Стивен Л. Брантон, Дж. Натан Куц

# Анализ данных в науке и технике

# Data-Driven Science and Engineering

## Machine Learning, Dynamical Systems, and Control

Steven L. Brunton, J. Nathan Kutz

CAMBRIDGE
UNIVERSITY PRESS

# Содержание

# От издательства

## *Отзывы и пожелания*

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

## *Скачивание исходного кода примеров*

Скачать файлы с дополнительной информацией для книг издательства «ДМК Пресс» можно на сайте www.dmkpress.com на странице с описанием соответствующей книги.

## *Список опечаток*

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

## *Нарушение авторских прав*

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Springer очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

# Об авторах

**Стивен Л. Брантон** – доцент факультета общего машиностроения в Вашингтонском университете. Также является внештатным сотрудником отделения прикладной математики и науки о данных Института eScience. Область его научных интересов охватывает применение науки о данных и машинного обучения к динамическим системам и управлению в области гидрогазодинамики, биомеханики, оптики, энергетических систем и производства. Является автором двух учебников, лауреатом премии армии и ВВС для молодых ученых, получил право преподавания в инженерном колледже Вашингтонского университета и удостоен премии для молодых преподавателей.

**Дж. Натан Куц** – профессор прикладной математики Вашингтонского университета, был деканом факультета до 2015 года. Также является внештатным профессором отделения электротехники и физики и старшим научным сотрудником отделения науки о данных в Институте eScience. Область научных интересов охватывает сложные системы и анализ данных, конкретно применение методов машинного обучения и динамических систем и управление в разнообразных приложениях. Автор двух учебников, лауреат премии Боинг за отличное преподавание прикладной математики, а также премии CAREER Национального научного фонда США.

# Предисловие

Эта книга посвящена растущей области знаний на пересечении методов обработки больших данных, прикладной оптимизации и классических дисциплин инженерной математики и математической физики. Мы готовили данный материал на протяжении ряда лет, в основном для лекций, читаемых студентам старших курсов и аспирантам технических и физических факультетов.

Обычно такие студенты имеют подготовку в области линейной алгебры, дифференциальных уравнений и научных расчетов, а инженеры также знакомы с теорией управления и (или) дифференциальными уравнениями в частных производных. Однако в большинстве учебных программ научно-технических вузов методы обработки данных и (или) оптимизации освещаются слабо или не включены вовсе. С другой стороны, студенты, обучающиеся по специальностям «информатика» и «статистика», плохо знакомы с динамическими системами и теорией управления. Нашей целью было написать введение в прикладную науку о данных для обеих групп. Включенные в книгу методы отбирались по трем критериям: (1) релевантность, (2) простота и (3) общность. Мы стремились представить широкий круг тем от вводного материала до методов, реально применяемых в исследованиях.

Открытие на основе анализа данных революционизировало наши подходы к моделированию, прогнозированию поведения и управлению сложными системами. Самые насущные научно-технические задачи нашего времени не поддаются эмпирическим моделям и выводам, основанным на первопринципах. Все чаще исследователи обращаются к подходам на основе анализа данных при изучении широкого спектра сложных систем, как то: турбулентность, науки о мозге, климатология, эпидемиология, финансы, робототехника, автономные системы. Такие системы обычно являются нелинейными, динамическими, многомасштабными в пространстве и во времени, многомерными и имеют доминирующие паттерны, которые необходимо охарактеризовать и смоделировать, чтобы в конечном итоге обеспечить сбор данных, прогнозирование, оценку и управление. Благодаря современным математическим методам вкупе с невиданной ранее доступностью данных и располагаемыми вычислительными ресурсами мы теперь можем подступиться к неприступным до недавнего времени проблемам. Упомянем лишь малую толику новых методов: надежное восстановление изображения по разреженным и зашумленным измерениям случайных пикселей, управление турбулентностью с помощью машинного обучения, оптимальное размещение датчиков и приводов, идентификация допускающих интерпретацию нелинейных динамических систем на основе одних лишь данных и модели пониженного порядка, позволяющие ускорить изучение и оптимизацию систем со сложной многомасштабной физикой.

Движущим началом современной науки о данных является доступность больших и постоянно увеличивающихся объемов данных вследствие заме-

чательных инноваций в области разработки дешевых датчиков, возросших на порядки вычислительных мощностей и практически неограниченной емкости устройств хранения и скорости передачи. Такое изобилие данных открывает перед учеными и инженерами во всех областях новые возможности для изобретений на основе анализа данных; часто в этой связи говорят о четвертой парадигме научного открытия [245]. Эта четвертая парадигма представляет собой естественную кульминацию первых трех: эмпирического эксперимента, аналитического вывода и вычислительного исследования. Интеграция всех трех методик создает новаторскую платформу для новых открытий на основе данных. Этот процесс научного открытия не нов и по сути дела повторяет усилия титанов научной революции: Иоганна Кеплера (1571–1630) и сэра Исаака Ньютона (1642–1727). Оба сыграли ключевую роль в разработке теоретических принципов небесной механики на базе сочетания эмпирических подходов, основанных на анализе данных, и аналитических вычислений. Наука о данных не заменяет математическую физику и технику, но дополняет ее с учетом достижений XXI века, что больше напоминает возрождение, нежели революцию.

Наука о данных сама по себе не нова, ее предложил больше 50 лет назад Джон Тьюки, предвидевший появление науки, в центре внимания которой будет обучение на данных, или *анализ данных* [152]. С тех пор в науке о данных преобладают два разных подхода [78]. Сообщество *машинного обучения* состоит в основном из специалистов по информатике и интересуется в первую очередь разработкой быстрых, масштабируемых и качественных алгоритмов прогнозирования. Сообщество же *статистического обучения*, которое вовсе необязательно во всем противопоставлять первому, больше сосредоточено на факультетах математической статистики и занимается выводом интерпретируемых моделей. Обе методологии могут похвастаться значительными успехами и закладывают математические и вычислительные основания методов науки о данных. Целью ученых и инженеров должно стать использование этих методов для выведения из результатов наблюдений и обсчета моделей (чаще нелинейных), которые правильно улавливают динамику системы *и* количественно и качественно обобщаются на ненаблюдавшиеся области фазового, параметрического или прикладного пространства. А в этой книге нашей целью будет применение статистических методов и методов машинного обучения к решению технических задач.

## Рассматриваемые вопросы

В книге обсуждается целый ряд ключевых тем. Во-первых, во многих сложных системах присутствуют доминирующие *низкоразмерные паттерны* данных, несмотря на быстрое увеличение разрешающей способности измерений и вычислений. Базовая структура открывает возможность эффективного размещения датчиков и компактного представления для моделирования и управления. Выделение паттернов тесно связано со второй темой: отысканием *преобразований координат*, позволяющих упростить систему.

Действительно, богатая история математической физики вращается вокруг преобразований координат (например, спектральные разложения, преобразование Фурье, обобщенные функции и т. д.), хотя эти методы в большинстве своем были ограничены простой идеализированной геометрией и линейной динамикой. Умение выводить преобразования *на основе данных* открывает возможность обобщить их на новые задачи с более сложной геометрией и граничными условиями. На протяжении всей книги мы будем интересоваться *динамическими системами и управлением*, т. е. применением методов, основанных на анализе данных, к моделированию и управлению систем, изменяющихся во времени. Красной нитью проходит тема *управляемой данными прикладной оптимизации*, поскольку едва ли не каждый рассматриваемый вопрос так или иначе связан с оптимизацией (например, нахождение *оптимальных* низкоразмерных паттернов, *оптимальное* расположение датчиков, *оптимизация* в машинном обучении, *оптимальное* управление и т. д.). И еще одна, даже более фундаментальная тема – большинство данных организовано в массивы для анализа, а широкое развитие численных инструментов линейной алгебры, начиная с 1960-х годов, лежит в основе математических методов матричных разложений и стратегий решения, встречающихся в этой книге.

# Благодарности

Мы в долгу перед многими талантливыми студентами, сотрудниками и коллегами, которые делились ценными замечаниями и предложениями и оказывали нам поддержку. Особенно мы благодарны Джошуа Проктору (Joshua Proctor), который стоял у истоков этой книги и помогал при планировании ее структуры и организации. Мы также извлекли много полезного из бесед с Бингом Брантоном (Bing Brunton), Игорем Мезичем (Igor Mezić), Берндом Ноаком (Bernd Noack) и Сэмом Тайрой (Sam Taira). Эта книга была бы невозможна без помощи наших сотрудников и коллег, исследования которых отражены в тексте.

На протяжении работы над книгой и чтения соответствующих курсов мы получали чрезвычайно ценные отзывы и комментарии от наших замечательных студентов и постдоков: Трэвиса Эшкама (Travis Askham), Майкла Ау-Юнга (Michael Au-Yeung), Цзе Бая (Zhe Bai), Идо Брайта (Ido Bright), Кэтлин Чемпион (Kathleen Champion), Эмили Кларк (Emily Clark), Чарльза Делаханта (Charles Delahunt), Даниэля Дылевски (Daniel Dylewski), Бена Эричсона (Ben Erichson), Чарли Фислера (Charlie Fiesler), Синь Фу (Xing Fu), Чена Гонга (Chen Gong), Тарена Гормана (Taren Gorman), Джейкоба Гросека (Jacob Grosek), Сета Хирша (Seth Hirsh), Микала Джонсона (Mikala Johnson), Юрики Кайзер (Eurika Kaiser), Мейсона Кема (Mason Kamb), Джеймса Кьюнерта (James Kunert), Бетани Луш (Bethany Lusch), Педро Майа (Pedro Maia), Критики Манохара (Krithika Manohar), Найла Мэнгана (Niall Mangan), Арианы Мендибль (Ariana Mendible), Томаса Морена (Thomas Mohren), Меган Моррисон (Megan Morrison), Маркуса Куэйда (Markus Quade), Сэма Руди (Sam Rudy), Сюзанны Саргсян

(Susanna Sargsyan), Изабель Шерл (Isabel Scherl), Эли Шлизермана (Eli Shlizerman), Джорджа Степанянца (George Stepaniants), Бена Строма (Ben Strom), Чань Суна (Chang Sun), Роя Тэйлора (Roy Taylor), Мегханы Велагар (Meghana Velagar), Джейка Вехолта (Jake Weholt) и Мэтта Уильямса (Matt Williams). Наши студенты подвигли нас на написание этой книги, благодаря им мы каждый день приходим на работу с радостью и волнением.

Мы также благодарны руководителю издательской группы в Cambridge University Press Лоран Коулз (Lauren Cowles), на которую могли положиться на протяжении всего процесса работы.

## Онлайновые материалы

Мы с самого начала предполагали, что к книге будут прилагаться обширные дополнительные материалы: код, данные, видео, домашние задания и рекомендуемые способы построения курса. Все эти материалы можно найти на сайте databookuw.com.

Код в сети полнее, чем в книге, в частности включен код генерации рисунков, пригодных для публикации. Визуализация данных была поставлена на первое место среди методов науки о данных в опросе «Состояние науки о данных и машинного обучения», проведенном на Kaggle 2017. Поэтому мы настоятельно рекомендуем читателям скачать код с сайта и в полной мере воспользоваться командами построения графиков и диаграмм.

Мы также записали и разместили на YouTube лекции по большинству тем, включенных в книгу. Есть дополнительные видео для студентов, желающих восполнить пробелы в подготовке по научным расчетам и основам прикладной математики. Этот текст задуман одновременно как справочное пособие и источник материалов к нескольким курсам, рассчитанным на студентов разного уровня. Большинство глав самостоятельны, на их основе можно разработать *курсы молодого бойца*, рассчитанные примерно на 10 часов каждый.

## Как пользоваться этой книгой

Книга рассчитана на начинающих аспирантов и продвинутых студентов старших курсов научных и технических факультетов. Поэтому методы машинного обучения излагаются с азов, но при этом мы предполагаем, что студенты умеют моделировать физические системы с помощью дифференциальных уравнений и решать их с помощью таких программ, как **ode45**. Рассматриваются как начальные вопросы, так и актуальные исследовательские методы. Наша цель – представить цельный взгляд и математический инструментарий для решения научно-технических задач. Но книга может быть также полезна студентам, изучающим информатику и статистику, которые зачастую мало знают о динамических системах и теории управления. На основе представленного материала можно разработать несколько курсов,

программы некоторых из них имеются на сайте книги и включают домашние задания, наборы данных и код.

Прежде всего мы хотели, чтобы книга была интересной, чтобы она вдохновляла, открывала глаза и вооружала знаниями молодых ученых и инженеров. Мы пытались по возможности не слишком усложнять, не жертвуя при этом глубиной и широтой охвата, без которых не может быть никакой исследовательской работы. Многие главы можно было бы развернуть в целые книги, и такие книги есть. Однако мы также стремились к полноте в той мере, в какой этого можно ожидать от книги, посвященной столь обширной и быстро развивающей области. Мы надеемся, что книга придется вам по вкусу, что вы овладеете всеми описанными в ней методами и измените мир с помощью прикладной науки о данных!

# Общеупотребительные методы оптимизации, уравнения, символы и акронимы

## Наиболее распространенные стратегии оптимизации

**Метод наименьших квадратов** (обсуждается в главах 1 и 4) минимизирует сумму квадратов разностей (ошибок) между фактическими данными и предсказаниями модели. В случае линейного метода наименьших квадратов, когда данные аппроксимируются линейной функцией, имеется решение в замкнутой форме, которое можно найти, приравняв к нулю производную ошибки по каждому неизвестному. Этот подход широко используется в технике и прикладных науках для аппроксимации полиномиальными функциями. Применение нелинейного метода наименьших квадратов обычно требует итеративного уточнения путем аппроксимации нелинейного решения линейным на каждой итерации.

**Градиентный спуск** (обсуждается в главах 4 и 6) – основной метод выпуклой оптимизации в многомерных системах. Для минимизации ошибки вычисляется градиент аппроксимирующей функции. Решение обновляется итеративно путем *спуска с горы* в пространстве решений. Одномерным вариантом градиентного спуска является метод Ньютона–Рафсона. В многомерном пространстве метод часто находит только локальный минимум. Важнейшими инновациями в приложениях больших данных являются стохастический градиентный спуск и алгоритм обратного распространения, благодаря чему оптимизация сводится к самому вычислению градиента.

**Чередующийся градиентный спуск** (Alternating Descent Method – ADM) (обсуждается в главе 4) позволяет избежать вычисления градиента за счет того, что на каждом шаге производится оптимизация по одной неизвестной. Таким образом, все неизвестные переменные считаются постоянными, за исключением одной, по которой производится линейный поиск (невыпуклая оптимизация). Эта переменная обновляется, после чего фиксируется, и то же самое повторяется для другой переменной. На одном шаге итерации пере-

бираются все неизвестные, а сами итерации продолжаются до тех пор, пока не будет достигнута желаемая точность.

**Расширенный метод Лагранжа** (Augmented Lagrange Method – ALM) (обсуждается в главах 3 и 8) – класс алгоритмов для решения задач условной оптимизации. Они похожи на методы штрафования тем, что заменяют задачу оптимизации с ограничениями последовательностью задач без ограничений и прибавляют к целевой функции штрафной член, который играет роль множителя Лагранжа. Расширенный метод Лагранжа – не то же самое, что метод множителей Лагранжа.

**Линейное программирование и симплекс-метод** – безотказные алгоритмы выпуклой оптимизации. В линейном программировании целевая функция линейно зависит от неизвестных, а ограничениями являются линейные равенства и неравенства. Вычислив область допустимых решений – выпуклый политоп, – алгоритм линейного программирования находит в полиэдре точку, в которой функция принимает наименьшее (или наибольшее) значение, если таковая существует. Симплекс-метод – это конкретная итеративная процедура линейного программирования, которая по заданному опорному допустимому решению пытается найти другое опорное решение, для которого целевая функция принимает меньшее значение, и тем самым производит оптимизацию.

# Наиболее употребительные уравнения и символы

## *Линейная алгебра*

*Линейная система уравнений*

$$\mathbf{Ax} = \mathbf{b}. \tag{0.1}$$

Матрица $\mathbf{A} \in \mathbb{R}^{p \times n}$ и вектор $\mathbf{b} \in \mathbb{R}^p$ обычно известны, а вектор $\mathbf{x} \in \mathbb{R}^n$ неизвестен.

*Уравнение для собственных значений*

$$\mathbf{AT} = \mathbf{T\Lambda}. \tag{0.2}$$

Столбец $\boldsymbol{\xi}_k$ матрицы $\mathbf{T}$ является собственным вектором матрицы $\mathbf{A} \in \mathbb{C}^{n \times n}$, соответствующим собственному значению $\lambda_k$: $\mathbf{A}\boldsymbol{\xi}_k = \lambda_k \boldsymbol{\xi}_k$. Матрица $\boldsymbol{\Lambda}$ – диагональная матрица, содержащая эти собственные значения, в простейшем случае все $n$ собственных значений различны.

*Замена координат*

$$\mathbf{x} = \boldsymbol{\Psi}\mathbf{a}. \tag{0.3}$$

Вектор $\mathbf{x} \in \mathbb{R}^n$ можно записать как $\mathbf{a} \in \mathbb{R}^n$ в системе координат, определенной столбцами матрицы $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$.

*Уравнение измерений*

$$\mathbf{y} = \mathbf{C}\mathbf{x}. \tag{0.4}$$

Вектор $\mathbf{y} \in \mathbb{R}^p$ является измерением состояния $\mathbf{x} \in \mathbb{R}^n$ в результате применения матрицы измерений $\mathbf{C} \in \mathbb{R}^{p \times n}$.

*Сингулярное разложение*

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \approx \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*. \tag{0.5}$$

Матрицу $\mathbf{X} \in \mathbb{C}^{n \times m}$ можно разложить в произведение трех матриц $\mathbf{U} \in \mathbb{C}^{n \times n}$, $\mathbf{\Sigma} \in \mathbb{C}^{n \times m}$ и $\mathbf{V} \in \mathbb{C}^{m \times m}$. Матрицы $\mathbf{U}$ и $\mathbf{V}$ *унитарные*, т. е. $\mathbf{U}\mathbf{U}^* = \mathbf{U}^*\mathbf{U} = \mathbf{I}^{n \times n}$ и $\mathbf{V}\mathbf{V}^* = \mathbf{V}^*\mathbf{V} = \mathbf{I}^{m \times m}$, где $^*$ обозначает операцию комплексного сопряжения и транспонирования. Столбцы $\mathbf{U}$ (соответственно $\mathbf{V}$) ортогональны и называются левыми (соответственно правыми) *сингулярными векторами*. На главной диагонали диагональной матрицы $\mathbf{\Sigma}$ находятся убывающие неотрицательные элементы, называемые *сингулярными значениями*.

Часто $\mathbf{X}$ аппроксимируется матрицей низкого ранга $\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$, где $\tilde{\mathbf{U}}$ и $\tilde{\mathbf{V}}$ содержат первые $r \ll n$ столбцов $\mathbf{U}$ и $\mathbf{V}$ соответственно, а $\tilde{\mathbf{\Sigma}}$ – левый верхний блок $\mathbf{\Sigma}$ размера $r \times r$. В контексте пространственных мод, моделей пониженного порядка и размещения датчиков матрица $\tilde{\mathbf{U}}$ часто обозначается буквой $\mathbf{\Psi}$.

## *Регрессия и оптимизация*

*Оптимизация переопределенных и недоопределенных линейных систем*

$$\operatorname{argmin}_{\mathbf{x}}(||\mathbf{A}\mathbf{x} - \mathbf{b}||_2 + \lambda g(\mathbf{x})) \text{ или} \tag{0.6a}$$

$$\operatorname{argmin}_{\mathbf{x}} g(\mathbf{x}) \text{ при условии } ||\mathbf{A}\mathbf{x} - \mathbf{b}||_2 \leq \varepsilon. \tag{0.6b}$$

Здесь $g(\mathbf{x})$ – штраф регрессии (со штрафным параметром $\lambda$ для переопределенных систем). Для переопределенных и недоопределенных систем линейных уравнений $\mathbf{A}\mathbf{x} = \mathbf{b}$, когда решений либо не существует, либо бесконечно много, для нахождения решения нужно задать ограничение или штраф; эта процедура называется *регуляризацией*.

*Оптимизация переопределенных и недоопределенных линейных систем*

$$\operatorname{argmin}_{\mathbf{x}}(f(\mathbf{A}, \mathbf{x}, \mathbf{b}) + \lambda g(\mathbf{x})) \text{ или} \tag{0.7a}$$

$$\operatorname{argmin}_{\mathbf{x}} g(\mathbf{x}) \text{ при условии } f(\mathbf{A}, \mathbf{x}, \mathbf{b}) \leq \varepsilon. \tag{0.7b}$$

Это обобщение линейной системы на нелинейную систему $f(\cdot)$ с регуляризацией $g(\cdot)$. Такие переопределенные и недоопределенные системы часто решаются методами градиентного спуска.

*Композиционная оптимизация для нейронных сетей*

$$\operatorname{argmin}_{\mathbf{A}_j}(f_M(\mathbf{A}_M, \cdots f_2(\mathbf{A}_2, (f_1(\mathbf{A}_1, \mathbf{x})) \cdots) + \lambda g(\mathbf{A}_j)). \tag{0.8}$$

Здесь $\mathbf{A}_k$ – матрицы весов связей между $k$-м и $(k+1)$-м слоями нейронной сети. Обычно это сильно недоопределенная система, которая регуляризи-

руется прибавлением $g(\mathbf{A}_j)$. Композиция и регуляризация весьма важны как для порождения выразительных представлений данных, так и для предотвращения переобучения.

### *Динамические системы и системы пониженного порядка*

*Нелинейное обыкновенное дифференциальное уравнение (динамическая система)*

$$\frac{d}{dt}\mathbf{x}(\mathrm{t}) = \mathbf{f}(\mathbf{x}(t), t; \boldsymbol{\beta}). \tag{0.9}$$

Вектор $\mathbf{x}(t) \in \mathbb{R}^n$ описывает состояние системы, изменяющейся во времени $t$, $\boldsymbol{\beta}$ – вектор параметров, а $\mathbf{f}$ – векторное поле. В общем случае $\mathbf{f}$ – липшицева функция, что гарантирует существование и единственность решения.

*Система с линейной зависимостью выхода от входа*

$$\frac{d}{dt}\mathbf{x} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{0.10a}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}. \tag{0.10b}$$

Состояние системы представлено вектором $\mathbf{x} \in \mathbb{R}^n$, входы (приводы) – вектором $\mathbf{u} \in \mathbb{R}^q$, а выходы (датчики) – вектором $\mathbf{y} \in \mathbb{R}^p$. Матрицы $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$ определяют динамику, управляющее воздействие, стратегию работы датчиков и эффект сквозного управления соответственно.

*Нелинейное отображение (динамические системы с дискретным временем)*

$$\mathbf{x}_{k+1} = \mathbf{F}(\mathbf{x}_k). \tag{0.11}$$

Состояние системы на $k$-й итерации представлено вектором $\mathbf{x}_k \in \mathbb{R}^n$, а $\mathbf{F}$ – потенциально нелинейное отображение. Часто это отображение описывает продвижение итераций во времени, т. е. $\mathbf{x}_k = \mathbf{x}(k\Delta t)$; в таком случае потоковое отображение обозначается $\mathbf{F}_{\Delta t}$.

*Операторное уравнение Купмана (с дискретным временем)*

$$\mathcal{K}_t g = g \circ \mathbf{F}_t \Rightarrow \mathcal{K}_t \varphi = \lambda\varphi. \tag{0.12}$$

Линейный оператор Купмана $\mathcal{K}_t$ экстраполирует функции измерения состояния $g(\mathbf{x})$ с помощью потока $\mathbf{F}_t$. Собственные значения и собственные векторы $\mathcal{K}_t$ обозначаются $\lambda$ и $\varphi(\mathbf{x})$ соответственно. Оператор $\mathcal{K}_t$ применяется к гильбертову пространству измерений.

*Нелинейные дифференциальные уравнения в частных производных (УрЧП)*

$$\mathbf{u}_t = \mathbf{N}(\mathbf{u}, \mathbf{u}_x, \mathbf{u}_{xx}, \cdots, x, t; \boldsymbol{\beta}). \tag{0.13}$$

Состояние УрЧП описывается вектором $\mathbf{u}$, $\mathbf{N}$ – нелинейный оператор эволюции, нижние индексы обозначают взятие частных производных, а $x$ и $t$ –

пространственная и временная переменные соответственно. УрЧП параметризуется значениями, собранными в векторе $\boldsymbol{\beta}$. Состояние УрЧП $\mathbf{u}$ может быть непрерывной функцией $u(x, t)$, а может быть дискретизировано в нескольких точках пространства, $\mathbf{u}(t) = [u(x_1, t)\ u(x_2, t) \cdots u(x_n, t)]^T \in \mathbb{R}^n$.

*Разложение Галеркина*
Непрерывное разложение Галеркина имеет вид:

$$u(x,t) \approx \sum_{k=1}^{r} a_k(t)\psi_k(x). \tag{0.14}$$

Функции $a_k(t)$ – коэффициенты, отражающие временную динамику, а $\psi_k(x)$ – пространственные моды. Для многомерного дискретизированного состояния разложение Галеркина принимает вид $\mathbf{u}(t) \approx \sum_{k=1}^{r} a_k(t)\boldsymbol{\psi}_k$. Пространственные моды $\boldsymbol{\psi}_k \in \mathbb{R}^n$ могут быть столбцами матрицы $\boldsymbol{\Psi} = \tilde{\mathbf{U}}$.

# Список обозначений

## Размерности

$K$ количество ненулевых элементов $K$-разреженного вектора $\mathbf{s}$
$m$ количество снимков данных (т. е. столбцов $\mathbf{X}$)
$n$ размерность состояния $\mathbf{x} \in \mathbb{R}^n$
$p$ размерность измерения, или выходной переменной $\mathbf{y} \in \mathbb{R}^p$
$q$ размерность выходной переменной $\mathbf{u} \in \mathbb{R}^q$
$r$ ранг усеченного сингулярного разложения или иной низкоранговой аппроксимации

## Скаляры

$s$ частота в лапласовой области
$t$ время
$\delta$ скорость обучения в методе градиентного спуска
$\Delta t$ временной шаг
$x$ пространственная переменная
$\Delta x$ пространственный шаг
$\sigma$ сингулярное значение
$\lambda$ собственное значение
$\lambda$ параметр разреженности при разреженной оптимизации (раздел 7.3)
$\lambda$ множитель Лагранжа (разделы 3.7, 8.4 и 11.4)
$\tau$ порог

## Векторы

$\mathbf{a}$ вектор амплитуд мод $\mathbf{x}$ в базисе $\boldsymbol{\Psi}$, $\mathbf{a} \in \mathbb{R}^r$
$\mathbf{b}$ вектор измерений в линейной системе $\mathbf{Ax} = \mathbf{b}$
$\mathbf{b}$ вектор амплитуд мод в разложении по динамическим модам (раздел 7.2)
$\mathbf{Q}$ вектор, содержащий функцию потенциала в алгоритме PDE-FIND

$\mathbf{r}$ вектор невязок
$\mathbf{s}$ разреженный вектор $\mathbf{s} \in \mathbb{R}^n$
$\mathbf{u}$ регулируемая переменная (главы 8, 9, 10)
$\mathbf{u}$ вектор состояния УрЧП (главы 11 и 12)
$\mathbf{w}$ экзогенные входы
$\mathbf{w}_d$ возмущения системы
$\mathbf{w}_n$ шум измерений
$\mathbf{w}_r$ опорная траектория
$\mathbf{x}$ состояние системы $\mathbf{x} \in \mathbb{R}^n$
$\mathbf{x}_k$ снимок данных в момент $t_k$
$\mathbf{x}_j$ пример данных $j \in Z := \{1, 2, \cdots, m\}$ (главы 5 и 6)
$\tilde{\mathbf{x}}$ упрощенное состояние $\tilde{\mathbf{x}} \in \mathbb{R}^r$, т. е. $\mathbf{x} \approx \tilde{\mathbf{U}}\tilde{\mathbf{x}}$
$\hat{\mathbf{x}}$ оценка состояния системы
$\mathbf{y}$ вектор измерений $\mathbf{y} \in \mathbb{R}^p$
$\mathbf{y}_j$ метка данных $j \in Z := \{1, 2, \cdots, m\}$ (главы 5 и 6)
$\hat{\mathbf{y}}$ оценка измерения выхода
$\mathbf{z}$ преобразованное состояние $\mathbf{x} = \mathbf{T}\mathbf{z}$ (главы 8 и 9)
$\boldsymbol{\varepsilon}$ вектор ошибок
$\boldsymbol{\beta}$ бифуркационные параметры
$\boldsymbol{\xi}$ собственный вектор оператора Купмана (разделы 7.4 и 7.5)
$\boldsymbol{\xi}$ разреженный вектор коэффициентов (раздел 7.3)
$\boldsymbol{\varphi}$ мода в разложении по динамическим модам
$\boldsymbol{\psi}$ мода собственного ортогонального разложения (POD)
$\boldsymbol{\Upsilon}$ вектор измерений УрЧП в алгоритме PDE-FIND

### *Матрицы*

$\mathbf{A}$ матрица системы уравнений, или динамики
$\tilde{\mathbf{A}}$ редуцированная динамика в $r$-мерном подпространстве POD
$\mathbf{A}_\mathbf{X}$ матричное представление линейной динамики с состоянием $\mathbf{x}$
$\mathbf{A}_\mathbf{Y}$ матричное представление линейной динамики с наблюдаемыми переменными $\mathbf{y}$
$(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{B})$ матрицы системы с непрерывным пространством состояний
$(\mathbf{A}_d, \mathbf{B}_d, \mathbf{C}_d, \mathbf{B}_d)$ матрицы системы с дискретным пространством состояний
$(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \hat{\mathbf{B}})$ матрицы пространства состояний системы в новых координатах $\mathbf{z} = \mathbf{T}^{-1}\mathbf{x}$
$(\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}}, \tilde{\mathbf{B}})$ матрицы пространства состояний упрощенной системы ранга $r$
$\mathbf{B}$ матрица входных данных с приводов
$\mathbf{C}$ матрица линейных измерений состояний
$\mathcal{C}$ матрица управляемости
$\mathcal{F}$ дискретное преобразование Фурье
$\mathbf{G}$ матричное представление линейной динамики состояний и входов $[\mathbf{x}^T\mathbf{u}^T]^T$
$\mathbf{H}$ матрица Ганкеля
$\mathbf{H}'$ матрица Ганкеля с временным сдвигом
$\mathbf{I}$ единичная матрица
$\mathbf{K}$ матричная форма оператора Купмана (глава 7)
$\mathbf{K}$ коэффициент усиления системы управления с замкнутым контуром (глава 8)

$\mathbf{K}_f$ коэффициент усиления фильтра Калмана
$\mathbf{K}_r$ коэффициент усиления линейно-квадратичного регулятора (ЛКР)
$\mathbf{L}$ низкоранговая часть матрицы $\mathbf{X}$ (глава 3)
$\boldsymbol{O}$ матрица наблюдаемости
$\mathbf{P}$ унитарная матрица, применяемая к столбцам $\mathbf{X}$
$\mathbf{Q}$ весовая матрица стоимости отклонений от нулевого состояния в ЛКР (раздел 8.4)
$\mathbf{Q}$ ортогональная матрица в QR-разложении
$\mathbf{R}$ весовая матрица стоимости управляющих воздействий в ЛКР (раздел 8.4)
$\mathbf{R}$ верхнетреугольная матрица в QR-разложении
$\mathbf{S}$ разреженная часть матрицы $\mathbf{X}$ (глава 3)
$\mathbf{T}$ матрица собственных векторов (глава 8)
$\mathbf{T}$ замена координат (главы 8 и 9)
$\mathbf{U}$ левые сингулярные векторы $\mathbf{X}$, $\mathbf{U} \in \mathbb{R}^{n \times n}$
$\hat{\mathbf{U}}$ левые сингулярные векторы экономичного сингулярного разложения $\mathbf{X}$, $\hat{\mathbf{U}} \in \mathbb{R}^{n \times n}$
$\tilde{\mathbf{U}}$ левые сингулярные векторы (POD-моды) усеченного сингулярного разложения $\mathbf{X}$, $\tilde{\mathbf{U}} \in \mathbb{R}^{n \times r}$
$\mathbf{V}$ правые сингулярные векторы $\mathbf{X}$, $\mathbf{V} \in \mathbb{R}^{m \times m}$
$\tilde{\mathbf{V}}$ правые сингулярные векторы (POD-моды) усеченного сингулярного разложения $\mathbf{X}$, $\tilde{\mathbf{V}} \in \mathbb{R}^{m \times r}$
$\mathbf{\Sigma}$ матрица сингулярных значений $\mathbf{X}$, $\mathbf{\Sigma} \in \mathbb{R}^{n \times m}$
$\hat{\mathbf{\Sigma}}$ матрица сингулярных значений экономичного сингулярного разложения $\mathbf{X}$, $\hat{\mathbf{\Sigma}} \in \mathbb{R}^{m \times m}$
$\tilde{\mathbf{\Sigma}}$ матрица сингулярных значений усеченного сингулярного разложения $\mathbf{X}$, $\tilde{\mathbf{\Sigma}} \in \mathbb{R}^{r \times r}$
$\mathbf{W}$ собственные векторы $\tilde{\mathbf{A}}$
$\mathbf{W}_c$ грамиан управляемости
$\mathbf{W}_o$ грамиан наблюдаемости
$\mathbf{X}$ матрица данных, $\mathbf{X} \in \mathbb{R}^{n \times m}$
$\mathbf{X}'$ матрица данных с временным сдвигом, $\mathbf{X}' \in \mathbb{R}^{n \times m}$
$\mathbf{Y}$ проекция матрицы $\mathbf{X}$ на ортогональный базис в рандомизированном сингулярном разложении (раздел 1.8)
$\mathbf{Y}$ матрица данных наблюдаемых величин, $\mathbf{Y} = \mathbf{g}(\mathbf{X})$, $\mathbf{Y} \in \mathbb{R}^{p \times m}$ (глава 7)
$\mathbf{Y}'$ матрица данных наблюдаемых величин со сдвигом, $\mathbf{Y}' = \mathbf{g}(\mathbf{X}')$, $\mathbf{Y}' \in \mathbb{R}^{p \times m}$ (глава 7)
$\mathbf{Z}$ эскиз матрицы для рандомизированного сингулярного разложения, $\mathbf{Z} \in \mathbb{R}^{n \times r}$ (раздел 1.8)
$\mathbf{\Theta}$ матрица измерений, умноженная на разреживающий базис, $\mathbf{\Theta} = \mathbf{C\Psi}$ (глава 3)
$\mathbf{\Theta}$ матрица функций-кандидатов для SINDy (раздел 7.3)
$\mathbf{\Gamma}$ матрица производных функций-кандидатов для SINDy (раздел 7.3)
$\mathbf{\Xi}$ матрица коэффициентов функций-кандидатов для SINDy (раздел 7.3)
$\mathbf{\Xi}$ матрица нелинейных снимков для DEIM (раздел 12.5)
$\mathbf{\Lambda}$ диагональная матрица собственных значений
$\mathbf{\Upsilon}$ матрица входных снимков, $\mathbf{\Upsilon} \in \mathbb{R}^{q \times m}$
$\mathbf{\Phi}$ матрица DMD-мод, $\mathbf{\Phi} \triangleq \mathbf{X}'\mathbf{V}'\mathbf{\Sigma}^{-1}\mathbf{W}$

$\boldsymbol{\Psi}$ ортонормированный базис (например, моды Фурье или POD-моды)

### *Тензоры*

$(\mathcal{A}, \mathcal{B}, \mathcal{M})$ тензоры $N$-мерных массивов размера $I_1 \times I_2 \times \cdots \times I_N$

### *Нормы*

$||\cdot||_0$ псевдонорма $\ell_0$ вектора $\mathbf{x}$: количество ненулевых элементов $\mathbf{x}$

$||\cdot||_1$ норма $\ell_1$ вектора $\mathbf{x}$: $||\mathbf{x}||_1 = \sum_{i=1}^{n} |x_i|$

$||\cdot||_2$ норма $\ell_2$ вектора $\mathbf{x}$: $||\mathbf{x}||_2 = \sqrt{\sum_{i=1}^{n} (x_i^2)}$

$||\cdot||_2$ норма $\ell_2$ матрицы $\mathbf{X}$: $||\mathbf{X}||_2 = \max_x \dfrac{||\mathbf{X}\mathbf{x}||_2}{||\mathbf{x}||_2}$

$||\cdot||_F$ норма Фробениуса матрицы $\mathbf{X}$: $||\mathbf{X}||_F = \sqrt{\sum_{i=1}^{n} \sum_{j-1}^{m} |X_{ij}|^2}$

$||\cdot||_*$ ядерная норма матрицы $\mathbf{X}$: $||\mathbf{X}||_* = \text{trace}\left(\sqrt{\mathbf{X}^*\mathbf{X}}\right) = \sum_{i=1}^{m} \sigma_i$ (для $m \leq n$)

$\langle \cdot, \cdot \rangle$ скалярное произведение. Для функций $\langle f(x), g(x) \rangle = \int_{-\infty}^{\infty} f(x) g^*(x) dx$

$\langle \cdot, \cdot \rangle$ скалярное произведение. Для векторов $\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u}^* \mathbf{v}$

### *Операторы, функции и отображения*

$\mathcal{F}$ преобразование Фурье

$\mathbf{F}$ отображение динамической системы с дискретным временем

$\mathbf{F}_t$ дискретное потоковое отображение динамической системы

$\mathbf{f}$ динамическая система с непрерывным временем

$G$ преобразование Габора

$\mathbf{G}$ передаточная функция, отображающая входы на выходы (глава 8)

$g$ скалярная функция измерения $\mathbf{x}$

$\mathbf{g}$ скалярная функция измерения $\mathbf{x}$

$J$ функция стоимости для регулирования

$\ell$ функция потерь в методе опорных векторов (глава 5)

$\mathcal{K}$ оператор Купмана (с непрерывным временем)

$\mathcal{K}_t$ оператор Купмана, ассоциированный с потоковым отображением

$\mathcal{L}$ преобразование Лапласа

$\mathbf{L}$ передаточная функция контура (глава 8)

$\mathbf{L}$ линейное дифференциальное уравнение в частных производных (главы 11 и 12)

$\mathbf{N}$ нелинейное дифференциальное уравнение в частных производных

$O$ порядок величины

$\mathbf{S}$ функция чувствительности (глава 8)

$\mathbf{T}$ дополнительная функция чувствительности (глава 8)

$\mathcal{W}$ вейвлет-преобразование

$\mu$ несогласованность между матрицей измерений $\mathbf{C}$ и базисом $\boldsymbol{\Psi}$

$\kappa$ число обусловленности

$\varphi$ собственная функция Купмана

$\nabla$ оператор градиента

$*$ оператор свертки

### *Наиболее употребительные акронимы*

БПФ быстрое преобразование Фурье
ГО глубокое обучение
ОДУ обыкновенное дифференциальное уравнение
СНС сверточная нейронная сеть
УрЧП дифференциальное уравнение в частных производных
DMD разложение по динамическим модам (dynamic mode decomposition)
PCA метод главных компонент (principal components analysis)
POD собственное ортогональное разложение (proper orthogonal decomposition)
ROM модель пониженного порядка (reduced order model)
SVD сингулярное разложение (singular value decomposition)

### *Прочие акронимы*

ДПФ дискретное преобразование Фурье
ИНС искусственная нейронная сеть
ЛДА линейный дискриминантный анализ
НУШ нелинейное уравнение Шрёдингера
ОПФ оконное преобразование Фурье (short time Fourier transform)
ПИД пропорционально-интегрально-дифференцирующий регулятор
РНС рекуррентная нейронная сеть
СГС стохастический градиентный спуск
ADM метод переменных направлений (alternating directions method)
AIC информационный критерий Акаике (Akaike information criterion)
ALM расширенный метод множителей Лагранжа (augmented Lagrange multiplier)
ARMA авторегрессионное скользящее среднее (autoregressive moving average)
ARMAX авторегрессионное скользящее среднее с экзогенным входом (autoregressive moving average with exogenous input)
BIC байесовский информационный критерий (Bayesian information criterion)
BPOD сбалансированное собственное ортогональное разложение (balanced proper orthogonal decomposition)
CCA канонический корреляционный анализ (canonical correlation analysis)
CFD вычислительная гидродинамика (computational fluid dynamics)
CoSaMP согласованное преследование со сжатой выборкой (compressive sampling matching pursuit)
CWT непрерывное вейвлет-преобразование (continuous wavelet transform)
DCT дискретное косинусное преобразование (discrete cosine transform)
DEIM дискретный эмпирический метод интерполяции (discrete empirical interpolation method)
DMDc разложение по динамическим модам с управлением (dynamic mode decomposition with control)

| | |
|---|---|
| DMDc | разложение по динамическим модам с управлением (dynamic mode decomposition with control) |
| DNS | прямое численное моделирование (direct numerical simulation) |
| DWT | дискретное вейвлет-преобразование |
| ECOG | электрокортикография (electrocorticography) |
| eDMD | расширенное DMD (extended DMD) |
| EIM | эмпирический метод интерполяции (empirical interpolation method) |
| EM | математическое ожидание-максимизация (expectation maximization) |
| EOF | эмпирические ортогональные функции (empirical orthogonal functions) |
| ERA | алгоритм реализации собственной системы (eigensystem realization algorithm) |
| ESC | управление с поиском экстремума (extremum-seeking control) |
| GMM | модель гауссовой смеси (Gaussian mixture model) |
| HAVOK | ганкелево альтернативное представление оператора Купмана (Hankel alternative view of Koopman) |
| ICA | метод независимых компонент (independent component analysis) |
| JL | Джонсона–Линденштраусса (Johnson–Lindenstrauss) |
| KL | Кульбака–Лейблера (Kullback–Leibler) |
| KLT | преобразование Карунена–Лоэва (Karhunen–Loève transform) |
| LAD | наименьшее абсолютное отклонение (least absolute deviations) |
| LASSO | оператор наименьшего абсолютного сжатия и выборки (least absolute shrinkage and selection operator) |
| LQE | линейно-квадратичная модель оценки (linear quadratic estimator) |
| LQG | линейно-квадратичный гауссов регулятор (linear quadratic Gaussian controller) |
| LQR | линейно-квадратичный регулятор |
| LTI | линейная стационарная система (linear time invariant system) |
| MIMO | с несколькими входами и несколькими выходами (multiple input, multiple output) |
| MLC | управление на основе машинного обучения (machine learning control) |
| MPE | оценка отсутствующей точки (missing point estimation) |
| mrDMD | многомасштабное разложение по динамическим модам (multiresolution dynamic mode decomposition) |
| NARMAX | нелинейная авторегрессионная модель с экзогенными входами (nonlinear autoregressive model with exogenous inputs) |
| OKID | идентификация наблюдателей с помощью фильтра Калмана (observer Kalman filter identification) |
| PBH | критерий Попова–Белевича–Хаутуса (Popov–Belevitch–Hautus test) |
| PCP | преследование главных компонент (principal component pursuit) |
| PDE-FIND | функциональная идентификация нелинейной динамики с уравнениями в частных производных (partial differential equation functional identification of nonlinear dynamics) |

| | |
|---|---|
| PDF | функция распределения вероятностей (probability distribution function) |
| PIV | анемометрия по изображениям частиц (particle image velocimetry) |
| RIP | свойство ограниченной изометрии (restricted isometry property) |
| RKHS | гильбертово пространство с воспроизводящим ядром (reproducing kernel Hilbert space) |
| RPCA | робастный метод главных компонент (robust principal components analysis) |
| rSVD | рандомизированное SVD (randomized SVD) |
| SINDy | разреженная идентификация нелинейных систем (sparse identification of nonlinear dynamics) |
| SISO | с одним входом и одним выходом (single input, single output) |
| SRC | разреженное представление для классификации (sparse representation for classification) |
| SSA | анализ сингулярного спектра (singular spectrum analysis) |
| STLS | последовательный метод наименьших квадратов с порогом (sequential thresholded least-squares) |
| SVM | метод опорных векторов (support vector machine) |
| TICA | метод независимых компонент с задержкой (time-lagged independent component analysis) |
| VAC | вариационный подход к конформационной динамике (variational approach of conformation dynamics) |

Часть I

# ПОНИЖЕНИЕ РАЗМЕРНОСТИ И ПРЕОБРАЗОВАНИЯ

# Глава 1

# Сингулярное разложение (SVD)

Сингулярное разложение (SVD) – одно из самых важных разложений матриц, появившихся в компьютерную эру, оно лежит в основе почти всех методов обработки данных, рассматриваемых в этой книге. SVD – это численно устойчивое разложение матрицы, которое применимо для самых разных целей и при этом гарантированно существует. Мы будем использовать SVD для получения низкоранговых аппроксимаций матриц и для вычисления псевдообращения неквадратных матриц, т. е. нахождения решения системы уравнений вида $\mathbf{Ax} = \mathbf{b}$. Еще одно важное применение SVD находит как алгоритм, лежащий в основе метода главных компонент (principal component analysis – PCA), идея которого – выделение статистически значимых факторов из многомерных данных. Комбинация SVD+PCA применяется к решению широкого круга научно-технических задач.

В каком-то смысле SVD является обобщением быстрого преобразования Фурье (БПФ) – темы следующей главы. Учебники прикладной математики часто начинаются с описания БПФ, поскольку это основа многих классических аналитических и численных результатов. Однако БПФ работает в идеализированной постановке, тогда как SVD – более общая техника, основанная на анализе предъявленных данных. Поскольку эта книга посвящена данным, мы начнем с сингулярного разложения, которое можно рассматривать как базис, созданный *специально* под конкретные данные, в отличие от БПФ, предоставляющего *общий* базис.

Во многих предметных областях сложные системы порождают данные, которые естественно организуются в виде больших матриц или, более общо, массивов. Например, временной ряд, получающийся в результате эксперимента или имитационного моделирования, можно представить в виде матрицы, каждый столбец которой содержит все измерения в один момент времени. Если данные в каждый момент времени многомерные, как при построении трехмерной модели погоды высокой разрешающей способности, то их можно *разгладить*, преобразовав в длинный вектор-столбец, и собрать такие столбцы в большую матрицу. Аналогично значения пикселей полутонового изображения можно сохранить в виде матрицы или преобразовать в длинные векторы-столбцы, которые в совокупности образуют матрицу, представляющую кадры фильма. Примечательно, что данные, порождаемые

такими системами, обычно имеют низкий ранг. Это означает, что можно выделить несколько доминирующих паттернов, которые объясняют многомерные данные. SVD как раз и является численно устойчивым и эффективным методом обнаружения таких паттернов в данных.

## 1.1. Общие сведения

Сейчас мы вкратце опишем SVD и разовьем интуицию, которая поможет продемонстрировать SVD на ряде мотивирующих примеров. SVD лежит в основе многих других методов, описанных в этой книге, в т. ч. методов классификации в главе 5, разложения по динамическим модам (DMD) в главе 7 и собственного ортогонального разложения (POD) в главе 11. Подробно математические свойства обсуждаются в последующих разделах.

Высокая размерность – самая главная трудность при обработке данных, порождаемых сложными системами. Такие наборы данных могут включать аудио, изображения и видео. Данные могут также генерироваться физической системой, например записи электрической активности мозга или измерения скорости течения жидкости в модели либо в эксперименте. Замечено, что во многих естественно возникающих системах в данных присутствуют преобладающие паттерны, характеризуемые аттрактором или многообразием низкой размерности [252, 251].

Рассмотрим, например, типичное изображение – оно содержит много измерений (пикселей) и, стало быть, является точкой в многомерном векторном пространстве. Однако большинство изображений отлично сжимаются, т. е. существенную информацию можно представить подпространством гораздо более низкой размерности. Сжимаемость изображений будет подробно обсуждаться на страницах этой книги. Сложные гидрогазодинамические системы, например атмосфера Земли или турбулентная спутная струя за кормой автомобиля, также дают убедительные примеры низкоразмерной структуры, скрывающейся в пространстве состояний высокой размерности. Хотя в точных моделях поведения жидкости или газа количество степеней свободы исчисляется миллионами и миллиардами, часто в потоке можно выделить доминирующие когерентные структуры, например периодические завихрения за кормой автомобиля или ураганы в атмосфере.

SVD предлагает систематический способ нахождения низкоразмерной аппроксимации данных высокой размерности в терминах доминирующих паттернов. Эту технику можно назвать *управляемой данными*, поскольку для обнаружения паттернов нужны только данные, без привлечения экспертных знаний или интуиции. Метод SVD обладает численной устойчивостью и дает иерархическое представление данных в новой системе координат, определяемой доминирующими корреляциями внутри данных. Кроме того, гарантируется, что сингулярное разложение, в отличие от спектрального, существует для любой матрицы.

SVD имеет много полезных приложений, помимо понижения размерности данных. Этот метод используется для вычисления псевдообратной матрицы для неквадратных матриц и тем самым позволяет найти решения недоопре-

деленных или переопределенных матричных уравнений вида $\mathbf{Ax} = \mathbf{b}$. Также мы будем использовать SVD для очистки наборов данных от шумов. Не менее важно, что SVD позволяет охарактеризовать входную и выходную геометрию линейного отображения векторных пространств. Все эти приложения будут рассмотрены в данной главе и дадут нам возможность развить интуитивное понимание матриц и данных высокой размерности.

# Определение SVD

В общем случае нас интересует анализ большого набора данных $\mathbf{X} \in \mathbb{C}^{n \times m}$:

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}. \tag{1.1}$$

Столбцы $\mathbf{x}_k \in \mathbb{C}^n$ могут представлять измерения, полученные в процессе моделирования или эксперимента. Например, это могут быть изображения, вытянутые в векторы-столбцы длиной, равной количеству пикселей в изображении. Векторы-столбцы могут также представлять состояние нестационарной физической системы, например скорости течения жидкости в выбранных точках, множество измерений электрической активности мозга или состояние метеорологической модели с разрешением 1 км.

Индекс $k$ – это метка, обозначающая $k$-й набор измерений. Во многих примерах из этой книги $\mathbf{X}$ будет содержать *временные ряды* данных, а $\mathbf{x}_k = \mathbf{x}(k\Delta t)$. Зачастую *размерность состояния* $n$ очень велика – порядка миллионов или миллиардов степеней свободы. Столбцы иногда называют (моментальными) *снимками*, а $m$ обозначает количество снимков в $\mathbf{X}$. Для многих систем $n \gg m$, поэтому получается *высокая и тощая* матрица, в отличие от *низкой и толстой* в случае, когда $n \ll m$.

SVD – это однозначно определенное разложение, существующее для любой комплексной матрицы $\mathbf{X} \in \mathbb{C}^{n \times m}$:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*, \tag{1.2}$$

где $\mathbf{U} \in \mathbb{C}^{n \times n}$ и $\mathbf{V} \in \mathbb{C}^{m \times m}$ – *унитарные* матрицы[1] с ортонормированными столбцами, а $\mathbf{\Sigma} \in \mathbb{R}^{n \times m}$ – матрица, диагональные элементы которой вещественны и неотрицательны, а все остальные равны нулю. Здесь символ $^*$ обозначает операцию комплексного сопряжения и транспонирования[2]. Ниже мы увидим, что унитарность $\mathbf{U}$ и $\mathbf{V}$ важна и часто используется.

Если $n \geq m$, то на диагонали матрицы $\mathbf{\Sigma}$ расположено не более $m$ ненулевых элементов, и ее можно записать в виде $\mathbf{\Sigma} = \begin{bmatrix} \hat{\mathbf{\Sigma}} \\ \mathbf{0} \end{bmatrix}$. Поэтому $\mathbf{X}$ можно *точно* представить, воспользовавшись *экономной* формой SVD:

---

[1] Квадратная матрица $\mathbf{U}$ называется унитарной, если $\mathbf{U}\mathbf{U}^* = \mathbf{U}^*\mathbf{U} = \mathbf{I}$.

[2] Для вещественных матриц эта операция совпадает с обычным транспонированием: $\mathbf{X}^* = \mathbf{X}^T$.

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \begin{bmatrix}\hat{\mathbf{U}} & \hat{\mathbf{U}}^\perp\end{bmatrix}\begin{bmatrix}\hat{\mathbf{\Sigma}} \\ \mathbf{0}\end{bmatrix}\mathbf{V}^* = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\mathbf{V}^*. \tag{1.3}$$

Полное и экономное SVD показаны на рис. 1.1. На столбцы матрицы $\hat{\mathbf{U}}^\perp$ натянуто векторное пространство, дополнительное и ортогональное к натянутому на столбцы $\hat{\mathbf{U}}$. Столбцы $\mathbf{U}$ называются *левыми сингулярными векторами* $\mathbf{X}$, а столбцы $\mathbf{V}$ – *правыми сингулярными векторами*. Диагональные элементы $\hat{\mathbf{\Sigma}} \in \mathbb{C}^{m\times m}$ называются *сингулярными значениями*, они расположены в порядке убывания. Ранг $\mathbf{X}$ равен количеству ненулевых сингулярных значений.

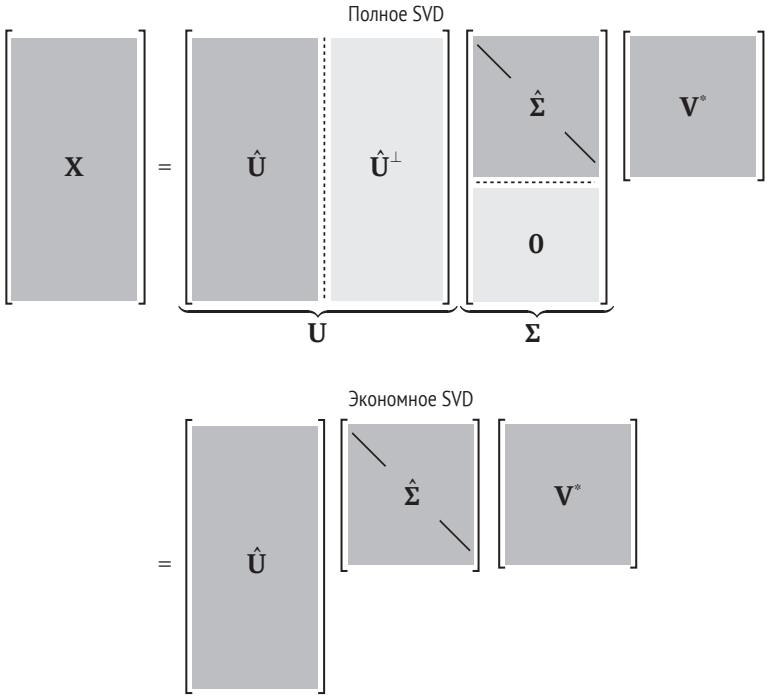


**Рис. 1.1** ❖ Схема матриц в полном и экономном SVD

# Вычисление SVD

SVD – краеугольный камень вычислительных методов в науке и технике, а численная реализация SVD важна и познавательна с математической точки зрения. Впрочем, большинство стандартных численных реализаций хорошо отработаны, к ним существует простой интерфейс из многих современных языков программирования, что позволяет нам абстрагироваться от деталей вычисления SVD. Как правило, мы будем просто использовать SVD как часть решения более крупной задачи и примем существование эффективных и устойчивых численных алгоритмов как данность. В последующих разделах мы продемон-

стрируем использование SVD на разных языках программирования, а также обсудим большинство стандартных стратегий вычисления и их ограничений. На тему вычисления SVD есть немало важных результатов [212, 106, 211, 292, 238]. Более полное обсуждение вычислительных аспектов можно найти в работе [214]. Для вычисления SVD очень больших матриц все чаще используют рандомизированные алгоритмы, этот вопрос обсуждается в разделе 1.8.

***MATLAB***. В MATLAB SVD вычисляется прямолинейно:

```
>>X = randn(5,3);   % создать случайную матрицу 5×3
>>[U,S,V] = svd(X); % сингулярное разложение
```

Для неквадратных матриц **X** экономное SVD эффективнее:

```
>>[Uhat,Shat,V] = svd(X,'econ'); % экономное SVD
```

***Python***

```
>>> import numpy as np
>>> X = np.random.rand(5, 3) % create random data matrix
>>> U, S, V = np.linalg.svd(X,full_matrices=True) % full SVD
>>> Uhat, Shat, Vhat = np.linalg.svd(X, full_matrices=False)
    % economy SVD
```

***R***

```
> X <- replicate(3, rnorm(5))
> s <- svd(X)
> U <- s$u
> S <- diag(s$d)
> V <- s$v
```

***Mathematica***

```
In:= X=RandomReal[{0,1},{5,3}]
In:= {U,S,V} = SingularValueDecomposition[X]
```

***Другие языки***

SVD реализовано и на других языках, например Fortran и C++. На самом деле большинство реализаций SVD основаны на библиотеке LAPACK (Linear Algebra Package) [13], написанной на Fortran. В LAPACK подпрограмма вычисления SVD называется **DGESVD**, а она уже обернута функциями C++ в библиотеках **Armadillo** и **Eigen**.

# Историческая справка

SVD имеет давнюю и богатую историю, от ранних работ, в которых был заложен теоретический фундамент, до современных исследований по численной устойчивости и эффективности. Отличный исторический обзор приведен в работе Stewart [502], где описан общий контекст и многие важные детали. В этом обзоре много внимания уделено ранним теоретическим работам Бельтрами и Джордана (1873), Сильвестра (1889), Шмидта (1907) и Вейля (1912). Обсуждаются также более поздние работы, включая основополагаю-

щие труды по вычислительной стороне проблемы Голуба с сотрудниками [212, 211]. Кроме того, в современных учебниках [524, 17, 316] имеются прекрасно написанные главы, посвященные SVD.

## Использование в этой книге и предположения о подготовке читателей

SVD – основа многих методов понижения размерности. К ним относятся метод главных компонент (PCA) в статистике [418, 256, 257], преобразование Карунена–Лоэва (KLT) [280, 340], эмпирические ортогональные функции (EOF) в изучении климата [344], собственное ортогональное разложение (POD) в гидродинамике [251] и канонический корреляционный анализ (CCA) [131]. Хотя многие из этих методов разрабатывались независимо для разных областей знания, различаются они только способами сбора и предварительной обработки данных. В работе Gerbrands [204] приведено превосходное обсуждение связи между SVD, KLT и PCA.

SVD широко используется для идентификации систем и в теории управления для получения моделей пониженного порядка, сбалансированных в том смысле, что состояния иерархически упорядочены в терминах доступности наблюдению и управляемости с помощью приводов [388].

В этой главе предполагается, что читатель знаком с линейной алгеброй и имеет некоторый опыт применения вычислительных методов. Если необходимо освежить познания, то существует много отличных книг по численной линейной алгебре, в т. ч. с обсуждением SVD, например [524, 17, 316].

# 1.2. Аппроксимация матриц

Пожалуй, самым полезным и определяющим свойством SVD является тот факт, что оно дает *оптимальную* низкоранговую аппроксимацию матрицы $\mathbf{X}$. На самом деле SVD позволяет построить *иерархию* низкоранговых аппроксимаций, поскольку для получения аппроксимации ранга $r$ нужно просто оставить первые $r$ сингулярных значений и векторов, а остальные отбросить.

Шмидт обобщил SVD на пространства функций и доказал теорему, устанавливающую, что усеченное SVD является оптимальной низкоранговой аппроксимацией исходной матрицы $\mathbf{X}$ [476]. Теорема Шмидта была заново открыта в работе Eckart and Young [170], поэтому иногда ее называют теоремой Эккарта–Янга.

**Теорема 1 (Eckart–Young [170]).** *Оптимальную в смысле наименьших квадратов аппроксимацию* $\mathbf{X}$ *ранга* $r$ *дает усеченное SVD* $\tilde{\mathbf{X}}$ *ранга* $r$:

$$\underset{\tilde{\mathbf{X}} \text{ такое, что } \operatorname{rank}(\tilde{\mathbf{X}})=r}{\operatorname{argmin}} ||\mathbf{X} - \tilde{\mathbf{X}}||_F = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*. \tag{1.4}$$

*Здесь* $\tilde{\mathbf{U}}$ *и* $\tilde{\mathbf{V}}$ *обозначают матрицы, образованные первыми* $r$ *столбцами* $\mathbf{U}$ *и* $\mathbf{V}$, *а* $\tilde{\mathbf{\Sigma}}$ *содержит левый верхний блок* $\mathbf{\Sigma}$ *размера* $r \times r$. $||\cdot||_F$ *– норма Фробениуса.*

Согласно этой нотации базис усеченного SVD (и аппроксимирующая матрица $\tilde{\mathbf{X}}$) обозначается $\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$. Поскольку $\mathbf{\Sigma}$ – диагональная матрица, SVD-аппроксимация ранга $r$ может быть представлена суммой $r$ матриц ранга 1:

$$\tilde{\mathbf{X}} = \sum_{k=1}^{r} \sigma_k \mathbf{u}_k \mathbf{v}_k^* = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^* + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^* + \cdots + \sigma_r \mathbf{u}_r \mathbf{v}_r^*. \tag{1.5}$$

Это так называемое *диадическое* суммирование. Для любого ранга $r$ не существует лучшей аппроксимации $\mathbf{X}$ в смысле нормы $\ell_2$, чем усеченная SVD-аппроксимация $\tilde{\mathbf{X}}$. Таким образом, данные высокой размерности хорошо описываются несколькими доминирующими паттернами, определяемыми столбцами $\tilde{\mathbf{U}}$ и $\tilde{\mathbf{V}}$.

Это важное свойство SVD, к которому мы будем много раз возвращаться. Есть многочисленные примеры наборов данных, содержащих измерения высокой размерности, которые приводят к большой матрице $\mathbf{X}$. Однако в данных часто присутствуют доминирующие паттерны низкой размерности, и базис усеченного SVD $\tilde{\mathbf{U}}$ определяет преобразование координат из пространства измерений высокой размерности в пространство паттернов низкой размерности. В результате уменьшается размер и размерность больших наборов данных, а значит, открывается возможность для визуализации и анализа. Наконец, многие системы, рассматриваемые в этой книге, *динамические* (см. главу 7), а базис SVD дает иерархию мод, характеризующую наблюдаемый аттрактор, на который мы можем спроецировать динамическую систему низкой размерности для получения моделей пониженного порядка (см. главу 12).

## Усечение

Усеченное SVD показано на рис. 1.2, где $\tilde{\mathbf{U}}$, $\tilde{\mathbf{\Sigma}}$ и $\tilde{\mathbf{V}}$ обозначают усеченные матрицы. Если $\mathbf{X}$ – матрица неполного ранга, то некоторые сингулярные значения в $\tilde{\mathbf{\Sigma}}$ могут быть равны нулю, и тогда усеченное SVD остается точным. Однако если $r$ меньше числа ненулевых сингулярных значений (т. е. ранга $\mathbf{X}$), то усеченное SVD является всего лишь аппроксимацией $\mathbf{X}$:

$$\mathbf{X} \approx \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*. \tag{1.6}$$

Есть много способов выбора ранга усеченной матрицы $r$, они обсуждаются в разделе 1.7. Если мы требуем, чтобы усеченная матрица содержала все ненулевые сингулярные значения, то равенство $\mathbf{X} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$ точное.

## Пример: сжатие изображения

Проиллюстрируем идею аппроксимации матриц на простом примере: сжатие изображения. Тема, которая красной нитью проходит через всю книгу, – наличие в больших наборах данных паттернов, благодаря чему возможны низкоранговые представления. Естественные изображения дают простой

и интуитивно понятный пример такой внутренне присущей *сжимаемости*. Полутоновое изображение можно рассматривать как вещественную матрицу $\mathbf{X} \in \mathbb{R}^{n \times m}$, где $n$ и $m$ – числа пикселей по вертикали и по горизонтали соответственно[1]. В зависимости от базиса представления (пространство пикселей, частотная область в смысле преобразования Фурье, преобразованные с помощью SVD координаты) изображение может иметь очень компактные аппроксимации.

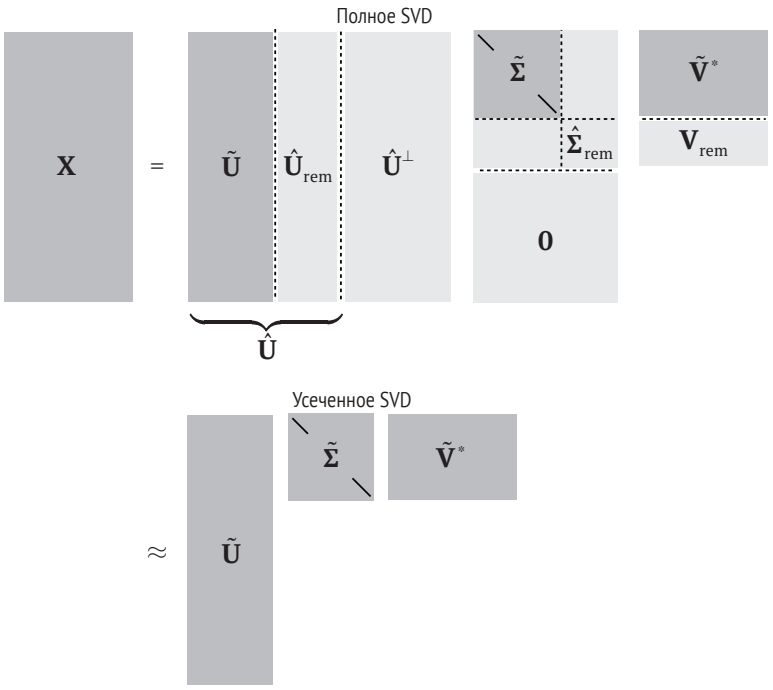


**Рис. 1.2** ❖ Схема усеченного SVD.
Нижний индекс «rem» обозначает остаток $\hat{\mathbf{U}}, \hat{\mathbf{\Sigma}}$ или $\mathbf{V}$ после усечения

Рассмотрим изображение собаки по кличке Мордекай на снегу (рис. 1.3). Его размер 2000×1500 пикселей. Мы можем вычислить SVD этого изображения и нанести на график сингулярные значения (рис. 1.4). На рис. 1.3 показаны приближенные матрицы $\tilde{\mathbf{X}}$ для разных значений $r$. При $r = 100$ реконструированное изображение вполне точное, а сингулярные значения отражают почти 80 % неоднородности изображения. Усечение SVD приводит к сжатию исходного изображения, поскольку в $\tilde{\mathbf{U}}$, $\tilde{\mathbf{\Sigma}}$ и $\tilde{\mathbf{V}}$ нужно хранить только первые 100 столбцов $\mathbf{U}$ и $\mathbf{V}$ плюс первые 100 диагональных элементов $\mathbf{\Sigma}$.

[1] Размер изображения часто задают, указывая сначала размер по горизонтали, а затем по вертикали, т. е. $\mathbf{X}^T \in \mathbb{R}^{m \times n}$, но мы будем придерживаться противоположного соглашения, совпадающего с общепринятым порядком обозначения размера матрицы.

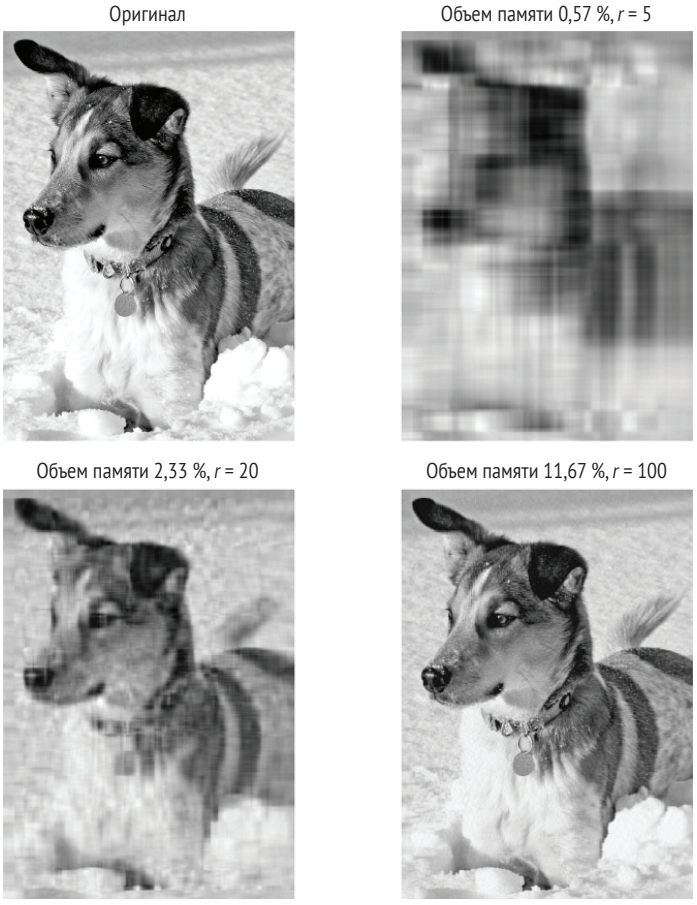


**Рис. 1.3** ❖ Сжатие изображения собаки Мордекая на снегу путем усечения SVD с различными значениями ранга *r*. Разрешение исходного изображения 2000×1500

Сначала загрузим изображение:

```
A=imread('../DATA/dog.jpg');
X=double(rgb2gray(A)); % преобразовать RBG в полутоновое, 256 бит->double.
nx = size(X,1); ny = size(X,2);
imagesc(X), axis off, colormap gray
```

и вычислим SVD:

```
[U,S,V] = svd(X);
```

Затем вычислим приближенную матрицу, используя усеченные SVD с разными рангами (r = 5, 20, 100):

```
for r=[5 20 100]; % Truncation value
    Xapprox = U(:,1:r)*S(1:r,1:r)*V(:,1:r)'; % Approx. image
```

```
    figure, imagesc(Xapprox), axis off
    title(['r=',num2str(r,'%d'),']);
end
```

Наконец, построим графики сингулярных значений и суммарной энергии, изображенные на рис. 1.4.

```
subplot(1,2,1), semilogy(diag(S),'k')
subplot(1,2,2), plot(cumsum(diag(S))/sum(diag(S)),'k')
```

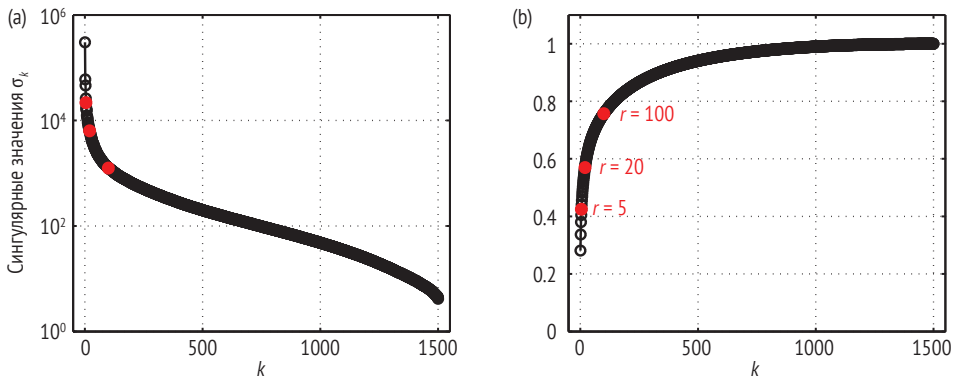


**Рис. 1.4** ❖ (a) Сингулярные значения $\sigma_k$. (b) Суммарная энергия в первых $k$ модах

# 1.3. Математические свойства и манипуляции

Опишем важные математические свойства SVD, в т. ч. геометрические интерпретации унитарных матриц **U** и **V**, а также обсудим SVD в терминах доминирующих корреляций, присутствующих в данных **X**. Связь между SVD и корреляциями данных мы объясним в разделе 1.5, посвященном методу главных компонент.

## Интерпретация с привлечением доминирующих корреляций

Сингулярное разложение тесно связано с задачей о собственных значениях, в которой фигурируют корреляционные матрицы $\mathbf{X}\mathbf{X}^*$ и $\mathbf{X}^*\mathbf{X}$, показанные на рис. 1.5 для конкретного изображения, а на рис. 1.6 и 1.7 для матриц общего вида. Подставив (1.3) в $\mathbf{X}\mathbf{X}^*$ и $\mathbf{X}^*\mathbf{X}$, получаем:

$$\mathbf{X}\mathbf{X}^* = \mathbf{U}\begin{bmatrix}\hat{\mathbf{\Sigma}}\\ \mathbf{0}\end{bmatrix}\mathbf{V}^*\mathbf{V}\begin{bmatrix}\hat{\mathbf{\Sigma}} & \mathbf{0}\end{bmatrix}\mathbf{U}^* = \mathbf{U}\begin{bmatrix}\hat{\mathbf{\Sigma}}^2 & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\mathbf{U}^*; \tag{1.7a}$$

$$\mathbf{X}^*\mathbf{X} = \mathbf{V}\begin{bmatrix}\hat{\mathbf{\Sigma}} & \mathbf{0}\end{bmatrix}\mathbf{U}^*\mathbf{U}\begin{bmatrix}\hat{\mathbf{\Sigma}} \\ \mathbf{0}\end{bmatrix}\mathbf{V}^* = \mathbf{V}\hat{\mathbf{\Sigma}}^2\mathbf{V}^*. \tag{1.7b}$$

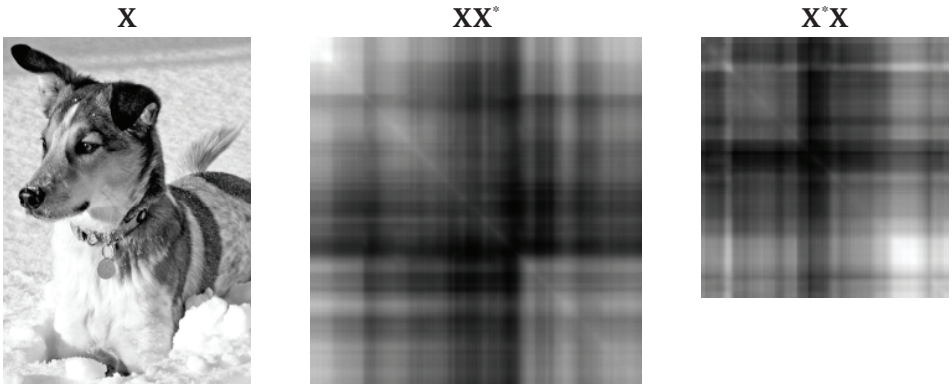


**Рис. 1.5** ❖ Корреляционные матрицы $\mathbf{XX}^*$ и $\mathbf{X}^*\mathbf{X}$ для матрицы **X**, полученной из изображения собаки. Заметим, что обе корреляционные матрицы симметричны

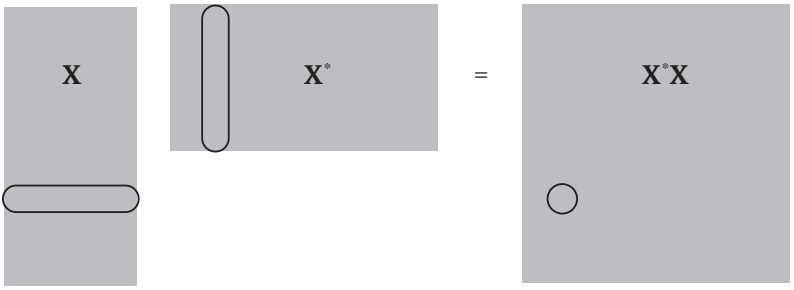


**Рис. 1.6** ❖ Корреляционная матрица $\mathbf{XX}^*$ получена взятием скалярных произведений строк **X**

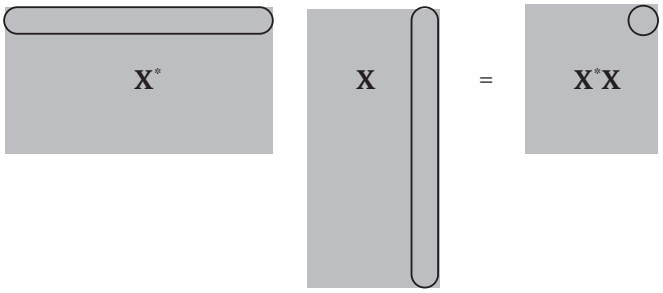


**Рис. 1.7** ❖ Корреляционная матрица $\mathbf{X}^*\mathbf{X}$ получена взятием скалярных произведений столбцов **X**

Учитывая, что **U** и **V** унитарны, **U**, **Σ** и **V** являются решениями следующих задач на нахождение собственных значений:

$$\mathbf{X}\mathbf{X}^{*}\mathbf{U} = \mathbf{U}\begin{bmatrix}\hat{\mathbf{\Sigma}}^{2} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}; \tag{1.8a}$$

$$\mathbf{X}^{*}\mathbf{X}\mathbf{V} = \mathbf{V}\hat{\mathbf{\Sigma}}^{2}. \tag{1.8b}$$

Иными словами, любое ненулевое сингулярное значение **X** является положительным квадратным корнем из какого-то собственного значения матриц $\mathbf{X}^{*}\mathbf{X}$ и $\mathbf{X}\mathbf{X}^{*}$, которые имеют одинаковые множества собственных значений. Отсюда следует, что если **X** самосопряженная (т. е. $\mathbf{X} = \mathbf{X}^{*}$), то сингулярные значения **X** равны абсолютным величинам собственных значений **X**.

В результате мы получаем интуитивно понятную интерпретацию SVD: столбцы **U** являются собственными векторами корреляционной матрицы $\mathbf{X}\mathbf{X}^{*}$, а столбцы **V** – собственными векторами $\mathbf{X}^{*}\mathbf{X}$. Мы упорядочиваем сингулярные значения в порядке убывания абсолютной величины, поэтому порядок столбцов **U** отражает, какую часть корреляции между столбцами **X** они улавливают; аналогично **V** улавливает корреляцию между строками **X**.

## Метод моментальных снимков

На практике часто невозможно построить матрицу $\mathbf{X}\mathbf{X}^{*}$, поскольку размерность состояния $n$ слишком велика; что уж говорить о нахождении собственных значений. Если **X** состоит из миллиона элементов, то число элементов $\mathbf{X}\mathbf{X}^{*}$ равно триллиону. В 1987 году Сирович (Sirovich) заметил, что можно не вычислять эту большую матрицу, а найти первые $m$ столбцов **U** с помощью метода, получившего название «метод (моментальных) снимков» [490].

Вместо того чтобы вычислять спектральное разложение $\mathbf{X}\mathbf{X}^{*}$ для получения левых сингулярных векторов **U**, мы вычисляем только спектральное разложение гораздо меньшей и простой для работы матрицы $\mathbf{X}^{*}\mathbf{X}$. Затем из (1.8b) находим **V** и $\hat{\mathbf{\Sigma}}$. Если у $\hat{\mathbf{\Sigma}}$ имеются нулевые сингулярные значения, то мы оставляем только ее часть $\tilde{\mathbf{\Sigma}}$, соответствующую $r$ ненулевым значениям, и соответственные столбцы $\tilde{\mathbf{V}}$ матрицы **V**. Зная эти матрицы, можно следующим образом аппроксимировать матрицу $\tilde{\mathbf{U}}$, состоящую из первых $r$ столбцов **U**:

$$\tilde{\mathbf{U}} = \mathbf{X}\tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}. \tag{1.9}$$

## Геометрическая интерпретация

Столбцы матрицы **U** образуют ортонормированный базис пространства столбцов **X**. Аналогично столбцы **V** образуют ортонормированный базис пространства строк **X**. Если столбцы **X** содержат пространственные измерения в разные моменты времени, то **U** кодирует пространственные паттерны, а **V** – временные паттерны.

Особенно полезным делает SVD тот факт, что $\mathbf{U}$ и $\mathbf{V}$ – *унитарные* матрицы, так что $\mathbf{U}\mathbf{U}^* = \mathbf{U}^*\mathbf{U} = \mathbf{I}_{n\times n}$ и $\mathbf{V}\mathbf{V}^* = \mathbf{V}^*\mathbf{V} = \mathbf{I}_{m\times m}$. Это означает, что для решения системы уравнений с матрицей $\mathbf{U}$ или $\mathbf{V}$ нужно просто умножить обе части на транспонированную матрицу. Сложность этой операции составляет $O(n^2)$, в отличие от традиционных методов обращения матрицы общего вида, имеющих сложность $O(n^3)$. Как отмечено в предыдущем разделе и в работе [57], SVD тесно связано со спектральными свойствами компактных самосопряженных операторов $\mathbf{X}\mathbf{X}^*$ и $\mathbf{X}^*\mathbf{X}$.

Сингулярное разложение $\mathbf{X}$ можно геометрически интерпретировать, рассмотрев отображение гиперсферы $S^{n-1} \triangleq \{\mathbf{x} \mid ||\mathbf{x}||_2 = 1\} \subset \mathbb{R}^n$ в эллипсоид, $\{\mathbf{y} \mid \mathbf{y} = \mathbf{X}\mathbf{x}$ для $\mathbf{x} \in S^{n-1}\} \subset \mathbb{R}^m$ посредством умножения на $\mathbf{X}$. Графически это показано на рис. 1.8 для сферы в $\mathbb{R}^3$ и отображения посредством умножения на $\mathbf{X}$ с тремя ненулевыми сингулярными значениями. Поскольку умножение на матрицу – линейное отображение, достаточно знать, как оно ведет себя на единичной сфере, чтобы вычислить образ любого вектора.

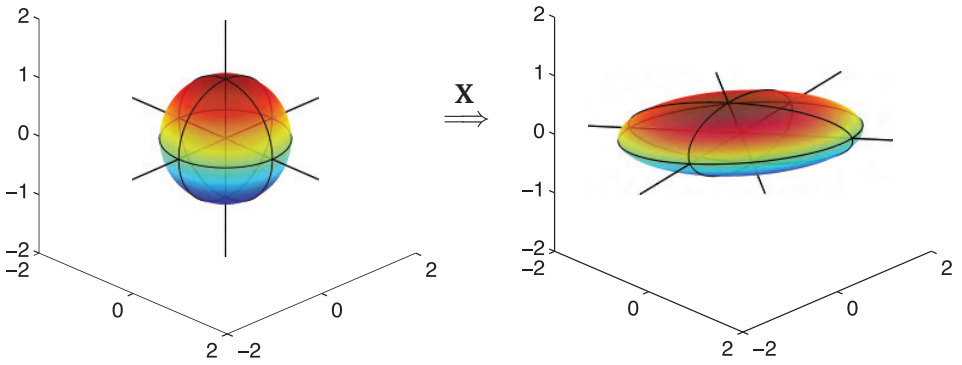


**Рис. 1.8** ❖ Геометрическая иллюстрация SVD как отображения сферы в $\mathbb{R}^n$ в эллипсоид в $\mathbb{R}^m$

Для частного случая на рис. 1.8 мы строим матрицу $\mathbf{X}$ из трех матриц поворота $\mathbf{R}_x$, $\mathbf{R}_y$ и $\mathbf{R}_z$ и четвертой матрицы, описывающей растяжение вдоль главных осей:

$$\mathbf{X} = \underbrace{\begin{bmatrix}\cos(\theta_3) & -\sin(\theta_3) & 0\\ \sin(\theta_3) & \cos(\theta_3) & 0\\ 0 & 0 & 1\end{bmatrix}}_{\mathbf{R}_z}\underbrace{\begin{bmatrix}\cos(\theta_2) & 0 & \sin(\theta_2)\\ 0 & 1 & 0\\ -\sin(\theta_2) & 0 & \cos(\theta_2)\end{bmatrix}}_{\mathbf{R}_y}$$
$$\times\underbrace{\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\theta_1) & -\sin(\theta_1)\\ 0 & \sin(\theta_1) & \cos(\theta_1)\end{bmatrix}}_{\mathbf{R}_y}\begin{bmatrix}\sigma_1 & 0 & 0\\ 0 & \sigma_2 & 0\\ 0 & 0 & \sigma_3\end{bmatrix}.$$

В этом случае $\theta_1 = \pi/15$, $\theta_2 = -\pi/9$ и $\theta_3 = -\pi/20$, а $\sigma_1 = 3$, $\sigma_2 = 1$, $\sigma_3 = 0.5$. Матрицы поворота не коммутируют, поэтому порядок поворотов имеет значение. Если одно из сингулярных значений равно нулю, то соответствующая размерность исчезает и эллипсоид становится фигурой в подпространстве меньшей размерности. Произведение $\mathbf{R}_x\mathbf{R}_y\mathbf{R}_z$ является унитарной матрицей $\mathbf{U}$ в сингулярном разложении $\mathbf{X}$. Матрица $\mathbf{V}$ в данном случае единичная.

**Листинг 1.1** ❖ Построение матриц поворота

```
theta = [pi/15; -pi/9; -pi/20];
Sigma = diag([3; 1; 0.5]);              % масштабировать x, y, z
Rx = [1 0 0;                            % поворот вокруг оси x
0 cos(theta(1)) -sin(theta(1));
0 sin(theta(1)) cos(theta(1))];
Ry = [cos(theta(2)) 0 sin(theta(2));    % поворот вокруг оси y
0 1 0;
-sin(theta(2)) 0 cos(theta(2))];
Rz = [cos(theta(3)) -sin(theta(3)) 0;   % поворот вокруг оси z
sin(theta(3)) cos(theta(3)) 0;
0 0 1];
X = Rz*Ry*Rx*Sigma;                     % повернуть и масштабировать
```

**Листинг 1.2** ❖ Изображение сферы

```
[x,y,z] = sphere(25);
h1=surf(x,y,z);
```

**Листинг 1.3** ❖ Отображение сферы путем умножения на **X** и изображение получившегося эллипсоида

```
xR = 0*x; yR = 0*y; zR = 0*z;
for i=1:size(x,1)
    for j=1:size(x,2)
        vecR = X*[x(i,j); y(i,j); z(i,j)];
        xR(i,j) = vecR(1);
        yR(i,j) = vecR(2);
        zR(i,j) = vecR(3);
    end
end
h2=surf(xR,yR,zR,z);      % координата z сферы задает цвет
```

# Инвариантность SVD относительно унитарных преобразований

У SVD есть полезное свойство: если умножить матрицу данных $\mathbf{X}$ слева или справа на унитарную матрицу, то члены сингулярного разложения не изменятся, за исключением левой или правой унитарной матрицы $\mathbf{U}$ или $\mathbf{V}$ соответственно. У этого факта есть важные следствия, поскольку дискретное преобразование Фурье (ДПФ, см. главу 2) $\mathcal{F}$ является унитарным преобразо-

ванием, а это значит, что SVD матрицы $\hat{\mathbf{X}} = \mathcal{F}\mathbf{X}$ совпадает с SVD матрицы $\mathbf{X}$ с тем отличием, что моды $\hat{\mathbf{U}}$ будут дискретными преобразованиями Фурье мод $\mathbf{U}$: $\hat{\mathbf{U}} = \mathcal{F}\mathbf{U}$. Кроме того, инвариантность SVD относительно унитарных преобразований позволяет использовать сжатые измерения для реконструкции мод SVD, разреженных в некотором базисе преобразования (см. главу 3).

Инвариантность SVD относительно унитарных преобразований геометрически очевидна, поскольку унитарное преобразование лишь поворачивает векторы в пространстве, но не изменяет их скалярные произведения и структуру корреляций. Будем обозначать левое унитарное преобразование $\mathbf{C}$, так что $\mathbf{Y} = \mathbf{CX}$, а правое унитарное преобразование $\mathbf{P}^*$, так что $\mathbf{Y} = \mathbf{XP}^*$. SVD матрицы $\mathbf{X}$ будем обозначать $\mathbf{U}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}\mathbf{V}_\mathbf{X}^*$, а SVD матрицы $\mathbf{Y}$ будет равно $\mathbf{U}_\mathbf{Y}\mathbf{\Sigma}_\mathbf{Y}\mathbf{V}_\mathbf{Y}^*$.

## Левые унитарные преобразования

Сначала рассмотрим левое унитарное преобразование $\mathbf{X}$: $\mathbf{Y} = \mathbf{CX}$. Вычисление корреляционной матрицы $\mathbf{Y}^*\mathbf{Y}$ дает

$$\mathbf{Y}^*\mathbf{Y} = \mathbf{X}^*\mathbf{C}^*\mathbf{CX} = \mathbf{X}^*\mathbf{X}. \tag{1.10}$$

Спроецированные данные имеют такой же спектральный состав, т. е. $\mathbf{V}_\mathbf{X}$ и $\mathbf{\Sigma}_\mathbf{X}$ не изменяются. Применив метод снимков для реконструкции $\mathbf{U}_\mathbf{Y}$, находим

$$\mathbf{U}_\mathbf{Y} = \mathbf{YV}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}^{-1} = \mathbf{CXV}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}^{-1} = \mathbf{CU}_\mathbf{X}. \tag{1.11}$$

Таким образом, $\mathbf{U}_\mathbf{Y} = \mathbf{CU}_\mathbf{X}$, $\mathbf{\Sigma}_\mathbf{Y} = \mathbf{\Sigma}_\mathbf{X}$ и $\mathbf{V}_\mathbf{Y} = \mathbf{V}_\mathbf{X}$. Тогда SVD матрицы $\mathbf{Y}$ равно:

$$\mathbf{Y} = \mathbf{CX} = \mathbf{CU}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}\mathbf{V}_\mathbf{X}^*. \tag{1.12}$$

## Правые унитарные преобразования

Для правого унитарного преобразования $\mathbf{Y} = \mathbf{XP}^*$ корреляционная матрица $\mathbf{Y}^*\mathbf{Y}$ равна

$$\mathbf{Y}^*\mathbf{Y} = \mathbf{PX}^*\mathbf{XP}^* = \mathbf{PV}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}^2\mathbf{V}_\mathbf{X}^*\mathbf{P}^* \tag{1.13}$$

и имеет такое спектральное разложение:

$$\mathbf{Y}^*\mathbf{YPV}_\mathbf{X} = \mathbf{PV}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}^2. \tag{1.14}$$

Таким образом, $\mathbf{V}_\mathbf{Y} = \mathbf{PV}_\mathbf{X}$ и $\mathbf{\Sigma}_\mathbf{Y} = \mathbf{\Sigma}_\mathbf{X}$. Мы можем воспользоваться методом снимков для реконструкции $\mathbf{U}_\mathbf{Y}$:

$$\mathbf{U}_\mathbf{Y} = \mathbf{YPV}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}^{-1} = \mathbf{XV}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}^{-1} = \mathbf{U}_\mathbf{X}. \tag{1.15}$$

Следовательно, $\mathbf{U}_\mathbf{Y} = \mathbf{U}_\mathbf{X}$, и SVD матрицы $\mathbf{Y}$ можно записать в виде:

$$\mathbf{Y} = \mathbf{XP}^* = \mathbf{U}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}\mathbf{V}_\mathbf{X}^*\mathbf{P}^*. \tag{1.16}$$

# 1.4. Псевдообращение, метод наименьших квадратов и регрессия

Многие физические системы можно представить линейной системой уравнений

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{1.17}$$

где матрица ограничений $\mathbf{A}$ и вектор $\mathbf{b}$ известны, а вектор $\mathbf{x}$ неизвестен. Если $\mathbf{A}$ – квадратная обратимая матрица (т. е. определитель $\mathbf{A}$ не равен нулю), то существует единственное решение $\mathbf{x}$ для любого $\mathbf{b}$. Но если $\mathbf{A}$ сингулярная или прямоугольная, то может существовать одно, ни одного или бесконечно много решений в зависимости от конкретного $\mathbf{b}$ и пространств столбцов и строк $\mathbf{A}$.

Сначала рассмотрим *недоопределенную систему*, т. е. случай, когда $\mathbf{A} \in \mathbb{C}^{n\times m}$ и $n \ll m$ ($\mathbf{A}$ – низкая и толстая матрица) – уравнений меньше, чем неизвестных. Такая система, скорее всего, будет иметь полный столбцовый ранг, поскольку число столбцов много больше, чем требуется для линейно независимого базиса[1]. В общем случае, если низкая и толстая матрица $\mathbf{A}$ имеет полный столбцовый ранг, то для каждого $\mathbf{b}$ существует бесконечно много решений $\mathbf{x}$. Такая система называется *недоопределенной*, потому что элементов $\mathbf{b}$ недостаточно, чтобы определить вектор $\mathbf{x}$ высокой размерности.

Точно так же рассмотрим *переопределенную систему*, когда $n \gg m$ (высокая и тощая матрица), т. е. уравнений больше, чем неизвестных. Эта матрица не может иметь полного столбцового ранга, поэтому гарантируется, что существуют такие векторы $\mathbf{b}$, для которых нет ни одного решения $\mathbf{x}$. На самом деле решение $\mathbf{x}$ существует, только если $\mathbf{b}$ принадлежит пространству столбцов $\mathbf{A}$, т. е. $\mathbf{b} \in \mathrm{col}(\mathbf{A})$.

Технически могут существовать $\mathbf{b}$, при которых имеется бесконечно много решений $\mathbf{x}$ для высокой и тощей матрицы $\mathbf{A}$, равно как и такие $\mathbf{b}$, при которых не существует ни одного решения для низкой и толстой матрицы. Пространство решений системы (1.17) определяется четырьмя фундаментальными подпространствами $\mathbf{A} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$, где ранг $r$ выбран так, чтобы все ненулевые сингулярные значения были включены:

- пространство столбцов, $\mathrm{col}(\mathbf{A})$, натянутое на столбцы $\mathbf{A}$; оно называется также *областью значений*. Пространство столбцов $\mathbf{A}$ совпадает с пространством столбцов $\tilde{\mathbf{U}}$;
- ортогональное дополнение к $\mathrm{col}(\mathbf{A})$ обозначается $\ker(\mathbf{A}^*)$ и совпадает с пространством столбцов матрицы $\hat{\mathbf{U}}^{\perp}$ на рис. 1.1;
- пространство строк, $\mathrm{row}(\mathbf{A})$, натянутое на строки $\mathbf{A}$ и совпадающее с линейной оболочкой столбцов $\tilde{\mathbf{V}}$. Имеет место равенство $\mathrm{row}(\mathbf{A}) = \mathrm{col}(\mathbf{A}^*)$;

[1] Легко построить вырожденные примеры низкой и толстой матрицы неполного столбцового ранга, например $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$.

- ядерное пространство, $\ker(\mathbf{A})$, являющееся ортогональным дополнением к $\text{row}(\mathbf{A})$ и называемое также *нуль-пространством*, или *ядром*. Ядро – это подпространство, состоящее из векторов, которые $\mathbf{A}$ отображает в нуль, т. е. $\mathbf{Ax} = \mathbf{0}$; оно совпадает с $\text{col}(\hat{\mathbf{V}}^{\perp})$.

Точнее, если $\mathbf{b} \in \text{col}(\mathbf{A})$ и $\dim(\ker(\mathbf{A})) \neq 0$, то существует бесконечно много решений $\mathbf{x}$. Заметим, что условие $\dim(\ker(\mathbf{A})) \neq 0$ гарантированно выполняется для низкой и толстой матрицы. Аналогично, если $\mathbf{b} \notin \text{col}(\mathbf{A})$, то решений не существует, и система уравнений (1.17) называется *несовместной*.

Описанные выше фундаментальные подпространства обладают следующими свойствами:

$$\text{col}(\mathbf{A}) \oplus \ker(\mathbf{A}^*) = \mathbb{R}^n \tag{1.18a}$$

$$\text{col}(\mathbf{A}^*) \oplus \ker(\mathbf{A}) = \mathbb{R}^n. \tag{1.18b}$$

**Замечание 1.** *Имеется обширная литература по теории случайных матриц, в которой перечисленные выше утверждения почти всегда верны, т. е. верны с высокой вероятностью. Например, крайне маловероятно, что система* $\mathbf{Ax} = \mathbf{b}$ *имеет решение для случайно выбранных матрицы* $\mathbf{A} \in \mathbb{R}^{n \times m}$ *и вектора* $\mathbf{b} \in \mathbb{R}^n$, *где* $n \gg m$, *т. к. мало шансов, что* $\mathbf{b}$ *принадлежит пространству столбцов* $\mathbf{A}$. *Эти свойства случайных матриц будут играть важную роль в теории сжатых измерений (см. главу 3).*

В переопределенном случае, когда решений не существует, нам часто хотелось бы найти вектор $\mathbf{x}$, который минимизирует сумму квадратов ошибок $||\mathbf{Ax} - \mathbf{b}||_2^2$, он называется решением *с наименьшей среднеквадратической ошибкой*. Заметим, что такое решение минимизирует также величину $||\mathbf{Ax} - \mathbf{b}||_2$. В недоопределенном случае, когда существует бесконечно много решений, часто требуется найти решение $\mathbf{x}$, для которого $\mathbf{Ax} = \mathbf{b}$, а норма $||\mathbf{x}||_2$ минимальна; оно называется решением с *минимальной нормой*.

SVD – общепризнанный метод решения этих важных задач оптимизации. Прежде всего если подставить точное усеченное SVD $\mathbf{A} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$ вместо $\mathbf{A}$, то каждую из матриц $\tilde{\mathbf{U}}$, $\tilde{\mathbf{\Sigma}}$ и $\tilde{\mathbf{V}}^*$ можно будет поочередно «обратить», что даст *левую псевдообратную матрицу* Мура–Пенроуза [425, 426, 453, 572] $\mathbf{A}^{\dagger}$:

$$\mathbf{A}^{\dagger} \triangleq \tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^* \;\Rightarrow\; \mathbf{A}^{\dagger}\mathbf{A} = \mathbf{I}^{m \times m}. \tag{1.19}$$

Ее можно использовать для нахождения решений системы (1.17) с наименьшей среднеквадратической ошибкой и с минимальной нормой:

$$\mathbf{A}^{\dagger}\mathbf{A}\tilde{\mathbf{x}} = \mathbf{A}^{\dagger}\mathbf{b} \;\Rightarrow\; \tilde{\mathbf{x}} = \tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^*\mathbf{b}. \tag{1.20}$$

Подставляя решение $\tilde{\mathbf{x}}$ обратно в (1.17), получаем:

$$\mathbf{A}\tilde{\mathbf{x}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*\tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^*\mathbf{b} \tag{1.21a}$$

$$\mathbf{A}\tilde{\mathbf{x}} = \tilde{\mathbf{U}}\tilde{\mathbf{U}}^*\mathbf{b}. \tag{1.21b}$$

Заметим, что $\tilde{\mathbf{U}}\tilde{\mathbf{U}}^*$ – необязательно единичная матрица, но является проекцией на пространство столбцов $\tilde{\mathbf{U}}$. Поэтому $\tilde{\mathbf{x}}$ будет точным решением системы (1.17), только если $\mathbf{b}$ принадлежит пространству столбцов $\tilde{\mathbf{U}}$, а значит, пространству столбцов $\mathbf{A}$.

Вычислить псевдообратную матрицу $\mathbf{A}^\dagger$ можно эффективно, если предварительно произвести затратное вычисление SVD. Обращение унитарных матриц $\tilde{\mathbf{U}}$ и $\tilde{\mathbf{V}}$ сводится к умножению на транспонированные матрицы, для чего требуется $O(n^2)$ операций. Обращение диагональной матрицы $\tilde{\mathbf{\Sigma}}$ еще эффективнее и требует всего $O(n)$ операций. С другой стороны, обращение плотной квадратной матрицы потребовало бы $O(n^3)$ операций.

## Одномерная линейная регрессия

Регрессия – важный статистический инструмент установления связи между величинами на основе имеющихся данных [360]. Рассмотрим набор данных, изображенный на рис. 1.9. Точки, обозначенные красными крестиками, получены прибавлением гауссова белого шума к черной прямой, как показано в листинге 1.4. Мы предполагаем, что между данными имеется линейная связь, как в (1.17), и используем псевдообратную матрицу для нахождения решения с наименьшей среднеквадратической ошибкой – синей штриховой прямой с угловым коэффициентом $x$, – как показано в листинге 1.5.

$$\begin{bmatrix} | \\ \mathbf{b} \\ | \end{bmatrix} = \begin{bmatrix} | \\ \mathbf{a} \\ | \end{bmatrix} x = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^* x. \tag{1.22a}$$

$$\Rightarrow x = \tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^*\mathbf{b}. \tag{1.22b}$$

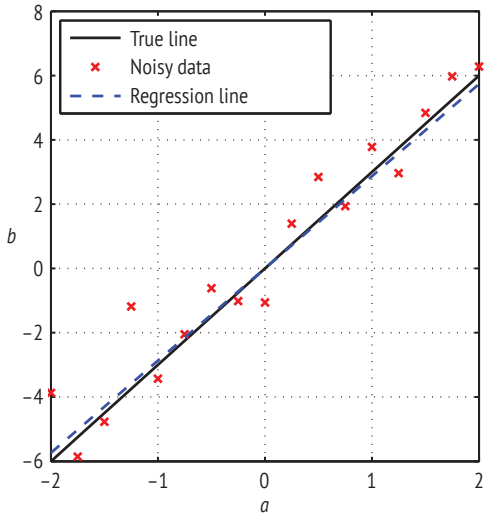


**Рис. 1.9** ❖ Линейная регрессия с зашумленными данными

В (1.22b) $\tilde{\mathbf{\Sigma}} = ||\mathbf{a}||_2, \tilde{\mathbf{V}} = 1, \tilde{\mathbf{U}} = \mathbf{a}/||\mathbf{a}||_2$. Умножение на левую псевдообратную матрицу дает

$$x = \frac{\mathbf{a}^*\mathbf{b}}{||\mathbf{a}||_2^2}. \tag{1.23}$$

Это имеет физический смысл, если интерпретировать $x$ как значение, которое дает наилучшее отображение нашего вектора **a** в вектор **b**. Такое наилучшее значение $x$ единственно и получается в результате вычисления скалярного произведения **b** с нормированным вектором в направлении **a**. И мы добавляем второй нормировочный коэффициент $||\mathbf{a}||_2$, потому что **a** в формуле (1.22a) не нормирован.

Отметим, что если в (1.22) взять векторы-строки вместо векторов-столбцов, то будут происходить странные вещи. Кроме того, если величина шума велика по сравнению с угловым коэффициентом $x$, то в точности псевдообратной матрицы произойдет фазовый переход, связанный с жесткой пороговой обработкой, описанной ниже.

**Листинг 1.4** ❖ Генерирование зашумленных данных для рис. 1.9

```
x = 3;                          % истинный угловой коэффициент
a = [-2:.25:2]';
b = a*x + 1*randn(size(a));     % добавить шум
plot(a,x*a,'k')                 % истинная связь
hold on, plot(a,b,'rx')         % зашумленные измерения
```

**Листинг 1.5** ❖ Вычисление аппроксимации методом наименьших квадратов для рис. 1.9

```
[U,S,V] = svd(a,'econ');
xtilde = V*inv(S)*U'*b;         % аппроксимации методом наименьших квадратов
plot(a,xtilde*a,'b--')          % нарисовать график
```

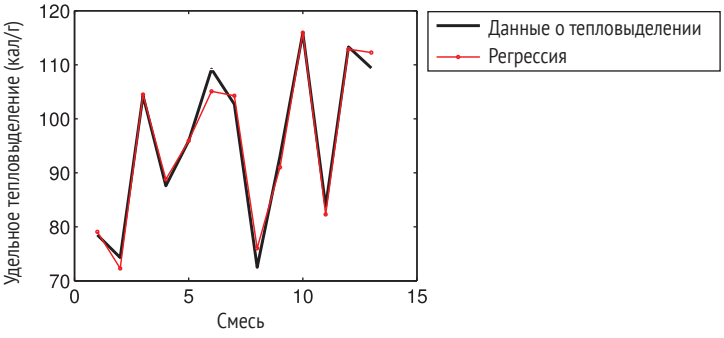


**Рис. 1.10** ❖ Данные о тепловыделении для четырехкомпонентных цементных смесей

В статистике описанная выше процедура называется *линейной регрессией*. В MATLAB есть команда **regress**, а также команда **pinv**, которую тоже можно использовать.

**Листинг 1.6** ❖ Альтернативные варианты метода наименьших квадратов в MATLAB

```
xtilde1 = V*inv(S)*U'*b
xtilde2 = pinv(a)*b
xtilde3 = regress(b,a)
```

# Полилинейная регрессия

### *Пример 1: данные о тепловыделении цементных смесей*

Начнем с простого встроенного в MATLAB набора данных, который описывает тепловыделение различных цементных смесей, составленных из четырех компонент. В этой задачи мы решаем систему (1.17) с $\mathbf{A} \in \mathbb{R}^{13\times 4}$, поскольку компонент четыре, а измерения производятся для 13 разных смесей. Наша цель – найти веса $\mathbf{x}$, описывающие связь между четырьмя ингредиентами и тепловыделением. В листинге 1.7 показано, как найти решение с минимальной ошибкой. Исследуются варианты с использованием функций **regress** и **pinv**.

**Листинг 1.7** ❖ Полилинейная регрессия для данных о тепловыделении цементных смесей

```
load hald;                  % загрузить набор данных Portlant Cement
A = ingredients;
b = heat;

[U,S,V] = svd(A,'econ');
x = V*inv(S)*U'*b;          % решить систему Ax=b методом SVD

plot(b,'k'); hold on        % построить график
plot(A*x,'r-o',);           % аппроксимация графика

x = regress(b,A);           % вариант 1 (regress)
x = pinv(A)*b;              % вариант 2 (pinv)
```

### *Пример 2: данные о недвижимости в Бостоне*

В этом примере мы исследуем более крупный набор данных, чтобы определить, какие факторы лучше всего предсказывают цены на бостонском рынке недвижимости [234]. Этот набор можно скачать из репозитория машинного обучения UCI Machine Learning Repository [24].

Существует 13 атрибутов, коррелирующих с ценой дома, например: индекс преступности на душу населения и ставка налога на имущество. Мы построили регрессионную модель влияния этих факторов на цену и на рис. 1.11 на-

рисовали график, на котором лучшее предсказание цены сопоставляется с истинной стоимостью дома. На рис. 1.12 показаны коэффициенты регрессии. Хотя стоимость дома предсказана неточно, тренды хорошо согласуются. Часто бывает, что простая линейная аппроксимация плохо улавливает выбросы с наибольшими значениями; такую ситуацию мы наблюдаем и в этом примере.

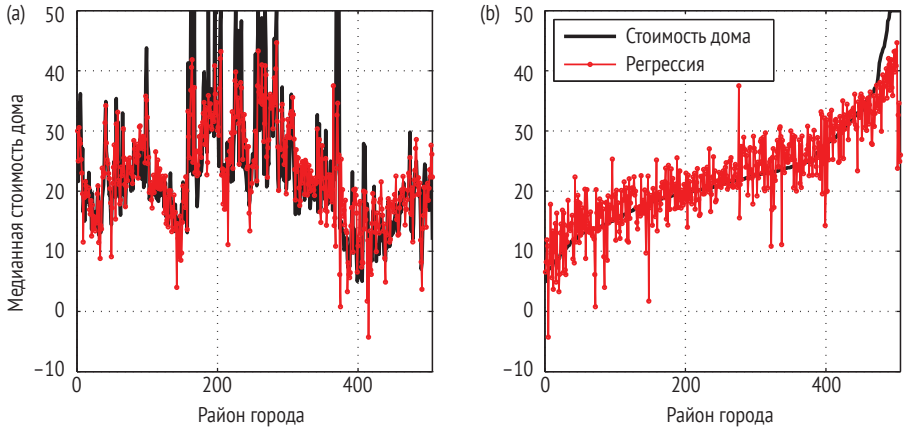


**Рис. 1.11** ❖ Полилинейная регрессия цен домов с различными факторами: (a) неотсортированные данные; (b) данные отсортированы по стоимости дома

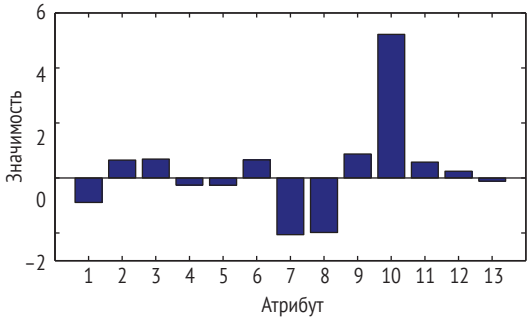


**Рис. 1.12** ❖ Значимость различных атрибутов в регрессии

Этот набор данных содержит цены и атрибуты для 506 домов, так что матрица атрибутов имеет размер $506 \times 13$. Важно дополнить эту матрицу столбцом, состоящим из единиц, чтобы учесть возможность ненулевого постоянного смещения в формуле регрессии. В случае одномерной регрессии это соответствует свободному члену.

**Листинг 1.8** ❖ Полилинейная регрессия для данных о стоимости домов в Бостоне

```
load housing.data

b = housing(:,14);                % стоимости домов в тысячах долларов
A = housing(:,1:13);              % прочие факторы
```

```
A = [A ones(size(A,1),1)];          % дополнить единичными свободными членами

x = regress(b,A);
plot(b,'k-o');
hold on, plot(A*x,'r-o');

[b sortind] = sort(housing(:,14));   % отсортированные стоимости
plot(b,'k-o')
hold on, plot(A(sortind,:)*x,'r-o')
```

## Предостережение

Вообще говоря, матрица $\mathbf{U}$, столбцами которой являются левые сингулярные векторы $\mathbf{X}$, – унитарная квадратная матрица. Поэтому $\mathbf{U}^*\mathbf{U} = \mathbf{U}\mathbf{U}^* = \mathbf{I}_{n\times n}$. Однако чтобы вычислить псевдообратную матрицу для $\mathbf{X}$, мы должны вычислить $\mathbf{X}^\dagger = \tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^*$, потому что только $\tilde{\mathbf{\Sigma}}$ обратима (если все сингулярные значения отличны от нуля), тогда как $\mathbf{\Sigma}$ в общем случае необратима (она даже в общем случае не является квадратной).

До сих пор мы предполагали, что $\mathbf{X} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$ – точное SVD, так что ранг $r$ включает все ненулевые сингулярные значения. Это гарантирует, что матрица $\tilde{\mathbf{\Sigma}}$ обратима.

Трудности начинаются при работе с усеченным базисом, образованным левыми сингулярными векторами $\tilde{\mathbf{U}}$. По-прежнему верно, что $\tilde{\mathbf{U}}^*\tilde{\mathbf{U}} = \mathbf{I}_{r\times r}$, где $r$ – ранг $\mathbf{X}$. Однако $\tilde{\mathbf{U}}\tilde{\mathbf{U}}^* \neq \mathbf{I}_{n\times n}$, что легко проверить численно на простом примере. Предположение о том, что $\tilde{\mathbf{U}}\tilde{\mathbf{U}}^*$ равно единичной матрице, – одно из самых типичных заблуждений при использовании SVD[1].

```
>> tol = 1.e-16;
>> [U,S,V] = svd(X,'econ')
>> r = max(find(diag(S)>max(S(:))*tol));
>> invX = V(:,1:r)*S(1:r,1:r)*U(:,1:r)'; % только приближенно
```

# 1.5. Метод главных компонент (PCA)

Метод главных компонент (PCA) – одно из основных применений SVD, он позволяет, ориентируясь только на данные, построить систему координат для представления коррелированных данных высокой размерности. При этом используются корреляционные матрицы, описанные в разделе 1.3. Важно, что в методе PCA, перед тем как применять SVD, производится предварительная обработка данных: вычитание среднего и приведение к единичной дисперсии. Геометрия результирующей системы координат определяется главными компонентами (principal component – PC), которые не коррелируют между собой (ортогональны), но имеют максимальную корреляцию с результатами измерений. Теория была разработана в 1901 году Пирсоном

[1] Его не избежали и авторы, по ошибке включив это неверное тождество в ранний вариант работы [96].

[418] и независимо в 1930-х годах Хотеллингом [256, 257]. Работа Jolliffe [268] содержит хорошее справочное пособие.

Обычно в одном эксперименте производится несколько измерений, которые образуют вектор-строку. Эти измерения могут быть свойствами наблюдаемой величины, например демографическими признаками одного человека. Выполняется несколько экспериментов, и все векторы-строки собираются в большую матрицу **X**. Если говорить о демографии, то экспериментальные данные могут быть собраны путем опроса. Заметим, что такое соглашение – **X** состоит из строк признаков – отличается от соглашения, принятого в других частях этой главы, где отдельные «снимки» признаков расположены по столбцам. Но мы решили в этом разделе не вступать в противоречие с литературой по PCA. Матрица по-прежнему имеет размер $n \times m$ и может содержать больше строк, чем столбцов, или наоборот.

## Вычисление

Теперь вычислим среднее по строкам $\bar{\mathbf{x}}$ (т. е. среднее всех строк) и вычтем его из **X**. Среднее $\bar{\mathbf{x}}$ определяется по формуле

$$\bar{\mathbf{x}}_j = \frac{1}{n}\sum_{i=1}^{n}\mathbf{X}_{ij}, \tag{1.24}$$

а средняя матрица – по формуле

$$\bar{\mathbf{X}} = \begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix}\bar{\mathbf{x}}. \tag{1.25}$$

Вычитая $\bar{\mathbf{X}}$ из **X**, получаем матрицу **B** с нулевым средним:

$$\mathbf{B} = \mathbf{X} - \bar{\mathbf{X}}. \tag{1.26}$$

Ковариационная матрица строк **B** определяется формулой

$$\mathbf{C} = \frac{1}{n-1}\mathbf{B}^{*}\mathbf{B}. \tag{1.27}$$

Первая главная компонента $\mathbf{u}_1$ определяется так:

$$\mathbf{u}_1 = \underset{||\mathbf{u}_1||=1}{\operatorname{argmax}}\,\mathbf{u}_1^{*}\mathbf{B}^{*}\mathbf{B}\mathbf{u}_1. \tag{1.28}$$

Это собственный вектор $\mathbf{B}^{*}\mathbf{B}$, соответствующий наибольшему собственному значению. Теперь понятно, что $\mathbf{u}_1$ – левый сингулярный вектор **B**, соответствующий наибольшему сингулярному значению.

Главные компоненты можно получить, вычислив спектральное разложение **C**:

$$\mathbf{CV} = \mathbf{VD}, \tag{1.29}$$

которое гарантированно существует, потому что матрица **C** эрмитова.

### *Команда pca*

В MATLAB имеются дополнительные команды **pca** и **princomp** (основанная на **pca**) для метода главных компонент:

```
>> [V,score,s2] = pca(X);
```

Матрица **V** эквивалентна матрице **V** из сингулярного разложения **X** с точностью до изменения знака столбцов. Вектор **s2** содержит собственные значения ковариационной матрицы **X**, которые называются также дисперсиями главных компонент; это квадраты сингулярных значений. Переменная **score** содержит координаты каждой строки **B** (данные после вычитания среднего) в направлениях главных компонент. Вообще говоря, мы часто предпочитаем использовать команду **svd** в сочетании с различными шагами постобработки, описанными ранее в этом разделе.

## Пример: данные с гауссовым шумом

Рассмотрим зашумленное облако данных на рис. 1.13 (a), построенном программой в листинге 1.9. Данные генерируются путем выборки 10 000 векторов из двумерного нормального распределения с нулевым средним и единичной дисперсией. Эти векторы затем масштабируются в направлениях $x$ и $y$ с коэффициентами в табл. 1.1 и поворачиваются на угол $\pi/3$. Наконец, все облако данных подвергается параллельному переносу, так что его центр располагается в точке $\mathbf{x}_C = [2 \;\; 1]^T$.

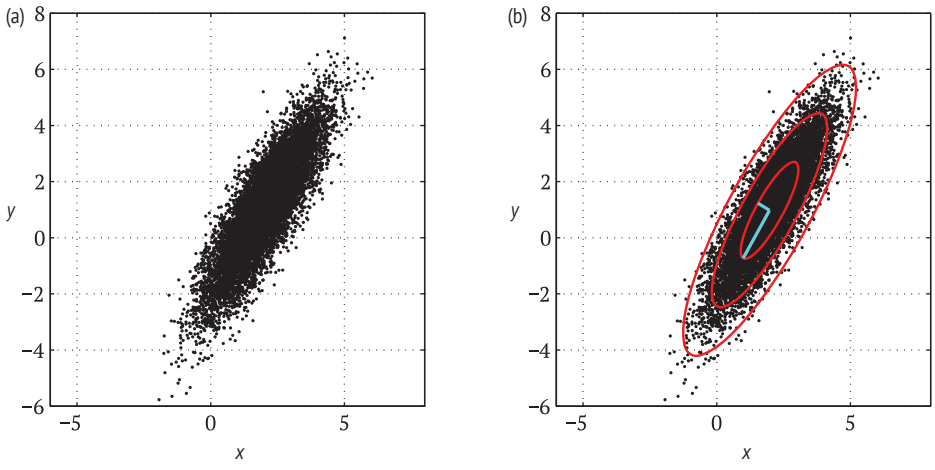


**Рис. 1.13** ❖ Главные компоненты улавливают дисперсию гауссовых данных после вычитания среднего (a). На рисунке (b) показаны эллипсы для первых трех стандартных отклонений (красным цветом) и два левых сингулярных вектора, умноженных на сингулярные значения ($\sigma_1 u_1 + x_C$ и $\sigma_2 u_2 + x_C$, голубым цветом)