина, и приняли, что  $\|\mathbf{w}\| = 1$ . В этом разделе мы придем к задаче максимизации отступа из других предпосылок. Вместо взятия нормализованного вектора параметров мы поменяем шкалу измерения для данных. Мы делаем это перенормирование так, чтобы значение предсказания  $\langle \mathbf{w}, \mathbf{x} \rangle + b$  на ближайшем объекте было равно 1. Обозначим ближайший к гиперплоскости пример за  $\mathbf{x}_a^1$ .

Рисунок 12.5 совпадает с рис. 12.4, за исключением перенормирования осей, так что пример  $\mathbf{x}_a$  лежит в точности на расстоянии отступа,  $\langle \mathbf{w}, \mathbf{x}_a \rangle + b = 1$ . Так как точка  $\mathbf{x}'_a$  — ортогональная проекция  $\mathbf{x}_a$  на гиперплоскость, она по определению лежит на гиперплоскости, то есть

$$\langle \mathbf{w}, \mathbf{x}'_a \rangle + b = 0. \quad (12.11)$$

Подставив (12.8) в (12.11), получим

$$\left\langle \mathbf{w}, \mathbf{x}_a - r \frac{\mathbf{w}}{\|\mathbf{w}\|} \right\rangle + b = 0. \quad (12.12)$$

---

<sup>1</sup> Напомним, что пока мы рассматриваем линейно разделимые выборки.

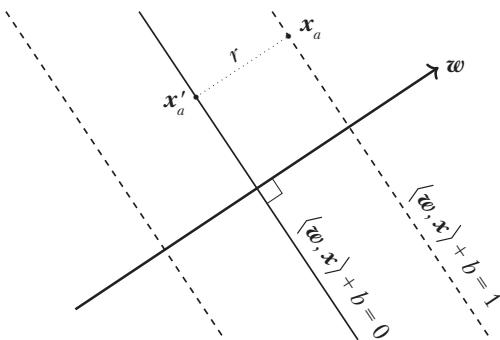


Рис. 12.5. Нахождение отступа:  $r = 1/\|w\|$

Пользуясь билинейностью скалярного произведения (раздел 3.2), приходим к

$$\langle w, x_a \rangle + b - r \frac{\langle w, w \rangle}{\|w\|} = 0. \quad (12.13)$$

Заметим, что благодаря нашему перенормированию первое слагаемое равно 1, то есть  $\langle w, x_a \rangle + b = 1$ .

Из формулы (3.16) раздела 3.1 мы знаем, что  $\langle w, w \rangle = \|w\|^2$ , и второе слагаемое превращается в  $r\|w\|$ . Благодаря этим упрощениям получаем

$$r = \frac{1}{\|w\|}. \quad (12.14)$$

Таким образом мы выразили расстояние  $r$  через нормальный к гиперплоскости вектор  $w$ . На первый взгляд, это равенство нелогично: мы выразили расстояние от гиперплоскости через длину вектора  $w$ , но этот вектор мы пока не знаем. Однако можно думать о расстоянии  $r$  как о временной величине, введенной только для данного рассуждения<sup>1</sup>. Далее в этой главе мы будем обозначать расстояние до гиперплоскости как  $\frac{1}{\|w\|}$ . В разделе 12.2.3 мы увидим, что выбор отступа, равного 1, эквивалентен нашему предыдущему предположению  $\|w\| = 1$  из раздела 12.2.1.

Мы хотим, чтобы и положительные, и отрицательные примеры находились хотя бы на единичном расстоянии от гиперплоскости. Воспользуемся тем же способом, что и при выводе (12.9), и запишем наши условия как

$$y_n (\langle w, x_n \rangle + b) \geq 1. \quad (12.15)$$

<sup>1</sup> Можно также рассматривать расстояние как «ошибку проекции»  $x_a$  на гиперплоскость.

Объединяя требования максимизации отступа и того, чтобы примеры находились с правильной стороны от гиперплоскости, получим

$$\max_{\mathbf{w}, b} \frac{1}{\|\mathbf{w}\|} \quad (12.16)$$

$$\text{при условиях } y_n(\langle \mathbf{w}, \mathbf{x}_n \rangle + b) \geq 1 \text{ для } n = 1, \dots, N. \quad (12.17)$$

Вместо максимизации обратной нормы, как в (12.16), часто минимизируют квадрат нормы<sup>1</sup>, нередко вводя множитель  $1/2$ , не влияющий на оптимальные значения  $\mathbf{w}, b$ , но дающий более приятный вид градиента. Тогда наша целевая функция превращается в

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 \quad (12.18)$$

$$\text{при условиях } y_n(\langle \mathbf{w}, \mathbf{x}_n \rangle + b) \geq 1 \text{ для } n = 1, \dots, N. \quad (12.19)$$

Задача (12.18) известна как *SVM с жестким зазором* (hard margin SVM). «Жесткость» проявляется в том, что данная формулировка не допускает нарушений условий использования зазора. В разделе 12.2.4 мы увидим, что это условие можно ослабить, допустив нарушения в тех случаях, когда данные не являются линейно разделимыми.

### 12.2.3. Почему можно взять зазор, равный 1

В разделе 12.2.1 мы договорились максимизировать значение  $r$ , расстояния от ближайшего примера до гиперплоскости. В разделе 12.2.2 мы изменяли масштаб данных так, чтобы ближайший пример лежал на расстоянии 1 от гиперплоскости. В данном разделе мы свяжем эти два подхода и поймем, что на самом деле они равносильны.

**Теорема 12.1.** *Максимизация зазора  $r$  при нормализованных весах, как в формуле (12.10):*

$$\max_{\mathbf{w}, b, r} \underbrace{\frac{r}{\|\mathbf{w}\|}}_{\text{зазор}} \quad (12.20)$$

при условиях  $y_n(\langle \mathbf{w}, \mathbf{x}_n \rangle + b) \geq r$ ,  $\|\mathbf{w}\| = 1$ ,  $r > 0$ ,

соответствие данным      нормализация

эквивалентна перенормированию данных, при котором зазор равен 1:

$$\min_{\mathbf{w}, b} \underbrace{\frac{1}{2} \|\mathbf{w}\|^2}_{\text{зазор}} \quad (12.21)$$

при условиях  $\underbrace{y_n(\langle \mathbf{w}, \mathbf{x}_n \rangle + b)}_{\text{соответствие данным}} \geq 1$ .

---

<sup>1</sup> Квадрат нормы дает выпуклую задачу квадратичного программирования (раздел 12.5).

*Доказательство.* Рассмотрим (12.20). Так как возведение в квадрат строго монотонно на положительной полуоси, точка достижения максимума не изменится, если брать в качестве целевой функции  $r^2$ . Так как  $\|\mathbf{w}\| = 1$ , можно перепараметризовать уравнение с ненормализованным вектором весов  $\mathbf{w}'$ , используя  $\frac{\mathbf{w}'}{\|\mathbf{w}'\|}$ . Мы получим

$$\max_{\mathbf{w}', b, r} \quad r^2$$

при условиях  $y_n \left( \left\langle \frac{\mathbf{w}'}{\|\mathbf{w}'\|}, \mathbf{x}_n \right\rangle + b \right) \geq r, \quad r > 0.$  (12.22)

Условия (12.22) явно задают, что  $r$  положительно. Поэтому можно разделить обе части первого условия на  $r^1$  и получить

$$\max_{\mathbf{w}', b, r} \quad r^2$$

при условиях  $y_n \left( \left\langle \underbrace{\frac{\mathbf{w}'}{\|\mathbf{w}'\|} r}_{\mathbf{w}''}, \mathbf{x}_n \right\rangle + \frac{b}{r} \right) \geq 1, \quad r > 0,$  (12.23)

переименовав параметры в  $\mathbf{w}''$  и  $b''$ . Так как  $\mathbf{w}'' = \frac{\mathbf{w}'}{\|\mathbf{w}'\| r}$ , приходим к

$$\|\mathbf{w}''\| = \left\| \frac{\mathbf{w}'}{\|\mathbf{w}'\| r} \right\| = \frac{1}{r} \cdot \left\| \frac{\mathbf{w}'}{\|\mathbf{w}'\|} \right\| = \frac{1}{r}. \quad (12.24)$$

После подстановки результата в (12.23) задача выглядит как

$$\max_{\mathbf{w}'', b''} \quad \frac{1}{\|\mathbf{w}''\|^2}$$

при условиях  $y_n (\langle \mathbf{w}'', \mathbf{x}_n \rangle + b'') \geq 1.$  (12.25)

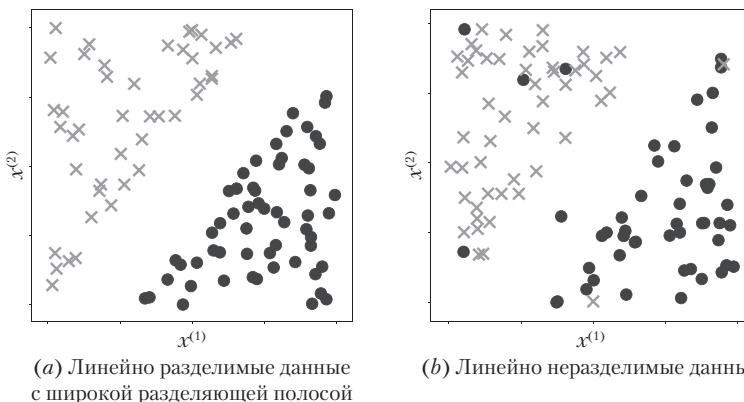
Последний шаг — убедиться, что максимизация  $\frac{1}{\|\mathbf{w}''\|^2}$  дает тот же результат, что и минимизация  $\frac{1}{2} \|\mathbf{w}''\|^2$ , и теорема 12.1 доказана.

#### 12.2.4. SVM с мягким зазором: геометрический подход

В случае, когда данные не являются линейно разделимыми, мы можем допустить, чтобы некоторые примеры попадали в разделяющую полосу или даже оказывались с неверной стороны от гиперплоскости, как показано на рис. 12.6.

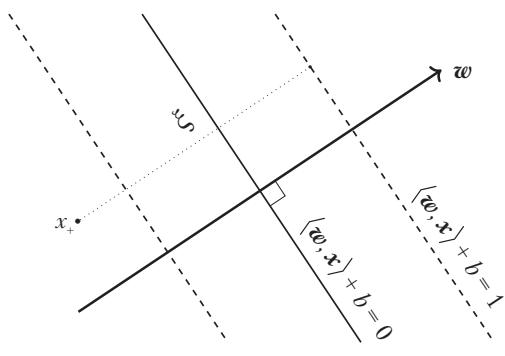
---

<sup>1</sup> Обратите внимание, что  $r > 0$ , поскольку мы предполагаем линейную сепарируемость, следовательно проблема с делением на  $r$  не возникает.



**Рис. 12.6.** (a) Линейно разделимые и (b) линейно неразделимые данные

Модель, допускающую ошибки классификации, называют *SVM с мягким зазором* (soft margin SVM). В этом разделе мы придем к соответствующей задаче оптимизации с помощью геометрических рассуждений. В разделе 12.2.5 мы выведем эквивалентную задачу оптимизации, пользуясь понятием функции потерь. Используя множители Лагранжа (раздел 7.2), мы поставим двойственную задачу оптимизации для SVM в разделе 12.3. Эта двойственная задача даст нам третий способ интерпретировать SVM: как проведение гиперплоскости, проходящей посередине между выпуклыми оболочками объектов положительного и отрицательного классов (раздел 12.3.2).



**Рис. 12.7.** Задача SVM с мягким зазором допускает, что объекты попадают в разделяющую полосу или даже оказываются с неверной стороны от гиперплоскости. Невязка  $\xi_n$ , если  $\mathbf{x}_+$  оказался с неверной стороны, измеряет расстояние от положительного примера  $\mathbf{x}_+$  до гиперплоскости положительного зазора  $\langle \mathbf{w}, \mathbf{x}_+ \rangle + b = 1$

Ключевая геометрическая идея — ввести для каждой пары «объект — метка»  $(\mathbf{x}_n, y_n)$  невязку  $\xi_n$ , позволяющую объекту попадать в разделяющую полосу или даже оказываться с неверной стороны от гиперплоскости (см. рис. 12.7). Значение  $\xi_n$  вычитается из зазора, при этом оно должно быть неотрицательным. Для поощрения правильной классификации мы прибавляем  $\xi_n$  к целевой функции

$$\min_{\boldsymbol{w}, b, \xi} \quad \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{n=1}^N \xi_n \quad (12.26a)$$

$$\text{при условиях } y_n (\langle \boldsymbol{w}, \mathbf{x}_n \rangle + b) \geq 1 - \xi_n \quad (12.26b)$$

$$\xi_n \geq 0, \quad (12.26c)$$

для  $n = 1, \dots, N$ . Данную задачу называют задачей оптимизации (12.18) уже не для SVM с жестким зазором, а для *SVM с мягким зазором*. Параметр  $C > 0$  отвечает за компромисс между шириной разделяющего зазора и общей суммой невязок. Его называют *параметром регуляризации*. Как мы увидим в следующем разделе, первое слагаемое  $\|\boldsymbol{w}\|^2$  в целевой функции (12.26a) выполняет роль *регуляризатора*, и во многих книгах по численной оптимизации именно его доминируют на параметр регуляризации (раздел 8.2.3). В этом разделе мы используем другую формулировку, в которой большие значения  $C$  соответствуют слабой регуляризации — давая больший вес поправкам и, соответственно, больший приоритет объектам, оказавшимся с неправильной стороны от гиперплоскости.

**ПРИМЕЧАНИЕ** В формулировке задачи SVM с мягким зазором (12.26a) регуляризован параметр  $\boldsymbol{w}$ , но не  $b$ , — регуляризатор  $b$  не содержит<sup>1</sup>. Этот нерегуляризованный член усложняет теоретический анализ (Steinwart and Christmann, 2008, глава 1) и ухудшает вычислительную эффективность (Fan et al., 2008). ◆

## 12.2.5. SVM с мягким зазором: подход с использованием функции потерь

Рассмотрим другой подход к постановке задачи SVM, следующий принципу минимизации эмпирического риска (раздел 8.2). Мы должны выбрать гиперплоскость из класса гипотез

$$f(\mathbf{x}) = \langle \boldsymbol{w}, \mathbf{x} \rangle + b. \quad (12.27)$$

В этом разделе мы увидим, что отступ соответствует регуляризатору. Остается выяснить, как выглядит *функция потерь*. В отличие от главы 9, где мы рассматривали задачу регрессии (в которой ответом является вещественное число), теперь мы рассматриваем задачу бинарной классификации (где предсказывается принадлежность объекта одному из классов  $\{+1, -1\}$ ). Таким образом, функция ошибок/потерь для каждой пары «объект — метка» должна подходить

---

<sup>1</sup> Существуют альтернативные параметризации этой регуляризации, поэтому задачу (12.26a) часто называют C-SVM.

для бинарной классификации. Например, квадратичная функция потерь (9.10b), используемая в задачах регрессии, для бинарной классификации не подходит.

**ПРИМЕЧАНИЕ** Идеальная функция потерь для бинарных меток подсчитывает количество несовпадений между предсказанием классификатора и настоящей меткой. Это значит, что для классификатора  $f$  и объекта  $\mathbf{x}_n$  мы сравниваем ответ классификатора  $f(\mathbf{x}_n)$  с меткой  $y_n$ . Функция потерь равна нулю, если они совпадают, и единице, если они не совпадают. Такую функцию обозначают как  $\mathbf{1}(f(\mathbf{x}_n) \neq y_n)$  и называют *нулевой потерей* (*zero-one loss*). К сожалению, при нулевой потере нахождение оптимальных параметров  $\mathbf{w}, b$  превращается в задачу комбинаторной оптимизации. Задачи комбинаторной оптимизации в общем случае гораздо сложнее, чем задачи непрерывной оптимизации, которые мы обсуждали в главе 7. ♦

Использование какой функции потерь приводит к методу опорных векторов? Рассмотрим разность между предсказанием классификатора  $f(\mathbf{x}_n)$  и меткой  $y_n$ . Функция потерь дает нам ошибку на обучающем множестве. Вывести формулу (12.26a) можно, используя *кусочно-линейную функцию потерь* (*hinge loss*):

$$\ell(t) = \max\{0, 1 - t\}, \quad \text{где } t = yf(\mathbf{x}) = y(\langle \mathbf{w}, \mathbf{x} \rangle + b). \quad (12.28)$$

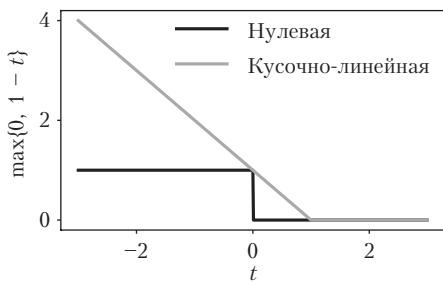
Если значение  $f(\mathbf{x})$  расположено с правильной стороны от гиперплоскости (совпадает с меткой  $y$ ) на расстоянии, большем 1, то  $t \geq 1$  и кусочно-линейная функция равна нулю. Если  $f(\mathbf{x})$  расположено с правильной стороны, но слишком близко к гиперплоскости ( $0 < t < 1$ ), объект  $\mathbf{x}$  лежит в разделяющей полосе и значение кусочно-линейной функции положительно. Если же объект оказался с неправильной стороны от гиперплоскости ( $t < 0$ ), функция потерь возвращает еще большее значение, которое притом линейно растет. Проще говоря, мы платим штраф, когда подходим к гиперплоскости слишком близко (даже при правильной классификации), при этом штраф линейно растет. Можно рассмотреть кусочно-линейную функцию как состоящую из двух линейных кусков

$$\ell(t) = \begin{cases} 0 & \text{если } t \geq 1 \\ 1 - t & \text{если } t < 1 \end{cases} \quad (12.29)$$

что изображено на рис. 12.8. Функция потерь, соответствующая SVM с жестким зазором (12.18), задается формулой

$$\ell(t) = \begin{cases} 0 & \text{если } t \geq 1 \\ \infty & \text{если } t < 1 \end{cases} \quad (12.30)$$

Можно истолковать эту формулу как запрет объектам попадать в разделяющую полосу.



**Рис. 12.8.** Кусочно-линейная функция потерь является выпуклой границей сверху для индикатора ошибки

Для заданной обучающей выборки  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$  мы минимизируем суммарные потери (эмпирический риск), применяя к целевой функции 2-регуляризацию (раздел 8.2.3). Использование кусочно-линейной функции (12.28) дает задачу безусловной (без ограничений) оптимизации

$$\min_{\mathbf{w}, b} \underbrace{\frac{1}{2} \|\mathbf{w}\|^2}_{\text{регуляризатор}} + C \underbrace{\sum_{n=1}^N \max\{0, 1 - y_n(\langle \mathbf{w}, \mathbf{x}_n \rangle + b)\}}_{\text{слагаемое ошибок}}. \quad (12.31)$$

Первое слагаемое в (12.31) называют *регуляризатором* (regularizer) (раздел 9.2.3), а второе — слагаемым ошибок (error term). Из раздела 12.2.4 мы знаем, что слагаемое  $\frac{1}{2} \|\mathbf{w}\|^2$  появляется непосредственно из значения отступа. Иными словами, максимизацию отступа можно рассматривать как *регуляризацию*.

Вообще говоря, задачу безусловной оптимизации (12.31) можно решить напрямую, используя методы (суб-)градиентного спуска, как описано в разделе 7.1. Чтобы понять, что (12.31) и (12.26а) эквивалентны, заметим, что функция потерь (12.28) состоит из двух слагаемых, как понятно из (12.29). Возьмем ее значение для пары «объект — метка» (12.28). Можно заменить минимизацию кусочно-линейной функции по  $t$  на минимизацию невязки  $\xi$  при условии двух ограничений. На языке формул

$$\min_t \max\{0, 1 - t\} \quad (12.32)$$

эквивалентно

$$\min_{\xi, t} \xi \quad (12.33)$$

при условиях  $\xi \geq 0, \xi \geq 1 - t$ .

Подставив это выражение в (12.31), получим в точности задачу SVM с мягким зазором (12.26а).

**ПРИМЕЧАНИЕ** Сравним нашу нынешнюю функцию потерь с функцией потерь для линейной регрессии в главе 9. Из раздела 9.2.1 мы знаем, что для нахождения оценки максимального правдоподобия обычно минимизируется взятый с противоположным знаком логарифм правдоподобия. Далее, так как слагаемое правдоподобия для линейной регрессии с гауссовым шумом будет гауссовым, взятый с противоположным знаком логарифм правдоподобия для каждого из примеров будет совпадать с квадратичной функцией ошибок. Именно квадратичная функция ошибок и будет функцией потерь, которую мы минимизируем при поиске решения максимального правдоподобия.



## 12.3. ДВОЙСТВЕННАЯ ЗАДАЧА SVM

Постановку задачи SVM в предыдущих разделах, в которой мы ищем  $\mathbf{w}$  и  $b$ , называют прямой задачей SVM. Вспомним, что входные значения  $\mathbf{x} \in \mathbb{R}^D$  имеют  $D$  признаков. Так как размерность вектора  $\mathbf{w}$  совпадает с размерностью  $\mathbf{x}$ , число параметров (размерность  $\mathbf{w}$ ) задачи оптимизации растет линейно с увеличением количества признаков.

Далее мы рассмотрим эквивалентную задачу оптимизации (используя понятие двойственности), не зависящую от количества признаков. В ней число параметров будет увеличиваться с увеличением размера обучающей выборки. С похожим приемом мы встречались в главе 19, когда записывали задачу в виде, не зависящем от количества признаков. Такой подход помогает в задачах, где признаков больше, чем примеров в обучающей выборке. Двойственная задача SVM хороша и тем, что в ней легко ввести ядро, — это мы увидим в конце данной главы. Слово «двойственный» часто встречается в математической литературе, сейчас мы имеем в виду двойственность для задач выпуклой оптимизации. Следующие главы, по сути, посвящены применению того, что мы узнали про двойственные задачи в разделе 7.2.

### 12.3.1. Двойственность и множители Лагранжа

Вернемся к прямой задаче SVM с мягким зазором (12.26a). Назовем присутствующие в ней переменные  $\mathbf{w}$ ,  $b$  и  $\xi$  прямыми переменными. Введем множитель Лагранжа  $\alpha_n \geq 0$ , соответствующий ограничению (12.26b) на правильность классификации, и множитель  $\gamma_n \geq 0$ , который соответствует условию неотрицательности невязок (12.26c)<sup>1</sup>. Лагранжиан выглядит как

---

<sup>1</sup> В главе 7 мы именовали множители Лагранжа как  $\lambda$ . В этом разделе мы следуем обозначениям, принятым в литературе по SVM, и используем  $\alpha$  и  $\gamma$ .

$$\mathcal{L}(\boldsymbol{w}, b, \xi, \alpha, \gamma) = \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{n=1}^N \xi_n - \underbrace{- \sum_{n=1}^N \alpha_n (y_n (\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) - 1 + \xi_n)}_{\text{ограничение (12.26b)}} - \underbrace{\sum_{n=1}^N \gamma_n \xi_n}_{\text{ограничение (12.26c)}}. \quad (12.34)$$

Дифференцируя лагранжиан (12.34) по трем прямым переменным  $\boldsymbol{w}$ ,  $b$  и  $\xi$  соответственно, получим

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \boldsymbol{w}^T - \sum_{n=1}^N \alpha_n y_n \boldsymbol{x}_n^T; \quad (12.35)$$

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{n=1}^N \alpha_n y_n; \quad (12.36)$$

$$\frac{\partial \mathcal{L}}{\partial \xi_n} = C - \alpha_n - \gamma_n. \quad (12.37)$$

Теперь найдем максимум лагранжиана, приравняв каждую из этих производных к нулю. Для (12.35) получим

$$\boldsymbol{w} = \sum_{n=1}^N \alpha_n y_n \boldsymbol{x}_n, \quad (12.38)$$

частный случай *теоремы о представителе* (Kimeldorf and Wahba, 1970). Уравнение (12.38) говорит нам, что оптимальным вектором весов будет линейная комбинация объектов  $\boldsymbol{x}_n$ . Из раздела 2.6.1 мы знаем, что это можно переформулировать так: решение задачи оптимизации принадлежит линейной оболочке обучающей выборки. Ограничение, полученное приравниванием к нулю выражения (12.36), говорит, что оптимальный вектор весов — аффинная комбинация объектов обучающей выборки. Теорема о представителе выполняется для очень общей задачи минимизации регуляризованного эмпирического риска (Hofmann et al., 2008; Argyriou and Dinuzzo, 2014). Теорема существует и в более общих формулировках (Schölkopf et al., 2001), необходимые и достаточные условия ее выполнения можно найти у Ю и др. (Yu et al, 2013)<sup>1</sup>.

**ПРИМЕЧАНИЕ** Теорема о представителе (12.38) объясняет название «метод опорных векторов». Для объектов  $\boldsymbol{x}_n$  соответствующие параметры  $\alpha_n = 0$  не вносят никакого вклада в решение  $\boldsymbol{w}$ . Другие объекты, для которых  $\alpha_n > 0$ , называют *опорными векторами* — на них «опирается» гиперплоскость. ◆

---

<sup>1</sup> Теорема о представителе — это фактически набор теорем, говорящих, что решение задачи минимизации эмпирического риска лежит в подпространстве (раздел 2.4.3), заданном обучающей выборкой.

Подставляя выражение для  $\mathbf{w}$  в лагранжиан (12.34), получим двойственный лагранжиан

$$\begin{aligned}\mathcal{D}(\xi, \alpha, \gamma) = & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^N y_i \alpha_i \left\langle \sum_{j=1}^N y_j \alpha_j, \mathbf{x}_i \right\rangle + \\ & + C \sum_{i=1}^N \xi_i - b \sum_{i=1}^N y_i \alpha_i + \sum_{i=1}^N \alpha_i - \sum_{i=1}^N \alpha_i \xi_i - \sum_{i=1}^N \gamma_i \xi_i.\end{aligned}\quad (12.39)$$

Заметим, что больше ни одно слагаемое не содержит прямой переменной  $\mathbf{w}$ . Приравняв (12.36) нулю, получим  $\sum_{n=1}^N y_n \alpha_n = 0$ . Таким образом, пропадает и слагаемое, содержащее  $b$ . Вспомним о симметричности и билинейности скалярного произведения (раздел 3.2). Поэтому первые два слагаемых в (12.39) можно объединить и получить

$$\mathcal{D}(\xi, \alpha, \gamma) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^N \alpha_i + \sum_{i=1}^N (C - \alpha_i - \gamma_i) \xi_i. \quad (12.40)$$

Последнее слагаемое в этом выражении объединяет все слагаемые, содержащие невязки  $\xi_i$ . Приравняв (12.37) к нулю, видим, что последнее слагаемое в (12.40) также равно нулю. Далее, из того же уравнения и неотрицательности множителей Лагранжа  $\gamma_i$  выводим, что  $\alpha_i \leq C$ . Мы получили двойственную задачу оптимизации для SVM, выраженную только через множители Лагранжа  $\alpha_i$ . Вспомним определение двойственности Лагранжа (определение 7.1) и то, что двойственная задача — задача максимизации. Можно взять целевую функцию с противоположным знаком и минимизировать ее, придав ей *двойственной задаче SVM*:

$$\begin{aligned}\min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^N \alpha_i; \\ \text{при условии} \quad & \sum_{i=1}^N y_i \alpha_i = 0; \\ & 0 \leq \alpha_i \leq C \quad \text{для всех } i = 1, \dots, N.\end{aligned}\quad (12.41)$$

Ограничение-равенство в (12.41) получено из приравнивания (12.36) к нулю. Ограничение  $\alpha_i \geq 0$  — условие, накладываемое на множители Лагранжа в неравенствах (раздел 7.2). Ограничение  $\alpha_i \leq C$  обсуждалось выше.

Ограничения-неравенства говорят нам, что вектор множителей Лагранжа  $\alpha = [\alpha_1, \dots, \alpha_N]^T \in \mathbb{R}^N$  лежит внутри квадрата со сторонами, параллельными осям координат (такие ограничения иногда называют квадратными, box constraints). Такие ограничения очень удобны для численных методов (Dostál, 2009, глава 5).

Найдя значения двойственных параметров  $\alpha$ , мы можем восстановить значения прямых параметров  $\mathbf{w}$  с помощью теоремы о представителе (12.38). Обозначим оптимальные прямые параметры  $\mathbf{w}^*$ . Однако остается вопрос, как найти пара-

метр  $b^*$ . Рассмотрим объект  $\mathbf{x}_n$ , лежащий точно на границе разделяющей полосы<sup>1</sup>, то есть  $\langle \mathbf{w}^*, \mathbf{x}_n \rangle + b = y_n$ . Мы знаем, что  $y_n$  равно +1 либо -1. Таким образом, неизвестным остается только  $b$ , которое можно вычислить как

$$b^* = y_n - \langle \mathbf{w}^*, \mathbf{x}_n \rangle. \quad (12.42)$$

**ПРИМЕЧАНИЕ** Вообще говоря, объектов, лежащих точно на границе разделяющей полосы, может и не существовать. В этом случае мы должны найти  $|y_n - \langle \mathbf{w}^*, \mathbf{x}_n \rangle|$  для всех опорных векторов и взять в качестве  $b^*$  медианное значение этого модуля разности. Доказательство можно найти на <http://fouryears.eu/2012/06/07/the-svm-bias-term-conspiracy/>. ♦

### 12.3.2. Двойственность и выпуклая оболочка

Альтернативный способ получить двойственную задачу SVM — геометрический. Рассмотрим множество примеров  $\mathbf{x}_n$  с одинаковыми метками. Мы хотим построить минимальное выпуклое множество, содержащее все эти примеры. Оно называется выпуклой оболочкой и изображено на рис. 12.9.

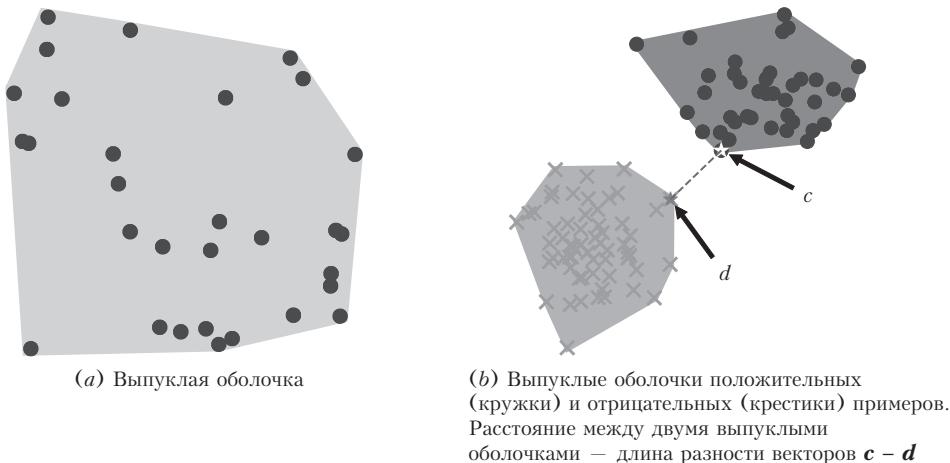
Познакомимся поближе с выпуклыми комбинациями точек. Рассмотрим две точки  $\mathbf{x}_1$  и  $\mathbf{x}_2$  и соответствующие неотрицательные веса  $\alpha_1, \alpha_2 \geq 0$ , такие что  $\alpha_1 + \alpha_2 = 1$ . Выражение  $\alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2$  задает любую точку на отрезке, соединяющем  $\mathbf{x}_1$  и  $\mathbf{x}_2$ . Посмотрим, что произойдет при добавлении третьей точки  $\mathbf{x}_3$  с весом  $\alpha_3 \geq 0$ , таким что  $\sum_{n=1}^3 \alpha_n = 1$ . Выпуклые комбинации трех точек  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$  задают двумерную фигуру. Ее *выпуклой оболочкой* будет треугольник, стороны которого соединяют каждую из пар точек. Если мы добавим больше точек, и их количество превысит размерность пространства, некоторые из точек окажутся внутри выпуклой оболочки, как мы видим на рис. 12.9(a).

В общем случае для получения выпуклой оболочки надо ввести неотрицательный вес  $\alpha_n \geq 0$  для каждого из объектов  $\mathbf{x}_n$ . Тогда выпуклую оболочку можно описать как множество

$$\text{conv}(\mathbf{X}) = \left\{ \sum_{n=1}^N \alpha_n \mathbf{x}_n \right\} \quad \text{с} \quad \sum_{n=1}^N \alpha_n = 1 \quad \text{и} \quad \alpha_n \geq 0 \quad (12.43)$$

для всех  $n = 1, \dots, N$ . Если два облака точек, соответствующие положительному и отрицательному классам, отделены друг от друга, выпуклые оболочки не пересекутся. Имея обучающую выборку  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ , мы построим две линейные оболочки, соответствующие положительному и отрицательному классам.

<sup>1</sup> Как выясняется, объекты, лежащие точно на границе разделяющей полосы, — это те же самые объекты, для которых двойственные параметры удовлетворяют строгим квадратным неравенствам  $0 < \alpha_i < C$ . Доказывается это через условия Каруша — Куна-Такера, см., например, Schölkopf and Smola (2002).



**Рис. 12.9.** Выпуклые оболочки. (a) Выпуклая оболочка точек, некоторые из которых лежат внутри ее границ. (b) Выпуклые оболочки множества положительных и отрицательных примеров

Возьмем точку  $\mathbf{c}$  в выпуклой оболочке множества положительных примеров, ближайшую к отрицательному классу. Точно так же возьмем точку  $\mathbf{d}$  в выпуклой оболочке множества отрицательных примеров, ближайшую к положительному классу, см. рис. 12.9(b). Обозначим разность  $\mathbf{d}$  и  $\mathbf{c}$  как

$$\mathbf{w} := \mathbf{c} - \mathbf{d}. \quad (12.44)$$

Требование, чтобы  $\mathbf{c}$  и  $\mathbf{d}$  были как можно ближе друг к другу, эквивалентно минимизации длины (нормы)  $\mathbf{w}$ , так что мы приходим к задаче оптимизации

$$\arg \min_{\mathbf{w}} \|\mathbf{w}\| = \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2. \quad (12.45)$$

Так как  $\mathbf{c}$  должна лежать в выпуклой оболочке положительных примеров, ее можно выразить как выпуклую комбинацию положительных примеров, то есть для некоторых неотрицательных коэффициентов  $\alpha_n^+$

$$\mathbf{c} = \sum_{n: y_n = +1} \alpha_n^+ \mathbf{x}_n. \quad (12.46)$$

В формуле (12.46) мы использовали обозначение  $n : y_n = +1$  для множества индексов  $n$ , таких что  $y_n = +1$ . Аналогично, для отрицательных примеров имеем

$$\mathbf{d} = \sum_{n: y_n = -1} \alpha_n^- \mathbf{x}_n. \quad (12.47)$$

Подставив (12.44), (12.46) и (12.47) в (12.45), получим целевую функцию

$$\min_{\alpha} \frac{1}{2} \left\| \sum_{n:y_n=+1} \alpha_n^+ \mathbf{x}_n - \sum_{n:y_n=-1} \alpha_n^- \mathbf{x}_n \right\|^2. \quad (12.48)$$

Обозначим через  $\alpha$  множество всех коэффициентов, то есть объединение  $\alpha^+$  и  $\alpha^-$ . Вспомним о требовании к элементам выпуклой оболочки — сумма коэффициентов равна единице:

$$\sum_{n:y_n=+1} \alpha_n^+ = 1 \quad \text{и} \quad \sum_{n:y_n=-1} \alpha_n^- = 1. \quad (12.49)$$

Приходим к ограничению

$$\sum_{n=1}^N y_n \alpha_n = 0. \quad (12.50)$$

Этот результат мы получаем, перемножив условия для каждого из классов:

$$\sum_{n=1}^N y_n \alpha_n = \sum_{n:y_n=+1} (+1)\alpha_n^+ + \sum_{n:y_n=-1} (-1)\alpha_n^- = \quad (12.51a)$$

$$= \sum_{n:y_n=+1} \alpha_n^+ - \sum_{n:y_n=-1} \alpha_n^- = 1 - 1 = 0. \quad (12.51b)$$

Целевая функция (12.48) и ограничения (12.50), вместе с предположением, что  $\alpha \geq 0$ , задают задачу выпуклой условной оптимизации. Можно показать, что это двойственная задача к SVM с жестким зазором (Bennett and Bredensteiner, 2000a).

**ПРИМЕЧАНИЕ** Чтобы получить задачу, двойственную к SVM с мягким зазором, мы рассматриваем выпуклую оболочку с ограничением сверху на коэффициенты  $\alpha$ . Ограничение на  $\alpha$  сжимает эту оболочку (Bennett and Bredensteiner, 2000b). ◆

## 12.4. ЯДРА

Рассмотрим формулировку задачи, двойственной к SVM (12.41). Заметим, что в целевой функции присутствуют только скалярные произведения объектов  $\mathbf{x}_i$  и  $\mathbf{x}_j$ , но не скалярные произведения объектов и параметров. Следовательно, если объект  $\mathbf{x}_i$  представлен набором признаков  $\phi(\mathbf{x}_i)$ , единственным изменением в двойственной задаче SVM будет замена скалярного произведения. Такая «модульность» задачи, когда выбор метода классификации (SVM) и представления признаков  $\phi(\mathbf{x})$  можно рассматривать отдельно, обеспечивает гибкость при решении. В данном разделе мы обсудим представление  $\phi(\mathbf{x})$  и вкратце опишем идею ядра, однако не станем вдаваться в технические детали.

Так как  $\phi(\mathbf{x})$  может быть нелинейной функцией, можно использовать SVM (который считается линейным классификатором) для построения классификаторов, не линейных по примерам  $\mathbf{x}_n$ . Это дает еще один путь работы с набором данных, который не является линейно разделимым (первый – использование мягкого отступа). На самом деле существует множество алгоритмов и статистических методов, обладающих тем же свойством, что и двойственная задача SVM: скалярные произведения вычисляются только между объектами обучающей выборки. Вместо того чтобы явно задать нелинейную карту признаков  $\phi(\cdot)$  и вычислить получившееся скалярное произведение  $\mathbf{x}_i$  и  $\mathbf{x}_j$ , мы определим функцию близости между  $\mathbf{x}_i$  и  $\mathbf{x}_j$ , обозначаемую  $k(\mathbf{x}_i, \mathbf{x}_j)$ . Некоторый класс функций близости – ядерные функции (ядра) – неявно задает нелинейную карту признаков  $\phi(\cdot)$ . Ядерные функции – это функции  $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ , для которых существует гильбертово пространство  $\mathcal{H}$  и карта признаков  $\phi : \mathcal{X} \rightarrow \mathcal{H}$ , такие что<sup>1</sup>

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}}. \quad (12.52)$$

С каждым ядром  $k$  связано единственное (воспроизводящее ядро) гильбертово пространство (Aronszajn, 1950; Berlinet and Thomas-Agnan, 2004). При этом  $\phi(\mathbf{x}) = k(\cdot, \mathbf{x})$  называют *канонической картой признаков*. Обобщение скалярного произведения до ядерной функции (12.52) известно как *ядерный трюк* (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004), прячущий нелинейное преобразование признаков.

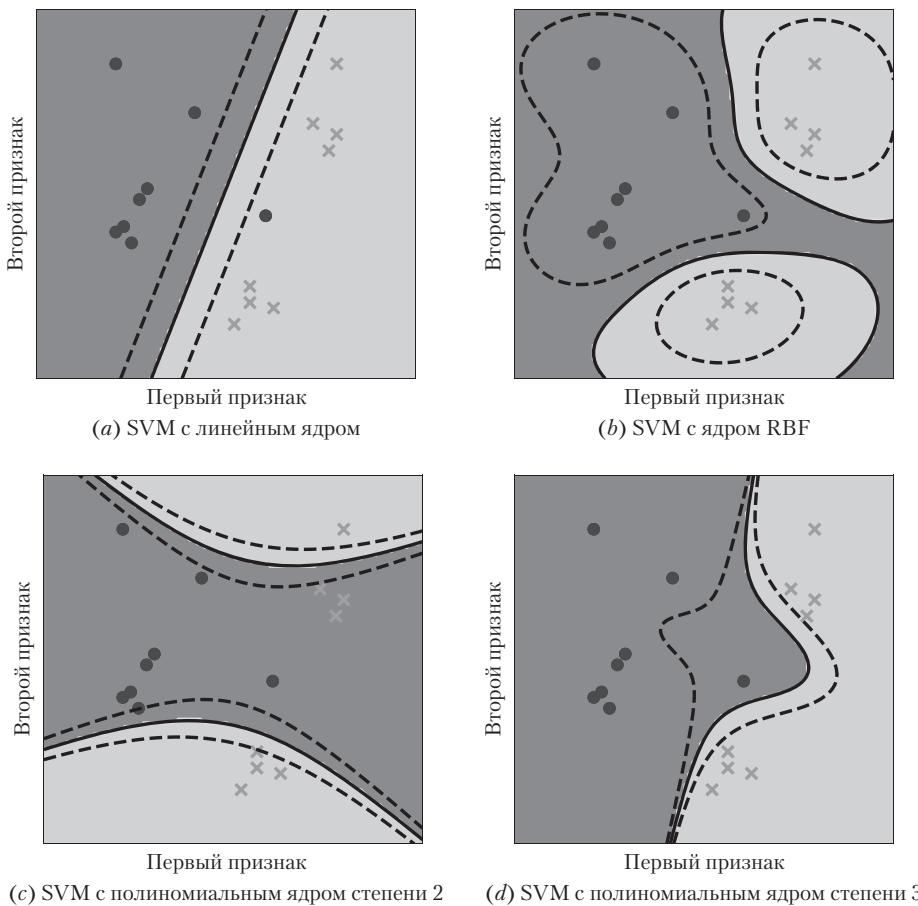
Матрица  $\mathbf{K} \in \mathbb{R}^{N \times N}$ , получающаяся применением скалярного произведения или ядерной функции  $k(\cdot, \cdot)$  к обучающей выборке, называется *матрицей Грама*, или *ядерной матрицей*. Ядра должны быть симметричными положительно полуопределенными функциями, так что любая ядерная матрица  $\mathbf{K}$  симметрична и положительно полуопределена (раздел 3.2.3):

$$\forall \mathbf{z} \in \mathbb{R}^N : \mathbf{z}^T \mathbf{K} \mathbf{z} \geq 0. \quad (12.53)$$

К широко применяемым ядерным функциям для многомерных вещественных данных  $\mathbf{x}_i \in \mathbb{R}^D$  относятся полиномиальное ядро, гауссова радиально-базисная функция (RBF) и рациональное квадратичное ядро (Schölkopf and Smola, 2002; Rasmussen and Williams, 2006). На рис. 12.10 показан эффект применения различных ядер к разделяющим гиперплоскостям на примере одного и того же набора данных. Заметим, что мы все еще ищем гиперплоскость, то есть классом гипотез по-прежнему являются линейные функции. Нелинейные поверхности появляются под действием ядерной функции.

---

<sup>1</sup> Область определения ядерной функции не обязана представлять собой  $\mathbb{R}^D$ .



**Рис. 12.10.** SVM с различными ядрами. Заметим, что хотя в решении границы нелинейна, внутри решается задача поиска разделяющей гиперплоскости (хотя и с использованием нелинейного ядра)

**ПРИМЕЧАНИЕ** К сожалению для начинающего изучать МО, слово «ядро» имеет много значений. В этой главе оно употребляется в смысле воспроизводящих ядер гильбертовых пространств (reproducing kernel Hilbert space, RKHS) (Aronszajn, 1950; Saitoh, 1988). Мы уже обсуждали понятие ядра в линейной алгебре (раздел 2.7.3), где оно является синонимом пространства решений однородной линейной системы. Третья область МО, где употребляется понятие ядра (сглаживающее ядро), — ядерные оценки плотности (раздел 11.5). ♦

Так как явное представление  $\varphi(\mathbf{x})$  математически эквивалентно ядерному представлению  $k(\mathbf{x}_i, \mathbf{x}_j)$ , на практике часто берут такую ядерную функцию, которую удобнее вычислять, чем скалярное произведение преобразованных признаков.

Рассмотрим, например, полиномиальное ядро (Schölkopf and Smola, 2002), где количество слагаемых в явном разложении растет чрезвычайно быстро (даже для многочленов небольшой степени), в то время как размерность входов велика. Ядерная функция требует числа умножений, равного размерности, что значительно упрощает вычисления. Другим примером может служить гауссова радиально-базисная функция (Schölkopf and Smola, 2002; Rasmussen and Williams, 2006), соответствующее которой пространство признаков бесконечномерно. В этом случае мы не можем напрямую работать с пространством признаков, но все еще в состоянии вычислять близость двух объектов с помощью ядра.

Другим полезным следствием ядерного трюка является отсутствие необходимости представлять исходные данные как многомерные векторы из вещественных чисел<sup>1</sup>. Вспомним, что скалярное произведение определено на выходных значениях функции  $\phi(\cdot)$ , которые не обязаны быть вещественными числами. Таким образом, функцию  $\phi(\cdot)$  и ядро  $k(\cdot, \cdot)$  можно определить на любых входных объектах — множествах, последовательностях, строках, графах, распределениях (Ben-Hur et al., 2008; Gärtner, 2008; Shi et al., 2009; Sriperumbudur et al., 2010; Vishwanathan et al., 2010).

## 12.5. ЧИСЛЕННОЕ РЕШЕНИЕ

Мы завершим разговор об SVM обсуждением того, как применять к задачам из данной главы концепции из главы 7. Рассмотрим два подхода к поиску оптимального решения задачи SVM. Сначала рассмотрим постановку задачи 8.2.2, использующую функцию потерь как задачу оптимизации без ограничений. Затем запишем ограниченные версии прямой и двойственной задач SVM в стандартной форме для задач квадратичного программирования 7.3.2.

Рассмотрим формулировку SVM (12.31), использующую функцию потерь. Это — задача безусловной оптимизации, однако кусочно-линейная функция (12.28) не дифференцируема. Поэтому надо применять субградиентный метод. Однако заметим, что не дифференцируема кусочно-линейная функция только в одной точке:  $t = 1$ . В ней градиентом можно считать множество значений между 0 и 1. Таким образом, субградиент  $g$  кусочно-линейной функции потерь задается формулой

$$g(t) = \begin{cases} -1 & t < 1 \\ [-1, 0] & t = 1. \\ 0 & t > 1 \end{cases} \quad (12.54)$$

---

<sup>1</sup> Ядро и его параметры часто выбирают с помощью вложенной кросс-валидации (раздел 8.6.1).

Вычислив субградиент, мы можем применить методы оптимизации из раздела 7.1. Как прямая, так и двойственная задачи SVM сводятся к выпуклой условной задаче квадратичного программирования. Заметим, что в прямой задаче (12.26a) переменные оптимизации имеют ту же размерность  $D$ , что и входные данные. В двойственной же задаче (12.41) они имеют размерность, равную размеру обучающей выборки  $N$ .

Чтобы записать прямую задачу SVM в стандартном виде задачи квадратичного программирования (7.45), будем использовать скалярное произведение (3.5). Перенесем в (12.26a) все переменные оптимизации в правую часть и приведем ограничение-неравенство к стандартному виду. Получаем задачу

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{n=1}^N \xi_n \quad (12.55)$$

при условиях

$$\begin{aligned} -y_n \mathbf{x}_n^\top \mathbf{w} - y_n b - \xi_n &\leq -1, \\ -\xi_n &\leq 0 \end{aligned}$$

$n = 1, \dots, N$ . Объединяя переменные  $\mathbf{w}, b, \mathbf{x}_n$  в один вектор и группируя слагаемые, получаем матричный вид задачи SVM с мягким зазором:

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2} \begin{bmatrix} \mathbf{w} \\ b \\ \xi \end{bmatrix}^\top \begin{bmatrix} \mathbf{I}_D & \mathbf{0}_{D, N+1} \\ \mathbf{0}_{N+1, D} & \mathbf{0}_{N+1, N+1} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ b \\ \xi \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{D+1, 1} & C \mathbf{1}_{N, 1} \end{bmatrix}^\top \begin{bmatrix} \mathbf{w} \\ b \\ \xi \end{bmatrix} \quad (12.56)$$

при условиях

$$\begin{bmatrix} -\mathbf{YX}^\top & -\mathbf{y} & -\mathbf{I}_N \\ \mathbf{0}_{N, D+1} & & -\mathbf{I}_N \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ b \\ \xi \end{bmatrix} \leq \begin{bmatrix} -\mathbf{1}_{N, 1} \\ \mathbf{0}_{N, 1} \end{bmatrix}.$$

В этой задаче минимизация происходит по переменным  $[\mathbf{w}^\top, b, \xi^\top]^\top \in \mathbb{R}^{D+1+N}$ . Мы используем обозначения  $\mathbf{I}_m$  для единичной матрицы размера  $m \times m$ ,  $\mathbf{0}_{m,n}$  для матрицы из нулей размера  $m \times n$  и  $\mathbf{1}_{m,n}$  для матрицы из единиц размера  $m \times n$ . Кроме того, обозначим через  $\mathbf{y}$  вектор меток  $[y_1, \dots, y_N]^\top$ ,  $\mathbf{Y} = \text{diag}(\mathbf{y})$  будет матрицей размера  $N \times N$ , у которой на диагонали расположены элементы  $\mathbf{y}$ , а  $\mathbf{X} \in \mathbb{R}^{N \times D}$  — матрица, содержащая все примеры из обучающей выборки.

Можно аналогичным образом сгруппировать слагаемые и в двойственной задаче SVM (12.41). Чтобы записать ее в стандартном виде, сначала введем матрицу  $\mathbf{K}$  с элементами  $K_{ij} = k(x_i, x_j)$ . Если  $x_i$  записан в виде вектора признаков, то мы определяем  $K_{ij} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ . Для удобства обозначений введем также матрицу  $\mathbf{Y} = \text{diag}(\mathbf{y})$ , у которой на диагонали стоят метки классов, а вне диагонали — нули. Двойственная задача SVM запишется как

$$\min_{\alpha} \frac{1}{2} \alpha^T Y K Y \alpha - \mathbf{1}_{N,1}^T \alpha$$

при условиях

$$\begin{bmatrix} \mathbf{y}^T \\ -\mathbf{y}^T \\ -\mathbf{I}_N \\ \mathbf{I}_N \end{bmatrix} \alpha \leq \begin{bmatrix} \mathbf{0}_{N+2,1} \\ C \mathbf{1}_{N,1} \end{bmatrix}. \quad (12.57)$$

**ПРИМЕЧАНИЕ** В разделах 7.3.1 и 7.3.2 мы договорились, что стандартной формой ограничений являются ограничения-неравенства. Выразим ограничение-равенство в двойственной задаче SVM как два неравенства:

$$A\mathbf{x} = \mathbf{b} \text{ заменяется на } A\mathbf{x} \leq \mathbf{b} \text{ и } A\mathbf{x} \geq \mathbf{b}. \quad (12.58)$$

Некоторые программные реализации выпуклой оптимизации допускают и наличие ограничений-равенств. ♦

Так как существует множество способов рассматривать задачу SVM, существует и множество подходов к решению полученной задачи оптимизации. Подход, изложенный здесь, — представление задачи SVM в стандартном виде задач выпуклой оптимизации — нечасто используется на практике. Две самые популярные реализации поиска решений SVM принадлежат Чангу и Линю (Chang and Lin, 2011) (программа с открытым исходным кодом) и Йоахимсу (Joachims, 1999). Так как SVM сводится к четко поставленной задаче оптимизации, можно применять различные методы численной оптимизации (Nocedal and Wright, 2006; Shawe-Taylor and Sun, 2011).

## 12.6. ДЛЯ ДАЛЬНЕЙШЕГО ЧТЕНИЯ

Метод опорных векторов — один из множества методов бинарной классификации. К другим методам относятся перцептроны, логистическая регрессия, дискриминант Фишера, метод ближайшего соседа, наивный Байес и случайный лес (Bishop, 2006; Murphy, 2012). Краткий справочник по использованию опорных векторов и ядер для дискретных последовательностей можно найти у Бен-Гура и др. (Ben-Hur et al., 2008). Развитие метода опорных векторов связано с задачей минимизации эмпирического риска, которую мы обсуждали в разделе 8.2. Поэтому SVM обладает свойствами, доказанными теоретически строго (Vapnik, 2000; Steinwart and Christmann, 2008). В книге по ядерным методам (Schölkopf and Smola, 2002) содержится много материала о методе опорных векторов и оптимизации. Более развернутое изложение ядерных методов (Shawe-Taylor and Cristianini, 2004) также содержит немало линейно-алгебраических подходов к задачам машинного обучения.

Альтернативный способ вывести двойственную задачу SVM основан на идее преобразования Лежандра – Фенхеля (раздел 7.3.3). При этом каждое слагаемое в безусловной формулировке задачи SVM (12.31) рассматривается отдельно и вычисляется его выпуклое сопряжение (Rifkin and Lippert, 2007). Читателей, которым интересен подход к SVM с точки зрения функционального анализа (а также методов регуляризации), отсылаем к книге Вахбы (Wahba, 1990). Изучение теории ядерных функций (Aronszajn, 1950; Schwartz, 1964; Saitoh, 1988; Manton and Amblard, 2015) требует знаний о линейных операторах (Akhiezer and Glazman, 1993). Понятие ядер обобщается на банаховы пространства (Zhang et al., 2009) и крейновы пространства (Ong et al., 2004; Loosli et al., 2016).

Заметим, что у кусочно-линейной функции существует три эквивалентных представления, как видно из формул (12.28) и (12.29) и условной задачи оптимизации (12.33). Вид (12.28) часто используется для сравнения функции потерь в SVM с другими функциями потерь (Steinwart, 2007). Кусочно-линейный вид (12.29) удобен для вычисления субградиентов. Вид (12.33), как можно увидеть из раздела 12.5, позволяет использовать методы выпуклого квадратичного программирования (раздел 7.3.2).

Бинарная классификация — очень распространенная и хорошо изученная задача машинного обучения, также известная под названиями задачи разделения и задачи принятия решений. Ответы бинарного классификатора могут быть трех типов. Первый тип — значение линейной функции, которое может быть любым вещественным числом. Эти числа можно использовать для ранжирования объектов, а классификация будет состоять в выборе порогового значения на объектах обучающей выборки (Shawe-Taylor and Cristianini, 2004). Во втором случае ответ бинарного классификатора пропущен через некоторую нелинейную функцию, которая загоняет ответ в заданный интервал, например  $[0, 1]$ . Типичным примером такой нелинейной функции является сигмоида (Bishop, 2006). Порой применение нелинейной функции дает нам вероятность (Gneiting and Raftery, 2007; Reid and Williamson, 2011), тогда мы говорим о задаче оценки вероятности класса. Третий тип ответа бинарного классификатора (чаще всего, говоря о задаче классификации, имеют в виду именно его) — просто итоговая метка класса  $\{+1, -1\}$ .

SVM относится к бинарным классификаторам, не имеющим естественной вероятностной интерпретации. Однако существует несколько методов, переводящих значения линейной функции в оценки вероятности классов ( $P(Y = 1 | X = \mathbf{x})$ ), включающих дополнительную калибровку (Platt, 2000; Zadrozny and Elkan, 2001; Lin et al., 2007). С точки зрения обучения, существуют родственные вероятностные методы. В конце раздела 12.2.5 мы упоминали о связи между функцией потерь и правдоподобием (сравните также с разделами 8.2 и 8.3). Подход максимального правдоподобия с удачным преобразованием во время

обучения называется логистической регрессией и принадлежит к классу обобщенных линейных моделей. Подробнее о логистической регрессии с этой точки зрения можно прочитать у Агрести (Agresti, 2002, глава 5) и у Маккаллаха и Нелдера (McCullagh and Nelder, 1989, глава 4). Можно принять байесовскую точку зрения на классификацию и оценить апостериорное распределение ответов с помощью байесовской линейной регрессии. Байесовский подход также включает оценку априорного распределения, для чего требуется выбирать, например, сопряженную функцию правдоподобия (раздел 6.6.1). Можно также рассматривать скрытые функции как априорные распределения и прийти к классификации на основе гауссовых процессов (Rasmussen and Williams, 2006, глава 3).

# Библиография

- Abel, Niels H. 1826. *Démonstration de l'Impossibilité de la Résolution Algébrique des Équations Générales qui Passent le Quatrième Degré*. Grøndahl & Søn.
- Adhikari, Ani, and DeNero, John. 2018. Computational and Inferential Thinking: The Foundations of Data Science. Gitbooks.
- Agarwal, Arvind, and Daumé III, Hal. 2010. A Geometric View of Conjugate Priors. *Machine Learning*, 81(1), 99–113.
- Agresti, A. 2002. Categorical Data Analysis. Wiley.
- Akaike, Hirotugu. 1974. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19(6), 716–723.
- Akhiezer, Naum I., and Glazman, Izrail M. 1993. Theory of Linear Operators in Hilbert Space. Dover Publications.
- Alpaydin, Ethem. 2010. Introduction to Machine Learning. MIT Press.
- Amari, Shun-ichi. 2016. Information Geometry and Its Applications. Springer.
- Argyriou, Andreas, and Dinuzzo, Francesco. 2014. A Unifying View of Representer Theorems. In: Proceedings of the International Conference on Machine Learning.
- Aronszajn, Nachman. 1950. Theory of Reproducing Kernels. *Transactions of the American Mathematical Society*, 68, 337–404.
- Axler, Sheldon. 2015. Linear Algebra Done Right. Springer.
- Bakir, Gökhan, Hofmann, Thomas, Schölkopf, Bernhard, Smola, Alexander J., Taskar, Ben, and Vishwanathan, S. V. N. (eds). 2007. Predicting Structured Data. MIT Press.
- Barber, David. 2012. Bayesian Reasoning and Machine Learning. Cambridge University Press.
- Barndorff-Nielsen, Ole. 2014. Information and Exponential Families: In Statistical Theory. Wiley.
- Bartholomew, David, Knott, Martin, and Moustaki, Irini. 2011. Latent Variable Models and Factor Analysis: A Unified Approach. Wiley.

- Baydin, Atilim G., Pearlmutter, Barak A., Radul, Alexey A., and Siskind, Jeffrey M. 2018. Automatic Differentiation in Machine Learning: A Survey. *Journal of Machine Learning Research*, 18, 1–43.
- Beck, Amir, and Teboulle, Marc. 2003. Mirror Descent and Nonlinear Projected Subgradient Methods for Convex Optimization. *Operations Research Letters*, 31(3), 167–175.
- Belabbas, Mohamed-Ali, and Wolfe, Patrick J. 2009. Spectral Methods in Machine Learning and New Strategies for Very Large Datasets. *Proceedings of the National Academy of Sciences*, 0810600105.
- Belkin, Mikhail, and Niyogi, Partha. 2003. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Computation*, 15(6), 1373–1396.
- Ben-Hur, Asa, Ong, Cheng Soon, Sonnenburg, Sören, Schölkopf, Bernhard, and Rätsch, Gunnar. 2008. Support Vector Machines and Kernels for Computational Biology. *PLoS Computational Biology*, 4(10), e1000173.
- Bennett, Kristin P., and Bredensteiner, Erin J. 2000a. Duality and Geometry in SVM Classifiers. In: *Proceedings of the International Conference on Machine Learning*.
- Bennett, Kristin P., and Bredensteiner, Erin J. 2000b. Geometry in Learning. Pages 132–145 in *Geometry at Work*. Mathematical Association of America.
- Berlinet, Alain, and Thomas-Agnan, Christine. 2004. Reproducing Kernel Hilbert Spaces in Probability and Statistics. Springer.
- Bertsekas, Dimitri P. 1999. *Nonlinear Programming*. Athena Scientific. Bertsekas, Dimitri P. 2009. *Convex Optimization Theory*. Athena Scientific.
- Bickel, Peter J., and Doksum, Kjell. 2006. *Mathematical Statistics, Basic Ideas and Selected Topics*. Vol. 1. Prentice Hall.
- Bickson, Danny, Dolev, Danny, Shental, Ori, Siegel, Paul H., and Wolf, Jack K. 2007. Linear Detection via Belief Propagation. In: *Proceedings of the Annual Allerton Conference on Communication, Control, and Computing*.
- Billingsley, Patrick. 1995. *Probability and Measure*. Wiley.
- Bishop, Christopher M. 1995. *Neural Networks for Pattern Recognition*. Clarendon Press.
- Bishop, Christopher M. 1999. Bayesian PCA. In: *Advances in Neural Information Processing Systems*.
- Bishop, Christopher M. 2006. *Pattern Recognition and Machine Learning*. Springer.
- Blei, David M., Kucukelbir, Alp, and McAuliffe, Jon D. 2017. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518), 859–877.

- Blum, Arvind, and Hardt, Moritz. 2015. The Ladder: A Reliable Leaderboard for Machine Learning Competitions. In: International Conference on Machine Learning.
- Bonnans, J. Frédéric, Gilbert, J. Charles, Lemaréchal, Claude, and Sagastizábal, Claudia A. 2006. Numerical Optimization: Theoretical and Practical Aspects. Springer.
- Borwein, Jonathan M., and Lewis, Adrian S. 2006. Convex Analysis and Nonlinear Optimization. 2nd edn. Canadian Mathematical Society.
- Bottou, Léon. 1998. Online Algorithms and Stochastic Approximations. Pages 9–42 in Online Learning and Neural Networks. Cambridge University Press.
- Bottou, Léon, Curtis, Frank E., and Nocedal, Jorge. 2018. Optimization Methods for Large-Scale Machine Learning. SIAM Review, 60(2), 223–311.
- Boucheron, Stephane, Lugosi, Gabor, and Massart, Pascal. 2013. Concentration Inequalities: A Nonasymptotic Theory of Independence. Oxford University Press.
- Boyd, Stephen, and Vandenberghe, Lieven. 2004. Convex Optimization. Cambridge University Press.
- Boyd, Stephen, and Vandenberghe, Lieven. 2018. Introduction to Applied Linear Algebra. Cambridge University Press.
- Brochu, Eric, Cora, Vlad M., and de Freitas, Nando. 2009. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. Tech. rept. TR-2009-023. Department of Computer Science, University of British Columbia.
- Brooks, Steve, Gelman, Andrew, Jones, Galin L., and Meng, Xiao-Li (eds). 2011. Handbook of Markov Chain Monte Carlo. Chapman and Hall/CRC.
- Brown, Lawrence D. 1986. Fundamentals of Statistical Exponential Families: With Applications in Statistical Decision Theory. Institute of Mathematical Statistics.
- Bryson, Arthur E. 1961. A Gradient Method for Optimizing Multi-Stage Allocation Processes. In: Proceedings of the Harvard University Symposium on Digital Computers and Their Applications.
- Bubeck, Sébastien. 2015. Convex Optimization: Algorithms and Complexity. Foundations and Trends in Machine Learning, 8(3-4), 231–357.
- Bühlmann, Peter, and Van De Geer, Sara. 2011. Statistics for High-Dimensional Data. Springer.
- Burges, Christopher. 2010. Dimension Reduction: A Guided Tour. Foundations and Trends in Machine Learning, 2(4), 275–365.

- Carroll, J Douglas, and Chang, Jih-Jie. 1970. Analysis of Individual Differences in Multidimensional Scaling via an N -Way Generalization of “Eckart–Young” Decomposition. *Psychometrika*, 35(3), 283–319.
- Casella, George, and Berger, Roger L. 2002. *Statistical Inference*. Duxbury.
- Çinlar, Erhan. 2011. *Probability and Stochastics*. Springer.
- Chang, Chih-Chung, and Lin, Chih-Jen. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2, 27:1–27:27. Software available at [www.csie.ntu.edu.tw/cjlin/libsvm](http://www.csie.ntu.edu.tw/cjlin/libsvm).
- Cheeseman, Peter. 1985. In Defense of Probability. In: *Proceedings of the International Joint Conference on Artificial Intelligence*.
- Chollet, Francois, and Allaire, J. J. 2018. *Deep Learning* with R. Manning Publications.
- Codd, Edgar F. 1990. *The Relational Model for Database Management*. Addison-Wesley Longman Publishing.
- Cunningham, John P., and Ghahramani, Zoubin. 2015. Linear Dimensionality Reduction: Survey, Insights, and Generalizations. *Journal of Machine Learning Research*, 16, 2859–2900.
- Datta, Biswa N. 2010. *Numerical Linear Algebra and Applications*. SIAM.
- Davidson, Anthony C., and Hinkley, David V. 1997. *Bootstrap Methods and Their Application*. Cambridge University Press.
- Dean, Jeffrey, Corrado, Greg S., Monga, Rajat, et al. 2012. Large Scale Distributed Deep Networks. In: *Advances in Neural Information Processing Systems*.
- Deisenroth, Marc P., and Mohamed, Shakir. 2012. Expectation Propagation in Gaussian Process Dynamical Systems. In: *Advances in Neural Information Processing Systems*.
- Deisenroth, Marc P., and Ohlsson, Henrik. 2011. A General Perspective on Gaussian Filtering and Smoothing: Explaining Current and Deriving New Algorithms. In: *Proceedings of the American Control Conference*.
- Deisenroth, Marc P., Fox, Dieter, and Rasmussen, Carl E. 2015. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2), 408–423.
- Dempster, Arthur P., Laird, Nan M., and Rubin, Donald B. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, 39(1), 1–38.
- Deng, Li, Seltzer, Michael L., Yu, Dong, Acero, Alex, Mohamed, Abdel-rahman, and Hinton, Geoffrey E. 2010. Binary Coding of Speech Spectrograms Using a Deep Auto-Encoder. *Proceedings of Interspeech*.
- Devroye, Luc. 1986. *Non-Uniform Random Variate Generation*. Springer.

- Donoho, David L., and Grimes, Carrie. 2003. Hessian Eigenmaps: Locally Linear Embedding Techniques for High-Dimensional Data. *Proceedings of the National Academy of Sciences*, 100(10), 5591–5596.
- Dostál, Zdeněk. 2009. Optimal Quadratic Programming Algorithms: With Applications to Variational Inequalities. Springer.
- Douven, Igor. 2017. Abduction. In: *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University. <https://plato.stanford.edu/cgi-bin/encyclopedia/archinfo.cgi?entry=abduction&archive=sum2017>.
- Downey, Allen B. 2014. Think Stats: Exploratory Data Analysis. 2nd edn. O'Reilly Media.
- Dreyfus, Stuart. 1962. The Numerical Solution of Variational Problems. *Journal of Mathematical Analysis and Applications*, 5(1), 30–45.
- Drumm, Volker, and Weil, Wolfgang. 2001. Lineare Algebra und Analytische Geometrie. Lecture Notes, Universität Karlsruhe (TH).
- Dudley, Richard M. 2002. Real Analysis and Probability. Cambridge University Press.
- Eaton, Morris L. 2007. Multivariate Statistics: A Vector Space Approach. Institute of Mathematical Statistics Lecture Notes.
- Eckart, Carl, and Young, Gale. 1936. The Approximation of One Matrix by Another of Lower Rank. *Psychometrika*, 1(3), 211–218.
- Efron, Bradley, and Hastie, Trevor. 2016. Computer Age Statistical Inference: Algorithms, Evidence and Data Science. Cambridge University Press.
- Efron, Bradley, and Tibshirani, Robert J. 1993. An Introduction to the Bootstrap. Chapman and Hall/CRC.
- Elliott, Conal. 2009. Beautiful Differentiation. In: International Conference on Functional Programming.
- Evgeniou, Theodoros, Pontil, Massimiliano, and Poggio, Tomaso. 2000. Statistical Learning Theory: A Primer. *International Journal of Computer Vision*, 38(1), 9–13.
- Fan, Rong-En, Chang, Kai-Wei, Hsieh, Cho-Jui, Wang, Xiang-Rui, and Lin, Chih-Jen. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9, 1871–1874.
- Gal, Yarin, van der Wilk, Mark, and Rasmussen, Carl E. 2014. Distributed Variational Inference in Sparse Gaussian Process Regression and Latent Variable Models. In: *Advances in Neural Information Processing Systems*.
- Gärtner, Thomas. 2008. Kernels for Structured Data. World Scientific.
- Gavish, Matan, and Donoho, David L. 2014. The Optimal Hard Threshold for Singular Values is  $4\sqrt{3}$ . *IEEE Transactions on Information Theory*, 60(8), 5040–5053.

- Gelman, Andrew, Carlin, John B., Stern, Hal S., and Rubin, Donald B. 2004. Bayesian Data Analysis. Chapman & Hall/CRC.
- Gentle, James E. 2004. Random Number Generation and Monte Carlo Methods. Springer.
- Ghahramani, Zoubin. 2015. Probabilistic Machine Learning and Artificial Intelligence. *Nature*, 521, 452–459.
- Ghahramani, Zoubin, and Roweis, Sam T. 1999. Learning Nonlinear Dynamical Systems Using an EM Algorithm. In: Advances in Neural Information Processing Systems.
- Gilks, Walter R., Richardson, Sylvia, and Spiegelhalter, David J. 1996. Markov Chain Monte Carlo in Practice. Chapman and Hall/CRC.
- Gneiting, Tilmann, and Raftery, Adrian E. 2007. Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American Statistical Association*, 102(477), 359–378.
- Goh, Gabriel. 2017. Why Momentum Really Works. Distill.
- Gohberg, Israel, Goldberg, Seymour, and Krupnik, Nahum. 2012. Traces and Determinants of Linear Operators. Birkhäuser.
- Golan, Jonathan S. 2007. The Linear Algebra a Beginning Graduate Student Ought to Know. Springer.
- Golub, Gene H., and Van Loan, Charles F. 2012. Matrix Computations. JHU Press.
- Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron. 2016. Deep Learning. MIT Press.
- Graepel, Thore, Candela, Joaquin Quiñonero-Candela, Borchert, Thomas, and Herbrich, Ralf. 2010. Web-Scale Bayesian Click-through Rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine. In: Proceedings of the International Conference on Machine Learning.
- Griewank, Andreas, and Walther, Andrea. 2003. Introduction to Automatic Differentiation. In: Proceedings in Applied Mathematics and Mechanics.
- Griewank, Andreas, and Walther, Andrea. 2008. Evaluating Derivatives, Principles and Techniques of Algorithmic Differentiation. SIAM.
- Grimmett, Geoffrey R., and Welsh, Dominic. 2014. Probability: An Introduction. Oxford University Press.
- Grinstead, Charles M., and Snell, J. Laurie. 1997. Introduction to Probability. American Mathematical Society.
- Hacking, Ian. 2001. Probability and Inductive Logic. Cambridge University Press.
- Hall, Peter. 1992. The Bootstrap and Edgeworth Expansion. Springer.
- Hallin, Marc, Paindaveine, Davy, and Šiman, Miroslav. 2010. Multivariate Quantiles and Multiple-Output Regression Quantiles: From  $\ell_1$  Optimization to Halfspace Depth. *Annals of Statistics*, 38, 635–669.

- Hasselblatt, Boris, and Katok, Anatole. 2003. *A First Course in Dynamics with a Panorama of Recent Developments*. Cambridge University Press.
- Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome. 2001. *The Elements of Statistical Learning – Data Mining, Inference, and Prediction*. Springer.
- Hausman, Karol, Springenberg, Jost T., Wang, Ziyu, Heess, Nicolas, and Riedmiller, Martin. 2018. Learning an Embedding Space for Transferable Robot Skills. In: *Proceedings of the International Conference on Learning Representations*.
- Hazan, Elad. 2015. Introduction to Online Convex Optimization. *Foundations and Trends in Optimization*, 2(3–4), 157–325.
- Hensman, James, Fusi, Nicolò, and Lawrence, Neil D. 2013. Gaussian Processes for Big Data. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.
- Herbrich, Ralf, Minka, Tom, and Graepel, Thore. 2007. TrueSkill(TM): A Bayesian Skill Rating System. In: *Advances in Neural Information Processing Systems*.
- Hiriart-Urruty, Jean-Baptiste, and Lemaréchal, Claude. 2001. *Fundamentals of Convex Analysis*. Springer.
- Hoffman, Matthew D., Blei, David M., and Bach, Francis. 2010. Online Learning for Latent Dirichlet Allocation. In: *Advances in Neural Information Processing Systems*.
- Hoffman, Matthew D., Blei, David M., Wang, Chong, and Paisley, John. 2013. Stochastic Variational Inference. *Journal of Machine Learning Research*, 14(1), 1303–1347.
- Hofmann, Thomas, Schölkopf, Bernhard, and Smola, Alexander J. 2008. Kernel Methods in Machine Learning. *Annals of Statistics*, 36(3), 1171–1220.
- Hogben, Leslie. 2013. *Handbook of Linear Algebra*. Chapman and Hall/CRC.
- Horn, Roger A., and Johnson, Charles R. 2013. *Matrix Analysis*. Cambridge University Press.
- Hotelling, Harold. 1933. Analysis of a Complex of Statistical Variables into Principal Components. *Journal of Educational Psychology*, 24, 417–441.
- Hyvärinen, Aapo, Oja, Erkki, and Karhunen, Juha. 2001. *Independent Component Analysis*. Wiley.
- Imbens, Guido W., and Rubin, Donald B. 2015. *Causal Inference for Statistics, Social and Biomedical Sciences*. Cambridge University Press.
- Jacob, Jean, and Protter, Philip. 2004. *Probability Essentials*. Springer.
- Jaynes, Edwin T. 2003. *Probability Theory: The Logic of Science*. Cambridge University Press.

- Jefferys, William H., and Berger, James O. 1992. Ockham's Razor and Bayesian Analysis. *American Scientist*, 80, 64–72.
- Jeffreys, Harold. 1961. *Theory of Probability*. Oxford University Press.
- Jimenez Rezende, Danilo, and Mohamed, Shakir. 2015. Variational Inference with Normalizing Flows. In: *Proceedings of the International Conference on Machine Learning*.
- Jimenez Rezende, Danilo, Mohamed, Shakir, and Wierstra, Daan. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In: *Proceedings of the International Conference on Machine Learning*.
- Joachims, Thorsten. 1999. Making Large-Scale SVM Learning Practical. Pages 169–184 in: *Advances in Kernel Methods – Support Vector Learning*. MIT Press.
- Jordan, Michael I., Ghahramani, Zoubin, Jaakkola, Tommi S., and Saul, Lawrence K. 1999. An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37, 183–233.
- Julier, Simon J., and Uhlmann, Jeffrey K. 1997. A New Extension of the Kalman Filter to Nonlinear Systems. In: *Proceedings of AeroSense Symposium on Aerospace/Defense Sensing, Simulation and Controls*.
- Kaiser, Marcus, and Hilgetag, Claus C. 2006. Nonoptimal Component Placement, but Short Processing Paths, Due to Long-Distance Projections in Neural Systems. *PLoS Computational Biology*, 2(7), e95.
- Kalman, Dan. 1996. A Singularly Valuable Decomposition: The SVD of a Matrix. *College Mathematics Journal*, 27(1), 2–23.
- Kalman, Rudolf E. 1960. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME – Journal of Basic Engineering*, 82(Series D), 35–45.
- Kamthe, Sanket, and Deisenroth, Marc P. 2018. Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics*.
- Katz, Victor J. 2004. *A History of Mathematics*. Pearson/Addison-Wesley.
- Kelley, Henry J. 1960. Gradient Theory of Optimal Flight Paths. *Ars Journal*, 30(10), 947–954.
- Kimeldorf, George S., and Wahba, Grace. 1970. A Correspondence between Bayesian Estimation on Stochastic Processes and Smoothing by Splines. *Annals of Mathematical Statistics*, 41(2), 495–502.
- Kingma, Diederik P., and Ba, Jimmy. 2014. Adam: A Method for Stochastic Optimization. In: *Proceedings of the International Conference on Learning Representations*.

- Kingma, Diederik P., and Welling, Max. 2014. Auto-Encoding Variational Bayes. In: Proceedings of the International Conference on Learning Representations.
- Kittler, Josef, and Föglein, Janos. 1984. Contextual Classification of Multispectral Pixel Data. *Image and Vision Computing*, 2(1), 13–29.
- Kolda, Tamara G., and Bader, Brett W. 2009. Tensor Decompositions and Applications. *SIAM Review*, 51(3), 455–500.
- Koller, Daphne, and Friedman, Nir. 2009. Probabilistic Graphical Models. MIT Press.
- Kong, Linglong, and Mizera, Ivan. 2012. Quantile Tomography: Using Quantiles with Multivariate Data. *Statistica Sinica*, 22, 1598–1610.
- Lang, Serge. 1987. Linear Algebra. Springer.
- Lawrence, Neil D. 2005. Probabilistic Non-Linear Principal Component Analysis with Gaussian Process Latent Variable Models. *Journal of Machine Learning Research*, 6(Nov.), 1783–1816.
- Leemis, Lawrence M., and McQueston, Jacquelyn T. 2008. Univariate Distribution Relationships. *American Statistician*, 62(1), 45–53.
- Lehmann, Erich L., and Romano, Joseph P. 2005. Testing Statistical Hypotheses. Springer.
- Lehmann, Erich Leo, and Casella, George. 1998. Theory of Point Estimation. Springer.
- Liesen, Jörg, and Mehrmann, Volker. 2015. Linear Algebra. Springer.
- Lin, Hsuan-Tien, Lin, Chih-Jen, and Weng, Ruby C. 2007. A Note on Platt's Probabilistic Outputs for Support Vector Machines. *Machine Learning*, 68, 267–276.
- Ljung, Lennart. 1999. System Identification: Theory for the User. Prentice Hall.
- Loosli, Gaëlle, Canu, Stéphane, and Ong, Cheng Soon. 2016. Learning SVM in Krein Spaces. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 38(6), 1204–1216.
- Luenberger, David G. 1969. Optimization by Vector Space Methods. Wiley.
- MacKay, David J. C. 1992. Bayesian Interpolation. *Neural Computation*, 4, 415–447.
- MacKay, David J. C. 1998. Introduction to Gaussian Processes. Pages 133–165 of: *Neural Networks and Machine Learning*. Springer.
- MacKay, David J. C. 2003. Information Theory, Inference, and Learning Algorithms. Cambridge University Press.
- Magnus, Jan R., and Neudecker, Heinz. 2007. Matrix Differential Calculus with Applications in Statistics and Econometrics. Wiley.

- Manton, Jonathan H., and Amblard, Pierre-Olivier. 2015. A Primer on Reproducing Kernel Hilbert Spaces. *Foundations and Trends in Signal Processing*, 8(1–2), 1–126.
- Markovsky, Ivan. 2011. *Low Rank Approximation: Algorithms, Implementation, Applications*. Springer.
- Maybeck, Peter S. 1979. *Stochastic Models, Estimation, and Control*. Academic Press.
- McCullagh, Peter, and Nelder, John A. 1989. *Generalized Linear Models*. CRC Press.
- McEliece, Robert J., MacKay, David J. C., and Cheng, Jung-Fu. 1998. Turbo Decoding as an Instance of Pearl's "Belief Propagation" Algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2), 140–152.
- Mika, Sebastian, Rätsch, Gunnar, Weston, Jason, Schölkopf, Bernhard, and Müller, Klaus-Robert. 1999. Fisher Discriminant Analysis with Kernels. Pages 41–48 of: *Proceedings of the Workshop on Neural Networks for Signal Processing*.
- Minka, Thomas P. 2001a. A Family of Algorithms for Approximate Bayesian Inference. Ph.D. thesis, Massachusetts Institute of Technology.
- Minka, Tom. 2001b. Automatic Choice of Dimensionality of PCA. In: *Advances in Neural Information Processing Systems*.
- Mitchell, Tom. 1997. *Machine Learning*. McGraw-Hill.
- Mnih, Volodymyr, Kavukcuoglu, Koray, & Silver, David, et al. 2015. Human-Level Control through Deep Reinforcement Learning. *Nature*, 518, 529–533.
- Moonen, Marc, and De Moor, Bart. 1995. *SVD and Signal Processing, III: Algorithms, Architectures and Applications*. Elsevier.
- Moustaki, Irini, Knott, Martin, and Bartholomew, David J. 2015. Latent-Variable Modeling. American Cancer Society. Pages 1–10.
- Müller, Andreas C., and Guido, Sarah. 2016. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Publishing.
- Murphy, Kevin P. 2012. *Machine Learning: A Probabilistic Perspective*. MIT Press.
- Neal, Radford M. 1996. *Bayesian Learning for Neural Networks*. Ph.D. thesis, Department of Computer Science, University of Toronto.
- Neal, Radford M., and Hinton, Geoffrey E. 1999. A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants. Pages 355–368 of: *Learning in Graphical Models*. MIT Press.
- Nelsen, Roger. 2006. *An Introduction to Copulas*. Springer.
- Nesterov, Yuri. 2018. *Lectures on Convex Optimization*. Springer.

- Neumaier, Arnold. 1998. Solving Ill-Conditioned and Singular Linear Systems: A Tutorial on Regularization. *SIAM Review*, 40, 636–666.
- Nocedal, Jorge, and Wright, Stephen J. 2006. *Numerical Optimization*. Springer.
- Nowozin, Sebastian, Gehler, Peter V., Jancsary, Jeremy, and Lampert, Christoph H. (eds). 2014. *Advanced Structured Prediction*. MIT Press.
- O'Hagan, Anthony. 1991. Bayes–Hermite Quadrature. *Journal of Statistical Planning and Inference*, 29, 245–260.
- Ong, Cheng Soon, Mary, Xavier, Canu, Stéphane, and Smola, Alexander J. 2004. Learning with Non-Positive Kernels. In: *Proceedings of the International Conference on Machine Learning*.
- Ormoneit, Dirk, Sidenbladh, Hedvig, Black, Michael J., and Hastie, Trevor. 2001. Learning and Tracking Cyclic Human Motion. In: *Advances in Neural Information Processing Systems*.
- Page, Lawrence, Brin, Sergey, Motwani, Rajeev, and Winograd, Terry. 1999. The PageRank Citation Ranking: Bringing Order to the Web. Tech. rept. Stanford InfoLab.
- Paquet, Ulrich. 2008. Bayesian Inference for Latent Variable Models. Ph.D. thesis, University of Cambridge.
- Parzen, Emanuel. 1962. On Estimation of a Probability Density Function and Mode. *Annals of Mathematical Statistics*, 33(3), 1065–1076.
- Pearl, Judea. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.
- Pearl, Judea. 2009. *Causality: Models, Reasoning and Inference*. 2nd edn. Cambridge University Press.
- Pearson, Karl. 1895. Contributions to the Mathematical Theory of Evolution. II. Skew Variation in Homogeneous Material. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 186, 343–414.
- Pearson, Karl. 1901. On Lines and Planes of Closest Fit to Systems of Points in Space. *Philosophical Magazine*, 2(11), 559–572.
- Peters, Jonas, Janzing, Dominik, and Schölkopf, Bernhard. 2017. *Elements of Causal Inference: Foundations and Learning Algorithms*. MIT Press.
- Petersen, Kaare B., and Pedersen, Michael S. 2012. *The Matrix Cookbook*. Tech. rept. Technical University of Denmark.
- Platt, John C. 2000. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In: *Advances in Large Margin Classifiers*.

- Pollard, David. 2002. *A User's Guide to Measure Theoretic Probability*. Cambridge University Press.
- Polyak, Roman A. 2016. The Legendre Transformation in Modern Optimization. Pages 437–507 of: *Optimization and Its Applications in Control and Data Sciences*. Springer.
- Press, William H., Teukolsky, Saul A., Vetterling, William T., and Flannery, Brian P. 2007. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press.
- Proschan, Michael A., and Presnell, Brett. 1998. Expect the Unexpected from Conditional Expectation. *American Statistician*, 52(3), 248–252.
- Raschka, Sebastian, and Mirjalili, Vahid. 2017. *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow*. Packt Publishing.
- Rasmussen, Carl E., and Ghahramani, Zoubin. 2001. Occam's Razor. In: *Advances in Neural Information Processing Systems*.
- Rasmussen, Carl E., and Ghahramani, Zoubin. 2003. Bayesian Monte Carlo. In: *Advances in Neural Information Processing Systems*.
- Rasmussen, Carl E., and Williams, Christopher K. I. 2006. *Gaussian Processes for Machine Learning*. MIT Press.
- Reid, Mark, and Williamson, Robert C. 2011. Information, Divergence and Risk for Binary Experiments. *Journal of Machine Learning Research*, 12, 731–817.
- Rifkin, Ryan M., and Lippert, Ross A. 2007. Value Regularization and Fenchel Duality. *Journal of Machine Learning Research*, 8, 441–479.
- Rockafellar, Ralph T. 1970. *Convex Analysis*. Princeton University Press.
- Rogers, Simon, and Girolami, Mark. 2016. *A First Course in Machine Learning*. Chapman and Hall/CRC.
- Rosenbaum, Paul R. 2017. *Observation and Experiment: An Introduction to Causal Inference*. Harvard University Press.
- Rosenblatt, Murray. 1956. Remarks on Some Nonparametric Estimates of a Density Function. *Annals of Mathematical Statistics*, 27(3), 832–837.
- Roweis, Sam T. 1998. EM Algorithms for PCA and SPCA. In: *Advances in Neural Information Processing Systems*.
- Roweis, Sam T., and Ghahramani, Zoubin. 1999. A Unifying Review of Linear Gaussian Models. *Neural Computation*, 11(2), 305–345.
- Roy, Anindya, and Banerjee, Sudipto. 2014. *Linear Algebra and Matrix Analysis for Statistics*. Chapman and Hall/CRC.

- Rubinstein, Reuven Y., and Kroese, Dirk P. 2016. *Simulation and the Monte Carlo Method*. Wiley.
- Ruffini, Paolo. 1799. *Teoria Generale delle Equazioni, in cui si Dimostra Impossibile la Soluzione Algebraica delle Equazioni Generali di Grado Superiore al Quarto*. Stamperia di S. Tommaso d'Aquino.
- Rumelhart, David E., Hinton, Geoffrey E., and Williams, Ronald J. 1986. Learning Representations by Back-Propagating Errors. *Nature*, 323(6088), 533–536.
- Sæmundsson, Steindór, Hofmann, Katja, and Deisenroth, Marc P. 2018. Meta Reinforcement Learning with Latent Variable Gaussian Processes. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.
- Saitoh, Saburou. 1988. *Theory of Reproducing Kernels and Its Applications*. Longman Scientific & Technical.
- Särkkä, Simo. 2013. *Bayesian Filtering and Smoothing*. Cambridge University Press.
- Schölkopf, Bernhard, Herbrich, Ralf, and Smola, Alexander J. 2001. A Generalized Representer Theorem. In: *Proceedings of the International Conference on Computational Learning Theory*.
- Schölkopf, Bernhard, and Smola, Alexander J. 2002. *Learning with Kernels – Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press.
- Schölkopf, Bernhard, Smola, Alexander J., and Müller, Klaus-Robert. 1997. Kernel Principal Component Analysis. In: *Proceedings of the International Conference on Artificial Neural Networks*.
- Schölkopf, Bernhard, Smola, Alexander J., and Müller, Klaus-Robert. 1998. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, 10(5), 1299–1319.
- Schwartz, Laurent. 1964. Sous Espaces Hilbertiens d'espaces Vectoriels Topologiques et Noyaux Associés. *Journal d'Analyse Mathématique*, 13, 115–256.
- Schwarz, Gideon E. 1978. Estimating the Dimension of a Model. *Annals of Statistics*, 6(2), 461–464.
- Shahriari, Bobak, Swersky, Kevin, Wang, Ziyu, Adams, Ryan P., and De Freitas, Nando. 2016. Taking the Human out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, 104(1), 148–175.
- Shalev-Shwartz, Shai, and Ben-David, Shai. 2014. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.
- Shawe-Taylor, John, and Cristianini, Nello. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.
- Shawe-Taylor, John, and Sun, Shiliang. 2011. A Review of Optimization Methodologies in Support Vector Machines. *Neurocomputing*, 74(17), 3609–3618.

- Shental, Ori, Siegel, Paul H., Wolf, Jack K., Bickson, Danny, and Dolev, Danny. 2008. Gaussian Belief Propagation Solver for Systems of Linear Equations. In: Proceedings of the International Symposium on Information Theory.
- Shewchuk, Jonathan R. 1994. An Introduction to the Conjugate Gradient Method without the Agonizing Pain.
- Shi, Jianbo, and Malik, Jitendra. 2000. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 888–905.
- Shi, Qinfeng, Petterson, James, Dror, Gideon, Langford, John, Smola, Alexander J., and Vishwanathan, S. V. N. 2009. Hash Kernels for Structured Data. *Journal of Machine Learning Research*, 2615–2637.
- Shiryayev, Albert N. 1984. Probability. Springer.
- Shor, Naum Z. 1985. Minimization Methods for Non-Differentiable Functions. Springer.
- Shotton, Jamie, Winn, John, Rother, Carsten, and Criminisi, Antonio. 2006. Texton-Boost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation. In: Proceedings of the European Conference on Computer Vision.
- Smith, Adrian F. M., and Spiegelhalter, David. 1980. Bayes Factors and Choice Criteria for Linear Models. *Journal of the Royal Statistical Society B*, 42(2), 213–220.
- Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In: Advances in Neural Information Processing Systems.
- Spearman, Charles. 1904. “General Intelligence,” Objectively Determined and Measured. *American Journal of Psychology*, 15(2), 201–292.
- Sriperumbudur, Bharath K., Gretton, Arthur, Fukumizu, Kenji, Schölkopf, Bernhard, and Lanckriet, Gert R. G. 2010. Hilbert Space Embeddings and Metrics on Probability Measures. *Journal of Machine Learning Research*, 11, 1517–1561.
- Steinwart, Ingo. 2007. How to Compare Different Loss Functions and Their Risks. *Constructive Approximation*, 26, 225–287.
- Steinwart, Ingo, and Christmann, Andreas. 2008. Support Vector Machines. Springer.
- Stoer, Josef, and Burlirsch, Roland. 2002. Introduction to Numerical Analysis. Springer.
- Strang, Gilbert. 1993. The Fundamental Theorem of Linear Algebra. *American Mathematical Monthly*, 100(9), 848–855.
- Strang, Gilbert. 2003. Introduction to Linear Algebra. Wellesley–Cambridge Press.

- Stray, Jonathan. 2016. The Curious Journalist's Guide to Data. Tow Center for Digital Journalism at Columbia's Graduate School of Journalism.
- Strogatz, Steven. 2014. Writing about Math for the Perplexed and the Traumatized. *Notices of the American Mathematical Society*, 61(3), 286–291.
- Sucar, Luis E., and Gillies, Duncan F. 1994. Probabilistic Reasoning in High-Level Vision. *Image and Vision Computing*, 12(1), 42–60.
- Szeliski, Richard, Zabih, Ramin, Scharstein, Daniel, et al. 2008. A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6), 1068–1080.
- Tandra, Haryono. 2014. The Relationship between the Change of Variable Theorem and the Fundamental Theorem of Calculus for the Lebesgue Integral. *Teaching of Mathematics*, 17(2), 76–83.
- Tenenbaum, Joshua B., De Silva, Vin, and Langford, John C. 2000. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500), 2319–2323.
- Tibshirani, Robert. 1996. Regression Selection and Shrinkage via the Lasso. *Journal of the Royal Statistical Society B*, 58(1), 267–288.
- Tipping, Michael E., and Bishop, Christopher M. 1999. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society: Series B*, 61(3), 611–622.
- Titsias, Michalis K., and Lawrence, Neil D. 2010. Bayesian Gaussian Process Latent Variable Model. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics*.
- Toussaint, Marc. 2012. Some Notes on Gradient Descent. <https://ipvs.informatik.uni-stuttgart.de/mlr/marc/notes/gradientDescent.pdf>.
- Trefethen, Lloyd N., and Bau III, David. 1997. *Numerical Linear Algebra*. SIAM.
- Tucker, Ledyard R. 1966. Some Mathematical Notes on Three-Mode Factor Analysis. *Psychometrika*, 31(3), 279–311.
- Vapnik, Vladimir N. 1998. *Statistical Learning Theory*. Wiley.
- Vapnik, Vladimir N. 1999. An Overview of Statistical Learning Theory. *IEEE Transactions on Neural Networks*, 10(5), 988–999.
- Vapnik, Vladimir N. 2000. *The Nature of Statistical Learning Theory*. Springer.
- Vishwanathan, S. V. N., Schraudolph, Nicol N., Kondor, Risi, and Borgwardt, Karsten M. 2010. Graph Kernels. *Journal of Machine Learning Research*, 11, 1201–1242.

- von Luxburg, Ulrike, and Schölkopf, Bernhard. 2011. Statistical Learning Theory: Models, Concepts, and Results. Pages 651–706 of: *Handbook of the History of Logic*, vol. 10. Elsevier.
- Wahba, Grace. 1990. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics.
- Walpole, Ronald E., Myers, Raymond H., Myers, Sharon L., and Ye, Keying. 2011. *Probability and Statistics for Engineers and Scientists*. Prentice Hall.
- Wasserman, Larry. 2004. *All of Statistics*. Springer.
- Wasserman, Larry. 2007. *All of Nonparametric Statistics*. Springer.
- Whittle, Peter. 2000. *Probability via Expectation*. Springer.
- Wickham, Hadley. 2014. Tidy Data. *Journal of Statistical Software*, 59, 1–23.
- Williams, Christopher K. I. 1997. Computing with Infinite Networks. In: *Advances in Neural Information Processing Systems*.
- Yu, Yaoliang, Cheng, Hao, Schuurmans, Dale, and Szepesvári, Csaba. 2013. Characterizing the Representer Theorem. In: *Proceedings of the International Conference on Machine Learning*.
- Zadrozny, Bianca, and Elkan, Charles. 2001. Obtaining Calibrated Probability Estimates from Decision Trees and Naive Bayesian Classifiers. In: *Proceedings of the International Conference on Machine Learning*.
- Zhang, Haizhang, Xu, Yuesheng, and Zhang, Jun. 2009. Reproducing Kernel Banach Spaces for Machine Learning. *Journal of Machine Learning Research*, 10, 2741–2775.
- Zia, Royce K. P., Redish, Edward F., and McKay, Susan R. 2009. Making Sense of the Legendre Transform. *American Journal of Physics*, 77(7), 614–622.

*Марк Питер Дайзенрот, Альдо Фейзал, Чен Сунь Он*

## **Математика в машинном обучении**

*Перевел с английского С. Черников*

Руководитель дивизиона	<i>Ю. Сергиенко</i>
Ведущий редактор	<i>Е. Строганова</i>
Научный редактор	<i>К. Кноп</i>
Литературный редактор	<i>А. Алимова</i>
Художественный редактор	<i>В. Мостицан</i>
Корректор	<i>М. Молчанова</i>
Верстка	<i>Л. Егорова</i>

Изготовлено в России. Изготовитель: ООО «Прогресс книга».

Место нахождения и фактический адрес: 194044, Россия, г. Санкт-Петербург,  
Б. Сампсониевский пр., д. 29А, пом. 52. Тел.: +78127037373.

Дата изготовления: 08.2023. Наименование: книжная продукция. Срок годности: не ограничен.

Налоговая льгота — общероссийский классификатор продукции ОК 034-2014, 58.11.12 — Книги печатные  
профессиональные, технические и научные.

Импортер в Беларусь: ООО «ПИТЕР М», 220020, РБ, г. Минск, ул. Тимирязева, д. 121/3, к. 214, тел./факс: 208 80 01.

Подписано в печать 12.07.23. Формат 70×100/16. Бумага офсетная. Усл. п. л. 41,280. Тираж 700. Заказ 0000.