1 Математика в программном коде

В этой главе

- ✓ Решение финансовых задач с помощью математики и программного обеспечения.
- ✓ Как избежать распространенных ошибок при изучении математики.
- ✓ От программирования к пониманию математики, основываясь на интуиции.
- ✓ Python как мощный и расширяемый калькулятор.

Математика подобна бейсболу, поэзии или хорошему вину. Одни настолько увлечены этой наукой, что посвящают ей всю свою жизнь, другим же кажется, что они просто не понимают ее. Вероятно, после 11 лет изучения математики в школе вы уже причислили себя к той или иной группе людей.

Представьте, что в школе вы бы изучали виноделие подобно тому, как изучали математику. Мне бы не понравилось, если бы мне читали лекции о сортах винограда и методах ферментации по часу в день шесть дней в неделю. Будь такой предмет в школе, то, может быть, мне приходилось бы выпивать по три-четыре стакана, чтобы выполнить домашнее задание. С одной стороны, это мог бы быть восхитительный образовательный опыт, но с другой — идти следующим утром на занятия с тяжелой головой — сомнительное удовольствие. Опыт, который я получил на уроках математики, был примерно таким же, и это на какое-то время отталкивало меня от предмета.

Проще всего решить, что вы либо созданы для математики, либо нет. Если вы уже верите в себя и хотите начать учиться — здорово! В остальном эта глава предназначена для тех, кто настроен менее оптимистично. Страх перед математикой настолько распространен, что даже назван *математической тревожностью (math anxiety)*. Я надеюсь рассеять ваше беспокойство и показать, что математика может быть очень интересным увлечением. Все, что вам нужно, — это правильные инструменты и правильный настрой.

Основным инструментом обучения в этой книге станет язык программирования Python. Полагаю, что когда вы изучали математику в школе, то учились по формулам, написанным на доске, а не по компьютерному коду. Это беда, потому что язык программирования высокого уровня намного мощнее школьной доски и намного универсальнее любого дорогого калькулятора, который вы могли бы использовать. Преимущество изучения математики на примерах в программном коде состоит в том, что идеи в этом случае выражаются достаточно точно, чтобы их мог понять компьютер, и нет необходимости спорить по поводу значений новых символов.

Как и при изучении любого нового предмета, лучший способ настроиться на успех — *захотеть* учиться. Тому может быть множество причин. Вас может заинтриговать красота математических понятий, или вам может доставлять удовольствие решение головоломных математических задач. Возможно, вы мечтаете создать приложение или игру, а для этого нужно реализовать математические вычисления в коде. Но я оставлю все эти причины в стороне и сосредоточусь на более прагматичной мотивации — решение математических задач с помощью программного обеспечения может принести вам много денег.

1.1. РЕШЕНИЕ ФИНАНСОВЫХ ЗАДАЧ С ПОМОЩЬЮ МАТЕМАТИКИ И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

На уроках математики от нерадивых учеников часто можно услышать вопрос: «Где в реальной жизни мне могут пригодиться эти знания?» Когда я учился в школе, учителя говорили нам, что математика поможет добиться профессионального успеха и заработать деньги. Я думаю, что они были правы, хотя иногда и приводили неправильные примеры. Например, я не рассчитываю банковские проценты вручную, как и мой банк. Может быть, если бы я стал геодезистом на стройплощадке, как предположил мой учитель тригонометрии, то использовал бы синусы и косинусы каждый день, чтобы заслужить свою зарплату.

Как оказалось, примеры из школьных учебников не так уж и практичны. И все же в реальной жизни есть примеры ошеломляюще прибыльного применения математики. Многие из них заключаются в воплощении правильной математической идеи в пригодном для использования программном обеспечении. Некоторыми из таких примеров я хочу с вами поделиться.

1.1.1. Прогнозирование движения финансового рынка

Все мы слышали легенды о биржевых трейдерах, которые зарабатывают миллионы долларов и умудряются покупать и продавать нужные акции в нужное время. По фильмам, которые я видел, у меня сложился стереотипный образ трейдера как мужчины средних лет в костюме, который кричит на своего брокера по мобильному телефону и разъезжает на спортивной машине. Возможно, когда-то это и соответствовало действительности, но сейчас все изменилось.

В офисах, разбросанных по всему Манхэттену, скрываются тысячи людей, которых называют *биржевыми аналитиками*. Они разрабатывают математические алгоритмы для автоматизации торговли акциями и получения прибыли. Они не носят костюмы и не кричат в мобильные телефоны, но я уверен, что у многих из них очень хорошие спортивные автомобили.

Но как биржевой аналитик может написать программу, которая автоматически зарабатывает деньги? Ответы на этот вопрос охраняются как самые ценные коммерческие секреты, но можете быть уверены: в них много математики. Рассмотрим короткий пример, чтобы понять, как может работать автоматизированная стратегия торговли.

Акции — это разновидность финансовых активов, представляющих доли участия в компаниях. Когда рынок понимает, что дела у компании идут хорошо, ее акции растут в цене — их покупка становится более дорогостоящей, а продажа — более прибыльной. Цены на акции меняются постоянно и хаотично. На рис. 1.1 показано, как может выглядеть график изменения цены акции в течение торгового дня.

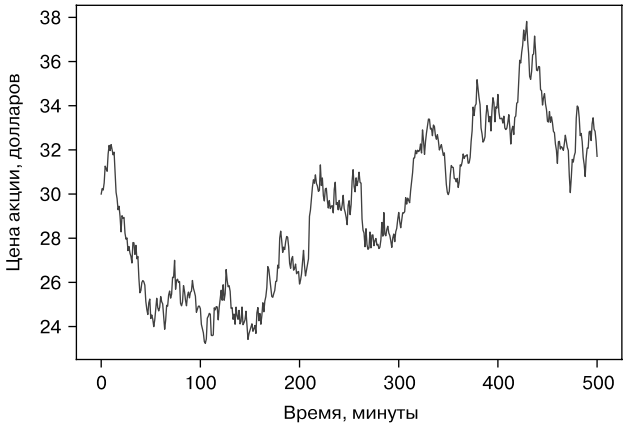

Рис. 1.1. Типичный график изменения цены акции с течением времени

Если купить 1000 этих акций по 24 доллара примерно на 100-й минуте и продать по 38 долларов на 400-й минуте, то можно заработать 14 000 долларов всего за один день. Неплохо! Однако проблема в том, что для этого нужно заранее знать, что акции вырастут и что 100-я и 400-я минуты являются лучшими моментами для покупки и продажи соответственно. Конечно, никто не сможет точно предсказать моменты, когда цена будет минимальной или максимальной, но можно попробовать найти относительно хорошее время для покупки и продажи в течение дня. Посмотрим, как эту задачу решить математически.

Мы могли бы определить, движется ли цена акций вверх или вниз, найдя линию наилучшего соответствия, которая примерно соответствует направлению движения цены. Этот процесс называется *линейной регрессией*, и мы рассмотрим его в части III. Основываясь на изменчивости данных, можно рассчитать еще две линии выше и ниже линии наилучшего соответствия, которые ограничивают область колебания цены. Как показано на рис. 1.2, эти три линии, наложенные на график колебания цены, довольно хорошо отражают общий тренд.

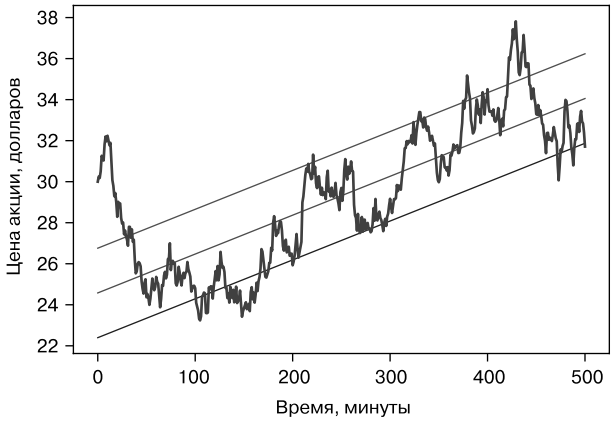

Рис. 1.2. Использование линейной регрессии для выявления тренда изменения цены на акции

Получив математическое понимание движения цены, можно написать код для автоматической покупки, когда цена колеблется возле нижней точки тренда, и продажи, когда она возвращается вверх. В частности, программа может подключаться к фондовой бирже по сети и покупать 100 акций, когда цена пересекает нижнюю линию, и продавать 100 акций, когда цена пересекает верхнюю линию. Рисунок 1.3 иллюстрирует одну из таких прибыльных сделок: покупка по цене около 27,8 доллара и продажа по цене около 32,6 доллара дают заработок 480 долларов в час.

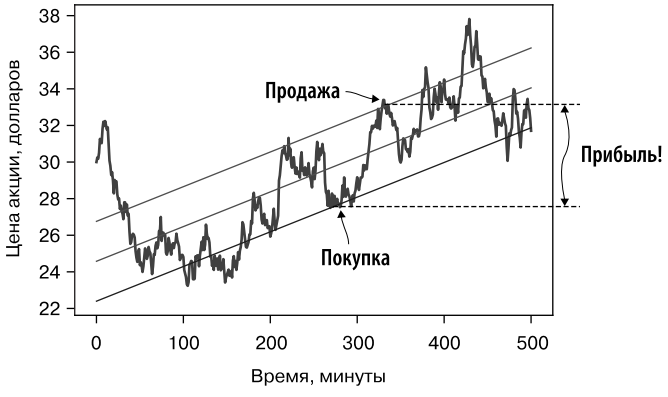

Рис. 1.3. Покупка и продажа в соответствии с установленными нами правилами обеспечивают прибыль

Я не берусь утверждать, что продемонстрировал полноценную или хотя бы жизнеспособную стратегию, но дело в том, что, имея правильную математическую модель, можно автоматически получать прибыль. Сейчас множество программ строят и обновляют модели, оценивающие прогнозируемую динамику изменения стоимости акций и других финансовых инструментов. Если вы напишете такую программу, то сможете наслаждаться отдыхом, в то время как она будет приносить вам деньги!

1.1.2. Поиск выгодной сделки

Наверняка у вас не настолько туго набитые карманы, чтобы пускаться в рискованные операции с акциями. Но математика все равно может помочь заработать и сэкономить деньги в других сферах, таких, например, как покупка подержанного автомобиля. С новой машиной все просто: если два дилера продают одинаковые автомобили, то вы, очевидно, пойдете к тому, который предложит наименьшую цену. Однако при покупке подержанного автомобиля приходится оценивать больше факторов — не только цену, но также пробег и год выпуска. Вы можете учитывать даже продолжительность нахождения конкретного подержанного автомобиля на рынке как косвенную оценку его качества: чем дольше он продается, тем более подозрительным кажется.

В математике объекты, которые можно описать с помощью упорядоченных списков чисел, называются *векторами*, и существует целая область математики, называемая линейной алгеброй, посвященная их изучению. Например, подержанный автомобиль может характеризоваться *четырехмерным* вектором, то есть четверкой чисел (2015, 41 429, 22,27, 16 980). Этот вектор содержит год выпуска модели, пробег, количество дней на рынке и запрашиваемую цену соответственно. У моего друга есть сайт CarGraph.com, на котором собраны данные о выставленных на продажу подержанных автомобилях. На момент написания

этих строк в списке насчитывался 101 автомобиль Toyota Prius, выставленный на продажу, и для каждого были предоставлены некоторые или все четыре элемента данных. Кроме того, оправдывая свое название, сайт предлагает визуальное представление данных в виде графика (рис. 1.4). Визуализировать четырехмерные объекты трудно, но если выбрать два измерения, такие как цена и пробег, то их можно изобразить в виде точек на точечной диаграмме.

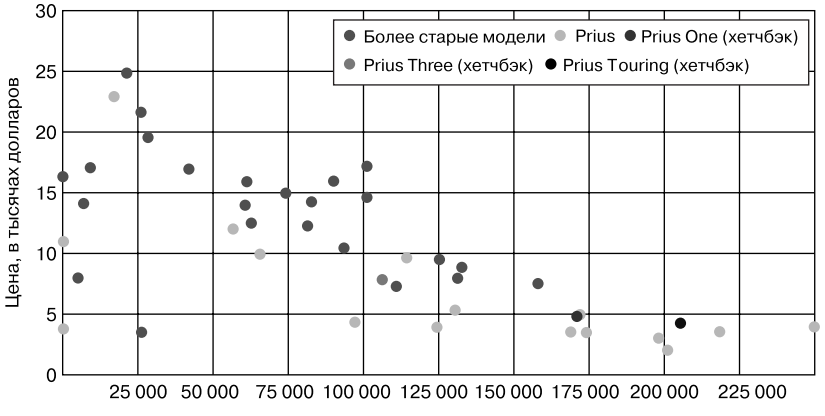

Рис. 1.4. Соотношение «цена/пробег» для автомобилей Toyota Prius, выставленных на продажу на сайте CarGraph.com

Было бы интересно попробовать построить линию тренда. Каждая точка на этом графике представляет чье-то мнение о справедливой цене, поэтому линия тренда объединит эти мнения в более надежную цену при любом пробеге. На рис. 1.5 я использовал *экспоненциальную* кривую вместо прямой и исключил из расчетов некоторые почти новые автомобили, продающиеся по цене ниже розничной.

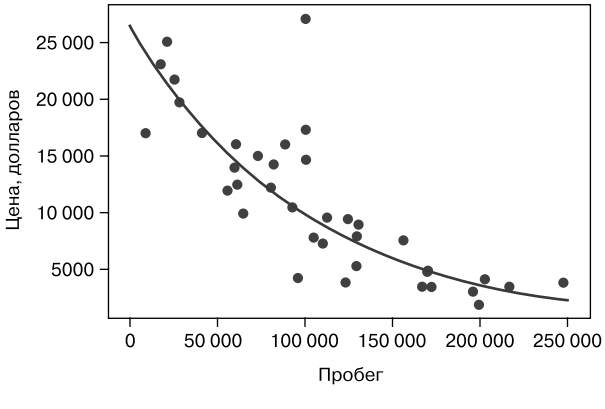

Рис. 1.5. Аппроксимация экспоненциальной кривой соотношения «цена/пробег» для подержанных автомобилей Toyota Prius

Для большего удобства я преобразовал значения пробега в десятки тысяч миль, поэтому пробег 5 соответствует 50 000 миль. Обозначив цену p, а пробег m, я вывел уравнение кривой наилучшего приближения:

$$p = 26\,500 \cdot 0{,}905^m. \tag{1.1}$$

Как показывает уравнение (1.1), в среднем цена подержанного автомобиля равна 26 500 долларов, умноженных на 0,905 в степени величины пробега. Подставив значения в уравнение, я выяснил, что, располагая 10 000 долларов, могу купить Prius с пробегом около 97 000 миль (рис. 1.6). Если считать эту кривую отражением *справедливой* цены, то автомобили ниже этой линии можно рассматривать как выгодные предложения.

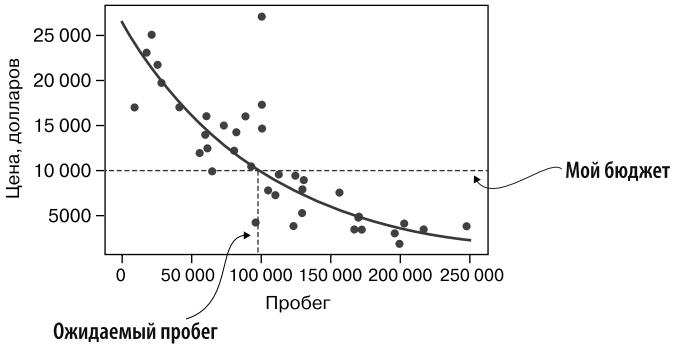

Рис. 1.6. Определение пробега автомобиля Prius, который можно приобрести при бюджете 10 000 долларов

Но уравнение (1.1) не только позволяет определить выгодность сделки. Оно рассказывает о том, как обесцениваются автомобили. Первое число в уравнении — 26 500 долларов — соответствует цене автомобиля с нулевым пробегом. Она удивительно близка к розничной цене нового Prius. Если бы наилучшей аппроксимацией тренда была прямая линия, то это означало бы, что стоимость Prius уменьшается на фиксированную сумму с каждой пройденной милей. Однако экспоненциальная форма кривой говорит о том, что стоимость уменьшается на фиксированный процент. Согласно этому уравнению, проехав 10 000 миль, Prius будет стоить 0,905, или 90,5 %, от первоначальной цены. После 50 000 миль пробега его цену нужно умножить на коэффициент $0{,}905^5 = 0{,}607$, то есть стоимость автомобиля составит 61 % от первоначальной цены.

Чтобы построить график, изображенный на рис. 1.6, я реализовал функцию `price(mileage)` на Python, которая принимает величину пробега (в десятках тысяч миль) и возвращает наиболее вероятную цену. Результаты вычислений `price(0) - price(5)` и `price(5) - price(10)` говорят мне, что первые и вторые 50 000 миль снижают стоимость примерно на 10 000 и 6300 долларов соответственно.

Аппроксимация тренда прямой линией означала бы, что стоимость автомобиля уменьшается на фиксированную величину 0,1 доллара за милю. То есть через каждые 50 000 миль пробега стоимость должна уменьшаться на одни и те же 5000 долларов. Но здравый смысл подсказывает, что первые мили пробега новой машины — самые дорогие, и экспоненциальная функция (уравнение (1.1)) хорошо согласуется с этим, а линейная модель — нет.

Но давайте не будем забывать, что это лишь *двухмерный* анализ. Мы построили математическую модель, использующую две характеристики из четырех, описывающих каждый автомобиль. В части I вы познакомитесь с векторами различных размерностей и узнаете, как манипулировать многомерными данными. В части II мы рассмотрим различные виды функций, такие как линейные и экспоненциальные, и сравним их, проанализировав скорости изменения. Наконец, в части III вы узнаете, как строить математические модели, которые включают *все* измерения, имеющиеся в наборе данных, и позволяют получить более точную картину.

1.1.3. Трехмерная графика и анимация

Многие из самых известных и финансово успешных программных проектов так или иначе обрабатывают многомерные данные, часто *трехмерные*. Здесь я имею в виду трехмерные анимационные фильмы и видеоигры, кассовые сборы которых исчисляются миллиардами долларов. Например, программное обеспечение Pixar для трехмерной анимации помогло заработать более 13 млрд долларов. Серия трехмерных экшен-игр *Call of Duty*, выпущенных компанией Activision, принесла ей более 16 млрд долларов, а *Grand Theft Auto V*, выпущенная компанией Rockstar, — 6 млрд долларов.

Каждый из этих проектов основан на вычислениях с трехмерными векторами, или тройками чисел, вида $v = (x, y, z)$. Тройки чисел достаточно, чтобы найти точку в трехмерном пространстве относительно контрольной точки, называемой началом координат. Как показано на рис. 1.7, каждое из трех чисел говорит, как далеко нужно пройти в одном из трех перпендикулярных направлений, чтобы достичь заданной точки.

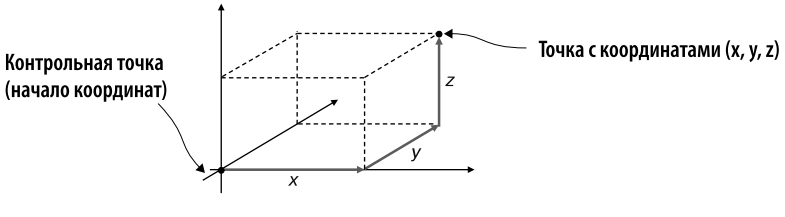

Рис. 1.7. Определение точки в трехмерном пространстве с помощью вектора из трех чисел: *x*, *y* и *z*

Любой трехмерный объект, от рыбы-клоуна в игре «В поисках Немо» до авианосца в *Call of Duty*, можно определить в компьютере как набор трехмерных векторов. В программе каждый из этих объектов выглядит как список троек чисел типа `float`. Три тройки чисел определяют три точки в пространстве, которые могут представлять треугольник (рис. 1.8), например:

```
triangle = [(2.3,1.1,0.9), (4.5,3.3,2.0), (1.0,3.5,3.9)]
```

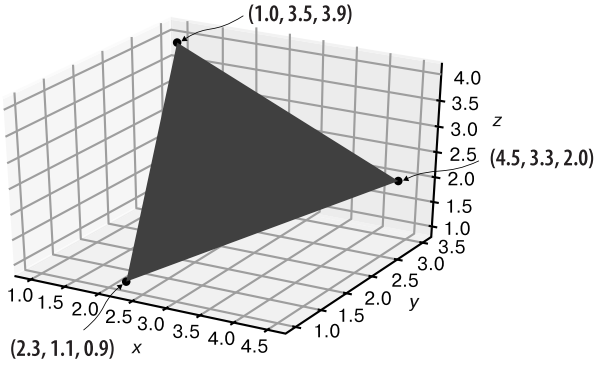

Рис. 1.8. Трехмерный треугольник, образованный тремя тройками чисел, обозначающими координаты вершин

Комбинируя множество треугольников, можно определить поверхность трехмерного объекта, и чем больше треугольников меньшего размера использовать, тем более гладким получится объект. На рис. 1.9 показаны варианты определения трехмерной сферы с помощью все большего числа треугольников все меньшего и меньшего размера.

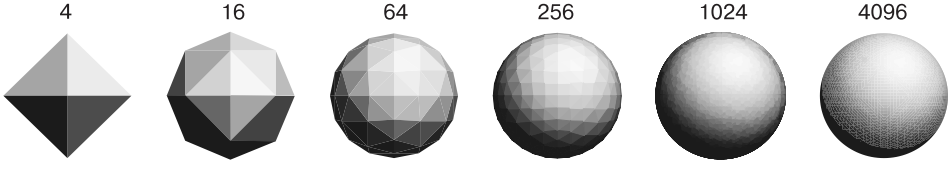

Рис. 1.9. Трехмерные сферы, образованные разным количеством треугольников

В главах 3 и 4 вы узнаете, как использовать трехмерную векторную математику для преобразования трехмерных моделей в двухмерные изображения с тенями, как на рис. 1.9. Кроме того, трехмерные модели должны быть гладкими, чтобы иметь реалистичный вид в игре или фильме, и должны двигаться и изменяться реалистичным образом. Это означает, что рисованные объекты должны подчиняться законам физики, которые тоже выражаются в терминах трехмерных векторов.