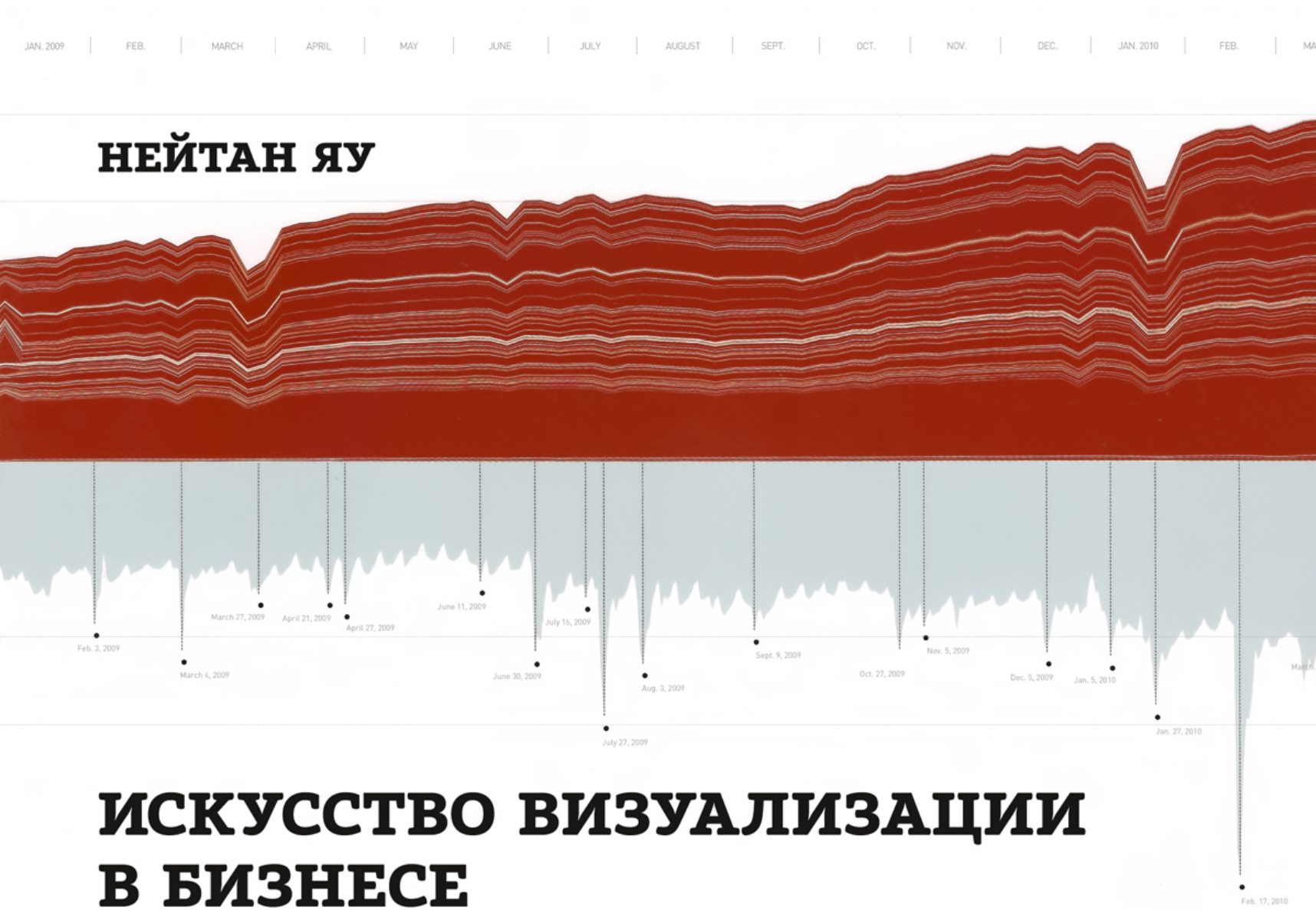
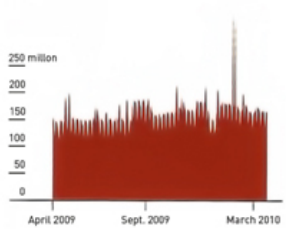
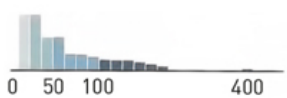
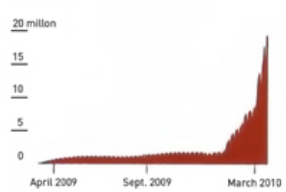


ИСКУССТВО ВИЗУАЛИЗАЦИИ В БИЗНЕСЕ

Как представить сложную информацию простыми образами







Visualize This

The FlowingData Guide to Design, Visualization, and Statistics

Nathan Yau

Wiley Publishing, Inc.

Искусство визуализации в бизнесе

Как представить сложную информацию простыми образами

Нейтан Яу

Перевод с английского Светланы Кировой

Издательство «Манн, Иванов и Фербер»
Москва
2013

УДК 65.012.2
ББК 65.291.34
Я88

Издано с разрешения John Wiley & Sons International Rights Inc.
и агентства Александра Корженевского.
На русском языке публикуется впервые

Яу Н.
Я88 Искусство визуализации в бизнесе. Как представить сложную информацию простыми образами / Нейтан Яу; пер. с англ. Светланы Кировой. — М. : Манн, Иванов и Фербер, 2013. — 352 с.

Визуализация в бизнесе — это умение представить данные в таком виде, который позволит их быстро анализировать, эффектно подавать, ну и конечно, применять в жизни. Прочитав эту книгу, вы научитесь собирать и форматировать информацию, создавать на ее основе диаграммы, графики и карты высокого качества.
Используя для целей визуализации Adobe Illustrator, интерактивную графику с HTML, CSS, JavaScript, Flash-графику, а также карты, созданные в R, Python и SVG, вы сможете креативно подавать свои данные и рассказывать с их помощью увлекательные истории.
Для тех, кто работает с большим количеством информации: руководителей проектов, аналитиков, консультантов, маркетологов.

УДК 65.012.2
ББК 65.291.34

ISBN 978-5-91657-737-2

Оглавление

Об авторе, о техническом редакторе, благодарности 7

Введение 9

1 Как рассказать историю с помощью данных 19

Больше чем числа 20
Что искать 26
Дизайн 30
Закругляясь 38

2 Как обращаться с данными 39

Сбор данных 40
Форматирование данных 54
Закругляясь 67

3 Выбор инструментов для визуализации данных 69

Готовые решения для визуализации 70
Программирование 78
Иллюстрирование 91
Маппинг 95
Изучите свои возможности 103
Закругляясь 104

4 Визуализация паттернов во времени 105

Что искать во времени 106
Дискретные моменты времени 107
Непрерывные данные 130
Закругляясь 142

5 Визуализация пропорций 145

Что искать в пропорциях 146
Части целого 146

Пропорции во времени 169
Закругляясь 183

6 Визуализация зависимостей 185

Какие зависимости искать 186
Корреляция 186
Распределение 205
Сравнение 218
Закругляясь 230

7 Как выявить отличия 231

Что искать 232
Сравнение по нескольким переменным 232
Сокращение размерности 263
Роиск выбросов 269
Закругляясь 274

8 Визуализация пространственных отношений 275

Что искать 276
Отдельные местоположения 276
Регионы 289
Во времени и пространстве 304
Закругляясь 324

9 Прицельный дизайн 325

Подготовьте себя 326
Подготовьте ваших читателей 327
Визуальные подсказки 331
Сто́ящая визуализация 337
Закругляясь 338

Об авторе

С 2007 года Нейтан Яу пишет статьи и создает графику для FlowingData, специализированного сайта, посвященного вопросам визуализации, статистики и дизайна. Работая с такими группами компаний, как New York Times, CNN, Mozilla и SyFy, автор убедился, что инфографика отлично подходит не только для анализа данных, но и для сторителлинга.

Нейтан Яу имеет магистерскую степень по статистике Университета Калифорнии в Лос-Анджелесе, Защитил кандидатскую диссертацию о визуализации персональных данных. Автор книги Data Points: Visualization That Means Something..

О техническом редакторе

Ким Рийс (Kim Rees) — сооснователь компании Periscopic, социально ответственной фирмы, занимающейся визуализацией информации. Сама Ким — личность знаменитая в кругу профессионалов в области визуализации. У нее семнадцатилетний опыт работы в интерактивных медиа. Она публиковалась в Journal of Information Mapping и в сборнике материалов InfoVIS 2010, а также выступала на множестве различных конференций и форумов, среди которых O'Reilly Strata Conference, WebVisions, AIGA Shift и Portland Data Visualization. Степень бакалавра в области компьютерных наук она получила в Нью-Йоркском университете. О ее компании Periscopic писали и в CommArts Insights, и в «Историях успеха» на сайте Adobe, ее работы были удостоены наград VAST Challenge, CommArts Web Picks, а также премии Communication Arts Interactive Annual. Недавно проект компании Periscopic оказался выдвинут на Национальную премию дизайна Cooper-Hewitt.

Благодарности

Эта книга не могла бы появиться без трудов ученых, занимающихся проблемами обработки данных и продолжающих создавать полезный открытый инструментарий для всеобщего пользования. Программные продукты этих разработчиков делают мою жизнь намного легче, и я уверен, что они продолжат нас удивлять — нет предела инновациям.

Хочу также выразить признательность читателям FlowingData, которые помогли мне установить контакт с таким огромным количеством людей, какого я себе даже представить не мог. Именно ради них главным образом я и сел за эту книгу.

Еще я хотел бы поблагодарить издательство Wiley Publishing, позволившее мне написать книгу так, как я хотел, а также Ким Рийс — за то, что помогла сделать эту работу достойной прочтения.

И в заключение я хочу сказать спасибо за поддержку моей жене и родителям, всегда поощрявшим меня в поисках того, что делает меня счастливым.

Моей любящей жене Бий

Введение

Нельзя сказать, что данные — это что-то новое. Люди занимаются количественными измерениями и составлением таблиц уже не одно столетие. Тем не менее в последние годы — с тех пор как я начал писать для FlowingData, моего сайта по дизайну, визуализации и статистике, — я наблюдаю настоящий бум в этой области, и конца-края ему не видно. Совершенствование технологий сделало сбор и хранение данных задачей чрезвычайно простой, а сеть позволяет получать к ним доступ в любой момент, когда захочется. Это изобилие данных, окажись они в хороших руках, способно стать кладезем информации, помогающей принимать более дальновидные решения, излагать свои идеи более убедительно и создавать более объективное представление о том, как люди смотрят на мир и на самих себя.

Значительные сдвиги в публикации правительственных данных произошли в середине 2009 года, когда Соединенные Штаты запустили портал Data.gov. Он представляет собой всеобъемлющий каталог информации, предоставляемой федеральными министерствами и ведомствами, и демонстрирует прозрачность и подотчетность всех этих организаций и должностных лиц. Сайт задумывался для того, чтобы предоставить гражданам страны возможность ознакомиться с тем, на что власти тратят налоговые поступления. А ведь до этого правительство больше походило на черный ящик. Значительная часть данных на Data.gov и раньше лежала в открытом доступе — она находилась на сайтах ведомств, но те были разбросаны по всей Сети. Теперь же все сведения собраны в одном месте и отформатированы так, что их анализ и визуализация стали намного проще. И у Организации Объединенных Наций есть подобный портал — UNdata. Прошло совсем немного времени, и этому примеру последовало Соединенное Королевство, организовав свой Data.gov.uk. Крупные города мира, такие как Нью-Йорк, Сан-Франциско и Лондон, также являются поставщиками большого количества данных.

Открытию коллективной сети также во многом поспособствовало появление тысяч интерфейсов программирования приложений (application programming interfaces, API). Задача была — приободрить разработчиков и соблазнить их сделать что-то со всеми этими данными. Такие приложения, как Twitter и Flickr, имеют API с широким функционалом, что делает возможным создание пользовательского интерфейса, совершенно отличного от того, который можно видеть на самих сайтах. На ProgrammableWeb — ресурсе, занимающемся каталогизацией API, — их представлено более двух тысяч. Не так давно появились и новые приложения, такие как Infochimps и Factual, разработанные специально для предоставления структурированных данных.

У себя, на индивидуальном уровне, вы можете обновить друзей в Facebook, поделиться сведениями о своем местонахождении через Foursquare или пощебетать о том, чем занимаетесь, в Twitter — все это можно сделать парой кликов мышью или нажатием нескольких клавиш на клавиатуре. Более специализированные приложения дают вам возможность вести учет того, что вы едите, сколько вы

весите, каково ваше настроение и многого другого. Какую бы информацию о себе вы ни захотели собрать, наверняка найдется приложение, которое поможет вам это сделать.

Если вокруг собрано столько данных — в магазинах, на складах и в базах, — значит, ситуация созрела для появления людей, которые способны их осмыслить. Сами по себе данные не так уж интересны (по крайней мере, для большей части человечества). Интересна информация, которую можно извлечь из этих данных. Люди хотят знать, о чем говорят их данные, и если вы способны помочь им в этом, то будете весьма востребованы. Вот почему Хол Вариан (Hal Varian), главный экономист Google, говорит, что статистик — самая секси работа грядущего десятилетия, и отнюдь не потому, что статистики — такие красавчики. (Хотя если взглянуть на нас сквозь призму гиковского* шика, то мы очень даже ничего.)

Визуализация

Один из лучших способов исследовать крупную базу данных и попытаться разобраться в ней — это визуализация. Поместите числа в видимое пространство и предоставьте мозгу — своему или ваших читателей — выявить паттерны. В этом деле мы все мастера. Вы сможете разглядеть истории, которые, возможно, никогда бы не увидели, применяя лишь формальные статистические методы.

Джон Тьюки (John Tukey), мой любимый статистик и отец разведочного анализа данных, разбирался в статистических методах и свойствах, как мало кто другой, и верил, что графические методы также имеют законное право на существование. Он был глубоко убежден, что картинки способны открывать нам неожиданное. Вы можете очень многое узнать из данных, просто визуализируя их, а в ряде случаев это и есть все, что вам нужно сделать, чтобы принять информированное решение или рассказать историю.

Например, в 2009 году в Соединенных Штатах произошел значительный рост уровня безработицы. В 2007 году он составлял в среднем 4,6 процента, в 2008 году поднялся до 5,8 процента, а к сентябрю 2009 года дошел уже до 9,8 процента. Однако такие средние по стране величины способны поведать лишь часть истории. Это обобщенные данные по США в целом. Но, может, были какие-то регионы, в которых уровень безработицы оказался выше, чем в других местах? Может, были регионы, которые эта беда вообще обошла стороной?

Карты, представленные на рис. 0.1, рассказывают более полную версию этой истории, вам достаточно лишь взглянуть на них — и вы сможете ответить на вопросы из предыдущего абзаца. Округа, окрашенные в более темный цвет, — это области, в которых уровень безработицы был сравнительно высоким, в то время как в округах, окрашенных светлым, ее уровень оказался относительно низким. В 2009 году вы уже видите множество регионов на западе, где уровень безработицы стал выше 10 процентов. Такая же ситуация сложилась и в большинстве регионов на востоке. А вот регионы Среднего Запада пострадали не так сильно (см. рис. 0.2).

---

* Гик (*англ.* geek) — человек, чрезмерно увлеченный какой-либо темой и по этой причине несколько выпадающий из реальности. В русском языке чаще применяется к людям, одержимым (компьютерными) технологиями. *Прим. пер.*

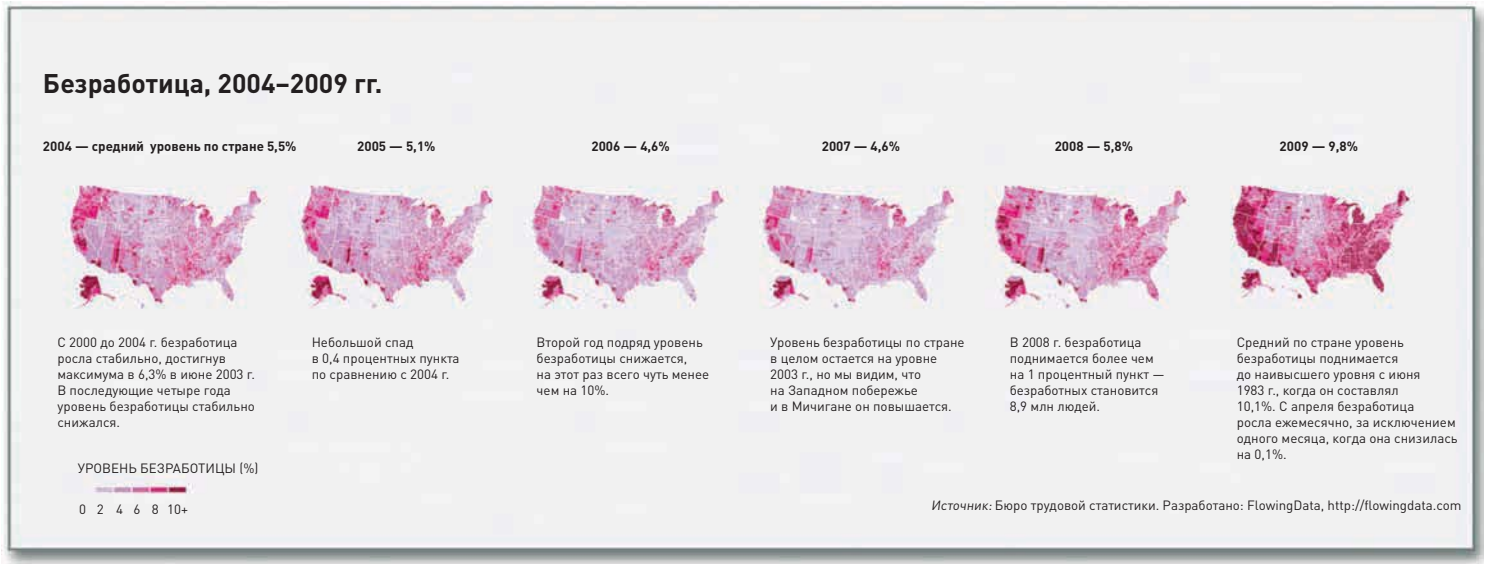


Рис. 0.1. Карты, демонстрирующие уровень безработицы в США с 2004 по 2009 гг.

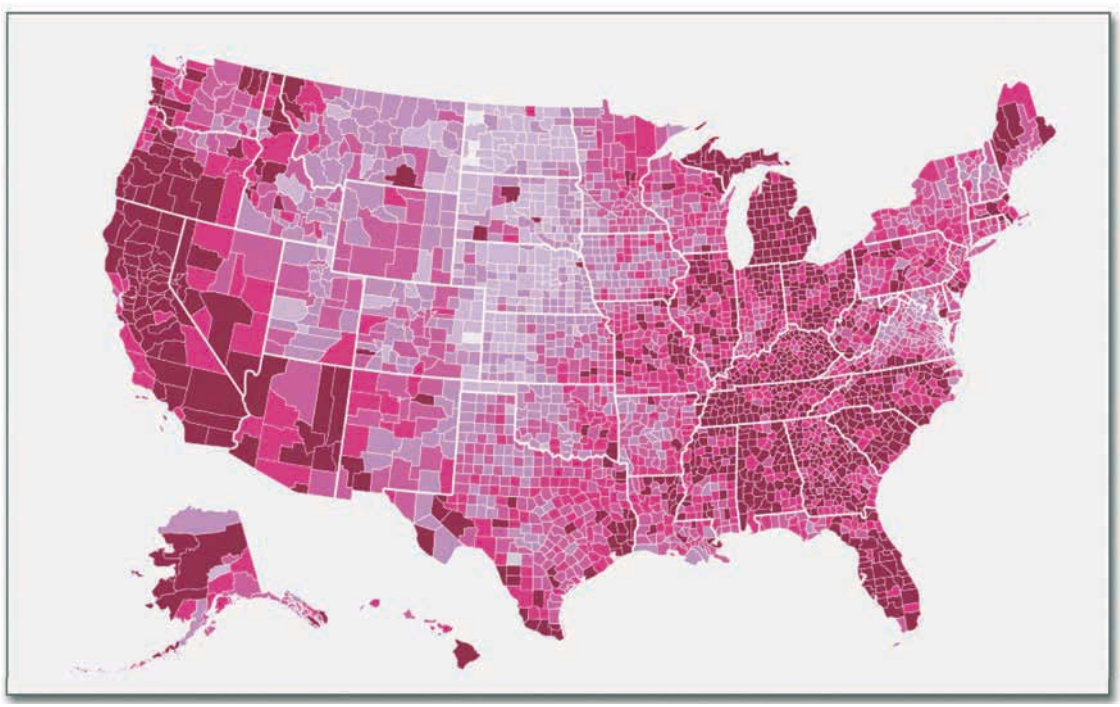


Рис. 0.2. Карта, демонстрирующая уровень безработицы в 2009 г.

Таблица 126. Браки и разводы — общее количество и в расчете на 1000 человек, по штатам, с 1990 по 2007 гг.

(Значение 2443,5 представляет собой 2 443 500 человек. По месту проживания.)

Штат	Браки[1]						Разводы[3]					
	Количество (в тыс.)			На 1000 человек[2]			Количество (в тыс.)			На 1000 человек[2]		
	1990	2000	2007	1990	2000	2007	1990	2000	2007	1990	2000	2007
США[4]	**2443,5**	**2329,0**	**2204,6**	**9,8**	**8,3**	**7,3**	**1182,0**	**НД**	**НД**	**4,7**	**4,1**	**3,6**
Алабама	43,1	45,0	42,4	10,6	10,3	9,2	25,3	23,5	19,8	6,1	5,4	4,3
Аляска	5,7	5,6	5,8	10,2	8,9	8,4	2,9	2,7	3,0	5,5	4,4	4,3
Аризона[5]	36,8	38,7	39,5	10,0	7,9	6,2	25,1	21,6	21,5	6,9	4,4	3,9
Арканзас	36,0	41,1	33,7	15,3	16,0	11,9	16,8	17,9	16,8	6,9	6,9	5,9
Калифорния	237,1	196,9	225,8	7,9	5,9	6,2	128,0	НД	НД	4,3	НД	НД
Колорадо	32,4	35,6	29,2	9,8	8,6	6,0	18,4	НД	21,2	5,5	НД	4,4
Коннектикут	26,0	19,4	17,3	7,9	5,9	4,9	10,3	6,5	10,7	3,2	2,0	3,1
Делавэр	5,6	5,1	4,7	8,4	6,7	5,5	3,0	3,2	3,9	4,4	4,2	4,5
Округ Колумбия	5,0	2,8	2,1	8,2	5,4	3,6	2,7	1,5	1,0	4,5	3,0	1,6
Флорида	141,8	141,9	157,6	10,9	9,3	8,6	81,7	81,9	86,4	6,3	5,3	4,7
Джорджия	66,8	56,0	64,0	10,3	7,1	6,7	35,7	30,7	НД	5,5	3,9	НД
Гавайи	18,3	25,0	27,3	16,4	21,2	21,3	5,2	4,6	НД	4,6	3,9	НД
Айдахо	14,1	14,0	15,4	13,9	11,0	10,3	6,6	6,9	7,4	6,5	5,4	4,9
Иллинойс	100,6	85,5	75,3	8,8	7,0	5,9	44,3	39,1	32,8	3,8	3,2	2,6
Индиана	53,2	334,5	51,2	9,6	5,8	8,1	НД	НД	НД	НД	НД	НД
Айова	24,9	20,3	20,1	9,0	7,0	6,7	11,1	9,4	7,8	3,9	3,3	2,6
Канзас	22,7	22,2	18,6	9,2	8,3	6,7	12,6	10,6	9,2	5,0	4,0	3,3
Кентукки	49,8	39,7	33,6	14,5	10,0	7,9	21,8	21,6	19,7	5,8	5,4	4,6
Луизиана	40,4	40,5	32,8	9,6	9,3	7,6	НД	НД	НД	НД	НД	НД
Мэн	11,9	10,5	10,1	9,7	8,3	7,7	5,3	5,8	5,9	4,3	4,6	4,5
Мэриленд	46,3	40,0	35,5	9,7	7,7	6,3	16,1	17,0	17,4	3,4	3,3	3,1
Массачусетс	47,7	37,0	38,4	7,9	6,0	6,0	16,8	18,6	14,5	2,8	3,0	2,2
Мичиган	76,1	66,4	59,1	8,2	6,7	5,9	40,2	39,4	35,5	4,3	4,0	3,5
Миннесота	33,7	33,4	29,8	7,7	6,9	5,7	15,4	14,8	НД	3,5	3,1	НД
Миссисипи	24,3	19,7	15,7	9,4	7,1	5,4	14,4	14,4	14,2	5,5	5,2	4,9
Миссури	49,1	43,7	39,4	9,6	7,9	6,7	26,4	26,5	22,4	5,1	4,8	3,8
Монтана	6,9	6,6	7,1	8,6	7,4	7,4	4,1	2,1	3,6	5,1	2,4	3,7
Небраска	12,6	13,0	12,4	8,0	7,8	7,0	6,5	6,4	5,5	4,0	3,8	3,1
Невада	120,6	144,3	126,4	99,0	76,7	49,3	13,3	18,1	16,6	11,4	9,6	6,5
Нью-Гэмпшир	10,5	11,6	9,4	9,5	9,5	7,1	5,3	7,1	5,1	4,7	5,8	3,9
Нью-Джерси	58,7	50,4	45,4	7,6	6,1	5,2	23,6	25,6	25,7	3,0	3,1	3,0
Нью-Мексико[5]	13,3	14,5	11,2	8,8	8,3	5,7	7,7	9,2	8,4	4,9	5,3	4,3
Нью-Йорк[5]	154,8	162,0	130,6	8,6	8,9	6,8	57,9	62,8	55,9	3,2	3,4	2,9
Северная Каролина	51,9	65,6	68,1	7,8	8,5	7,5	34,0	36,9	37,4	5,1	4,8	4,1
Северная Дакота	4,8	4,6	4,2	7,5	7,3	6,6	2,3	2,0	1,5	3,6	3,2	2,4
Огайо	98,1	88,5	70,9	9,0	7,9	6,2	51,0	49,3	37,9	4,7	4,4	3,3
Оклахома	33,2	15,6	26,2	10,6	4,6	7,3	24,9	12,4	18,8	7,7	3,7	5,2
Орегон	25,3	26,0	29,4	8,9	7,8	7,8	15,9	16,7	14,8	5,5	5,0	4,0
Пенсильвания	84,9	73,2	71,1	7,1	6,1	5,7	40,1	37,9	35,3	3,3	3,2	2,8
Род-Айленд	8,1	8,0	6,8	8,1	8,0	6,4	3,8	3,1	3,0	3,7	3,1	2,8
Южная Каролина	55,8	42,7	31,4	15,9	10,9	7,1	16,1	14,4	14,4	4,5	3,7	3,3
Южная Дакота	7,7	7,1	6,2	11,1	9,6	7,7	2,6	2,7	2,4	3,7	3,6	3,1
Теннесси	68,0	88,2	65,6	13,9	15,9	10,6	32,3	33,8	29,9	6,5	6,1	4,9
Техас	178,6	196,4	179,9	10,5	9,6	7,5	94,0	85,2	79,5	5,5	4,2	3,3
Юта	19,4	24,1	22,6	11,2	11,1	8,6	8,8	9,7	8,9	5,1	4,5	3,4
Вермонт	6,1	6,1	5,3	10,9	10,2	8,6	2,6	5,1	2,4	4,5	8,6	3,8
Вирджиния	71,0	62,4	58,0	11,4	9,0	7,5	27,3	30,2	29,5	4,4	4,3	3,8
Вашингтон	46,6	40,9	41,8	9,5	7,0	6,5	28,8	27,2	28,9	5,9	4,7	4,5
Западная Вирджиния	13,0	15,7	13,0	7,2	8,7	7,2	9,7	9,3	9,0	5,3	5,2	5,0
Висконсин	38,9	36,1	32,2	7,9	6,8	5,8	17,8	17,6	16,1	3,6	3,3	2,9
Вайоминг	4,9	4,9	4,8	10,7	10,3	9,3	3,1	2,8	2,9	6,6	5,9	5,5

НД — нет данных. 1. Данные на основе подсчета заключенных браков, за исключением отмеченных. 2. На основании общего количества населения, проживающего на территории; по данным пересчета от 1 апреля 1990 и 2000 гг. и по оценкам к 1 июля за все остальные годы. 3. Включая аннулированные браки. 4. Данные о количестве браков и разводов по США в целом оценочные и включают также штаты, по которым нет информации. Начиная с 2000 г. данные об уровне разводов основываются на комбинированной системе подсчета населения в отчетных штатах и в округе Колумбия. Сбор детализированных данных о браках и разводах был отменен в январе 1996 г. 5. Некоторые данные основаны на количестве выданных свидетельств о браке.

Источник: Национальный центр статистики в области здравоохранения США. "Births, Marriages, Divorces and Deaths: Provisional Data for 2007, Vol. 56, №21. July 14, 2008" и предыдущие отчеты

Рис. 0.3. Таблица из «Статистического ежегодника Соединенных Штатов»

Вы бы не смогли выявить эти географические и временны́е закономерности так быстро, если бы перед вами была только сводная таблица, и уж точно не сумели бы это сделать, располагая лишь средними по стране величинами. И хотя при наличии данных на уровне округов картина становится более сложной, большинство людей тем не менее способны интерпретировать эти карты. Такие карты помогают политикам решать, куда направлять финансовую помощь или другие формы поддержки.

А самое замечательное — то, что все данные, использованные для создания представленных выше карт, абсолютно бесплатны и доступны широкой общественности на сайте Бюро трудовой статистики. И хотя «нарыть» их было не то чтобы очень просто с такой устаревшей системой представления данных, как у них, так или иначе все тамошние цифры в вашем распоряжении: они сидят и ждут, чтобы кто-нибудь их малость обработал визуально.

«Статистический ежегодник Соединенных Штатов», например, существует в виде сотен таблиц данных (рис. 0.3), но в нем нет ни одного графика. Это неплохая возможность представить всеобъемлющую картину страны. Вот уж действительно интересная штука. Некоторое время назад я перевел часть таблиц в диаграммы просто так, ради идеи. На рис. 0.4 вы можете увидеть динамику браков и разводов, почтовых тарифов, потребления электроэнергии и кое-чего еще. В первом варианте,

то есть в таблице, разобраться в данных непросто, и все, что у вас получится извлечь из них, — это отдельные величины. А вот в графическом варианте вы легко можете заметить тенденции и паттерны и с одного взгляда провести сравнение.

Такие поставщики новостей, как New York Times и Washington Post, прекрасно справляются с задачей делать данные более доступными и наглядными. Они, наверное, лучше многих других пользуются всей этой открытой информацией, каждый день рассказывая читателям все новые и новые истории. Иногда графики с данными применяются для обогащения истории иной точкой зрения, а в других случаях графики, собственно, и излагают всю историю.

С развитием онлайн-медиа диаграммы и графики получили еще большее распространение. Сегодня в новостных компаниях существуют специальные службы, которые занимаются только интерактивами, или только диаграммами, или только картами. У New York Times, например, есть даже отдел новостей, созданный исключительно для работы с тем, что они называют «компьютеризированными репортажами». Тамошние журналисты специализируются на новостях с цифрами. И отделу графики в New York Times не привыкать к работе с огромными массивами данных.

Визуализация пробилась также в поп-культуру. Фирма Stamen Design, занимающаяся визуализацией и широко известная своими онлайн-интерактивами, в последние несколько лет готовила для ежегодной церемонии вручения премий MTV Video Music Awards трекеры твитов.

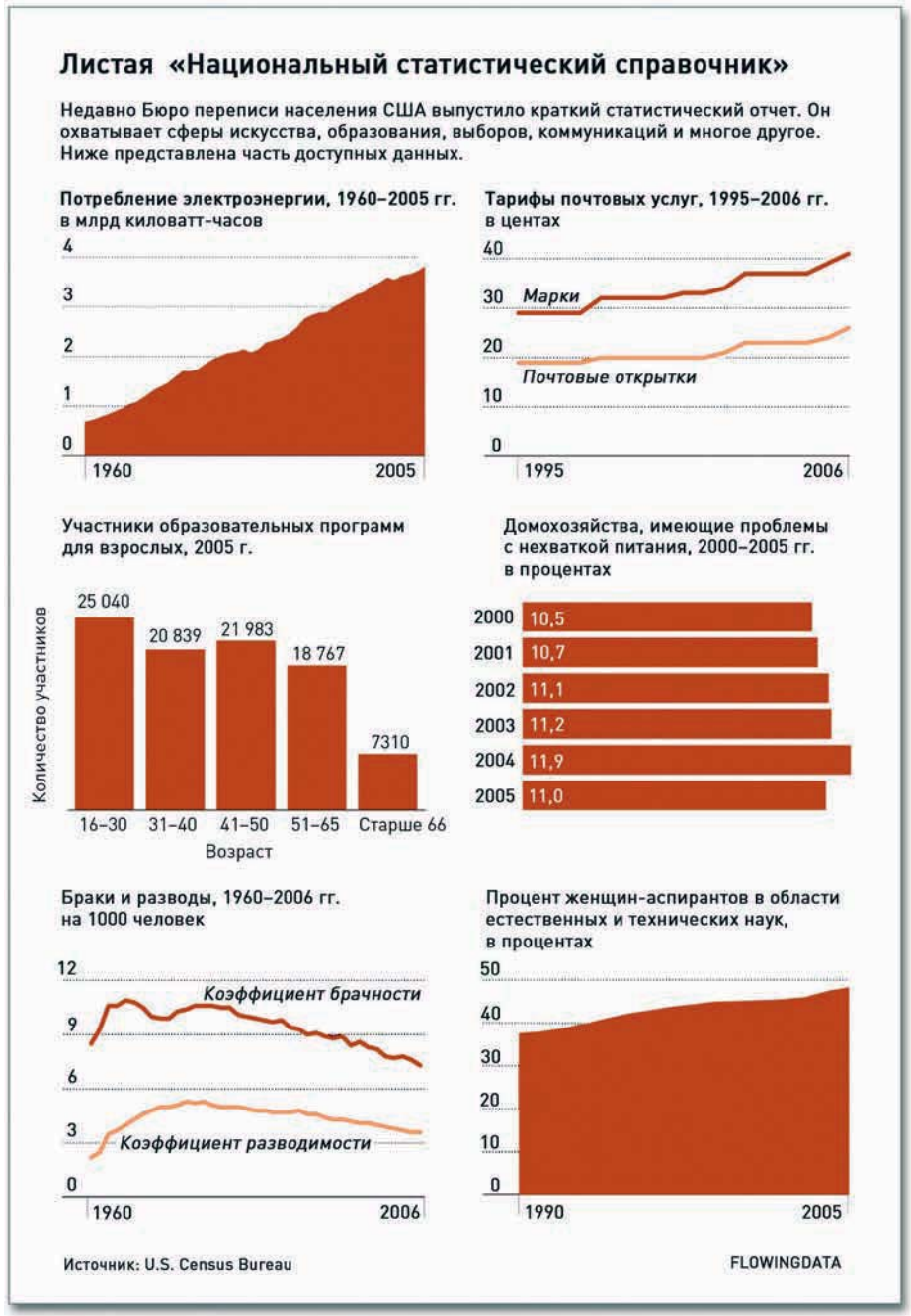


Рис. 0.4. Графический вид данных из «Статистического ежегодника Соединенных Штатов»

Каждый год Stamen делала что-то новое, но суть была одна: показать, о чем говорят люди в Twitter в реальном времени. Когда в 2009 году во время речи получившей награду Тейлор Свифт (Taylor Swift) произошла известная неприятность с Канье Уэстом (Kanye West)*, через трекер можно было сразу увидеть, что люди об этом думают.

На данном этапе вы подходите к визуализации не столько аналитически, сколько на уровне ощущений. Определение визуализации кажется несколько туманным. На протяжении длительного времени визуализация была чем-то связанным с количественными показателями. От вас требуется своими инструментами выявлять паттерны, а от паттернов — каким-то образом помогать вам в процессе анализа. Визуализация — это не просто получение объективных голых фактов. Как и в случае с трекером Stamen, она почти всегда связана в первую очередь с фактором развлечения. Это способ дать зрителям возможность смотреть шоу с присуждением призов и в процессе взаимодействовать с остальными людьми. Другим прекрасным тому примером может служить творчество Джонатана Харриса (Jonathan Harris). Харрис выстраивает свои работы, такие как «Все у нас хорошо» и «Охота на китов», скорее, вокруг историй, нежели вокруг аналитических прозрений, и эти истории базируются на человеческих эмоциях, вызванных цифрами и аналитикой.

Схемы и диаграммы также переросли состояние просто инструментов и служат теперь средствами коммуницирования идей — они способны даже на шутки. Такие сайты, как GraphJam и Indexed, используют диаграммы Венна, секторные диаграммы и пр. для того, чтобы представлять популярные песни или показывать, что комбинирование

Рис. 0.5. Цитаты из кинофильмов в графической форме

* Во время выступления Тейлор Свифт, победившей в номинации за лучшее женское видео, Канье Уэст поднялся на сцену, отобрал у нее микрофон и заявил, что лучшим он считает ролик Бейонсе. *Прим. пер.*

черного, белого и красного равносильно выпуску коммунистической газеты или убийству панды. «Недозагрузка данных» (Data Underload) — серия юмористических постов, которые я публикую на FlowingData, — это моя собственная «проба пера» в данном жанре. Каждый день я веду наблюдения и выкладываю их в форме диаграмм. На рис. 0.5 таким образом проиллюстрированы цитаты из известных фильмов, вошедших в рейтинги Американского института киноискусства. Получилось нечто абсолютно нелепое, но забавное (по крайней мере, для меня).

Итак, что же такое визуализация? Ну, все зависит от того, с кем вы разговариваете. Некоторые люди утверждают, что визуализация — это традиционные схемы и графики. Другие придерживаются более либеральных взглядов. Для них все, что способно иллюстрировать данные, и есть визуализация, и не имеет значения, что это: произведение data-арта или таблица в Microsoft Excel. Я лично больше склоняюсь к последним, хотя, бывает, иногда обнаруживаю себя среди членов первой группы. В конце концов, все это не так уж и важно. Просто делайте то, что подходит для ваших целей.

На каком бы определении визуализации вы ни остановились, когда вы станете создавать диаграммы для своей презентации, анализировать большой массив цифровой информации или готовить новостной репортаж, содержащий некие данные, вы в конечном счете будете искать правды. В какой-то момент ложь и статистика стали почти синонимами, но на самом деле лгут не числа. Лгут люди, использующие числа. Иногда они делают это намеренно, реализуя какой-то план, но в большинстве случаев подобное происходит неумышленно. Когда вы не знаете, как правильно составить диаграмму или как подать данные беспристрастно, есть большая вероятность, что у вас появится некая информационная свалка, дающая совершенно искривленное представление о реальности. Однако если вы усвоите годные приемы визуализации и научитесь работать с данными, вы сможете уверенно излагать свои идеи и радоваться своим открытиям.

▶ Другие публикации из серии «Недозагрузка данных» («Data Underload») можно найти на сайте FlowingData по адресу: `http://datafl.ws/underload`

Как научиться работать с данными

Я начал изучать статистику на первом курсе колледжа. Это был обязательный учебный предмет для получения никак не связанной со статистикой ученой степени по электротехнике. Я слышал много ужасных историй, но, в отличие от описываемых в этих байках преподавателей, мой профессор оказался большим энтузиастом своего дела и занимался им с явным удовольствием. Читая свои лекции, он быстро передвигался вверх-вниз по ступеням зала и размашисто жестикулировал, задевая студентов, мимо которых проходил. До того дня, как мне кажется, у меня не было более вдохновленного учителя, и это, несомненно, повлияло на меня — я увлекся миром данных. В итоге четыре года спустя я закончил колледж специалистом в области статистики.

На протяжении всего моего обучения в бакалавриате статистика сводилась для меня к анализу данных, распределению и проверке гипотез, и мне это нравилось. Было забавно разглядывать

наборы данных и выявлять в них различные тенденции, паттерны и корреляции. Но когда я перешел в магистратуру, мои взгляды изменились, и все стало еще интереснее.

Теперь заниматься статистикой не означало лишь выявлять паттерны и проверять гипотезы (что, как оказалось, во многих случаях не так уж и полезно). Стоп, нет, последние слова я беру назад. Статистика по-прежнему состояла из этого, но тем не менее все воспринималось по-другому. Статистика, как я понял, — это рассказывание историй посредством данных. Вы берете кучку данных, описывающих ваш материальный мир, и анализируете их не только ради того, чтобы найти некие корреляции, но и чтобы выяснить, что же происходит вокруг вас. И когда вы это выясняете, полученные истории оказываются способны помочь вам решить некие проблемы и задачи реального мира — такие как снижение уровня преступности, улучшение здравоохранения и облегчение движения автотранспорта — или хотя бы оставаться человеком, информированным по всем этим вопросам.

Многие люди не видят связи между данными и реальной жизнью. Именно поэтому, когда я сообщал, что учусь в магистратуре на статистика, многие ребята говорили мне, что в колледже ненавидели курс статистики. Я знаю, вы не допустите подобной ошибки, верно? Вы ведь уже начали читать эту книгу.

Как обрести нужные умения, чтобы научиться извлекать пользу из данных? Для этого вы можете прослушать курс лекций, как я, но можете учиться и самостоятельно, опытным путем. Это ведь именно то, чем люди занимаются большую часть времени в магистратуре.

Точно так же обстоят дела и с визуализацией и информационной графикой. Вам не нужно быть великим дизайнером, чтобы делать классную графику. Равно как не нужно быть кандидатом наук в области статистики. Вам достаточно иметь страстное желание учиться и — как это бывает почти с каждым делом в жизни — практиковаться, чтобы совершенствоваться.

Первую диаграмму с данными я создал классе в четвертом. Сделал я ее, участвуя в школьной ярмарке научных проектов. Мой партнер по проекту и я исследовали (очень углубленно, можете не сомневаться), по какой поверхности улитки передвигаются быстрее всего. Мы ставили улиток на гладкие и шершавые поверхности и засекали время, чтобы посмотреть, сколько минут им понадобится, чтобы пройти определенное расстояние. Таким образом мы получили данные о времени для различных поверхностей. На их основе я создал столбцовую диаграмму. Я не помню, догадался ли я выстроить столбцы по величине от наименьшего к наибольшему, но зато я хорошо помню, как боролся с программой Excel. Однако на следующий год, когда мы выяснили, что именно предпочитают есть мучные каштановые хрущаки, диаграмма получилась просто блестящей. Когда вы освоите основной набор функциональных возможностей и научитесь работать с программным обеспечением, разобраться с остальным не составит трудности. И если это не отличный пример обучения на собственном опыте, тогда я не знаю, что еще может быть примером. Кстати, быстрее всего улитки передвигались по стеклу, а каштановые хрущаки предпочитали сухой завтрак Grape Nuts — на тот случай, если вам это интересно.

Здесь мы будем говорить о самых основополагающих моментах, но по сути своей процесс выглядит аналогично, какую бы программу или язык программирования вы ни взялись изучать.

Если вы за всю жизнь не написали ни строчки кода, тогда R* — вычислительная среда, которой отдают предпочтение многие статистики, — может показаться вам пугающей. Но после того как вы изучите несколько примеров, вы быстро набьете руку. Данная книга поможет вам в этом.

Я говорю вам это, потому что именно так я и учился. Помню, как я впервые углубился в дизайнерские аспекты визуализации. Было это летом после второго курса магистратуры, когда я получил потрясающее известие: меня взяли на стажировку в должности редактора графики в New York Times. До того момента графика для меня всегда была лишь инструментом анализа (в том числе и столбцовые диаграммы для школьной ярмарки научных проектов), а эстетика и дизайн не имели особо большого значения — если они вообще имели для меня хоть какое-нибудь значение. Мне и в голову не приходило, сколь велика роль данных в журналистике.

Чтобы подготовиться к стажировке, я прочитал все книги по дизайну, которые смог достать, и проштудировал руководство пользователя Adobe Illustrator, потому что знал: в New York Times работают именно с этой программой. Но только когда я действительно начал делать диаграммы и графики, я стал по-настоящему чему-то учиться. Обучаясь в процессе работы, вы окажетесь вынуждены очень быстро набираться знаний и навыков, необходимых в деле. И по мере того, как вы будете работать со все бóльшими массивами данных и создавать все больше диаграмм, ваши умения станут развиваться активнее.

Как читать эту книгу

Эта книга построена на примерах и написана с целью дать вам знания, необходимые для создания различных объектов информационной графики от начала и до конца. Вы можете прочитать ее от корки до корки, а можете выхватить только нужные вам идеи, если у вас уже есть какие-то данные или если вы владеете какими-то способами визуализации. Главы организованы так, что примеры в них независимы и самодостаточны. Если вы новичок в этой области, первые главы будут вам особенно полезны. В них рассказывается о том, как подходить к имеющимся данным, что в них искать и какими инструментами пользоваться. Вы узнаете, где именно находить данные и как их форматировать и готовить для визуализации. Далее представлены приемы визуализации, структурированные по типу данных и по типу историй, которые с их помощью вы можете рассказать. Но помните: говорить всегда должны сами данные.

Какой бы подход к чтению этой книги вы ни выбрали, я горячо рекомендую вам читать ее, сидя за работающим компьютером, чтобы прорабатывать примеры шаг за шагом и просматривать все ресурсы, упомянутые в примечаниях и ссылках. Вы можете также скачать коды и файлы данных и опробовать работающие демоверсии на сайтах `http://www.wiley.com/go/visualizethis` и `http://book.flowingdata.com`.

Чтобы сделать сказанное выше предельно ясным, на рис. 0.6 я представляю вам схему, которая поможет понять, что именно вам необходимо. Развлекайтесь в свое удовольствие!

---

* R — язык программирования для статистической обработки данных и работы с графикой; свободная программная среда с открытым исходным кодом. *Прим. пер.*

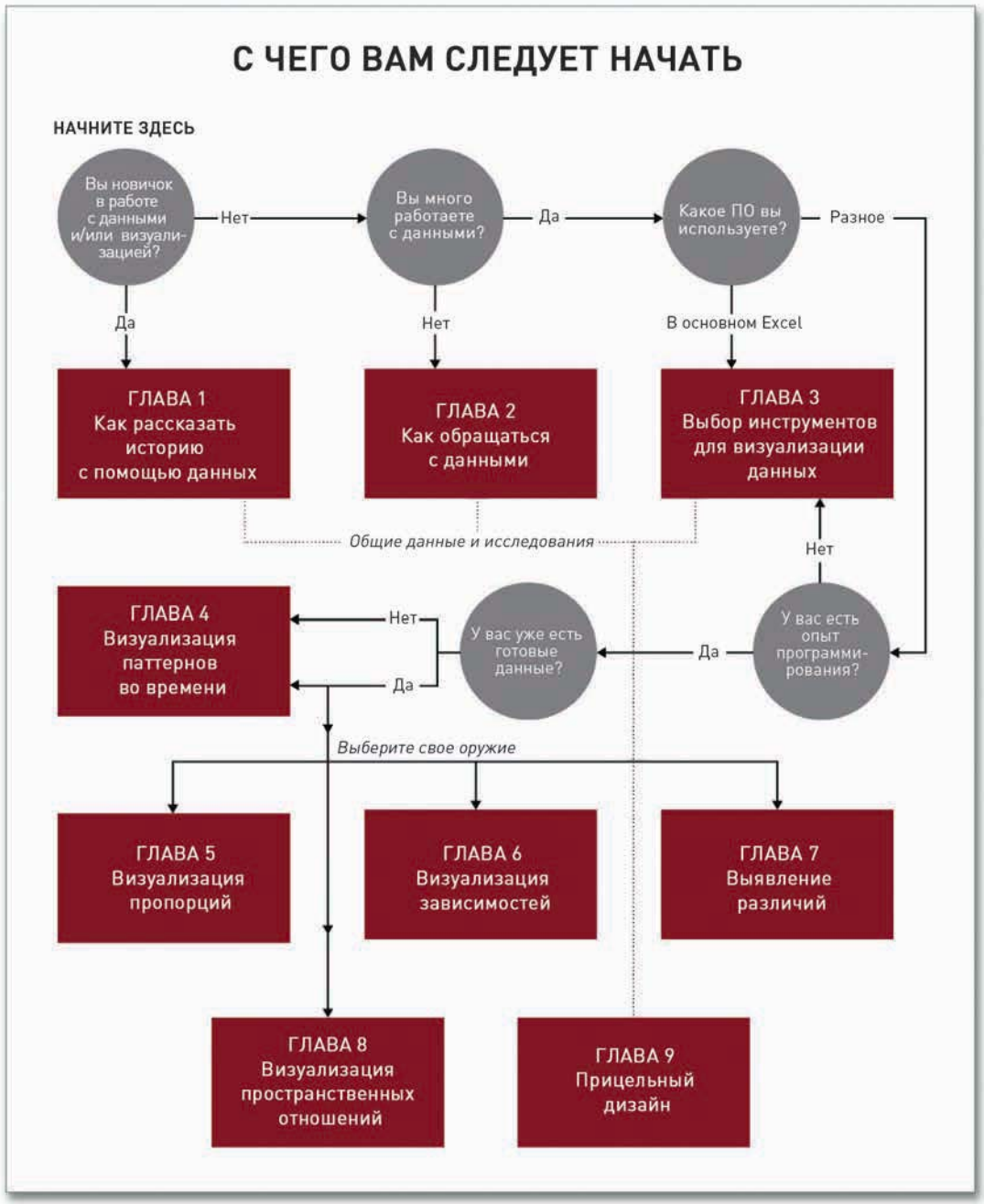


Рис. 0.6. С какого места следует начать читать эту книгу

Как рассказать историю с помощью данных

1

Подумайте обо всех популярных книгах по визуализации, которые существуют, — о тех, про которые вы постоянно слышите на лекциях или про которые читаете в блогах, а еще о тех, которые пришли вам на ум прямо сейчас, пока вы пробегали глазами по этим строчкам. Что у них всех общего? Они все рассказывают интересные истории. Вероятно, эти истории должны были убедить вас в чем-то. Или, возможно, подтолкнуть вас сделать что-то, познакомить с новой информацией или заставить пересмотреть свои предвзятые представления о реальности. Как бы то ни было, по-настоящему хорошая визуализация данных — неважно, большая или маленькая, созданная из любви к искусству или для презентации — помогает вам понять, о чем именно способны рассказать данные.

Больше чем числа

Взглянем правде в глаза. Данные могут быть чем-то довольно досадным и раздражающим, если вы не знаете, что вы ищете, или не понимаете, есть ли в них нечто такое, что следует искать в первую очередь. Тогда они превращаются в груду цифр и слов, которые не имеют иного смысла, кроме своего непосредственного значения. В этом и состоит великий смысл статистики и визуализации — в том, что они помогают увидеть, что именно стоит за всем этим. Помните: данные есть отражение реальной жизни. Это не просто груда каких-то чисел. В этой груде содержится множество историй. В ней есть и смысл, и правда, и красота. И, как и в реальной жизни, эти истории иногда бывают простыми и искренними, а иногда — сложными и иносказательными. Некоторые истории как будто взяты из учебника. Другие похожи на роман. И от вас — статистиков, программистов, дизайнеров и специалистов в обработке данных — зависит, как рассказать ту или иную историю.

В этом и состоял один из первых уроков, которые я усвоил, начав заниматься в магистратуре. Должен признаться, что до своего поступления туда я думал, что статистика — это чистый анализ, а данные — результат механического процесса сбора и обработки информации. В большинстве случаев так оно и есть. Ну, я же окончил колледж по специальности «электротехника», а потому неудивительно, что данные виделись мне именно в таком свете.

Не поймите меня неправильно. Я не хочу сказать этим ничего плохого. Просто со временем я убедился, что данные, будучи вещью объективной, очень часто имеют при этом и человеческое измерение.

В качестве примера давайте посмотрим еще раз на безработицу. Выдать средние значения по штатам нетрудно, но, как вы сами убедились, положение дел даже в пределах одного штата может сильно колебаться. В двух соседних районах ситуация может существенно разниться. Возможно, среди ваших знакомых есть человек, который потерял работу в последние несколько лет, и, как говорится, он для вас не просто статистическая единица, правильно? Числа представляют собой живых людей, так что вам следует подходить к данным именно так. Вам не нужно рассказывать историю каждого из людей. И все же есть тонкое, но от этого не менее важное различие между увеличением уровня безработицы на 5 процентов и несколькими сотнями тысяч людей, оставшихся не у дел. В первом случае сказанное воспринимается как довольно абстрактная цифра, а во втором данные имеют совсем иную смысловую нагрузку.

Журналистика

В полной мере все вышесказанное до меня дошло во время стажировки в газете New York Times. Длилась она всего три летних месяца после второго курса магистратуры, но изменила мой подход к данным навсегда. Я не только научился создавать графику для новостей. Я научился преподносить данные как новость, а вместе с тем узнал многое о дизайне, организации и проверке фактов, а также об их поиске и изучении.

Однажды я посвятил день задаче проверки трех чисел из целого массива данных, а все потому, что когда отдел графики New York Times создает некую диаграмму, он должен удостовериться в точности сведений. Только убедившись в надежности полученной информации, мы переходили к созданию презентации. Именно такое внимание к деталям делает графику в этой газете столь хорошей.

Посмотрите на любую схему или диаграмму в New York Times. Данные в ней всегда представлены кратко, ясно и чрезвычайно красиво. Однако что это значит? Когда вы смотрите на инфографику, вы получаете возможность понять данные. Важные моменты или области сопровождаются комментариями; символы и цвета аккуратно объяснены в легенде или в примечаниях; а еще Times делает все для того, чтобы история, содержащаяся в данных, прочитывалась аудиторией легко. Это уже не просто диаграмма или схема. Это графика.

Графика, представленная на рис. 1.1, похожа на ту, которую вы можете найти в New York Times. На ней показано, как увеличивается вероятность того, что вы умрете в ближайший год, в зависимости от вашего возраста.

Рис. 1.1. Вероятность смерти с учетом вашего возраста

▶ Некоторые из лучших диаграмм и графиков New York Times можно посмотреть на странице `http://datafl.ws/nytimes`.

В основе этого рисунка лежит простой линейный график. Однако элементы дизайна помогают лучше рассказать историю. Подписи и примечания поясняют контекст и помогают понять, почему эти данные интересны; а толщина линий и цвет фокусируют глаза на том, что важно.

Требования к дизайнеру графиков и диаграмм не сводятся только к визуализации статистических данных — от него требуется также дать понять, что именно выражает конкретный изображенный элемент.

Искусство

ПРИМЕЧАНИЕ

Дополнительную информацию о том, как журналисты используют данные при подготовке репортажей о текущих событиях, можно получить из документального фильма Джеффа Макги «Журналистика в эпоху данных» (Geoff McGhee "Journalism in the Age of Data"). В нем вы увидите потрясающие интервью с некоторыми из лучших профессионалов в этой области.

New York Times — газета объективная. Она излагает вам данные и дает факты. В этом ей нет равных. На другом конце спектра находится визуализация, которая в меньшей степени связана с аналитикой и в большей — с погружением в эмоции. Именно этим занимаются Джонатан Харрис и Сеп Камвар в своей работе «Все у нас хорошо» (Jonathan Harris, Sep Kamvar, We Feel Fine), представленной на рис. 1.2.

Рис. 1.2. «Все у нас хорошо» (Джонатан Харрис, Сеп Камвар)

В их интерактивном творении собраны предложения и фразы из различных персональных блогов, и все они визуализированы как пузырьки, плывущие в пространстве. Каждый пузырек представляет собой эмоцию и окрашен в соответствующий цвет. В целом создается впечатление, как будто все эти люди несутся в космосе, но если вы посмотрите подольше, то заметите, как пузырьки начинают сбиваться вместе. Попробуйте отсортировать и категоризировать их, используя интерфейс, и вы увидите, как эти, казалось бы, произвольно разбросанные крошечные штучки объединяются. Кликните на какой-нибудь пузырек, чтобы узнать чью-то индивидуальную историю. Это так поэтично и одновременно так познавательно!

Можно привести еще много примеров. Скажем, работа Голана Левина «Мусорка» (Golan Levin, The Dumpster). В ней исследуются записи в блогах, в которых говорится о расставаниях со значимым человеком. Или «Sumedicina» Кима Азендорфа (Kim Asendorf), в которой рассказывается выдуманная история мужчины, убегающего от пораженной коррупцией организации, но не словами, а схемами и диаграммами. Или скульптуры Андреаса Николаса Фишера (Andreas Nicolas Fischer), изображающие экономический спад в Соединенных Штатах.

Основная идея заключается в том, что данные и визуализация не обязательно должны передавать лишь голые, холодные факты. Иногда то, что вам необходимо, это вовсе не аналитические прозрения. Порой бывает полезнее рассказать историю с эмоциональной точки зрения, чтобы подтолкнуть зрителей к размышлениям над имеющимися данными. Подумайте над этим. Не всегда кино должно быть документальным, и не всегда визуализация должна быть в виде традиционных графиков и диаграмм.

▶ Исследуйте человеческие эмоции в интерактивном режиме с живым онлайн-произведением Джонатана Харриса и Сепа Камвара на сайте `http://wefeelfine.org`.

▶ Еще большее количество примеров произведений искусства, созданных с использованием данных, вы найдете на сайте FlowingData: `http://datafl.ws/art`.

Развлечение

Где-то между журналистикой и искусством визуализация нашла себе лазейку и пробилась в область развлечений. Если подумать о данных в более отвлеченном плане — выйти за рамки электронных таблиц и текстовых файлов с разделителями-запятыми и перенестись туда, где оцениваются фотографии и статусы, — заметить это будет нетрудно.

Facebook использовал обновление статусов для определения самого счастливого дня в году, а сайт онлайн-знакомств OkCupid использовал размещаемую на нем информацию для оценки того, в чем и как люди лгут, пытаясь сделать свое цифровое «я» лучше, чем есть на самом деле (рис. 1.3). Такой анализ данных имеет весьма отдаленную связь с развитием бизнеса, увеличением доходов и нахождением сбоев в системе. Но его результаты охватили Всемирную паутину как пожар — именно из-за своей развлекательной ценности. Эти данные раскрывают нам кое-что о нас самих и о нашем обществе.

Так, Facebook обнаружил, что самый счастливый день — это День благодарения, а OkCupid выяснил, что люди склонны к преувеличениям чаще всего тогда, когда говорят о своем росте, и любят приписать себе примерно 5 лишних сантиметров.

▶ Зайдите на блог OkTrends, и вы узнаете еще много чего интересного об онлайн-свиданиях — например, что на самом деле нравится белым людям и как случайно не оплошать: `http://blog.okcupid.com`.

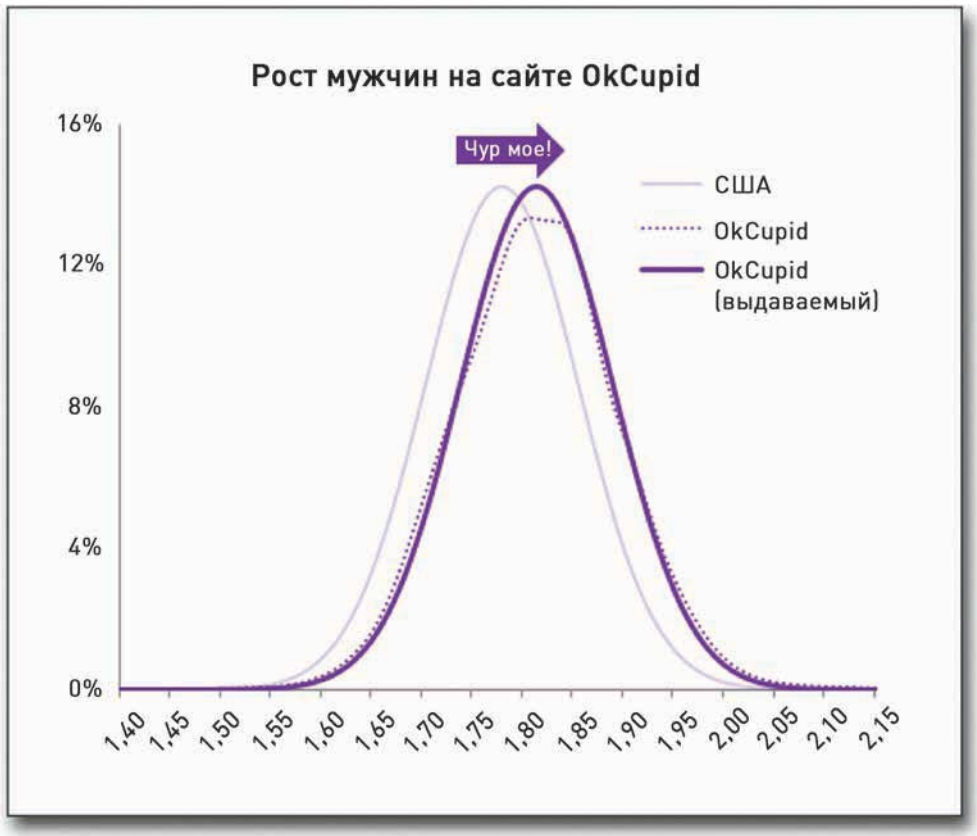


Рис. 1.3. Распределение значений роста мужчин на сайте OkCupid

Побуждение

Конечно, задача историй не всегда состоит в том, чтобы информировать или развлекать людей. Иногда истории используются, чтобы донести мысль о необходимости срочно предпринять некие меры или чтобы подвигнуть людей на какие-то действия. Кто может забыть тот момент в «Неудобной правде», когда Ал Гор встает на ножничный подъемник, чтобы продемонстрировать, как поднимается уровень углекислого газа в атмосфере?

Однако, на мой взгляд, никто не сумел справиться с задачей побуждения людей к действию так хорошо, как это удалось Хансу Рослингу (Hans Rosling), профессору в области международного здравоохранения и директору фонда Gapminder. Используя инструмент под названием «трендалайзер» (рис. 1.4), Рослинг создал анимированное изображение, показывающее изменения уровня бедности в разных странах. Выступление у него получилось просто изумительным, так что если вы его еще не видели, я горячо рекомендую вам это сделать.

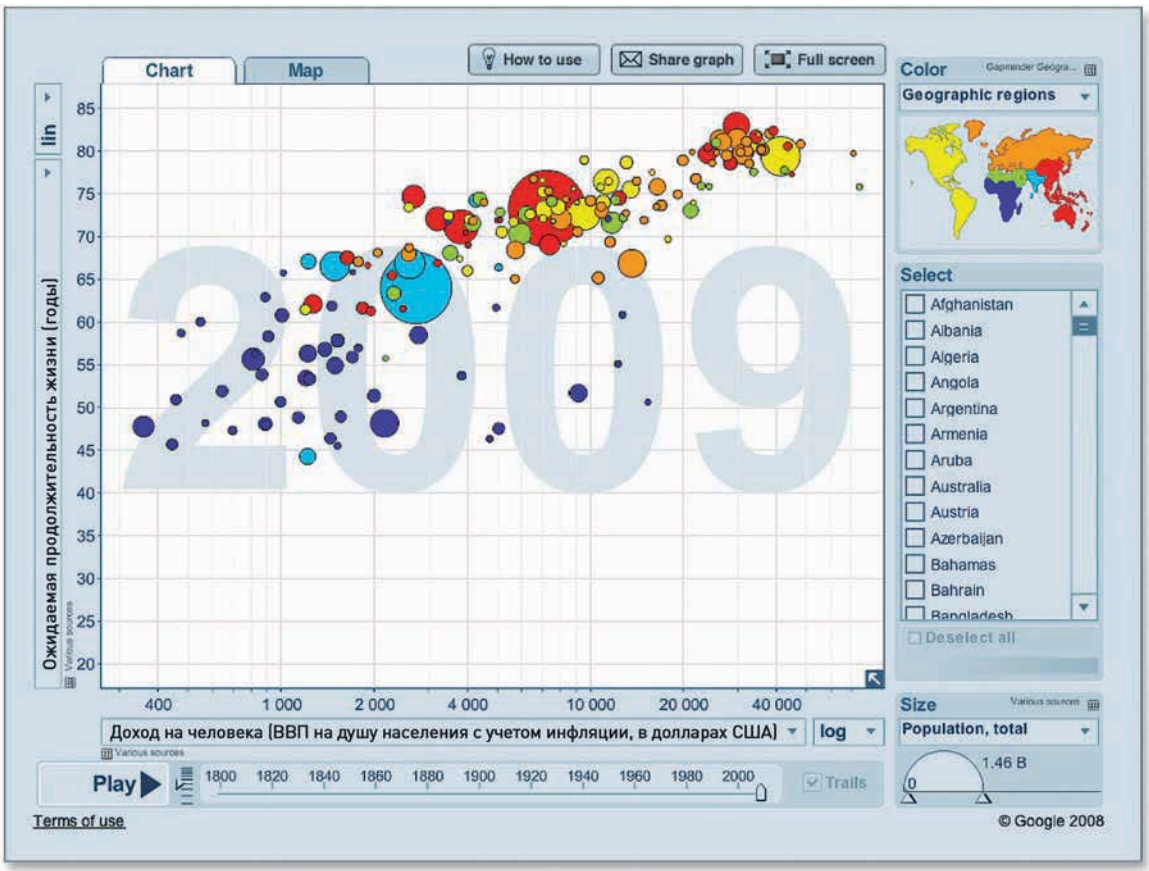


Рис. 1.4. Трендалайзер фонда Gapminder

С точки зрения визуализации тут все довольно элементарно. Это просто динамическая диаграмма. Пузырьки обозначают различные государства и двигаются в соответствии с динамикой показателей бедности в конкретной стране на протяжении определенного года. Так в чем секрет популярности выступления Рослинга? В том, что он говорит с убеждением и волнением. Он рассказывает историю. Сколько раз вы видели презентации со схемами и диаграммами, от которых все зевают?! Но у Рослинга все иначе. Он извлек смысл, заложенный в данных, и в полной мере раскрыл его. А если к этому добавить еще и глотание шпаги в конце лекции, все становится на свои места — в зале не остается равнодушных. Посмотрев лекцию Рослинга, мне захотелось взять в руки эти данные и самому их рассмотреть. Мне не терпелось вникнуть в эту историю поглубже.

Позже я посмотрел другую презентацию фонда Gapminder на ту же самую тему, с той же формой визуализации данных, но с другим докладчиком. Получилось отнюдь не так волнующе. Если говорить честно, я чуть не задремал. Эмоций было ноль. Данные не показались мне

ни убедительными, ни увлекательными. Так что дело не только в данных — не они одни делают рассказ интересным. Важно то, как вы о них рассказываете и как их подаете. От этого зависит, запомнят ли их люди.

▶ Посмотрите, как Ханс Рослинг вызывает восторг аудитории с помощью данных и потрясающих слайдов, на странице `http://datafl.ws/hans`.

Когда все материалы собраны, вам необходимо запомнить: подходи́те к визуализации так, как будто вы собираетесь рассказать историю. Какого рода историю вы хотите поведать? Это будет репортаж? Или рассказ о жизни? Вы хотите убедить людей в необходимости каких-то действий?

Продумайте развитие персонажей. За каждой строчкой в массиве данных стоит история — точно так же, как у каждого героя в книге имеется свое прошлое, настоящее и будущее. Все эти числа находятся между собой в каких-то связях и отношениях. От вас требуется обнаружить их. Конечно, эксперты в сторителлинге, прежде чем засесть за написание текста, должны научиться составлять предложения.

Что искать

Хорошо, истории. Договорились. А какого типа истории можно рассказать с помощью данных? Ну, все зависит от того, какими именно данными вы располагаете, но в общем случае, чему бы ни была посвящена ваша графика, вам следует всегда искать две вещи: паттерны и зависимости.

Паттерны

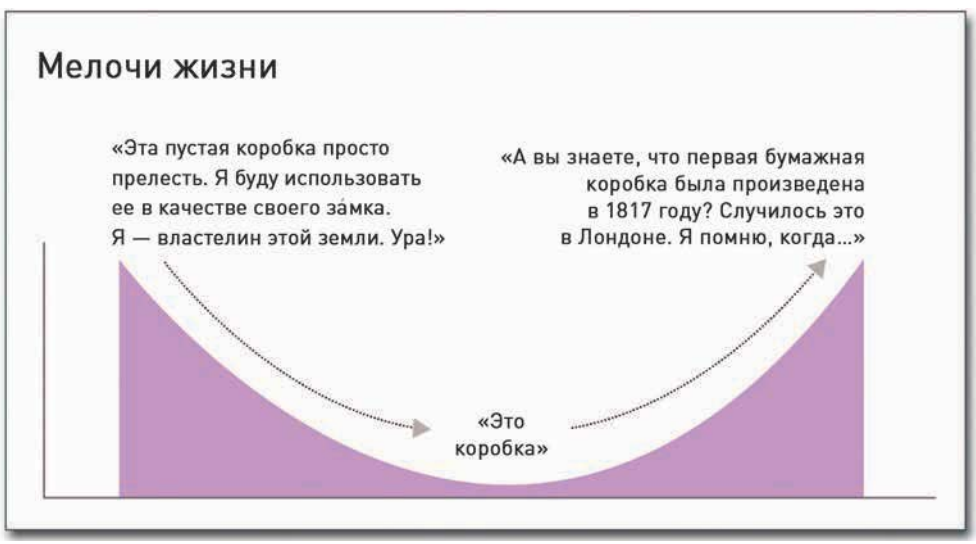


Рис. 1.5. Веселый взгляд на старение

Со временем все меняется. Вы становитесь старше, ваши волосы седеют, а зрение слабеет (рис. 1.5). Цены меняются. Логотипы меняются. Компании появляются. Компании исчезают. Иногда эти перемены случаются внезапно и без предупреждения. А иногда они происходят так медленно, что вы их даже не замечаете.

На какие бы данные вы ни смотрели, тенденция сама по себе может быть интересной, как может быть интересен любой процесс изменений. И здесь вы можете изучать паттерны во времени. Допустим, к примеру, что вы изучаете, как с ходом времени менялся курс акций. Он, конечно, то поднимается, то снижается, но на

сколько пунктов он меняется за сутки? А за неделю? А за месяц? Существуют ли периоды, в которые курс поднимался сильнее, чем обычно? И если да, то почему это происходило? Имели ли место некие события, которые могли спровоцировать подобные изменения?

Как вы убедитесь сами, стоит начать с одного-единственного вопроса, взятого в качестве отправной точки, и тут же за ним потянутся следующие. Причем так обстоят дела не только с временны́ми рядами, но и со всеми типами данных. Попробуйте подойти к вашему массиву данных в большей степени как исследователь, и тогда вы, скорее всего, получите в итоге более интересные ответы.

Данные типа временны́х рядов можно разделить по-разному. Иногда есть смысл представить значения по дням или по часам. В других случаях может оказаться, что полезнее взглянуть на данные по месяцам или по годам. В первом случае в графике, возможно, окажется больше шума, а во втором случае вы получите более обобщенный взгляд.

Те из вас, у кого есть сайт и кто пользуется той или иной компьютерной программой для анализа его посещаемости, могут в этом быстро убедиться. Если вы посмотрите на трафик на вашем сайте на базе посещаемости по дням, как это показано на рис. 1.6, то вы увидите, что он весьма бугристый. Колебания в нем довольно велики.

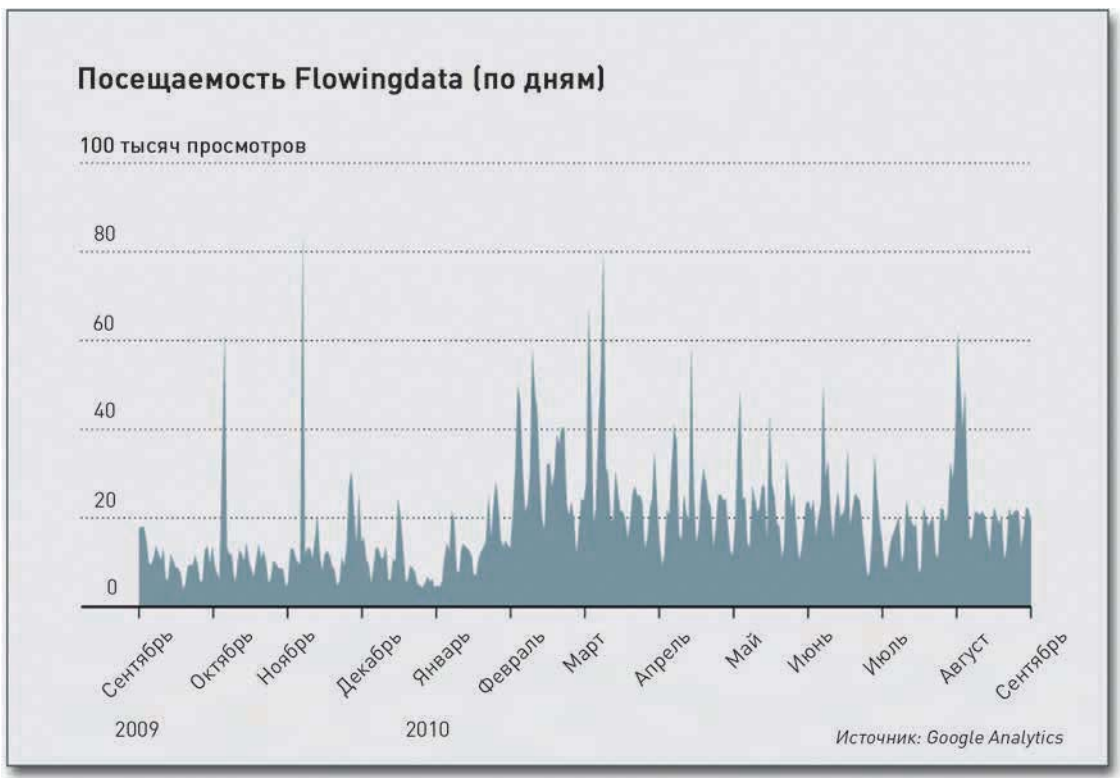


Рис. 1.6. Количество уникальных посетителей сайта FlowingData в день

А если посмотрите на график, выстроенный на базе посещаемости по месяцам (рис. 1.7), то увидите, что, хотя он охватывает тот же промежуток времени, точек на нем меньше и его линия более гладкая.

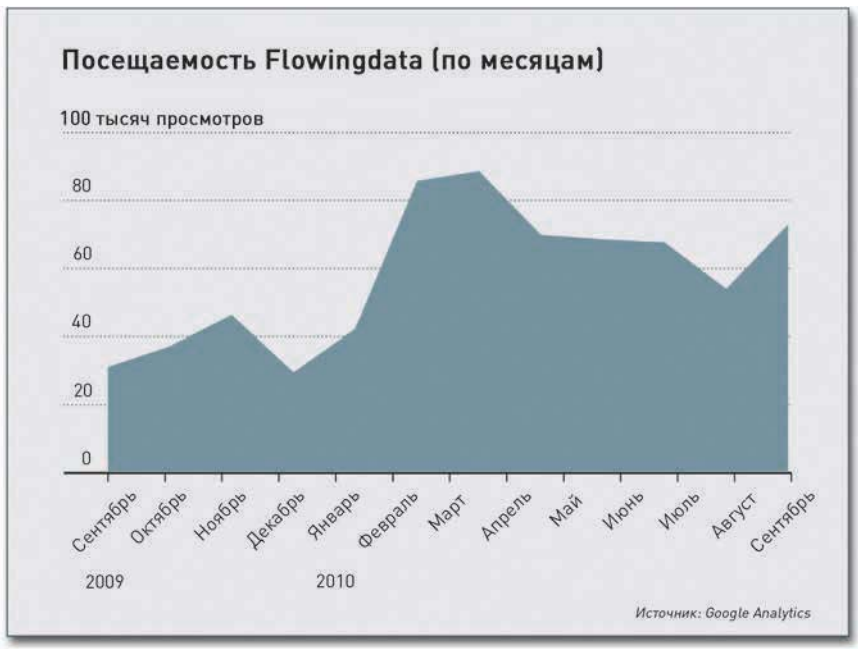


Рис. 1.7. Количество уникальных посетителей сайта FlowingData в в месяц

Я не хочу сказать, что один график лучше другого. На самом деле они могут дополнять друг друга. То, как вы проводите разбивку данных, зависит от того, какое количество деталей вам нужно (или не нужно).

Конечно, временные зависимости — это не единственное, что можно искать. Определенные паттерны можно найти также и в в совокупностях, которые способны помочь вам сравнить некие группы людей или других объектов. Что вы чаще всего едите в течение недели? О чем обычно говорит президент в своем Послании Конгрессу? Какие штаты обычно голосуют за республиканцев? В данном случае полезно поискать паттерны по географическим регионам. И, как вы убедитесь, читая следующие главы, типы вопросов и данных могут быть разными, но ваш подход останется почти неизменным.

Зависимости

Вам наверняка когда-нибудь попадались на глаза рисунки с с целым набором графиков и диаграмм, расположенных как будто в совершенно случайном порядке? Я говорю о рисунках,

которым очевидно чего-то не хватает — как будто графическому дизайнеру некогда было поразмыслить над данными, и он второпях придал им некую форму, лишь бы успеть к крайнему сроку. Очень часто то, чего так явно не хватает в подобных случаях, — и есть зависимость.

В статистике под зависимостью, как правило, подразумевают корреляцию и причинность. Множественные переменные могут быть переплетены между собой сложными связями. Шестая глава, «Визуализация зависимостей», посвящена именно этим понятиям и тому, как их визуализировать.

Однако и на более абстрактном плане, на котором вы не думаете об уравнениях и критериях проверки гипотез, вы можете разрабатывать диаграммы и графики для визуального сравнения и сопоставления величин и распределений. В качестве простого примера можно привести схему проникновения технологий в жизнь, взятую из «Доклада о мировом прогрессе» и представленную на рис. 1.8.

Рис. 1.8. Проникновение технологий в жизнь во всем мире

▶ «Доклад о мировом прогрессе» — это графический отчет, в котором сравниваются показатели прогресса во всем мире и для составления которого используются данные портала UNdata. Его полную версию вы можете найти на странице `http://datafl.ws/12i`.

Столбцовые диаграммы показывают количество пользователей интернета, а также количество абонентов с коммутируемым доступом и широкополосных сетей на 100 жителей. Отметьте, что разброс у интернет-пользователей (от 0 до 95 на 100 жителей) гораздо шире, чем у двух других групп.

Проще и быстрее всего — предоставить компьютерной программе, которая у вас установлена, самой решить, какой диапазон использовать для каждой столбцовой диаграммы. Однако все

столбцовые диаграммы здесь выстроены на базе одного и того же диапазона, несмотря на то что нет ни одной страны, у которой имелось бы по 95 абонентов с коммутируемым доступом или пользователей широкополосных сетей на 100 жителей. Но это дает вам возможность легко сравнивать распределения между группами.

А потому, когда вам приходится работать с множеством различных наборов данных, постарайтесь думать о них как о группе, а не как об отдельных, никак не связанных между собой массивах. Так вы сможете получить гораздо более интересные результаты.

Сомнительные данные

Пытаясь разглядеть истории, скрывающиеся за вашими данными, стоит всегда подвергать сомнению то, что вы видите. Помните: то, что это — числа, еще не означает, что они верны.

Я должен признаться, что для меня проверка данных — самый нелюбимый этап создания графики. Конечно, когда некий человек, группа или организация предоставляет вам груду данных, то это должна быть их обязанность — проверить, все ли они достоверны. Но хорошие дизайнеры графики не принимают все на веру. Ведь и хорошие строители не берут для возведения фундамента дома абы какой цемент. Так что и вам для создания графики не следует брать абы какие данные.

Проверка и верификация данных — одна из важнейших, если не самая важная, часть графического дизайна.

По сути, вам необходимо просмотреть данные на предмет нахождения в них чего-то такого, что лишено смысла. Может, кто-то допустил ошибку при вводе и добавил лишний ноль или, наоборот, пропустил его. Может, во время сбора данных возникли некие проблемы со связью и часть информации дошла в искаженном виде. Во всех случаях, если что-то покажется вам подозрительным, вам необходимо будет свериться с первоисточником.

У человека, занимающегося сбором данных, обычно возникает некое чувство того, что именно можно ожидать. Если сбором данных занимались вы сами, тогда просто спросите себя: это нормально — что один штат дает 90 процентов чего бы там ни было, а все остальные, вместе взятые, умещаются в диапазоне 10–20 процентов от общего объема? Что там такое происходит?

Очень часто выясняется, что аномалия в ваших данных — всего лишь опечатка. Хотя бывает и такое, что она оказывается самым интересным фактом во всем массиве данных и становится двигателем вашей истории. Просто проверьте, с чем именно вы столкнулись в каждом конкретном случае.

Дизайн

Когда уже все данные в порядке, вы можете приступать к визуализации. Над чем бы вы ни работали — будь то доклад, онлайн-инфографика или произведение data-арта, — вам следует

соблюдать несколько основных принципов. Каждый из них дает пространство для маневра, так что вам лучше воспринимать их, скорее, как некие ориентиры, а не как свод жестких правил, но если в графическом дизайне вы новичок, тогда вам именно с них и стоит начать.

Объясните кодировки

Разработка любой диаграммы протекает примерно по одному и тому же сценарию. Вы собираете данные; вы кодируете данные с помощью кругов, столбцов и цветов; а затем предлагаете другим людям прочитать их. В этот момент читатели должны расшифровать ваши кодировки. Что означают все эти круги, столбцы и цвета?

В работе Уильяма Кливленда (William Cleveland) и Роберта Макгила (Robert McGill) вопросы кодировки рассмотрены в деталях. Некоторые типы кодировок работают лучше, чем другие. Но какой именно подход к кодировке вы выберете, будет не так уж и важно, если в конечном счете читатели не смогут понять, что именно выражают те или иные обозначения. Если они не смогут раскодировать ваше послание, значит, время, которое вы посвятили созданию диаграммы, окажется потраченным впустую.

Возможно, вы отмечали про себя, что в рисунках, находящихся где-то между data-артом и инфографикой, порой отсутствует контекст, а уж с произведениями data-арта это и вовсе обычная ситуация. Легенды и метки способны полностью разрушить впечатление от работы, но вы можете по крайней мере дать немного информации, описав самое необходимое в одном коротком абзаце. Это поможет другим оценить ваш труд.

ПРИМЕЧАНИЕ

За дополнительной информацией о том, как люди кодируют и декодируют цвета и формы, обратитесь к публикации Кливленда и Макгила Graphical Methods for Analyzing Data («Восприятие графики и графические методы анализа данных»).

Однако с подобным отсутствием пояснений иногда можно столкнуться и в тех случаях, когда основная задача диаграмм и графиков как раз и состоит в представлении данных. В итоге читатели бывают разочарованы, а вам наверняка не хочется допускать этого. Происходит такое порой потому, что, работая с данными, вы с ними срастаетесь и знаете наизусть, что означает каждый элемент диаграммы. Читателям же все внове. Не зная контекста, с которым вы сами свыклись, анализируя данные, они оказываются словно слепыми — не могут увидеть то, что видите вы.

Итак, как вы можете убедиться, что читателям удастся расшифровать ваши кодировки? Просто объясните им значение каждого элемента с помощью подписей, легенд и ключей. Что именно выбрать — зависит от ситуации. Посмотрите, например, на карту мира на рис. 1.9, которая дает представление об использовании программы Firefox в различных странах земного шара.

Вы видите, что разные страны окрашены в разные оттенки голубого, но что означают эти оттенки? Темный цвет — это больше или меньше потребителей? И если больше, то больше какого именно количества? В том виде, в котором карта представлена на рисунке, она довольно-таки бесполезна. Но если вы снабдите ее легендой, такой, как на рис. 1.10, все сразу прояснится. В данном случае цветовая легенда выполняет и еще одну функцию: она представляет собой столбцовую диаграмму, демонстрирующую распространенность программы на основании количества пользователей.

Рис. 1.9. Использование программы Firefox в мире, по странам

Если у вас есть достаточно места или слишком много категорий, вы можете вставить подписи непосредственно рядом с объектами вашей диаграммы, как это продемонстрировано на рис. 1.11. На данной диаграмме показано количество номинаций, которые тот или иной актер получил перед тем, как был удостоен премии «Оскар» за лучшую мужскую роль.

Рис. 1.10. Легенда к карте использования Firefox

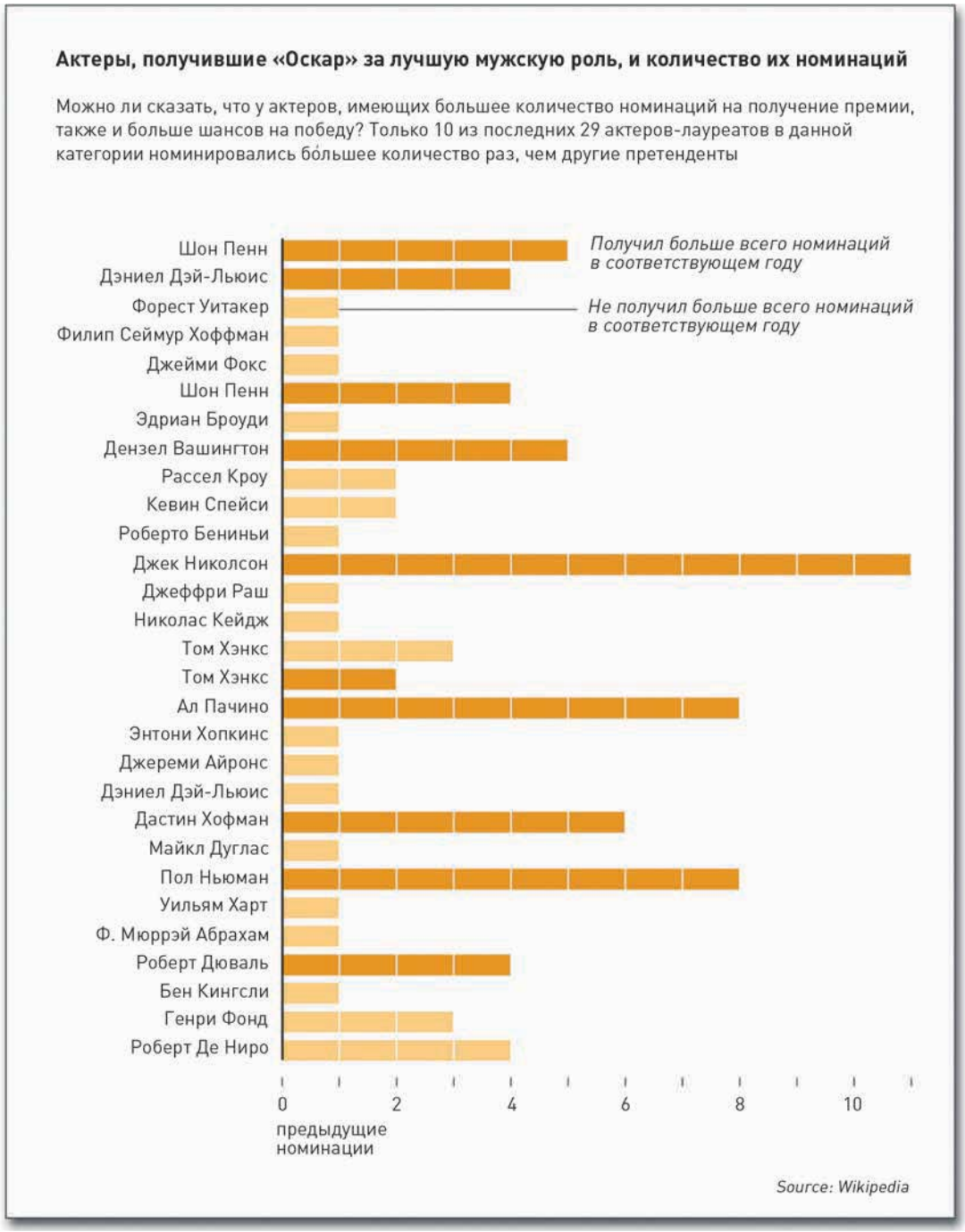


Рис. 1.11. Непосредственное обозначение объектов

По интернету в свое время разгуливала теория о том, что статуэтку, как правило, выигрывает тот актер, который в соответствующий год получил больше всего номинаций среди представителей своей когорты. Как следует из подписей, оскароносцы, отмеченные темно-оранжевым, действительно получили больше всего номинаций, а светло-оранжевым отмечены те, кто не получил больше всего номинаций, однако тем не менее был удостоен статуэтки в соответствующем году.

Как видите, у вас есть множество вариантов действия. Все эти детали не требуют больших усилий в использовании, они вроде мелкие, но разница от их применения может быть огромной для восприятия вашей работы.

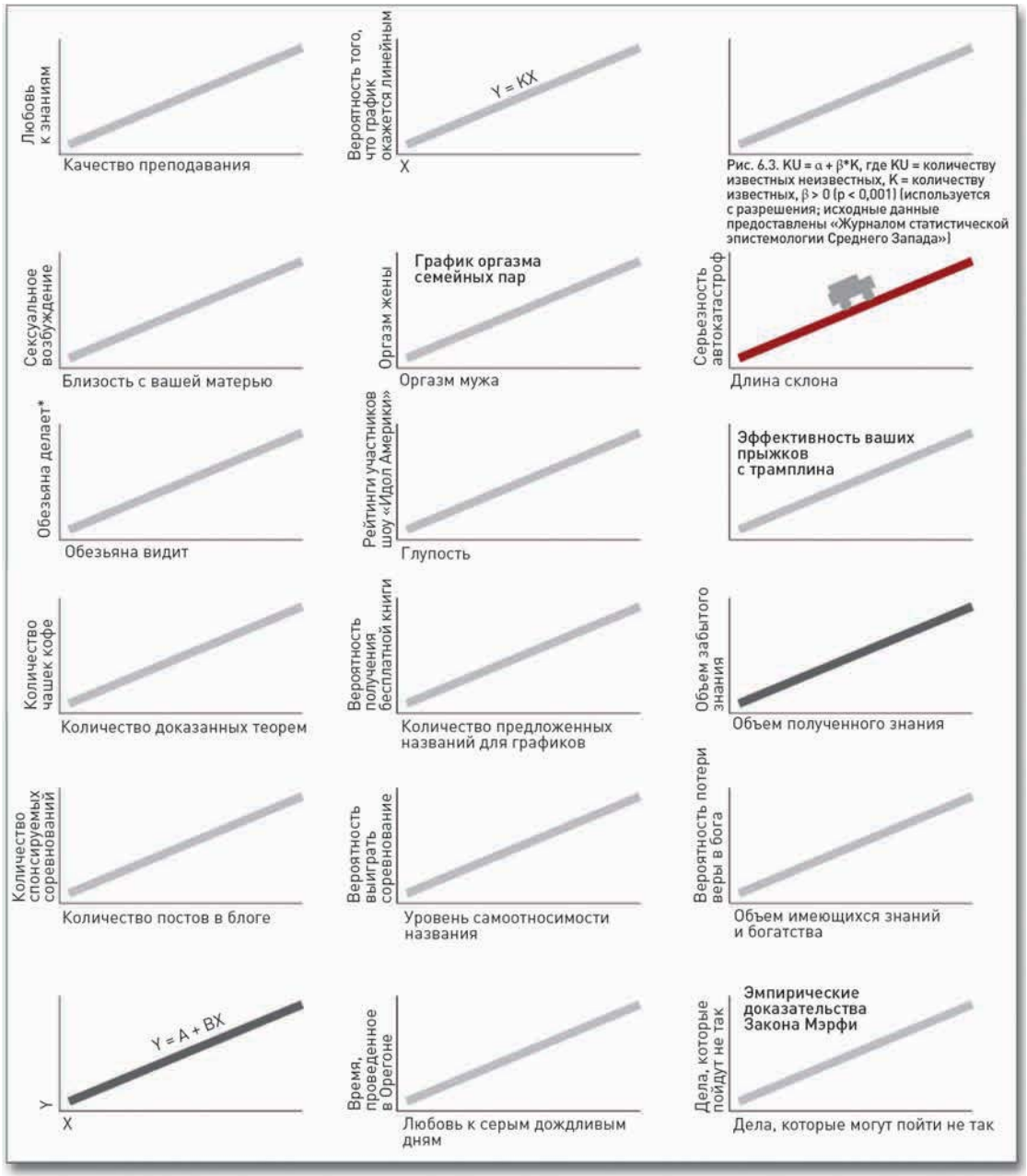
Дайте осям названия

Вы уже поняли, почему необходимо всегда давать пояснения к кодировке. Точно так же вам следует всегда давать названия осям в ваших графиках и диаграммах. Без подписей и объяснений оси будут исполнять лишь декоративную роль. Дайте им названия, чтобы читатели знали, какая именно шкала применяется в каждом конкретном случае. Логарифмическая, дифференциальная, экспоненциальная или вы измеряете все в расчете на 100 сливных бачков унитазов? Лично я, когда не вижу у осей названий, всегда по умолчанию принимаю, что речь идет именно о последнем.

Чтобы продемонстрировать свою мысль, я вернусь к конкурсу, который организовал пару лет назад на сайте FlowingData. Я разместил на страничке изображение, представленное на рис. 1.12, и попросил читателей дать самые интересные названия осям.

Рис. 1.12. Вставьте название

Для одного и того же графика я получил около 60 различных вариантов названий. Часть из них представлена на рис. 1.13.



* «Обезьяна видит — обезьяна делает» — ямайская поговорка, описывающая процесс копирования чужого поведения без понимания его смысла. *Прим. пер.*

Рис. 1.13. Некоторые из результатов, полученных во время соревнования по придумыванию названий на FlowingData

Как вы можете убедиться, притом что график везде один, в результате простой смены названий осей получаются совершенно разные истории. Конечно, все это делалось забавы ради. А теперь представьте себе, что данный график должен был рассказать о чем-то очень серьезном. Без подписей и названий ваша диаграмма просто бессмысленна.

Держите геометрию под контролем

Когда вы создаете диаграмму, вы используете геометрические фигуры. В столбцовой диаграмме вы используете прямоугольники. При этом величины вы выражаете высотой прямоугольников. В точечной диаграмме индикатором величины служит местоположение. То же самое и с графическим представлением временны́х рядов. В круговых диаграммах для выражения величины используется угол, а сумма величин всегда равняется 100 процентам (рис. 1.14). Это несложно, но будьте начеку, потому что так же несложно все перепутать. Если вы не проявите достаточного внимания, вы легко можете допустить ошибку, и когда вы облажаетесь, люди, особенно в Сети, не побоятся вам об этом сказать прямым текстом.

Рис. 1.14. Правильное и неправильное составление круговых диаграмм

Другая распространенная ошибка обычно происходит тогда, когда дизайнеры используют для обозначения величин двухмерные формы, но измеряют их так, как если бы они использовали только одно измерение. Прямоугольники в столбцовой диаграмме двухмерные, но в качестве индикатора используется только их высота. Ширина не означает ничего. Однако, создавая пузырьковую диаграмму, для обозначения величин следует использовать площадь кругов. Начинающие дизайнеры вместо этого часто применяют радиус или диаметр, и соотношение получается в корне неверным.

На рис. 1.15 показаны два круга, размер которых задавался по площади. Это правильный подход.

Рис. 1.15. Правильный подход при определении размеров кругов в диаграммах

На рис. 1.16 представлена пара кругов, чьи размеры определялись по диаметру. Первый круг диаметром в два раза меньше второго, но по площади он меньше в четыре раза.

Рис. 1.16. Неправильный подход при определении размеров кругов в диаграммах

То же самое происходит и с прямоугольниками, например в тримапах. В качестве индикатора величины там используется площадь, а не высота или ширина.

Укажите источники

Казалось бы, не нужно говорить о необходимости указывать источники данных — но уж слишком много людей забывают это сделать. Откуда взялись все эти данные? Если вы посмотрите на диаграммы в газетах, вы увидите, что источник в них всегда указан — обычно мелким

шрифтом где-нибудь внизу. Вы должны поступать так же. Иначе читатели не будут знать, насколько достоверными являются ваши графики и диаграммы.

Как еще они могут убедиться в том, что данные не были просто выдуманы? Конечно, вы бы такое никогда не сделали, но не все об этом знают. Если вы укажете в своей работе источник, читатели не только будут относиться к ней с бóльшим уважением, но они смогут также самостоятельно проверить и проанализировать данные.

Включение источника данных способствует также обогащению контекста. Вполне очевидно, что опрос, проведенный на региональной ярмарке, будет интерпретирован иначе, чем тот, который осуществлялся во время переписи населения страны.

Учитывайте аудиторию

И наконец, всегда помните о вашей аудитории и о цели вашей работы. Например, диаграмма, создаваемая для слайда презентации, должна быть простой. Вы, конечно, можете включить в нее массу деталей, но их увидят только те, кто сидит в первом ряду. А вот если вы делаете постер, который люди будут пристально разглядывать и изучать, тогда вы можете включить в диаграмму гораздо больше подробностей.

Вы готовите деловой отчет? Тогда не пытайтесь создать самое красивое произведение data-арта, которое когда-либо видел мир. Вместо этого постарайтесь разработать четкую и ясную диаграмму, в которой все строго по делу. Графика нужна вам как подспорье в процессе анализа? В таком случае результат визуализации данных увидите только вы и, следовательно, не стоит особо тратить время на оформление и аннотации. А может, ваша диаграмма или график предназначается для печати в средствах массовой информации? Тогда вы должны стремиться к простоте и пояснять все мысли, восприятие которых может оказаться затруднительным для аудитории.

Закругляясь

Короче говоря, начните с вопроса, осмотрите данные критическим взглядом и определитесь с целью визуального средства, а также узнайте, для кого оно предназначено. Это поможет вам разработать понятную диаграмму или график, которые будут заслуживать того, чтобы люди потратили время на их рассмотрение. И неважно, какого именно типа графикой вы воспользуетесь.

Как это делать, вы научитесь в следующих главах. Вы поймете, как подавать и визуализировать данные. Вы узнаете, как разрабатывать диаграммы с самого начального этапа и до конца. А затем вы примените полученные знания к вашему собственному массиву данных. Вы определитесь с тем, какую историю вы хотите рассказать, и создадите соответствующий графический объект.

Как обращаться с данными 2

Прежде чем перейти к собственно визуальной части работы с данными, вам необходимо их получить.Именно данные делают визуализацию интересной. Если у вас нет интересных сведений, вы в конечном итоге получите лишь незапоминающуюся диаграмму или красивую, но бесполезную картинку. Где можно найти хорошие данные? Как до них добраться? Вот какие вопросы стоят перед вами на этом этапе.

Когда же данные будут у вас на руках, вам понадобится отформатировать их, чтобы загрузить в компьютерную программу. Возможно, вы получили данные в виде текстового файла с разделителями-запятыми или в виде таблицы Excel, и вам нужно конвертировать их во что-то типа XML. Или наоборот. Или, возможно, необходимые вам данные доступны лишь в разрозненном виде через интернет-приложение, а вы хотите получить таблицу целиком.

Вам нужно научиться добывать и перерабатывать данные, и тогда ваши способности в области визуализации существенно возрастут.

Сбор данных

Данные представляют собой основу любой визуализации. К счастью, есть множество мест, откуда их добывают. Вы можете получить их от экспертов в интересующей вас области, почерпнуть из целого ряда онлайн-приложений или собрать собственноручно.

Данные, предоставленные другими людьми

Это проторенная дорога, особенно если вы дизайнер-фрилансер или работаете в отделе графики крупной организации. В большинстве случаев это очень хорошо — когда кто-то другой выполняет за вас работу по сбору данных. Но вам все равно необходимо быть внимательным. Прежде чем красиво отформатированная таблица попадет к вам в руки, в нее может закрасться масса ошибок.

Если таблицу, с которой вам предстоит работать, вы получили от других людей, знайте, что самые распространенные ошибки, на наличие которых вам нужно будет проверить данные, — это опечатки. Нет ли, например, каких-нибудь потерянных нулей? Может, ваш клиент или поставщик данных хотел вставить их шесть, а не пять. На определенном этапе данные обычно считываются из одного источника и вносятся в другой, скажем, в Excel или в какую-нибудь иную программу табличных вычислений (за исключением тех случаев, когда целиком импортируется текстовый файл с разделителями), и в такие моменты какая-нибудь невинная ошибка легко может преодолеть «барьеры» этапа контроля и попасть в массив данных.

А еще вам необходимо изучить контекст. Не нужно становиться экспертом в той области, из которой были почерпнуты данные, однако вам следует знать, из какого именно первоисточника они взяты и о чем говорят. Это поможет вам создать более качественную графику и поведать через нее более цельную историю. Допустим, к примеру, что перед вами лежат результаты некоего опроса. Когда именно он состоялся? Кто его проводил? Кто отвечал на вопросы? Очевидно, что результаты опроса, проведенного в 1970 году, будут нести несколько иной смысл, нежели результаты сегодняшнего дня.

Самостоятельный поиск данных

Если вам не были предоставлены данные, то пойти и найти их — ваша обязанность, что, конечно, прибавит вам работы, и это плохая новость. Однако есть и хорошая новость: в наши дни становится все проще и проще находить релевантные машиночитаемые данные, которые легко можно загрузить в программу. И вот с чего вам следует начинать поиски.

ПОИСКОВЫЕ СИСТЕМЫ

Как люди сегодня находят что-либо в Сети? Они «гуглят» — ищут с помощью Google. Для этого большого ума не надо, но вы будете удивлены, если я скажу вам, как часто люди шлют мне

письма, спрашивая, не знаю ли я, где можно найти данные по той или иной тематике, — и это в то время, когда даже поверхностный поиск дает вполне удовлетворительный результат. Лично я в подобных ситуациях обычно обращаюсь к Google, а время от времени и к Wolfram|Alpha, поисковой машине с вычислительными возможностями.

▶ Вы можете познакомиться с Wolfram|Alpha на `http://wolframalpha.com`. Эта поисковая машина может оказаться особенно полезной при поиске базовых статистических данных по разным темам.

НАПРЯМУЮ ИЗ ПЕРВОИСТОЧНИКА

Если запрос данных в поисковой машине не дал ничего полезного, попробуйте найти ученых, которые специализируются в той области, сведения из которой вы пытаетесь найти. Иногда они размещают внушительные массивы данных на своих персональных сайтах. Если и это не поможет, просканируйте их публикации и труды — вполне возможно, что найдете там хорошие зацепки. А еще вы можете попробовать связаться с ними по электронной почте, но сначала убедитесь, что они действительно занимались изучением интересующего вас вопроса. Иначе вы просто впустую потратите и свое, и их время.

Различные источники данных вы можете также подсмотреть в диаграммах и графиках, публикуемых в средствах массовой информации, таких как New York Times. Обычно источник данных указан мелким шрифтом где-то на самом графическом объекте. Если его там нет, тогда упоминание о нем должно содержаться в связанной с изображением статье. Такое направление поиска бывает особенно полезным тогда, когда вам на глаза попадается газета или онлайн-издание, использующая данные, в изучении которых вы в настоящий момент заинтересованы. Поищите сайты с подходящей информацией, откуда можно почерпнуть необходимые данные.

Это будет работать не всегда. Когда в письме указано, что вы — репортер такой-то газеты, находить общий язык с собеседниками оказывается немного проще, но в любом случае этот способ стоит попробовать.

УНИВЕРСИТЕТЫ

Когда я стал магистрантом, я часто пользовался академическими ресурсами, которыми располагал, а именно библиотекой. Многие библиотеки существенно продвинулись в своем технологическом оснащении и к настоящему времени располагают огромными архивами данных. Различные факультеты и кафедры также поддерживают массу файлов данных, значительная часть которых находится в открытом доступе. Правда, многие из этих доступных массивов данных предназначены в первую очередь для использования при написании курсовых работ и выполнении домашних заданий. Я предлагаю вам зайти и посмотреть следующие ресурсы.

- Data and Story Library (DASL) (`http://lib.stat.cmu.edu/DASL`) — онлайн-библиотека файлов с данными и историями, иллюстрирующими применение основных статистических методов; относится к Университету Карнеги-Меллон.
- Berkeley Data Lab (`http://sunsite3.berkeley.edu/wikis/datalab`) — часть библиотечной системы Калифорнийского университета в Беркли.

- Массивы статистических данных Калифорнийского университета в Лос-Анджелесе (`http://www.stat.ucla.edu/data`) — часть данных, которые кафедра статистики университета использует при выполнении лабораторных работ и заданий.

ИСТОЧНИКИ ДАННЫХ ОБЩЕГО ХАРАКТЕРА

Сегодня появляется все больше приложений, из которых можно почерпнуть различные данные. Некоторые приложения предлагают большие файлы данных, которые можно скачать безвозмездно или за определенную плату. Часть из них изначально создавалась с мыслью о разработчиках, и доступ к их данным осуществляется через интерфейсы программирования приложений (API). Это позволяет вам использовать данные различных сервисов, таких как, например, Twitter, и интегрировать их в свои собственные приложения. Далее представлены некоторые рекомендуемые ресурсы.

- Freebase (`http://www.freebase.com`) — база данных, появившаяся в результате усилий большого сообщества и предоставляющая сведения главным образом о людях, местах и товарах. Немного напоминает «Википедию», но информация в ней более структурирована. Можете пользоваться ею для скачивания информации или в качестве базы данных типа back-end* для вашего приложения.
- Infochimps (`http://infochimps.org`) — рынок с платными и бесплатными массивами данных. Доступ к некоторым из них можно получить через их API.
- Numbrary (`http://numbrary.com`) — онлайн-каталог данных, по большей части связанных с национальной статистикой.
- AggData (`http://aggdata.com`) — еще одно хранилище платных массивов данных, сфокусированных в основном на составлении исчерпывающих перечней торговых объектов.
- Amazon Public Data Sets (`http://aws.amazon.com/publicdatasets`) — особого роста здесь не наблюдается, но тем не менее ресурс содержит весьма солидные подборки научных данных.
- Wikipedia (`http://wikipedia.org`) — в этой энциклопедии, поддерживаемой на общественных началах, можно найти множество небольших подборок данных в виде HTML-таблиц.

ТЕМАТИЧЕСКИЕ ДАННЫЕ

Помимо поставщиков данных общего характера существует также множество сайтов по конкретным тематикам, предлагающих уйму бесплатных данных. Далее следует лишь маленькая выборка того, что доступно по тем или иным направлениям.

---

* Back-end — база данных, к которой пользователи обращаются не напрямую или путем низкоуровневых манипуляций (например SQL-запросов), а через специально разработанное приложение. *Прим. пер.*

География

- У вас есть программа для локализации объектов, но нет географических данных? Вам повезло. В вашем распоряжении огромное количество географических файлов, в том числе в векторном формате (шейп-файлов).
- TIGER (`http://www.census.gov/geo/www/tiger`) — сайт Бюро переписи населения США предоставляет, возможно, самые дорогие, но и самые детальные базы данных об автострадах, железных дорогах, реках и почтовых индексах, которые только можно найти.
- OpenStreetMap (`http://www.openstreetmap.org`) — один из лучших примеров баз данных, созданных на общественных началах.
- Geocommons (`http://www.geocommons.com`) — это одновременно и база данных, и картографическое приложение.
- Flickr Shapefiles (`http://www.flickr.com/services/api`) — географические границы в том виде, в каком они представлены пользователями Flickr.

Спорт

Люди любят спортивную статистику, и вы можете получить доступ к данным этого типа за несколько последних десятилетий. Найти их можно на сайте журнала Sports Illustrated и на сайтах соответствующих спортивных команд и организаций, а также на сайтах, посвященных сугубо спортивной статистике.

- Basketball Reference (`http://www.basketball-reference.com`) — источник исчерпывающих данных и сведений вплоть до прямых репортажей с игр Национальной баскетбольной ассоциации.
- Baseball DataBank (`http://baseball-databank.org`) — сверхосновательный сайт, с которого можно скачать исчерпывающие наборы данных о бейсболе.
- DatabaseFootball (`http://www.databasefootball.com`) — здесь можно посмотреть данные об играх Национальной футбольной лиги США по командам, по игрокам и по сезонам.

Мир в целом

Несколько заслуживающих внимания международных организаций также поддерживают базы данных о мире в целом, главным образом по таким показателям, как здравоохранение и уровень развития. И все же, работая с ними, вам придется кое-что отсеивать, так как многие массивы данных не очень хорошо структурированы. Нелегко получить стандартизированные данные, собирая их из разных стран разными методами.

- Global Health Facts (`http://www.globalhealthfacts.org`) — данные из области здравоохранения в разных странах мира.
- UNdata (`http://data.un.org`) — агрегатор данных из разных источников со всего мира.

- World Health Organization (`http://www.who.int/research/en`) — снова обилие данных из области здравоохранения, таких как уровень смертности и ожидаемая продолжительность жизни.
- OECD Statistics (`http://stats.oecd.org`) — один из крупнейших ресурсов, посвященных экономическим показателям.
- World Bank (`http://data.worldbank.org`) — данные по сотням показателей, причем представленные в очень удобном для разработчиков виде.

Государственные и политические организации

В последние годы все больше говорится о доступности информации и прозрачности, а потому многие ведомства регулярно предоставляют сведения о своей работе, а организации, такие как Sunlight Foundation, стимулируют разработчиков и дизайнеров использовать эти данные. На протяжении некоторого времени государственные организации сами публиковали информацию о себе, но с момента запуска Data.gov большинство этих данных уже доступны в одном месте. А еще вы можете найти массу неправительственных сайтов, чья цель — заставить политиков отчитываться перед гражданами.

- Бюро переписи населения (`http://www.census.gov`) — богатый источник демографических данных.
- Data.gov (`http://data.gov`) — каталог данных, поставляемых органами правительства. Хотя и относительно новый, но пополняется из большого количества источников.
- Data.gov.uk (`http://data.gov.uk`) — британский эквивалент data.gov.
- DataSF (`http://datasf.org`) — данные, связанные конкретно с Сан-Франциско.
- NYC DataMine (`http://nyc.gov/data`) — ресурс, аналогичный предыдущему, только о Нью-Йорке.
- Follow the Money (`http://www.followthemoney.org`) — большой набор инструментов и массивов данных для отслеживания денежных потоков в политической жизни страны.
- OpenSecrets (`http://www.opensecrets.org`) — еще один ресурс, поставляющий сведения о государственных расходах и лоббировании.

Извлечение данных

Очень часто бывает так, что необходимые данные найти можно, но есть одна загвоздка. Они находятся не в одном месте и не в одном файле, а разбросаны по разным HTML-страницам или даже по разным сайтам. Как вам быть?

Самый лобовой, но и самый затратный по времени метод — это посетить по очереди все страницы и вручную ввести интересующие вас крупицы данных в свою таблицу. Если страниц всего несколько, то, конечно, проблем с этим не возникнет.

Ну а что делать, если страниц тысячи? Такая работа займет слишком много времени. Даже если их не тысячи, а сотни — процесс уже становится утомительным. Было бы гораздо лучше, если б вы

могли автоматизировать процесс. Для этого и применяется *скрейпинг (анализ и извлечение) данных*. Вы создаете некий код для автоматического посещения определенного перечня страниц, получения конкретного контента с этих страниц и сохранения его в базе данных или в текстовом файле.

ПРИМЕР: АНАЛИЗ САЙТА

Лучший способ научиться извлекать необходимые данные — это сразу же перейти к примерам. Скажем, вы хотели скачать данные по температуре за прошедший год, но у вас не получается найти источник, который предоставил бы вам все сведения за нужный отрезок времени или по нужному городу. Вы можете пойти на любой сайт о погоде, но самое большее, что вы там найдете, — подробный прогноз на ближайшие десять дней. Это совсем не то, что вам нужно. Вам нужны реальные показатели температуры из прошлого, а не прогнозы на будущее.

К счастью, сайт Weather Underground предоставляет исторические данные о погоде. И плохая новость: на одной странице сведения можно получить только за один день.

Чтобы разговор стал конкретнее, посмотрите температуру в городе Буффало, США. Зайдите на сайт Weather Underground и введите BUF в окно поиска. После этого вы попадете на страницу о погоде в районе расположенного в Буффало аэропорта Ниагара Интернешнл (рис. 2.1).

ПРИМЕЧАНИЕ

Хотя самый гибкий метод извлечения необходимых вам данных — это программирование, вы можете попробовать еще и такие инструменты, как Needlebase и PDF-конвертер Able2Extract. Используйте их по назначению — они способны существенно сэкономить вам время.

▶ Посетите сайт Weather Underground: `http://wunderground.com`.

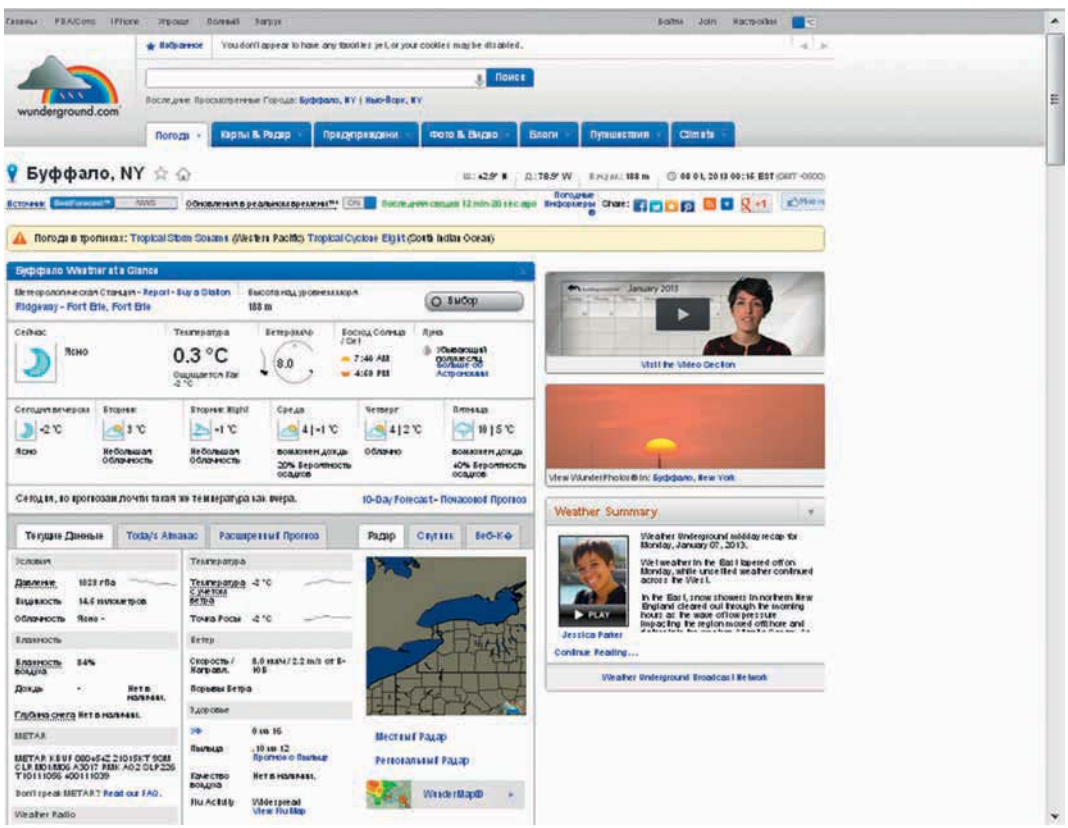


Рис. 2.1. Температура в Буффало по данным Weather Underground

Рис. 2.2. Выпадающее меню для получения исторических сведений по конкретной дате

Вверху страницы дается информация о температуре в настоящий момент, пятидневный прогноз погоды и некоторые другие детали о сегодняшнем дне. Прокрутите страницу вниз, поближе к середине, и найдите панель «История и альманах» («History & Almanac»), как показано на рис. 2.2. Обратите внимание на выпадающее меню, в котором вы можете выбрать определенную дату.

Настройте меню на показ данных за 8 января 2013 года и нажмите кнопку «Просмотр» (View). Откроется другая страница, где вам будут представлены подробные данные о погоде в выбранный вами день (рис. 2.3).

Там будет и температура, и градусо-день, и влажность, и осадки, и множество других данных, но вас интересует только максимальная температура в тот день. Ее вы можете найти во втором столбце, во второй строчке сверху вниз. Восьмого января 2013 года максимальная температура в Буффало составляла 0 °C.

Получить этот один показатель вышло довольно легко. Но как можно добыть сведения о максимальной температуре за каждый день на протяжении всего 2012 года? Самый простой, лобовой метод — это поочередно менять дату в выпадающем меню. Повторите процедуру 365 раз — и готово.

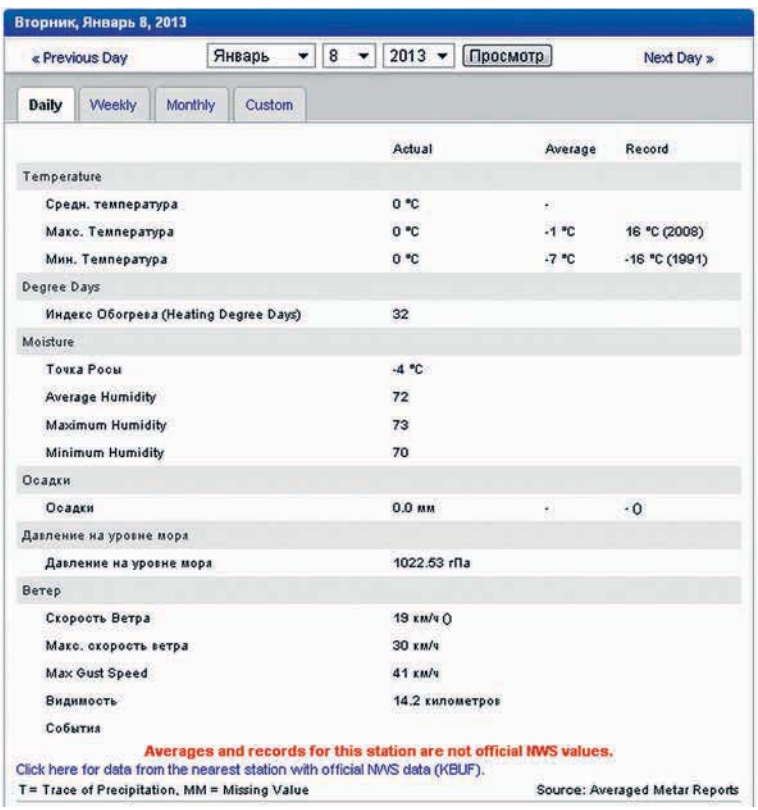


Рис. 2.3. Температурные данные за один конкретный день

Разве это не кажется вам увлекательным? Нет? Тогда вы можете ускорить процесс с помощью небольшого кода и кое-какого ноу-хау. Для этого обратитесь к языку программирования Python и к библиотеке под названием Beautiful Soup, созданной Леонардом Ричардсоном (Leonard Richardson).

В течение следующих нескольких абзацев вы впервые в жизни ощутите вкус создания кодов. Если у вас есть опыт в программировании, следующий этап вы пройдете относительно быстро. Однако если у вас такого опыта нет, не беспокойтесь — я проведу вас через всю процедуру шаг за шагом. Многие люди предпочитают оставаться в рамках интерфейсов, где все делается одним кликом, но доверьтесь мне. Достаточно освоить самую малость программистских умений, и вы сможете открыть для себя целый мир новых возможностей в работе с данными. Готовы? Поехали.

Во-первых, вам необходимо убедиться, что у вас в компьютере установлено подходящее программное обеспечение. Если вы работаете под Mac OS X, тогда Python у вас уже должен быть инсталлирован. Чтобы начать, откройте приложение Terminal и введите: **python** (рис. 2.4).

Рис. 2.4. Запуск Python в Mac OS X

▶ Для скачивания и установки Python зайдите на `http://python.org`. Не беспокойтесь, это не так уж трудно.

Если вы работаете под Windows, вы можете зайти на соответствующий сайт и, следуя инструкциям, скачать и установить Python у себя.

Затем вам необходимо скачать Beautiful Soup, библиотеку, которая поможет вам считывать веб-страницы легко и быстро. Поместите файл Beautiful Soup Python (он имеет расширение .py) в директории, в которой вы собираетесь сохранить и свой код. Если вы знаете путь до Python, можете поместить Beautiful Soup также у себя в библиотеке — на результате это никак не скажется.

▶ Чтобы скачать Beautiful Soup, зайдите на `http://www.crummy.com/software/BeautifulSoup`. Скачайте ту версию, которая подходит к используемой вами версии Python.

После того как вы установите Python и скачаете Beautiful Soup, создайте файл в вашем любимом текстовом редакторе или в редакторе кодов и сохраните его как `get-weather-data.py`. Теперь вы можете заняться написанием кода.

Первое, что вам необходимо сделать, — это загрузить страницу, на которой представлены исторические сведения о погоде. URL у страницы с информаций о погоде в Буффало за 1 октября 2010 выглядит так:

```
www.wunderground.com/history/airport/KBUF/2010/10/1/DailyHistory.html?req_city=NA&req_
state=NA&req_statename=NA
```

Если в этом адресе вы удалите все, что следует за .html, страница будет по-прежнему загружаться, так что избавьтесь от этих лишних символов. На данный момент они вас не интересуют. Должно получиться:

```
www.wunderground.com/history/airport/KBUF/2010/10/1/DailyHistory.html
```

В URL дата указана как /2010/10/1. Используя выпадающее меню, замените ее на новую: на 1 января 2009 года, так как вы собираетесь извлекать данные по температуре за весь 2009 год. После этого URL должен выглядеть так:

```
www.wunderground.com/history/airport/KBUF/2009/1/1/DailyHistory.html
```

В адресе все осталось по-прежнему за исключением той части, в которой указана дата. Теперь она записана так: /2009/1/1/. Интересно. А как можно загрузить страницу со сведениями о 2 января 2009 года без использования выпадающего меню? Просто измените параметр даты так, чтобы URL выглядел следующим образом:

```
www.wunderground.com/history/airport/KBUF/2009/1/2/DailyHistory.html
```

Загрузите новый URL в ваш веб-браузер, и вы получите историческую информацию о 2 января 2009 года. Итак, все, что вам необходимо сделать, чтобы узнать погоду за определенную дату, — модифицировать URL страницы Weather Underground. Запомните это — мы еще вернемся к данному вопросу.

Теперь загрузите одну-единственную страницу с помощью Python, используя библиотеку urllib2, для чего импортируйте ее, введя следующую строчку кода:

```
import urllib2
```

Чтобы загрузить страницу с данными о погоде за 1 января с помощью Python, используйте функцию urlopen.

```
page = urllib2.urlopen("www.wunderground.com/history/airport/KBUF/2009/1/1/DailyHistory.
html")
```

Таким образом вы загрузите весь HTML-файл, на который указывает URL в переменной страницы. Следующим шагом будет извлечение интересующего вас значения максимальной температуры из этого HTML. Beautiful Soup сделает выполнение данной задачи намного проще. Вслед за urllib2 импортируйте Beautiful Soup следующим образом:

```
from BeautifulSoup import BeautifulSoup
```

В конце вашего файла используйте Beautiful Soup, чтобы прочитать страницу, то есть произвести ее анализ.

```
soup = BeautifulSoup(page)
```

Не вдаваясь во все детали, скажу, что эта строчка кода прочитывает HTML, который представляет по сути одну длинную строку, а затем сохраняет элементы страницы, такие как заголовок или изображения, в удобном для работы виде.
Например, если вы хотите найти все изображения на странице, вы можете использовать вот что:

```
images = soup.findAll('img')
```

В результате вы получите перечень всех изображений на странице Weather Underground, отображаемых с помощью HTML-тега `<img />`. Вам нужно первое изображение на странице? Сделайте так:

```
first_image = images[0]
```

Хотите второе изображение? Измените ноль на единицу. Если вам нужно значение `src` в первом теге `<img />`, тогда используйте:

```
src = first_image['src']
```

Ладно, вас ведь не интересуют изображения. Вы просто хотите одно-единственное значение: максимальную температуру в городе Буффало, штат Нью-Йорк, в день 1 января 2009 года. Тогда она составляла 26 °F (или –3 °C). Чтобы найти в «супе» данное значение, нужно будет потрудиться чуть подольше, чем когда вы искали изображения, но метод применяется все тот же. Вам всего лишь нужно выяснить, что именно следует вставить в `findAll()`, а потому просмотрите исходный HTML.

Это легко делается в любом из популярных веб-браузеров. В Firefox, например, необходимо открыть меню «Вид» (View) и выбрать «Исходный код страницы» (Page Source). Появится окно с HTML-кодом текущей страницы, как это показано на рис. 2.5.

ПРИМЕЧАНИЕ

Beautiful Soup снабжена хорошей документацией и наглядными примерами, поэтому если что-то из описываемого здесь вас смущает, я горячо рекомендую вам поискать дополнительную информацию на том же сайте Beautiful Soup, с которого вы скачали библиотеку.

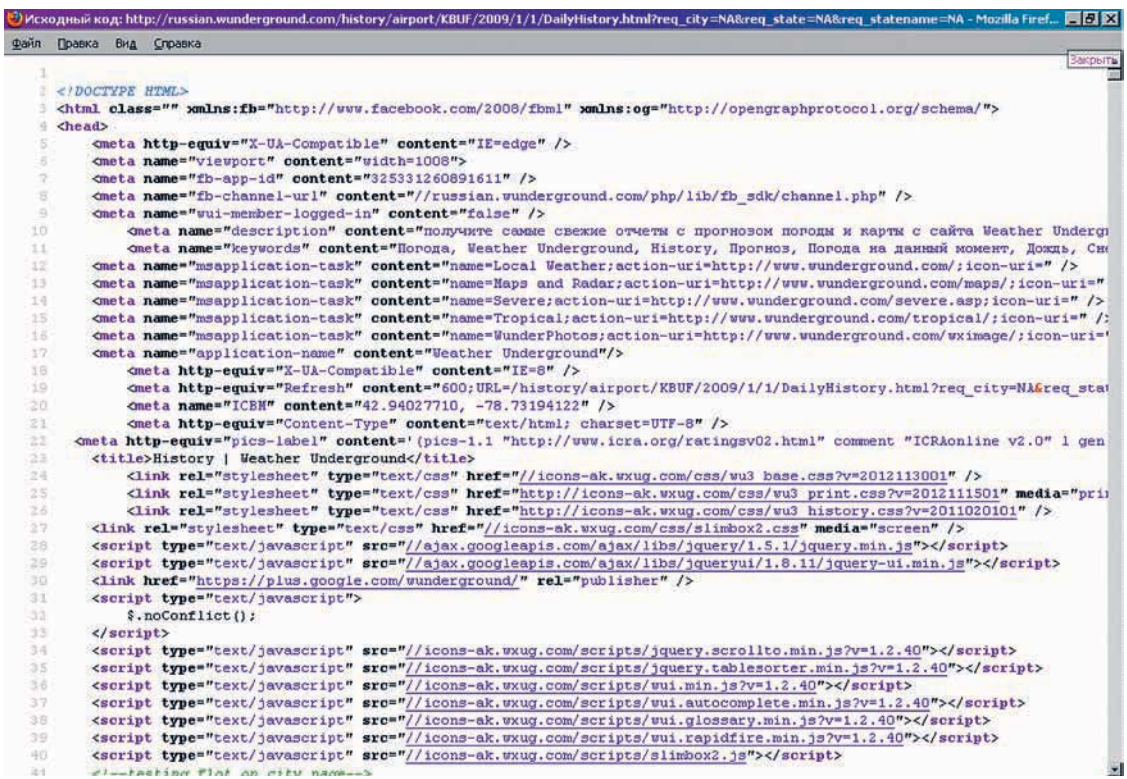


Рис. 2.5. Исходный HTML-код страницы Weather Underground

Прокрутите вниз до того места, где показана средняя температура, или просто воспользуйтесь функцией поиска в браузере, что будет быстрее. Отметьте 26 (–3, если по Цельсию). Это именно то, что вы хотите извлечь.

Строка замыкается тегом с классом nobr. Это и есть ваш ключ. Вы теперь можете найти все элементы на странице с классом nobr.

```
nobrs = soup.findAll(attrs={"class":"nobr"})
```

Как и в предыдущем примере, это дает вам в руки перечень всех случаев употребления nobr. Вас интересует шестой из них, который вы найдете с помощью следующей строчки:

```
print nobrs[5]
```

Это выдаст вам весь элемент, но вас интересует только 26 (–3). Внутри тега с классом nobr есть другой тег , а за ним идет 26 (–3). Вот то, что вам необходимо использовать.

```
dayTemp = nobrs[5].span.string
print dayTemp
```

Вуаля! Вы впервые в жизни извлекли нужное вам значение из HTML-кода веб-страницы. Следующий шаг — проанализировать все страницы за 2009 год и извлечь из них необходимые данные. Для этого вернитесь к первоначальному URL.

```
www.wunderground.com/history/airport/KBUF/2009/1/1/DailyHistory.html
```

Помните, как вы вручную изменили адрес, чтобы получить сведения о той дате, которая вас интересовала? Приведенный выше код относится к 1 января 2009 года. Если вам нужна страница за 2 января 2009 года, просто измените ту часть URL, в которой указана дата, на соответствующую. Чтобы получить данные за все дни 2009 года, загрузите все месяцы (с 1-го по 12-й), а затем загрузите каждый день каждого месяца. Ниже представлен весь скрипт с комментариями. Сохраните его в вашем файле get-weather-data.py.

```
import urllib2
from BeautifulSoup import BeautifulSoup

# Create/open a file called wunder.txt (which will be a comma-delimited file)
f = open('wunder-data.txt', 'w')

# Iterate through months and day
for m in range(1, 13):
   for d in range(1, 32):

      # Check if already gone through month
      if (m == 2 and d > 28):
         break
      elif (m in [4, 6, 9, 11] and d > 30):
         break
```

```
      # Open wunderground.com url
      timestamp = '2009' + str(m) + str(d)
      print "Getting data for " + timestamp
      url = "http://www.wunderground.com/history/airport/KBUF/2009/" +
str(m) + "/" + str(d) + "/DailyHistory.html"
      page = urllib2.urlopen(url)

      # Get temperature from page
      soup = BeautifulSoup(page)
      # dayTemp = soup.body.nobr.b.string
      dayTemp = soup.findAll(attrs={"class":"nobr"})[5].span.string

      # Format month for timestamp
      if len(str(m)) < 2:
         mStamp = '0' + str(m)
      else:
         mStamp = str(m)

      # Format day for timestamp
      if len(str(d)) < 2:
         dStamp = '0' + str(d)
      else:
         dStamp = str(d)

      # Build timestamp
      timestamp = '2009' + mStamp + dStamp

      # Write timestamp and temperature to file
      f.write(timestamp + ',' + dayTemp + '\n')

# Done getting data! Close file.
f.close()
```

Вы наверняка узнали первые две строчки кода, с помощью которых вы импортировали необходимые библиотеки, — urllib2 и BeautifulSoup.

```
import urllib2
from BeautifulSoup import BeautifulSoup
```

Далее создается текстовый файл под названием `wunder-data-txt` с правами на запись, используя метод `open()`. Все данные, которые вы извлечете, будут сохраняться в этом текстовом файле, расположенном в той же директории, куда вы сохранили свой скрипт.

```
# Create/open a file called wunder.txt (which will be a comma-delimited file)
f = open('wunder-data.txt', 'w')
```

С помощью следующей строчки кода, используя цикл for, компьютеру отдается команда просмотреть каждый месяц. Число, обозначающее месяц, указывается в переменной m. Следующий цикл сообщает компьютеру, что надо посетить каждый день каждого месяца. Каждый отдельный день месяца указывается в переменной d.

```
# Iterate through months and day
for m in range(1, 13):
   for d in range(1, 32):
```

▶ За дополнительной информацией о том, как работают циклы и происходит итерация, обратитесь к документации Python, расположенной по адресу http://docs.python.org/reference/compound_stmts.html

Обратите внимание на то, что для повторения операции по дням используется range (1, 32). Это означает, что повтор будет производиться для каждого числа, начиная с 1 и по 31. Однако не в каждом месяце в году есть 31 день. В феврале их 28; в апреле, июне, сентябре и ноябре — по 30. Температурных показателей за 31 апреля нет, потому что такого дня не существует. Обратите внимание на то, о каком месяце идет речь, и действуйте сообразно этому. Если текущий месяц — февраль, и число больше 28, прервитесь и переходите к следующему месяцу. Если вы хотите извлечь данные по нескольким годам, тогда вам необходимо использовать дополнительный оператор if, чтобы справиться с високосными годами.

Точно так же, если речь идет не о феврале, а об апреле, июне, сентябре или ноябре, и если значение текущего дня больше 30, нужно перейти к следующему месяцу.

```
# Check if already gone through month
if (m == 2 and d > 28):
   break
elif (m in [4, 6, 9, 11] and d > 30):
   break
```

И снова следующие строчки кода должны показаться вам знакомыми. Вы использовали их для извлечения данных из одной конкретной страницы сайта Weather Underground. Отличие состоит в переменной месяца и дня в URL. Ее просто нужно менять для каждого дня, а не оставлять статичной — все прочее тут без изменений. Загрузите страницу, используя библиотеку urllib2, произведите анализ контента с помощью Beautiful Soup, а затем извлеките максимальную температуру, но ищите шестое появление класса nobr.

```
# Open wunderground.com url
timestamp = '2009' + str(m) + str(d)
print "Getting data for " + timestamp
url = "http://www.wunderground.com/history/airport/KBUF/2009/" +
str(m) + "/" + str(d) + "/DailyHistory.html"
page = urllib2.urlopen(url)

# Get temperature from page
soup = BeautifulSoup(page)
# dayTemp = soup.body.nobr.b.string
dayTemp = soup.findAll(attrs={"class":"nobr"})[5].span.string
```

Предпоследний кусок кода отвечает за составление метки времени исходя из года, месяца и дня. Метка времени дается в формате «ггггммдд». Здесь вы можете задать любой формат, но на данном этапе чем проще — тем лучше.

```
# Format day for timestamp
if len(str(d)) < 2:
   dStamp = '0' + str(d)
else:
   dStamp = str(d)

# Build timestamp
timestamp = '2009' + mStamp + dStamp
```

И наконец, в файл `'wunder-data.txt'` с помощью метода `write()` добавляются температура и временна́я отметка.

```
# Write timestamp and temperature to file
f.write(timestamp + ',' + dayTemp + '\n')
```

По окончании работы со всеми месяцами и днями применяется `close()`.

```
# Done getting data! Close file.
f.close()
```

Осталось лишь запустить код, что вы и сделаете в своем терминале с помощью вот такой команды:

```
$ python get-weather-data.py
```

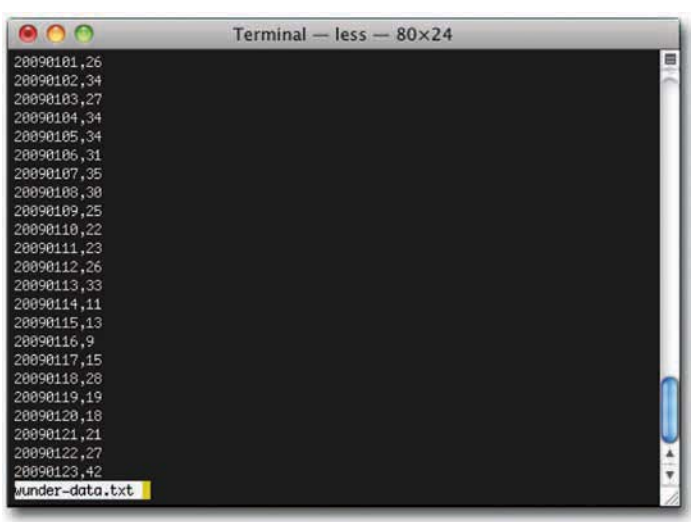


Рис. 2.6. Извлеченные данные о температуре за год

Прогон займет некоторое время, так что наберитесь терпения. По сути, в процессе выполнения программы ваш компьютер загрузит по очереди 365 страниц — по одной на каждый день 2009 года. Когда выполнение скрипта завершится, у вас в рабочем каталоге появится файл под названием `wunder-data.txt`. Откройте его, и там вы найдете нужные вам данные в формате с разделителями-запятыми. В первой колонке вы увидите отметку о времени, во второй колонке — температуру. Выглядеть все будет примерно так, как показано на рис. 2.6.

ГЕНЕРАЛИЗАЦИЯ ПРИМЕРА

Хотя вы всего лишь извлекли данные о погоде с сайта Weather Underground, этот процесс можно генерализировать для использования в работе с другими источниками. Извлечение данных, как правило, состоит из трех этапов:

1. Выявление паттернов.
2. Произведение итерации.
3. Сохранение данных.

В рассмотренном выше примере вы, чтобы получить данные по температуре, должны были найти два паттерна. Первый из них содержался в URL, а второй — в загружаемой веб-странице. Чтобы загрузить страницу о другом дне в 2009 году, вы изменяли часть URL с указанием дня и месяца. Значение температуры содержалось в шестом случае употребления класса `nobr` на странице. Если в URL нет очевидных паттернов, попробуйте сообразить, как можно получить URL всех страниц, которые вы хотите подвергнуть анализу, чтобы извлечь необходимые данные. Возможно, у сайта есть карта страниц, или вы можете пробежаться по индексу с помощью поиска. Так или иначе, в конечном итоге в вашем распоряжении должны оказаться URL всех страниц с данными.

После того как вы найдете паттерны, приступайте к итерации. Иными словами, вам нужно будет посетить — программным способом — все страницы по очереди, загрузить их и проанализировать. В предыдущем примере вы сделали это с помощью библиотеки Beautiful Soup, которая делает парсинг* XML и HTML в Python довольно легким. Если вы выберете другой язык программирования, для него, вероятно, найдется похожая библиотека.

И наконец, вам понадобится где-то сохранить полученные данные. Самое простое решение — это записать их в текстовый файл, в котором значения разделены запятыми. Но если у вас уже создана некая база данных, вы можете записать полученные значения в нее.

Задача становится несколько сложнее, когда приходится иметь дело с веб-страницами, использующими для загрузки данных JavaScript, но по своей сути процесс остается таким же.

Форматирование данных

Разные инструменты для визуализации используют различные форматы данных. Структура зависит от того, какую историю вы хотите рассказать. Поэтому чем больше гибкости вы проявите в работе со структурой ваших данных, тем больше у вас будет возможностей. Используйте приложения для форматирования данных, добавьте к этому немного ноу-хау в области

* Парсинг — процесс сбора контента (обычно с сайта) и его синтаксического анализа с целью перевода данных в более удобную для обработки структуру; осуществляется с помощью специальной программы — парсера. *Прим. пер.*

программирования, и у вас появится возможность отобразить данные в любом желаемом вами формате так, чтобы они отвечали вашим нуждам в конкретный момент.

Самый легкий путь — это найти программиста, который сможет отформатировать все ваши данные и произвести их анализ, но в таком случае вы всегда будете зависеть от кого-то другого. Подобная зависимость становится особенно очевидной на ранних этапах любого проекта, когда итерация и изучение данных имеют ключевое значение для создания дельной визуализации. Честно говоря, если бы я занимался подбором кадров, я предпочел бы нанять кого-то, кто знает, как работать с данными, а не того, кому каждый раз на начальном этапе проекта будет нужна помощь со стороны.

ЧТО Я УЗНАЛ О ФОРМАТИРОВАНИИ

Когда я только начал изучать статистику в средней школе, данные нам всегда предоставлялись в аккуратном табличном формате. Все, что от меня требовалось сделать, — это вставить несколько чисел в таблицу Excel или в мой потрясающий графический калькулятор (что было предпочтительнее, так как позволяло играть в «Тетрис», делая вид, будто я выполняю классное задание). На протяжении всех лет учебы в бакалавриате ситуация оставалась практически без изменений. Поскольку я занимался изучением методов и теорем анализа, мои преподаватели не тратили лишнего времени на работу с сырыми, неподготовленными данными. Создавалось впечатление, будто данные всегда существуют в удобном для работы виде.

Это и вполне понятно, учитывая ограничения во времени и т. п. Но в магистратуре я осознал, что в реальном мире данные почти никогда не встречаются в нужном вам формате. Какие-то значения отсутствуют вовсе, другие появляются без какого-либо контекста или сопровождаются нелогичными, противоречивыми подписями и опечатками. Очень часто необходимые данные разбросаны по разным таблицам, а вам нужно, чтобы они все оказались в одной-единственной, причем выстроенными по одному какому-то показателю, такому как имя или уникальный идентификатор.

С этой же проблемой я столкнулся и тогда, когда начал заниматься визуализацией. И проблема становилась тем острее, чем сильнее мне хотелось добиться большего с помощью имеющихся в наличии данных. Сегодня для меня уже не является чем-то необычайным то, что на получение данных в нужном мне формате я трачу столько же времени, сколько собственно на выполнение работы по их визуализации. Иногда на наведение порядка в данных у меня уходит даже больше времени. Поначалу это может показаться странным, но вы убедитесь, что создание графики становится намного легче, когда данные у вас организованы так четко, как это было во времена вводного курса в статистику в средней школе.

Далее вы познакомитесь с различными форматами данных, с доступными инструментами, с помощью которых вы сможете работать с этими форматами, и, наконец, обретете некоторые знания в области программирования, применяя те же принципы, что и при извлечении данных в предыдущем примере.

Форматы данных

Большинство людей привыкли работать с данными в Excel. И в этом нет ничего плохого, если вы будете делать в данной программе все — от анализа до визуализации. Но если вам понадобится выйти за эти рамки, тогда вам придется ознакомиться и с другими форматами данных. Смысл этих форматов в том, чтобы сделать ваши данные машиночитаемыми. Иными словами, структурировать ваши данные таким образом, чтобы их мог понять компьютер. Какой именно формат данных использовать, будет зависеть от инструмента и целей визуализации, но в большинстве случаев ваша база данных окажется в каком-то из следующих трех форматов: текст с разделителями, объектная нотация JavaScript или расширяемый язык разметки.

ТЕКСТ С РАЗДЕЛИТЕЛЯМИ

Текст с разделителями знаком большинству людей. Вы только что, в примере с извлечением данных, создали текстовый файл с разделителями-запятыми. Если вы привыкли думать о наборах данных в виде строчек и столбцов, то в текстовом файле с разделителями позиции будут разграничены именно разделителем. В файле с разделителями-запятыми это запятые. Помимо них в роли разделителя может выступать символ табуляции, а также пробелы, точки с запятыми, двоеточия, слеши — все что угодно, хотя чаще всего применяются запятые и табуляция.

Текст с разделителями-запятыми имеет очень широкое применение и читается большинством программ с таблицами, таких как Excel или Google Documents. Электронные таблицы вы также можете экспортировать в виде текста с разделителями. Если у вас в книге Excel много таблиц, тогда у вас и много файлов с разделителями, если вами не задано что-то иное.

Этот формат удобен также для обмена данными с другими людьми, так как он не зависит от какой-либо конкретной программы.

ОБЪЕКТНАЯ НОТАЦИЯ JAVASCRIPT (JSON)

Это один из самых распространенных форматов, который предлагают веб-API. Он разработан таким образом, чтобы его могли читать и машины, и люди, хотя если вам придется работать с таким файлом и внимательно всматриваться в него, вы вскоре окосеете. Он основан на нотации JavaScript, но является языконезависимым. Для JSON существует множество спецификаций, но для начала работы вам хватит понимания азов.

JSON работает с ключевыми словами и значениями и относится к ним как к объектам. Если вы конвертируете данные в формате JSON в значения, разделенные запятыми (CVS), каждый объект станет строчкой.

Далее в этой книге вы найдете множество приложений, языков и библиотек, принимающих JSON в качестве формата вводимых данных. Если вы собираетесь создавать диаграммы и графики для Сети, вы наверняка столкнетесь с этим форматом.

▶ Чтобы ознакомиться с полной спецификацией JSON, посетите `http://json.org`. Вам не нужно штудировать все детали, но данный ресурс может оказаться полезен, если вдруг вы не сумеете разобраться в данных, представленных в этом формате.

РАСШИРЯЕМЫЙ ЯЗЫК РАЗМЕТКИ (XML)

Другой распространенный в сети формат — это XML. Его часто применяют для передачи данных через различные API. Существует множество разных типов XML и спецификаций, но на самом базовом уровне его можно охарактеризовать как формат текстового документа со значениями, заключенными в теги. Например, фид RSS (Really Simple Syndication) — который люди используют, подписываясь на блоги вроде FlowingData, — по сути, представляет собой XML-файл (рис. 2.7).

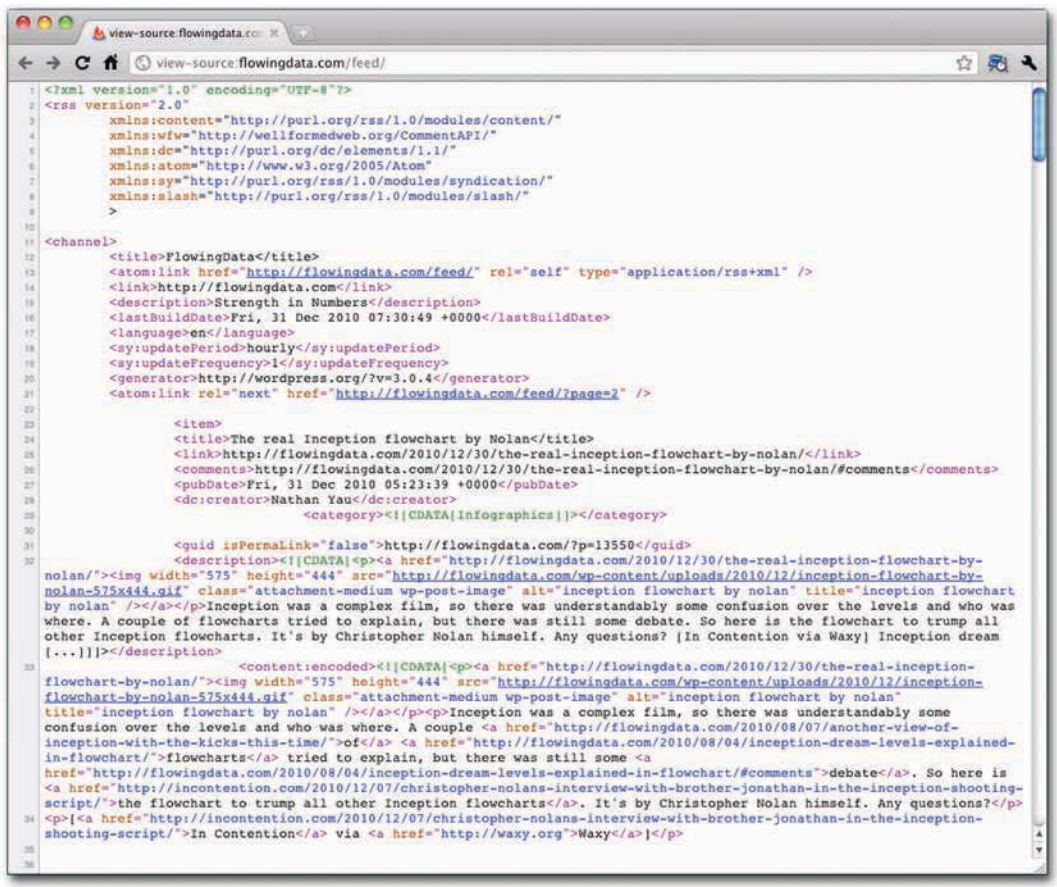


Рис. 2.7. Фрагмент RSS-фида FlowingData

В RSS перечислены недавно опубликованные статьи. Они заключены в теги `<item></item>`, и по каждой из них дается заголовок, описание, автор, дата публикации и еще кое-какие другие детали.

Производить парсинг XML относительно несложно, если использовать такие библиотеки, как Beautiful Soup для Python. После прочтения следующих разделов книги вы начнете лучше разбираться в XML, а также в CSV и JSON.

Инструменты форматирования

Еще пару лет назад для форматирования данных всегда приходилось писать скрипты. Стоит создать несколько скриптов, как начинаешь подмечать определенные паттерны в логике, и чем дальше, тем становится проще, однако во всех случаях на это уходит время. К счастью, вместе с ростом объемов данных стали появляться и некоторые инструменты для выполнения этой типовой процедуры.

GOOGLE REFINE

Google Refine — это результат эволюции Freebase Gridworks. Gridworks разрабатывался поначалу для открытой платформы данных Freebase в качестве собственного инструментального средства. Однако позже Freebase была приобретена компанией Google, отсюда и новое наименование. Google Refine представляет собой по сути Gridworks 2.0, но с более удобным в употреблении интерфейсом (см. рис. 2.8) и с бóльшим набором возможностей.

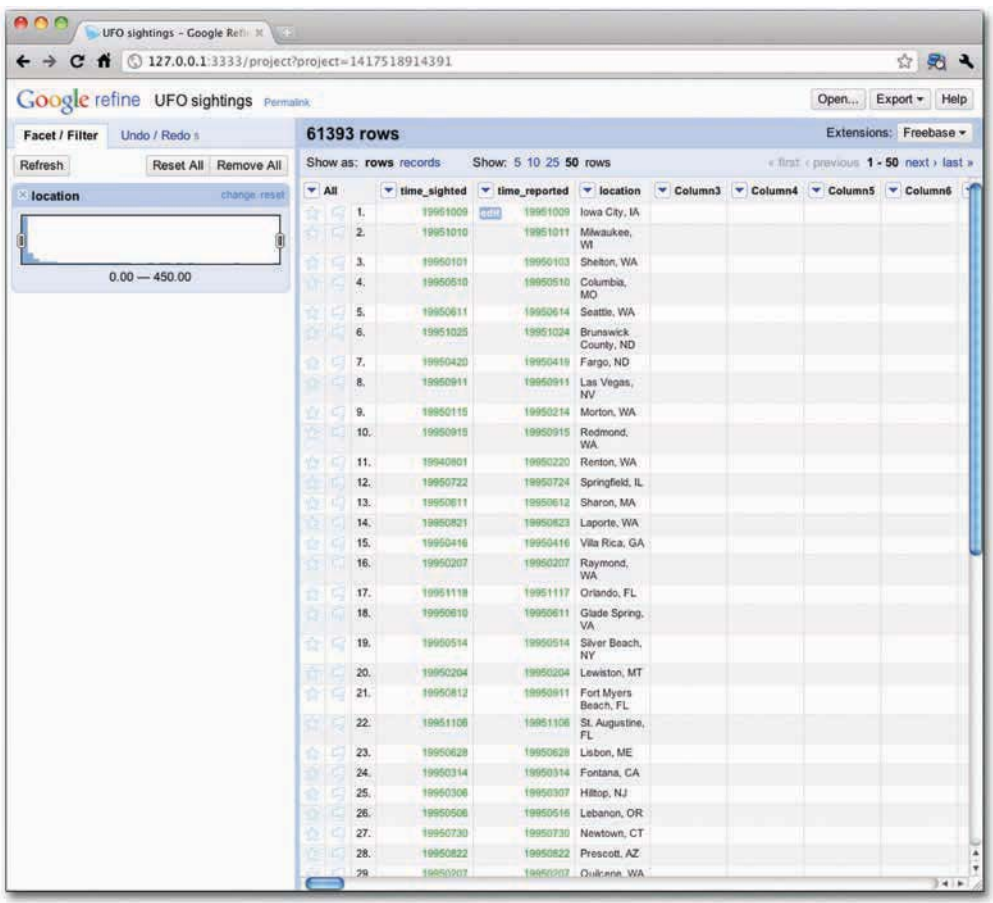


Рис. 2.8. Пользовательский интерфейс Google Refine

Программа запускается на вашем компьютере (только через браузер), что само по себе очень удобно, так как вам не придется беспокоиться из-за необходимости загружать закрытую информацию на сервера Google. Вся обработка данных происходит у вас на компьютере. У Refine по-прежнему открытый исходный код, так что если вами овладеют амбиции, вы сможете приспособить инструмент к вашим собственным «расширенным» потребностям.

Когда вы откроете Refine, вы увидите знакомый табличный интерфейс с вашими строчками и столбцами. В нем вы легко можете сортировать данные по полям и искать определенные величины. А еще вы можете относительно быстро находить ошибки и проводить операции объединения данных.

Допустим, по какой-то причине у вас на руках оказалась инвентарная опись вашей кухни. Вы можете загрузить данные в Refine и быстро найти нестыковки, такие как опечатки или разнящиеся классификации. Может, в слове «вилки» допущена ошибка и в перечне значатся «илки». Или вы хотите изменить классификацию и перевести все вилки, ложки и ножи в категорию «утварь». С Refine вы легко найдете все эти пункты и внесете необходимые правки. Если в результате изменения вам не понравятся или вы допустите ошибку, вы сможете вернуться к предыдущему варианту простой командой «Отменить» (Undo).

Когда вы перейдете к более сложным процедурам, то сможете также объединять различные источники данных, скажем, ваши собственные данные с массивами из Freebase, и создавать расширенные данные.

В любом случае Google Refine — это хороший инструмент, который всегда стоит иметь под рукой. Он мощный, и скачать его можно бесплатно, так что я горячо рекомендую вам хотя бы опробовать его.

▶ Скачайте Google Refine с `http://code.google.com/p/google-refine` и просмотрите обучающее руководство, чтобы узнать, как извлечь из этого инструмента максимум пользы.

MR. DATA CONVERTER

Часто бывает и так, что у вас есть возможность получить все данные в Excel, но затем необходимо конвертировать их в другой формат, который в большей степени соответствует вашим потребностям. Подобное происходит почти всегда в случае создания графики для Сети. Вы уже умеете экспортировать таблицы Excel в формате CSV, но как быть, если вам нужно сделать что-то еще? Вам поможет Mr. Data Converter.

Mr. Data Converter — это простой и бесплатный инструмент. Его создал Шэн Картер (Shan Carter), редактор графики в New York Times. Бóльшую часть своего рабочего времени Картер проводит в создании интерактивной графики для онлайн-версии газеты. Ему часто приходится конвертировать данные, чтобы они подходили для использования в программах, с которыми он работает, а потому неудивительно, что он создал инструмент, упрощающий этот процесс.

Пользоваться Mr. Data Converter легко. Как вы можете убедиться, взглянув на рис. 2.9, интерфейс совершенно простой. Все, что

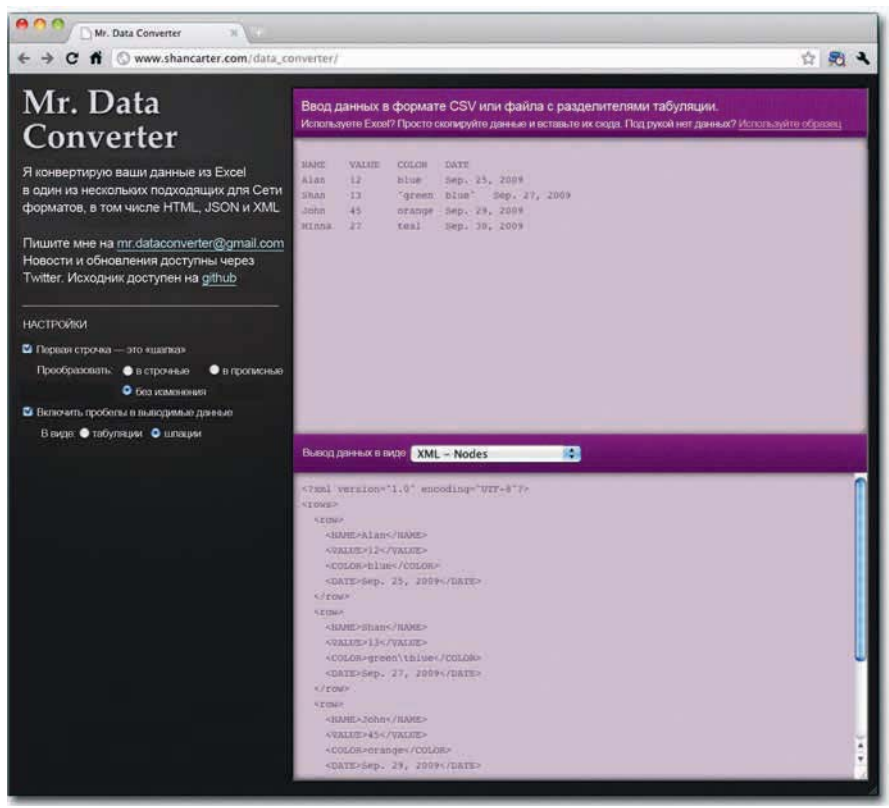


Рис. 2.9. Mr. Data Converter существенно упрощает перевод данных из одного формата в другой

▶ Опробуйте Mr. Data Converter на сайте `http://www.shancarter.com/data_converter` или скачайте исходник с GitHub по адресу at `https://github.com/shancarter/Mr-Data-Converter`, чтобы конвертировать свои таблицы Excel в формат, удобный для размещения в Сети.

вам необходимо сделать, — это скопировать данные из Excel и вставить их в область ввода в верхней части экрана, а внизу указать, в каком формате вы хотите получить их. На выбор у вас XML, JSON и ряд других форматов.

Исходный код Mr. Data Converter также доступен — на тот случай, если вы захотите сделать свой собственный вариант или дополнить существующий.

MR. PEOPLE

Вдохновленный примером Картера с его Mr. Data Converter, заместитель руководителя отдела графики New York Times Мэтью Эриксон (Matthew Ericson) создал свой инструмент — Mr. People. Как и Mr. Data Converter, Mr. People дает возможность копировать и вставлять данные в текстовое поле, а затем анализирует и извлекает их за вас. Однако, как вы могли догадаться, Mr. People предназначен специально для выполнения парсинга имен.

Может, у вас есть длинный перечень имен, не отформатированных ни по какому принципу, и вы хотите установить для каждого включенного в список человека его имя, фамилию, а заодно и инициалы второго имени, титул, если таковой есть, и пр. А может, множество людей перечислены друг за другом в один ряд. Вот здесь-то на сцену и выходит Mr. People. Скопируйте имена и вставьте их так, как показано на рис. 2.10. Вы получите красивую аккуратную таблицу, которую вы сможете использовать в вашей любимой программе для работы с таблицами (рис. 2.11).

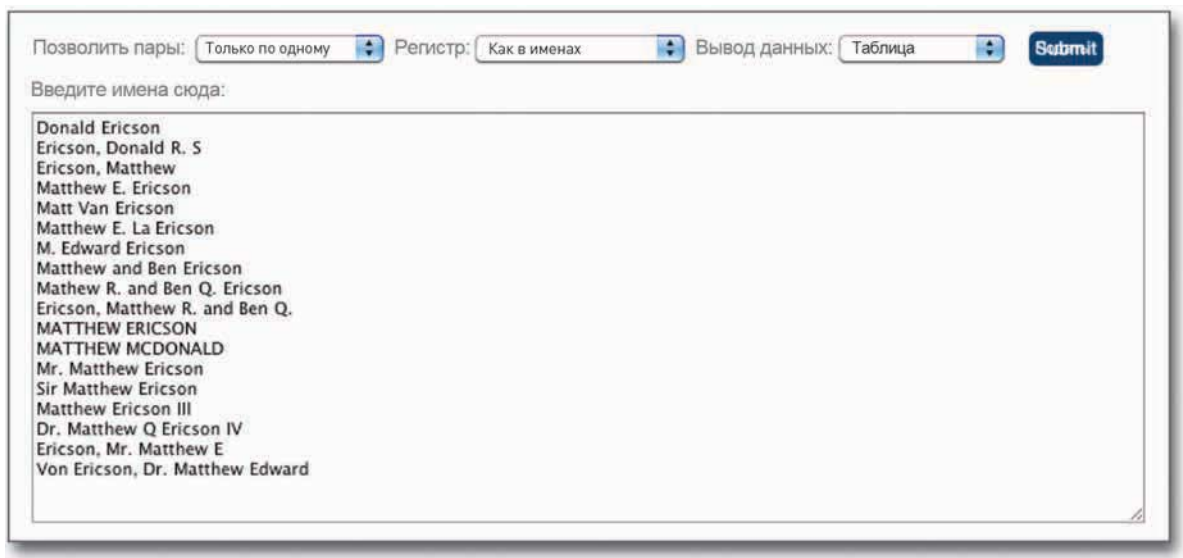
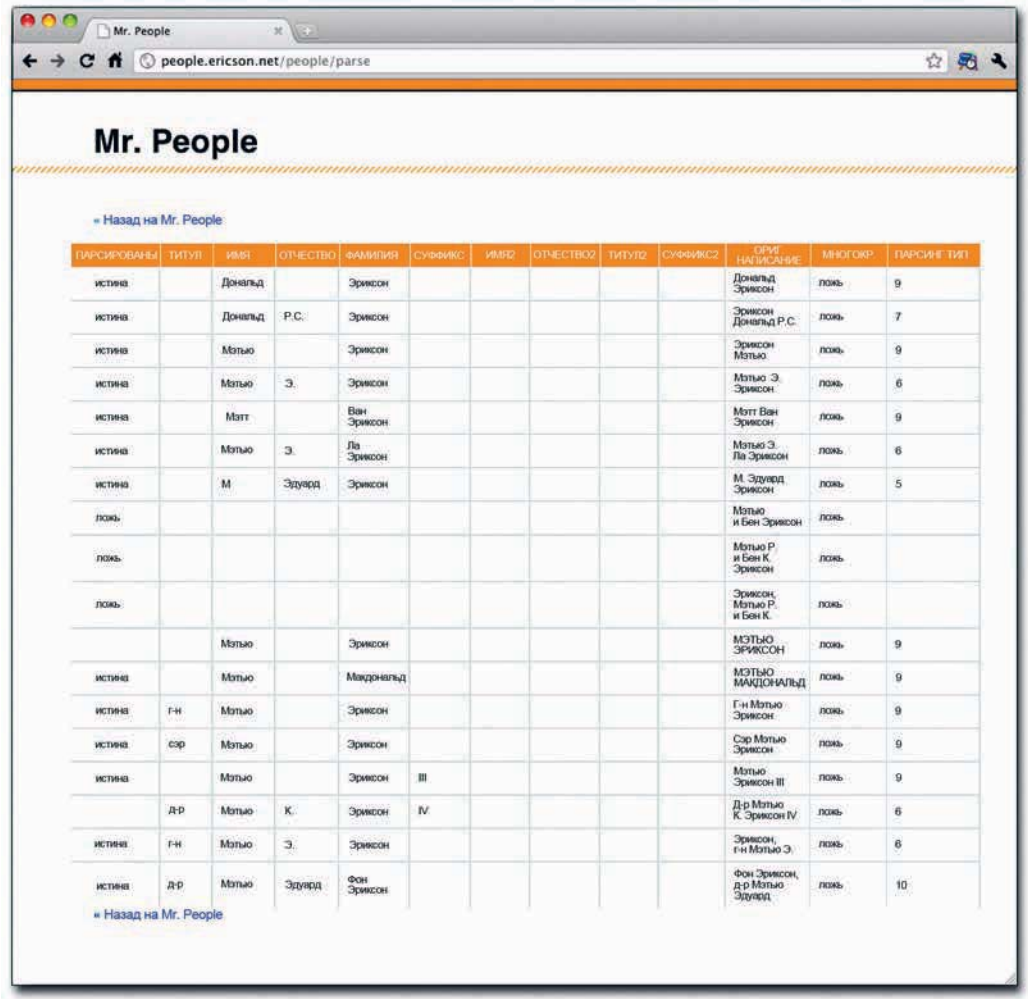


Рис. 2.10. Страница ввода имен в Mr. People



ПАРСИРОВАНЫ	ТИТУЛ	ИМЯ	ОТЧЕСТВО	ФАМИЛИЯ	СУФФИКС	ИМЯ2	ОТЧЕСТВО2	ТИТУЛ2	СУФФИКС2	ОРИГ НАПИСАНИЕ	МНОГОКР	ПАРСИНГ ТИП
истина		Дональд		Эриксон						Дональд Эриксон	ложь	9
истина		Дональд	Р.С.	Эриксон						Эриксон Дональд Р.С.	ложь	7
истина		Мэтью		Эриксон						Эриксон Мэтью	ложь	9
истина		Мэтью	Э.	Эриксон						Мэтью Э. Эриксон	ложь	6
истина		Мэтт		Ван Эриксон						Мэтт Ван Эриксон	ложь	9
истина		Мэтью	Э.	Ла Эриксон						Мэтью Э. Ла Эриксон	ложь	6
истина		М	Эдуард	Эриксон						М. Эдуард Эриксон	ложь	5
ложь										Мэтью и Бен Эриксон	ложь	
ложь										Мэтью Р. и Бен К. Эриксон	ложь	
ложь										Эриксон, Мэтью Р. и Бен К.	ложь	
		Мэтью		Эриксон						МЭТЬЮ ЭРИКСОН	ложь	9
истина		Мэтью		Макдональд						МЭТЬЮ МАКДОНАЛЬД	ложь	9
истина	г-н	Мэтью		Эриксон						Г-н Мэтью Эриксон	ложь	9
истина	сэр	Мэтью		Эриксон						Сэр Мэтью Эриксон	ложь	9
истина		Мэтью		Эриксон	III					Мэтью Эриксон III	ложь	9
	д-р	Мэтью	К.	Эриксон	IV					Д-р Мэтью К. Эриксон IV	ложь	6
истина	г-н	Мэтью	Э.	Эриксон						Эриксон, г-н Мэтью Э.	ложь	6
истина	д-р	Мэтью	Эдуард	Фон Эриксон						Фон Эриксон, д-р Мэтью Эдуард	ложь	10

Рис. 2.11. Имена в формате таблицы после обработки в Mr. People

▶ Используйте Mr. People на `http://people.ericson.net` или скачайте исходный код на языке Ruby с GitHub, чтобы использовать парсер имен в своих собственных скриптах. Для этого зайдите на `http://github.com/mericson/people`.

Подобно Mr. Data Converter, Mr. People также является программой с открытым исходным кодом, и ее можно скачать с GitHub.

ПРОГРАММЫ ТАБЛИЧНЫХ ВЫЧИСЛЕНИЙ

Конечно, если вам нужно просто рассортировать некие данные или внести небольшие изменения в отдельные пункты, вы всегда сможете воспользоваться вашей любимой табличной программой. Если вам предстоят небольшие правки, идите проторенной дорогой. Если же нет, тогда попробуйте сначала то, о чем говорилось выше (особенно если у вас огромный массив данных), или подумайте о создании кода, специально предназначенного для конкретной задачи.

Форматирование с помощью кода

Хотя программы, работающие по методу «указал и щелкнул», могут быть полезны, иногда они выдают не совсем то, что вам нужно, особенно если вы работаете с данными достаточно длительное время. Некоторые программы не очень хорошо справляются с объемными файлами данных, их работа замедляется или они и вовсе выходят из строя.

Что делать в подобной ситуации? Вы можете поднять белый флаг и сдаться, хотя это будет непродуктивное решение. Но вы также можете написать небольшую программу и выполнить работу с ее помощью. Располагая кодом, вы станете намного гибче и сумеете подогнать свой скрипт под имеющиеся данные.

Так что давайте, с ходу смотрите пример того, как можно легко переключаться с одного формата данных на другой с помощью всего нескольких строчек кода.

ПРИМЕР: ПЕРЕКЛЮЧЕНИЕ МЕЖДУ РАЗЛИЧНЫМИ ФОРМАТАМИ ДАННЫХ

В этом примере применяется Python, но вы, конечно, можете использовать тот язык, которому отдаете предпочтение. Логика остается такой же, только синтаксис будет несколько иным. (Я люблю разрабатывать приложения в Python, а потому производственный процесс у меня хорошо согласуется с обработкой исходных данных с помощью Python.)

Вернитесь к предыдущему примеру с анализом данных и используйте полученный файл `wunder-data.txt`, в котором содержатся даты и показания о температуре в Буффало за 2009 год. Первые строчки выглядят так:

```
20090101,26
20090102,34
20090103,27
20090104,34
20090105,34
20090106,31
20090107,35
20090108,30
20090109,25
...
```

Это CSV-файл, но, допустим, вы хотите, чтобы данные были в XML, вот в таком формате:

```
<weather_data>
   <observation>
      <date>20090101</date>
```

```
      <max_temperature>26</max_temperature>
   </observation>
   <observation>
      <date>20090102</date>
      <max_temperature>34</max_temperature>
   </observation>
   <observation>
      <date>20090103</date>
      <max_temperature>27</max_temperature>
   </observation>
   <observation>
      <date>20090104</date>
      <max_temperature>34</max_temperature>
   </observation>
   …
</weather_data>
```

Температура за каждый день содержится в тегах <observation>, включающих теги <date> и <max_temperature>.

Чтобы конвертировать CSV в формат XML, как это сделано выше, вы можете использовать следующий фрагмент кода:

```
import csv
reader = csv.reader(open('wunder-data.txt', 'r'), delimiter=",")
print '<weather_data>'

for row in reader:
   print '<observation>'
   print '<date>' + row[0] + '</date>'
   print '<max_temperature>' + row[1] + '</max_temperature>'
   print '</observation>'

print '</weather_data>'
```

Необходимые модули вы импортируете, как и прежде. В данном случае, чтобы прочитать `wunder-data.txt`, вам нужен только CSV-модуль.

```
import csv
```

Вторая строчка кода открывает для чтения `wunder-data.txt`, используя `open()`, а затем загружает его при помощи метода `csv.reader()`.

```
reader = csv.reader(open('wunder-data.txt', 'r'), delimiter=",")
```

Обратите внимание на то, что в качестве разделителя здесь стоит запятая. Если в файле для этой цели используется табуляция, тогда вам следует указать разделитель в виде `'\t'`.

Затем в строке 3 вы вводите в XML-файл первый тег.

```
print '<weather_data>'
```

В основном фрагменте цикл повторяется для каждой строчки данных, и они отображаются в том формате, в каком вам нужно. В настоящем примере каждая строчка в заголовке CSV эквивалентна каждому наблюдению (observation) в XML.

```
for row in reader:
   print '<observation>'
   print '<date>' + row[0] + '</date>'
   print '<max_temperature>' + row[1] + '</max_temperature>'
   print '</observation>'
```

В каждой строчке есть два значения: даты и максимальной температуры.
Завершите перевод в XML закрывающим тегом:

```
print '</weather_data>'
```

Главную роль здесь играют два основных момента. Вы вводите данные и затем производите итерацию по ним, меняя каждую строчку определенным образом. Если бы вам предстояло конвертировать полученный XML обратно в CSV, логика действий была бы точно такой же. Как видно из следующего фрагмента, отличие состоит лишь в том, что вы используете для парсинга XML-файла другой модуль.

```
from BeautifulSoup import BeautifulStoneSoup

f = open('wunder-data.xml', 'r')
xml = f.read()

soup = BeautifulStoneSoup(xml)
observations = soup.findAll('observation')
for o in observations:
   print o.date.string + "," + o.max_temperature.string
```

Код выглядит по-другому, но, по сути, вы делаете то же самое. Вместо CSV-модуля вы импортируете BeautifulStoneSoup с BeautifulSoup. Вспомните, что вы уже использовали BeautifulSoup для анализа HTML-сайта Weather Underground. BeautifulStoneSoup используется для парсинга более распространенного XML.

Вы можете открыть XML-файл для чтения, используя `open()`, а затем загрузить контент в переменную `xml`. На данном этапе контент сохраняется в виде строки. Для начала парсинга передайте `xml`-строку в `BeautifulStoneSoup` для итерации через каждый `<observation>` в XML-файле. Используйте `findAll()` для того, чтобы извлечь все наблюдения и, наконец, как и при

переводе CSV в XML, прогнать цикл по каждому наблюдению и отобразить значения в нужном вам формате.

Это приведет вас туда, откуда вы начинали:

```
20090101,26
20090102,34
20090103,27
20090104,34
…
```

Для полноты картины вот вам еще и код, с помощью которого можно конвертировать CSV в формат JSON.

```
import csv
reader = csv.reader(open('wunder-data.txt', 'r'), delimiter=»,»)

print "{ observations: ["
rows_so_far = 0
for row in reader:

   rows_so_far += 1

   print '{'
   print '"date": ' + '"' + row[0] + '", '
   print '"temperature": ' + row[1]

   if rows_so_far < 365:
      print " },"
   else:
      print " }"

print "] }"
```

Пробегитесь глазами по строчкам, чтобы понять, что здесь происходит. Мы снова имеем дело с той же логикой, только с иным результатом. Если вы запустите приведенный выше код, то получите свои данные в формате JSON. Вот как они будут выглядеть:

```
{
   "observations": [
      {
         "date": "20090101",
         "temperature": 26
      },
      {
         "date": "20090102",
```

```
         "temperature": 34
      },
      …
   ]
}
```

Это все те же данные с датой и температурой, но в еще одном формате. Компьютеры просто обожают разнообразие.

Внесите в цикл логику

Если вы посмотрите на код для конвертирования CSV-файла в JSON, вы наверняка заметите оператор `if-else` в цикле `for` после трех строчек `print`. Его назначение — проверить, не является ли текущая итерация по рядам с данными последней. Если нет, тогда в конце записи точка не нужна. В противном случае поставьте ее. Это часть спецификации JSON. И здесь у вас больше возможностей.

Вы можете проверить, например, не поднималась ли максимальная температура выше определенного уровня, и создать новое поле со значением 1, если температура в конкретный день оказывалась выше некоего заданного порога, или 0, если нет. Вы можете также создать категории или пометить дни, за которые нет данных.

На самом деле проверка данных на предмет надпороговых значений — это не единственный доступный вариант. Вы можете подсчитать скользящее среднее или разницу между текущим и предыдущим днями. В рамках цикла вы можете много чего сделать, чтобы обогатить сырые данные. Здесь упомянуто далеко не все, поскольку вам позволено делать что угодно — от поверхностных правок до глубокого анализа. Посмотрите на следующий простой пример.

Вернитесь к вашему первоначальному файлу `wunder-data.txt` и создайте третью колонку, показывающую, превышает ли дневная температура точку замерзания. Так, 0 будет значить, что температура превысила заданный порог, а 1 — что она оказалась на уровне точки замерзания или ниже.

```
import csv
reader = csv.reader(open('wunder-data.txt', 'r'), delimiter=",")
for row in reader:
   if int(row[1]) <= 32:
      is_freezing = '1'
   else:
      is_freezing = '0'

   print row[0] + "," + row[1] + "," + is_freezing
```

Как и прежде, данные считываются из CSV-файла в Python, после чего производится итерация по каждой строчке. Все дни проверяются и помечаются соответствующим образом.

Пример, конечно, простой, но он наглядно демонстрирует, как вы можете развить эти принципы для форматирования или обогащения данных как вашей душе угодно. Запомните эти три шага: загрузка, цикл, обработка — и действуйте свободно.

Закругляясь

В настоящей главе обсуждалось, как находить необходимые данные и что с ними делать после нахождения. В процессе визуализации это очень важный этап, чтобы не сказать — самый важный. Информационная графика интересна настолько, насколько интересны лежащие в ее основе данные. Вы можете украсить свою диаграмму или график как хотите, но именно данные (или результаты вашего анализа этих данных) будут оставаться основой основ. И теперь, когда вы знаете, откуда и как доставать данные, вы готовы двигаться дальше.

Вы также впервые попробовали свои силы в программировании. Вы извлекли данные с сайта, а затем отформатировали и перегруппировали их. Этот прием еще сослужит вам добрую службу в следующих главах. Однако самое важное в коде — логика. Вы использовали Python, но с таким же успехом вы могли бы обратиться и к Ruby, и к Perl, и к PHP. Каким бы языком вы ни пользовались, принцип действия остается неизменным. Когда вы освоите один из языков программирования — а если вы уже состоявшийся программист, вы подтвердите мои слова, — вам будет гораздо легче затем освоить еще один.

Не всегда нужно прибегать к кодам. В некоторых случаях приложения, действующие по принципу перетаскивания (drag and drop), способны существенно облегчить вашу работу, и в таких случаях вам стоит этим пользоваться. В конечном счете, чем больше инструментов будет в вашем инструментарии, тем меньше опасность, что вы увязнете в процессе работы.

Итак, данные у вас уже есть. Теперь пора приступить к визуализации.

Выбор инструментов для визуализации данных

Из предыдущей главы вы узнали, как находить данные и как приводить их к тому формату, который вам необходим. Теперь вы готовы приступить к визуализации. Один из вопросов, которые люди чаще всего задают мне на этом этапе, звучит так: «Какое программное обеспечение мне следует использовать для визуализации данных?»

К счастью, вариантов у вас много. Часть программ уже предустановлена у вас на компьютере и работает по принципу перетаскивания (drag and drop). Чтобы пользоваться другими инструментами, вам потребуются некоторые навыки программирования. А есть и такие средства, которые не были разработаны специально для графического представления данных, но тем не менее могут оказаться вам полезными. Настоящая глава посвящена теме выбора программного обеспечения для визуализации данных.

Чем бóльшим количеством инструментов вы овладеете, чем лучше станете разбираться в их преимуществах, тем меньше вероятность, что вы растеряетесь, не зная, что делать далее с имеющимся у вас массивом данных, и тем выше шанс, что вы создадите диаграмму или график, достойно представляющие ваше ви́дение.

Готовые решения для визуализации

Готовые решения — это самый простой вариант, с которого могут начать новички. Скопируйте и вставьте некоторое количество данных или загрузите CSV-файл — и готово. Остается лишь щелкнуть мышью по типу диаграммы, которая вам нужна, ну и, может быть, поменять кое-какие опции тут и там.

Опции

Выбор готовых инструментов не так уж и богат и зависит от приложения, для которого они были разработаны. Некоторые из этих инструментов, такие как Microsoft Excel или Google Documents, предназначены для управления данными и графикой на базисном уровне, в то время как другие сконструированы для более углубленного анализа и визуального исследования.

MICROSOFT EXCEL

С этой программой вы наверняка знакомы. Перед вами — всем известная таблица, куда вы обычно вводите свои данные (рис. 3.1).

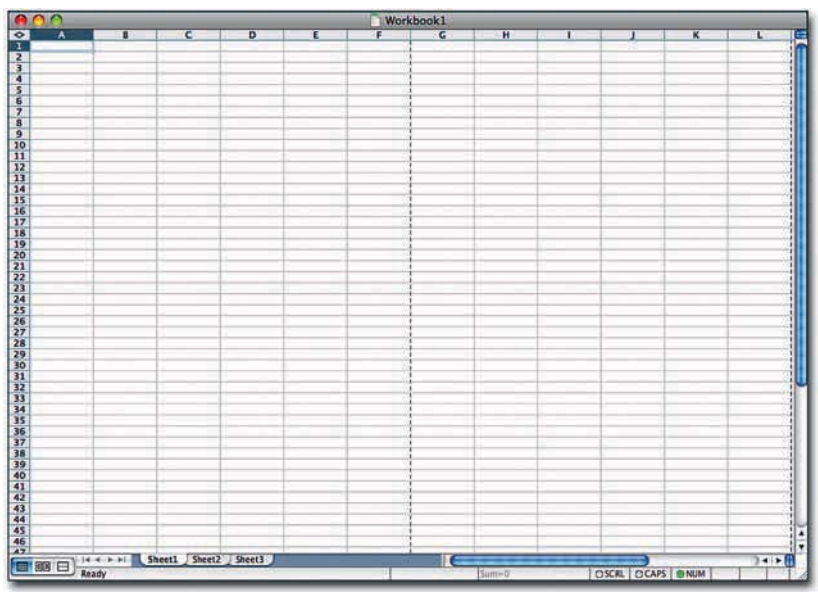


Рис. 3.1. Таблица в Microsoft Excel

Вы можете щелкнуть мышью по кнопке с маленьким графиком и сделать такую диаграмму, какую хотите. Перед вами на выбор все их стандартные типы (рис. 3.2), такие как гистограмма, линейный график, круговая и точечная диаграммы.

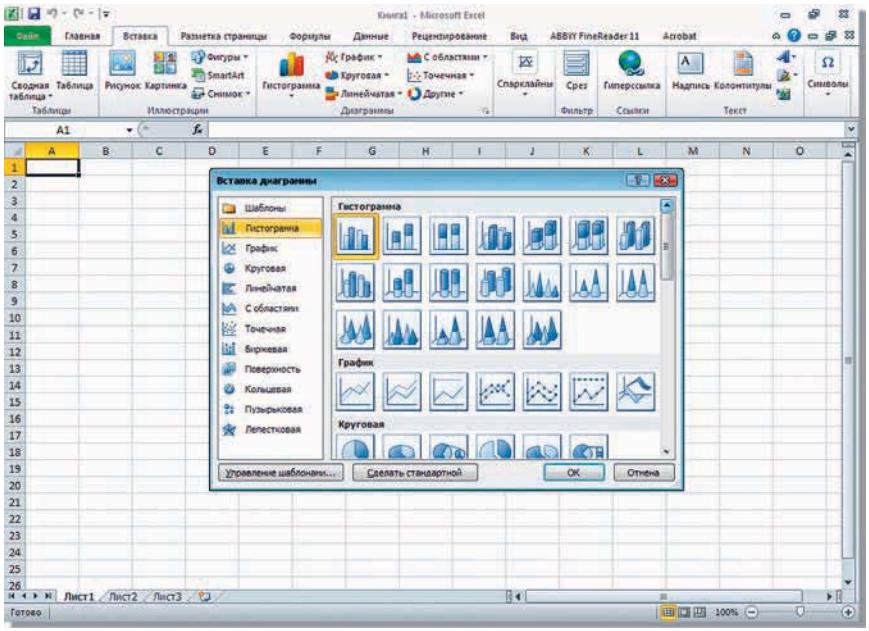


Рис. 3.2. Варианты диаграмм в Microsoft Excel

Некоторые люди не воспринимают Excel всерьез, но эта программа отнюдь не плоха, если применять ее в подходящих целях. Я, например, не использую Excel, если мне необходимо провести глубокий анализ или разработать графический объект для публикации. Но если я получаю небольшой набор данных в файле формата Excel, как оно часто бывает, и хочу быстренько составить общее представление о том, что же это передо мной, то я предпочитаю парой щелчков мышью состряпать диаграмму прямо в этой всеми любимой табличной программе.

ДИАГРАММЫ И ВПРАВДУ МОГУТ БЫТЬ ЗАБАВНЫМИ

Первая диаграмма, которую я построил на компьютере, делалась в Microsoft Excel, и создал я ее в пятом классе для школьной ярмарки научных проектов. Я и мой партнер по этой работе пытались выяснить, по какой поверхности улитки передвигаются быстрее всего. Смею вас заверить, это было революционное исследование.

Даже в те времена, как сейчас помню, я изрядно позабавился, создавая диаграмму. Мне понадобилась целая вечность, чтобы освоиться в программе (тогда компьютер был для меня новинкой), но когда я в конце концов разобрался, что к чему, процесс принес мне настоящее удовольствие. Я ввел числа в таблицу и тут же получил диаграмму, которую можно было раскрасить в любой цвет, какой захочется. Я выбрал ярко-желтый, от которого аж глаза слепнут, — мне показалось, что это самое то.

Легкость в использовании — вот что делает программу Excel столь привлекательной для большого количества людей, и это замечательно. Если же вам нужна более качественная информационная графика, тогда не останавливайтесь на этом. Для подобных целей вам лучше подойдут другие инструменты.

GOOGLE SPREADSHEETS

Google Spreadsheets — это, по сути, облачная версия Microsoft Excel с обычным табличным интерфейсом, в чем вы можете сами убедиться, посмотрев на рис. 3.3.

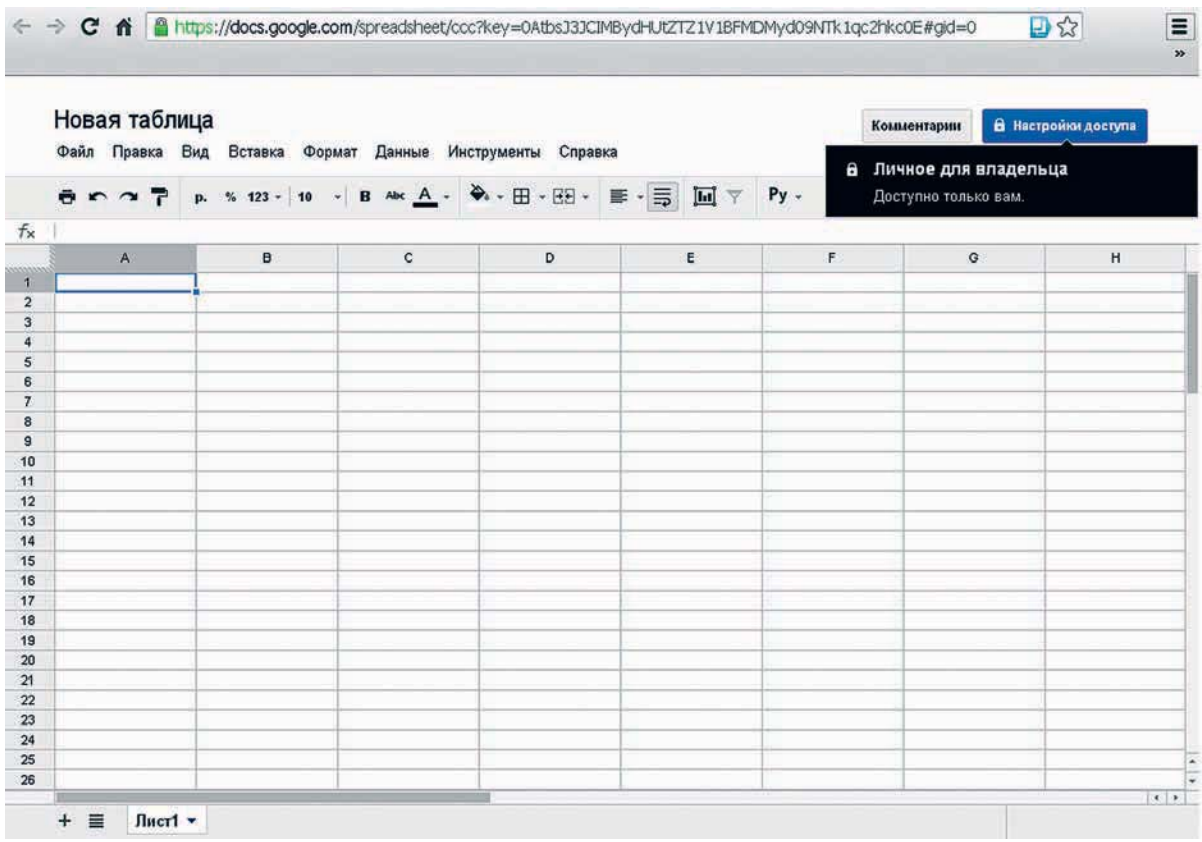


Рис. 3.3. Таблицы в Google Spreadsheets

Данная программа также предлагает стандартный набор диаграмм, как это показано на рис. 3.4.

Однако у Google Spreadsheets есть и некоторые преимущества перед Microsoft Excel. Во-первых, благодаря тому, что данные сохраняются на серверах Google, вы можете получить к ним доступ с любого компьютера, на котором установлен веб-браузер. Просто зайдите со своей учетной записи и действуйте. Вы можете легко обмениваться своими таблицами с другими людьми и работать

с ними вместе в реальном времени. Google Spreadsheets предлагает также и некоторые дополнительные возможности для вычерчивания диаграмм с помощью гаджетов, как это показано на рис. 3.5.

Многие из доступных гаджетов бесполезны, но среди них есть и несколько весьма толковых. Например, имея временны́е ряды данных, вы легко можете создать динамическую диаграмму (такую же, как у Ханса Рослинга). А с интерактивной диаграммой временных рядов, представленной на рис. 3.6, вы, возможно, уже знакомы, если заходили на Google Finance.

Рис. 3.4. Варианты графиков и диаграмм в Google Spreadsheets

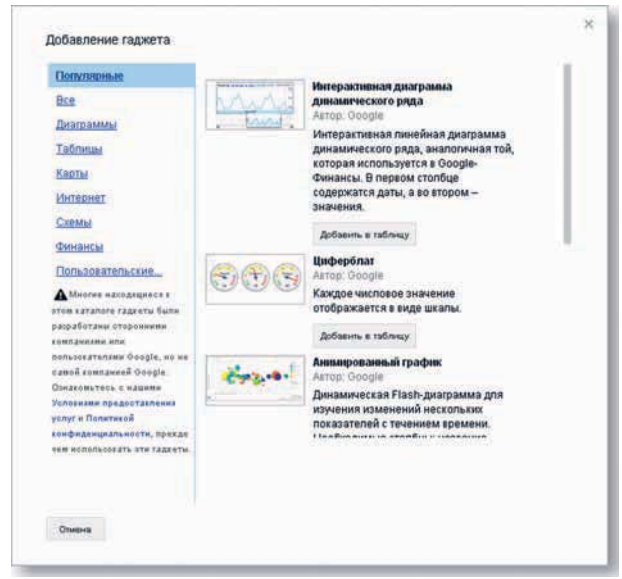


Рис. 3.5. Гаджеты Google

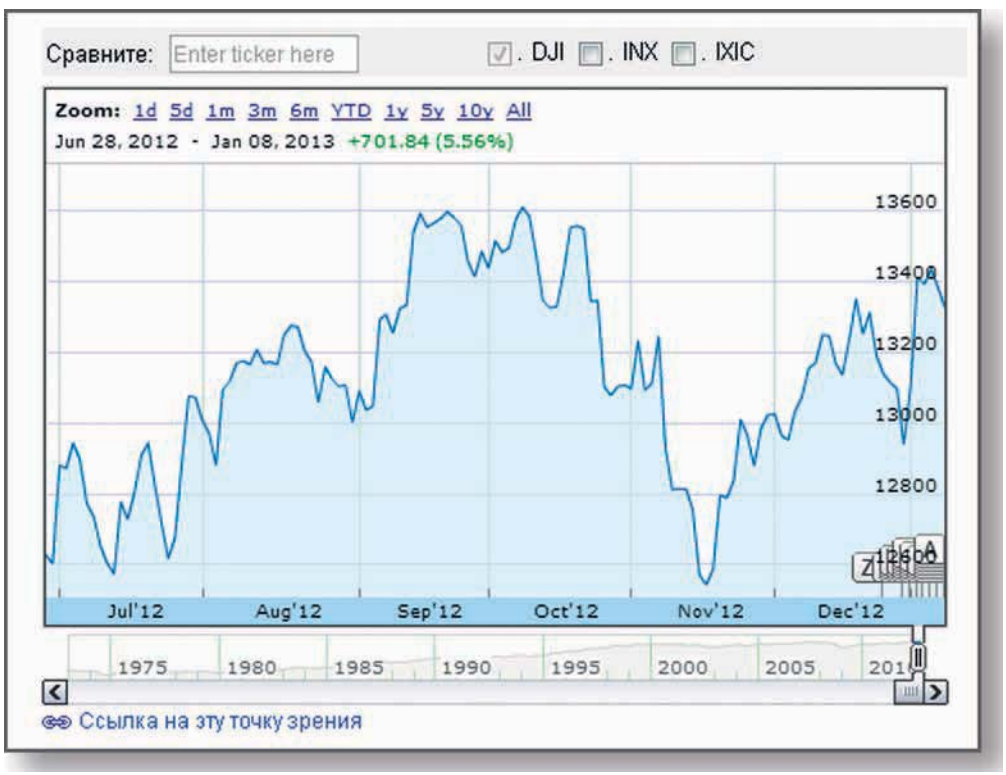


Рис. 3.6. Google Finance

▶ Посетите Google Docs на `http://docs.google.com` и попробуйте поработать с таблицами.

MANY EYES

Many Eyes — это действующий исследовательский проект Лаборатории визуальной коммуникации IBM. Представляет собой онлайн-приложение, дающее возможность загружать свои данные в виде текстового файла с разделителями и изучать этот файл с помощью набора интерактивных инструментов визуализации. Изначально сервис Many Eyes создавался с целью выяснить, способны ли люди исследовать большие массивы данных в группах — отсюда и название*. Другими словами, сможет ли группа (множество глаз) высмотреть в большом массиве данных интересные моменты быстрее и эффективнее, чем если бы это делал один человек, или найти в данных что-то такое, чего бы не нашел одиночка.

Хотя Many Eyes как инструмент для анализа социальных данных не получил особого распространения, он тем не менее может быть вам полезен. В нем есть почти все традиционные типы визуализации, такие как линейный график (рис. 3.7) и диаграмма рассеяния (рис. 3.8).

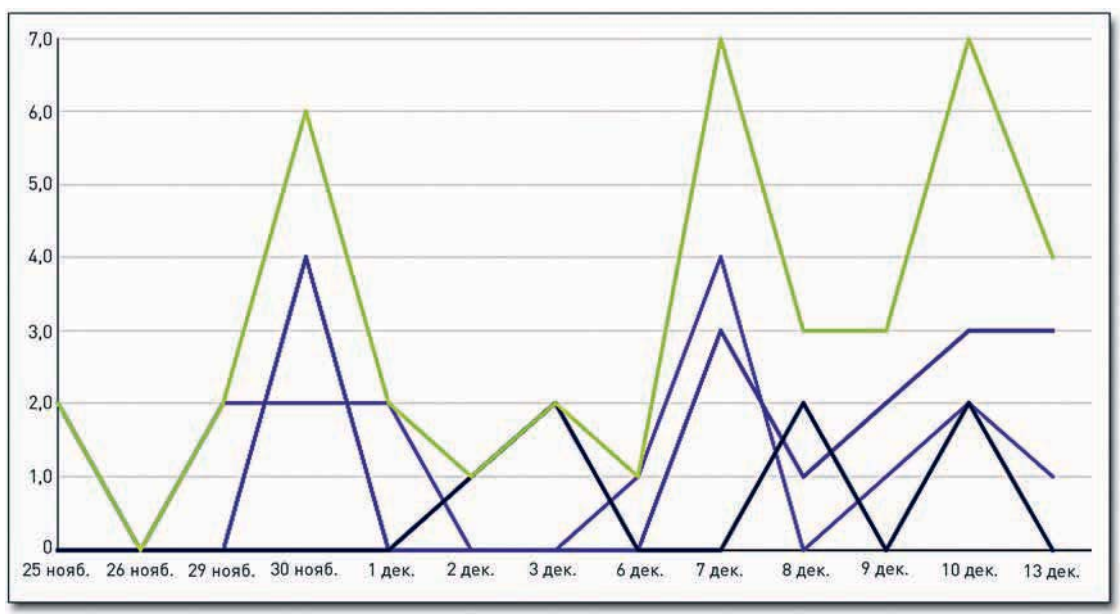


Рис. 3.7. Линейный график на Many Eyes

Но, помимо этого, Many Eyes предлагает целый набор различных, более продвинутых и экспериментальных типов визуализации, а также ряд базовых инструментов для составления карт. Дерево слов поможет вам исследовать текст, например, книги или статьи целиком. Выберите слово или фразу, и вы узнаете, как они применяются в тексте, увидите, что следует за ними. В качестве примера на рис. 3.9 представлены результаты поиска для слова «право» («right») в Конституции Соединенных Штатов.

* Many Eyes — множество глаз (*англ.*). *Прим. пер.*

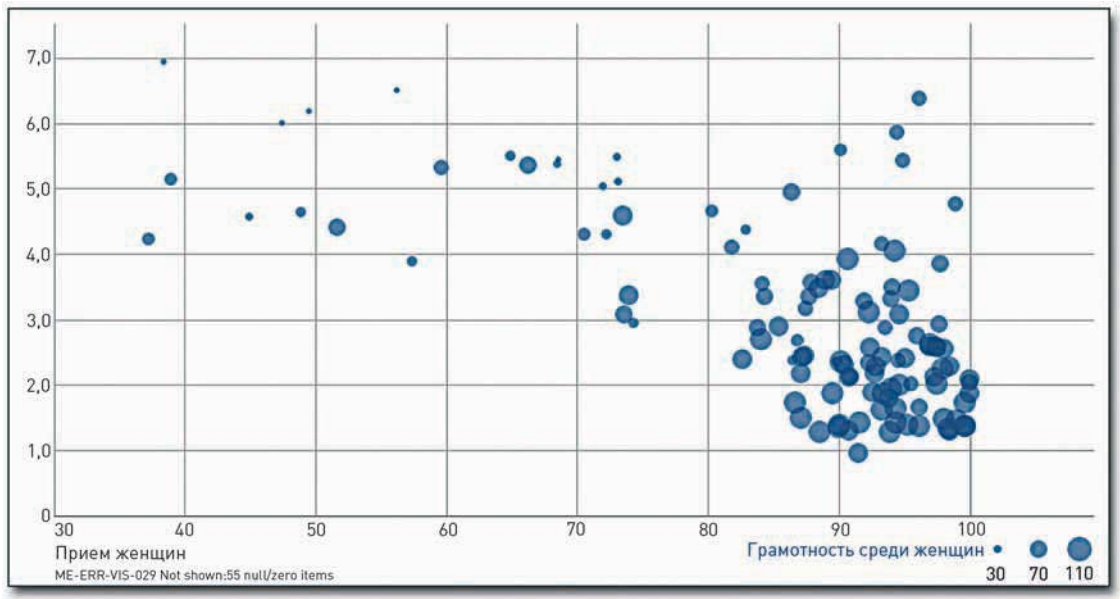


Рис. 3.8. Диаграмма рассеяния на Many Eyes

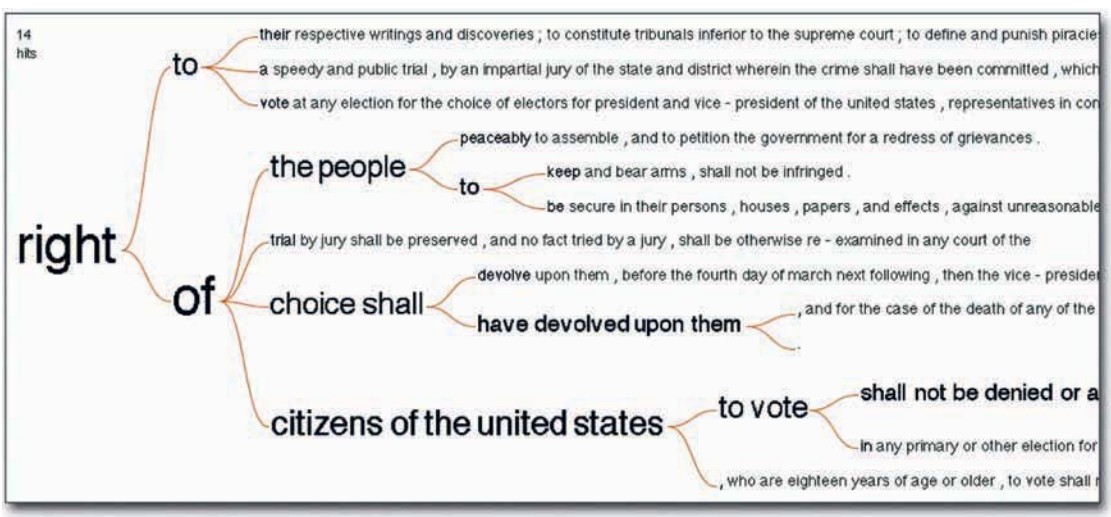


Рис. 3.9. Дерево слова «право» в Конституции США на Many Eyes

При этом вы легко можете переключаться с одного инструмента на другой, используя одни и те же данные. На рис. 3.10 Конституция визуализирована с помощью стилизованного облака слов, известного как Wordle. Чем чаще встречается слово, тем оно больше по размеру.

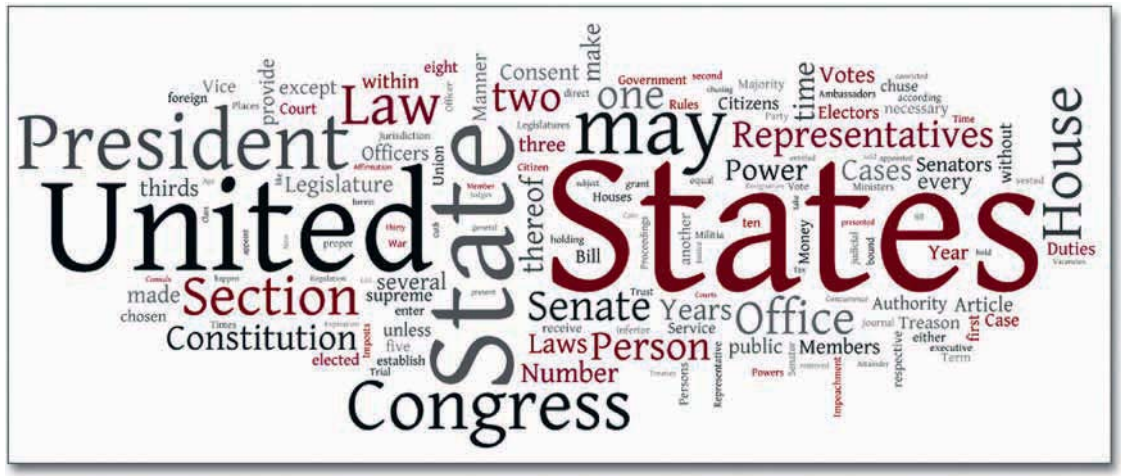


Рис. 3.10. Текстовое облако из слов Конституции США

Как вы видите, у Many Eyes есть много опций, которые помогут вам опробовать различные варианты визуализации ваших данных. В настоящий момент это пока что самый богатый по возможностям (и, на взгляд многих, лучший) бесплатный набор инструментов для исследования данных. Но тем не менее мне следует вас предостеречь. Во-первых, большинство инструментов являются Java-апплетами, а потому вы немногого добьетесь, если у вас не стоит Java. (Для большинства людей это не проблема, но я знаю нескольких человек, которые по разным причинам очень разборчиво относятся к тому, что следует, а что не следует инсталлировать в компьютер.)

▶ Попробуйте загрузить свои данные на `http://many-eyes.com` и визуализировать их.

Второе предостережение, которое может иметь для вас решающее значение, связано с тем, что все данные, которые вы загрузите на сайт, окажутся достоянием общественности. А потому вы не сможете пользоваться Many Eyes для того, например, чтобы покопаться в информации о клиентах или в данных о продажах, реализованных вашей компанией.

TABLEAU SOFTWARE

Tableau Software, программа, работающая только под Windows, относительно нова, но в последние пару лет быстро набирает популярность. Она создана главным образом для визуального исследования и анализа данных. Совершенно очевидно, что ее разработчики уделили особое внимание эстетике и дизайну. Потому-то программа и нравится многим.

Tableau Software предлагает обилие интерактивных инструментов визуализации, а еще очень хорошо умеет управлять данными. Их можно загружать как из Excel, так и из текстовых файлов и с серверов баз данных. В вашем расположении стандартные диаграммы временны́х рядов, гистограммы, круговые диаграммы, основные типы картографических способов визуализации

и т. д. Вы можете смешивать и сочетать способы отображения, можете залезть в источник динамических данных, чтобы составить собственное мнение о нем, или в инструментальную панель, чтобы получить представление о том, что происходит с вашими данными.

Относительно недавно компания Tableau выпустила Tableau Public — бесплатный инструмент, предлагающий в десктоп-редакциях определенное подмножество функциональных возможностей полной версии. Вы можете загрузить ваши данные на сервер Tableau, создать интерактивный объект и опубликовать его на вашем сайте или в блоге. Однако любые данные, которые вы загрузите на сервер, как и в случае с Many Eyes, станут общедоступными, так что помните об этом.

Если вы хотите использовать Tableau, сохраняя при этом свои данные закрытыми, тогда вам придется привыкнуть к десктоп-редакциям. На момент написания этой книги программные средства для настольных систем оставались несколько дороговатыми — их цены колебались в диапазоне от 999 до 1999 долларов США (для индивидуального и корпоративного пользования соответственно).

▶ Посетите сайт Tableau Software по адресу `http://tableausoftware.com`. Там вы сможете бесплатно опробовать полнофункциональную версию.

YOUR.FLOWINGDATA

Мой личный интерес к теме сбора персональных данных вдохновил меня на создание собственного сервиса: your.flowingdata (YFD). Это онлайн-приложение, дающее возможность собирать данные через Twitter, а затем исследовать существующие в них паттерны и взаимосвязи при помощи набора интерактивных инструментов визуализации. Некоторые люди отслеживают таким образом свой режим питания или сна. Другие регистрируют то, как их малыши овладевают новыми навыками, и создают своего рода детский альбом, только заполненный не фотографиями, а данными.

YFD изначально создавался с мыслью о личной информации, но многие находят это приложение полезным также для сбора данных более общего характера, таких как статистика сетевой активности или информация о прибытии и отправлении поездов.

▶ Опробуйте процесс сбора персональных данных через Twitter на сайте `http:// your.flowingdata.com`.

Компромиссы

Несмотря на то что все эти инструменты просты в применении, у них есть и некоторые недостатки. Чтобы не лишать себя удобной возможности работать методом перетаскивания (drag and drop), вам придется несколько ограничить себя в гибкости относительно того, что можно сделать с данными. Как правило, вам доступно изменение цветов, шрифтов и названий, но вы не можете делать того, чего в программе не предусмотрено. Если в ней нет кнопки для того типа диаграммы, которую вы хотите построить, то с этим вам придется смириться.

Бывает и наоборот: некоторые сервисы могут иметь огромное множество функций, но у них окажется и масса кнопок, которые вам придется изучить. Например, существовала одна программа (не упомянутая выше), по которой я как-то прошел ускоренный курс обучения, — было вполне очевидно, что она способна на многое, если вложить в ее освоение достаточно

времени. Однако процесс выполнения задач в ней оказался настолько далек от интуитивно понятного, что я в конце концов потерял желание изучать ее далее. Помимо прочего, было очень непросто повторять процедуру для различных наборов данных, так как приходилось запоминать все, по чему я щелкал мышью. Когда вы собственноручно создаете код для управления данными, потом бывает очень легко применить его повторно и подключить к обработке другого набора данных.

Не поймите меня неправильно. Я не говорю, что необходимо по возможности избегать готовых программ. Они способны помочь быстро и легко окинуть взглядом ваши данные. Но по мере того, как вы начнете работать со все бóльшими объемами информации, чаще станут возникать ситуации, в которых существующего программного обеспечения вам будет недостаточно. И когда наступит этот момент, вы сможете обратиться к программированию.

Программирование

Я никогда не устану повторять: стоит вам обрести хоть немножечко знаний в области программирования, и вы сможете делать со своими данными гораздо больше, чем если вы будете пользоваться лишь готовым ПО. Умение писать программы позволяет вам поступать более гибко и лучше адаптироваться к различным типам данных.

Если когда-нибудь вам доводилось видеть информационную графику, которая производила на вас особое впечатление своим нестандартным видом, скорее всего, она была создана с помощью специально разработанного кода или спроектирована в программе для работы с иллюстрациями. Или же, как это оказывается во многих случаях, имело место и то, и другое. Но о программах для работы с иллюстрациями мы поговорим позже.

Написание кода может показаться новичкам настоящим таинством. Я знаю это по себе. Проще думать о нем как о письме на новом для вас языке, потому что именно этим он по сути и является. Каждая строка кода говорит компьютеру, что именно нужно сделать. Ваш компьютер не понимает языка, с помощью которого вы общаетесь с вашими друзьями, а потому вам следует говорить с ним на его языке со специфическим синтаксисом.

Как это бывает с любым иностранным языком, вы не сможете сходу завести непринужденную беседу. Начните с самых азов, а затем потихоньку продвигайтесь вперед. Еще до того, как вы выучите язык, вы уже начнете писать коды. Прелесть программирования заключается в том, что стоит вам освоить всего один язык, и дальше будет уже намного легче изучить остальные, так как логика везде схожая.

Опции

Итак, вы решили замарать руки программированием. Поздравляю! Перед вами открывается множество возможностей. Некоторые языки лучше подходят для выполнения одних задач, другие — для других. Какие-то способны управлять огромными массивами данных, в то время

как другие не так надежны в этом деле, но позволяют создавать гораздо более привлекательные визуальные объекты или придавать им интерактивность. На каком языке будет лучше остановиться, зависит от того, какие цели вы ставите перед конкретной диаграммой и в работе с каким из языков вы чувствуете себя более уверенно.

Некоторые люди придерживаются одного определенного языка и со временем осваивают его очень хорошо. Это замечательно также и в том случае, если вы новичок в области программирования. Я горячо рекомендую вам придерживаться данной стратегии. Сперва ознакомьтесь с основополагающими и важными принципами кода.

Используйте тот язык, который лучше всего подходит для ваших нужд. Однако есть особое удовольствие в том, чтобы изучать новые языки и осваивать новые приемы в игре с данными. Так что вам следует накопить приличный опыт в программировании, прежде чем решить, что вам больше нравится.

PYTHON

В предыдущей главе мы говорили о том, как управлять данными с помощью языка Python. В данном деле Python зарекомендовал себя с лучшей стороны — он способен без сбоев управлять большими массивами данных. Это делает настоящий язык особенно полезным для анализа и для мощных вычислений.

Помимо прочего, у Python опрятный и удобочитаемый синтаксис, который нравится программистам. С ним вы сможете отработать множество модулей, чтобы создать информационную графику, подобную представленной на рис. 3.11.

С точки зрения эстетики, данный объект не представляет собой чего-то выдающегося. Вы вряд ли захотите отдать только что полученный с помощью Python график в печать. Результат обычно выглядит неаккуратным. Тем не менее это хорошее начало на пути изучения данных. А еще вы можете экспортировать изображение, а затем дообработать его или добавить в него информацию, используя графический редактор.

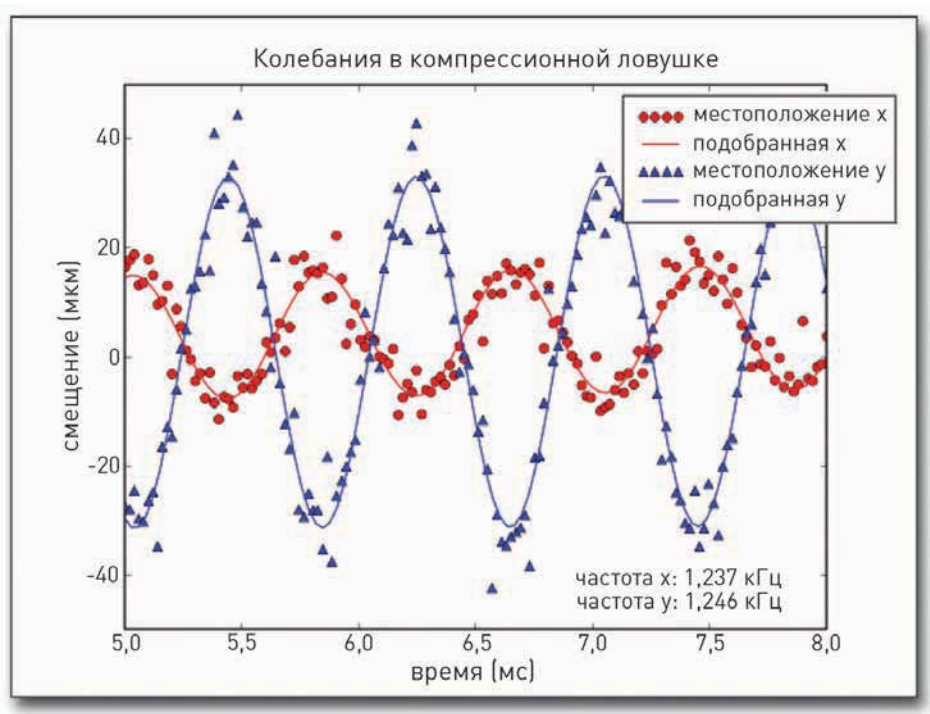


Рис. 3.11. График, созданный в Python

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С PYTHON:

- официальный сайт Python (`http://python.org`);
- NumPy и SciPy (`http://numpy.scipy.org`) — научные инструменты для решения вычислительных задач.

PHP

PHP был первым языком, который я выучил, когда начал заниматься созданием программ для Сети. Некоторые люди утверждают, что он неупорядоченный. Возможно, их слова недалеки от истины. Но PHP так же несложно и держать в порядке. Его, как правило, легко установить, на большинстве веб-серверов он уже инсталлирован, а потому можно сразу же браться за него.

Рис. 3.12. Искрографики, созданные с применением графической библиотеки PHP

Существует довольно гибкая графическая библиотека для PHP под названием CD, которая обычно включена в стандартный установочный пакет. Библиотека дает вам возможность создавать изображения с нуля и подгонять уже готовые. Помимо нее есть еще множество других PHP-библиотек для построения диаграмм, дающих возможность создавать все основные виды графических объектов. Самая популярная из них — это библиотека для создания искрографиков Sparkline Graphing Library, позволяющая вставлять маленькие, размером со слово, графики в текст или добавлять визуальные компоненты в числовые таблицы, как показано на рис. 3.12.

В большинстве случаев PHP связан с некоей базой данных, например с MySQL. Таким образом, вместо того чтобы работать с множеством CSV-файлов, вы можете иметь дело с крупными массивами данных.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С PHP:

- официальный сайт PHP (`http://php.net`);
- Sparkline PHP Graphing Library (`http://sparkline.org`).

PROCESSING

Processing — это язык программирования с открытым исходным кодом, предназначенный для дизайнеров и художников, работающих в области data-арта. Поначалу он задумывался как записная книжка с кодами, позволяющая быстро создавать графические объекты. Однако

с тех времен Processing претерпел значительное развитие, и сегодня на нем разрабатываются многие высококачественные проекты. Например, упомянутое в первой главе произведение «Все у нас хорошо» было создано именно на этом языке.

У языка Processing есть явное достоинство: освоиться в нем можно довольно быстро. Среда программирования очень простая, и вы можете создавать динамическую и интерактивную графику с помощью всего нескольких строк кода. Эта графика, конечно, будет самого базового уровня, но, поскольку Processing разрабатывался с мыслью о создании визуальных объектов, вы легко научитесь творить и более совершенные произведения.

Хотя аудитория у Processing поначалу состояла преимущественно из дизайнеров и художников, сегодня данное сообщество представляет собой довольно разнородную группу людей. Наличие множества библиотек поможет вам добиться большего.

Однако у Processing есть и недостатки, и один из них заключается в том, что в качестве результата работы вы получаете Java-апплет, который на компьютерах у части пользователей будет загружаться относительно медленно, кроме того, не у всех стоит Java (хотя у большинства все-таки есть). Впрочем, для этой проблемы существует решение. Недавно была разработана JavaScript-версия Processing, и она готова к употреблению.

Во всех случаях новичкам есть с чего начать. Даже те, у кого нет никакого опыта в программировании, способны сделать что-то полезное.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С PROCESSING:

- официальный сайт Processing (`http://processing.org`).

FLASH И ACTIONSCRIPT

В большинстве случаев интерактивная и динамическая информационная графика в Сети, особенно на крупнейших новостных сайтах, таких как сайт New York Times, разрабатывается с использованием Flash и ActionScript. Графику можно создавать и только во Flash, тем более что интерфейс программы работает по принципу перетаскивания (drag and drop), однако ActionScript позволяет лучше контролировать процесс взаимодействия пользователей с интерактивными объектами. Многие приложения написаны целиком на ActionScript без использования среды Flash. Тем не менее код компилируется как Flash-приложение.

Например, интерактивная карта роста торговой сети Walmart, которая показана на рис. 3.13, была разработана на ActionScript. Для этого использовалась Modest Maps — интерактивная библиотека для создания карт на основе тайлов. Она распространяется на условиях лицензии BSD, то есть бесплатна, и вы можете использовать ее для чего хотите.

ПРИМЕЧАНИЕ

Хотя существует множество ActionScript-библиотек с открытым исходным кодом, Flash и Flash Builder стоят недешево, что следует учитывать при выборе программного обеспечения.

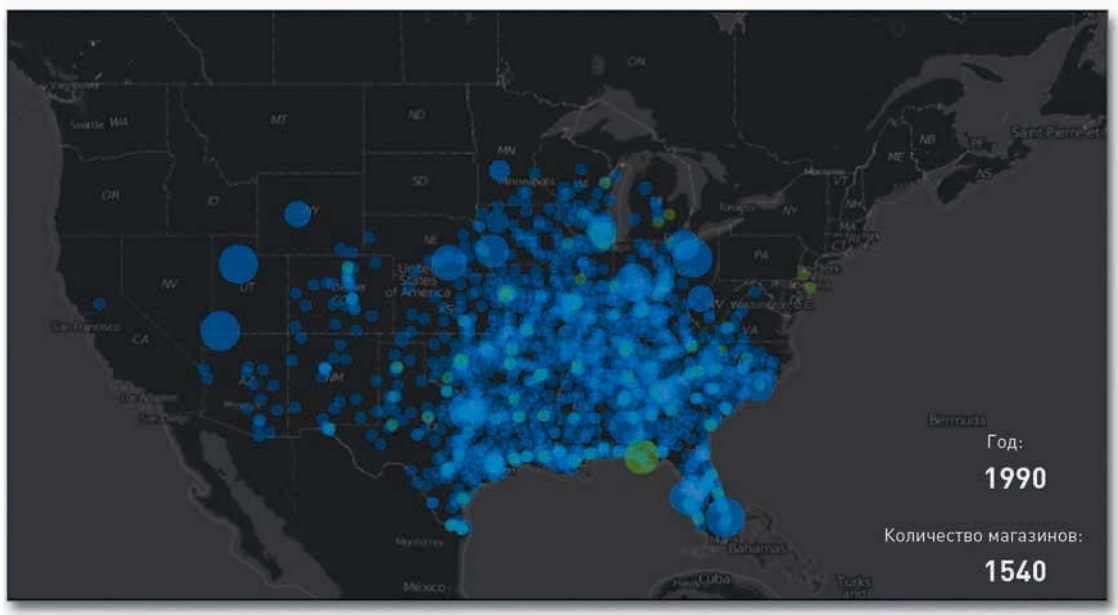


Рис. 3.13. Анимированная карта роста Walmart, созданная на ActionScript

Интерактивная штабельная диаграмма с областями, представленная на рис. 3.14, также была написана на ActionScript. Она дает возможность отслеживать различные категории расходов по годам. Для выполнения большей части тяжелой работы использовалась библиотека Flare ActionScript, созданная в Лаборатории по визуализации при Калифорнийском университете в Беркли.

Если вы хотите заниматься созданием интерактивной графики для Сети, тогда Flash и ActionScript — отличный выбор. Flash-приложения загружаются относительно быстро, и у большинства людей на компьютерах Flash Player уже установлен.

Это не самый простой для изучения язык. Синтаксис у него не такой уж сложный, но структура и организация кода способны запутать новичков. В отличие от Processing, при работе с Flash у вас не получится запустить приложение с помощью всего нескольких строк кода. Следующие главы книги проведут вас по основным этапам создания Flash-приложений, кроме того, в Сети вы можете найти множество полезных обучающих онлайн-руководств, так как у Flash действительно очень широкий спектр применения.

А поскольку веб-браузеры постоянно совершенствуются в плане скорости и эффективности, перед вами раскрываются все бóльшие возможности.

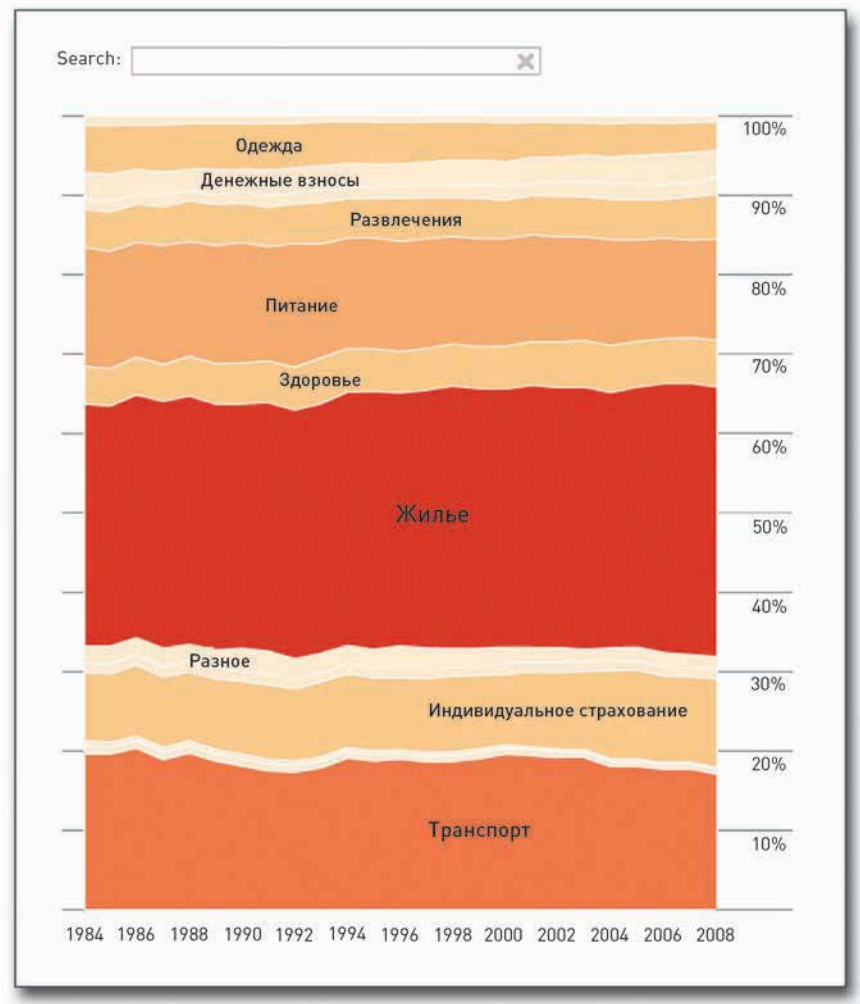


Рис. 3.14. Интерактивная штабельная диаграмма с областями показывает распределение расходов потребителей (создана с помощью ActionScript)

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С FLASH И ACTIONSCRIPT:

- служба поддержки Adobe Support (`http://www.adobe.com/products/flash/whatisflash`) — официальная документация Flash и ActionScript (а также других продуктов Adobe);
- набор инструментальных средств для визуализации Flare (`http://flare.prefuse.org`);
- Modest Maps (`http://modestmaps.com`).

HTML, JAVASCRIPT И CSS

Веб-браузеры становятся все быстрее, их функционал — все богаче и разнообразнее. У многих людей сегодня браузер — самое используемое приложение на компьютере. В последнее время произошел заметный сдвиг в сторону визуализации, которая легко запускается без «посторонней помощи» в браузере через HTML, JavaScript и CSS. Раньше информационная графика, если в ней присутствовал элемент интерактивности, создавалась преимущественно на Flash и ActionScript — или же сохранялась как статичное изображение. И по сей день почти все делается именно так, но прежде просто не было других вариантов, кроме этих двух, а теперь есть.

Существуют несколько надежных в работе пакетов и библиотек, которые помогают быстро создавать интерактивные и статичные визуальные объекты. Они также предоставляют множество возможностей для подстройки инструментов под конкретные потребности при работе с данными.

Например, Protovis — бесплатная библиотека с открытым исходным кодом, поддерживаемая Stanford Visualization Group, — дает возможность создавать визуальные объекты, специально предназначенные для Сети. Protovis предлагает множество готовых решений, но вы отнюдь не ограничены в том, что и как делать, с точки зрения геометрии. На рис. 3.15 представлена штабельная диаграмма с областями, которая может быть интерактивной.

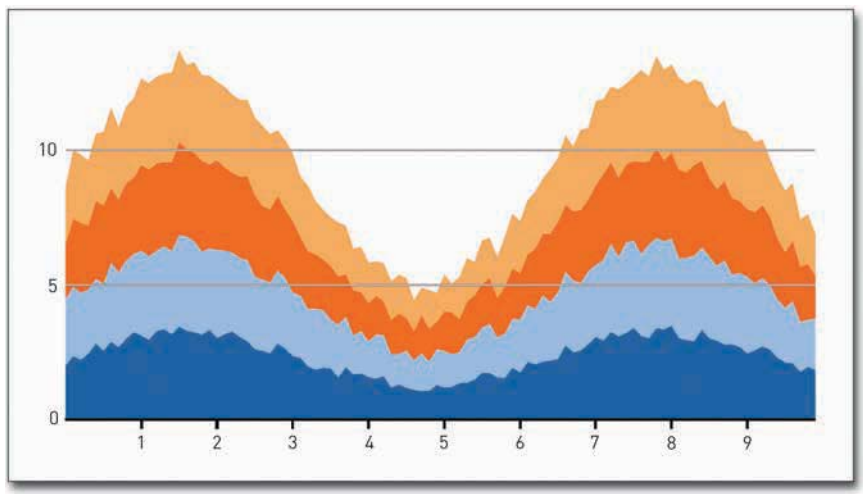


Рис. 3.15. Штабельная диаграмма с областями, созданная с помощью Protovis

Данный тип диаграмм встроен в Protovis, но вы можете создать и менее традиционную диаграмму с областями в виде «потокового графика» (streamgraph*), такую, как показана на рис. 3.16.

---

* Потоковый график (streamgraph) — разновидность штабельной диаграммы с областями, выстроенной со смещением вокруг центральной оси, в результате чего возникают красивые плавные формы (особенно при работе с большими массивами данных). Разработан Ли Байроном в 2008 году. Генератор можно скачать с GitHub: https://github.com/leebyron/streamgraph_generator. *Прим. пер.*

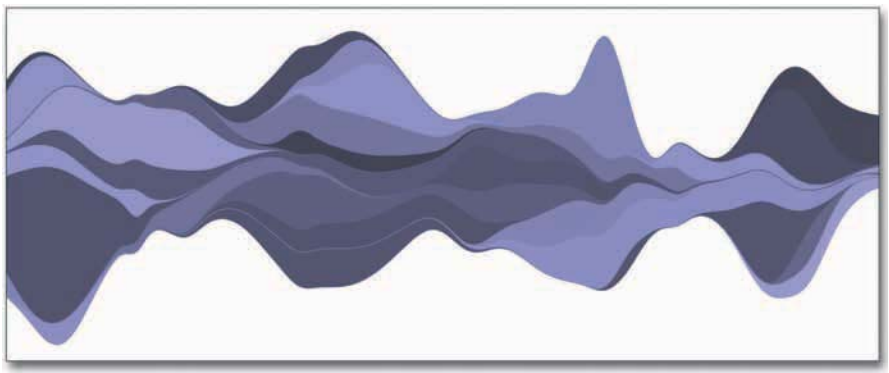


Рис. 3.16. Потоковый график, созданный по индивидуальному заказу с помощью Protovis

Для обогащения функционала вы можете пользоваться несколькими библиотеками. Во Flash это возможно, JavaScript дает меньше пространства для таких экспериментов. Зато JavaScript намного проще читать и использовать с такими библиотеками, как jQuery и MooTools. Они не специализированы в области визуализации, но могут быть полезными, так как предоставляют множество базовых функциональных возможностей, и с ними вам понадобится всего несколько строчек собственного кода. Без библиотек вам придется писать намного больше, а написанный в спешке код бывает довольно запутанным.

Плагины для библиотек также могут пригодиться вам при создании некоторых основных видов диаграмм и графиков. Например, вы можете использовать плагин Sparkline для jQuery, чтобы делать такие маленькие диаграммы, как представленная на рис. 3.17.

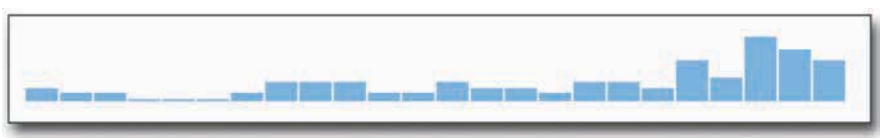


Рис. 3.17. Искрографики, созданные с применением плагина jQuery Sparklines

Сделать это можно и с PHP, но у данного метода есть несколько преимуществ. Во-первых, графика генерируется в браузере пользователя, а не на сервере. Это снимает нагрузку с ваших машин, что может быть важным, если у вашего сайта большой трафик.

Другое преимущество состоит в том, что вам не нужно устанавливать на свой сервер графическую библиотеку PHP. Многие сервера оснащены ею, но бывает, что ее нет. Установка может вызвать у вас затруднения, если вы не знакомы с системой.

Вы можете и вовсе не пользоваться плагинами. Можете разрабатывать визуальные объекты по индивидуальному заказу стандартными методами веб-программирования. Для примера на рис. 3.18 представлен интерактивный календарь, который способен также работать в качестве карты интенсивности кликов на сайте your.flowingdata.

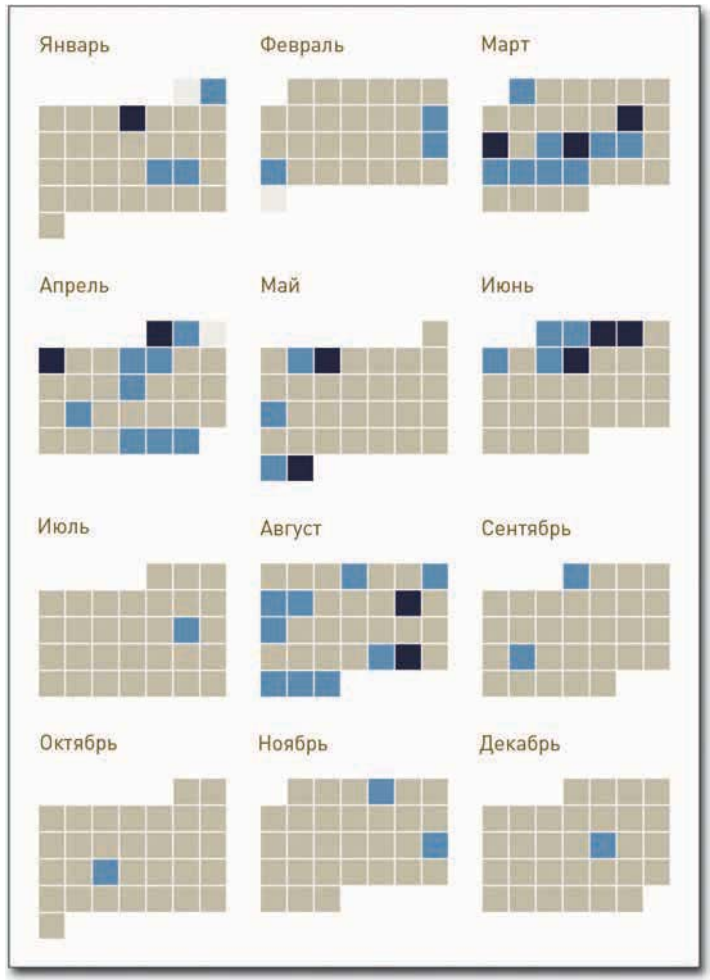


Рис. 3.18. Интерактивный календарь на your.flowingdata, который может работать также и как карта интенсивности кликов

Есть, однако, и некоторые недостатки. Поскольку программы и технология в целом относительно новы, в различных браузерах ваша графика может отображаться по-разному. Некоторые из упомянутых выше инструментов не будут работать корректно в старых браузерах, таких как Internet Explorer 6. Однако эта проблема постепенно решается сама собой, так как все больше людей начинают пользоваться современными браузерами вроде Firefox или Google Chrome. В конечном счете все зависит от вашей аудитории. Скажем, только 5% посетителей FlowingData пользуются старыми версиями Internet Explorer, и несовместимость уже не кажется большой проблемой.

Однако есть также еще один недостаток, связанный с возрастом технологии: количество доступных библиотек для визуализации на JavaScript не так велико, как для Flash и ActionScript. Вот почему многие крупнейшие СМИ все еще активно используют Flash. Впрочем, с развитием технологий это изменится.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С HTML, JAVASCRIPT И CSS:

- jQuery (`http://jquery.com`) — JavaScript-библиотека, которая делает написание кодов на этом языке гораздо более эффективным, а конечный продукт — более удобочитаемым;
- jQuery Sparklines (`http://omnipotent.net/jquery.sparkline`) — плагин для создания статических и динамических искрографиков на JavaScript;
- Protovis (`http://vis.stanford.edu/protovis`) — ориентированная на визуализацию JavaScript-библиотека, специально разработанная так, чтобы можно было учиться на примерах;
- JavaScript InfoVis Toolkit (`http://datafl.ws/15f`) — еще одна библиотека, чье назначение — совершенствовать процесс визуализации, хотя она и не такая богатая, как Protovis;
- Google Charts API (`http://code.google.com/apis/chart`) — позволяет создавать традиционные карты на лету, просто меняя URL.

R

Если вы читаете FlowingData, вы, наверное, знаете, что R — мой излюбленный инструмент для создания информационной графики. Это свободная программная среда для статистических вычислений с открытым исходным кодом и очень хорошим графическим функционалом. Большинство статистиков предпочитают работать именно с R. Существуют также и платные альтернативы, такие как S-plus и SAS, но им тяжело тягаться с бесплатным вариантом и с активным сообществом его сторонников.

Одно из преимуществ R перед упомянутым выше программным обеспечением состоит в том, что он специально создавался для анализа данных. HTML разрабатывался для создания веб-страниц, а Flash используется для тысячи разных целей, таких как создание видео и анимированных рекламных роликов. R же создавался (и ныне поддерживается) статистиками и для статистиков — это и хорошо, и плохо, в зависимости от того, под каким углом посмотреть.

Существует множество R-пакетов, которые дают возможность создавать информационную графику с помощью всего нескольких строк кода. Загрузите ваши данные в R, и вы получите график или диаграмму, написав всего одну строчку. Например, используя пакет Portfolio, вы можете очень быстро сделать тримап (рис. 3.19).

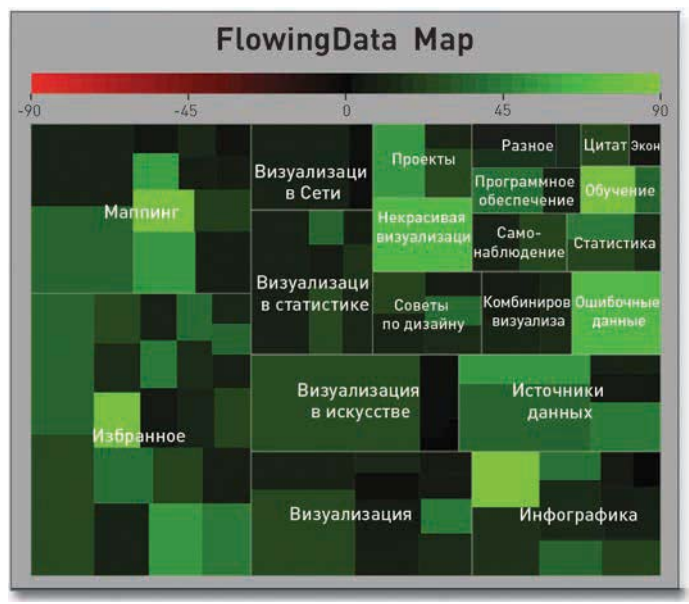


Рис. 3.19. Тримап, созданный в R с пакетом Portfolio

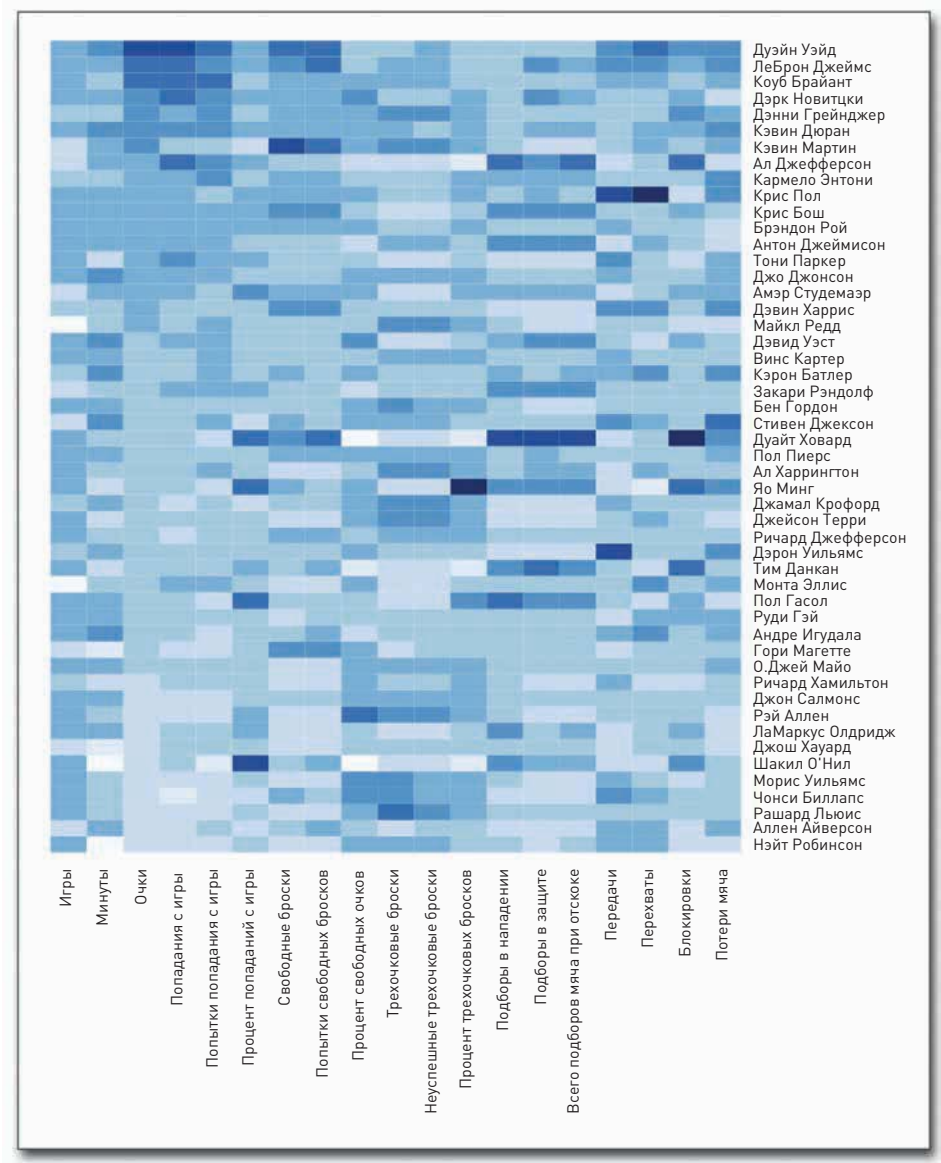


Рис. 3.20. Карта интенсивности, созданная в R

С той же легкостью можно построить и карту интенсивности (рис. 3.20).

Конечно, вы можете создавать и более традиционные виды графики, такие как точечные диаграммы и диаграммы временны́х рядов, о которых мы подробнее поговорим в четвертой главе, «Визуализация паттернов во времени».

А вот сайт R, если быть до конца откровенным, смотрится ужасно несовременно (рис. 3.21), да и сама программа не сильно помогает свежеиспеченным пользователям осваивать новую среду. Однако вам необходимо помнить, что R — это язык программирования, и подобная ситуация у вас будет возникать с любым языком, который вы решите использовать. То немногое плохое, что я читал об R, как правило, написано людьми, привыкшими работать с кнопками и методом перетаскивания. Так что когда вы доберетесь до R, не думайте, что увидите интерфейс, где все можно делать мышью, — в противном случае этот интерфейс действительно покажется вам «недружественным».

Однако если вы примете эти особенности, перед вами раскроются большие возможности. Вы сможете делать графику типографского качества (или по крайней мере на таком уровне, с которого это качество начинается) и полюбите гибкость R. Если вы захотите, то сможете написать свои собственные функции и пакеты, чтобы создавать такую графику, какая нужна именно вам. А можете пользоваться теми, которые другие люди сделали доступными в библиотеке R.

R предоставляет все основные функции рисования, которые, по сути, позволяют создавать практически любые объекты, которые могут вам понадобиться. Во фреймворке рисования вы

можете вычерчивать линии, формы и оси, и, опять-таки, как в случае и с другими программными решениями, вы будете ограничены только рамками своего воображения. С помощью разных пакетов R доступны практически все виды диаграмм.

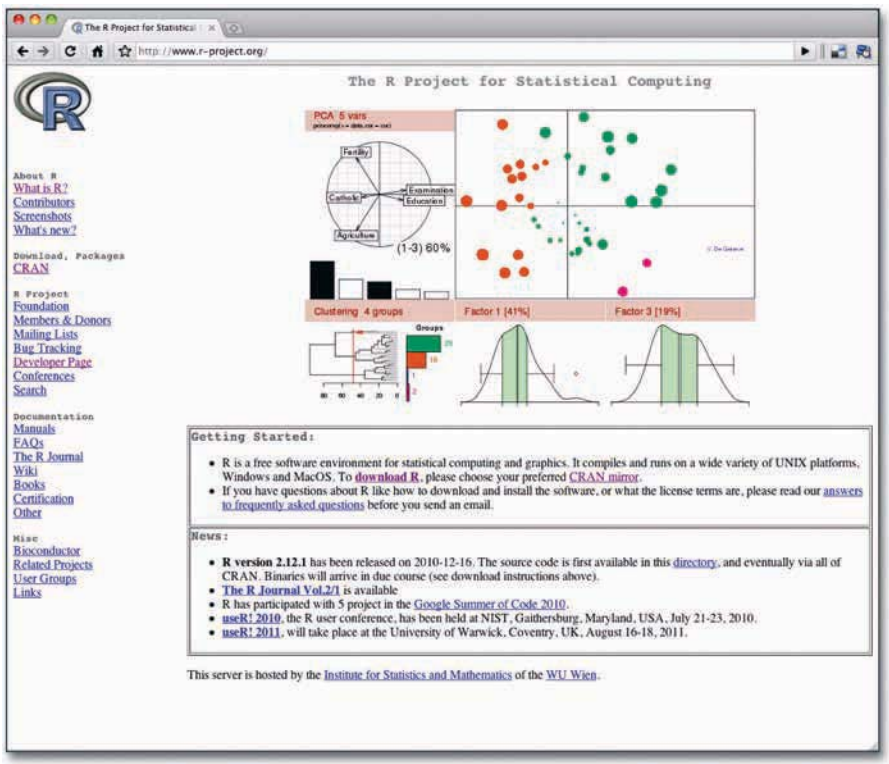


Рис. 3.21. Главная страница сайта R: http://www.r-project.org

ПОДСКАЗКА

Когда вы будете искать что-нибудь, связанное с R, в интернете, поисковые системы могут иногда не учитывать такое короткое название и выдавать сообщение об ошибке или неправильный результат. Поэтому попробуйте указать в своем запросе «r-project», а не просто «R». Результаты поиска должны оказаться более релевантными.

Тогда зачем пользоваться еще чем-нибудь, помимо R? Почему бы не делать с его помощью все? Вот вам несколько причин. R работает на вашем десктопе, так что он не подходит для динамических веб-страниц. Сохранить и разместить диаграммы и изображения на интернет-странице — не проблема, но все это делается отнюдь не автоматически. Вы можете на лету создавать графику прямо в Сети, но на данном этапе существующие для этого решения не слишком устойчивы, особенно если сравнивать их с такими специально разработанными для сети продуктами, как JavaScript.

R также мало подходит для интерактивной графики и анимации. Конечно, вы можете делать с R и это, но существуют более гибкие, более элегантные способы выполнить подобную задачу, например с помощью Flash или Processing.

И наконец, вы, возможно, заметили, что графическим объектам, представленным на рис. 3.19 и 3.20, чуточку не хватает лоска. Вы вряд ли когда-нибудь обнаружите подобную графику

в газетах или журналах. Вы можете подтянуть дизайн в R до определенного уровня, подключив разные опции или написав дополнительный код, но я сам обычно применяю другую стратегию: я делаю основу графического объекта в R, а затем редактирую его и совершенствую в каком-нибудь приложении для подготовки макетов документов, таком как Adobe Illustrator, — об этом мы еще поговорим далее. Для анализа «сырой» продукт R подходит отлично, но для презентаций и сторителлинга лучше будет немного поработать над эстетикой.

ПОЛЕЗНЫЙ РЕСУРС:

- проект для статистических расчетов R (`http://www.r-project.org`).

Компромиссы

Изучать новую программу означает изучать новый язык. Язык, на котором говорит ваш компьютер, — это язык, состоящий из битов и обладающий собственной логикой. Когда вы работаете, например, с Excel или Tableau, вы, по сути, работаете с переводчиком. Интерфейс говорит с вами на вашем языке, и когда вы щелкаете мышью по кнопке, программа переводит команду и затем посылает перевод компьютеру. После чего компьютер выполняет ее и делает для вас что-то, скажем, создает график или обрабатывает какие-то данные.

И здесь время определенно становится серьезным препятствием. Время требуется на изучение нового языка. Для многих людей такое препятствие оказывается непосильным, и я могу их понять. Вам необходимо выполнить работу сейчас, потому что перед вами целая куча данных и люди ждут не дождутся результатов. Если у вас дела обстоят именно так — у вас есть одна-единственная задача, связанная с обработкой данных, и больше таких задач в будущем не предвидится, — тогда, возможно, действительно лучше ограничиться готовыми инструментами для визуализации.

Однако если вы хотите разобраться в ваших данных и вам с большой долей вероятности придется и впредь работать над разными проектами, связанными с обработкой данных, — тогда время, потраченное на изучение программирования сегодня, может завтра обернуться экономией времени для других проектов, которые к тому же будут иметь более впечатляющие результаты. С каждым новым проектом ваши умения в программировании станут совершенствоваться, и оно будет даваться вам все легче и легче. Как и с любым иностранным языком, вы не сразу начинаете писать на нем романы. Нет, вы начнете с азов, а затем постепенно расширите свои знания.

Можно взглянуть на все это и по-другому. Представьте себе, что вас забросили в чужую страну, а вы не говорите на тамошнем языке. Однако у вас есть переводчик. (Выслушайте меня до конца, я говорю по делу.) Чтобы общаться с местными, вам нужно сперва озвучить свою мысль, а затем переводчик должен донести ваше послание. А что делать, если переводчику

неизвестно значение только что произнесенного вами слова или он не знает, какое именно слово употребить, чтобы передать сказанное вами? Он может это слово просто опустить или, если он достаточно сообразительный, заглянуть в словарь.

Программа для готовых визуальных решений является тем самым переводчиком. Если она не знает, как делается что-то, то вы оказываетесь в тупике или вам придется попробовать пойти другим путем. В отличие от человека-переводчика, программа неспособна на ходу усваивать новые слова либо, как в нашем случае, новые типы диаграмм и графиков или новые средства обработки данных. Дополнительные функции поступают к ней в виде обновления программы, появления которого приходится ждать. Так почему бы вам самому не выучить язык?

И опять-таки я не призываю вас избегать готовых инструментов. Я сам постоянно ими пользуюсь. Они делают множество нудных задач легкими и быстро разрешимыми, и это здорово. Просто не позволяйте программному обеспечению ограничивать вас.

Как вы убедитесь, читая следующие главы, программирование способно помочь вам проделать больше работы за меньшее время и с меньшими усилиями, чем если вы будете делать все вручную. Конечно, есть и такие задачи, которые лучше делать вручную, особенно когда вы рассказываете истории с помощью данных. И это подводит нас к следующему пункту, который лежит на противоположном конце спектра визуализации: к иллюстрированию.

Иллюстрирование

Давайте заглянем в вотчину графических дизайнеров. Если вы аналитик или у вас более техническая специальность, эта территория для вас, вероятно, незнакомая. Сочетая коды и готовые инструменты визуализации, можно добиться очень многого, но графический объект, который вы в итоге получите, почти всегда будет выглядеть немного грубовато — как нечто, сгенерированное автоматически. Возможно, подписи окажутся не совсем в том месте, или легенда будет немного перегруженной. Для анализа такой результат, как правило, вполне годится — вы же знаете, на что смотрите.

Но когда вы делаете диаграмму или график для презентации, для отчета или для публикации, как правило, вы должны навести лоск, чтобы люди могли четко понять, какую историю вы им рассказываете.

Например, на рис. 3.19 представлен «сырой» результат работы в R. Он показывает количество просмотров и комментариев на сайте FlowingData для 100 самых популярных постов. Посты рассортированы по категориям. Чем ярче зеленый цвет, тем больше комментариев вызвал конкретный пост, а чем больше размер прямоугольника, тем больше было просмотров. По первому варианту тримапа вы бы об этом не догадались, но когда я смотрел на числа, я знал, что именно я вижу, так как я собственноручно писал этот код.

На рис. 3.22 представлен переработанный вариант того же тримапа. Названия размещены так, чтобы их было видно; наверху я добавил вводный текст, чтобы читатели понимали, что это такое перед ними; красный участок цветовой легенды я удалил вовсе, так как это нонсенс — пост

с отрицательным количеством комментариев. А еще я изменил фон с серого на белый просто потому, что мне показалось — так будет лучше.

Рис. 3.22. Тримап, созданный в R и отредактированный в Adobe Illustrator

Я мог бы отредактировать код таким образом, чтобы он отвечал всем моим потребностям, но было намного проще щелкнуть мышью по объекту и перетащить его в Adobe Illustrator. Вы можете с нуля создать график или диаграмму в программе для работы с иллюстрациями, а можете импортировать в нее графический объект, который вы разработали, скажем, в R, и отредактировать его так, как вам хочется. В первом случае вы ограничены в выборе вариантов, так как визуализация не является первоочередной задачей данной категории программного обеспечения. Для всего, что сложнее столбцовой диаграммы, вам лучше будет прибегнуть к импорту. В противном случае вам придется очень многое делать вручную, а это чревато ошибками.

Что хорошо в использовании программ — графических редакторов, так это то, что вы лучше контролируете отдельные элементы и все можете делать методом перетаскивания (drag and drop). Поменять цвет столбцов или одного-единственного столбца, увеличить или уменьшить толщину осевых линий, снабдить примечаниями самые важные характеристики — и все это несколькими щелчками мыши.

Опции

Существует много программ для работы с иллюстрациями, но среди них таких, которые бы использовало большинство людей, всего несколько, а используемая повсеместно — и вовсе одна. Скорее всего, решающим фактором для вас будет цена. Цены варьируются от нуля (бесплатные программы с открытым исходным кодом) до нескольких сотен долларов.

ADOBE ILLUSTRATOR

Любая графика, построенная на статистических данных, которая выглядит как сделанная на заказ или появляется в том или ином крупном издании, скорее всего, на одном из этапов прошла обработку в Adobe Illustrator. Эта программа — отраслевой стандарт. Каждый графический объект, который печатается на страницах New York Times, был или создан, или отредактирован в Illustrator.

Популярность Illustrator в типографском деле объясняется тем, что в данной программе вы работаете с векторами, а не с пикселами. Это означает, что вы можете создавать большие графические объекты без снижения качества изображения. Пример обратного — это когда вам приходится увеличивать фотографию с низким разрешением, и в итоге вы получаете изображение, разбитое на цветные квадратики-пикселы.

Данная программа изначально была разработана для конструирования шрифтов, но позже обрела популярность у дизайнеров-иллюстраторов как средство для создания логотипов и арт-графики. И именно для этого Adobe Illustrator главным образом используется и по сей день.

Тем не менее программа предоставляет доступ к некоторым основным функциональным возможностям визуализации данных через инструмент Graph (Диаграмма). С его помощью вы можете создавать практически все основные типы диаграмм и графиков, такие как гистограммы, круговые диаграммы и диаграммы временны́х рядов. Вам нужно ввести числа в маленькую таблицу, однако этим и исчерпываются возможности управления данными.

Лучшее, что есть в Illustrator в плане информационной графики, — это гибкость и легкость работы при наличии большого числа кнопок и функциональных возможностей. Поначалу их обилие может несколько смутить вас, но в них можно быстро освоиться, и вы в этом сами убедитесь, прочитав четвертую главу («Визуализация паттернов во времени»). Именно эта гибкость позволяет лучшим дизайнерам информационной графики создавать лаконичные и понятные объекты.

Illustrator существует в варианте для Windows и для Mac. Однако у данной программы есть и один недостаток: она недешевая. Особенно высокой цена начинает казаться тогда, когда

подумаешь, сколько всего можно сделать с помощью кода, который вообще бесплатен (при условии, что у вас уже есть машина, на которую его можно загрузить). Однако если сравнивать по цене данную программу с другими готовыми решениями, Illustrator не покажется таким уж дорогим.

На момент написания книги самая свежая версия Illustrator стоила на сайте Adobe 599 долларов США, в других местах можно было получить значительные скидки (или взять себе более старую версию). Кроме того, Adobe предлагает существенные скидки студентам и другим членам научного сообщества, так что, возможно, программа достанется вам существенно дешевле. (Это самая дорогая программа, которую я когда-либо покупал, но я использую ее почти ежедневно.)

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С PHP ADOBE ILLUSTRATOR:

- страница Illustrator на вкладке «Продукты» сайта Adobe (`http://www.adobe.com/products/illustrator`);
- VectorTuts (`http://vectortuts.com`) — простые и подробные обучающие руководства по программе Illustrator.

INKSCAPE

Inkscape — это бесплатная (с открытым исходным кодом) альтернатива Adobe Illustrator. Если вы хотите обойтись без затрат, тогда Inkscape — ваш лучший выбор. Я всегда пользуюсь Illustrator, потому что когда я только начинал осваивать тонкости информационной графики, все пользовались именно им, и это показалось мне самым разумным вариантом действий. Но я слышал хорошие отзывы об Inkscape, и, поскольку программа бесплатная, не будет вреда, если попробуете ее. Только не надейтесь найти такое же, как для Illustrator, количество учебных веб-ресурсов на эту тему.

ПОДСКАЗКА

Часть графических объектов в этой книге редактировалась в Adobe Illustrator, но вряд ли вам было бы сложно сообразить, как сделать то же самое в Inkscape. Многие инструменты и функции двух программ имеют схожие названия.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С INKSCAPE:

- сайт Inkscape (`http://inkscape.org`);
- обучающие руководства по Inkscape (`http://inkscapetutorials.wordpress.com`).

ДРУГИЕ

Illustrator и Inkscape, безусловно, не единственные программы, с помощью которых можно создавать и доводить до блеска свои диаграммы и графики. Просто ими пользуется большинство

людей. Но есть и специалисты, которые предпочитают Corel Draw. Данная программа существует только в варианте для Windows и стоит примерно столько же, сколько и Illustrator. Можно найти ее и чуть подешевле, если знать, где искать.

Существуют и другие программы, такие как Raven от Aviary и Lineform, но они предлагают меньший набор инструментов. Помните, что Illustrator и Inkscape — это основные инструменты графических дизайнеров, и они обладают наиболее богатым функционалом. Но если вы хотите всего лишь чуть подкорректировать парочку существующих диаграмм, тогда можете выбрать и более простое (дешевое) программное обеспечение.

Компромиссы

Программы типа Illustrator и Inkscape предназначены лишь для одного: для иллюстрирования. Они не созданы специально для разработки информационной графики. Их основная задача — графический дизайн, а потому многие люди не пользуются всеми функциональными возможностями, предлагаемыми Illustrator и Inkscape. Обе они также не очень хорошо подходят для управления большими массивами данных и не выдерживают сравнения ни с программами, которые вы сами пишете для конкретных целей, ни с другими инструментами, которые специально созданы для визуализации данных.

Иными словами, графические редакторы необходимы, если вы хотите делать графику достаточно высокого уровня, чтобы ее можно было публиковать. Они помогают не только в плане эстетики, но также делают объект более читабельным и понятным, чего часто сложно добиться, работая с автоматически сгенерированными результатами.

Маппинг

Возможности инструментов для маппинга частично совпадают с возможностями инструментов для визуализации, о которых мы говорили выше. Однако в последние годы объемы географических данных значительно возросли, а вместе с ними увеличилось и количество способов, которыми можно пользоваться для создания карт. Услуги мобильного позиционирования находятся на подъеме, все больше становятся массивы данных с привязанными к ним широтой и долготой. Помимо прочего, карты — невероятно интуитивный способ визуализации данных, и они заслуживают более внимательного рассмотрения.

В первые годы существования Сети создание карт было дело непростым. Да и результат не отличался элегантностью. Вы помните те дни, когда приходилось обращаться к MapQuest, выполнять массу инструкций и в конечном итоге получать малюсенькую статичную карту? В какой-то момент и у Yahoo был такой сервис.

Длилось это до тех пор, пока Google не реализовал принцип *подвижной карты* (рис. 3.22). Хотя технология была изобретена раньше, она не находила применения, пока скорость интернета

у большинства людей не выросла достаточно, чтобы обеспечивать непрерывное обновление данных. Сегодня мы уже привыкли к подвижным картам. Мы с легкостью их прокручиваем и увеличиваем, а в некоторых случаях карты оказываются нужны нам не только для определения направления движения — они становятся основным интерфейсом для просмотра набора данных.

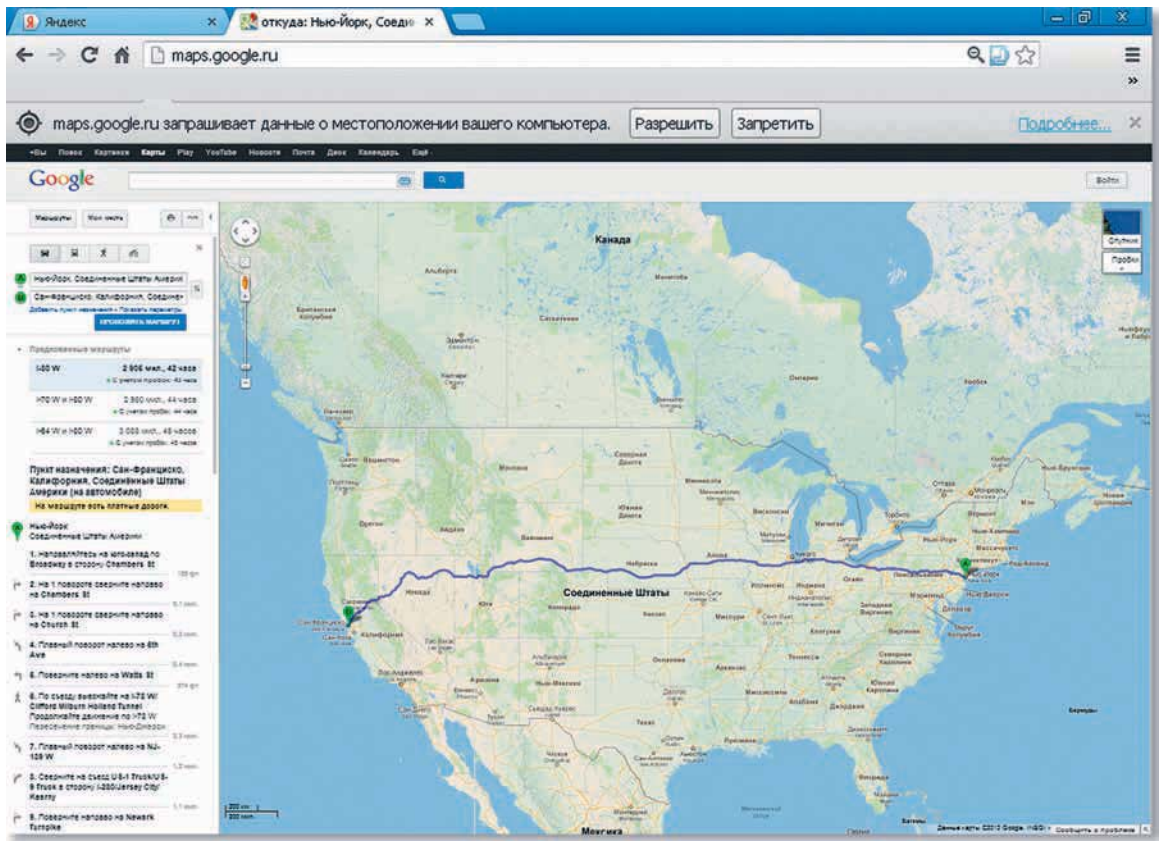


Рис. 3.23. На Google Maps можно получить также и инструкции

ПРИМЕЧАНИЕ

Подвижные карты — это принцип реализации картографических данных, который сегодня стал практически универсальным. Большие карты, которые в иной ситуации не поместились бы на экране, делятся на меньшие изображения (или тайлы). Вы видите только те тайлы, которые приходятся на ваше окно, а все остальные спрятаны. Однако стоит только протащить карту мышью, как появляются другие тайлы, и таким образом создается впечатление, что вы перемещаетесь по одной большой карте. Подобный принцип отображения вы могли видеть также и при просмотре фотографий с высоким разрешением.

Опции

По мере того как географические данные все больше становятся общественным достоянием, появляются все новые и все более разнообразные инструменты для составления карт с применением этих данных. В случае с некоторыми из них требуется всего лишь толика умений в области программирования, чтобы можно было с их помощью что-то создать и это что-то запустить. Работа с иными инструментами предполагает несколько бóльшие вложения труда и времени. Но существуют также и решения, не требующие программистских навыков.

КАРТЫ GOOGLE, YAHOO И MICROSOFT

Это самое простое онлайновое решение, но и оно требует, чтобы вы хоть немного ориентировались в программировании. Чем лучше вы умеете писать коды, тем большего вы сможете добиться с помощью API для создания карт, предлагаемых Google, Yahoo и Microsoft.

Основной функционал во всех трех случаях довольно схож, но если вы делаете только первые шаги на данном поприще, я рекомендую вам начать с Google. Мне кажется, это самый надежный вариант. У Google есть API для создания карт как на JavaScript, так и на базе Flash, а помимо этого еще и другие связанные с географией сервисы, такие как геокодирование и прокладывание маршрутов. Просмотрите обучающее руководство о том, как начать работу с системой, а затем уже углубляйтесь в изучение других тем, таких, например, как наносить метки, находить оптимальные маршруты и добавлять слои. Всеобъемлющие инструкции с фрагментами кодов и рекомендациями помогут вам быстро освоиться.

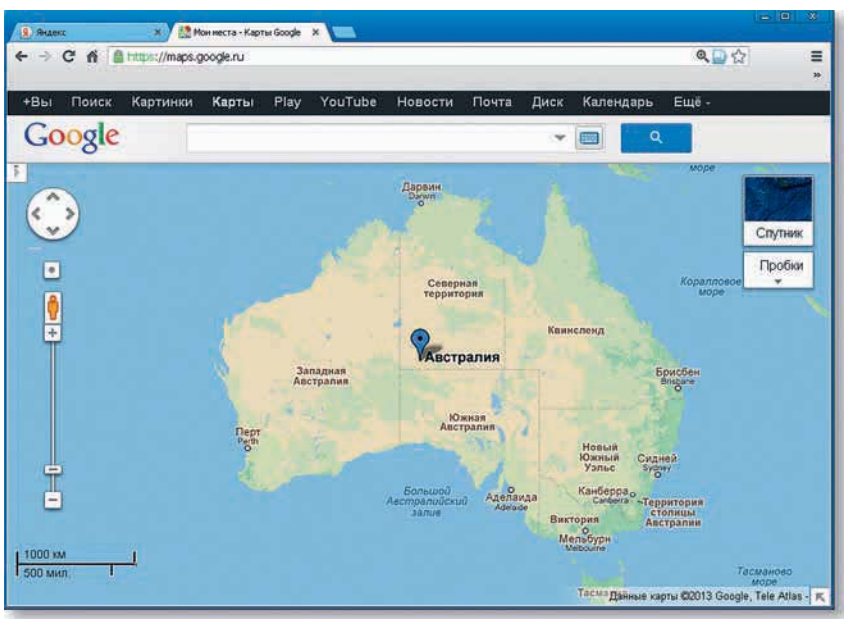


Рис. 3.24. Нанесение меток на Google Maps

У Yahoo также имеется API для создания карт с помощью JavaScript и Flash плюс некоторое количество геосервисов, но я не уверен, как долго они будут доступны, учитывая текущее состояние компании. Когда писалась эта книга, Yahoo переключила свое внимание с разработки приложений на контент-провайдинг. Microsoft также предлагает API для JavaScript (с названием Bing) и еще одно для Silverlight — платформы, которая разрабатывалась этой корпорацией как ответ на Flash.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ СОЗДАНИЯ КАРТ:

- семейство Google Maps API (`http://code.google.com/apis/maps`);
- Yahoo! Maps Web Services (`http://developer.yahoo.com/geo/placefinder`);
- Bing Maps API (`http://www.microsoft.com/maps/developers/web.aspx`).

ARCGIS

Упомянутые выше онлайн-сервисы для создания карт довольно незатейливы в плане того, что они вообще умеют делать. Если вам нужно создавать более сложные карты, вам, скорее всего, придется самостоятельно реализовывать функционал. Однако есть еще ArcGIS, которая разрабатывалась как настольное приложение для создания карт. Это увесистая программа, которая позволяет переносить на карту огромное количество данных и выполнять множество действий, таких как сглаживание и обработка. Все это вы можете делать через пользовательский интерфейс, так что вам не потребуется писать коды.

Почти все службы и отделы графики, в которых работают специалисты по созданию карт, используют ArcGIS. Некоторые люди без ума от нее. Так что если вы интересуетесь созданием детальных карт, вам стоит присмотреться к ArcGIS.

Я использовал ArcGIS в работе над несколькими проектами, потому что я предпочитаю идти «программным путем» и мне все эти функциональные возможности просто были не нужны. У такого богатого инструментария есть и обратная сторона: приходится иметь дело с таким же большим количеством кнопок и меню. Имеются в наличии также онлайн- и серверные решения, но в сравнении с другими разработками они кажутся несколько неуклюжими.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С ARCGIS:

- страница Products на сайте ArcGIS (`http://www.esri.com/software/arcgis`).

MODEST MAPS

Я уже упоминал Modest Maps выше, когда приводил пример, представленный на рис. 3.13. В нем демонстрировался рост торговой сети Walmart. Modest Maps — это Flash- и ActionScript-библиотека для карт на основе тайлов, и она также совместима с Python. Библиотека поддерживается группой людей, которые разбираются в онлайн-маппинге и делают прекрасную работу как для своих клиентов, так и для собственного удовольствия, а это многое говорит о качестве библиотеки.

Самое забавное то, что Modest Maps — это в большей степени фреймворк, нежели API для создания карт вроде тех, что предлагает Google. Она обеспечивает абсолютный минимум того, что необходимо для создания онлайн-карты, а после уже не вмешивается в вашу работу и предоставляет вам возможность реализовать то, что вы хотите. Вы можете использовать тайлы от разных провайдеров, а можете кастомизировать карту так, чтобы она подходила для вашего приложения. Например, на рис. 3.13 представлена карта в сине-черном оформлении, но вы можете с легкостью поменять ее на бело-красную, как это показано на рис. 3.25.

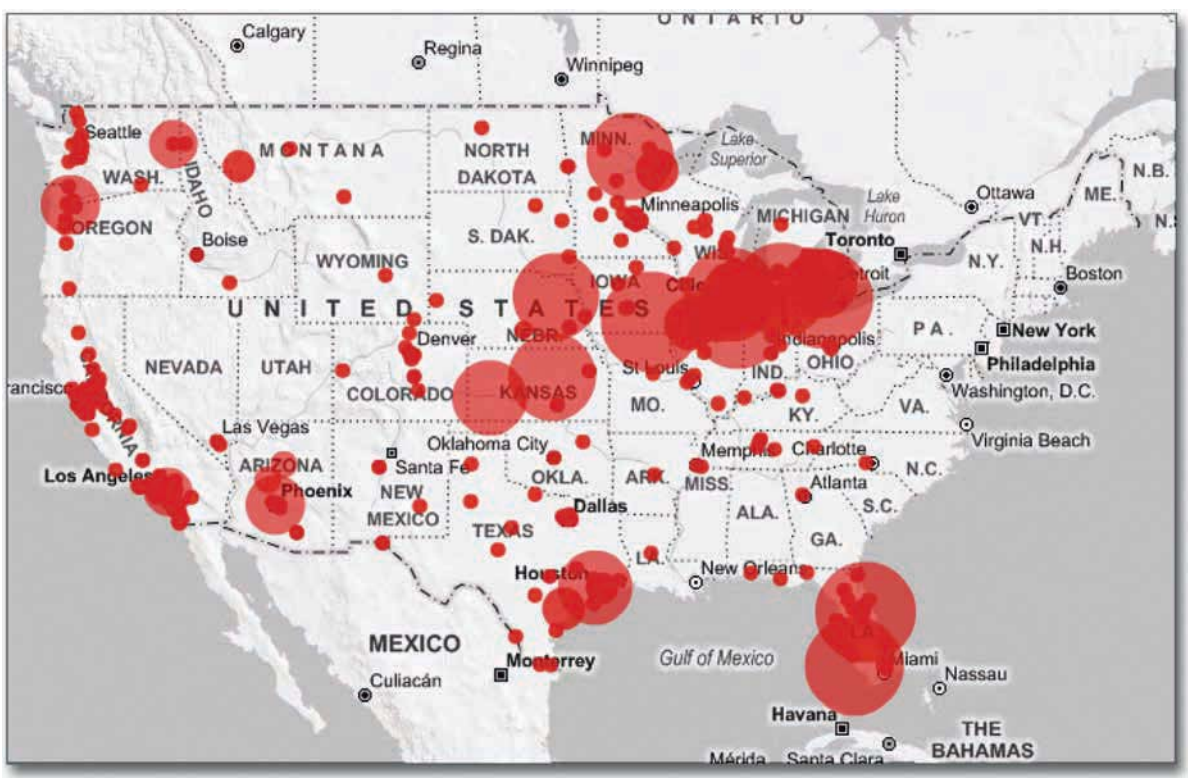


Рис. 3.25. Карта в бело-красном оформлении с использованием Modest Maps

Modest Maps распространяется на условиях лицензии BSD, так что вы можете делать с ней все, что захотите, совершенно бесплатно. Правда, для этого вам необходимо будет научиться работать с Flash и ActionScript, но о них мы еще поговорим в восьмой главе («Визуализация пространственных отношений»).

POLYMAPS

Polymaps — это своего рода JavaScript-версия библиотеки Modest Maps. Она была разработана и поддерживается отчасти теми же самыми людьми и предлагает примерно тот же функционал, но и много чего еще в придачу. Modest Maps предоставляет только базовый функционал в области создания карт, в то время как Polymaps имеет такие встроенные функции, как картограммы (рис. 3.26) и пузырьковые диаграммы.

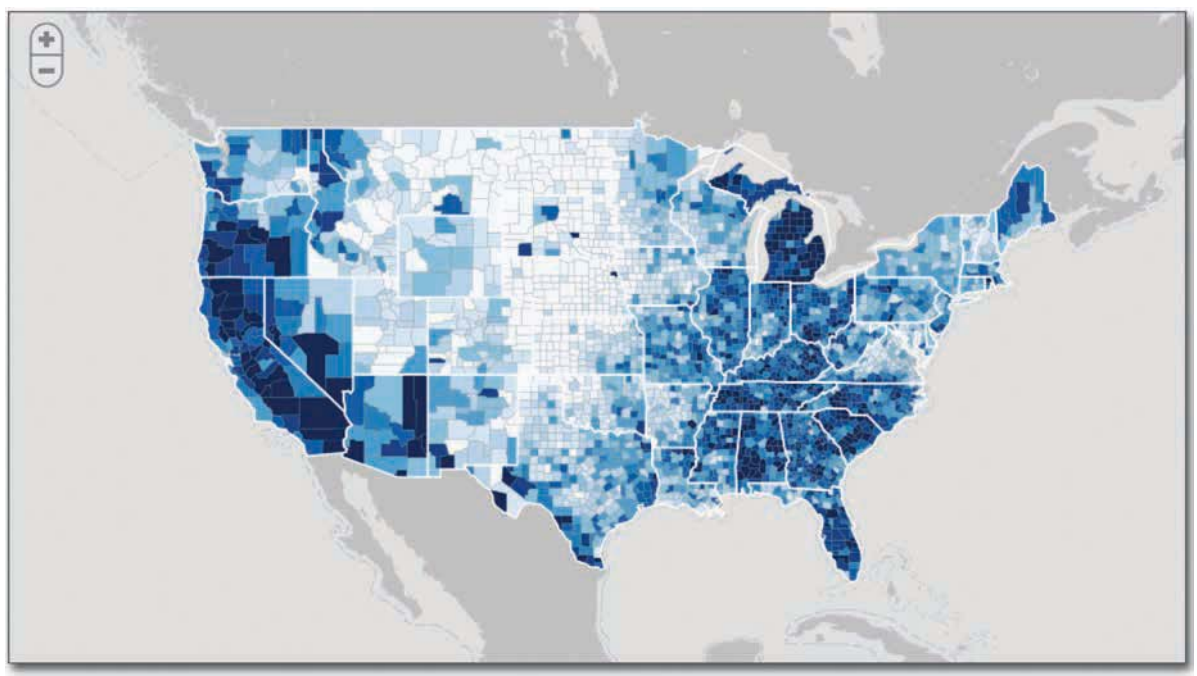


Рис. 3.26. Картограмма, демонстрирующая уровень безработицы, реализованная в Polymaps

Поскольку все это JavaScript, объект кажется более легким (так как требует меньше строк кода) и работает в современных браузерах. Для демонстрации данных Polymaps использует масштабируемую векторную графику (SVG), а потому не работает в старых версиях Internet Explorer, хотя большинство людей идет в ногу со временем. К слову сказать, только 5% посетителей сайта FlowingData пользуются устаревшими веб-браузерами, и я подозреваю, что вскоре их количество упадет до нуля.

Лично я в библиотеках для создания карт на JavaScript ценю больше всего то, что код выполняется в браузерах без сучка и задоринки. Вам ничего не приходится делать — никакого компилирования, никакого экспорта Flash. В результате все легко запускается и легко обновляется.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С POLYMAPS:

- Polymaps (`http://polymaps.org`).

R

В базовом варианте распространения R не обладает функционалом для создания карт, но существует несколько пакетов, которые позволяют вам заниматься в R также и маппингом. На рис. 3.27 представлена маленькая карта, которую я сделал в R. Аннотации были добавлены позже, уже в Adobe Illustrator.

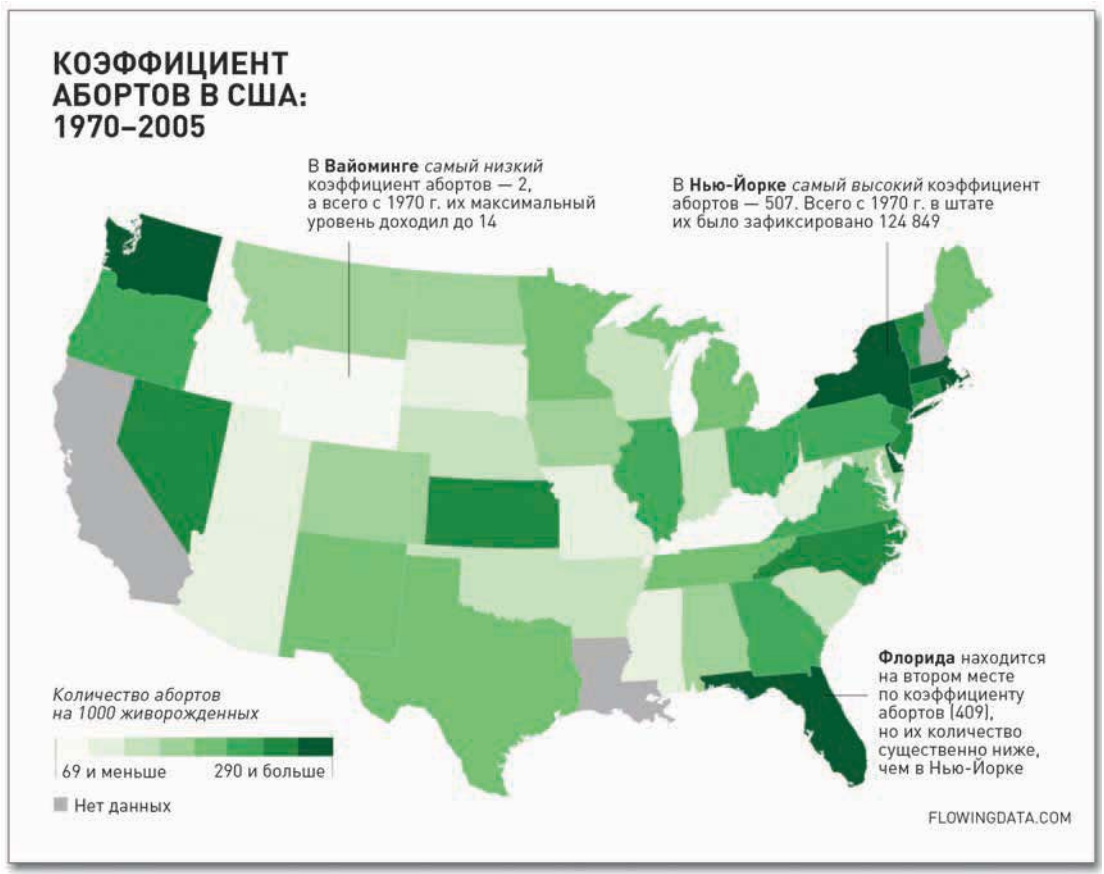


Рис. 3.27. Карта США, созданная в R

Карты, сделанные в R, ограничены в возможностях, да и документация там не на высоте, так что я использую этот способ создания карт, только если мне нужно сделать что-то несложное и по случайности я в этот момент как раз работаю с R. В остальных случаях я предпочитаю прибегать к инструментам, о которых я уже говорил выше.

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ РАБОТЫ С R ПРИ СОЗДАНИИ КАРТ:

- «Анализ пространственных данных» (`http://cran.r-project.org/web/views/Spatial.html`) — исчерпывающий список пакетов на R для пространственного анализа;
- «Практические рекомендации по геостатистическому картостроению» (A Practical Guide to Geostatistical Mapping) — свободно скачиваемая с `http://spatial-analyst.net/book/download` книга о том, как использовать R и другие инструменты для работы с пространственными данными.

ОНЛАЙН-РЕШЕНИЯ

Существует также несколько онлайн-решений для создания карт, которые позволяют с легкостью визуализировать географические данные. В большинстве случаев они берут за основу карты, которые люди используют чаще всего, и удаляют с них все лишнее. Получается нечто похожее на упрощенный ArcGIS. К двум из этих ресурсов доступ свободный. Это Many Eyes и GeoCommons. Первый из них — о нем мы говорили выше — обладает лишь базовым функционалом для работы с данными по странам или по штатам США. А вот GeoCommons предоставляет более широкие функциональные возможности и более богатый инструментарий для взаимодействия. А еще GeoCommons поддерживает самые распространенные форматы файлов с картографическими данными, такие как шейп-файлы и KML.

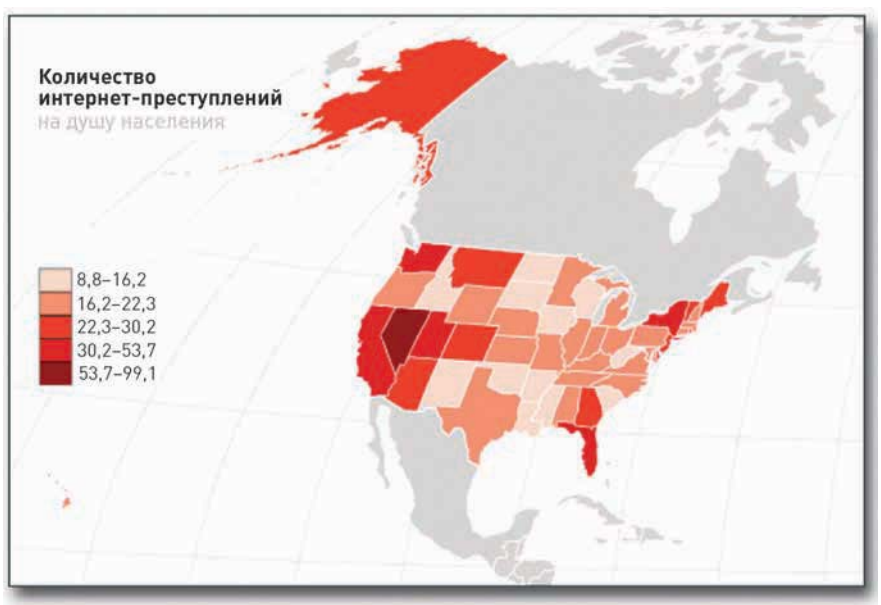


Рис. 3.28. Картограмма, созданная в Indiemapper

Имеется также множество платных решений, самые полезные из них — это Indiemapper и SpatialKey. SpatialKey пригоден в большей степени для бизнеса и принятия решений, в то время как Indiemapper отлично подходит для задач картографов и дизайнеров. На рис. 3.28 представлен пример картограммы, которую я состряпал в Indiemapper всего за несколько минут.

Компромиссы

Программы для создания карт встречаются самых разных видов и предназначены для удовлетворения различных потребностей. Было бы замечательно — освоить одну-единственную программу и иметь возможность создавать все мыслимые виды карт. К несчастью, так не получится.

ArcGIS, например, обладает множеством функций, но вам, возможно, не стоит тратить время на ее изучение и деньги на ее покупку, если нужно создавать только простенькие карты. А вот бесплатный R с его базовым функционалом, наоборот, может оказаться слишком простым для того, что вы хотите сделать. Если ваша цель — интерактивные онлайн-карты, вас вполне могут устроить Modest Maps и Polymaps, но тогда вам понадобятся более серьезные навыки в сфере программирования. Больше информации о том, как пользоваться имеющимися возможностями в области маппинга, вы получите в главе 8.

Изучите имеющиеся возможности

Данный перечень инструментов отнюдь не исчерпывает все варианты, которыми вы можете пользоваться для визуализации данных, но на первых порах вам его должно хватить. Тут есть много о чем подумать и много с чем поиграть. То, какими инструментами вы в конечном счете будете пользоваться, во многом зависит от того, чего вы хотите добиться, причем всегда есть множество подходов к выполнению задачи даже в рамках одной-единственной программы. Хотите создать статичную информационную диаграмму? Может быть, стоит остановиться на R или Illustrator. Хотите построить интерактивный инструмент для веб-приложения? Тогда попробуйте JavaScript или Flash.

На сайте FlowingData я провел опрос, пытаясь выяснить, чем люди в основном пользуются для анализа и визуализации данных. Ответили чуть более 1000 человек. Результаты представлены на рис. 3.29.

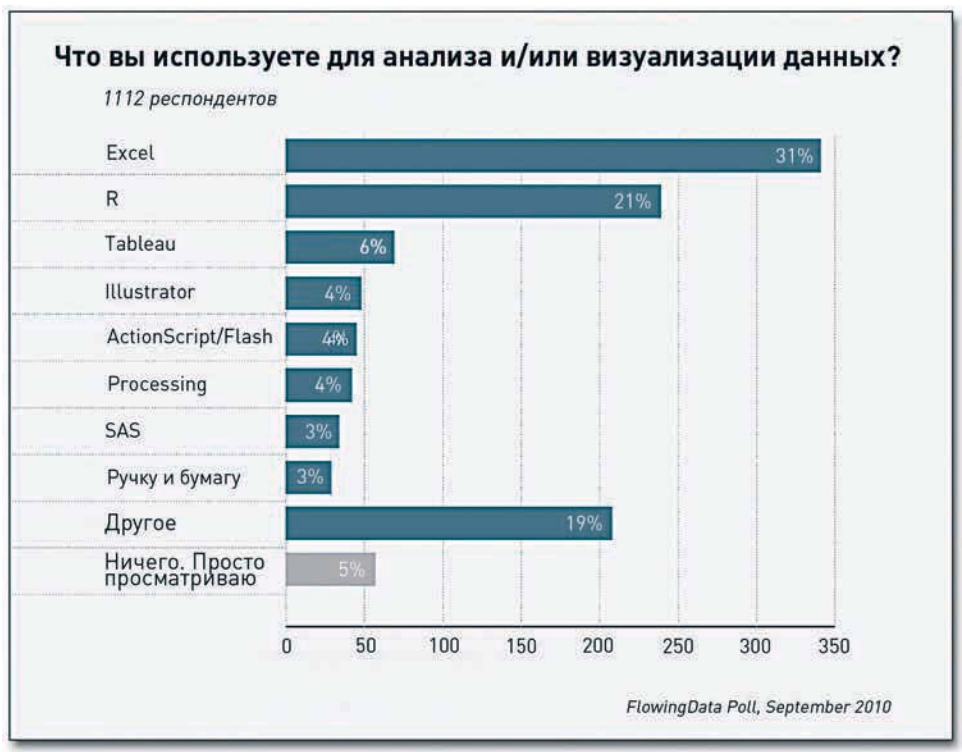


Рис. 3.29. Что используют читатели FlowingData для анализа и визуализации данных

Среди ответов есть несколько очевидных лидеров, особенно учитывая тематику FlowingData. На первом месте идет Excel, за ним следует R. А дальше во мнениях и предпочтениях при выборе программного обеспечения наступает разнобой. Более 200 человек выбрали категорию «Другое». В своих комментариях многие заявляли о том, что они комбинируют инструменты для удовлетворения разных потребностей — а в долгосрочном плане это обычно является наиболее эффективным подходом.

Комбинирование возможностей

Многие любят работать только с одной программой — это легко и удобно. Не нужно изучать ничего нового. Если этого хватает для удовлетворения ваших потребностей в области визуализации, тогда не надо отступать от данного принципа. Но после того как вы поработаете с данными достаточно долго, наступит момент, когда вы поймете, что возможности программного обеспечения исчерпаны. Вы будете знать, что нужно сделать с данными или как их визуализировать, однако программа не позволит вам это осуществить или сделает процесс более трудоемким, чем следовало бы.

Вы можете смириться с таким положением, а можете начать использовать другие программы, на изучение которых уйдет время, но которые помогут вам осуществить свой дизайнерский замысел. Я вам предлагаю пойти вторым путем. Владение разнообразными инструментами гарантирует, что вы не запутаетесь в данных и что вам хватит гибкости, чтобы выполнять все многообразие визуальных задач и получать реальные результаты.

Закругляясь

Помните: ни один из этих инструментов не панацея. В конечном итоге анализ данных и дизайн всегда будут зависеть от вас. Ведь инструменты являются всего лишь инструментами. То, что у вас есть молоток, вовсе не означает, что вы способны построить дом. Точно так же вы можете иметь в своем распоряжении отличную программу и суперкомпьютер, но если вы не будете знать, как пользоваться этими инструментами, считайте, что их у вас нет. Именно вы решаете, какие вопросы задавать, какие данные использовать и какие их грани высветить, а понимание этого приходит с опытом.

Но вам повезло! Ведь именно этому посвящена вся оставшаяся часть книги. В следующих главах вы познакомитесь с основными концепциями информационного дизайна и научитесь, как претворять теорию на практике, применяя ту или иную комбинацию инструментов, о которых мы говорили выше. Вы узнаете, что именно нужно искать в имеющихся у вас данных и как эти данные визуализировать.

Визуализация паттернов во времени

4

Данные в виде временны́х рядов окружают нас буквально повсюду. Общественное мнение меняется, население меняется, бизнес меняется... Мы изучаем временные ряды, чтобы понять, насколько сильно все это изменилось. Эта глава посвящена дискретным и непрерывным данным, потому что способ графического изображения зависит от типа данных, которыми вы располагаете. А еще здесь вы сможете попробовать свои силы в работе с R и Adobe Illustrator — программами, которые отлично сочетаются одна с другой.

Что искать во времени

Вы смотрите на время каждый день. Оно у вас в компьютере, на часах, на телефоне и практически везде, куда бы вы ни бросили взгляд. Время вы ощущаете даже без часов — когда просыпаетесь или засыпаете, когда солнце встает или заходит. Так что для человека вполне естественно, что данные располагаются во времени. Это дает возможность видеть, как именно происходят изменения.

Чаще всего во временны́х рядах, или в темпоральных данных, люди стараются разглядеть тенденции. Что происходит с тем или иным объектом — рост или падение? Наблюдаются ли какие-нибудь сезонные циклы? Чтобы найти эти паттерны, вам необходимо посмотреть дальше, за пределы одного измерения, и увидеть всю картину. Легко взять единичные значения в какой-то конкретный момент времени и решить, что дело сделано. Но если вы взглянете на данные за период до этого момента и после него, у вас сложится более хорошее понимание того, что означают единичные данные, а чем больше вы будете знать о своих данных, тем лучше получится история, которую вы хотите рассказать.

Администрация Обамы, например, через год после его вступления в должность президента выпустила следующую диаграмму (рис. 4.1). На ней представлена ситуация с безработицей в конце периода правления администрации Буша и в первый год президентства Обамы.

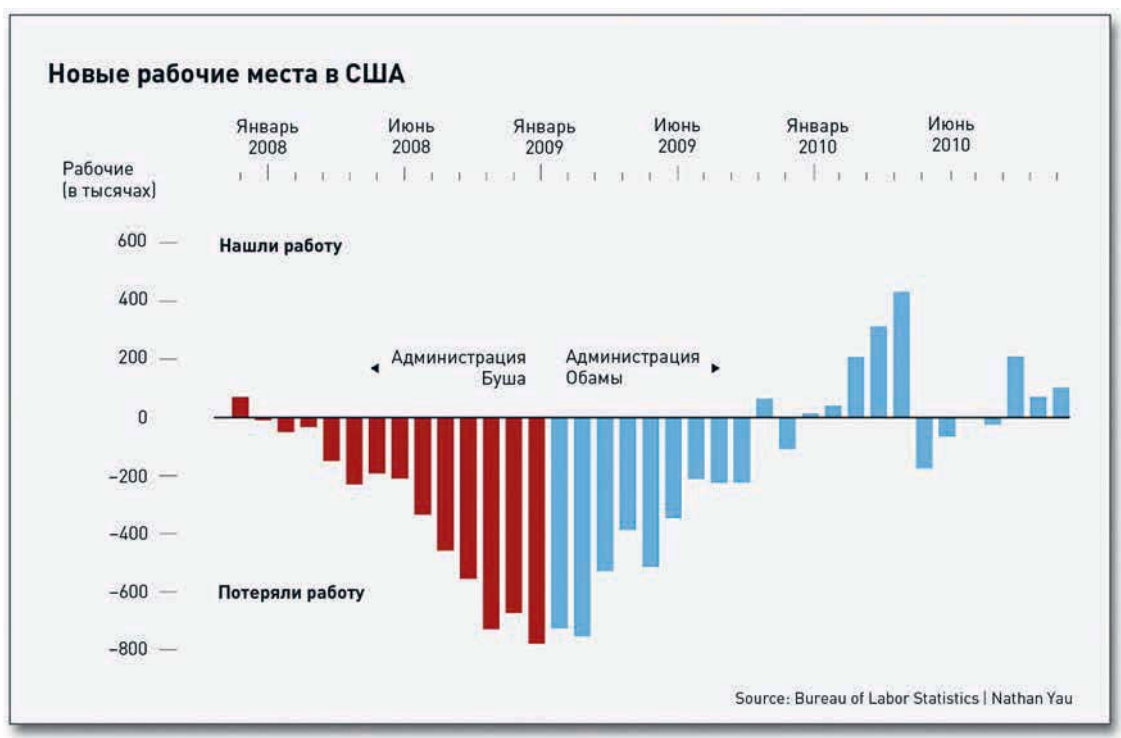


Рис. 4.1. Изменения в ситуации с безработицей с момента вступления Барака Обамы в должность президента

Складывается впечатление, что администрация существенным образом улучшила ситуацию с безработицей. А что будет, если уменьшить масштаб и посмотреть на то, как складывались дела в этой области на протяжении более длительного периода времени, как это показано на рис. 4.2? Замечаете разницу?

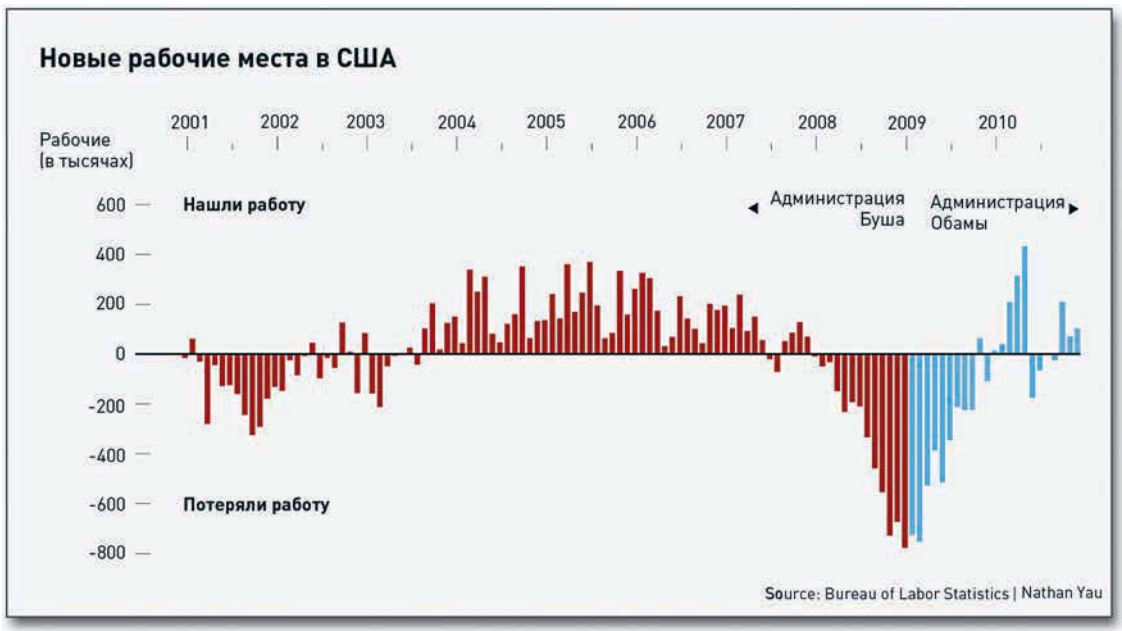


Рис. 4.2. Изменения в ситуации с безработицей с 2001 по 2010 годы

Хотя всегда хочется увидеть картину целиком, бывает не менее полезно рассмотреть данные в деталях. Нет ли среди них каких-нибудь флуктуаций, значений, которые явно выбиваются из общей массы? Нет ли данных, которые кажутся находящимися не на своем месте? Нет ли каких-нибудь пиков и провалов? Если да, то что происходило в это время? Очень часто именно такие неровности и оказываются теми самыми моментами, на которых вы захотите сконцентрироваться. А бывает и так, что в конечном счете выясняется: это всего лишь ошибки, произошедшие при вводе данных. Представление о картине целиком, о контексте поможет вам определить, что есть что.

Дискретные моменты времени

Темпоральные данные можно поделить на дискретные и непрерывные. Зная, к какой из этих двух категорий принадлежат ваши данные, вы сможете решить, как именно визуализировать их. В случае с дискретными данными значения относятся к неким конкретным моментам или

периодам времени, и существует конечное число этих значений. Например, процент людей, которые проходят экзамен каждый год, дискретен. Люди сдают экзамен, и всё. Их оценки впоследствии уже не меняются, и дата проведения экзамена тоже. Другие же явления, такие как температура, непрерывны. Ее можно измерять в любое время суток на протяжении любого интервала, и она будет постоянно варьировать.

В этом разделе вы узнаете, какие типы диаграмм и графиков могут помочь вам визуализировать дискретные темпоральные данные, и посмотрите на конкретные примеры того, как создавать подобные графические объекты в R и Illustrator. В начале будут изложены основные принципы, а затем вы сможете использовать полученные шаблоны на протяжении всей главы. Эта часть книги очень важна. Хотя примеры даны для конкретных типов диаграмм, вы можете применять данные принципы во всех видах визуализации. Помните: это все делается для картины в целом.

Столбцы

Столбцовые диаграммы — один из самых популярных типов диаграмм. Наверняка вы видели множество подобных фигур. Столбцовые диаграммы годятся для самых разных типов данных, но сейчас мы посмотрим, как их можно использовать для темпоральных данных.

На рис. 4.3 представлена принципиальная структура такой диаграммы. Ось времени (горизонтальная ось, она же ось абсцисс, или ось X) — это место, где обозначаются моменты времени, выстраиваемые в хронологическом порядке. В данном случае такими моментами времени являются месяцы (от января до июня 2011 года), но с таким же успехом здесь мог бы оказаться ряд годов или дней либо ряд каких-то других единиц времени. Ширина столбцов и расстояние между ними, как правило, не выражают никаких значений.

Рис. 4.3. Структура столбцовой диаграммы

На оси значений (вертикальной, она же ось ординат, или ось Y) наносится шкала диаграммы. Шкала на рис. 4.3 линейная — единицы расположены на равном расстоянии друг от друга по всей длине оси. Высота столбцов определяется по оси значений. Первый столбец, например, доходит до первого деления, а самый высокий столбец — до четвертого.

Это важно. Высота столбца является визуальным выражением значения. Чем меньше значение, тем ниже будет столбец. Чем больше значение, тем столбец будет выше. Как вы видите, столбец высотой до четвертого деления в апреле в два раза выше, чем февральский, чья высота — две единицы.

Многие программы в качестве точки отсчета оси ординат по умолчанию ставят минимальное значение, содержащееся в наборе данных, как это показано на рис. 4.4. В данном случае минимальное значение — это 1. Однако если вы начнете ось значений с единицы, тогда высота февральского столбца будет уже не в два раза меньше высоты апрельского столбца, и может сложиться впечатление, что он составляет всего треть от высоты апрельского. А январский столбец и вовсе перестанет существовать. Вывод: всегда начинайте ось значений с нуля. Иначе соотношение величин на вашей столбцовой диаграмме будет отображаться некорректно.

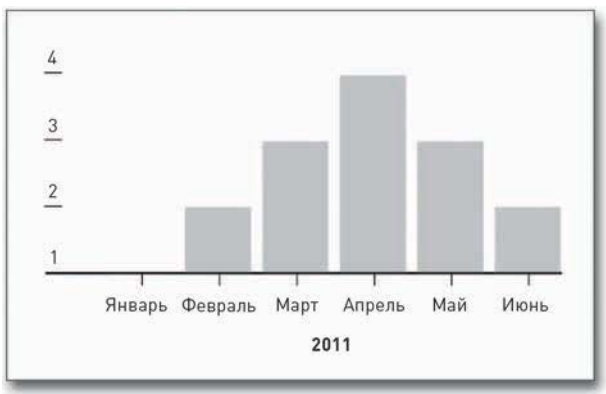


Рис. 4.4. Столбцовая диаграмма, в которой ось начинается не с нуля

ПОДСКАЗКА

Когда вы имеете дело с положительными величинами, всегда начинайте ось значений столбцовой диаграммы с нуля. Любой иной подход затруднит визуальное сравнение столбцов по высоте.

СОЗДАЙТЕ СТОЛБЦОВУЮ ДИАГРАММУ

Пора вам сделать первую диаграмму, используя реальные данные, и они относятся к важному разделу истории, которую должны знать все, называющие себя «человеками». Это результаты Нейтановского турнира по поеданию хот-догов за последние три десятилетия. О да!

На рис. 4.5 представлен окончательный вариант диаграммы, которую вы будете стараться построить. Вы сделаете это в два этапа: сначала создадите базовый вариант столбцовой диаграммы в R, а затем обработаете ее в Illustrator.

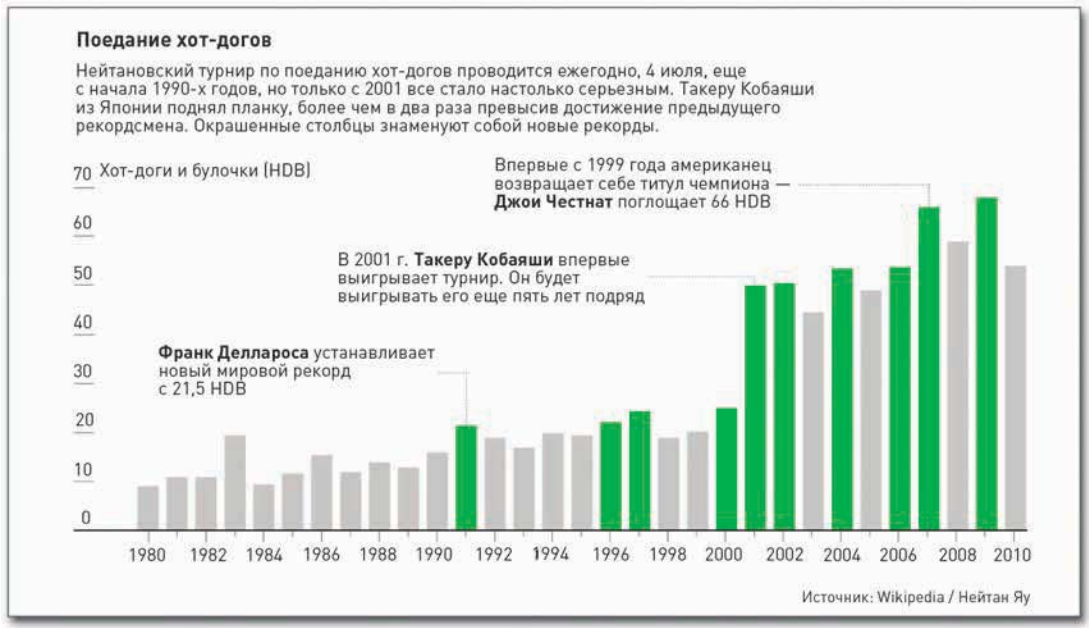


Рис. 4.5. Столбцовая диаграмма, демонстрирующая результаты Нейтановского турнира по поеданию хот-догов

На тот случай, если вы не в курсе событий в области соревновательного поедания пищевых продуктов, то Нейтановский турнир по поеданию хот-догов — это ежегодное мероприятие, которое проходит 4 июля. В Соединенных Штатах это День независимости. Соревнование стало настолько популярным, что его даже показывают по телевидению на канале ESPN. С конца 1990-х годов примерно за 15 минут победители съедали от 10 до 20 хот-догов и булочек (HDB). Однако в 2001 году Такеру Кобаяши (Takeru Kobayashi), профессиональный едок из Японии, наголову разбил конкурентов, съев 50 HDB. Это в два с лишним раза больше того количества, которое кто-либо в мире съедал до него. И вот с этого момента начинается история.

В «Википедии» хранятся результаты турнира еще с 1916 года, но поедание хот-догов стало регулярным мероприятием только в 1980-х годах. Данные представлены в виде HTML-таблицы и включают в себя год (Year), имя победителя (Winner) и количество съеденных хот-догов (Dogs eaten), а также страну (Country), которую представлял победитель. Я скомпилировал данные в CSV-файл, который вы можете скачать с `http://datasets.flowingdata.com/hot-dog-contest-winners.csv`.

Вот как выглядят первые пять строчек данных:

```
"Year","Winner","Dogs eaten","Country","New record"
1980,"Paul Siederman & Joe Baldini",9.1,"United States",0
1981,"Thomas DeBerry ",11,"United States",0
1982,"Steven Abrams ",11,"United States",0
1983,"Luis Llamas ",19.5,"Mexico",1
1984,"Birgit Felden ",9.5,"Germany",0
```

▶ Скачайте данные в формате CSV с `http://datasets.flowingdata.com/hot-dog-contest-winners.csv`. Посмотрите страницу, посвященную Нейтановскому турниру по поеданию хот-догов (Nathan's Hot Dog Eating Contest) в «Википедии», чтобы ознакомиться с исходными данными и историей соревнования.

Чтобы загрузить данные в R, используйте команду `read.csv()`. Вы можете или загрузить файл локально с вашего собственного компьютера, или обратиться к нему по URL. Для второго варианта введите в код на R следующую строку:

```
hotdogs <-
   read.csv("http://datasets.flowingdata.com/hot-dog-contest-winners.csv",
   sep=",", header=TRUE)
```

Если же вы хотите загрузить данные локально, используйте главное меню и поместите свой рабочий каталог в R в ту же директорию, в которой находится и ваш файл с данными. Или же вы можете воспользоваться функцией `setwd()`.

Если вы новичок в области программирования, все это, скорее всего, покажется вам какой-то тайнописью, а потому давайте разберем фрагмент по частям, чтобы вы все поняли. Это строчка кода на R. Командой `read.csv()` вы загружаете данные. У нее есть три аргумента. Первый — это местоположение ваших данных. В приведенном примере это URL.

Второй аргумент, `sep`, определяет, какой знак разделяет колонки в файле с данными. В нашем случае это файл с разделителями-запятыми, а потому указывается запятая. Если бы это был файл, в котором разделителем является символ табуляции, тогда бы вам пришлось использовать не запятую, а `\t`.

Последний аргумент — `header` — сообщает R, что у файла с данными есть шапка, которая содержит названия всех колонок. Первая колонка — это год, вторая — имя победителя, третья — количество съеденных хот-догов и булочек, четвертая — страна, которую представлял победитель. Как вы, возможно, заметили, я добавил еще одно поле: новый рекорд (New record). Если в соответствующем году мировой рекорд оказался побит, в ячейке будет стоять 1, в противном случае — 0. Скоро вы воспользуетесь этой информацией.

Итак, данные загружены в R и доступны через переменную `hotdogs`. Точнее говоря, данные сохранены в виде таблицы, но это пока не имеет принципиального значения. Вот как будет выглядеть начало таблицы данных, если вы наберете **hotdogs**:

```
  Year              Winner              Dogs.eaten      Country      New.record
1 1980    Paul Siederman & Joe Baldini       9.10     United States        0
2 1981    Thomas DeBerry                    11.00     United States        0
3 1982    Steven Abrams                     11.00     United States        0
4 1983    Luis Llamas                       19.50     Mexico               1
5 1984    Birgit Felden                      9.50     Germany              0
```

Пробелы в названиях колонок были заменены точками. Вместо `Dogs eaten` теперь имеем `Dogs.eaten`. То же самое и с `New.record`. Для обращения к конкретной колонке данных вам нужно использовать название таблицы, за которым следует знак доллара ($), а за ним название колонки. Например, если вы хотите обратиться к `Dogs.eaten`, необходимо ввести следующее:

```
hotdogs$Dogs.eaten
```

Теперь, когда данные уже в R, вы можете сразу же перейти к вычерчиванию диаграммы с помощью команды `barplot()`.

```
barplot(hotdogs$Dogs.eaten)
```

Таким образом вы отдаете распоряжение R начертить диаграмму на основе данных из колонки `Dogs.eaten`. В результате вы получите столбцовую диаграмму, представленную на рис. 4.6.

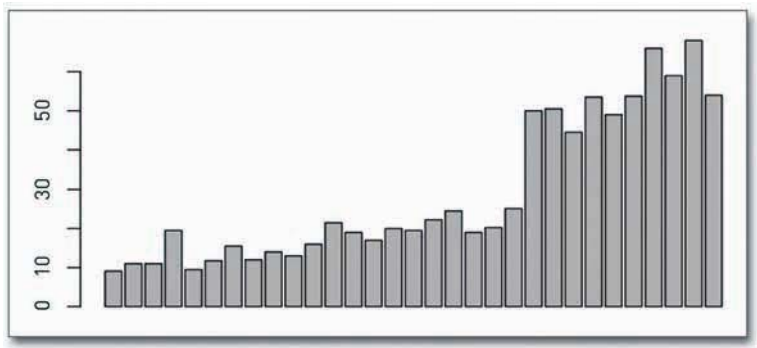


Рис. 4.6. Диаграмма по умолчанию, созданная с использованием barplot() в R, на которой представлено количество съеденных хот-догов

Получилось неплохо, но вы можете добиться лучшего. Используйте аргумент `names.arg` в `barplot()`, чтобы дать названия столбцам. В данном случае это год проведения каждого соревнования.

```
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year)
```

В результате вы получите то, что изображено на рис. 4.7: теперь внизу есть подписи.

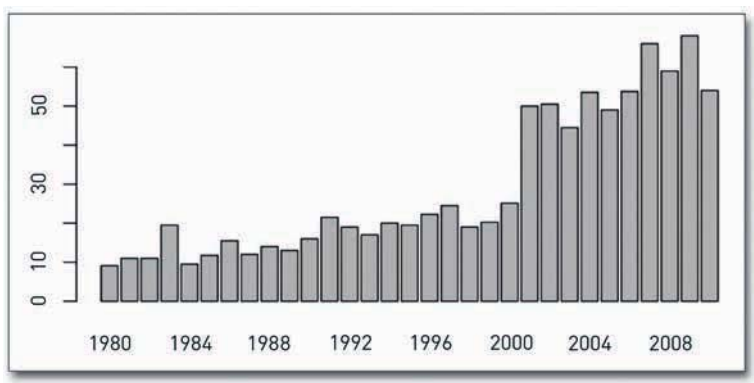


Рис. 4.7. Столбцовая диаграмма с обозначениями года

Вы можете использовать и другие аргументы. Можете разместить подписи к осям, изменить рамки и добавить цвета, как показано на рис. 4.8.

```
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year, col="red",
   border=NA, xlab="Year", ylab="Hot dogs and buns (HDB) eaten")
```

Аргумент `col` служит для задания цвета (возможные варианты перечислены в документации к R) или шестнадцатеричного числа, такого как `#821122`. В данном случае был выбран вариант без границ — `NA` (это логическая константа, означающая «без значения»). Кроме того, были добавлены подписи к осям X и Y: «Year» («Год») и «Hot dogs and buns (HDB) eaten» («Съеденные хот-доги и булочки (HDB)») соответственно.

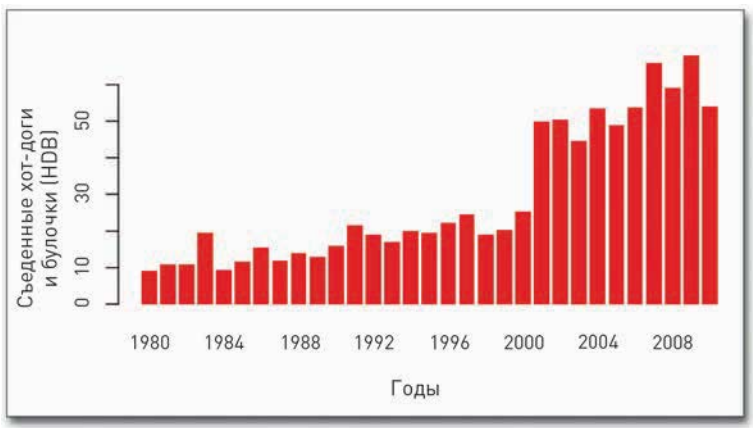


Рис. 4.8. Столбцовая диаграмма с окрашенными столбцами и подписями к осям

Однако вы отнюдь не ограничены одним-единственным цветом. Вы можете перечислить в `barplot()` множество цветов, раскрасив каждый столбик, как хотите. Допустим, вам нужно выделить годы, когда Соединенные Штаты выигрывали соревнование. Эти годы вы можете окрасить в темно-красный цвет (`#821122`), а все остальные — в светло-серый, как показано на рис. 4.9.

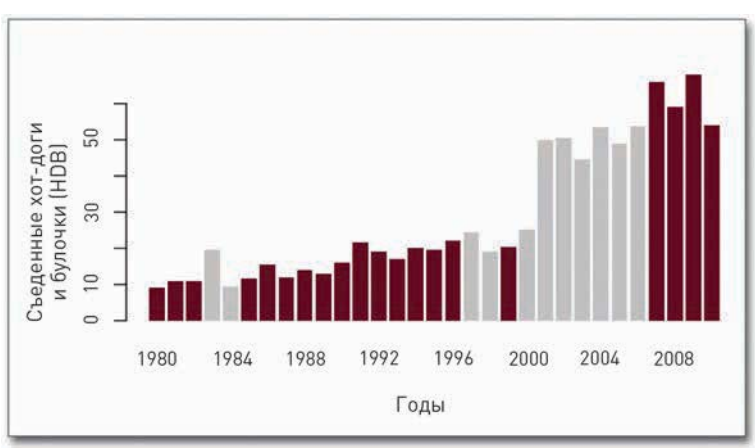


Рис. 4.9. Столбцовая диаграмма с индивидуально окрашенными столбцами

Чтобы сделать это, вам необходимо составить список — или вектор (в R) — цветов. Ознакомьтесь со всеми годами и решите, какого цвета нужно сделать соответствующий столбец. Если выиграли Соединенные Штаты, пусть столбец будет красным. В противном случае покрасьте его серым. Вот код, с помощью которого можно это осуществить:

```
fill_colors <- c()
for ( i in 1:length(hotdogs$Country)) {
   if (hotdogs$Country[i] == "United States") {
      fill_colors <- c(fill_colors, "#821122")
   } else {
      fill_colors <- c(fill_colors, "#cccccc")
   }
}
```

Первая строка создает пустой вектор под именем `fill_colors`. В R для создания векторов используется команда `c()`.

Следующая строка запускает цикл `for`. Вы можете поручить R повторить цикл для всего индекса `i` от 1 до числа строк в вашей таблице данных `hotdogs`. Точнее говоря, вы можете взять одну-единственную колонку, `Country`, из таблицы данных `hotdogs`, и вычислить длину. Если бы вы использовали `length()` только с `hotdogs`, тогда вы бы получили число колонок, которое в данном случае составляет 5, но вас интересует количество строк, которых 31. Для каждого года с 1980 по 2010 существует отдельная строка, а значит, цикл выполнит код внутри скобок 31 раз, и с каждым разом индекс `i` будет увеличиваться на единицу.

ПРИМЕЧАНИЕ

Большинство языков программирования используют массивы (векторы), в которых точкой отсчета является 0 и первый пункт обозначается как 0-index. Однако R использует векторы с точкой отсчета, равной 1.

Итак, при первой итерации, когда `i` равняется 1, производится проверка того, является ли страна в первом ряду (то есть страна победителя 1980 года) США. Если да, то с помощью `fill_colors` присоединяется цвет `#821122` (это и есть тот самый оттенок красного в шестнадцатеричной форме). В противном случае присоединяется `#cccccc` (светло-серый).

В 1980 году победителем стал представитель США, так что проведите описанную выше операцию. Далее цикл повторяется еще 30 раз для всех остальных годов. Чтобы увидеть, чем все закончится, введите в консоль R `fill_colors`. Это тот самый вектор цветов, который вам нужен.

Передайте вектор `fill_colors` в аргумент `col` для `barplot()` следующим образом:

```
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year, col=fill_colors,
   border=NA, xlab="Year", ylab="Hot dogs and buns (HDB) eaten")
```

Код остался тем же, что и прежде, за тем исключением, что в аргументе `col` вы используете `fill_colors`, а не `"red"`.

В окончательном варианте столбцовой диаграммы (см. рис. 4.5) выделены, однако, не те годы, когда побеждали Соединенные Штаты, а те, когда устанавливался новый мировой рекорд. Однако процесс и логика там точно такие же. Просто нужно изменить некоторые из условий. Информация о новых рекордах в вашей таблице данных содержится в колонке `New.record`,

и если там стоит 1, то столбец должен быть темно-красным, в противном случае — серым. Вот как это делается в R:

```
fill_colors <- c()
for ( i in 1:length(hotdogs$New.record)) {
   if (hotdogs$New.record[i] == 1) {
      fill_colors <- c(fill_colors, "#821122")
   } else {
      fill_colors <- c(fill_colors, "#cccccc")
   }
}
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year, col=fill_colors,
   border=NA, xlab="Year", ylab="Hot dogs and buns (HDB) eaten")
```

Здесь все то же самое, как в примере с Соединенными Штатами, только оператор `if` иной. Полученный вами результат должен походить на то, что изображено рис. 4.10.

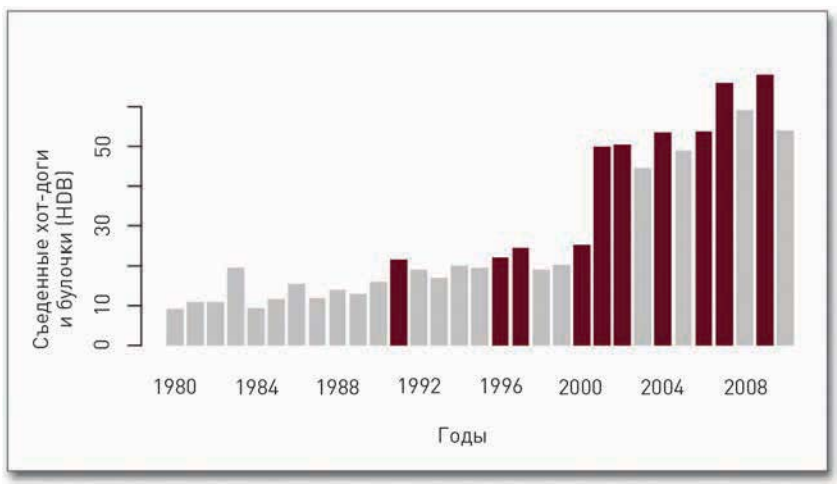


Рис. 4.10. Столбцовая диаграмма с индивидуально окрашенными столбцами, но с использованием других операторов, нежели на рис. 4.9

На этом этапе вы можете поэкспериментировать и с другими вариантами `barplot()`, например изменить расстояние между столбцами или добавить заголовок.

```
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year, col=fill_colors,
   border=NA, space=0.3, xlab="Year", ylab="Hot dogs and buns (HDB)
eaten")
```

В результате вы получите диаграмму, представленную на рис. 4.11. Обратите внимание на то, что интервал между столбцами стал больше, чем прежде, и что наверху появился заголовок.

ПОДСКАЗКА

Будьте осторожны при выборе расстояния между столбцами. Если оно окажется близким к ширине самих столбцов, возникает визуальный эффект вибрации — как будто столбцы и интервалы меняются ролями.

Рис. 4.11. Столбцовая диаграмма с измененными интервалами между столбцами и заголовком

Вуаля! Вот вы и добились первого успеха в R.

В меню File есть опция сохранения диаграммы. Сохраните ее в формате PDF. Она вам скоро понадобится.

ПОДСКАЗКА

Чтобы просмотреть документацию по любой функции R, просто наберите знак вопроса и за ним название функции. Например, чтобы получить информацию о `barplot()`, просто наберите **`?barplot`**. Тут же появится описание функции, а также пояснения к доступным аргументам. Все это обычно сопровождается рабочими примерами, которые бывают исключительно полезными.

ОТРЕДАКТИРУЙТЕ СВОЮ ДИАГРАММУ В ILLUSTRATOR

Теперь у вас есть базовый вариант столбцовой диаграммы. Смотрится она неплохо, и если вы собираетесь использовать ее только для анализа, можете оставить ее как есть. Но если вы хотите сделать из нее нечто выдающееся, тогда стоит еще кое-что изменить для повышения ее читабельности.

Посмотрите на данный объект уже с точки зрения сторителлинга. Представьте себе, что вы видите рис. 4.11 впервые. Вы — читатель и случайно наткнулись на эту диаграмму. Какую информацию вы можете почерпнуть из нее? Вы знаете, что на диаграмме отображено поедание хот-догов и булочек по годам. Значит ли это, что речь идет о пищевых привычках конкретного человека? Для одного человека хот-догов явно многовато. Может, это корм для животных?

А остатки отдают птичкам? Или здесь представлено количество хот-догов, поедаемых людьми за год в среднем? И почему столбцы окрашены?

Поскольку вы — тот самый человек, который делал эту диаграмму, вам известен контекст, стоящий за каждым числом. Но читатели ничего не знают, а потому вам следует объяснить им, что здесь происходит. Хороший дизайн помогает читателям лучше вникнуть в историю. И в этом вас выручит программа Illustrator — она позволяет вручную манипулировать каждым из элементов графики. Вы можете задать другие шрифты, добавить примечания, видоизменить оси, отредактировать цвета и вообще сделать все, что подскажет вам ваша фантазия.

ПОДСКАЗКА

Когда вы разрабатываете информационную графику, поставьте себя на место читателя и задайтесь вопросом: какие части данных нуждаются в пояснении?

В этой книге рассказывается о самых простых операциях в Illustrator, но по мере того как вы будете изучать все больше и больше примеров и начнете создавать свои собственные графические объекты, вы убедитесь, что эти элементарные действия способны приносить огромную пользу, делая вашу графику ясной и лаконичной.

Однако не будем забегать вперед. Сперва откройте PDF-файл с вашей столбцовой диаграммой в Illustrator. В окне вы увидите собственно диаграмму, а рядом, скорее всего, откроются несколько маленьких окошек с инструментами, цветами, шрифтами и много с чем еще. Главное окно, на которое стоит обратить внимание, — это окно инструментов (рис. 4.12). Им вы будете пользоваться часто. Если вы не видите этого окна, то в пункте главного меню Window (Окно) выберите команду Tools (Инструменты).

Рис. 4.12. Окно инструментов в Illustrator

ПОДСКАЗКА

Если у вас нет Illustrator, можете попробовать Inkscape — бесплатную альтернативу с открытым исходным кодом. Хотя функции и кнопки будут выглядеть не совсем так, тем не менее в Inkscape вы найдете много сходств с Illustrator.

Черная стрелка — это инструмент Selection (Выделение). Выберите его, и указатель вашей мыши превратится в черную стрелку. Щелкните ею по рамке и чуть потяните ее. Рамка станет выделенной (рис. 4.13). В Illustrator это называется обтравочной маской (clipping mask). Она может пригодиться вам в самых разных ситуациях, но сейчас она вас не интересует, так что нажмите клавишу Delete на клавиатуре и избавьтесь от маски. Если в результате будет удалена вся диаграмма, отмените правку и используйте для выделения обтравочной маски инструмент Direct Selection (Прямое выделение), который обозначен белой стрелкой.

А теперь попытайтесь изменить шрифт. Сделать это легко. Снова, используя инструмент Selection (Выделение), щелкните по тексту, чтобы отметить, что именно вы хотите изменить. Поменяйте шрифт на любой другой с помощью выпадающих меню окна Font (Шрифт), как это показано на рис. 4.14. Шрифты вы можете поменять также и через меню Type (Текст). На рис. 4.15 шрифт заменен на Georgia Regular.

Вернитесь в меню Object (Объект) и выберите Transform → Transform Each (Трансформировать → Трансформировать каждый). Как это показано на рис. 4.16, измените угол вращения (rotation angle) на –90 градусов. Щелкните по кнопке OK. Подписи выстроятся по горизонтали.

Рис. 4.13. Удаление обтравочной маски из PDF-документа

Рис. 4.14. Окно со шрифтами в Illustrator

Рис. 4.15. Столбцовая диаграмма, в которой шрифт изменен на Georgia Regular

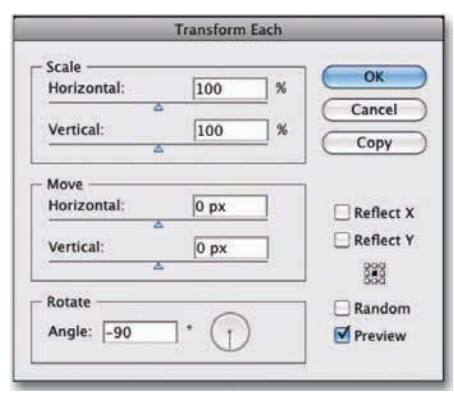


Рис. 4.16. Меню Transform (Трансформирование)

Пока вы все еще там, поднимите подписи (только подписи, не линии разметки) вверх и вправо, чтобы они оказались над метками на оси, а не слева от них. Сделайте это или с помощью клавиш-стрелок на клавиатуре, или мышью. Вы можете также напрямую задать единицы

измерения в хот-догах и булочках (HDB), а не оставлять их сбоку. Это опять-таки повысит читабельность диаграммы. В результате у вас должно получиться что-то похожее на рис. 4.17.

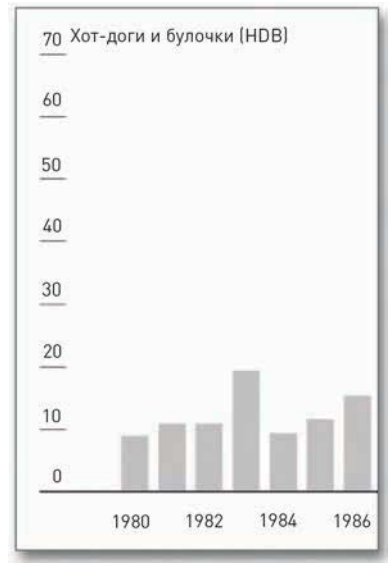


Рис. 4.17. Столбцовая диаграмма с упрощенной осью значений

Это уже становится похожим на окончательный вариант диаграммы, представленной на рис. 4.5. Хотя кое-чего все еще не хватает. На горизонтальной оси нет никаких меток, отсутствуют аннотации, да и не мешало бы добавить зеленого — одного из цветов, содержащихся в логотипе Нейтановского турнира по поеданию хот-догов.

Вы можете также упростить диаграмму, удалив вертикальную линию оси значений. Она не несет читателю никакой особой информации. Обратите внимание на то, что в окончательном варианте диаграммы мы имеем только штриховые метки. Если вы выберете вертикальную линию инструментом Selection (Выделение), подписи также окажутся выделенными. Так происходит, потому что они являются частью группы. Чтобы выбрать одну линию, используйте инструмент Direct Selection (Прямое выделение). После того как линия окажется выделенной, нажмите клавишу Delete, и она исчезнет.

Существует множество способов создания линий разметки, но мы сейчас остановимся только на одном из них. Используя инструмент Pen (Перо), вы можете с легкостью чертить прямые линии. Выберите его из окна инструментов, а затем установите стиль ваших линий в окне Stroke (Обводка). Задайте толщину штриха 0,3 pt и убедитесь, что флажок Dashed Line (Пунктирная линия) не поставлен.

Чтобы начертить линию, щелкните мышью в том месте, где вы хотите расположить первую точку линии (или штриха разметки, как в данном случае), а затем щелкните там, где хотите поставить вторую точку. Если во время второго щелчка вы будете держать клавишу Shift, то автоматически получите прямую линию. Пока у вас есть одна метка. Но вам нужны еще 30 — по метке на каждый год.

ПОДСКАЗКА

Информационная графика предназначена для того, чтобы проливать свет на ваши данные. Постарайтесь удалить все элементы, которые не помогают вам в этом.

Вы можете проставить их все вручную, но есть способ лучше. Если вы работаете на Mac, зажмите клавишу Option, а если у вас PC — клавишу Alt. Таким образом будет создана копия, и у вас окажутся две метки. Далее нажмите Command + D для Mac или Control + D для PC. Сделав это, вы продублируете новую метку, причем с таким же интервалом от предыдущей, как между первыми двумя. Нажимайте Command/Control + D до тех пор, пока не получите необходимое количество меток.

Наконец, вам нужно расположить все метки правильно. Сдвиньте последнюю из них так, чтобы она оказалась в центре под последним столбцом. Первая метка и так уже должна находиться в центре под первым столбцом. Теперь выделите все метки и выберите в окне Align (Выравнивание) вариант Horizontal Distribute Center (По горизонтали по центру), как это показано на рис. 4.18.

После этого все метки распределятся на равных расстояниях между двумя крайними. Если вы захотите, то с помощью инструмента для выделения (Selection) можете выбрать каждую вторую метку и изменить ее размер по вертикали, сделав ее короче. Так с первого взгляда будет

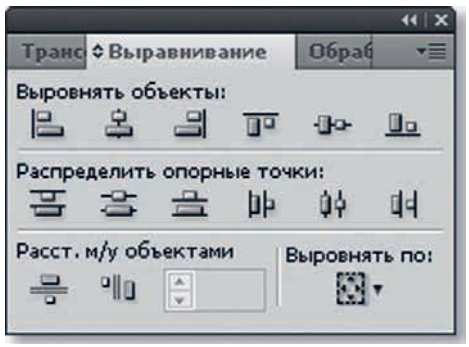


Рис. 4.18. Окно Align (Выравнивание) в Illustrator

понятно, что более длинные метки относятся к сопроводительным подписям с годами.

Чтобы изменить цвет заливки с красного на зеленый, вы можете вернуться к инструменту частичного выделения (Direct Selection) и выбрать по очереди все красные прямоугольники. Поскольку их относительно мало, это можно выполнить довольно быстро. Но что делать, если таких элементов много? Тогда лучше будет поступить следующим образом: щелкнуть по одному-единственному красному столбцу, а затем выбрать Select → Same → Fill Color (Выделение → По общему признаку → С одинаковым цветом заливки). Случится то, чего вы и ожидали: программа выделит все столбцы с красной заливкой. Теперь просто поменяйте цвет на тот, который вам больше нравится, через окно Color (Цвет). Здесь вы можете перекрасить и границы, и заливку — однако сейчас вам нужно изменить только заливку, как это показано на рис. 4.19.

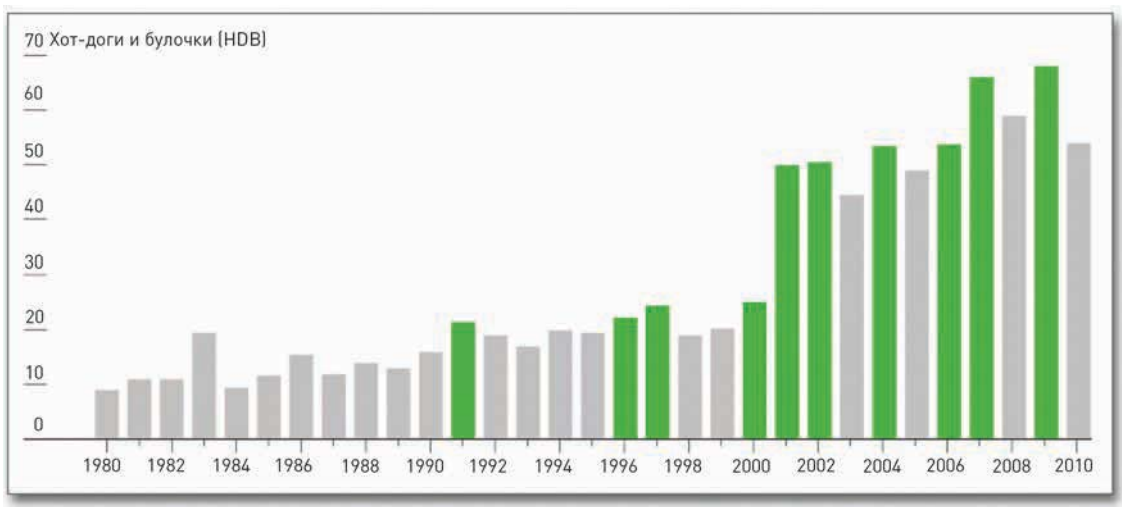


Рис. 4.19. Изменение цвета элементов диаграммы

Используйте инструмент Type (Текст), который находится в окне Tools (Инструменты), чтобы добавить к диаграмме текстовые окошки. Это ваш шанс объяснить читателям, что именно они видят перед собой, глядя на диаграмму, и прояснить для них любые непонятные моменты. Выберите шрифт, который кажется вам подходящим, — причем установите другой кегль и гарнитуру, чтобы подписи отличались от остальных элементов диаграммы, таких как названия осей.

В случае с этой сверхважной хот-договой диаграммой особо стоит выделить первый с 1980 года рекорд, а также период, когда доминировал Такеру Кобаяши, а также сегодняшнее лидерство Джои Честната. Добавьте также заголовок и вводный абзац, объясняющий суть диаграммы.

И вот что еще немаловажно: не забудьте указать источник данных. Без этого не будет никакого способа определить, насколько ваша диаграмма верна.

Сложите все вместе, и вы получите окончательный вариант диаграммы, который представлен на рис. 4.5.

Я знаю, что освоить это было непросто, но чем больше диаграмм вы сделаете, тем легче в итоге окажется ваша работа. Вы увидите, как процесс составления кода на R или на любом другом подходящем для нашего дела языке следует определенному алгоритму. И хотя у программы Illustrator имеется множество инструментов, вы приучитесь работать только с теми из них, которые необходимы для решения стоящих перед вами задач.

Примеры, которые мы рассмотрим далее, касаются других типов диаграмм для визуализации темпоральных данных, и работа с ними в R и Illustrator более длительная. Однако мы пройдем их довольно быстро, так как вы уже владеете азами этих двух программ.

ПОДСКАЗКА

Всегда указывайте в диаграммах источник данных. Это не только повышает доверие к информации, но и обогащает контекст.

Сложите столбцы в штабеля

Как видно из рис. 4.20, конфигурация штабельных столбцовых диаграмм схожа с обычными столбцовыми диаграммами. Главное отличие, конечно, состоит в том, что прямоугольники уложены друг на друга в стеки. Штабельные столбцовые диаграммы используются тогда, когда есть некие подкатегории и сумма этих подкатегорий почему-либо имеет значение.

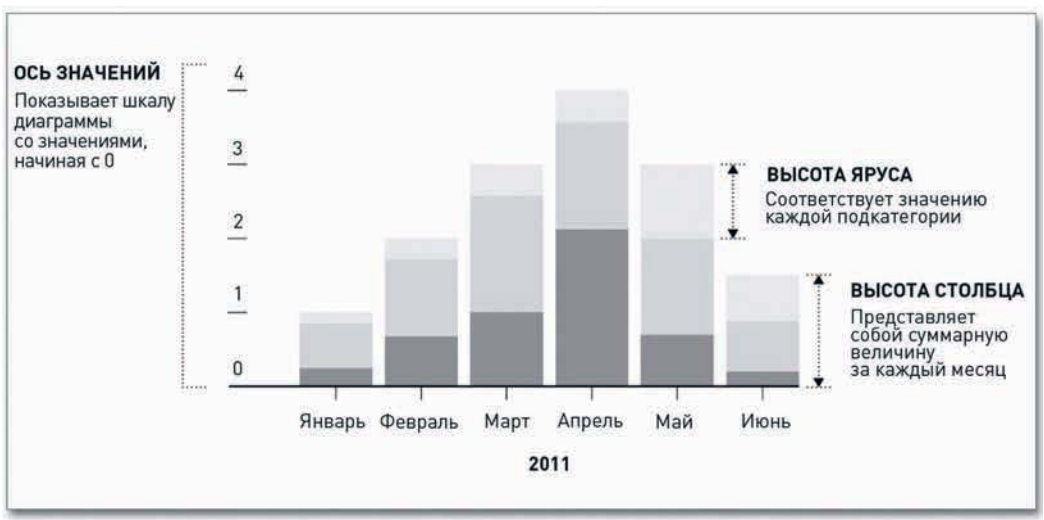


Рис. 4.20. Структура штабельной столбцовой диаграммы

Как и обычные столбцовые диаграммы, штабельные используются для визуализации не только темпоральных, но также и категорийных данных. Но в примере, представленном на рис. 4.20, категориями являются месяцы.

СОЗДАЙТЕ ШТАБЕЛЬНУЮ СТОЛБЦОВУЮ ДИАГРАММУ

Поскольку штабельные столбцовые диаграммы весьма распространены, существует множество способов их создания (не меньше, чем способов создания их нештабелированных родственников). Но вопрос тут вот в чем: как это делается в R? Алгоритм схож с тем, который вы использовали, выстраивая обычную столбцовую диаграмму. Ему и следуйте.

1. Загрузите данные.
2. Убедитесь, что данные правильно отформатированы.
3. Используйте функцию R, чтобы создать диаграмму.

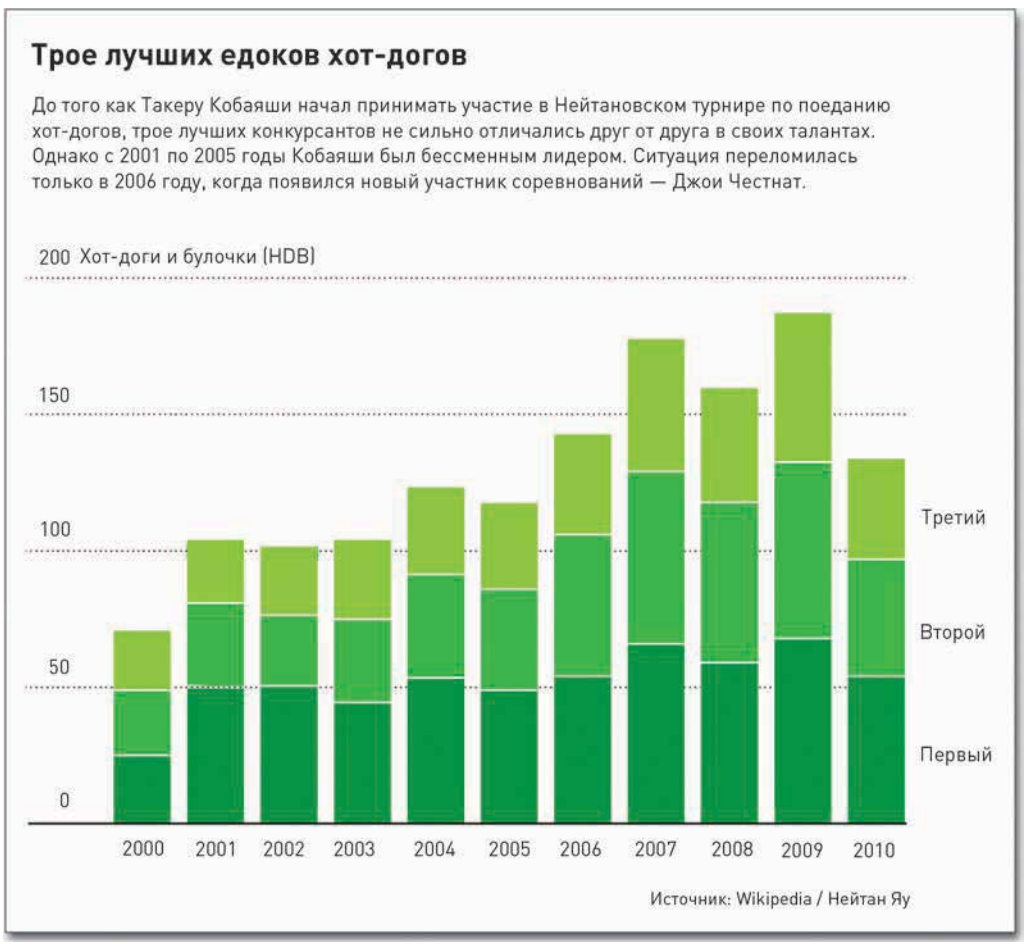


Рис. 4.21. Штабельная столбцовая диаграмма с результатами трех лучших едоков хот-догов

Это примерно то, что вам придется делать каждый раз, когда вы будете использовать R для создания информационной графики. Иногда на какой-либо из этапов уходит больше времени, чем на другие. Возможно, вы потратите изрядное время на перевод данных в правильный формат или напишете свою собственную функцию на R, чтобы получить именно то, чего хотите. В любом случае вы так или иначе будете проходить через эти три этапа. И, как вы убедитесь в следующих главах, переход на другие языки не меняет процедуры.

Однако вернемся к нашей штабельной столбцовой диаграмме и к Нейтановскому турниру по поеданию хот-догов. Вы смотрите на данные по поеданию хот-догов в последний раз, так что проникнитесь важностью момента. На рис. 4.21 представлена диаграмма, которую вам нужно сделать.

Вместо того чтобы смотреть лишь на количество съеденных победителями хот-догов и булочек, взгляните на результаты трех лучших едоков за каждый год. Каждый штабель (или стек) представляет один год, и в каждом содержится по три яруса — по числу лучших едоков. В «Википедии» можно найти соответствующие данные в полном объеме только с начала 2000-х годов, так что мы и начнем с этого момента.

Но давайте делать все по очереди. Сначала вам нужно загрузить данные в R. Вы можете получить к ним доступ прямо по URL с помощью следующих строчек:

```
hot_dog_places <-
   read.csv('http://datasets.flowingdata.com/hot-dog-places.csv',
   sep=",", header=TRUE)
```

Наберите **hot_dog_places**, чтобы вывести данные на экран. Каждая колонка показывает результаты за конкретный год, а каждая строчка — за места от первого по третье.

```
  X2000 X2001 X2002 X2003 X2004 X2005 X2006 X2007 X2008 X2009 X2010
1    25  50.0  50.5  44.5  53.5    49    54    66    59  68.0    54
2    24  31.0  26.0  30.5  38.0    37    52    63    59  64.5    43
3    22  23.5  25.5  29.5  32.0    32    37    49    42  55.0    37
```

Обратите внимание на то, что все названия колонок предваряются знаками «X». Они были добавлены по умолчанию в тот момент, когда вы загружали данные, потому что названия в шапке состоят из чисел. R это не нравится, вот он и добавляет буквы, чтобы сделать названия более похожими на слова или, говоря компьютерным языком, превратить их в *строку*. Названия в шапке понадобятся вам для подписей в штабельной столбцовой диаграмме, а потому лучше вернуть им первоначальный вид.

```
names(hot_dog_places) <- c("2000", "2001", "2002", "2003", "2004",
   "2005", "2006", "2007", "2008", "2009", "2010")
```

Используйте кавычки и оградите ими каждый год, чтобы уточнить, что это именно строка. Наберите снова **hot_dog_places**, теперь шапка должна появиться в нормальном виде — с годами.

```
  2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
1   25 50.0 50.5 44.5 53.5   49   54   66   59 68.0   54
2   24 31.0 26.0 30.5 38.0   37   52   63   59 64.5   43
3   22 23.5 25.5 29.5 32.0   32   37   49   42 55.0   37
```

Как и в предыдущем примере, вы будете использовать функцию `barplot()`, но данные у вас теперь в другом формате. Чтобы передать все верхние данные в `barplot()`, вам необходимо

конвертировать hot_dog_places в матрицу. Сейчас это таблица данных. В R таблица и матрица — разные структуры, но сейчас отличия не так важны. А что для нас важно, так это знать, как конвертировать таблицу данных в матрицу.

```
hot_dog_matrix <- as.matrix(hot_dog_places)
```

Только что созданная матрица была сохранена как hot_dog_matrix. Теперь можно передавать ее в barplot().

```
barplot(hot_dog_matrix, border=NA, space=0.25, ylim=c(0, 200),
   xlab="Year", ylab="Hot dogs and buns (HDB) eaten",
   main="Hot Dog Eating Contest Results, 1980-2010")
```

Эти строки определяют, что вокруг столбцов не будет границ, что расстояние между столбцами должно составлять 0,25 от ширины одного столбца и что значения на оси ординат будут в пределах от 0 до 200. Здесь же задаются заголовок и названия осей. Результат представлен на рис. 4.22.

Неплохой результат для каких-то нескольких строк кода. Теперь можно приступить к наведению лоска. Сохраните изображение как PDF и откройте его в программе Illustrator. Используйте те же самые инструменты, что и в предыдущем примере. Вы можете добавить немного текста с помощью инструмента Type (Текст), изменить шрифты, упростить вертикальную ось, отредактировать цвета, используя возможность выделять элементы с одинаковой заливкой. И не забудьте включить источник данных. Итог того, что вам предстоит сделать, показан на рис. 4.23.

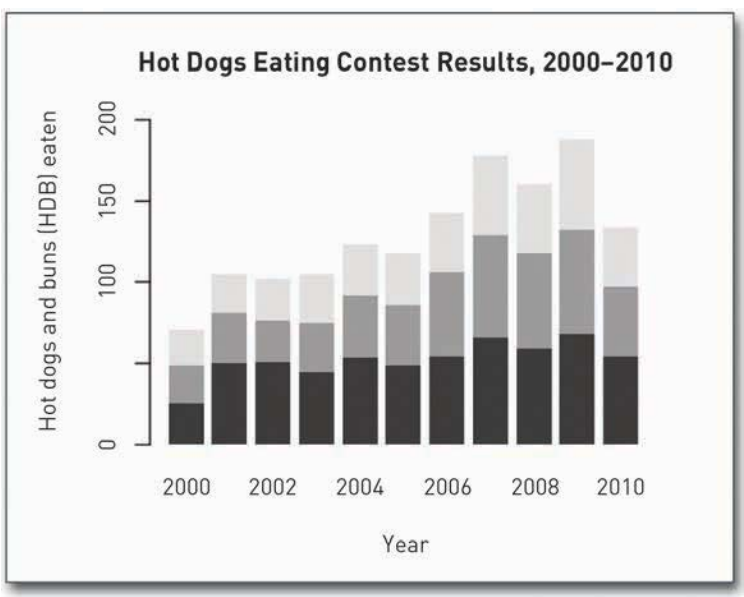


Рис. 4.22. Штабельная столбцовая диаграмма, созданная с использованием R

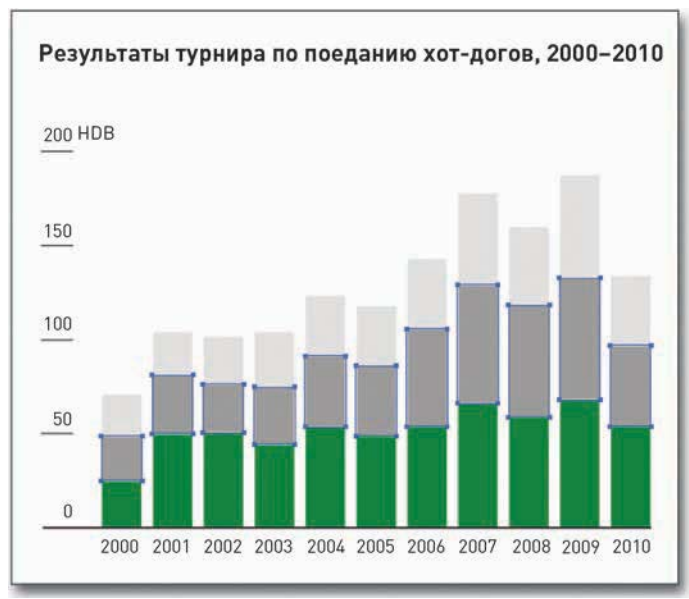


Рис. 4.23. Штабельная диаграмма, отредактированная в Illustrator

Добавьте небольшой вводный текст и измените заголовок на свой вкус. Конечный результат представлен на рис. 4.21.

Следующий раздел посвящен дальней родственнице штабельной столбцовой диаграммы — штабельной диаграмме с областями. Конфигурация у нее весьма похожая, нужно только представить, что вы объединили все штабеля в один непрерывный поток.

Точки

Иногда лучше использовать не столбцы, а точки. Они занимают меньше места и, поскольку нет брусков, более удачно создают ощущение непрерывного потока, переходящего от одной позиции к другой. На рис. 4.24 представлена стандартная конфигурация при использовании точек для визуализации темпоральных данных.

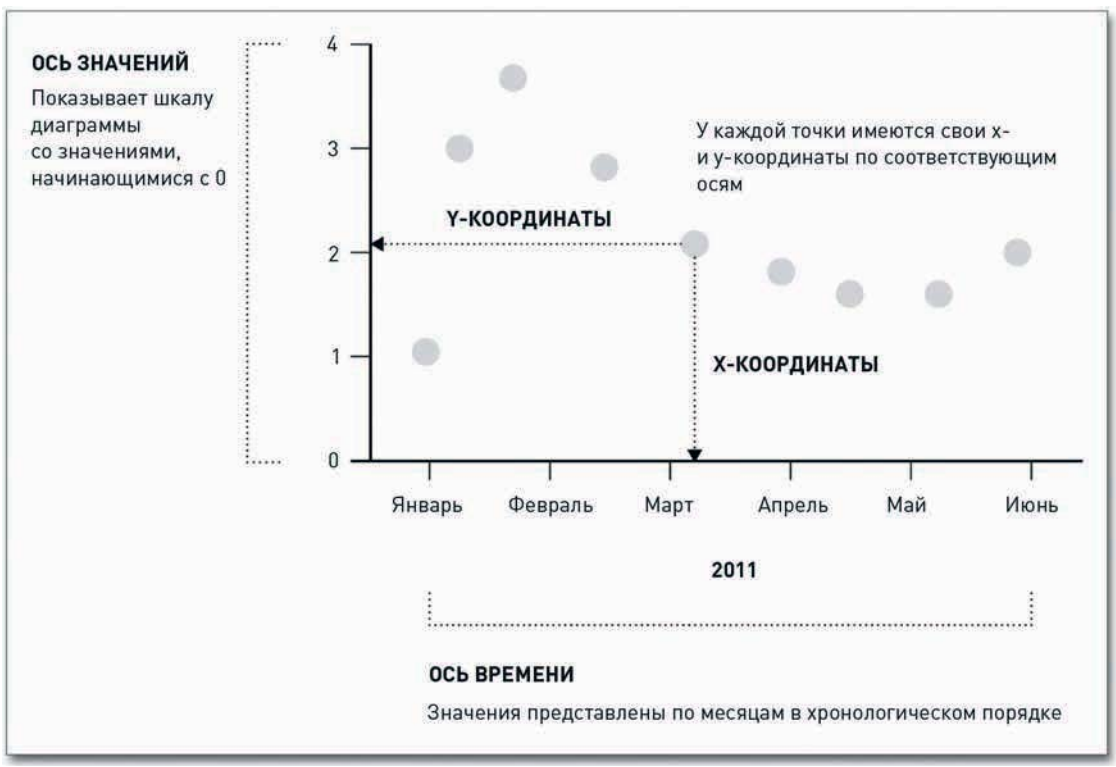


Рис. 4.24. Структура точечной диаграммы

Данный тип диаграмм известен больше как диаграмма рассеяния, и ее можно использовать для визуализации не только темпоральных данных. Нередко такую диаграмму применяют для демонстрации зависимости между двумя переменными, о чем мы подробнее поговорим

в шестой главе («Визуализация зависимостей»). В случае с темпоральными данными время тянется вдоль горизонтальной оси, а значения или измерения отмечаются на вертикальной.

В отличие от столбцовой диаграммы, где в качестве визуальной подсказки используется высота столбца, в диаграмме рассеяния для этого используют местоположение точек. У каждой из них есть свои *x*- и *y*-координаты, и вы можете сравнивать во времени одну точку с другими на основании того, где именно они располагаются.

Вот почему в диаграммах рассеяния ось значений не обязательно должна начинаться с нуля, хотя это и считается хорошей привычкой.

СОЗДАЙТЕ ДИАГРАММУ РАССЕЯНИЯ

Создать точечную диаграмму в R очень просто — через функцию `plot()`, но можно использовать и другие варианты, все зависит от данных, которыми вы располагаете. На рис. 4.25 представлен окончательный вариант диаграммы.

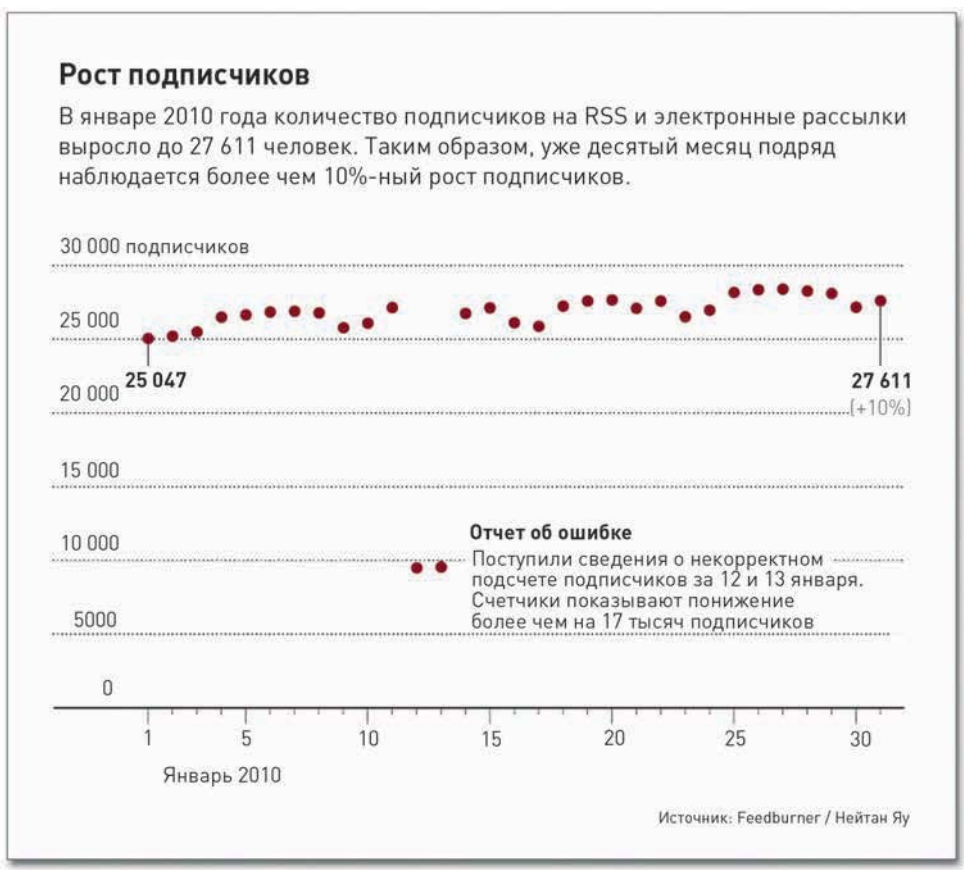


Рис. 4.25. Диаграмма рассеяния, созданная в R и обработанная в Illustrator

Перед вами история подсчета подписчиков FlowingData в январе 2010 года по данным сервиса Feedburner, отслеживающего, сколько людей ежедневно читают FlowingData. 1 января 2010 года подписчиков было 25 047 человек, а к концу месяца их число выросло до 27 611. Но, вероятно, наиболее интересная часть — это то, о чем стало известно в середине месяца. Неужели я действительно сказал нечто такое, что могло обидеть 17 тысяч читателей и заставить их разом отказаться от подписки? Маловероятно.

ПОДСКАЗКА

Данные сами по себе не есть факты. В них могла закрасться ошибка, могла произойти опечатка или что-то еще, из-за чего картина реальности получилась искаженной.

Вы помните, что необходимо делать в первую очередь, создавая диаграмму в R? Загрузить данные. Используйте `read.csv` и загрузите информацию о подписчиках (subscribers) прямо с URL.

```
subscribers <-
   read.csv("http://datasets.flowingdata.com/flowingdata_subscribers.csv",
   sep=",", header=TRUE)
```

Чтобы посмотреть первые пять рядов данных, введите:

```
subscribers[1:5,]
```

Вот как они выглядят:

```
      Date        Subscribers     Reach     Item.Views      Hits
  1 01-01-2010        25047         4627         9682        27225
  2 01-02-2010        25204         1676         5434        28042
  3 01-03-2010        25491         1485         6318        29824
  4 01-04-2010        26503         6290        17238        48911
  5 01-04-2010        26654         6544        16224        45521
```

У нас имеется пять колонок: с датой (Date), количеством подписчиков (Subscribers), охватом (Reach), просмотрами публикаций (Item.Views) и хитами (Hits). Нас интересуют только подписчики.

Вы можете включить также и дату, но подсчет и без того ведется в хронологическом порядке, так что для нанесения количества подписчиков на диаграмму первая колонка вам не нужна. Чтобы нарисовать точки, введите приведенную ниже строчку, и в результате вы получите диаграмму, как на рис. 4.26.

```
plot(subscribers$Subscribers)
```

Легко, не правда ли? Функция `plot()` на самом деле позволяет создавать несколько различных типов диаграмм, но по умолчанию рисуется именно диаграмма рассеяния. Сейчас вы использовали только данные о количестве подписчиков. Когда вы предоставляете функции `plot()` всего один массив данных, она решает, что этот массив содержит значения, и автоматически генерирует индекс для x-координат.

А теперь давайте зададим конкретно, какой именно тип диаграммы нам нужно получить, и установим диапазон значений вертикальной оси от 0 до 30 000.

```
plot(subscribers$Subscribers, type="p", ylim=c(0, 30000))
```

На рис. 4.27 представлены те же самые данные, что и на рис. 4.26, только диапазон значений вертикальной оси стал шире, как это и было задано в аргументе `ylim`. Обратите внимание на аргумент `type`, указывающий, что нужно использовать точки. Если вы поменяете `type` на `h`, тогда R нарисует плотные вертикальные линии.

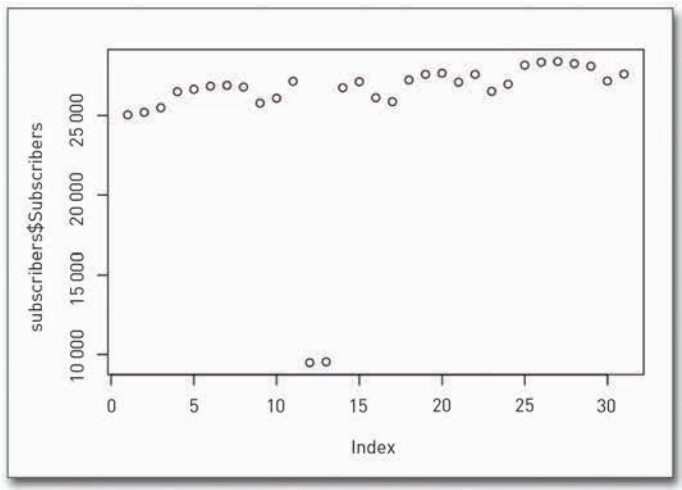


Рис. 4.26. Диаграмма по умолчанию, созданная в R

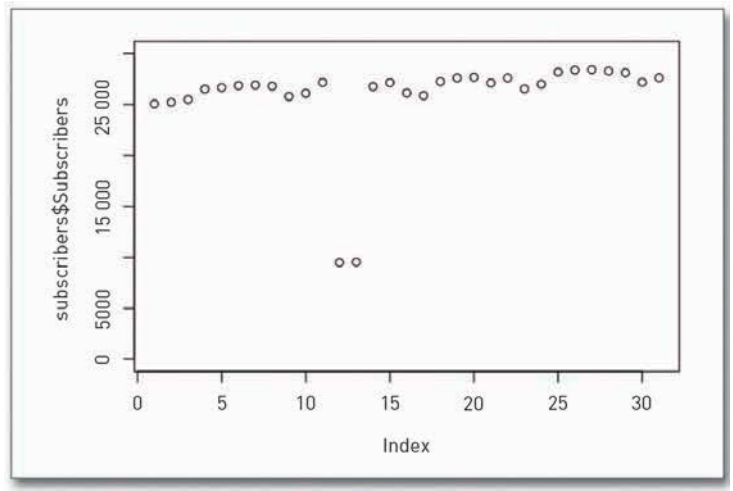


Рис. 4.27. Точечная диаграмма в R с заданными предельными значениями оси Y

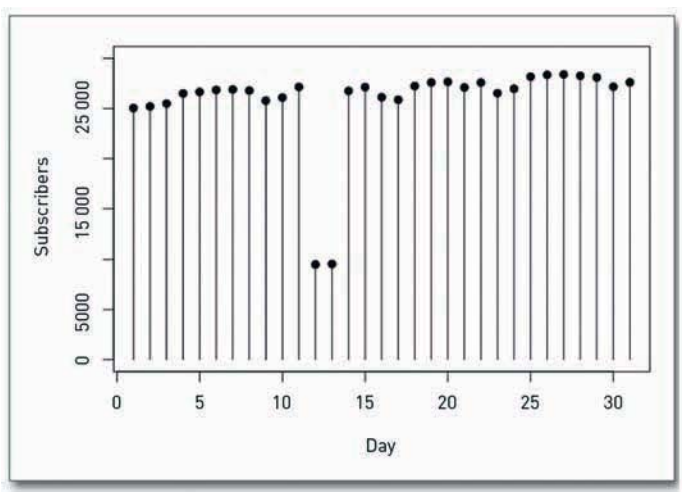


Рис. 4.28. Диаграмма с плотными вертикальными линиями

Вы можете использовать эти два варианта в комбинации, как показано на рис. 4.28. Однако вам понадобится применить также метод `points()`. Каждый раз при обращении к функции `plot()` вы станете создавать новую диаграмму, а не добавлять элементы к существующей. А вот как можно было бы сочетать вертикальные линии и точки, если б у вас вдруг возникло такое желание:

```
plot(subscribers$Subscribers, type="h", ylim=c(0, 30000),
   xlab="Day", ylab="Subscribers")
points(subscribers$Subscribers, pch=19, col="black")
```

Сначала чертится диаграмма с вертикальными линиями, на этот раз с названиями осей. Затем к существующей диаграмме добавляются точки. Аргумент `pch` отвечает за размер, а аргумент `col`, который мы уже использовали со столбцовой диаграммой, определяет цвет заливки.

Давайте вернемся к рис. 4.27. Сохраните диаграмму как PDF и откройте ее в Illustrator, чтобы заняться дизайном.

Используя инструмент Select (Выделение), выберите подписи и поменяйте шрифт в них на такой, какой вам нравится. Затем разгруппируйте их так, чтобы вы могли редактировать подписи по вертикальной оси по одной. Используйте Transform → Transform Each (Трансформировать → Трансформировать каждый), чтобы развернуть подписи по горизонтали. Затем возьмите инструмент Direct Selection (Частичное выделение) и удалите вертикальную линию оси — она вам не нужна, только место занимает.

И наконец, выделите собственно точки, которые представляют собой простые белые кружочки, и используйте опции в окне Color (Цвет), чтобы выбрать цвет для их обводки (Stroke) и заливки (Fill). Возможно, вам придется поменять схему цветов с Grayscale (шкалы полутонов серого) на CMYK (то есть Cyan — голубой, Magenta — пурпурный, Yellow — желтый, и black — черный цвет). Тогда выбор в окне Color (Цвет) станет богаче (рис. 4.29).

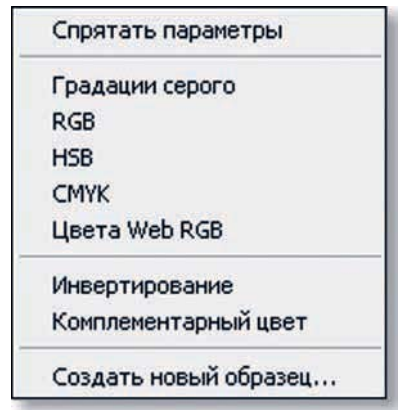


Рис. 4.29. Меню с выбором позиций в окне Color (Цвет)

В конце концов вы получите то, что изображено на к рис. 4.30. Это изображение уже в большей степени похоже на конечный вариант диаграммы.

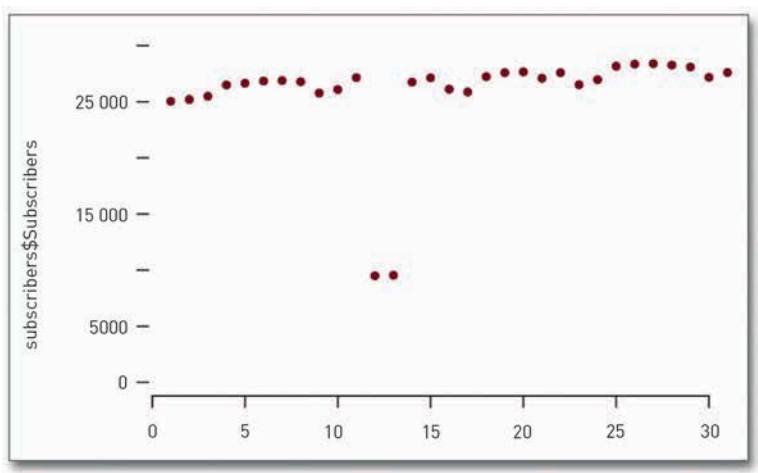


Рис. 4.30. Точечная диаграмма после редактирования вертикальной оси и цвета

А теперь попробуйте добавить сетку таким образом, чтобы стало легче увидеть, какие значения у точек в правом углу и как они соотносятся с предыдущими точками. С помощью инструмента Select (Выделение) выделите метки на оси значений, а затем щелкните мышью и протащите их так, чтобы растянуть через всю диаграмму. В таком виде они выглядят несколько грубовато, поэтому подправьте их стиль. Как и раньше, вы можете изменить стиль линий в опциях в окне Stroke (Обводка). Используйте опции, представленные на рис. 4.31, если хотите сделать линии тонкими и пунктирными.

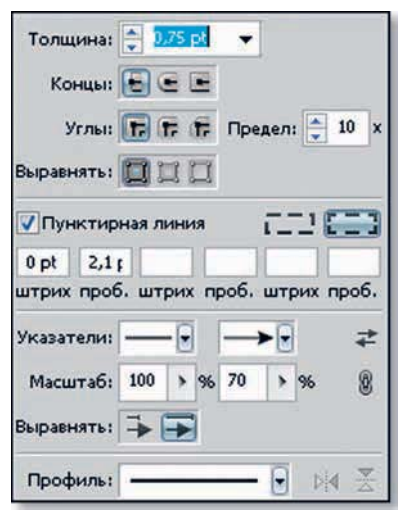


Рис. 4.31. Опции для пунктирных линий в окне Stroke (Обводка)

На рис. 4.32 показано что у вас должно получиться после внесения всех правок.

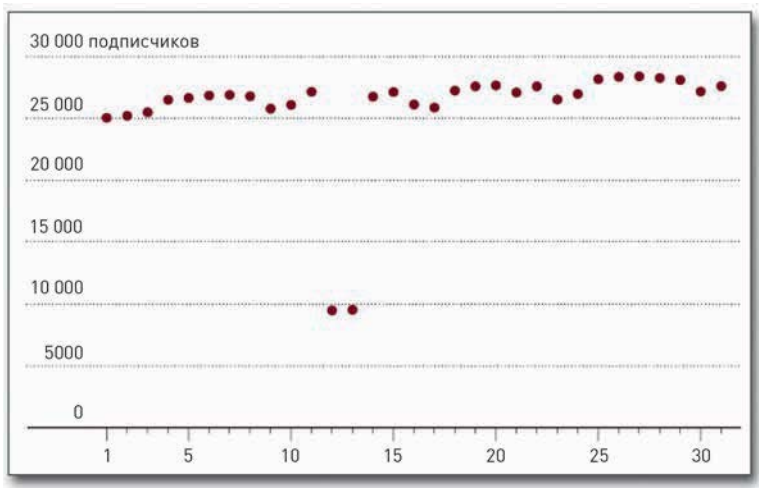


Рис. 4.32. Диаграмма с добавлением пунктирных линий и чисел на оси значений

Теперь, используя те же инструменты и приемы, придайте диаграмме, представленной на рис. 4.32, тот конечный вид, который она должна была обрести. С помощью инструмента Pen (Перо) нанесите метки по горизонтальной оси, а затем отредактируйте их и добавьте подписи, используя инструмент Type (Текст). Не забудьте указать внизу источник данных. После этого история будет готова.

Непрерывные данные

Визуализация непрерывных временны́х рядов не имеет принципиальных отличий от визуализации дискретных данных. В конечном счете вы все равно имеете дело с ограниченным числом точек ввода данных, даже если набор данных и относится к непрерывным. Структура в обоих случаях будет одинаковая. Отличие между ними состоит в том, какую часть материального мира они описывают. Как уже говорилось выше, непрерывные данные представляют постоянно меняющиеся явления, а потому вам необходимо визуализировать их таким образом, чтобы эта непрерывность была видна.

Соедините точки

Такое вам наверняка знакомо. Диаграмма временны́х рядов похожа на точечную, с той лишь разницей, что точки между собой соединены. Но часто бывает и так, что точек не видно. На рис. 4.33 представлена конфигурация распространенного типа диаграммы.

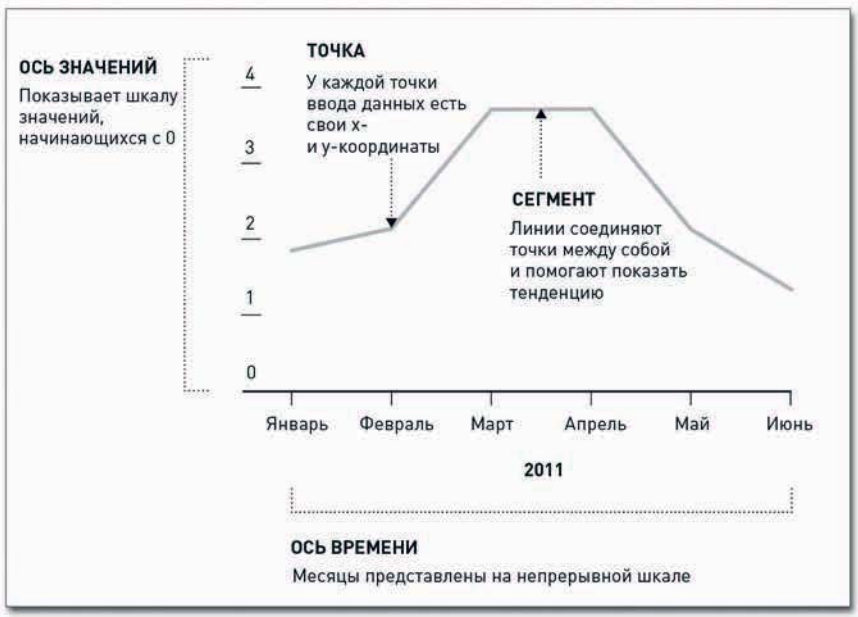


Рис. 4.33. Структура диаграммы временны́х рядов

На ней есть точки с *x*- и *y*-координатами и сегменты, или соединительные линии, которые помогают выявить в данных тенденции. Как правило, бывает полезно начинать ось значений с нуля, так как любая другая точка отсчета может исказить шкалу.

То, насколько вы растянете горизонтальную ось, также способно повлиять на отображение тенденций. Если сделаете ее слишком длинной, вы можете не заметить паттерны.

СОЗДАЙТЕ ДИАГРАММУ ВРЕМЕННО́ГО РЯДА

Если вы знаете, как делаются диаграммы рассеяния в R, значит, вы умеете создавать и диаграммы временны́х рядов. Загрузите ваши данные и примените функцию `plot()`, но в аргументе `type` вместо `p` используйте `l`, что означает «линия».

В качестве примера используйте данные Всемирного банка о народонаселении мира за 1960–2009 годы. Как обычно, загрузите данные с помощью функции `read.csv()`.

```
population <-
   read.csv("http://datasets.flowingdata.com/world-population.csv",
   sep=",", header=TRUE)
```

Ниже приводятся первые несколько рядов с данными. Здесь только год (Year) и население (Population).

```
     Year      Population
1    1960      3028654024
2    1961      3068356747
3    1962      3121963107
4    1963      3187471383
5    1964      3253112403
```

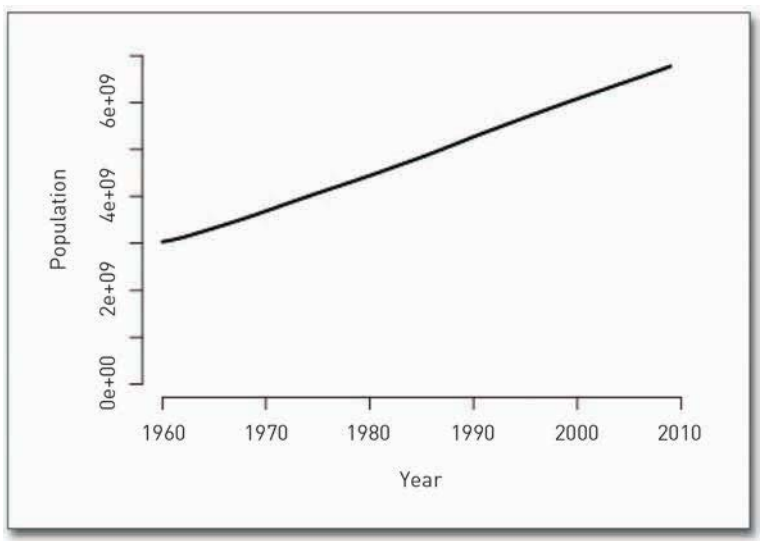


Рис. 4.34. Диаграмма временно́го ряда по умолчанию в R

Используйте функцию `plot()` и задайте *x*- и *y*-координаты, тип диаграммы, диапазон оси значений и названия осей.

```
plot(population$Year, population$Population, type="l",
   ylim=c(0, 7000000000), xlab="Year",
   ylab="Population")
```

Вы получите диаграмму, похожую на ту, которая представлена на рис. 4.34.

На этом этапе вы уже можете сохранить изображение в формате PDF и отредактировать его в программе Illustrator, как вы делали с предыдущими двумя диаграммами, однако давайте-ка в этот раз попробуем кое-что новенькое, а именно: начертим всю диаграмму в Illustrator с помощью инструмента Line Graph (Линейная диаграмма). Это один из нескольких инструментов в Illustrator, дающих возможность создавать основные типы диаграмм и графиков (рис. 4.35).

Рис. 4.35. Инструменты для создания диаграмм в Illustrator

Для начала в окне Tools (Инструменты) выберите Line Graph (Линейная диаграмма). Для этого сначала найдите среди инструментов иконку с изображением какого-нибудь типа диаграммы. Щелкните по иконке и удерживайте кнопку мыши нажатой, чтобы выбрать нужный тип графического объекта.

Затем загрузите данные о народонаселении из `http://datasets.flowingdata.com/world-population.csv`. В Adobe Illustrator вы не можете загрузить данные напрямую с URL, как вы это делали в R, а потому вам необходимо сохранить файл на своем компьютере. Откройте CSV-файл в Excel или Google Documents, чтобы получить таблицу, похожую на ту, что представлена на рис. 4.36. Скопируйте все строчки за исключением шапки таблицы (то есть ее верхней строки). Эти данные вам нужно вставить в Illustrator.

Вернитесь обратно к Illustrator. Используя инструмент Line Graph (Линейная диаграмма), который вы уже выбрали в окне Tools (Инструменты), щелкните мышью и перетащите курсор по диагонали, очертив прямоугольную область, примерно равную по размеру тому графику, который вы хотите получить в конечном итоге. Появится таблица вроде той, что показана на рис. 4.37.

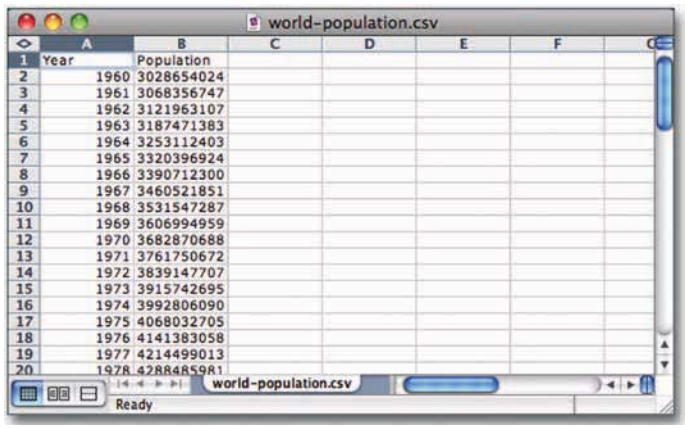


Рис. 4.36. CSV-файл, открытый в Excel

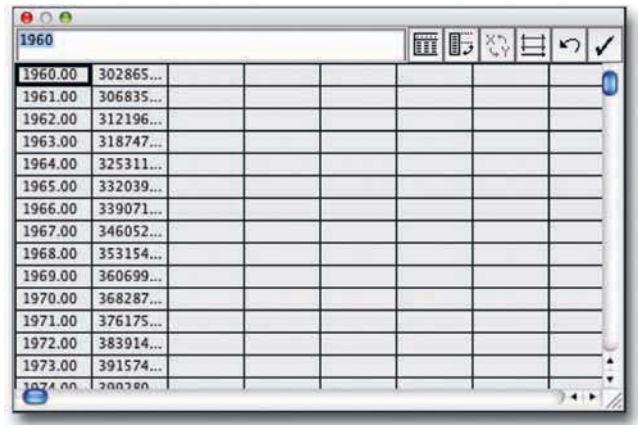


Рис. 4.37. Таблица для ввода данных в Illustrator

Вставьте в таблицу данные, которые вы скопировали в Excel, а затем щелкните по кнопке с галочкой в верхнем правом углу. Вы увидите изображение как на рис. 4.38.

Вот вы и получили базовый вариант диаграммы, теперь вам нужно изменить некоторые настройки, чтобы линейная диаграмма не выглядела такой неаккуратной. Щелкните правой кнопкой мыши и выберите Type (Тип). Снимите флажок Mark Data Points (Пометить точки), как это показано на рис. 4.39.

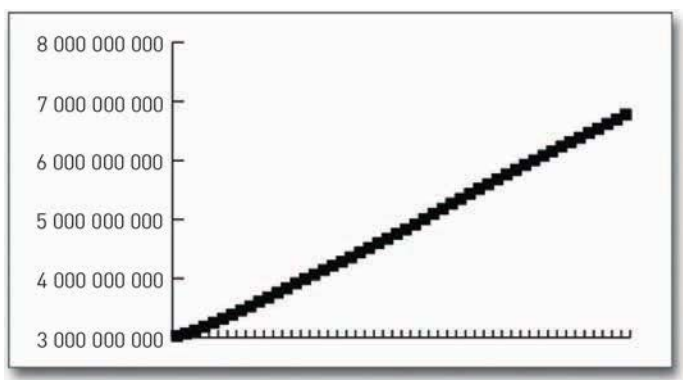


Рис. 4.38. Линейный график, создаваемый по умолчанию в Illustrator

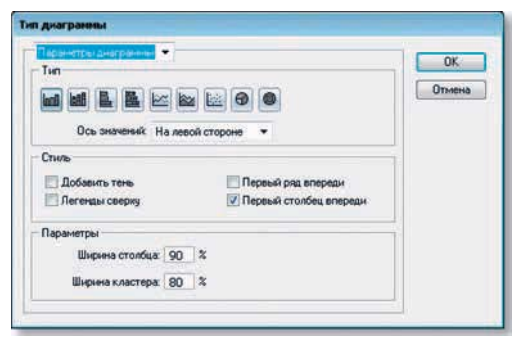


Рис. 4.39. Варианты диаграмм в Illustrator

ПОДСКАЗКА

Если вы собираетесь создавать диаграмму какого-нибудь базового типа и затем редактировать ее в Illustrator, возможно, вы сэкономите немного времени, если нарисуете ее в этой программе с нуля. Вам не обязательно все делать сначала в R. Точно так же вам не обязательно все делать сразу в Illustrator.

Из выпадающего меню выберите ось категорий (Category Axis) и в опциях для длины (Length) делений (Tick marks) выберите None (Не создавать). Нажмите OK. Таким образом вы получите более чистое и менее перегруженное изображение. А далее следуйте тому же алгоритму, что вы использовали, редактируя диаграммы, созданные в R.

Вы можете удалить вертикальную ось, упростить подписи со значениями, вставить метки и пояснения на горизонтальной оси с годами и добавить заголовок и немного текста. Вы можете также изменить стиль, чтобы линия больше выделялась. Светло-серый, который задается по умолчанию, зачастую создает впечатление, что данные являются каким-то фоном, хотя на самом деле они должны занимать наиболее престижные места — быть на первом плане в центре. Когда вы внесете все эти изменения, у вас должна получиться диаграмма, похожая на рис. 4.40.

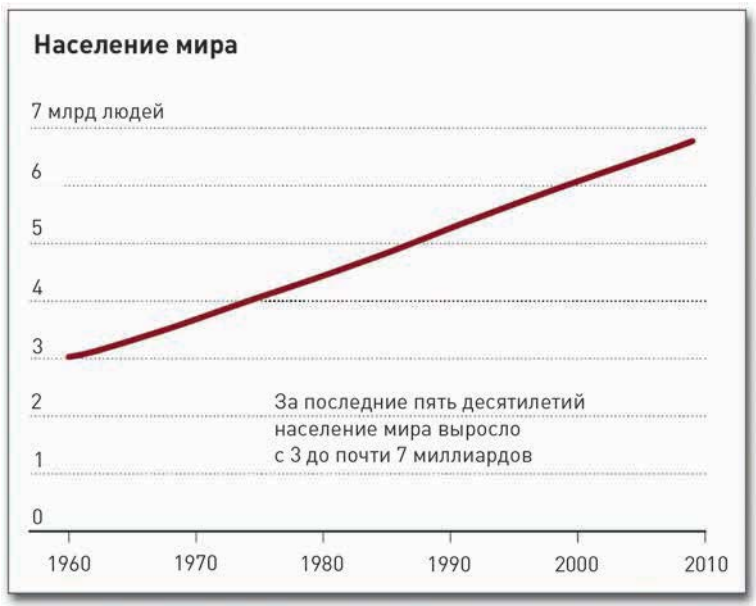


Рис. 4.40. Изменение численности населения мира за последние пять десятилетий

Самый важный вывод здесь — что вы можете сделать одну и ту же диаграмму и в Illustrator, и в R — и в обоих случаях вы получите одно и то же. А потому решайте сами, каким инструментом вам удобнее пользоваться, и действуйте. Главное — получить результат.

Поднимитесь на ступеньку

Один из недостатков стандартного линейного графика состоит в том, что он внушает мысль о стабильном движении от точки A к точке B. Иногда все так и обстоит, например, когда мы имеем дело с ростом населения Земли. Но в других случаях после длительного времени

пребывания в неизменном состоянии вдруг происходит резкий скачок, неважно, вверх или вниз. Процентные ставки, например, могут держаться на одном уровне на протяжении многих месяцев, а затем за один день упасть. Для такого типа данных лучше использовать ступенчатый график, показанный на рис. 4.41.

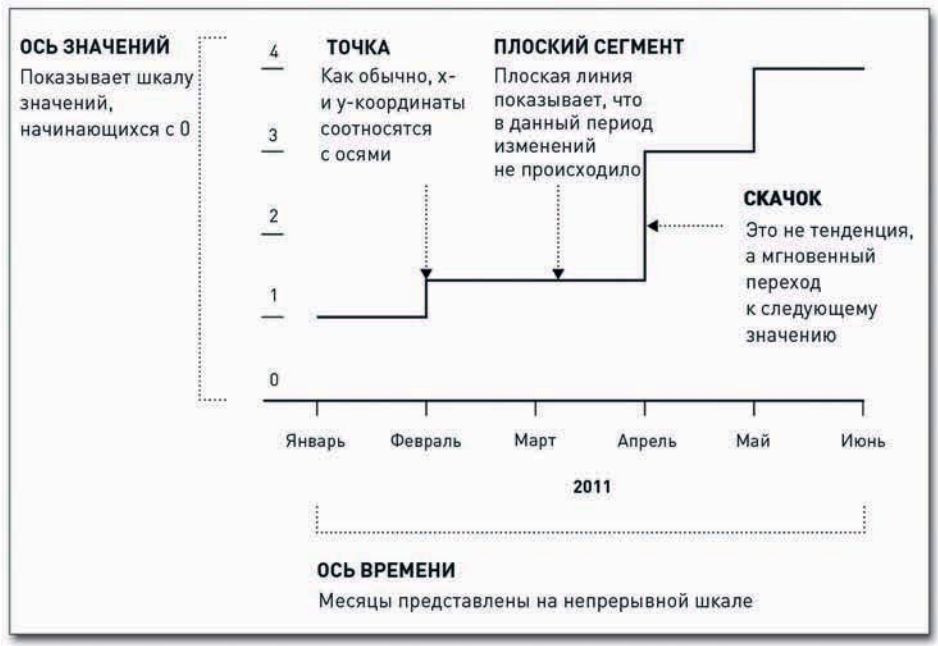


Рис. 4.41. Базовая структура ступенчатого графика

Вместо того чтобы соединять точки A и B напрямую, кратчайшим путем, линия остается на уровне одного значения, пока не происходит перемена. В точке перемены линия прыгает вверх (или вниз) к следующему значению. В итоге вы получаете последовательность ступеней.

СОЗДАЙТЕ СТУПЕНЧАТУЮ ДИАГРАММУ

В программе Illustrator нет инструмента для простого создания ступенчатых графиков. Зато в R такой инструмент есть. Вы можете создать базовый вариант диаграммы в R, а затем отредактировать в Illustrator. Вы уже начинаете улавливать в этой последовательности некую закономерность?

На рис. 4.42 представлен конечный вариант графика. Он показывает, как менялись тарифы на пересылку писем, отправляемых через Почтовую службу США.

С 1995 до 1999 года цена держалась на уровне 32 центов. То есть четыре года подряд она не менялась. А в последнее время — с 2006 по 2009 годы — изменения в тарифы вносились ежегодно.

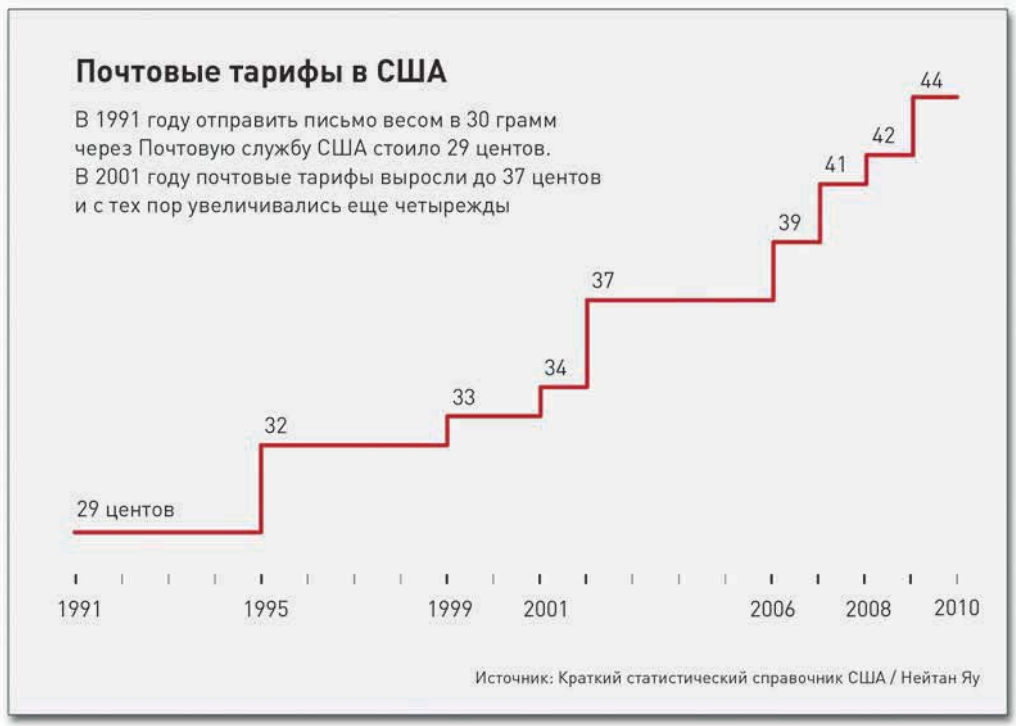


Рис. 4.42. Изменение почтовых ставок

Чтобы создать ступенчатую диаграмму в R, следуйте тому же алгоритму, который вы использовали во всех примерах в этой главе:

1. Загрузите данные.
2. Убедитесь, что данные правильно отформатированы.
3. Используйте функцию R, чтобы создать диаграмму.

Исторические данные о почтовых тарифах (наряду с множеством других сведений) вы можете почерпнуть из Краткого статистического справочника США. Я ввел их в CSV-файл, который можно найти по адресу `http://datasets.flowingdata.com/us-postage.csv`. Вставьте этот URL в `read.csv()` в качестве источника, чтобы загрузить файл с данными в R.

```
postage <- read.csv("http://datasets.flowingdata.com/us-postage.csv",
   sep=",", header=TRUE)
```

Ниже приводится весь набор данных. Он состоит всего из десяти пунктов, по одному на каждое изменение почтовых тарифов с 1991 по 2009 годы, а последний пункт показывает текущий уровень. В первой колонке идет год, во второй — тариф в долларах США.

```
   Year  Price
1  1991   0.29
2  1995   0.32
3  1999   0.33
4  2001   0.34
5  2002   0.37
6  2006   0.39
7  2007   0.41
8  2008   0.42
9  2009   0.44
10 2010   0.44
```

Функция `plot()` существенно облегчает создание ступенчатого графика. Как вы, наверное, и ожидали, в качестве *x*-координаты необходимо ввести год (Year), в качестве *y*-координаты — цену (Price), а аргументом `type` указать `s`, что, конечно, означает «ступенчатый».

```
plot(postage$Year, postage$Price, type="s")
```

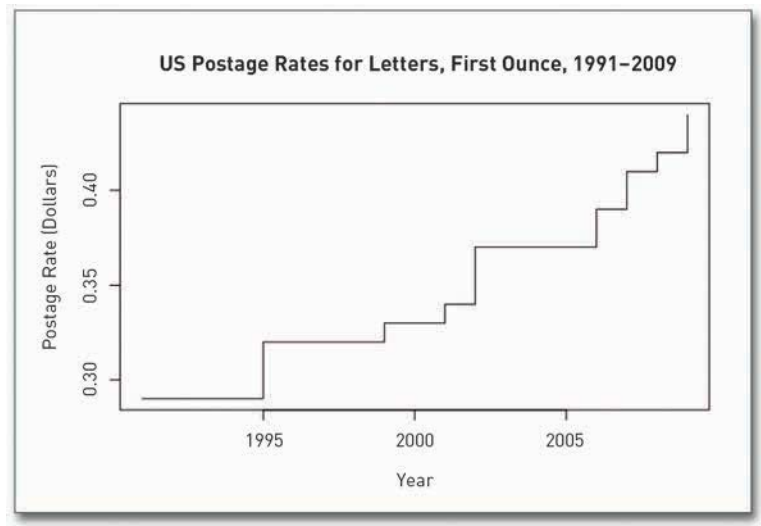


Рис. 4.43. Ступенчатая диаграмма, созданная в R

Кроме того, вы можете ввести заголовок и названия осей — какие захотите. В качестве заголовка я ввел «US Postage Rates for Letters, First Ounce, 1991–2010» («Почтовые тарифы на письма в США, за первые 30 граммов, 1991–2010»), а в качестве названий осей «Year» («Год») и « Postage Rate (Dollars)» («Почтовый тариф, в долларах»).

```
plot(postage$Year, postage$Price, type="s",
   main="US Postage Rates for Letters, First Ounce, 1991-2010",
   xlab="Year", ylab="Postage Rate (Dollars)")
```

В результате я получил ступенчатую диаграмму почтовых тарифов, изображенную на рис. 4.43.

Забавы ради посмотрите, как выглядело бы изображение, если б вы сделали график не ступенчатым, а линейным (рис. 4.44).

Тенденция повышения налицо, но вы обратили внимание, как она выглядит? Как постоянный рост. В период с 2001 по 2006 год тариф ни разу не составлял 38 центов, но вы об этом так и не узнаете, если будете смотреть только на линейный график и не обратитесь к исходным данным.

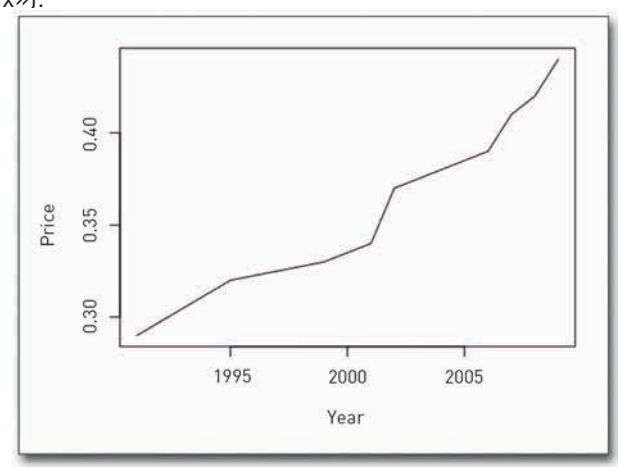


Рис. 4.44. Линейный график почтовых тарифов

ПОДСКАЗКА

Когда вы имеете дело с небольшим набором данных, бывает полезно вместо оси значений и сетки поставить подписи непосредственно рядом с точками на графике. Таким образом вы еще больше привлечете внимание к данным, а поскольку точек не так много, изображение не получится перегруженным и запутанным.

Сохраните изображение в формате PDF, а затем откройте его в программе Illustrator. Следуйте тому же алгоритму, который вы использовали ранее, и отредактируйте ступенчатую диаграмму на свой вкус. Лично я начисто избавился от вертикальной оси и с помощью инструмента Type (Текст) добавил подписи прямо к каждому скачку. Еще на оси времени я разместил метки с равным интервалом, но подписал только те годы, в которые происходили перемены.

В самом конце я сделал фон серым. Это результат моих личных предпочтений, но фон еще и помогает выделить диаграмму, особенно когда она размещена внутри текста. Таким образом изображение начинает казаться больше, но при этом не становится крикливым и раздражающим. Чтобы оформить фон, находящийся за линией и текстом, и не залить краской все, вам необходимо создать в Illustrator новый слой. Это вы можете сделать — правильно, где же еще?! — в окне Layers (Слои). Щелкните по кнопке, чтобы создать новый слой. По умолчанию он будет помещен сверху, но вам необходимо, чтобы он оказался внизу, а потому щелкните и перетащите новый слой под Layer 1 (Слой 1), как это показано на рис. 4.45.

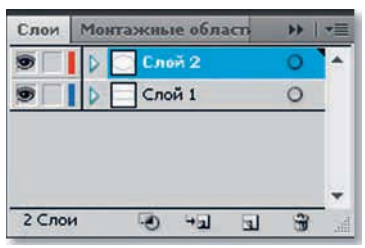


Рис. 4.45. Окно Layers (Слои) в Illustrator

Вы можете переименовывать слои, это будет особенно полезным, когда вы начнете создавать более сложные диаграммы и графики; новому слою я дал название «background» («фон»). Используя инструмент Rectangle (Прямоугольник), начертите соответствующую фигуру. Щелкните мышью и потяните прямоугольник за край, чтобы задать нужный размер, а потом измените цвет в окне Color (Цвет).

Сглаживание и оценка

Порой, когда приходится работать с большими массивами данных или когда данные сопровождаются «шумом», бывает довольно сложно разглядеть в них тенденции и паттерны. Чтобы облегчить задачу, вы можете оценить линию тренда. Общий смысл такого рода действий выражен на рис. 4.46.

Начертите линию, проходящую через максимально возможное количество точек, таким образом, чтобы суммарное суммарное расстояние от точек до подобранной линии было минимальным. Самый прямолинейный подход — это создать прямую подобранную линию, используя базовое уравнение с угловым коэффициентом, которое вы наверняка проходили в средней школе:

$$y = mx + b,$$

где m — это угловой коэффициент прямой, а b — величина отрезка. Что происходит, когда тенденция нелинейна? Нет смысла подгонять прямую линию к данным, которые демонстрируют только повороты и изгибы. Вместо этого лучше воспользоваться статистическим методом, разработанным Уильямом Кливлендом (William Cleveland) и Сьюзен Дэвлин (Susan Devlin) и названным «локально взвешенным сглаживанием диаграммы рассеяния» (ЛВСДР). Этот

метод позволяет подогнать кривую под ваши данные.

ЛВСДР стартует с исходной точки данных и делает маленькие срезы. На каждом срезе определяется низшая степень полинома для данных только из этого среза. Таким образом ЛВСДР передвигается далее по данным, выстраивая серию миниатюрных кривых, которые складываются вместе, формируя одну кривую. Дополнительную информацию вы можете найти в Google. По этой теме вышло уже несколько публикаций. А пока давайте поговорим о том, как вы можете использовать ЛВСДР в своей работе с данными.

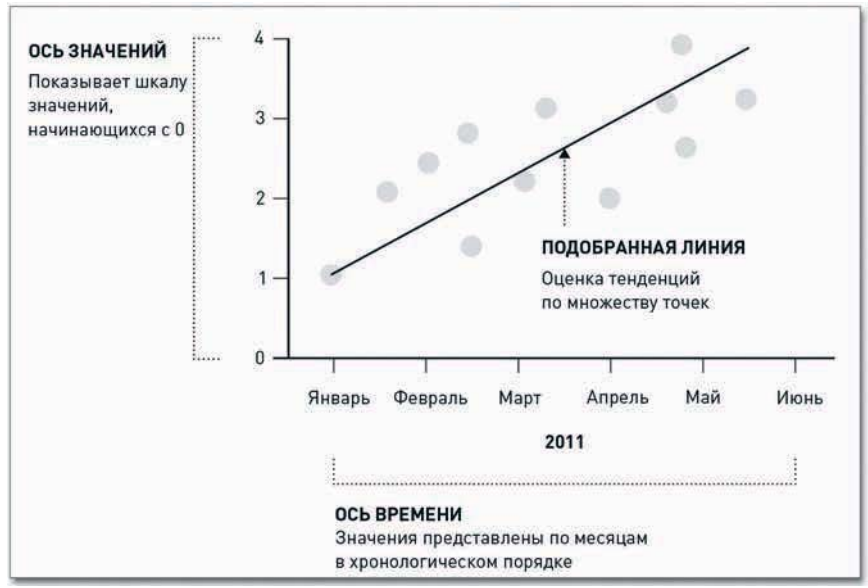


Рис. 4.46. Подгонка линии по точкам ввода данных

ПОДБЕРИТЕ КРИВУЮ МЕТОДОМ ЛВСДР

История, на которую вы сейчас смотрите, — это история безработицы в Соединенных Штатах за последние несколько десятилетий. Были в ней и взлеты, и падения, и сезонные колебания. Но как выглядит общая тенденция? Вы можете видеть на рис. 4.47, что до своих максимальных значений уровень безработицы доходил в 1980-х годах, затем на протяжении 1990-х снижался, а потом около 2008 года снова взлетел.

▶ Для более подробного знакомства с методом ЛВСДР посмотрите статью У. Кливленда «Робастная локально взвешенная регрессия и сглаживание диаграмм рассеяния» в Journal of the American Statistical Association.

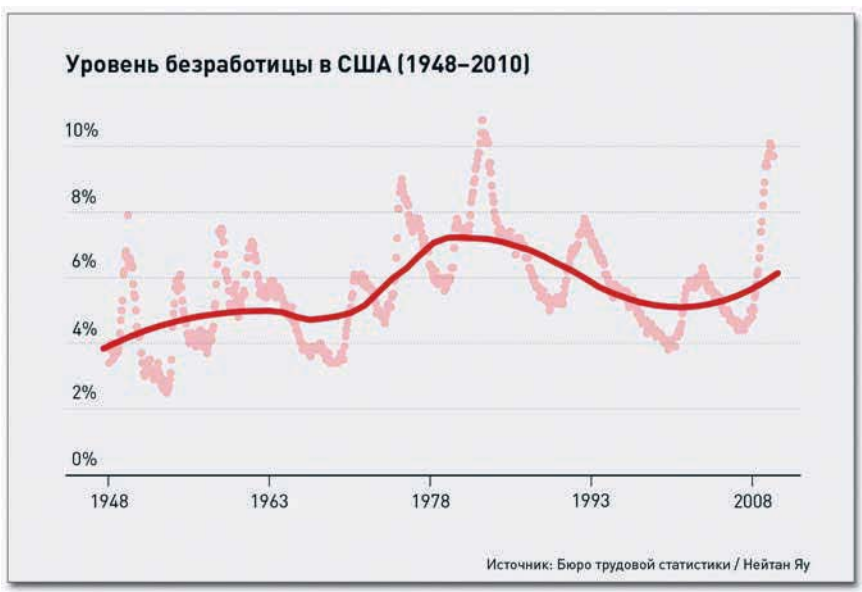


Рис. 4.47. Уровень безработицы со сглаженной методом ЛВСДР кривой

На рис. 4.48 показано, как выглядит диаграмма уровня безработицы, состоящая только из точек и составленная в R с применением функции `plot()`.

```
# Load data
unemployment <-
   read.csv(
     "http://datasets.flowingdata.com/unemployment-rate-1948-2010.csv",
     sep=",")
unemployment[1:10,]

# Plain scatter plot
plot(1:length(unemployment$Value), unemployment$Value)
```

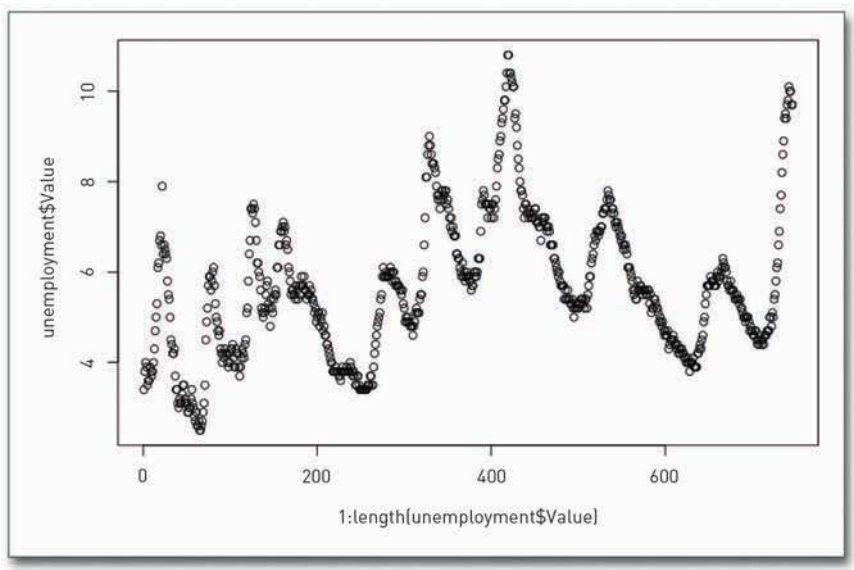


Рис. 4.48. Диаграмма уровня безработицы с использованием точек

А вот как выглядела бы прямая подобранная линия (рис. 4.49).

Пользы от нее мало. Создается впечатление, что она начисто игнорирует все колебания в уровне безработицы. Поэтому, чтобы создать кривую методом ЛВСДР, необходимо использовать функцию `scatter.smooth()`.

```
scatter.smooth(x=1:length(unemployment$Value), y=unemployment$Value)
```

Полученный результат вы можете увидеть на рис. 4.50. Теперь линия изогнута с учетом резкого подъема уровня безработицы в 1980-х годах. Это уже чуть лучше.

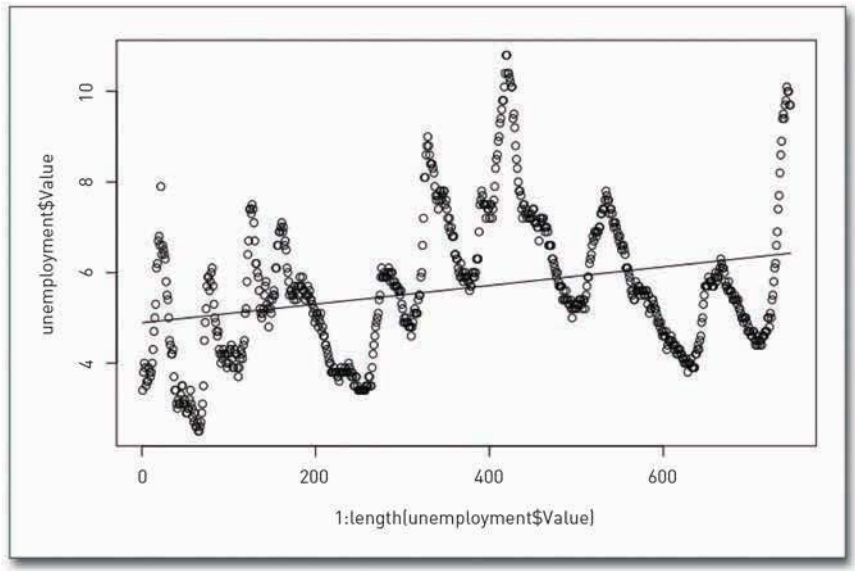


Рис. 4.49. Прямая подобранная линия

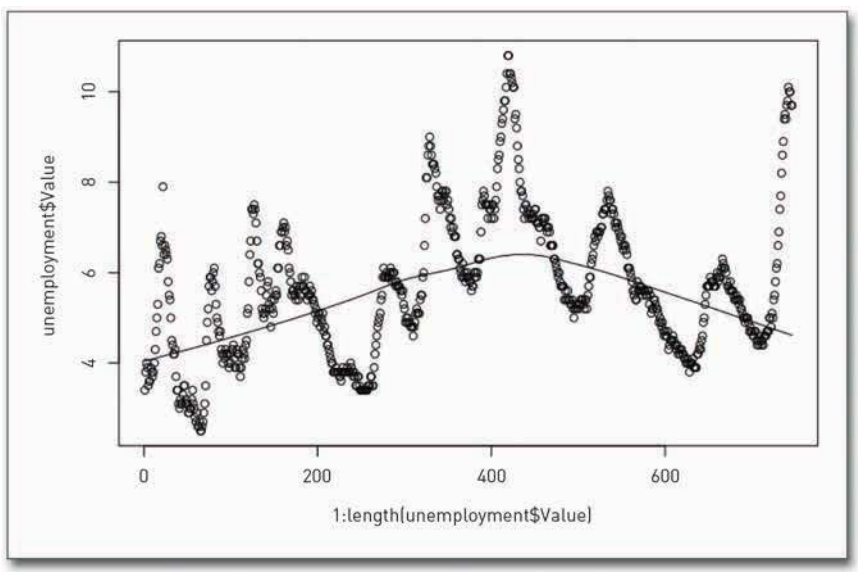


Рис. 4.50. Подбор кривой, созданной методом ЛВСДР

То, насколько сглаженной должна получиться кривая, можно задать через аргументы `degree` (степень) и `span` (интервал) в функции `scatter.smooth()`. Первый из них контролирует степень полиномов, а второй — то, насколько гладкой получится кривая. Чем ближе интервал к нулю,

тем точнее будет подгонка. На рис. 4.51 показано, что́ именно вы получите, если поменяете степень на 2, а интервал — на 0,5. В конце измени́те цвета и настройте интервал значений осей.

```
scatter.smooth(x=1:length(unemployment$Value),
   y=unemployment$Value, ylim=c(0,11), degree=2, col="#CCCCCC", span=0.5)
```

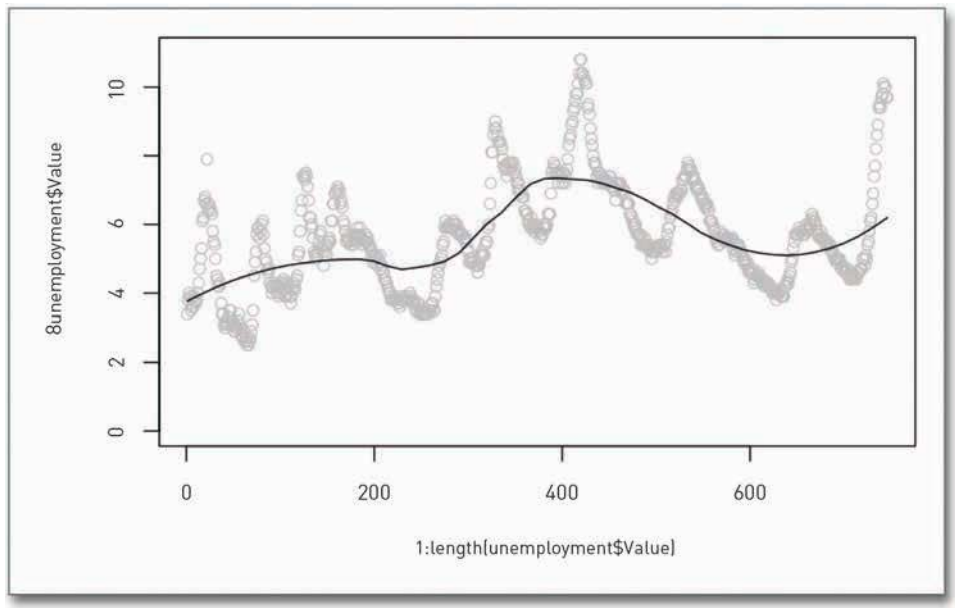


Рис. 4.51. Сглаженная кривая по методу ЛВСДР с более низкой степенью сглаженности и более высокой степенью полинома

С такими настройками колебания уровня безработицы на полученной кривой уже более заметны. Попробуйте поиграть с интервалом, чтобы прочувствовать, как он меняет степень сглаженности кривой.

Чтобы получить конечный вариант диаграммы, представленный на рис. 4.47, сохраните изображение как PDF и откройте его в Illustrator. Используйте те же самые инструменты — Selection (Выделение), Type (Текст), Pen (Перо), — чтобы оформить заголовок, фон и метки на горизонтальной оси. Сглаженная линия сделана более яркой, чтобы привлекать внимание к тенденции, а не к отдельным точкам данных.

Закругляясь

Изучать паттерны во времени очень интересно. Время является неотъемлемой частью нашей повседневной жизни и так тесно переплетено с ней, что многие подходы к визуализации

темпоральных данных вполне понятны на интуитивном уровне. Вы знаете, что все меняется и развивается, и сложность здесь состоит в другом — определить степень этих изменений и научиться понимать, что именно следует искать в своих диаграммах.

Это нетрудно — бросить взгляд на диаграмму и сказать, что в чем-то наблюдается рост и что это хорошо. Собственно, для того и существует визуализация: она позволяет быстро составить общее впечатление о картине, складывающейся из массива данных. Но вы можете пойти дальше. Вы можете использовать визуализацию в качестве исследовательского инструмента. Рассмотреть в приближении те или иные отрезки времени и задаться вопросом: почему в такой-то день произошел маленький всплеск, какого больше ни разу не наблюдалось, или почему в другой день произошел такой-то перепад. Вот тогда данные станут интересными и увлекательными — а чем больше вы знаете о своих данных, тем лучше получится у вас история, которую собираетесь рассказать.

Когда вы разберетесь, о чем на самом деле говорят ваши данные, тогда и поясните эти детали в инфографике. Выделите интересные элементы, чтобы ваши читатели поняли, на что смотреть. Может, диаграмма без подписей говорит вам о многом, но для остальных без контекста она будет безликой.

Чтобы добиться достойных результатов, вы использовали R и Illustrator. В R вы создавали основу, а Illustrator применяли для оформления диаграмм, чтобы показать, что именно важно в собранных данных. Рассмотренные нами типы диаграмм представляют собой, конечно, только небольшое подмножество того, что вы можете делать с темпоральными данными. Вы откроете еще целый набор хитроумных приемов, когда присовокупите к уже освоенным методам анимацию и интерактив — с ними вы познакомитесь в следующей главе. Но и с переходом к новому типу данных — к пропорциям — вы сможете применять тот же процесс программирования и те же принципы дизайна, которые использовали в этой главе, даже когда будете создавать коды на других языках.

Визуализация пропорций

Временны́е ряды самым естественным образом группируются по... времени. Череда событий происходит в определенных временных рамках. Данные о пропорциях также группируются, но по категориям, подкатегориям и совокупностям. Говоря слово «совокупность», я не имею в виду лишь совокупность людей. Совокупность в данном случае — скорее, все многообразие вариантов принятия решений или вариантов исхода. Это территория выборок.

При проведении опросов людей часто спрашивают, каково их мнение по определенной проблеме — положительное, отрицательное или нейтральное. Каждая из этих категорий представляет собой некую часть чего-то, а сумма частей складывается в целое.

Эта глава посвящена тому, как можно представить отдельные категории, но при этом не потерять из виду и картину целиком и показать, как каждый из вариантов соотносится с остальными. Здесь вы сможете применить на практике часть всего того, что узнали из предыдущих глав, и впервые попробуете создать интерактивную графику, используя HTML, CSS и JavaScript, а в конце познакомитесь и с Flash.

Что искать в пропорциях

В случае с пропорциями обычно ищут три параметра: минимум, максимум и распределение в целом. Первые два выявляются с ходу. Выстройте свои данные по порядку от наименьшего к наибольшему, возьмите те значения, что находятся на двух концах, — вот вам и минимум с максимумом. Если бы вы имели дело с результатами опроса, это могло бы означать наиболее и наименее популярные ответы участников. А если бы вы изучали калорийность разных компонентов блюда, то выяснили бы, что именно вносит наибольшую лепту в общую калорийность яства, а что — наименьшую.

Однако для демонстрации минимума и максимума вам не нужна диаграмма. Самое интересное — это распределение пропорций. Как соотносится группа людей, сделавшая тот или иной выбор, с остальными? Как распределяются калории между жирами, белками и углеводами — поровну, или одна из этих групп доминирует? Следующие типы диаграмм способны помочь вам при решении подобных задач.

Части целого

Части целого — это самый простой вид пропорций. У вас есть набор пропорций, которые в сумме составляют единицу, или набор процентных показателей, которые складываются в 100 процентов. В таких случаях вы обычно хотите показать, как именно отдельные части относятся к другим частям, но при этом стараетесь сохранить ощущение целостности.

Круг

Круговые диаграммы — старый испытанный вариант. И сегодня их можно увидеть повсюду — от бизнес-презентаций до сайтов, использующих их как материал для шуток. Первая круговая диаграмма, о которой доподлинно известно, была опубликована в 1801 году Уильямом Плейфэром (William Playfair), человеком, который придумал также линейный график и столбцовую диаграмму. Толковый был парень.

Вы знаете, как работает данный тип диаграмм. Как показано на рис. 5.1, все начинается с круга, который представляет собой целое, а затем вы нарезаете его на дольки (сектора), как если бы он был пирогом. Каждая такая долька является частью целого. Не забывайте об этом, так как многие новички нередко допускают такую ошибку. Сумма всех долек должна составлять 100 процентов. Если сумма получается отличной от этого числа, значит, вы в чем-то ошиблись.

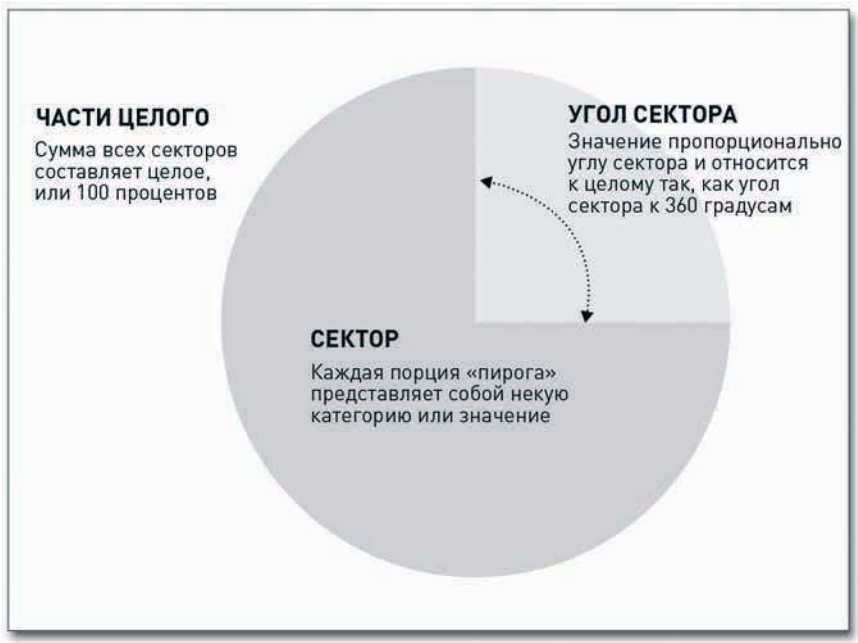


Рис. 5.1. Круговая диаграмма в обобщенном виде

Круговые диаграммы часто клеймят за то, что они не так точны, как столбцовые диаграммы или линейные графики, а потому некоторые люди думают, что их лучше совсем избегать. Оценить на глаз высоту намного проще, чем измерить площадь или угол. Но это не означает, что вам следует отказаться от этого типа диаграмм.

Вы можете использовать их без проблем до тех пор, пока будете помнить об определенных ограничениях, связанных с их применением. Все очень просто: держите свои данные в порядке и не разбивайте круг на слишком большое количество секторов.

СОЗДАЙТЕ КРУГОВУЮ ДИАГРАММУ

Несмотря на то что практически все графические программы позволяют создавать круговые диаграммы, вы можете, как и в предыдущей главе, сделать свою диаграмму в Illustrator. Процесс введения данных, создания диаграммы по умолчанию и дальнейшего ее «облагораживания» должен быть уже знаком вам.

Выстроить базовый вариант диаграммы — собственно круг — дело довольно простое. После того как вы создадите новый документ, из окна Tool выберите инструмент Pie Graph (Круговая диаграмма), как это показано на рис. 5.2. Щелкните и протащите прямоугольник, чтобы он получился примерно такого размера, какого вы хотите видеть свою диаграмму. Позже размер можно будет изменить.

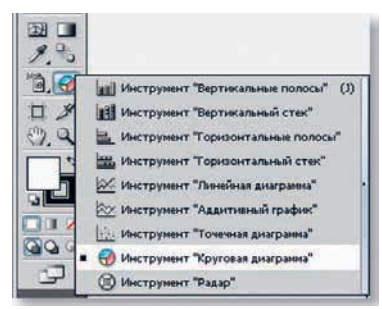


Рис. 5.2. Окно инструментов в Illustrator

Когда вы отпустите кнопку мыши, появится окно с таблицей, куда вы можете поместить ваши данные. Чтобы создать круговую диаграмму, вводите точки данных в строгой последовательности слева направо — на диаграмме значения появятся в том же порядке.

В качестве примера используйте результаты опроса, проведенного на сайте FlowingData. Читателей спрашивали, какая область, связанная с данными, их интересует больше всего. Был получен 831 ответ.

Сфера интересов	Количество голосов
Статистика (Statistics)	172
Дизайн (Design)	136
Бизнес (Business)	135
Картография (Cartography)	101
Информатика (Information Science)	80
Веб-аналитика (Web Analytics)	68
Программирование (Programming)	50
Инженерное дело (Engineering)	29
Математика (Mathematics)	19
Другое (Other)	41

ПОДСКАЗКА

Таблица в Illustrator предельно упрощенная, так что вам будет нелегко в ней манипулировать своими данными и перестраивать их на свой вкус. Один из способов решить проблему — это выполнить все операции с данными в Microsoft Excel, а затем скопировать результат и вставить его в Illustrator.

Введите числа в таблицу в программе Illustrator, как показано на рис. 5.3. Порядок введения чисел будет соответствовать порядку, в котором сегменты выстроятся на вашей круговой диаграмме, начиная сверху и далее по часовой стрелке.

Рис. 5.3. Таблица в Illustrator

Обратите внимание, что результаты опроса организованы от наибольшего к наименьшему, а заканчивается все категорией «Другое». Данный способ сортировки должен повысить читабельность вашей диаграммы. Когда вы закончите, щелкните по кнопке с галочкой в верхнем правом углу окна с таблицей.

Перед вами появится круговая диаграмма, по умолчанию имеющая черную обводку и раскрашенная восемью оттенками серого,

выстроенными в случайном порядке (рис. 5.4). Диаграмма немного похожа на серый леденец на палочке, но ситуацию нетрудно исправить. Важнее всего то, что у вас уже есть базовый вариант.

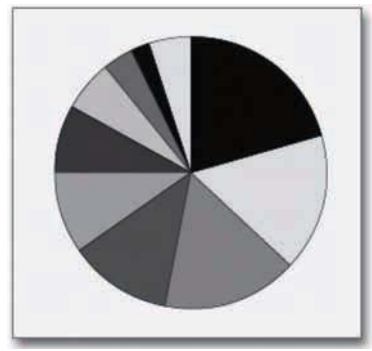


Рис. 5.4. Круговая диаграмма по умолчанию

А теперь сделайте круговую диаграмму более читаемой, изменив некоторые цвета и добавив текст, чтобы объяснить людям, на что они смотрят. В том виде, в каком диаграмма находится сейчас, цвета́ мало о чем говорят. Они служат лишь для разграничения сегментов, а вы можете использовать их для того, чтобы подсказать читателям, на что смотреть и в каком порядке.

Если вы начнете с 12 часов и пойдете далее по часовой стрелке, вы увидите данные в порядке убывания. Однако поскольку цветовая схема выстроена по случайному принципу, некоторые из секторов меньшего размера оказались выделены более темными тонами. Темные тона действуют как маркеры, а потому вам следует сделать наоборот: крупные сектора окрасить в более насыщенные цвета, а сектора поменьше — в более светлые. Если по той или иной причине вы захотите привлечь внимание к ответам, которые дало меньшее количество респондентов, тогда, возможно, вам стоит поступить наоборот. В этом случае, однако, мы хотим знать, какие сферы деятельности, связанные с работой с данными, оказались наиболее популярными.

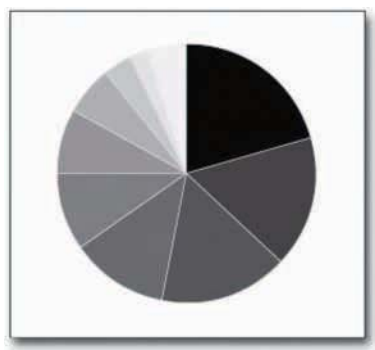


Рис. 5.5. Круговая диаграмма, цвета в которой выстроены от темного к светлому

В окне инструментов выберите Direct Selection (Частичное выделение) и затем кликните по одному из секторов. Через элементы управления в окне Color (Цвет) измените параметры заливки и обводки. На рис. 5.5 представлена та же самая круговая диаграмма, только с белой обводкой и с секторами, окрашенными в оттенки серого от наиболее темного к наиболее светлому. Теперь уже легко заметить, что сектора выстроены по размеру от наибольшего к наименьшему, за исключением последнего сектора — «Другое».

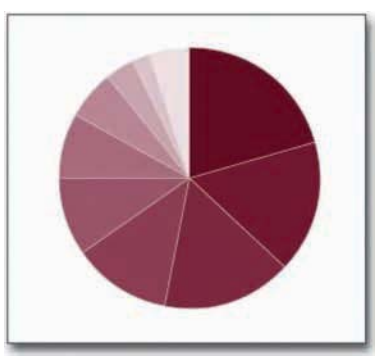


Рис. 5.6. Окрашенная круговая диаграмма

Конечно, вам не обязательно пользоваться цветами настолько экономно. Вы можете взять такие цвета, какие захотите (пример на рис. 5.6). Однако принято считать, что пользоваться яркими цветами не очень хорошая идея — вы же не хотите, чтобы у ваших читателей стало рябить в глазах. Впрочем, если тема предусматривает подобный эффект, тогда не стесняйтесь.

Поскольку данный опрос проводился на сайте FlowingData, я решил использовать тот вариант красного, который присутствует в логотипе FlowingData, а более светлые оттенки для других секторов получил, изменив непрозрачность (opacity). Эту опцию вы найдете в окне Transparency

ПОДСКАЗКА

Цвет может играть важную роль в том, как люди воспримут вашу диаграмму. И это не вопрос одной лишь эстетики, хотя иногда бывает и так. Цвет способен служить визуальной подсказкой точно так же, как длина или площадь, так что вдумчиво подходите к его выбору.

ПРИМЕЧАНИЕ

Когда вы используете непрозрачность, цвет заливки того сегмента, параметры которого вы меняете, смешивается с цветом фона. В данном случае фон белый, и чем выше будет прозрачность, тем более блеклый цвет вы получите. А вот окажись фон синим, сектор приобрел бы фиолетовый оттенок.

(Прозрачность). При 0 процентов непрозрачности цвет заливки проявится полностью, а при 100 процентах заливки не будет виден вовсе.

В конце, пользуясь инструментом Type (Текст), добавьте заголовок, вводное предложение и впишите названия осей. По мере накопления опыта вы начнете лучше разбираться в том, какие шрифты больше или меньше подходят для заголовков и для текста. Но какие бы шрифты вы ни выбрали, когда дело дойдет до размещения текста, лучшего друга, чем инструмент Alignment (Выравнивание), в Illustrator вам не найти. Правильно выровненный текст и равноотстоящие подписи сделают вашу диаграмму гораздо более читабельной. Вы также можете воспользоваться инструментом Pen (Перо), чтобы снабдить последние три сектора указателями, как показано на рис. 5.7. Эти сектора слишком маленькие, чтобы разместить подписи внутри них, и слишком прижаты друг к другу, чтобы можно было расположить подписи рядом.

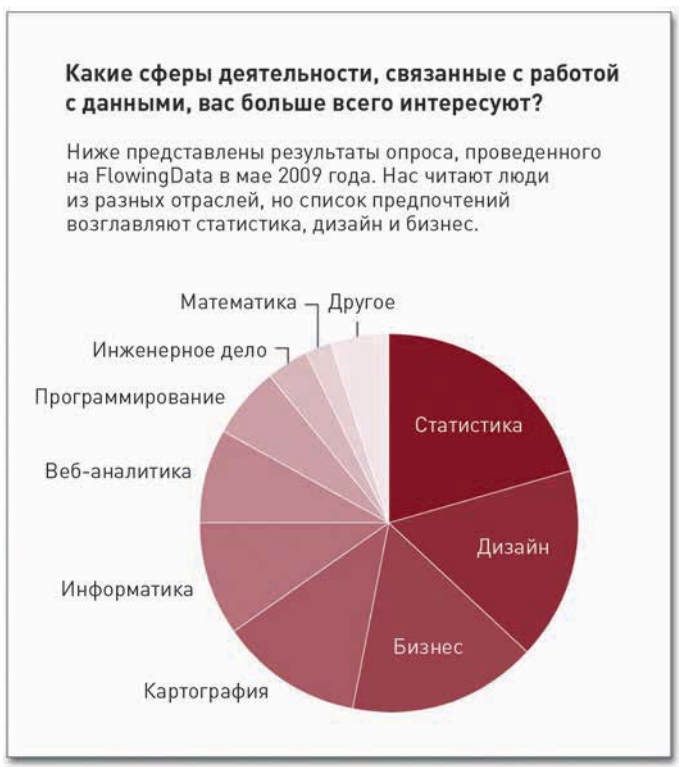


Рис. 5.7. Окончательный вариант круговой диаграммы с подписями и вводным текстом

Кольцо

У вашей хорошей подруги, круговой диаграммы, есть младшая кузина: кольцевая диаграмма. Она похожа на круговую, только с дыркой в центре, как у бублика (см. рис. 5.8).

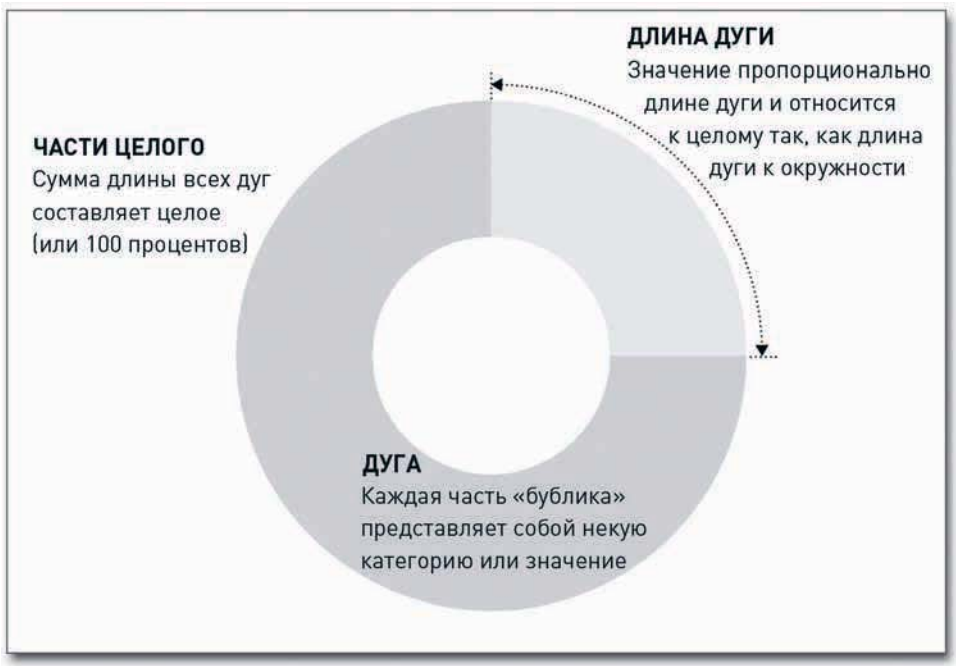


Рис. 5.8. Структура кольцевой диаграммы

Поскольку в середине есть дырка, мы уже не используем угол в качестве меры. Вместо него для оценки значения применяется длина дуги. А потому, в случае, если вам придется работать с большим количеством категорий, вы столкнетесь с немалой частью прежних проблем. А вот если вы имеете дело с небольшим количеством категорий, тогда кольцевая диаграмма может сослужить вам хорошую службу.

СОЗДАЙТЕ КОЛЬЦЕВУЮ ДИАГРАММУ

Создать кольцевую диаграмму в Illustrator очень просто. Сперва создайте круговую диаграмму, как вы это уже делали. А затем вставьте в центр залитый цветом круг, как показано на рис. 5.9. И, опять-таки, используйте цвета, которые будут направлять взгляд читателя в нужную сторону.

Во многих случаях центр кольцевой диаграммы используется для размещения подписи или другого текста, как сделано и в этом случае.

ПОДСКАЗКА

При использовании круговой или кольцевой диаграмм очень важно постоянно помнить о том, что их легко перегрузить. Они не предназначены для визуализации большого количества значений.

А теперь давайте создадим такую же диаграмму, используя Protovis — бесплатный инструмент для визуализации с открытым исходным кодом. Protovis — это JavaScript-библиотека, позволяющая использовать возможности современных браузеров для работы с масштабируемой векторной графикой (Scalable Vector Graphics, SVS). Поскольку графические объекты генерируются динамически, это дает возможность делать их анимированными и интерактивными. А потому Protovis — отличный выбор для создания онлайн-графики.

▶ Загрузите Protovis с `http://vis.stanford.edu/protovis` и разместите в той же самой директории, которую используете для сохранения файлов с примерами.

Хотя вы собираетесь погрузиться в другой язык программирования, вы по-прежнему будете следовать той же модели, что и тогда, когда работали с R и Illustrator. Первым делом загрузите данные, затем постройте базовый вариант диаграммы и наконец, придайте ей эстетичный вид.

На рис. 5.10 показано, к чему вы будете стремиться. Диаграмма на ней очень похожа на ту, которая представлена на рис. 5.9, с той лишь разницей, что подписи размещены под углом, и когда указатель мыши оказывается поверх того или иного сектора, вы видите, сколько именно человек проголосовало за данную категорию. Интерактив может быть и более продвинутым, но прежде чем давать волю фантазии, необходимо освоить азы.

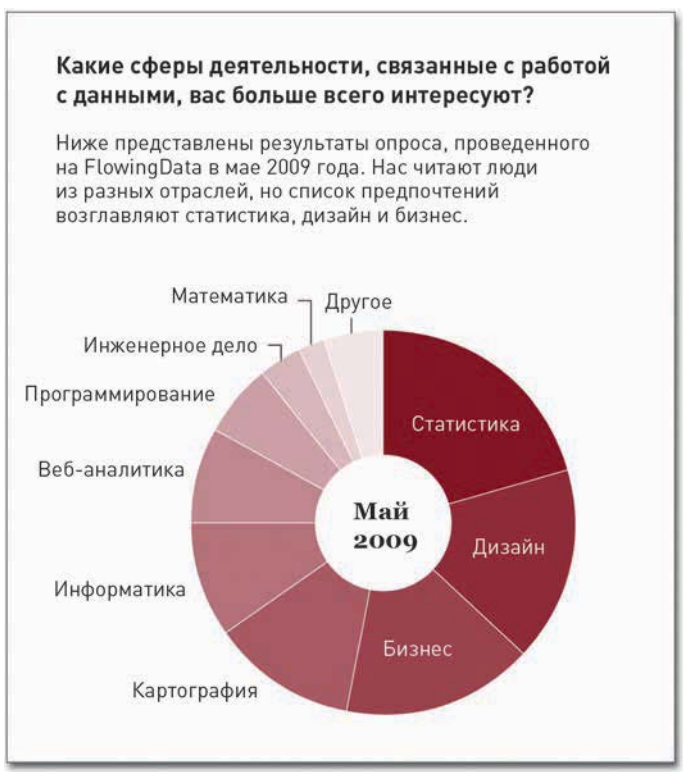


Рис. 5.9. Из круговой диаграммы — в кольцевую

Рис. 5.10. Кольцевая диаграмма, созданная с помощью Protovis

Первым делом вам необходимо создать HTML-страницу. Назовите ее **donut.html**.

```
<html>
<head>
   <title>Donut Chart</title>
   <script type="text/javascript" src="protovis-r3.2.js"></script>
   <style type="text/css">
      #figure {
         width: 400px;
         height: 400px;
      }
   </style>
</head>
<body>
   <div id="figure ">

   </div><!-- @end figure -->
</body>
</html>
```

Если вы когда-либо прежде создавали веб-страницы, то вам не надо ничего объяснять. А для тех, кто никогда не делал ничего такого, скажу, что это основы HTML, и подобное вы найдете в сети буквально повсюду. Каждая страница начинается с тега `<html>`. За ним следует тег `<head>`, в котором содержится информация о странице, но который не отображается в окне вашего браузера. А вот все, что заключено в теге `<body>`, вполне видимо. В нашем примере страница получила название **`Donut Chart`** («Кольцевая диаграмма»), и на нее была загружена библиотека Protovis — это один JavaScript-файл — с помощью тега `<script>`. Далее идет фрагмент CSS для определения стиля страницы. Чем он проще — тем лучше. С помощью `id "figure"` ширина и высота блока `<div>` устанавливаются равными 400 пикселам. Это то пространство, где будет располагаться диаграмма. Приведенный фрагмент HTML, по сути, не является частью диаграммы, но он нужен, чтобы следующий далее код JavaScript правильно загрузился в веб-браузер. Если вы сейчас откроете файл `donut.html` в браузере, то увидите лишь чистую страницу.

Внутри `<div>` нужно внести уточнение о том, что код, который вы собираетесь написать, будет на JavaScript. Все остальное пойдет внутри тега `<script>`.

```
<script type="text/javascript+protovis">
</script>
```

Хорошо. Но давайте все делать по очереди. Сначала — данные. Перед вами результаты опроса, проведенного на сайте FlowingData, которые записаны в виде двух последовательностей. Данные о числе голосов сохранены в одном массиве, а соответствующие им категории — в другом.

```
var data = [172,136,135,101,80,68,50,29,19,41];
var cats = ["Statistics", "Design", "Business", "Cartography",
```

```
   "Information Science", "Web Analytics", "Programming",
   "Engineering", "Mathematics", "Other"];
```

Далее нужно задать ширину и высоту кольцевой диаграммы, а также радиус и шкалу для определения длины дуг.

```
var w = 350,
   h = 350,
   r = w / 2,
   a = pv.Scale.linear(0, pv.sum(data)).range(0, 2 * Math.PI);
```

Высота и ширина кольцевой диаграммы будут по 350 пикселов, а радиус (то есть расстояние от центра диаграммы до ее внешней кромки), равный половине ширины, составит 175 пикселов. Четвертая строка задает шкалу дуги. Расшифровывается она следующим образом: настоящие данные представлены на линейной шкале от нуля до суммы всех голосов, и эту шкалу необходимо преобразовать в шкалу кольца, которая имеет диапазон от 0 до 2π радиан (или от 0 до 360 градусов, если вам так проще).

Затем идет цветовая шкала. Чем больше голосов получила та или иная категория, тем насыщеннее должен быть красный цвет. В Illustrator вы делали это вручную, но Protovis способен подобрать цвета за вас. Вам только нужно определить, какую цветовую гамму вы предпочитаете.

```
var depthColors = pv.Scale.linear(0, 172).range("white", "#821122");
```

Теперь у вас есть цветовая шкала от белого до темно-красного (#821122), соответствующая линейному диапазону от 0 до 172 (наибольшего количества голосов, набранного одной категорией). Иными словами, категория, получившая 0 голосов, будет белого цвета, а категория с 172 голосами — темно-красного. Категории с промежуточным количеством голосов будут и по цвету где-то между белым и красным.

С переменными покончено. Размер и шкала определены. Чтобы нарисовать собственно диаграмму, сначала нужно создать чистую панель с размерами 350 (w, или ширина) на 350 (h, или высота) пикселов.

```
var vis = new pv.Panel()
   .width(w)
   .height(h);
```

Затем панель нужно наполнить содержимым, в данном случае сегментами. Следующий фрагмент может показаться вам несколько запутанным, но ниже мы разберем его строчка за строчкой.

```
vis.add(pv.Wedge)
   .data(data)
   .bottom(w / 2)
```

```
    .left(w / 2)
    .innerRadius(r - 120)
    .outerRadius(r)
    .fillStyle(function(d) depthColors(d))
    .strokeStyle("#fff")
    .angle(a)
    .title(function(d) String(d) + " votes")
    .anchor("center").add(pv.Label)
      .text(function(d) cats[this.index]);
```

Первая строка сообщает, что вы добавляете сегменты на панель по одному для каждого пункта в массиве данных. Свойства `bottom()` и `left()` ориентируют сегменты так, чтобы их верхушки оказались в центре круга. `innerRadius()` определяет радиус дырки «бублика», а `outerRadius()` — радиус всей окружности. Этим задается структура кольцевой диаграммы.

При указании стиля заливки цвета́ берутся не какие-то фиксированные, а определяются значением каждого конкретного элемента данных, после чего цветовая шкала сохраняется как `depthColors`. Граница задана с помощью `strokeStyle()` — она белая (`#fff`). Круговая шкала сама способна определить угол каждого сегмента.

Для создания контекстного окна (тултипа), которое при наведении на сегмент указателя мыши должно сообщать, сколько человек отдали свои голоса соответствующей категории, используется `title()`. Можно было бы вместо этого создать событие «mouseover», чтобы определить, что именно должно происходить, когда указатель мыши оказывается над объектом. Но поскольку браузер автоматически показывает значение атрибута `title` (название), проще использовать именно его. В приведенном выше фрагменте название состоит из значений элементов данных и слова "votes" (голоса). Далее идут подписи к каждому сегменту. Единственное, что остается сделать — это добавить текст "May 2009" (май 2009 г.) в центре диаграммы.

```
vis.anchor("center").add(pv.Label)
    .font("bold 14px Georgia")
    .text("May 2009");
```

Этот фрагмент можно прочитать как команду вставить в центр диаграммы подпись полужирным шрифтом Georgia 14-м кеглем «Май 2009».

Диаграмма выстроена, можно ее визуализировать.

```
vis.render();
```

Когда вы откроете файл `donut.html` в браузере, вы должны увидеть то, что изображено на рис. 5.10.

Если вы новичок в программировании, данный раздел мог показаться вам чересчур сложным. Но есть и хорошая новость: Protovis разработана так, чтобы ее можно было изучать на примерах. На сайте этой библиотеки вы найдете множество примеров кода, которые вы сможете использовать и для обучения, и в работе с вашими собственными данными. Там есть

▶ Зайдите на `http://book.flowingdata.com/ch05/donut.html`, чтобы увидеть диаграмму «вживую» и посмотреть исходный код в полном объеме.

и традиционные формы статистической графики, и более продвинутые образцы интерактивной и анимированной графики. Так что если вы и растерялись немного в какой-то момент, не стоит паниковать. Те усилия, которые вы сейчас вложите в изучение материала, окупятся сторицей, когда вы начнете применять полученные знания на практике. В следующем разделе у вас будет возможность продолжить знакомство с библиотекой Protovis.

Сложите их

В предыдущей главе вы использовали штабельные столбцовые диаграммы для визуализации данных во времени, но такие диаграммы подходят не только для работы с темпоральными данными. Как показано на рис. 5.11, вы можете использовать штабельные столбцовые диаграммы также и для визуализации категорийных данных.

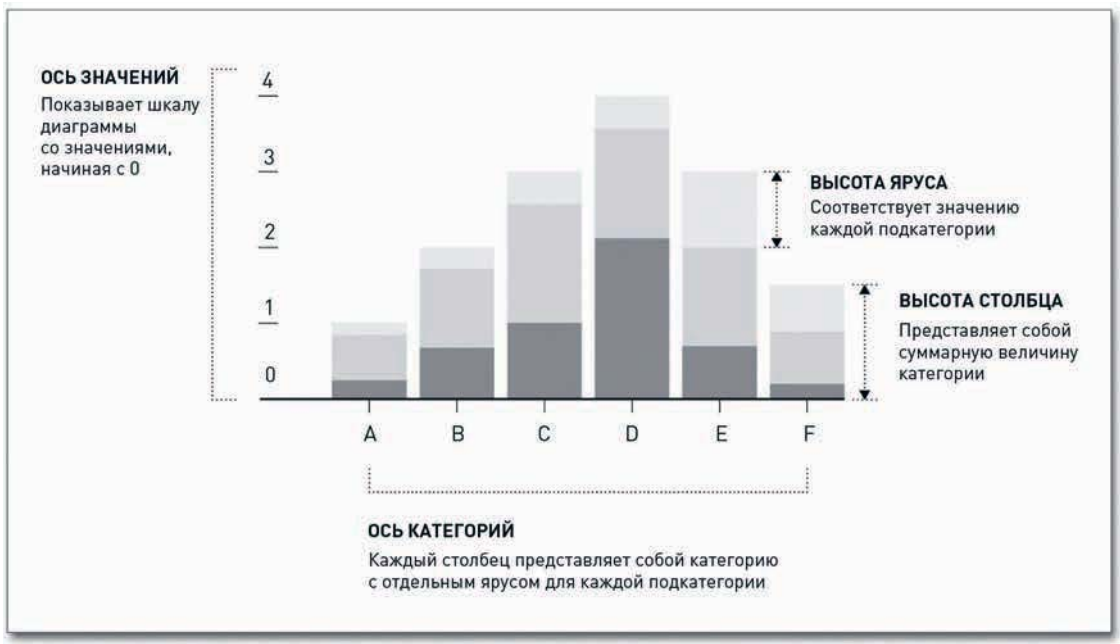


Рис. 5.11. Штабельная столбцовая диаграмма с категориями

В качестве примера давайте рассмотрим рейтинг Барака Обамы, который был составлен по итогам опросов, проведенных компаниями Gallup и CBS в июле и августе 2010 года. Участников спрашивали, одобряют они или не одобряют политику Обамы при решении следующих 13 важнейших для страны вопросов.

Вот эти вопросы с ответами в виде таблицы.

Вопрос	Одобряют	Не одобряют	Не имеют мнения
Межрасовые отношения (Race relations)	52	38	10
Образование (Education)	49	40	11
Терроризм (Terrorism)	48	45	7
Энергетическая политика (Energy policy)	47	42	11
Международные отношения (Foreign affairs)	44	48	8
Экология (Environment)	43	51	6
Ситуация в Ираке (Situation in Iraq)	41	53	6
Налоги (Taxes)	41	54	5
Здравоохранение (Healthcare policy)	40	57	3
Экономика (Economy)	38	59	3
Ситуация в Афганистане (Situation in Afghanistan)	36	57	7
Бюджетный дефицит (Federal budget deficit)	31	64	5
Иммиграция (Immigration)	29	62	9

Один из вариантов действий — это создать по одной круговой диаграмме для каждого вопроса, как показано на рис. 5.12. Чтобы сделать это в программе Illustrator, вам понадобится вместо одной строки с данными ввести несколько. Для каждой будет создана отдельная диаграмма.

Однако штабельная столбцовая диаграмма позволит с большей легкостью сопоставить рейтинги одобрения по различным вопросам, потому что сравнивать высоту столбцов легче, чем углы секторов. Так что давайте попробуем этот вариант. В предыдущей главе мы рисовали диаграмму в Illustrator с использованием инструмента для создания штабельных диаграмм (Stacked Graph). На этот раз мы придадим диаграмме кое-какие простые интерактивные свойства.

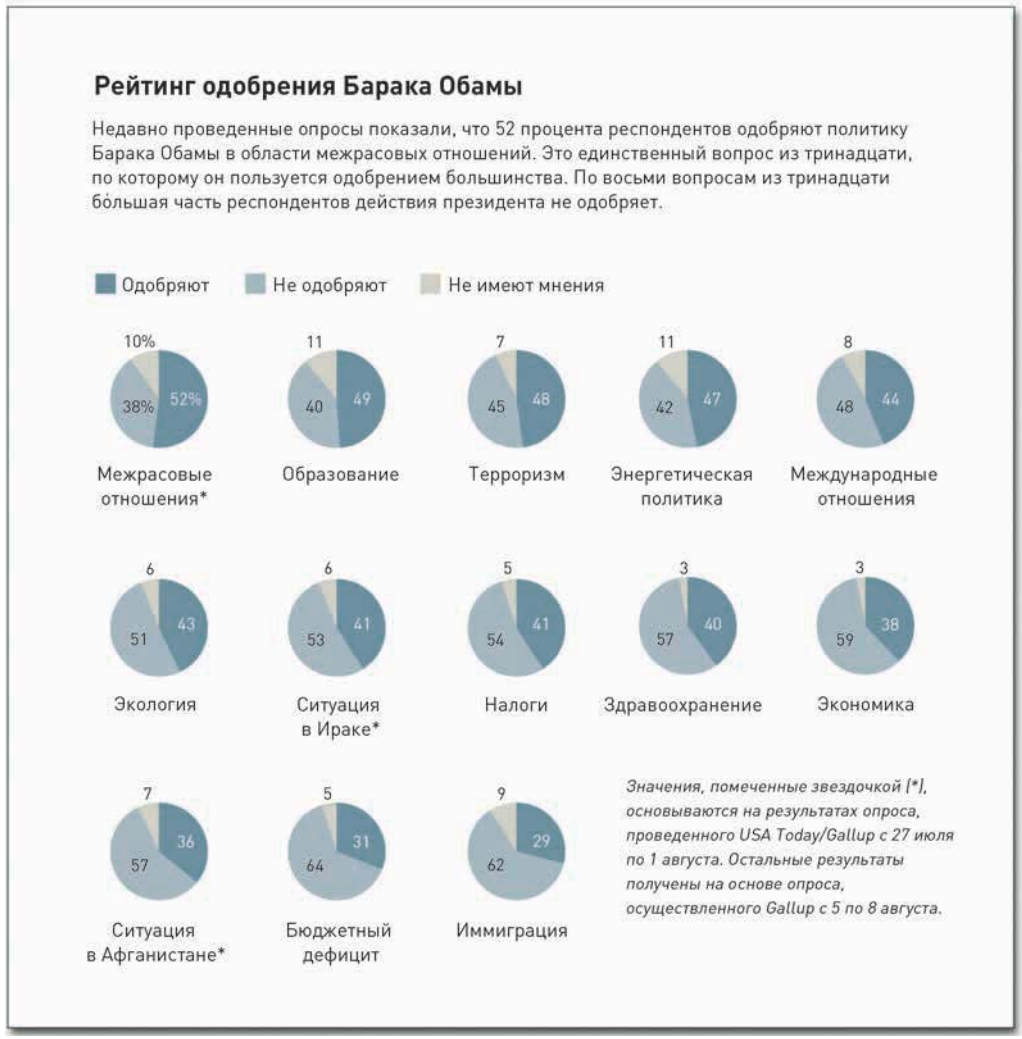


Рис. 5.12. Серия круговых диаграмм

СОЗДАЙТЕ ИНТЕРАКТИВНУЮ ШТАБЕЛЬНУЮ СТОЛБЦОВУЮ ДИАГРАММУ

Как и в примере с кольцевой диаграммой, вам предстоит, используя Protovis, создать интерактивную штабельную столбцовую диаграмму. На рис. 5.13 представлен окончательный ее вариант. Нужно реализовать два основных вида взаимодействия. Во-первых, когда указатель мыши будет проходить над столбцом, на нем должны появляться соответствующие значения в процентах. Второй тип взаимодействия предусматривает выделение разных ярусов столбцов по категориям — «одобряют» (approve), «не одобряют» (disapprove), «не имеют мнения» (no opinion) — в зависимости от того, куда вы направляете мышь.

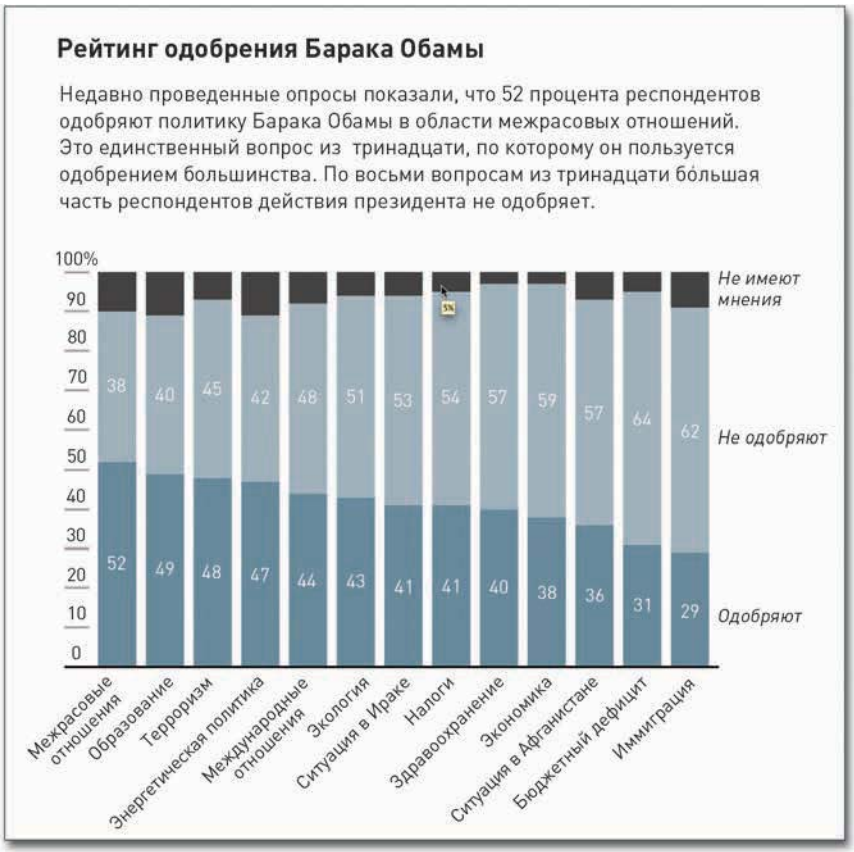


Рис. 5.13. Интерактивная штабельная столбцовая диаграмма, созданная с помощью Protovis

Для начала нужно создать HTML-страницу и загрузить необходимый JavaScript-файл Protovis.

```
<html>
<head>
   <title>Stacked Bar Chart</title>
   <script type="text/javascript" src="protovis-r3.2.js"></script>
</head>
<body>
   <div id="figure-wrapper">
      <div id="figure">

   </div><!-- @end figure -->
   </div><!-- @end figure-wrapper -->
</body>
</html>
```

Кажется знакомым? Вы уже делали это, когда создавали с помощью Protovis кольцевую диаграмму. Отличие состоит лишь в заголовке страницы — на этот раз он гласит: «Stacked Bar Chart» («Штабельная столбцовая диаграмма»), да еще есть дополнительный `<div>` с `id="figure-wrapper"`. Кроме того, на странице нет никаких CSS-стилей — мы прибережем эту операцию для более позднего этапа.

Вернемся к JavaScript. Внутри `<div>` нужно загрузить данные (то есть рейтинги Обамы) в массивы.

```
<script type="text/javascript+protovis">
   var data = {
      "Issue":["Race Relations","Education","Terrorism","Energy Policy",
         "Foreign Affairs","Environment","Situation in Iraq",
         "Taxes","Healthcare Policy","Economy","Situation in Afghanistan",
         "Federal Budget Deficit","Immigration"],
      "Approve":[52,49,48,47,44,43,41,41,40,38,36,31,29],
      "Disapprove":[38,40,45,42,48,51,53,54,57,59,57,64,62],
      "None":[10,11,7,11,8,6,6,5,3,3,7,5,9]
   };
</script>
```

Все это можно прочитать следующим образом: по вопросу межрасовых отношений рейтинг одобрения составил 52 процента, а неодобрения — 38 процентов. Далее, по вопросу ситуации в образовании рейтинг одобрения составил 49 процентов, а неодобрения — 40 процентов, и т. д.

Чтобы облегчить процесс составления кода для этой диаграммы, вы можете разбить данные и сохранить их в двух переменных:

```
var cat = data.Issue;
var data = [data.Approve, data.Disapprove, data.None];
```

Массив вопросов сохраняется в `cat`, а данные теперь представляют собой массив из массивов.

Переменные для ширины (width), высоты (height), шкалы (scale) и цветов (colors) задаются следующим образом:

```
var w = 400,
h = 250,
x = pv.Scale.ordinal(cat).splitBanded(0, w, 4/5),
y = pv.Scale.linear(0, 100).range(0, h),
fill = ["#809EAD", "#B1C0C9", "#D7D6CB"];
```

Диаграмма получится размером 400 пикселов в ширину и 250 пикселов в высоту. Горизонтальная шкала будет порядковая (а не непрерывная), иными словами, для нее задаются некие категории. Категориями станут темы, охваченные опросами. Четыре пятых ширины диаграммы будет использовано под столбцы, а остальная часть — под интервалы между столбцами.

Вертикальная ось с процентами будет иметь линейную шкалу от 0 до 100. Высота столбцов может варьировать в диапазоне от 0 пикселов до высоты диаграммы, то есть 250 пикселов.

И наконец, заливка. В массивах заливка определяется шестнадцатеричными числами. В данном случае были использованы темно-голубой — для одобряющих, светло-голубой — для неодобряющих и светло-серый — для тех, кто не определился со своим мнением. Вы можете изменить цвета на любые другие по своему усмотрению.

А теперь следующий шаг: нам нужно инициализировать визуализацию с определенной шириной и высотой. Остальная часть пойдет на создание «воздушной подушки» вокруг диаграммы, чтобы вы могли разместить названия осей. Так, `bottom(90)` сдвигает нулевую позицию оси вверх на 90 пикселов. Следующую процедуру можно сравнить с установкой чистого холста.

```
var vis = new pv.Panel()
   .width(w)
   .height(h)
   .bottom(90)
   .left(32)
   .right(10)
   .top(15);
```

Для того чтобы добавить на холст штабелированные столбцы, можно воспользоваться специальной схемой для штабельных диаграмм (стеков), которую предоставляет Protovis и которая так и называется: Stack. Хотя в данном примере вы воспользуетесь ею для штабельных диаграмм, эта схема также может быть полезной при создании штабельных диаграмм с областями и потоковых графиков. Сохраните новую схему в переменной bar («столбец»).

```
var bar = vis.add(pv.Layout.Stack)
   .layers(data)
   .x(function() x(this.index))
   .y(function(d) y(d))
   .layer.add(pv.Bar)
      .fillStyle(function() fill[this.parent.index])
   .width(x.range().band)
   .title(function(d) d + "%")
   .event("mouseover", function() this.fillStyle("#555"))
   .event("mouseout", function()
      this.fillStyle(fill[this.parent.index]));
```

Впрочем, к данной диаграмме можно подойти и по-другому: как к состоящей из трех слоев — по одному для одобряющих, неодобряющих и не имеющих мнения. Вы помните, что эти данные у нас были структурированы как массив из трех массивов? Об этом сказано в `layers()`, где x и y выводятся из уже созданных шкал.

▶ Если вы не уверены, какие именно цвета следует использовать, тогда вам стоит начать с ColorBrewer — с `http://colorbrewer2.org`. Данный инструмент поможет вам точно определить номера и тип ваших цветов и предоставит цветовую шкалу, которую вы можете скопировать в разных форматах. Еще есть 0to255 (на `http://0to255.com`) — инструмент для работы с цветом более общего характера, я пользуюсь им довольно часто.

Для каждого слоя нужно добавить столбцы, используя `pv.Bar`. Стиль заливки задается с помощью `fillStyle()`. Обратите внимание на то, что мы использовали функцию `this.parent.index`. В результате этого штабелированный столбец окрасится в такие цвета и таким образом, как это задано для разных ярусов, каковых у нас три. Если бы мы использовали `this.index`, то потребовалось бы определить отдельно цвет каждого яруса каждого столбца, то есть всего 39 (по 3 яруса в 13 столбцах). Ширина столбцов остается постоянной, вы можете понять это из порядковой шкалы, которую уже задали.

Последние три строчки представленного выше кода и есть то, что делает диаграмму интерактивной. Использование `title()` в Protovis эквивалентно использованию атрибута `title` для HTML-элемента, скажем, для изображения. Если вы дадите картинке название, то в момент, когда на веб-странице указатель мыши окажется поверх этой картинки, появится всплывающая подсказка. Схожим образом устроено и здесь — когда указатель задерживается на секунду поверх столбца, всплывает подсказка. В данном случае мы сделали так, что в подсказке содержится только число (процентное значение) соответствующего яруса столбца и знак процента (%).

ПОДСКАЗКА

Интерактивные возможности в Protovis не ограничиваются лишь действиями, выполняемыми при входе указателя мыши в зону объекта и выходе из нее. Вы можете также запрограммировать событие для таких действий, как одиночный и двойной щелчок мышью. За дополнительной информацией обратитесь к документации Protovis.

Чтобы добиться эффекта, когда при наведении мыши на тот или иной участок штабелированного столбца подсвечивается соответствующий ярус всех столбцов, мы использовали `event()`. При наведении на объект курсора мыши ("mouseover") цвет заливки меняется на темно-серый (#555), а когда указатель мыши уходит с объекта, ярус принимает первоначальный цвет, для чего используется событие "mouseout".

Чтобы диаграмма появилась, необходимо визуализировать ее. Для этой цели в конце JavaScript-кода нужно ввести следующую строчку:

```
vis.render();
```

Это, по сути, команда: «Вот и всё, мы сложили все кусочки. Теперь приступай к визуализации». Откройте страницу в веб-браузере (в одном из современных — Firefox или Safari), и вы увидите нечто, похожее на рис. 5.14.

Наведите указатель мыши на столбец, и соответствующий ярус подсветится, а еще появится всплывающая подсказка. Но кое-чего здесь не хватает, а именно — осей и подписей к ним. Давайте добавим их.

В диаграмме на рис. 5.13 столбцы сопровождаются множеством подписей. Но все они расположены на широких ярусах, а на сером ярусе их нет. Задать, чтобы все происходило именно так, можно следующим образом (не забывайте, что приведенный ниже фрагмент кода необходимо вставить до `vis.render()`, визуализация всегда должна идти в самом конце).

```
bar.anchor("center").add(pv.Label)
   .visible(function(d) d > 11)
   .textStyle("white")
   .text(function(d) d.toFixed(0));
```

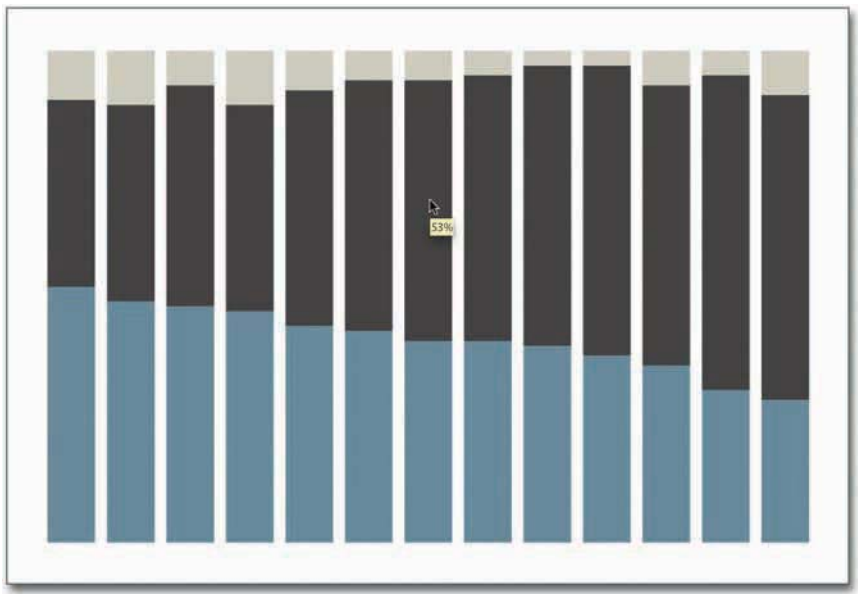


Рис. 5.14. Штабельная столбцовая диаграмма без каких-либо подписей

Иными словами: необходимо проверить значение каждого яруса, и если оно окажется больше 11, снабдить ярус белой подписью, которая должна появиться по его центру, показывая процент, округленный до ближайшего целого числа.

А теперь давайте добавим на оси X подписи с тематикой вопросов о политике Обамы. В идеале хотелось бы, чтобы все подписи располагались горизонтально, но здесь для них места явно недостаточно. Если бы это была горизонтальная столбцовая диаграмма, мы могли бы приладить к ней горизонтальные подписи, но в данном случае придется разместить их под углом в 45 градусов. Подписи можно сделать и вертикальными, но так они станут менее читабельными.

```
bar.anchor("bottom").add(pv.Label)
   .visible(function() !this.parent.index)
   .textAlign("right")
   .top(260)
   .left(function() x(this.index)+20)
   .textAngle(-Math.PI / 4)
   .text(function() cat[this.index]);
```

Все это работает таким же образом, как и тогда, когда мы размещали подписи с числами в середине каждого яруса каждого столбца. Однако теперь нужно добавить подписи лишь к нижнему ярусу столбцов, то есть к ярусу одобряющих. Затем с помощью `textAlign()` и `top()` нужно выровнять текст по правому краю и задать абсолютную позицию подписей по вертикали. Их x-позиция определяется тем, к какому столбцу они относятся, при этом они все располагаются под углом в 45 градусов, а текстом является соответствующая категория.

Итак, мы получили подписи с категориями. Подписи со значениями к вертикальной оси добавляются точно таким же способом, но к ним нужно добавить еще и штриховые метки.

```
vis.add(pv.Rule)
   .data(y.ticks())
   .bottom(y)
   .left(-15)
   .width(15)
   .strokeStyle(function(d) d > 0 ? "rgba(0,0,0,0.3)" : "#000")
   .anchor("top").add(pv.Label)
   .bottom(function(d) y(d)+2)
   .text(function(d) d == 100 ? "100%" : d.toFixed(0));
```

Данный фрагмент кода вводит линейку (`Rule`), или штрихи, в соответствии с `y.ticks()`. Если метка представляет собой что-то иное, кроме нулевой линии, она окажется окрашена в серый цвет. В противном случае она будет черной. Вторая секция кода снабжает метки подписями.

Однако у нас все еще нет горизонтальной оси, так что давайте добавим еще одну линейку, чтобы диаграмма стала походить на ту, которая изображена на рис. 5.15.

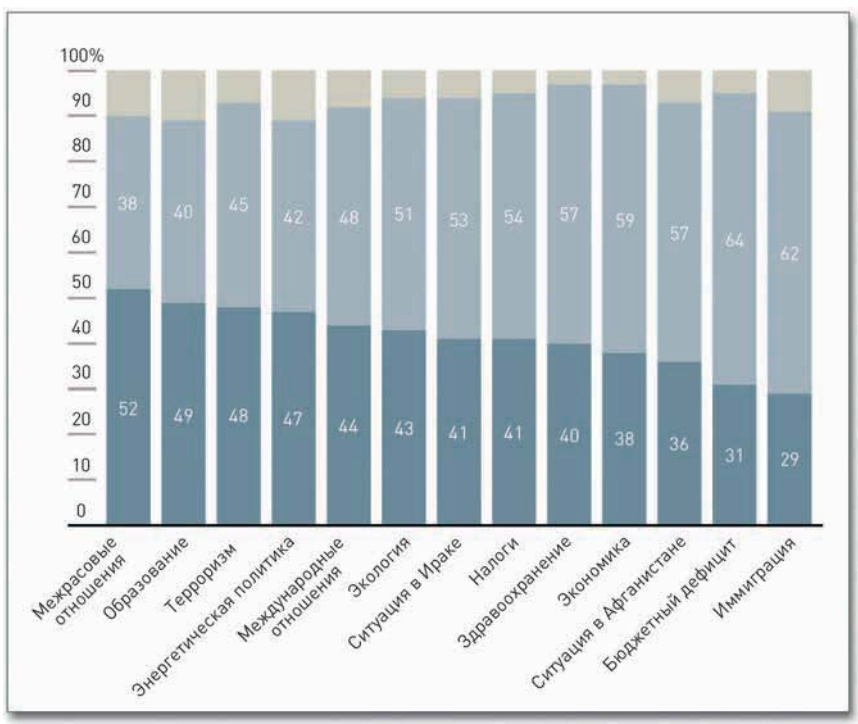


Рис. 5.15. Добавление горизонтальной оси

Вводный текст и остальные подписи добавляются с помощью HTML и CSS. Но так как теме веб-дизайна посвящена не одна книга, я на этом и закруглюсь. Что особенно здорово при работе с такими форматами, так это то, что вы легко можете комбинировать HTML и CSS с Protovis, которая представляет собой JavaScript, и при этом переходов между ними совершенно не будет заметно.

▶ Чтобы увидеть готовую штабельную столбцовую диаграмму и испытать ее в интерактивном режиме, зайдите на `http://book.flowingdata.com/ch05/stacked-bar.html`. Изучите исходный код страницы, чтобы увидеть, как стыкуются вместе HTML, CSS и JavaScript.

Иерархия и прямоугольники

В 1990 году Бен Шнайдерман (Ben Shneiderman) из Университета Мэриленда захотел визуализировать то, что творится у него на вечно переполненном жестком диске. Он решил выяснить, что же именно занимает столько места. Учитывая иерархическую структуру директорий и файлов, он сначала попытался создать древовидную диаграмму. Однако она вскоре стала чересчур громоздкой и, как следствие, бесполезной. Там было слишком много узлов. И слишком много ветвей.

Но Шнайдерман все же решил проблему, и этим решением стал тримап. Как видно из рис. 5.16, речь идет о способе визуализации, основанном на областях, при котором размер каждого прямоугольника выражает собой количественный показатель. Внешние прямоугольники представляют родительские категории, а прямоугольники внутри родительских категорий — это своего рода подкатегории. Вы можете использовать тримап для визуализации прямых пропорций, но чтобы опробовать технологию в полном объеме, ее лучше применять с иерархическими или, точнее, с древовидно структурированными данными.

▶ Чтобы ознакомиться с дополнительными примерами тримапов и подробностями истории их создания в изложении автора, Бена Шнайдермана, зайдите на `http://datafl.ws/11m`.

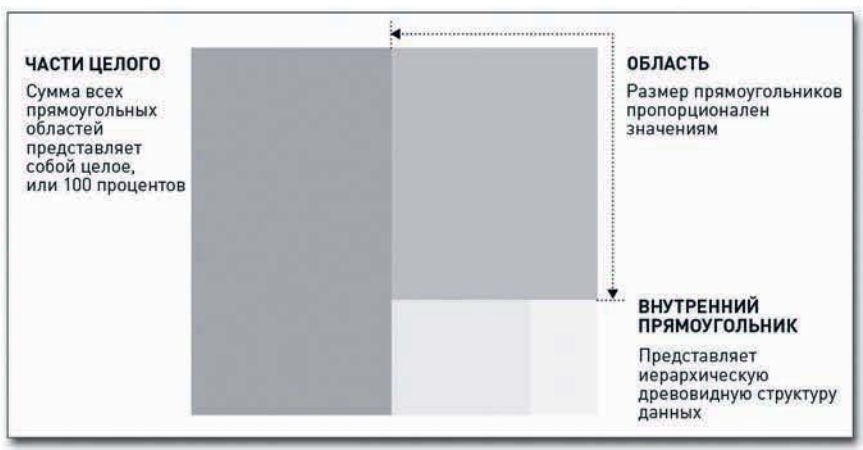


Рис. 5.16. Общая структура тримапа

СОЗДАЙТЕ ТРИМАП

В программе Illustrator нет инструмента для создания тримапов, но он есть в R, в пакете, разработанном Джеффом Эносом (Jeff Enos) и Дэвидом Кейном (David Kane) и называющемся Portfolio («Портфель»). Первоначально пакет создавался для визуализации портфелей ценных бумаг (отсюда и название), но его успешно можно применять и к другим типам данных. Давайте приглядимся к статистике количества просмотров 100 самых популярных постов на сайте FlowingData и оставленных к ним комментариев и разделим их по категориям, таким как «визуализация» или «инфографика».

Как всегда, первым шагом будет загрузка данных в R. Сделать это можно непосредственно с компьютера или указав на конкретный URL. В настоящем примере лучше использовать второй вариант, потому что данные уже есть в Сети. Однако если вы предпочитаете первый вариант и хотите проделать дальнейшие шаги применительно к своим собственным данным, тогда просто позаботьтесь о том, чтобы разместить свой файл с данными в рабочем каталоге R.

ПОДСКАЗКА

R — это программная среда с открытым исходным кодом для статистических вычислений. Вы можете скачать ее бесплатно с `http://www.r-project.org`. Что самое замечательное в R, так это то, что вокруг нее существует активное сообщество программистов, которые постоянно создают новые пакеты и развивают функционал. Если вам необходимо построить статичную диаграмму и вы не знаете, с чего начать, архивы R — это как раз то место, куда стоит заглянуть.

Загрузить CSV-файл с URL нетрудно. Нужна всего одна строчка кода на R с функцией `read.csv()` — рис. 5.17.

```
posts <- read.csv("http://datasets.flowingdata.com/post-data.txt")
```

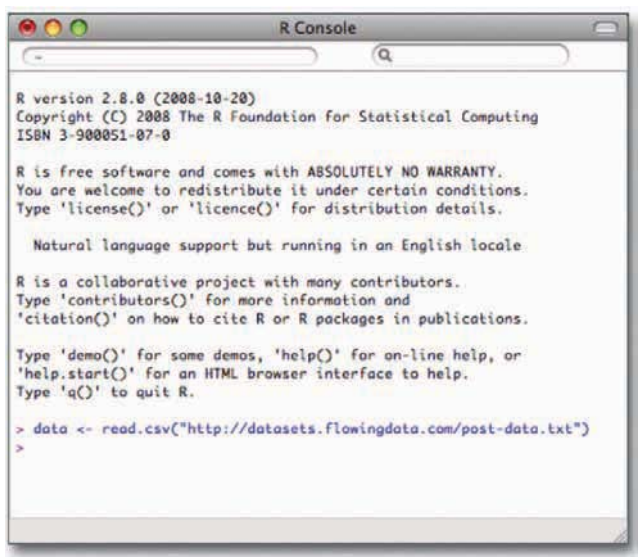


Рис. 5.17. Загрузка CSV в R

Легко, не правда ли? Мы загрузили текстовый файл (в формате CSV), используя `read.csv()`, и сохранили значения для просмотров и комментариев в переменной `posts`. Как уже упоминалось в предыдущей главе, функция `read.csv()` по умолчанию принимает, что файл у вас создан с разделителями-запятыми. Если бы в роли разделителя в файле с данными выступал, скажем, знак табуляции, тогда бы следовало использовать аргумент `sep`, а в качестве его значения задать `\t`. Если хотите загрузить данные из локальной директории, тогда предыдущая строчка должна выглядеть так:

```
posts <- read.csv("post-data.txt")
```

В этом случае принимается, что вы соответствующим образом изменили свой рабочий каталог. Чтобы ознакомиться с другими возможностями и получить дополнительные инструкции о том, как загружать данные, используя функцию `read.csv()`, наберите в консоли R:

```
?read.csv
```

Идем дальше. Теперь, когда данные сохранены в переменной `posts`, нам нужна следующая строчка, чтобы увидеть первые пять рядов данных.

```
posts[1:5,]
```

Перед вами должны появиться четыре колонки, соответствующие исходному CSV-файлу: *id, views, comments* и *category*. Сейчас, когда данные уже загружены в R, можно воспользоваться пакетом Portfolio. Попробуйте загрузить его с помощью следующей строчки:

```
library(portfolio)
```

Получили ошибку? Возможно, вам необходимо установить пакет, прежде чем начать. Для этого введите:

```
install.packages("portfolio")
```

Теперь пакет точно должен загрузиться. Все прошло без ошибок? Хорошо. Тогда переходим к следующему шагу.

Пакет Portfolio выполняет всю тяжелую работу с помощью функции под названием `map.market()`. У нее есть несколько аргументов, но мы будем использовать только пять из них.

```
map.market(id=data$id, area=posts$views, group=posts$category,
   color=posts$comments, main="FlowingData Map")
```

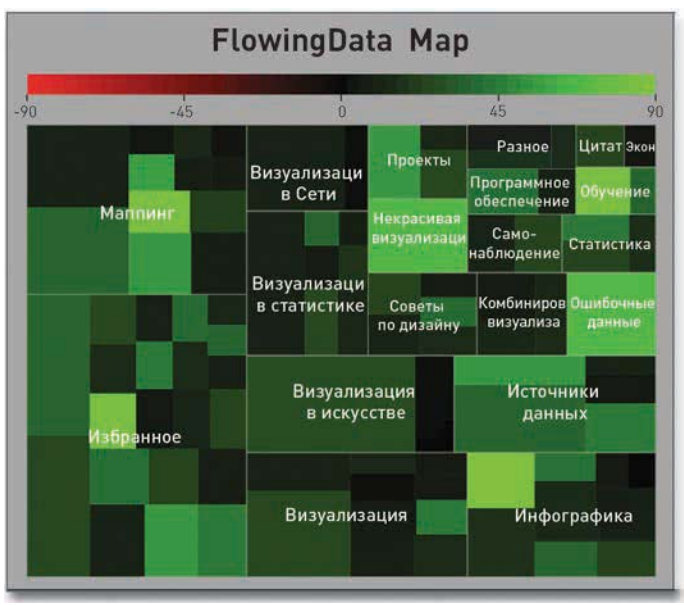


Рис. 5.18. Тримап по умолчанию в R

Итак, `id` — это та колонка, которая указывает на конкретные элементы, и мы отдаем команду R использовать *views*, чтобы определить площадь прямоугольников в тримапе, категории (чтобы сформировать группы) и количество комментариев к постам (чтобы определить их цвет). **FlowingData Map** в конце — это основной заголовок.

Нажмите клавишу Enter на клавиатуре — и вы увидите тримап, похожий на тот, что представлен на рис. 5.18.

Полученный графический объект все еще немного неаккуратный, но основа создана, иерархия выстроена, а это и есть самая сложная часть работы. Как и требовалось, прямоугольники,

ПОДСКАЗКА

Вы можете установить пакеты R и через пользовательский интерфейс. Перейдите на вкладку Packages & Data → Package Installer (Пакеты и данные → Инсталлятор пакетов). Щелкните по Get List (Получить список) и в появившемся перечне найдите интересующий вас пакет. Двойной щелчок мыши запустит установку.

каждый из которых представляет собой тот или иной пост, имеют размеры, соответствующие количеству просмотров страницы, и они рассортированы по категориям. Чем светлее оттенок зеленого, тем больше комментариев получил соответствующий пост. Посты с бóльшим числом просмотров не обязательно имеют и много комментариев.

Можете сохранить изображение в R как PDF, а затем открыть файл в Illustrator. К нему применимы все стандартные опции. Вы можете менять обводку, цвет заливки, шрифты, можете удалить все лишнее, а если захотите, то и добавить комментарии.

В данном случае необходимо изменить шкалу легенды, которая идет от –90 до 90. Отрицательная часть шкалы здесь не имеет смысла, поскольку не бывает отрицательного количества комментариев. А еще можно подправить подписи. Некоторые из них — те, что в прямоугольниках поменьше, — частично загорожены и не видны. Пусть размер подписей соответствует популярности категории, а не будет выстроен по единой шкале, как сейчас. Для этого следует использовать инструмент Selection (Выделение). Стоит также увеличить толщину границ категорий, чтобы они стали более заметными. В результате должна получиться диаграмма, похожая на ту, что представлена на рис. 5.19.

▶ В своей статье How the Giants of Finance Shrank, Then Grew, Under the Financial Crisis газета New York Times использовала анимированный тримап, чтобы показать, какие изменения происходили на фондовой бирже во время финансового кризиса. Посмотреть на него в действии можно на странице `http://nyti.ms/9JUkWL`.

Рис. 5.19. Тримап, созданный в R и отредактированный в Illustrator

Вот это уже другое дело. Теперь читать диаграмму стало намного легче: подписи не перекрываются другими элементами, да и цветовая шкала стала более вразумительной. Мы также избавились от темно-серого фона, и изображение стало чище. О да, конечно, мы включили еще и заголовок и вводный текст, коротко объяснив, о чем повествует этот тримап.

Так как бóльшая часть тяжелой работы была выполнена пакетом Portfolio, единственное, что может вас затруднить, когда вы решите применить метод к своим собственным данным, — это перевод их в нужный формат. Помните: вам необходимы три вещи: уникальный id для каждого ряда, количественные показатели для определения размера прямоугольников и родительские категории. По желанию вы можете использовать и четвертый показатель для определения цвета прямоугольников. Вернитесь ко второй главе «Как обращаться с данными» и просмотрите рекомендации о том, как перевести данные в нужный вам формат.

Пропорции во времени

Довольно часто приходится работать с пропорциями во времени. На месте результатов единственного опроса с определенным количеством тем могут оказаться результаты серии опросов, которые проводились каждый месяц на протяжении года. Ведь иногда интересно получить не только одномоментную картину мнений, но также посмотреть, как эти мнения развиваются во времени, например, как изменились взгляды с прошлого года по сегодняшний день.

Конечно, это относится не только к опросам. Не счесть распределений, которые меняются со временем. В следующих примерах вы ознакомитесь с распределением возрастных групп в Соединенных Штатах с 1860 по 2005 годы. С улучшением здравоохранения и уменьшением размера среднестатистической семьи население в целом живет дольше, чем предыдущие поколения.

Штабелированные непрерывные

Представьте себе, что у вас есть несколько диаграмм временны́х рядов. Теперь разместите каждую линию поверх предыдущей. Заполните пустое пространство. То, что вы получите в итоге, называется штабельной диаграммой с областями, где на горизонтальной оси располагается время, а вертикальная ось имеет диапазон от 0 до 100 процентов, как показано на рис. 5.20.

Так что если бы вам надо было снять вертикальный срез с диаграммы с областями, вы бы получили распределение за этот конкретный временной интервал. А еще данный тип диаграмм можно рассматривать как серию штабельных столбцовых диаграмм, столбцы которых соединены во времени.

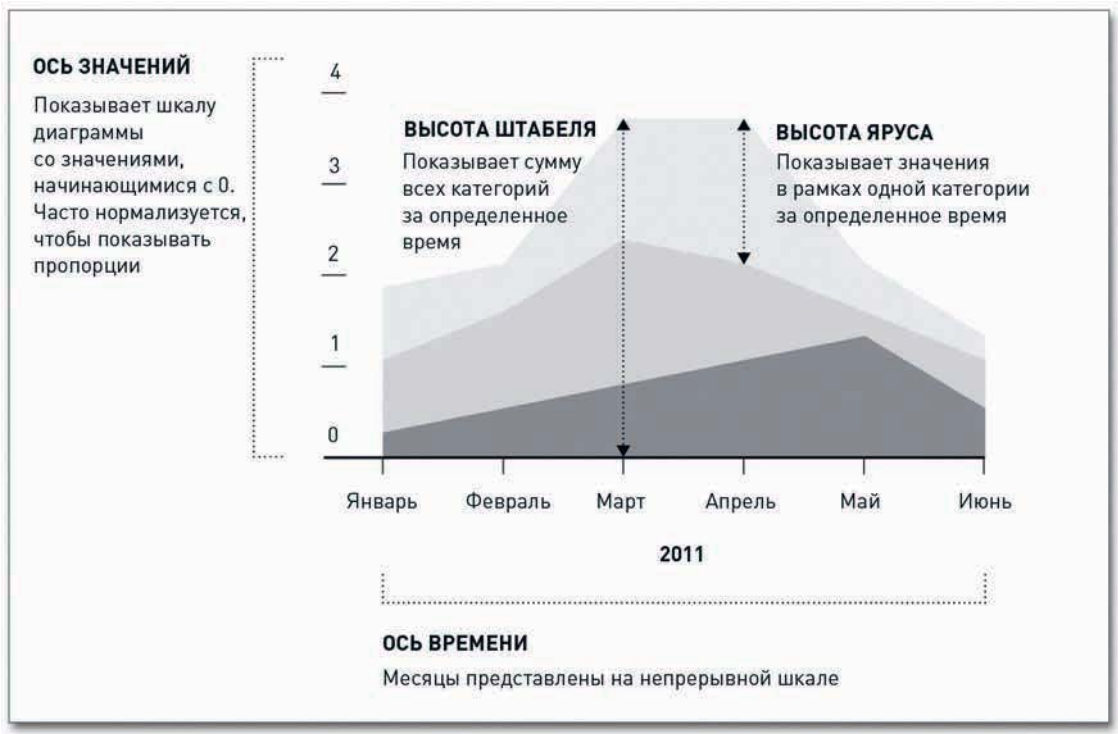


Рис. 5.20. Обобщенная схема штабельной диаграммы с областями

СОЗДАЙТЕ ШТАБЕЛЬНУЮ ДИАГРАММУ С ОБЛАСТЯМИ

В данном примере мы будем изучать старение населения. Загрузите данные с `http://book.flowingdata.com/ch05/data/us-population-by-age.xls`. За прошедшие десятилетия медицина и здравоохранение значительно развились, и средняя продолжительность жизни неуклонно увеличивается. В результате процент населения в старшей возрастной группе вырос. Насколько сильно распределение населения по возрастным группам изменилось с годами? Данные Бюро переписи населения США способны помочь разобраться в этом, для чего понадобится штабельная диаграмма с областями. Мы хотим увидеть, как увеличивалась доля старших возрастных групп и как сокращалась доля младших.

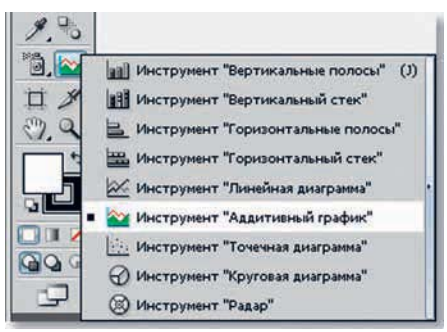


Рис. 5.21. Инструмент «Аддитивный график» (Area Graph), или диаграмма с областями

Можно сделать это разными способами, но сначала давайте воспользуемся программой Illustrator. В ней для создания штабельных диаграмм с областями существует инструмент «Аддитивный график» (Area Graph) — рис. 5.21.

Щелкните мышью и протащите указатель в каком-нибудь месте в новом документе, затем введите данные в таблицу, которая появится перед вами.

Процесс загрузки данных, генерирования диаграммы и ее обработки вам уже знаком, не правда ли?

После того как вы введете данные, вы увидите штабельную диаграмму с областями, похожую на ту, что представлена на рис. 5.22.

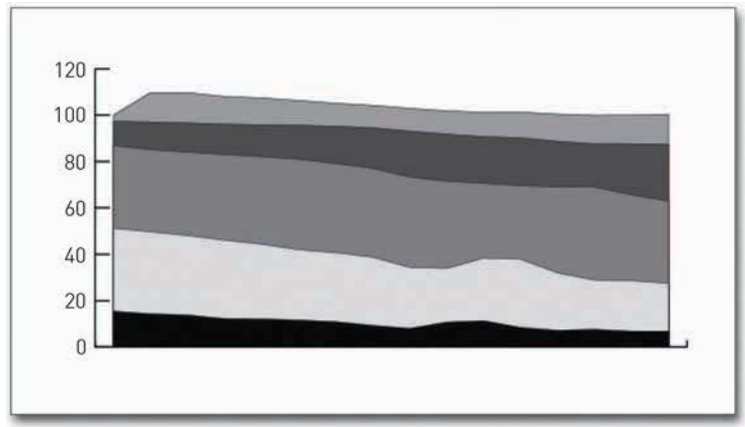


Рис. 5.22. Штабельная диаграмма с областями по умолчанию в Illustrator

Верхний ярус пересек линию 100 процентов. Это произошло потому, что штабельные диаграммы с областями применяются не только для нормализованных пропорций или для наборов значений, составляющих в сумме 100 процентов. Их можно применять также и в случае с необработанными значениями, так что если вы хотите, чтобы отдельные временны́е интервалы не выходили за 100 процентов, вам придется нормализовать данные. В действительности верхнее изображение явилось результатом допущенной мной ошибки: я сам ввел данные некорректно. Та-а-ак! Кое-что немного подправим, и теперь вы можете увидеть диаграмму как на рис. 5.23. Хотя вы, возможно, ввели данные правильно с первого раза, и уже поджидаете меня здесь.

ПОДСКАЗКА

Будьте осторожны, когда вводите данные вручную. Множество глупых ошибок происходит именно при переносе данных из одного источника в другой.

Всегда нужно быть начеку и не допускать, чтобы с вашими диаграммами происходило подобное. Лучше отлавливать опечатки и небольшие помарки на ранних этапах, еще при вводе данных, чтобы потом не приходилось возвращаться назад, пытаясь сообразить, что именно и когда пошло не так.

Теперь, когда у вас уже есть правильная основа, почистите оси и линии. Для выделения отдельных элементов используйте инструмент Direct Selection (Прямое выделение). Я предпочитаю удалять вертикальные линии и оставлять только тоненькие штрихи меток, так как это придает диаграмме более опрятный и не такой громоздкий вид. Добавьте знак процента к числам, так как именно с процентами мы и работаем. Как правило, я также меняю цвет обводки зон заливки с черного (по умолчанию) на белый. А еще можно добавить немного синего цвета разных оттенков. В конце вы должны прийти к варианту, изображенному на рис. 5.24.

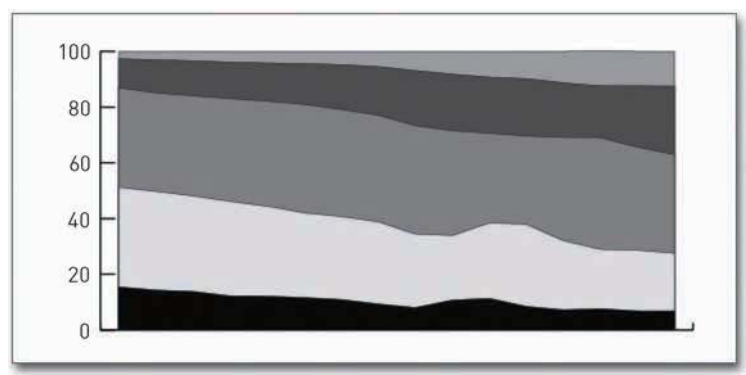


Рис. 5.23. Исправленная диаграмма с областями

И, опять-таки, это всего лишь мои предпочтения в области дизайна, а вы можете делать все, что хотите. Выбор цвета также зависит от конкретного случая. Чем больше диаграмм вы сделаете, тем тоньше начнете чувствовать, что вам нравится и что лучше работает.

ПОДСКАЗКА

Используйте цвета, которые подходят для вашей темы и оттенки которых помогают читателям ориентироваться в том, что они видят.

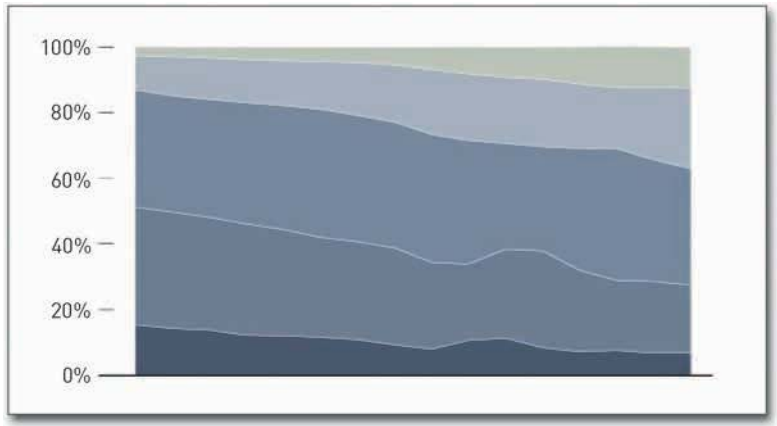


Рис. 5.24. Изменение цветов по умолчанию на другие

Может, чего-то не хватает? Ну да, нет подписей на горизонтальной оси. Вставьте их. И пока вы еще там, вставьте также подписи к областям, чтобы обозначить возрастные группы (рис. 5.25).

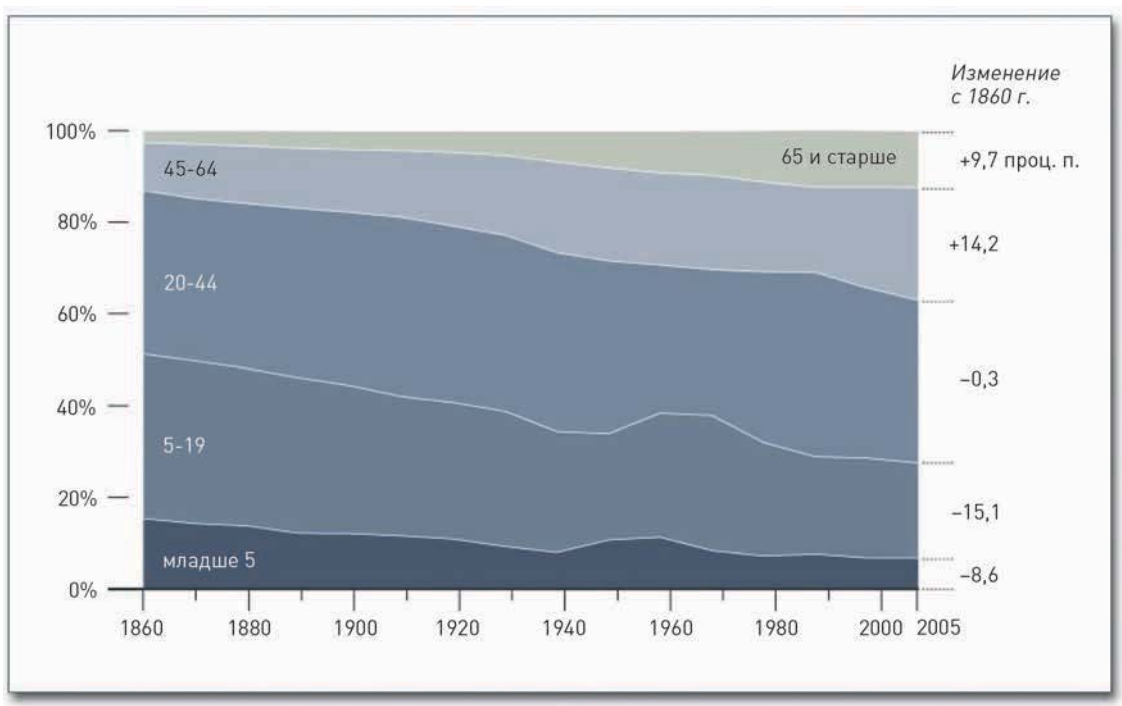


Рис. 5.25. Штабельная диаграмма с областями, сопровождаемая подписями

Помимо этого, справа от диаграммы я добавил примечание. В данном случае нас прежде всего интересует изменение в распределении населения по возрастным группам. Мы это видим из самой диаграммы, хотя реальные цифры помогают донести мысль яснее.

И наконец, вставьте заголовок и вводный текст, а также укажите источник данных внизу. Немного подкорректируйте цвета подписей справа, чтобы сделать их осмысленнее, — и все. У вас на руках окончательный вариант диаграммы, как он показан на рис. 5.26.

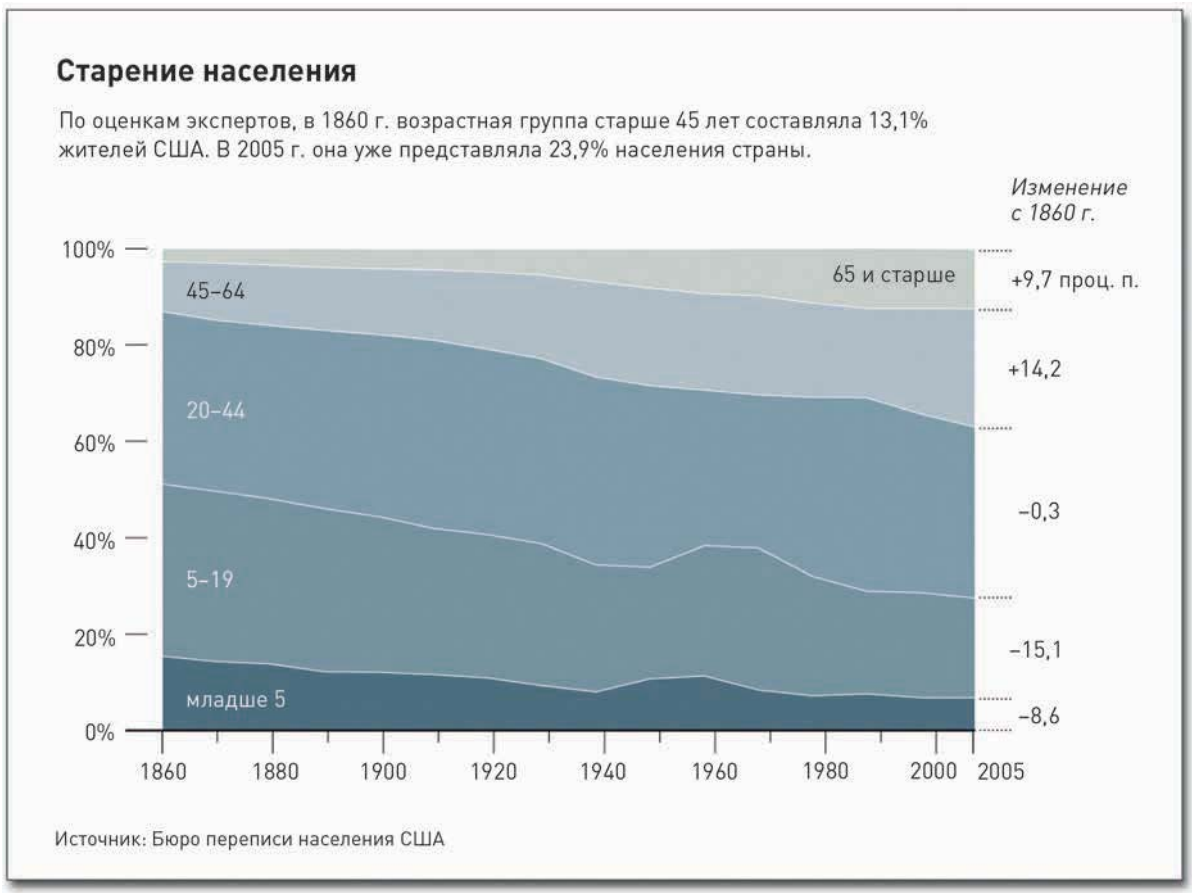


Рис. 5.26. Штабельная диаграмма с областями, сопровождаемая подписями

СОЗДАЙТЕ ИНТЕРАКТИВНУЮ ШТАБЕЛЬНУЮ ДИАГРАММУ С ОБЛАСТЯМИ

Один из недостатков использования штабельных диаграмм с областями состоит в том, что они становятся нечитабельными и бесполезными, если у вас много категорий и много точек данных. В случае с разбивкой на возрастные группы диаграмма работает, так как категорий всего пять.

Но если добавлять их еще и еще, ярусы в конце концов начнут выглядеть как узенькие полоски. Ситуация усложняется и тогда, когда у вас есть одна категория с относительно небольшой долей в общей сумме — она легко может затеряться между более внушительными категориями. Однако если сделать штабельную диаграмму с областями интерактивной, это проблема становится решаемой.

Мы можем предложить читателям способ искать и выделять различные категории, а ось настроить так, чтобы изображение того сегмента, которым заинтересовался читатель, увеличивалось в масштабе. Всплывающие подсказки также способны помочь рассмотреть значения в тех местах, которые слишком малы, чтобы поставить в них подписи. На практике мы можем взять данные, которые не будут работать в статичной штабельной диаграмме с областями, и использовать их в интерактивном варианте этой диаграммы, где их легко рассмотреть и изучить.

Можно было бы сделать это на JavaScript с Protovis, но, чтобы овладеть другими инструментами (что само по себе круто), в данном случае мы реализуем это с помощью Flash и ActionScript.

▶ Интерактивные штабельные диаграммы с областями сделал популярными Мартин Ваттенберг (Martin Wattenberg) с его NameVoyager. Данный инструмент используется для визуализации выбора имен для новорожденных во времени. Когда вы вводите имя в окошко поиска, диаграмма автоматически обновляется. Попробуйте сами на `http://www.babynamewizard.com/voyager`.

ПРИМЕЧАНИЕ

Сегодня онлайн-визуализация начала переходить от Flash к JavaScript и HTML5, но это небыстрый процесс, так как HTML5 поддерживается не всеми браузерами (а точнее, не поддерживается Internet Explorer). Помимо этого, поскольку Flash существует уже довольно давно, для него есть различные библиотеки и пакеты, которые делают выполнение некоторых задач существенно легче, чем если бы вы попытались сделать то же самое, используя лишь «родной» функционал браузера.

К счастью, вам не придется начинать все с чистого листа. Бóльшая часть работы за вас уже выполнена, и благодарить нужно Flare — инструмент, разработанный и поддерживаемый Лабораторией по визуализации при Калифорнийском университете в Беркли. Flare представляет собой ActionScript-библиотеку для визуализации, которая по сути является «наследником» Java-инструмента под названием Prefuse. Мы опробуем одно из представленных на сайте Flare приложений, а именно JobVoyager, которое очень похоже на NameVoyager, но исследует ситуацию с востребованностью различных профессий. После того как вы установите среду разработки, вам останется лишь подключить данные и настроить внешний вид и эффекты.

Вы можете создать код целиком на ActionScript и компилировать его во Flash-файл. По сути это означает, что вы напишете код (команды на языке, понятном для человека), а затем воспользуетесь компилятором для перевода кода в биты, чтобы ваш компьютер или Flash-плеер могли понять, чего именно вы от них хотите. Иными словами, вам нужно место, где писать, и способ, при помощи которого компилировать.

Более трудный вариант — это написать код в стандартном текстовом редакторе, а затем воспользоваться одним из бесплатных компиляторов от Adobe. Я говорю «трудный», потому что

это означает, что вы пойдете кружным путем, да к тому же вам придется еще кое-что установить у себя в компьютере.

Более легкий способ выполнить задачу — его я вам горячо рекомендую, если вы планируете много работать на Flash и ActionScript, — это использовать Adobe Flex Builder. Он значительно ускоряет нудную часть программирования на ActionScript, так как дает вам возможность и писать код, и компилировать, и устранять ошибки — и все это в одном месте. Минус заключается в том, что он стоит денег, хотя для студентов программа бесплатна.

Если вы не уверены, что готовы раскошелиться, вы всегда можете скачать бесплатную пробную версию и принять решение позже. Но давайте вернемся к примеру со штабельной диаграммой с областями. Я объясню, какие шаги вам необходимо предпринять во Flex Builder, чтобы выполнить задание.

ПОДСКАЗКА

Скачать Flare бесплатно можно с `http://flare.prefuse.org`.

ПРИМЕЧАНИЕ

Пока шла подготовка книги к выходу, компания Adobe в очередной версии изменила название продукта с Flex Builder на Flash Builder. Программы похожи, но между ними есть и некоторые отличия. Описанные далее шаги осуществлялись в Flex Builder, но вы можете выполнить их и в Flash Builder. Скачайте Flash Builder со страницы `http://www.adobe.com/products/flashbuilder`. Если вы студент — обязательно воспользуйтесь предоставляемой скидкой. Просто отошлите копию своего студенческого билета и получите сертификат на право пользования программой бесплатно. В качестве альтернативы вы можете порыться на сайте и найти старую, более дешевую версию Flex Builder.

После загрузки и установки программы Flex Builder запустите ее, и вы увидите окно, похожее на то, что представлено на рис. 5.27.

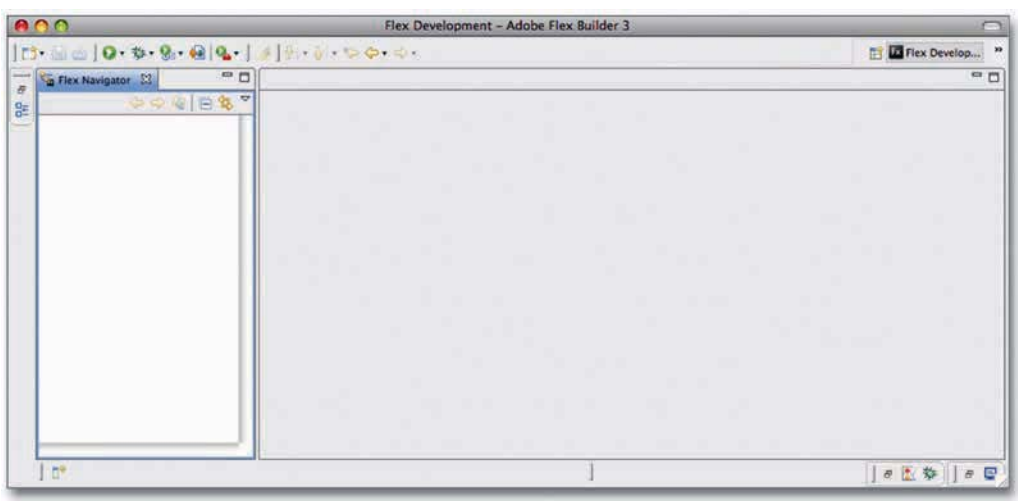


Рис. 5.27. Стартовое окно при открытии Flex Builder

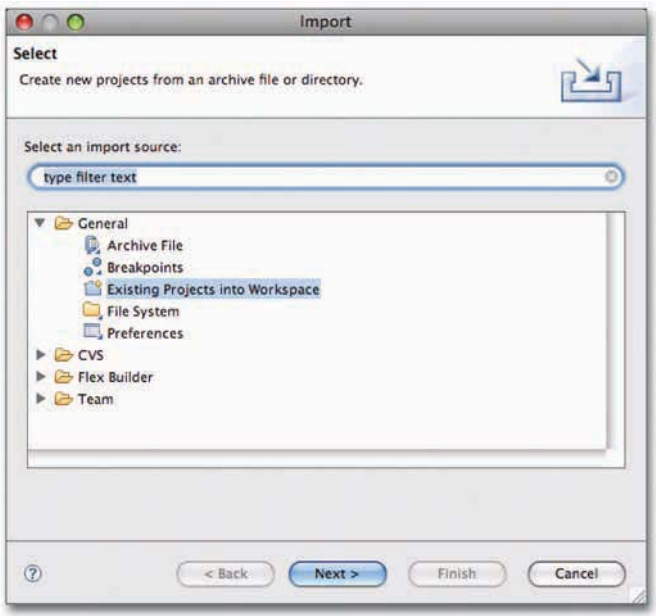


Рис. 5.28. Окно для импортирования файлов в Flex Builder

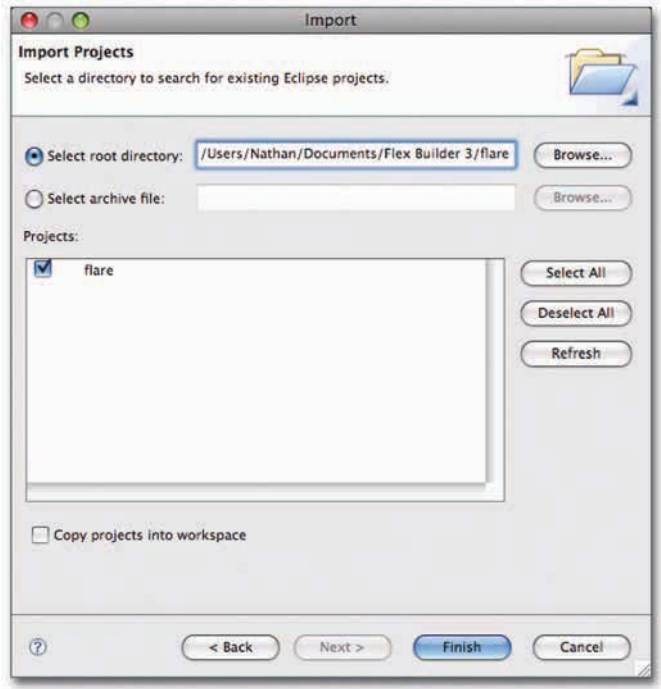


Рис. 5.29. Окно существующих проектов

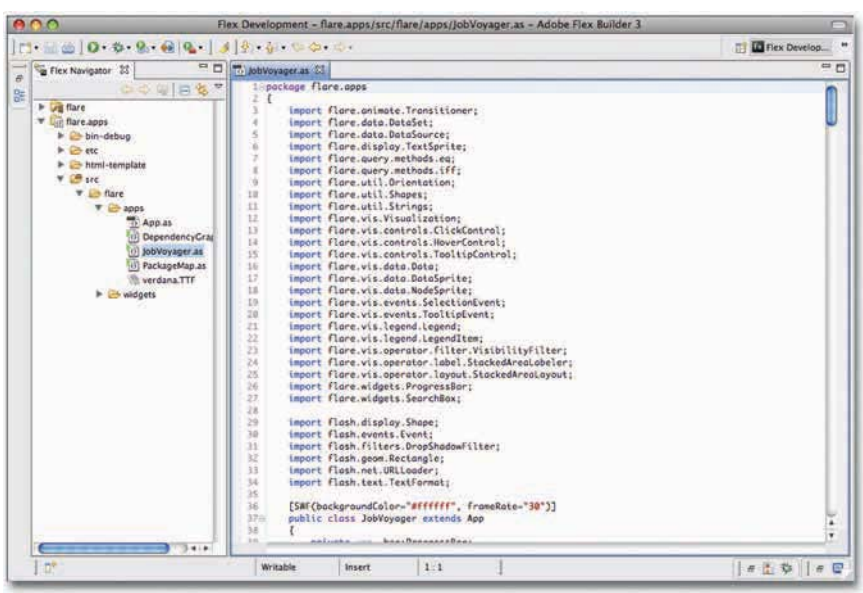


Рис. 5.30. Код JobVoyager в развернутом виде

Щелкните правой кнопкой мыши по навигатору (Flex Navigator) или по блоку слева и выберите Import (импортировать). Вы увидите всплывающее окно вроде того, что на рис. 5.28.

Выберите Existing Project into Workspace (Существующий проект в рабочую область), затем нажмите Next (Далее). Найдите то место, куда вы хотите сохранять Flare-файлы, и убедитесь, что напротив Flare в окне проектов стоит галочка, как показано на рис. 5.29.

Проделайте то же самое и с папкой flare.apps. После того как вы развернете папку flare.apps/flare/apps/ и щелкнете мышью JobVoyager.as, окно Flex Builder у вас должно выглядеть так, как показано на рис. 5.30.

Если вы прямо сейчас нажмете кнопку пуска (наверху слева, она зеленая с белым треугольником), то вы увидите JobVoyager за работой, как показано на рис. 5.31. А как только он запустится, считайте, вы закончили самую тяжелую часть — установку. Теперь вам надо всего лишь вставить свои данные и отредактировать все на свой вкус. Звучит знакомо, не правда ли?

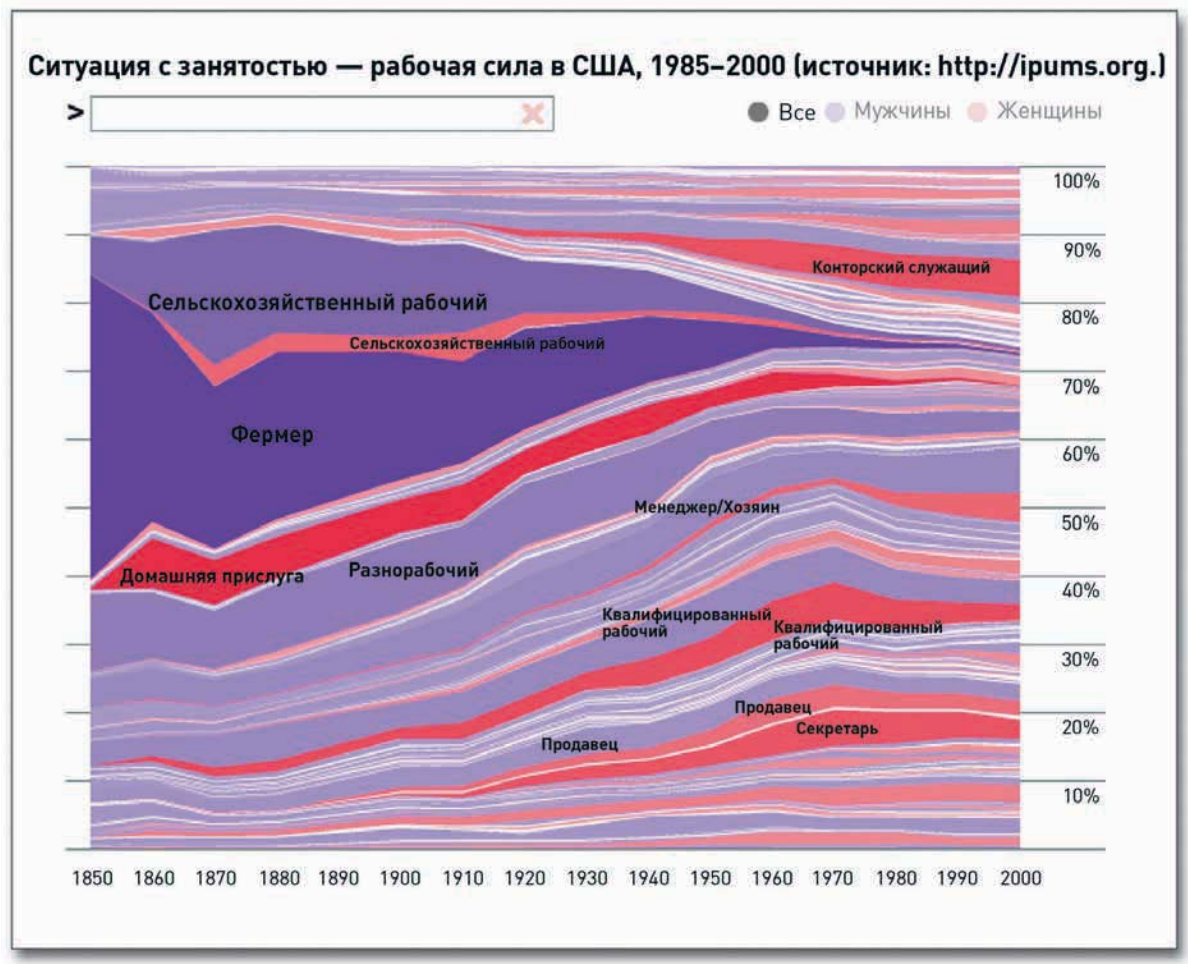


Рис. 5.31. Приложение JobVoyager

На рис. 5.32 показано, к чему вам следует стремиться. Это «вояджер» потребительских расходов с 1984 по 2008 год (на базе данных Бюро переписи населения США). Горизонтальная ось по-прежнему обозначает время, но вместо профессий у нас теперь другие категории типа жилья и питания.

▶ Зайдите на `http://datafl.ws/16r` и опробуйте конечный вариант визуализации, чтобы увидеть, как работает диаграмма с потребительскими расходами.

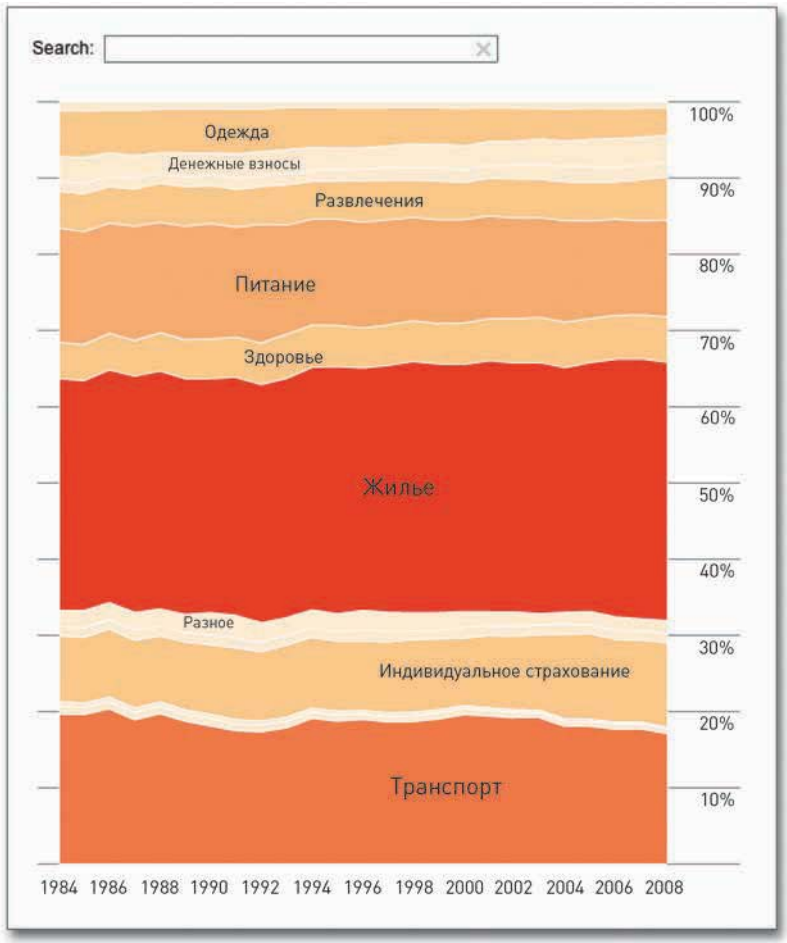


Рис. 5.32. Интерактивный «вояджер» потребительских расходов

Сначала вам необходимо изменить источник данных, который указан в 57-й строке JobVoyager.as.

```
private var _url:String = "http://flare.prefuse.org/data/jobs.txt";
```

Измените `_url` так, чтобы он указывал на имеющийся источник данных о потребительских тратах: `http://datasets.flowingdata.com/expenditures.txt`. Как и в `jobs.txt`, данные там хранятся в файле с разделителями в виде знаков табуляции. Первая колонка — это год, вторая — категория, а последняя — расходы.

```
private var _url:String =
   "http://datasets.flowingdata.com/expenditures.txt";
```

Теперь программа вместо данных по профессиям введет ваши данные по тратам. Пока что все идет гладко.

Следующие две строки, 58-я и 59-я, содержат имена, входящие в колонку, в данном случае — годы, по которым имеются данные о профессиях. Они выстроены по десятилетиям с 1850 по 2000 год. Лучше было бы найти соответствующие годы в загруженной информации, но, поскольку данные не меняются, вы можете сэкономить себе немного времени и задать годы «открытым текстом».

Данные по расходам собирались ежегодно с 1984-го по 2008-й, а потому вам нужно изменить строчки 58 и 59 соответствующим образом.

```
private var _cols:Array =
   [1984,1985,1986,1987,1988,1989,1990,1991,1992,
   1993,1994,1995,1996,1997,1998,1999,2000,2001,2002,
   2003,2004,2005,2006,2007,2008];
```

Следующее изменение, которое вы внесете, касается шапки данных. У исходного файла (jobs.txt) колонок четыре: год, профессия, количество людей и пол. Данные о тратах размещены только в трех колонках: год, категория и объем расходов. Вам необходимо адаптировать код к этой новой структуре данных.

К счастью, делается это легко. Колонка с годом остается в том же виде, вам достаточно будет вместо ссылок на количество людей вставить ссылки на объем расходов (вертикальная ось), а ссылки на род занятий заменить ссылками на категории (ярусы). И наконец, удалите все указания на пол.

На 74-й строчке данным придается новая форма — они подготавливаются для выстраивания штабельной диаграммы с областями. Данные специфицируются по профессии (occupation) и полу (sex), то есть определяются ярусы, и указывается, что годы (year) будут размещены по оси X, а люди (people) — по оси Y.

```
var dr:Array = reshape(ds.nodes.data, ["occupation","sex"],
   "year", "people", _cols);
```

Поменяйте эту строчку следующим образом:

```
var dr:Array = reshape(ds.nodes.data, ["category"],
   "year", "expenditure", _cols);
```

Теперь у вас есть только одна категория (без пола), и это... просто категория (category). Ось X все так же представляет годы, а ось Y — расходы (expenditure).

Строчка 84 сортирует данные по профессиям (в алфавитном порядке) и по полу (по номерам). Вам нужно сделать так, чтобы сортировка шла только по категории:

```
data.nodes.sortBy("data.category");
```

ПОДСКАЗКА

В сегменте визуализации появляются некоторые поистине классные приложения с открытым исходным кодом, и хотя перспектива написания кода поначалу может показаться вам обескураживающей, во многих случаях удается запросто использовать уже существующие коды для подстановки своих данных, просто меняя в них переменные. Сложность здесь заключается в том, чтобы прочитать код и понять, как он работает.

Начинаете улавливать идею? Уже почти все сделано за вас. Вам необходимо лишь подстроить переменные и подогнать данные.

Строчка 92 задает цвет по полу, но у вас в данных нет такого разделения, а потому вам это не нужно. Удалите строчку целиком:

```
data.nodes.setProperty("fillHue", iff(eq("data.sex",1), 0.7, 0));
```

К настройке цвета заливки ярусов мы вернемся несколько позже.

Строка 103 добавляет подписи на основе профессий (occupation).

```
_vis.operators.add(new StackedAreaLabeler("data.occupation"));
```

В нашем случае нужно, чтобы подписи отражали категорию (category) расходов, а потому измените строчку следующим образом:

```
_vis.operators.add(new StackedAreaLabeler("data.category"));
```

В JobVoyager строки 213–231 осуществляют фильтрацию. Сначала идет фильтр мужчина/женщина, затем фильтр по профессиям. Первый фильтр вам не нужен, так что избавьтесь от строк 215–218, а строку 219 превратите в простой условный оператор `if`.

И еще один аналогичный момент: строки 264–293 создают кнопки для активизации фильтра мужчина/женщина. От этих строк также следует избавиться.

Вы уже почти полностью закончили перестройку «вояджера» под данные о расходах. Вернитесь к функции `filter()` в 213-й строке. Вам необходимо снова подкорректировать функцию так, чтобы вы могли осуществлять фильтрацию по категории расходов, а не по профессии.

Вот как выглядит строка 222 до правки:

```
var s:String = String(d.data["occupation"]).toLowerCase();
```

Поменяйте профессию (occupation) на категорию (category):

```
var s:String = String(d.data[«category»]).toLowerCase();
```

Следующим пунктом в перечне задач идет цвет. Если вы скомпилируете код и запустите его сейчас, вы получите красноватого цвета штабельную диаграмму с областями, похожую на ту, что представлена на рис. 5.33. Однако нам нужно более контрастное изображение.
Цвета определяются в двух местах. Сначала строки 86–89 задают цвет обводки и окрашивают все в красный цвет:

```
shape: Shapes.POLYGON,
lineColor: 0,
fillValue: 1,
fillSaturation: 0.5
```

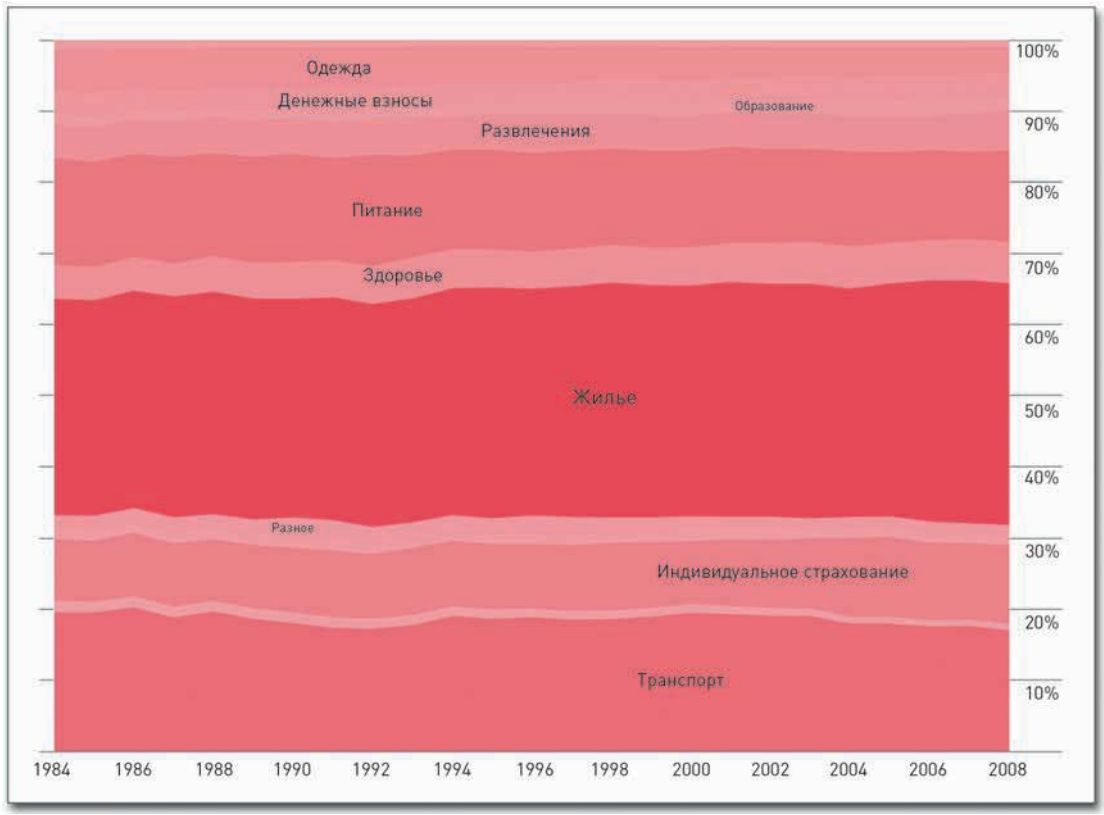


Рис. 5.33. Штабельная диаграмма с областями до изменения цветовой схемы

Затем строка 105 модифицирует насыщенность цвета (степень красного) по результатам подсчета. Код для `SaturationEncoder()` содержится в строчках 360–383. Принцип насыщения цвета нам не нужен. Вместо этого давайте зададим определенную цветовую схему.

Сначала измените строки 86–89 следующим образом:

```
shape: Shapes.POLY shape: Shapes.POLYGON,
ON,lineColor: 0xFFFFFFFF
```

А теперь с помощью `lineColor` сделайте цвет обводки белым. Если бы категорий расходов было больше, вы, наверное, не стали бы этого делать, чтобы не перегружать диаграмму. Но в данном случае их не так много, и белый цвет облегчит восприятие.

Создайте массив из цветов, которые вы хотите использовать, и выстройте их по ярусам. Вставьте этот фрагмент кода повыше, около 50-й строки:

```
private var _reds:Array = [0xFFFEF0D9, 0xFFFDD49E, 0xFFFDBB84, 0xFFFC8D59,
0xFFE34A33, 0xFFB30000];
```

Для определения этих цветов я использовал ColorBrewer (о котором уже рассказывал вам), он предлагает цветовые схемы на основе заданных критериев. Этот инструмент разрабатывался для подбора цветов при создании карт, но он прекрасно справляется также и с более общими задачами в области визуализации.

Добавьте новый ColorEncoder около строчки 110:

```
var colorPalette:ColorPalette = new ColorPalette(_reds);
vis.operators.add(new ColorEncoder("data.max", "nodes",
   "fillColor", null, colorPalette));
```

ПРИМЕЧАНИЕ

Если при попытке компиляции кода вы получите ошибку, вернитесь в начало JobVoyager.as, чтобы увидеть, заданы ли следующие две строчки для импортирования объектов Encoder и ColorPallete. Если нет, то добавьте их:

```
mport flare.util.palette.*;
import flare.vis.operator.encoder.*;
```

Вуаля! Теперь у нас есть нечто похожее на то, к чему мы стремились (рис. 5.32). Конечно, не следует останавливаться на достигнутом. Вы можете сделать еще много чего. Вы можете применить этот алгоритм к своим собственным данным, использовать другую цветовую схему и дальше подгонять код к своим собственным нуждам. Можете поменять шрифт или формат всплывающих подсказок. А еще вы можете проявить фантазию и интегрировать ваше творение с другими инструментами, добавить ActionScript — и т. д., и т. п.

Точка за точкой

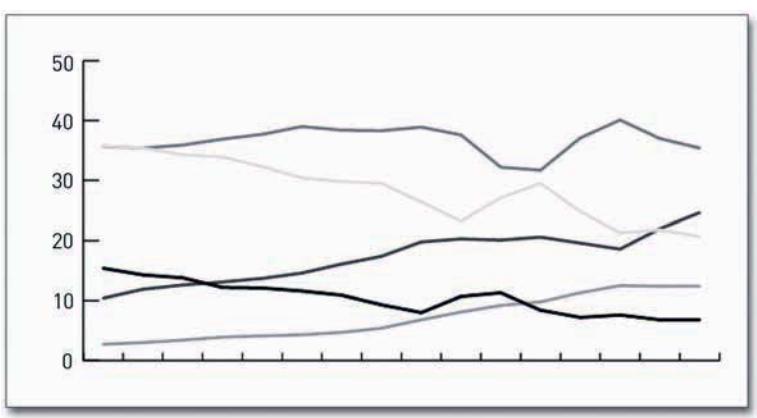


Рис. 5.34. Линейный график по умолчанию

Один из недостатков штабельных диаграмм с областями состоит в том, что в них тяжело разглядеть тенденции в той или иной отдельной группе, потому что местоположение каждой точки зависит от точек, которые ниже нее. А потому иногда бывает лучше представить пропорции как простые временны́е ряды, работу с которыми мы подробно рассматривали в предыдущей главе.

К счастью, переключаться между различными типами диаграмм в Illustrator легко. Принцип ввода данных одинаковый, так что вам всего лишь надо будет поменять тип диаграммы. Вместо штабельной диаграммы с областями выберите линейный график, и вы по умолчанию получите то, что изображено на рис. 5.34.

Подчистите его и отформатируйте на свой вкус точно так же, как вы это делали в примере с временны́ми рядами, и вы сможете посмотреть на те же самые данные с другой точки зрения (рис. 5.35).

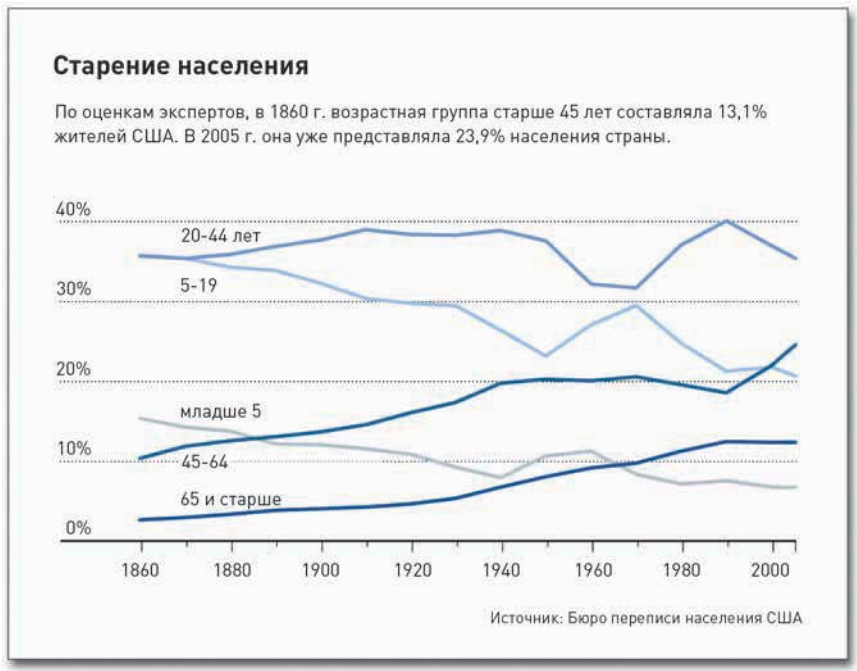


Рис. 5.35. Отредактированная линейная диаграмма с подписями

Теперь, с этой диаграммой временных рядов, стало легче рассмотреть отдельные тенденции в каждой возрастной группе. С другой стороны, потерялось ощущение целого и представление о том, как распределяются различные группы в населении. Какой вид диаграммы выбрать, зависит от того, какую идею вы собираетесь донести и что именно хотите найти в ваших данных. Если у вас достаточно много места, вы можете показать и оба варианта сразу.

Закругляясь

Главное, чем пропорции отличаются от других типов данных, состоит в том, что они представляют собой части целого. Каждая отдельная величина имеет свой смысл, как имеют смысл и сумма всех частей, и отдельные подмножества частей. Способ визуализации, который вы выберете, должен отражать эти идеи.

У вас всего несколько значений? Тогда, возможно, круговая диаграмма — ваш лучший выбор. Кольцевые диаграммы используйте с осторожностью. Если у вас несколько значений

и несколько категорий, тогда подумайте о создании не большого количества круговых диаграмм, а штабельной диаграммы с областями. Если вы выискиваете паттерны во времени, тогда обратитесь к штабельной диаграмме с областями или к классической диаграмме временных рядов. С таким прочным фундаментом ваши пропорции будут готовы блеснуть во всей красе.

Когда же дело дойдет до разработки и реализации, спросите себя, что вам необходимо узнать о ваших данных, и действуйте далее, исходя из этого. Сможет ли статичная диаграмма рассказать вашу историю в полном объеме? Во многих случаях ответ будет «да», и это замечательно. Однако если вы решите, что вам необходима некая форма интерактивной графики, составьте себе на бумаге план того, что именно должно происходить во время щелчков мышью по объекту, а что — не должно. Если вы придадите диаграмме слишком много функциональных возможностей, все очень быстро запутается, так что постарайтесь придерживаться принципа: лучше проще, да лучше. Обратитесь за помощью к другим людям: пусть они попробуют повзаимодействовать с вашим детищем, а вы посмотрите, понимают ли они, что происходит.

И наконец, когда вы станете писать программу — особенно если вы новичок в составлении кодов, — вы, несомненно, дойдете до того момента, когда не будете знать, что делать дальше. Со мной это происходит постоянно. Когда вы увязнете, вам не найти лучшего места для поиска решений, чем Сеть. Просмотрите документацию, если она доступна, или изучите примеры, схожие с той задачей, которую вы пытаетесь выполнить. Обращайте внимание не только на синтаксис. Изучите логику, потому что она для вас наиболее полезна. К счастью, существуют такие библиотеки, как Protovis и Flare, которые содержат множество примеров и имеют прекрасную документацию.

В следующей главе мы перейдем к более глубокому анализу и интерпретации данных и вернемся к старому доброму R. Вы зададите ему изрядно работы, когда станете изучать зависимости между наборами данных и переменными. Готовы? Поехали.

Визуализация зависимостей 6

Поиск зависимостей между данными — этим как раз и занимается статистика. Каковы сходства и отличия между несколькими группами? А внутри групп? А внутри подгрупп? Один из видов зависимостей в статистике — корреляция — знаком большинству людей. Например, по мере того как средний рост населения увеличивается, можно ожидать, что увеличиваться будет и средний вес. Это пример простой положительной корреляции. Однако зависимости в ваших данных, так же как и в реальной жизни, могут стать сложнее, когда вы начнете учитывать больше факторов или находить паттерны, которые не столь линейны. Эта глава посвящена тому, как использовать визуализацию для выявления подобных зависимостей и чем они могут оказаться полезны для сторителлинга.

По мере того как в этой и последующих главах вы будете осваивать все более сложную статистическую графику, вам придется все активнее использовать R. Вот здесь-то программное обеспечение с открытым исходным кодом и проявит себя во всей своей красе. Как и в предыдущих главах, R выполнит за вас всю тяжелую работу, а вы затем сможете воспользоваться программой Illustrator и сделать графику более читабельной для аудитории.

Какие зависимости искать

До сих пор вы занимались исследованием зависимостей и выявлением паттернов во времени, а также изучением пропорций. Вы узнали, как визуализировать тенденции во временны́х рядах и каким образом следует сравнивать пропорции, чтобы выяснить, что больше, что меньше и как все выстраивается между этими двумя крайностями. На следующем этапе вы научитесь искать зависимости между различными переменными. Когда что-нибудь поднимается, обязательно ли что-то другое падает, и о какой связи между этими процессами в таком случае можно говорить — о причинно-следственной или о корреляции? Первое бывает обычно довольно трудно доказать количественными данными, а графикой — тем более. А вот корреляция, наоборот, легко поддается визуализации и тем самым дает возможность провести более глубокий анализ.

Вы можете также сделать шаг назад, чтобы взглядом охватить всю картину — или распределение ваших данных — целиком. И понять, какова ситуация в действительности: идут ли данные с широким разбросом или они сгруппированы с очень маленькими интервалами. Подобные сравнения способны помочь вам рассказать увлекательные истории о жителях разных стран или о том, каковы лично вы по сравнению с теми, кто вас окружает. Вы сможете понять, как государства соотносятся друг с другом или как развивается мир в целом, и это позволит вам, например, принимать более взвешенные решения о том, куда следует направлять гуманитарную помощь.

Вы сможете провести сравнения нескольких распределений и тем самым получить еще более широкое представление о ваших данных. Как меняется структура населения во времени? Или как она остается неизменной?

И, что самое важное, в конце, когда все диаграммы и графики будут уже лежать перед вами, спроси́те себя: что означают полученные результаты? Они похожи на то, что вы ожидали? Вас ничего не удивляет?

Возможно, все это сейчас звучит для вас абстрактно и витиевато, а потому давайте сразу же погрузимся в конкретный пример и посмотрим, как исследовать зависимости в данных.

Корреляция

Корреляция — наверное, это первое, о чем вы подумали, когда услышали про поиск зависимостей в данных. Второе — скорее всего, это причинно-следственная связь. А теперь вы наверняка вспомнили мантру о том, что корреляция и причинно-следственная связь — это «две большие разницы». Первое — корреляция — означает, что некий объект склонен меняться определенным образом вслед за изменениями, наблюдаемыми в другом объекте. Например, между ценой литра молока и ценой литра бензина существует положительная корреляция. И то, и другое с годами повышается. Различие между корреляцией и причинно-следственной связью заключается вот в чем. Если вы увеличите цены на бензин, означает ли это, что и цены на молоко поднимутся? И, что еще более важно: если цены на молоко действительно поднялись, произошло это из-за увеличения цен на бензин или по причине некоего внешнего фактора, такого, например, как забастовка молочников?

Очень непросто учесть все внешние, или вмешивающиеся, факторы (конфаундеры), которые затрудняют процесс доказательства причинно-следственной связи. У ученых на решение подобных проблем порой уходят годы. А вот корреляцию вы можете найти и продемонстрировать довольно легко, что также бывает полезно, и далее вы в этом убедитесь сами.

Корреляция способна помочь вам спрогнозировать некие количественные показатели на основе имеющихся сведений о других количественных показателях. Чтобы рассмотреть эту зависимость, давайте вернемся к диаграммам рассеяния.

Еще с точками

В главе 4, «Визуализация паттернов во времени», для наглядного представления измерений во времени вы использовали диаграмму рассеяния. Там время располагалось по горизонтальной оси, а интересующие вас количественные показатели — по вертикальной. Данный подход помогал разглядеть темпоральные изменения (или убедиться в отсутствии таковых). Тогда вы исследовали зависимость между временем и другим фактором или переменной. Однако, как видно на рис. 6.1, диаграмму рассеяния можно использовать и для других типов данных, а не только для времени — например, для визуализации зависимостей между двумя переменными.

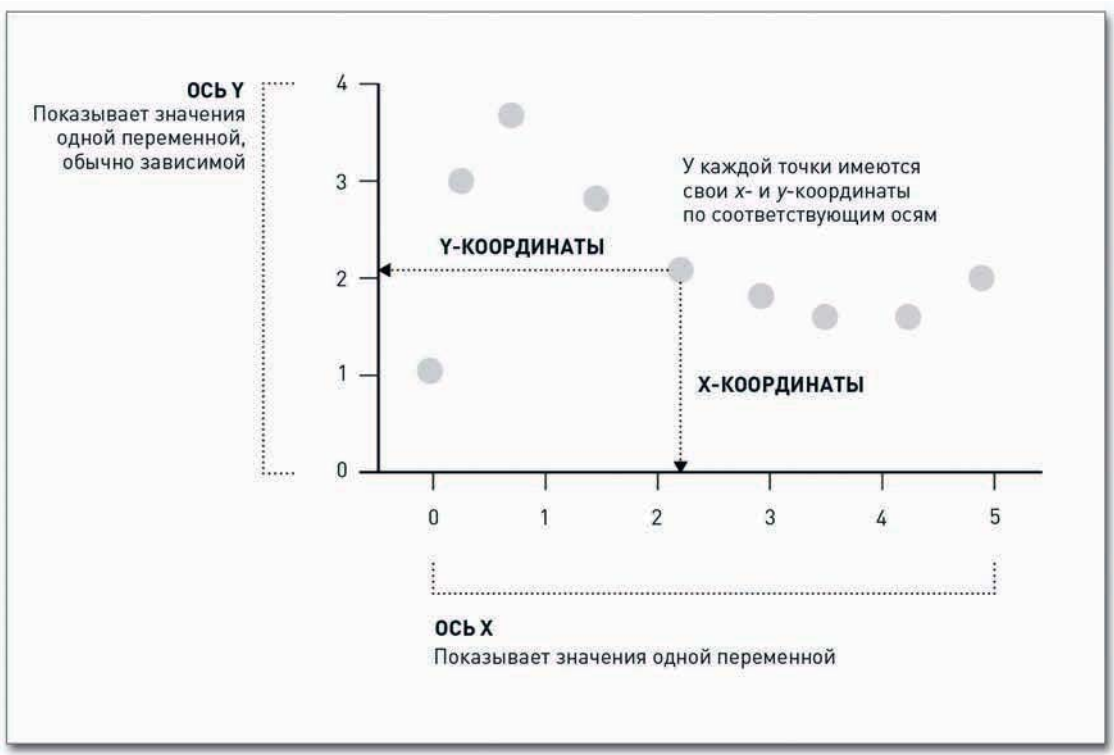


Рис. 6.1. Структура диаграммы рассеяния для сравнения двух переменных

Если между двумя рядами количественных данных существует положительная корреляция (на рис. 6.2 слева), то когда вы станете читать диаграмму слева направо, точки будут подниматься выше и выше. И наоборот, если существует отрицательная корреляция, тогда по мере продвижения слева направо точки будут опускаться все ниже и ниже (как показано на средней диаграмме на рис. 6.2).

Рис. 6.2. Виды корреляции, демонстрируемые диаграммами рассеяния

Иногда зависимость бывает прямой, как, скажем, зависимость между ростом и весом людей. Ведь по мере увеличения роста обычно увеличивается также и вес. Но случаются ситуации, когда зависимость не столь очевидна, например, как в случае со здоровьем человека и индексом массы его тела. Высокий индекс массы тела обычно свидетельствует о том, что у человека избыточный вес. Однако люди с сильно развитой мускулатурой, активно занимающиеся спортом, тоже могут иметь высокий индекс массы тела. Допустим, если обследуемая совокупность состоит из культуристов и футболистов. Как в этом случае будет выглядеть зависимость между здоровьем и индексом массы тела?

Не забывайте, что диаграмма — это всего лишь часть истории. И по-прежнему от вас зависит, как интерпретировать полученные результаты. Данное правило тем более следует учитывать, когда речь идет о зависимостях. Вы можете поддаться искушению и допустить, что налицо причинно-следственная связь, но в большинстве случаев это будет не так. Тот факт, что со временем и цены на бензин, и народонаселение мира растут, вовсе не означает, что для снижения темпов роста населения необходимо снизить цены на бензин.

СОЗДАЙТЕ ДИАГРАММУ РАССЕЯНИЯ

В этом примере мы рассмотрим уровень преступности в США по штатам за 2005 год в пересчете на 100 000 человек (по данным Бюро переписи населения). Всего мы рассмотрим семь типов преступлений, а именно: убийства (murder), изнасилования (forcible rape), грабежи (robbery),

нападения с применением физического насилия (aggravated assault), квартирные кражи со взломом (burglary), кражи имущества (larceny-theft) и кражи транспортных средств (motor vehicle theft). Приглядитесь для начала к двум из них: к квартирным кражам со взломом и к убийствам. Как они связаны между собой? Можно ли ожидать, что в штатах с относительно высоким уровнем убийств будет наблюдаться также и высокий уровень квартирных краж? Чтобы выяснить это, давайте обратимся к R.

Как всегда, первым делом необходимо загрузить данные в R с помощью `read.csv()`. Нужный CSV-файл вы можете найти по адресу `http://datasets.flowingdata.com/crimeRatesByState2005.csv`, но на этот раз вы загрузите его напрямую в R, указав URL.

```
# Load the data
crime <-
   read.csv('http://datasets.flowingdata.com/crimeRatesByState2005.csv',
   sep=",", header=TRUE)
```

Проверьте первые несколько строчек данных, набрав переменную **`crime`** и следом за ней указав строчки, которые хотите посмотреть.

```
crime[1:3,]
```

Вот как будут выглядеть первые три строчки данных:

```
        state           murder       forcible_rape   robbery   aggravated_assault   burglary
 1   United States        5.6             31.7         140.7          291.1            726.7
 2   Alabama              8.2             34.3         141.4          247.8            953.8
 3   Alaska               4.8             81.1          80.9          465.1            622.5
     larceny_theft   motor_vehicle_theft    population
 1   2286.3              416.7           295 753 151
 2   2650.0              288.3            4 545 049
 3   2599.1              391.0              669 488
```

В первой колонке идет название штата, а далее представлены данные по различным типам преступлений. Например, средний по США уровень преступности по части грабежей в 2005 году составлял 140,7 на 100 000 жителей. Используя `plot()`, создайте диаграмму рассеяния по умолчанию, сопоставив данные по убийствам (murder) и по кражам со взломом (burglary), как это показано на рис. 6.3.

```
plot(crime$murder, crime$burglary)
```

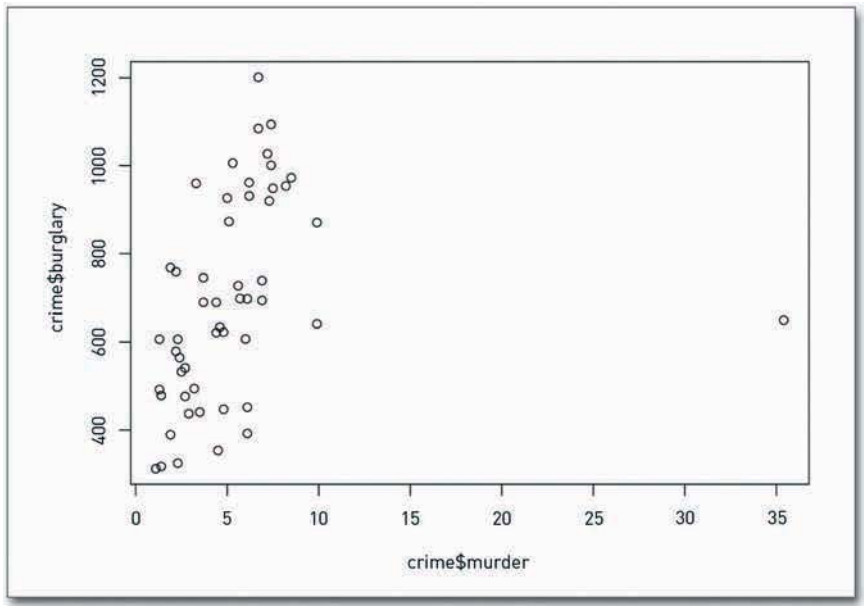


Рис. 6.3. Диаграмма рассеяния по умолчанию, сопоставляющая убийства с кражами со взломом

Похоже, что здесь есть положительная корреляция. Штаты, в которых уровень преступности в части убийств выше, склонны демонстрировать также и более высокий уровень по кражам со взломом, но разглядеть данную тенденцию непросто из-за одной точки на правой стороне диаграммы. Эта одинокая точка — или выброс — вынуждает горизонтальную ось сильно вытягиваться в ширину. Точка представляет Вашингтон, округ Колумбия , где уровень по убийствам весьма высок — 35,4. В следующих сразу за ним штатах — Луизиане и Мэриленде — данный показатель составляет 9,9.

Чтобы диаграмма получилась более наглядной и полезной, удалите Вашингтон, а заодно, пока вы все еще там, удалите и средние по США показания и сконцентрируйтесь только на данных по отдельным штатам.

```
crime2 <- crime[crime$state != "District of Columbia",]
crime2 <- crime2[crime2$state != "United States",]
```

Первая строчка сохраняет все ряды, которые не касаются округа Колумбия в *crime2*. Аналогичным образом вторая строчка отфильтровывает средние значения по Соединенным Штатам в целом.

Теперь, выстроив диаграмму, сопоставляющую убийства с кражами со взломом, вы получите уже более ясную картину, как показано на рис. 6.4.

```
plot(crime2$murder, crime2$burglary)
```

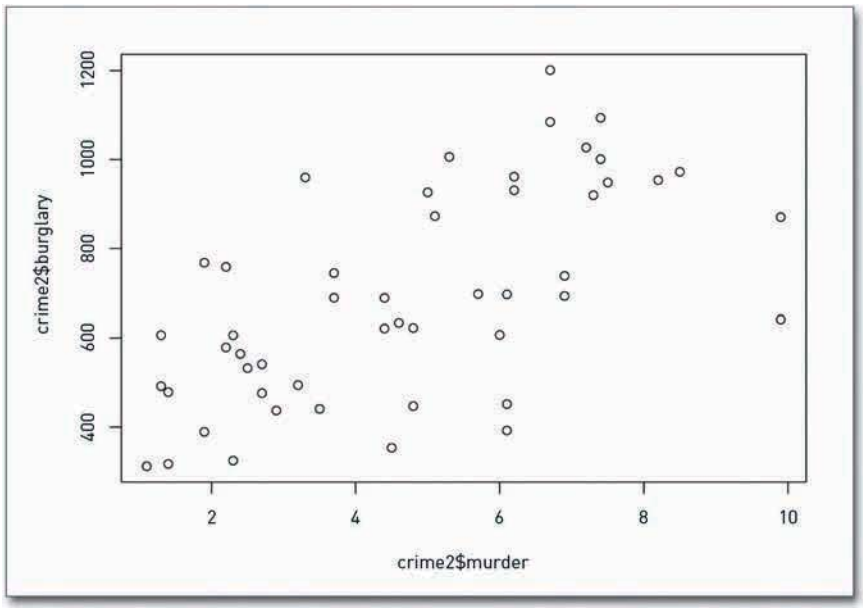


Рис. 6.4. Диаграмма рассеяния после фильтрации данных

Хотя было бы лучше, если б оси начинались с нуля, так что подкорректируйте и это тоже. Ось X должна иметь охват от 0 до 10, а ось Y — от 0 до 1200. Таким образом точки сместятся выше и правее, как показано на рис. 6.5.

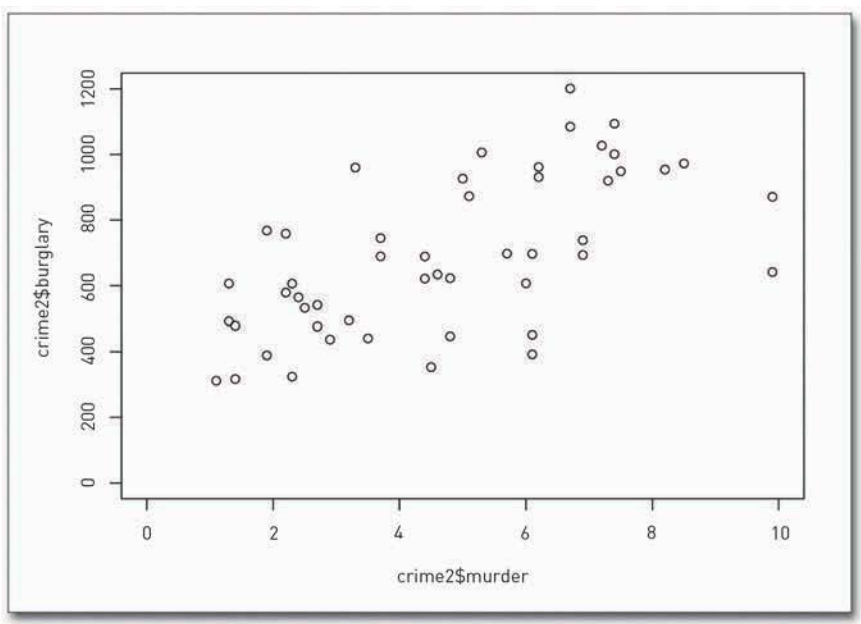


Рис. 6.5. Диаграмма рассеяния с осями, начинающимися с нуля

```
plot(crime2$murder, crime2$burglary, xlim=c(0,10), ylim=c(0,1200))
```

А знаете, как можно сделать эту диаграмму еще более наглядной? С помощью кривой, подобранной по методу локально взвешенного сглаживания диаграммы рассеяния (ЛВСДР), вроде той, с которой вы ознакомились в четвертой главе. Она поможет вам увидеть зависимость между показателями по убийствам и по кражам со взломом. Результаты представлены на рис. 6.6.

```
scatter.smooth(crime2$murder, crime2$burglary,
   xlim=c(0,10), ylim=c(0,1200))
```

ПРИМЕЧАНИЕ

Для простоты я удалил Вашингтон из массива данных, чтобы можно было лучше разглядеть все остальные показатели. Однако выбросы в данных имеют большое значение и им следует уделять особое внимание. Подробнее мы поговорим об этом в главе 7, «Выявление различий».

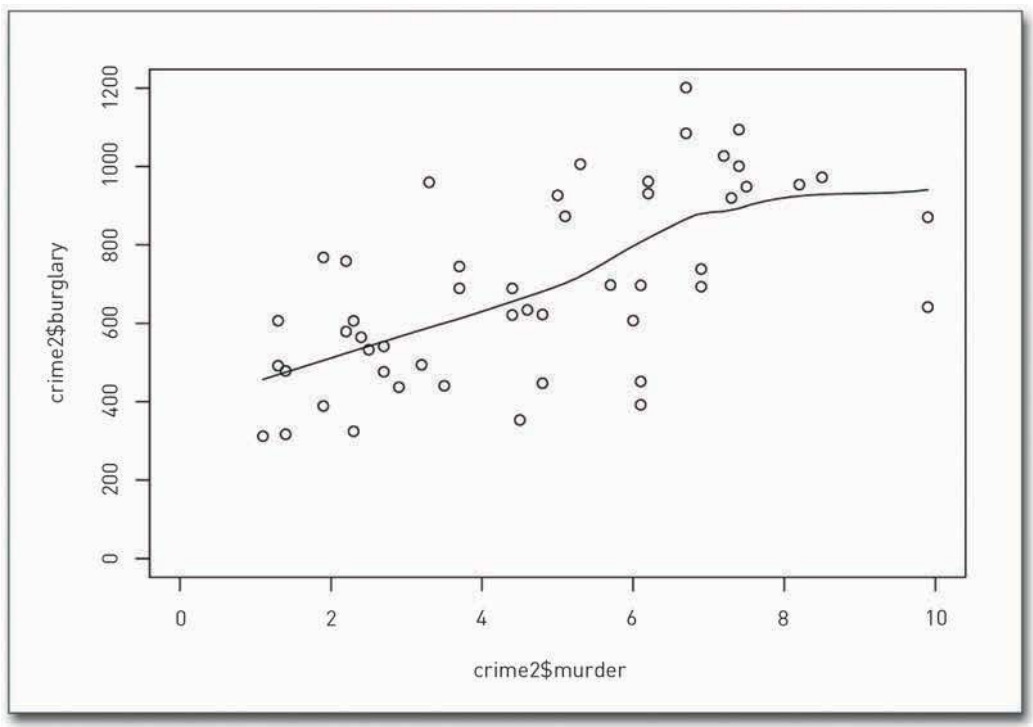


Рис. 6.6. Использование кривой для оценки зависимости

Совсем неплохо для базового варианта диаграммы. И если бы графика создавалась только для целей анализа, вы могли бы на этом остановиться. Тем не менее если аудитория у вас состоит не из одного человека, тогда вам придется существенно улучшить читабельность диаграммы, внеся несколько небольших, но значимых изменений, как показано на рис. 6.7.

Я снял визуальный акцент с рамок изображения, полностью избавившись от линий границ, а саму кривую сделал толще и темнее, чем окружающие ее точки, чтобы направить внимание именно на нее.

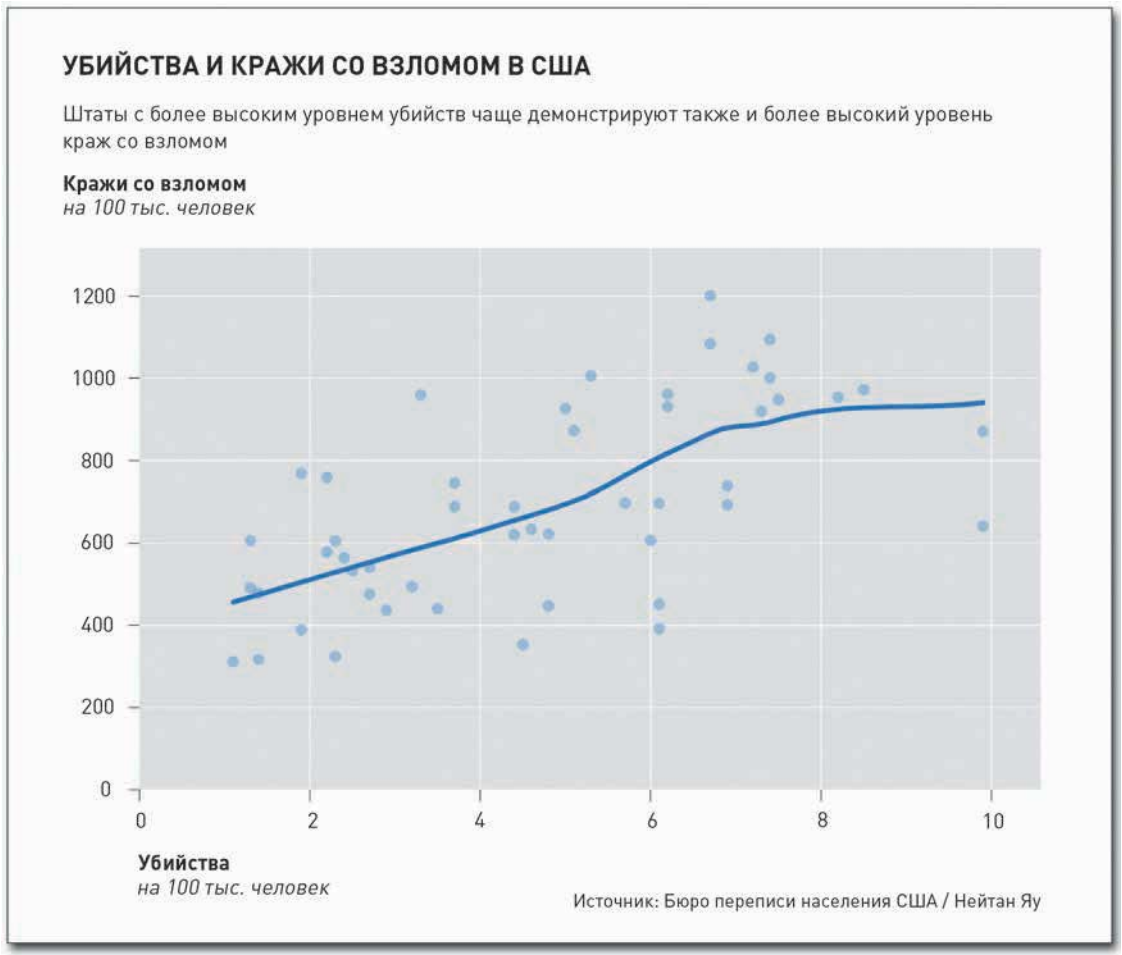


Рис. 6.7. Переработанная диаграмма рассеяния, сопоставляющая данные по убийствам и по кражам со взломом

Исследование нескольких переменных

Теперь, когда вы выстроили диаграмму, сопоставляя две переменные, вполне очевидно, что следующим шагом станет сравнение нескольких переменных. Вы могли бы по-прежнему выбирать и сравнивать переменные попарно, создавая диаграммы рассеяния для всех этих пар поочередно, но подобная практика может в итоге окончится упущенными возможностями и проигнорированными интересными моментами в ваших данных. А допустить это вам вряд ли захочется. Вот почему, чтобы получить все возможные парные сравнения, я предлагаю вам воспользоваться матрицей диаграмм рассеяния (рис. 6.8).

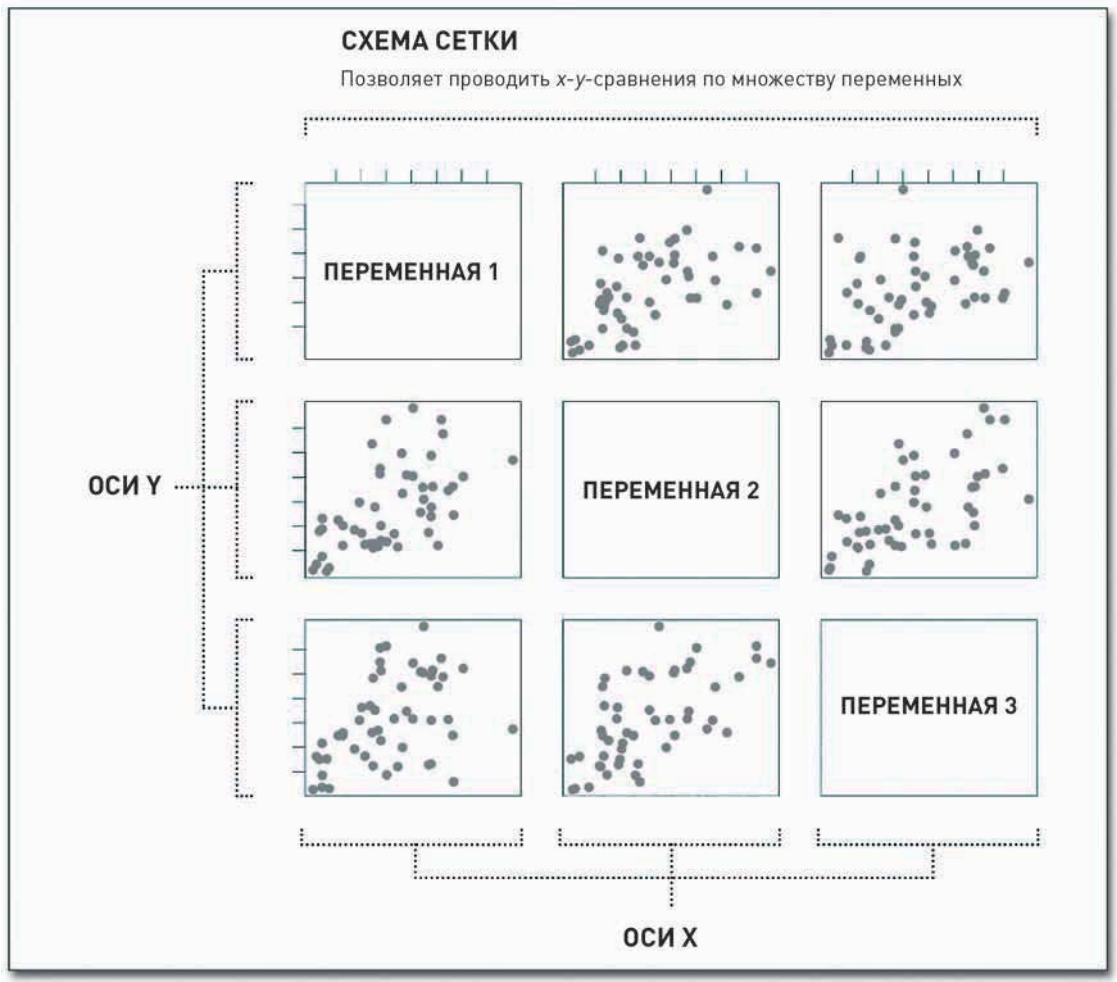


Рис. 6.8. Структура матрицы диаграмм рассеяния

ПОДСКАЗКА

Чтобы рассказать полную историю, вы должны понимать данные. Чем больше вы о них будете знать, тем лучше получится история, которую сможете поведать.

Данный метод особенно полезен на этапе изучения данных. Бывают ситуации, когда перед вами лежит массив данных, а вы и понятия не имеете, с чего начать и вообще о чем здесь речь. Если вы сами не знаете, о чем эти данные, то ваши читатели и подавно.

Матрица диаграмм рассеяния выглядит так, как вы и ожидали. Обычно это квадратная сетка со всеми переменными, выстроенными по вертикали и по горизонтали. Каждая колонка представляет одну переменную по горизонтальной оси, а каждый ряд — одну переменную по вертикальной оси. Таким образом вы получаете все возможные пары, а диагональ остается для подписей, так как нет никакого смысла сравнивать переменную с самой собой.

СОЗДАЙТЕ МАТРИЦУ ДИАГРАММ РАССЕЯНИЯ

Теперь давайте вернемся к данным о преступности. У вас есть семь переменных (или показателей по разным типам преступлений), но в предыдущем примере вы сравнивали только два из них: убийства и кражи со взломом. Теперь же с матрицей диаграмм рассеяния вы можете сопоставить все виды преступлений. На рис. 6.9 показано, что́ вы получите в итоге.

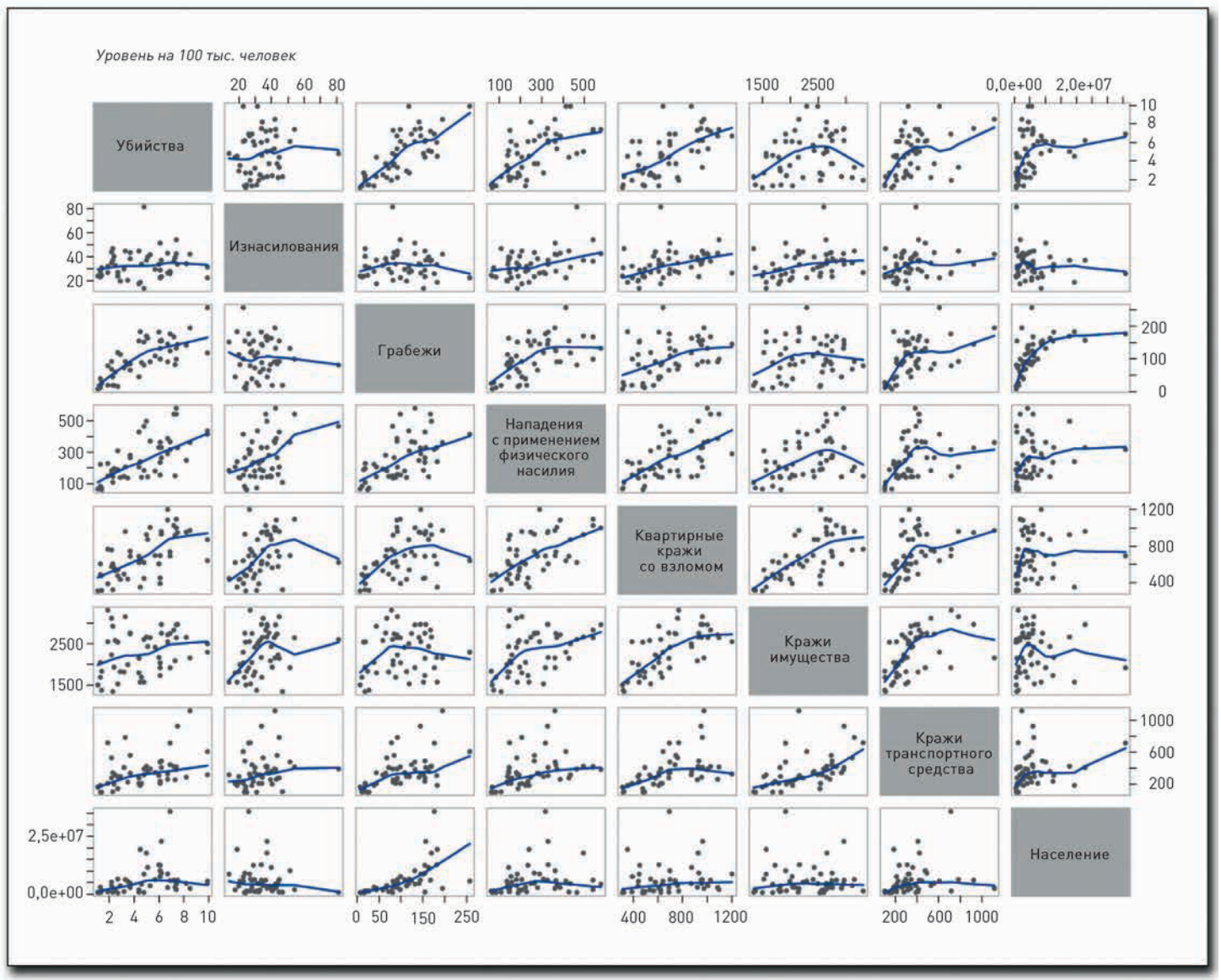


Рис. 6.9. Матрица диаграмм рассеяния для сопоставления уровней различных типов преступности

Как вы, возможно, и ожидали, налицо множество положительных корреляций. Например, между кражами со взломом и нападениями с применением физического насилия, похоже, налицо весьма высокая степень корреляции. Когда первый показатель повышается, второй также устремляется вверх, и наоборот. А вот связь между убийствами и кражами имущества не так очевидна. Тут не нужно быть семи пядей во лбу, чтобы понять: матрица диаграмм рассеяния может оказаться весьма полезной. На первый взгляд кажется, что все несколько запутанно с этими линиями и диаграммами, но начните читать матрицу слева направо и сверху вниз, и вы сможете извлечь из нее массу ценной информации.

К счастью, R превращает создание матричных диаграмм рассеяния в столь же простое дело, как и начертание единичной диаграммы, хотя процесс создания матрицы протекает все же не так быстро. Вы снова воспользуетесь функцией `plot()`, но вместо двух колонок загрузите всю таблицу данных за исключением первой колонки, так как она содержит только названия штатов.

```
plot(crime2[,2:9])
```

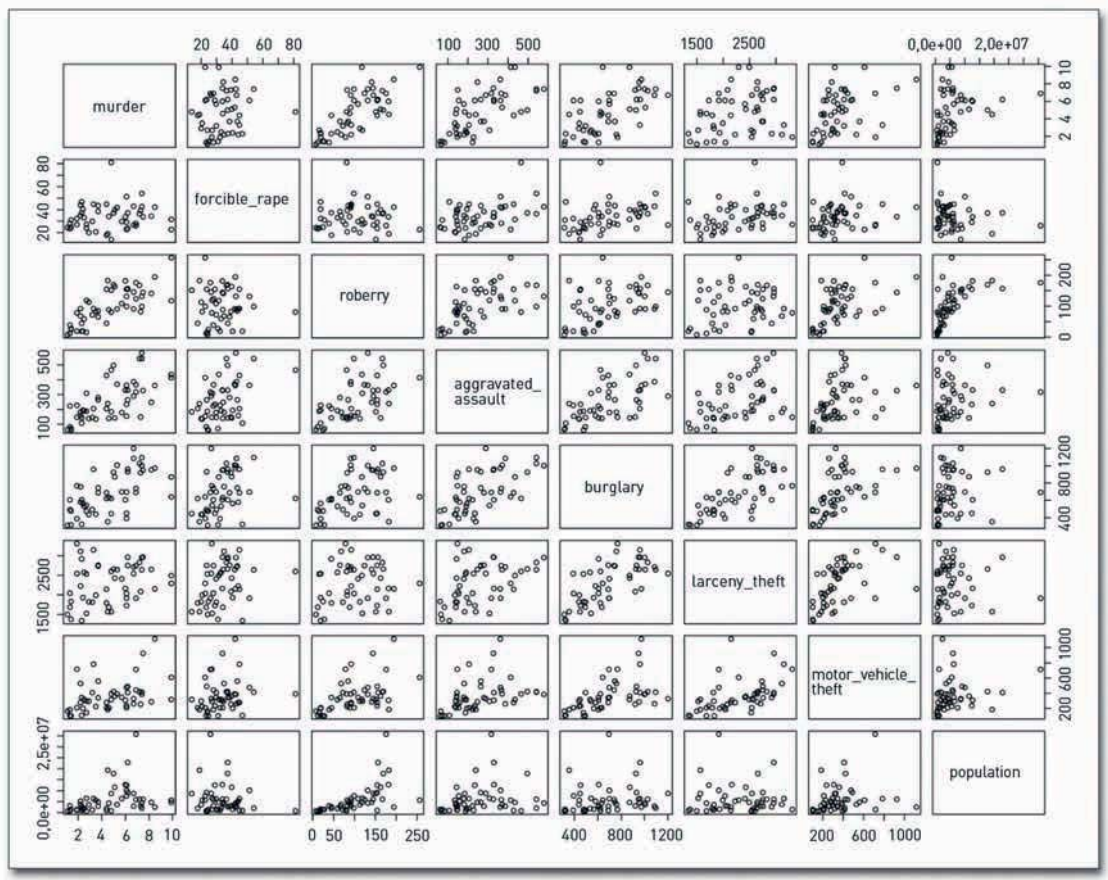


Рис. 6.10. Матрица диаграмм рассеяния, созданная в R по умолчанию

Этот шаг даст вам матрицу, представленную на рис. 6.10, которая весьма похожа на то, к чему вы стремитесь. Но ей пока что недостает подобранных кривых, которые помогают продемонстрировать зависимость более наглядно.

Чтобы создать матрицу со сглаженными кривыми, вместо функции, к которой вы обращались перед этим, на сей раз вы воспользуетесь функцией `pairs()`. Не беспокойтесь, сама процедура останется столь же простой, как и прежде. Результат представлен на рис. 6.11.

```
pairs(crime2[,2:9], panel=panel.smooth)
```

ПОДСКАЗКА

Аргумент `panel` в `pairs()` позволяет назначить функцию для преобразования данных и отображения их в виде *x*- и *y*-координат. В данном примере вы будете использовать `panel.smooth()`, являющуюся для R «родной» функцией и выдающую в результате сглаженную кривую. Но вы можете задать здесь и некую другую функцию.

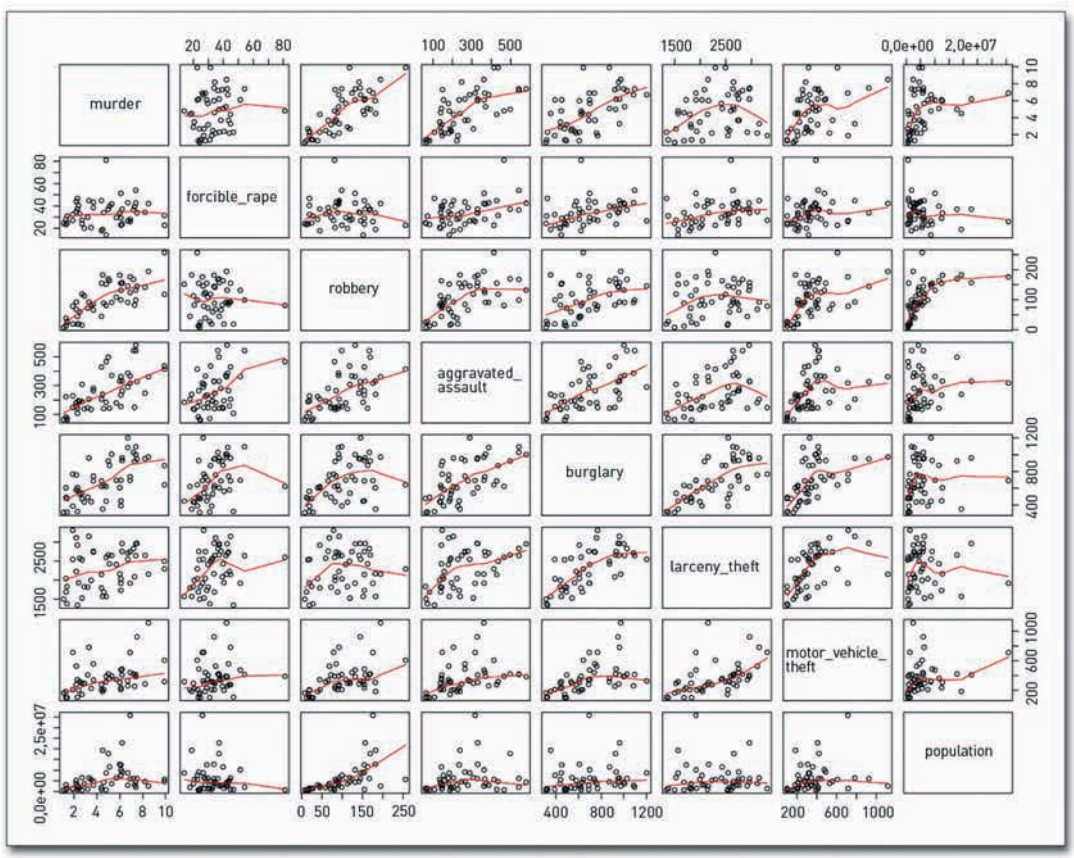


Рис. 6.11. Матрица диаграмм рассеяния со сглаженными кривыми

ПОДСКАЗКА

Решите, какую часть истории вы хотите рассказать, и придайте графике такой вид, который привлечет внимание к нужным моментам, только позаботьтесь, чтобы при этом другие факты не оказались вовсе вне поля зрения.

Теперь у вас уже есть замечательная основа для того, чтобы завершить работу и сделать графику более читабельной, ориентируясь на рис. 6.9. Сохраните полученное изображение как PDF и откройте файл в Illustrator.

Самое главное, что вам предстоит, — это избавиться от всего, перегружающего графику, и сделать так, чтобы самое важное стало также и самым заметным. А это, в первую очередь, типы преступлений и линии тренда, во вторую очередь — точки и наконец — оси. Причем этот порядок следует соблюдать при выборе и цвета, и размера. Так, размер подписей по диагонали следует увеличить, а ячейки окрасить в серый. Затем надо будет сделать линии тренда более толстыми и увеличить контраст между линиями и точками. И наконец — осветлить границы и линии сетки, сделав тоньше штрих и изменив цвет на светло-серый. Сравните рис. 6.9 с рис. 6.11 — визуальная вибрация в нем стала намного слабее, правда?!

Пузырьки

▶ Просмотрите ставшие уже знаменитыми выступления Ханса Рослинга на сайте Gapminder по адресу `www.gapminder.org`, в том числе и документальный фильм BBC об удовольствии занятия статистикой.

С тех пор как Ханс Рослинг, профессор в области международного здравоохранения Каролинского института и директор фонда Gapminder, представил свою историю о благосостоянии и здравоохранении в различных странах мира с помощью динамической диаграммы, пузырьковое отображение пропорций по осям X и Y обрело значительную популярность. И хотя динамические диаграммы используют анимацию с целью продемонстрировать изменения во времени, вы можете создать также и статичный вариант такой диаграммы, а именно — пузырьковую диаграмму.

Пузырьковая диаграмма может состоять и просто из пузырьков, чьи размеры пропорциональны неким значениям, но давайте сейчас подумаем о том, как можно создать разновидность такой диаграммы, которая походила бы на диаграмму рассеяния, но при этом имела еще и третье, «пузырьковое» измерение.

Преимущество данного типа диаграмм состоит в том, что они дают возможность сравнивать три переменные одновременно, как это показано на рис. 6.12. Одна переменная располагается на оси X, другая — на оси Y, а выражением третьей служит размер площади пузырьков.

ПОДСКАЗКА

При использовании кругов для визуализации данных определяйте их размер по площади, а не по диаметру, радиусу или окружности.

На последнюю деталь обратите особое внимание, так как люди часто определяют размер пузырька неправильно. Как мы уже говорили в главе 1, «Как рассказать историю с помощью данных», размер пузырьков необходимо задавать по площади. Их не следует измерять по радиусу, диаметру или окружности. А ведь допустить подобную ошибку легко. Если вы будете пользоваться настройками вашего программного обеспечения по умолчанию, все может кончиться тем, что пузырьки у вас получатся чересчур большими или чересчур маленькими.

А теперь посмотрите на простой пример, иллюстрирующий сказанное. Представьте себе, что вы отвечаете за рекламу в вашей компании и должны протестировать два баннера на одном сайте. Вам нужно узнать, какой из них работает лучше. В течение месяца оба рекламных объявления были показаны одинаковое количество раз, однако число кликов по ним оказалось разным. Первый баннер принес 150 кликов, а второй — 100 кликов. Следовательно, первый баннер

принес на 50 процентов больше кликов. На рис. 6.13 показаны два круга, по одному за каждый баннер, размеры которых определялись по площади. Круг, представляющий первый баннер, на 50 процентов больше второго.

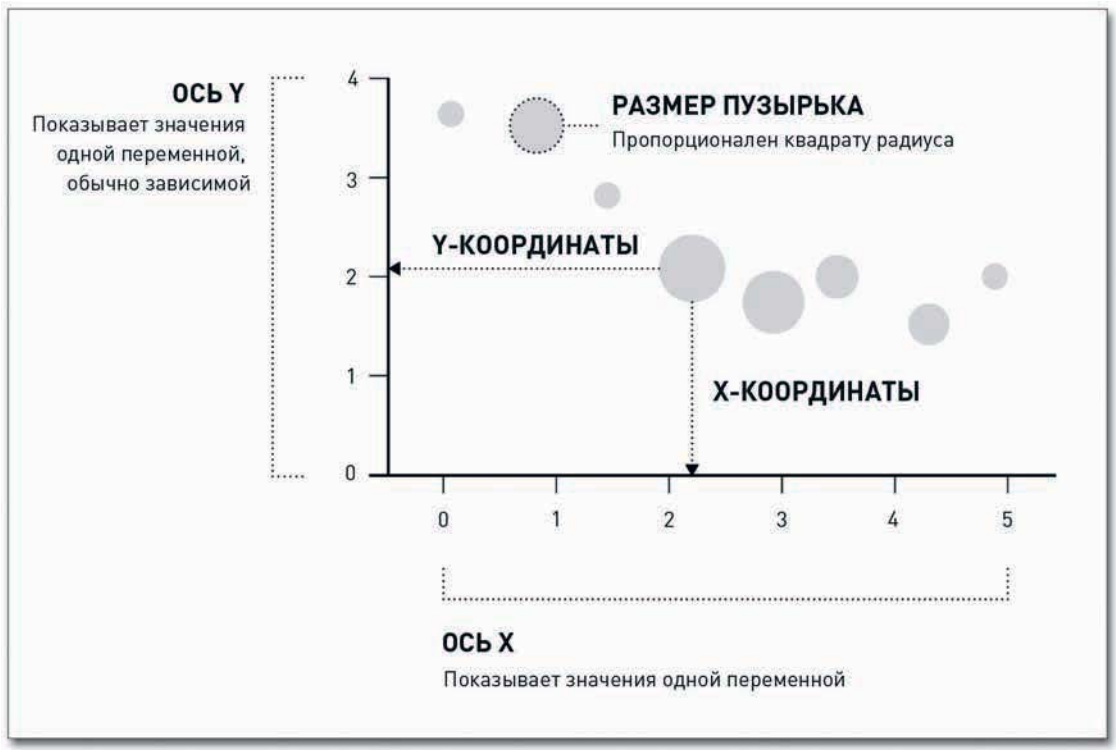


Рис. 6.12. Структура пузырьковой диаграммы

На рис. 6.14 вы видите, как соотносятся друг с другом пузырьки, если определять их размер по радиусу.

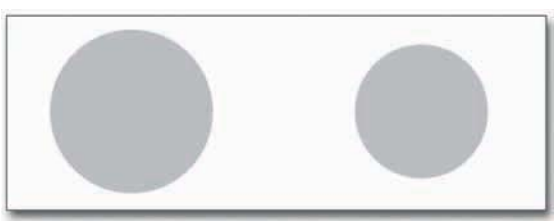


Рис. 6.13. Пузырьки с размером, заданным по площади

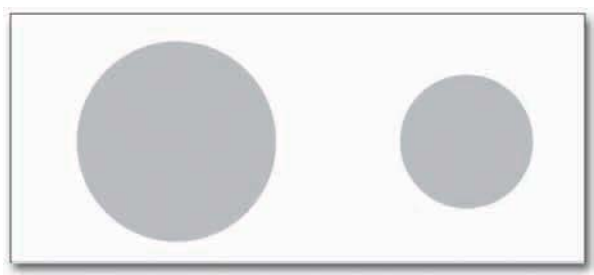


Рис. 6.14. Пузырьки с размером, заданным по радиусу

Радиус первого круга, представляющего первый баннер, на 50 процентов больше радиуса второго круга-баннера, в результате чего площадь первого получилась в два раза больше площади второго. Сейчас, когда вы сравниваете всего два элемента, которые и так нетрудно сопоставить, это кажется ерундой, но когда вы начнете работать с бóльшим количеством данных, проблема станет очевидной.

СОЗДАЙТЕ ПУЗЫРЬКОВУЮ ДИАГРАММУ

Посмотрите на окончательный вариант диаграммы, представленный на рис. 6.15, чтобы увидеть, чего нам предстоит добиваться. Это те же самые данные об уровне преступности по части убийств и краж со взломом в различных штатах, но в них было включено также и население штатов в качестве третьей величины. Можно ли говорить, что уровень преступности выше в штатах с бóльшим количеством населения? Ответ отнюдь не очевиден (как, впрочем, обычно и бывает). Большие штаты, такие, как Калифорния, Флорида и Техас, находятся довольно близко к верхнему правому квадранту, а вот в Нью-Йорке и Пенсильвании уровень краж со взломом относительно низок. Аналогичным образом обстоит ситуация с Луизианой и Мэрилендом — у них население поменьше, но они намного «правее» других.

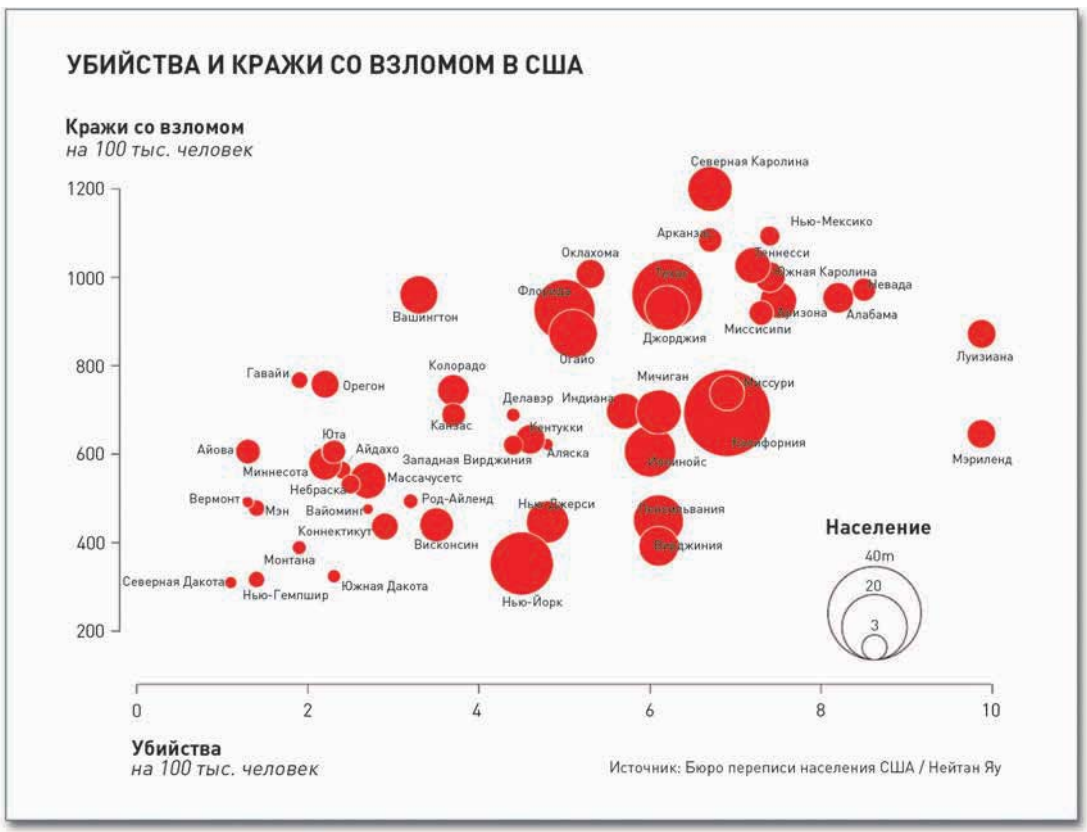


Рис. 6.15. Пузырьковая диаграмма, демонстрирующая уровень преступности в США

Для начала загрузите данные в R с помощью `read.csv()`. Это те же самые данные, которые вы только что использовали, с той лишь разницей, что здесь также добавлена колонка с численностью населения и в массив не включен Вашингтон. А еще в данном случае в качестве разделителя в файле используется табуляция, а не запятые. Ничего страшного. Просто измените в функции аргумент *sep*.

```
crime <-
   read.csv("http://datasets.flowingdata.com/crimeRatesByState2005.tsv",
   header=TRUE, sep="\t")
```

А затем сразу же приступайте к созданию пузырьков с помощью функции `symbols()`. На оси X представлен уровень убийств, на оси Y — уровень краж со взломом, а радиус пузырьков, соответственно, пропорционален численности населения в различных штатах. Результат показан на рис. 6.16. Хотите еще поиграть с `symbols()`? Скоро вам представится такая возможность.

```
symbols(crime$murder, crime$burglary, circles=crime$population)
```

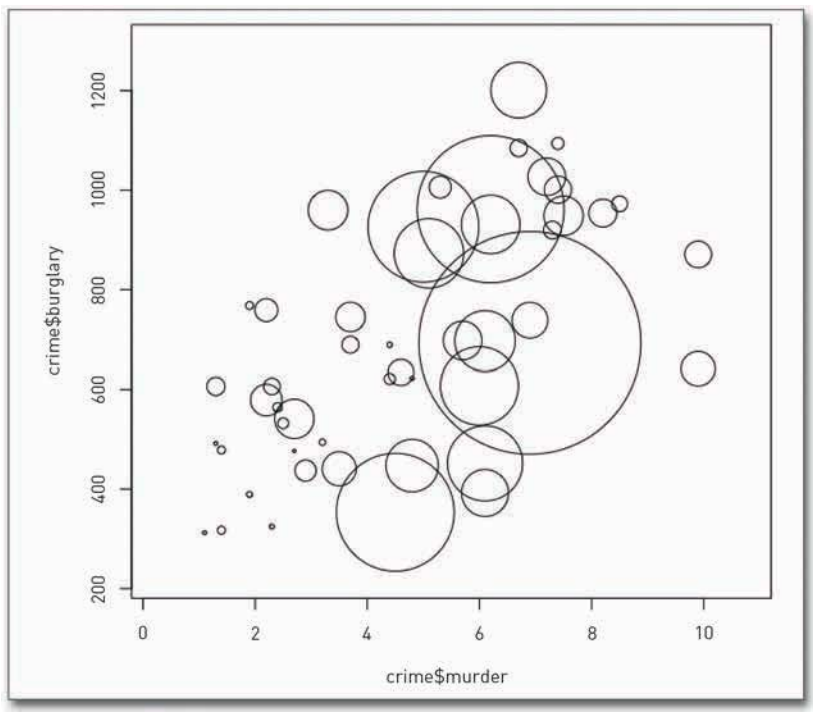


Рис. 6.16. Пузырьковая диаграмма по умолчанию

Все получилось правильно? Нет? Это был тест. Предыдущая строчка определяет размер кругов таким образом, что их радиус прямо пропорционален численности населения. А вам необходимо задать размер пузырьков так, чтобы не радиус, а их площадь была пропорциональна численности населения. Если вы станете измерять круги по радиусу, то все соотношения и пропорции полетят в тартарары. Неужели и вправду численность населения Калифорнии, представленная этим огромным кругом в центре диаграммы, настолько превосходит численность населения всех остальных штатов?

Чтобы определить размер радиусов корректно, вспомните уравнение для вычисления площади круга:

$$\text{Площадь круга} = \pi r^2.$$

Площадь каждого пузырька выражает численность населения. А вам нужно выяснить, как определить размер радиуса. Так вот: выделите радиус, и вы поймете, что он должен быть пропорционален квадратному корню из площади круга:

$$r = \sqrt{(\text{Площадь круга} / \pi)}.$$

На самом деле вы можете вообще избавиться от π, так как это константа, но пока оставьте его для пущей ясности. А теперь, чтобы определить радиусы кругов, используйте не `crime$population`, а извлеките квадратный корень и затем вставьте его в `symbols()`.

```
radius <- sqrt( crime$population/ pi )
symbols(crime$murder, crime$burglary, circles=radius)
```

Первая строчка кода просто создает новый вектор значений квадратного корня (`sqrt`) и сохраняет их в *radius*. На рис. 6.17 представлена диаграмма, радиусы пузырьков которой определены правильно. Но она выглядит запутанно, потому что пузырьки штатов, население которых меньше, чем у Калифорнии, стали крупнее.

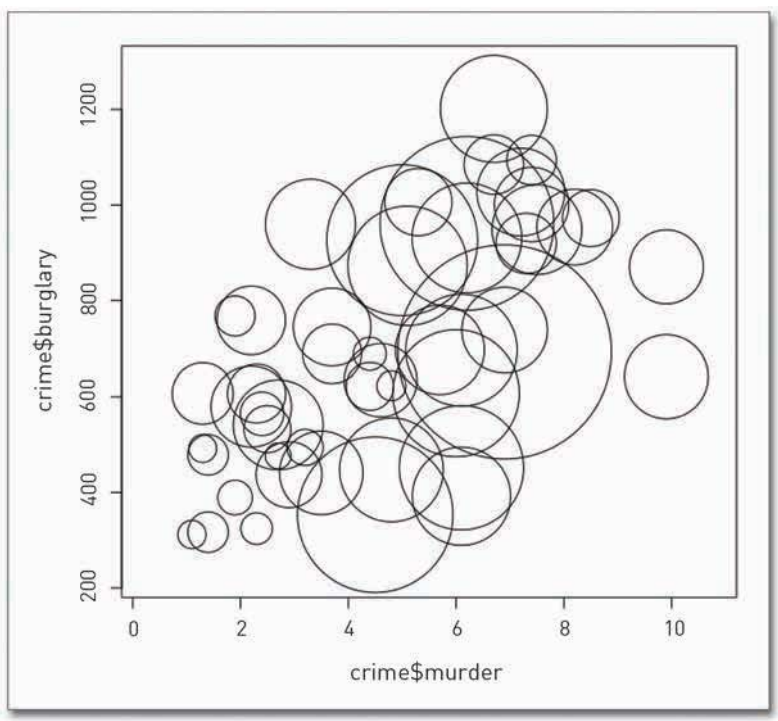


Рис. 6.17. Пузырьковая диаграмма по умолчанию с правильно определенным размером кругов

Чтобы увидеть ясную картину и понять, что происходит, вам необходимо уменьшить размер всех кругов. Аргумент *inches* в `symbols()` задает размер самого большого круга в дюймах. По умолчанию он устанавливается на уровне 1 дюйм (2,54 см), а потому на рис. 6.17 круг, обозначающий Калифорнию, получил размер в 1 дюйм, а все остальные были подстроены под него. Но вы можете сделать максимальный размер меньше — скажем, 0,35 дюйма (0,89 см), и при этом сохранить правильные пропорции. Вы можете также переопределить цвет, используя *fg* и *bg*, чтобы задать цвета обводки и заливки соответственно. А еще вы можете добавить к осям свои собственные подписи. Результат представлен на рис. 6.18.

```
symbols(crime$murder, crime$burglary, circles=radius, inches=0.35,
   fg="white", bg="red", xlab="Murder Rate", ylab="Burglary Rate")
```

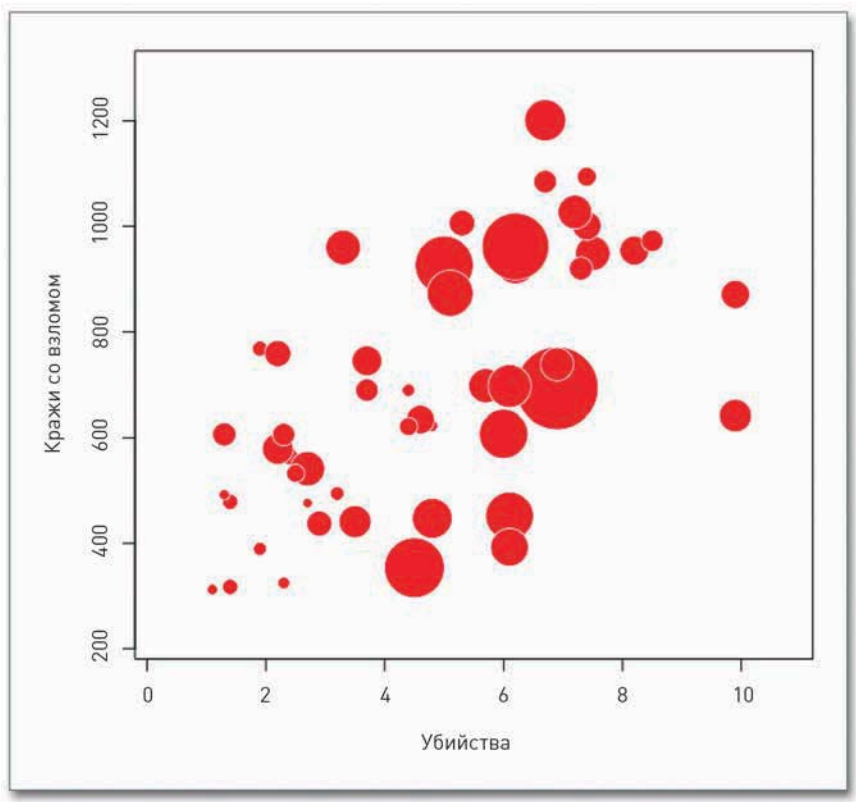


Рис. 6.18. Пузырьковая диаграмма с уменьшенным размером пузырьков

Это уже на что-то похоже.

Также с `symbols()` вы можете создать диаграмму не с кругами, а с другими фигурами: с квадратами, прямоугольниками, термометрами, ящичками и звездочками. Только аргументы

будут использоваться в таком случае другие, не те, что применяются при создании пузырьков. Например, для определения размеров квадрата используется длина стороны, но так же, как и с пузырьками, вам необходимо задавать размер сторон по квадратному корню из площади. На рис. 6.19 показано, как будут выглядеть квадраты, если используется следующая строчка кода:

```
symbols(crime$murder, crime$burglary,
   squares=sqrt(crime$population), inches=0.5)
```

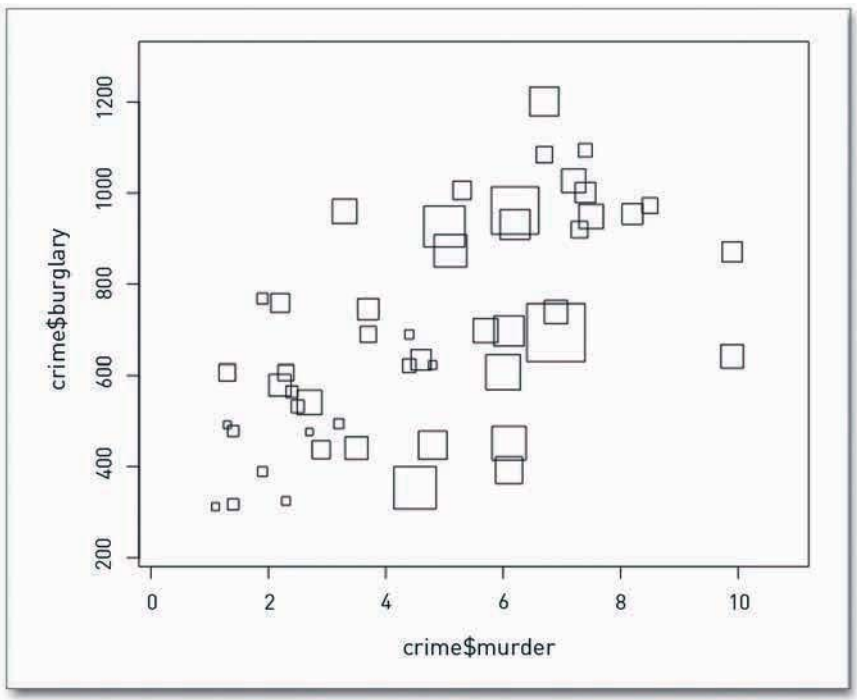


Рис. 6.19. Использование квадратов вместо кругов

Но все-таки давайте пока продолжим с кругами. Диаграмма на рис. 6.18 дает некое представление о распределении, но вы не знаете, какой круг представляет тот или иной штат. А потому следует добавить подписи с помощью `text()`, где аргументами будут выступать *x*-координаты, *y*-координаты и собственно текст. Все это у вас есть. Как и с пузырьками, по оси X пойдут убийства, а по Y — кражи со взломом. В качестве подписей будут выступать названия штатов, которые приведены в первой колонке таблицы с данными.

```
text(crime$murder, crime$burglary, crime$state, cex=0.5)
```

Аргумент cex контролирует размер текста, по умолчанию он равен 1. Значения больше единицы делают подписи крупнее, и наоборот — меньшие значения делают шрифт мельче. Подписи можно центрировать по *x*- и *y*-координатам, как это показано на рис. 6.20.

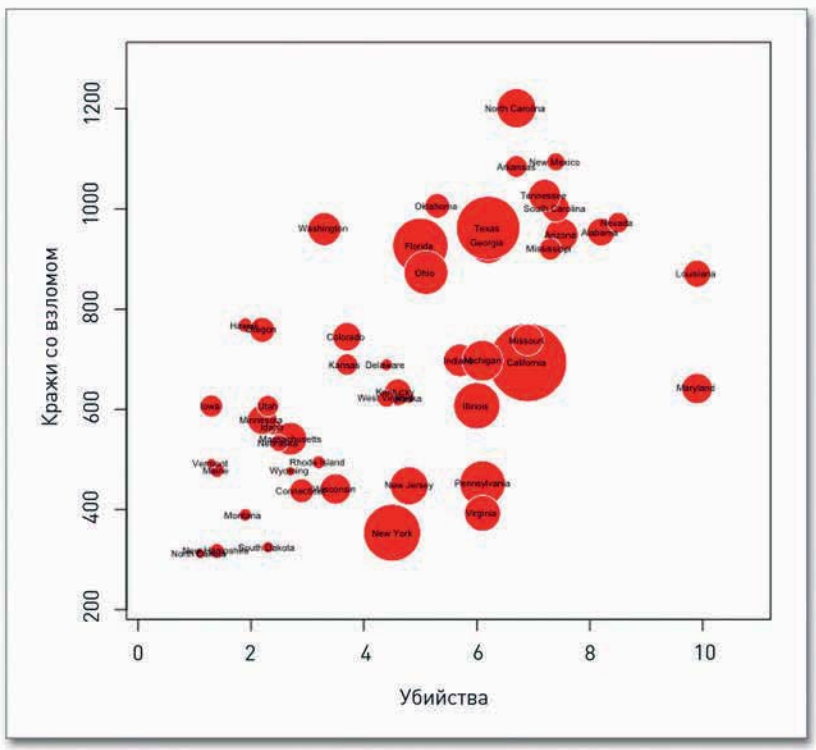


Рис. 6.20. Пузырьковая диаграмма с подписями

Теперь вам уже не нужно вносить массу правок, чтобы диаграмма обрела свой окончательный вид и стала походить на рис. 6.15. Сохраните созданную в R диаграмму как PDF, а затем откройте ее в вашей любимой программе для работы с иллюстрациями, чтобы окончательно отредактировать. Чтобы упростить и облегчить изображение, вы можете сделать линии осей тоньше, а границы диаграммы удалить полностью. Еще вы можете немного изменить местоположение подписей, особенно тех, что находятся внизу слева, чтобы названия штатов хорошо читались, а затем вынести пузырек Джорджии на передний план — в данный момент он прячется за более крупным пузырьком Техаса.

Вот это другое дело! А теперь наберите `?symbols` в R, чтобы посмотреть и другие опции для создания диаграмм. Вперед, не стесняйтесь!

Распределение

Вы наверняка и прежде слышали термины «среднее», «медиана» и «мода». Все это вы должны были проходить в школе. Средним называется сумма всех значений, поделенная на их количество. Чтобы найти медиану, вам нужно выстроить все данные по величине

от наименьшего значения к наибольшему и выделить то, которое окажется в центре. Мода — это число, которое встречается в массиве данных чаще всего. Определить все эти значения — проще простого, хотя они все равно не поведают вам всю историю. Однако они покажут вам, как распределяются различные части ваших данных. И если вы их все визуализируете, то увидите целостное распределение.

Уклон влево означает, что основной объем ваших данных сгруппирован в нижней части диапазона. Уклон вправо говорит об обратном. Плоская линия — это равномерное распределение, а кривая в форме колокола свидетельствует о группировании вокруг среднего и постепенном уменьшении в обе стороны от него.

Далее мы рассмотрим классический график, главным образом чтобы прочувствовать распределение, а затем перейдем к более практичным гистограммам и графикам плотности распределения.

Старое доброе распределение

В 1970-х годах, когда компьютеры не были так широко распространены, графики и диаграммы чертились в основном от руки. Часть советов знаменитого статистика Джона Тьюки, которые он давал в своей книге «Разведочный анализ данных»*, касались использования ручек и карандашей для достижения плотности линий и оттенков. А еще можно было использовать мелкий узор в качестве заливки для выделения различных переменных.

Так возникла и диаграмма типа «стебель с листьями» (или «опора и консоль»). Все, что вам нужно сделать, — это написать числа в определенном порядке, и в результате вы получите общее представление о распределении. Данный метод пользовался особой популярностью в 1980-х годах (когда применение статистических графиков для анализа данных только набирало обороты), потому что включить такой график в документ было легко, даже если текст набирался на печатной машинке.

И хотя сегодня существуют более простые и быстрые способы визуализации распределений, все же полезно ознакомиться и с этим, так как некоторые принципы создания диаграмм типа «стебель с листьями» вы сможете применять и при создании гистограмм.

СОЗДАЙТЕ ДИАГРАММУ «СТЕБЕЛЬ С ЛИСТЬЯМИ»

Если вы хотите испытать себя, то можете начертить диаграмму типа «стебель с листьями», используя ручку и бумагу, но в R это делается гораздо быстрее. На рис. 6.21 представлена подобная диаграмма, на которой показан уровень рождаемости в различных странах мира в 2008 году (по оценкам Всемирного банка).

Как вы сами видите, все элементарно. «Столбовые» значения идут слева, а «ответвления» — справа. В данном случае десятичная запятая находится у разделителя (|). Следовательно,

* *Тьюки Дж.* Анализ результатов наблюдений. Разведочный анализ. — М.: Мир, 1981.

интервал с наибольшим количеством стран в нем — это тот, который охватывает диапазон от 10 до 12 живорожденных на 1000 человек. Еще есть одна страна (Нигер) с уровнем рождаемости между 52 и 54.

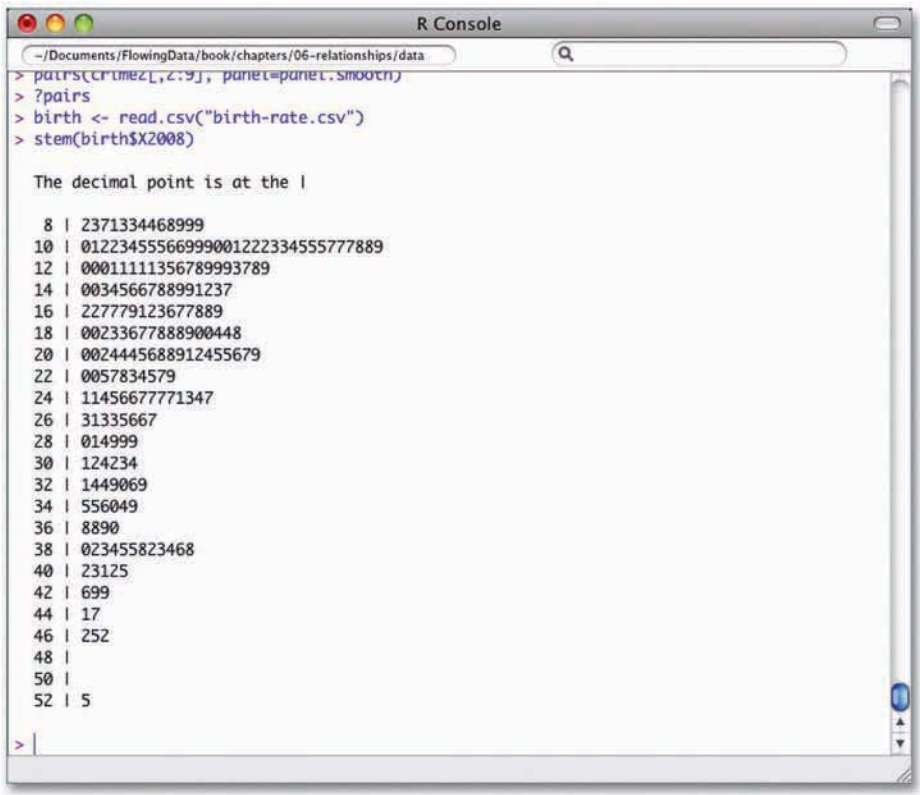


Рис. 6.21. Диаграмма типа «стебель с листьями», демонстрирующая уровень рождаемости в мире

А вот как можно создать такую диаграмму от руки. Напишите сверху вниз числа от 8 до 52 с интервалом в 2 единицы. Справа от полученной колонки чисел проведите черту. Затем, спускаясь вниз строчка за строчкой, добавляйте соответствующие числа. Если у страны уровень рождаемости составляет 8,2, добавьте цифру 2 справа от 8. Если у страны уровень рождаемости 9,9, тогда ее тоже нужно будет занести в строчку справа от 8, только туда вы теперь впишете цифру 9.

Занятие довольно утомительное, особенно если вам нужно ввести большой массив данных. Поэтому давайте посмотрим, как можно сделать такую диаграмму в R. После того как вы загрузите данные, просто используйте функцию `stem()`.

```
birth <- read.csv("http://datasets.flowingdata.com/birth-rate.csv")
stem(birth$X2008)
```

Вот и все. А если вы захотите оформить диаграмму покрасивее (как показано на рис. 6.22), то можете в R скопировать текст, а затем вставить его в другую программу. Однако данный метод считается уже устаревшим, так что лучше, наверное, будет воспользоваться гистограммой. По сути, гистограмма является более наглядной версией диаграммы «стебель с листьями».

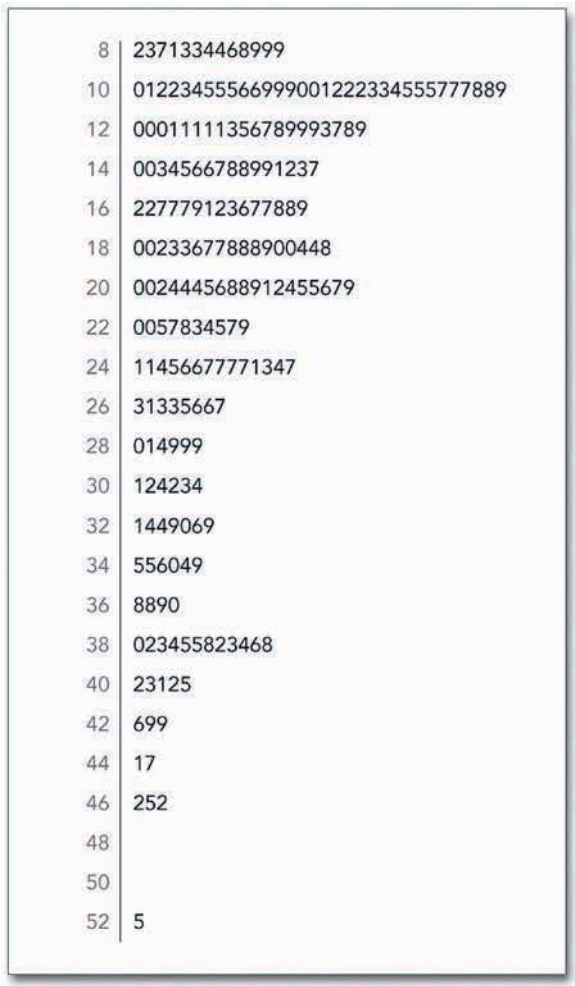


Рис. 6.22. Переработанный вариант диаграммы «стебель с листьями»

Столбцы распределения

Глядя на диаграмму «стебель с листьями» на рис. 6.22, вы сразу видите, какова частота в различных диапазонах. Чем больше стран демонстрируют уровень рождаемости в том или ином диапазоне, тем больше чисел вписывается в конкретную строчку и тем длиннее получается

сама эта строчка. А теперь переверните диаграмму набок так, чтобы строчки превратились в колонки. Чем выше колонка — тем больше стран в соответствующем диапазоне. Если превратить колонку чисел в простой прямоугольник, вы получите гистограмму, похожую на ту, что показана на рис. 6.23.

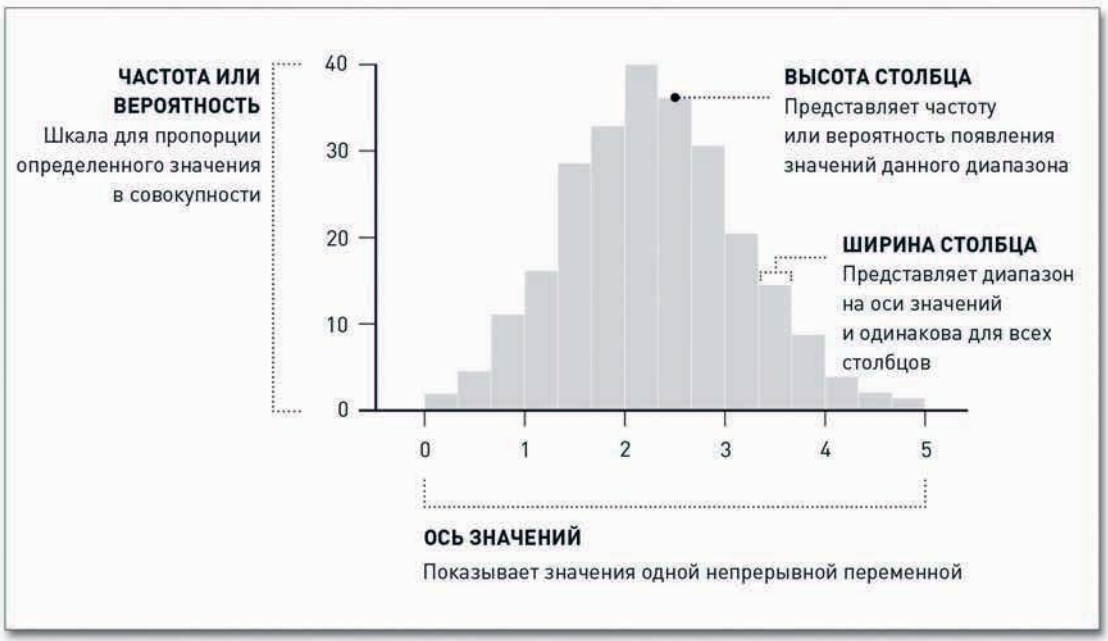


Рис. 6.23. Структура гистограммы

Высота столбцов обозначает частоту, а ширина не обозначает ничего. Горизонтальная и вертикальная оси в гистограмме непрерывны — в отличие от столбцовой диаграммы, в которой горизонтальная ось дискретна (ведь в случае со столбцовой диаграммой вы обычно работаете с категориями, и между столбцами есть расстояние).

Нередко люди, которым приходится иметь дело с диаграммами и графиками лишь изредка, неверно принимают горизонтальную ось за время. Конечно, на ней может быть представлено и время тоже, но это не обязательно. Однако данный факт необходимо учитывать, думая о своей аудитории. Если ваша диаграмма предназначена для широкого круга людей, вам необходимо будет объяснить им, как именно читать диаграмму и на что следует обратить внимание. А еще не забывайте, что многие совершенно не разбираются в распределениях. Однако, создав понятную диаграмму, вы сможете их кое-чему научить.

СОЗДАЙТЕ ГИСТОГРАММУ

Создать гистограмму в R так же легко, как и диаграмму «стебель с листьями». Сейчас вы, используя функцию `hist()`, создадите еще одну диаграмму распределения стран по уровню

рождаемости — вроде той, что показана на рис. 6.24. Вы заметили, как сильно форма столбцов напоминает форму диаграммы «стебель с листьями» на рис. 6.22?

Рис. 6.24. Распределение стран по уровню рождаемости

Полагая, что данные из предыдущего примера у вас уже загружены, предлагаю вам к тем же самым показателям за 2008 год теперь применить функцию `hist()`.

```
hist(birth$X2008)
```

Гистограмма, которую вы получите по умолчанию, показана на рис. 6.25.

В гистограмме по умолчанию десять столбцов, или диапазонов, но вы можете изменить это число на свой вкус с помощью аргумента *breaks*. Например, вы можете задать разбивку на меньшее количество столбцов, но сделать их более широкими, как показано на рис. 6.26. Здесь столбцов всего пять.

```
hist(birth$X2008, breaks=5)
```

Вы также можете пойти другим путем и сделать гистограмму с более узкими столбцами в количестве, скажем, 20 штук (рис. 6.27).

```
hist(birth$X2008, breaks=20)
```

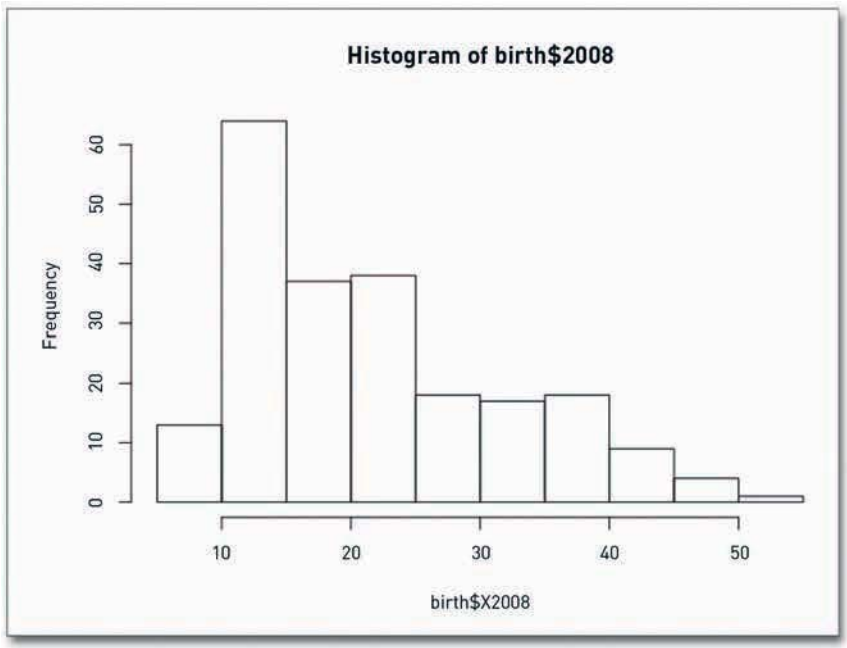


Рис. 6.25. Гистограмма по умолчанию

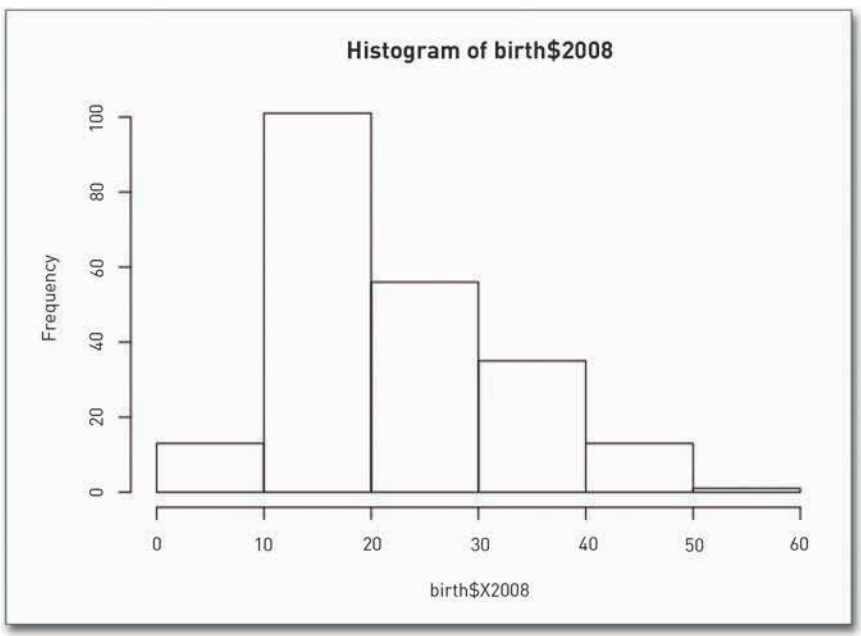


Рис. 6.26. Гистограмма с 5 столбцами

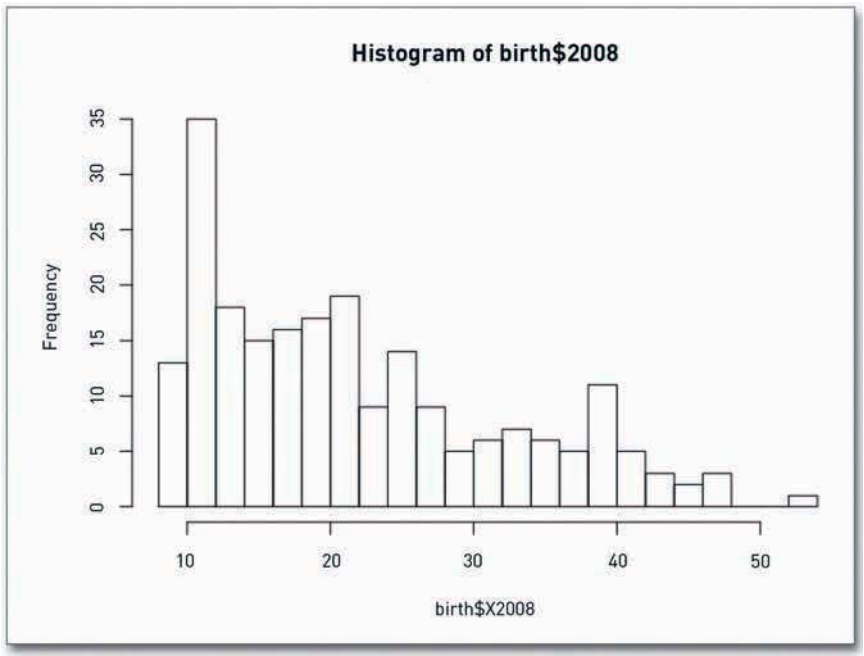


Рис. 6.27. Гистограмма с 20 столбцами

ПОДСКАЗКА

Количество делений в гистограмме по умолчанию не всегда является самым подходящим. Поиграйте с опциями и решите сами, что представляется вам более разумным для конкретного набора данных.

Какое количество столбцов выбрать, будет зависеть от самих данных, которые вы собираетесь визуализировать. Если бóльшая их часть сгруппирована где-то в начале, вы можете захотеть сделать деление более подробным, чтобы вместо создания одного высокого столбца получить возможность рассмотреть вариации. А если у вас не так много данных или числа распределены относительно равномерно, тогда широкие столбцы могут оказаться более подходящими. Хорошая новость состоит в том, что все это очень легко меняется и вы можете экспериментировать в свое удовольствие.

В случае с информацией об уровне рождаемости количество делений по умолчанию — это как раз то, что надо. Видно, что есть страны, где уровень рождаемости ниже 10, но чаще всего коэффициент живорожденных детей на 1000 жителей варьирует от 10 до 25. Есть и такие страны, где уровень рождаемости выше 25, но их сравнительно мало.

ПОДСКАЗКА

В R вы легко можете найти среднее, медиану, моду и квартиль с помощью функции `summary()`.

Теперь вы можете сохранить результат как PDF и внести дальнейшие правки уже в программе Illustrator. Бóльшая часть таковых будет похожа на то, что вы уже делали, работая со столбцовыми диаграммами в главе 4, но есть и специфические детали, которые необходимо учесть, чтобы сделать гистограмму более понятной и объяснить своим читателям, чему она посвящена.

В окончательном варианте (рис. 6.24) вы можете выделить некоторые из самых важных характеристик распределения, скажем, медиану, минимум и максимум. Вводный текст — это еще одна возможность помочь читателям разобраться, что к чему. А еще вы можете добавить немного цвета, чтобы диаграмма не смотрелась как проволочный каркас.

Непрерывная плотность

Хотя ось значений непрерывная, распределение все равно разбивается на дискретное число столбцов. Каждый столбец представляет собой некую совокупность объектов, в случае с предыдущим примером — совокупность стран. Но что происходит внутри каждого такого деления, каждой «корзины»? Ведь с диаграммой типа «стебель с листьями» вы могли видеть все числа, и вам все-таки было сложно оценить величину различий. Ситуация, довольно схожая с использованием метода локально взвешенного сглаживания диаграмм рассеяния (ЛВСДР), разработанного Кливлендом и Дэвлин, который вы применяли в главе 4 для более наглядного представления тенденций. Для визуализации малых по масштабу колебаний внутри распределения вы можете использовать график плотности распределения.

На рис. 6.28 показано использование кривой вместо столбцов. Общая площадь под кривой равняется единице, а вертикальная ось представляет вероятность или пропорцию данного значения в этой самой совокупности.

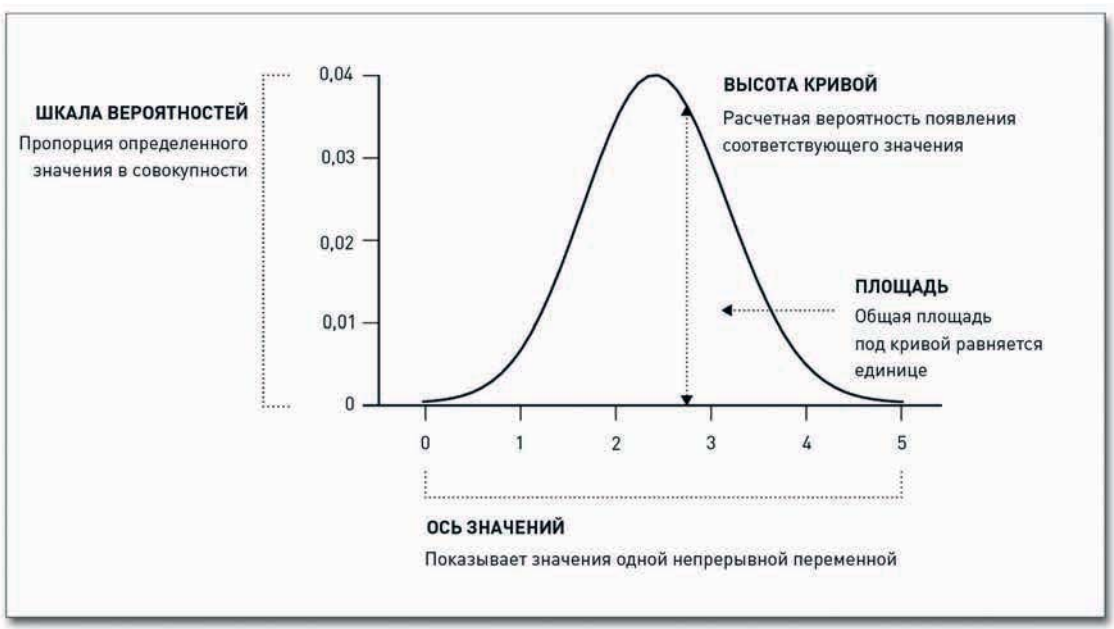


Рис. 6.28. Структура графика плотности

СОЗДАЙТЕ ГРАФИК ПЛОТНОСТИ

Вернитесь к данным об уровне рождаемости. На этот раз, чтобы создать график плотности распределения, вам потребуется проделать одну дополнительную операцию. Вам нужно будет воспользоваться функцией `density()`, чтобы определить точки для кривой. Однако пробелы

ПРИМЕЧАНИЕ

Позиции с отсутствующими данными удалены ради упрощения примера. Но когда вы будете визуализировать и анализировать другие наборы данных, вы должны обратить особое внимание на подобные пропуски и подумать, почему данные отсутствуют и что следует сделать: вставить туда нули или удалить строчки вовсе.

в данных в этом случае недопустимы, а у нас в массиве данных за 2008 год есть 15 позиций, которые не содержат данных.

В R подобные строчки с отсутствующими данными помечаются буквами NA. К счастью, такие строчки легко отфильтровываются.

```
birth2008 <- birth$X2008[!is.na(birth$X2008)]
```

Приведенная выше строка кода извлекает из таблицы колонку с данными об уровне рождаемости за 2008 год, а затем, как вы и велели, сохраняет строчки, в которых нет отсутствующих данных, в *birth2008*. Технически это выглядит так: `is.na()` проверяет каждую позицию данных в векторе *birth$X2008* и возвращает вектор одинаковой длины с истинными (true) и ложными (false) значениями, известными как *булевы*, или логические, значения. Когда вы передаете вектор булевых значений в индекс вектора, возвращаются только те позиции, которые соответствуют истинным значениям. Не смущайтесь, если все сказанное звучит для вас несколько запутанно. Вам не обязательно разбираться в деталях технологии, чтобы она работала. Однако если вы планируете писать свои собственные функции, то я советую вам выучить этот язык. Тогда вам будет проще читать документацию, хотя в процессе работы вы и так наберетесь знаний.

Теперь у вас уже есть очищенный массив данных об уровне рождаемости, который вы сохранили в *birth2008*, так что сейчас вы можете передать его в функцию `density()` для высчитывания кривой и сохранения результатов в *d2008*.

```
d2008 <- density(birth2008)
```

Таким образом вы получите координаты для кривой. Что здесь особенно приятно, так это то, что вы можете сохранить координаты в текстовый файл на тот случай, если захотите использовать их для начертания кривой в другой программе. Наберите в консоли R `d2008`, чтобы посмотреть, что и как сохранилось в переменной. Вот что вы получите:

```
Call:
   density.default(x = birth2008)

 Data: birth2008 (219 obs.);      Bandwidth 'bw' = 3.168

             x                          y
 Min.        :-1.299         Min.      :6.479e-06
 1st Qu.     :14.786         1st Qu.   :1.433e-03
 Median      :30.870         Median    :1.466e-02
 Mean        :30.870         Mean      :1.553e-02
 3rd Qu.     :46.954         3rd Qu.   :2.646e-02
 Max.        :63.039         Max.      :4.408e-02
```

Вас интересуют прежде всего *x* и *y*. Здесь показано распределение координат, но если вы хотите получить их все, тогда введите вот что:

```
d2008$x
d2008$y
```

А затем сохраните их в текстовом файле, используя `write.table()`. В качестве аргументов вам нужны данные, которые вы хотите сохранить, имя файла, используемый разделитель (например, запятая или знак табуляции) и кое-что еще. Чтобы сохранить данные в качестве простого текстового файла с разделителями в виде знаков табуляции, сделайте следующее:

```
d2008frame <- data.frame(d2008$x, d2008$y)
write.table(d2008frame, "birthdensity.txt", sep="\t")
```

В вашем рабочем каталоге уже должен был появиться файл `birthdensity.txt`. Если вы не хотите, чтобы строчки оказались пронумерованы, или предпочитаете в качестве разделителя видеть запятые, это можно легко устроить:

```
write.table(d2008frame, "birthdensity.txt", sep=",", row.names=FALSE)
```

Сейчас вы можете загрузить данные в Excel, Tableau, Protovis или в другую программу, принимающую текст с разделителями, — то есть практически в любую.

А теперь давайте вернемся к нашему графику плотности распределения. Вспомните, что у вас уже есть необходимые координаты для вычерчивания графика. Вам осталось только перевести их в визуальную форму, используя, естественно, функцию `plot()`. Результат представлен на рис. 6.29.

ПРИМЕЧАНИЕ

Функция `write.table()` сохраняет новые файлы в вашем текущем рабочем каталоге. Изменить рабочий каталог можно через главное меню или с помощью функции `setwd()`.

ПОДСКАЗКА

Если вы предпочитаете выполнить работу по вычерчиванию графика в другой программе, а не в R, но при этом хотите воспользоваться вычислительным функционалом R, тогда сохраните часть результатов или их все, используя `write.table()`.

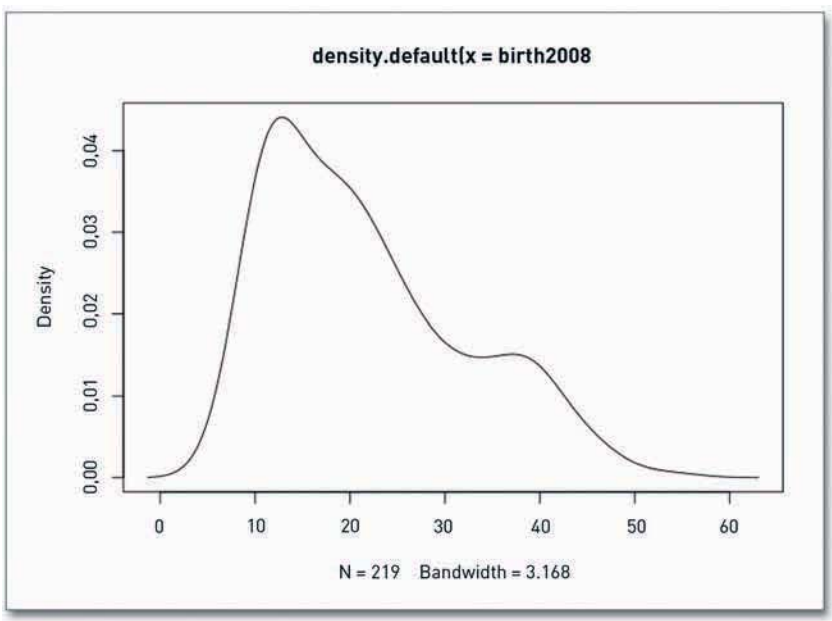


Рис. 6.29. График плотности по умолчанию

```
plot(d2008)
```

А еще, если захотите, вы можете создать график плотности с заливкой, используя наряду с функцией `plot()` также и `polygon()`, как показано на рис. 6.30. Первую из функций вы используете для определения осей, но при этом `type` будет установлен на `"n"`, чтобы процедура вычерчивания не запускалась, а уже затем для вычерчивания формы вы используете вторую функцию. Задайте в качестве цвета заливки темно-красный, а границы пусть будут светло-серыми.

```
plot(d2008, type="n")
polygon(d2008, col="#821122", border="#cccccc")
```

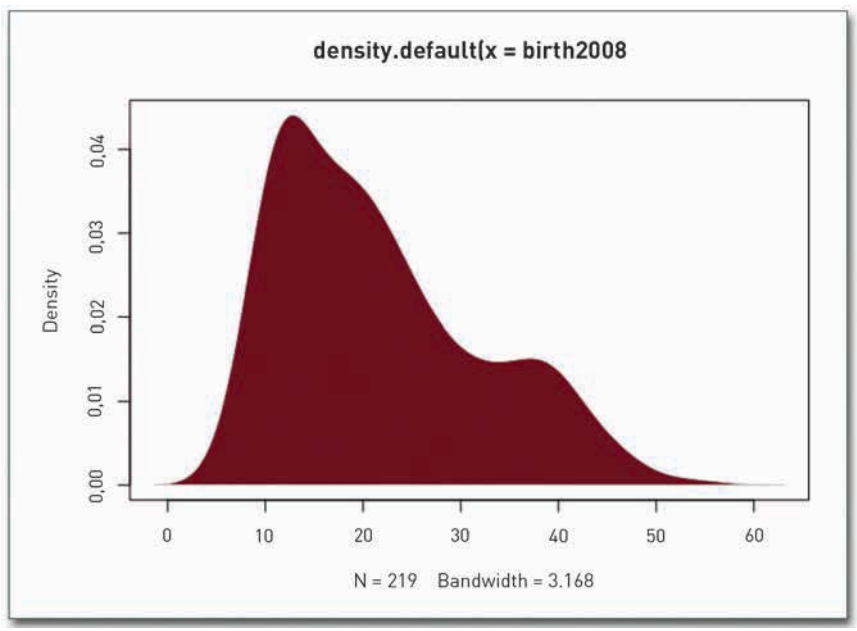


Рис. 6.30. График плотности с заливкой

Но нет нужды на этом останавливаться. Вы можете начертить вместе и гистограмму, и график плотности распределения и таким образом получить одновременно и точную частоту, представляемую столбцами, и расчетные пропорции, которые дает кривая (рис. 6.31). Используйте функцию `histogram()` из пакета `lattice` и функцию `lines()`. Первая из них создаст новый график, а вторая добавит к нему линии.

```
library(lattice)
histogram(birth$X2008, breaks=10)
lines(d2008)
```

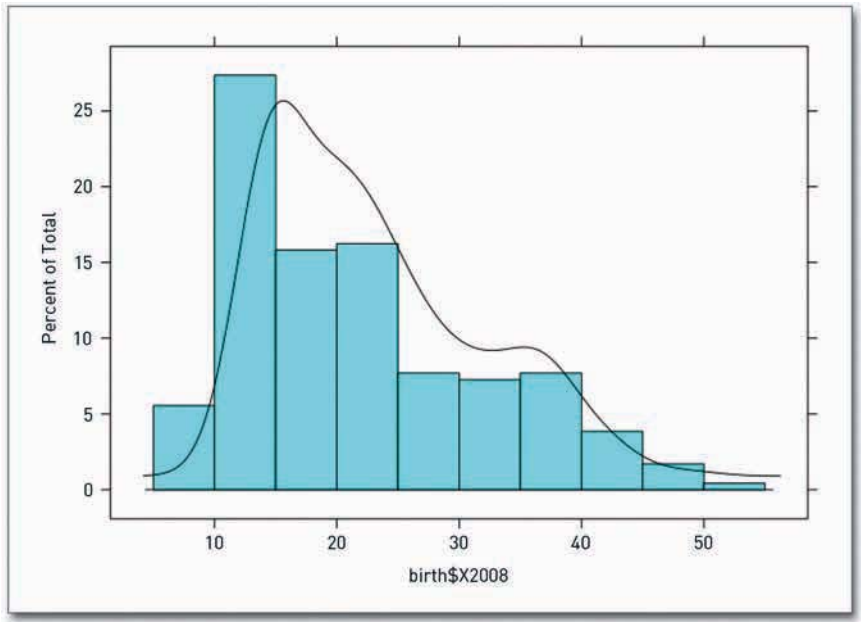


Рис. 6.31. Совмещение гистограммы и графика плотности

Так что вы можете много чего сделать — вариантов уйма, и при всем при том математика и геометрия остаются точно такими же, как и при создании диаграммы «стебель с листьями». Вы просто считаете, составляете совокупности и группируете. А какому именно варианту визуализации лучше отдать предпочтение, будет зависеть от самих данных, которыми вы располагаете. На рис. 6.32 представлен уже более-менее завершенный вариант графики. Я ослабил акцентированность линий осей, подкорректировал подписи и снабдил медиану указателем. Вертикальная ось, которая представляет плотность, в данном случае не особо нужна, но я оставил ее для придания графику завершенности.

Рис. 6.32. График плотности распределения уровня рождаемости в мире в 2008 году

Сравнение

Очень часто бывает полезнее сравнивать несколько распределений, а не просто средние, медианы и моды. Ведь подобная сводная статистика описывает картину в целом, но крупными «мазками», а потому способна поведать лишь часть истории.

Например, я мог бы сказать вам, что средний уровень рождаемости в мире в 2008 году составлял 19,98 живорожденных детей на 1000 человек населения и что в 1960 году данный показатель был на уровне 32,87, так что можно говорить о его 39-процентном снижении с 1960 до 2008 года. Однако это свидетельствует лишь о том, что происходит в центре распределения. А может, в 1960-х годах было всего несколько стран с высоким уровнем рождаемости, которые, собственно, и подняли среднее значение? Как изменилась ситуация в разных странах за прошедшие десятилетия? Различия между странами усилились или ослабли?

Есть множество способов для проведения сравнений. Вы можете уйти в чистую аналитику и вовсе не прибегать к визуализации. (В магистратуре я провел целый год за изучением статистических методов — и то мне удалось обозреть лишь верхушку айсберга.) А можете поступить наоборот и использовать визуализацию. Ваш результат не будет столь же точен, как ответ, полученный в процессе тщательного статистического анализа, но он может быть вполне пригодным для принятия информированного решения о том, чем вы сейчас занимаетесь. Очевидно, что вы пойдете визуализационным путем.

Множественные распределения

До сих пор вы занимались лишь единичными распределениями, а именно — распределением уровня рождаемости в мире в 2008 году. Но если вы посмотрите оригинальный файл с данными или таблицу в R, вы увидите, что у вас есть сведения об уровне рождаемости по годам аж с 1960-го. А если не посмотрите, тогда... ну что ж, они все равно у вас есть, все эти данные. Я говорил выше, уровень рождаемости в мире существенно снизился, но как изменилось собственно распределение?

Вот и идите, создайте по гистограмме за каждый год, а затем уложите их в замечательно организованную матрицу. Идея здесь примерно такая же, как и в случае с матрицей диаграмм рассеяния, которую вы создали в начале этой главы.

СОЗДАЙТЕ МАТРИЦУ ГИСТОГРАММ

Пакет lattice в R позволит вам с легкостью создать целую охапку гистограмм при помощи всего нескольких строк кода, но здесь есть одна небольшая хитрость. Вам нужно подать данные в таком виде, в котором их хочет видеть сама функция. Ниже представлен фрагмент первоначального варианта загруженного вами текстового файла.

```
Country,1960,1961,1962,1963...
Aruba,36.4,35.179,33.863,32.459...
Afghanistan,52.201,52.206,52.208,52.204...
...
```

Каждая строчка — это отдельная страна. В первой колонке идет ее название (country), затем следуют колонки с данными по каждому году, то есть вы имеете 30 колонок и 234 ряда данных плюс шапка. А вам необходимо, чтобы данные были представлены в двух колонках: в одной — год (year), а во второй — уровень рождаемости (rate). В этом случае названия стран вам, по сути, не нужны, так что первые несколько строк данных будут выглядеть так:

```
year,rate
1960,36.4
1961,35.179
1962,33.863
1963,32.459
1964,30.994
1965,29.513
...
```

Если вы сравните фрагмент, который вам нужен, с текущим, вы наверняка заметите, что значения во втором фрагменте соответствуют значениям по государству Аруба. Итак, у вас есть по строчке на каждое значение уровня рождаемости, и это число сопровождается годом. А всего рядов данных — 9870, не считая шапки.

Как можно получить данные в нужном формате? Помните, что вы делали с Python в главе 2? Вы загрузили CSV-файл в Python, а затем проитерировали все строчки подряд и распечатали полученные значения в желаемом формате. Здесь вы можете повторить то же самое. Создайте новый файл под названием `transform-birth-rate.py` в вашем любимом текстовом редакторе. Убедитесь, что он находится в том же каталоге, что и `birth-rate.csv`. А затем введите следующий скрипт:

```
import csv

reader = csv.reader(open('birth-rate.csv', 'r'), delimiter=",")

rows_so_far = 0
print 'year,rate'
for row in reader:
   if rows_so_far == 0:
      header = row
      rows_so_far += 1
   else:

      for i in range(len(row)):
         if i > 0 and row[i]:
            print header[i] + ',' + row[i]

      rows_so_far += 1
```

Этот скрипт наверняка покажется вам знакомым. Однако прервите его. Импортируйте пакет `csv` и следом загрузите `birth-rate.csv`. Затем выведите шапку и начните итерацию по всем строчкам

ПОДСКАЗКА

Если вы хотите, чтобы кодирование у вас целиком проходило в R, тогда можете попробовать воспользоваться пакетом reshape, разработанным Хэдли Уикхэмом (Hadley Wickham). Он поможет перевести порцию данных в нужный вам формат.

и колонкам, чтобы получить данные в требуемом формате. Скрипт запустите из своей консоли, а результат сохраните в новом CSV-файле под названием `birth-rate-yearly.csv`.

```
python transform-birth-rate.py > birth-rate-yearly.csv
```

Замечательно. А теперь для матрицы вы будете использовать `histogram()`. Вернитесь в R и с помощью `read.csv()` загрузите новый файл с данными. В случае если вы пропустили весь этот этап с форматированием (или оставили его на потом), я скажу, что новый файл с данными уже размещен в Сети и вы можете просто загрузить его с URL:

```
birth_yearly <-
   read.csv("http://datasets.flowingdata.com/birth-rate-yearly.csv")
```

Теперь передайте данные в `histogram()`, чтобы сделать матрицу 10×5, в которой уровень рождаемости (rate) будет категоризирован по годам (year). Результат представлен на рис. 6.33.

```
histogram(~ rate | year, data=birth_yearly, layout=c(10,5))
```

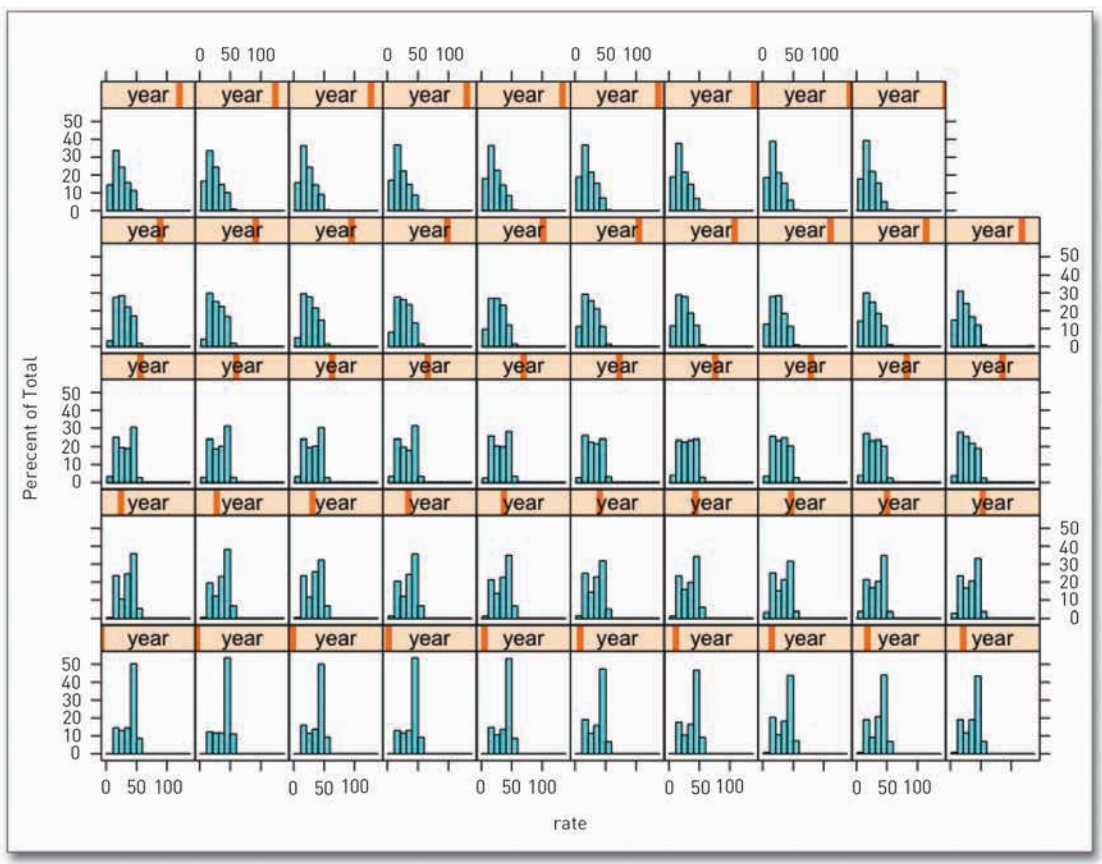


Рис. 6.33. Матрица гистограмм по умолчанию

Ну что ж, неплохо. Но вы можете улучшить результат. Обратите внимание, во-первых, на выброс в дальнем правом углу, который выталкивает все столбцы влево. Во-вторых, на то, что указателем года служит оранжевая полоска в годовых ячейках матрицы, которая перемещается слева направо. Гистограмму было бы проще читать, если б на этом месте стояли подписи с годами в виде чисел. И наконец (хотя трудно говорить наверняка, так как нет подписей с годами), кажется, что порядок выстраивания гистограмм не совсем работает. Первый год — 1960-й — находится в нижнем левом углу, а 1969-й — в нижнем правом. Следующий, 1970-й, находится над 1960-м. Иными словами, очередность выстроена снизу вверх и слева направо. Это представляется странным.

Чтобы найти выброс, снова воспользуйтесь `summary()` для *`birth_yearly`*.

```
      year               rate
 Min.      :1960   Min.      :6.90
 1st Qu.   :1973   1st Qu.   :18.06
 Median    :1986   Median    :29.61
 Mean      :1985   Mean      :29.94
 3rd Qu.   :1997   3rd Qu.   :41.91
 Max.      :2008   Max.      :132.00
```

Максимальный уровень составляет 132. Кажется, это число попало сюда случайно. Ни один другой показатель и близко к нему не стоит. Что тут происходит? Оказывается, 132 — это показатель рождаемости в республике Палау, зафиксированный за 1999 год. Скорее всего, имеет место опечатка, так как уровень рождаемости в Палау до и после 1999 года не превышает 20. Надо полагать, здесь должно было стоять число 13,2. Однако вам придется покопаться, чтобы выяснить все наверняка. Давайте на время удалим эту ошибку.

```
birth_yearly.new <- birth_yearly[birth_yearly$rate < 132,]
```

Далее перейдем к подписям с годами. Когда используемые для подписей значения сохраняются в числовом виде, решетчатая функция автоматически использует в качестве индикатора значения оранжевую полоску. А если подписи состоят из букв, функция использует строчку. Так что действуйте:

```
birth_yearly.new$year <- as.character(birth_yearly.new$year)
```

Осталось изменить очередность представления данных по годам, но сначала давайте создадим матрицу гистограмм и сохраним ее в переменной.

```
h <- histogram(~ rate | year, data=birth_yearly.new, layout=c(10,5))
```

А теперь нужно использовать функцию `update()`, чтобы изменить порядок гистограмм в матрице.

```
update(h, index.cond=list(c(41:50, 31:40, 21:30, 11:20, 1:10)))
```

Таким образом вы, по сути, перевернули порядок выстраивания рядов. Как показано на рис. 6.34, вы получили матрицу гистограмм с внятными подписями и с более разумным распределением (после того как удалили опечатку). При этом гистограммы выстроены в более логичном порядке — слева направо и сверху вниз. Вы можете читать лишь по одной ячейке в каждом ряду и, перемещая взгляд сверху вниз, ухватить, как меняется распределение по десятилетиям.

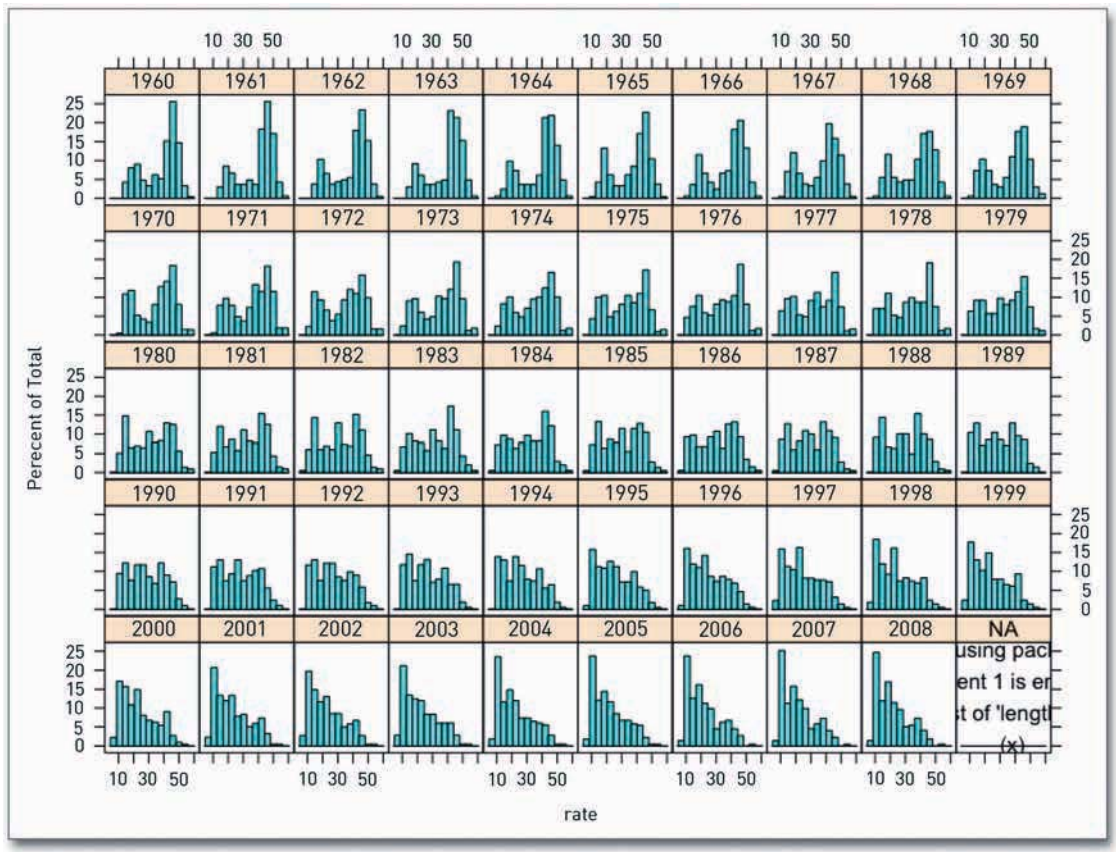


Рис. 6.34. Переработанная матрица гистограмм

Теперь уже получилось хорошо. Если вы захотите отредактировать полученную графику в Illustrator, вы можете уменьшить подписи, изменить границы и цвет заливки и подчистить еще кое-что, как показано на рис. 6.35. Таким образом вы улучшите читабельность матрицы. Но чтобы сделать ее еще более наглядной и при этом дополнить историю, вы можете также добавить подходящий вводный текст, указать источник данных и отметить, как распределение смещается влево в сторону более низкого уровня рождаемости во всем мире. Продемонстрировать все это с помощью одного отдельного графика было бы слишком сложно, так как вам пришлось бы обстоятельно знакомить читателей с контекстом, чтобы они смогли вникнуть и оценить данные в полной мере.

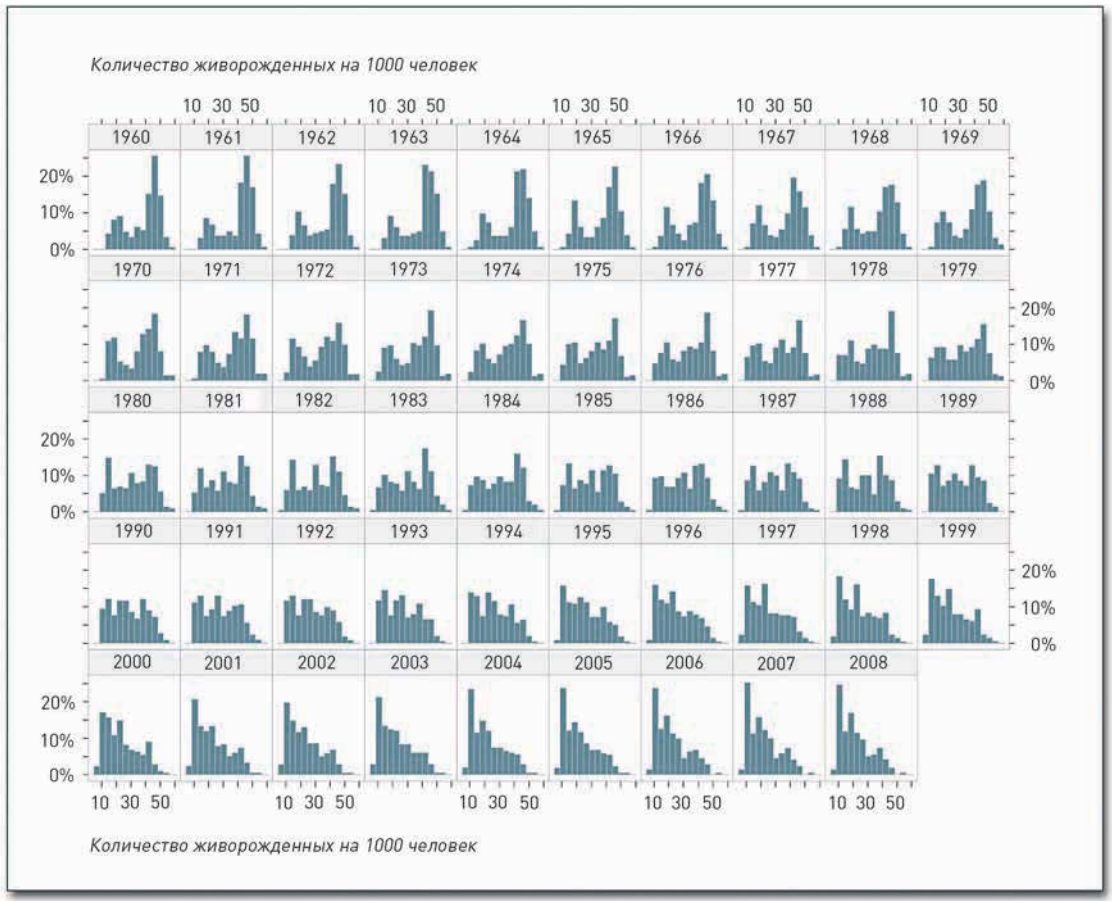


Рис. 6.35. Матрица гистограмм, отредактированная в Illustrator

Но всего этого можно добиться и другими средствами, скажем, с помощью Processing, Protovis, PHP или любого инструмента, умеющего чертить столбцовые диаграммы. Даже в самом R существует не один способ создания такого тип матриц. Например, я как-то создал для FlowingData графику, демонстрирующую распределение размеров телеэкранов в период с 2002 по 2009 годы (рис. 6.36).

Код выглядит несколько иначе, не как тот, который вы только что использовали, однако логика действий схожая. Я загрузил данные, применил кое-какие фильтры, чтобы отсеять выбросы, а затем начертил множество гистограмм. Отличие состоит в том, что для данной цели я не использовал функцию `histogram()` из пакета lattice. Вместо этого я задал структуру с помощью функции `par()`, которая в R применяется универсально для определения параметров, а затем, чтобы начертить все диаграммы, воспользовался функцией `hist()`.

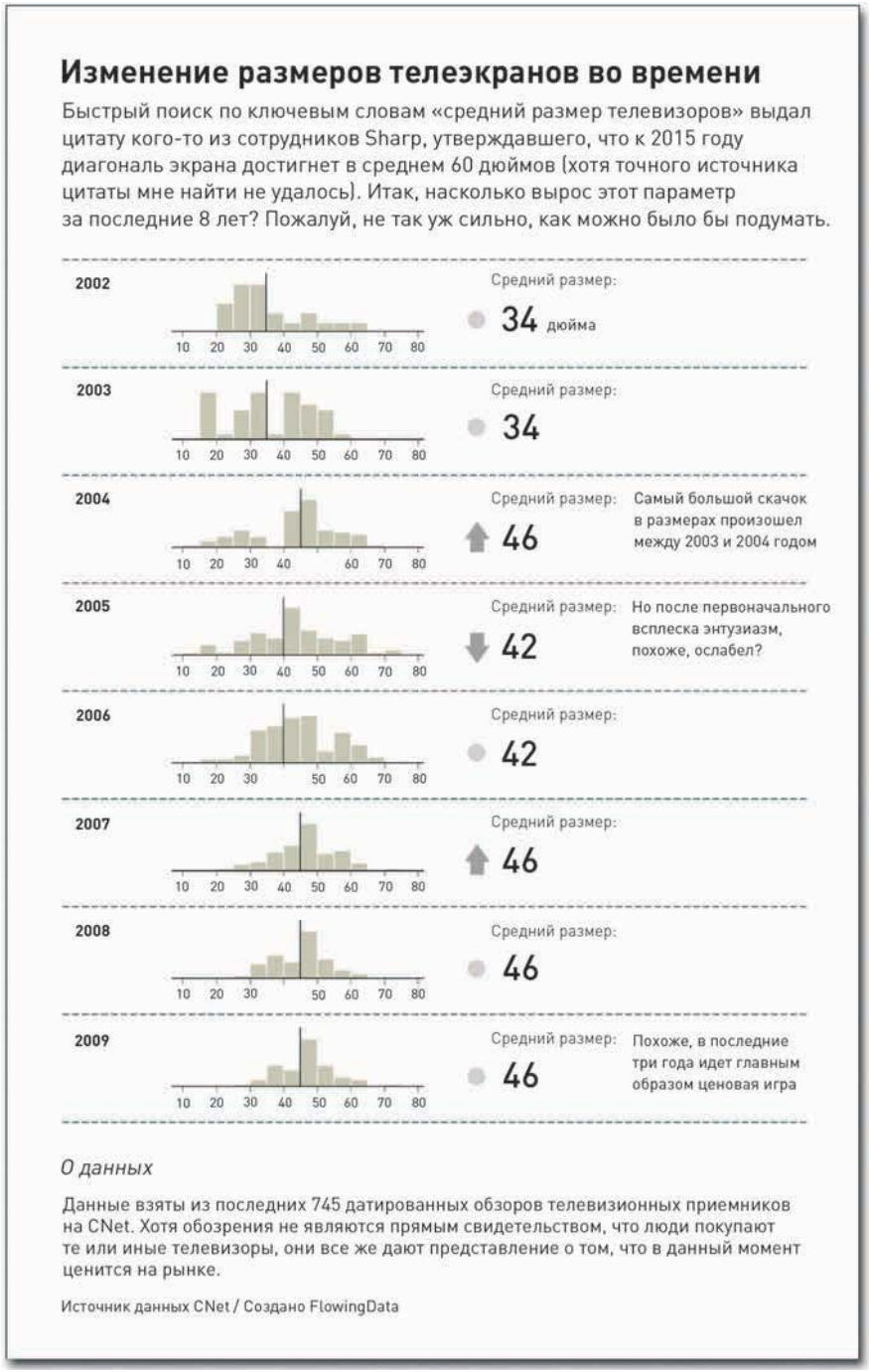


Рис. 6.36. Распределение размеров телевизионных экранов во времени

```
# Load data
tvs <- read.table('http://datasets.flowingdata.com/tv_sizes.txt',
   sep="\t", header=TRUE)

# Filter outliers
tvs <- tvs[tvs$size < 80, ]
tvs <- tvs[tvs$size > 10, ]

# Set breaks for histograms
breaks = seq(10, 80, by=5)

# Set the layout
par(mfrow=c(4,2))

# Draw histograms, one by one
hist(tvs[tvs$year == 2009,]$size, breaks=breaks)
hist(tvs[tvs$year == 2008,]$size, breaks=breaks)
hist(tvs[tvs$year == 2007,]$size, breaks=breaks)
hist(tvs[tvs$year == 2006,]$size, breaks=breaks)
hist(tvs[tvs$year == 2005,]$size, breaks=breaks)
hist(tvs[tvs$year == 2004,]$size, breaks=breaks)
hist(tvs[tvs$year == 2003,]$size, breaks=breaks)
hist(tvs[tvs$year == 2002,]$size, breaks=breaks)
```

Графический результат данного кода представлен на рис. 6.37. Здесь у нас четыре ряда и две колонки, что было задано в аргументе *mfrow* функции `par()`. В окончательном варианте я все же выстроил результаты в одну колонку, но здесь важнее всего то, что мне не пришлось вводить целую кучу данных в Illustrator или Excel и вручную создавать восемь диаграмм.

МАЛЕНЬКИЕ ПАНЕЛИ

Прием объединения в один графический объект серии небольших диаграмм часто называют «маленькими панелями» (small multiples*). Данный подход облегчает для читателя проведение сравнений как между группами и категориями, так и внутри этих групп и категорий. Если подобную панельную диаграмму хорошо организовать, вы можете вместить в нее великое множество «миниатюр».

---

* Термин small multiples был введен Эдвардом Тафти для описания множества небольших по размеру, однотипных по структуре, но многовариантных (в плане данных) диаграмм или графиков, которые легко сравнивать. Как и большинство терминов Тафти, small multiples тяжело поддается переводу. На русском встречаются различные варианты: «типовые множества», «маленькие однотипности», «размноженные миниатюры», «мультиграфики» и даже «раскадровки». В данном переводе использован термин «маленькие панели». В некоторых случаях в качестве синонима встречается также термин «панельная диаграмма» (*англ.* panel chart). *Прим. пер.*

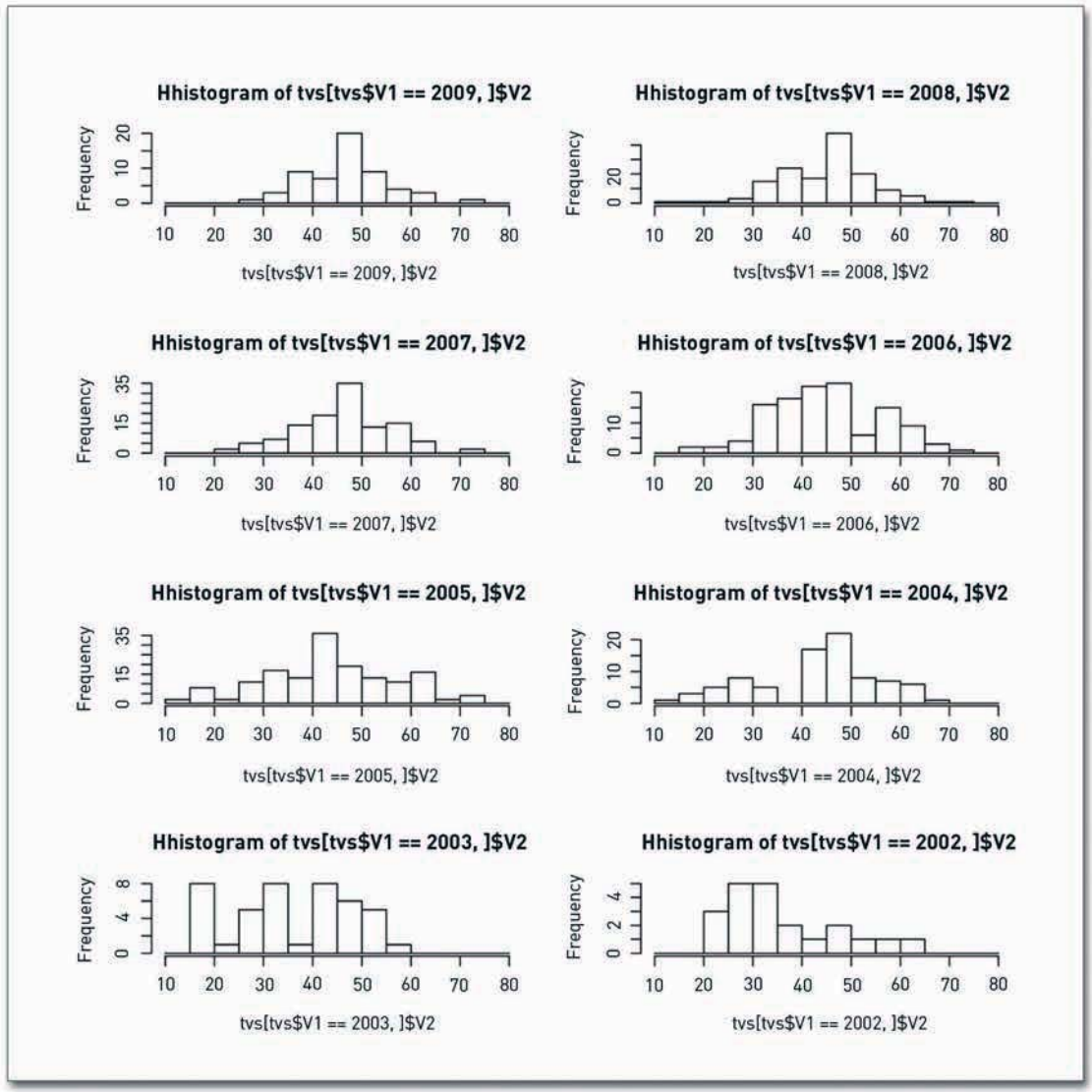


Рис. 6.37. Схема сетки для гистограмм

Приведу один пример. Как-то раз я посмотрел рейтинги кинолент на сайте Rotten Tomatoes («Гнилые помидоры»), а точнее, выборку по трилогиям. В случае если вы никогда не слышали о таком сайте, сообщу, что Rotten Tomatoes собирает рецензии на кинофильмы и помечает их как положительные или отрицательные. Когда как минимум 60 процентов критиков говорят, что фильм им понравился, его помечают как «свежий». В противном случае — как «гнилой».

Я захотел выяснить, как продолжения в трилогиях соотносятся с оригиналами* по своей «свежести». Оказалось, что сравнение не в их пользу (рис. 6.38). Медианный рейтинг финальных картин в трилогиях оказался на 37 процентных пунктов ниже, чем медианный рейтинг начальных. Иными словами, большинство первых фильмов были «свежими», а большинство заключительных — «гнилыми».

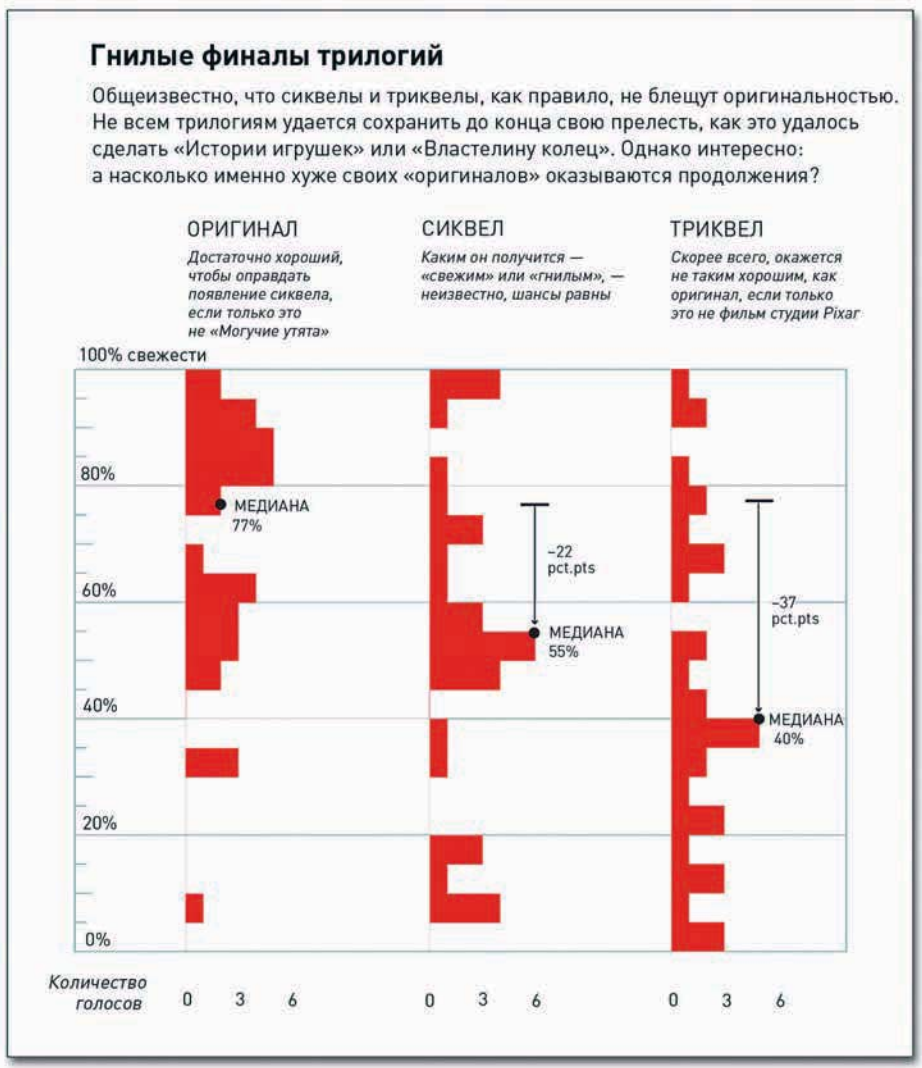


Рис. 6.38. Рейтинги трилогий от оригинала до финала

* Далее в тексте примеров использованы распространенные (применительно к фильмам в нескольких частях) термины «оригинал» (о первой ленте), «сиквел» (о ее сюжетном продолжении), «триквел» (о продолжении сиквела, иными словами, о третьем фильме) и «приквел» (о картине, описывающей более ранние в хронологическом плане события, но снятой позже «оригинала»). *Прим. пер.*

На самом деле на рис. 6.38 представлены три гистограммы, положенные набок. А первоначальный вариант этих гистограмм в R показан на рис. 6.39. Я всего лишь подредактировал их в Illustrator.

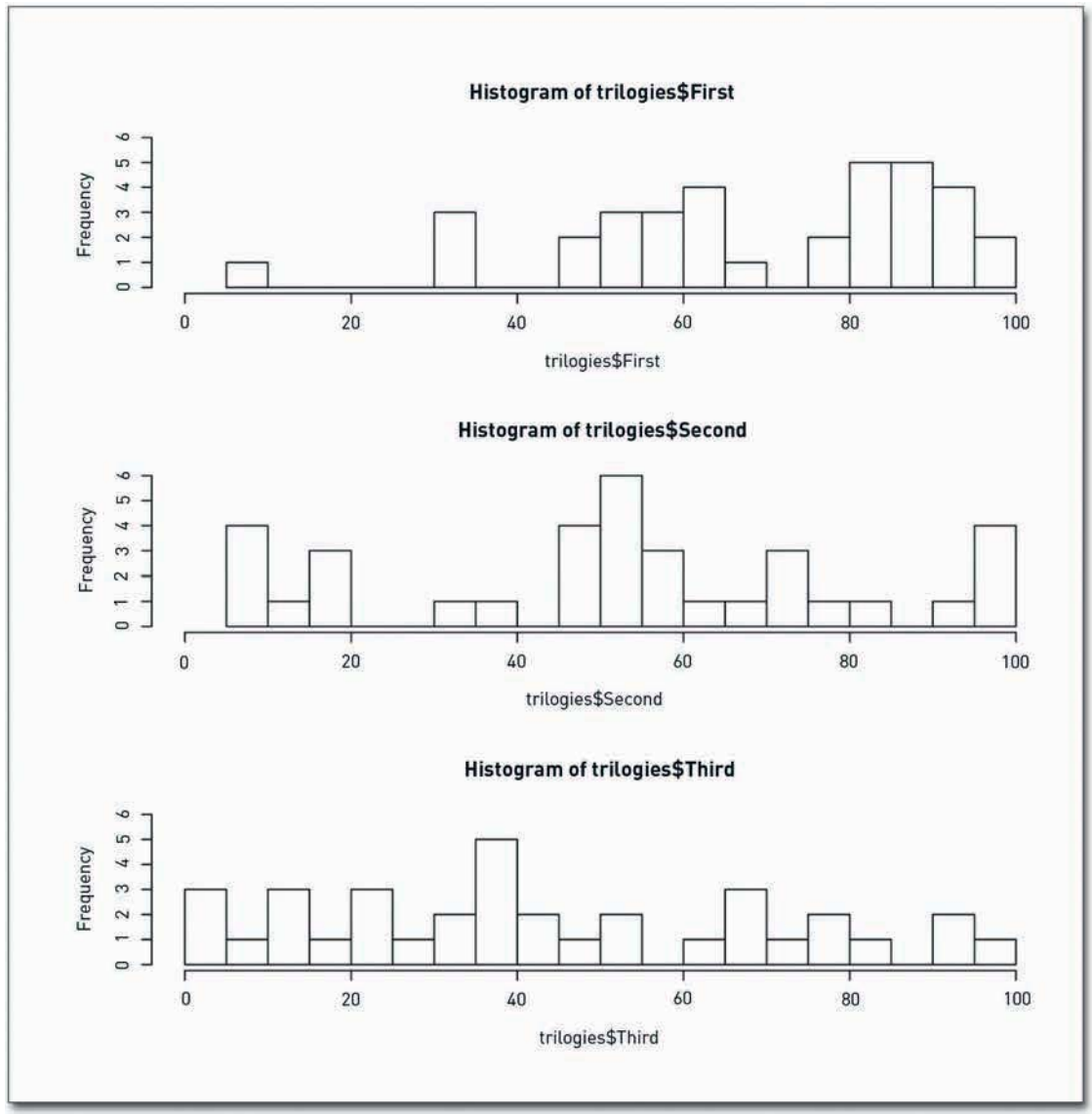


Рис. 6.39. Первоначальный вариант гистограмм трилогий

Вообще-то читатели FlowingData в большинстве своем способны понять любую графику. Они — народ сообразительный. Однако впоследствии на эту гистограмму дали ссылку на сайте

IMDB (Internet Movie Database, или «База данных фильмов в Интернете»). А там аудитория более широкая и менее подготовленная — так что, судя по комментариям, менее смышленые в области работы с данными читатели столкнулись с трудностями, пытаясь интерпретировать представленные таким способом распределения.

А вот второй вариант того же графика (рис. 6.40) определенно оказался более легким для понимания. Он является примером использования панельной диаграммы, в которой каждый столбец представляет рейтинг одной киноленты. Столбцы окрашены в красный, если фильм признан «гнилым», и в зеленый — если «свежим».

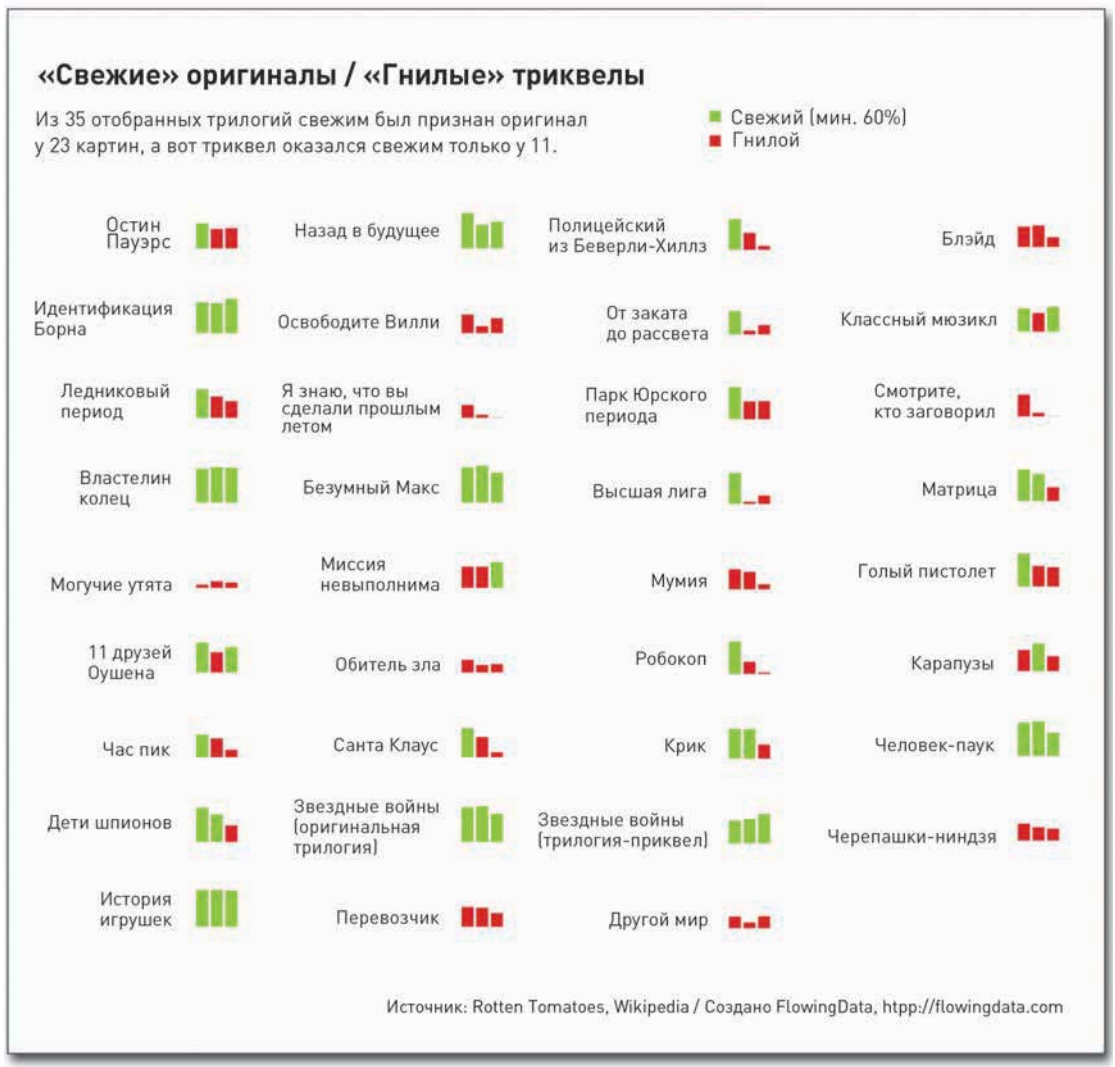


Рис. 6.40. Панельная диаграмма рейтингов трилогий

В случае если вы не знаете, как такое сделать, — имейте в виду, что это просто набор столбцовых диаграмм, так что вы можете изменить параметр *mfrow* так же, как вы это уже делали ранее, и использовать функции `plot()` или `polygon()`. Хотя я воспользовался инструментом Column Graph (Вертикальные полосы) в Illustrator просто потому, что в тот момент у меня эта программа была открыта.

После размещения диаграммы на сайте я узнал еще кое-что. Первое и самое важное — что отнюдь не все люди каждый день имеют дело с множествами и распределениями, так что вам необходимо всегда уделять повышенное внимание пояснению данных и донесению до читателей истории в полном объеме. А второе — что разным людям нравится разное кино, и когда вы говорите, что такой-то фильм — отстой, многие начинают принимать это слишком близко к сердцу.

Закругляясь

Поиск зависимостей в данных иногда бывает непростой задачей и требует проявлять критическое мышление, а не просто слепо наносить на диаграмму числа. Но этот поиск может также стать занятием очень результативным и увлекательным. Ведь именно то, как ваши данные — или, точнее, как явления, которые ваши данные описывают, — связаны и как они взаимодействуют между собой, и делает историю интересной.

В этой главе мы говорили о том, как искать корреляции между многими переменными и вместе с тем как объяснять эти связи в более общем контексте. Используя распределения, попытайтесь понять, как именно различные элементы соотносятся между собой и как они связаны в единое целое. Попробуйте разглядеть в распределениях выбросы и паттерны, а затем поразмышляйте над контекстом всего того, что вы видите. И когда вы заметите нечто интересное, спросите себя: как и почему оно возникло? Подумайте о контексте данных и о возможных объяснениях имеющимся фактам.

Это самый приятный момент в игре с данными — когда вы исследуете их, пытаетесь понять, о чем они говорят, и, возможно, откопать что-нибудь захватывающее. А затем, когда накопаетесь вдоволь, вы сможете объяснить читателям, что именно вы нашли. Помните: не все люди говорят на языке чисел, так что придерживайтесь человеческого языка, понятного широкой аудитории. Однако не бойтесь показаться «ботаном», если у вас понимающие слушатели.

Выявление различий

7

Спортивные комментаторы любят раздавать титулы и называть некоторых атлетов не иначе как «суперзвездами» или «элитой спорта», а остальных объявлять середнячками или ролевыми игроками. Такие классификации обычно возникают не столько на основании какой-то спортивной статистики, сколько после просмотра значительного количества игр. О подобном типе суждений говорят так: «узна́ю, когда увижу». В этом подходе нет ничего плохого. Комментаторы обычно знают, о чем говорят, и всегда учитывают контекст чисел. Я получаю огромное удовольствие, когда удается понаблюдать, как группа спортивных аналитиков обсуждает показатели игры разных спортсменов, — рано или поздно кто-нибудь из них обязательно скажет: «Нельзя судить только по цифрам. Класс игрока проявляется в неуловимых вещах». Вот вам и статистика.

Очевидно, что это относится не только к спорту. Вам может потребоваться найти лучшие рестораны в определенном районе или выявить лояльных клиентов. Вместо того чтобы категорировать все элементы множества, вы можете поискать кого-то или что-то, что выделяется на общем фоне. Эта глава посвящена именно тому, как обособлять группы в рамках совокупности, причем делать это по большому числу критериев, а также как выявлять выбросы, применяя здравый смысл.

Что искать

Проводить сравнения на основании единственной переменной нетрудно. В одном доме больше квадратных метров, чем в другом, и одна кошка весит больше, чем другая. Сравнение по двум параметрам дается сложнее, но и это вполне осуществимо. В первом доме больше квадратных метров, но во втором больше ванных комнат. Первая кошка весит больше, и у нее шерсть короче, а вторая весит меньше, и шерсть у нее длиннее.

Но что делать, если вам необходимо классифицировать сотню домов или сотню кошек? И как быть, если по каждому дому у вас множество переменных, таких как количество спальных комнат, размер сада и объем выплат в кондоминиум? В итоге вам придется сравнивать все количество элементов по количеству переменных. Да, это будет сделать посложнее, и на решении данной проблемы мы сейчас и сфокусируемся.

Допустим, у вас есть множество переменных, но вы хотите классифицировать или сгруппировать элементы множества (скажем, людей или мест) в некие категории и найти выбросы или отклонения. Вам необходимо просмотреть все переменные на предмет различий, но вам также нужно поискать общие черты, имеющие место и по всем переменным. Двое баскетболистов при совершенно разных средних показателях по числу заброшенных мячей могут иметь практически одинаковое количество подборов мяча, перехватов и сыгранных минут в каждой игре. Вам необходимо найти различия, но не следует забывать и о сходствах и зависимостях, точно так же, как это делают... спортивные комментаторы.

Сравнение по нескольким переменным

Одна из главных сложностей при работе с множеством переменных состоит в том, чтобы определить, с чего именно начать. Иногда приходится иметь дело с таким количеством наборов данных и вариантов их сопоставления, что можно и растеряться. Порой бывает лучше всего постараться охватить все данные одним взглядом, и тогда необычные элементы в них сами укажут вам направление движения к следующему интересному моменту.

Теплее, еще теплее

Один из самых прямолинейных способов визуализации таблицы данных — показать ее одновременно всю. Но при этом вместо чисел в качестве индикатора значений вы можете использовать цвета, как показано на тепловой карте (или карте интенсивности) на рис. 7.1.

В итоге вы получите сетку того же размера, что и первоначальная таблица данных, но по ней, благодаря цвету, вы легко сможете определить относительно высокие и низкие значения. Как правило, более интенсивные, темные цвета символизируют бóльшие значения, а светлые — меньшие, но это легко изменить у себя в приложении.

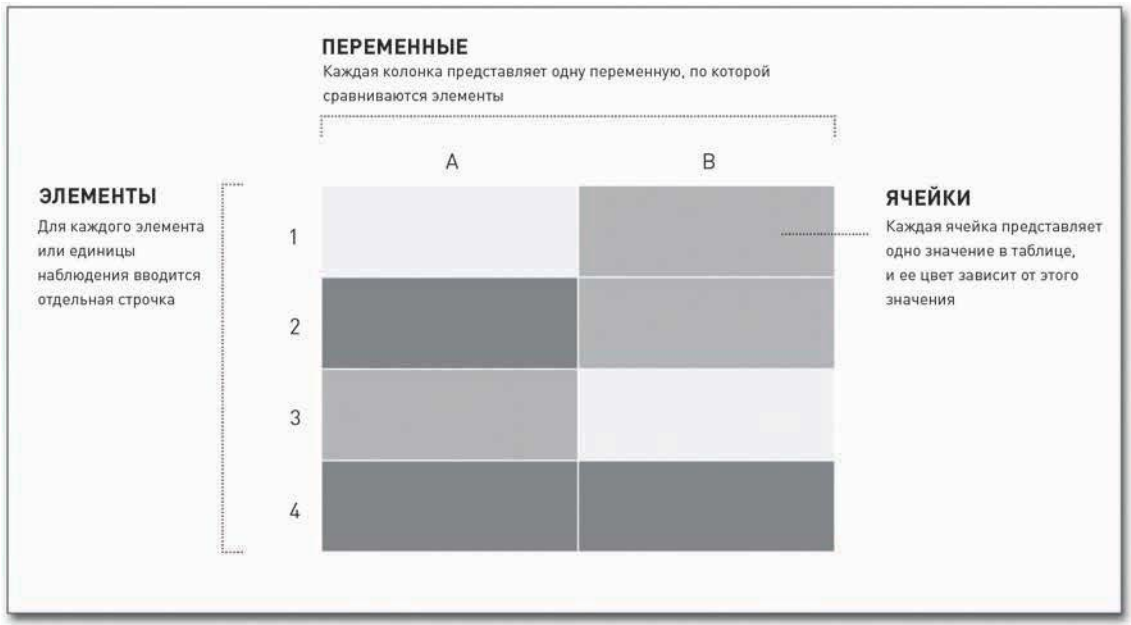


Рис. 7.1. Структура тепловой карты

Тепловую карту (тепловую матрицу) читают так же, как и таблицу. Вы можете прочитать строчки слева направо, чтобы увидеть значения всех переменных для одного отдельного элемента, или можете просмотреть все элементы в сравнении друг с другом по одной переменной.

Такой способ представления данных, скорее всего, покажется вам несколько сложным и запутанным, особенно если таблица данных большая, но, подобрав правильную цветовую схему и немного поработав над сортировкой данных, вы можете создать графику, которая сослужит вам хорошую службу.

СОЗДАЙТЕ ТЕПЛОВУЮ КАРТУ

Создавать тепловые карты в R проще простого. Для этого существует функция `heatmap()`, которая выполнит все математические операции, а вам останется лишь подобрать подходящие для ваших данных цвета и организовать подписи таким образом, чтобы они оставались читаемыми, даже если строк и колонок будет много. Иными словами, R определит структуру, а вы займетесь дизайном. Пока что все должно звучать для вас знакомо.

В данном примере мы обратимся к статистике игр Национальной баскетбольной ассоциации за 2008 год. Данные вы можете скачать в виде CSV-файла с `http://datasets.flowingdata.com/ppg2008.csv`. Итак, у вас 22 колонки: в первой даны имена игроков, а далее идет статистика — очки за игру, процент трехочковых бросков и т. д. Чтобы загрузить данные в R, можете воспользоваться `read.csv()`. А теперь посмотрите на первые пять строчек, чтобы создать себе представление о структуре данных (рис. 7.2).

```
bball <-
   read.csv("http://datasets.flowingdata.com/ppg2008.csv",
   header=TRUE)
bball[1:5,]
```

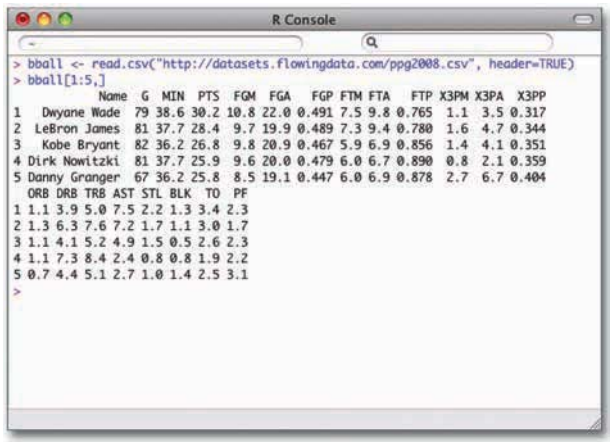


Рис. 7.2. Структура первых пяти строк данных

В настоящий момент игроки выстроены по очкам за игру от наибольшего значения к наименьшему, но вы можете рассортировать их по показателям любой колонки, например по количеству подборов мяча при отскоке или по проценту попаданий с игры, с помощью `order()`.

```
bball_byfgp <- bball[order(bball$FGP, decreasing=TRUE),]
```

Если вы сейчас посмотрите на первые пять строк `bball_byfgp`, вы увидите, что список возглавляют Шакил О'Нил, Дуайт Ховард и По Газоль, а не Дуэйн Уэйд, ЛеБрон Джеймс и Коуб Брайант. Но для данного примера вам необходимо выстроить игроков обратно по количеству очков за игру.

```
bball <- bball[order(bball$PTS, decreasing=FALSE),]
```

В таком виде, в каком она сейчас, колонка с именами соответствует заголовку CSV-файла. Это именно то, что вам нужно. Однако вам еще необходимо вместо номеров присвоить строчкам имена игроков, а потому перенесите первую колонку в названия строк.

```
row.names(bball) <- bball$Name
bball <- bball[,2:20]
```

В данном фрагменте сначала названия строк меняются на первую колонку таблицы данных, затем выбираются все колонки со 2-й по 20-ю, и это подмножество данных пересылается обратно в *bball*.

ПОДСКАЗКА

С помощью аргумента *decreasing* в `order()` вы определяете, в каком порядке хотите видеть данные — в нисходящем или восходящем.

А еще желательно, чтобы данные были скорее в формате матрицы, нежели просто таблицы. Если вы попробуете использовать таблицу данных с функцией `heatmap()`, то вы получите ошибку. По сути, таблица данных — это набор векторов, в котором каждая колонка представляет те или иные количественные показатели. У колонок могут быть различные форматы — числа или некая другая последовательность символов. Матрица же используется, как правило, для представления двухмерного пространства, и данные в ней, от первой до последней ячейки, должны быть единообразны по своему типу.

ПОДСКАЗКА

Процесс визуализации часто требует заниматься также сбором и подготовкой данных. Редко бывает так, чтобы все данные приходили к вам в нужном виде. Готовьтесь к тому, что вам придется перекраивать их так и этак, прежде чем вы сможете перейти к визуализации.

```
bball_matrix <- data.matrix(bball)
```

Итак, данные выстраиваются в нужном вам порядке и форматируются по вашему желанию. Далее вы можете передать их в `heatmap()` и собрать «урожай». Когда для аргумента *scale* вы задаете значение "column", вы тем самым поручаете R при определении цветового градиента использовать наибольшие и наименьшие значения каждой колонки, а не минимум и максимум всей матрицы.

```
bball_heatmap <- heatmap(bball_matrix, Rowv=NA,
   Colv=NA, col = cm.colors(256), scale="column", margins=c(5,10))
```

Полученный результат должен походить на рис. 7.3. С помощью `cm.colors()` был установлен цветовой диапазон от голубого (cyan) до пурпурного (magenta). Функция создает по умолчанию вектор шестнадцатеричных цветов в диапазоне от голубого до пурпурного, при этом количество оттенков между ними можно менять (в данном случае их 256). Обратите внимание на то, что третья колонка (в которой представлено количество очков за игру) начинается с пурпурного, означающего самые высокие показатели Дуэйна Уэйда и Леброна Джеймса, и затем медленно переходит в голубые оттенки до самого темного цвета в конце колонки, где представлены показатели Аллена Айверсона и Нэйта Робинсона. Так же легко вы можете заметить и другие ярко-пурпурные ячейки, представляющие лучшие показатели по отскокам (у Дуайта Ховарда) и лучшего игрока по передачам (это Крис Пол).

Может, вы предпочли бы другую цветовую схему? Для этого достаточно изменить аргумент *col*, представленный как `cm.colors(256)` в строчке кода, которую вы только что выполнили. Наберите `?cm.colors`, чтобы получить справку о том, какие цвета могут использоваться в R. Например, на рис. 7.4 показаны более теплые цвета.

```
bball_heatmap <- heatmap(bball_matrix,
   Rowv=NA, Colv=NA, col = heat.colors(256), scale="column",
   margins=c(5,10))
```

Если в консоли R вы введете `cm.colors(10)`, то получите набор из десяти цветов в диапазоне от голубого до пурпурного, после чего `heatmap()` автоматически подыщет цвет, соответствующий каждому из значений на базе линейной шкалы.

```
[1] "#80FFFFFF" "#99FFFFFF" "#B3FFFFFF" "#CCFFFFFF" "#E6FFFFFF"
[6] "#FFE6FFFF" "#FFCCFFFF" "#FFB3FFFF" "#FF99FFFF" "#FF80FFFF"
```

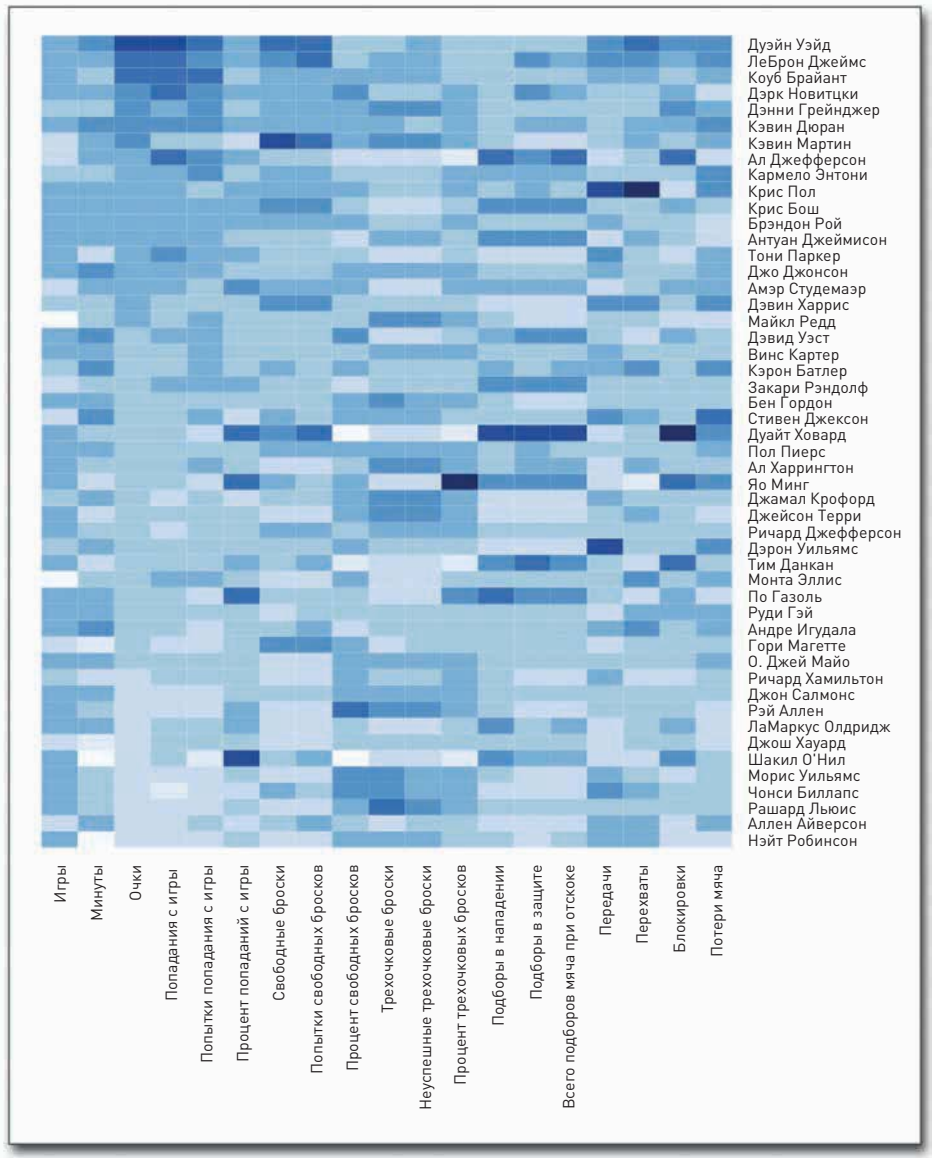


Рис. 7.3. Тепловая карта по умолчанию, выстроенная по показателю «очки за игру» (PTS)

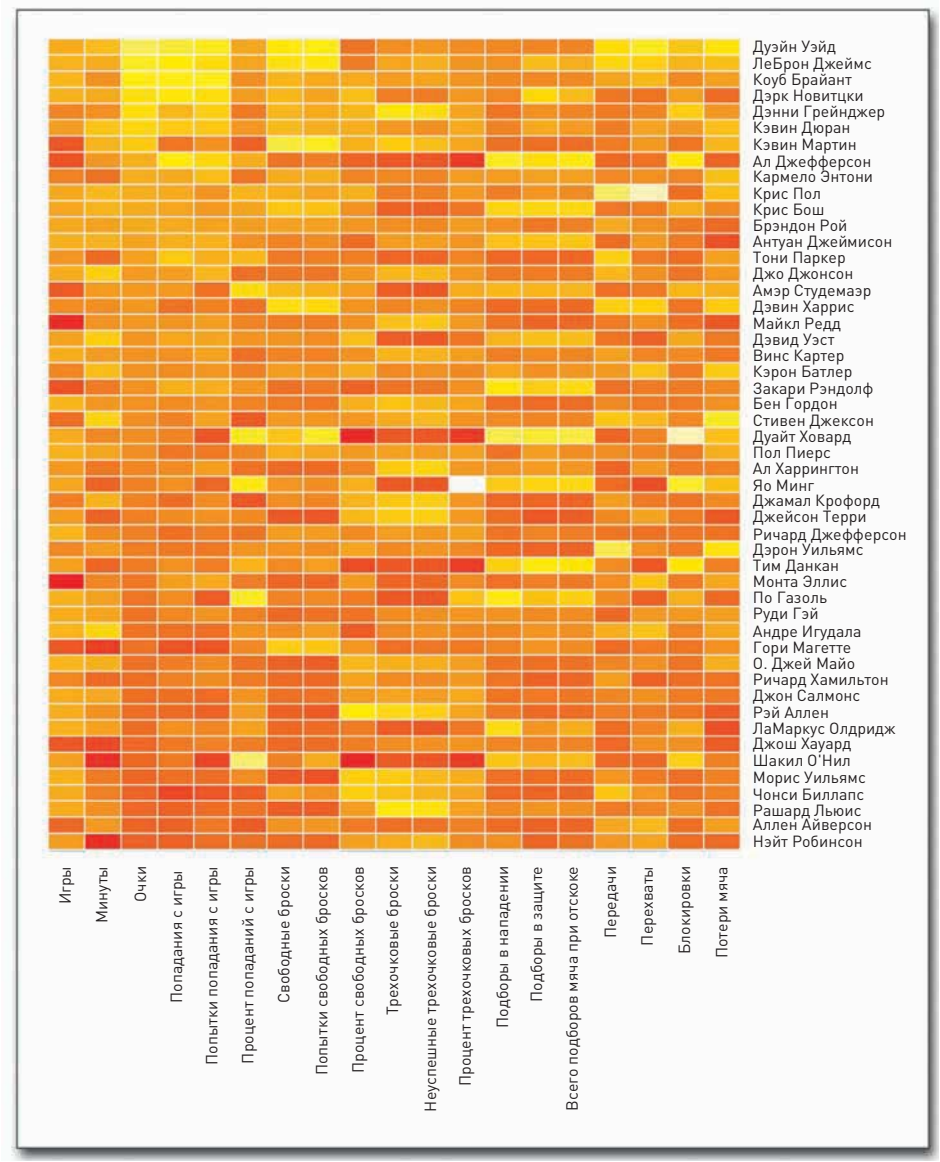


Рис. 7.4. Тепловая карта с желто-красной цветовой шкалой

И это замечательно, это позволяет вам с легкостью создавать свои собственные цветовые шкалы. Например, вы можете сходить на сайт `0to255.com` и там выбрать главный цвет, а затем на его основе выстроить все остальное. На рис. 7.5 представлен градиент на базе красного. Вы можете выбрать несколько цветов от светлого до темного и затем просто передать их в `heatmap()`, как показано на рис. 7.6. Вместо того чтобы использовать R для создания вектора цветов, в переменной *red_colors* нужно указать свои собственные значения.

```
red_colors <- c("#ffd3cd", "#ffc4bc", "#ffb5ab",
   "#ffa69a", "#ff9789", "#ff8978", "#ff7a67", "#ff6b56",
   "#ff5c45", "#ff4d34")
bball_heatmap <- heatmap(bball_matrix, Rowv=NA,
   Colv=NA, col = red_colors, scale="column", margins=c(5,10))
```

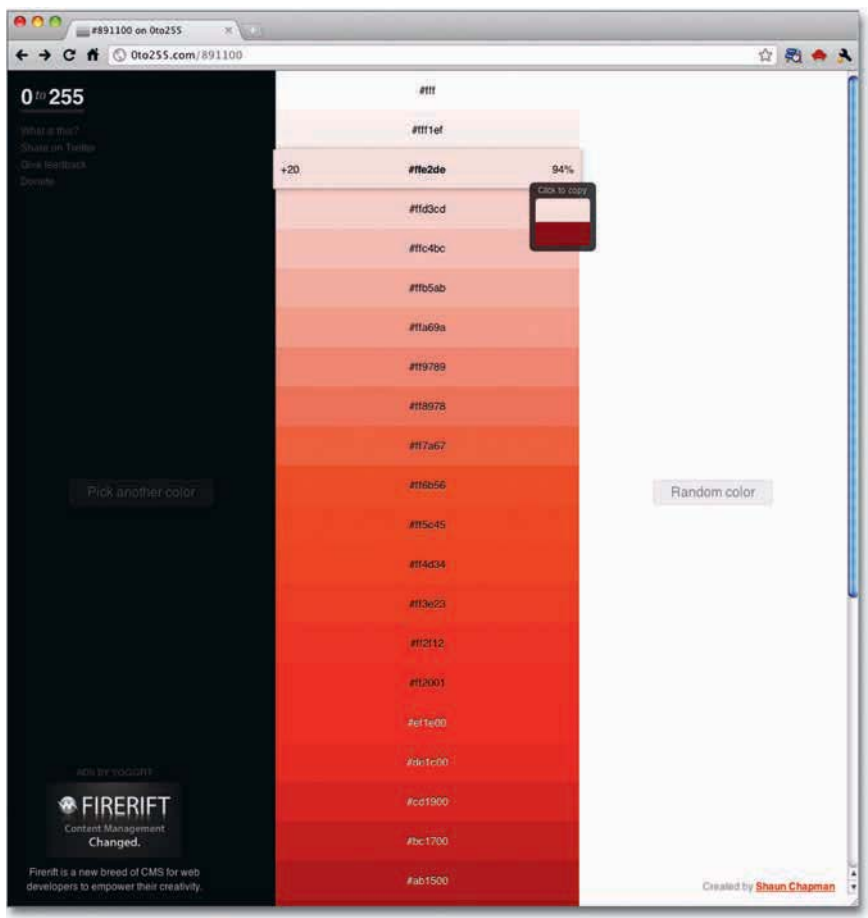


Рис. 7.5. Красный градиент от 0to255.com

ПОДСКАЗКА

Выбирайте цвет осторожно, так как он способен задать тон всей вашей истории. Например, если тема у вас мрачная, наверное, лучше отдать предпочтение более нейтральным, приглушенным тонам, а если тема радостная или легкомысленная, то можете использовать жизнерадостные цвета.

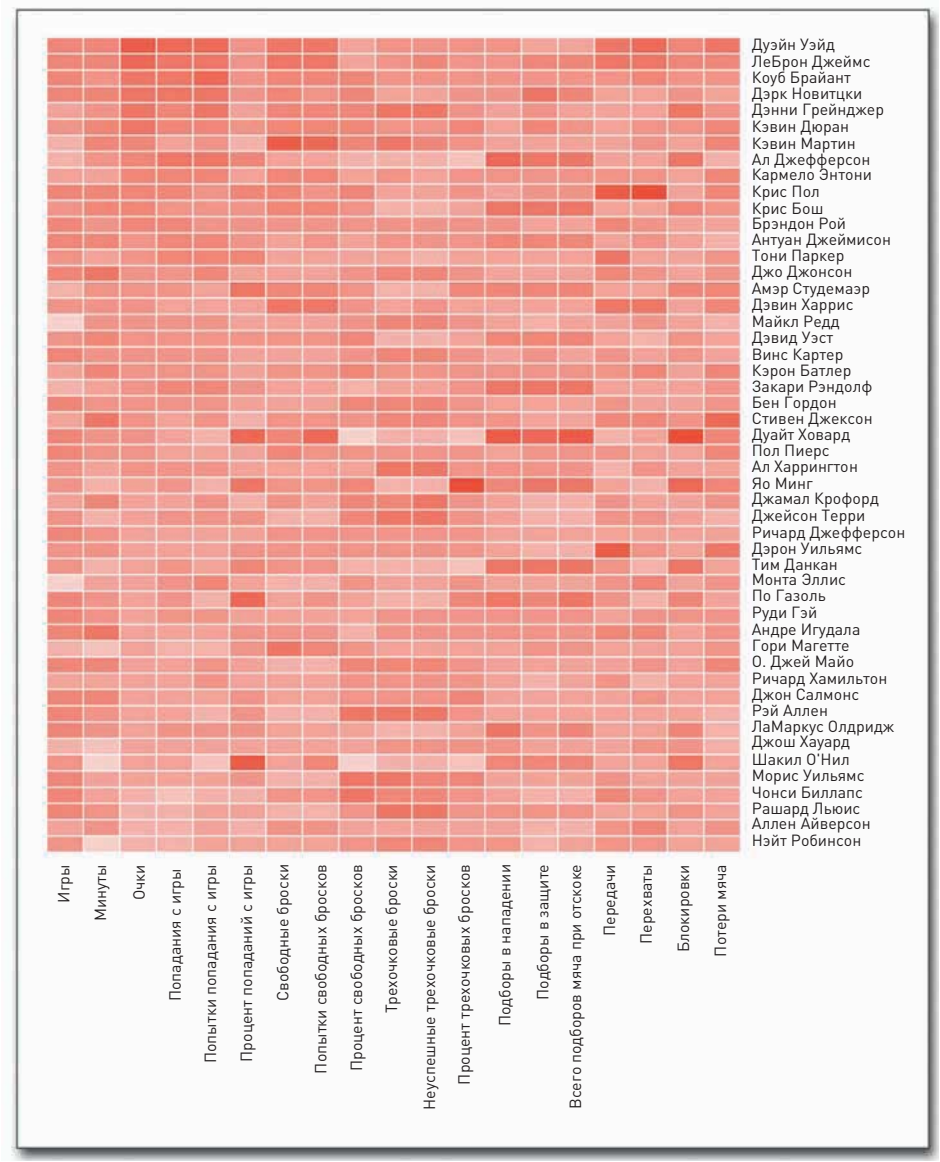


Рис. 7.6. Тепловая карта с использованием красной шкалы

Если вы хотите самостоятельно подбирать цвета, можете воспользоваться пакетом RColorBrewer. Он не стоит по умолчанию, так что вам придется загрузить и установить его с помощью Package Installer, если вы этого еще не сделали. RColorBrewer был разработан картографом Синтией Брюэр (Cynthia Brewer) и изначально предназначался для использования при рисовании карт, однако он будет полезен при создании любого другого типа инфографики.

▶ Посмотрите интерактивную версию ColorBrewer по адресу `http://colorbrewer2.com`. Там вы сможете поиграть с различными опциями в выпадающем меню и поглядеть, как будут смотреться те или иные цветовые схемы применительно к карте, представленной на сайте в качестве образца.

С ним вы можете выбирать из множества опций: там есть и последовательные цветовые палитры, и расходящиеся, а также огромное количество оттенков. Для целей нашего примера подойдет и простая голубая палитра. Наберите **`?brewer.pal`** в консоли R, чтобы ознакомиться с дополнительными опциями — это довольно увлекательно. Если вы уже установили RColorBrewer, вам будет достаточно ввести следующий код, чтобы создать тепловую карту, используя голубую палитру с десятью оттенками (рис. 7.7).

```
library(RColorBrewer)
bball_heatmap <- heatmap(bball_matrix, Rowv=NA,
   Colv=NA, col = brewer.pal(9, "Blues"),
   scale="column", margins=c(5,10))
```

Теперь вы можете перенести карту, представленную на рис. 7.7, в Illustrator и там чуть-чуть украсить ее. Ей не нужно многого, но подписи лучше сделать более читаемыми и смягчить оттенки так, чтобы изображение легче читалось.

Что касается первого, лучше расшифровать аббревиатуры в подписях. Как фанат баскетбола, я точно знаю, какое сокращение что означает, но для человека, не столь сильно погруженного в тему, это все может оказаться абракадаброй. А если говорить по второму вопросу — о цветах, — вы можете приглушить контрастность, используя прозрачность, для чего нужно открыть окно Color (Цвет) в Illustrator. Границы придадут ячейкам отчетливости, и карту станет легко читать слева направо и сверху вниз. Окончательный вариант представлен на рис. 7.8.

Читаем по лицам

Что хорошо с тепловыми картами, так то, что они дают возможность охватить массив данных одним взглядом, но при этом все сфокусировано на отдельных показателях. Таким образом легко разглядеть самые высокие и самые низкие значения по очкам или по подбору мяча за игру, но вот сравнивать одного спортсмена с другим при таком подходе становится уже несколько сложнее.

А ведь очень часто хочется осмотреть один элемент в полном объеме, а не в разбивке по разным показателям. Сделать это можно с помощью «лиц Чернова»*. Данный метод не является особо точным и неподготовленного читателя он может затруднить. Однако в определенных ситуациях «лица Чернова» могут быть весьма полезными, да и людям, много работающим с данными, применение этого метода зачастую приносит массу удовольствия, так что освоить его определенно стоит.

Смысл метода «лица Чернова» состоит в одновременной демонстрации множества переменных посредством специфического размещения частей человеческого лица, таких как уши, волосы,

* «Лица Чернова» — метод отображения многомерных данных в виде человеческого лица, придуманный американским математиком Германом Черновым и основанный на сильно развитой способности людей к восприятию лиц и мельчайших изменений в них. *Прим. пер.*

глаза и нос, на основе чисел определенного набора данных (рис. 7.9). Логический посыл таков: если вы в реальной жизни легко «прочитываете» лица людей, то вы сможете так же легко подметить даже мельчайшие изменения в них и тогда, когда их черты будут представлять некие данные. Предположение довольно смелое, но можно допустить, что верное.

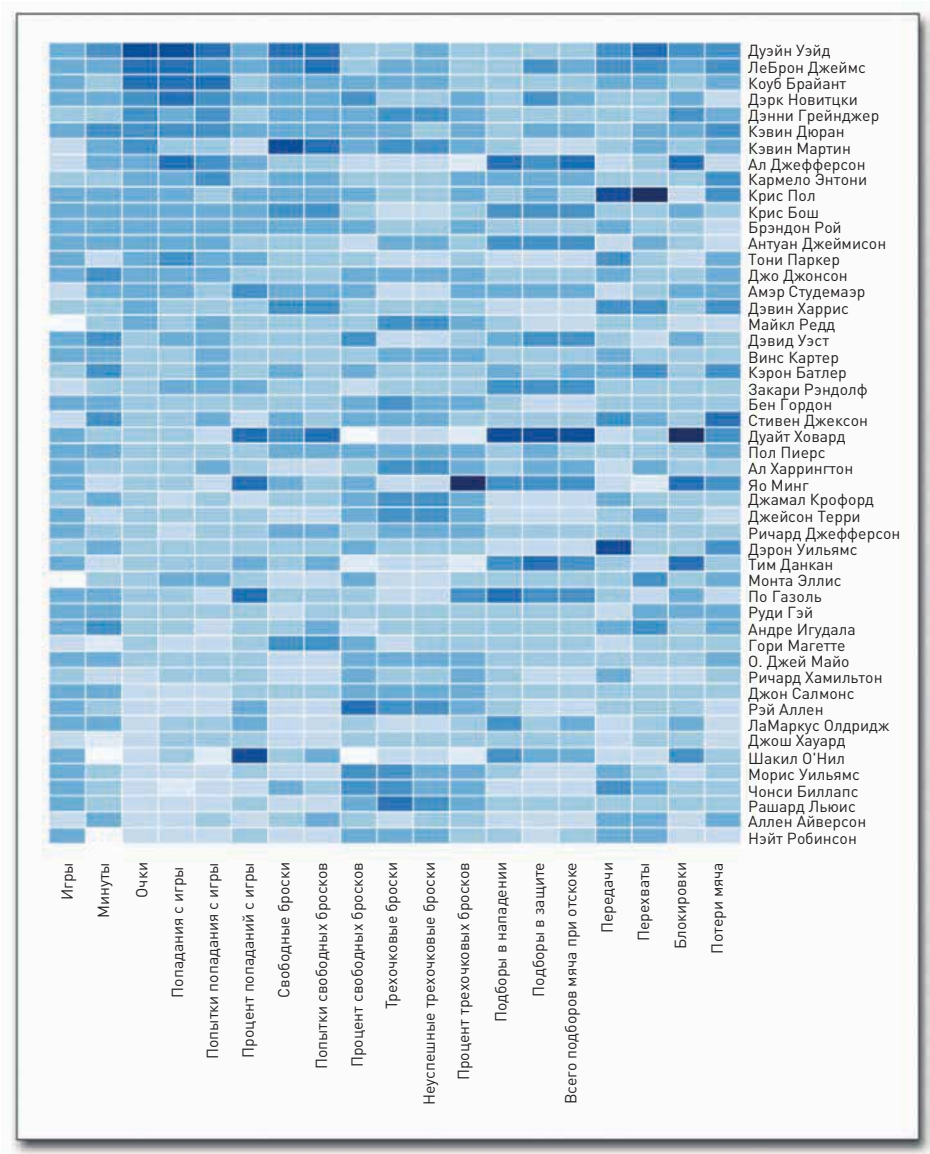
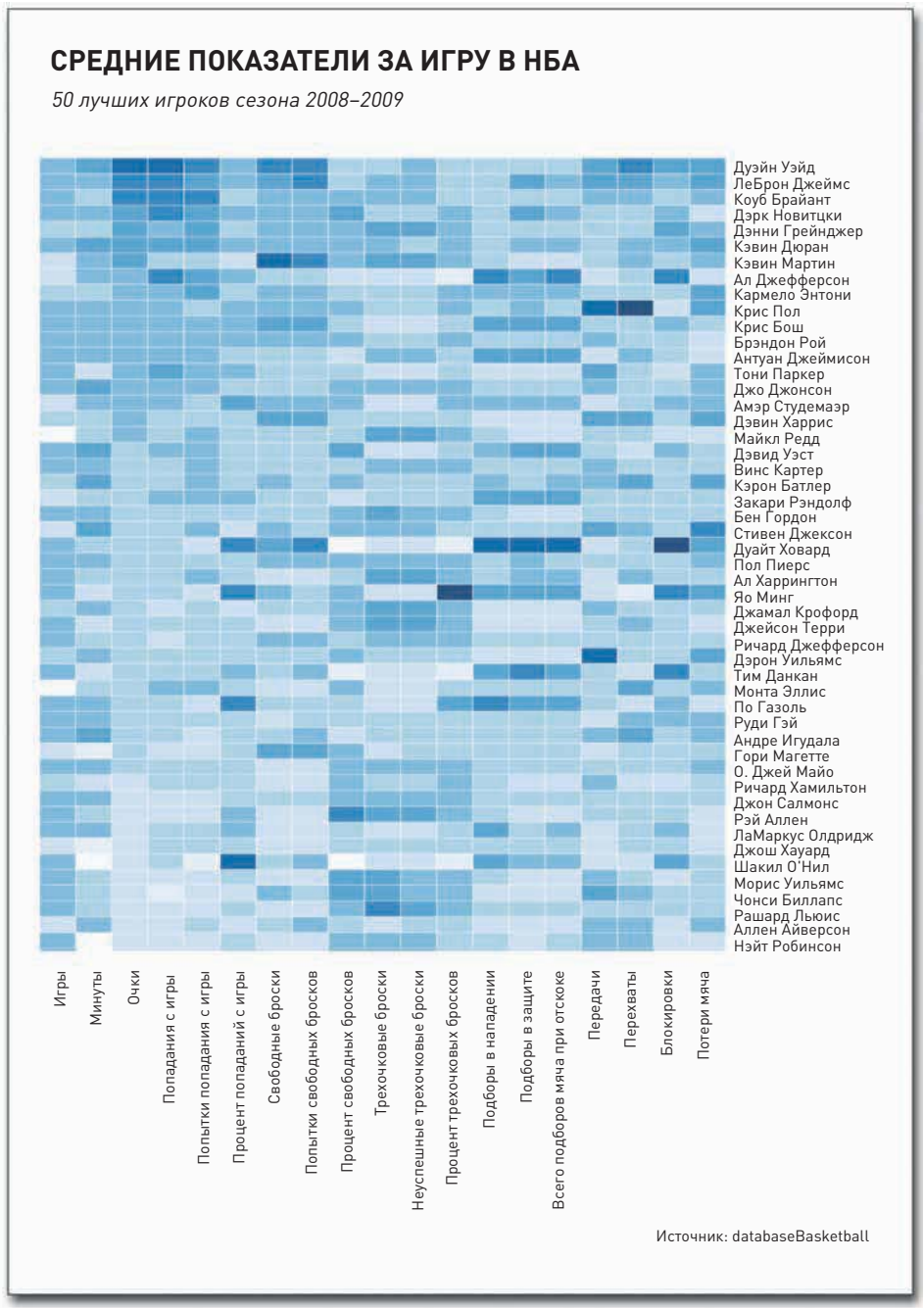


Рис. 7.7. Тепловая карта с использованием RColorBrewer для подбора цветовой палитры



▶ Майк Бостоk (Mike Bostock) перенес эту карту в Protovis, так что вы можете найти ее и в тамошнем разделе с примерами. Внешний вид графики не изменился, но возможности стали богаче: когда мышка оказывается поверх той или иной ячейки, всплывает подсказка с дополнительной информацией.

Рис. 7.8. Тепловая карта, демонстрирующая средние показатели за игру 50 лучших игроков НБА сезона 2008–2009 гг.

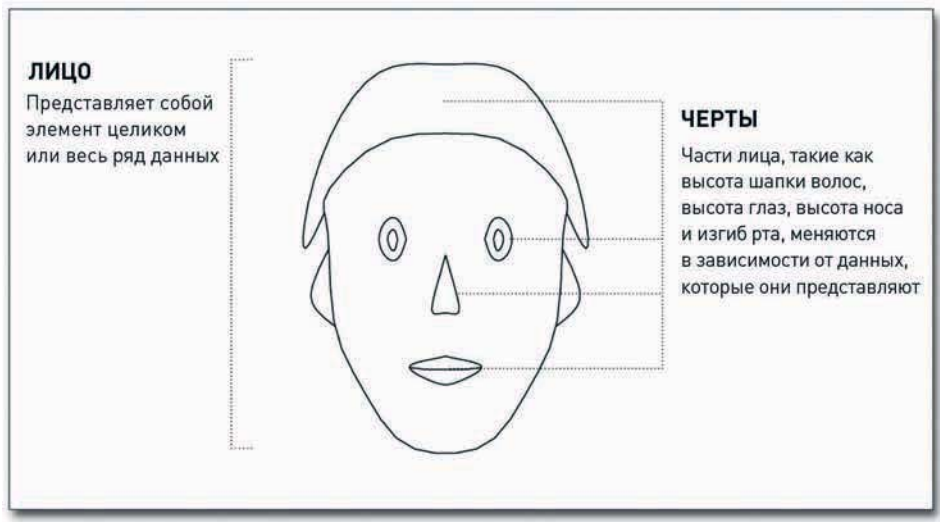


Рис. 7.9. Структура «лиц Чернова»

Как вы увидите в следующем примере, бóльшие значения принимают форму больших глаз или большой шапки волос, а меньшие значения делают черты лица мельче. Помимо размера, значение имеют и такие особенности, как изгиб рта или форма лица.

СОЗДАЙТЕ «ЛИЦА ЧЕРНОВА»

Давайте вернемся к данным, содержащим информацию о 50 лучших игроках НБА сезона 2008–2009 годов. Нам нужно будет создать по одному лицу для каждого игрока. Не беспокойтесь — вам не придется вручную вычерчивать их все по очереди. В пакете `aplpack` в R есть функция `faces()`, которая поможет вам получить все необходимое.

Если данный пакет у вас еще не установлен, сделайте это сейчас с помощью `install.packages()` или через Package Installer. Если вы теряетесь в догадках, сообщу, что название `aplpack` расшифровывается как «another plotting package» («другой пакет для графопостроения»), а создал его Ханс Петер Вольф (Hans Peter Wolf). По завершении установки пакет обычно загружается автоматически, но если загрузки не произойдет, тогда вам необходимо будет сделать это самостоятельно.

```
library(aplpack)
```

Вы должны были еще раньше загрузить баскетбольные данные, когда создавали тепловую карту. Но если вы этого до сих пор не сделали, воспользуйтесь снова `read.csv()`, чтобы загрузить данные напрямую с URL.

```
bball <- read.csv("http://datasets.flowingdata.com/ppg2008.csv",
header=TRUE)
```

ПРИМЕЧАНИЕ

Новейшая версия позволяет функции `faces()` добавлять также и цвета. В данном примере, однако, в качестве значения *ncolors* вы зададите 0, чтобы получить черно-белое изображение. Но посмотрите, как можно было бы использовать цветовые векторы — принцип действия точно такой же, как и в предыдущем примере с тепловой картой.

После того как пакет и данные загрузятся, сражу же можно будет приступить к созданию «лиц Чернова» с помощью функции `faces()`, как это показано на рис. 7.10.

```
faces(bball[,2:16], ncolors=0)
```

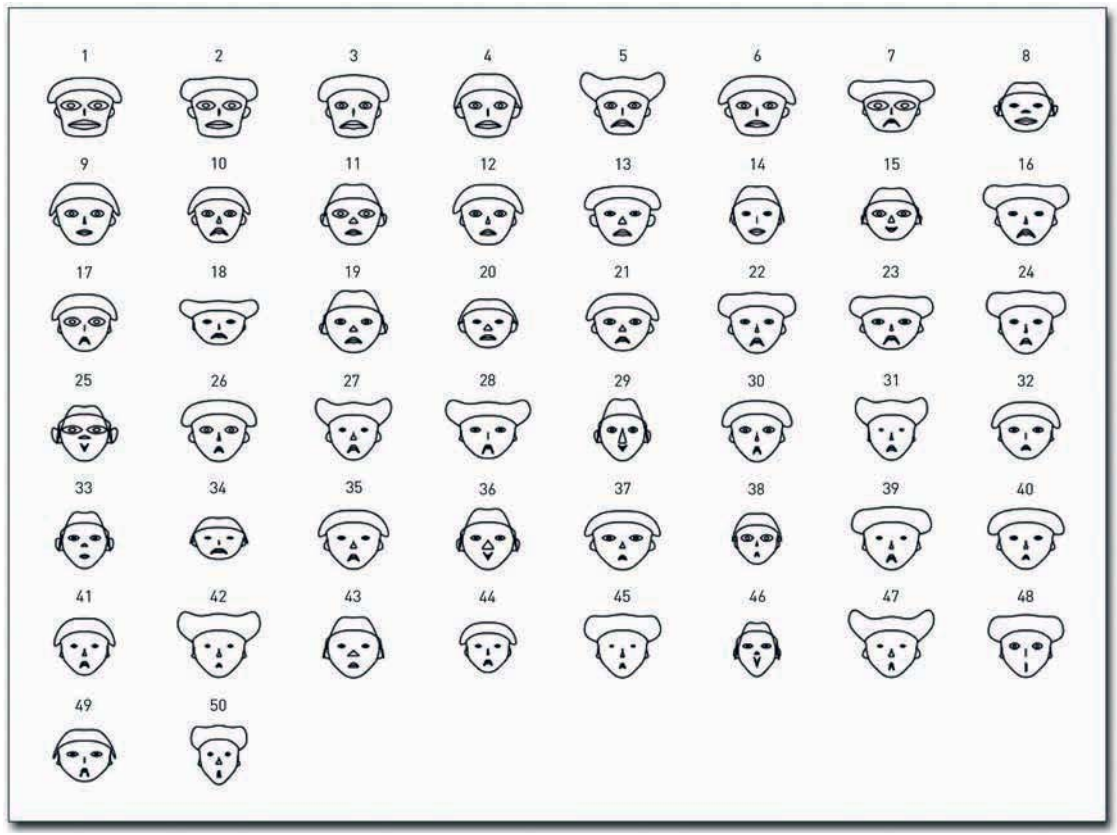


Рис. 7.10. «Лица Чернова» по умолчанию

У вашего набора данных есть 20 переменных плюс имена игроков. Однако функция `faces()`, предлагаемая `aplpack`, дает возможность работать не более чем с 15 переменными, потому что в ней есть только 15 черт лица, которые можно менять. Вот почему вы создадите подмножество из данных, содержащихся в колонках от 2 до 16.

Что будет представлять каждое лицо? Функция `faces()` меняет черты в следующем порядке, соответствующем порядку колонок данных:

1. Высота лица.
2. Ширина лица.
3. Форма лица.
4. Высота рта.
5. Ширина рта.
6. Изгиб рта.

7. Высота глаз.
8. Ширина глаз.
9. Высота шапки волос.
10. Ширина волос.
11. Оформление волос.
12. Высота носа.
13. Ширина носа.
14. Ширина ушей.
15. Высота ушей.

В нашем примере высота лица будет представлять количество сыгранных игр, а высота рта — количество попаданий с игры в среднем. Пока что все это немного бесполезно, так как у лиц нет имен, но тем не менее вы можете заметить, что у первых нескольких баскетболистов более хорошо оформленный игровой процесс, чем у остальных, и что у игрока под номером 7, например, более широкая шапка волос, что соответствует большему количеству трехочковых бросков.

Используя аргумент *labels* в функции `faces()`, дайте лицам имена, как показано на рис. 7.11.

```
faces(bball[,2:16], labels=bball$Name)
```

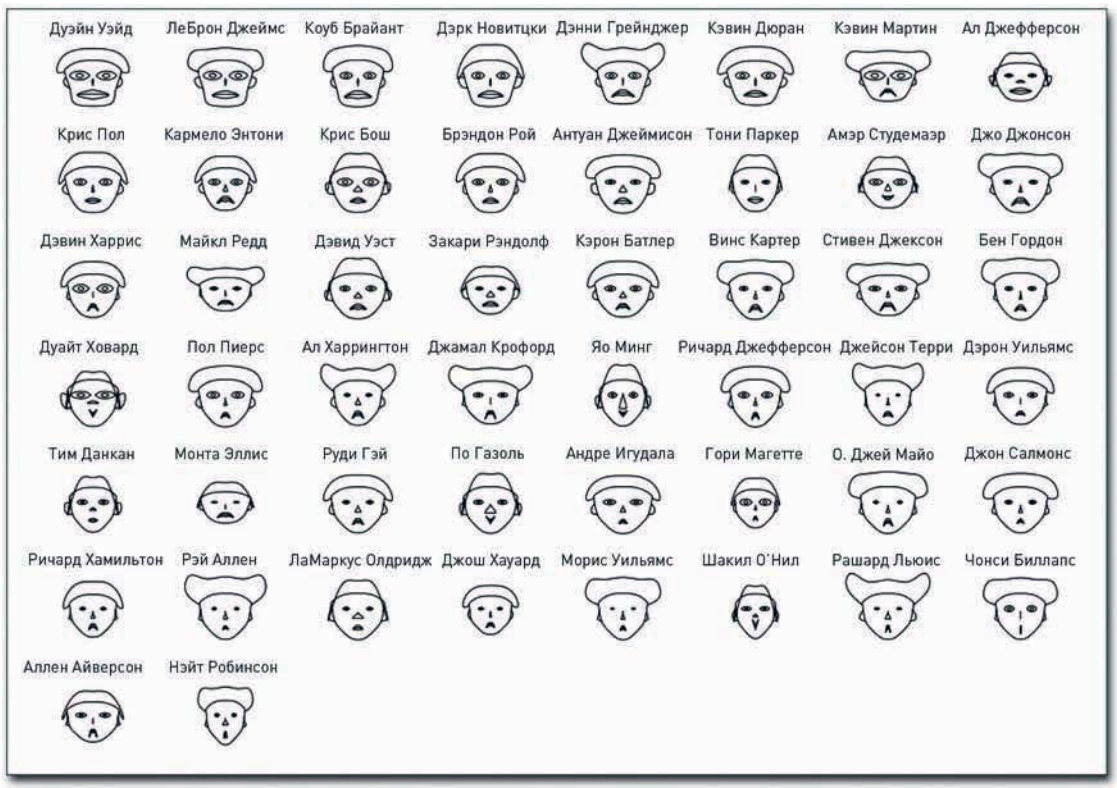


Рис. 7.11. «Лица Чернова» с именами игроков

ПОДСКАЗКА

Иногда при работе с большим количеством индивидуумов оказывается полезно сгруппировать их по категориям, чтобы лица было проще «сканировать». В этом примере, скажем, вы могли бы поделить лица по занимаемым игроками позициям: на форвардов, защитников и центровых.

ПОДСКАЗКА

Свободное пространство (пробелы) способно сделать графический объект более читабельным, особенно при наличии существенного количества элементов, которые необходимо рассмотреть и оценить.

Так уже лучше. Теперь видно, какое лицо кому из игроков «принадлежит». Чтобы идентифицировать разыгрывающих защитников, вы можете начать, допустим, с Криса Пола и поискать похожие лица, такие как лица Дэвина Харриса или Дэрона Уильямса. Чонси Биллапс в нижнем правом углу также является разыгрывающим защитником, но его лицо выглядит не так, как у остальных. Его волосы выше, а ширина рта небольшая, что означает высокий процент свободных бросков и попыток попадания с игры соответственно.

Чтобы сделать графику более читабельной, вы можете увеличить пробел между рядами и хотя бы коротко пояснить, что означают различные черты лица (рис. 7.12). Обычно я вставляю графическую легенду, но так как мы ввели в употребление все черты лица, будет не так уж просто дать такое количество указателей к одному лицу.

И опять-таки полезность «лиц Чернова» может варьировать в зависимости от набора данных и от аудитории, так что вам решать, стоит ли использовать этот метод или нет. Во всех случаях следует учитывать, что люди, незнакомые с данным методом, склонны воспринимать нарисованное буквально, как будто лицо, представляющее, скажем, Шакила О'Нила, должно походить на самого игрока, а ведь почти всем известно, что он — один из самых крупных по телосложению баскетболистов всех времен и народов.

ПОДСКАЗКА

Когда вы будете создавать графику, попробуйте поставить себя на место читателя. В отличие от вас, он не так хорошо знаком с методами визуализации и не осведомлен о характере данных, так что это ваша задача — задача сторителлера — все ему объяснить.

В том же духе с помощью «лиц Чернова» я создал графику преступности в Соединенных Штатах (рис. 7.13), в результате чего получил даже комментарий с обвинениями в расизме из-за того, как выглядят лица, представляющие штаты с высоким уровнем правонарушений. Мне самому подобное сравнение никогда в голову не приходило — для меня изменение черт лиц сродни изменению высоты столбцов в диаграмме, но здесь явно есть над чем поразмыслить.

В лучах радара

Для наглядного представления многомерных данных ту же самую идею можно применить, используя, однако, не лица, а другую графическую реализацию. Так, в зависимости от значений различных показателей можно менять не черты лиц, а очертания фигур. В этом и состоит идея диаграмм-радаров, также известных как паутинные, или лепестковые, диаграммы (а еще их иногда называют диаграммами-звездами).

Как показано на рис. 7.14, вы можете начертить несколько осей, по одной для каждой переменной, начинающихся с центра и расположенных на равном удалении от соседних, составляя таким образом круг («колесо»). Центр — это минимальное значение каждой из переменных, а конец оси — ее максимум. Если вы создаете диаграмму для одного элемента, начните с одной переменной и проведите соединительную линию до соответствующей точки на соседней оси. В конечном итоге вы получите нечто похожее на радар (или на паутину, на цветок, на звезду).

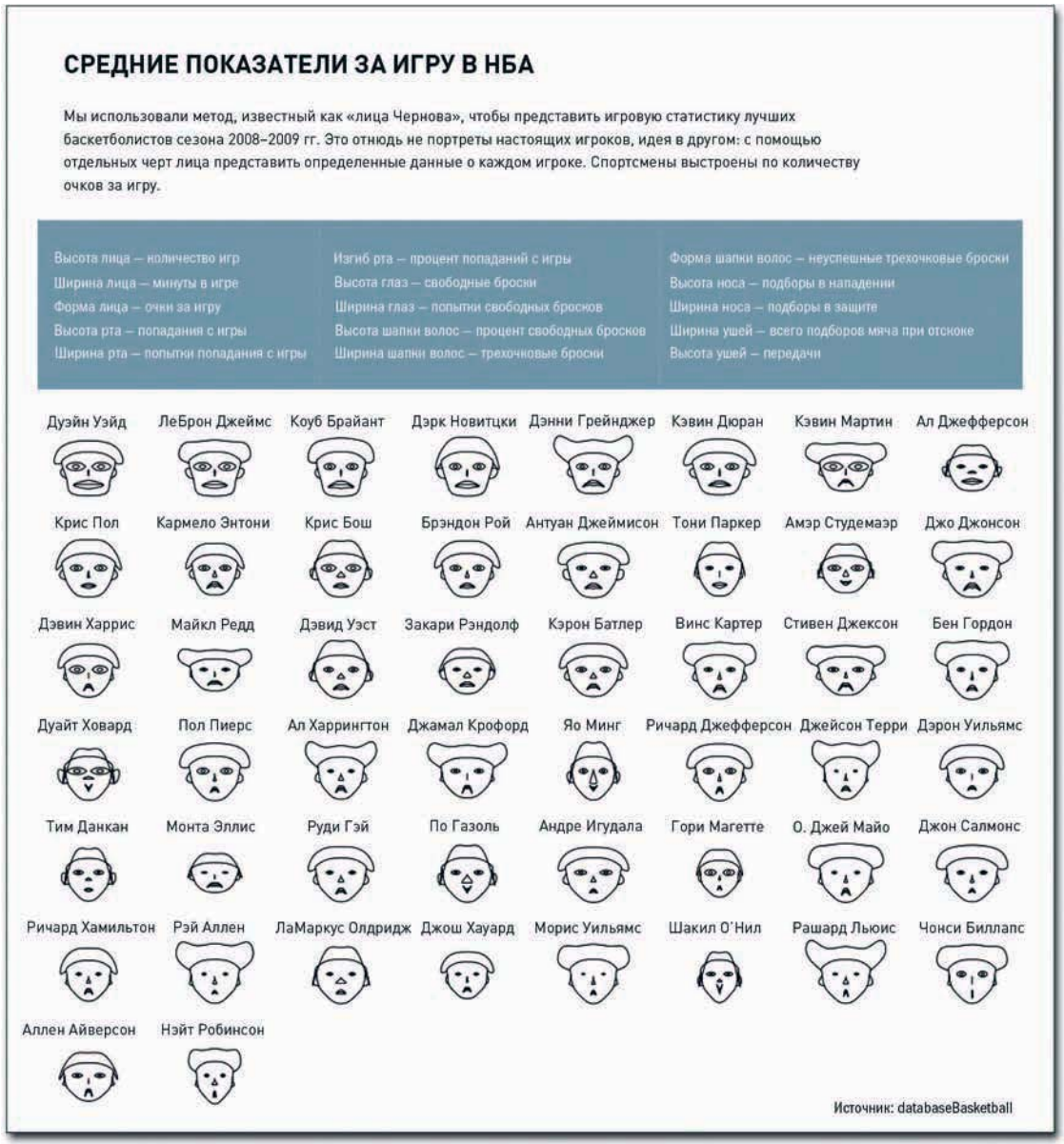


Рис. 7.12. «Лица Чернова» для лучших игроков НБА сезона 2008–2009 гг.

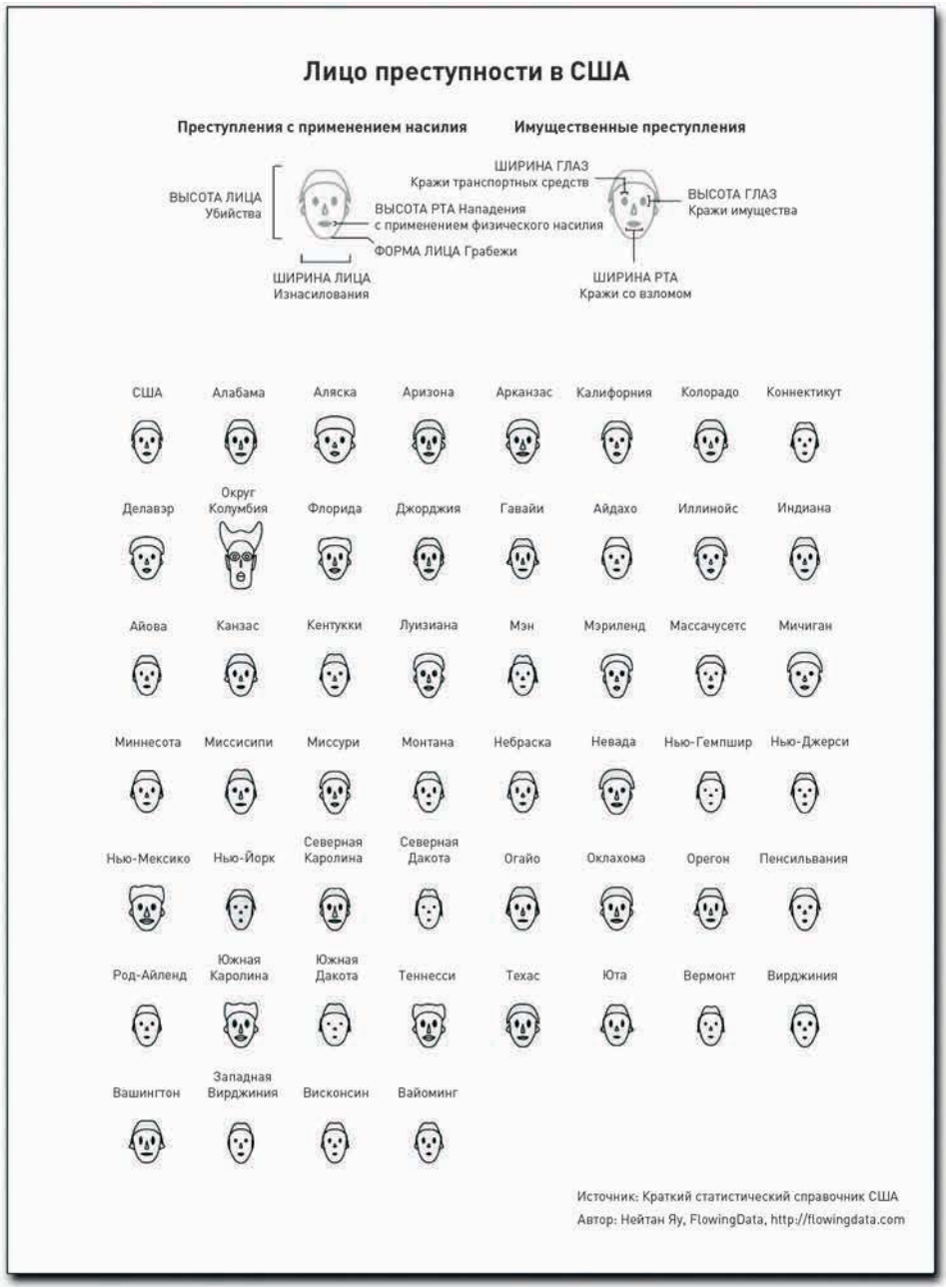


Рис. 7.13. Преступность в США, представленная «лицами Чернова» (каждое лицо — отдельный штат)

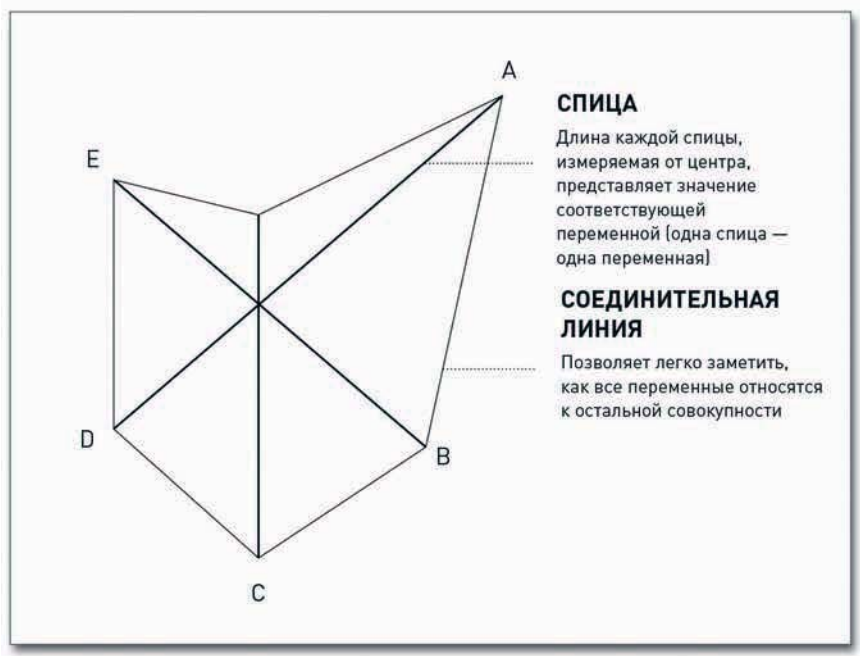


Рис. 7.14. Структура диаграммы-радара

На одной диаграмме можно представить не только один, но и несколько элементов. Однако в спешке легко напортачить и сделать графику бесполезной, иными словами, плохо рассказать историю. Так что лучше придерживаться принципа «один элемент — одна диаграмма» и сравнивать их между собой.

СОЗДАЙТЕ ДИАГРАММУ-РАДАР

Теперь используйте те же данные по преступности, что и для рис. 7.13, и посмотрите, в чем диаграмма-радар как средство визуализации лучше или хуже. Давайте начнем с начала. С загрузки данных в R.

```
crime <- read.csv("http://datasets.flowingdata.com/
crimeRatesByState-formatted.csv")
```

После этого создать диаграмму-радар становится так же просто, как и нарисовать «лица Чернова». Воспользуйтесь функцией `stars()`, которая поставляется с базовым пакетом R.

```
stars(crime)
```

Диаграммы, полученные по умолчанию, представлены на рис. 7.15. Будем надеяться, что уж они-то никого не обидят. Конечно, вам по-прежнему нужны подписи с названиями штатов,

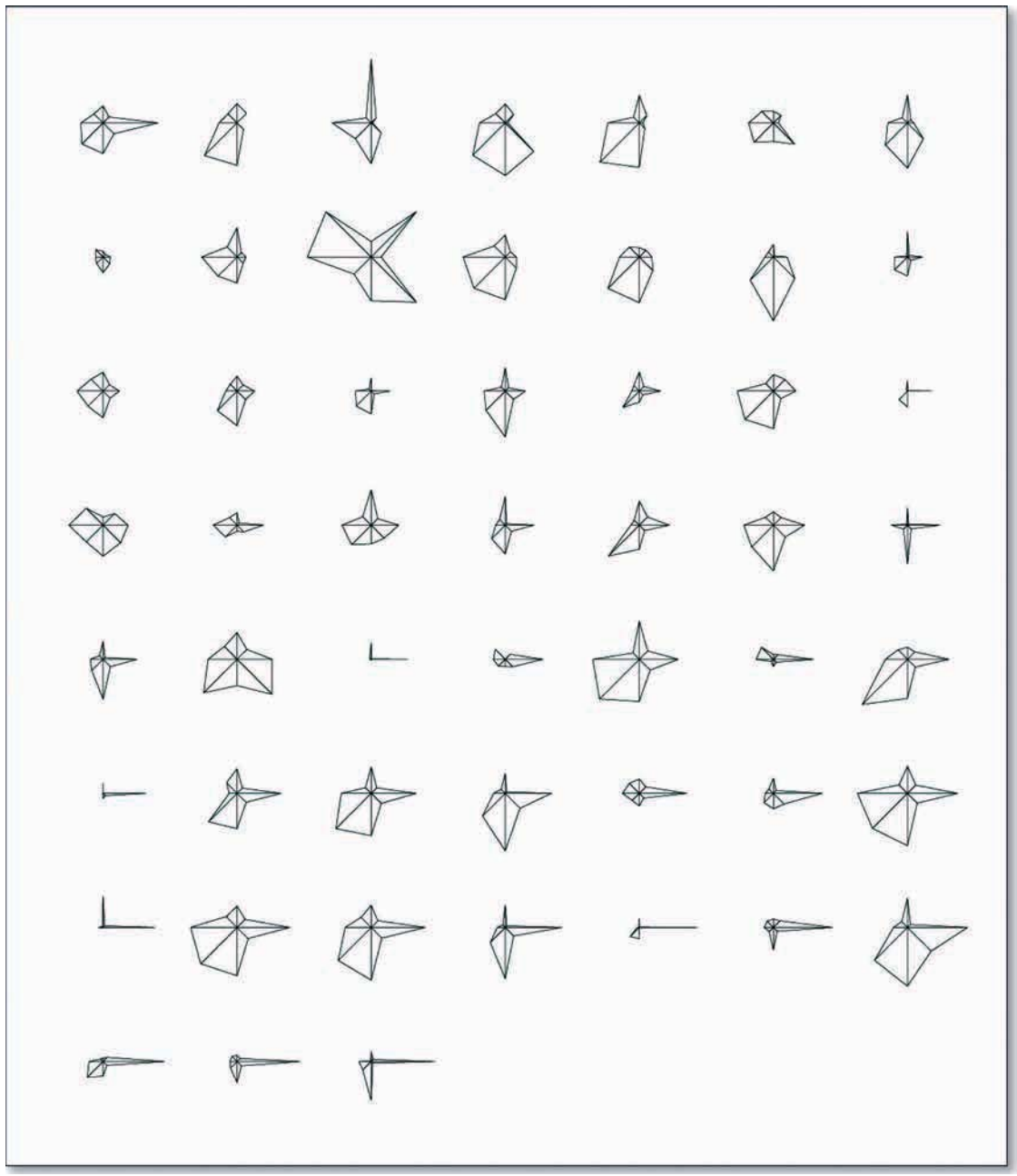


Рис. 7.15. Диаграммы-радары по умолчанию, демонстрирующие преступность в США по штатам

но, помимо этого, вам необходимо дать также ключ к пониманию, что есть что. Здесь так же, как и с функцией `faces()`, соблюдается определенный порядок, но вы не знаете, где какая переменная и откуда все начинается. А потому позаботьтесь и о том, и о другом. Обратите внимание еще и вот на что: вы можете изменить первую колонку таким образом, чтобы в ней выстроился ряд названий штатов. Вы это уже делали, создавая тепловую карту. И вам нужно будет установить `flip.labels` на *FALSE*, так как вы не хотите, чтобы подписи имели разную высоту. Результаты представлены на рис. 7.16.

```
row.names(crime) <- crime$state
crime <- crime[,2:7]
stars(crime, flip.labels=FALSE, key.loc = c(15, 1.5))
```

Теперь уже легко заметить сходства и отличия. В варианте с лицами Чернова округ Колумбия, по сравнению с другими штатами, выглядел как клоун с выпученными глазами, а в варианте с диаграммами-радарами, как вы сами видите, он демонстрирует относительно высокие показатели по некоторым видам преступлений, но вместе с тем у него сравнительно низкие показатели по статьям «изнасилования» и «кражи со взломом». А еще легко можно выявить штаты с относительно невысоким уровнем преступности, такие, как Нью-Гемпшир и Род-Айленд. А есть и такие штаты, как Северная Каролина, у которых высоки показатели только по одному типу преступлений.

На мой взгляд, для этого набора данных больше подходит именно такой метод визуализации. У него есть еще две разновидности, которые, возможно, вы захотите опробовать применительно к вашим наборам данных. Первая разновидность ограничивает пространство размещения данных только верхней половиной круга, как это показано на рис. 7.17.

```
stars(crime, flip.labels=FALSE, key.loc = c(15, 1.5), full=FALSE)
```

Вторая разновидность вместо размещения точек использует длину сегментов, как показано на рис 7.18. На самом деле это уже не диаграмма-радар, а диаграмма Найтингейл* (также известная как полярная диаграмма), но идея по сути такая же. Если вы решите остановиться на данной разновидности, вы, возможно, захотите опробовать и другие цветовые схемы, отличные от безумного варианта, установленного по умолчанию.

```
stars(crime, flip.labels=FALSE, key.loc = c(15, 1.5), draw.segments=TRUE)
```

Однако, как я уже говорил, меня вполне устраивает первоначальный вариант (рис. 7.16), так что вы можете перенести его в Illustrator и кое-что подчистить. При этом сильно менять ничего не нужно. Добавьте побольше пространства между рядами, чтобы стало яснее, к чему относится каждая подпись, и, возможно, стоит также перенести ключ наверх, чтобы читатели знали, во что они собираются углубиться (рис. 7.19). В остальном все вполне приемлемо.

---

* Флоренс Найтингейл (Florence Nightingale) — английская медсестра, участник Крымской войны и общественный деятель, а также способный математик, статистик и новатор в использовании методов инфографики. *Прим. пер.*

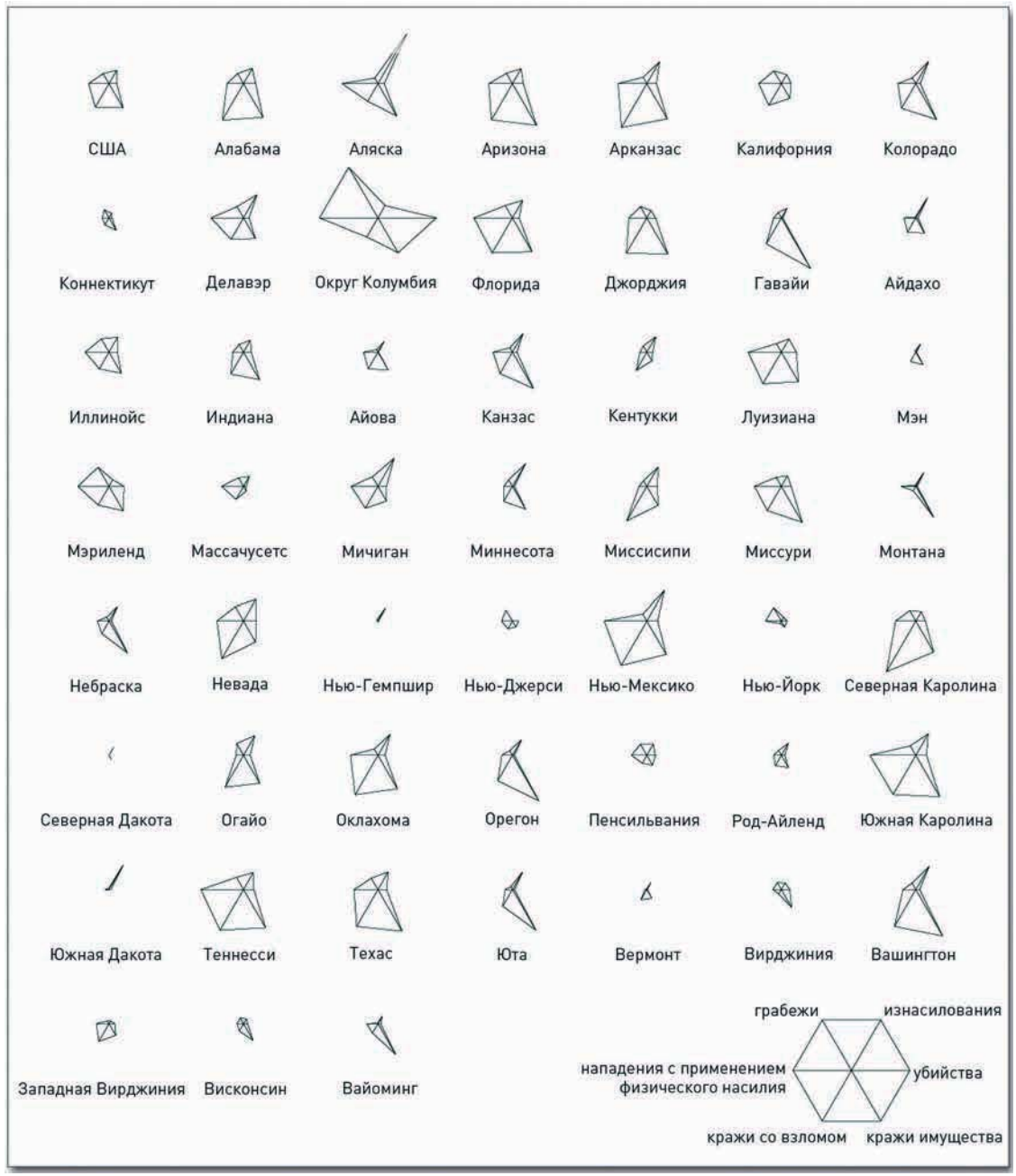


Рис. 7.16. Диаграммы-радары с подписями и ключом

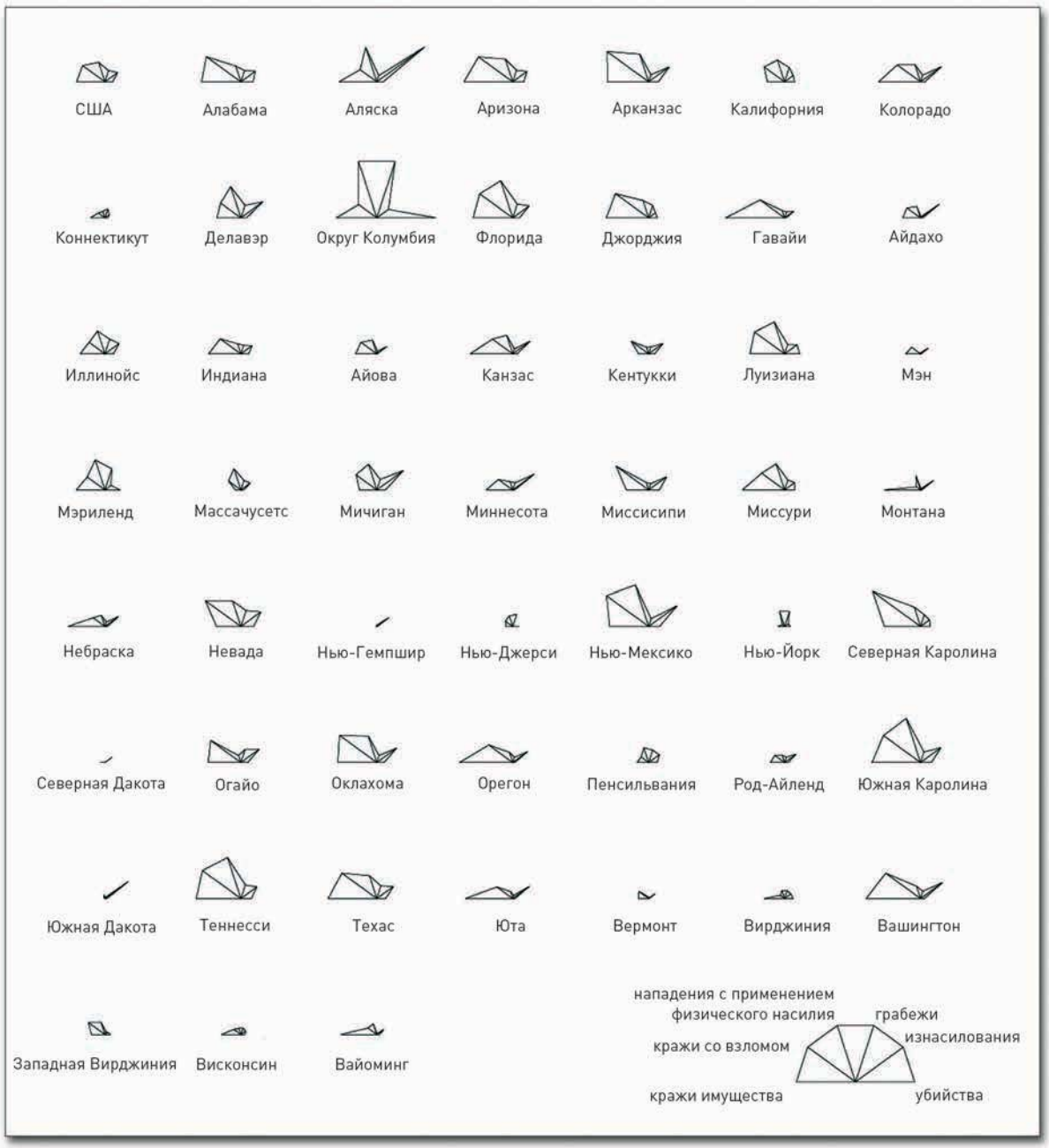


Рис. 7.17. Диаграммы-радары, ограничивающиеся только верхней половиной круга

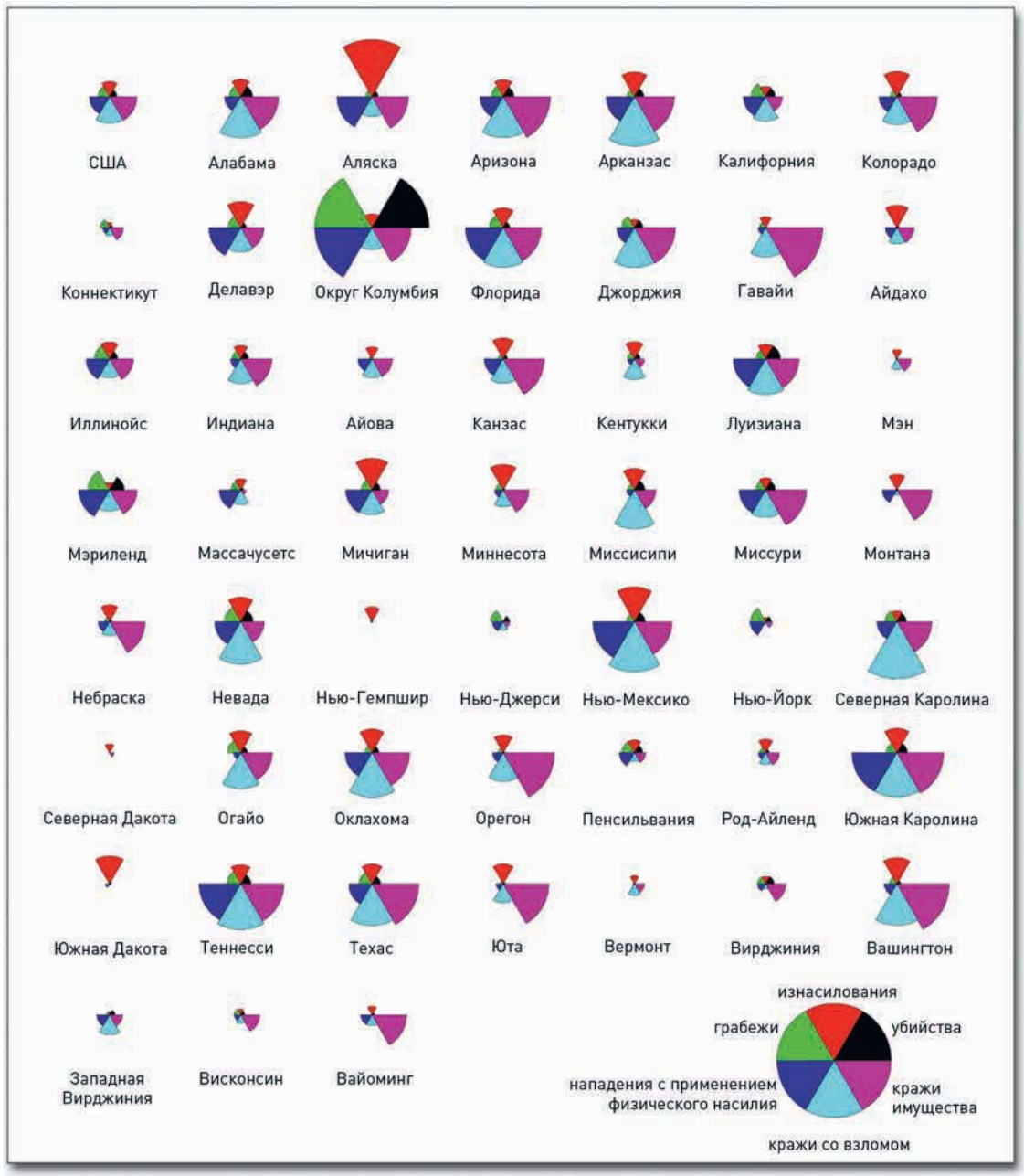


Рис. 7.18. Преступность, представленная с помощью диаграмм Найтингейл

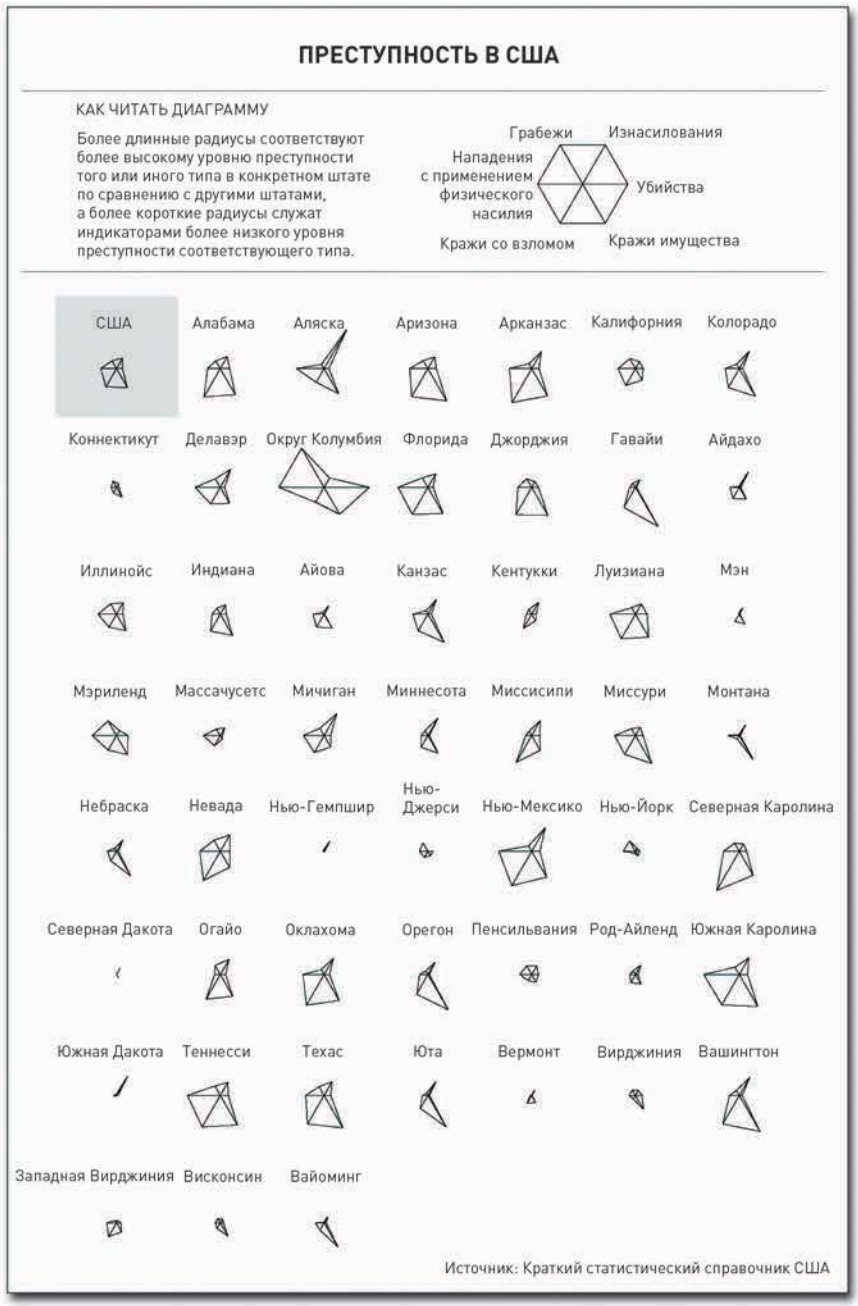


Рис. 7.19. Серия диаграмм-радаров, демонстрирующая уровень преступности по штатам

Двигаясь параллельно

Диаграмма-радар и «лица Чернова» позволяют легко выявлять элементы, отличающиеся от остальных в группе, однако выделить с их помощью отдельные группы или рассмотреть, как различные переменные соотносятся друг с другом, становится уже весьма затруднительно. В этом нам помогут параллельные координаты, изобретенные в 1885 году Морисом д'Оканем (Maurice d'Ocagne).

Как видно на рис. 7.20, в этом типе графики вы размещаете множество осей параллельно друг другу. Верхушка каждой оси представляет максимальное значение переменной, а ее нижняя точка — минимальное. Для каждого элемента слева направо вычерчивается линия, которая двигается вверх и вниз в зависимости от значений конкретного элемента.

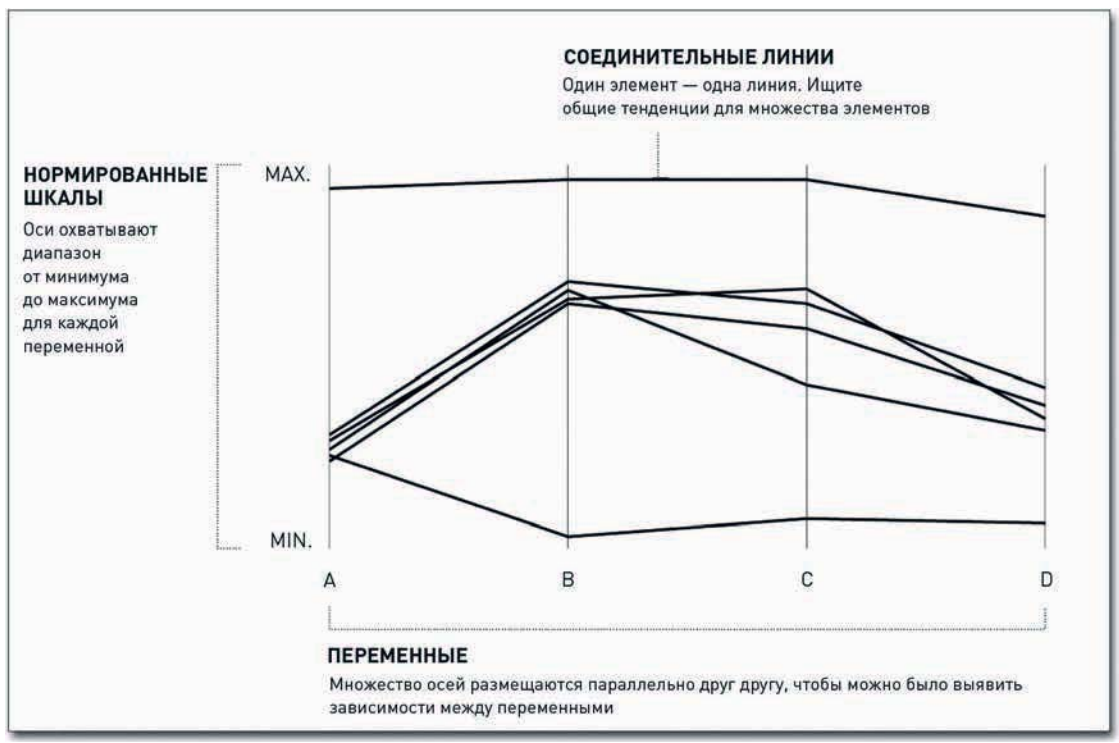


Рис. 7.20. Структура диаграммы с параллельными координатами

Например, представьте себе, что вы создали диаграмму, используя баскетбольные данные из предыдущего параграфа этой главы. Простоты ради вы учитывали только три показателя: очки, подборы мяча и фолы именно в этом порядке. Теперь представьте себе, что у вас есть один игрок, который набрал много очков, однако сделал мало подборов, да и нафолил от души. Линия этого игрока в параллельных координатах начнется с высокой точки, затем опустится вниз и потом снова поднимется.

Если таким образом нанести показатели для большого количества элементов, этот метод поможет вам выявить некие группы и тенденции. В следующем примере мы используем этот тип диаграмм с данными Национального центра статистики в области образования.

СОЗДАЙТЕ ДИАГРАММУ С ПАРАЛЛЕЛЬНЫМИ КООРДИНАТАМИ

При работе с параллельными координатами у вас есть несколько опций для создания интерактивных диаграмм. Если вам понадобится какая-то особая диаграмма, вы можете построить ее в Protovis, а можете просто ввести данные в такой исследовательский инструмент, как GGobi. И в том, и в другом случае вы будете иметь возможность отфильтровать и выделить те моменты в наборе данных, которые вас больше всего интересуют. Однако я тем не менее предпочитаю статичный вариант диаграмм с параллельными координатами, так как они позволяют сравнивать различные фильтры. В интерактивной версии у вас есть только одна диаграмма, и бывает не так уж и просто разобраться в том, что же вы видите перед собой, когда одновременно столько выделений.

▶ Вы можете скачать GGobi бесплатно с `http://ggobi.org`.

Первый шаг вам уже известен. Прежде чем приступить к визуализации данных, вам необходимо добыть сами данные. Загрузите статистику по образованию в R с помощью `read.csv()`.

```
education <- read.csv("http://datasets.flowingdata.com/education.csv",
header=TRUE)
education[1:10,]
```

Здесь у нас семь колонок. В первой даны названия штатов (state) плюс «США» для средних по стране значений. В следующих трех колонках идут средние баллы за анализ текста (reading), математику (math) и эссе (writing) — три основных раздела Школьного оценочного теста (ШОТ). Пятая колонка — это процент выпускников (percentage of graduates), которые реально сдали тест, а в последних двух колонках содержатся данные о численном соотношении учеников и учителей (pupil-to-staff ratio) и об уровне отсева (dropout rate) в средней школе. Нас интересует, прежде всего, связаны ли как-нибудь между собой эти переменные и как они группируются (если вообще группируются). Например, можно ли утверждать, что в штатах с высоким процентом отсева средние баллы, полученные на ШОТ, будут ниже?

Базовый дистрибутив R не предлагает прямых путей к созданию диаграмм с параллельными координатами, но вам поможет пакет lattice. Давайте, загрузите его и установите, если вы этого еще не сделали.

```
library(lattice)
```

Дальше все пойдет как по накатанной. В пакете lattice имеется функция `parallel()`, которой вы сейчас и воспользуетесь.

```
parallel(education)
```

В результате вы получите диаграмму, представленную на рис. 7.21. Она, по сути, довольно-таки бесполезная. По ней разбросано множество каких-то линий, да и переменные идут не слева направо, а сверху вниз. Сейчас это больше похоже на вываливающиеся цветные спагетти.

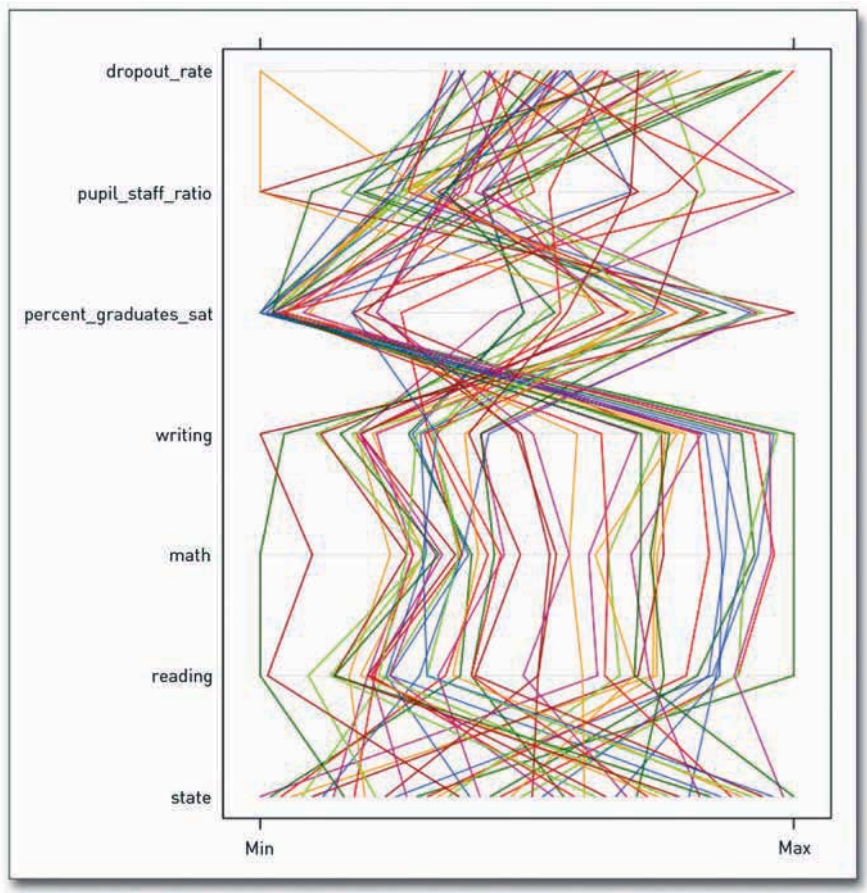


Рис. 7.21. Диаграмма с параллельными координатами, созданная по умолчанию с помощью пакета lattice

Как видоизменить эту диаграмму, чтобы из нее действительно можно было почерпнуть какую-нибудь информацию? Для начала переверните ее набок. Это не то чтобы правило, скорее, личное предпочтение, но параллельные координаты, выстроенные слева направо, выглядят как-то более осмысленно (рис. 7.22)

```
parallel(education, horizontal.axis=FALSE)
```

А еще вам на самом деле не нужна колонка *state* с названиями штатов, во-первых, потому что она категориальная, а во-вторых, потому что у каждого штата свое название. И еще стоит изменить цвет линий на черный. Я всегда выступаю за цвет, но здесь его уж слишком много. Выполните следующую строчку кода — и вы получите результат как на рис. 7.23.

```
parallel(education[,2:7], horizontal.axis=FALSE, col="#000000")
```

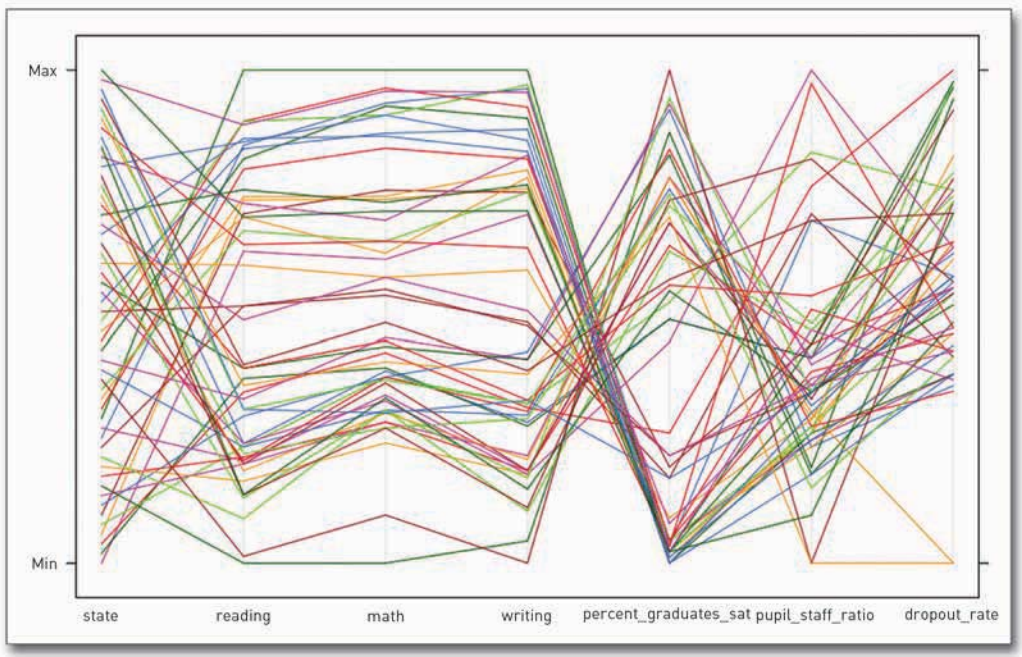


Рис. 7.22. Диаграмма с горизонтальными параллельными координатами

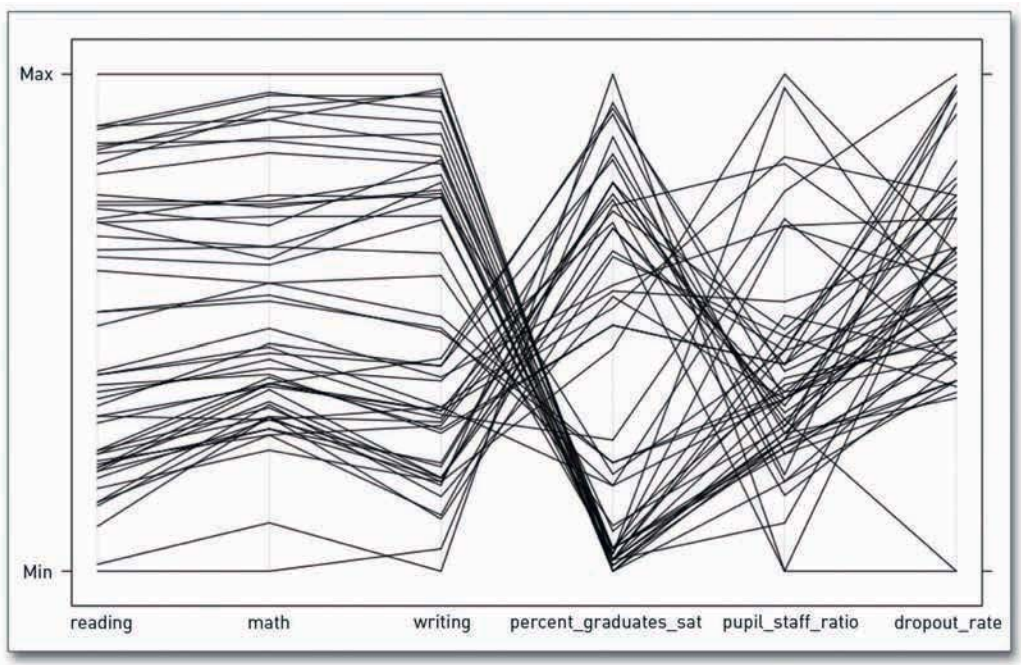


Рис. 7.23. Упрощенный вариант диаграммы с параллельными координатами

Так уже немного лучше. Линии, представляющие анализ текста, математику и чтение, редко пересекаются и идут почти параллельно друг другу. Это означает, что штаты, в которых ученики на экзамене продемонстрировали высокие показатели в части анализа текста, отличились также и в математике и эссе. Аналогично штаты с низкими баллами за анализ текста, как правило, имеют низкие баллы и по двум другим дисциплинам.

Когда вы переключаетесь с баллов на процент выпускников, которые сдали ШОТ, происходит нечто весьма любопытное. Похоже, в штатах с более высокими средними баллами за тест процент выпускников, сдавших этот самый тест, в большинстве случаев оказывается ниже. И наоборот: в штатах с более низкими средними баллами процент сдавших выше. Я не специалист в области образования, но рискну предположить, что в некоторых штатах тест сдают буквально все ученики, а в других — только те, кто собирается затем поступать в колледж. Таким образом, когда к экзамену привлекают также и людей, которых не волнует его результат, средние показатели падают.

Вы можете сделать данный вывод более наглядным, добавив контрастный цвет. Функция `parallel()` позволяет вам контролировать цвета с помощью аргумента *col*. Перед этим вы использовали один-единственный цвет (#000000), но вы можете задать их массу — по одному на каждую строчку данных. Давайте сделаем штаты в 50-м перцентиле баллов за анализ текста черными, а те, что из нижней половины, — серыми. Чтобы найти медианы в данных Национального центра статистики в области образования, используйте функцию `summary()`. Просто введите в консоль `summary(education)`. Это даст вам сводную статистику по всем колонкам, и вы быстро выясните, что медианой (median) для анализа текста (reading) является число 523.

```
      state              reading          math            writing
 Alabama      : 1    Min.   :466.0   Min.   :451.0   Min.   :455.0
 Alaska       : 1    1st Qu.:497.8   1st Qu.:505.8   1st Qu.:490.0
 Arizona      : 1    Median :523.0   Median :525.5   Median :510.0
 Arkansas     : 1    Mean   :533.8   Mean   :538.4   Mean   :520.8
 California   : 1    3rd Qu.:571.2   3rd Qu.:571.2   3rd Qu.:557.5
 Colorado     : 1    Max.   :610.0   Max.   :615.0   Max.   :588.0
 (Other)      :46

 percent_graduates_sat   pupil_staff_ratio      dropout_rate
Min.          : 3.00    Min.      : 4.900   Min.      : -1.000
1st Qu.       : 6.75    1st Qu.   : 6.800   1st Qu.   : 2.950
Median        :34.00    Median    : 7.400   Median    : 3.950
Mean          :37.35    Mean      : 7.729   Mean      : 4.079
3rd Qu.       :66.25    3rd Qu.   : 8.150   3rd Qu.   : 5.300
Max.          :90.00    Max.      :12.100   Max.      : 7.600
```

А теперь давайте пройдемся по всем строчкам данных, проверяя, выше они или ниже этого значения, и задавая им соответствующий цвет. Директива `c()` создает пустой вектор, пополняемый при каждой итерации.

```
reading_colors <- c()
for (i in 1:length(education$state)) {

   if (education$reading[i] > 523) {
      col <- "#000000"
   } else {
      col <- "#cccccc"
   }
   reading_colors <- c(reading_colors, col)
}
```

Затем нам нужно будет передать массив `reading_colors` в `parallel` вместо единственного черного цвета ("#000000"), который там сейчас есть. Таким образом мы получим рис. 7.24. Теперь уже мощный сдвиг вниз стал намного заметнее.

```
parallel(education[,2:7], horizontal.axis=FALSE, col=reading_colors)
```

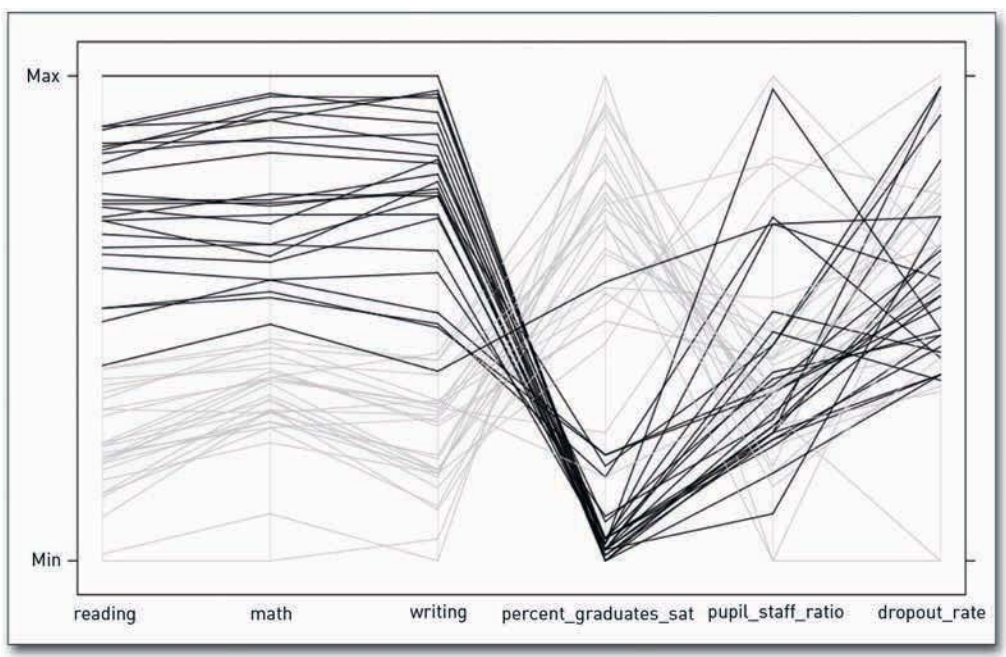


Рис. 7.24. Штаты с самыми высокими баллами за анализ текста (reading) выделены цветом

А как обстоят дела с уровнем отсева? Что будет, если вы проделаете с процентом отсева то же самое, что вы сейчас провернули с баллами за анализ текста, с той лишь разницей, что вместо медианы вы используете третий квартиль? Квартиль составляет 5,3 процента. Сейчас мы снова проитерируем все строчки данных подряд, но на этот раз будем проверять уровень отсева, а не баллы за анализ текста.

```
dropout_colors <- c()
for (i in 1:length(education$state)) {

   if (education$dropout_rate[i] > 5.3) {
      col <- "#000000"
   } else {
      col <- "#cccccc"
   }
   dropout_colors <- c(dropout_colors, col)
}
parallel(education[,2:7], horizontal.axis=FALSE, col=dropout_colors)
```

В итоге должна получиться диаграмма, представленная на рис. 7.25. Она не так однозначна и убедительна, как предыдущая. В ней не наблюдается таких очевидных группирований по всем переменным, как на рис. 7.24.

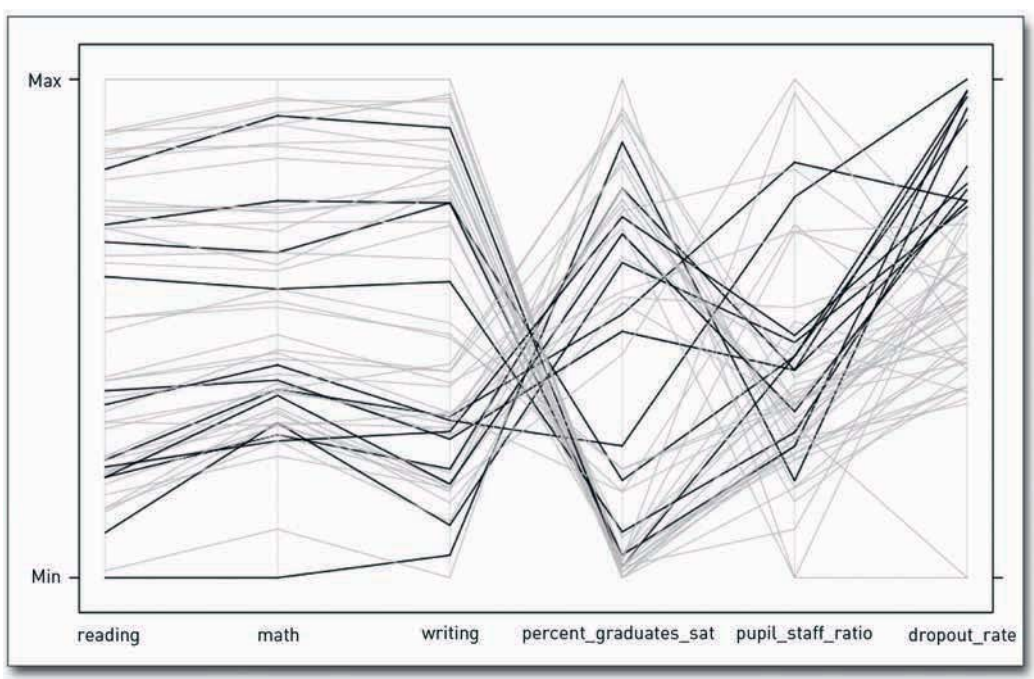


Рис. 7.25. Штаты с самым высоким уровнем отсева (dropout rate) выделены цветом

Вы можете продолжить изучение данных и самостоятельно. Вернитесь к рис. 7.24 и подредактируйте его. Неплохо будет сделать подписи более читабельными. Возможно, стоит вместо серой шкалы использовать какие-нибудь цвета. И как насчет короткой аннотации о том, почему верхние 50 процентов штатов выделены? В итоге вы получите рис. 7.26.

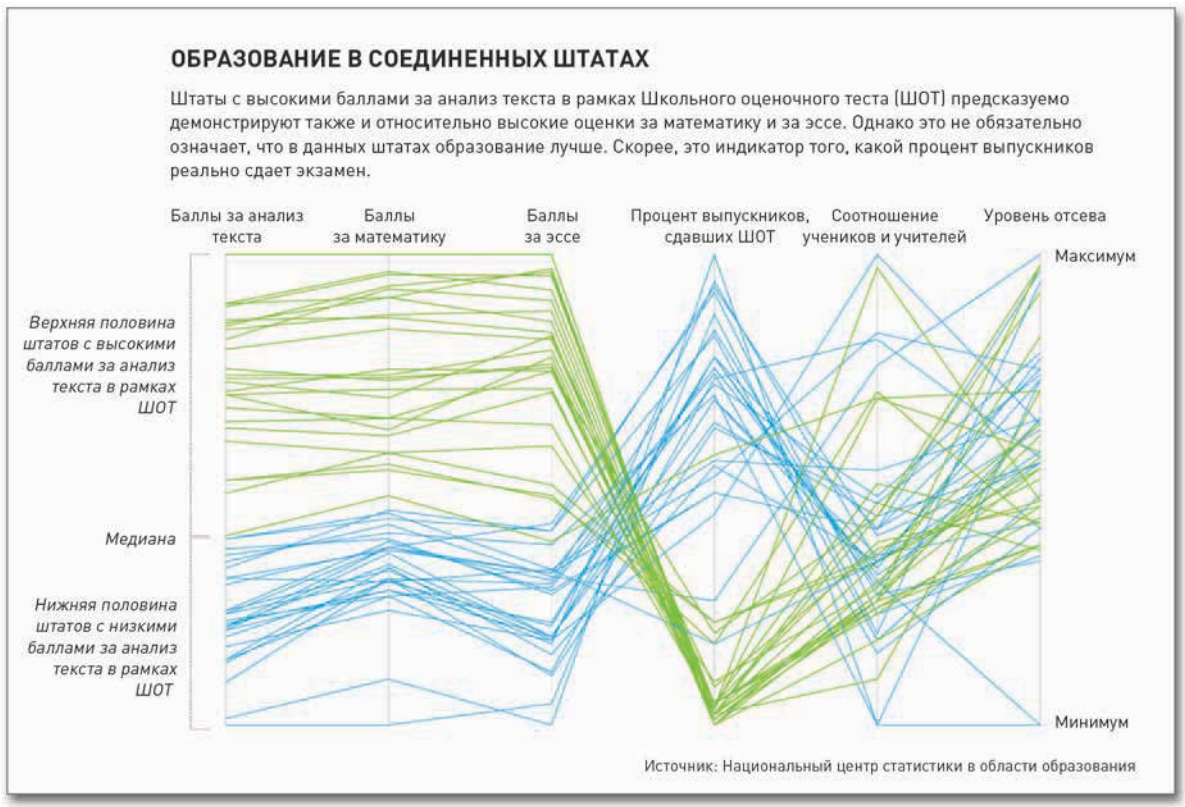


Рис. 7.26. Диаграмма с параллельными координатами, отражающая результаты ШОТ

Сокращение размерности

Когда вы используете «лица Чернова» или параллельные координаты, ваша основная задача сводится к сокращению. Вы хотите вычленить в наборе данных или в совокупности несколько основных групп. Трудность заключается в том, что вы не всегда знаете, с какого места начать разглядывать лица или соединительные линии. А потому было бы замечательно, если бы вы могли сгруппировать объекты на основе нескольких критериев. В этом и состоит одна из целей многомерного шкалирования. Вы все учитываете, а затем более схожие элементы размещаете на диаграмме ближе друг к другу.

▶ Чтобы подробнее ознакомиться с данным методом, поищите дополнительную информацию о многомерном шкалировании или об анализе главных компонент.

Этой теме посвящены целые книги, так что, как вы догадываетесь, объяснение может показаться вам чересчур мудреным, но простоты ради я постараюсь не особо углубляться и математическую сторону вопроса оставлю на другой раз. При этом я замечу, что многомерное шкалирование было одной из первых концепций, которые мы изучали в магистратуре. Лежащий в ее основе принцип стоит того, чтобы его изучить, если вы собираетесь всерьез заниматься такими делами.

Представьте себе, что вы находитесь в пустой квадратной комнате и что помимо вас в ней есть еще два человека. Ваша задача — определить, где именно им следует располагаться согласно их росту. Чем более они похожи по этому показателю, тем ближе должны стоять, и, соответственно, чем существенней разница в их росте, тем дальше они должны оказаться друг от друга. Итак, допустим, что один из них действительно ну очень высокий, а другой — очень низкий. Где они должны стоять? Эти два антипода должны стоять в противоположных углах комнаты.

А теперь в комнату входит третий человек. Он среднего роста. Придерживаясь принятой схемы, новый человек должен встать в центре комнаты, ровно посередине между первыми двумя. Его рост в одинаковой степени отличается от роста и высокого, и низкого, а потому он должен и находиться на равном расстоянии от них. При этом долговязый и коротышка будут по-прежнему стоять на максимальном расстоянии друг от друга.

Хорошо. А теперь введем еще одну переменную: вес. Вам известен и рост, и вес всех троих. Коротышка и средний по росту человек на самом деле весят одинаково, а долговязый, допустим, на треть тяжелее их. Как вы теперь на основе двух показателей — роста и веса — расставите людей в комнате? Наверное, первых двоих (высокого и низкого) вы оставите на прежнем месте — в противоположных углах комнаты, а третьего (среднего роста) вам придется сдвинуть поближе к коротышке, потому что у них одинаковый вес.

Теперь понимаете, что происходит? Чем более схожи между собой два человека, тем ближе они будут стоять друг к другу. В этом простом примере у вас есть только три человека и только две переменные, так что нетрудно проделать работу и вручную. Но представьте себе, что у вас есть 50 человек и вам необходимо их расставить по местам в комнате на основе, скажем, пяти критериев. Задачка посложнее, не правда ли? Вот для этого и существует многомерное шкалирование.

Использование многомерного шкалирования

Метод многомерного шкалирования проще понять на конкретном примере, а потому перейдем сразу к нему. Вернитесь к данным из области образования, а если вы их еще не загрузили в R, тогда сделайте это сейчас.

```
education <-
   read.csv("http://datasets.flowingdata.com/education.csv",
   header=TRUE)
```

Помните, что для каждого штата, включая округ Колумбия, имеется отдельная строка, и еще есть строчки со средними по США данными. По каждому штату есть шесть переменных: баллы за анализ текста, математику и эссе плюс процент выпускников, сдающих экзамен, соотношение ученики/учителя и уровень отсева.

Все обстоит точно так же, как и в метафоре с комнатой, но вместо квадратной комнаты у вас есть квадратная диаграмма, вместо людей — штаты, а вместо роста и веса — количественные показатели в области образования. Цель же остается прежней: вы хотите разместить штаты по осям X и Y диаграммы таким образом, чтобы схожие между собой штаты оказались ближе друг к другу.

Первым делом вам нужно посчитать, на каком расстоянии друг от друга должны располагаться штаты. Примените функцию `dist()` — именно для этого она и предназначена. Вы будете использовать только шесть колонок — со второй по седьмую, так как первая — это названия штатов, а вы и так знаете, что они все разные.

```
ed.dis <- dist(education[,2:7])
```

Если сейчас вы наберете в консоли **`ed.dis`**, вы увидите серию матриц. Каждая ячейка будет содержать информацию, как далеко (на каком евклидовом пиксельном расстоянии) один штат должен отстоять от другого. Например, значение во второй строке / втором столбце представляет расстояние, на которое Алабама должна отстоять от Аляски. Но в данный момент эти элементы не так важны. Гораздо большее значение имеют относительные разности.

Как можно перенести эту матрицу 51×51 в *x*-*y*-диаграмму? Сейчас — никак, пока вы не получите координаты *x* и *y* каждого штата. Для этого существует функция `cmdscale()`. В нее вы вводите матрицу расстояний, а на выходе получаете набор точек, разности между которыми окажутся примерно такими же, как было задано в матрице.

```
ed.mds <- cmdscale(ed.dis)
```

Наберите в консоли **`ed.mds`**, и вы увидите, что у вас уже есть координаты *x* и *y* для каждой строки данных. Сохраните их в переменных *x* и *y*, а затем перекиньте их в `plot()`. Вот теперь уже вы получите нечто, похожее на рис. 7.27.

```
x <- ed.mds[,1]
y <- ed.mds[,2]
plot(x,y)
```

Получилось неплохо. Каждая точка представляет собой отдельный штат. Однако есть одна загвоздка: вы не знаете, какой штат у вас где. Вам нужны подписи, а потому, как и в прошлый раз, воспользуйтесь функцией `text()`, чтобы вставить названия штатов в тех местах, где сейчас точки, как это показано на рис. 7.28.

```
plot(x, y, type="n")
text(x, y, labels=education$state)
```

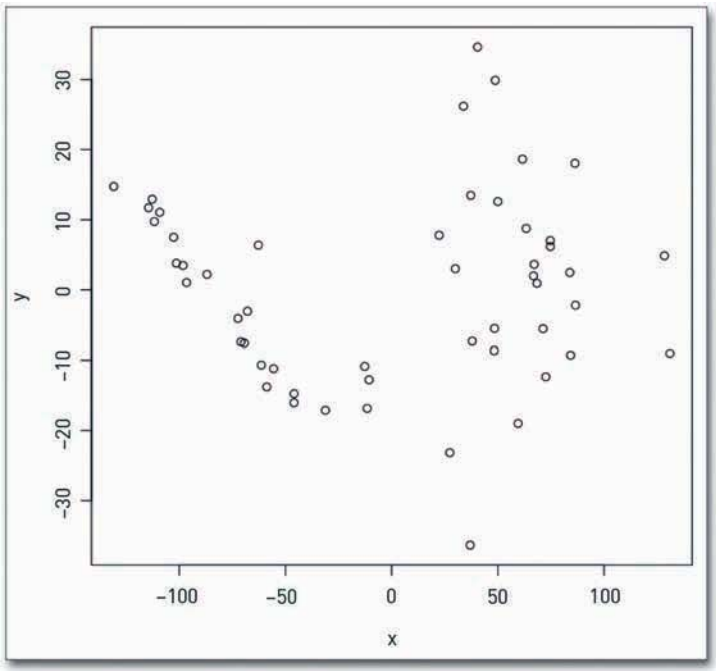


Рис. 7.27. Точечная диаграмма, демонстрирующая результаты многомерного шкалирования

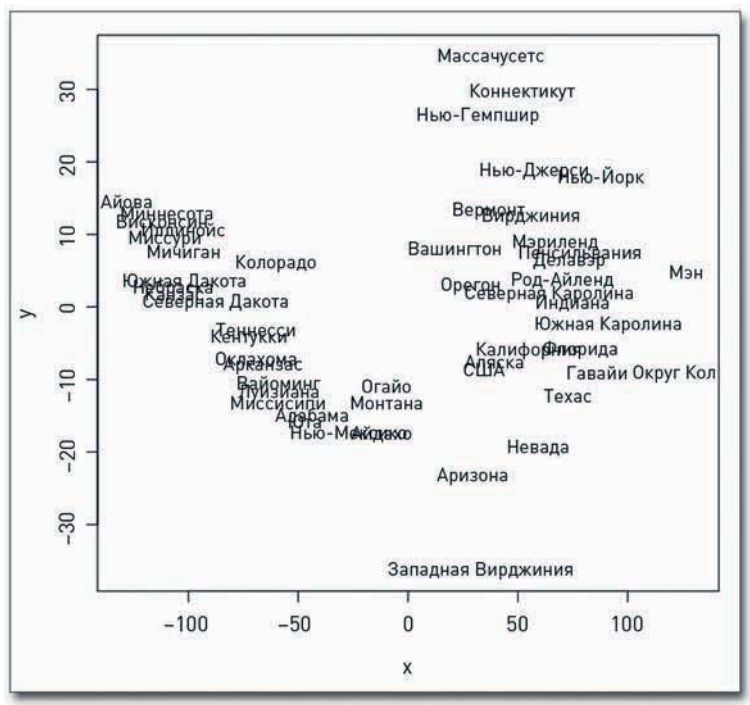


Рис. 7.28. Использование названий штатов вместо точек позволяет увидеть, какое место занимает тот или иной штат

Так уже лучше. Вы видите, что возникла пара кластеров, один слева, а другой справа. Соединенные Штаты находятся в нижней части правого кластера, ближе к середине, что представляется вполне логичным. На данном этапе вам самостоятельно придется определяться с тем, что означают кластеры, но в любом случае это уже неплохая отправная точка для начала погружения в исследование данных.

Вы могли бы, например, окрасить штаты по проценту отсева (с помощью `dropout_colors`), как вы уже делали, создавая диаграмму с параллельными координатами. В итоге вы получите нечто похожее на рис. 7.29. Не то чтобы это много о чем говорило, но по крайней мере диаграмма подтверждает то, что вы видели на рис. 7.25.

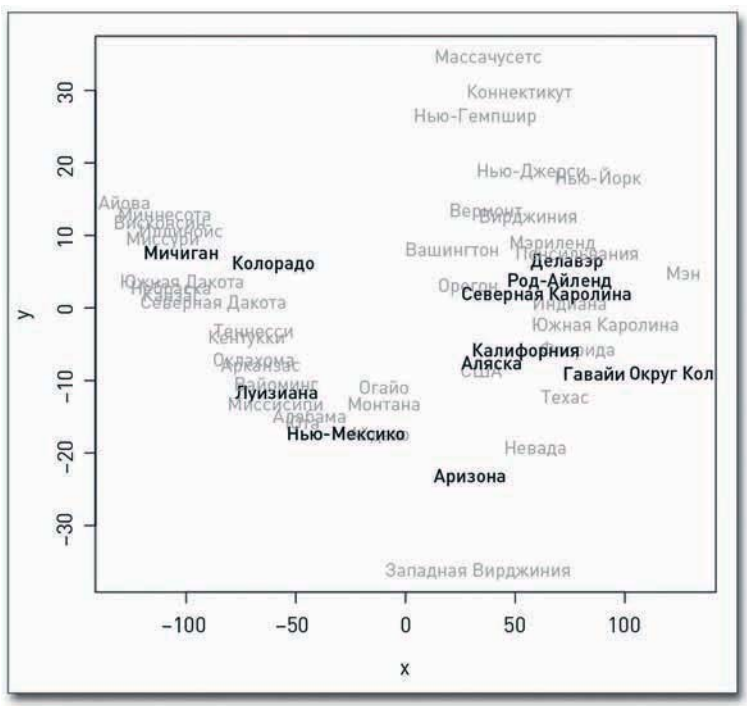


Рис. 7.29. Штаты окрашены в зависимости от уровня отсева

А как насчет того, чтобы окрасить штаты в соответствии с баллами за анализ текста? Да, вы вполне можете сделать и это тоже, как показано на рис. 7.30. И вот теперь уже, кажется, начинает просматриваться четкий паттерн. Похоже, слева у нас высокие баллы, а справа — низкие, правильно? Тогда почему Вашингтон находится там, где он находится? Подумайте над этим.

Если вы хотите подойти к делу с воображением, то можете опробовать так называемый модельный метод кластеризации. Я не буду пускаться в подробное описание, просто покажу вам, как его можно применять, а вы поверьте мне на слово, что здесь нет никакой магии. Это

чистая математика. По сути, мы используем пакет `mclust`, чтобы выявить кластеры в диаграмме многомерного шкалирования. Если `mclust` у вас еще не установлен, сделайте это сейчас. А затем запустите приведенный ниже код — и вы получите диаграммы, представленные на рис. 7.31.

```
library(mclust)
ed.mclust <- Mclust(ed.mds)
plot(ed.mclust, data=ed.mds)
```

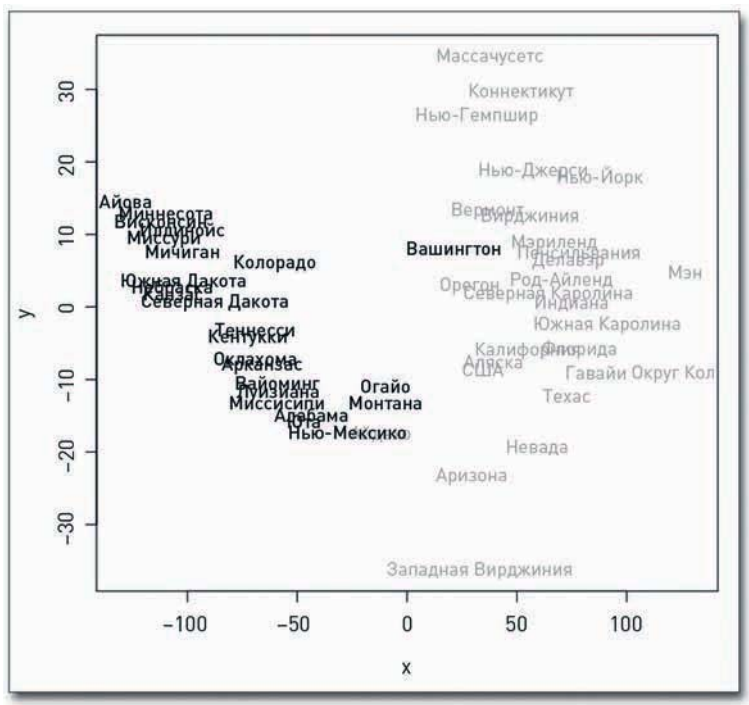


Рис. 7.30. Штаты окрашены в соответствии с баллами за анализ текста

На первом графике вверху слева показаны результаты выполнения алгоритма для нахождения идеального количества кластеров в наборе данных. На остальных трех показаны сами кластеры. Впечатляет, не правда ли? Вы получили те же два кластера, только более четко определенные, демонстрирующие штаты с хорошими и с плохими показателями.

Раньше на этом этапе я предлагал вам перенести PDF-файл в программу Illustrator и нанести последние штрихи, но сейчас я не очень-то уверен, что имеет смысл выставлять эти графики на всеобщее обозрение. Они слишком абстрактные для неподготовленной аудитории. Человек другого, не технического склада ума просто не поймет, что здесь происходит. Эти графики хороши для исследовательских целей. Однако если вы все же захотите их опубликовать, можете применить к ним все стандартные принципы дизайна. Подумайте, что именно вам понадобится, чтобы рассказать внятную историю, и удалите все остальное.

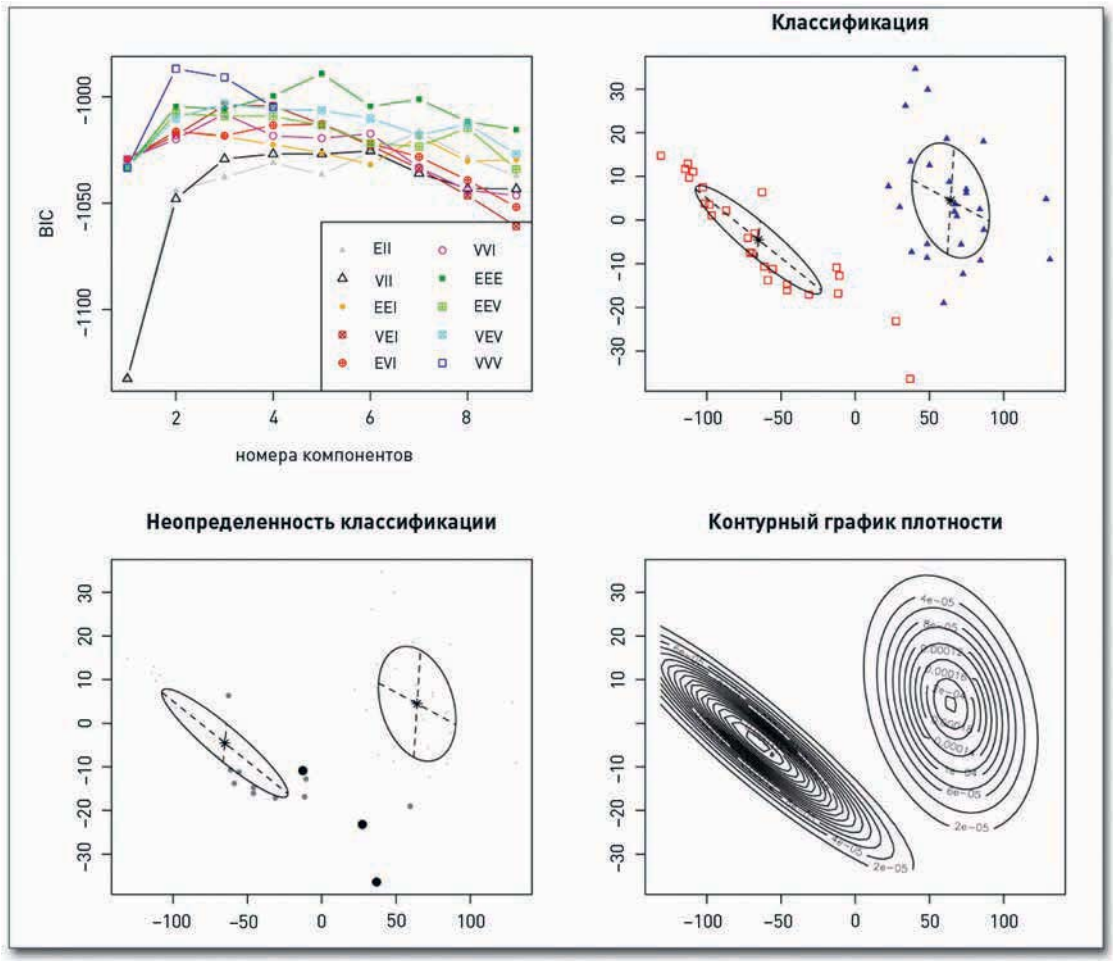


Рис. 7.31. Результаты модельного метода кластеризации

Поиск выбросов

Помимо того, как группируются отдельные элементы данных, вас должно интересовать и то, как они *не* группируются. Иными словами, очень часто в наборах данных обнаруживаются такие элементы, которые выбиваются из общего ряда и которые мы называем, как вы сами догадались, *выбросами*. Это точки, отличающиеся от всей совокупности. Иногда они могут оказаться самой интересной частью вашей истории, в других случаях — просто досадными опечатками с пропущенными нулями. В любом случае вам необходимо проверить их, чтобы

выяснить, что же там происходит. Вы же не хотите построить сложнейшую диаграмму на основании конкретного выброса, а впоследствии узнать от какого-нибудь въедливого читателя, что ваше произведение лишено смысла и весь тяжкий труд был проделан впустую?

Для выявления выбросов существуют специально разработанные типы графиков, но по своему опыту могу сказать: ничто не сравнится с обыкновенными графиками и здравым смыслом. Изучите контекст данных, если не уверены в чем-то, посоветуйтесь с экспертами. А когда вы определитесь с выбросами, можете выделить их для читателей, применяя графические приемы, которые уже опробовали: используя различные цвета, вставляя указатели и изменяя толщину границ.

Давайте рассмотрим простой пример. На рис. 7.32 показан вариант графика временны́х рядов с данными о погоде, извлеченными с сайта Weather Underground (таким же способом, как вы это делали в главе 2, «Как обращаться с данными») и охватывающими период с 1980 по 2005 годы. Как вы могли ожидать, на нем видны определенные сезонные циклы, но что там происходит в середине? Все выглядит необычайно гладко, в то время как в остальной части данных явственно присутствуют «шумы». Конечно, это не повод для особого расстройства, но если бы вам вдруг понадобилось выстроить модель погоды на основе этих данных, вы, возможно, захотели бы выяснить, что именно оценивалось и каковы настоящие данные.

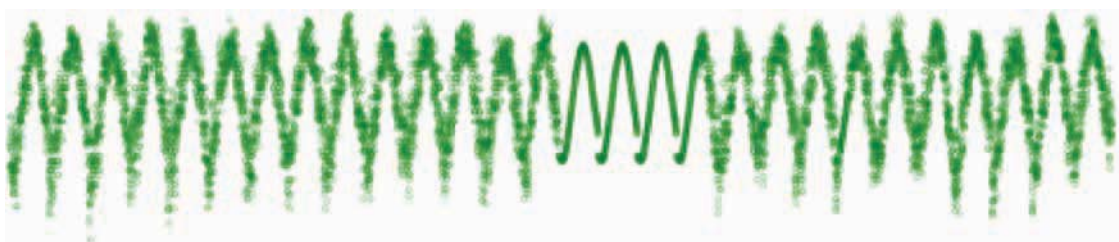


Рис. 7.32. Оценочные данные о погоде с Weather Underground

Аналогичным образом, разглядывая диаграммы-радары, которые вы создали для наглядного представления данных о преступности, вы можете заметить, что округ Колумбия выделяется среди остальных штатов. С таким же успехом вы бы это заметили и на простой столбцовой диаграмме, как показано на рис. 7.33. Но насколько честно сравнивать Вашингтон, округ Колумбия, с другими штатами, учитывая то, что данная административная единица является все же, скорее, городом? Это уж вам решать.

А как быть с подсчетом подписчиков, о котором мы говорили в главе 3, «Выбор инструментов для визуализации данных», и который представлен на рис. 7.34? Что делать с этим огромным провалом в середине, из-за которого создается впечатление, что FlowingData потерял более половины своих читателей?

Вы можете взглянуть также на распределение в целом, которое представлено на гистограмме на рис. 7.35. Результаты почти всех подсчетов у нас собрались справа и только два результата оказались слева, а в середине — пустота.

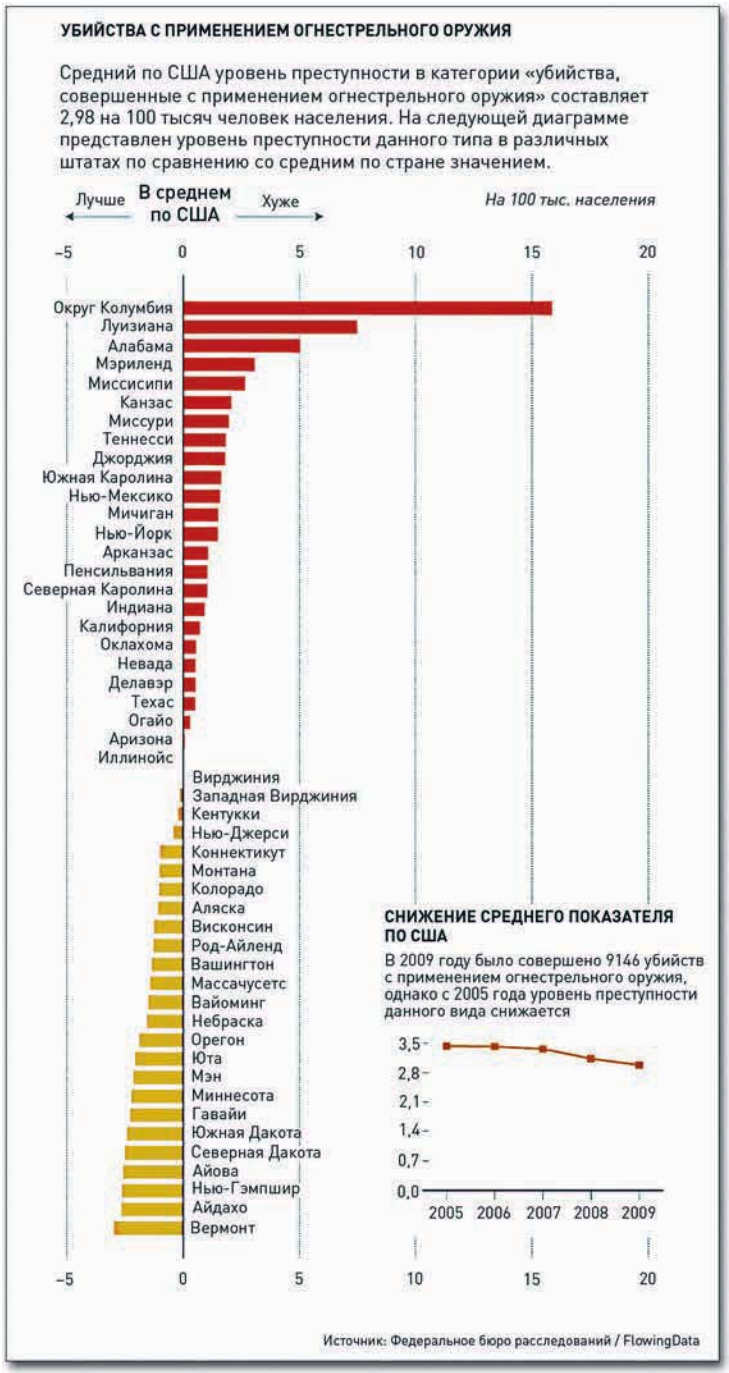


Рис. 7.33. Убийства с применением огнестрельного оружия в США

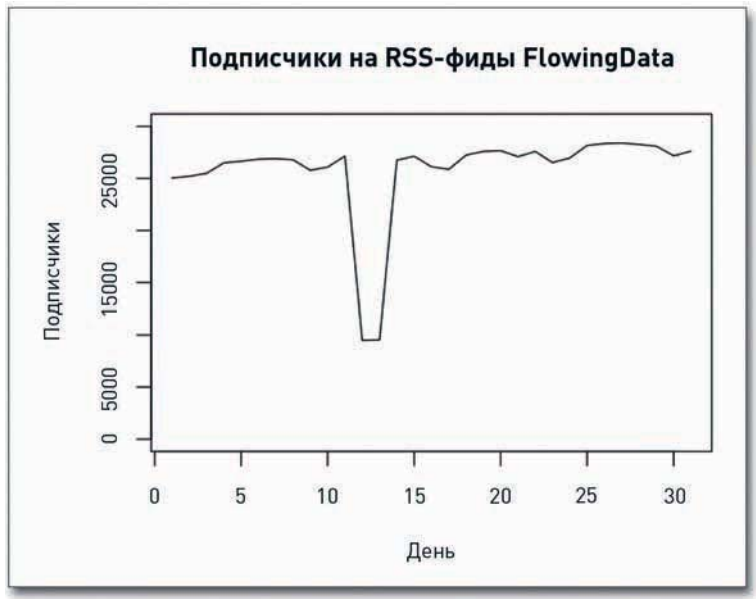


Рис. 7.34. Количество подписчиков FlowingData во времени

Рис. 7.35. Гистограмма, демонстрирующая распределение подсчетов подписчиков

Для большей конкретики вы можете воспользоваться коробчатой диаграммой, демонстрирующей квартили распределения. Коробчатые диаграммы, создаваемые в R с помощью функции `boxplot()`, дают возможность автоматически выделять значения, которые

более чем в полтора раза превышают верхний квартиль или же не добирают до нижнего (рис. 7.36).

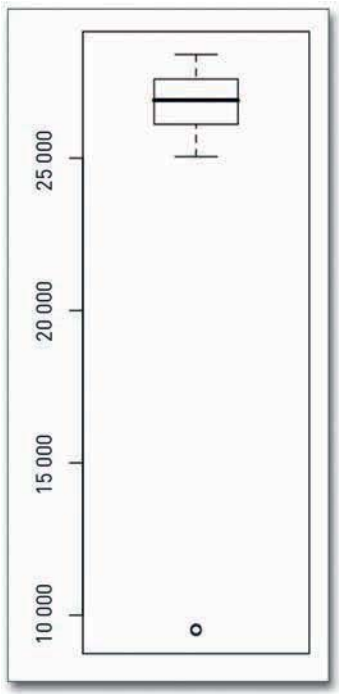


Рис. 7.36. Коробчатая диаграмма, демонстрирующая распределение подсчетов подписчиков

▶ Квартиль — это одна из трех точек в наборе данных, которые делят его на четыре части. Средний квартиль — это медиана, или центр набора данных; верхний квартиль фиксирует точку, относительно которой 25 процентов данных имеют большее значение; нижний квартиль отделяет меньшие 25 процентов.

Если бы я имел небольшое количество подписчиков, исчисляемое однозначным числом, тогда, конечно, подобный значительный (в процентном измерении) спад был бы возможен. Однако с трудом верится, что я сказал нечто настолько оскорбительное, что заставило десятки тысяч читателей отказаться от подписки (а затем через пару дней возобновить ее). Гораздо вероятнее, что Feedburner, веб-сервис, регулирующий потоки фидов, услугами которого я пользуюсь, допустил ошибку в отчетах.

Приведенные в качестве примеров выбросы вполне очевидны, так как мы все же располагаем определенными сведениями о них. Но если бы это был незнакомый набор данных, все могло бы оказаться не столь очевидным. Когда оказываешься в подобной ситуации, полезно обратиться непосредственно к источнику данных и к тому, кто за них отвечает. Человек или организация, курирующие статистику, обычно бывают рады тому, что кто-то нашел применение их данным, и всегда готовы помочь советом. Однако если вам не удастся найти дополнительную информацию, вы по крайней мере сможете сказать, что сделали все возможное, чтобы ее разыскать, и вставить примечание о наличии неопределенности, когда будете пояснять свой график или диаграмму.

Закругляясь

Для начинающих одна из самых больших сложностей в разработке инфографики состоит в том, чтобы определить, с чего следует начать. Перед вами целая гора данных и ни одной подсказки, о чем они и чего от них можно ожидать. Как правило, начинать следует с вопроса, касающегося этих данных, и нахождения на него ответа, но что делать, если вы не знаете даже, о чем спрашивать? Методы, описанные в настоящей главе, как раз и должны вам в этом помочь. Они дадут вам возможность охватить все данные одним взглядом, в результате чего вам будет легче решить, какую их часть затем стоит рассмотреть подробнее.

Но не останавливайтесь на этом. Используйте новообретенные навыки и умения как отправную точку в изучении того, что покажется особенно интересным. Полученных в этой и в предыдущих главах знаний должно уже быть достаточно, чтобы вы могли разобраться в ситуации независимо от того, с каким именно типом данных вам довелось работать. Ну, за исключением разве что одного типа данных, которому и будет посвящена следующая глава, а именно: пространственные данные. Приготовьтесь нарисовать несколько карт.

Визуализация пространственных отношений

Карты — это форма визуализации, которая обладает дополнительным преимуществом: она невероятно интуитивна. Даже ребенком я умел читать карты. Помню, как сидел на пассажирском сиденье дедушкиной машины и громко раздавал указания, изучая фантастически огромную карту, развернутую у меня на коленях. Сегодня направление движения мне диктует австралийская дамочка с хоть и несколько механическим, но тем не менее успокаивающим голосом из маленькой коробочки на приборной панели. В любом случае карты — замечательный способ разобраться в массиве данных. Они представляют собой уменьшенную в масштабах версию реального мира, и они буквально повсюду. В этой главе вы углубитесь в изучение различных наборов данных, выискивая паттерны во времени и в пространстве. Вы создадите несколько базовых карт в R, а затем перейдете к более продвинутым методам составления карт с помощью Python и SVG. И в завершение освоите создание интерактивных и анимированных карт с ActionScript и Flash.

Что искать

Карты вы читаете точно так же, как и статистические диаграммы и графики. Когда вы смотрите в конкретное место на карте, вы все равно выискиваете некие скопления чего бы то ни было в конкретном регионе или, например, сравниваете один регион с другими. Разница состоит лишь в том, что вместо *x*- и *y*-координат здесь вы имеете дело с широтой и долготой. На карте координаты связаны друг с другом точно так же, как один город — с другим. Точка A отстоит от точки B на определенное количество километров, и, чтобы добраться от одной до другой, требуется определенное количество времени. А вот на точечной диаграмме расстояния абстрактные и, как правило, не имеют единиц измерения.

Однако это «небольшое» отличие сопряжено со значительным количеством тонкостей, которые следует учитывать при работе с картами и картограммами. Вот почему в отделе графики газеты New York Times есть специальное подразделение людей, которые работают исключительно с картами. Ведь в каждом случае необходимо убедиться в том, что местоположение определено верно, что цвета подобраны логично, что подписи не закрывают собой важные точки на карте и что проекция выбрана правильно.

Эта глава охватывает только часть основных положений при работе с картами. Они помогут вам продвинуться довольно далеко в умении раскрывать истории в имеющихся данных, однако не забывайте, что это лишь базовый уровень и вам есть к чему еще стремиться.

А когда вы добавите время, все станет еще интереснее. Одна карта — это один срез времени, но вы можете представить множество срезов времени с помощью нескольких карт. А еще вы можете анимировать изменения, чтобы показать, скажем, рост (или спад) деловой активности в определенном географическом регионе. Таким образом, всплески на конкретных территориях станут очевидными, и если карта будет еще и интерактивной, то читатели смогут легко сконцентрироваться на интересующей их области, чтобы увидеть, какие изменения и как именно на ней происходили. Со столбцовыми и точечными диаграммами такого эффекта не добиться — ведь с картами данные могут в одно мгновение стать чем-то очень личным.

Отдельные местоположения

Перечень местоположений — это самый простой тип пространственных данных, с которыми вам может довестись работать. У вас есть широта и долгота некоей кучки мест, и вам нужно нанести их на карту. Может, вы собираетесь показать, где происходили какие-нибудь события, скажем, преступления, или хотите найти области, где эти пункты сгруппированы. Сделать такое нетрудно, методов существует множество.

В Сети самый распространенный способ определить на карте некую точку — это воспользоваться Google или Bing Maps. Через их API вы получаете интерактивную карту, которую

можете изменять в масштабе и прокручивать с помощью всего нескольких строк JavaScript. Там же, в Сети, вы найдете тонны обучающих руководств, так что передаю это дело в ваши руки.

Однако у этих API есть и свои недостатки. Вы можете лишь адаптировать тамошние карты под свои потребности, но в конце концов по ним почти всегда будет видно, что карты взяты из Google или Microsoft. Я не хочу сказать, что они некрасивые, но когда вы разрабатываете приложение или создаете графику для печати, как правило, желательно, чтобы карта органично вписывалась в общий дизайн вашего документа. Конечно, если постараться, во многих случаях удается найти способ обойти эти препятствия, однако оно того не стоит, если подобного результата можно добиться с бóльшим успехом, применяя иной инструмент.

ПРИМЕЧАНИЕ

Google и Microsoft предлагают прекрасные обучающие руководства, посвященные именно их картографическим API, так что не забудьте свериться с ними, если захотите воспользоваться их маппинг-функционалом.

Нахождение широты и долготы

Прежде чем приступить к созданию карт, подумайте о данных, которые у вас есть и которые вам необходимы. Если вы не располагаете нужными данными, вам и визуализировать будет нечего, верно? В большинстве самых практичных приложений вам необходимо располагать широтой и долготой точек, которые вы собираетесь нанести на карту, а данные чаще всего приходят отнюдь не в таком виде — как правило, вы получаете просто перечень адресов.

И как бы вам того ни хотелось, вы не можете просто ввести названия улиц и номера почтовых индексов и получить симпатичную карту. Вам сначала необходимо найти широту и долготу нужных точек, а для этого вам придется обратиться к *геокодированию*. В общих чертах процесс выглядит так: вы берете адрес и отдаете его в сервис, который обращается к своей базе данных, выискивая в ней нечто сопоставимое, после чего выдает вам широту и долготу, которые, по мнению этого сервиса, описывают местоположение вашего адреса на земном шаре.

Подобных сервисов существует множество. Если вам необходимо геокодировать буквально несколько адресов, проще всего будет зайти на нужный веб-сайт и ввести их вручную. Geocoder.us — хороший бесплатный вариант для этого. Если вам не нужно, чтобы местоположение было определено с абсолютной точностью, тогда вы можете попробовать Google Maps Latitude Longitude Popup. Этот простой интерфейс для Google Maps, разработанный Пьером Гориссеном (Pierre Gorissen), определяет широту и долготу любой точки на карте, в которую вы ткнете указателем мыши.

Однако если вам необходимо геокодировать множество пространственных точек, тогда вам лучше сделать это программными средствами. Вам не стоит тратить время на бесконечное копирование и вставку. И Google, и Yahoo, и Geocoder.us, и Mediawiki — каждый из этих сервисов предлагает свои API для геокодирования. А Geopy — геокодирующий инструмент для работы с Python — объединяет их все в один пакет.

Зайдите на страницу проекта Geopy. Там вы получите инструкции, как установить у себя этот пакет, а также сможете ознакомиться с множеством примеров того, как с ним работать. А сейчас мы перейдем к практике (предполагается, что вы уже установили данный пакет у себя на компьютере).

ПОЛЕЗНЫЕ РЕСУРСЫ ДЛЯ ДЛЯ ГЕОКОДИРОВАНИЯ:

- Geocoder.us, `http://geocoder.us` — предоставляет удобный интерфейс, в который можно вставить адрес и получить широту и долготу. Предлагает также и API.
- Latitude Longitude Popup, `www.gorissen.info/Pierre/maps/` — мэшап, объединяющий картографические данные Google Maps. Стоит кликнуть по точке на карте, и приложение выдаст вам ее широту и долготу.
- Geopy, `http://code.google.com/p/geopy/` — геокодирующий инструментарий для работы с Python. Объединяет множество API для геокодирования в один пакет.

После того как вы инсталлировали Geopy, скачайте данные с `http://book.flowingdata.com/ch08/geocode/costcos-limited.csv`. Это CSV-файл, содержащий адреса всех магазинов-складов сети Costco на территории США, но в нем нет их координат с широтой и долготой. Найти их — уже ваша задача.

Сперва вам нужно создать новый файл и сохранить его как `geocode-locations.py`, а также обычным образом импортировать пакеты, необходимые для остальной части скрипта.

ПРИМЕЧАНИЕ

Зайдите на `http://code.google.com/apis/maps/signup.html`, чтобы зарегистрироваться и получить бесплатно API-ключ для Google Maps API. Это делается очень просто и отнимает всего пару минут.

```
from geopy import geocoders
import csv
```

А еще вам нужен API-ключ для каждого сервиса, который вы собираетесь использовать. Для целей данного примера вам необходим такой ключ от Google.

Сохраните ваш API-ключ в переменной под названием `g_api_key`, а затем воспользуйтесь им, когда будете обращаться к геокодеру.

```
g_api_key = 'INSERT_YOUR_API_KEY_HERE'
g = geocoders.Google(g_api_key)
```

Загрузите файл `costcos-limited.csv` с данными и запустите итерацию. Таким образом вы будете создавать полный адрес для каждой строчки, а затем передавать его для геокодирования.

```
costcos = csv.reader(open('costcos-limited.csv'), delimiter=',')
next(costcos) # Skip header

# Print header
print "Address,City,State,Zip Code,Latitude,Longitude"
for row in costcos:

    full_addy = row[1] + "," + row[2] + "," + row[3] + "," + row[4]
    place, (lat, lng) = list(g.geocode(full_addy, exactly_one=False))[0]
    print full_addy + "," + str(lat) + "," + str(lng)
```

Вот и все. Запустите скрипт и сохраните полученный результат как `costcos-geocoded.csv`. Ниже показано, как будут выглядеть первые несколько строк данных.

```
Address,City,State,Zip Code,Latitude,Longitude
1205 N. Memorial Parkway,Huntsville,Alabama,35801-5930,34.7430949,
-86.6009553
3650 Galleria Circle,Hoover,Alabama,35244-2346,33.377649,-86.81242
8251 Eastchase Parkway,Montgomery,Alabama,36117,32.363889,-86.150884
5225 Commercial Boulevard,Juneau,Alaska,99801-7210,58.3592,-134.483
330 West Dimond Blvd,Anchorage,Alaska,99515-1950,61.143266,-149.884217
…
```

Классно, правда?! По какому-то счастливому стечению обстоятельств для каждого адреса были найдены соответствующие координаты широты и долготы. Но, как правило, все происходит не так гладко. И если вы столкнетесь с подобной проблемой, нам необходимо будет добавить проверку ошибок, начиная со второй и до последней строчки предыдущего скрипта.

```
   try:
      place, (lat, lng) = list(g.geocode(full_addy, exactly_one=False))
[0]
      print full_addy + "," + str(lat) + "," + str(lng)
   except:
      print full_addy + ",NULL,NULL"
```

Таким образом вы отдаете поручение найти координаты широты и долготы, а если это сделать не получится, то напечатать строку с адресом и нулевыми значениями координат. Запустите Python'овский скрипт и сохраните полученные результаты в файле, а затем вернитесь к нулям. Что делать с ними? Вы можете проверить адреса без данных через Geopy или просто ввести их вручную в Geocoder.us.

Только точки

Теперь, когда у вас есть точки с широтой и долготой, вы можете нанести их на карту. Самый простой способ — это реализовать компьютерный эквивалент прикалывания канцелярских кнопок к бумажной карте на доске. То есть для каждого пункта вы расставите метки на карте, как это показано на схеме на рис. 8.1.

Концепция простая, но вы уже можете отслеживать в данных такие характеристики, как кластеры, распределения и выбросы.

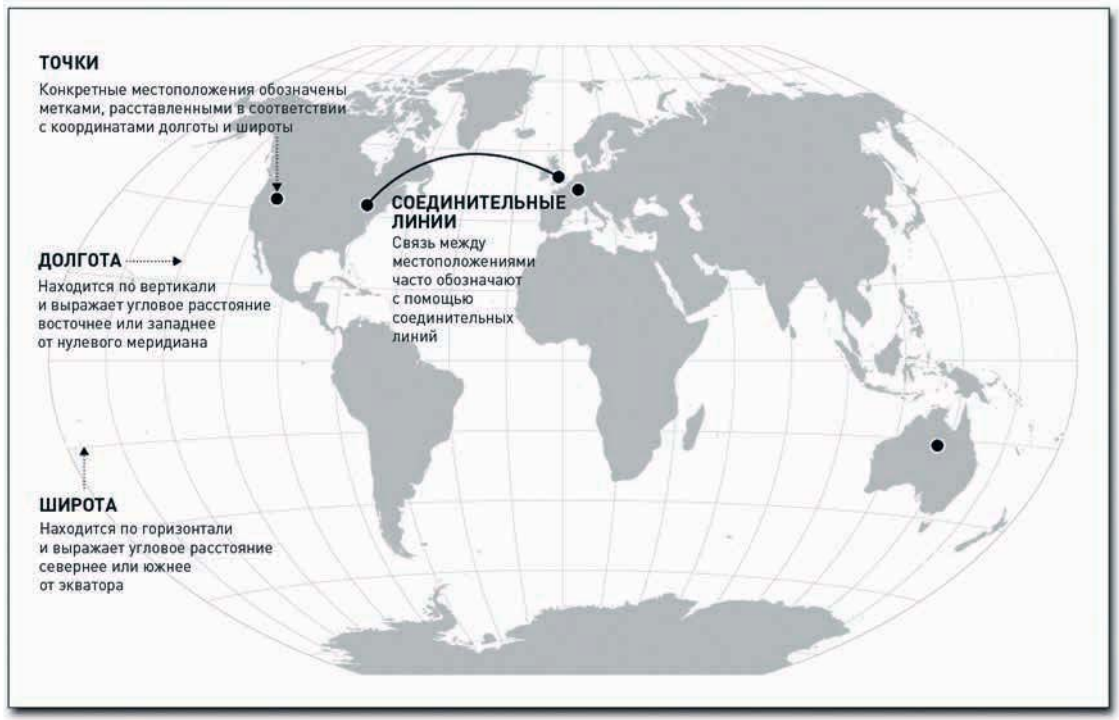


Рис. 8.1. Схема нанесения точек на карте

КАРТА С ТОЧКАМИ

Хотя функционал R в области маппинга довольно ограничен, тем не менее он существенно облегчает процесс размещения точек на карте. Почти всю работу за вас выполнит пакет `maps`. Установите его. Можете сделать это через Package Installer или с помощью `install.packages()` из консоли. Когда закончите инсталляцию, загрузите пакет в рабочую область памяти.

```
library(maps)
```

Следующий шаг — загрузка данных. Вы можете свободно воспользоваться перечнем местоположений магазинов Costco, который вы уже геокодировали, но для большего удобства я разместил уже обработанные данные в Сети, так что вам лучше будет загрузить их напрямую с URL.

```
costcos <-
   read.csv("http://book.flowingdata.com/ch08/geocode/costcos-geocoded
   .csv", sep=",")
```

А теперь давайте приступим к маппингу. Когда создаете свои собственные карты, бывает полезно думать о них как о слоях (вне зависимости от того, каким именно программным обеспечением вы пользуетесь). Нижний слой — это обычно базовая карта с нанесенными на нее географическими границами, а поверх нее вы будете наносить слои с данными. В нынешнем случае нижний слой представляет собой карту Соединенных Штатов, а верхний — местоположения магазинов Costco. И вот как можно создать первый слой, показанный на рис. 8.2.

```
map(database="state")
```

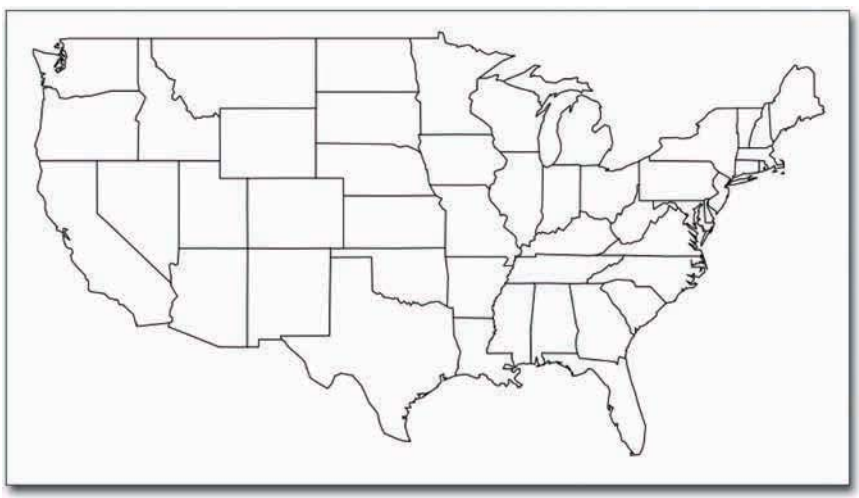


Рис. 8.2. Простая карта Соединенных Штатов

Теперь вам нужно нанести поверх первого слоя второй (с магазинами Costco), для чего вам следует воспользоваться функцией `symbols()`. Это та же самая функция, которую вы использовали для создания пузырьковых диаграмм в главе 6, «Визуализация зависимостей», и применять вы ее станете точно так же, с той лишь разницей, что вместо *x*- и *y*-координат вы будете передавать в нее широту и долготу. А еще установите *add* на *TRUE*, чтобы знаки наносились на карту, а не использовались для создания новой диаграммы.

```
symbols(costcos$Longitude, costcos$Latitude,
   circles=rep(1, length(costcos$Longitude)), inches=0.05, add=TRUE)
```

Результат показан на рис. 8.3. Все круги одинакового размера, потому что для *circles* вы задали массив из единиц, по длине равный количеству местоположений, а еще установили аргумент *inches* на 0,05, что и сделало круги именно такого размера. Если вы хотите, чтобы метки были еще мельче, то вам следует уменьшить это значение.

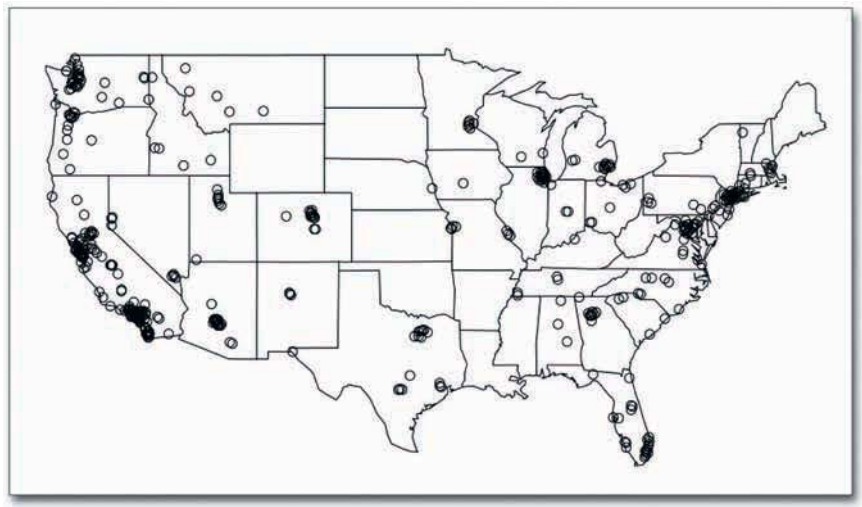


Рис. 8.3. Карта местоположения магазинов Costco

Как и раньше, вы можете изменить цвет и самой карты, и кругов таким образом, чтобы местоположение магазинов выделялось, а линии границ штатов оставались на заднем фоне, как показано на рис. 8.4. Давайте, измените цвет точек на фирменный красный, а границы штатов — на светло-серый.

```
map(database="state", col="#cccccc")
symbols(costcos$Longitude, costcos$Latitude, bg="#e2373f", fg="#ffffff",
   lwd=0.5, circles=rep(1, length(costcos$Longitude)),
   inches=0.05, add=TRUE)
```

На рис. 8.3 незаполненные круги и сама карта были одного цвета, с обводкой одинаковой толщины, а потому все смешивалось в кучу. Правильно подобранные цвета сразу же выводят данные на первый план и фокусируют внимание именно на них.

Неплохая работа, особенно учитывая то, что она была выполнена с помощью всего нескольких строк кода. Компания Costco явно нацелена на открытие магазинов на двух побережьях. Особенно много их

Рис. 8.4. Использование цвета для выделения отдельных точек на карте

сгруппировано в северной Калифорнии и северо-западном Вашингтоне, а также на северо-востоке страны.

Однако здесь налицо явное упущение. Точнее, два упущения. Где Аляска и где Гавайи? Они также являются частью Соединенных Штатов, но мы их не видим, хотя с `map()` использовали «штатовскую» базу данных. Дело в том, что эти два штата находятся во «всемирной» базе данных, так что если вы захотите увидеть местоположение магазинов Costco на Аляске и на Гавайях, вам придется составить карту всего мира, как показано на рис. 8.5.

```
map(database="world", col="#cccccc")
symbols(costcos$Longitude, costcos$Latitude, bg="#e2373f", fg="#ffffff",
   lwd=0.3, circles=rep(1, length(costcos$Longitude)),
   inches=0.03, add=TRUE)
```

Рис. 8.5. Карта мира с магазинами Costco

Это лишняя трата места, я понимаю. В документации вы найдете различные опции, которые можно использовать для решения данной проблемы, но можно также и просто перенести карту в Illustrator и там отредактировать ее, дав Соединенные Штаты крупным планом или просто удалив другие страны.

Но бывают и такие ситуации, когда вы хотите наоборот — уменьшить охват карты, скажем, картографировать местоположение магазинов Costco только в нескольких штатах. Это вы можете сделать с помощью аргумента *region*.

```
map(database="state", region=c("California", "Nevada", "Oregon",
   "Washington"), col="#cccccc")
symbols(costcos$Longitude, costcos$Latitude, bg="#e2373f", fg="#ffffff",
   lwd=0.5, circles=rep(1, length(costcos$Longitude)), inches=0.05,
   add=TRUE)
```

ПОДСКАЗКА

Когда во время работы с R вас охватывают сомнения, сразу же обращайтесь к документации, описывающей функцию или пакет, в которых вы «застряли», для чего просто наберите ее/его название, предварив его вопросительным знаком.

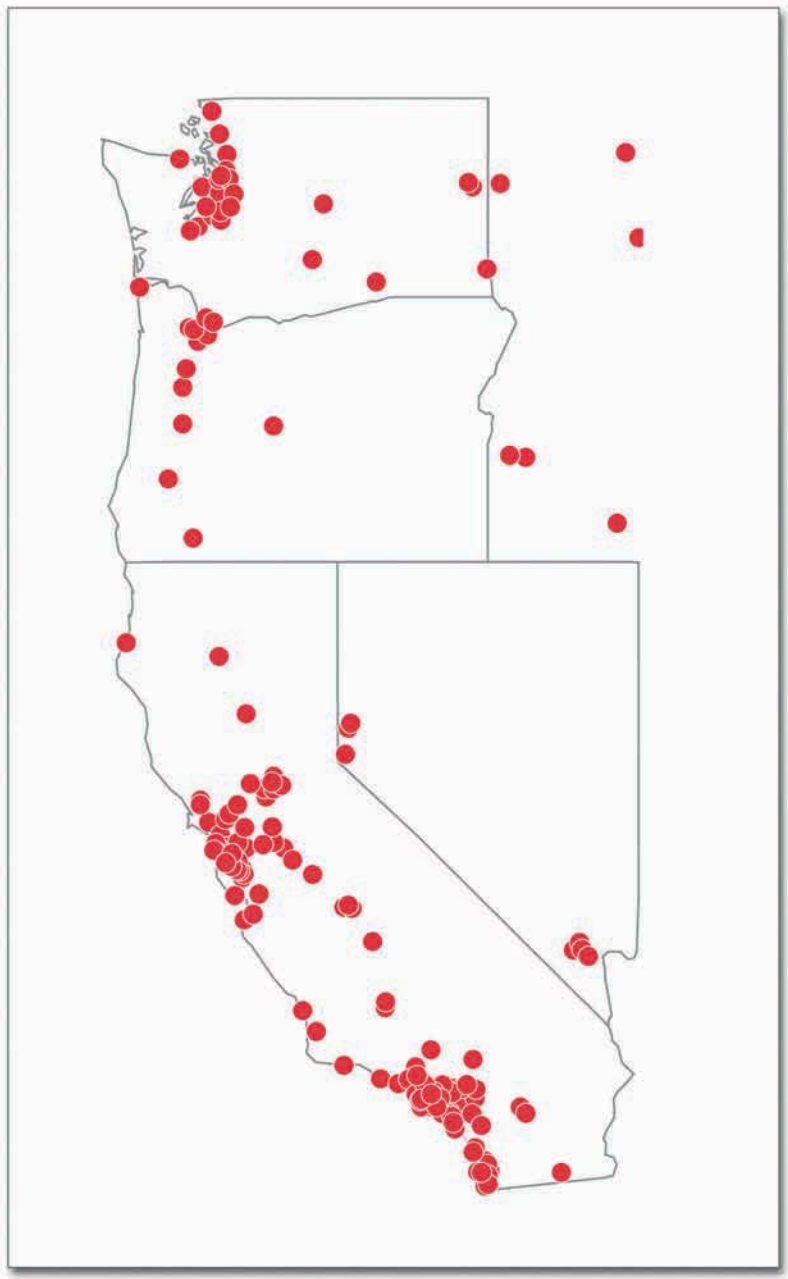


Рис. 8.6. Местоположение магазинов Costco в отобранных штатах

Таким образом, как это показано на рис. 8.6, вы создадите нижний слой с изображенными на нем границами штатов Калифорния, Невада, Орегон и Вашингтон. Поверх него вы создадите слой с данными. Некоторые из точек находятся вне территории этих штатов, но они попадают в вычерчиваемый регион, а потому и появляются на карте. И опять-таки это дело элементарное — удалить их в любимой программе — редакторе векторной графики.

КАРТА С ЛИНИЯМИ

В некоторых случаях бывает полезно соединить линиями точки на карте, особенно когда важна их очередность. С ростом популярности таких сервисов по определению локализации, как Foursquare, трассировка местоположений уже отнюдь не является малораспространенным занятием. Самый легкий способ начертить линии — это воспользоваться функцией, закономерно названной `lines()`. Чтобы продемонстрировать, как она работает, я поделюсь с вами сведениями о местах, которые я объехал во время своей семидневной (и семиночной) командировки в качестве тайного агента правительства Шпионляндии. Как обычно, начните с загрузки и создайте базовую карту мира.

```
faketrace <-
   read.csv("http://book.flowingdata.com/ch08/
   points/fake-trace.txt",sep="\t")
map(database="world", col="#cccccc")
```

Чтобы взглянуть на данные, наберите в консоли R **`faketrace`**. Как вы видите, здесь только две колонки с широтой и долготой и всего восемь пунктов. Можно предположить, что пункты уже выстроены в том порядке, в каком я их посетил за долгие семь ночей.

```
     latitude      longitude
1    46.31658      3.515625
2    61.27023      69.609375
3    34.30714      105.468750
4    -26.11599     122.695313
5    -30.14513     22.851563
6    -35.17381     -63.632813
7    21.28937      -99.492188
8    36.17336      -115.180664
```

Вперед, начертите линии. Просто вставьте эти две колонки в `lines()`. И еще задайте цвет (*col*) и толщину линии (*lwd*).

```
lines(faketrace$longitude, faketrace$latitude, col="#bb4cd4", lwd=2)
```

А теперь добавьте точки так же, как вы это делали, когда наносили местоположение магазинов Costco. В итоге вы получите изображение, похожее на то, что представлено на рис. 8.7.

```
symbols(faketrace$longitude, faketrace$latitude, lwd=1, bg="#bb4cd4",
fg="#ffffff", circles=rep(1, length(faketrace$longitude)), inches=0.05,
add=TRUE)
```

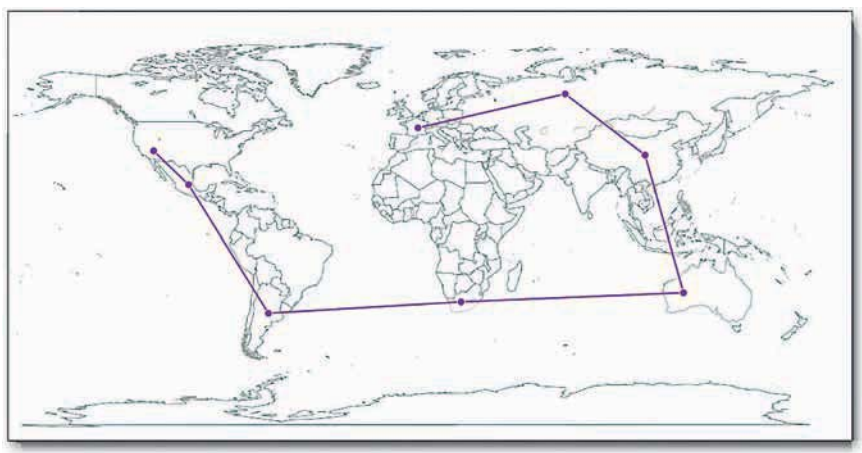


Рис. 8.7. Трассировка маршрута

По истечении этих семи дней и ночей я решил, что шпионское дело — не для меня. Занятие оказалось отнюдь не таким увлекательным, каким его демонстрирует Джеймс Бонд. Однако мне

все же удалось наладить связи во всех странах, которые я посетил. Может, вам будет интересно провести линии из моего настоящего местоположения до всех перечисленных точек, как это показано на рис. 8.8.

```
map(database="world", col="#cccccc")
for (i in 2:length(faketrace$longitude)-1) {
   lngs <- c(faketrace$longitude[8], faketrace$longitude[i])
   lats <- c(faketrace$latitude[8], faketrace$latitude[i])
   lines(lngs, lats, col="#bb4cd4", lwd=2)
}
```

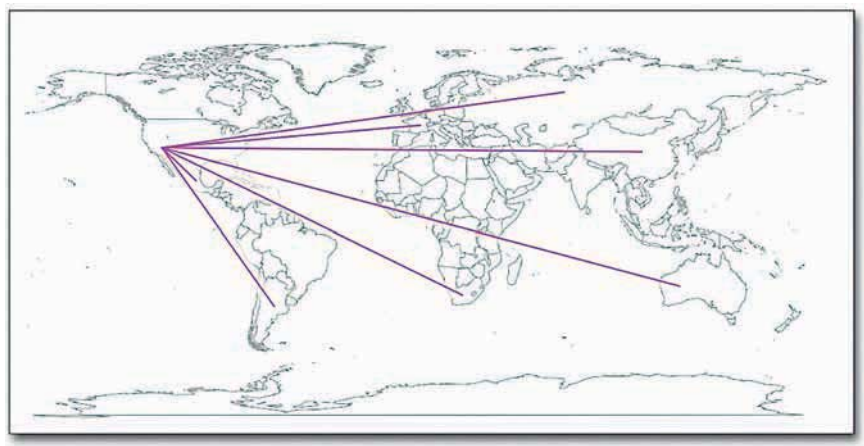


Рис. 8.8. Визуализация мировых связей

После того как создадите базовую карту, вы пройдете в цикле по каждому пункту и начертите линию от последнего пункта в наборе данных до каждого из остальных. Не то чтобы такая графика была особенно информативной, но в некоторых случаях ей можно найти очень полезное применение. Основная же идея здесь заключается в том, что вы можете создать карту, а затем, применяя другие графические функции R, начертить на ней все, что вам захочется, изначально располагая лишь координатами широты и долготы.

Кстати сказать, я на самом деле никогда не служил агентом Шпионляндии. Я просто пошутил.

Масштабированные точки

А теперь давайте вернемся назад к настоящим данным и более интересным темам, нежели мои вымышленные шпионские проделки. Дело в том, что в реальной жизни данные обычно содержат не только координаты неких местоположений. Как правило, к ним привязана и еще какая-нибудь величина, такая как объем продаж компании или население города. Вы

и в этом случае можете создать карту с точками или даже вспомнить принципы пузырьковой диаграммы — и применить их на карте.

Мне не нужно еще раз объяснять, почему размер пузырьков следует определять по площади, а не по радиусу, правда?! Отлично, договорились.

КАРТА С ПУЗЫРЬКАМИ

В этом примере мы рассмотрим данные об уровне подростковой рождаемости, приведенные в «Отчете о развитии человечества», составляемом ООН. То есть речь пойдет о количестве живорожденных в 2008 году на 1000 женщин в возрасте от 15 до 19 лет. Геокординаты предоставлены сервисом GeoCommons. Вам необходимо сделать размер пузырьков пропорциональным соответствующим коэффициентам.

Код будет похож на тот, который вы применяли, когда наносили на карту магазины Costco. Только не забывайте, что в прошлый раз, когда вы устанавливали размер кружков, вы просто передавали в функцию `symbols()` вектор, составленный из единиц. Теперь же для определения размера вы будете использовать `sqrt()` с коэффициентами рождаемости (fertility).

```
fertility <-
   read.csv("http://book.flowingdata.com/ch08/points/adol-fertility.csv")
map('world', fill = FALSE, col = "#cccccc")
symbols(fertility$longitude, fertility$latitude,
   circles=sqrt(fertility$ad_fert_rate), add=TRUE,
   inches=0.15, bg="#93ceef", fg="#ffffff")
```

Результат представлен на рис. 8.9. Сразу видно, что в африканских странах, похоже, самый высокий уровень подростковой рождаемости, а в Европе этот коэффициент относительно низок. Но из одних только географических данных не понять, какова именно величина каждого из этих кружков, так как нет легенды. Больше информации вы получите, воспользовавшись `summary()` в R.

```
summary(fertility$ad_fert_rate)
```

```
   Min.   1stQu.   Median   Mean    3rdQu.   Max.     NA's
   3.20   16.20    39.00    52.89   78.20    201.40   1.00
```

Для вас или для меня — для аудитории, состоящей из одного человека, — этого вполне достаточно. Но если вы хотите, чтобы вашу графику поняли и другие люди, которые сами данные не видели, вам придется кое-что им объяснить. Например, вы можете добавить аннотации, выделяя страны с самым высоким и самым низким уровнем подростковой рождаемости, вставить указатель с данными по той стране, где проживает большинство ваших читателей (в моем случае это США), и написать небольшой вводный текст, чтобы подготовить читателей к тому, что они увидят. Эти изменения представлены на рис. 8.10.

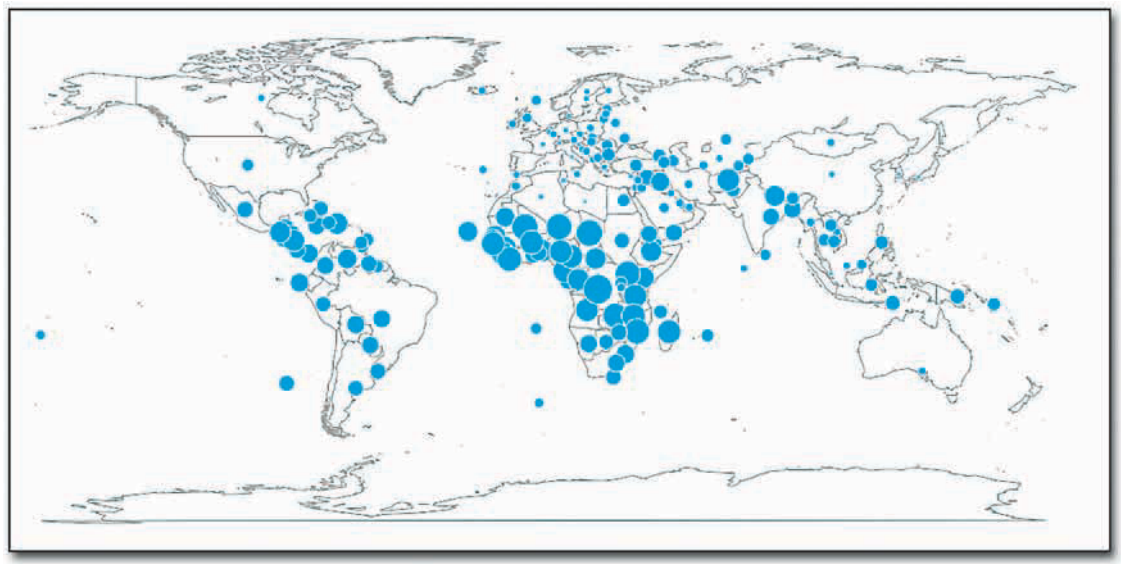


Рис. 8.9. Уровень подростковой рождаемости в мире

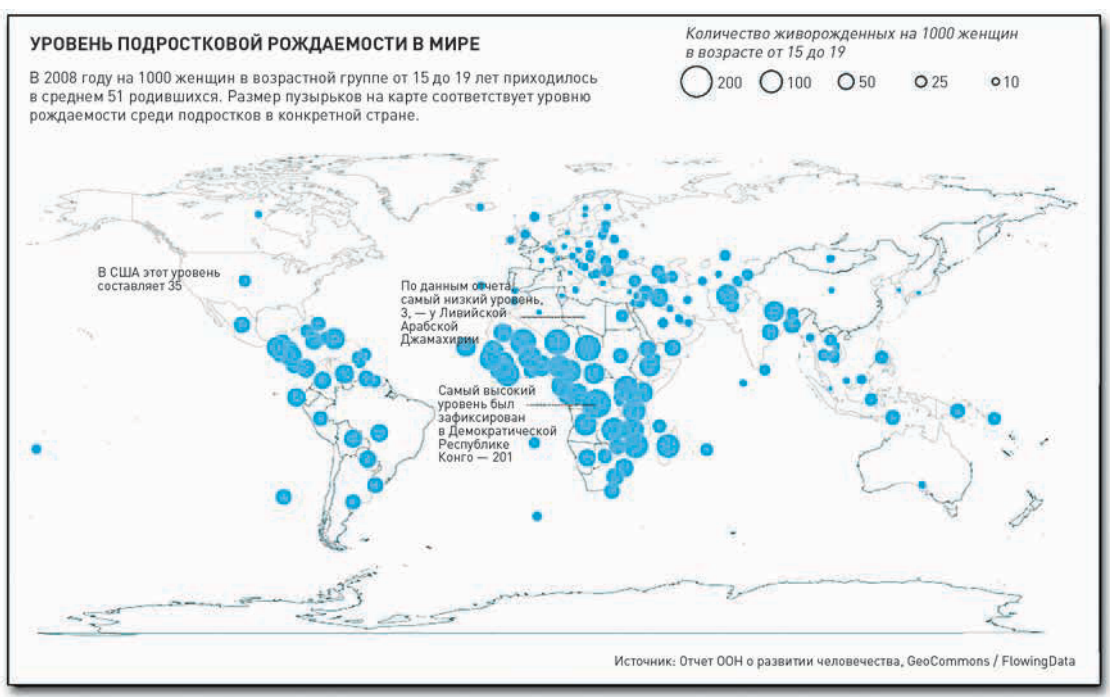


Рис. 8.10. Данные об уровне подростковой рождаемости с пояснениями для более широкой аудитории

Регионы

Добиться большего с помощью точек на карте нельзя, так как они представляют собой лишь отдельные местоположения. Округа, штаты, государства и континенты — это целые регионы с границами, и географические данные по ним обычно поставляются в обобщенном виде. Например, намного легче найти данные о состоянии здоровья населения государства или штата, нежели отдельных людей или пациентов конкретной больницы. Как правило, такие сведения засекречены в целях сохранения конфиденциальности, хотя часто подобное происходит и по другой причине: просто обобщенные данные легче распространять. Так или иначе, в большинстве случаев вы будете использовать пространственные данные именно в таком виде, а потому научитесь их визуализировать.

Окрашивание в соответствии с данными

Самый распространенный способ нанесения на карту региональных данных — это картограмма. В ней регионы окрашиваются в различные цвета на базе неких количественных показателей и в соответствии с разработанной вами цветовой шкалой, как это показано на рис. 8.11. Области и места уже определены, так что вам остается лишь подобрать для использования подходящую цветовую шкалу.

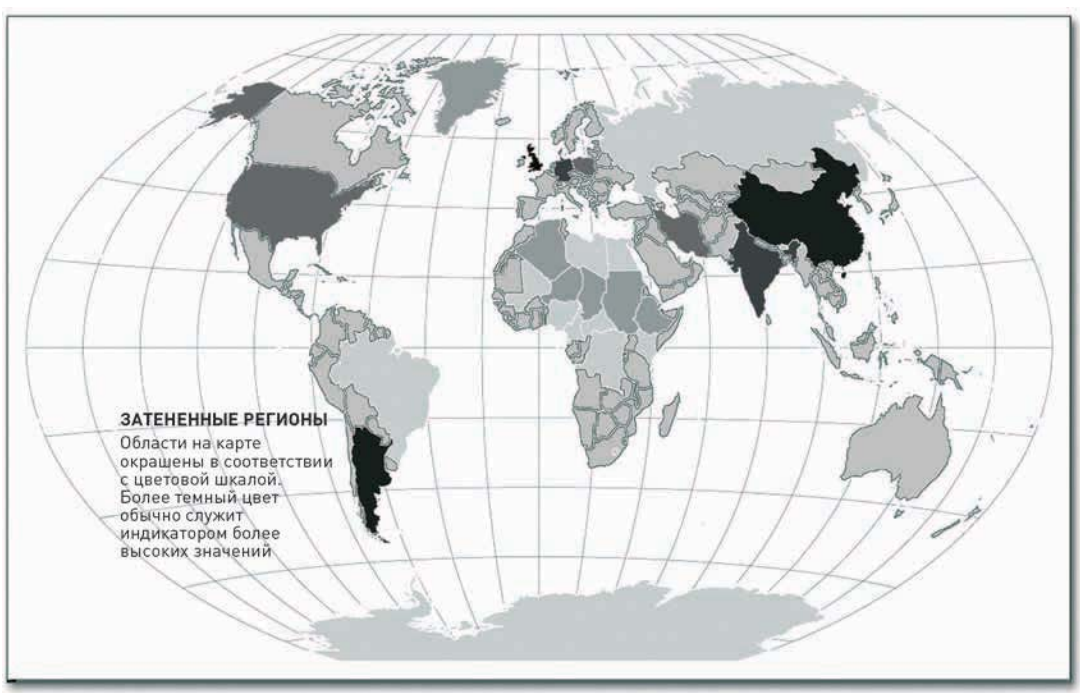


Рис. 8.11. Структура картограммы

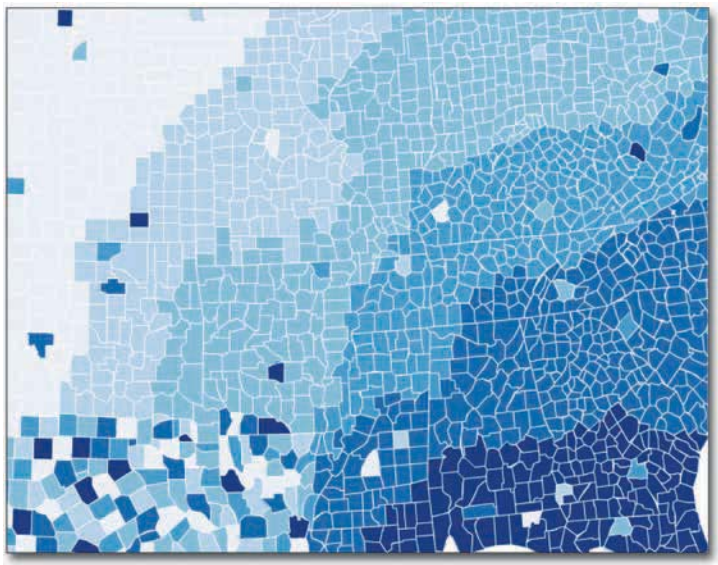


Рис. 8.12. Последовательная цветовая схема, разработанная с помощью ColorBrewer

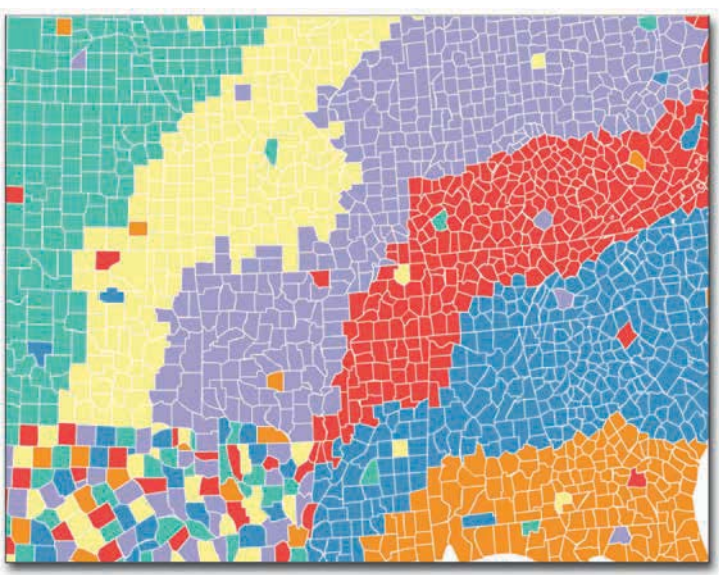


Рис. 8.14. Квалитативная цветовая схема, разработанная с помощью ColorBrewer

Как уже говорилось в предыдущей главе, ColorBrewer Синтии Брюэр — отличное средство для подбора цветов или, по крайней мере, для того, чтобы начать с него разработку цветовой палитры. Если данные у вас непрерывные, то, возможно, вы захотите, чтобы цветовая шкала также была непрерывной и шла от светлого до темного оттенков какого-то одного цвета (или охватывала множество аналогичных цветов, расположенных на цветовом круге последовательно), как на рис. 8.12.

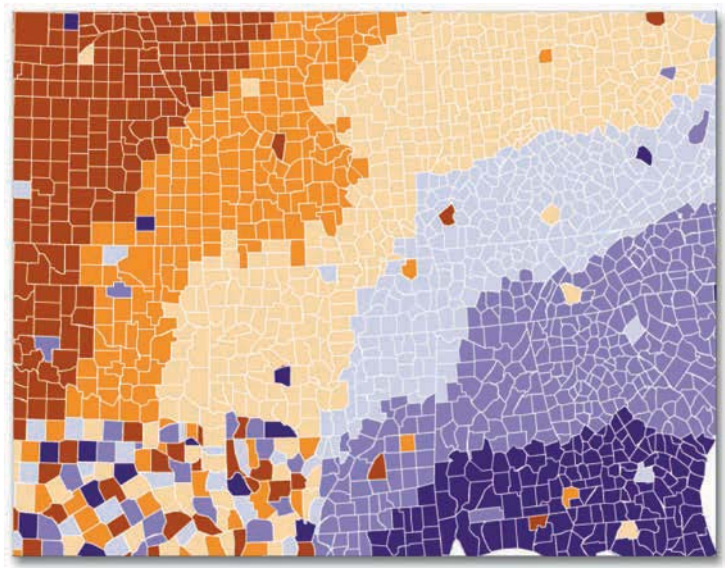


Рис. 8.13. Расходящаяся цветовая схема, разработанная с помощью ColorBrewer

А если ваши данные характеризуются двухсторонне, скажем, они бывают лучше и хуже или выше и ниже определенного порогового значения, тогда вы можете отдать предпочтение расходящейся цветовой схеме, типа той, что продемонстрирована на рис. 8.13.

И наконец, если вам приходится работать с качественными данными, в которых выделяются различные классы или категории, тогда вы, возможно, захотите использовать отдельный цвет для каждой из этих категорий (рис. 8.14).

Когда вы определитесь с цветовой схемой, вам останется сделать две вещи. Во-первых, решить, как выбранные вами цвета будут соотноситься с охватом данных, и во-вторых, присвоить каждому региону определенный цвет, основываясь на выработанном вами принципе. И то, и другое вы можете выполнить с помощью Python и Scalable Vector Graphics (SVG), как показано в следующих примерах.

НАНЕСИТЕ НА КАРТУ ДАННЫЕ ПО ОКРУГАМ

Каждый месяц Бюро трудовой статистики предоставляет новые данные об уровне безработицы в США по округам. С сайта Бюро вы можете скачать как свежайший выпуск, так и сведения о состоянии дел несколько лет назад. Однако браузер на сайте этой организации несколько устаревший и неудобный, так что я вам советую простоты ради (и на тот случай, если сайт Бюро поменяет свой адрес) скачать данные с `http://book.flowingdata.com/ch08/regions/unemployment-aug2010.txt`. Они выстроены в шести колонках. В первой содержится код Бюро трудовой статистики. Следующие две колонки, вместе взятые, — это индивидуальный идентификатор конкретного округа. Четвертая и пятая колонки содержат названия округов и месяца, за который производилась оценка уровня безработицы, соответственно. Последняя колонка — это, собственно, процент людей в округе, которые оказались без работы. В данном конкретном примере вас будет интересовать только идентификатор округа (то есть код по ФСОИ — Федеральному стандарту обработки информации) и уровень безработицы.

Давайте приступим к маппингу. В предыдущих примерах вы создавали базовые карты в R, однако теперь вы в этих же целях будете использовать другие инструменты, а именно Python и SVG. Первый из них нужен вам для обработки данных, а второй — для создания самой карты. Однако вам не придется начинать все с нуля. Вы можете взять чистую карту (такую, как показано на рис. 8.15) с «Викисклада» (Wikimedia Commons) по адресу http://commons.wikimedia.org/wiki/File:USA_Counties_with_FIPS_and_names.svg. На этой странице вы найдете ссылки на карту в четырех размерах в формате PNG и еще на одну в формате SVG. Вам нужна та, что в SVG. Скачайте файл и сохраните его как `counties.svg` в той же директории, в которой вы сохранили данные об уровне безработицы.

Для тех, кто не знаком с SVG, скажу, что важнее всего здесь то, что это, по сути, XML-файл, текст с тегами, и вы можете обрабатывать его в текстовом редакторе так же, как вы делаете это с HTML-файлами.

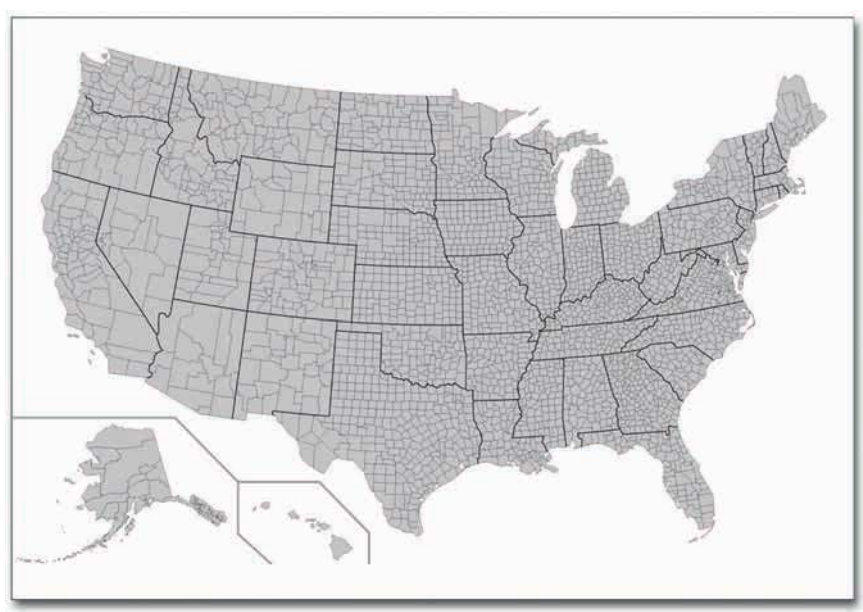


Рис. 8.15. Чистая карта с округами США с «Викисклада»

ПОДСКАЗКА

SVG-файлы — это XML-файлы, в которые легко можно вносить изменения в текстовом редакторе. А еще отсюда следует, что вы можете производить парсинг SVG-кода и менять его программным способом.

Браузер и программа для просмотра изображений прочитывает XML, а XML говорит им, что именно показывать: какие цвета использовать и какие формы нарисовать.

Чтобы понять это в полной мере, откройте SVG-файл с картой в текстовом редакторе и посмотрите, с чем, собственно, вы имеете дело. По большей части файл состоит из SVG-объявлений и всяких шаблонов, которые вас не особо интересуют в данный момент.

Прокрутите файл еще немного вниз, пока не появятся теги `<path>` (контур), как показано на рис. 8.16. Все эти числа в каждом из тегов определяют границы округа. Вы их трогать не будете. Вам нужно лишь изменить цвет заливки каждого округа так, чтобы он соответствовал конкретному уровню безработицы. Для этого вам придется отредактировать `style` во всех контурах.

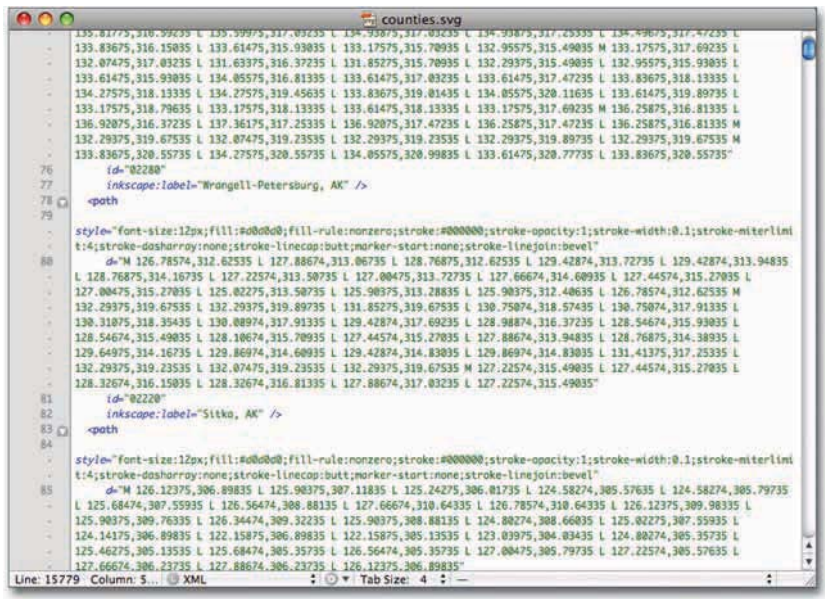


Рис. 8.16. Описание контуров в SVG-файле

Вы обратили внимание на то, что все теги `<path>` начинаются с атрибута *style*? Те из вас, кто когда-либо писал CSS, должны были заметить это сразу же. А еще в них есть атрибут *fill* (заливка), за которым следует шестнадцатеричный цвет, и если вы измените это значение в SVG-файле, вы измените и цвет конечного изображения. Вы можете изменить их все вручную, но округов более 3000, и у вас на это уйдет уйма времени. Так что вспомните о своем старом друге, Beautiful Soup, Python'овском пакете, который делает процесс парсинга XML и HTML довольно-таки легким занятием.

Создайте новый файл в той же директории, в которой находятся SVG-карта и данные по безработице. Сохраните его как `colorize_svg.py`. После этого вам необходимо будет импортировать CSV-файл с данными и произвести парсинг SVG-файла с помощью Beautiful Soup, так что начните с импорта необходимых пакетов.

```
import csv
from BeautifulSoup import BeautifulSoup
```

Затем откройте CSV-файл и сохраните его так, чтобы можно было запустить итерацию по строчкам, используя `csv.reader()`. Обратите внимание на то, что 'r' в функции `open()` означает лишь то, что вы хотите открыть файл и прочитать его содержимое, а не, скажем, дописать в него некие новые строчки.

```
reader = csv.reader(open('unemployment-aug2010.txt', 'r'), delimiter=",")
```

А теперь загрузите чистую SVG-карту округов.

```
svg = open('counties.svg', 'r').read()
```

Круто. Вы загрузили все, что вам необходимо для создания картограммы. На этом этапе сложность заключается в том, чтобы неким образом связать данные с SVG. Но как? Что между ними общего? Я вам подскажу. Это имеет отношение к индивидуальным идентификаторам каждого округа, как я уже говорил выше. Если вы подумали, что речь идет о кодах по ФСОИ, то вы правы!

У каждого контура в SVG-файле есть индивидуальный id, который объединяет в себе код штата и код округа по ФСОИ. В данных по безработице также каждая строчка содержит коды конкретного штата и округа, но здесь они разделены. Например, код штата Алабама по ФСОИ — 01, а код округа Отога в штате Алабама — 001. А в id-контуре округа Отога в SVG-файле эти два кода объединены: 01001.

Вам необходимо сохранить данные по безработице таким образом, чтобы вы могли в процессе итерации по каждому контуру извлечь сведения об уровне безработицы в разных округах по их кодам по ФСОИ. Если вы чувствуете, что начинаете теряться, не отходите от меня — когда перейдем к написанию компьютерного кода, все станет яснее. А пока что вам стоит запомнить самое важное: коды по ФСОИ — это и есть связующее звено между SVG и CSV, и вы можете им воспользоваться.

ПОДСКАЗКА

Контуры в SVG-файлах, особенно географических, обычно имеют индивидуальный id. Им не всегда является код по ФСОИ, но принцип работы с ним остается в силе.

Чтобы сохранить данные по безработице в таком виде, который максимально облегчит впоследствии доступ к ним по ФСОИ-коду, используйте структуру в Python под названием dictionary (словарь). Он дает вам возможность сохранять и извлекать значения по ключевому слову. В данном случае вашим ключевым словом будет комбинированный (штатский плюс окружной) код по ФСОИ (FIPS), как это показано в следующем коде:

```
unemployment = {}
min_value = 100; max_value = 0
for row in reader:
   try:
      full_fips = row[1] + row[2]
      rate = float( row[8].strip() )
      unemployment[full_fips] = rate
   except:
      pass
```

Затем с помощью BeautifulSoup произведите парсинг SVG-файла. Большинство тегов имеют открывающий и закрывающий элементы, но там есть еще и парочка самозакрывающихся тегов, которые вам необходимо определить. Воспользуйтесь функцией `findAll()` чтобы отыскать все контуры на карте.

```
soup = BeautifulSoup(svg, selfClosingTags=['defs','sodipodi:namedview'])
paths = soup.findAll('path')
```

Далее сохраните цвета, которые я подобрал в ColorBrewer, в виде списка в Python. Я остановился на последовательной цветовой схеме с множеством оттенков от пурпурного до красного.

```
colors = ["#F1EEF6", "#D4B9DA", "#C994C7", "#DF65B0", "#DD1C77", "#980043"]
```

Мы уже приближаемся к кульминации. Как я уже говорил, вам нужно изменить атрибут *style* для всех контуров в SVG. Вас интересует только цвет заливки, но, упрощая себе задачу, вы можете изменить стиль целиком — чтобы не заниматься парсингом и не менять один лишь цвет. Я изменил шестнадцатеричное значение после `stroke` на `#ffffff`, то есть на белый. Теперь границы будут уже не серыми, как до сих пор, а белыми.

```
path_style = 'font-size:12px;fill-rule:nonzero;stroke:#ffffff;strokeopacity:
1;stroke-width:0.1;stroke-miterlimit:4;strokedasharray:
none;stroke-linecap:butt;marker-start:none;strokelinejoin:
bevel;fill:'
```

А еще я сдвинул `fill` в конец и оставил значение пустым, так как это именно та часть, которая зависит от уровня безработицы в каждом округе.

Теперь уже вы готовы изменить цвета! Вы можете проитерировать все контуры (за исключением границ штатов и линий, отделяющих Гавайи и Аляску) и окрасить их сообразно уровню безработицы. Если этот уровень выше 10, используйте самый темный оттенок, а все округа, где этот уровень ниже 2, пусть будут окрашены в самый светлый оттенок.

```
for p in paths:

   if p['id'] not in ["State_Lines", "separator"]:
      # pass
   try:
      rate = unemployment[p['id']]
   except:
      continue

   if rate > 10:
      color_class = 5
   elif rate > 8:
      color_class = 4
```

```
elif rate > 6:
   color_class = 3
elif rate > 4:
   color_class = 2
elif rate > 2:
   color_class = 1
else:
   color_class = 0

color = colors[color_class]
p['style'] = path_style + color
```

Остался последний шаг: вывести на экран SVG-файл. Для этого вы воспользуетесь функцией `prettify()`. Данная функция преобразовывает ваш «суп» в строку, которую браузер уже понимает и может интерпретировать.

```
print soup.prettify()
```

Теперь остается только запустить Python'овский скрипт и сохранить полученный результат как новый SVG-файл под названием, скажем, `colored_map.svg` (рис. 8.17).

▶ Весь скрипт целиком вы найдете здесь: `http://book.flowingdata.com/ch08/regions/colorize_svg.py.txt`.

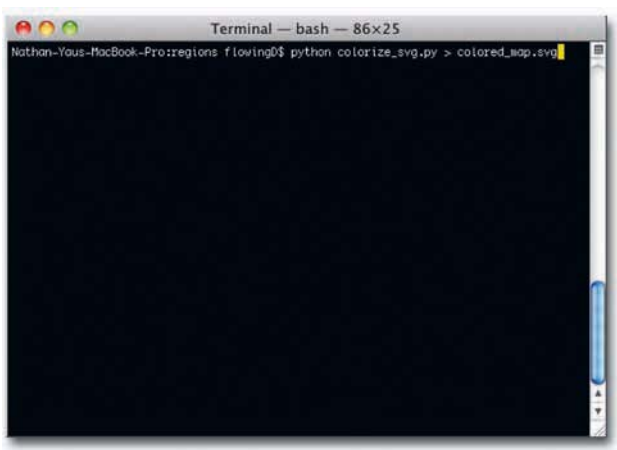


Рис. 8.17. Выполнение Python'овского скрипта и сохранение результата в виде нового SVG-файла

Откройте свою новехонькую, с иголочки, картограмму в Illustrator или в каком-нибудь современном браузере, таком как Firefox, Safari или Chrome, чтобы увидеть плоды своего труда, которые должны выглядеть, как показано на рис. 8.18. Теперь уже нетрудно заметить, в каких регионах страны уровень безработицы в августе 2010 года был выше, а в каких — ниже. Совершенно очевидно, что на западном побережье и во многих округах на юго-востоке страны безработица оказалась весьма высокой, как, впрочем, и на Аляске и в Мичигане. А во многих округах в центре страны уровень безработицы был относительно низким.

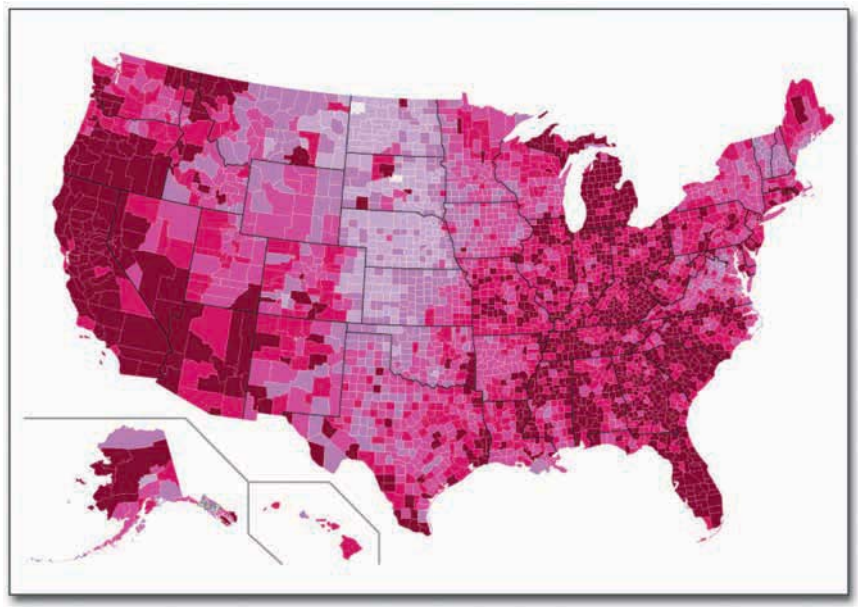


Рис. 8.18. Картограмма, демонстрирующая уровень безработицы

Теперь, когда трудная часть упражнения осталась позади, вы вольны дооформить карту, как вашей душе угодно. Вы можете отредактировать SVG-файл в Illustrator, можете изменить цвет и толщину границ, а также добавить аннотации, чтобы придать своему произведению законченный вид и сделать его понятным для более широкой аудитории (подсказка: картограмме недостает легенды).

А самая хорошая новость состоит в том, что верхний код можно использовать многократно, применяя его с другими наборами данных, использующими коды по ФСОИ. Да и даже с этим самым набором данных вы можете использовать его вторично, пробуя различные цветовые схемы и выискивая самый подходящий вариант для конкретной тематики.

В зависимости от данных вы можете менять также и пороговые значения для окрашивания каждого региона. В примерах до сих пор мы использовали равные пороговые значения, и регионы были окрашены в 6 тонов — каждые 2 процентных пункта составляли новую категорию. Все округа с уровнем безработицы выше 10 процентов входили в одну категорию; далее следовали округа с уровнем между 8 и 10 процентами, затем между 6 и 8, и т. д. Другой распространенный способ определять пороговые значения — по квартилям. В таком случае вы станете использовать 4 цвета, и каждый будет представлять четвертую часть регионов.

Так, нижний, средний и верхний квартили данных по безработице будут находиться соответственно на уровне 6,9, 8,7 и 10,8 процента. Это означает, что в четверти округов уровень безработицы был ниже 6,9 процента, в другой четверти он находился в диапазоне от 6,9 до 8,7, еще один — между 8,7 и 10,8, и в последней четверти округов уровень безработицы был выше

10,8 процента. Чтобы окрасить округа в соответствии с этим распределением, вам необходимо изменить список цветов в вашем скрипте примерно так, как показано ниже. Это пурпурная цветовая схема, в которой каждой четверти присваивается свой оттенок.

```
colors = ["#f2f0f7", "#cbc9e2", "#9e9ac8", "#6a51a3"]
```

Затем вам нужно изменить условия по цвету в цикле `for`, используя верхние квартили.

```
if rate > 10.8:
   color_class = 3
elif rate > 8.7:
   color_class = 2
elif rate > 6.9:
   color_class = 1
else:
   color_class = 0
```

Запустите скрипт и сохраните результат, как и в предыдущий раз. Так вы получите картограмму как на рис. 8.19. Обратите внимание на то, что теперь в светлый оттенок окрашено больше округов.

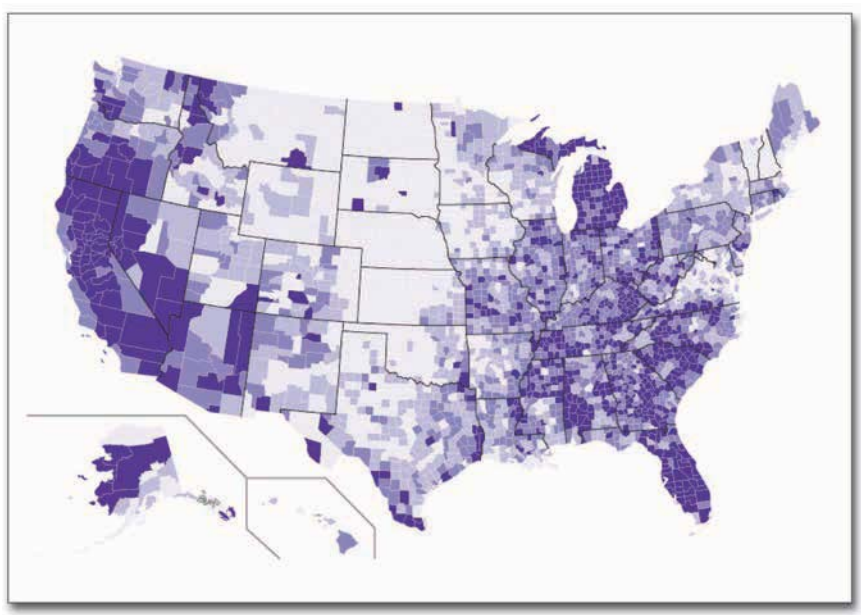


Рис. 8.19. Уровень безработицы с разбивкой по квартилям

Чтобы расширить возможности многократного применения данного кода, вы можете квартили не задавать жестко, а подсчитать их программным способом. В Python это делается легко.

Необходимо сохранить список значений, рассортировать их от наименьшего к наибольшему и найти значения, отмеченные как одна четверть, половина и три четверти. А точнее, если рассматривать вопрос в контексте данного примера, вы можете видоизменить первый цикл в colorize_svg.py таким образом, чтобы сохранить лишь значения уровня безработицы.

```
unemployment = {}
rates_only = [] # To calculate quartiles
min_value = 100; max_value = 0; past_header = False
for row in reader:
   if not past_header:
      past_header = True
      continue

   try:
      full_fips = row[1] + row[2]
      rate = float( row[5].strip() )
      unemployment[full_fips] = rate
      rates_only.append(rate)
   except:
      pass
```

Затем вы можете рассортировать массив и найти квартили.

```
# Quartiles
rates_only.sort()
q1_index = int( 0.25 * len(rates_only) )
q1 = rates_only[q1_index]     # 6.9

q2_index = int( 0.5 * len(rates_only) )
q2 = rates_only[q2_index]     # 8.7

q3_index = int( 0.75 * len(rates_only) )
q3 = rates_only[q3_index]     # 10.8
```

Вместо того чтобы жестко закреплять в вашем коде значения 6,9, 8,7 и 10,8, вы можете вместо них вставить q1, q2 и q3 соответственно. Преимущество подсчета этих значений программным способом заключается в том, что в подобном случае вы сможете использовать этот же код и с другим набором данных, просто заменив CSV-файл новым.

Какой цветовой шкале отдать предпочтение, зависит от данных, с которыми вы работаете, и от того, какое послание вы хотите донести. Для этого конкретного набора данных я выбрал линейную шкалу, так как она нагляднее передает распределение и подчеркивает относительно высокие значения уровня безработицы в разных уголках страны. Взяв за отправную точку рис. 8.18, вы можете придать картограмме более завершенный вид, добавив в нее легенду, заголовок и вводный абзац, как показано на рис. 8.20.

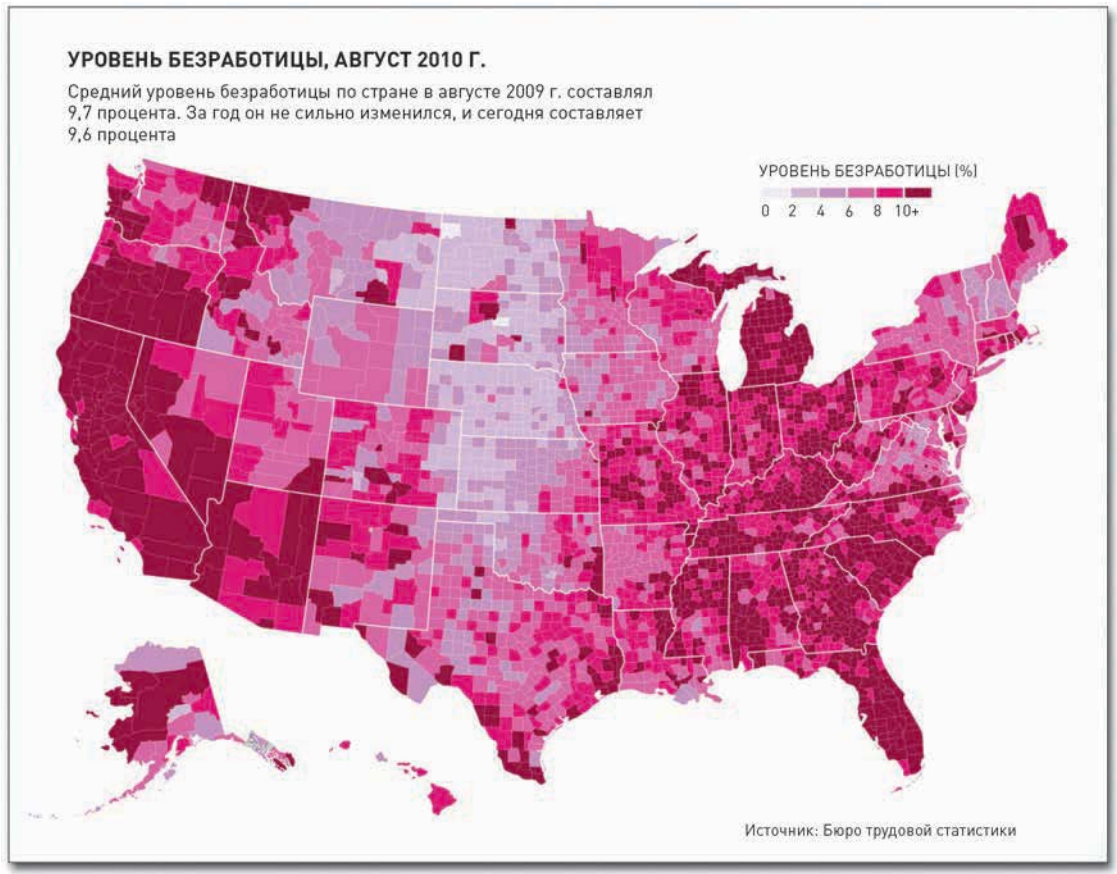


Рис. 8.20. Законченная карта с заголовком, вводным текстом и легендой

НАНЕСИТЕ НА КАРТУ ДАННЫЕ ПО ГОСУДАРСТВАМ

Процесс окрашивания округов в предыдущем примере применим не только к этому типу регионов. Те же самые шаги вы можете проделать и при окрашивании отдельных штатов или государств. Все, что вам нужно, — это SVG-файл с индивидуальными идентификаторами для каждого региона, который вы хотите окрасить (подобные данные легко добыть из «Википедии»), а также данные с идентификаторами, которые вы им сопоставите. Попробуйте сделать это с данными Всемирного банка, находящимися в открытом доступе.

Посмотрите на проценты городского населения, имеющего доступ к источникам очищенной воды, в разных странах в 2008 году. Excel-файл вы можете скачать с сайта Всемирного банка по адресу http://data.worldbank.org/indicator/SH.H2O.SAFE.UR.ZS/countries. Для вашего удобства я разместил уже обработанные данные в виде CSV-файла на сайте FlowingData: `http://book.flowingdata.com/ch08/worldmap/water-source1.txt`. По некоторым государствам данные

ПОДСКАЗКА

Всемирный банк — один из самых богатых источников демографических данных с разбивкой по странам. Лично я всегда начинаю поиски именно с него.

отсутствуют, но это обычная ситуация, когда имеешь дело с многонациональной статистикой, а потому я удалил эти строчки из CSV-файла.

Итак, у нас есть семь колонок. В первой идут названия стран, во второй — коды стран (как вы думаете, они смогут сыграть для вас роль индивидуального идентификатора?), в последних пяти колонках содержатся данные о соответствующих процентах за время с 1990 по 2008 годы.

Базовый вариант карты снова можно взять из «Википедии». Там есть множество вариантов карты мира в формате SVG, но эту конкретную я нашел здесь: http://en.wikipedia.org/wiki/File:BlankMap-World6.svg. Скачайте SVG-файл с максимальным разрешением и сохраните его в той же самой директории, в которой находятся данные. Как видно на рис. 8.21, это чистая карта мира, окрашенная в серый цвет и с белыми границами.

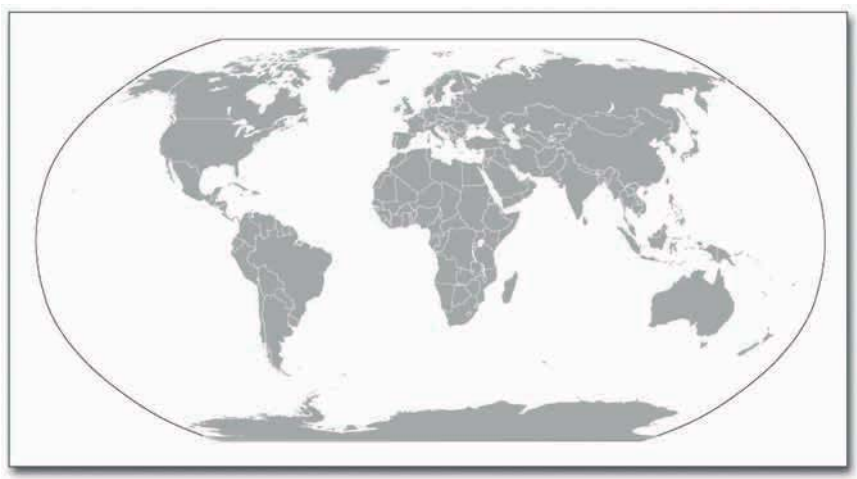


Рис. 8.21. Чистая карта мира

Откройте SVG-файл в текстовом редакторе. Конечно, он имеет вид текста и отформатирован как XML, хотя и несколько по-другому, нежели файл с данными по округам из предыдущего примера. Контуры не содержат так полезных нам id, да и атрибут *style* не используется. Зато у контуров есть классы, которые выглядят как коды государств. Однако состоят они только из двух букв. А коды государств, используемые Всемирным банком, состоят из трех букв.

Как указывается в документах Всемирного банка, эта организация применяет трехбуквенные коды стандарта ISO 3166-1 alpha 3. SVG-файл из «Википедии», однако, использует двухбуквенные коды стандарта ISO 3166-1 alpha 2. Названия стандартов звучат ужасно, я понимаю, но не переживайте, вам не придется их заучивать. Все, что вам нужно запомнить, — это то, что в «Википедии» имеется переводная таблица и находится она по адресу http://en.wikipedia.org/wiki/ISO_3166-1. Я скопировал ее и вставил в Excel, а затем сохранил важную информацию как текстовый файл. В нем две колонки: одна для alpha 2, другая для alpha 3. Скачать файл вы можете отсюда: `http://book.flowingdata.com/ch08/`

worldmap/country-codes.txt. Используйте содержащуюся в нем таблицу для перевода кодов из одного стандарта в другой.

А теперь давайте перейдем к оформлению стран. Эту процедуру мы также выполним по-другому. Вместо того чтобы вносить изменения в атрибуты прямо в тегах path, для окрашивания регионов мы используем CSS *снаружи* контуров. Давайте сразу посмотрим, как это делается.

Создайте файл под названием generate_css.py в той же директории, в которой находятся SVG- и CSV-файлы. И опять импортируйте пакет CSV, чтобы загрузить данные из CSV-файлов с кодами стран и процентами населения, имеющего доступ к очищенной воде.

```
import csv
codereader = csv.reader(open('country-codes.txt', 'r'), delimiter="\t")
waterreader = csv.reader(open('water-source1.txt', 'r'), delimiter="\t")
```

Затем сохраните коды государств таким образом, чтобы можно было легко переключаться с alpha 3 на alpha 2.

```
alpha3to2 = {}
i = 0
next(codereader)
for row in codereader:

   alpha3to2[row[1]] = row[0]
```

Этот фрагмент сохраняет коды в словаре Python, где alpha 3 является ключом, а alpha 2 — значением.

А теперь, как и в предыдущем примере, произведите итерацию по каждой строчке данных о водных ресурсах и присвойте ей цвет на основании значения для текущей страны.

```
i = 0
next(waterreader)
for row in waterreader:

   if row[1] in alpha3to2 and row[6]:
      alpha2 = alpha3to2[row[1]].lower()
      pct = int(row[6])
      if pct == 100:
         fill = "#08589E"
      elif pct > 90:
         fill = "#08589E"
      elif pct > 80:
         fill = "#4EB3D3"
      elif pct > 70:
         fill = "#7BCCC4"
```

```
        elif pct > 60:
            fill = "#A8DDB5"
        elif pct > 50:
            fill = "#CCEBC5"
        else:
            fill = "#EFF3FF"
        print '.' + alpha2 + ' { fill: ' + fill + ' }'

    i += 1
```

Этот фрагмент скрипта выполняет следующие шаги:

1) игнорирует шапку CSV;
2) начинает цикл итерации по данным о воде;
3) если имеется код alpha 2, сопоставленный коду alpha 3 из CSV, и есть данные по стране за 2008 год, он находит соответствующий alpha 2;
4) основываясь на процентах, выбирает подходящий цвет;
5) выдает строку CSS для каждой строки данных.

Запустите `generate_css.py` и сохраните результат как **`style.css`**. Первые несколько строк CSS будут выглядеть так:

```
.af { fill: #7BCCC4 }
.al { fill: #08589E }
.dz { fill: #4EB3D3 }
.ad { fill: #08589E }
.ao { fill: #CCEBC5 }
.ag { fill: #08589E }
.ar { fill: #08589E }
.am { fill: #08589E }
.aw { fill: #08589E }
.au { fill: #08589E }
…
```

Это стандартный CSS. Первая строчка, например, меняет цвет заливки всех контуров с классом .af на #7BCCC4.

Откройте в текстовом редакторе файл `style.css` и скопируйте все его содержимое. Затем откройте SVG-карту и вставьте это содержимое примерно на уровне 135-й строки, под скобками для *`.oceanxx`*. Поздравляю, вы только что создали картограмму мира, окрашенную в соответствии с процентами населения, имеющего доступ к очищенной воде (рис. 8.22). Темно-синий цвет соответствует 100 процентам, а самые светлые зеленые оттенки указывают на наиболее низкие проценты. Серыми остались те страны, по которым нет данных.

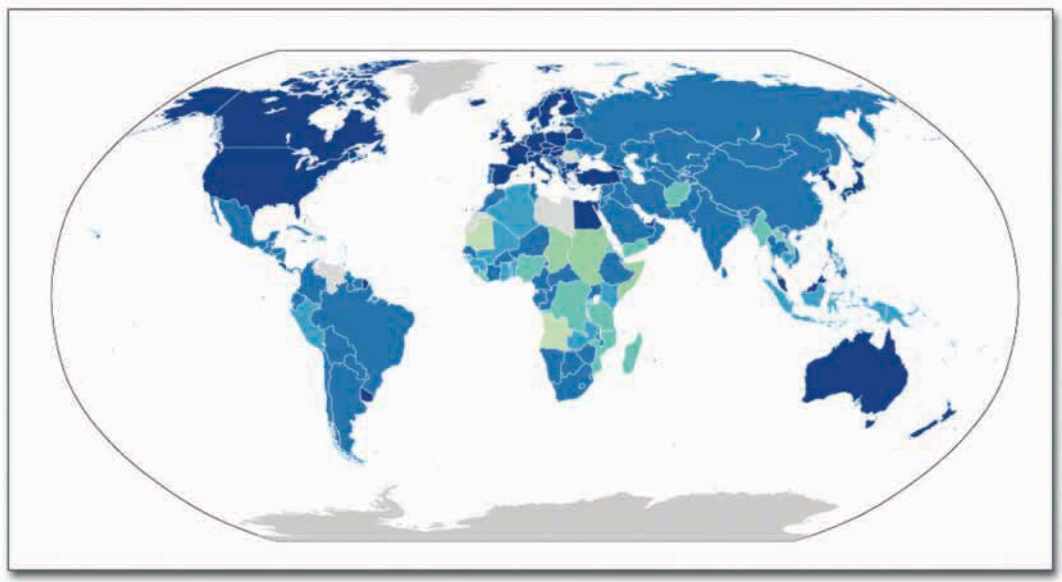


Рис. 8.22. Картограмма мира, показывающая доступ к источникам очищенной воды

Что особенно приятно во всем этом, так то, что теперь вы можете скачать практически любые наборы данных Всемирного банка (а их довольно-таки много) и создать картограмму мира буквально в два счета, лишь подправив несколько строчек кода. Чтобы придать рис. 8.22 более нарядный вид, вы можете открыть SVG-файл в программе Illustrator и заняться редактированием. Карте нужны прежде всего заголовок и легенда, которая бы объясняла, что означают разные оттенки (рис. 8.23).

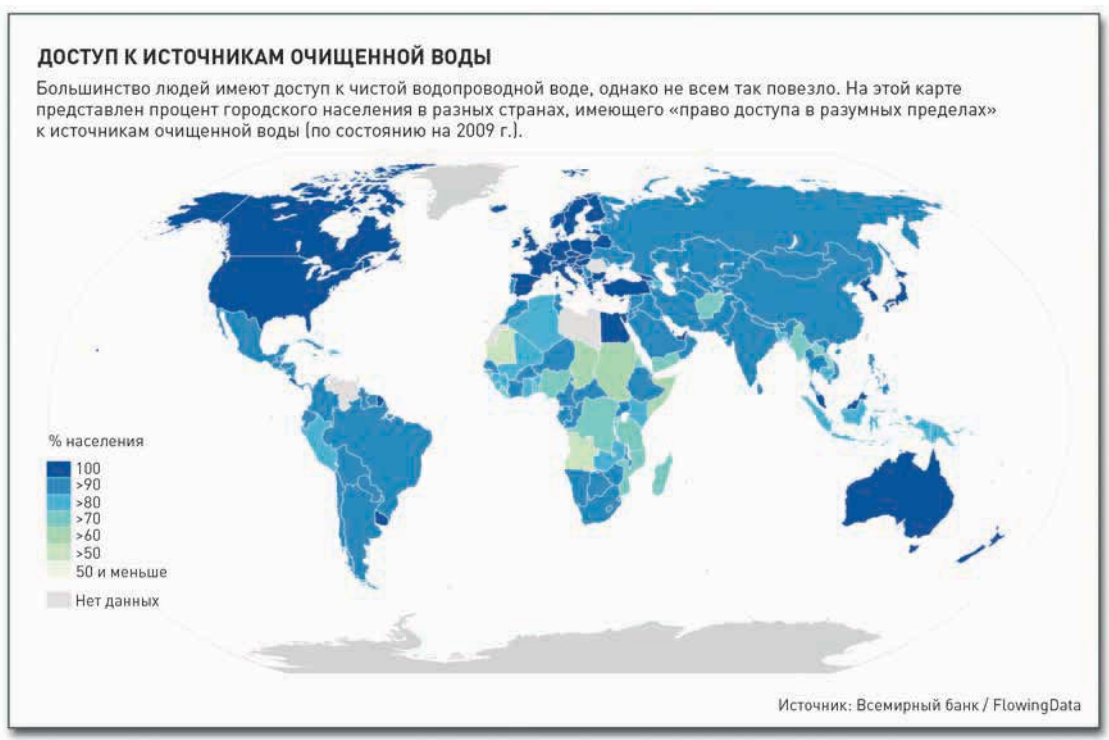


Рис. 8.23. Карта мира в законченном виде

Во времени и пространстве

Все предыдущие примеры имели своей целью научить вас визуализировать различные типы данных, как качественных, так и количественных. Вы уже знаете, как менять цвета, категории и символы таким образом, чтобы они соответствовали истории, которую вы хотите рассказать. А еще вы умеете дополнять карты аннотациями и выделять отдельные регионы или свойства, а также увеличивать и уменьшать масштаб, концентрируя внимание на определенных регионах и странах.

Но не спешите, это еще не все! Вам не следует останавливаться на достигнутом. Если вы добавите еще одно измерение данных, вы сможете рассмотреть изменения одновременно и во времени, и в пространстве.

В главе 4 вы визуализировали время более абстрактно, с помощью линий и графиков, что само по себе полезно, но когда к подобного рода данным добавляется еще и местоположение, может оказаться, что карты — это более интуитивно понятный способ представить паттерны и изменения. Так бывает легче разглядеть кластеры или группы регионов, которые находятся близко друг к другу в физическом плане.

А самое приятное то, что вы сможете применить обретенные вами знания о визуализации данных и во времени, и в пространстве одновременно.

Маленькие панели

С методом маленьких панелей вы ознакомились в главе 6 и использовали его для визуализации зависимостей между категориями. Но, как показано на рис. 8.24, его можно применять также и с пространственными данными. Вместо маленьких столбцовых диаграмм вы будете использовать маленькие карты, по одной для каждого среза времени. Выстройте их слева направо или сложите в штабель сверху вниз, и ваши глаза легко проследят за изменениями.

Например, в конце 2009 года я создал графику, демонстрирующую уровень безработицы в стране по округам (рис. 8.25). На самом деле я взял вариацию кода, который вы использовали в предыдущем разделе, но я применил его к нескольким временны́м срезам.

Рис. 8.24. Маленькие панели с картами

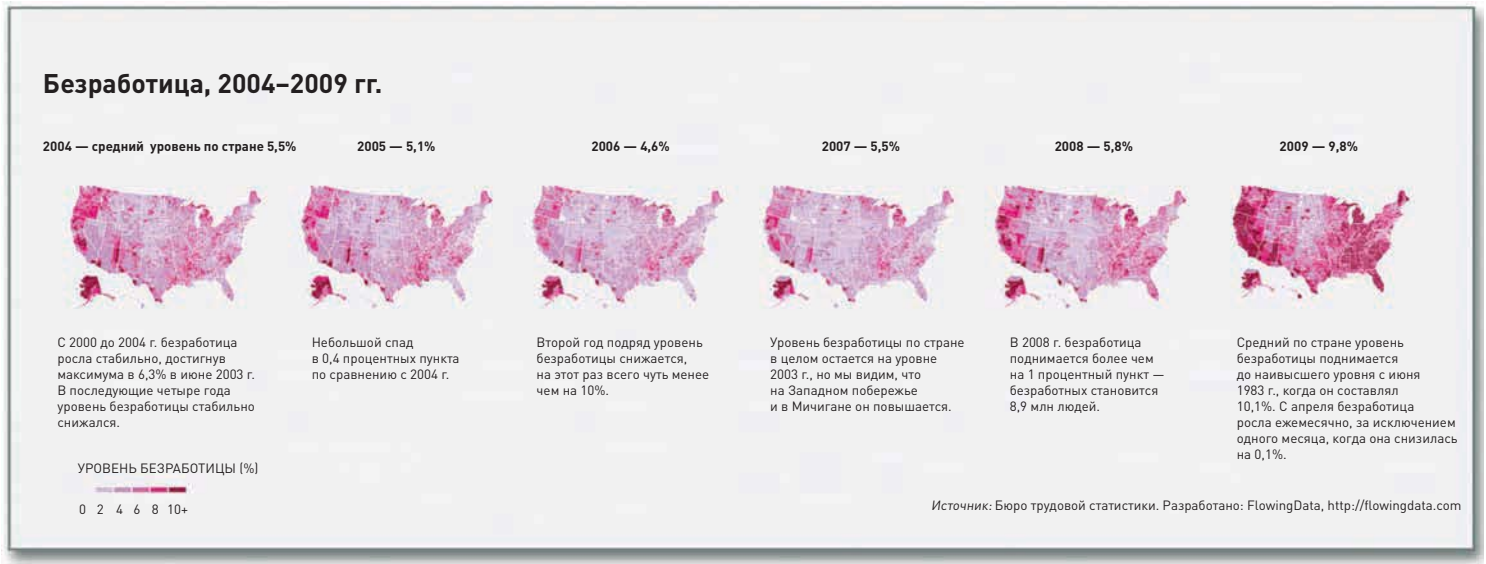


Рис. 8.25. Уровень безработицы с 2004 по 2009 гг.

В таком виде заметить изменения (или их отсутствие) совсем несложно. Давайте присмотримся к картограммам за 2004–2006 годы (рис. 8.26). В это время уровень безработицы в национальном масштабе, как можно видеть, снижался.

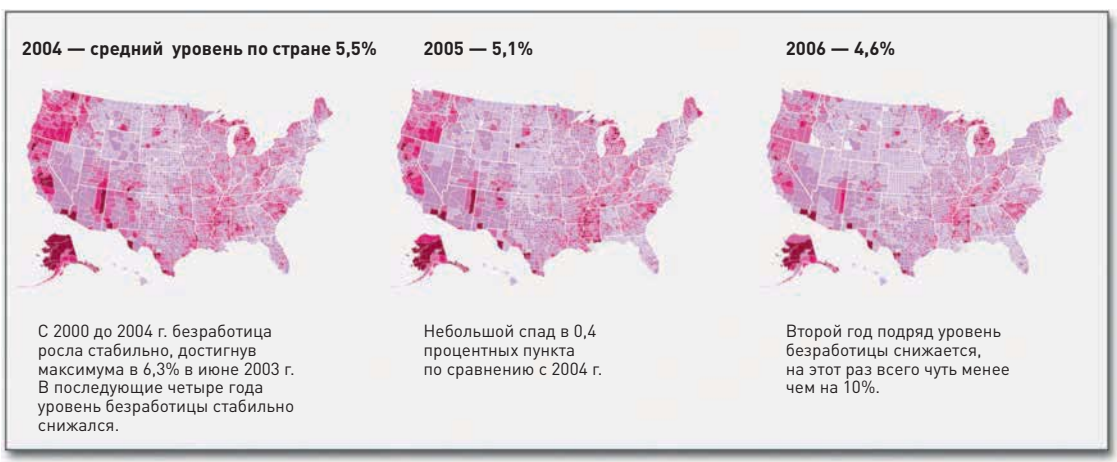


Рис. 8.26. Уровень безработицы с 2004 по 2006 гг.

А затем наступил 2008 год (рис. 8.27), и уже можно заметить определенный рост уровня безработицы, особенно в Калифорнии, Орегоне и Мичигане, а также в некоторых округах на юго-востоке страны.

Включаем быструю перемотку и переходим к 2009 году. Налицо явное различие (рис. 8.28). Уровень безработицы в среднем по стране подскочил на 4 процентных пункта, и округа́ окрасились в темный цвет.

Рис. 8.27. Уровень безработицы в 2008 г.

Рис. 8.28. Уровень безработицы в 2009 г.

▶ Если изображение высокого разрешения получается слишком большим, чтобы вывести его целиком на монитор, полезно вставить его в OpenZoom Viewer (`http://openzoom.org`), чтобы можно было охватить взглядом картинку целиком, а затем вывести крупным планом нужные детали.

Это одна из самых популярных графических работ, которые я публиковал на сайте FlowingData, потому что на ней хорошо видны драматические изменения, наступившие после нескольких лет относительной стабильности. К тому же я использовал программу просмотра OpenZoom Viewer, которая позволяет давать изображение крупным планом в высоком разрешении, так что вы можете сфокусироваться на вашем собственном регионе и проследить, как он менялся.

Я мог бы визуализировать эти данные и в виде диаграммы временны́х рядов, в которой каждая линия представляла бы отдельный округ, но таковых в США более 3000. Диаграмма получилась бы чрезмерно перегруженной, и вы вряд ли смогли бы разобраться, где какой округ.

Уловите разницу

Впрочем, для того чтобы продемонстрировать изменения, вам не всегда придется создавать множество маленьких карт. Иногда более разумно бывает визуализировать сами различия в одной-единственной карте. Таким образом можно и место сэкономить, и подчеркнуть изменения (рис. 8.29).

Рис. 8.29. Фокусировка на изменениях

Если бы вам нужно было скачать статистику Всемирного банка об уровне урбанизации, вы бы получили набор данных, похожий на тот, что использовали в недавнем примере с доступом к очищенной воде. Каждая строка — это страна, а каждый столбец — год. Однако данные об уровне урбанизации — результат грубой оценки количества людей в стране, живущих в городах. Если вы решите нанести данные в таком виде на картограмму, в итоге у вас неминуемо «выделятся» наиболее крупные страны, а произойдет это по той простой причине, что у больших стран население в целом больше. Две карты, демонстрирующие уровень урбанизации в 2005 и 2009 годах, также вряд ли сослужат вам хорошую службу, если только вы не измените значения на пропорции. Чтобы сделать это, вам придется скачать данные за 2005 и 2009 годы по всем странам и затем просто произвести некоторые несложные математические расчеты. Сделать такое нетрудно, но это все равно еще один дополнительный шаг. К тому же, если изменения незначительные, их будет трудно подметить при сравнении нескольких карт.

Вместо этого вы можете просто взять и нанести различия на одной карте. Вам понадобится подсчитать все в Excel или видоизменить предыдущий Python-скрипт, а затем создать одну карту, как показано на рис. 8.30.

Когда вы визуализируете сами различия, легко заметить, в каких именно странах произошли наиболее радикальные перемены, а в каких — самые незначительные. А теперь давайте сравним этот подход с другим. На рис. 8.31 показана доля городских жителей в населении каждой страны в целом по состоянию на 2005 г.

А на рис. 8.32 представлены аналогичные данные за 2009 год. Рисунок настолько похож на предыдущий, что различий практически не видно.

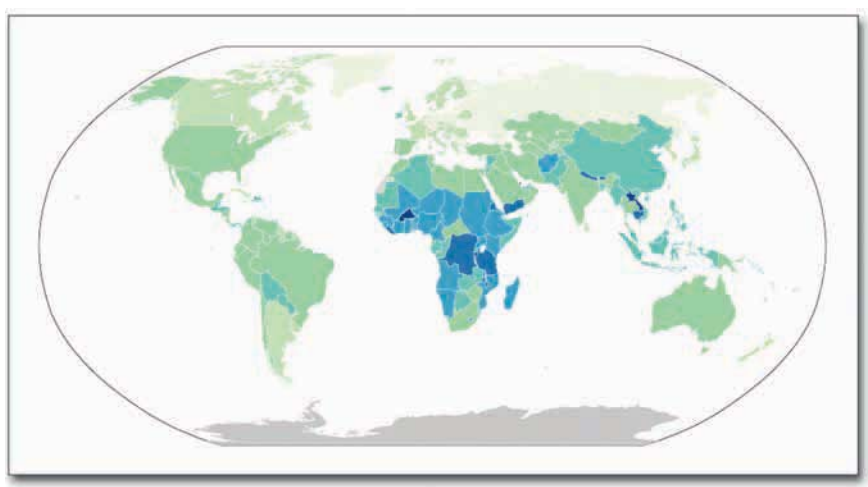


Рис. 8.30. Изменения в уровне урбанизации за период 2005–2009 гг.

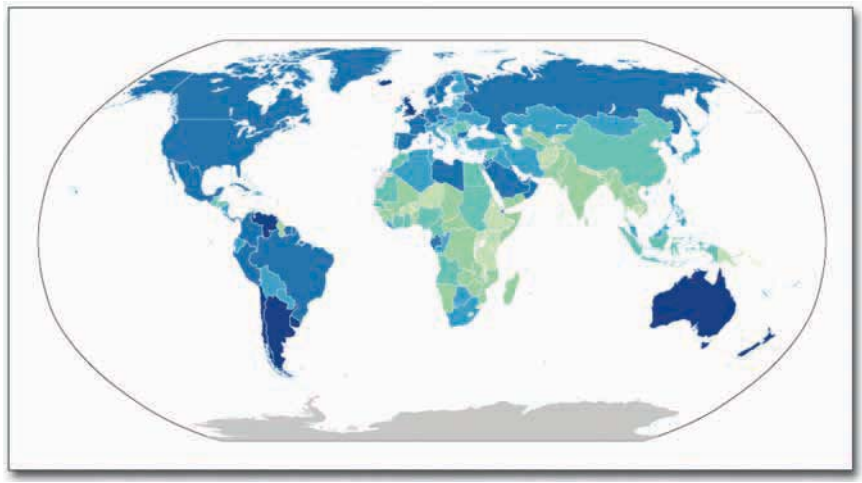


Рис. 8.31. Доля городского населения в 2005 г.

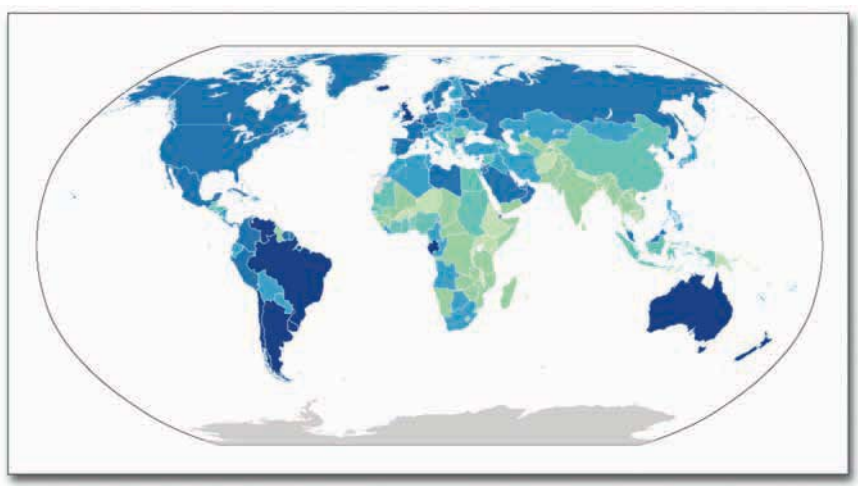


Рис. 8.32. Доля городского населения в 2009 г.

Очевидно, что для этого конкретного примера вариант с одной картой будет более информативным, чем с двумя. Так вам придется проделать гораздо меньше умственной работы, чтобы разгадать, какие изменения произошли за эти 4 года. И сразу становится понятно: хотя в Африке гораздо больше стран с относительно невысокой долей городского населения по сравнению с другими регионами мира, именно на этом континенте в последние годы происходят самые динамичные изменения в процессе урбанизации.

Не забудьте добавить заголовок и легенду, а также указать источник, если собираетесь представить карту вниманию широкой аудитории (рис. 8.33).

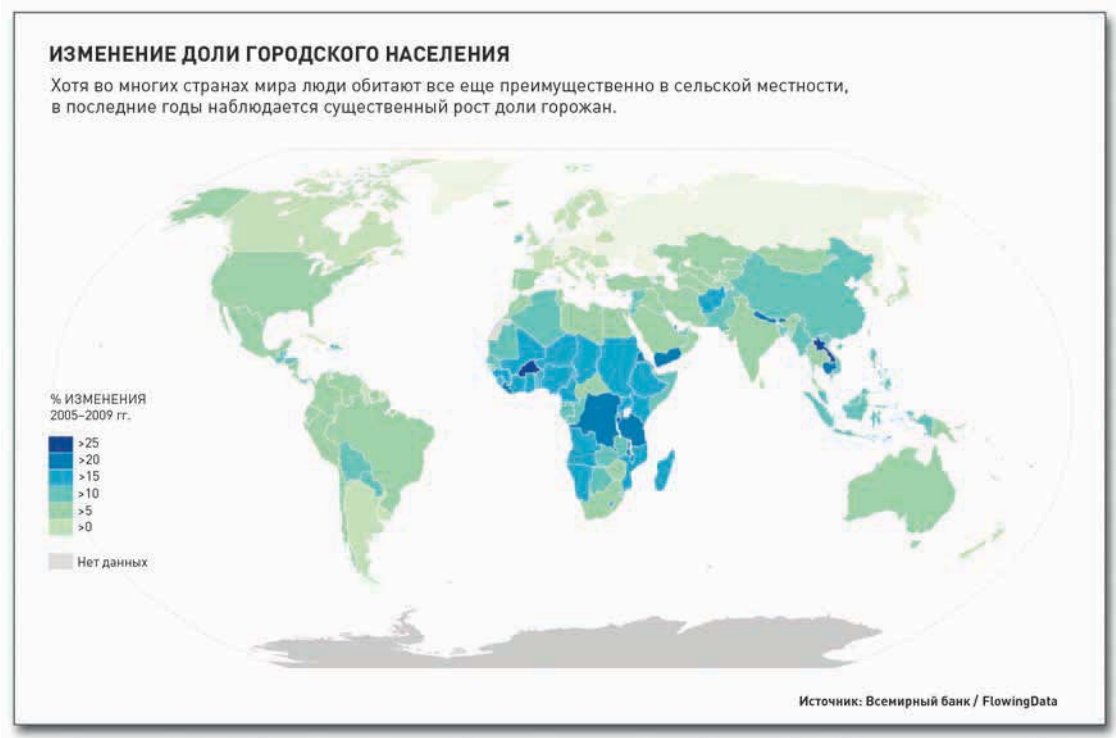


Рис. 8.33. Аннотированная карта различий

Анимация

Один из наиболее очевидных способов визуализации изменений во времени и пространстве — это анимация данных. Вместо того чтобы показывать отдельные карты для различных срезов времени, вы можете продемонстрировать изменения в динамике на одной-единственной интерактивной карте. Подобный подход сохраняет интуитивность изображения и вместе с тем позволяет читателям исследовать данные самостоятельно.

Несколько лет назад я разработал карту, показывающую распространение магазинов сети Walmart на территории Соединенных Штатов (рис. 8.34). Анимация стартует с открытия первого магазина в городе Роджерс, штат Арканзас, в 1962 году и продолжается вплоть до 2010 года. Для каждого открывшегося магазина на карте появляется новая точка. Поначалу рост происходит медленно, но затем точки начинают распространяться по стране чуть ли не как вирус. Количество магазинов все растет и растет, а местами, в тех регионах, где компания осуществляет большие приобретения, этот рост подобен даже взрыву. И прежде, чем вы успеваете сориентироваться, Walmart оказывается уже повсюду.

▶ Посмотреть карту Walmart в динамике можно по адресу `http://datafl.ws/197`.

Рис. 8.34. Анимированная карта США, демонстрирующая рост количества магазинов Walmart

В то время, когда я создавал эту карту, я просто пытался на практике освоить Flash и ActionScript, но карта разошлась по Сети и собрала несколько миллионов просмотров. Позже я создал похожую карту, показывающую рост сети Target (рис. 8.35), и она пользовалась не меньшим успехом.

▶ Посмотреть рост количества магазинов Target можно на: `http://datafl.ws/198`.

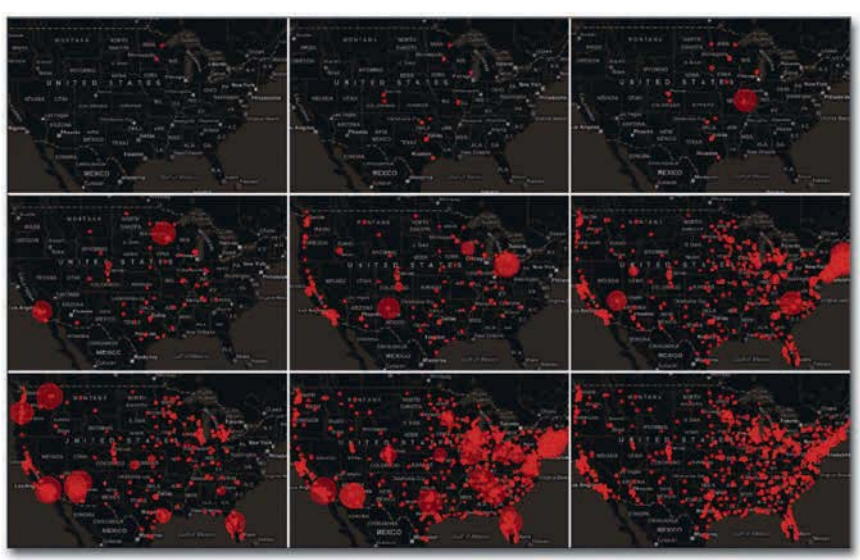


Рис. 8.35. Анимированная карта США, демонстрирующая рост количества магазинов Target

Такой интерес к себе эти карты вызвали, главным образом, по двум причинам. Во-первых, потому, что анимированная карта дает вам возможность увидеть паттерны так четко, как не может никакая диаграмма временных рядов. Обычный график или диаграмма покажут лишь количество магазинов, открывающихся каждый год, что само по себе замечательно, если это как раз та история, которую вы хотели рассказать. Но анимированные карты показывают рост более органично и последовательно, что особенно заметно с картой магазинов Walmart.

Вторая причина заключается в том, что широкая аудитория легко ориентируется в картах. И когда начинается анимация, вы понимаете, что вы видите. Этим я не хочу сказать, что визуализация, требующая времени для ее понимания, не имеет ценности; часто все обстоит ровно наоборот. Однако в Сети временной порог весьма низок, и поскольку карты интуитивно понятны (да к тому же люди могут вывести интересующие их области крупным планом), это, безусловно, подогревало желание посетителей поделиться ссылкой.

СОЗДАЙТЕ АНИМИРОВАННУЮ КАРТУ РОСТА

В следующем примере вы создадите карту роста Walmart в ActionScript. Для этого вы будете использовать Modest Maps — картографическую библиотеку ActionScript, которая поможет вам с базовой картой и выстраиванием интерактива. Остальную часть кода вы создадите самостоятельно. Скачайте весь исходный код с `http://book.flowingdata.com/ch08/Openings_src.zip`. В этом параграфе мы не будем анализировать каждую строчку и каждый файл, а остановимся лишь на самом главном.

▶ Скачайте Modest Maps с `http://modestmaps.com`.

В главе 5, «Визуализация пропорций», когда вы создавали штабельную диаграмму с областями с помощью ActionScript и визуализационного инструментария Flare, я горячо рекомендовал вам использовать Adobe Flex Builder. Эта программа значительно упрощает работу в ActionScript и помогает с организацией кода. Конечно, вы можете выполнить всю процедуру создания кода и в стандартном текстовом редакторе, но Flex Builder объединяет в себе широкие возможности: позволяет и код писать, и ошибки выискивать и устранять, и компилированием заниматься — и все это в одном флаконе. Далее в примере мы будем действовать, исходя из предположения, что Flex Builder у вас уже установлен, хотя вы, конечно, можете просто скачать компилятор ActionScript 3 с сайта Adobe.

ПОДСКАЗКА

Недавно на смену Adobe Flex Builder пришел Adobe Flash Builder. Между ними есть небольшие отличия, но вы можете использовать и тот, и другой продукт.

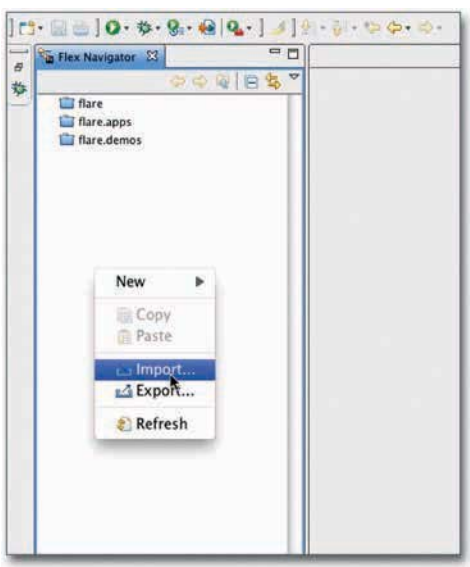


Рис. 8.36. Импортирование ActionScript-проекта

Для начала откройте Flex Builder 3 и щелкните правой кнопкой мыши по левой боковой панели, в которой представлен текущий перечень проектов. Выберите Import (Импортировать), как показано на рис. 8.36.

▶ Чтобы выполнить пример, скачайте весь код карты роста с `http://book.flowingdata.com/ch08/Openings_src.zip`.

Выберите Existing Projects Into Workspace (Существующие проекты в рабочую область), как показано на рис. 8.37.

Затем, как показано на рис. 8.38, пролистайте и найдите директорию, в которой вы сохранили код. После того как вы выберете корневой каталог, на экране появится проект Openings.

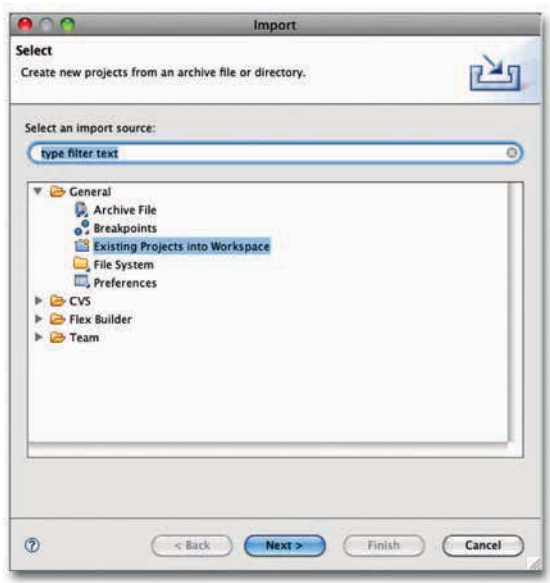


Рис. 8.37. Существующий проект

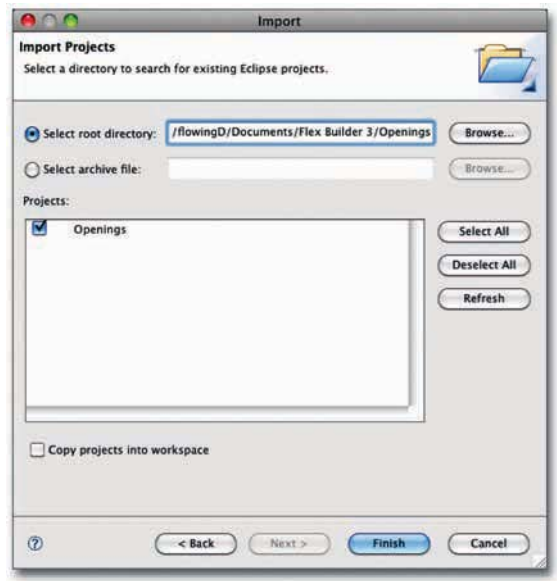


Рис. 8.38. Импортирование проекта Openings

Теперь рабочая область у вас во Flex Builder должна выглядеть примерно так, как на рис. 8.39.

Весь код находится в папке src. Сюда входят также Modest Maps (в папке com) и TweenFilterLite (в папке gs), которые помогут с анимацией переходов.

Теперь, когда проект Openings импортирован, пора приступить к построению карты. Это вы сделаете в два приема: сначала создадите интерактивный базовый вариант карты, а на втором этапе добавите в нее метки.

Добавьте интерактивную базовую карту

Первые строчки кода в Openings.as импортируют необходимые пакеты.

```
import com.modestmaps.Map;
import com.modestmaps.TweenMap;
import com.modestmaps.core.MapExtent;
import com.modestmaps.geo.Location;
import com.modestmaps.mapproviders.OpenStreetMapProvider;
```

```
import flash.display.Sprite;
import flash.display.StageAlign;
import flash.display.StageScaleMode;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.filters.ColorMatrixFilter;
import flash.geom.ColorTransform;
import flash.text.TextField;
import flash.net.*;
```

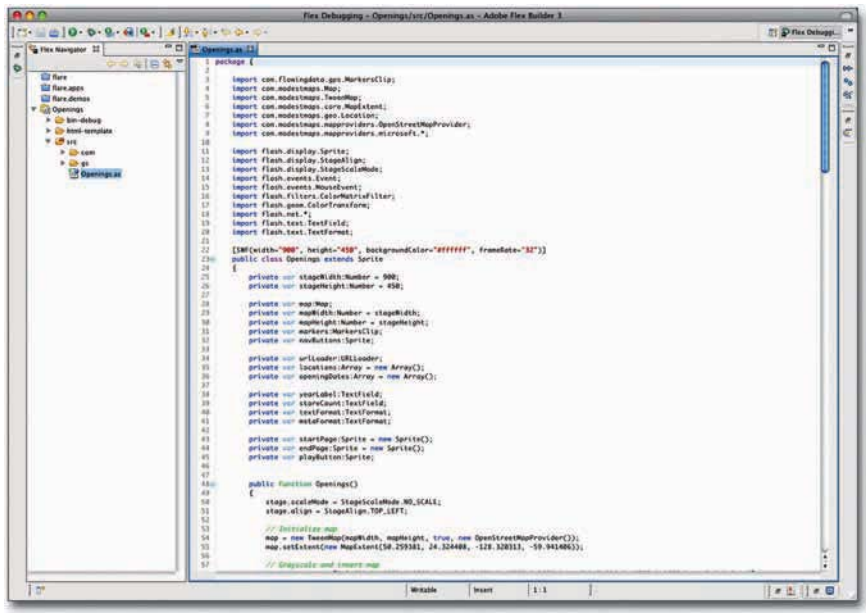


Рис. 8.39. Рабочая область после импортирования

Первая часть импортирует классы из пакета Modest Maps, а вторая импортирует объекты отображения и классы событий, поддерживаемые Flash. В данный момент названия отдельных классов нас не особо интересуют. Когда вы начнете их использовать, то сами разберетесь. Однако важно обратить внимание на то, что паттерн присваивания имен в первой части соответствует структуре директории: сначала идет `com`, затем `modestmaps` и в конце `Map`. Именно в такой очередности вы и будете по большей части импортировать классы, когда начнете сами писать коды на языке ActionScript.

Выше строки `public class Openings extends Sprite` инициализированы несколько переменных компилированного Flash-файла: ширина (width), высота (height), цвет фона (background color) и частота смены кадров (frame rate).

```
[SWF(width="900", height="450", backgroundColor="#ffffff", frameRate="32")]
```

Далее, после объявления класса вам необходимо определить некоторые переменные и инициализировать объект `Map` (то есть карту).

```
private var stageWidth:Number = 900;
private var stageHeight:Number = 450;
private var map:Map;
private var mapWidth:Number = stageWidth;
private var mapHeight:Number = stageHeight;
```

Теперь между скобками функции `Openings()` вы можете с помощью Modest Maps создать вашу первую интерактивную карту.

```
stage.scaleMode = StageScaleMode.NO_SCALE;
stage.align = StageAlign.TOP_LEFT;
```

```
// Initialize map
map = new TweenMap(mapWidth, mapHeight, true, new OpenStreetMapProvider());
map.setExtent(new MapExtent(50.259381, 24.324408, -128.320313, -59.941406));
addChild(map);
```

Как и во время работы в программе Illustrator, вы можете рассматривать интерактивную графику целиком как набор слоев. В ActionScript и Flash первый слой — это сцена (stage). Вы ее устанавливаете так, чтобы не масштабировать объекты, когда будете «приближать» ее. Сцену вы выравниваете по верхнему левому краю (top left). Затем инициализируете карту с помощью *mapWidth* и *mapHeight*, которые вы специфицировали в переменных, включаете интерактив и используете для карты тайлы из OpenStreetMap. Устанавливая размер карты, как указано в верхнем фрагменте кода, вы тем самым ограничиваете ее рамками Соединенных Штатов.

Числа в `MapExtent()` — это координаты долготы и широты, которые задают ограничивающий прямоугольник для того региона мира, который вы хотите показать. Первое и третье число — это широта и долгота верхнего левого угла карты, а второе и четвертое число — широта и долгота нижнего правого угла.

И наконец, с помощью `addChild()` вы добавляете к сцене саму карту. На рис. 8.40 представлен результат, который вы получите после компиляции кода и до добавления к карте фильтров. Вы можете или нажать кнопку Play в верхнем левом углу Flex Builder, или из главного меню выбрать Run → Run Openings.

Рис. 8.40. Простая карта с использованием тайлов OpenStreetMap

Когда вы запустите Openings, результат откроется на экране в том браузере, который установлен у вас по умолчанию. На карте пока еще ничего нет, но вы можете ее выбрать и перетащить, что уже круто. А еще, если вам эти тайлы не очень нравятся, вы можете использовать дорожные карты Microsoft (рис. 8.41) или гибридные карты Yahoo (рис. 8.42).

Рис. 8.41. Простая карта с использованием дорожной карты Microsoft

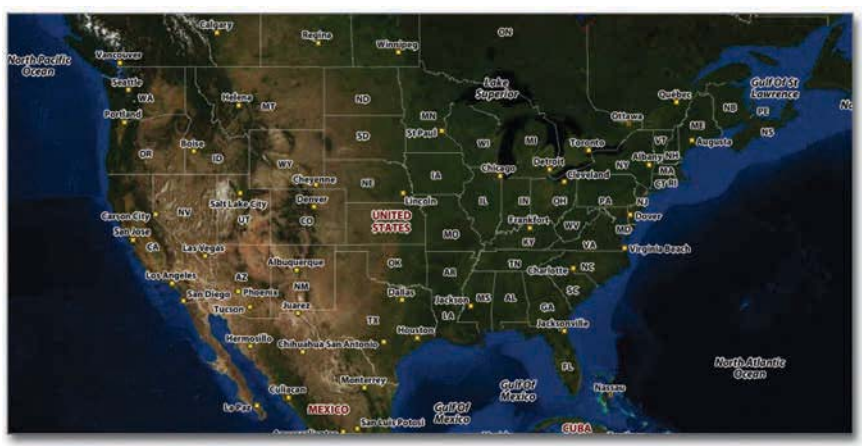


Рис. 8.42. Простая карта с использованием гибридной карты Yahoo

Еще вы можете поэкспериментировать с цветами карты, применяя разные фильтры. Например, можете перевести карту в шкалу оттенков серого, разместив приведенные ниже строки сразу после фрагмента кода, который вы только что написали. Массив mat имеет длину 20 и принимает значения от 0 до 1. Каждое значение указывает на то, сколько красной, зеленой, синей и альфа-компоненты получит каждый пиксель.

```
var mat:Array = [0.24688,0.48752,0.0656,0,44.7,0.24688,0.48752,
   0.0656,0,44.7,0.24688,0.48752,0.0656,0,44.7,0,0,0,1,0];
var colorMat:ColorMatrixFilter = new ColorMatrixFilter(mat);
map.grid.filters = [colorMat];
```

▶ Если вы захотите, то можете использовать также и свои собственные тайлы. На сайте Modest Maps вы найдете очень хорошее руководство, обучающее, как их можно сделать.

▶ Дополнительную информацию о том, как пользоваться цветовыми матрицами для кастомизации объектов в ActionScript, можно найти по адресу `http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/flash/filters/ColorMatrixFilter.html`.

Как видно на рис. 8.43, карта уже целиком окрашена в серые тона, что, возможно, покажется вам вполне подходящим фоном для тех данных, которые вы собираетесь на ней расположить. Таким образом сама карта не будет конкурировать с данными в борьбе за читательское внимание.

Рис. 8.43. Карта в серой цветовой шкале после применения фильтра

А еще вы можете инвертировать цвета с помощью `color transform`.

```
map.grid.transform.colorTransform =
   new ColorTransform(-1,-1,-1,1,255,255,255,0);
```

Белое было преобразовано в черное, а черное — в белое, как показано на рис. 8.44.

Рис. 8.44. Черно-белая карта после инвертирования цветов

Чтобы сформировать кнопки изменения масштаба изображения, сначала необходимо написать функцию, создающую эти самые кнопки. Возможно, вы ожидали, что для такого рода целей тоже

существует простой способ по умолчанию, однако правда состоит в том, что для выполнения подобных задач все еще приходится немного заниматься программированием. Описание функции makeButton() находится в конце класса Openings.

```
public function makeButton(clip:Sprite, name:String, labelText:String,
action:Function):Sprite
{
   var button:Sprite = new Sprite();
   button.name = name;
   clip.addChild(button);

   var label:TextField = new TextField();
   label.name = 'label';
   label.selectable = false;
   label.textColor = 0xffffff;
   label.text = labelText;
   label.width = label.textWidth + 4;
   label.height = label.textHeight + 3;
   button.addChild(label);

   button.graphics.moveTo(0, 0);
   button.graphics.beginFill(0xFDBB30, 1);
   button.graphics.drawRect(0, 0, label.width, label.height);
   button.graphics.endFill();

   button.addEventListener(MouseEvent.CLICK, action);
   button.useHandCursor = true;
   button.mouseChildren = false;
   button.buttonMode = true;

   return button;
}
```

Далее вам нужно создать еще одну функцию, которая будет использовать вышеприведенную и начертит нужные вам кнопки. Следующий код разместит в нижнем левом углу вашей карты две кнопки, используя makeButton(), — для выведения изображения крупным планом (zoom in) и для его уменьшения (zoom out).

```
// Draw navigation buttons
private function drawNavigation():void
{
   // Navigation buttons (zooming)
   var buttons:Array = new Array();
   navButtons = new Sprite();
```

```
        addChild(navButtons);
        buttons.push(makeButton(navButtons, 'plus', '+', map.zoomIn));
        buttons.push(makeButton(navButtons, 'minus', '-', map.zoomOut));
        var nextX:Number = 0;
        for(var i:Number = 0; i < buttons.length; i++) {
            var currButton:Sprite = buttons[i];
            Sprite(buttons[i]).scaleX = 3;
            Sprite(buttons[i]).scaleY = 3;
            Sprite(buttons[i]).x = nextX;
            nextX += 3*Sprite(buttons[i]).getChildByName('label').width;
        }
        navButtons.x = 2; navButtons.y = map.height-navButtons.height-2;
    }
```

Но, поскольку это функция, код ее не будет выполнен, пока вы ее не вызовете. А потому в `Openings()` — функцию-конструктор — сразу под фильтрами добавьте `drawNavigation()`. Теперь вы можете увеличивать и уменьшать масштаб интересующих вас регионов, как показано на рис. 8.45.

Рис. 8.45. Карта с работающим зумом

Это все, что вам нужно для базового варианта карты. Вы отобрали тайлы, присвоили значения переменным и включили интерактивный режим.

Добавьте метки

Следующий шаг предусматривает загрузку данных о местоположении магазинов Walmart и создание метки для каждого новооткрытого магазина. Приведенный ниже код в конструкторе загружает XML-файл с URL. Когда загрузка файла будет завершена, вызывается функция под названием `onLoadLocations()`.

```
   var urlRequest:URLRequest =
      new URLRequest('http://projects.flowingdata.com/walmart/walmarts_new.xml');
   urlLoader = new URLLoader();
   urlLoader.addEventListener(Event.COMPLETE, onLoadLocations);
   urlLoader.load(urlRequest);
```

Вполне очевидно, что теперь следует создать функцию `onLoadLocations()`. Она прочитывает XML-файл и сохраняет данные в массивах так, чтобы впоследствии ими можно было легко пользоваться. Однако прежде чем вы это сделаете, вам необходимо инициализировать еще несколько переменных после *navButtons*.

```
private var urlLoader:URLLoader;
private var locations:Array = new Array();
private var openingDates:Array = new Array();
```

Эти переменные используются в `onLoadLocations()`. Широта и долгота местоположений магазинов сохраняются в *locations*, а даты их открытия (в годовом формате) — в *openingDates*.

```
private function onLoadLocations(e:Event):void {
   var xml:XML = new XML(e.target.data);
   for each(var w:* in xml.walmart) {
      locations.push(new Location(w.latitude, w.longitude));
      openingDates.push(String(w.opening_date));
   }
   markers = new MarkersClip(map, locations, openingDates);
   map.addChild(markers);
}
```

Далее необходимо создать класс MarkersClip. Следуя структуре, о которой мы говорили выше, в директории com вы найдете директорию flowingdata, а в ней — gps. Таким образом класс MarkersClip окажется в com → flowingdata → gps. Это будет контейнер для хранения всех меток или, точнее, слоев с данными вашей интерактивной карты.

Как и раньше, вам потребуется импортировать классы, которые вы будете использовать. Обычно они добавляются в код по мере необходимости, но простоты ради вы можете добавить их все разом.

```
import com.modestmaps.Map;
import com.modestmaps.events.MapEvent;

import flash.display.Sprite;
import flash.events.TimerEvent;
import flash.geom.Point;
import flash.utils.Timer;
```

Первые два класса — из Modest Maps, а последние четыре — «родные». Затем вам нужно присвоить значения переменным — это делается как раз перед функцией `MarkersClip()`. И опять их, как правило, следует вставлять по мере необходимости, но можно добавить сразу, чтобы сократить процедуру и поскорее добраться до сути этого класса — до функций.

```
protected var map:Map;  // Base map
public var markers:Array;       // Holder for markers
public var isStationary:Boolean;

public var locations:Array;
private var openingDates:Array;

private var storesPerYear:Array = new Array();
private var spyIndex:Number = 0;       // Stores per year index
private var currentYearCount:Number = 0; // Stores shown so far
private var currentRate:Number;        // Number of stores to show
private var totalTime:Number = 90000; // Approx. 1.5 minutes
private var timePerYear:Number;
public var currentYear:Number = 1962; // Start with initial year

private var xpoints:Array = new Array(); // Transformed longitude
private var ypoints:Array = new Array(); // Transformed latitude

public var markerIndex:Number = 0;
private var starting:Point;
private var pause:Boolean = false;
public var scaleZoom:Boolean = false;
```

А теперь сохраните переменные, которые будут передаваться в класс, в конструкторе `MarkersClip()` и выполните кое-какие расчеты, например, задайте время появления данных за следующий год и координаты магазинов. Думайте об этой процедуре как о настройке.

Переменная *storesPerYear* содержит информацию о том, сколько магазинов открылось в течение определенного года. Например, в первый год открылся единственный магазин, а в следующем году не появилось ни одного нового. Когда вы станете использовать этот код с вашими собственными данными, вам необходимо будет соответственно обновить и *storesPerYear*. А еще вы могли бы написать функцию, которая бы сама рассчитывала количество и местоположение открывшихся магазинов за каждый год, и тем самым повысить пригодность вашего кода для повторного использования. Здесь же приводится жестко заданный код, и это сделано исключительно в целях упрощения примера.

```
this.map = map;

this.x = map.getWidth() / 2;
this.y = map.getHeight() / 2;
```

```
      this.locations = locations;
      setPoints();
      setMarkers();

      this.openingDates = openingDates;

      var tempIndex:int = 0;

      storesPerYear = [1,0,1,1,0,2,5,5,5,15,17,19,25,19,27,
         39,34,43,54,150,63,87,99,110,121,142,125,131,178,
         163,138,156,107,129,53,60,66,80,105,106,114,96,
         130,118,37];
      timePerYear = totalTime / storesPerYear.length;
```

В классе MarkersClip есть еще две функции: `setPoints()` и `setMarkers()`. Первая переводит координаты широты и долготы в *x*- и *y*-координаты, а вторая размещает метки на карте, при этом реально не показывая их. Далее следует определение функции `setPoints()`. Таким образом встроенная функция (предоставляемая Modest Maps) используется для определения *x* и *y* и сохранения новых координат в *xpoints* и *ypoints*.

```
   public function setPoints():void {
      if (locations == null) {
         return;
      }
      var p:Point;
      for (var i:int = 0; i < locations.length; i++) {
         p = map.locationPoint(locations[i], this);
         xpoints[i] = p.x;
         ypoints[i] = p.y;
      }
   }
```

Вторая функция, `setMarkers()`, использует точки, которые сохранила `setPoints()`, и ставит метки (маркеры) в соответствующих местах.

```
   protected function setMarkers():void
   {
      markers = new Array();
      for (var i:int = 0; i < locations.length; i++)
      {
         var marker:Marker = new Marker();
         addChild(marker);
         marker.x = xpoints[i]; marker.y = ypoints[i];
         markers.push(marker);
```

```
        }
    }
```

Данная функция использует также специальный класс, который вы найдете в com → flowingdata → gps → Marker.as. (Я говорю так, полагая, что вы скачали весь код целиком.) По сути это скрытая метка, и когда вы вызываете ее функцией `play()`, она «подсвечивается».

К настоящему моменту координаты местоположения магазинов и метки загружены в карту. Но если вы скомпилируете код и проиграете файл прямо сейчас, вы по-прежнему увидите лишь чистую карту. Требуется сделать следующий шаг: запустить выполнение цикла по меткам, чтобы они зажигались в нужное время.

Функция `playNextStore()` просто вызывает `play()` очередной метки и переходит в состояние готовности проиграть следующую за ней. Функции `startAnimation()` и `onNextYear()` используют таймеры и демонстрируют открытие каждого магазина с определенным шагом.

```
private function playNextStore(e:TimerEvent):void
{
    Marker(markers[markerIndex]).play();
    markerIndex++;
}
```

Если вы сейчас скомпилируете и запустите анимацию, вы получите точки (как на рис. 8.46), но зум и прокрутка работать не будут, а если попытаетесь подвигать карту туда-сюда либо увеличить или уменьшить масштаб изображения, пузырьки, представляющие различные магазины, будут оставаться неподвижными.

Рис. 8.46. Карта роста сети Walmart с неработающей прокруткой и зумом

В конструктор необходимо добавить слушатели событий, чтобы точки двигались вместе с картой. И каждый раз при запуске MapEvent со стороны Modest Maps станет вызываться

соответствующая функция, определенная в MarkersClip.as. Возьмем, к примеру, первую строчку приведенного ниже кода: когда пользователь щелкает по кнопке зума на карте, вызывается функция `onMapStartZooming()`.

```
this.map.addEventListener(MapEvent.START_ZOOMING,onMapStartZooming);
this.map.addEventListener(MapEvent.STOP_ZOOMING,onMapStopZooming);
this.map.addEventListener(MapEvent.ZOOMED_BY,onMapZoomedBy);
this.map.addEventListener(MapEvent.START_PANNING,onMapStartPanning);
this.map.addEventListener(MapEvent.STOP_PANNING,onMapStopPanning);
this.map.addEventListener(MapEvent.PANNED,onMapPanned);
```

После этого вы получите окончательный вариант карты, представленный на рис. 8.47.

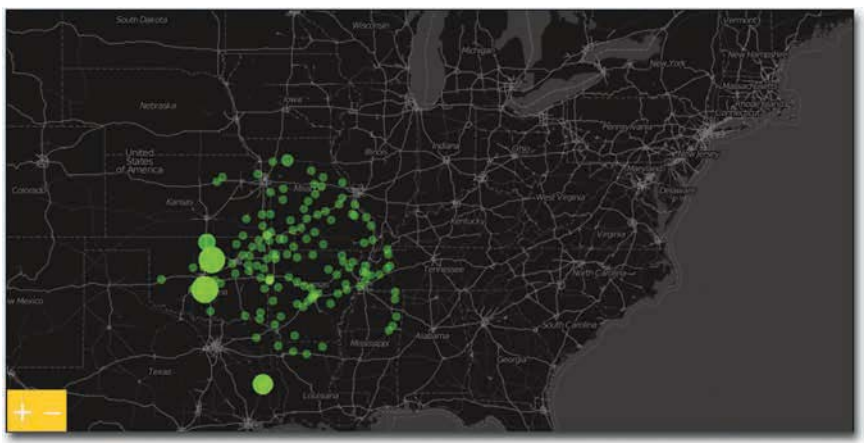


Рис. 8.47. Полностью интерактивная карта роста, демонстрирующая открытие магазинов Walmart

История с открытием новых магазинов Walmart — это история естественного, постепенного развития. Все началось с одного-единственного магазина, и затем происходило медленное распространение вширь. Но так бывает отнюдь не всегда. Например, рост компании Target не выглядит таким плавным и продуманным. Рост Costco не такой резкий, возможно, потому, что у него и магазинов меньше, но, похоже, в его стратегию входит освоение сначала двух побережий и уже затем — проникновение вглубь страны.

В любом случае карты роста — очень интересный и увлекательный метод визуализации данных. Они способны тормошить воображение людей и возбуждать у них желание узнать о том, например, как шло развитие McDonald's или Starbucks. И теперь, когда вы уже располагаете кодом, вам будет намного проще удовлетворять это любопытство. Самое сложное — найти данные.

Закругляясь

Карты — это довольно сложный способ визуализации, потому что помимо данных вам приходится иметь дело и с географическими координатами. Но так как карты удивительно интуитивны, они считаются также и очень плодотворным способом визуализации, причем как тогда, когда необходимо представить данные другим людям, так и когда речь идет о самостоятельном исследовании, требующем более глубокого анализа, чем можно провести с помощью статистических графиков.

Как вы поняли из примеров в этой главе, существует множество возможностей для визуализации пространственных данных. Даже обладая лишь некоторыми базовыми умениями, можно визуализировать самые различные наборы данных и рассказывать самые разные интересные истории. Но это лишь верхушка айсберга. Что я имею в виду? Вы только подумайте: люди годами обучаются в колледжах, а затем продолжают учиться и работать в магистратуре и докторантуре, чтобы получить ученую степень в области картографии и географии, так что можете себе представить, сколько всего еще кроется там, под поверхностью. Но вы уже умеете играть с картограммами и сортировать географические регионы в соответствии с некими количественными показателями; вы освоили технологию, как можно придать им интерактивности с помощью Flash; а еще вы научились комбинировать карты с диаграммами для более детального и наглядного представления данных.

Сегодня онлайн-карты получили очень широкое распространение, и их популярность будет только расти по мере совершенствования браузеров и других инструментов. В примере с картой роста мы использовали ActionScript и Flash, но то же самое можно реализовать и с помощью JavaScript. Какой именно инструмент выбрать — зависит от поставленных перед вами задач. А если не имеет значения, каким инструментом пользоваться, тогда лучше отдать предпочтение тому, с которым вы себя чувствуете более уверенно. Самое важное — вне зависимости от того, какое программное обеспечение будет использоваться, — это логика. Синтаксис может меняться. Важно то, что вы станете делать с данными и какую историю расскажете.

Прицельный дизайн

Когда вы занимаетесь самостоятельным изучением данных, вам не нужно думать о сторителлинге. В конце концов, вы и есть автор истории. Однако в тот момент, когда вы решите использовать эти данные, чтобы донести некую информацию до других — будь то один человек, или несколько тысяч, или даже миллионы, — голой диаграммы окажется уже недостаточно.

Конечно, вам бы хотелось, чтобы другие люди сами интерпретировали полученные результаты и, возможно, придумывали свои собственные истории. Однако читателям обычно бывает непросто сообразить, какие именно вопросы следовало бы задать, особенно если они ничего не знают о данных, которые видят перед собой. Это ваша работа и ваша ответственность — подготовить для них почву. От того, какой у ваших диаграмм или графиков будет дизайн, зависит то, как читатели станут интерпретировать данные, заложенные в их основу.

Подготовьте себя

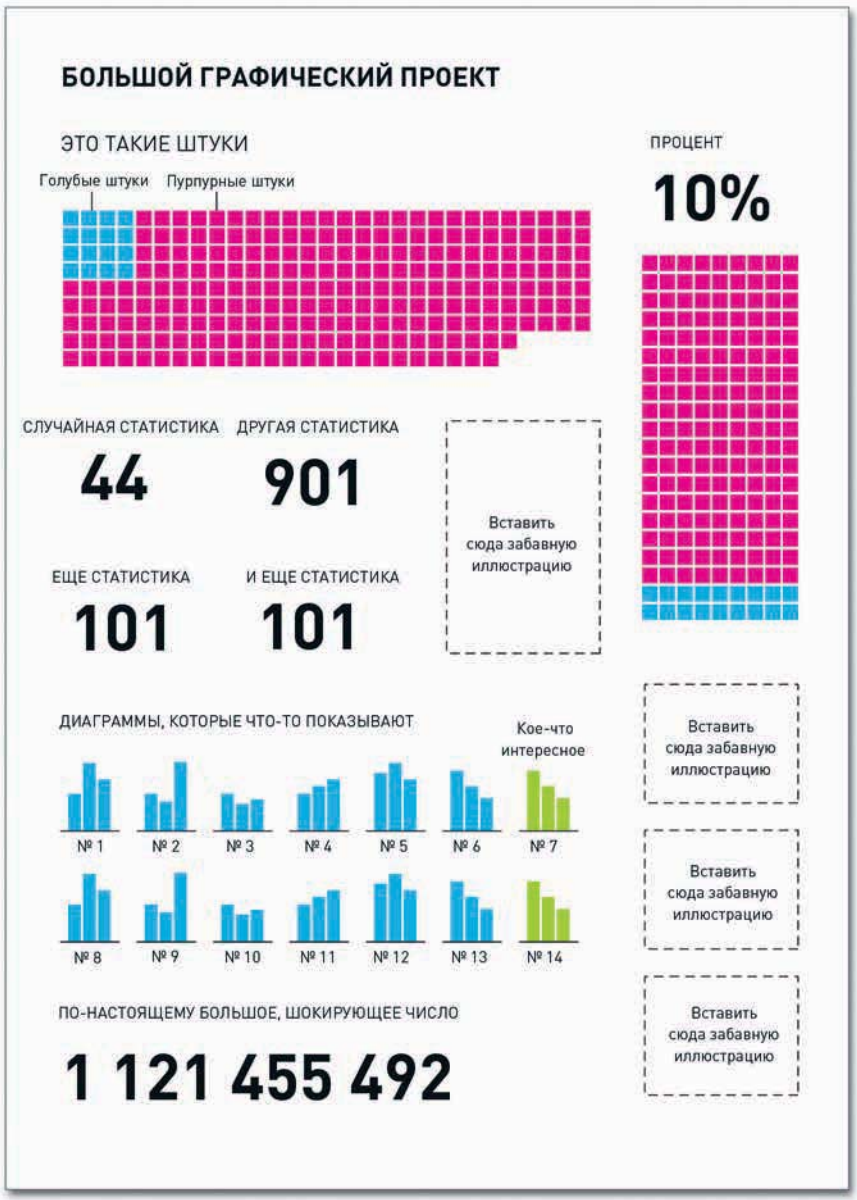


Рис. 9.1. Большой графический проект. Или делайте как надо, или вообще не беритесь

Чтобы рассказать хорошую историю с данными, вам необходимо знать их источник. Этой операцией, пожалуй, чаще всего пренебрегают во время подготовки инфографики. Когда начинаешь новый проект, обычно не терпится поскорее увидеть конечный результат. Вам хочется побыстрее создать нечто прекрасное, удивительное, восхитительное — и это замечательно. Только вам не добиться цели, если вы себе не представляете, что именно вы визуализируете. И в итоге вы получите что-то похожее на рис. 9.1. Как вы сможете объяснить интересные моменты в данных, если вы в этих данных сами не разбираетесь?

Изучите все числа и количественные показатели. Узнайте, откуда они взялись и как собирались, посмотрите, есть ли вообще в них смысл. Именно этот начальный процесс сбора данных и делает графику New York Times настолько качественной. В сети и в газетах вы видите лишь конечный результат, а весь труд, который вкладывается в графический объект еще до того момента, когда вычерчивается первый штрих, остается для вас невидимым. Хотя он отнимает огромное количество времени. Сбор и приведение данных в порядок — процедура намного более трудоемкая, нежели собственно создание графики.

А потому в следующий раз, когда вы получите некий набор данных, постарайтесь не бросаться сразу что-то вычерчивать. Так поступают только ленивцы, и по результату это всегда заметно. Выделите время для ознакомления с данными и изучения контекста чисел.

Вбейте несколько чисел в R, прочитайте имеющуюся сопроводительную документацию, чтобы понять, что и как измерялось, и посмотрите, нет ли в числах каких-нибудь отдельных странностей. Если что-то насторожит вас и вы не поймете, почему это именно так, вы всегда можете обратиться к источнику. Люди обычно рады узнать, что кто-то нашел применение опубликованным ими данным, и всегда готовы устранить ошибки, если таковые имеются.

После того как вы узнаете о своих данных все, что можно, вы будете готовы приступить к созданию графики. И лучше думать об этом вот в каком ключе. Помните тот момент в фильме «Малыш-каратист», когда Дэниел только начинает осваивать боевые искусства? Мистер Мияги поручает ему отполировать кучу автомобилей, надраить деревянные полы и покрасить забор, и Дэниел впадает в отчаяние, так как ему кажется: все это — бесполезные задания. Но затем он, конечно, выясняет, что и удар, и блокировка у него вдруг стали получаться сами собой, а все потому, что он отработал нужные движения. То же самое и с данными. Изучите все, что связанно с данными, и визуализация начнет получаться у вас естественно и непринужденно. Если же вы кино не видели, просто кивните в знак согласия. А затем добавьте «Малыш-каратист» к вашему списку фильмов для просмотра в Netflix.

ПОДСКАЗКА

Визуализация — это способ коммуницирования данных, так что найдите время изучить то, что составляет основу вашей графики. Иначе все кончится лишь потоком чисел.

Подготовьте своих читателей

Ваша работа в качестве дизайнера инфографики состоит в донесении того, что вы знаете, до вашей аудитории, которая, скорее всего, данных не знает, а потому, если не получит каких-нибудь объяснений или установок, может и не разглядеть в графике то, что видите вы. Я на опыте убедился и привык по умолчанию принимать, что люди выходят на мои работы вслепую; учитывая, что они нередко приходят по гиперссылкам из Facebook, Twitter и различных блогов, это, по сути, так и есть.

На рис. 9.2, например, представлен скриншот с анимированной карты, которую я однажды сделал. Если вы не видели ее ранее, вы вряд ли догадаетесь, что перед вами. В лучшем случае, памятуя о примерах, которые рассматривались в главе 8, вы можете подумать, что речь идет об открытии магазинов какой-нибудь сети.

▶ Увидеть анимированную карту в полном объеме можно на `http://datafl.ws/19n`.

На самом деле карта демонстрирует геопривязанные твиты, которые люди публиковали по всему земному шару во время инаугурации президента Барака Обамы, которая состоялась во вторник 20 января 2009 года в полдень по Восточному поясному времени. Анимация начинается с раннего утра понедельника и заканчивается в обед следующего за инаугурацией дня. С приближением события количество твитов в час увеличивается. Когда Соединенные Штаты засыпают, подключается Европа. И вот наступает четверг. На карте видно, как все больше и больше людей просыпается и начинает писать твиты. В какой-то момент следует бум — в ходе самого события волнение зашкаливает. Вы это легко заметите по рис. 9.3. Если бы я снабдил рис. 9.2 подобным комментарием, вы бы наверняка извлекли больше информации из увиденного.

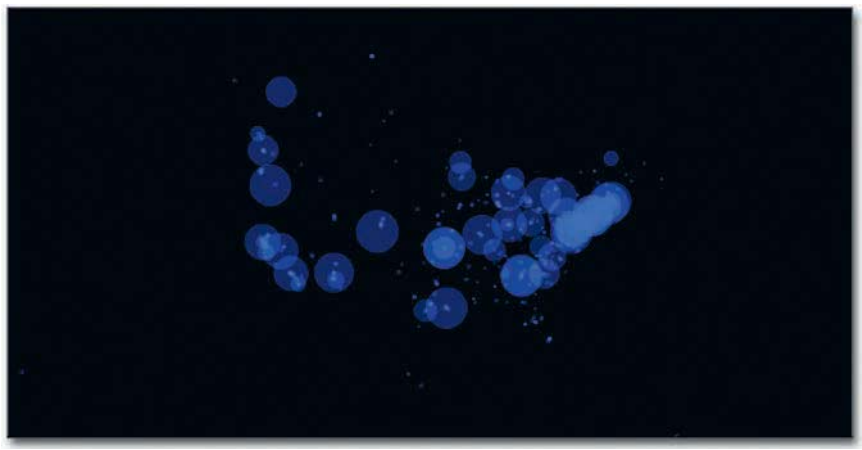


Рис. 9.2. Карта без заголовка и контекста

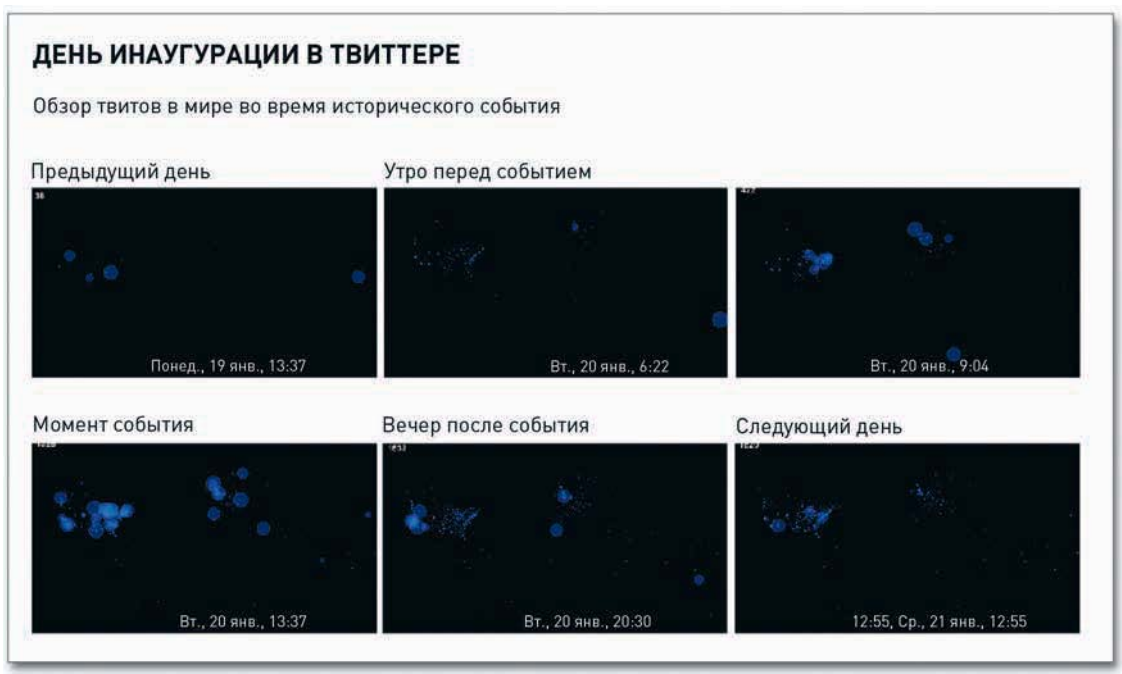


Рис. 9.3. Твиты во время инаугурации президента Барака Обамы

Вам не нужно писать целые эссе, чтобы сопровождать ими каждую вашу работу, но заголовок и кое-какие пояснения во вводном абзаце всегда бывают полезны. Не помешает также вставить ссылку где-нибудь в самом изображении, чтобы люди могли найти аннотацию, даже если они увидят вашу графику опубликованной на другом сайте. Иначе все быстро превратится

в игру в испорченный телефон, и прежде чем вы сообразите, что происходит, графика, в которую вы вложили столько сил и времени, станет служить иллюстрацией мыслей, в корне противоположных тем, которые вы хотели донести до аудитории. Интернет в этом плане обладает поистине сверхъестественными способностями.

Или вот другой пример. На рис. 9.4 представлен простой временной график, показывающий десять самых крупных утечек данных с 2000 по 2008 год.

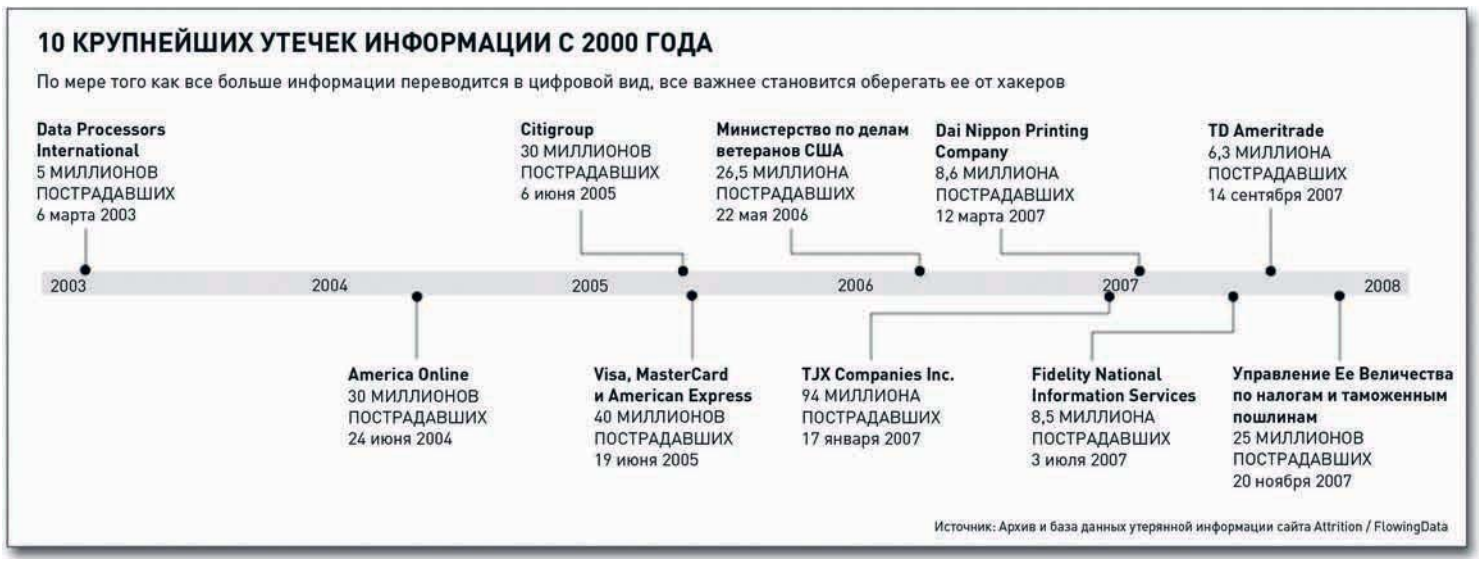


Рис. 9.4. Крупнейшие утечки информации с 2000 года

График совсем элементарный. На нем есть только 10 точек ввода данных, но когда я размещал его на сайте FlowingData, я хотел привлечь внимание к тому, что в период с 2000 по 2008 год утечки начали происходить со всевозрастающей частотой. Все кончилось тем, что график разошелся по Сети, а один из его вариантов был даже опубликован в журнале Forbes. И практически все разделяли вложенную в него мысль. Но я не думаю, что люди уделили бы графику столько внимания, если бы я не дополнил его констатацией этого простого факта.

Вывод: не принимайте по умолчанию, что ваши читатели в курсе всего или что они и сами разберутся в вашей графике. Такого не будет, особенно в Сети, где все привыкли, чуть что, сразу щелкать по следующей ссылке и переходить далее.

Однако это вовсе не означает, что люди не станут тратить время на изучение информации. Как вы, возможно, убедились, в блоге OkCupid часто печатаются довольно длинные посты, представляющие результаты тщательного анализа собранных данных об онлайн-знакомствах. Например, такие как «Лучшие вопросы для первого свидания» или «Математика красоты».

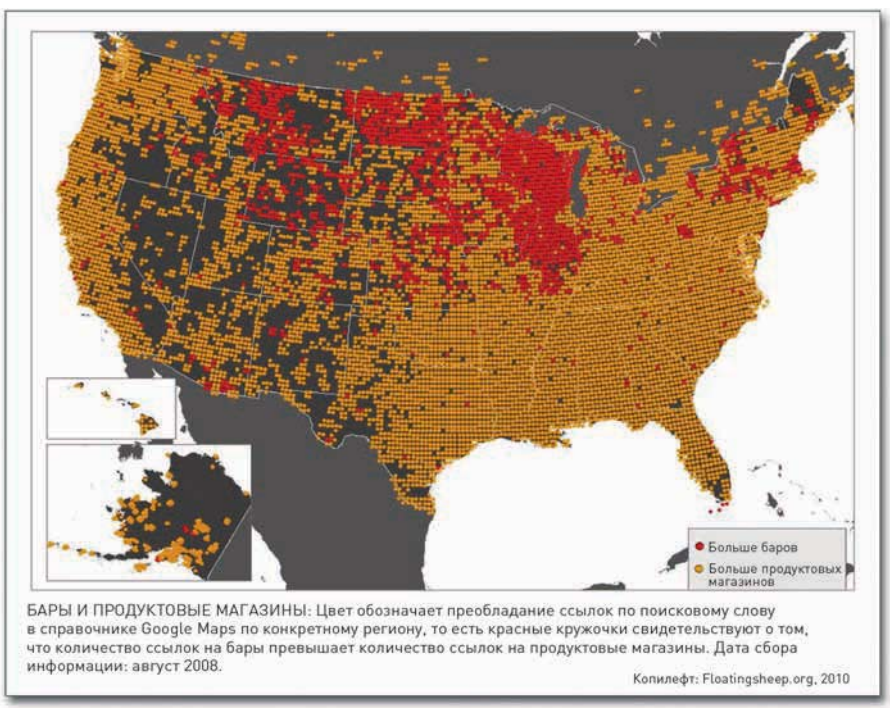
Но эти посты иногда собирают миллионы просмотров — людям интересно то, что ребята из OkCupid могут рассказать. Вдобавок к тоннам контекста, содержащегося в постах, читатели,

приходя в блог, привносят в него и свои крупицы контекста. Поскольку здесь собрана информация, связанная с отношениями с противоположным полом, людям легко проводить сравнения со своим собственным опытом. На рис. 9.5, например, представлена графика, демонстрирующая, что обычно нравится парням из Азии. Она взята из поста OkCupid, посвященного предпочтениям людей в зависимости от расы и пола. Ну-ка, ну-ка! Я же сам парень и я из Азии! Связь устанавливается мгновенно.

С другой стороны, если тема вашей графической работы — уровень загрязнения окружающей среды или общий мировой долг, тогда трудно ожидать, что широкая аудитория сама все поймет, так что вам придется хорошенько потрудиться и объяснить ей, что к чему.

Но иногда бывает и так, что, как бы вы ни старались, людям просто не нравится изучать информацию онлайн и они уходят со страницы. Например, я однажды разместил на FloatingSheep карту, на которой сравнивал количество баров в США с количеством

Рис. 9.5. Что нравится парням из Азии (на базе онлайн-профилей сайта знакомств OkCupid)



продуктовых магазинов (рис. 9.6). Красный цвет обозначает области, в которых баров больше, нежели продуктовых магазинов, а оранжевый — те, где ситуация обратная. Ребята из FloatingSheep назвали эту часть карты «пивной животик» Америки.

В конце поста я выразил сомнение по поводу точности карты и закончил словами: «Может, кто-нибудь из живущих в этом регионе озаботится и подтвердит, так ли это. Я готов к тому, что ваши комментарии окажутся полны опечаток и продраться через их смысл будет непросто. Ну, и амбре от них пойдет соответствующее». Каков вывод? Грубый юмор и сарказм в Сети воспринимаются не очень хорошо, особенно если люди не привыкли к вашему стилю.

Рис. 9.6. Где в США баров больше, чем продуктовых магазинов

На самом деле я не ожидал, что комментарии будут как-то пахнуть. И большинство людей шутку поняли, но оказалось и приличное количество обиженных жителей Висконсина. Как я говорил, Сеть — занятное место (в хорошем смысле слова).

Визуальные подсказки

В главе 1 вы увидели, как работает кодировка. По сути, у вас есть данные, и вы кодируете их с помощью геометрии, цвета или анимации. А затем читатели декодируют эти фигуры, цвета и движения, представляя себе их снова в виде чисел. Это и есть основа визуализации. Кодирование означает перевод на визуальный язык. Декодирование помогает взглянуть на данные под другим углом и разглядеть паттерны, которые вы бы не заметили, если б смотрели на таблицу.

Такой тип кодировки обычно довольно прямолинеен, так как основывается на математических правилах. Более длинные столбцы представляют более крупные значения, а кружочки меньшего размера представляют меньшие значения. И хотя в процессе вашему компьютеру приходится принимать массу решений, все равно именно от вас зависит выбор подходящей кодировки для конкретного массива данных.

На всех примерах, приведенных в предыдущих главах, вы могли убедиться, что хороший дизайн работает не только на эстетику. Он делает графику более понятной и способен менять отношение читателей к данным или к истории, которую вы пытаетесь им рассказать. Графики, создаваемые в R или Excel с настройками по умолчанию, смотрятся сыровато и похожи на штамповку. Это не всегда плохо. Этого вам может быть вполне достаточно, если вы пишете научный отчет или если ваша графика — всего лишь приложение к более важному труду. Тогда, наверное, вам действительно не стоит отвлекаться от того, на чем люди в действительности будут фокусироваться. На рис. 9.7 представлена столбцовая диаграмма, созданная без регулирования настроек. Она предельно проста.

Но если вы хотите сделать свою графику более заметной, вам для этого, возможно, будет достаточно просто добавить в нее цвет. Рис. 9.8 точно такой же, как предыдущий, с той лишь разницей, что цвета фона и заливки другие.

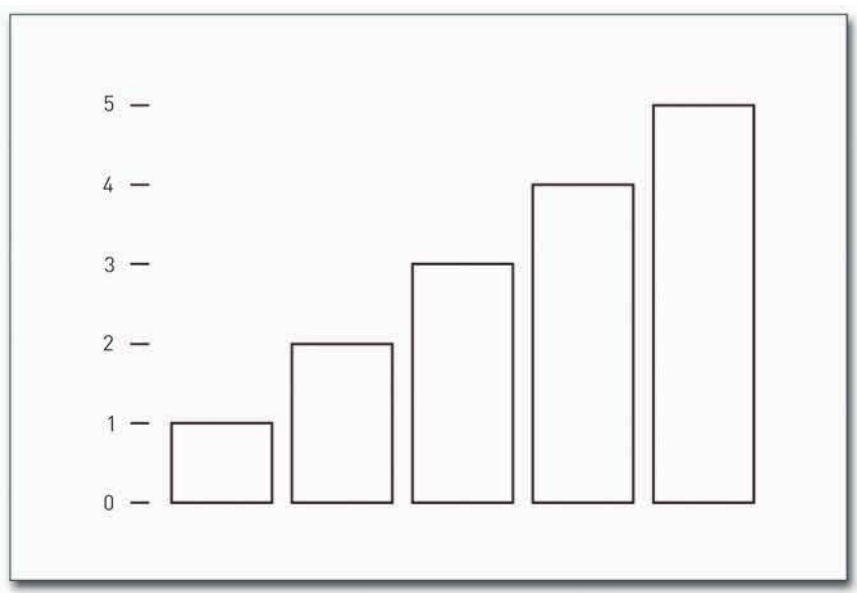


Рис. 9.7. Простая столбцовая диаграмма

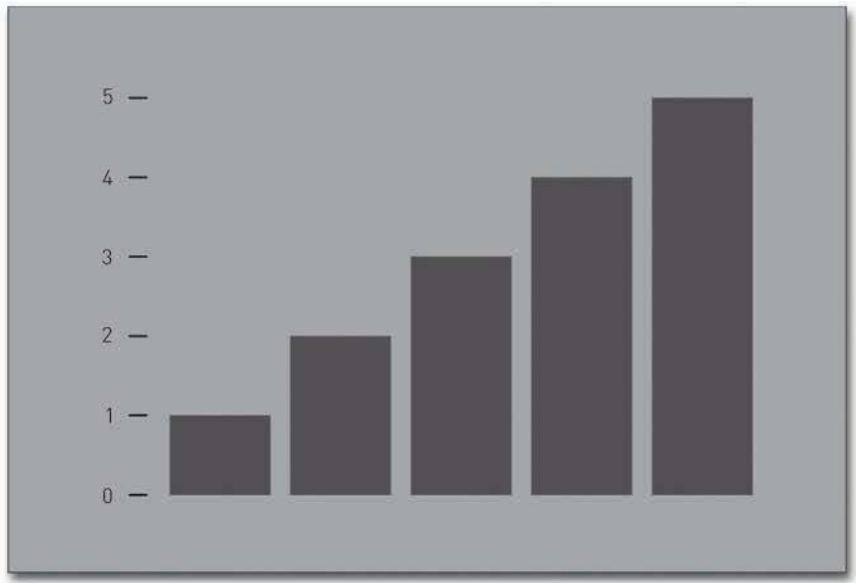


Рис. 9.8. Диаграмма по умолчанию с применением темной цветовой схемы

Более темные тона лучше применять с мрачными темами, а более яркие цветовые схемы создают радостное, беззаботное настроение (рис. 9.9).

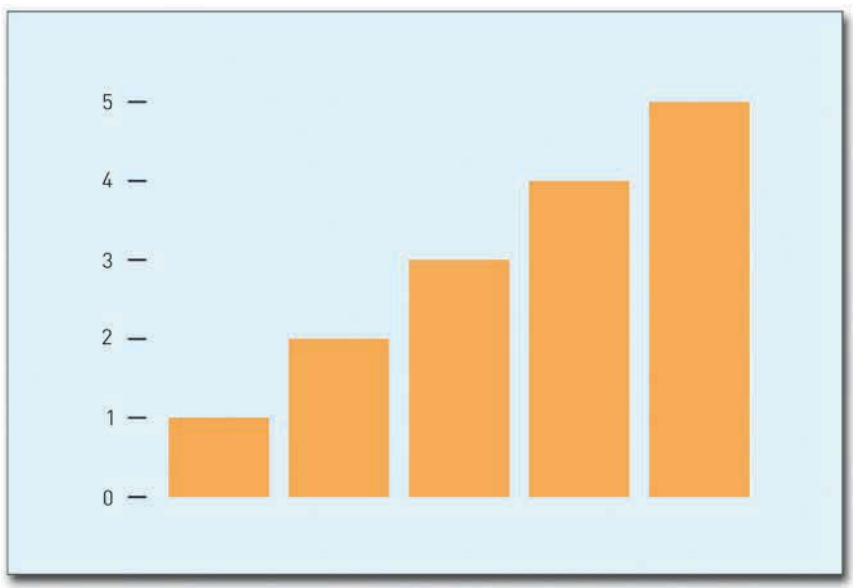


Рис. 9.9. Диаграмма по умолчанию со светлой цветовой схемой

Конечно, вам не обязательно пользоваться какой-нибудь однозначно говорящей цветовой темой. Вы можете выбрать нейтральную палитру, как показано на рис. 9.10.

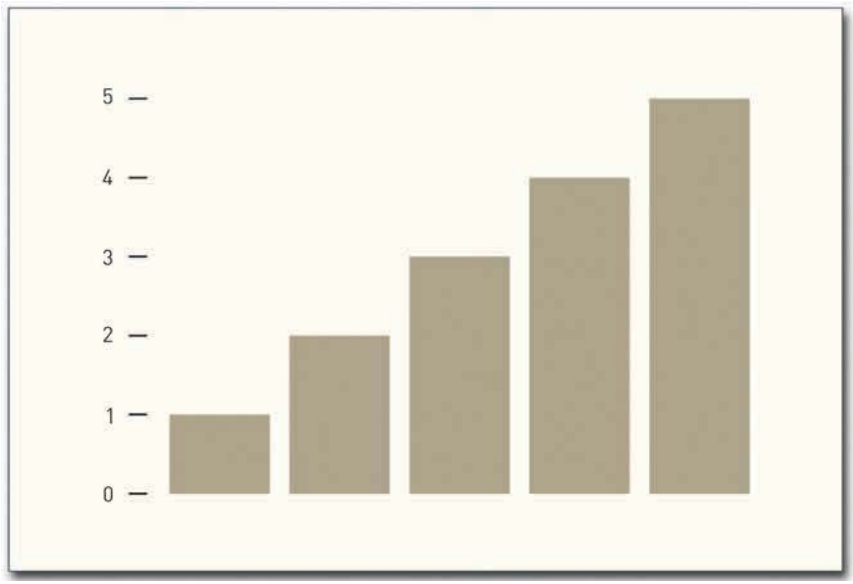


Рис. 9.10. Диаграмма по умолчанию с нейтральной цветовой схемой

Основная идея заключается в том, что в инфографике выбор цвета играет важную роль. Цвет способен вызывать (или не вызывать) эмоции, он помогает доносить информацию. И это ваша ответственность — выбрать цвета, которые правильно передадут ваше послание. Цвета должны соответствовать истории, которую вы пытаетесь рассказать. Как видно на рис. 9.11, простая смена цвета способна перевернуть смысл данных. Представленная на рисунке графика была разработана Дэвидом Маккэндлессом (David McCandless)* и дизайнерским дуэтом Always With Honor**. В ней исследуется значение цветов в различных культурах. Например, черное и белое нередко используются для обозначения смерти; в мусульманском мире и южноамериканской культуре для этого используются синий и зеленый соответственно.

Аналогичным образом вы можете менять и геометрию, чтобы взглянуть на данные под иным углом и увидеть другие их грани. Например, на рис. 9.12 представлена штабельная столбцовая диаграмма, произвольно сгенерированная с помощью Data-Driven Documents*** Майка Бостока (Mike Bostock). У нее прямые линии и хорошо выделяющиеся точки, а также максимумы и минимумы.

---

* Презентацию Д. Маккэндлесса «Красота визуализации данных» на TED 2010 можно посмотреть по адресу http://www.ted.com/talks/lang/ru/david_mccandless_the_beauty_of_data_visualization.html. *Прим. пер.*

** В дизайнерский дуэт Always With Honor («Всегда с честью») входят Тайлер Ланг (Tyler Lang) и Эльза Чавес (Elsa Chaves). *Прим. пер.*

*** Data-Driven Documents (или D3) — JavaScript-библиотека для визуализации данных. *Прим. пер.*

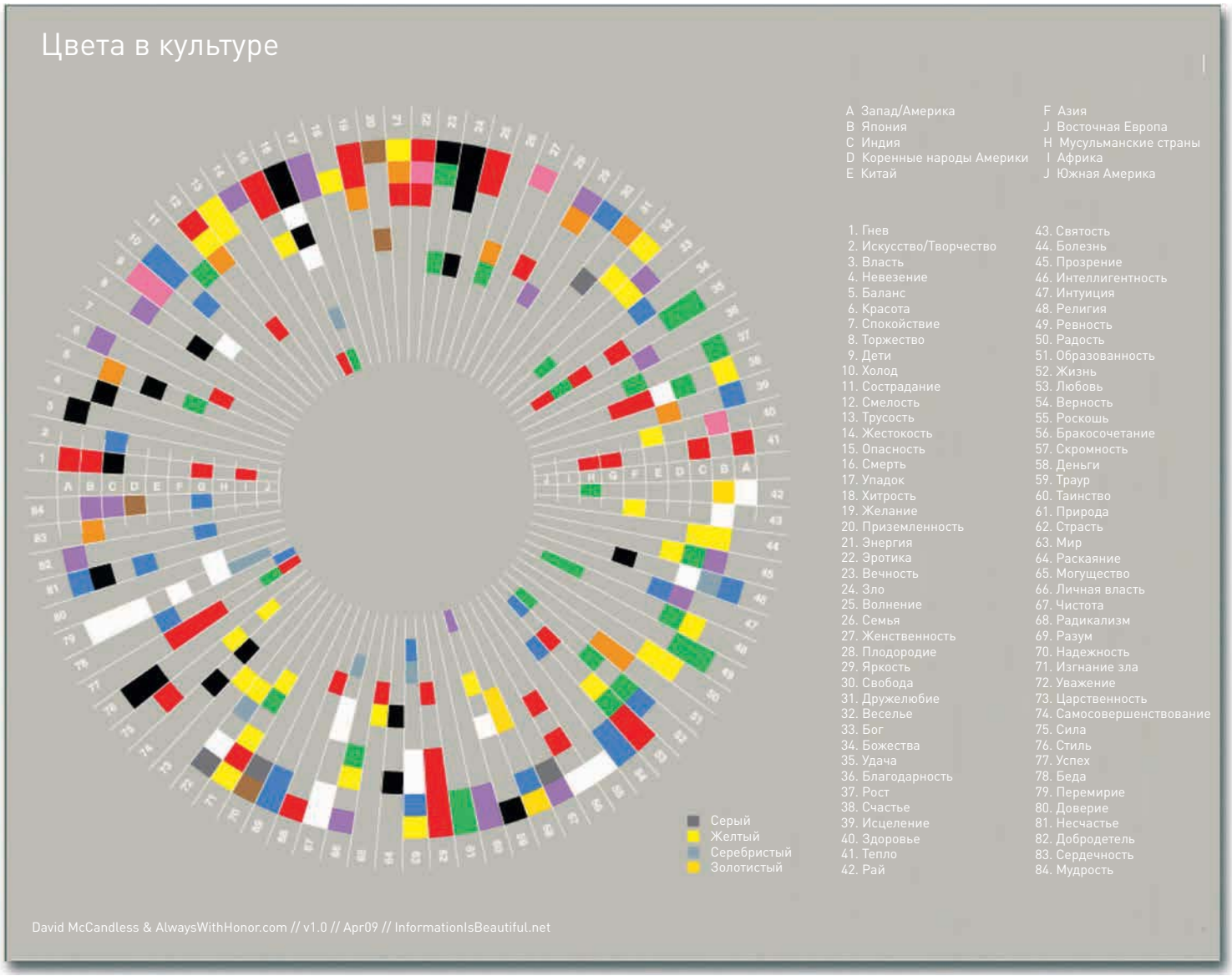


Рис. 9.11. Цвета в различных культурах (согласно Дэвиду Маккэндлессу и Always With Honor)

А если для демонстрации тех же чисел вы бы решили использовать потоковый график, как показано на рис. 9.13, то данные предстали бы перед вами в совершенно ином свете. В этом случае отчетливо возникает ощущение свободного течения и непрерывности, а вместо пиков и спадов у нас теперь есть уплотнения и расширения. И в то же время геометрия этих двух

графических объектов по сути схожая. Потоковый график, по большому счету, — это сглаженная штабельная столбцовая диаграмма, горизонтальная ось которой находится в центре, а не внизу.

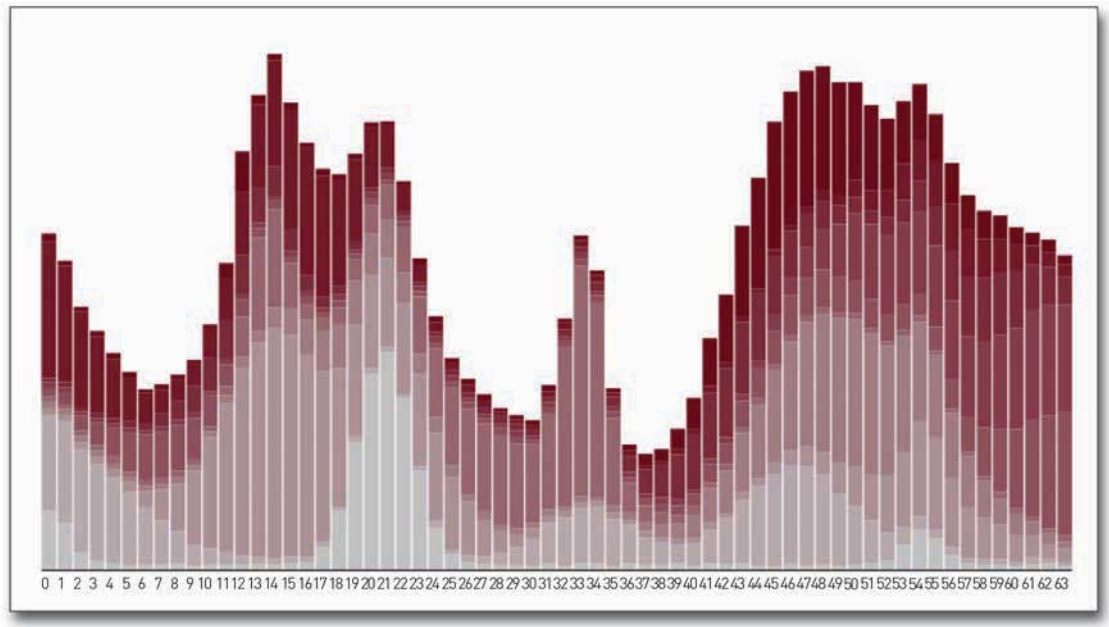


Рис. 9.12. Произвольно сгенерированная штабельная столбцовая диаграмма

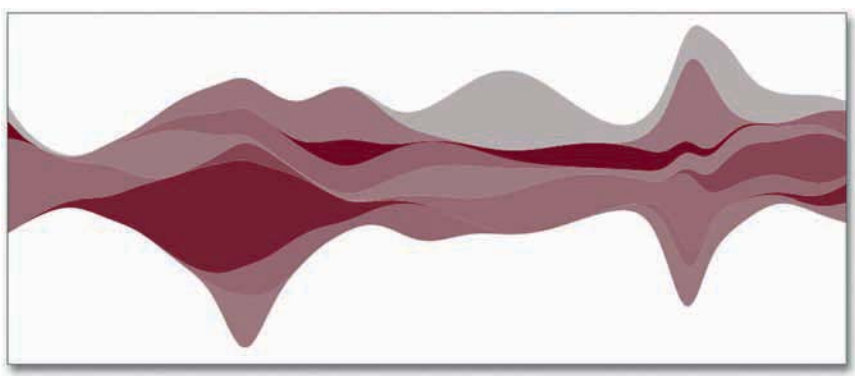


Рис. 9.13. Произвольно сгенерированный потоковый график

▶ Дополнительную информацию о потоковых графиках вы можете почерпнуть из публикации Ли Байрона (Lee Byron) и Мартина Ваттенберга (Martin Wattenberg) «Штабельные диаграммы — геометрия и эстетика». Если захотите сами создать подобный график, можете воспользоваться каким-либо из пакетов вроде Protovis и D3.

Иногда контекст может возникнуть просто из того, как вы организуете различные формы и цвета. На рис. 9.14 представлена моя графическая работа, которую я создал забавы ради, чтобы отметить наступление праздника. В верхней ее части представлены ингредиенты для маринования индейки, а в нижней — чем ее посыпать во время жарки.

Рис. 9.14. Рецепт рождественской индейки

Итог: по большому счету, визуализация — это превращение данных (будь то числа, текст, категории или любое другое множество единиц) в визуальные элементы. Некоторые визуальные коды работают лучше других. Их применимость варьирует и зависит также от конкретного набора данных. Метод, который считается в корне ошибочным для визуализации одного набора, может оказаться идеально подходящим для другого. Со временем и опытом вы научитесь быстро определять, какой метод лучше всего подходит для ваших конкретных целей.

Сто́ящая визуализация

Хотя люди занимаются визуализацией данных, составлением карт и вычерчиванием графиков и диаграмм на протяжении нескольких веков, только в последние десятилетия ученые всерьез взялись за исследование того, что именно работает, а что — нет. В этом смысле визуализация — относительно новая область знаний. Все еще нет консенсуса насчет того, что на самом деле представляет собой визуализация. Следует ли считать ее чем-то, генерируемым компьютером в соответствии с определенными правилами? И как быть, если к процессу создания графического объекта приложил руку также и человек — можно ли считать, что это уже не является визуализацией? И можно ли относить информационную графику к формам визуализации, или это совсем иная категория?

Если вы поищете ответы на такого рода вопросы в интернете, то найдете множество форумов, где обсуждаются сходства и отличия между визуализацией и инфографикой, а также публикации и эссе, пытающиеся дать определение тому, что такое визуализация. И всё всегда перетекает в бесконечные споры с доводами «за» и «против», а вопрос так и остается нерешенным. В процессе подобных дискуссий выкристаллизовываются также и различные критерии оценки того, какую графику следует считать плохой, а какую — хорошей.

Статистики и аналитики, например, обычно смотрят на визуализацию как на традиционную статистическую графику, которую они используют в своих анализах. Если диаграмма или интерактивный график не способствуют анализу, тогда они никчемны. Такие графические объекты считаются неудачными. Однако если вы поговорите о тех же самых диаграммах и графиках с дизайнерами, они могут посчитать их очень даже удачными — наглядно демонстрирующими данные и подающими их весьма увлекательно.

Что делать? А просто объединить этих людей или хотя бы собирать их в одной комнате почаще. Людям с аналитическим складом ума есть много чему поучиться у дизайнеров в том, как представлять данные более наглядно и понятно, а дизайнеры могут перенять у своих коллег-аналитиков привычку копать в данных поглубже.

Я не пытаюсь дать определение тому, что такое визуализация, поскольку моя работа не зависит от дефиниций. Я считаюсь в первую очередь с аудиторией и с данными, находящимися передо мной, и спрашиваю себя, какой смысл несет в себе полученная в итоге графика. Говорит ли она мне то, что я хочу узнать? Ели да — то замечательно. Если нет — тогда я возвращаюсь к чистому листу и начинаю думать, как можно улучшить графику так, чтобы она отвечала на вопросы,

которые у меня возникают по поводу этих данных. В конечном счете все сводится к тому, какие цели вы ставите перед графикой, какую историю хотите рассказать и кому вы будете говорить. Если вы учтете все вышесказанное, то вы молодец.

Закругляясь

Многие люди смотрят на дизайн как на способ сделать графику красивой. Конечно, в этом есть доля истины, но задача дизайна отнюдь не сводится к наведению «марафета». Задача дизайна — сделать графику более читабельной, понятной и полезной. Вы способны помочь людям понять данные лучше, глубже, чем если бы они разглядывали диаграмму, созданную по умолчанию. Вы можете почистить «шумы», подчеркнуть самое важное и даже вызвать эмоциональный отклик. Графическое представление данных бывает забавным, увлекательным, информативным. Иногда все может ограничиться и только первым вариантом — это зависит от ваших целей, но как бы то ни было, чем бы вы ни занимались — визуализацией, инфографикой или data-артом, — пусть всегда именно данные руководят вашей работой.

Иногда, особенно когда вам приходится иметь дело с огромными объемами информации, бывает и так, что вы не знаете, с какой стороны подступиться к работе. Тогда лучше всего начинать с вопроса: что именно вы хотите узнать. Вы пытаетесь найти сезонные паттерны? Или изучить зависимости между множеством переменных? Или найти выбросы? Или исследовать пространственные зависимости? Приглядитесь к своим данным и подумайте, сможете ли вы найти в них ответ на свой вопрос. Если вам недостает нужных данных, тогда найдите их сами.

Когда у вас будут все данные, вы сможете применить знания и умения, полученные на примерах этой книги, чтобы рассказать интересную историю. Но не останавливайтесь на достигнутом. Смотрите на освоенный вами материал лишь как на фундамент. В основе всех ваших любимых графических работ лежит тип данных и метод визуализации, с которыми вы уже знаете как работать. Так встаньте на этот фундамент и дерзайте, создавайте еще более продвинутую и сложную графику. Добавьте интерактива, комбинируйте диаграммы или дополните графики фотографиями и словами, чтобы обогатить их контекст.

Помните: данные — это отражение реальной жизни. Когда вы визуализируете их, вы визуализируете то, что происходит вокруг вас и в мире. Вы можете увидеть, что случается на микроуровне с отдельными людьми, или посмотреть, что творится в гораздо более широких масштабах — в масштабах вселенной. Изучайте данные, и вы сможете рассказывать истории, о которых люди даже не догадываются, но которые им не терпится услышать. Сегодня у нас в распоряжении есть такие объемы числовой информации, какие раньше люди себе и представить не могли, и все хотят узнать, о чем говорят данные. Теперь вы можете им это сказать. Наслаждайтесь.