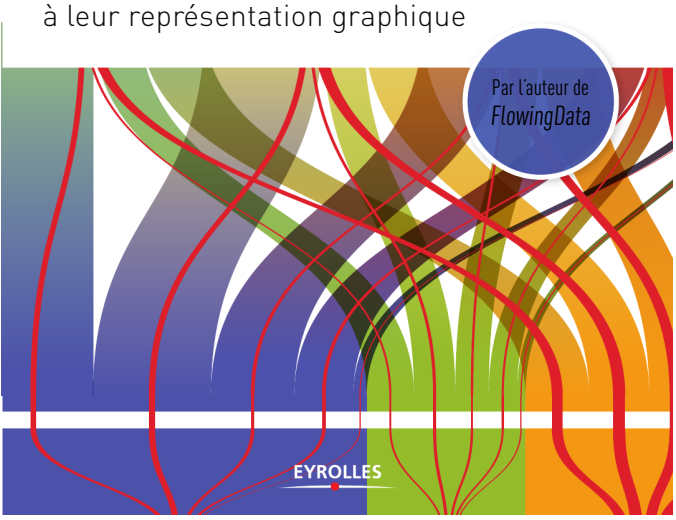
Nathan Yau

DATA VISUALISATION

De l'extraction des données
à leur représentation graphique



EYROLLES

DATA VISUALISATION

De l'extraction des données à leur représentation graphique

L'essor des nouvelles technologies et du Web a extraordinairement accéléré et simplifié la collecte, le stockage et l'accès aux données. Analysées et présentées de façon explicite et sensée, elles contribuent à faciliter la prise de décision, partager les connaissances et les idées, porter à un regard plus objectif sur le monde.

La data visualisation est ainsi devenue une discipline à part entière, outil privilégié des datajournalistes, scientifiques, statisticiens, ingénieurs, graphistes, designers, chercheurs en sciences de l'information, spécialistes du marketing. Pionnier de cette approche innovante, Nathan Yau présente dans cet ouvrage les meilleurs moyens de collecter, d'explorer, d'analyser et de représenter de façon créative de larges ensembles de données.

Nathan Yau est diplômé en statistiques à l'Université de Californie, Los Angeles, et prépare un doctorat sur la visualisation et les données personnelles. Depuis 2007, il conçoit et crée des graphiques pour FlowingData, site web de référence dédié aux statistiques et à la conception de visualisations de données. Il collabore régulièrement avec *The New York Times*, CNN, Mozilla et SyFy.

AU SOMMAIRE

Code éditeur : G13599
ISBN : 978-2-212-13599-2
www.editions-eyrolles.com

Illustration de couverture : *"The poverty-net threat"* © Mario Porpora

DATA VISUALISATION

Chez le même éditeur

M. Lima, *Cartographie des réseaux – L'art de représenter la complexité*, 2013, 272 pages.

D. Rosenberg, A. Grafton, *Cartographie du temps – Des frises chronologiques aux nouvelles timelines*, 2013, 272 pages.

J. Gray, L. Bounegru, L. Chambers, N. Kayser-Bril, *Guide du datajournalisme – Collecter, analyser et visualiser les données*, 2013, 220 pages.

M. Briggs, *Manuel du journalisme web – Blogs, réseaux sociaux, multimédia, info mobile*, 2013, 280 pages.

Nathan Yau

DATA VISUALISATION

De l'extraction des données à leur représentation graphique

Traduction et adaptation Xavier Guesnu
Validation technique Jérôme Cukier

EYROLLES

Éditions Eyrolles
61, bd Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

À Bea, *ma femme bien aimée*

Remerciements

Ce livre n'aurait pas pu voir le jour sans le travail des spécialistes en données qui m'ont précédé et qui ont développé, et développent encore, les outils indispensables et ouverts pour que chacun puisse les utiliser. Les logiciels créés par ces généreux développeurs m'ont grandement facilité la vie et je suis sûr que ces derniers poursuivront leurs innovations.

Mes remerciements vont aussi aux lecteurs de FlowingData, grâce auxquels j'ai pu atteindre plus de personnes que je ne l'aurais jamais rêvé. Ils sont l'une des principales raisons de l'écriture de cet ouvrage.

Merci à Wiley Publishing, qui m'a permis de faire le livre que je souhaitais, et à Kim Rees qui qui m'a aidé à écrire un livre digne d'être lu.

Et enfin, je ne voudrais pas manquer de remercier mon épouse pour son soutien, ainsi que mes parents qui m'ont toujours encouragé à rechercher ce qui me rendait heureux.

Table des matières

Introduction

Les données n'ont rien de nouveau. L'être humain quantifie et classe des informations sous forme de tableaux depuis des siècles. Ces toutes dernières années cependant, tandis que j'écrivais pour mon site web consacré à la conception à la visualisation et aux statistiques, FlowingData, j'ai remarqué un extraordinaire essor, qui ne cesse de se poursuivre. Les améliorations apportées à la technologie ont grandement simplifié la collecte et le stockage de données, tandis que le Web permet d'y accéder en continu. Entre de bonnes mains, cette richesse des données peut être une mine d'informations pour aider à l'amélioration de la prise de décision, à une communication des idées plus claire et à un regard plus objectif sur le monde et sur soi-même.

À la mi-2009, le lancement de Data.gov par les États-Unis a marqué une étape importante dans la diffusion des données gouvernementales. Il s'agit d'un catalogue détaillé des données fournies par les agences fédérales et qui traduit la transparence des groupes et de l'administration. L'idée est que le citoyen sache comment le gouvernement dépense l'argent des impôts – jusque-là, l'administration évoquait plutôt une boîte noire. Un grand nombre de données de Data.gov étaient déjà accessibles sur les sites des agences éparpillés sur le Web, désormais elles sont rassemblées en un même lieu et leur mise en forme a été améliorée à des fins d'analyse et de visualisation. Les Nations unies en possèdent un équivalent, ou presque, avec UNdata ; peu après, le Royaume-Uni a lancé Data.gov.uk, tandis que des grandes villes comme New York, San Francisco et Londres ont aussi pris part à la mise à disposition de leurs données.

Les sites web collectifs se sont également orientés vers une plus grande ouverture grâce à des milliers d'API (*Application Programming Interface*) destinées à encourager et à inciter les développeurs à utiliser concrètement les données disponibles. Des applications telles que Twitter et Flickr en proposent certaines dont les interfaces utilisateur sont totalement différentes de celles des sites actuels. Le site ProgrammableWeb recense par exemple plus de 2 000 API, et de

nouvelles applications, comme Infochimps et Factual, lancées récemment, ont été spécifiquement développées pour fournir des données structurées.

À un niveau plus individuel, chacun peut mettre à jour ses listes d'amis sur Facebook, faire part de sa localisation sur Foursquare ou échanger de très brefs messages à propos de ses activités sur Twitter, le tout en quelques clics de souris ou entrées au clavier. Des applications plus spécialisées permettent d'enregistrer ses comportements alimentaires, son poids, son humeur et beaucoup d'autres informations. Il existe probablement une application qui vous aidera à suivre… tout ce que vous aurez l'idée de suivre !

Avec toutes ces données stockées dans des bases de données, des entrepôts ou des magasins, le domaine est mûr pour qu'elles soient utilisées de façon sensée. Pour la plupart des gens, les données ne sont pas intéressantes en elles-mêmes, ce sont les informations qu'ils peuvent en extraire qui le sont. Ils veulent savoir ce que « disent » leurs données ; si vous pouvez les y aider, vous allez être quelqu'un de très sollicité. Ce n'est pas sans raison que Hal Varian, économiste en chef de Google, annonce que le travail de statisticien sera le poste le plus « sexy » des dix prochaines années – et il ne fait pas seulement référence à la séduction des statisticiennes et statisticiens de l'entreprise…

Visualisation

La visualisation est l'un des meilleurs moyens d'explorer et de comprendre un large ensemble de données. Disposez visuellement les chiffres dans un espace et laissez votre cerveau ou celui du lecteur rechercher les modèles. Nous excellons en cela. Vous lirez souvent des histoires que vous n'auriez jamais découvertes à l'aide de simples méthodes statistiques formelles.

John Tukey, mon statisticien favori et père de l'analyse de données exploratoires, était rompu aux propriétés et méthodes statistiques, mais pensait aussi que les techniques graphiques avaient leur place. Il croyait fermement à la possibilité de découvrir l'imprévu au travers d'images. Vous pouvez en apprendre beaucoup sur les données, simplement en les visualisant, et la plupart du temps cela suffira à ce que vous preniez des décisions fondées ou puissiez raconter une histoire.

Par exemple, les États-Unis ont connu en 2009 une augmentation importante du nombre de chômeurs. En 2007, la moyenne nationale était de 4,6 %, puis elle est passée à 5,8 % en 2008 et a atteint 9,8 % en septembre 2009. Ces moyennes nationales ne racontent qu'une partie de l'histoire ; elles constituent une généralisation sur l'ensemble du pays. Y avait-il des régions qui connaissaient un taux de chômage plus élevé que d'autres ? Y avait-il des régions qui ne semblaient pas touchées par le chômage ? Les cartes de la figure I-1 racontent une histoire plus détaillée et permettent de répondre à ces questions d'un simple coup d'œil. Les comtés les plus sombres sont des zones aux taux de chômage relativement plus élevés, tandis que les comtés en clair ont des taux relativement

plus bas. En 2009, vous remarquez que beaucoup de régions de l'ouest et la plupart de celles de l'est ont des taux supérieurs à 10 %. Les zones du Middle West étaient touchées moins durement (figure I-2).

Vous n'auriez pas pu trouver ces modèles géographiques et temporels aussi rapidement avec une simple feuille de calcul, et certainement pas avec les seules moyennes nationales. En outre, même si les données pour chaque comté sont

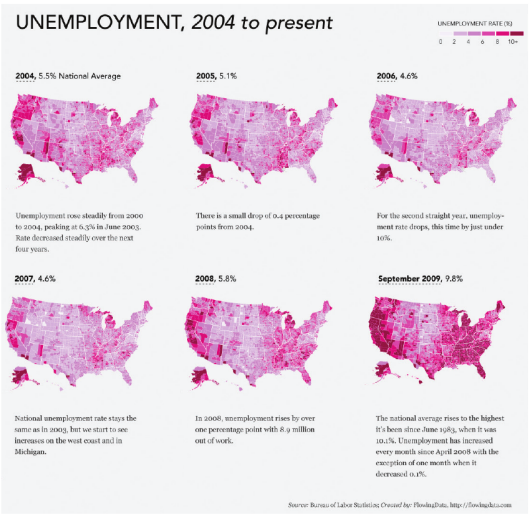


Figure I-1 *Cartes du chômage aux États-Unis de 2004 à 2009*

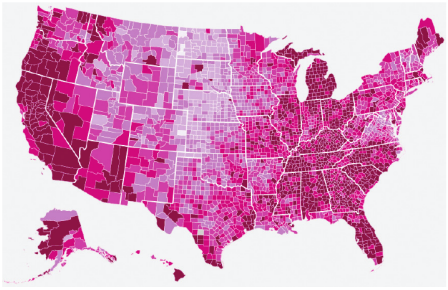


Figure I-2 *Carte du chômage pour 2009*

plus complexes à analyser, la plupart des personnes peuvent continuer à interpréter les cartes. Lesquelles peuvent, à leur tour, aider les décideurs politiques à déterminer l'affectation des caisses de secours ou autres types de prise en charge.

L'une des grandes aubaines pour la réalisation de ces cartes est la disponibilité gratuite de toutes les données au *Bureau of Labor Statistics*. Bien que qu'elles ne soient pas très aisées à retrouver à partir d'un navigateur obsolète, les chiffres existent et sont à votre disposition pour être traités visuellement.

Le *Statistical Abstract of the United States*, par exemple, comporte des centaines de tableaux de données (figure I-3) qui permettent pourtant de donner l'image détaillée d'un pays. J'ai représenté sous forme graphique certains tableaux à titre de démonstration de faisabilité, comme illustré à la figure I-4 : taux de divorce, tarifs postaux et consommation électrique, entre autres. Les tableaux sont en eux-mêmes difficiles à lire et vous n'en tirez que des valeurs individuelles, alors que les graphiques permettent de repérer aisément des tendances et des modèles et de procéder à des comparaisons en un coup d'œil.

Les journaux tels que *The New York Times* et *The Washington Post* font un excellent travail pour rendre les données plus accessibles et visuelles. Probablement sont-ils même parvenus à la meilleure utilisation possible des données en termes de narration. Parfois, les graphiques de données racontent une histoire dans sa totalité, parfois ils permettent de l'enrichir grâce à un point de vue différent.

Table 126. **Marriages and Divorces—Number and Rate by State: 1990 to 2007**

[**2,443.5 represents 2,443,500.** By place of occurence. See Appendix III]

State	Marriages [1]						Divorces [2]					
	Number (1,000)			Rate per 1,000 population [3]			Number (1,000)			Rate per 1,000 population [3]		
	1990	2000	2007	1990	2000	2007	1990	2000	2007	1990	2000	2007
U.S. [4]	**2,443.5**	**2,329.0**	**2,204.6**	**9.8**	**8.3**	**7.3**	**1,182.0**	**(NA)**	**(NA)**	**4.7**	**4.1**	**3.6**
Alabama	43.1	45.0	42.4	10.6	10.3	9.2	25.3	23.5	19.8	6.1	5.4	4.3
Alaska [5]	5.7	5.6	5.8	10.2	8.9	8.4	2.9	2.7	3.0	5.5	4.4	4.3
Arizona [5]	36.8	38.7	39.5	10.0	7.9	6.2	25.1	21.6	24.5	6.9	4.4	3.9
Arkansas	36.0	41.1	33.7	15.3	16.0	11.9	16.8	17.9	16.8	6.9	6.9	5.9
California	237.1	196.9	225.8	7.9	5.9	6.2	128.0	(NA)	(NA)	4.3	(NA)	(NA)
Colorado	32.4	35.6	29.2	9.8	8.6	6.0	18.4	(NA)	21.2	5.5	(NA)	4.4
Connecticut	26.0	19.4	17.3	7.9	5.9	4.9	10.3	6.5	10.7	3.2	2.0	3.1
Delaware	5.6	5.1	4.7	8.4	6.7	5.5	3.0	3.2	3.9	4.4	4.2	4.5
District of Columbia	5.0	2.8	2.1	8.2	5.4	3.6	2.7	1.5	1.0	4.5	3.0	1.6
Florida	141.8	141.9	157.6	10.9	9.3	8.6	81.7	81.9	86.4	6.3	5.3	4.7
Georgia	66.8	56.0	64.0	10.3	7.1	6.7	35.7	30.7	(NA)	5.5	3.9	(NA)
Hawaii	18.3	25.0	27.3	16.4	21.2	21.3	5.2	4.6	(NA)	4.6	3.9	(NA)
Idaho	14.1	14.0	15.4	13.9	11.0	10.3	6.6	6.9	7.4	6.5	5.4	4.9
Illinois	100.6	85.5	75.3	8.8	7.0	5.9	44.3	39.1	32.8	3.8	3.2	2.6
Indiana	53.2	34.5	51.2	9.6	5.8	8.1	(NA)	(NA)	(NA)	(NA)	(NA)	(NA)
Iowa	24.9	20.3	20.1	9.0	7.0	6.7	11.1	9.4	7.8	3.9	3.3	2.6
Kansas	22.7	22.2	18.6	9.2	8.3	6.7	12.6	10.6	9.2	5.0	4.0	3.3
Kentucky	49.8	39.7	33.6	13.5	10.0	7.9	21.8	21.6	19.7	5.8	5.4	4.6
Louisiana	40.4	40.5	32.8	9.6	9.3	7.6	(NA)	(NA)	(NA)	(NA)	(NA)	(NA)
Maine	11.9	10.5	10.1	9.7	8.3	7.7	5.3	5.8	5.9	4.3	4.6	4.5
Maryland	46.3	40.0	35.5	9.7	7.7	6.3	16.1	17.0	17.4	3.4	3.3	3.1
Massachusetts	47.7	37.0	38.4	7.9	6.0	6.0	16.8	18.6	14.5	2.8	3.0	2.2
Michigan	76.1	66.4	59.1	8.2	6.7	5.9	40.2	39.4	35.5	4.3	4.0	3.5
Minnesota	33.7	33.4	29.8	7.7	6.9	5.7	15.4	14.8	(NA)	3.5	3.1	(NA)
Mississippi	24.3	19.7	15.7	9.4	7.1	5.4	14.4	14.4	14.2	5.5	5.2	4.9
Missouri	49.1	43.7	39.4	9.6	7.9	6.7	26.4	26.5	22.4	5.1	4.8	3.8
Moritana	6.9	6.6	7.1	8.6	7.4	7.4	4.1	2.1	3.6	5.1	2.4	3.7
Nebraska	12.6	13.0	12.4	8.0	7.8	7.0	6.5	6.4	5.5	4.0	3.8	3.1
Nevada	120.6	144.3	126.4	99.0	76.7	49.3	13.3	18.1	16.6	11.4	9.6	6.5
New Hampshire	10.5	11.6	9.4	9.5	9.5	7.1	5.3	7.1	5.1	4.7	5.8	3.9
New Jersey	58.7	50.4	45.4	7.6	6.1	5.2	23.6	25.6	25.7	3.0	3.1	3.0
New Mexico [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
New York [6]	154.8	162.0	130.6	8.6	8.9	6.8	57.9	62.8	55.9	3.2	3.4	2.9
North Carolina	51.9	65.6	68.1	7.8	8.5	7.5	34.0	36.9	37.4	5.1	4.8	4.1
North Dakota	4.8	4.6	4.2	7.5	7.3	6.6	2.3	2.0	1.5	3.6	3.2	2.4
Ohio	98.1	88.5	70.9	9.0	7.9	6.2	51.0	49.3	37.9	4.7	4.4	3.3
Oklahoma	33.2	15.6	26.2	10.6	4.6	7.3	24.9	12.4	18.8	7.7	3.7	5.2
Oregon	25.3	26.0	29.4	8.9	7.8	7.8	15.9	16.7	14.8	5.5	5.0	4.0
Pennsylvania	84.9	73.2	71.1	7.1	6.1	5.7	40.1	37.9	35.3	3.3	3.2	2.8
Rhode Island	8.1	8.0	6.8	8.1	8.0	6.4	3.8	3.1	3.0	3.7	3.1	2.8
South Carolina	55.8	42.7	31.4	15.9	10.9	7.1	16.1	14.4	14.4	4.5	3.7	3.3
South Dakota	7.7	7.1	6.2	11.1	9.6	7.7	2.6	2.7	2.4	3.7	3.6	3.1
Tennessee	68.0	88.2	65.6	13.9	15.9	10.6	32.3	33.8	29.9	6.5	6.1	4.9
Texas	178.6	196.4	179.9	10.5	9.6	7.5	94.0	85.2	79.5	5.5	4.2	3.3
Utah	19.4	24.1	22.6	11.2	11.1	8.6	8.8	9.7	8.9	5.1	4.5	3.4
Vermont	6.1	6.1	5.3	10.9	10.2	8.6	2.6	5.1	2.4	4.5	8.6	3.8
Virginia	71.0	62.4	58.0	11.4	9.0	7.5	27.3	30.2	29.5	4.4	4.3	3.8
Washington	46.6	40.9	41.8	9.5	7.0	6.5	28.8	27.2	28.9	5.9	4.7	4.5
West Virginia	13.0	15.7	13.0	7.2	8.7	7.2	9.7	9.3	9.0	5.3	5.2	5.0
Wisconsin	38.9	36.1	32.2	7.9	6.8	5.8	17.8	17.6	16.1	3.6	3.3	2.9
Wyoming	4.9	4.9	4.8	10.7	10.3	9.3	3.1	2.8	2.9	6.6	5.9	5.5

NA Not available. [1] Data are counts of marriages performed, except as noted. [2] Based on total population residing in area; population enumerated as of April 1 for 1990 and 2000; estimated as of July 1 for all other years. [3] Includes annulments. [4] U.S. total for the number of divorces is an estimate which includes states not reporting. Beginning 2000, divorce rates based solely on the combined counts and populations for reporting states and the District of Columbia. The collection of detailed data of marriages and divorces was suspended in January 1996. [5] Some figures for marriages are marriage licenses issued. [6]

Source: U.S. National Center for Health Statistics, National Vital Statistics Reports (NVSR), *Births, Marriages, Divorces, and Deaths: Provisional Data for 2007*, Vol. 56, No. 21, July 14, 2008 and prior reports.

Figure I-3 *Tableau extrait du* Statistical Abstract of the United States

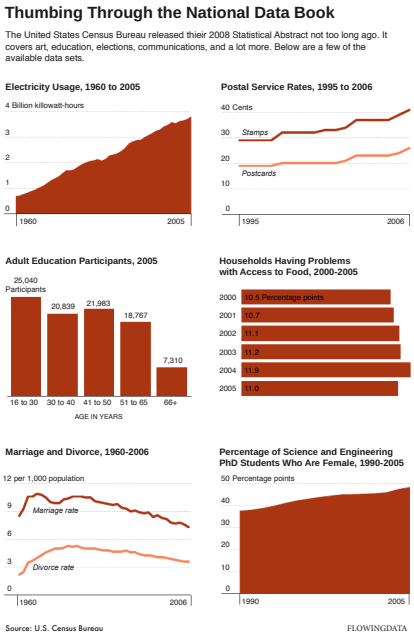


Figure I-4 *Vue graphique des données extraites du* Statistical Abstract of the United States

Les graphiques occupent une place de plus en plus grande dans les médias en ligne. Il existe désormais des départements qui ne gèrent que les animations interactives ou que les graphiques ou que les cartes. *The New York Times*, par exemple, dispose d'un service d'informations spécifiquement dédié à ce qu'il appelle le « journalisme assisté par ordinateur ». Il s'agit de journalistes spécialisés dans la narration des informations à partir de chiffres. Le bureau graphique du *New York Times* sait aussi parfaitement gérer des volumes importants de données.

La visualisation s'est également frayé un chemin dans la culture populaire. Stamen Design, entreprise de visualisation réputée pour ses animations interactives en ligne, a par exemple proposé ces dernières années un logiciel tracker de Twitter pour l'émission MTV *Video Music Awards* – chaque année Stamen conçoit des applications nouvelles mais qui montrent toujours ce dont les personnes parlent sur Twitter en temps réel. C'est ainsi que l'on a pu voir de quel côté penchait l'opinion lors de l'interruption du discours de remerciements de la chanteuse Taylor Swift – qui venait de recevoir le prix du meilleur clip féminin – par le rappeur Kayne West qui soutenait Beyonce.

À ce stade, vous entrez dans un domaine de la visualisation qui a plus à faire avec le sentiment qu'avec l'analyse. La définition de la visualisation commence à devenir quelque peu floue. Pendant une longue période, la visualisation concernait des faits quantitatifs. Il était possible, à l'aide de certains outils, de dégager des modèles qui aidaient à l'analyse. La visualisation ne consiste pas simplement à obtenir les faits purs et durs. Dans le cas du logiciel tracker de Stamen, elle est presque affaire de divertissement uniquement. Il s'agit pour les spectateurs de pouvoir regarder la remise des prix et d'interagir dans le même temps avec les autres spectateurs. Le travail de Jonathan Harris constitue un autre exemple remarquable. Harris l'articule, comme dans *We Feel Fine* et *Whale Hunt*, autour d'histoires plutôt que d'idées analytiques, et ces histoires tournent plus autour des émotions humaines que des chiffres et des analyses.

Les graphiques ont évolué pour ne plus être seulement des outils, mais également des supports de communication aux idées – et même aux plaisanteries. Les sites comme GraphJam et Indexed utilisent diagrammes de Venn, graphiques en camembert et autres pour représenter les chansons populaires ou montrer qu'une combinaison de rouge, noir et blanc peut aussi bien évoquer un journal communiste que le meurtre d'un panda. Data Underload, sorte de bande dessinée mettant en scène les données que je publie sur FlowingData, constitue mon propre apport au genre. Je recueille les observations quotidiennes et les dispose sous forme graphique. La figure I-5 illustre de célèbres passages de films répertoriés par l'*American Film Institute*. Le résultat est totalement ridicule, mais amusant (à mes yeux, du moins).

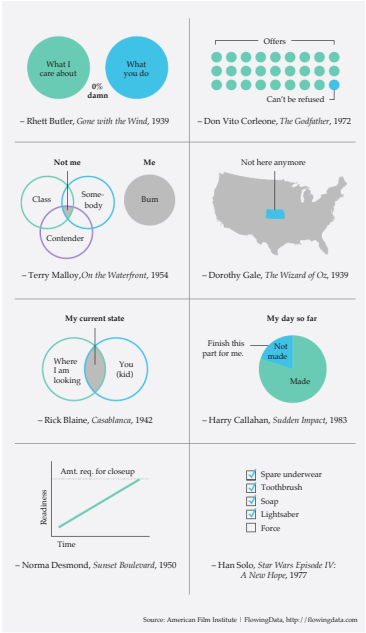


Figure I-5 *Célèbres passages de films sous forme graphique*

Découvrez Data Underload sur FlowingData, à l'adresse http://datafl.ws/underload

Par conséquent, qu'est-ce que la visualisation ? La réponse varie en fonction de la personne à laquelle vous vous adressez. Pour certains, il s'agit strictement des graphiques traditionnels, alors que pour d'autres, à la vision moins restrictive, elle peut être définie comme tout affichage de données, qu'il s'agisse d'art de données ou d'une feuille de calcul Excel. Je tends plutôt pour cette seconde approche, mais il m'arrive aussi d'adopter la première. Au final, cela n'a pas une réelle importance, l'essentiel étant de créer quelque chose en harmonie avec son propre objectif.

Que vous créiez des graphiques pour une présentation ou analysiez un vaste ensemble de données, votre recherche doit être celle de la vérité. Quand mensonges et statistiques deviennent presque synonymes, ce ne sont pas les chiffres qui mentent mais les personnes qui les utilisent, parfois à dessein pour satisfaire un propos donné mais la plupart du temps par inadvertance. Si vous ne savez pas comment créer un graphique correctement ou comment communiquer avec les données de manière impartiale, il y a de fortes chances que jaillissent des absurdités. Mais si vous apprenez les bonnes techniques de visualisation et d'utilisation des données, alors vous pourrez présenter vos informations en toute tranquillité et vous sentir à l'aise avec vos conclusions.

Apprentissage des données

J'ai découvert les statistiques lors de ma première année d'étudiant en génie électrique. Il s'agissait d'un cours d'introduction obligatoire. Contrairement à certaines histoires horribles que j'avais pu entendre, mon professeur enseignait avec un grand enthousiasme et adorait son sujet. Il ne cessait de monter et descendre les marches à grande vitesse tout en parlant. Il accompagnait ses propos de grands gestes des mains et suscitait l'implication des étudiants quand il passait à côté d'eux. À ce jour, je ne crois pas avoir jamais eu un enseignant ou un professeur aussi passionné, et à n'en pas douter, cette passion m'a conduit à m'intéresser de plus près aux données et à passer mon diplôme de statistiques quatre années plus tard.

Les années de premier cycle furent consacrées à l'analyse des données, aux distributions et aux tests d'hypothèses. J'ai aimé ces années-là et pris un grand plaisir à examiner un ensemble de données et à rechercher des tendances, des modèles et des corrélations. Les années suivantes, ma vision évolua et les choses devinrent encore plus intéressantes.

Les statistiques ne concernaient plus les tests d'hypothèses (qui se révèlent ne pas être si utiles dans nombre de cas) et la recherche de modèles. Pour être exact, en fait les statistiques avaient toujours partie liée avec les tests d'hypothèses et la recherche de modèles, mais le sentiment était différent. Elles exprimaient avant tout la capacité à raconter une histoire à l'aide de données.

Vous récoltez un tas de données, qui représentent le monde physique, puis vous analysez ces données pas seulement pour trouver des corrélations, mais aussi pour savoir ce qui se passe autour de vous. Ces histoires peuvent vous aider à résoudre des problèmes concrets, comme la diminution des crimes et délits, l'amélioration des services de santé ou le trafic autoroutier, ou simplement vous permettre d'être mieux informé. Beaucoup de personnes ne font pas ce lien entre données et vie réelle. Je pense que c'est pour cette raison que tant de gens me disent qu'ils ont détesté les statistiques à l'université. Vous ne ferez pas la même erreur, n'est-ce pas ? Vous êtes en train de lire ce livre après tout…

Comment acquérir les compétences nécessaires pour utiliser les données ? Au travers d'études, à mon exemple, mais aussi par le biais de votre propre expérience. C'est ce qui se passe pendant une grande partie de votre cursus universitaire, quel qu'il soit. Il en est de même avec la visualisation et les graphiques d'information. Nul besoin d'être concepteur graphique pour créer des graphiques de qualité. Nul besoin non plus d'être docteur en statistiques. Vous avez juste besoin d'être désireux d'apprendre, et comme dans presque tout domaine de l'existence, de pratiquer pour vous améliorer.

Je repense à l'un de mes premiers graphiques de données que je fis en quatrième année universitaire pour l'un de mes projets. Mon binôme et moi-même réfléchissions (très intensément) à la surface sur laquelle les escargots se déplaçaient le plus rapidement. Nous plaçâmes des escargots sur une surface rugueuse et sur une surface lisse, et nous les chronométrâmes. Les données dont nous disposions étaient les temps de parcours, je pus ainsi créer un graphique en barres. Je ne me souviens pas si j'eus l'idée de trier les temps du plus court au plus long, mais me rappelle que j'ai eu fort à faire avec Excel. L'année suivante, toutefois, quand nous étudiâmes quelle farine avait la préférence des scarabées de farine rouge, les graphiques furent un régal. Une fois que vous avez appris les fonctionnalités de base et exploré le logiciel, le reste est assez simple à comprendre. Si ce n'est pas là un exemple d'apprentissage à partir de l'expérience, alors je ne sais pas ce dont il s'agit. (À propos, les escargots avancent le plus vite sur verre et les scarabées de farine rouge préfèrent Grape Nuts, si cela vous intéresse.)

Nous n'évoquons ici que des exemples simples, mais ce sera essentiellement le même processus avec tout logiciel ou langage de programmation que vous apprendrez. Si vous n'avez jamais écrit une ligne de code, R, environnement de développement de nombreux informaticiens, peut paraître intimidant, mais une fois que vous aurez reproduit quelques exemples, vous acquerrez rapidement le coup de main. Cet ouvrage peut vous y aider. C'est ainsi que j'ai appris. Je me souviens lorsque je me suis mis à approfondir les aspects de conception de la visualisation : c'était au cours de l'été après ma deuxième année universitaire, je venais juste d'apprendre que j'allais travailler comme rédacteur graphique

au *New York Times*. Jusque-là, les graphiques avaient toujours été pour moi un outil destiné à l'analyse, et l'esthétique et le design n'avaient pas tant d'importance à mes yeux, si ce n'est aucune. Le rôle des données dans le journalisme ne me venait pas à l'esprit. À titre de préparation, je lus tous les livres sur le design de graphiques que je pus trouver, ainsi qu'un manuel sur Illustrator, car je savais que c'était le logiciel utilisé par *The New York Times*. Cependant, ce n'est que lorsque je me mis à créer des graphiques que je commençais réellement à comprendre.

Quand vous apprenez par l'expérience, vous êtes conduit à choisir ce qui est nécessaire, et vos compétences évoluent au fur et à mesure que vous gérez de plus en plus de données, et concevez de plus en plus de graphiques.

Comment lire ce livre

L'écriture de cet ouvrage s'appuie sur de nombreux exemples et vise à vous donner les compétences nécessaires pour créer un graphique de A à Z. Vous pouvez lire le livre du début à la fin, ou sélectionner telle ou telle partie si vous avez déjà en tête un jeu de données ou une visualisation. Les chapitres sont organisés de telle sorte que les exemples soient indépendants. Si vous débutez dans le monde des données, les premiers chapitres vous seront particulièrement utiles. Ils expliquent comment approcher les données, ce que vous devez rechercher, et les outils dont vous disposez. Vous apprendrez où rechercher les données et comment les structurer et les préparer en vue de la visualisation. Ensuite, les techniques de visualisation sont réparties par type de données et en fonction du type d'histoire recherché. N'oubliez pas, faites en sorte que ce soient toujours les données qui parlent.

De quelque façon que vous lisiez ce livre, je vous recommande vivement de le faire à côté d'un ordinateur, afin que vous puissiez parcourir les exemples pas à pas et consulter les sources identifiées dans les notes et les références. Vous pouvez aussi télécharger le code et les fichiers de données, et interagir avec les démonstrations opérationnelles disponibles sur www.wiley.com/go/visualizethis et http://book.flowingdata.com.

Pour que les choses soient bien claires, vous trouverez ci-après un organigramme pour vous aider à sélectionner les chapitres. Bonne lecture !

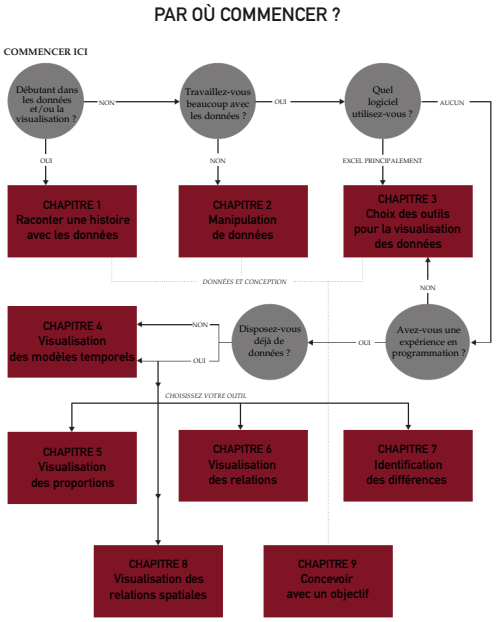


Figure I-6 *Par où commencer la lecture du livre*

À propos de l'auteur

Depuis 2007, Nathan Yau conçoit et crée des graphiques pour FlowingData, site dédié à la visualisation, aux statistiques et à la conception. Il travaille avec des entreprises aussi diverses que *The New York Times*, CNN, Mozilla et SyFy. À ses yeux, les graphiques, s'ils sont excellents pour l'analyse, conviennent aussi parfaitement à la narration d'histoires à partir de données. Nathan Yau est titulaire d'un diplôme de statistiques de l'université de Californie et prépare un doctorat sur la visualisation et les données personnelles.

Kim Rees, son éditrice, est cofondatrice de Periscopic, entreprise de visualisation des informations en prise avec la réalité sociale. Personnalité éminente du monde de la visualisation, Kim possède plus de dix-sept années d'expérience dans le secteur des applications interactives. Elle est l'auteur d'articles dans le *Parsons Journal of Information Mapping* et les *InfoVIS 2010 Proceedings*, et est notamment intervenue lors de divers événements (O'Reilly Strata Conference, WebVisions, AIGA Shift et Portland Data Visualization). Kim est diplômée en informatique de l'université de New York. Periscopic a été notamment mentionnée dans les CommArts Insights et les Adobe Success Stories, et récompensée à plusieurs reprises (VAST Challenge, CommArts Web Picks et Communication Arts Interactive Annual). Récemment, le travail de Periscopic a reçu une nomination pour les Cooper-Hewitt National Design Awards.

Chapitre 1

Raconter une histoire avec les données

Si vous réfléchissez au fonctionnement des systèmes de visualisation de données, qu'ils soient proposés dans les conférences ou dans les blogs, quel est leur point commun ? Tous racontent une histoire qui doit retenir votre intérêt. Elle a peut-être pour but de vous convaincre, de vous inviter à agir, de vous éclairer à l'aide de nouvelles informations ou de vous obliger à remettre en question votre vision de la réalité. Quoi qu'il en soit, la meilleure visualisation des données, indépendamment de son format et de sa présentation, est celle qui permet de voir ce que les données ont à dire.

Plus que des chiffres

Avouons-le, la lecture de données peut être fastidieuse si nous ignorons ce que nous cherchons. De prime abord, ce n'est qu'un amas de chiffres et de mots, sans autre signification que leur valeur brute. La grande force des statistiques et de la visualisation est de permettre de voir au-delà. Souvenez-vous que les données sont une représentation de la vie réelle. Ce ne sont pas que de simples chiffres. Les données racontent une histoire dont les principaux protagonistes ont pour nom sens, vérité ou beauté. Et, comme dans la « vraie » vie, les histoires sont parfois simples et claires ou parfois complexes et alambiquées. Certaines histoires conviennent mieux à un manuel, d'autres à une forme romanesque. C'est à vous, le statisticien, le programmeur, le designer ou le *data scientist* (spécialiste des données), de décider sous quelle forme l'histoire doit être racontée.

Ce fut l'une des premières choses que j'ai apprises quand je préparais mon diplôme de statisticien. Je dois reconnaître que, jusque-là, je considérais les statistiques comme de l'analyse pure et les données comme le résultat d'un processus mécanique. C'est bien souvent le cas. Dans la mesure où je m'étais spécialisé en tant qu'ingénieur électricien, il est vrai que je ne pouvais envisager les données sous une autre approche.

Ne vous en offusquez pas car ce n'est pas nécessairement négatif, mais au fil des années, j'ai réalisé que les données, outre leur valeur objective, présentaient souvent une dimension humaine.

Reprenons le cas du chômage aux États-Unis, par exemple. Il est facile de calculer des moyennes pour chaque État américain, mais elles varient beaucoup au sein même de ces États, voire d'un quartier à l'autre. Peut-être que l'une de vos connaissances a perdu son travail au cours des dernières années, et comme le dit l'adage, c'est la seule statistique qui compte… « Les chiffres représentant des êtres humains », c'est ainsi que nous devrions approcher les données. Inutile, bien sûr, de raconter l'histoire de chacun. Il existe une différence, subtile mais réelle, entre un taux de chômage qui croît de 5 % (ou 5 points) et plusieurs centaines de milliers de personnes qui se retrouvent sans emploi. La première valeur se lit comme un chiffre sans véritable contexte, tandis que la seconde est plus concrète.

Journalisme

Le documentaire vidéo de Geoff McGhee, *Journalism in the Age of Data*, montre très bien comment les journalistes utilisent les données pour rendre compte des événements. Il contient de remarquables entretiens avec quelques-uns des meilleurs professionnels.

Ce fut un stage au journal *The New York Times* qui finit par me convaincre. Il ne dura que les trois mois d'été après ma deuxième année universitaire, mais eut un effet durable sur mon approche des données. J'y appris non seulement à créer des graphiques pour les journaux, mais aussi à rendre compte des données comme éléments d'information. Cet apprentissage s'accompagna d'un intense travail de design (compétences graphiques), d'organisation, de vérification des faits et de recherche.

Un jour, ma seule tâche fut de vérifier trois nombres au sein d'un ensemble de données afin de contrôler l'exactitude des informations avant de les publier. Ce ne fut qu'une fois la fiabilité des chiffres établie que l'on passa à leur présentation. C'est ce soin accordé au détail qui fait toute la valeur d'un graphique.

Observez n'importe quel graphique du *New York Times*. Il présente les données de façon claire, concise et toujours élégante. Que pouvons-nous en déduire ? En réalité, quand vous regardez un graphique, l'occasion vous est offerte de comprendre les données. Les points importants sont annotés, tandis que les couleurs et les symboles sont soigneusement expliqués sous forme de légende ou autre. *The New York Times* permet aux lecteurs de distinguer facilement l'histoire transmise par les données. Ce n'est pas juste une courbe ou un tracé, mais un *graphique*.

Consultez certains des meilleurs graphiques du journal *The New York Times* à l'adresse suivante : *http://datafl.ws/nytimes*.

Le graphique de la figure 1-1 est similaire à ce que vous pouvez trouver dans *The New York Times*. Il montre l'évolution annuelle de l'espérance de vie.

À la base, c'est un simple graphique linéaire. Cependant, certains éléments aident à mieux construire l'histoire. Les diverses indications fournissent un contexte et aident à cerner l'intérêt des données, tandis que la couleur et la largeur de la ligne orientent le regard vers les informations importantes.

La création d'un graphique ne consiste pas seulement à visualiser les données statistiques, mais aussi à expliquer ce que la visualisation montre.

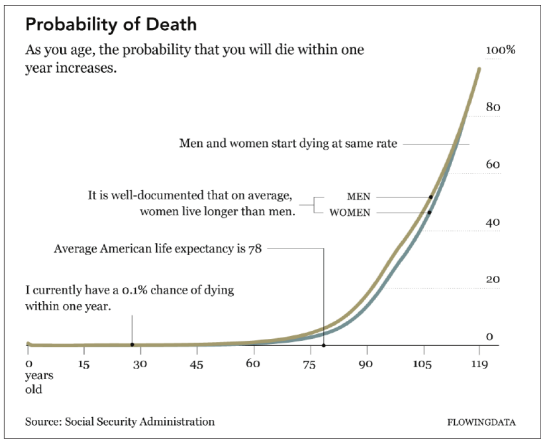


Figure 1-1 *Espérance de vie en fonction de l'âge*

Art

The New York Times est un journal objectif. Il présente les données et expose les faits. Reconnaissons qu'il y réussit parfaitement bien. La visualisation de données permet l'analyse mais répond aussi à la volonté de susciter une émotion. Jonathan Harris et Sep Kamvar y sont parfaitement parvenus dans *We Feel Fine* (figure 1-2).

La création interactive emprunte des phrases et des expressions à des blogs publics, puis les visualise sous forme de bulles flottantes. Chaque bulle représente une émotion et un code couleur lui est attribué en conséquence. Si nous regardons plus attentivement, nous pouvons remarquer que les bulles commencent à se grouper. À l'aide de l'interface, il est possible d'appliquer des critères de tri ou de classification pour voir comment ces bulles, en apparence aléatoires, se

connectent. Cliquez sur l'une d'entre elles et un texte s'affiche. C'est tout à la fois poétique et suggestif.

Explorez les émotions des individus dans la création de Jonathan Harris et de Sep Kamvar (*http://wefeelfine.org*) et interagissez avec elles.

Figure 1-2 We Feel Fine *(Jonathan Harris et Sep Kamvar)*

Consultez les nombreux exemples de création artistique et de données du site FlowingData, à l'adresse suivante : *http://datafl.ws/art*.

Il existe de nombreux autres exemples, tels que *The Dumpster* de Golan Levin, qui explore les entrées de blogs relatives aux ruptures amoureuses des adolescents, *Sumadicina* de Kim Asendorf, qui raconte à l'aide de seuls graphiques l'histoire d'un homme qui tente d'échapper à une entreprise corrompue, ou encore les sculptures d'Andreas Nicolas Fischer, qui illustrent la crise économique aux États-Unis.

Le point essentiel est que les données et la visualisation ne concernent pas toujours que les faits bruts. Parfois, vous ne recherchez pas une compréhension analytique, mais simplement à raconter l'histoire d'un point de vue émotionnel qui encourage le lecteur à réfléchir sur les données. Pensez-y ainsi. Tous les films ne sont pas des documentaires et toutes les visualisations n'ont pas à être de simples graphiques.

Loisirs

Entre journalisme et art, la visualisation s'est aussi frayé un chemin dans le domaine des loisirs. Si vous pensez aux données au sens le plus abstrait, en dehors des feuilles de calcul et des fichiers texte délimités par des virgules,

où les photos et les mises à jour de statut sont acceptées, vous comprendrez aisément.

Facebook exploitait les mises à jour de statut pour évaluer le plus beau jour de l'année, tandis que le site de rencontre en ligne OkCupid se servait des informations en ligne pour évaluer les mensonges que les individus racontaient afin de parfaire leurs egos numériques (figure 1-3). Ces analyses avaient peu à faire avec l'amélioration d'une activité, l'augmentation d'un chiffre d'affaires ou la recherche de bogues dans un système. Elles parcouraient le Web comme une traînée de poudre en raison de leur valeur divertissante. Les données dévoilaient un petit peu de nous-mêmes et de notre société.

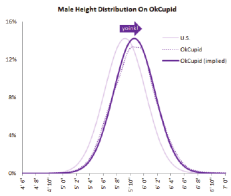


Figure 1-3 *Répartition de la taille masculine sur OkCupid*

Consultez le blog OkTrends (*http://blog.okcupid.com*), pour découvrir ce que révèlent les rencontres en ligne.

Facebook établit que le jour le plus heureux était celui de Thanksgiving et OkCupid que les membres inscrits sur son site avaient tendance à se grandir de 5 cm.

Fascinant

Bien sûr, les histoires n'ont pas toujours pour but d'informer ou de divertir. Parfois, leur mission est de répondre à une urgence ou d'inciter à l'action. Dans *An Inconvenient Truth*, par exemple, Al Gore se dresse sur un monte-charge et montre l'urgence à agir compte tenu de l'augmentation du niveau de dioxyde de carbone.

À mon avis, personne ne l'a mieux prouvé que Hans Rosling, professeur de santé internationale et président de la Fondation Gapminder. À l'aide d'un outil appelé Trendalyzer (figure 1-4), Rosling propose une animation qui illustre, par pays, les évolutions en termes de pauvreté. Il la diffuse dans le cadre d'une conférence qui vous emmène au cœur des données et vous fait vous lever et applaudir à la fin du « voyage ». Cette conférence fascinante, je ne saurais que trop la recommander.

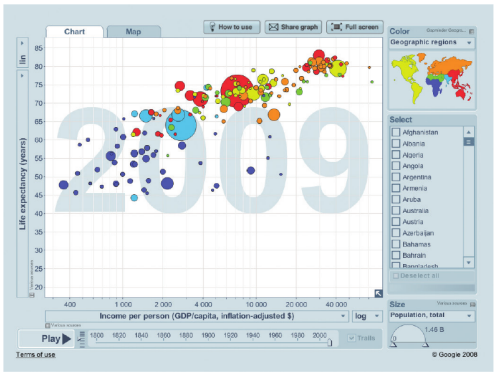


Figure 1-4 *L'outil Trendalyzer, développé par la Fondation Gapminder*

La visualisation en elle-même est assez simple. Il s'agit d'un graphique mobile. Chaque bulle représente un pays et son déplacement est fonction de la pauvreté du pays correspondant sur une année donnée. Pourquoi cet exposé est-il si populaire ? Parce que Rosling s'exprime avec enthousiasme et passion. Il raconte une histoire. Souvent, les présentations qui s'appuient abondamment sur les graphiques sont une excellente invitation à… dormir. Rosling, lui, extrait la signification des données et l'utilise à son propre avantage. De plus, lorsqu'il avale une épée à la fin de son discours, il marque vraiment un point. Après avoir vu l'exposé de Rosling, je n'aspirais qu'à me procurer les données et à les examiner par moi-même. C'était une histoire que je voulais explorer, moi aussi.

Admirez Hans Rosling en train d'embarquer ses auditeurs avec des données et une incroyable démonstration à l'adresse suivante : *http://data-fl.ws/hans*.

Par la suite, je vis un exposé de Gapminder sur le même sujet et avec les mêmes animations, mais présenté par un autre orateur. Ce ne fut pas aussi passionnant. Pour être honnête, je n'avais qu'une envie : faire la sieste. Il n'y avait pas la moindre émotion. Je ne ressentais ni conviction ni enthousiasme. Ainsi, ce ne sont pas seulement les données qui rendent le propos intéressant, c'est la

façon dont vous les présentez et les agencez qui laisse une empreinte chez les auditeurs.

En fin de compte, retenez ceci : abordez la visualisation comme si vous racontiez une histoire. Quelle sorte d'histoire voulez-vous narrer ? S'apparente-t-elle plus à un rapport ou à un roman ? Voulez-vous convaincre le public de la nécessité d'agir ?

Pensez au développement d'un personnage. Chaque point de donnée possède une histoire sous-jacente, tout comme un personnage, dans un livre, a un passé, un présent et un futur. Il existe des interactions et des relations entre ces points de données. Il vous appartient de les trouver. Bien sûr, avant qu'un romancier n'écrive des romans, il doit apprendre à construire des phrases.

Que chercher ?

D'accord, une histoire. Maintenant, quel type d'histoire raconter avec les données ? Chaque ensemble de données présente une particularité, mais plus généralement, soyez toujours à l'affût de deux éléments : les modèles et les relations. Et ce, quel que soit le graphique.

Modèles

Les temps changent, les choses évoluent. Nous vieillissons, nos cheveux commencent à grisonner et notre vue baisse (figure 1-5). Les prix évoluent, de même que les logos. Des entreprises sont créées alors que d'autres ferment. Parfois, ces changements interviennent brutalement et sans prévenir. D'autres fois, ils sont si lents que nous ne les remarquons même pas.

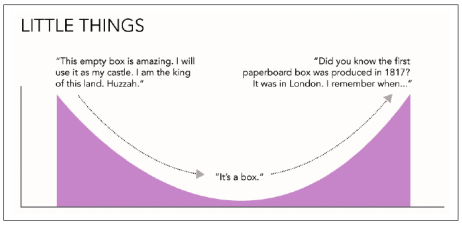


Figure 1-5 *Un regard amusé sur le vieillissement*

Quoi que vous regardiez, le résultat du changement peut être aussi intéressant que le changement lui-même. C'est à ce stade que vous pouvez explorer les modèles au fil du temps. Imaginons, par exemple, que vous vous intéressiez à l'évolution du cours des actions. Bien sûr, leur valeur augmente et diminue, mais de combien par jour ? Par semaine ? Par mois ? Existe-t-il des périodes où la hausse est plus forte qu'à l'ordinaire ? Si oui, quelle en est la raison ? Des événements précis sont-ils à l'origine de ce changement ?

Comme vous le constatez, une simple question initiale peut conduire à une multitude de questions supplémentaires. Cela ne vaut pas seulement pour les données de séries temporelles, mais pour tous les types de données. Approchez vos données de manière plus exploratoire et, très probablement, vous vous retrouverez avec des réponses plus intéressantes.

Vous pouvez partager les données de séries temporelles de différentes façons. Dans certains cas, il est préférable de s'appuyer sur des valeurs horaires ou quotidiennes ; dans d'autres, sur des valeurs mensuelles ou annuelles. Dans la première représentation, le tracé pourrait entraîner un bruit important, tandis que la seconde se rapproche plus d'une vue d'ensemble.

Ceux qui disposent d'un site web et d'un logiciel d'analyse peuvent visualiser rapidement les différences. Si vous étudiez le trafic quotidien sur votre site (figure 1-6), vous constatez que le graphique est plus irrégulier et affiche un plus grand nombre de fluctuations.

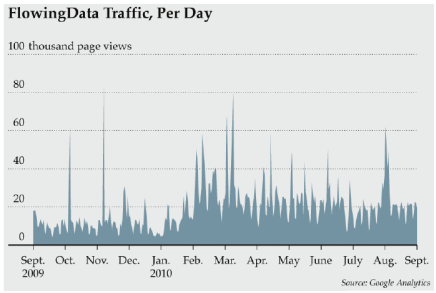


Figure 1-6 *Visiteurs quotidiens du site FlowingData*

Si vous examinez le trafic sous un angle mensuel (figure 1-7), le graphique couvre la même période, présente moins de points de données et offre, de ce fait, un aspect moins irrégulier.

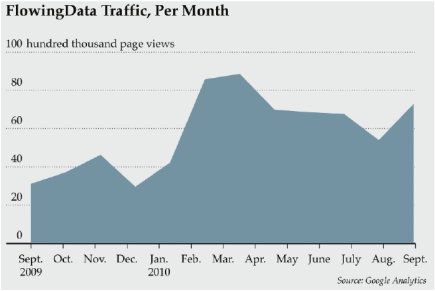


Figure 1-7 *Visiteurs mensuels du site FlowingData*

Je ne dis pas qu'un graphique est meilleur que l'autre, en réalité ils se complètent l'un l'autre. La façon dont vous découpez les données dépend du degré de détail souhaité.

Bien sûr, les modèles temporels ne sont pas les seuls à rechercher. Vous pouvez aussi chercher des modèles dans les agrégats qui vous aident à comparer les groupes, les personnes et les objets. Qu'avez-vous tendance à manger ou à boire chaque semaine ? Quels sujets aborde généralement le président américain lors de son discours annuel sur l'état de l'Union ? Quels États américains votent traditionnellement pour le candidat républicain ? L'examen de modèles à partir de régions géographiques serait utile dans ce cas. Même si les questions et les types de données sont différents, l'approche demeure similaire, comme vous le verrez dans les prochains chapitres.

Relations

N'avez-vous jamais vu un graphique composé de plusieurs éléments ou tracés qui paraissent avoir été placés de façon aléatoire ? Je fais allusion à ces graphiques où il semble manquer une information importante, comme si l'auteur n'avait accordé que peu de réflexion aux données elles-mêmes, puis produit

en toute hâte un graphique afin de respecter un délai. Souvent, l'information importante qui fait défaut n'est autre que la relation.

En termes statistiques, nous l'appelons corrélation et causalité. Plusieurs variables peuvent être associées d'une façon ou d'une autre. Le chapitre 6, « Visualisation des relations » traite de ces concepts et de leur visualisation.

À un niveau plus abstrait, où vous ne réfléchissez ni aux équations ni aux hypothèses à tester, vous pouvez concevoir votre graphique afin de comparer et de mettre en contraste visuellement les valeurs et les diverses distributions. Pour un simple exemple, reportez-vous à l'extrait du *World Progress Report* de la figure 1-8.

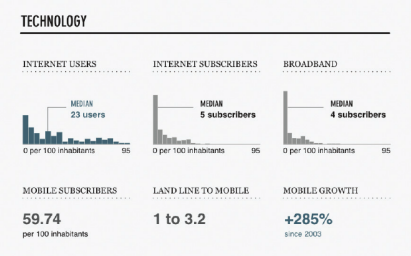


Figure 1-8 *Adoption de la technologie à travers le monde*

Le *World Progress Report* est un rapport graphique sur l'évolution à travers le monde à l'aide de données provenant d'UNdata. La version intégrale du rapport est disponible à l'adresse : *http://datafl.ws/12i.*

Ces histogrammes montrent le nombre d'internautes, d'abonnements Internet et la bande passante correspondante. Notez que la plage pour les internautes (0 à 95 pour 100 habitants) est plus étendue que celle des deux autres ensembles de données.

Une chose simple et facile à faire aurait été de laisser le logiciel choisir lui-même la plage la plus appropriée pour chaque histogramme. Cependant, chacun d'entre eux a été conçu sur la même plage, même s'il n'y a pas de pays ayant 95 abonnés Internet ou utilisateurs de la bande passante pour 100 habitants. Cette approche permet de comparer aisément les distributions entre groupes.

Par conséquent, quand vous vous retrouvez en présence de différents ensembles de données, pensez à les aborder comme plusieurs groupes, plutôt que comme compartiments cloisonnés sans interaction les uns avec les autres. Les résultats peuvent se révéler beaucoup plus intéressants.

Données en question

Lorsque vous recherchez l'histoire adaptée à vos données, vous devez toujours mettre en doute ce que vous voyez. N'oubliez pas : un nombre n'est pas vrai par essence !

Je dois l'avouer, la vérification des données est l'aspect que j'aime le moins dans la création de graphiques. Je m'explique. Quand un individu, un groupe ou un service vous fournit un tas de données, il lui appartient en principe de s'assurer de l'exactitude de ces données. Et c'est ce que feront à leur tour les auteurs de graphiques consciencieux. Après tout, de même qu'un maçon n'utilise pas un ciment de mauvaise qualité pour les fondations d'une maison, n'employez pas des données bâclées pour votre graphique.

La vérification et la validation des données constituent l'un des aspects les plus importants – si ce n'est *le* plus important – du design de graphiques.

En fait, vous recherchez une information qui n'a pas de sens. Peut-être y a-t-il eu une erreur au niveau de la saisie des données et un zéro a-t-il été ajouté ou omis par inadvertance ? Peut-être s'est-il produit un problème de connexion lors du regroupement des données et certaines se sont-elles retrouvées tronquées ? Quoi qu'il en soit, vous devez vérifier auprès de la source la présence de toute anomalie éventuelle.

La personne qui fournit les données a généralement une idée de ce qu'elle attend. Si vous êtes celui ou celle qui recueille les données, demandez-vous simplement si elles ont un sens : cet État présente une valeur égale à 90 % et tous les autres États se situent dans une plage uniquement comprise entre 10 et 20 %. Pourquoi ?

Souvent, l'anomalie n'est qu'une simple coquille, et parfois, elle caractérise un point réellement intéressant de l'ensemble de données et pourrait constituer l'élément moteur de votre histoire. Assurez-vous simplement de la nature exacte de l'anomalie.

Design

Une fois que vous possédez toutes les données et qu'elles sont en ordre, vous êtes prêt à les visualiser. Que vous fassiez un rapport, une infographie en ligne ou une représentation artistique de données, il importe de suivre quelques règles élémentaires. Bien sûr, elles offrent toutes une certaine marge de manœuvre et vous devrez davantage considérer les directives qui suivent comme un cadre que comme un ensemble de règles intangibles. Cependant, il s'agit d'un excellent point de départ si vous débutez dans le design de graphiques de données.

Explication des codes

Le design d'un graphique obéit à un flux familier. Vous récupérez les données, vous les codez à l'aide de cercles, de barres et de couleurs, et vous confiez la lecture du graphique à d'autres. À ce stade, le lecteur du graphique doit décoder les conventions adoptées. Que représentent ces cercles, barres et couleurs ?

Consultez l'article de William Cleveland et Robert McGill, *Graphical Perception and Graphical Methods for Analyzing Data*, pour plus d'informations sur la façon dont les individus codent les formes et les couleurs.

William Cleveland et Robert McGill ont écrit de façon détaillée sur le codage. Certains codages fonctionnent mieux que d'autres. Mais peu importe celui que vous adoptez si le lecteur ne sait pas ce que le codage représente. S'il est dans l'incapacité de décoder le graphique, le temps consacré à son design est un temps perdu.

Vous êtes parfois confronté à ce manque de contexte dans le cas de graphiques qui se situent à mi-chemin de la création artistique de données et de l'infographie. C'est assurément le cas avec l'« art des données ». Une légende ou un libellé peut complètement ruiner l'aura d'une œuvre, mais au moins, vous pouvez inclure certaines informations au sein d'un bref paragraphe. Il permet à autrui d'apprécier vos efforts.

En d'autres occasions, vous rencontrerez la même difficulté avec un graphique de données traditionnel, ce qui se révèle frustrant pour le lecteur et à l'opposé de vos souhaits. Il se peut que parfois vous l'oubliiez, car comme vous manipulez vous-mêmes les données, vous savez ce que chacune d'elles signifie. Le lecteur, quant à lui, se présente en quelque sorte en aveugle devant le graphique, sans la connaissance du contexte que vous avez acquise grâce à vos analyses.

Aussi comment vous assurer que vos lecteurs comprennent les différents codes utilisés ? Expliquez la signification des libellés, des légendes et des symboles. Vos choix peuvent varier en fonction de la situation. Par exemple, observez la carte du monde de la figure 1-9 qui illustre l'utilisation de Firefox par pays.

Vous apercevez bien les diverses nuances de bleu en fonction des pays, mais que signifient-elles ? Le bleu foncé indique-t-il une utilisation faible ou importante ? Et s'il signifie une utilisation élevée, que faut-il entendre par « utilisation élevée » ? En tant que telle, cette carte ne nous est pas d'une grande utilité. Mais si vous y ajoutez la légende de la figure 1-10, les choses s'éclaircissent. La légende de couleur sert à deux reprises en tant qu'histogramme représentant la distribution de l'utilisation par nombre d'utilisateurs.

Vous pouvez aussi libeller directement les formes et les objets du graphique, si vous disposez d'un espace suffisant et si n'avez pas trop de catégories (figure 1-11). Le graphique montre le nombre de nominations reçues par un acteur avant de gagner l'Oscar du meilleur acteur.

Une théorie circulait sur le Web selon laquelle les acteurs qui étaient les plus nominés parmi leurs compères une année donnée gagnaient l'Oscar. Ainsi étiqueté, l'orange foncé représente les acteurs qui eurent le plus de nominations, tandis que l'orange clair illustre ceux qui en eurent le moins.

Comme vous pouvez le constater, vous avez à votre disposition une multitude d'options. Elles sont faciles à utiliser, mais ces petits détails peuvent entraîner une énorme différence dans l'interprétation des graphiques.

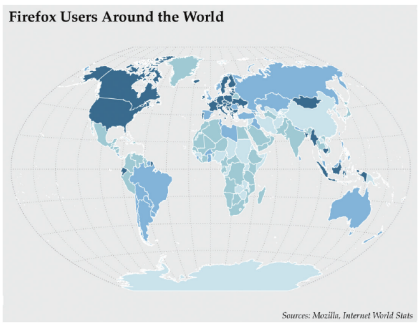


Figure 1-9 *Utilisation de Firefox dans le monde par pays*

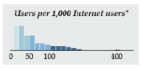


Figure 1-10 *Légende de la carte précédente*

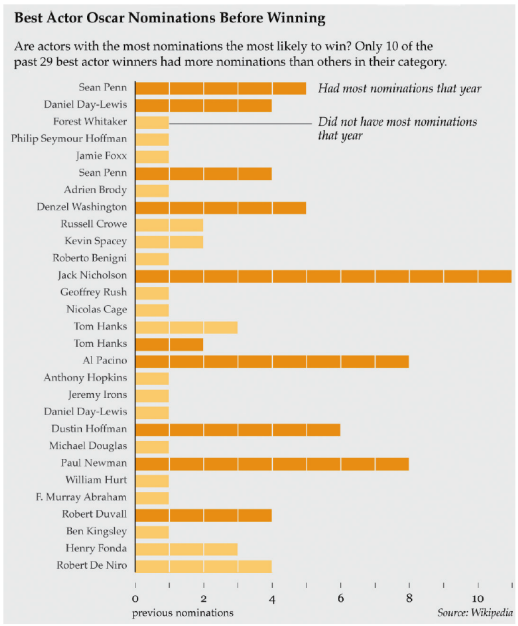


Figure 1-11 *Objets directement libellés*

Libeller les axes

Dans le même ordre d'idée que l'explication des codes utilisés, vous devez toujours libeller les axes. Dans le cas contraire, ceux-ci risquent de n'être là que pour la décoration. Libellez les axes de telle sorte que le lecteur connaisse le type d'échelle tracé. Est-il logarithmique, incrémentiel, exponentiel ou représente-t-il… 100 toilettes? Personnellement, quand je ne vois aucun libellé, je considère toujours que je suis dans le dernier cas.

Pour illustrer ma remarque, revenons sur un concours que j'ai organisé il y a quelques années sur FlowingData. J'avais publié l'image de la figure 1-12 et demandé aux lecteurs de libeller les axes au gré de leur fantaisie.

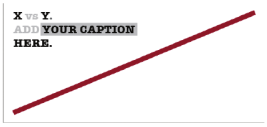


Figure 1-12 *Proposez votre légende.*

Il y eut environ 60 légendes différentes pour le même graphique. La figure 1-13 en propose quelques-unes. Même si tous les lecteurs avaient le même graphique, un simple changement dans les libellés des axes conduisait à une histoire totalement différente. Bien sûr, ce concours n'avait pas d'autre but que de s'amuser. Maintenant, imaginons un instant qu'il eût fallu prendre ce graphique au sérieux. Sans libellés, il n'aurait aucun sens.

Une géométrie sous contrôle

Lorsque vous concevez un graphique, vous vous servez de formes géométriques. Un graphique en barres utilise des rectangles, dont la différence de longueur correspond aux valeurs représentées. Dans un tracé en points, la position indique la valeur – comme dans le cas d'un graphique chronologique standard. Les graphiques en camembert symbolisent les valeurs à l'aide d'angles et la somme des valeurs est toujours égale à 100 % (figure 1-14). Rien de très compliqué, mais c'est aussi un bon moyen de se tromper. Vous commettrez aisément une erreur si vous ne prêtez pas suffisamment attention et les internautes ne manqueront pas de vous le faire savoir à grand renfort de messages.

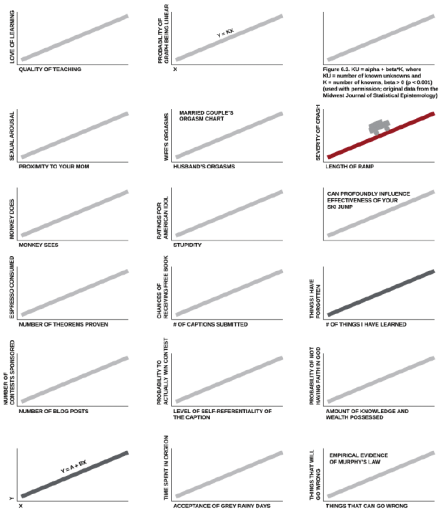


Figure 1-13 *Quelques résultats d'un concours de légendes sur FlowingData*

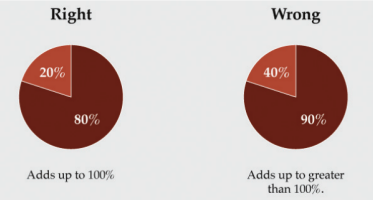


Figure 1-14 *La bonne et la mauvaise méthode pour créer un graphique en camembert*

Les auteurs font souvent une autre erreur, celle de représenter les valeurs à l'aide de formes bidimensionnelles dont ils n'utilisent qu'une seule dimension. Les rectangles d'un graphique en barres sont bidimensionnels, mais seule la longueur sert d'indicateur. La largeur ne représente rien. Cependant, quand vous créez un graphique en bulles, vous recourez à une surface pour représenter les valeurs. Les débutants utilisent souvent le rayon ou le diamètre, et l'échelle est alors totalement décalée. La figure 1-15 montre deux cercles dont la surface est proportionnelle. C'est la bonne méthode.

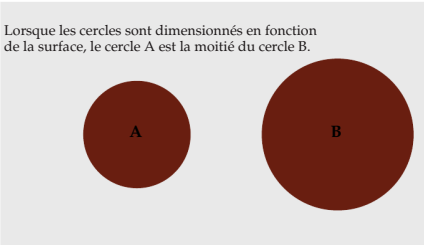


Figure 1-15 *La bonne méthode pour créer des bulles ayant les bonnes proportions*

La figure 1-16 représente deux cercles dimensionnés en fonction de la surface. Le second cercle possède un diamètre deux fois plus grand que celui du premier cercle, mais la surface est quatre fois supérieure.

Il en va de même avec les rectangles, comme dans une arborescence de rectangles (*treemap*). Vous utilisez la surface des rectangles pour indiquer les valeurs, et non la longueur ou la largeur.

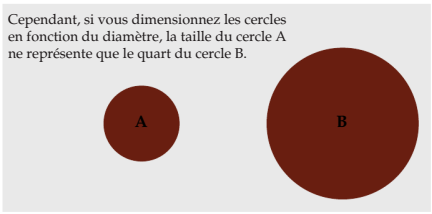


Figure 1-16 *La mauvaise méthode pour créer des bulles qui ne sont pas proportionnelles*

Inclure les sources

Cela va sans dire, mais beaucoup l'oublient. D'où viennent vos données ? Si vous regardez les graphiques des journaux, vous verrez toujours la source mentionnée, en bas et en caractères plus petits. Procédez de même. Sinon, les lecteurs n'auront aucune idée du degré réel de précision de votre graphique.

Ils n'ont en effet aucun moyen de s'assurer que vous n'avez pas tout simplement inventé les données. Bien sûr, vous ne le feriez pas, mais comment le lecteur peut-il le savoir ? Outre le fait d'asseoir l'honorabilité de votre graphique, la mention des sources permet au lecteur de contrôler les informations ou d'analyser les données.

Elle offre aussi un contexte plus large aux chiffres. De toute évidence, une enquête réalisée au cours d'un événement national aura une interprétation différente de celle effectuée au porte à porte dans le cadre d'un recensement.

Respecter les lecteurs

Enfin, prenez toujours en compte votre auditoire et l'objectif du graphique. Par exemple, un graphique destiné à un diaporama doit être simple. Vous pouvez inclure une multitude de détails, mais ceux-ci ne seront visibles que par les

seules personnes assises au premier rang. En revanche, si vous créez un poster qui a pour vocation d'être étudié et examiné, vous pouvez insérer un plus grand nombre de détails.

Vous travaillez sur un rapport d'activités ? Inutile alors de créer la plus belle œuvre que l'art des données ait jamais vue ! Privilégiez plutôt un graphique clair et allant droit au but. L'utilisez-vous dans le cadre d'une analyse ? Le graphique vous est alors destiné à vous seul et il n'est probablement pas nécessaire que vous consacriez beaucoup de temps à son esthétique et à ses annotations. Sera-t-il publié sur un support qui s'adresse à une audience de masse ? Restez simple et expliquez les éventuels concepts difficiles.

Pour résumer

En bref, commencez par une question, examinez les données d'un œil critique et définissez le but du graphique, ainsi que le public ciblé. Vous parviendrez à créer un graphique clair, quel que soit son type, et qui mérite que le lecteur y consacre de son temps.

Dans les chapitres suivants, vous apprendrez à procéder ainsi. Vous découvrirez comment gérer et visualiser les données, et comment créer un graphique du début à la fin. Ensuite, vous appliquerez ce que vous vous aurez appris à vos propres données. Cherchez l'histoire que vous voulez raconter et concevez le graphique en conséquence.

Chapitre 2

Manipulation de données

Avant de commencer à travailler sur l'aspect visuel proprement dit, vous avez besoin de données. Ce sont elles qui font l'intérêt d'une visualisation. Si vous n'avez aucune donnée intéressante, vous risquez de vous retrouver avec un graphique sans caractère ou une image très belle, mais inutilisable. Où trouver des données de qualité ? Et comment y accéder ?

Une fois en votre possession, les données doivent être mises en forme afin que vous puissiez les télécharger dans votre logiciel. Vous disposez peut-être de données sous la forme d'un fichier texte séparé par des virgules ou d'une feuille de calcul Excel, et vous avez besoin de les convertir en XML, ou inversement. Il se peut aussi que les données soient accessibles point par point à partir d'une application web, mais que vous vouliez une feuille de calcul complète… Apprenez à accéder aux données et à les traiter. Vous n'en maîtriserez que mieux la visualisation.

Collecter les données

Les données sont le cœur de toute visualisation. Heureusement, il existe de nombreux endroits où les trouver. Vous pouvez vous les procurer auprès d'experts du domaine qui vous intéresse ou au sein d'une grande variété d'applications en ligne. Vous pouvez aussi les collecter vous-même.

Données fournies par les tiers

Il s'agit de la solution la plus usuelle si vous travaillez en indépendant ou au sein du département graphique d'une grande entreprise. La collecte des données par un tiers allège le travail, mais vous devez être vigilant. Beaucoup d'erreurs peuvent se produire avant que vous ne teniez entre vos mains une feuille de calcul élégamment mise en forme.

Lorsque vous partagez des données avec des feuilles de calcul, les coquilles constituent l'erreur la plus fréquente à rechercher. Est-ce qu'il manque des zéros ? Le client ou le fournisseur de données voulait-il dire six au lieu de cinq ? À un certain moment, les données sont lues à partir d'une source, puis saisies dans Excel ou autre tableur (à moins que vous n'ayez importé un fichier texte délimité) et, par conséquent, une erreur de frappe innocente peut très bien avoir échappé aux différents contrôles et se retrouver sous vos yeux.

Il est nécessaire également de contrôler le contexte. Nul besoin que vous deveniez expert dans le domaine concerné, mais il faut savoir d'où proviennent les données, comment elles ont été recueillies et sur quoi elles portent. Cela vous aidera à créer un graphique de meilleure qualité et à raconter une histoire plus complète. Imaginons, par exemple, que vous étudiez les résultats d'un sondage. Quand celui-ci a-t-il eu lieu ? Qui l'a réalisé ? Qui a répondu ? De toute évidence, les résultats d'un sondage réalisé en 1970 auront une toute autre signification que ceux d'un sondage effectué aujourd'hui.

Recherche des sources

Si les données ne viennent pas à vous, il vous appartient d'aller les trouver. La mauvaise nouvelle est que cela entraîne un travail supplémentaire, mais la bonne nouvelle est qu'il est de plus en plus aisé de trouver des données pertinentes et qu'un ordinateur peut lire et charger dans un logiciel. C'est là où commence votre recherche.

Moteurs de recherche

Le moteur de recherche Wolfram|Alpha est disponible à l'adresse *http://wolframalpha.com*. Il peut s'avérer particulièrement utile si vous recherchez quelques statistiques élémentaires sur un thème donné.

Comment rechercher en ligne ? Utilisez Google. Une évidence peut-être, mais vous seriez surpris du nombre de courriers électroniques que je reçois où l'on me demande où l'on peut trouver tel ou tel ensemble de données ou obtenir des résultats pertinents à partir d'une recherche rapide. Personnellement, en dehors de Google, il m'arrive occasionnellement d'effectuer mes recherches avec Wolfram|Alpha, le moteur de recherche formel.

Directement à partir de la source

Si la requête directe des « données » ne se révèle pas utile, orientez vos recherches vers les sites personnels des universitaires spécialisés dans le domaine qui vous intéresse. Parfois, ils y publient leurs données. Sinon, explorez leurs articles et leurs travaux en vue d'y trouver d'éventuelles pistes. Vous pouvez aussi leur adresser un courrier électronique, mais assurez-vous au préalable que vous ne vous trompez pas d'interlocuteur. Sinon, chacun perdra son temps.

Vous pouvez également identifier des sources dans les graphiques publiés par les services de presse tels que *The New York Times*. Généralement, les sources de données sont mentionnées en petits caractères sur le graphique. Si tel n'est pas le cas, elles doivent être précisées dans l'article associé. Cela est particulièrement utile lorsque vous trouvez un graphique créé à partir de données qu'il

vous intéresse plus particulièrement d'explorer. Visitez le site correspondant à la source, car il se peut que les données y soient disponibles.

Cela ne fonctionne pas toujours, il semblerait que la recherche de contacts soit un peu plus aisée quand vous adressez un courrier électronique en vous présentant comme journaliste, mais ça vaut le coup d'essayer.

Universités

En tant qu'étudiant en doctorat, j'utilise fréquemment les ressources universitaires auxquelles j'ai accès, à savoir les bibliothèques. Beaucoup d'entre elles ont considérablement enrichi leurs ressources technologiques et possèdent quelques archives exhaustives. Un certain nombre de départements statistiques gèrent aussi une liste de fichiers de données, dont la plupart sont accessibles publiquement, même si beaucoup d'ensemble de données mis à disposition par ces départements sont destinés à être utilisés dans le cadre des travaux dirigés. Je vous suggère de consulter les ressources suivantes :

- Data and Story Library (DASL, *http://lib.stat.cmu.edu/DASL/*) – Bibliothèque en ligne de l'université Carnegie-Mellon qui propose des fichiers de données et des articles qui illustrent l'utilisation de méthodes statistiques de base.
- Berkeley Data Lab (*http://sunsite3.berkeley.edu/wikis/datalab/*) – Université de Berkeley, Californie.
- UCLA Statistics Data Sets (*http://www.stat.ucla.edu/data/*) – Certaines des données utilisées par le département de statistiques de l'UCLA dans ses laboratoires.

Applications de données générales

Le nombre d'applications de fourniture de données ne cesse de croître. Certaines d'entre elles proposent de volumineux fichiers de données que vous pouvez télécharger gratuitement ou pas. D'autres sont créées dans l'objectif de permettre aux développeurs d'accéder aux données via les API (*Application Programming Interface*) : vous pouvez ainsi utiliser les données d'un service tel que Twitter et les intégrer à votre propre application.

Voici quelques suggestions de ressources.

- Freebase (*http://www.freebase.com*) – Communauté qui propose principalement des données sur les personnes, les lieux et les objets. C'est un peu le Wikipédia des données, mais en plus structuré. Téléchargez les données ou utilisez-les comme support de votre application.
- Infochimps (*http://infochimps.org*) – Données gratuites ou payantes. Possibilité aussi d'accéder à certaines données via les API.
- Numbrary (*http://numbrary.com*) – Catalogue de données (principalement administratives) sur le Web.
- AggData (*http://aggdata.com*) – Autre référentiel de données payantes, principalement axé sur les listes complètes des emplacements de commerces de détail.

- Amazon Public Data Sets (*http://aws.amazon.com/publicdatasets*) – Propose quelques vastes ensembles de données scientifiques.
- Wikipédia (*http://wikipedia.org*) – Beaucoup d'ensembles de données de moindre taille sous forme de tableaux HTML.

Données thématiques

En dehors des fournisseurs de données générales, il existe une multitude de sites qui proposent en téléchargement gratuit des données plus spécifiques.

Les sections suivantes proposent quelques liens qui vous permettront d'accéder aux données relatives à un thème en particulier.

Géographie

Vous possédez un logiciel de cartographie, mais vous manquez de données géographiques ? Vous avez de la chance. Une multitude de fichiers sont à votre disposition.

- TIGER (*http://www.census.gov/geo/www/tiger/*) – Données du bureau américain du recensement. Probablement les plus exhaustives et les plus détaillées sur les routes, voies ferrées, fleuves et codes postaux des États-Unis que vous puissiez trouver.
- OpenStreetMap (*http://www.openstreetmap.org/*) – L'un des meilleurs exemples de données et d'initiative d'une communauté.
- Geocommons (*http://www.geocommons.com/*) – Données et cartographie.
- Flickr Shapefiles (*http://code.flickr.net/2011/01/08/flickr-shapefiles-public-dataset-2-0/*)– Frontières géographiques définies par les utilisateurs de Flickr.

Sports

Nombreux sont ceux qui aiment les statistiques sportives et vous trouverez des décennies de données précieuses. Elles sont disponibles sur *Sports Illustrated* ou sur le site des fédérations sportives américaines, mais également sur des sites qui leur sont dédiés spécifiquement.

- Basketball Reference (*http://www.basketball-reference.com/*) – Fournit les données, match par match, des rencontres de la NBA.
- Baseball DataBank (*http://baseball-databank.org/*) – Excellent site à partir duquel vous pouvez télécharger des ensembles de données complets.
- databaseFootball (*http://www.databasefootball.com/*) – Explorez les matchs de la NFL par équipe, joueur et saison.

Monde

Plusieurs organisations internationales réputées conservent diverses données sur le monde, notamment les indicateurs de santé et de développement. Cependant, un certain criblage est nécessaire, car beaucoup de données sont assez

disparates. Il n'est pas aisé d'obtenir des données standardisées entre des pays utilisant des méthodes différentes.

- Global Health Facts (*http://www.globalhealthfacts.org/*) – Données relatives à la santé dans le monde.
- UNdata (*http://data.un.org/*) – Agrégat de données mondiales provenant de différentes sources.
- Organisation mondiale de la santé (*http://www.who.int/research/fr/index.html*) – Autres ensembles de données sur la santé, notamment sur la mortalité et sur l'espérance de vie.
- Statistiques de l'OCDE (*http://www.oecd-ilibrary.org/fr/statistiques*) – Principale source d'indicateurs économiques.
- Banque mondiale (*http://www.banquemondiale.org/*) – Données de centaines d'indicateurs à la disposition des développeurs.

Administration et politique

L'accent ayant été mis ces dernières années sur les données et la transparence, de nombreuses organisations gouvernementales proposent leurs données et des groupes tels que la Sunlight Foundation encouragent développeurs et concepteurs à les utiliser. Désormais, avec le lancement du site Data.gov (data.gouv.fr en France), la plupart de ces données sont disponibles à partir d'un seul et même emplacement. Vous pouvez également trouver plusieurs sites non gouvernementaux qui cherchent à rendre les hommes politiques plus responsables.

- United States Census Bureau (*http://www.census.gov/*) – Données démographiques.
- Data.gov (*http://data.gov/*) – Catalogue des données fournies par les organisations gouvernementales américaines. Encore relativement nouveau, mais propose une multitude de sources.
- Data.gov.uk (*http://data.gov.uk/*) – Équivalent du Data.gov américain pour le Royaume-Uni.
- data.gouv.fr (*http://data.gouv.fr/*) – Équivalent du Data.gov américain pour la France.
- San Francisco Data (*http://datasf.org/*) – Données spécifiques à la ville de San Francisco.
- NYC OpenData (*http://nyc.gov/data/*) – Données propres à la ville de New York.
- ParisData (*http://opendata.paris.fr/*) – Ensemble des jeux de données publiés par les services de la ville de Paris sous licence libre.
- Opendata-map.org (*http://www.opendata-map.org/*) – Mise à disposition de données publiques transversales, cartographiées sous trois catégories.

- Follow the Money (*http://www.followthemoney.org/*) – Nombreux outils et ensembles de données pour analyser l'utilisation de l'argent dans la politique des États américains.
- OpenSecrets (*http://www.opensecrets.org/*) – Fournit des détails sur les dépenses gouvernementales et le lobbying.

Récupération des données

Trouver les données exactes dont vous avez besoin n'est pas le plus compliqué. En revanche, elles seront souvent éparpillées dans plusieurs pages HTML ou plusieurs sites web et figureront rarement dans un seul fichier ou au même endroit. Que faire dans ce cas ?

La méthode la plus simple, mais la plus chronophage, consiste à visiter chaque page et à entrer manuellement les données qui vous intéressent dans une feuille de calcul. Si vous n'avez que quelques pages, pas de problème.

Même si le codage constitue la solution la plus souple pour récupérer les données dont vous avez besoin, vous pouvez aussi recourir à des outils tels que Google Refine, ScraperWiki et le convertisseur Able2Extract PDF. Ils sont simples d'utilisation et pourront vous faire gagner un temps précieux.

Mais comment faire si vous avez des milliers de pages à analyser ? L'opération serait alors très fastidieuse. Il serait bien plus simple d'automatiser le processus, ce qui est le but du *scraping de données* (récupération des données). Vous écrivez le code qui permet de visiter plusieurs pages automatiquement, d'extraire le contenu souhaité de chacune d'elles et de le stocker dans une base de données ou dans un fichier texte.

Exemple : récupérer les données d'un site web

Rien ne vaut un exemple pour apprendre à récupérer les données. Imaginons que vous vouliez télécharger les températures de l'année écoulée, mais que vous ne parveniez pas à trouver une source fournissant l'ensemble des valeurs de la période ou de la ville souhaitée. Quel que soit le site web consacré à la météorologie, vous ne trouverez au mieux que les prévisions de température pour les 10 prochains jours. Et ce n'est pas ce que vous souhaitez : vous voulez savoir ce que furent les températures à telle ou telle époque et non ce qu'elles seront demain ou après-demain.

Heureusement, le site Weather Underground (*http://wunderground.com*) propose les données recherchées, même si vous ne pouvez consulter qu'un seul jour à la fois.

Pour être plus concret, nous allons examiner les températures à Buffalo. Connectez-vous au site Weather Underground et saisissez « BUF » dans le champ de recherche et cliquez sur le bouton pour valider. La page météo de l'aéroport de Buffalo s'affiche alors.

Figure 2-1 *Températures à Buffalo (New York), selon Weather Underground*

Le haut de la page affiche la température du jour et les prévisions pour 5 jours, ainsi que d'autres informations sur la journée en cours.

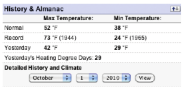
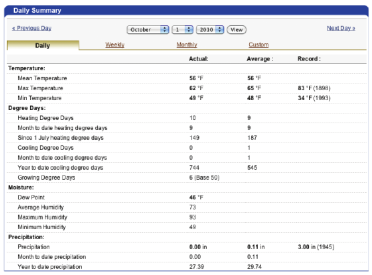


Figure 2-2 *Menu déroulant permettant de sélectionner une date*

Sélectionnez une date et cliquez sur le bouton View (Affichage, en français). Les données relatives à cette date s'affichent à l'écran.



Daily Summary

« Previous Day | October 1 2010 View | Next Day »

Daily | Weekly | Monthly | Custom

	Actual:	Average:	Record:
Temperature:			
Mean Temperature	**56** °F	**56** °F	
Max Temperature	**62** °F	**65** °F	**83** °F (1988)
Min Temperature	**49** °F	**46** °F	**34** °F (1993)
Degree Days:			
Heating Degree Days	10	9	
Month to date heating degree days	9	9	
Since 1 July heating degree days	149	167	
Cooling Degree Days	0	1	
Month to date cooling degree days	0	1	
Year to date cooling degree days	744	845	
Growing Degree Days	6 (Base 50)		
Moisture:			
Dew Point	**46** °F		
Average Humidity	73		
Maximum Humidity	90		
Minimum Humidity	49		
Precipitation:			
Precipitation	**0.00** in	**0.11** in	**3.00** in (1945)
Month to date precipitation	0.00	0.11	
Year to date precipitation	27.39	29.74	

Figure 2-3 *Informations détaillées pour la date du 1er octobre 2010*

Sont mentionnés la température, les degrés-jours, l'humidité, les précipitations et une multitude d'autres informations. Pour l'heure, seule la température maximale par jour nous intéresse, disponible dans la deuxième ligne, deuxième colonne.

Le 1er octobre 2010, la température maximale à Buffalo était de 17 °C (62 °F). L'obtention de cette valeur a été assez simple. Maintenant, comment faire pour récupérer la température maximale de chaque jour de l'année 2009 ? La méthode la plus simple et la plus directe consisterait à changer à chaque fois la date dans la liste déroulante.

Après avoir effectué cette action 365 fois, vous aurez certes obtenu les données recherchées, mais cela vous aura pris pas mal de temps et surtout, la tâche est très répétitive et ennuyeuse. Vous pouvez accélérer le processus grâce à un peu de code et de savoir-faire. Pour cela, nous allons utiliser le langage de programmation Python et la bibliothèque Python de Leonard Richardson, Beautiful Soup.

Vous allez avoir un avant-goût des premières lignes de code dans les prochains paragraphes. Si vous avez une expérience en programmation, vous pouvez les parcourir assez rapidement. Dans le cas contraire, ne vous inquiétez pas, je vous guiderai pas à pas. Beaucoup de personnes aiment s'en remettre à quelques clics sur une interface, mais faites-moi confiance. Si vous acquérez quelques compétences en programmation, vous vous ouvrirez une multitude de possibi-

lités quant à l'utilisation des données. Alors prêt ? Allons-y ! Tout d'abord, vous devez vous assurer que tous les logiciels appropriés sont installés sur votre ordinateur. Si vous travaillez sous Mac OS X, Python doit déjà être installé. Ouvrez l'application Terminal et saisissez « python » (figure 2-4).

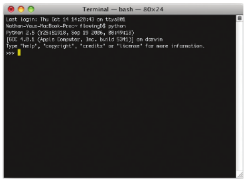


Figure 2-4 *Démarrage de Python sous Mac OS X*

Si vous utilisez un ordinateur Windows, rendez-vous sur le site Python (*http://python.org*) afin de télécharger la version appropriée à votre système d'exploitation et procédez à son installation. Ensuite, téléchargez la bibliothèque Beautiful Soup (*http://www.crummy.com/software/BeautifulSoup/*), ce qui vous aidera à lire les pages web rapidement et facilement. Choisissez la version de Beautiful Soup correspondant à la version de Python que vous utilisez. Enregistrez le fichier Python (`.py`) Beautiful Soup dans le répertoire où vous prévoyez de sauvegarder le code. Si vous connaissez déjà Python, vous pouvez aussi placer Beautiful Soup dans le chemin d'accès de votre bibliothèque, mais le résultat sera similaire. Ouvrez un nouveau fichier dans votre éditeur de texte et enregistrez-le sous le nom `get-weather-data.py`. Vous pouvez à présent écrire le code.

La première chose à faire consiste à charger la page qui affiche les données météorologiques du 1er octobre 2010 à Buffalo :

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

Si vous supprimez tout ce qui suit `.html` dans l'URL précédente, la même page se charge :

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

La date est mentionnée dans l'URL par `/2010/10/1`. À l'aide des menus déroulants, modifiez la date pour afficher l'historique des données du 1er janvier 2009 car vous voulez récupérer les températures de toute l'année 2009. L'URL devient alors :

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

L'URL est identique à celle du 1er octobre 2010, à l'exception de la partie qui indique la date et qui affiche désormais `/2009/1/1`. Intéressant. Sans recourir aux menus déroulants, comment charger la page du 2 janvier 2009 ? Modifiez simplement le paramètre date de telle sorte que l'URL se présente ainsi :

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

Chargez l'URL précédente dans votre navigateur et vous obtenez le récapitulatif historique du 2 janvier 2009. Par conséquent, pour obtenir la météo d'un jour donné, il suffit de modifier l'URL de Weather Underground. Gardez cela présent à l'esprit pour la suite. Maintenant, chargez une page avec Python et importez la bibliothèque urllib2 avec la ligne de code suivante :

```
import urllib2
```

Pour charger la page du 1er janvier 2009 avec Python, utilisez la méthode `urlopen`.

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

Cette instruction charge tout le code HTML vers lequel pointe l'URL de la variable `page`. L'étape suivante consiste à extraire du code HTML la température maximale qui vous intéresse. Beautiful Soup va vous simplifier la tâche. Après la bibliothèque urllib2, importez Beautiful Soup de la façon suivante :

```
from BeautifulSoup import BeautifulSoup
```

À la fin du fichier, utilisez Beautiful Soup pour lire (c'est-à-dire décoder) la page.

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

Sans trop rentrer dans les détails, cette instruction lit le code HTML – qui n'est qu'une longue chaîne de caractères – et stocke les éléments de la page, tels que l'en-tête ou les images, de façon à ce qu'ils soient plus simples à utiliser. Par exemple, pour trouver toutes les images de la page, vous pouvez écrire :

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

Vous obtenez la liste de toutes les images de la page Weather Underground qui s'affichent avec la balise HTML `<img />`. Pour que ce soit la première image de la page, écrivez :

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

Pour obtenir la deuxième image, remplacez le zéro (`0`) par un (`1`). Pour afficher la valeur `src` dans la première balise `<img />`, écrivez :

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

Bien, vous ne voulez pas d'images. Vous ne voulez qu'une seule valeur : la température maximale le 1er janvier 2009 à Buffalo. Il faisait -3 °C (26 degrés Fahrenheit).

La recherche de cette valeur est un peu plus complexe que celle des images, mais la méthode demeure identique. Il vous suffit de savoir quoi écrire dans `findAll()`. Pour cela, étudiez le code source HTML.

Vous pouvez facilement l'afficher dans tous les principaux navigateurs. Dans Firefox, rendez-vous dans le menu Outils>Développeur web>Code source de la page. Le code HTML de la page active s'affiche, comme illustré à la figure 2-5.

Rendez-vous dans le menu Edition>Rechercher et saisissez « -3 » dans le champ de recherche car c'est la valeur que vous souhaitez extraire. La ligne est encadrée par la balise `<span>` et la classe `nobr`. C'est l'élément clé. Vous pouvez trouver tous les éléments de la page avec la classe `nobr`.

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

Beautiful Soup propose une bonne documentation et des exemples simples. Aussi, si ces lignes de code vous semblent déroutantes, je vous invite vivement à les vérifier sur le site Beautiful Soup.

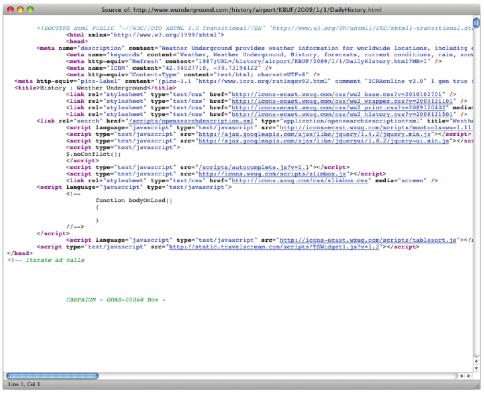


Figure 2-5 *Code source HTML d'une page de Weather Underground*

Comme précédemment, vous obtenez la liste de toutes les occurrences de `nobr`. Celle qui vous intéresse est la sixième, que vous retrouvez avec l'instruction suivante :

```
print nobrs[5]
```

Vous récupérez la totalité de l'élément, alors que seule la valeur -3 vous intéresse. À l'intérieur de la balise `<span>`, outre la classe `nobr`, se trouve une autre balise `<span>` et la valeur -3. Voici ce que vous devez écrire :

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

Bravo ! Vous avez récupéré votre première valeur d'une page web HTML. L'étape suivante consiste à explorer toutes les pages de l'année 2009. Pour cela, affichez à nouveau l'URL suivante dans votre navigateur :

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

N'oubliez pas que vous avez modifié l'URL manuellement pour obtenir les données météorologiques de la date souhaitée. Le code précédent se rapporte au 1er janvier 2009. Si vous voulez la page du 2 janvier 2009, modifiez simplement la partie de l'URL correspondant à la date. Pour obtenir les données de chaque jour de l'année 2009, chargez chaque mois (1 à 12), puis chargez chaque jour du mois. Le script complet est indiqué ci-après, accompagné de commentaires. Enregistrez-le dans votre fichier `get-weather-data.py`.

```
import urllib2
from BeautifulSoup import BeautifulSoup

# Créer/ouvrir un fichier nommé wunder.txt (valeurs séparées par des
  virgules)
f = open('wunder-data.txt', 'w')

# Parcourir les mois et les jours
for m in range(1, 13):
    for d in range(,1, 32):
      # Vérifier si le mois a déjà été lu
      if (m == 2 and d > 28):
         break
      elif (m in [4, 6, 9, 11] and d > 30):
         break

      # Ouvrir wunderground.com url
      timestamp = '2009' + str(m) + str(d)
      print "Récupération des données du " + timestamp
      url = "http://www.wunderground.com/history/airport/
      ➥KBUF/2009/" + str(m) + "/" + str(d) + "/DailyHistory.html"
      page = urllib2.urlopen(url)
```

```
        # Récupérer la température à partir de la page
        soup = BeautifulSoup(page)
        # dayTemp = soup.body.nobr.b.string
        dayTemp = soup.findAll(attrs={"class":"nobr"})[5].span.string

        # Mettre en forme le mois pour l'horodatage
        if len(str(m)) < 2:
            mStamp = '0' + str(m)
        else:
            mStamp = str(m)

        # Mettre en forme le jour pour l'horodatage
        if len(str(d)) < 2:
            dStamp = '0' + str(d)
        else:
            dStamp = str(d)

         Créer l'horodatage
        timestamp = '2009' + mStamp + dStamp

        # Écrire l'horodatage et la température dans le fichier
        f.write(timestamp + ',' + dayTemp + '\n')

# Fin de traitement des données, fermer le fichier
f.close()
```

Vous devriez reconnaître les deux premières lignes de code qui importent les bibliothèques nécessaires, soit urllib2 et BeautifulSoup.

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

Ensuite, créez un fichier texte `wunder-data.txt` contenant des autorisations d'écriture, à l'aide de la méthode `open()`. Toutes les données récupérées seront stockées dans ce fichier texte, dans le même répertoire que celui où vous avez enregistré le script.

```
# Créer/ouvrir un fichier nommé wunder.txt (valeurs séparées par des
  virgules)
f = open('wunder-data.txt', 'w')
```

Dans la première ligne de code suivante, la boucle `for` demande à l'ordinateur de parcourir chaque mois. Le numéro du mois est stocké dans la variable `m`. La boucle suivante demande à l'ordinateur de parcourir chaque jour de chaque mois. Le numéro du jour est stocké dans la variable `d`.

Consultez la documentation Python pour plus d'informations sur le fonctionnement des boucles et de l'itération : *http://docs.python.org/reference/compound_stmts.html*

```
# Parcourir les mois et les jours
for m in range(1, 13):
      for d in range(1, 32):
```

Notez que nous utilisons `range (1, 32)` pour parcourir tous les jours. Autrement dit, nous parcourons les nombres de 1 à 31. Cependant, seuls certains mois de l'année ont 31 jours. Février n'a que 28 jours ; avril, juin, septembre et novembre n'ont que 30 jours. Il n'y a pas de température pour le 31 avril, parce que ce jour n'existe pas. Aussi, pour chaque mois, agissez en conséquence. Si le mois en cours est février et que le jour est supérieur à 28, interrompez la recherche et passez au mois suivant. Pour récupérer les températures de plusieurs années, il vous faut utiliser une instruction `if` supplémentaire afin de gérer les années bissextiles. De même, si le mois est avril, juin, septembre ou novembre, passez au mois suivant après le 30.

```
      # Vérifier si le mois a déjà été lu
      if (m == 2 and d > 28):
         break
      elif (m in [4, 6, 9, 11] and d > 30):
         break
```

Les quelques lignes de code suivantes vont vous sembler familières. En effet, vous les avez déjà utilisées pour extraire une simple page de Weather Underground. La différence réside dans la variable mois et jour de l'URL. Modifiez la valeur pour chaque jour au lieu de la laisser statique ; le reste est identique. Chargez la page avec la bibliothèque urllib2, analysez le contenu avec Beautiful Soup et récupérez la température maximale, mais recherchez la sixième occurrence de la classe `nobr`.

```
      # Ouvrir wunderground.com url
      url = "http://www.wundergound.com/history/airport/
      ➥KBUF/2009/" + str(m) + "/" + str(d) + "/DailyHistory.html"
      page = urllib2.urlopen(url)

      # Récupérer la température à partir de la page
      soup = BeautifulSoup(page)
      # dayTemp = soup.body.nobr.b.string
      dayTemp = soup.findAll(attrs={"class":"nobr"})[5].span.string
```

Les dernières lignes de code créent un horodatage à partir de l'année, du mois et du jour, au format aaaammjj. Vous pouvez choisir un autre format, mais restons simple pour l'instant.

```
      # Mettre en forme le jour pour l'horodatage
      if len(str(d)) < 2:
         dStamp = '0' + str(d)
```

```
    else:
        dStamp = str(d)

    # Créer l'horodatage
    timestamp = '2009' + mStamp + dStamp
```

Enfin, la température et l'horodateur sont écrits dans le fichier `wunder-data.txt` à l'aide de la méthode `write()`.

```
# Écrire l'horodatage et la température dans le fichier
f.write(timestamp + ',' + dayTemp + '\n')
```

Puis, utilisez `close()` lorsque vous en avez fini avec tous les mois et tous les jours.

```
    # Fin de traitement des données, fermer le fichier
    f.close()
```

Il ne reste plus qu'à exécuter le code, à l'aide de la commande suivante :

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

Soyez patient car l'exécution nécessite un certain temps. L'ordinateur est en train de charger 365 pages, une pour chaque jour de l'année 2009. À la fin de l'exécution, vous devez avoir dans votre répertoire de travail un fichier intitulé `wunder-data.txt`. Ouvrez-le, il contient les données présentées sous forme de valeurs séparées par des virgules (figure 2-6). La première colonne correspond aux horodatages, la seconde aux températures.

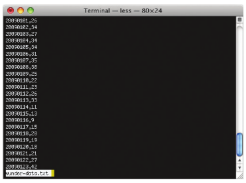


Figure 2-6 *Une année de températures*

Généralisation de l'exemple

Vous pouvez généraliser le processus ayant servi à récupérer les données de Weather Undergound et l'utiliser avec d'autres sources de données. La récupération de données se compose généralement de trois étapes :

1. Identifier les modèles.

2. Parcourir toutes les données.
3. Stocker les données.

Dans cet exemple, nous avions deux modèles. Le premier se trouvait dans l'URL et le second dans la page web chargée pour obtenir les températures réelles. Afin de charger la page d'un autre jour de l'année 2009, vous avez changé les parties mois et jour de l'URL. La valeur de la température résidait dans la sixième occurrence de la classe `nobr` de la page HTML. Si aucun modèle d'URL évident ne se dégage, essayez d'établir comment vous pourriez obtenir l'URL de toutes les pages à récupérer. Peut-être le site possède-t-il un plan ou peut-être pouvez-vous explorer l'index via un moteur de recherche. Pour finir, vous devez connaître toutes les URL des pages de données.

Une fois que vous avez trouvé les modèles, vous entamez l'itération sur les données : vous visitez toutes les pages par programme, les chargez et les analysez. Précédemment, vous avez utilisé Beautiful Soup, qui simplifie l'analyse XML et HTML en Python. Il existe probablement une bibliothèque similaire si vous choisissez un autre langage de programmation.

Il faut aussi stocker les données. La solution la plus simple consiste à enregistrer les données sous forme de fichier texte dont les valeurs sont séparées par des virgules. Cependant, si vous avez une base de données, vous pouvez aussi les y stocker.

Le processus peut être plus compliqué si vous traitez des pages web qui utilisent du JavaScript pour récupérer les données à afficher, mais le processus demeure identique.

Structurer des données

Les outils de visualisation emploient différents formats de données et la structure choisie varie en fonction de l'histoire à raconter. Plus la structure de vos données offre de souplesse, plus vous gagnez en possibilités. Appuyez-vous sur les applications de mise en forme des données, complétez-les d'un peu de programmation et de savoir-faire, et vous obtiendrez le format de données dont vous avez besoin.

Bien sûr, le mieux serait de trouver un programmeur à même de structurer et d'analyser l'ensemble de vos données, mais vous risquez d'attendre longtemps. Cela est particulièrement évident pendant les premières étapes d'un projet où l'exploration et l'itération des données jouent un rôle essentiel dans la conception d'une visualisation efficace. Sincèrement, si j'avais à embaucher, je privilégierais la recherche d'une personne qui sache manipuler les données et non une personne qui a besoin d'aide au début de chaque projet.

Les prochaines sections sont consacrées aux formats de données, aux outils qui permettent de les traiter et à la programmation nécessaire et qui repose sur la même logique que celle utilisée dans l'exemple précédent.

Ce que j'ai appris sur le format de données

Quand j'ai commencé à apprendre les statistiques au lycée, les données se présentaient toujours sous une belle forme rectangulaire. Il me suffisait d'entrer quelques nombres dans une feuille Excel ou dans ma géniale calculatrice (le meilleur moyen de faire croire que vous écoutiez en classe, alors qu'en réalité vous jouiez à Tetris). Il en fut ainsi durant toutes mes années de lycée. Comme j'apprenais alors les techniques et les théorèmes d'analyse, mes professeurs ne perdaient pas de temps à travailler sur les données brutes, prétraitées. Les données semblaient toujours au bon format.

Ce sont là des contraintes de temps parfaitement compréhensibles et par-là même, une fois en troisième cycle, je compris que les données ne semblent jamais être dans la réalité au format que vous souhaitez. Il manque des valeurs, les libellés ne sont pas cohérents, il y a des coquilles ou bien aucun contexte n'est fourni. Les données sont souvent réparties sur plusieurs tableaux, alors que vous souhaitez que tout soit regroupé en un seul, à l'aide d'une.valeur, comme un nom ou un identifiant unique.

Il en fut de même quand je commençais à utiliser la visualisation. Elle prit une place croissante, parce que je souhaitais obtenir plus de données en ma possession. Désormais, il n'est pas rare que je passe autant de temps à obtenir les données au format souhaité qu'à élaborer l'aspect visuel du graphique. Parfois, je consacre même plus de temps à traiter mes données. Cela peut paraître étrange de prime abord, mais vous constaterez que le design de graphiques est nettement simplifié si les données sont clairement organisées, comme à l'époque où je découvris les statistiques au lycée.

Formats de données

La plupart des personnes sont habituées à manipuler les données avec Excel. L'application est parfaite si vous y réalisez la totalité de votre projet, depuis l'analyse jusqu'à la visualisation. Mais, si vous souhaitez aller au-delà, vous devrez vous familiariser avec d'autres formats de données. L'objet de ces formats est de rendre les données lisibles par l'ordinateur ou, en d'autres termes, de les structurer de telle sorte qu'un ordinateur puisse les comprendre. Le choix même du format des données peut varier en fonction de l'objectif et de l'outil de visualisation. Les trois formats suivants peuvent répondre à la plupart de vos besoins : texte délimité, JSON (*JavaScript Object Notation*) et XML (*eXtensible Markup Language*).

Texte délimité

Nombre d'entre vous connaissent ce format et, dans notre exemple précédent, nous ne faisions rien d'autre que de créer un fichier texte délimité. Si vous considérez un ensemble de données sous forme de lignes et de colonnes, un fichier texte délimité sépare les colonnes à l'aide d'un délimiteur. Celui-ci est généralement une virgule ou une tabulation, mais ce peut être aussi un espace, un point-virgule, une barre oblique ou tout autre caractère de votre choix.

Le texte délimité est très utilisé et peut être lu par la plupart des tableurs comme Excel ou Google Drive. Vous pouvez aussi exporter les feuilles de calcul comme

du texte délimité. Si votre classeur Excel contient plusieurs feuilles, vous avez autant de fichiers délimités, sauf spécification contraire.

Le format convient aussi parfaitement au partage de données, car il ne dépend d'aucun programme particulier.

L'ensemble des spécifications du format JSON est disponible à l'adresse suivante : *http://json.org/*. Il n'est pas nécessaire de connaître chaque particularité du format, mais si vous ne comprenez pas une source de données JSON, n'hésitez pas à consulter les spécifications.

JSON

Il s'agit d'un format souvent proposé par les API du Web. Il est à la fois compréhensible par l'ordinateur et par l'être humain, même si, en présence d'un texte abondant, vous risquez fort de vous surprendre à loucher… Le format repose sur la notation JavaScript, mais ne dépend pas de ce langage. Les spécifications pour JSON sont nombreuses, mais vous pouvez fort bien, pour l'essentiel, vous contenter des notions de base.

JSON utilise des propriétés et des valeurs, et traite les données comme des objets. Si vous devez convertir des données JSON en un format de texte délimité CSV (Comma-Separated Values), chaque objet peut se présenter sous forme d'une ligne.

Comme nous le verrons par la suite, un certain nombre d'applications, de langages et de bibliothèques acceptent le JSON en entrée. Si vous prévoyez de créer des graphiques de données pour le Web, vous rencontrerez très probablement ce format.

XML

Le format XML est également très répandu sur le Web et il souvent utilisé pour transférer les données via les API. Il existe de nombreux types et spécifications pour le XML, mais au niveau le plus élémentaire, il s'agit d'un document texte dont les valeurs sont encadrées par des balises. Par exemple, le flux RSS (*Really Simple Syndication*) que les personnes utilisent pour s'inscrire à un blog, tel que FlowingData, est un fichier XML (figure 2-7).

Le RSS répertorie les derniers éléments publiés à l'intérieur des balises `<item>` `</item>` et chaque élément possède un titre, une description, un auteur, une date de publication, ainsi que quelques autres attributs.

XML est relativement facile à analyser avec les bibliothèques telles que Beautiful Soup dans Python. Vous aurez un meilleur aperçu de XML, ainsi que de CSV et JSON, dans les sections qui suivent.

Outils de mise en forme

Il y a quelques années à peine, la gestion et la mise en forme des données s'effectuaient toujours à partir de scripts. Après en avoir écrit quelques-uns, vous commenciez à repérer des modèles dans la logique et il devenait assez aisé d'écrire de nouveaux scripts pour des ensembles de données spécifiques, même si cela prenait du temps. Heureusement, les volumes croissants de données ont favorisé le développement d'outils pour gérer les routines ordinaires.

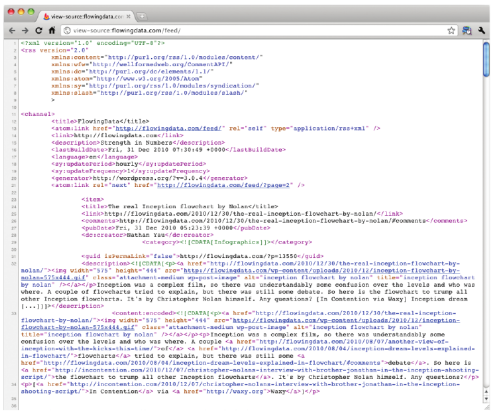


Figure 2-7 *Extrait du flux RSS de FlowingData*

Google Refine

Google Refine est le successeur de l'application Freebase Gridworks. Cette dernière a d'abord été développée comme outil interne d'une plate-forme de données ouverte, Freebase. Cette plate-forme fut ensuite rachetée par Google et renommée Google Refine. Elle est désormais constituée de Gridworks 2.0 et d'une interface plus simple d'utilisation (figure 2-8) et pourvue d'un plus grand nombre de fonctionnalités.

Elle s'exécute directement sur votre ordinateur (via le navigateur), ce qui évite d'avoir à télécharger des données privées sur les serveurs Google. Tout le traitement s'effectue sur votre ordinateur. Google Refine est aussi une application open source : vous pouvez tout à fait adaptez l'outil à vos propres besoins en lui ajoutant des extensions.

Quand vous ouvrez Google Refine, l'interface familière d'une feuille de calcul apparaît, avec des lignes et des colonnes. Il est possible d'effectuer un tri par champ et de rechercher des valeurs particulières. Vous pouvez également déceler des incohérences dans vos données et consolider ces dernières d'une façon relativement nouvelle.

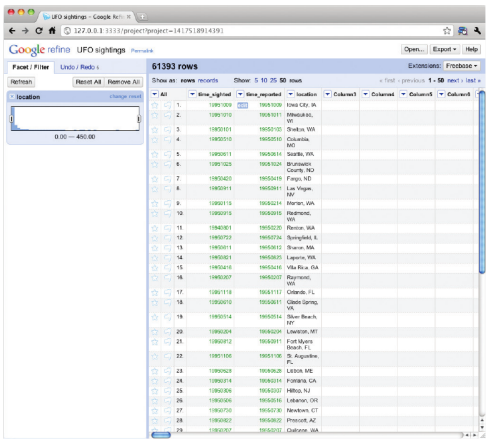


Figure 2-8 *Interface utilisateur de Google Refine*

Imaginons, par exemple, que vous disposiez de la liste des ustensiles de votre cuisine. Vous pouvez charger les données dans Google Refine et trouver rapidement des incohérences, telles que fautes de frappe ou classifications divergentes. Il se peut que le mot « fourchette » ait été mal orthographié en « furchette » ou peut-être souhaitez-vous reclasser les fourchettes, cuillères et couteaux comme ustensiles. Vous pouvez retrouver sans peine ces informations

avec Google Refine et apporter les modifications nécessaires. Si vous ne voulez pas conserver les changements effectués ou avez commis une erreur, vous pouvez revenir à l'état d'origine à l'aide d'une simple annulation.

Google Refine propose également des fonctions plus avancées. Vous pourrez ainsi incorporer des sources de données, par exemple la vôtre, à un ensemble de données de Freebase pour en créer un plus riche.

Google Refine est un outil fort utile, puissant et gratuit, que je vous invite vivement à découvrir.

Téléchargez l'application open source Google Refine et consultez les tutoriels consacrés à cet outil à l'adresse : *http://code.google.com/p/google-refine/*.

Mr. Data Converter

Il arrive souvent que vous ayez toutes vos données dans Excel, mais que vous deviez les convertir en un autre format. Tel est presque toujours le cas lorsque vous créez des graphiques pour le Web. Vous pouvez déjà exporter les feuilles Excel au format CSV, mais qu'en est-il si le format est autre ? L'outil Mr. Data Converter vous sera d'une grande aide dans ce cas.

Mr. Data Converter est un outil simple et gratuit créé par Shan Carter, rédacteur graphique au *The New York Times*. Carter consacre la plus grande partie de son activité professionnelle à créer des graphiques interactifs pour la version en ligne du journal. Comme il doit souvent convertir les données pour les adapter aux logiciels qui les manipulent, il n'est pas étonnant qu'il ait créé un outil pour optimiser le processus.

Il est simple d'utilisation et son interface est claire et intuitive (figure 2-9). Il suffit de copier les données dans Excel et de les coller dans la section d'entrée, en haut, puis de sélectionner le format de sortie dans la partie inférieure de l'écran. Vous avez le choix entre les formats XML, JSON et d'autres encore.

Si vous voulez créer votre propre convertisseur ou étendre les fonctionnalités de Mr. Data Converter, son code source est également disponible.

Essayez l'outil Mr. Data Converter à l'adresse *http://www.shancarter.com/data_converter/* (ou téléchargez le code source sur GitHub à l'adresse suivante : *https://github.com/shancarter/Mr-Data-Converter*) pour convertir vos feuilles Excel en un format web convivial.

Mr. People

Inspiré par Mr. Data Converter, Matthew Ericson, directeur graphique du *The New York Times*, a créé Mr. People. Comme Mr. Data Converter, Mr. People permet de copier et de coller des données dans un champ texte, puis de les analyser et de les extraire automatiquement. Mr. People, toutefois, comme vous l'aurez deviné, ne concerne que l'analyse des noms.

Utilisez Mr. People à l'adresse *http://people.ericson.net/* ou téléchargez son code source Ruby sur le site GitHub afin d'exploiter l'analyseur de noms dans vos propres scripts : *http://github.com/mericson/people*.

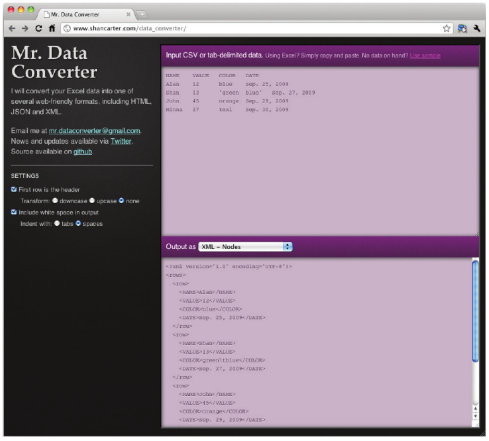


Figure 2-9 *Mr. Data Converter facilite la conversion des données d'un format vers un autre.*

Figure 2-10 *Noms en entrée sur Mr. People*

Peut-être avez-vous une longue liste de noms sans mise en forme particulière et voulez-vous identifier le prénom et le nom, ainsi que l'initiale intermédiaire, le préfixe et le suffixe ? Peut-être plusieurs personnes apparaissent-elles sur une même ligne ? C'est là qu'intervient Mr. People. Copiez et collez les noms (figure 2-10), et vous obtenez un joli tableau, bien propre, qu'il ne vous reste plus qu'à copier dans votre tableur favori (figure 2-11).

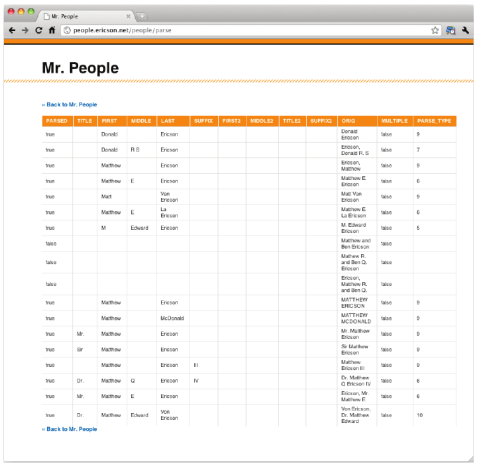


Figure 2-11 *Noms analysés selon un format tabulaire avec Mr. People*

Le code source de Mr. People est également disponible librement sur le site GitHub.

Tableurs

Si vous n'avez besoin que de trier les données ou d'y apporter quelques modifications, vous pouvez simplement utiliser votre tableur favori si vous êtes à l'aise avec la modification manuelle des données. Dans le cas contraire, essayez d'abord la solution précédente (notamment si l'ensemble de données est gigantesque) ou optez pour une solution personnalisée à base de code.

Structurer avec le code

Même s'il peut être utile de disposer d'un logiciel clés en main, vous constaterez que les applications ne répondent pas toujours à vos attentes et que certaines d'entre elles ne permettent pas de gérer les fichiers de données volumineux – elles ralentissent ou plantent.

Que faire dans ce cas ? Vous pouvez lever les mains au ciel et abandonner, mais cette attitude n'est pas très productive. Une autre solution consiste à écrire vous-même le code pour que le travail soit exécuté automatiquement. Ceci vous permettra d'être beaucoup plus souple et vous pourrez adapter les scripts à vos données.

L'exemple suivant va vous montrer à quel point il est aisé de passer d'un format de données à un autre avec quelques lignes de code seulement.

Exemple : passage d'un format de données à un autre

Cet exemple utilise Python, mais vous pouvez tout à fait choisir un autre langage. La logique reste la même, seule la syntaxe sera différente. (Comme j'aime développer les applications en Python, la gestion des données brutes avec ce même langage convient parfaitement à mon flux de travail.)

Retournons à l'exemple précédent sur la récupération des données et utilisons le fichier résultant `wunder-data.txt`, qui contient les températures de Buffalo (New York) pour l'année 2009. Les premières lignes se présentent ainsi :

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

Il s'agit d'un fichier CSV, mais imaginons que vous vouliez les données en XML selon le format suivant :

```
<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>
```

Chaque température du jour est encadrée par la balise `<observation>` ainsi que les balises `<date>` et `<max_temperature>`.

Pour convertir le fichier CSV au format XML précédent, vous pouvez utiliser l'extrait de code suivant :

```
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>'
```

Comme précédemment, vous importez les modules nécessaires. Dans le cas présent, vous avez uniquement besoin du module `csv` pour lire le fichier `wunder-data.txt`.

```
import csv
```

La deuxième ligne de code ouvre le fichier `wunder-data.txt`avec `open()` et le lit avec la méthode `csv.reader()`.

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

Remarquez que le séparateur spécifié est la virgule. Dans le cas d'une tabulation, nous aurions écrit \t.

Nous affichons ensuite la ligne d'ouverture du fichier XML à la ligne 3.

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

Dans la partie principale du code, nous parcourons chaque ligne de données l'une après l'autre et l'affichons selon le format XML souhaité. Dans cet exemple, chaque ligne de l'en-tête CSV est équivalente à chaque observation en XML.

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

Chaque ligne comporte deux valeurs : la date et la température maximale.

Terminons la conversion en XML avec sa balise de fermeture.

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

Ici, deux choses importantes sont en jeu. Tout d'abord, nous lisons les données, puis nous les parcourons ligne après ligne en modifiant chacune d'entre elles d'une certaine façon. C'est la même logique que si vous convertissiez à nouveau le XML obtenu en CSV. Comme illustré dans l'extrait de code suivant, la différence est que nous utilisons un module différent pour analyser le fichier XML.

```
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
```

Le code paraît différent, mais, pour l'essentiel, nous faisons la même chose. Au lieu d'importer le module `csv`, nous importons `BeautifulStoneSoup` à partir de `BeautifulSoup`. Souvenons-nous que nous avons utilisé `BeautifulSoup` pour analyser le code HTML à partir de Weather Underground. `BeautifulStoneSoup` analyse le code XML plus général.

Nous pouvons ouvrir le fichier XML et le lire avec `open()`, puis charger le contenu dans la variable `xml`. À ce stade, le contenu est stocké en tant que chaîne de caractères. Pour procéder à l'analyse, transmettons la chaîne `xml` à `BeautifulStoneSoup` pour passer en revue chaque `<observation>` du fichier `XML`. Utilisons `findAll()` pour extraire toutes les observations, et enfin, comme nous l'avons fait pour la conversion de CSV en XML, exécutons une boucle pour parcourir chaque observation, en imprimant les valeurs au format souhaité.

Nous revenons ainsi à notre point de départ :

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

Pour vous convaincre, voici le code de conversion du format CSV au format 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 "] }"
```

Lisez chaque ligne pour comprendre ce qui se passe, mais une fois encore, c'est la même logique avec une sortie différente. Voici ce à quoi ressemble le fichier JSON si nous exécutons le code précédent.

```
{
    "observations": [
         {
            "date": "20090101",
            "temperature": 26
         },
         {
            "date": "20090102",
            "temperature": 34
         },
         ...
    ]
}
```

Les données sont les mêmes, à savoir date et température maximale, mais présentées dans un format différent. Les ordinateurs aiment la variété, vous l'ignoriez ?

Mettre la logique en boucle

Si nous étudions le code qui convertit le fichier CSV en JSON, nous notons la présence d'une instruction `if-else` dans la boucle `for`, après les trois lignes `print`. Cette instruction vérifie si l'itération en cours porte sur la dernière ligne de données. Si tel est le cas, nous n'insérons rien en fin d'observation, sinon nous plaçons une virgule. Ceci est une partie de la spécification JSON. Nous pouvons faire plus encore, ici-même.

Nous pouvons vérifier si la température maximale dépasse une certaine valeur et créer un champ qui vaut 1 si la température est supérieure au seuil et 0 dans le cas contraire. Nous pouvons créer des catégories ou des jours repères avec les valeurs manquantes.

Rien n'impose qu'il ne s'agisse que d'un simple contrôle du seuil de température. Nous pouvons calculer une moyenne mobile ou la différence entre le jour en cours et le précédent. Nous pouvons accomplir une multitude d'actions au sein de la boucle pour enrichir les données brutes. Nous ne les aborderons pas toutes ici, car il peut s'agir aussi bien de simples modifications que d'analyses élaborées. Contentons-nous d'un simple exemple.

Retournons à notre fichier CSV d'origine, `wunder-data.txt`, et créons une troisième colonne qui indique si la température maximale d'une journée était inférieure ou égale à zéro. La valeur 0 indique une température supérieure à zéro et 1 une température inférieure à zéro.

```
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
```

Comme auparavant, lisons les données du fichier CSV en Python, et parcourons chaque ligne l'une après l'autre. Vérifions chaque jour et plaçons un repère en conséquence.

Bien sûr, cet exemple est simple, mais il est aisé de voir comment la logique peut être enrichie pour mettre en forme les données ou les compléter à votre goût. Souvenez-vous des trois étapes – chargement, boucle et traitement – et élargissez à partir de là.

Pour résumer

Dans ce chapitre, nous avons vu où vous pouviez trouver les données et comment les gérer une fois qu'elles étaient en votre possession. Il s'agit d'une étape essentielle, si ce n'est la plus importante, du processus de visualisation. Un graphique n'est intéressant que par les données qui le sous-tendent. Vous pouvez l'embellir autant que vous voulez, les données (ou les résultats de votre analyse des données) en demeurent la substance ; et maintenant que vous savez où et comment vous procurer les données, vous avez fait un grand pas en avant.

Vous avez également eu un avant-goût de ce qu'était la programmation. Vous avez récupéré les données d'un site web, puis vous les avez structurées et réorganisées, ce qui vous sera utile dans les prochains chapitres. Cependant, l'aspect déterminant est la logique du code. Vous avez utilisé Python, mais vous pourriez aussi avoir utilisé Ruby, Perl ou PHP. La logique demeure la même d'un langage à l'autre. Lorsque vous connaissez un langage de programmation (et si vous êtes un programmeur confirmé, vous pouvez l'attester), il est beaucoup plus facile d'apprendre d'autres langages par la suite.

Vous n'avez pas toujours à recourir au code. Parfois, certaines applications clés en main rendent le travail plus facile ; n'hésitez pas à en tirer parti le cas échéant. En fin de compte, plus vous avez d'outils dans votre boîte à outils, moins vous êtes susceptible de vous retrouver bloqué au milieu du processus.

Vous avez les données, il est temps maintenant de passer à la visualisation !

Chapitre 3

Choix des outils pour la visualisation des données

Dans le précédent chapitre, vous avez appris à rechercher et trouver les données et aussi à les récupérer au format requis. Vous voici donc prêt à les visualiser. L'une des questions que l'on me pose le plus fréquemment à ce stade est la suivante : « Quel logiciel dois-je utiliser pour visualiser mes données ? »

Heureusement, vous avez le choix. Certains logiciels, de type glisser-déplacer, sont prêts à l'emploi. D'autres nécessitent un peu de programmation. Cependant, il existe aussi des outils qui n'ont pas été conçus spécifiquement pour les graphiques de données, mais qui se révèlent néanmoins utiles. Le présent chapitre traite de ces différentes options.

Mieux vous savez utiliser les outils de visualisation et en tirer parti, moins vous risquez de vous retrouver à ne savoir que faire d'un ensemble de données et plus vous avez de chances de créer un graphique correspondant à votre vision.

Visualisation prête à l'emploi

Les solutions prêtes à l'emploi sont de loin les plus aisées pour les novices. Copiez et collez quelques données ou bien chargez un fichier CSV, et vous êtes prêt ! Cliquez simplement sur le graphique de votre choix et modifiez éventuellement une ou deux options de-ci de-là.

Options

Ces outils prêts à l'emploi varient selon l'application pour laquelle ils ont été conçus. Certaines applications, comme Excel, Google Drive ou encore Google Sheets, sont destinées à la gestion des données et aux graphiques de base, tandis que d'autres prennent en charge des analyses plus approfondies et l'exploration visuelle.

Microsoft Excel

Vous reconnaissez bien sûr la feuille de calcul illustrée à la figure 3-1, dans laquelle vous allez insérer les données.

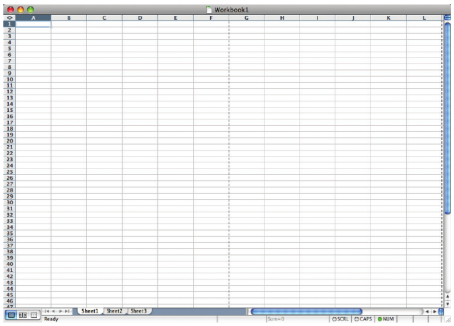


Figure 3-1 *Feuille de calcul Excel*

Cliquez ensuite sur le bouton représentant un graphique en barres afin de créer le graphique de votre choix (figure 3-2) : graphique en barres, linéaire, en camembert ou nuage de points.

Certaines personnes dédaignent Excel, alors que c'est une application assez efficace pour les tâches simples. Je n'utilise pas Excel pour les analyses approfondies ou les graphiques destinés à une publication. Mais si je me trouve avec un petit ensemble de données dans un fichier Excel, comme c'est souvent le cas, et que je veux savoir rapidement ce que j'ai sous les yeux, je prépare un graphique en quelques clics seulement.

Cette simplicité d'utilisation est ce qui rend Excel si séduisant aux yeux du public, et ce n'est que justice. Mais, si vous souhaitez des graphiques de données d'une qualité supérieure, ne vous arrêtez pas là et employez des outils plus appropriés.

Les graphiques peuvent être ludiques

C'est dans Excel que j'ai créé mon premier graphique, dans le cadre d'un projet universitaire. Mon partenaire de projet et moi-même essayions de découvrir sur quelle surface les escargots se déplaçaient le plus rapidement. Je peux vous assurer que c'était une recherche innovante !

Même alors j'ai aimé créer un graphique. Il me fallut du temps pour apprendre (je découvrais aussi l'ordinateur, à l'époque), mais quand finalement, j'y parvins, ce fut une agréable satisfaction. J'entrais les nombres dans le tableur et obtenais instantanément un graphique que je pouvais modifier dans la couleur de mon choix – jaune vif, en l'occurrence.

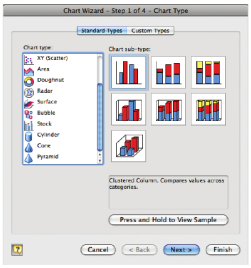


Figure 3-2 *Options des graphiques Excel*

Google Sheets

Google Sheets correspond plus ou moins à la version cloud d'Excel, et reprend l'interface familière de ce dernier (figure 3-3).

L'application propose aussi les types de graphiques standards (figure 3-4).

Google Sheets offre toutefois certains avantages par rapport à Excel. Tout d'abord, comme vos données sont stockées sur les serveurs Google, vous pouvez les visualiser sur n'importe quel ordinateur, sous réserve qu'un navigateur web y soit installé. Connectez-vous à votre compte Google et en avant ! Vous pouvez aussi partager aisément votre feuille de calcul et collaborer en temps réel. Google Sheets propose également quelques options de graphiques supplémentaires, via la commande Gadget (figure 3-5).

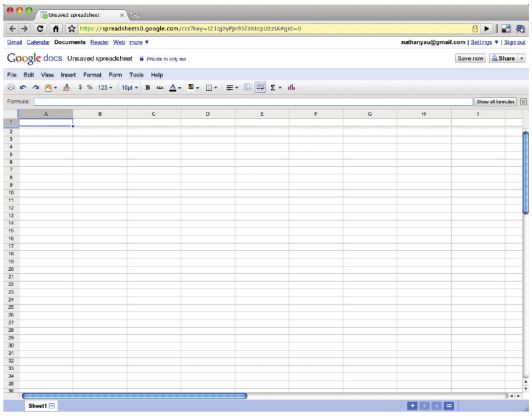


Figure 3-3 *Google Sheets*

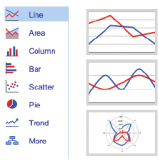


Figure 3-4 *Google Spreadsheets*

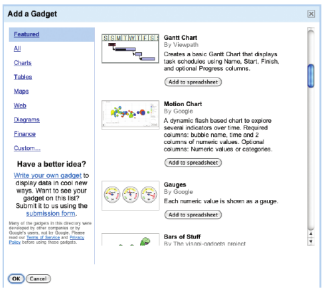


Figure 3-5 *Gadgets Google*

Si beaucoup de gadgets sont sans utilité, certains sont très utiles. Par exemple, vous pouvez aisément créer un graphique en mouvement avec vos données de séries temporelles (tout comme Hans Rosling). Vous trouverez également un graphique chronologique interactif avec lequel vous devriez être familier si vous visitez Google Finance (figure 3-6).

Visitez Google Drive à l'adresse *http://drive.google.com* et découvrez les feuilles de calcul Google Sheets.

Figure 3-6 *Google Finance*

Many Eyes

Many Eyes est le nom d'un portail de visualisation géré par IBM. Il s'agit d'une application en ligne qui permet de charger les données en tant que fichier texte délimité et d'explorer ce dernier via un ensemble d'outils de visualisation interactifs. L'objectif initial de Many Eyes était de voir s'il était possible d'analyser des ensembles de données conséquents comme groupes, d'où le nom du projet. Si plusieurs yeux sont braqués sur un ensemble de données volumineux, un groupe peut-il détecter des points de données intéressants plus rapidement ou plus efficacement, ou découvrir parmi les données des informations qu'un individu seul n'aurait pu trouver ?

Même si les analyses sociales de données n'ont pas gagné en popularité avec Many Eyes, les outils peuvent continuer à être utiles pour le particulier. La plupart des types classiques de visualisation sont disponibles, comme les graphiques linéaires (figure 3-7) et les nuages de points (figure 3-8).

L'une des grandes qualités de toutes les visualisations de Many Eyes est qu'elles sont interactives et proposent un certain nombre d'options de personnalisation. Les nuages de points, par exemple, permettent de mettre les points à l'échelle à l'aide d'une troisième mesure et vous pouvez afficher les valeurs individuelles en survolant avec la souris un point particulier.

Figure 3-7 *Graphiques linéaires dans Many Eyes*

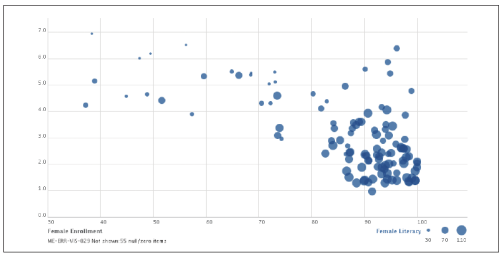


Figure 3-8 *Nuage de points dans Many Eyes*

Many Eyes propose aussi une grande variété de visualisations plus avancées et expérimentales, ainsi que quelques outils élémentaires de cartographie. L'arborescence lexicale permet ainsi d'explorer le corps complet d'un texte, comme dans un livre ou un article. Vous choisissez un mot ou une expression, puis vous observez comment votre sélection est utilisée à travers le texte en examinant les phrases qui suivent. La figure 3-9, par exemple, montre les résultats de la recherche du mot « right » dans la Constitution des États-Unis.

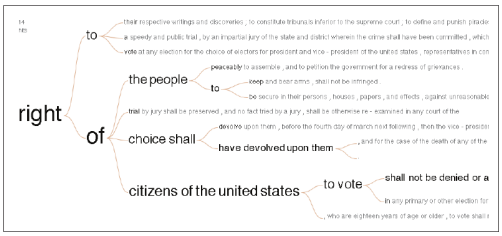


Figure 3-9 *Outil d'arborescence lexicale de Many Eyes illustrant les différentes parties de la Constitution des États-Unis*

De même, vous pouvez aisément passer d'un outil à un autre, en utilisant les mêmes données. La figure 3-10 illustre la Constitution visualisée à l'aide d'un nuage lexical stylisé appelé Wordle. La taille des mots est proportionnelle à leur fréquence.

Figure 3-10 *Wordle de la Constitution des États-Unis*

Comme vous pouvez le constater, Many Eyes contient beaucoup d'options qui vous aideront à manipuler les données. C'est de loin l'outil gratuit le plus polyvalent (et à mes yeux, le meilleur) dédié à l'exploration des données. Quelques mises en garde s'imposent néanmoins. La première est que la plupart des outils sont des applets Java et que, par conséquent, il faut que l'application Java soit installée sur votre ordinateur. Ce n'est pas un énorme problème, mais je connais certaines personnes qui, pour une raison ou une autre, font très attention à ce qu'elles installent sur leur ordinateur.

Téléchargez et visualisez vos propres données à l'adresse *http://many-eyes.com*.

L'autre restriction est que toutes les données que vous chargez sur le site font partie du domaine public. Autrement dit, vous ne pouvez pas utiliser Many Eyes, par exemple, pour décortiquer les informations sur vos clients ou les ventes réalisées par votre entreprise.

Tableau Software

Tableau Software, logiciel Windows uniquement, est relativement récent, mais sa popularité n'a cessé de croître ces dernières années. Il est principalement conçu pour explorer et analyser les données visuellement. Il est clair qu'un grand soin a été apporté à son esthétique et à sa conception, ce qui explique l'engouement en sa faveur.

Tableau Software propose plusieurs outils de visualisation interactifs et permet de parfaitement gérer les données, que vous pouvez importer depuis Excel, les fichiers texte et les serveurs de base de données. Les graphiques de séries temporelles, en barres, en camembert, les cartes simples, etc., sont

aussi disponibles. Vous pouvez assortir les différents affichages et associer une source de données dynamique à une vue personnalisée, ou à un tableau de bord, pour obtenir une vue d'ensemble instantanée de vos données.

Plus récemment, la société Tableau a proposé Tableau Public, logiciel gratuit composé d'un sous-ensemble des fonctionnalités de l'édition complète. Vous pouvez charger vos données sur les serveurs de Tableau, créer un affichage interactif et le publier aisément sur votre site web ou votre blog. Toutefois, les données que vous chargez sur les serveurs, comme Many Eyes, deviennent aussitôt disponibles publiquement, ne l'oubliez pas.

Visitez le site Tableau Software à l'adresse *http://tableau-software.com/fr-fr*. Une version d'évaluation complète est disponible gratuitement.

Si vous voulez utiliser Tableau et que vos données demeurent confidentielles, choisissez l'édition intégrale Tableau Desktop. Au moment où j'écris ces lignes, il vous en coûtera respectivement 999 $ et 1 999 $ pour les éditions Personnel et Professionnel.

your.flowingdata

Mon intérêt pour la collecte de données personnelles m'a inspiré ma propre application, your.flowingdata (YFD). Il s'agit d'une application en ligne qui permet de recueillir les données via Twitter, puis d'explorer les modèles et les relations avec un ensemble d'outils de visualisation interactifs. Certaines personnes aiment noter leurs habitudes en matière d'alimentation et de sommeil, tandis que d'autres tiennent une sorte d'album de tous les faits et gestes de leur bambin. À chacun ses centres d'intérêt !

Découvrez la collecte de données personnelles via Twitter à l'adresse *http://your.flowingdata.com*.

YFD a été conçu à l'origine en pensant aux données personnelles, mais nombreux ont trouvé l'application utile pour des données plus générales, comme l'activité sur le Web ou les horaires des trains.

Compromis

Même si ces outils sont simples d'utilisation, ils présentent quelques inconvénients. En échange de quelques opérations de glisser-déplacer, vous abandonnez une certaine flexibilité dans ce que vous pouvez faire. Vous pouvez généralement modifier les couleurs, les polices et les titres, mais vous êtes limité aux fonctions du logiciel. S'il n'existe pas de bouton pour le graphique que vous souhaitez, pas de chance, voilà tout !

À l'opposé, certains logiciels proposent un très grand nombre de fonctions, mais en retour une multitude de boutons que vous devez découvrir. Par exemple, il y avait un programme (non répertorié ici) pour lequel je suivis un cours intensif sur un week-end et, de toute évidence, il était très puissant si j'y consacrais le temps nécessaire. Cependant, l'interface était si peu intuitive que je finis par renoncer.

En outre, il était difficile de reproduire mon travail pour différents ensembles de données, car je devais me souvenir de chaque bouton sur lequel j'avais cliqué.

Par comparaison, quand vous gérez vos données à l'aide de code, il est souvent facile de réutiliser celui-ci et de l'appliquer à un autre ensemble de données.

Ne vous méprenez pas. Je ne dis pas qu'il faut éviter à tout prix les logiciels prêts à l'emploi. Ils vous permettent d'explorer vos données rapidement et aisément. Mais, lorsque vous travaillerez avec un plus grand nombre d'ensembles de données, parfois le logiciel ne sera pas adapté et vous devrez vous tourner vers la programmation.

Programmation

Je ne le soulignerai jamais assez : acquérez quelques compétences en programmation et vous pourrez obtenir bien plus de vos données qu'à l'aide d'un simple logiciel prêt à l'emploi. Les compétences en programmation vous procurent une plus grande souplesse et une meilleure adaptabilité aux différents types de données.

S'il vous est arrivé d'être impressionné par un graphique de données qui paraissait fait sur mesure, il avait été probablement créé à l'aide de code ou dans un logiciel d'illustration. Et même avec les deux, la plupart du temps. Je reviendrai plus tard sur les logiciels d'illustration.

Le code peut sembler énigmatique aux yeux des débutants (j'en ai fait partie). Aussi considérez-le comme une nouvelle langue, car c'est bien de cela qu'il s'agit. Chaque ligne du code ordonne à l'ordinateur de faire telle ou telle action. Comme votre ordinateur ne comprend pas la façon dont vous parlez à vos amis, vous devez vous adresser à l'ordinateur dans sa propre langue ou syntaxe.

Comme pour toute langue, vous ne pouvez pas démarrer une conversation immédiatement. Commencez d'abord par les bases, puis gravissez les échelons l'un après l'autre ! Avant même de vous en rendre compte, vous serez en train de coder. L'un des bons côtés de la programmation est qu'une fois que vous avez appris un langage, il est beaucoup plus aisé d'en apprendre un nouveau, car la logique est identique.

Options

Ainsi vous décidez de mettre les mains dans le cambouis… pardon, dans le code ! Excellente initiative. Un grand nombre d'options sont disponibles gratuitement. Certains langages sont plus efficaces que d'autres pour telle ou telle tâche. Certaines solutions peuvent gérer des quantités importantes de données, tandis que d'autres ne sont pas aussi robustes sur ce point, mais peuvent produire de bien meilleurs visuels ou fournir une interaction. Le choix du langage dépend grandement de votre aisance à le manipuler et des objectifs assignés à votre graphique de données.

Certains développeurs s'en tiennent à un seul langage et s'attellent à le connaître parfaitement. Si vous débutez en programmation, je recommande vivement cette stratégie. Familiarisez-vous avec les bases et les concepts importants du code.

Choisissez le langage qui correspond le mieux à vos besoins. Cependant, il est plaisant d'apprendre de nouveaux langages et de nouvelles façons de manipuler les données ; par conséquent, acquérez une bonne expérience en programmation avant de choisir votre solution favorite.

Python

Le précédent chapitre traitait de la façon dont Python peut gérer les données. Python excelle en la matière et peut gérer de grandes quantités de données sans aucun problème. Le langage est ainsi particulièrement utile pour les analyses et les calculs volumineux. Python possède aussi une syntaxe claire et facile à lire, que les programmeurs apprécient. Il est également possible de tirer parti d'une multitude de modules pour créer des graphiques de données, comme que celui de la figure 3-11.

D'un point de vue esthétique, le résultat est moins brillant. Vous ne proposerez pas directement à la publication un graphique créé en Python. À coup sûr, il n'est pas d'un raffinement extrême. Néanmoins, ce peut être un bon point de départ pour explorer les données. Vous pouvez aussi exporter les images et les retoucher ou ajouter des informations à l'aide d'un logiciel d'édition graphique.

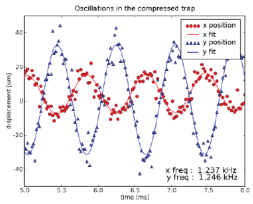


Figure 3-11 *Graphique créé en Python*

Ressources Python utiles

- Site web officiel de Python (*http://python.org*)
- NumPy and SciPy (http://www.scipy.org/Download) – Informatique scientifique

PHP

PHP fut le premier langage que j'ai appris lorsque j'ai commencé la programmation pour le Web. Certains le trouvent quelque peu touffu – ce qu'il peut être – mais vous pouvez sans peine y établir une certaine organisation. Sa configuration est simple, parce que la plupart des serveurs web en sont déjà pourvus et que, de ce fait, il est facile de… sauter à l'eau !

Il existe une bibliothèque graphique flexible appelée GD, généralement présente dans les installations standards. Elle permet de créer des images à partir de rien ou de manipuler celles existantes. Il existe également un certain nombre de bibliothèques graphiques PHP qui rendent possible la création de graphiques élémentaires. La plus connue est la bibliothèque graphique PHP Sparkline, qui permet d'incorporer à un texte de petits graphiques de la taille d'un mot et d'ajouter un composant visuel à un tableau numérique (figure 3-12).

Souvent, PHP est couplé avec une base de données telle que MySQL. Cela évite d'avoir à travailler avec une multitude de fichiers CSV et permet d'optimiser l'utilisation du langage et de manipuler des ensembles de données volumineux.

Ressources PHP utiles

- Site web officiel de PHP (*http://php.net*)
- Bibliothèque graphique PHP Sparkline (*http://sparkline.org*)

Figure 3-12 *Sparklines utilisant une bibliothèque graphique PHP*

Processing

Processing est un langage de programmation open source destiné aux auteurs-et aux artistes des données. Le langage a commencé comme « carnet à dessin informatique » grâce auquel vous pouviez créer des graphiques rapidement. Depuis, il a connu de nombreux développements et plusieurs projets de haute qualité ont été créés avec Processing, par exemple *We Feel Fine* (cf. chapitre 1).

La grande force de Processing est qu'il est très rapidement opérationnel. L'environnement de programmation est léger, et quelques lignes de code suffisent pour créer un graphique animé et interactif. Il sera, certes, assez élémentaire, mais comme il a été conçu dans le but de créer des visuels, vous apprendrez sans peine à produire des graphiques plus élaborés.

Même si, à l'origine, le public ciblé se composait d'auteurs et d'artistes, la communauté autour de Processing s'est peu à peu diversifiée. De nombreuses bibliothèques vous aideront à tirer le meilleur parti du langage.

L'un des inconvénients de Processing est que vous vous retrouvez avec une applet Java, dont le chargement peut être lent sur certains ordinateurs, et par ailleurs, tous les utilisateurs n'ont pas Java. Il existe une solution à cela, à savoir une version JavaScript de Processing dont le développement s'est achevé récemment et qui est prête à être utilisée.

Néanmoins, il s'agit là d'un excellent point de départ pour les débutants. Même les personnes n'ayant aucune expérience en programmation peuvent créer quelque chose d'utile.

Ressources Processing utiles

Processing (*http://processing.org*) – Site web officiel de Processing

Flash et ActionScript

La majorité des graphiques de données interactifs et animés que l'on trouve sur le Web. Il est possible de créer des graphiques uniquement avec Flash, interface de type cliquer-déplacer, mais ActionScript permet un meilleur contrôle des interactions. Beaucoup d'applications sont écrites totalement en ActionScript, sans l'environnement Flash. Cependant, le code se compile comme une application Flash.

Même s'il existe de nombreuses bibliothèques ActionScript open source gratuites, le logiciel et les développeurs Flash sont coûteux. Tenez-en compte au moment de choisir votre application.

Par exemple, la carte interactive qui montre la croissance de Walmart sous forme animée (figure 3-13), a été écrite en ActionScript. La bibliothèque utilisée, Modest Maps, est une bibliothèque d'interaction et d'affichage destinée aux cartes modulaires. Elle est sous licence BSD, ce qui signifie qu'elle est gratuite et que vous pouvez l'exploiter à votre guise.

Le graphique interactif de la figure 3-14 a également été écrit en ActionScript. Il permet d'explorer les différentes catégories de dépenses au fil des ans. Le plus dur ou presque a été réalisé à l'aide de la bibliothèque Flare ActionScript de l'UC Berkeley Visualization Lab.

Pour créer des graphiques interactifs sur le Web, Flash et ActionScript sont une excellente option. Les applications Flash se chargent relativement vite et Flash est déjà installé sur la plupart des ordinateurs.

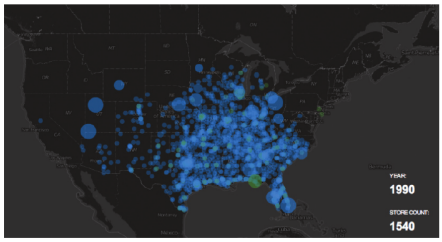


Figure 3-13 *Carte animant la croissance de Walmart et écrite en ActionScript*

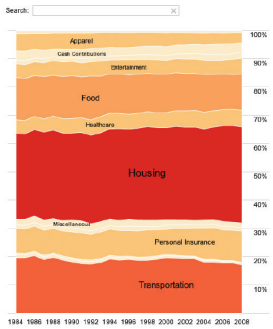


Figure 3-14 *Graphique interactif écrit en ActionScript et illustrant la ventilation des dépenses de consommation*

Ce n'est pas le langage le plus aisé à assimiler, pas tant à cause de la syntaxe, relativement simple, que parce que l'installation et l'organisation du code peuvent être rebutantes pour le débutant. Vous n'obtiendrez pas une application qui s'exécute en quelques lignes de code, comme avec Processing. Les prochains chapitres vous conduiront à travers les étapes élémentaires et vous trouverez en ligne un grand nombre de didacticiels utiles, Flash étant très largement utilisé.

De même, comme les navigateurs web ne cessent de s'améliorer en vitesse et en efficacité, un nombre croissant d'alternatives s'offrent à vous.

Ressources Flash et ActionScript utiles

- Adobe Support (*http://www.adobe.com/products/flash/whatisflash/*) – Documentation officielle de Flash et ActionScript (et autres produits Adobe)
- Boîte à outils Flare pour la création de visualisations (*http://flare.prefuse.org*)
- Modest Maps (*http://modestmaps.com*)

HTML, JavaScript et CSS

Les navigateurs web s'exécutent de plus en plus vite et leurs fonctionnalités ne cessent de s'améliorer. Nombre de personnes utilisent infiniment plus le navigateur que toute autre application de leur ordinateur. Plus récemment, il s'est produit une évolution vers la visualisation s'exécutant en mode natif dans le navigateur via HTML, JavaScript et CSS. Les graphiques de données étaient pour l'essentiel créés en Flash et ActionScript s'ils comportaient un composant interactif ou enregistrés comme image statique. C'est encore souvent le cas, mais il n'existait à dire vrai pas beaucoup d'autres options.

Désormais, plusieurs packages et bibliothèques peuvent vous aider à développer rapidement des visualisations interactives ou statiques. Ils proposent aussi une multitude d'options grâce auxquelles vous personnaliserez les outils en fonction de vos besoins en termes de données.

Par exemple, D3, géré par le Stanford Visualization Group, est une bibliothèque de visualisations gratuite et open source, qui permet de créer des visualisations en mode natif sur le Web. D3 propose un certain nombre de visualisations prêtes à l'emploi, mais vous n'êtes en rien limité par ce que vous pouvez faire, géométriquement parlant. La figure 3-15 illustre un graphique empilé, qui peut être interactif. Ce type de graphique est intégré à D3, mais vous pouvez aussi vous satisfaire d'un graphique de flux, comme celui de la figure 3-16.

Pour bénéficier de fonctionnalités accrues, vous pouvez aussi exploiter plusieurs bibliothèques. Cela est possible en Flash, mais JavaScript peut se révéler beaucoup moins lourd en termes de code. JavaScript est aussi plus facile à lire et à utiliser avec des bibliothèques telles que jQuery et MooTools. Ces outils ne sont pas dédiés spécifiquement à la visualisation, mais ils sont utiles. Ils assurent un grand nombre de fonctions élémentaires en quelques lignes de code seulement.

Sans les bibliothèques, vous auriez à écrire beaucoup plus de lignes et votre code deviendrait très vite embrouillé.

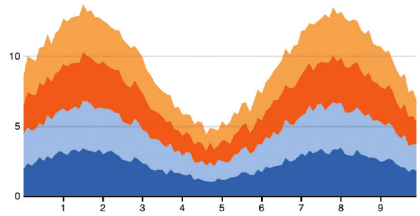


Figure 3-15 *Graphique empilé créé avec D3*

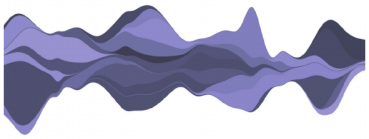


Figure 3-16 *Graphique de flux réalisé sur mesure avec D3*

Les plug-ins des bibliothèques vous aideront aussi pour certains de vos graphiques de base. Par exemple, vous pouvez utiliser un plug-in Sparkline pour jQuery et créer un graphique de taille réduite (figure 3-17).

Figure 3-17 *Sparklines avec le plug-in jQuery*

Le même résultat peut être obtenu avec PHP, mais la présente méthode offre quelques avantages. Tout d'abord, le graphique est généré dans le navigateur de l'utilisateur, et non sur le serveur. La charge de vos ordinateurs s'en trouve

allégée, ce qui est précieux lorsque vous possédez un site web confronté à un trafic important.

L'autre avantage est que vous n'avez nul besoin de configurer votre serveur avec la bibliothèque graphique PHP. La plupart des serveurs sont configurés avec les fonctionnalités graphiques, mais tel n'est pas toujours le cas. L'installation peut alors être fastidieuse, si le système ne vous est pas familier.

Vous pouvez également ne pas utiliser du tout de plug-in. Vous pouvez aussi concevoir une visualisation personnalisée à l'aide d'une programmation web standard. La figure 3-18, par exemple, représente un calendrier interactif qui fait aussi office de « carte chaude » (*heatmap*)dans your.flowingdata.

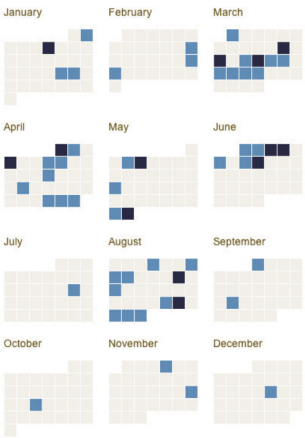


Figure 3-18 *Calendrier interactif servant aussi de carte chaude dans your.flowingdata*

Tout n'est pas parfait, cependant. Comme ces logiciels et ces technologies sont relativement récents, vos créations n'auront peut-être pas le même aspect dans les différents navigateurs. Certains outils précédemment mentionnés ne fonctionneront pas correctement dans un ancien navigateur tel qu'Internet Explorer 8. À dire vrai, ce n'est plus guère un problème, car la plupart des personnes utilisent des navigateurs modernes tels que Firefox ou Google Chrome. En fin de compte, tout dépend de vos visiteurs. Moins de 5 % de ceux de FlowingData ont recours à d'anciennes versions d'Internet Explorer et la compatibilité n'est donc pas réellement un problème.

Pour des raisons d'ancienneté également, il n'existe pas autant de bibliothèques accessibles pour la visualisation en JavaScript qu'il n'y en a en Flash. Telle est la raison pour laquelle la plupart des grandes entreprises d'informations continuent à utiliser Flash, mais la situation évoluera au fur et à mesure du développement.

Ressources HTML, JavaScript et CSS utiles

- jQuery (*http://jquery.com/*) – Bibliothèque JavaScript qui permet d'accroître l'efficacité du code et de simplifier la lecture du produit final.
- jQuery Sparklines (*http://omnipotent.net/jquery.sparkline/*) – Créer des sparklines statiques et animées en JavaScript.
- D3 (*http://d3js.org/*) – Bibliothèque JavaScript propre à la visualisation et conçue pour apprendre par l'exemple.
- JavaScript InfoVis Toolkit (*http://datafl.ws/15f*) – Autre bibliothèque de visualisation, moins riche que D3.
- paperjs (*paperjs.org*) – Qui a la particularité d'être plus orienté sur l'élément html canvas.
- raphael (*raphaeljs.com*) – Dont la particularité est la compatibilité avec IE6.
- *mootools.net* – Environnement dont le périmètre est comparable à celui de jQuery.
- Google Charts API (*http://code.google.com/apis/chart/*) – Créer des graphiques classiques à la volée, en modifiant simplement l'URL.

Logiciel R

Si vous lisez FlowingData, vous savez probablement que mon logiciel favori pour les graphiques de données est R. Il s'agit d'un logiciel de calcul statistique gratuit et open source, qui offre aussi de bonnes fonctionnalités graphiques en matière de statistiques. C'est aussi le choix de la plupart des statisticiens comme logiciel d'analyse. Il existe d'autres solutions payantes, comme S-plus et SAS, mais il est difficile de rivaliser avec la gratuité et une communauté de développement active.

L'un des avantages de R sur les logiciels précités est qu'il a été spécifiquement conçu pour analyser les données. HTML permet de créer des pages web et Flash bien d'autres choses, comme les publicités animées ou les vidéos. R, de son

côté, a été conçu et continue d'être maintenu par des statisticiens pour des statisticiens, ce qui peut être une bonne ou une mauvaise chose selon le point de vue que l'on adopte.

Il existe une multitude de packages R qui permettent de créer des graphiques de données avec quelques lignes de code seulement après avoir chargé vos données dans R. Par exemple, vous pouvez rapidement créer une arborescence de rectangles (*treemap*) avec le package Portfolio (figure 3-19). Tout aussi simplement, vous pouvez créer une carte chaude (figure 3-20). Et, bien sûr, vous pouvez aussi créer d'autres graphiques statistiques traditionnels, comme les graphiques en nuage de points ou les graphiques chronologiques, traités au chapitre 4.

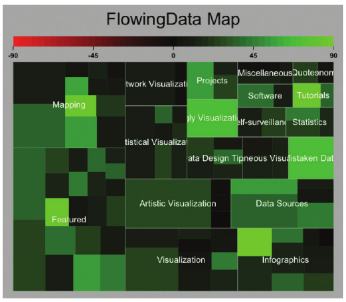


Figure 3-19 *Arborescence de rectangles générée dans le logiciel R avec le package Portfolio*

Pour être complètement honnête, toutefois, le site de R semble horriblement obsolète (figure 3-21) et le logiciel lui-même n'est guère efficace pour guider les nouveaux utilisateurs. Cependant, n'oubliez pas que R est un langage de programmation et que vous vous retrouverez dans la même situation avec n'importe quel autre langage. Les quelques critiques que j'ai lues sur R sont généralement écrites par des personnes habituées surtout à cliquer sur un bouton ou à glisser-déplacer un élément. Par conséquent, dans le cas de R, ne vous attendez pas à une interface à base de clics, sans quoi elle risque fort de vous paraître peu conviviale.

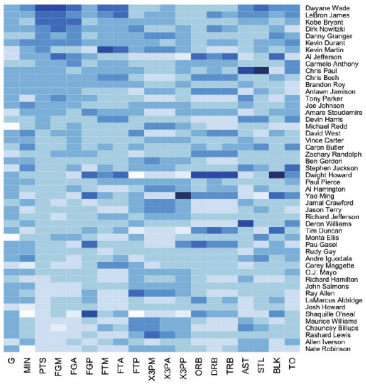


Figure 3-20 *Carte chaude générée dans R*

Ceci dit, le logiciel R vous permettra de réaliser beaucoup de choses. Vous pourrez créer des graphiques (ou du moins leurs prémisses) dont la qualité équivaut à celle d'une publication ou apprendre à adopter la souplesse de R. Si vous le désirez, vous pourrez écrire vos propres fonctions et packages pour personnaliser les graphiques, ou utiliser ceux créés par d'autres et mis à disposition dans la bibliothèque R.

R propose des fonctions de dessin élémentaires qui, pour l'essentiel, vous permettent de dessiner ce que vous souhaitez. Vous pouvez tracer des lignes, des formes ou des axes, et comme pour les autres solutions de programmation, une seule limite : votre imagination ! Là encore, pratiquement tous les types de graphique sont disponibles via un package R ou un autre.

Pourquoi utiliser une autre application en plus de R ? Pourquoi ne pas simplement tout faire dans R ? Voici quelques raisons. Dans la mesure où le logiciel R est à utiliser sur un ordinateur, il n'est pas idéalement adapté au Web dynamique. L'enregistrement de graphiques, d'images ou leur insertion sur une page

web n'est pas un problème, mais ces actions n'interviennent pas automatiquement. Il est possible de générer des graphiques à la volée sur le Web, mais jusqu'à présent, les solutions ne sont pas particulièrement robustes si vous les comparez à ce que permettent des solutions natives du Web comme JavaScript.

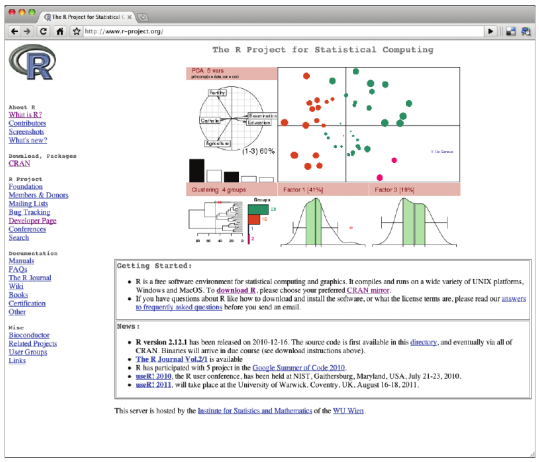


Figure 3-21 *Page d'accueil du site officiel de R (http://www.r-project.org)*

R n'est pas aussi efficace en ce qui concerne les graphiques interactifs et les animations. Bien sûr, vous pouvez en créer dans R, mais il existe des solutions bien meilleures et plus élégantes pour y parvenir, ne serait-ce qu'avec Flash ou Processing.

Enfin, il se peut que vous ayez remarqué que les graphiques des figures 3-19 et 3-20 manquent d'un certain éclat. Il est peu probable que vous les retrouviez

dans un journal. Vous pouvez améliorer l'aspect des graphiques de R en bricolant différentes options ou en écrivant du code supplémentaire. Cependant, ma stratégie consiste généralement à créer le graphique de base en R, puis à le retoucher et à l'embellir dans un logiciel tel qu'Illustrator, que j'évoquerai bientôt. Pour les analyses, la sortie brute de R convient parfaitement, mais pour la présentation et la narration, il est préférable de peaufiner l'esthétique.

Ressources R utiles

R Project for Statistical Computing (*http://www.r-project.org*)

Conseil

Lorsque vous effectuez une recherche sur R à l'aide de moteurs de recherche du Web, vous risquez d'être quelque peu troublé par les résultats obtenus. Aussi, au lieu de saisir comme mot-clé « R », préférez « r-project », complété par l'objet proprement dit de votre recherche. Les résultats obtenus seront bien plus pertinents.

Compromis

Apprendre à programmer revient à découvrir un nouveau langage. Il s'agit, en quelque sorte, de la langue de votre ordinateur, composée de bits, et de sa logique. Lorsque vous utilisez Excel ou Tableau, par exemple, vous travaillez essentiellement avec un traducteur. Les boutons et les menus sont affichés dans votre langue et, lorsque vous cliquez sur un élément, le logiciel traduit votre interaction et transmet le résultat de la traduction à votre ordinateur. Celui-ci exécute alors l'action souhaitée, comme créer un graphique ou traiter certaines données.

À n'en pas douter, le temps est un obstacle majeur et il en faut pour s'initier à un nouveau langage. Pour beaucoup de personnes, cette difficulté est de taille et je suis sensible à leur argument. Elles ont besoin que le travail soit effectué sur-le-champ, car elles ont sous les yeux une multitude de données et plusieurs utilisateurs attendent les résultats. Si tel est le cas, à savoir que vous n'avez que cette tâche concernant les données et rien d'autre en perspective, il est préférable que vous recouriez aux outils de visualisation prêts à l'emploi.

Cependant, si vous voulez traiter vous-même vos données et qu'à l'avenir, vous serez amené à gérer beaucoup de projets liés aux données, le temps consacré à l'apprentissage de la programmation pourra se révéler être un gain de temps précieux et se traduire par des résultats impressionnants. Vous progresserez en programmation à chacun de vos projets et trouverez l'écriture du code de plus en plus simple. Comme pour une langue étrangère, vous ne commencez pas par écrire des livres dans cette langue, mais par les notions de base, puis vous étendez vos capacités.

Il est possible de voir les choses d'une autre façon. Imaginez que vous vous retrouviez dans un pays étranger, dont vous ne parlez pas la langue. Pour communiquer, vous faites appel à un traducteur. Mais que se passe-t-il si le traducteur ne connaît pas la signification ou le mot exact correspondant à ce que vous voulez dire ? Il peut ignorer le mot ou le rechercher dans un dictionnaire.

Pour les outils de visualisation prêts à l'emploi, le logiciel équivaut au traducteur. S'il ne sait accomplir telle ou telle opération, vous êtes bloqué et devez essayer une autre méthode. Contrairement au traducteur oral, le logiciel n'apprend pas instantanément de nouveaux mots ou, dans notre cas, de nouveaux types de graphiques ou de nouvelles fonctions de gestion des données. Ces dernières se présentent sous la forme de mises à jour logicielles, qu'il vous faut attendre. Et si vous appreniez le langage (la langue) vous-même ?

Une fois encore, je ne suis pas en train de dire qu'il faut éviter les outils prêts à l'emploi. Je les utilise en permanence. Ils rendent simples et rapides un grand nombre de tâches fastidieuses, ce qui est excellent. Simplement, il ne faut pas que le logiciel vous restreigne.

Comme nous le verrons dans les prochains chapitres, la programmation permet de faire beaucoup avec un effort moindre que si vous exécutiez la même opération manuellement. Ceci dit, certaines choses sont mieux faites à la main, notamment quand il s'agit de raconter une histoire à partir de données. Ceci nous conduit à la prochaine section sur l'illustration : l'extrémité opposée du spectre de la visualisation.

Illustration

Nous voici maintenant dans la zone de confort des graphistes. Si vous êtes analyste ou intervenez dans un domaine plus technique, il s'agit probablement d'un espace inhabituel pour vous. Vous pouvez obtenir d'excellents résultats en combinant code et outils de visualisation prêts à l'emploi, mais les graphiques de données que vous obtiendrez donneront presque toujours l'impression d'avoir été générés automatiquement. Peut-être les étiquettes ne sont-elles pas à leur place ou une légende paraît-elle surchargée. Pour les analyses, pas de réel problème, car vous savez en principe ce que vous avez sous les yeux.

Cependant, lorsque vous créez des graphiques pour une présentation, un rapport ou une publication, plus ils sont bien conçus, plus il est possible de voir clairement l'histoire que vous racontez.

Par exemple, la figure 3-19 est la sortie brute de R. Elle illustre les vues et les commentaires sur FlowingData de 100 billets populaires, classés par catégorie, telle que Mapping. Plus le vert est clair, plus le billet contient de commentaires et plus le rectangle est grand, plus le nombre de vues est élevé. L'original ne vous l'aurait pas appris, mais lorsque je regardais les nombres, je savais ce que j'avais sous les yeux, car c'était moi qui avais écrit le code en R.

La figure 3-22 propose une version révisée. Les étiquettes ont été ajustées afin d'être toutes lisibles ; une introduction a été ajoutée en haut pour informer le lecteur du contenu proposé et la partie rouge de la légende a été supprimée. J'ai également modifié la couleur de fond de gris en blanc pour la rendre plus séduisante.

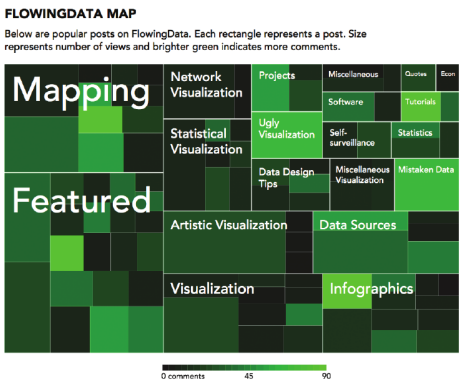


Figure 3-22 *Arborescence de rectangles créée en R et modifiée dans Illustrator*

J'aurais pu aussi modifier le code pour l'adapter à mes besoins spécifiques, mais il fut beaucoup plus simple d'utiliser le glisser-déplacer dans Illustrator. Vous pouvez créer intégralement un graphique à l'aide d'un logiciel d'illustration ou importer un graphique réalisé avec R, par exemple, et le modifier à votre guise. Dans le premier cas, les choix de visualisation sont limités, parce que la visualisation n'est pas l'objectif principal du logiciel. Pour un graphique plus complexe qu'un simple graphique en barres, le mieux est de recourir à la seconde solution. Sinon, vous aurez un travail conséquent à exécuter manuellement, ce qui est source d'erreurs.

La grande force d'un logiciel d'illustration est que vous avez un meilleur contrôle de chaque élément et que vous pouvez tout faire en cliquant et en déplaçant : par exemple, modifier la couleur d'une ou de plusieurs barres, modifier la largeur des axes ou annoter les caractéristiques importantes en quelques clics de souris.

Options

Il existe de nombreux programmes d'illustration, mais la plupart d'entre nous n'en utilisons que quelques-uns, pour ne pas dire un seul. Le prix constitue sans doute le critère majeur, sachant qu'il varie de zéro (dans le cas des logiciels gratuits et open source) à plusieurs centaines d'euros.

Illustrator

Il y a fort à parier que tout graphique de données statiques personnalisé ou publié dans un grand journal soit passé, à un moment ou un autre, dans Illustrator, qui représente le logiciel standard de l'industrie de l'illustration. Chaque graphique imprimé dans *The New York Times* a été créé ou retouché dans Illustrator.

Illustrator est très réputé dans le monde de l'impression, car il utilise des vecteurs à la place des pixels. Cela signifie que vous pouvez agrandir le graphique sans diminuer la qualité de l'image. Par comparaison, si vous deviez agrandir une photo de faible résolution, à savoir définie avec un certain nombre de pixels, vous obtiendrez une image pixélisée.

À l'origine, le logiciel a été conçu pour le développement de polices ; il s'est ensuite répandu parmi les concepteurs d'illustrations telles que logos ou graphiques à vocation plus artistique. Et telle est aujourd'hui encore la principale utilisation d'Illustrator.

Cependant, Illustrator propose quelques fonctionnalités graphiques élémentaires via son outil dédié, Graphe. Il est possible de créer les types de graphiques les plus simples, tels que les graphiques en barres, en camembert ou graphiques de séries temporelles. Vous pouvez coller les données dans une feuille de calcul réduite, mais c'est tout ce que vous pouvez faire en termes de gestion des données.

Le bon côté d'Illustrator, en termes de graphiques de données, est sa flexibilité et sa simplicité d'utilisation, avec un grand choix de boutons et de fonctions. Cela peut paraître déroutant de prime abord, car les boutons sont extrêmement nombreux ; cependant, il est facile de choisir le bouton approprié, comme nous le verrons au chapitre 4. C'est cette souplesse qui permet aux meilleurs concepteurs de données de créer les graphiques les plus clairs et les plus concis.

Illustrator est disponible pour Windows et Mac. Son inconvénient majeur est son coût. En effet, celui-ci est élevé si vous le comparez à ce que vous pouvez faire gratuitement en codant, en considérant évidemment que vous possédez l'ordinateur sur lequel installer les éléments nécessaires. Cependant, au regard

de certaines solutions prêtes à l'emploi, le prix d'Illustrator peut paraître relativement raisonnable.

Au moment où ces lignes sont écrites, la dernière version d'Illustrator coûte environ 780 € sur le site Adobe, mais vous devriez pouvoir trouver des offres plus intéressantes sur d'autres sites (sinon, choisissez une version plus ancienne). Comme Adobe propose aussi des réductions importantes aux étudiants, ne manquez pas de les consulter. (C'est le logiciel le plus onéreux que j'ai jamais acheté, mais je l'utilise presque tous les jours).

Ressources Illustrator utiles

- Page produit Illustrator (*http://www.adobe.com/products/illustrator/*)
- VectorTuts (*http://vectortuts.com*) – Didacticiels complets et simples sur l'utilisation d'Illustrator

Inkscape

Les graphiques de données de cet ouvrage ont été améliorés dans Illustrator. Les mêmes opérations peuvent être réalisées avec Inkscape, qui propose de nombreux outils et fonctions identiques et ayant les mêmes noms.

Inkscape est l'alternative gratuite et open source d'Illustrator. Si vous voulez éviter de trop dépenser, Inkscape est le meilleur choix que vous puissiez faire. J'utilise toujours Illustrator, parce que lorsque j'ai commencé à apprendre les aspects les plus pointus des graphiques de données, Illustrator était le logiciel que tout le monde se servait au travail. J'ai entendu beaucoup de bien sur Inkscape et comme il est gratuit, vous ne vous engagez à rien en l'essayant. À noter toutefois qu'il existe peu de ressources consacrées à l'utilisation du logiciel.

Ressources Inkscape utiles

- Inkscape (*http://inkscape.org*)
- Didacticiels Inkscape (*http://inkscapetutorials.wordpress.com/*)

Autres

Illustrator et Inkscape ne sont certainement pas les seules options disponibles pour créer et améliorer des graphiques de données. Mais il s'agit des deux programmes les plus couramment utilisés. Peut-être adorerez-vous un logiciel tel que Corel Draw, qui ne fonctionne que sous Windows et dont le prix est similaire, voire légèrement inférieur à celui d'Illustrator.

Parmi les autres programmes, citons Raven (Aviary) et Lineform, qui proposent un ensemble d'outils moins étendu. N'oubliez pas qu'Illustrator et Inkscape sont des outils généraux pour les concepteurs graphiques et que, par conséquent, ils proposent un grand nombre de fonctionnalités. Bien sûr, si vous n'apportez que quelques retouches à vos graphiques, une solution plus simple (et moins onéreuse) conviendra parfaitement.

Compromis

Les logiciels d'illustration sont destinés à… l'illustration. Ils ne sont pas conçus spécifiquement pour les graphiques de données. Comme ils sont dédiés à la création graphique, la plupart des utilisateurs ne recourent pas aux fonctions proposées par Illustrator ou Inkscape. Ils n'excellent pas non plus dans la manipulation de données volumineuses, si nous les comparons à la programmation ou à l'emploi d'outils de visualisation. Pour cette raison, il n'est pas possible d'explorer les données à l'aide de ces programmes.

Ceci dit, ils sont indispensables pour créer des graphiques de données de qualité satisfaisante pour la publication. Ils ne se contentent pas de contribuer à l'esthétique, mais participent aussi à la lisibilité et à la clarté, souvent difficiles à obtenir avec une sortie générée automatiquement.

Cartographie

Les outils de visualisation présentés dans les sections précédentes et ceux que vous utilisez pour cartographier les données géographiques se recouvrent partiellement. Cependant, la quantité de données géographiques s'est considérablement accrue ces dernières années, de même que les solutions de cartographie disponibles. Avec l'augmentation des services d'emplacement mobiles, il y aura de plus en plus de données auxquelles seront associées leurs coordonnées exprimées en latitude et longitude. Les cartes constituent aussi une solution incroyablement intuitive pour visualiser les données, et cela mérite que nous nous y attardions quelques instants.

Au début du Web, la cartographie n'était ni une tâche aisée, ni une tâche élégante. Souvenez-vous de l'époque où sur MapQuest, vous recherchiez une direction et obteniez une petite carte statique ? Yahoo! ne fit guère mieux pendant un certain temps.

Jusqu'à ce que, quelques années plus tard, Google propose une *carte glissante* (figure 3-23). La technologie existait depuis un moment, mais elle ne se révéla utile que lorsque la connexion Internet fut assez rapide pour permettre la mise à jour en continu. Aujourd'hui, les cartes glissantes sont celles auxquelles nous sommes habitués. Nous pouvons choisir sans peine un zoom avant ou arrière et, dans certains cas, les cartes ne servent pas qu'aux seules directions ; elles constituent la principale interface pour explorer un ensemble de données.

Les cartes glissantes sont devenues pratiquement universelles. Les grandes cartes, qui en principe ne pourraient pas être affichées sur un écran d'ordinateur, sont divisées en images plus petites, appelées aussi mosaïques ou tuiles. Seules s'affichent les tuiles qui correspondent à la taille de la fenêtre de l'écran, tandis que le reste demeure masqué à la vue. Lorsque vous faites glisser la carte, les autres tuiles apparaissent et vous donnent ainsi l'illusion de vous déplacer autour d'une seule et même carte. Peut-être avez-vous déjà vu un résultat similaire avec les photographies haute résolution.

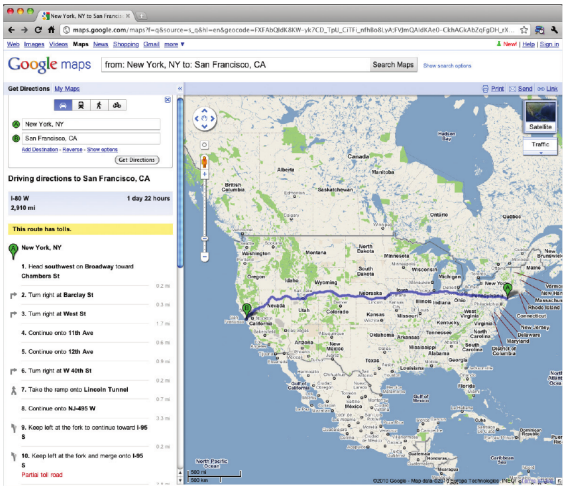


Figure 3-23 *Google Maps : recherche d'un itinéraire*

Options

Tandis que la création de données géographiques se développait au sein du grand public, divers outils permettant de cartographier ces données ont aussi vu le jour. Certains ne nécessitent qu'une légère dose de programmation pour être pleinement opérationnels, tandis que d'autres nécessitent un peu plus de travail. Il existe aussi quelques autres solutions qui ne nécessitent pas de programmation.

Google, Yahoo! et Microsoft Maps

C'est la solution en ligne la plus simple, même si elle requiert quelques lignes de programmation. Meilleur est le code, meilleur est le résultat que vous obtenez des API de cartographie proposées par Google, Yahoo! et Microsoft.

Les fonctionnalités de base des trois services sont assez similaires, mais si vous débutez, commencez par Google. Il semble que ce soit la solution la plus fiable. Google propose des API de cartographie écrites en JavaScript et en Flash, ainsi que quelques autres services tels que le géocodage ou le calcul d'itinéraire long. Parcourez le didacticiel d'initiation, puis intéressez-vous aux autres éléments, comme le placement de marqueurs (figure 3-24), le tracé d'itinéraires et l'ajout de calques. Le riche ensemble d'extraits de code et de didacticiels doit vous permettre de devenir rapidement opérationnel.

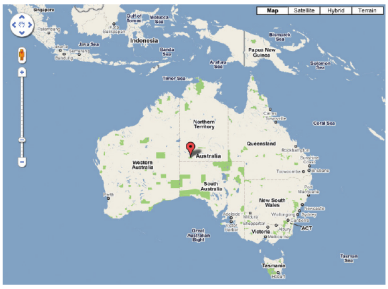


Figure 3-24 *Placement de marqueurs sur Google Maps*

Yahoo! propose aussi des API en JavaScript et en Flash pour la cartographie, ainsi que quelques services de géolocalisation. Je ne sais pas combien de temps ils seront disponibles, au regard de la situation actuelle de l'entreprise. Au moment où j'écris ces lignes, Yahoo! semble avoir délaissé les applications et le développement au profit de la fourniture de contenu. Microsoft propose aussi une API JavaScript (sous le nom Bing) et une en Silverlight, la réponse de Microsoft à Flash.

Ressources d'API de cartographie utiles

- API pour Google Maps (*http://code.google.com/apis/maps/*)
- Services web Yahoo! Maps (*http://developer.yahoo.com/maps/*)
- API pour Bing Maps (*http://www.microsoft.com/maps/developers/web.aspx*)

ArcGIS

Les services de cartographie précédemment mentionnés sont assez rudimentaires quant aux fonctions proposées. Si vous souhaitez des fonctionnalités plus élaborées, vous devrez probablement les implémenter vous-même. L'application ArcGIS, conçue pour la cartographie sur ordinateur de bureau, se situe à l'opposé. Il s'agit d'un programme imposant qui permet de cartographier une multitude de données et de leur appliquer divers traitements. Toutes les opérations s'effectuent via l'interface utilisateur et aucun code n'est requis.

Presque tous les départements graphiques avec lesquels collaborent des spécialistes en cartographie utilisent ArcGIS. Les cartographes professionnels eux-mêmes se servent d'ArcGIS. Certains l'aiment au-delà de toute mesure. Aussi, si vous souhaitez créer des cartes détaillées, ArcGIS mérite plus qu'un détour.

Je n'ai utilisé ArcGIS que pour quelques projets, car j'ai plutôt tendance à choisir la voie de la programmation quand je le peux et, en outre, je n'ai pas besoin de toute la panoplie de fonctionnalités proposées. L'inconvénient d'un ensemble aussi riche est qu'il y a une infinité de boutons et de menus à parcourir. Les solutions en ligne et serveur sont également disponibles, mais elles semblent peu pratiques comparées aux autres implémentations.

Ressources ArcGIS utiles

Page produit ArcGIS (*http://www.esri.com/software/arcgis/*)

Modest Maps

J'ai déjà mentionné Modest Maps et en ai proposé un exemple à la figure 3-13 qui illustre la croissance de Walmart. Modest Maps est une bibliothèque Flash et ActionScript pour les cartes composées de tuiles. La prise en charge de Python est également assurée. L'application est maintenue par un groupe d'individus qui connaissent la cartographie en ligne et font un excellent travail pour leurs clients, mais aussi pour le plaisir – ce qui en dit long sur l'extraordinaire qualité de la bibliothèque.

L'aspect original de Modest Maps est qu'il s'agit plus d'une infrastructure que d'une API de cartographie, comme celle proposée par Google. Cette infrastructure fournit le strict minimum de ce qui est nécessaire pour créer une carte en ligne, puis elle vous laisse implémenter ce que vous souhaitez. Vous pouvez

utiliser les tuiles de différents fournisseurs et personnaliser les cartes pour les adapter au mieux à votre application. Par exemple, la figure 3-13 décline un thème noir et bleu, mais vous pouvez facilement le modifier en blanc et rouge (figure 3-25).

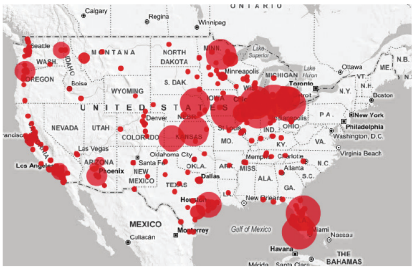


Figure 3-25 *Carte Modest Maps en rouge et blanc*

Comme l'application est sous licence BSD, vous pouvez pour ainsi dire faire ce que vous voulez, et ce sans frais. En revanche, vous devrez connaître les ficelles Flash et ActionScript, mais j'en propose les notions de base au chapitre 8.

Polymaps[1]

Polymaps constitue en quelque sorte la version JavaScript de Modest Maps. Elle a été développée et continue d'être maintenue par quelques-unes des mêmes personnes, et propose les mêmes fonctionnalités – et plus encore. Si Modest Maps ne couvre que les bases de la cartographie, Polymaps contient quelques fonctions intégrées, telles que les cartes choroplèthes (figure 3-26) et les bulles.

Comme il s'agit d'une application JavaScript, elle est plus légère (car nécessitant moins de code) et fonctionne avec les navigateurs récents. Polymaps utilisant la technologie SVG (*Scalable Vector Graphics*) pour afficher les données, l'application n'est pas compatible avec les anciennes versions d'Internet Explorer, mais, en

1. NDT. La compagnie qui s'occupait de Polymats ayant été rachetée, son développement a été arrêté. On retrouve les mêmes fonctions dans D3, et aussi dans d'autres bibliothèques plus spécialisées, comme leaflet.

principe, la plupart des utilisateurs sont à jour. À titre de référence, seulement 5 % environ des visiteurs de FlowingData se servent d'un navigateur trop ancien – je pense que ce pourcentage approchera bientôt de zéro.

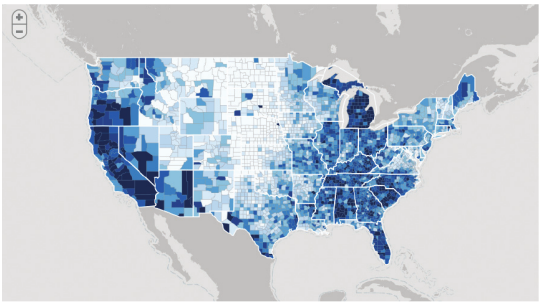


Figure 3-26 *Carte choroplèthe illustrant le chômage et implémentée dans Polymaps*

Le grand avantage, à mes yeux, d'une bibliothèque de cartographie en JavaScript est que tout le code s'exécute en mode natif dans le navigateur. Aucune compilation ou exportation Flash n'est nécessaire, ce qui facilite l'exécution et permet de procéder ultérieurement aux mises à jour.

Ressources Polymaps utiles

Polymaps (*http://polymaps.org/*)

R

R ne propose pas les fonctionnalités de cartographie dans la distribution de base, mais elles peuvent être ajoutées au moyen de packages. La figure 3-27 correspond à une carte créée dans R. Les annotations furent ajoutées par la suite dans Illustrator.

Les cartes créées dans R ont des capacités limitées et la documentation n'est pas exceptionnelle. Aussi, j'utilise R pour la cartographie si j'ai quelque chose de simple ; dans le cas contraire, je privilégie les outils mentionnés précédemment.

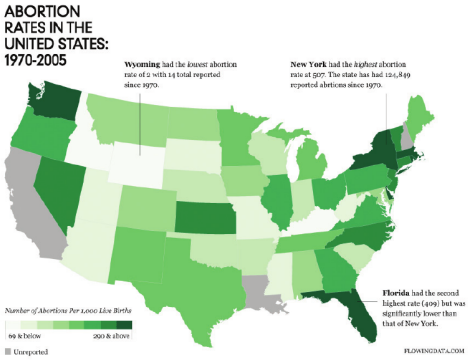


Figure 3-27 *Carte des États-Unis créée avec R*

Ressources utiles de cartographie avec R

- Analyse des données spatiales (*http://cran.r-project.org/web/views/Spatial.html*) – Liste complète des packages en R consacrés à l'analyse spatiale
- Guide pratique de la cartographie géostatistique (*http://spatial-analyst.net/book/download*) – Manuel gratuit et disponible en téléchargement sur l'utilisation de R et d'autres outils pour les données spatiales

Solutions en ligne

Quelques solutions de cartographie en ligne facilitent la visualisation de vos données géographiques. Dans l'ensemble, elles s'appuient sur les types de cartes le plus fréquemment utilisés et constituent une sorte de version ArcGIS simplifiée. Deux solutions gratuites parmi d'autres : Many Eyes et GeoCommons. La première, déjà évoquée, ne possède que les fonctionnalités de base pour les données réparties par pays ou par État dans le cas des États-Unis. GeoCommons

offre des fonctions et une interaction plus élaborées. L'application gère aussi les formats des fichiers géospatiaux, tels que shapefiles et KML.

Parmi les solutions payantes, Indiemapper et SpatialKey sont les plus précieuses. SpatialKey est plus orientée vers le secteur professionnel et la prise de décision, tandis qu'Indiemapper s'adresse surtout aux cartographes et aux concepteurs. La figure 3-28 illustre une carte que j'ai réalisée en quelques minutes à peine dans Indiemapper.

Compromis

Les logiciels de cartographie existent dans toutes les formes et tailles adaptées pour répondre à des besoins très différents. Il serait appréciable de pouvoir apprendre une seule application et d'être capable de créer tout type de carte imaginable. Malheureusement, il n'en est pas ainsi.

Par exemple, ArcGIS comporte de multiples fonctions, mais il ne serait pas plus judicieux de l'acquérir que de l'apprendre si vous ne voulez créer que des cartes simples. D'un autre côté, R, qui possède les fonctionnalités de cartographie élémentaires et qui est gratuit, peut se révéler trop simple pour ce que vous souhaitez. Si les cartes en ligne et interactives constituent votre objectif, vous pouvez choisir des applications open source comme Modest Maps ou Polymaps, mais elles nécessitent des compétences en programmation plus avancées. Vous en apprendrez plus sur l'utilisation de ce qui est disponible au chapitre 8.

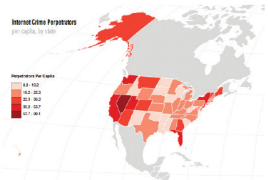


Figure 3-28 *Carte choroplèthe créée dans Indiemapper*

Enquête sur les options

Ceci n'est pas une liste exhaustive de ce que vous pouvez utiliser pour visualiser les données, mais elle est suffisante pour démarrer. Il y a beaucoup de points à prendre en compte ici. Les outils que vous finirez par utiliser dépendent grandement de ce que vous souhaiterez accomplir, et il existe maintes façons

d'exécuter une simple tâche, même au sein d'un seul logiciel. Vous voulez créer des graphiques de données statiques ? Essayez peut-être R ou Illustrator. Vous souhaitez développer un outil interactif pour une application web ? Essayez JavaScript ou Flash.

Sur FlowingData, j'ai procédé à un sondage où je demandais aux personnes ce qu'elles utilisaient principalement pour analyser et visualiser les données. Un peu plus de 1 000 personnes ont répondu. Les résultats sont illustrés à la figure 3-29.

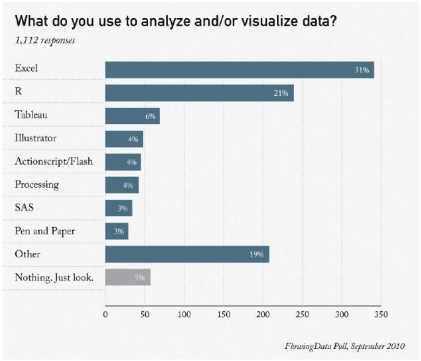


Figure 3-29 *Outils utilisés par les lecteurs de FlowingData pour analyser et visualiser les données*

Certains logiciels mènent la danse. Excel arrive en première position et R en deuxième. Mais derrière, les autres applications se tiennent au coude à coude. Il est à noter que plus de 200 personnes ont choisi la catégorie Autre. Dans leurs commentaires, de nombreuses personnes ont déclaré qu'elles se servaient d'une combinaison d'outils pour couvrir différents besoins, ce qui constitue habituellement la meilleure voie pour le long terme.

Combinaison des outils

Nombre d'utilisateurs aiment s'en tenir à un seul programme, solution simple et confortable. Ils n'ont pas à apprendre de nouvelles applications. Si le programme leur donne satisfaction, il est évident qu'ils doivent le conserver. Cependant, arrive un moment où, lorsque vous avez travaillé longtemps avec les données, vous atteignez les limites du logiciel. Vous savez ce que vous voulez faire avec vos données ou comment les visualiser, mais le logiciel ne vous le permet pas ou rend le processus bien plus complexe qu'il ne devrait l'être.

Vous pouvez accepter cet état de fait… ou choisir d'utiliser un autre logiciel. Certes, il vous faudra du temps pour l'apprendre, mais il vous aidera à donner forme à votre vision. L'apprentissage de différents outils vous garantit de ne pas vous retrouver bloqué par un ensemble de données ; en outre, si vous êtes polyvalent, l'exécution de différentes tâches de visualisation vous permettra d'obtenir des résultats effectifs.

Pour résumer

Aucun de ces outils ne constitue la panacée, bien sûr. En dernier ressort, les analyses et la conception des données sont toujours de votre fait. Les outils ne sont que ce qu'ils sont, à savoir des outils. De même que le fait d'avoir un marteau ne signifie pas que vous allez construire une maison. Idem, vous pouvez avoir un excellent logiciel et un super ordinateur, mais si vous ne savez pas comment utiliser ces outils, ils pourraient tout aussi bien ne pas exister. Vous décidez des questions à poser, des données à utiliser et des facettes à mettre en évidence ; et avec la pratique, vous gagnez en aisance.

Autant dire que vous êtes chanceux ! C'est à cela qu'est consacré le reste du livre. Les chapitres suivants traitent de concepts importants de la création de données et vous apprennent à passer de la théorie à la pratique, en utilisant une combinaison des outils que nous venons de présenter. Vous apprendrez ce que vous devez rechercher dans vos données et à le visualiser.

Chapitre 4

Visualisation des modèles temporels

Les données de séries temporelles sont omniprésentes ou presque. L'opinion publique change, les populations se déplacent et les entreprises se développent. Les données de séries temporelles permettent de mesurer l'ampleur de ces changements. Le présent chapitre s'intéresse aux données discrètes et continues, car le type de données graphiques que vous utiliserez dépendra du type des données elles-mêmes. Vous devrez également manipuler R et Illustrator, deux programmes dont l'association est excellente.

Que chercher au fil du temps ?

Vous regardez l'heure tous les jours. Elle est indiquée sur votre ordinateur, votre montre, votre téléphone… Même sans horloge, vous ressentez le temps, que vous vous réveilliez ou vous endormiez, que le soleil se lève ou se couche. Aussi, est-il tout à fait naturel que vous ayez des données au fil du temps. Grâce à elles, vous pouvez évaluer le degré de changement.

Les tendances constituent les informations les plus courantes que vous puissiez rechercher dans les séries temporelles. Illustrent-elles une augmentation ou une diminution ? Existe-t-il des cycles saisonniers ? Pour découvrir ces modèles et obtenir une vision d'ensemble, vous devez regarder au-delà des points de données individuels. Il est facile d'extraire une valeur d'un point dans le temps et de l'appeler « jour », mais si vous examinez ce qui la précède et la suit, vous comprendrez mieux ce que cette seule valeur signifie. Mieux vous connaissez vos données, meilleure sera l'histoire que vous pourrez raconter.

Par exemple, l'administration Obama a diffusé un graphique relatif à la première année de la présidence (figure 4-1). Il montre les pertes d'emploi durant la dernière année de l'administration Bush et la première année de la présidence Obama.

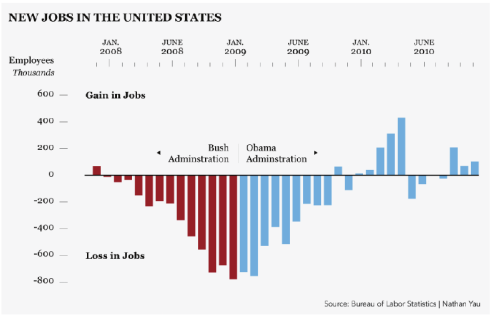


Figure 4-1 *Modification dans les chiffres du chômage depuis l'arrivée de Barack Obama à la présidence*

Il semble que la nouvelle administration ait eu un effet positif important sur le chômage, mais qu'en est-il si nous élargissons le cadre temporel, comme illustré à la figure 4-2 ? En résulte-t-il une différence d'interprétation ?

Même si vous désirez toujours bénéficier d'une vue d'ensemble, il est aussi utile de regarder les données de façon détaillée. Y a-t-il des cas isolés ? Y a-t-il des périodes qui semblent hors de propos ? Y a-t-il des pointes et des creux ? Si tel est le cas, que s'est-il passé pendant cette période ? Souvent, ce sera sur ces irrégularités que vous souhaiterez concentrer votre attention. Parfois, les cas aberrants se révèleront n'être qu'une erreur de saisie des données. Observer la vue d'ensemble, ou le contexte, vous aidera à déterminer la nature des choses.

Points discrets dans le temps

Les données temporelles peuvent se classer comme discrètes ou comme continues. Le fait de savoir à quelle catégorie vos données appartiennent vous aidera à déterminer comment les visualiser. Dans le cas des données discrètes, les valeurs proviennent de points ou de blocs de temps spécifiques, et il existe un nombre fini de valeurs possibles. Par exemple, le pourcentage de personnes qui réussissent un test chaque année est une valeur discrète. Les personnes réussissent le test, point à la ligne. Leurs résultats ne changent pas après coup et le test se déroule un jour donné. La température, quant à elle, est une donnée continue. Elle peut être mesurée à n'importe quel moment de la journée, pendant un intervalle, et elle change en permanence.

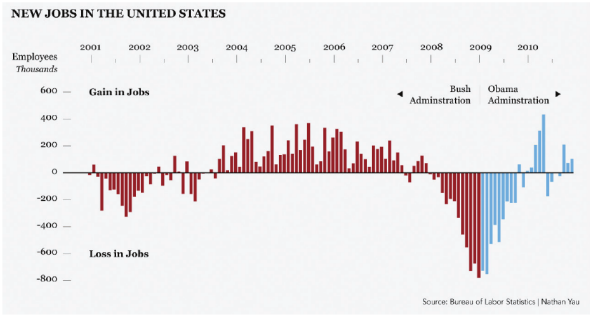


Figure 4-2 *Évolution du chômage aux États-Unis entre 2001 et 2010*

Dans cette section, vous vous intéresserez aux types de graphiques qui aident à visualiser les données temporelles discrètes. Des exemples concrets de création de ces graphiques dans R et Illustrator seront proposés. Une fois que j'en aurai terminé avec la présentation, vous pourrez appliquer les mêmes modèles de conception à l'ensemble du chapitre. Il s'agit là d'une partie importante. Même si les exemples concernent des graphiques spécifiques, vous pouvez appliquer les mêmes principes à toutes sortes de visualisation.

Barres

Le graphique en barres est l'un des plus répandus. Il est plus que probable que vous en ayez vu des centaines ou des milliers, et vous en avez sans doute déjà créés. Le graphique en barres peut être utilisé pour différents types de données, mais pour l'heure intéressons-nous à son emploi avec les données temporelles.

La figure 4-3 illustre un cadre de base. L'axe temporel (l'axe horizontal, axe des *x*) accueille les points dans le temps classés chronologiquement. Dans le cas présent, ils correspondent aux mois, de janvier à juin 2011, mais ce pourrait très bien être les années, les jours ou toute autre unité temporelle. La largeur des barres et leur espacement ne correspondent à aucune valeur.

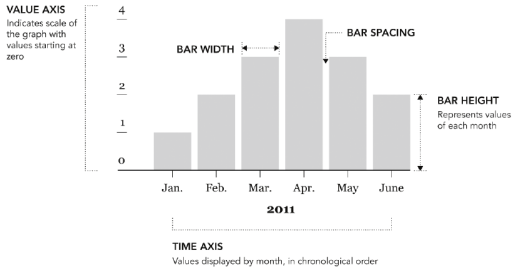


Figure 4-3 *Graphique en barres*

L'axe des valeurs (l'axe vertical, axe des *y*) indique l'échelle du graphique. La figure 4-3 illustre une échelle linéaire où les unités sont espacées de façon égale sur l'ensemble de l'axe. La hauteur des barres est en rapport avec l'axe des valeurs. La première barre, par exemple, équivaut à une unité, tandis que la barre la plus haute atteint quatre unités.

Ceci est important. L'indice visuel de la valeur est la hauteur des barres. Plus la valeur est faible, plus la barre est basse. Plus la valeur est élevée, plus la barre

est haute. Vous pouvez ainsi constater que la barre du mois d'avril est deux fois plus haute que celle du mois de février.

De nombreux programmes, par défaut, définissent la valeur la plus basse de l'axe des valeurs avec le minimum du jeu de données, comme illustré à la figure 4-4. Dans ce cas, le minimum est 1. Cependant, si vous deviez démarrer l'axe des valeurs à 1, la hauteur de la barre de février ne serait plus égale à la moitié de la barre de février. Elle paraîtrait être égale au tiers de celle d'avril. La barre de janvier elle-même serait non existante. En résumé, démarrez toujours l'axe des valeurs à zéro faute de quoi le graphique en barres pourrait afficher des relations incorrectes.

Figure 4-4 *Graphique en barres avec axe des y commençant à une valeur différente de zéro*

Démarrez toujours l'axe des valeurs de votre graphique en barres à la valeur zéro lorsque vous manipulez des valeurs positives. Toute autre valeur rend difficile la comparaison visuelle de la hauteur des barres.

Créer un graphique en barres

Il est temps pour vous de créer votre premier graphique. Vous allez utiliser des données réelles qui concernent un aspect primordial de l'histoire humaine… Elles correspondent aux résultats des trois dernières décennies du Nathan's Hot Dog Eating Contest, le concours du plus gros mangeur de hot-dogs. Super !

La figure 4-5 montre le graphique final que vous devez obtenir. Vous allez procéder en deux étapes : la création d'un simple graphique en barres dans R, puis son amélioration dans Illustrator. Au cas où vous l'ignoreriez, le Nathan's Hot Dog Eating Contest se déroule tous les ans, le 4 juillet, le jour même de la fête de l'indépendance des États-Unis. L'événement est devenu si populaire qu'il est même retransmis à la télévision sur la chaîne ESPN.

Au cours des années 1990, les vainqueurs mangèrent de 10 à 20 HDB (*Hot Dogs and Buns*) en 15 minutes environ. Cependant, en 2001, Takeru Kobayashi, mangeur professionnel japonais, écrasa la compétition en consommant 50 HDB, soit plus du double que le précédent record. Et c'est ici que l'histoire commence.

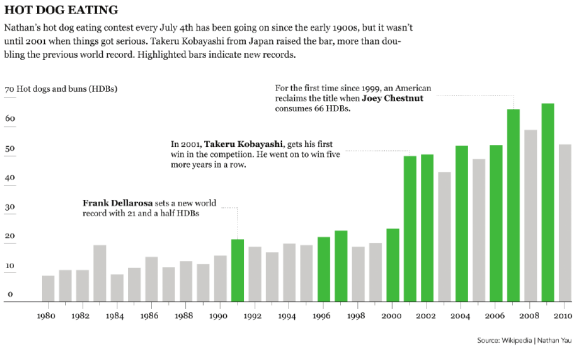


Figure 4-5 *Graphique en barres illustrant les résultats du Nathan's Hot Dog Eating Contest*

Wikipédia propose les résultats de la compétition depuis 1916, mais comme l'événement n'a eu lieu régulièrement qu'à partir de 1980, nous commencerons à cette date. Les données figurent dans un tableau HTML et incluent l'année, le nom, le nombre de HDB mangés et le pays d'origine du vainqueur. J'ai compilé les données dans un fichier CSV que vous pouvez télécharger à l'adresse suivante : *http://datasets.flowingdata.com/hot-dog-contest-winners.csv*. Les cinq premières lignes sont les suivantes :

```
"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
```

Pour charger les données dans R, utilisez la commande `read.csv()`. Vous pouvez charger le fichier localement à partir de votre propre ordinateur ou utiliser une URL. Dans ce cas, saisissez la ligne de code suivante :

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

Téléchargez les données au format CSV à partir de l'adresse suivante : *http://datasets.flowingdata.com/hot-dog-contestwinners.csv*. Reportez-vous à la page de Wikipédia intitulée « Nathan's Hot Dog Eating Contest » (*http://en.wikipedia.org/wiki/Nathan%27s_Hot_Dog_Eating_Contest*) pour obtenir les données précompilées et l'historique du concours.

Si vous souhaitez charger les données localement à partir de votre propre ordinateur, utilisez le menu principal de R pour définir votre répertoire de travail comme étant le même répertoire que celui du fichier de données. Vous pouvez aussi vous servir de la fonction `setwd()`.

Si vous débutez en programmation, cela vous paraîtra énigmatique. Aussi, décomposons les tâches afin que vous compreniez bien. Une ligne de code R suffit. Vous chargez les données avec la commande `read.csv()` qui comporte trois arguments.

- Le premier correspond à l'emplacement des données, ici une URL.
- Le deuxième argument, `sep`, spécifie le caractère séparateur des colonnes dans le fichier de données, en l'occurrence la virgule. Si le caractère avait été la tabulation, vous auriez précisé `\t` au lieu d'une virgule.
- Le dernier argument, `header`, indique à R que le fichier de données a un entête, qui contient le nom de chaque colonne. La première colonne est l'année, la deuxième le nom du vainqueur, la troisième le nombre de HDB mangés et la quatrième le pays de résidence. J'ai aussi ajouté un nouveau champ, comme vous l'aurez peut-être remarqué : nouvel enregistrement. Si le record mondial a été battu une année, la valeur est 1. Sinon, la valeur est 0. Vous en tirerez parti bientôt.

Les données sont maintenant chargées dans R et disponibles via la variable `hotdogs`. Techniquement, les données sont stockées comme un tableau de données, ce qui n'est pas essentiel mais mérite d'être noté. Voici à quoi ressemble le début de la structure si vous tapez `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
```

Les espaces dans les noms des colonnes ont été remplacés par des points. `Dogs eaten` est devenu `Dogs.eaten`. De même pour `New record`. Pour accéder à une colonne de données spécifique, vous utilisez le nom du tableau de données,

suivi du signe `$` et du nom de la colonne. Par exemple, pour accéder à `Dogs.eaten`, vous entrez :

```
hotdogs$Dogs.eaten
```

Maintenant que vous avez les données dans R, vous pouvez passer directement au graphique avec la commande `barplot()`.

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

R sait alors qu'il doit tracer le graphique de la colonne `Dogs.eaten`. Vous devez obtenir le graphique illustré à la figure 4-6.

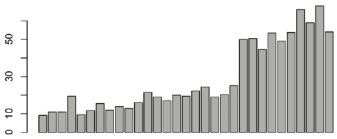


Figure 4-6 *Graphique par défaut du nombre de HDB mangés, tracé avec la commande barplot() dans R*

Pas mal, mais le résultat peut être meilleur. Utilisez l'argument `names.arg` dans la commande `barplot()` pour spécifier les noms de chaque barre. Dans ce cas, il s'agit de l'année de chaque compétition.

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

Vous obtenez alors le graphique de la figure 4-7 dans lequel les années apparaissent (en bas).

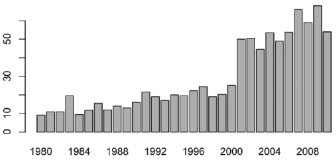


Figure 4-7 *Graphique en barres affichant les années*

Vous pouvez appliquer un certain nombre d'autres arguments. Il est ainsi possible d'ajouter les libellés des axes, modifier les bordures ou encore changer les couleurs (figure 4-8).

```
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year, col="red",
      border=NA, xlab="Année", ylab="Hot-dogs et sandwichs avalés")
```

L'argument `col` peut être le nom de la couleur (voir la documentation R) ou un nombre hexadécimal tel que `#821122`. Ici, vous spécifiez l'absence de bordure avec `NA`, constante logique qui signifie l'absence de valeur. L'axe *x* est défini par `Année` et l'axe *y* par `Hot-dogs et sandwichs avalés`.

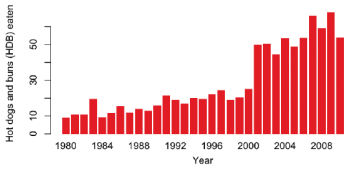


Figure 4-8 *Graphique en barres de couleur affichant les intitulés des axes*

Pourquoi vous limitez à une seule couleur ? Vous pouvez spécifier plusieurs couleurs à la commande `barplot()` pour colorer chaque barre à votre guise. Imaginons, par exemple, que vous vouliez mettre en évidence les années où les États-Unis ont remporté la compétition. Vous pouvez les colorer en rouge foncé (`#821122`) afin de les distinguer des autres années, qui apparaissent quant à elles en gris clair (figure 4-9).

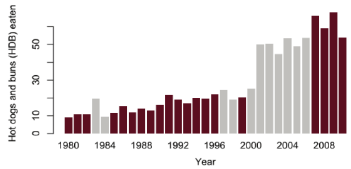


Figure 4-9 *Graphique en barres avec plusieurs couleurs*

Pour cela, vous devez construire une liste, ou vecteur dans R, de couleurs. Passez en revue chaque année et décidez de quelle couleur doit être la barre correspondante. Si les États-Unis ont emporté le concours, spécifiez la couleur rouge, sinon la couleur grise. Le code correspondant est le suivant :

```
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")
    }
}
```

La première ligne initie un vecteur vide, nommé `fill_colors`. En R, vous créez des vecteurs avec `c()`.

La ligne suivante démarre une boucle `for`. Vous pouvez demander à R de lire en boucle un index, baptisé `i`, de 1 jusqu'au nombre de lignes du tableau de données `hotdogs`. Plus précisément, vous pouvez extraire une simple colonne, `Country`, de la structure de données `hotdogs` et calculer la longueur. Si vous utilisiez `length()` avec seulement `hotdogs`, vous obtiendriez le nombre de colonnes, à savoir `5` dans le cas présent, alors que vous voulez le nombre de lignes, en l'occurrence `31`. Comme il y a une ligne par année entre 1980 et 2010, la boucle exécute le code à l'intérieur des crochets 31 fois, et à chaque boucle, l'index `i` augmente d'une unité.

Ainsi, dans la première itération, où `i` vaut 1, vérifiez si le pays de la première ligne (à savoir, le vainqueur de 1980) est bien les États-Unis. Si tel est le cas, ajoutez la couleur `#821122`, qui est une couleur rougeâtre au format hexadécimal, à `fill_colors`. Sinon, ajoutez `#cccccc`, qui correspond à gris clair.

En 1980, comme les vainqueurs venaient des États-Unis, optez pour la première solution. La boucle parcourt encore 30 fois les années restantes. Saisissez `fill_colors` dans la console R pour afficher les résultats. Il s'agit d'un vecteur de couleurs, exactement comme vous le souhaitiez.

Transmettez le vecteur `fill_colors` dans l'argument `col` de `barplot()`, comme suit :

La plupart des langages utilisent les tableaux ou les vecteurs de base 0, où le premier élément est la référence avec un index égal à 0. R, cependant, emploie les vecteurs de base 1.

```
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year, col=fill_colors,
    border=NA, xlab="Année", ylab="Hot-dogs et sandwichs avalés")
```

Le code est le même que précédemment, si ce n'est que vous utilisez `fill_colors` au lieu de `red` dans l'argument `col`.

Le graphique en barres final de la figure 4-5 met en évidence les années où un record a été battu – et non quand les États-Unis ont remporté le concours. Le processus et la logique sont les mêmes. Vous devez simplement modifier

certaines conditions. La colonne `New.record` de votre tableau de données indique les nouveaux enregistrements : si la valeur est 1, vous colorez la barre en rouge foncé, sinon en gris. Le code en R est le suivant :

```
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="Année", ylab="Hot-dogs et sandwichs avalés")
```

L'exemple est identique à celui des États-Unis, si ce n'est que les instructions `if` sont différentes. Vous devez obtenir un graphique tel que celui illustré à la figure 4-10.

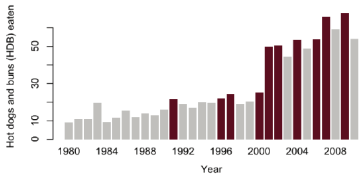


Figure 4-10 *Graphique en barres de couleur, obtenu avec des conditions différentes par rapport au graphique précédent*

À ce stade, vous pouvez essayer d'autres options de `barplot()` comme l'espacement ou l'ajout d'un titre.

```
barplot(hotdogs$Dogs.eaten, names.arg=hotdogs$Year, col=fill_colors,
    border=NA, space=0.3, xlab="Année", ylab="Hot-dogs et sandwichs
    ➥avalés"
```

Le résultat obtenu est illustré à la figure 4-11. Notez que l'espacement entre les barres est plus large que précédemment et qu'un titre a été ajouté au-dessus du graphique.

Faites attention lors du choix de l'espacement des barres. Si la valeur de l'espacement est proche de la largeur des barres, il risque de se produire un effet de vibration visuelle, comme si les rôles de la barre et de l'espacement avaient permuté.

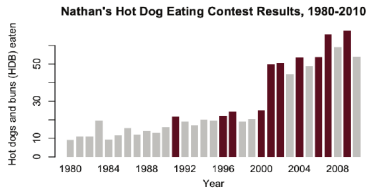


Figure 4-11 *Graphique en barres avec espacement personnalisé et titre principal*

Voilà ! Vous avez eu votre premier aperçu de R.

Dans le menu Fichier, utilisez l'option qui permet d'enregistrer le graphique comme fichier PDF. Vous en aurez besoin sous peu.

Pour afficher la documentation sur une fonction quelconque de R, saisissez simplement un point d'interrogation suivi du nom de la fonction. Par exemple, pour en savoir plus sur la commande `barplot()` dans R, tapez `?barplot`. Vous obtenez alors une description de la fonction, accompagnée des arguments disponibles. Il y a aussi généralement des exemples, qui peuvent être extrêmement précieux.

Améliorer le graphique dans Illustrator

Vous disposez maintenant d'un graphique en barres. Il est assez agréable à regarder et, si vous ne l'utilisez qu'à des fins d'analyse, il convient parfaitement. Cependant, si vous voulez le transformer en graphique autonome, quelques opérations supplémentaires le rendront plus lisible.

Maintenant, regardons-le du point de vue de l'histoire. Faites comme si la figure 4-11 était seule et que vous étiez un lecteur qui tombait dessus par hasard. Que pouvez-vous recueillir de votre graphique de base ? Vous savez qu'il illustre le nombre de hot-dogs et de sandwichs mangés chaque année. S'agit-il des habitudes de consommation d'une personne ? Cela fait beaucoup de hot-dogs pour une seule personne… S'agit-il de nourriture pour animaux ? De restes laissés aux oiseaux ? Du nombre moyen de hot-dogs mangés par personne et par an ? Pourquoi les barres sont-elles colorées ?

Comme c'est vous qui avez créé le graphique, vous connaissez le contexte sous-jacent aux nombres, mais pas vos lecteurs : vous devez donc leur expliquer de quoi il s'agit. Un bon design des données peut aider vos lecteurs à comprendre l'histoire plus clairement. Illustrator, qui permet de modifier les éléments manuellement, peut vous y aider. Vous pouvez changer les polices, les axes, les couleurs, ajouter des notes ou encore effectuer toutes sortes de modifications.

Dans ce livre, les retouches effectuées dans Illustrator resteront simples, mais au fur et à mesure que vous travaillerez sur un plus grand nombre d'exemples et commencerez à créer vos propres graphiques, vous verrez à quel point ces petites modifications peuvent être d'une grande aide pour la clarté et la concision de vos graphiques.

Commençons par le début. Lancez Illustrator et ouvrez le fichier PDF correspondant à votre graphique en barres. Une nouvelle fenêtre apparaît affichant votre graphique et un certain nombre de boîtes de dialogue permettant de modifier, entre autres, les couleurs et les polices. La boîte de dialogue la plus importante est la boîte Outils, illustrée à la figure 4-12. Vous l'utiliserez très souvent. Si elle n'apparaît pas, sélectionnez le menu Fenêtre et cliquez sur Outils pour l'activer.

Figure 4-12 *Boîte de dialogue Outils dans Illustrator*

Mettez-vous à la place du lecteur lorsque vous concevez les graphiques de données. Quelles parties de ces données nécessitent une explication ?

La flèche noire correspond à l'outil Sélection. Cliquez dessus, le pointeur de la souris se transforme alors en flèche noire (si ce n'était déjà le cas). Cliquez sur la bordure du graphique et déplacez-la. Les bordures sont alors mises en évidence, comme illustré à la figure 4-13. Dans Illustrator, cet effet est appelé masque de détourage. Il peut être utile dans différentes situations. Pour le moment, vous n'en avez pas besoin aussi appuyez sur la touche Suppr du clavier pour le faire disparaître. Si cette action fait disparaître tout le graphique, annulez la modification. Cliquez ensuite sur l'outil Sélection directe (flèche blanche) pour sélectionner uniquement le masque de détourage et non tout le graphique.

Si vous n'avez pas Illustrator, vous pouvez utiliser Inkscape, logiciel gratuit et open source. Les icônes et fonctions d'Inkscape ne sont pas les mêmes que celles d'Illustrator mais vous retrouverez néanmoins de nombreux éléments similaires.

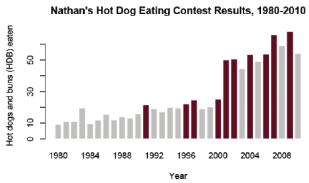


Figure 4-13 *Suppression du masque de détourage du fichier PDF ouvert dans Illustrator*

Essayez maintenant de modifier les polices, ce qui ne présente aucune difficulté. Choisissez à nouveau l'outil Sélection et sélectionnez le texte que vous voulez modifier. Utilisez les menus déroulants de la boîte de dialogue Police pour remplacer par la police de votre choix. Vous pouvez aussi modifier les polices via le menu Texte. Sur la figure 4-15, la police d'origine a été est remplacée par la police Georgia Regular.

Nous allons ensuite intervenir sur les étiquettes numériques de l'axe des valeurs. Les nombres sont listés verticalement, mais ils devraient être positionnés horizontalement pour une meilleure lisibilité. Cliquez sur les nombres afin de les sélectionner. Comme vous pouvez le constatez, cette action active également d'autres éléments. Ceci s'explique par le fait que les nombres sélectionnés et les éléments du graphique sont regroupés. Vous devez les dégrouper afin de pouvoir faire pivoter chacun des nombres. Dans le menu Objet, cliquez sur Annuler

regroupement. Annulez la sélection des étiquettes numériques, puis sélectionnez-les à nouveau. À présent, seuls les nombres sont sélectionnés, ce que vous vouliez. Une autre solution consiste à utiliser l'outil Sélection directe.

Figure 4-14 *Boîte de dialogue Police dans Illustrator*

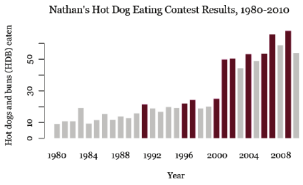


Figure 4-15 *Graphique dont les polices ont été remplacées par la police Georgia Regular*

Revenez dans le menu Objet et sélectionnez Transformation>Transformation répartie. Comme illustré à la figure 4-16, modifiez la valeur de l'angle de rotation afin qu'elle soit égale à -90 degrés. Cliquez sur OK. Les étiquettes sont maintenant du bon côté.

Profitez-en pour décaler les étiquettes (pas les graduations) vers le haut et vers la droite afin qu'elles soient au-dessus des graduations et non à leur gauche. Déplacez les éléments à l'aide des touches de direction du clavier ou en les glissant-déplaçant à la souris. Vous pouvez aussi spécifier directement les unités comme « Hot dogs and buns eaten » (« Hot-dogs et sandwichs avalés »). Une fois encore, la lisibilité est améliorée quand les yeux du lecteur se déplacent

de la gauche vers la droite. Le résultat obtenu doit être semblable au graphique de la figure 4-17.

Votre graphique commence à ressembler un peu plus au graphique final de la figure 4-5. Toutefois, il manque encore quelques éléments. L'axe horizontal ne comporte aucune graduation, il n'y a aucune annotation et il serait bien que vous incorporiez un peu de vert, l'une des couleurs du logo du concours du plus gros mangeur de hot-dogs.

Les graphiques de données sont censés apporter un éclairage sur vos données. Aussi, supprimez les éléments qui ne concourent pas à cet objectif.

Vous pouvez aussi simplifier le graphique en supprimant la ligne verticale de l'axe des valeurs car elle n'aide pas à communiquer les données plus clairement. Notez que dans le graphique final, il y a uniquement des graduations. Si vous cliquez sur la ligne verticale avec l'outil Sélection, les libellés sont aussi sélectionnés. La raison en est qu'ils font tous partie d'un groupe. Pour sélectionner uniquement la ligne, utilisez l'outil Sélection directe et appuyez sur la touche Suppr pour la supprimer.

Figure 4-16 *Boîte de dialogue Transformation répartie*

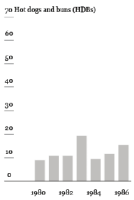


Figure 4-17 *Graphique en barres avec axe des valeurs simplifié*

Il existe plusieurs façons de créer des graduations. L'une d'entre elles consiste à utiliser l'outil Plume, qui permet de tracer facilement des lignes droites. Sélectionnez-le dans la boîte de dialogue Outils, puis spécifiez le style du trait via la boîte de dialogue Trait. Choisissez une épaisseur de 0,3 pt et assurez-vous que la case à cocher En pointillé n'est pas activée.

Pour tracer une ligne, cliquez au point de départ de la ligne, puis sur son point d'arrivée. Si, lors du second clic, vous appuyez sur la touche Maj, la ligne est automatiquement droite. Vous ne devez avoir maintenant qu'une seule graduation. Il en faut 30 autres, car vous avez besoin d'une graduation pour chaque année.

Vous pouvez toutes les tracer à la main, mais il existe une solution bien meilleure qui consiste à utiliser la touche Option sur Mac ou Alt sur PC. Cliquez sur la seule graduation avec l'outil Sélection et faites-la glisser à l'endroit où vous souhaitez insérer la graduation suivante, tout en maintenant enfoncée la touche Option ou Alt. Une copie de la première graduation est créée. Appuyez maintenant sur les touches Cmd + D sur Mac ou Ctrl + D sur PC. Une nouvelle graduation est créée, espacée de la seconde comme celle-ci l'était de la première. Appuyez sur les touches Cmd/Ctrl + D autant de fois que nécessaire pour obtenir toutes les graduations souhaitées.

Enfin, disposez toutes les graduations correctement. Déplacez la dernière de manière à ce qu'elle soit centrée par rapport à la dernière barre. La première graduation doit déjà être centrée par rapport à la première barre. Sélectionnez ensuite toutes les graduations, et cliquez sur l'icône Distribution horizontale de la boîte de dialogue Alignement (figure 4-18).

Les graduations sont alors réparties afin d'être espacées de façon égale entre les première et dernière graduations. Le cas échéant, vous pouvez sélectionner une graduation sur deux avec l'outil Sélection et la redimensionner verticalement pour la réduire. Il apparaît ainsi clairement que les graduations les plus longues concernent les libellés des années.

Figure 4-18 *Boîte de dialogue Alignement dans Illustrator*

Pour modifier la couleur de remplissage de rouge en vert, activez à nouveau l'outil Sélection directe et sélectionnez chaque zone rouge. Ici, la sélection est rapide car il y a peu de barres rouges, mais qu'en serait-il s'il y en avait beaucoup ? Une solution plus efficace consiste à cliquer sur une seule barre rouge

et à choisir ensuite le menu Sélection>Identique>Couleur de fond. Cette commande, comme vous pouviez vous y attendre, sélectionne toutes les barres ayant un fond rouge. Il suffit maintenant de changer la couleur à votre convenance via la boîte de dialogue Couleur. Il est possible de modifier les couleurs de fond et de contour, nous nous contenterons ici de la couleur de fond (figure 4-19).

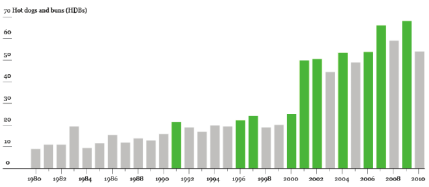


Figure 4-19 *Modification de la couleur des éléments graphiques*

À l'aide de l'outil Texte de la boîte de dialogue Outils, vous pouvez ajouter des zones de texte au graphique. Vous avez ainsi la possibilité d'expliquer aux lecteurs ce qu'ils ont sous les yeux et de clarifier les points éventuellement confus. Choisissez les polices qui vous semblent adaptées, en jouant sur la taille et le style pour différencier le libellé des éléments graphiques tels que les libellés d'axes.

Mentionnez toujours la source des données dans vos graphiques. Elle assure non seulement sa crédibilité, mais permet aussi de comprendre le contexte.

Dans le cas présent, mettez en valeur le premier record depuis 1980, la domination de Takeru Kobayashi et le règne actuel de Joey Chestnut. Ajoutez aussi un titre, ainsi qu'un préambule qui explique l'idée générale du graphique.

Dernier point très important : pensez à inclure la source des données. Si vous ne la mentionnez pas, il n'existe aucun moyen de savoir si votre graphique est exact. Une fois tous les éléments ajoutés, vous obtenez le graphique final, illustré à la figure 4-5.

Plus vous travaillerez sur les graphiques, plus la tâche sera aisée. Vous apprendrez comment l'écriture de code en R, ou autre langage, obéit à un certain modèle, et même si Illustrator propose une boîte à outils très étendue, vous venez de découvrir ceux qui se rapportent à la tâche en cours.

L'exemple suivant concerne d'autres types de graphiques de données temporelles et utilise plus abondamment R et Illustrator. Vous irez maintenant plus vite, puisque nous venons de voir certains aspects élémentaires des deux outils.

Empilement des barres

Comme illustré à la figure 4-20, la géométrie des graphiques en barres empilées est similaire à celle des graphiques en barres classiques. La différence est que les rectangles sont empilés l'un sur l'autre. Les graphiques en barres empilées sont à privilégier quand il existe des sous-catégories et que la somme de ces dernières est significative.

Comme les graphiques en barres, les graphiques en barres empilées ne concernent pas seulement les données temporelles et peuvent être utilisés pour d'autres données catégorielles. Les catégories du graphique de la figure 4-20 correspondent aux mois.

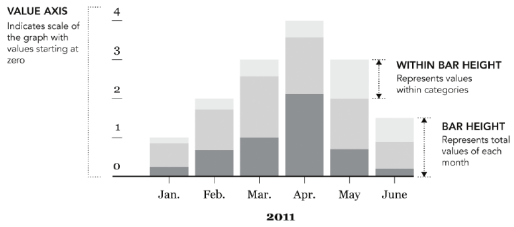


Figure 4-20 *Graphiques en barres empilées*

Créer un graphique en barres empilées

Ce type graphique est relativement courant et il existe de multiples façons de le créer. Nous vous indiquons ici comment procéder avec R. Le processus est semblable à celui qui vous a permis de créer précédemment un graphique en barres régulier.

1. Chargez les données dans R.
2. Assurez-vous que les données sont correctement mises en forme.
3. Utilisez une fonction R pour créer un tracé.

Vous procéderez ainsi à chaque fois que vous créerez un graphique de données avec R. Vous consacrerez parfois plus de temps à un point plutôt qu'à un autre. Le formatage des données pourra nécessiter plus de temps selon les cas, ou peut-être aurez-vous à écrire vos propres fonctions en R pour obtenir exactement

ce que vous souhaitez. Quoi qu'il en soit, vous passerez presque toujours par ces trois étapes et vous les retrouverez dans d'autres langages, comme nous le verrons dans les prochains chapitres.

Revenons à notre graphique en barres empilées et au Nathan's Hot Dog Contest. La figure 4-21 représente le graphique que vous souhaitez créer.

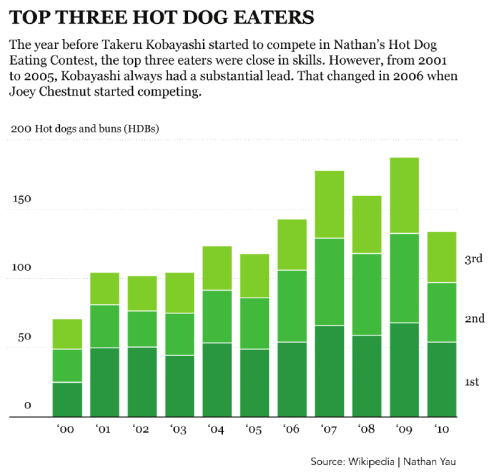


Figure 4-21 *Graphiques en barres empilées illustrant les trois plus gros mangeurs de hot-dogs de 2000 à 2010*

Au lieu de ne regarder que le nombre de hot-dogs et sandwichs avalés par les vainqueurs, intéressons-nous aux trois premiers de chaque année. Chaque pile représente une année et se compose de trois barres, une pour chacun des trois

premiers concurrents. Dans la mesure où Wikipédia ne fournit ces données de façon régulière que depuis 2000, commençons par là.

Chargez les données dans R. Vous les chargez directement depuis l'URL avec le code suivant :

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

Tapez `hot_dog_places` pour afficher les données. Chaque colonne affiche le résultat d'une année et chaque ligne correspond respectivement aux places 1, 2 et 3.

```
  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
```

Remarquez la présence de la lettre `X` devant tous les noms de colonne, ajouté par défaut lors du chargement des données. Cette lettre a été ajoutée par R car les noms d'en-tête étaient des nombres. Grâce à cet ajout, les noms d'en-tête sont traités comme des *chaînes de caractères* et non des nombres. Comme vous devez utiliser les noms d'en-tête pour les libellés du graphique en barres empilées, modifions-les à nouveau.

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

Placez des guillemets autour de chaque année pour spécifier qu'il s'agit d'une chaîne. Tapez à nouveau `hot_dog_places`, à présent les en-têtes correspondent bien aux années.

```
  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
```

Comme précédemment, ayez recours à la fonction `barplot()`, mais utilisez les données dans un format différent. Pour transmettre toutes les valeurs précédentes à `barplot()`, vous devez convertir `hot_dog_places` en une matrice. Pour l'heure, les valeurs se présentent sous la forme d'un tableau de données. Ce sont des structures différentes dans R, mais, pour l'instant, ces différences ne sont pas déterminantes. En revanche, vous devez savoir convertir un tableau de données en une matrice.

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

Vous avez stocké la matrice nouvellement créée comme `hot_dog_matrix`. Vous pouvez la passer à `barplot()`.

```
barplot(hot_dog_matrix, border=NA, space=0.25, ylim=c(0, 200),
    xlab="Année", ylab="Hot-dogs et sandwichs avalés",
    main="Résultats du concours du plus gros mangeur de hot-dogs,
    ➥1980-2010")
```

Vous n'avez spécifié aucun contour de barre et vous avez défini l'espacement à 0,25 de la largeur des barres et un axe des valeurs compris entre 0 et 200, ainsi que le titre du graphique et les libellés des axes. La figure 4-22 illustre le résultat obtenu.

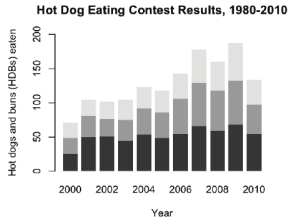


Figure 4-22 *Graphique en barres empilées créé en R*

Pas mal pour quelques lignes de code seulement, mais vous n'avez pas terminé pour autant. Vous pouvez maintenant affiner votre graphique. Enregistrez l'image au format PDF et ouvrez-la dans Illustrator. Utilisez les mêmes outils que dans l'exemple précédent. Vous pouvez ajouter du texte avec l'outil Texte, changer de polices, simplifier l'axe vertical, modifier les couleurs avec la possibilité de sélectionner les éléments ayant le même remplissage, et, bien sûr, inclure les sources des données (figure 4-23).

Ajoutez un texte d'introduction et modifiez le titre à votre guise. La figure 4-21 illustre le résultat final.

Le prochain chapitre traite d'un cousin du graphique en barres empilées, à savoir le graphique en zones empilées. La géométrie étant similaire, imaginez simplement que tous les empilements sont reliés de façon continue.

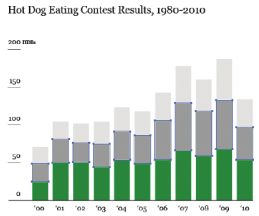


Figure 4-23 *Modifications dans Illustrator*

Points

Parfois, l'emploi de points à la place de barres est plus justifié. Ils prennent moins d'espace et assurent une meilleure impression de continuité d'un point à un autre. La figure 4-24 illustre la géométrie courante obtenue lorsque vous utilisez des points pour créer un graphique de données temporelles.

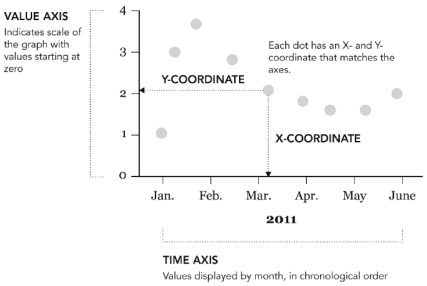


Figure 4-24 *Utilisation de points dans un graphique*

Ce type de graphique, communément appelé nuage de points, permet lui aussi de visualiser des données non temporelles. Il sert souvent à montrer la relation entre deux variables (voir chapitre 6). Dans le cas des données temporelles, le temps est représenté sur l'axe horizontal et les valeurs (ou les mesures) sur l'axe vertical.

Contrairement au graphique en barres, qui utilise la longueur comme indice visuel, les nuages de points recourent à la position. Vous pouvez penser à chaque point en termes de coordonnées X et Y, et le comparer à d'autres points en fonction de leur emplacement. Pour cette raison, l'axe des valeurs d'un nuage de points ne doit pas nécessairement commencer à zéro, même s'il s'agit là d'une pratique recommandée.

Créer un nuage de points

R simplifie la création d'un nuage de points grâce à la fonction `plot()`, mais vous pouvez choisir des variantes selon les données que vous examinez. La figure 4-25 illustre le nuage de points final.

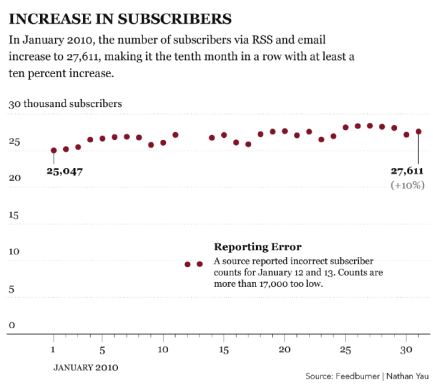


Figure 4-25 *Nuage de points créé dans R et mis en forme dans Illustrator*

Le nuage de points concerne le nombre d'abonnés à FlowingData en janvier 2010, rapporté par Feedburner, service assurant le suivi du nombre quotidien de lecteurs de FlowingData. Le 1er janvier 2010, il y avait 25 047 abonnés, et à la fin de ce même mois, 27 611. Toutefois, la donnée la plus intéressante n'apparaît pas, elle correspond à un événement survenu au milieu du mois. Ai-je réellement dit quelque chose qui a offensé les abonnés et conduit 17 000 d'entre eux à annuler leur abonnement ? Peu probable.

Le fait qu'il s'agisse de données ne les rend pas sûres pour autant. Elles peuvent contenir des coquilles, des erreurs d'information ou autres, qui engendrent une déformation de la réalité.

Vous souvenez-vous de la première chose à faire quand vous créez un graphique dans R ? Vous chargez les données. Utilisez `read.csv()` pour charger directement les données à partir d'une URL.

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

Pour voir les cinq premières lignes de données, entrez l'instruction suivante :

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

Et voici le résultat obtenu :

```
    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
```

Parmi les cinq colonnes, seule la colonne du nombre d'abonnés nous intéresse.

Vous pouvez aussi intégrer la date, mais comme les données se présentent déjà par ordre chronologique, la première colonne n'est pas réellement utile. Pour tracer le graphique, entrez l'instruction suivante afin d'obtenir le résultat de la figure 4-26.

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

La fonction `plot()` permet de créer plusieurs types de graphiques. Par défaut, elle crée un nuage de points. Ici, vous n'avez utilisé que la colonne du nombre d'abonnés. Lorsque vous fournissez un seul tableau de données à la fonction `plot()`, elle présume que le tableau contient des valeurs et génère automatiquement un index pour les coordonnées en *x*.

À présent, spécifiez explicitement le type de points souhaité et définissez pour l'axe vertical une plage allant de 0 à 30 000.

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

La figure 4-27 est similaire à la figure 4-26, mais avec un axe vertical plus large tel que vous l'avez spécifié avec l'argument `ylim`. Notez l'emploi de l'argument `type` pour dire à R d'utiliser des points. Si vous modifiez le type en `h`, R créera des lignes verticales plus denses.

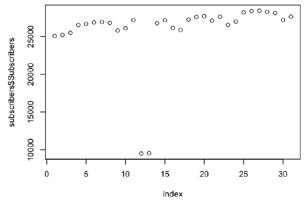


Figure 4-26 *Tracé par défaut dans R*

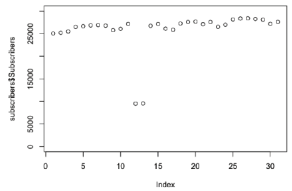


Figure 4-27 *Tracé dans R avec spécification des limites en* y

Vous pouvez aussi combiner les deux, comme illustré à la figure 4-28. Cependant, il vous faut aussi utiliser la méthode `points()`. Chaque fois que vous utilisez la fonction `plot()`, vous créez un graphique au lieu d'ajouter de nouveaux éléments à un graphique existant. Voici comment combiner les lignes verticales et les points :

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

Le tracé avec les lignes verticales est dessiné en premier, cette fois avec les libellés des axes. Les points sont ensuite ajoutés au tracé existant. L'argument `pch` spécifie la taille et l'argument `col`, déjà utilisé dans le graphique en barres, la couleur de remplissage.

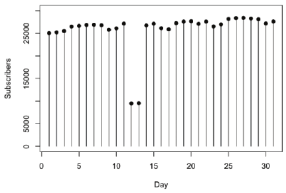


Figure 4-28 *Tracé avec lignes verticales denses*

Revenons à la figure 4-27. Enregistrez-la comme fichier PDF et ouvrez-la dans Illustrator afin de lui apporter quelques modifications.

Avec l'outil Sélection, sélectionnez les libellés et modifiez la police à votre guise. Dégroupez ensuite les libellés de façon à pouvoir modifier séparément les libellés de l'axe vertical. Utilisez le menu Transformation>Transformation répartie pour afficher les libellés horizontalement. Puis, avec l'outil Sélection directe, supprimez la barre verticale de l'axe – vous n'en avez pas besoin, elle ne fait qu'occuper de l'espace.

Enfin, sélectionnez les points eux-mêmes (simples cercles blancs) et utilisez les options de la boîte de dialogue Couleur pour choisir la couleur de remplissage et la couleur de trait. Il se peut que vous ayez à modifier le modèle de couleur de Niveaux de gris à CMJN (Cyan, Magenta, Jaune et Noir) pour obtenir un plus grand choix de couleurs. Pour cela, utilisez le menu Options de la boîte de dialogue Couleur (figure 4-29). Vous obtenez alors la figure 4-30, qui ressemble davantage au graphique final.

Essayons maintenant d'ajouter une grille afin de voir plus facilement à quelles valeurs correspondent les points à l'extrémité droite et comment ils se rapportent aux points précédents. Pour cela, sélectionnez tout d'abord les graduations sur l'axe des valeurs avec l'outil Sélection. Effectuez un glisser-déplacer afin de positionner les graduations tous le long du graphique. Modifiez éventuellement le style des traits à l'aide des options disponibles de la boîte de dialogue Trait. Utilisez les options de la figure 4-31 pour obtenir des lignes pointillées fines. La figure 4-32 illustre le résultat obtenu après les modifications.

Figure 4-29 *Menu Options de la boîte de dialogue Couleur*

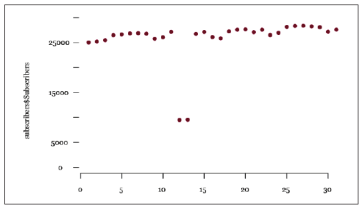


Figure 4-30 *Tracé des points après modification de l'axe vertical et de la couleur*

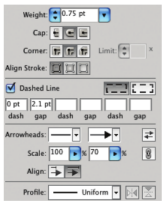


Figure 4-31 *Options pour les lignes pointillées dans la boîte de dialogue Trait*

Utilisez les mêmes outils et techniques que précédemment pour obtenir le graphique final de la figure 4-32. Créez des graduations sur l'axe horizontal avec l'outil Plume, puis ajoutez et modifiez des libellés avec l'outil Texte. N'oubliez pas d'ajouter la source de vos données sous le graphique afin qu'il soit complet.

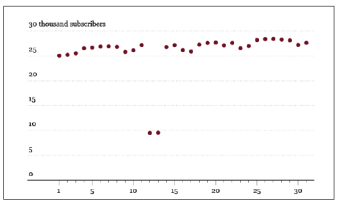


Figure 4-32 *Ajout de lignes de grille et de graduations sur l'axe des valeurs*

Données continues

La visualisation de données temporelles continues est similaire à la visualisation de données discrètes. Après tout, vous avez bel et bien un nombre discret de points de données, même si l'ensemble des données est continu. La structure des données continues et des données discrètes est la même. Leur différence réside dans ce qu'elles représentent dans le monde réel. Comme évoqué précédemment, les donnés continues représentent des phénomènes en évolution constante ; vous voudrez que la visualisation illustre cet aspect.

Connecter les points

À ce stade, rien de nouveau. Le graphique chronologique est similaire au dessin de points, à la différence que vous connectez les points à l'aide de lignes. Souvent, les points n'apparaissent pas, comme l'illustre la figure 4-33 qui montre la géométrie du type de graphique répandu.

Nous avons les nœuds, ou points, de coordonnées X et Y, puis les lignes continues qui permettent de visualiser les tendances. Il est généralement recommandé de démarrer l'axe des valeurs à zéro, car toute autre valeur de départ pourrait affecter l'échelle.

L'extension de l'axe horizontal peut aussi avoir une influence sur l'aspect des tendances. S'il est trop écrasé, une augmentation d'un point à un point peut sembler plus grande qu'elle ne l'est ; s'il est trop étiré, aucun modèle ne risque d'apparaître.

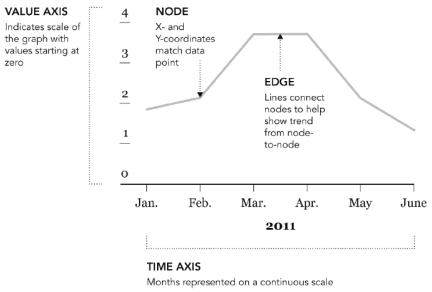


Figure 4-33 *Structure d'un graphique chronologique Créer un graphique chronologique*

Si vous savez créer un nuage de points dans R, vous savez créer un graphique chronologique. Chargez vos données et utilisez la fonction `plot()`, mais au lieu d'employer `p` dans l'argument `type`, choisissez `l`, ce qui correspond à ligne.

En guise de démonstration, utilisez les données de la population mondiale en provenance de la Banque mondiale, de 1960 à 2009. Comme d'ordinaire, chargez les données avec la fonction `read.csv()`.

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

Voici à quoi ressemblent les toutes premières lignes, avec uniquement l'année et la population.

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

Utilisez la fonction `plot()` et spécifiez les coordonnées X et Y, le type, les limites de l'axe des valeurs et les libellés des axes.

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

Votre graphique doit ressembler à celui de la figure 4-34.

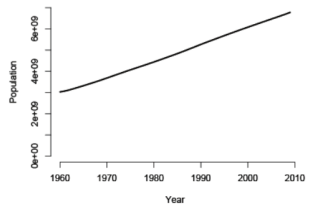


Figure 4-34 *Graphique chronologique par défaut dans R*

Vous pourriez enregistrer l'image comme fichier PDF et la modifier dans Illustrator comme vous l'avez déjà fait, mais nous allons essayer une autre solution. Vous allez créer la totalité du graphique dans Illustrator avec l'outil Graphe linéaire. Il s'agit de l'un des outils de graphe qui permettent de créer des graphiques de base dans Illustrator (figure 4-35).

Figure 4-35 *Outils de Graphe dans Illustrator*

Sélectionnez donc l'outil Graphe linéaire de la boîte de dialogue Outils en cliquant sur l'icône correspondante et en maintenant le bouton de la souris enfoncé afin de sélectionner le type de graphique souhaité.

Téléchargez les données relatives à la population mondiale à l'adresse *http://datasets.flowingdata.com/world-population.csv*. Dans Illustrator, vous ne pouvez pas charger directement les données depuis une URL comme c'est le cas en R. Vous devez enregistrer le fichier sur votre ordinateur. Ouvrez ensuite le fichier CSV dans Excel (figure 4-36) ou Google Drive. Copiez toutes les lignes, à l'exception de celle de l'en-tête qui contient les noms des colonnes. Vous collerez les données dans Illustrator par la suite.

Retournez dans Illustrator et resélectionnez l'outil Graphe linéaire. Cliquez sur un rectangle et faites-le glisser jusqu'à ce qu'il ait la taille approximative du graphique souhaitée. Une feuille de calcul s'affiche (figure 4-37).

Collez les données copiées depuis Excel, puis cliquez sur la coche en haut à droite. Vous devez alors voir apparaître un graphique qui ressemble à celui de la figure 4-38.

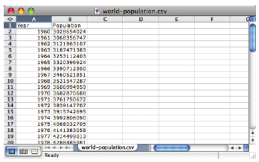


Figure 4-36 *Fichier CSV ouvert dans Excel*

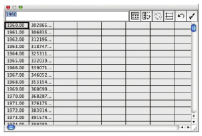


Figure 4-37 *Feuille de calcul pour saisir les données dans Illustrator*

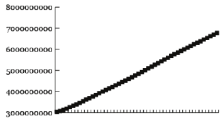


Figure 4-38 *Graphique linéaire par défaut dans Illustrator*

Si vous utilisez un type de graphique de base que vous modifiez par la suite dans Illustrator, vous pouvez gagner du temps et créer directement le graphique dans Illustrator.

Cependant, il vous reste à modifier certaines options pour que le graphique soit plus élaboré. Cliquez avec le bouton droit de la souris et sélectionnez Texte. Désactivez la case à cocher Marquer les points (figure 4-39).

Sélectionnez Axe des catégories dans le menu déroulant, puis choisissez Aucune comme longueur de graduation. Cliquez sur OK. Vous obtiendrez ainsi un graphique plus clair, moins encombré. Poursuivez en procédant comme vous l'avez fait pour modifier les graphiques générés dans R.

Vous pouvez supprimer l'axe vertical, simplifier les libellés des valeurs, ajouter sur l'axe horizontal des graduations et des libellés pour les années ou encore insérer un titre et une brève présentation. Vous pouvez aussi modifier le style de trait pour que la ligne se distingue mieux. Le gris clair, valeur par défaut, donne le sentiment que les données se trouvent à l'arrière-plan alors qu'elles devraient se trouver au premier plan et au centre. Une fois que vous aurez procédé aux modifications nécessaires, vous devriez obtenir un graphique similaire à celui de la figure 4-40.

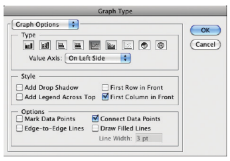


Figure 4-39 *Options de Graphe dans Illustrator*

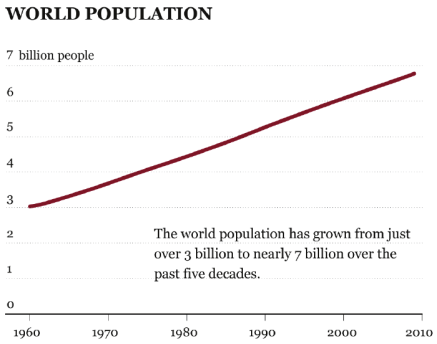


Figure 4-40 *Population mondiale sur les cinq dernières décennies*

Le point principal est qu'il est possible de créer le même graphique dans Illustrator et dans R – vous obtiendriez bel et bien le même résultat final. À vous donc de choisir l'outil avec lequel vous êtes le plus à l'aise. Le plus important est d'obtenir des résultats.

Encore un effort !

L'un des inconvénients du tracé de ligne standard est qu'il implique un changement régulier entre le point A et le point B. Aucun problème dans le cas d'une mesure comme celle de la population mondiale, mais il arrive que certains phénomènes demeurent à une certaine valeur pendant une longue période, puis qu'ils subissent une accélération ou une dégradation soudaines. Les taux d'intérêt, par exemple, peuvent demeurer identiques pendant des mois, puis chuter brusquement en une journée. Pour ce type de données, utilisez un graphique en escaliers, comme illustré à la figure 4-41.

Au lieu de connecter directement le point A et le point B, la ligne conserve la même valeur jusqu'à ce qu'il y ait un changement, auquel cas la ligne grimpe (ou redescend) jusqu'à la prochaine valeur. Vous vous retrouvez ainsi avec un ensemble de marches.

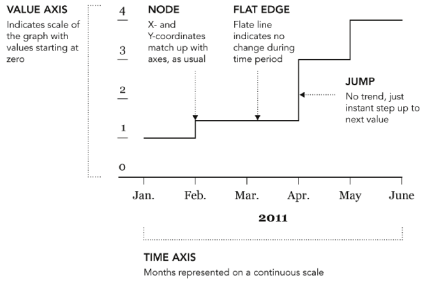


Figure 4-41 *Structure de base du graphique en escaliers*

Créer un graphique en escaliers

Illustrator ne propose pas d'outil pour créer facilement un graphique en escaliers, contrairement à R. Aussi, vous pouvez créer un graphique en escaliers dans R et le modifier dans Illustrator. Commencez-vous à voir un modèle ici ?

La figure 4-42 illustre le graphique final. Celui-ci représente l'évolution du prix des timbres pour les lettres aux États-Unis. Notez que l'évolution ne se produit pas sur une base régulière. De 1995 à 1999, le timbre est resté fixe à 32 cents, soit 4 années sans aucune modification. Cependant, de 2006 à 2009, le prix a augmenté chaque année.

Pour créer le graphique en escaliers dans R, suivez le même processus que celui utilisé dans l'ensemble du chapitre :

1. Chargez les données.
2. Assurez-vous que les données soient correctement mises en forme.
3. Utilisez une fonction de R pour créer un tracé.

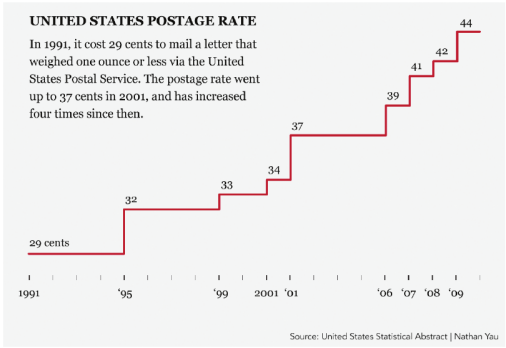


Figure 4-42 *Graphique illustrant l'augmentation du prix du timbre*

Vous trouverez les prix du timbre américain, ainsi que de nombreux autres ensembles de données, auprès de l'*United States Statistical Abstract*. Je les ai placés dans un fichier CSV disponible à l'adresse *http://datasets.flowingdata.com/us-postage.csv*. Associez cette URL à la fonction `read.csv()` comme source de chargement du fichier de données dans R.

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

Vous trouverez ci-après l'ensemble complet des données. Elles ne comportent que neuf points, un par évolution du prix entre 1991 et 2009, et un dixième point de données qui indique le prix en 2010. La première colonne correspond à l'année et la seconde au prix en dollars américains.

```
     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
```

La fonction `plot()` simplifie la création d'un graphique en escaliers. Vous entrez l'année comme coordonnée X, le prix comme coordonnée Y et vous utilisez l'argument `s` (qui correspond à *step*, soit escalier) pour le `type`.

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

Vous pouvez également spécifier le titre principal et les libellés des axes, si vous le désirez.

```
plot(postage$Year, postage$Price, type="s",
    main="US Postage Rates for Letters, First Ounce, 1991-2010",
    xlab="Année", ylab="Prix du timbre (dollars)")
```

Vous obtenez ainsi le graphique en escaliers correspondant au prix du timbre, comme illustré à la figure 4-43.

Juste pour le plaisir, regardez à quoi le graphique ressemblerait si vous aviez utilisé un tracé linéaire (figure 4-44).

Il existe une tendance à l'augmentation, mais comment se rend-on compte qu'il s'agit d'une hausse régulière ? À aucun moment entre 2001 et 2006 le prix n'a été de 38 cents, mais il serait impossible de le déterminer à partir de ce tracé linéaire, sauf à regarder les données brutes.

Figure 4-43 *Graphique en escaliers créé dans R*

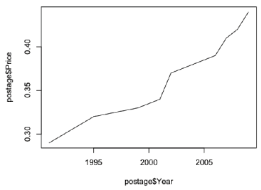


Figure 4-44 *Tracé linéaire des prix du timbre*

Lorsque vous avez un ensemble de données réduit, il peut parfois être utile de libeller directement les points au lieu d'utiliser un axe des valeurs et une grille. Cela permet de mieux souligner les données et, comme il n'y a pas tant de points que cela, les libellés ne se confondront pas les uns avec les autres.

Enregistrez l'image au format PDF, puis ouvrez-la dans Illustrator. Comme précédemment, modifiez le graphique en escaliers à votre convenance. Soucieux de la présentation, j'ai supprimé l'axe vertical et j'ai nommé directement chaque « bond » avec l'outil Texte. J'ai également espacé régulièrement les graduations, mais je n'ai fait apparaître que les années où une modification s'était produite.

Pour finir, j'ai choisi un arrière-plan gris. C'est une préférence personnelle, mais l'arrière-plan contribue à mettre le graphique en valeur, notamment quand il est placé à l'intérieur du texte. Il offre un meilleur environnement, sans être trop clinquant. Pour insérer un arrière-plan derrière le graphique et le texte, vous devez créer un calque dans Illustrator à l'aide de la boîte de dialogue Calques. Cliquez sur le bouton correspondant pour créer un calque. Par défaut, le calque se place en haut ; pour le faire apparaître en bas, cliquez sur le nom du calque dans la boîte de dialogue et déplacez-le sous Layer 1, comme illustré à la figure 4-45.

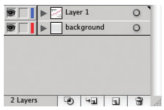


Figure 4-45 *Boîte de dialogue Calques dans Illustrator*

Vous pouvez renommer les calques, ce qui peut être particulièrement utile si vous créez des graphiques plus complexes. J'ai intitulé le nouveau calque « background ». Dessinez ensuite un rectangle avec l'outil Rectangle. Cliquez dessus pour l'agrandir à la taille souhaitée et changez la couleur via la boîte de dialogue Couleur.

Lissage et estimation

Lorsque vous avez une multitude de données, ou que les données contiennent beaucoup de « bruit », il peut être difficile de tracer des tendances et des modèles. Aussi, pour vous faciliter la tâche, vous pouvez estimer une courbe de tendances. La figure 4-46 illustre le principe de base.

Tracez une ligne qui traverse le plus grand nombre de points possible, et réduisez la distance cumulée entre les points et la ligne ajustée. Le plus simple consiste à créer une ligne droite ajustée à l'aide de l'équation d'interception de pente d'une ligne droite, que vous avez sans doute apprise lors de vos études.

```
y = mx + b
```

La pente est symbolisée par la lettre `m` et l'interception par la lettre `b`. Que se passe-t-il quand la tendance n'est pas linéaire ? Cela n'a aucun sens d'ajuster une ligne droite à des données qui décrivent sans cesse des creux et des courbes. Utilisez à la place la méthode statistique créée par William Cleveland et Susan Devlin, baptisée « LOESS » (technique de lissage d'une moyenne mobile). Elle permet d'ajuster une courbe aux données.

Figure 4-46 *Ajustement d'une ligne aux points de données*

La méthode LOESS est utilisée dès les premières données et procède par tranches. À chacune d'elles, elle estime une forme polynomiale de faible degré pour les seules données de la tranche. En se déplaçant le long des données, la méthode LOESS ajuste une multitude de courbes minuscules, qui forment ensemble une seule courbe. Pour plus d'informations sur cette méthode, effectuez une recherche dans Google avec comme mots-clés « méthode LOESS ». Maintenant, apprenons à appliquer la méthode LOESS à vos données.

Ajuster une courbe LOESS

L'histoire qui nous intéresse ici est celle du chômage aux États-Unis au cours des toutes dernières décennies. Il y a eu des hauts et des bas, ainsi que des variations saisonnières. Quelle est l'ampleur des tendances générales ? Comme illustré à la figure 4-47, le taux de chômage a atteint un point culminant dans les années 1980, a décliné au cours des années 1990, puis est monté en flèche vers 2008.

Pour plus d'informations sur la méthode LOESS, consultez l'article « Robust Locally Weighted Regression and Smoothing Scatterplots » publié dans *Journal of the American Statistical Association* (William Cleveland).

Figure 4-47 *Taux de chômage avec ajustement de la courbe LOESS*

La figure 4-48 montre à quoi ressemble le graphique du chômage en n'utilisant que les points avec la fonction `plot()` en R.

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

# Nuage de points simple
plot(1:length(unemployment$Value), unemployment$Value)
```


Figure 4-48 *Tracé du chômage à l'aide de points uniquement*

Maintenant, voici à quoi ressemblerait une ligne droite ajustée (figure 4-49). Elle n'est guère utile. Elle semble ignorer toutes les fluctuations que connaît le taux de chômage. Pour ajuster une courbe LOESS, il faut utiliser à la place la fonction `scatter.smooth()`.

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

Figure 4-49 *Ligne droite ajustée*

Le résultat obtenu, plus satisfaisant, est représenté à la figure 4-50. La ligne s'incurve vers le haut, en rendant compte du pic des années 1980.

Figure 4-50 *Ajustement d'une courbe LOESS*

Vous pouvez régler le degré d'ajustement de la courbe via les arguments `degree` et `span` de la fonction `scatter.smooth()`. Le premier contrôle le degré des formes polynomiales ajustées et le second le degré de lissage de la courbe.

Plus l'argument `span` est proche de zéro, plus l'ajustement est proche. La figure 4-51 illustre le résultat obtenu après avoir attribué la valeur 2 à `degree` et la valeur `0.5` à `span`. À présent, modifiez les couleurs et ajustez les limites des axes.

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

Les hauts et les bas de la courbe sont plus manifestes avec ces valeurs. Amusez-vous à varier les valeurs de `span` pour mieux comprendre comment le lissage s'en trouve affecté.

Figure 4-51 *Courbe LOESS ajustée avec un lissage moindre et une forme polynomiale de plus haut degré*

Pour obtenir le graphique final de la figure 4-47, enregistrez l'image au format PDF et ouvrez-la dans Illustrator. Les outils utilisés précédemment (Sélection, Texte, Plume…) permettent d'ajouter des titres, un arrière-plan et des graduations sur l'axe horizontal. La ligne ajustée a été rendue plus visible pour mieux mettre en évidence les tendances que les points de données individuelles.

Variez les styles de couleur et de trait pour mettre en valeur les parties du graphique qui sont les plus importantes par rapport à l'histoire que vous racontez.

Pour résumer

Il est plaisant d'explorer les modèles au fil du temps. Le temps est à ce point intégré à notre vie quotidienne que de nombreux aspects de la visualisation des données temporelles sont pratiquement intuitifs. Vous comprenez que les

choses changent et évoluent, la difficulté est de savoir jusqu'à quel point et d'apprendre ce qu'il faut rechercher dans les graphiques.

Il est facile de jeter un coup d'œil à quelques courbes d'un tracé et de dire que telle ou telle chose augmente, et c'est parfait. C'est à cela que sert la visualisation : avoir rapidement un aperçu général des données. Mais elle permet aussi d'aller plus loin. La visualisation peut être utilisée comme outil d'exploration. Effectuez un zoom avant sur les sections temporelles et demandez-vous pourquoi, un jour et seulement ce jour-là, la courbe affiche un petit accident, ou pourquoi, tel autre jour, il y a un pic. C'est à ce moment-là que les données sont intéressantes : plus vous en savez sur vos données, meilleure sera l'histoire que vous raconterez.

Après avoir appris ce sur quoi portent vos données, expliquez ces informations détaillées dans votre graphique de données. Mettez en évidence les parties intéressantes afin que vos lecteurs sachent où regarder. Un graphique ordinaire peut vous paraître convenir, mais, sans contexte, il sera ennuyeux pour quiconque.

Vous avez utilisé R avec Illustrator pour parvenir à vos fins. R a construit la base et Illustrator a permis de créer les graphiques qui soulignaient les aspects importants des données. Les types de graphiques abordés n'étaient qu'un sous-ensemble de ce qu'il est possible de faire avec les données temporelles. Vous ouvrez tout un nouveau sac d'astuces lorsqu'à l'ensemble, vous ajoutez de l'animation et de l'interaction, ce que nous ferons dans le prochain chapitre. Lorsque vous aborderez un nouveau type de données, en l'occurrence les proportions, vous pourrez appliquer le même processus de programmation et les mêmes principes que ceux étudiés dans ce chapitre, même si vous utilisez un autre langage de programmation.

Chapitre 5

Visualisation des proportions

Les données temporelles sont naturellement regroupées par période. En effet, une série d'événements se produit pendant une période de temps spécifique. Les données relatives aux proportions sont aussi regroupées, mais par catégorie, sous-catégorie et population. Par population, je n'entends pas simplement la population humaine. Ici, la population représente tous les choix ou résultats possibles. C'est l'espace des échantillons.

Dans un sondage, il est demandé aux personnes si elles sont d'accord, en désaccord ou sans opinion à propos de telle ou telle affirmation. Chaque catégorie représente quelque chose, et la somme des parties représente un tout.

Ce chapitre traite de la représentation des catégories individuelles, mais propose aussi une vue d'ensemble des liens entre les différents choix. Ce que vous avez appris au chapitre précédent va vous être utile pour aborder ici les graphiques interactifs, élaborés à l'aide de code HTML, de feuilles de style (*Cascading Style Sheets*, CSS) et de JavaScript. Vous pourrez ensuite vous intéresser aux graphiques conçus avec Flash.

Que rechercher dans les proportions ?

Dans le cas des proportions, vous recherchez généralement trois choses : le maximum, le minimum et la distribution globale. Les deux premières valeurs s'obtiennent directement. Triez les données de la plus petite à la plus grande, et choisissez les extrémités, qui correspondent respectivement au maximum et au minimum. S'il s'agit de traiter les résultats d'un sondage, ces deux valeurs pourraient être la réponse la plus fréquente et la réponse la moins fréquente formulée par les participants ; si vous étiez en train d'établir le graphique des calories des différentes parties d'un repas, elles symboliseraient le plus grand contributeur et le plus petit contributeur au nombre total de calories.

Vous n'avez pas besoin d'un graphique, pour illustrer le minimum et le maximum. Ce qui vous intéresse le plus est la distribution des proportions. Comment la sélection de l'un des choix du sondage se compare-t-elle aux autres sélections possibles ? Les calories sont-elles distribuées de façon égale entre les graisses, les protéines et les glucides, ou bien un groupe domine-t-il ? Les types de graphiques suivants devraient vous aider à y voir plus clair.

Parties d'un tout

Il s'agit des proportions dans leur forme la plus simple. Vous avez un ensemble de proportions dont le total est égal à 1 ou un ensemble de pourcentages dont le total est égal à 100 %. Vous voulez montrer chaque partie par rapport aux autres, mais vous souhaitez aussi conserver l'impression d'un tout.

Graphique en camembert

Les graphiques en camembert sont la solution la plus répandue. Ils sont omniprésents, des présentations commerciales aux sites qui utilisent les graphiques comme support de plaisanteries. Le premier graphique en camembert connu fut publié par William Playfair, qui inventa aussi le graphique linéaire (courbe) et le graphique en barres (histogramme), en 1801. Merci, William !

Le principe des graphiques en camembert est simple : vous disposez d'un cercle, qui représente un tout, puis vous coupez des parts, comme vous le feriez avec une tarte. Chaque part représente une portion du tout (figure 5-1).

Figure 5-1 *Graphique en camembert généralisé*

La somme des pourcentages de toutes les parts doit être égale à 100 %. Si tel n'est pas le cas, vous avez commis une erreur.

Il est reproché aux graphiques en camembert de ne pas être aussi précis que les graphiques en barres ou que les visuels basés sur les positions. Aussi, certains pensent qu'ils devraient être complètement évités. Il est plus facile d'estimer une longueur que des surfaces et des angles. Cependant, cela ne signifie pas pour autant que vous devez absolument vous en passer.

Vous pouvez utiliser sans aucun problème le graphique en camembert, à condition de bien connaître ses limites : organisez bien vos données et ne découpez pas le graphique en un trop grand nombre de parts.

Créer un graphique en camembert

Presque tous les programmes de création de graphique permettent de générer des graphiques en camembert. Vous pouvez aussi utiliser Illustrator, abordé au chapitre précédent. Le processus qui consiste à ajouter les données, à créer un graphique par défaut et à l'affiner, doit vous sembler familier à présent.

Créer la base du graphique, à savoir le cercle, est assez simple. Ouvrez un nouveau document et sélectionnez l'outil Graphe sectoriel de la boîte de dialogue Outils (figure 5-2). Cliquez et dessinez un rectangle de la taille souhaitée (vous pourrez le redimensionner ultérieurement).

Figure 5-2 *Boîte de dialogue Outils dans Illustrator*

Lorsque vous relâchez le bouton de la souris, une feuille de calcul s'affiche dans laquelle vous pouvez saisir vos données. Pour un graphique en camembert simple, entrez chaque point de données de la gauche vers la droite ; les valeurs apparaîtront dans le même ordre sur votre graphique.

Pour cet exemple, vous allez utiliser les résultats d'un sondage réalisé sur le site FlowingData. Il a été demandé aux lecteurs d'indiquer quel domaine les intéressait le plus. Il y eut 831 réponses.

Tableau 5-1

Zone d'intérêt	Nombre de votes
Statistiques *(Statistics)*	172
Conception *(Design)*	136
Business *(Business)*	135
Cartographie *(Cartography)*	101
Sciences de l'information *(Information Science)*	80
Analyse web *(Web Analytics)*	68
Programmation *(Programming)*	50
Ingénierie *(Engineering)*	29
Mathématiques *(Mathematics)*	19
Autre *(Other)*	41

Saisissez les nombres dans la feuille de calcul, comme illustré à la figure 5-3. L'ordre dans lequel vous les entrez correspondra à l'ordre de parts dans le graphique en camembert, en commençant par le haut et en poursuivant dans le sens des aiguilles d'une montre.

Notez que les résultats du sondage sont classés du plus grand au plus petit, et qu'ils se terminent par la catégorie « Autre ». Ce type de tri peut permettre de rendre vos graphiques plus faciles à lire. Cliquez sur la case à cocher dans le coin supérieur de la feuille lorsque vous avez terminé.

Figure 5-3 *Feuille de calcul dans Illustrator*

Le graphique en camembert par défaut s'affiche avec huit nuances de gris, selon un ordre apparemment aléatoire (figure 5-4). Il ressemble quelque peu à une sucette à niveaux de gris, mais il est aisé d'y remédier. Le point important, ici, est que vous disposez de la base de votre graphique en camembert.

Nous allons rendre le graphique plus lisible en modifiant certaines couleurs et en ajoutant un texte pour expliquer aux lecteurs ce qu'ils ont sous les yeux. Telles qu'elles sont pour l'instant, les couleurs n'ont pas beaucoup de sens. Elles se contentent simplement de séparer les parts. Mais vous pouvez les utiliser pour indiquer aux lecteurs ce qu'ils doivent regarder et dans quel ordre. Vous allez bien devoir trier les données de la plus grande à la plus petite.

Si vous commencez à 12 heures et continuez dans le sens des aiguilles d'une montre, vous devez voir un ordre décroissant. Cependant, en raison du jeu de couleurs arbitraire, certaines des parts les plus petites sont mises en valeur à l'aide de nuances plus sombres. La nuance sombre faisant en quelque sorte office de surligneur, assombrissez les plus grosses parts et éclaircissez les plus petites. Si pour une raison ou autre, vous voulez mettre en valeur les réponses les moins nombreuses, vous pouvez inverser les couleurs. Dans le cas présent, vous voulez savoir quels domaines suscitent le plus grand intérêt.

La feuille de calcul proposée dans Illustrator étant assez rudimentaire, il est difficile de manipuler ou réorganiser les données. Pour contourner cet obstacle, effectuez la gestion des données dans Excel, copiez-les puis collez-les dans Illustrator.

Figure 5-4 *Graphique en camembert par défaut*

La couleur peut jouer un rôle important dans la façon dont votre graphique est lu. Il ne s'agit pas seulement d'un composant esthétique, même si cela peut être le cas parfois. La couleur peut aussi être un indice visuel comme la longueur ou la surface.

Lorsque vous modifiez la valeur de l'opacité, la couleur de remplissage de la forme que vous êtes en train de modifier se mélangera à la couleur de l'arrière-plan. Dans l'exemple de la figure 5-4, l'arrière-plan est blanc donc plus la valeur d'opacité est faible, plus le blanc apparaît en transparence. Si l'arrière-plan avait été bleu, la couleur de remplissage de la forme aurait eu un aspect délavé.

Choisissez l'outil Sélection directe dans la boîte de dialogue Outils, puis cliquez sur une part du camembert. Modifiez la couleur de remplissage et la couleur de

trait à l'aide des options de la boîte de dialogue Couleur. La figure 5-5 représente le même graphique avec un trait de couleur blanche et les parts du camembert colorées de la plus foncée à la plus claire. Il est désormais plus facile de voir que les nombres sont classés du plus grand au plus petit, à l'exception de la dernière part (« Other »).

Figure 5-5 *Graphique dont les couleurs sont disposées de la plus sombre à la plus claire*

Bien sûr, rien ne vous oblige à être aussi économe en termes de couleurs. Vous pouvez utiliser celles de votre choix, comme illustré à la figure 5-6. Même s'il est généralement conseillé de ne pas employer de couleurs vives pour ne pas aveugler le lecteur, si elles conviennent à votre sujet, ne vous en privez pas !

Figure 5-6 *Graphique en camembert de couleur*

Comme il s'agit d'un sondage FlowingData, j'ai utilisé la nuance de rouge du logo FlowingData, puis j'ai progressivement diminué la valeur d'opacité, disponible dans la boîte de dialogue Transparence, pour chacune des parts du camembert. Avec une opacité de 0 %, la couleur de remplissage est totalement transparente et donc invisible ; avec une opacité de 100 %, la couleur de remplissage est complètement visible et opaque, l'arrière-plan est caché.

Enfin, ajoutez un titre à votre graphique, une phrase d'introduction et des libellés à l'aide de l'outil Texte. Avec un peu de pratique, vous connaîtrez mieux les

différentes polices proposées que vous voulez utiliser pour les en-têtes et pour le texte. Par ailleurs, aidez-vous des outils d'alignement d'Illustrator, qui seront vos meilleurs alliés pour placer précisément vos textes. Si ces derniers sont correctement alignés et régulièrement espacés, vos graphiques seront plus lisibles. Vous pouvez aussi utiliser l'outil Plume pour créer des flèches (figure 5-7) pour les trois dernières catégories du sondage. En effet, ces sections sont trop petites pour y insérer des libellés et trop proches pour disposer les libellés de façon adjacente.

Figure 5-7 *Graphique en camembert final avec libellés et texte d'introduction*

Graphique en anneau

Le graphique en camembert possède un petit cousin, le graphique en anneau. Il ressemble au graphique en camembert, mais avec un trou central qui lui donne l'aspect d'un beignet, comme illustré à la figure 5-8.

Figure 5-8 *Structure du graphique en anneau*

En raison du trou central, vous n'estimez plus les valeurs en fonction de l'angle, mais de la longueur d'arc. Vous pouvez vous retrouver avec les mêmes problèmes que lorsque vous utilisez un simple graphique avec un nombre trop élevé de catégories. En revanche, si les catégories sont moins nombreuses, le graphique en anneau est pratique.

Le point le plus important dont vous devez vous souvenir, que vous utilisiez un graphique en camembert ou un graphique en anneau, est qu'ils peuvent rapidement devenir encombrés. Ils ne sont pas destinés à représenter une multitude de valeurs.

Téléchargez D3 à l'adresse suivante : http://d3js.org/ et placez-le dans le même répertoire que celui employé pour enregistrer les exemples de fichiers.

Créer un graphique en anneau

Illustrator permet de générer facilement des graphiques en anneau. Pour cela, créez tout d'abord un graphique en camembert, puis dessinez un cercle au milieu. Comme précédemment, employez la couleur pour guider le lecteur.

Très souvent, la partie centrale des graphiques en anneau est utilisée pour accueillir un libellé ou tout autre contenu, comme illustré à la figure 5-9.

Créons maintenant le même graphique avec D3, boîte à outils de visualisation gratuite et open source. Il s'agit d'une bibliothèque implémentée en JavaScript et qui reprend les fonctionnalités SVG (*Scalable Vector Graphics*) des navigateurs modernes. Les graphiques sont générés dynamiquement et permettent l'animation et l'interactivité. L'utilisation de D3 est donc parfaitement adaptée pour créer des graphiques en ligne.

Figure 5-9 *Du graphique en camembert au graphique en anneau*

Même si vous faites le choix d'un autre langage de programmation, le processus sera le même que celui utilisé avec R et Illustrator. Ainsi, vous chargerez les données, puis vous construirez la base et enfin vous personnaliserez la présentation.

La figure 5-10 illustre le graphique final que nous souhaitons obtenir. Les libellés sont ici orientés en biais par rapport à la figure 5-9, qui sans cela est similaire, et lorsque la souris survole un arc de cercle, une infobulle affiche le nombre de votes pour la catégorie correspondante. Il est possible de définir une interaction bien plus élaborée, mais acquerez d'abord les bases pour pouvoir vous lancer ensuite dans plus de sophistication.

Figure 5-10 *Graphique en anneau avec D3*

La première à chose à faire est de créer une page HTML, que nous appellerons `donut.html`.

```
<html>
<head>
    <title>Graphique en anneau</title>
```

```
    <script type="text/javascript" src="d3.v3.min.js"></script>
    <style type="text/css">
      #figure {
          width: 400px;
          height: 400px;
      }
    </style>
  </head>
  <body>
    <div id="figure">
    </div><!-- @end figure -->
  </body>
  </html>
```

Si vous avez déjà créé une page web, les lignes précédentes doivent vous paraître simples. Dans le cas contraire, sachez qu'il s'agit de code HTML, tel que vous en trouvez presque partout en ligne. Chaque page commence par une balise `<html>` suivie d'une balise `<head>` contenant des informations sur la page, mais n'apparaissant pas dans la fenêtre de votre navigateur. Tout le contenu encadré par la balise `<body>` est visible. Intitulez la page `Graphique en anneau`. Chargez le fichier JavaScript correspondant à la bibliothèque D3 grâce à la balise `<script>`. Déclarez les feuilles de style utilisées pour mettre en forme la page HTML. Ensuite, définissez la largeur et la hauteur (`400px`) de la balise `<div>` à l'aide du sélecteur d'ID `figure`. C'est à cet emplacement que vous tracerez votre graphique. Ce code HTML ne fait pas réellement partie du graphique, mais il est nécessaire pour que les instructions JavaScript qui suivent soient correctement chargées dans votre navigateur. Pour l'instant, si vous chargez le fichier `donut.html` dans votre navigateur, vous ne voyez qu'une page blanche.

À l'intérieur de la balise `<div>`, spécifiez que le code que vous allez écrire est du JavaScript. Tout le reste prend place entre les balises `<script>`.

```
  <script type="text/javascript">
  </script>
```

Intéressons-nous tout d'abord aux données. Il s'agit des résultats du sondage sur FlowingData, lesquels sont stockés dans des tableaux. Les nombres de votes sont sauvegardés dans un tableau et les noms des catégories correspondantes dans un autre.

```
  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"];
```

Précisez ensuite la largeur et la hauteur du graphique en anneau, ainsi que la longueur du rayon et l'échelle de la longueur d'arc.

```
var w = 350,
    h = 350,
    r = w / 2,
```

La largeur et la hauteur du graphique en anneau sont toutes deux de 350 pixels, tandis que le rayon est égal à la moitié de la largeur, soit 175 pixels.

Créez ensuite une échelle de couleurs. Plus une catégorie reçoit de votes, plus le rouge doit être foncé. Dans Illustrator, vous l'avez fait manuellement, mais D3 peut choisir les couleurs à votre place. Il suffit pour cela de sélectionner la plage de couleurs souhaitée.

```
var depthColors = d3.scale.linear().domain([0,172]).range
➥(["white","#821122"]);
```

Vous disposez ainsi d'une gamme de couleurs allant du blanc au rouge foncé (`#821122`) sur une plage linéaire comprise entre 0 et 172, le nombre de votes le plus élevé. Autrement dit, une catégorie avec 0 vote sera blanche et une catégorie avec 172 votes sera rouge foncé. Les catégories dont les nombres de votes sont compris entre ces deux valeurs auront une couleur située entre le blanc et le rouge foncé.

Pour le moment, vous n'avez que des variables. Vous avez spécifié la taille et l'échelle. Pour créer le graphique proprement dit, définissez un conteneur SVG de 350 pixels (largeur) par 350 pixels (hauteur). Nous lui rajoutons un groupe `svg ("g")` centré au milieu du conteneur, pour pouvoir plus facilement placer les éléments du graphique.

```
var vis = d3.select("#figure")
    .append("svg").attr({width:w,height:h})
      .append("g").attr("transform","translate("+w/2+","+h/2+")");
```

Complétez le panneau en lui ajoutant les différents éléments. Le code peut vous paraître un peu confus, nous allons l'examiner ligne par ligne un peu plus loin.

```
var pie = d3.layout.pie()
     .sort(null)
     .value(function(d) {return d;});

var arc = d3.svg.arc()
     .outerRadius(r)
     .innerRadius(r - 120);

var g= vis.selectAll(".arc")
   .data(pie(data))
      .enter()
      .append("g")
```

```
          .attr("class", "arc");

g.append("path")
        .attr("d", arc)
        .style({fill: function(d) {return depthColors(d.data);},
        ➥stroke:"white"})
                  .append("title").text(function(d) {return d.data+"
                  ➥votes";})

g.append("text")
        .attr("transform",function(d) {return "translate
        ➥("+arc.centroid(d) +") rotate ("+((90*(d.startAngle+d.
        ➥endAngle)/Math.PI-90))+")";})
        .attr("dy", ".35em")
        .style("text-anchor", "middle")
        .text(function(d,i) {return cats[i];});
```

Dans un premier temps, nous créons une fonction `pie` qui va transformer nos données afin de les convertir en informations qui seront directement visualisables, telles que l'angle de début ou de fin de chaque segment. Ensuite, nous créons une fonction `arc` qui va calculer le tracé de chaque segment sur la base de ces informations.

Puis, nous allons ajouter un groupe (`"g"`) pour chaque élément de données. Pour chacun de ces groupes, nous allons dessiner un segment (`"path"`). Nous allons utiliser les fonctions préalablement définies pour passer les données appropriées et récupérer le tracé de chaque segment.

Au lieu que le style de remplissage soit défini avec une nuance statique, les couleurs de remplissage sont déterminées par la valeur du point de données et l'échelle de couleurs stockée comme propriété `depthColors` – autrement dit, la couleur est déterminée par une fonction de chaque point. Nous choisissons une bordure blanche (`"white"`), spécifiée par `stroke`.

Pour afficher une infobulle qui indique le nombre de votes obtenus lorsque la souris survole une section, nous employons un élément `"title"`. Une autre option consisterait à créer un événement `mouseover` dans lequel nous indiquerions ce qui se passe quand l'utilisateur place un pointeur sur un objet. Mais comme les navigateurs affichent automatiquement la valeur de l'élément `title`, il est plus facile de l'utiliser. Le titre se compose ainsi de la valeur de chaque point de données suivi de la chaîne « votes ». Enfin, nous ajoutons les libellés pour chaque section. Et il ne reste plus qu'à ajouter « May 2009 » au centre du graphique.

```
vis.append("text").test("May 2009").style("font","bold 14px Georgia");
```

Ces instructions peuvent s'interpréter ainsi : « placer le libellé May 2009 au centre du graphique, en police Georgia 14 pixels et en caractères gras ».

Le graphique est maintenant prêt à être affiché. Lorsque vous ouvrez le fichier `donut.html` dans votre navigateur, vous devez maintenant obtenir un graphique semblable à celui de la figure 5-10.

Rendez-vous à l'adresse *http://book.flowing-data.com/ch05/donut.html* pour visualiser le graphique en direct et afficher la source du code dans son intégralité.

Si vous débutez en programmation, il se peut que cette section ait quelque chose d'intimidant. Mais la bonne nouvelle est que D3 a été conçu selon le principe de l'apprentissage par l'exemple. Le site de la bibliothèque contient de nombreux exemples opérationnels qui vous seront très utiles et que vous pouvez utiliser avec vos propres données. Il propose aussi bien des graphiques statistiques traditionnels que des graphiques interactifs et animés plus avancés. Par conséquent, ne vous découragez pas si vous vous sentez un peu perdu. Vous finirez par être récompensé de vos efforts ! La section suivante va vous permettre de vous familiariser davantage avec D3.

Empiler

Dans le précédent chapitre, vous avez utilisé le graphique en barres empilées pour illustrer les données au fil du temps. Ce type de graphique peut être employé pour d'autres types de données comme les données de catégorie (figure 5-11).

Figure 5-11 *Graphique en barres empilées avec catégories*

Par exemple, nous allons examiner les résultats d'un sondage sur Barack Obama réalisé en juillet et en août 2010 par Gallup et CBS. Il a été demandé aux personnes interrogées si elles approuvaient ou désapprouvaient la politique du Président Obama concernant 13 grands thèmes. Les résultats sont indiqués dans le tableau 5-2.

Tableau 5-2

Thème	D'accord	Pas d'accord	Sans opinion
Relations inter-raciales *(Race relations)*	52	38	10
Éducation *(Education)*	49	40	11
Terrorisme *(Terrorism)*	48	45	7
Politique énergétique *(Energy policy)*	47	42	11
Affaires étrangères *(Foreign affairs)*	44	48	8
Environnement *(Environment)*	43	51	6
Situation en Irak *(Situation in Iraq)*	41	53	6
Impôts *(Taxes)*	41	54	5
Politique de santé *(Healthcare policy)*	40	57	3
Économie *(Economy)*	38	59	3
Situation en Afghanistan *(Situation in Afghanistan)*	36	57	7
Déficit du budget federal *(Federal budget deficit)*	31	64	5
Immigration *(Immigration)*	29	62	9

Une option consisterait à créer un graphique en camembert pour chaque thème, comme illustré à la figure 5-12. Pour les créer dans Illustrator, il suffit d'entrer plusieurs lignes de données au lieu d'une seule. Un graphique en camembert sera ainsi généré pour chaque ligne.

Un graphique en barres empilées permettrait de comparer plus facilement les taux d'approbation obtenus pour les différents thèmes, car il est plus facile d'estimer la longueur d'une barre que l'angle d'un morceau. Au chapitre précédent, vous avez obtenu dans Illustrator un graphique en barres empilées. Vous allez ici ajouter quelques interactions simples.

Figure 5-12 *Ensemble de graphiques en camembert*

Créer un graphique en barres empilées interactif

Comme pour l'exemple du graphique en anneau, utilisez D3 pour créer un graphique en barres empilées interactif. La figure 5-13 représente le graphique final. Deux interactions élémentaires sont à implémenter : la première affiche la valeur en pourcentage d'un empilement donné lorsqu'il est survolé par la souris, la seconde met en valeur les barres dans les catégories « Approve » (« D'accord »), « Disapprove » (« Pas d'accord ») et « No opinion » (« Sans opinion »), selon l'emplacement du curseur de la souris.

Figure 5-13 *Graphique en barres empilées interactif dans D3*

Pour commencer, configurez la page HTML et chargez le fichier D3 JavaScript requis.

```
<html>
<head>
    <title>Graphique en barres empilées</title>
    <script type="text/javascript" src="d3.v3.min.js"></script>
</head>
<body>
    <div id="figure-wrapper">
        <div id="figure">
    </div><!-- @end figure -->
    </div><!-- @end figure-wrapper -->
</body>
</html>
```

Le code doit vous sembler familier. Vous avez procédé de même pour créer un graphique en anneau avec D3. La seule différence ici est que le titre de la page est « Graphique en barres empilées » et qu'il existe une autre balise `<div>` avec le sélecteur d'ID `figure-wrapper`. Nous n'avons pas non plus ajouté de feuilles de style, ce que nous ferons plus tard.

Passons maintenant à JavaScript. À l'intérieur de la balise `<div>` avec le sélecteur d'ID `figure`, chargez les données et préparez-les sous forme de tableaux.

```
<script type="text/javascript+protovis">
    var layers= "[ {"name":"approve","values":[{"x":0,"y":52},
                  {"x":1,"y":49},{"x":2,"y":48},{"x":3,"y":47},
                  {"x":4,"y":44},{"x":5,"y":43},{"x":6,"y":41},
                  {"x":7,"y":41},{"x":8,"y":40},{"x":9,"y":38},
                  {"x":10,"y":36},{"x":11,"y":31},
                  {"x":12,"y":29}]},{"name":"disapprove",
                   "values":[{"x":0,"y":38},{"x":1,"y":40},
                  {"x":2,"y":45},{"x":3,"y":42},{"x":4,"y":48},
                  {"x":5,"y":51},{"x":6,"y":53},{"x":7,"y":54},
                  {"x":8,"y":57},{"x":9,"y":59},
                  {"x":10,"y":57},{"x":11,"y":64},
                  {"x":12,"y":62}]},{"name":"no
                     opinion","values":[{"x":0,"y":10},
                  {"x":1,"y":11},{"x":2,"y":7},{"x":3,"y":11},
                  {"x":4,"y":8},{"x":5,"y":6},{"x":6,"y":6},
                  {"x":7,"y":5},{"x":8,"y":3},{"x":9,"y":3},
                  {"x":10,"y":7},{"x":11,"y":5},
                  {"x":12,"y":9}]}]"
</script>
```

Par exemple, pour les relations entre les différentes races, 52 % des personnes approuvent la politique du Président Obama et 38 % la désapprouvent. De même, en matière d'éducation, 49 % sont d'accord et 40 % en désaccord.

Pour que le code soit plus facile à écrire, vous pouvez scinder les données et les stocker dans deux variables.

```
var cat = ["Race Relations","Education","Terrorism",
           "Energy Policy","Foreign Affairs","Environment",
           "Situation in Iraq","Taxes","Healthcare Policy",
           "Economy","Situation in Afghanistan",
           "Federal Budget Deficit","Immigration"]
```

Le tableau des thèmes est stocké dans `cat` et les données se présentent désormais sous la forme d'un tableau de tableaux.

Configurez les variables requises pour la largeur (`w`), la hauteur (`h`), l'échelle (`x`, `y`) et les couleurs (`fill`) comme suit :

```
var w = 400,
h = 250,
x = d3.scale.ordinal().domain(cat).rangeBands([0,w],.2),
y = d3.scale.linear().domain([0,100]).range([0,h]),
fill = ["#809EAD", "#B1C0C9", "#D7D6CB"];
```

La largeur du graphique est égale à 400 pixels et la hauteur à 250 pixels. L'échelle horizontale est ordinale, ce qui signifie que vous avez défini des catégories, par opposition à une échelle continue. Les catégories correspondent aux thèmes traités par le sondage. Les 4/5 de la largeur du graphique sont utilisés pour les barres, et le 1/5 restant pour le remplissage entre les barres.

L'axe vertical, qui représente les pourcentages, est une échelle linéaire comprise entre 0 et 100 %. Les barres peuvent se trouver à n'importe quelle hauteur entre 0 pixel et la hauteur du graphique, soit 250 pixels.

Enfin, le remplissage est spécifié dans un tableau grâce à des valeurs hexadécimales. Le bleu foncé correspond à une opinion favorable, le bleu clair à une opinion défavorable et le gris clair à l'absence d'opinion. Vous pouvez modifier ces couleurs à votre convenance.

L'étape suivante consiste à initialiser la visualisation avec la largeur et la hauteur spécifiées. Comme la partie restante fournit un remplissage autour du graphique réel, vous pouvez ajuster les libellés des axes.

```
var vis = d3.select("#figure")
    .append("svg")
    .attr({width:w+100,height:h+150});
    .append("g")
    .attr("transform","translate(100,20)")
```

Pour pouvoir ajouter les barres empilées à votre toile, D3 propose une méthode de disposition particulière pour les graphiques empilés et judicieusement intitulée « stack ». Dans le cas présent, nous l'utilisons pour un graphique en barres empilées, mais cette méthode de disposition peut également être employée avec les graphiques en aire ou en flux empilés. Stockez la nouvelle disposition dans la variable `bar`.

```
var bar=d3.layout.stack().values(function(d) {return d.values;})
var g=vis.selectAll("g").data(bar(layers)).enter().append("g").
➥style("fill",function(d,i) {return fill[i]});
var rect=
g.selectAll("rect").data(function(d) {return d.values;}).enter().
➥append("rect");
rect.attr({y: function(d) {return y(100-d.y0-d.y)
```

```
,height:function(d) {return y(d.y);},x:function(d,i) {return x
➥(cat[i])},width:x.rangeBand()})
rect.on("mouseover",function() {d3.select(this).style
➥("fill","#555");});
rect.on("mouseout",function() {d3.select(this).style("fill",null)})
rect.append("title").text(function(d) {return d.y+"%";})
```

Si vous hésitez sur le choix des couleurs, servez-vous de l'outil ColorBrewer (*http://colorbrewer2.org*) pour commencer. Il permet de spécifier le nombre et le type de couleurs à employer, et propose une échelle de couleurs que vous pouvez copier selon différents formats. L'outil 0to255 (*http://0to255.com*), est quant à lui plus général, mais je l'utilise souvent.

Une autre solution consiste à dessiner chacune des couches individuellement en calculant au préalable la position de chacune des barres.

L'interaction dans D3 ne se limite pas aux déplacements de la souris. Il est également possible d'associer des événements au clic et au double-clic. Pour plus d'informations, consultez la documentation de D3.

Ici, nous utilisons la fonction `bar` pour effectuer ces calculs. Nous créons trois groupes auxquels nous attribuons trois couleurs de remplissage différentes, puis nous ajoutons à chaque groupe un rectangle par barre. Ces rectangles récupèrent de la fonction `bar` appliquée à nos données `layers` les informations qui permettent de les dessiner : position sur l'axe des x, hauteur et position de la ligne de base. La largeur de chaque barre reste identique et vous pouvez l'obtenir à partir de l'échelle ordinale que vous avez déjà spécifiée.

Grâce à la dernière ligne de l'extrait de code précédent, le graphique est interactif. L'ajout d'un élément `title` est équivalent à la définition de l'attribut `title` d'un élément HTML tel qu'une image. Lorsque le curseur de la souris survole une image sur une page web, une infobulle apparaît si vous avez défini le titre de cette image. Il en va de même ici : une infobulle apparaît lorsque le curseur de la souris reste positionné sur une barre pendant une seconde. L'infobulle permet d'afficher la valeur en pourcentage représentée par la barre, suivie du signe pourcentage (%).

Pour que les couches apparaissent en surbrillance chaque fois que vous placez le pointeur de la souris sur une barre, utilisez la fonction `on` associée à des événements. L'événement `mouseover` change la couleur de remplissage en gris foncé (#555), laquelle revient à sa couleur d'origine grâce à l'événement `mouseout` lorsque le curseur de la souris ne se trouve plus sur la barre.

Ouvrez la page dans votre navigateur web (Firefox ou Safari, par exemple). Vous devez voir apparaître un graphique semblable à celui de la figure 5-14.

Déplacez le curseur de la souris sur une barre, la couche s'affiche en surbrillance. Il manque encore quelques informations, comme les axes et les libellés. Ajoutons-les maintenant.

Sur la figure 5-13, un certain nombre de libellés se trouvent sur les barres, mais uniquement sur les plus grandes d'entre elles, à savoir celles de couleur grise. Voici comment procéder.

```
g.selectAll("text").data(function(d) {return d.values;}).enter().
➥append("text")
.text(function(d) {if(d.y>11) return +d.y;})
.attr({x:function(d,i) {return x(cat[i])+x.rangeBand()/2;},
y:function(d) {return y(d.y0)-y(d.y)/2;},"text-anchor":"middle"})
.style("font-color":"white")
.textStyle("white")
.text(function(d) d.toFixed(0));
```

Figure 5-14 *Graphique en barres empilées sans libellé*

Pour chaque barre, regardez si la valeur est supérieure à 11 %. Si tel est le cas, un libellé de couleur blanche indiquant le pourcentage arrondi à l'entier le plus proche apparaît au milieu de la barre.

Ajoutez à présent les libellés de chaque thème sur l'axe des x. Idéalement, vous aimeriez que tous les libellés se lisent horizontalement, mais vous n'avez manifestement pas l'espace nécessaire. S'il s'agissait d'un graphique

en barres horizontales, vous pourriez ajuster des libellés horizontaux. Dans le cas présent, vous choisissez de les afficher avec un angle de 45 degrés. Il serait possible d'afficher les libellés à 90 degrés, mais leur lecture en serait plus difficile.

```
vis.selectAll(".labelx").data(cat).enter().append("g").classed("labelx",1).
➥attr("transform",function(d) {return "translate("+x(d)+","+h)}
.append("text");
.text(String).attr({x:0,y:10,"text-anchor":"end","transform":"
➥rotate(-45)"})
```

Le fonctionnement est le même que celui obtenu lorsque vous avez ajouté les libellés des nombres au milieu de chaque barre. Cependant, nous allons cette fois-ci créer des groupes pour ces libellés que nous allons placer en bas de chaque barre. Nous allons ensuite ajouter à chacun de ces groupes un libellé correspondant à chaque catégorie. Finalement, nous allons orienter ces libellés à 45 degrés et nous assurer qu'ils sont alignés à droite.

Nous obtenons ainsi les libellés des catégories. Les libellés correspondant aux valeurs de l'axe vertical sont ajoutés de la même manière, il reste encore à placer les graduations.

```
var ticks= vis.selectAll(".ticks").data(y.ticks(10)).enter().
➥append("g").classed("ticks",1);
ticks.attr("trnasform",function(d) {return "translate(0,"+y(100-d)+");})
ticks.append("path").attr("d","M0,0h-20").style({stroke:"black",opacity:
➥function(d){return d?.3:1;})
ticks.append("text").text(function(d) {return (d==100)?"100%":d;})
.attr({x:-10,y:5,"text-anchor":"middle"})
```

Ces instructions ajoutent onze groupes, chacun correspondant à `y.ticks(10)` (soit 0,10, ..., 100). Une ligne (`path`) est ajoutée à chaque groupe. Si la graduation n'est pas celle de la ligne 0, sa couleur est grise, sinon elle est noire. La seconde section ajoute alors les libellés au-dessus des graduations.

Comme l'axe horizontal est toujours manquant, nous ajoutons une autre ligne, de façon distincte, pour obtenir l'équivalent de la figure 5-15.

```
vis.append("path").attr("d","M-20"+h+"h"+(w+20)).style("stroke","black")
```

Le texte d'introduction et les libellés restants sont ajoutés avec des balises HTML et les feuilles de style CSS. Comme il existe de multiples ouvrages sur la conception web, je n'insisterai pas. Sachez toutefois que vous pouvez facilement associer HTML et CSS avec D3.

Figure 5-15 *Ajout de l'axe horizontal*

Hiérarchie et rectangles

En 1990, Ben Shneiderman (Université du Maryland) souhaita visualiser ce qui se passait sur son disque dur toujours plein. Il désirait savoir ce qui occupait autant d'espace. À partir de la structure hiérarchique des dossiers et des fichiers, il établit un premier diagramme arborescent. Cependant, celui-ci était trop grand pour s'avérer utile. Il comportait beaucoup trop de nœuds et beaucoup trop de branches.

La solution qu'il adopta porte le nom de *treemap* (arborescence de rectangles) Comme illustré à la figure 5-16, il s'agit d'une visualisation basée sur la surface, où la taille de chaque rectangle représente une mesure. Les rectangles externes correspondent aux catégories parentes et les rectangles au sein d'une catégorie parente représentent les sous-catégories. Vous pouvez vous servir d'une arborescence de rectangles pour visualiser directement les proportions, mais pour

que la technique soit utilisée au mieux, il est préférable de l'appliquer à des données hiérarchiques ou structurées de façon arborescente.

Figure 5-16 *Arborescence de rectangles généralisée*

Pour visualiser et interagir avec le graphique en barres empilées, rendez-vous à l'adresse *http://book.flowingdata.com/ch05/stacked-bar.html*. Afficher le code source pour voir comment les langages HTML, CSS et JavaScript peuvent être associés.

Pour obtenir un historique complet des arborescences de rectangles et des exemples supplémentaires décrits par leur créateur, Ben Shneiderman, rendez-vous à l'adresse suivante : *http://datafl.ws/11m*.

R est un environnement logiciel open source dédié aux calculs statistiques. Il peut être téléchargé gratuitement à l'adresse *http://www.r-project.org/*. L'un des grands atouts de R est sa communauté active dont les membres développent sans cesse des packages pour ajouter de nouvelles fonctionnalités au logiciel. Si vous prévoyez de créer un graphique statique et ne savez pas par où commencer, les archives de R constituent un excellent emplacement.

Créer une arborescence de rectangles

Illustrator ne propose pas d'outil permettant de créer directement une arborescence de rectangles. Vous pourrez cependant utiliser le package R intitulé Portofolio, conçu par Jeff Enos et David Kane. À l'origine, il était destiné à visualiser les portefeuilles d'action, d'où son nom, mais vous pouvez l'appliquer sans

peine à vos propres données. En guise d'exemple, nous allons nous intéresser au nombre de visites par page et au nombre de commentaires des 100 billets populaires sur le site FlowingData. Nous allons les classer par catégorie de billet, comme « conseils de visualisation » ou « conception des données ».

Comme toujours, la première étape consiste à charger les données dans R. Vous pouvez les charger directement depuis votre ordinateur ou pointer vers une URL. Dans cet exemple, optez pour la seconde solution, car les données sont déjà disponibles en ligne. Toutefois, si vous préférez la première solution pour gérer vos propres données, assurez-vous d'avoir placé le fichier de données dans votre répertoire de travail dans R. Il est possible de modifier le répertoire de travail via le menu Miscellaneous.

Le chargement d'un fichier CSV à partir d'une URL est simple. Il suffit de saisir la ligne de code suivante, utilisant la fonction `read.csv()` (figure 5-17).

```
posts <- read.csv("http://datasets.flowingdata.com/post-data.txt")
```

Figure 5-17 *Chargement d'un fichier CSV dans R*

Facile, non ? Nous avons chargé un fichier texte (au format CSV) à l'aide de `read.csv()` et nous avons stocké les valeurs correspondant au nombre de visites par page et au nombre de commentaires dans la variable `posts`. Comme mentionné dans le chapitre précédent, la fonction `read.csv()` présume que votre fichier de données est délimité par la virgule. Si la tabulation avait été employée à la place, vous auriez utilisé l'argument `sep` et défini la valeur sur `\t`. Pour charger les données à partir d'un répertoire local, la ligne de code précédente se présenterait de la façon suivante :

```
posts <- read.csv("post-data.txt")
```

Cette instruction suppose que vous avez modifié le répertoire de travail en conséquence. Pour obtenir plus d'options et des instructions sur le chargement des données avec la fonction `read.csv()`, tapez l'instruction suivante dans la console R :

```
?read.csv
```

Maintenant que les données sont stockées dans la variable `posts`, entrez la ligne suivante pour afficher les cinq premières lignes de données.

```
posts[1:5,]
```

Vous devez voir apparaître quatre colonnes qui correspondent au fichier CSV original, intitulées « id », « views », « comments » et « category ». Maintenant que les données sont chargées dans R, nous allons utiliser le package Portfolio. Essayez de le charger ainsi :

```
library(portfolio)
```

Vous obtenez une erreur ? Il est probable que vous ayez besoin d'installer le package avant de commencer :

```
install.packages("portfolio")
```

Réeesayez à présent de charger le package. Aucune erreur ne se produit, nous pouvons passer à l'étape suivante.

Le package Portfolio effectue le plus dur avec une fonction intitulée `map.market()`.

La fonction accepte plusieurs arguments, mais vous n'en utiliserez que cinq.

```
map.market(id=data$id, area=posts$views, group=posts$category,
     color=posts$comments, main="FlowingData Map")
```

L'argument `id` correspond à la colonne qui indique un point unique et vous demandez à R d'utiliser `views` pour déterminer les surfaces des rectangles de l'arborescence de rectangles, les catégories pour former les groupes et le nombre de commentaires d'un billet pour choisir la couleur. Enfin, saisissez « FlowingData Map » comme titre principal. Appuyez sur la touche Entrée du clavier pour obtenir une arborescence de rectangles, comme illustrée à la figure 5-18.

Il y a encore des améliorations à apporter, mais la base et la hiérarchie sont là, ce qui constitue la part la plus difficile. Tout comme vous l'avez spécifié, les rectangles (chacun d'entre eux représente un billet) sont dimensionnés en fonction du nombre de visites et triés par catégorie. Les nuances de vert les plus claires correspondent aux billets ayant reçu le plus de commentaires ;

les billets les plus visités ne sont pas nécessairement ceux qui ont le plus de commentaires.

Figure 5-18 *Arborescence de rectangles par défaut dans R*

Vous pouvez enregistrer l'image au format PDF dans R, puis ouvrir le fichier dans Illustrator. Pour améliorer votre graphique, utilisez toutes les options abordées précédemment afin de modifier les couleurs de trait et de remplissage, les polices, supprimer les éléments superflus et ajouter éventuellement des commentaires.

Pour ce graphique particulier, vous devez modifier l'échelle de la légende, comprise entre –90 et 90. Il n'y a aucune raison d'avoir une échelle négative, car il ne peut pas y avoir un nombre négatif de commentaires. Vous pouvez aussi corriger les libellés. Certains paraissent dissimulés dans les petits rectangles. Dimensionnez les libellés par popularité à l'aide de l'outil Sélection, en remplacement de l'échelle uniforme. Épaississez aussi les bordures des catégories pour qu'elles soient plus prononcées. Vous devriez obtenir un graphique similaire à celui de la figure 5-19.

Le graphique est bien plus lisible maintenant, ses libellés apparaissent clairement et son échelle de couleurs est bien plus logique. Vous pouvez aussi supprimer l'arrière-plan gris foncé, ce qui rendra le graphique plus net. Et, bien sûr, n'oubliez pas le titre et le texte d'introduction pour expliquer brièvement l'objet du graphique.

Vous pouvez aussi installer les packages R via l'interface utilisateur du logiciel. Pour cela, sélectionnez le menu Packages & Data>Package Installer. Cliquez sur Get List, puis recherchez le package qui vous intéresse. Double-cliquez sur son nom pour l'installer.

Figure 5-19 *Arborescence de rectangles corrigée dans Illustrator*

The New York Times se servait d'une arborescence de rectangles animée pour illustrer les variations de la Bourse pendant la crise financière. Pour voir l'animation, rendez-vous à l'adresse *http://nyti.ms/9JUkWL*.

Le package Portfolio accomplit la plus grande partie du travail de création du graphique. Il ne vous reste qu'à faire en sorte que vos propres données soient au bon format. Souvenez-vous, vous avez besoin de trois choses : un identifiant unique pour chaque ligne, une mesure pour dimensionner les rectangles et des catégories parentes. Le cas échéant, vous pouvez utiliser une quatrième mesure pour colorier les rectangles. Si besoin, consultez à nouveau le chapitre 2, « Manipulation des données » qui explique comment convertir les données au format requis.

Proportions dans le temps

Vous aurez fréquemment à traiter un ensemble de proportions dans le temps. Au lieu de résultats correspondant à un ensemble de questions à partir d'une seule session d'enquête, vous pourriez avoir des résultats provenant du même sondage exécuté tous les mois pendant une année. Vous n'êtes pas simplement

intéressé par les résultats des sondages individuels ; vous voulez également voir de quelle façon les visites ont progressé/baissé au fil du temps ou comment l'opinion a évolué au cours de l'année évaluée ?

Ce point ne s'applique pas qu'aux sondages, bien sûr. Il existe une multitude de distributions qui changent au fil du temps. Dans les exemples suivants, nous allons nous intéresser à la distribution des groupes d'âge aux États-Unis entre 1860 et 2005. Avec l'amélioration des soins et la réduction de la taille moyenne d'une famille, la population actuelle vit plus longtemps que la génération précédente.

Empilement continu

Imaginez que vous ayez plusieurs graphiques chronologiques. Placez chaque ligne au-dessus de l'autre, puis complétez l'espace vide. Vous obtenez un graphique en surfaces empilées (graphique en couches), où l'axe horizontal correspond au temps et l'axe vertical à une plage qui s'étend de 0 à 100 %, comme illustré à la figure 5-20.

Figure 5-20 *Graphique en surfaces empilées généralisé*

Si vous extrayez une tranche verticale du graphique en surfaces, vous obtenez la distribution correspondante. Une autre façon de la voir consiste à la considérer comme une série de graphiques en barres empilées reliés par le temps.

Créer un graphique en surfaces empilées

Dans cet exemple, nous allons nous intéresser à la population vieillissante aux États-Unis. Téléchargez les données à l'adresse *http://book.flowingdata.com/ch05/data/us-population-by-age.xls*. La médecine et la santé se sont améliorées au cours des dernières décennies et la durée de vie moyenne ne cesse de croître. En conséquence, le pourcentage de la population dans les tranches les plus âgées a augmenté. Comment évaluer cette distribution des âges au fil des ans ? Les données en provenance de l'U.S. Census Bureau peuvent vous aider via un graphique en surfaces empilées. Vous voulez voir de combien la proportion des groupes d'âge les plus âgés a augmenté et de combien celle des groupes d'âge les plus jeunes a diminué.

Vous pouvez procéder de différentes façons, nous vous présentons tout d'abord celle utilisant Illustrator. Pour créer un graphique en surfaces empilées dans Illustrator, vous allez vous servir de l'outil Graphe en couches (figure 5-21).

Figure 5-21 *L'outil Graphe en couche*

Ouvrez un nouveau document et cliquez n'importe où à l'aide de l'outil Graphe en couches. Effectuez un cliquer-glisser, puis entrez les données dans la feuille de calcul qui s'affiche. Le chargement des données, la génération du graphique et son amélioration vous sont désormais familiers, n'est-ce-pas ?

La figure 5-22 illustre un graphique en surfaces empilées, après que vous avez saisi les données.

La couche supérieure dépasse la ligne des 100 %. La raison en est que le graphique en surfaces empilées ne convient pas seulement aux proportions normalisées ou aux ensembles de valeurs dont le total est égal à 100 %. Il peut aussi être utilisé pour les valeurs brutes et, par conséquent, si vous voulez que la somme des tranches de temps atteigne 100 %, vous devez normaliser les données. Le graphique de l'image 5-22 a été obtenu suite à une erreur de ma part ; je n'ai pas saisi les données correctement. Une rapide correction et nous obtenons le graphique de la figure 5-23. Cependant, comme vous ne vous êtes sans doute pas trompé, vous avez sûrement le bon graphique.

Figure 5-22 *Graphique en surfaces empilées par défaut dans Illustrator*

Figure 5-23 *Graphique en surfaces empilées corrigé*

Faites attention à ce type d'anomalie lorsque vous créez des graphiques. Il est préférable de repérer les coquilles et les erreurs de saisie des données tout au début, plutôt que d'avoir à retrouver tout à la fin à quel endroit l'erreur a été commise.

Soyez vigilant lorsque vous saisissez les données manuellement. Un grand nombre d'erreurs se produisent lors du transfert des données d'une source à une autre.

Maintenant que le graphique dispose d'une base correcte, effacez les lignes et les axes. Utilisez l'outil Sélection directe pour sélectionner des éléments spécifiques. J'aime retirer la ligne de l'axe vertical et conserver des graduations plus fines, afin que le graphique offre un aspect plus net et moins encombré. J'ajoute également le symbole % car c'est bien de pourcentages dont il s'agit ici. Je remplace aussi généralement la couleur de remplissage noire (par défaut) des couches du graphique par du blanc. Introduisez aussi quelques nuances de bleu, vous obtenez alors la figure 5-24.

Il s'agit ici de mes préférences personnelles, libre à vous de choisir des couleurs différentes et d'effectuer d'autres types de modifications. La sélection des couleurs varie aussi selon les cas. Plus vous créerez de graphiques, plus vous saurez ce qui vous plaît le plus et fonctionne le mieux.

Manque-t-il quelque chose d'autre ? Effectivement, il n'y a pas de libellés pour l'axe horizontal. Nous allons les insérer et, dans le même temps, nommer les couches pour indiquer les groupes d'âge (figure 5-25).

Figure 5-24 *Modification des couleurs par rapport aux couleurs par défaut*

Figure 5-25 *Graphique en couches empilées avec libellés*

Utilisez des couleurs adaptées au thème et guidez les yeux du lecteur avec diverses nuances.

J'ai également ajouté une annotation à droite du graphique. Ce qui nous intéresse le plus ici est la modification survenue dans la distribution des âges. Nous pouvons l'observer à partir du graphique, mais les nombres réels peuvent aider à enfoncer le clou.

Pour finir, placez le titre et le texte d'introduction, ainsi que la source des données en bas à gauche du graphique. Vous pouvez également modifier légèrement les couleurs des annotations. Vous obtenez alors le graphique final tel que représenté à la figure 5-26.

Figure 5-26 *Graphique en couches empilées final*

Créer un graphique en couches empilées interactif

L'un des inconvénients de l'utilisation des graphiques en couches empilées est qu'ils finissent par être difficiles à lire et pratiquement inutiles si vous avez beaucoup de catégories et de points de données. Ce type de graphique a fonctionné pour la ventilation des âges, car il n'y avait que cinq catégories. Si vous en ajoutez d'autres, les couches vont peu à peu ressembler à de fines bandes et seront illisibles. De même, si une catégorie offre des nombres relativement petits, elle peut facilement se retrouver éclipsée par les catégories plus importantes. Toutefois, il existe une solution pour résoudre ce type de problème : rendre le graphique interactif.

Vous pouvez proposer au lecteur de rechercher une catégorie, puis ajuster l'axe pour effectuer un zoom avant sur les points intéressants. Grâce aux infobulles, le lecteur peut visualiser les valeurs aux endroits qui sont trop petits pour y insérer des libellés. En résumé, vous pouvez extraire les données qui ne fonctionneraient pas sous forme de graphique en couches empilées statique et les rendre plus faciles à parcourir et à explorer. Vous pouvez le faire en JavaScript avec D3, mais pour le plaisir de découvrir de nouveaux outils, nous allons utiliser Flash et ActionScript.

La visualisation en ligne est progressivement passée de Flash à JavaScript et HTML5, mais certains navigateurs ne prennent pas en charge ce dernier, notamment Internet Explorer 8. De même, comme Flash existe depuis des années, les bibliothèques et les packages qui lui sont associés rendent certaines tâches plus aisées que si vous aviez à les exécuter avec les fonctionnalités natives du navigateur.

Le *NameVoyager* de Martin Wattenberg a rendu célèbre le graphique en couches empilées interactif. Il permet d'afficher les noms des bébés au fil du temps et le graphique est automatiquement mis à jour lorsque vous saisissez des noms dans la zone de recherche. Découvrez-le à l'adresse suivante : *http://www.babynamewizard.com/voyager*.

Heureusement, vous n'avez pas à commencer à partir de zéro. La plus grande part du travail a déjà été effectuée pour vous via la boîte à outils de visualisation Flare, conçue et maintenue par l'UC Berkeley Visualization Lab. Il s'agit d'une bibliothèque ActionScript, qui faisait partie d'une autre boîte à outils de visualisation Java intitulée Prefuse. Nous nous appuierons sur l'un des exemples d'application du site Flare, JobVoyager, qui s'apparente à NameVoyager, mais qui permet de rechercher des emplois. Une fois votre environnement de développement configuré, il suffit de reporter vos données et de personnaliser la présentation.

Téléchargez gratuitement Flare à l'adresse suivante : *http://flare.prefuse.org/*.

Vous pouvez écrire le code totalement en ActionScript, puis le compiler dans un fichier Flash. Ceci implique que vous écriviez le code, lequel est un langage que vous comprenez, puis que vous utilisiez un compilateur pour traduire le code en bits de telle sorte que votre ordinateur, ou le lecteur Flash, puisse comprendre ce que vous lui demandez de faire. Autrement dit, deux choses vous sont nécessaires : un emplacement où écrire et une solution de compilation.

La solution à la dure consiste à écrire le code dans un éditeur de texte standard, puis à utiliser l'un des compilateurs gratuits d'Adobe. Je dis « à la dure », car les étapes sont à coup sûr plus alambiquées et vous devrez installer différentes choses sur votre ordinateur.

La solution la plus simple – et c'est celle que je vous recommande vivement si vous prévoyez de beaucoup utiliser Flash et ActionScript – est d'employer Flex Builder. Cet outil rend plus rapide la programmation fastidieuse avec ActionScript, parce que vous codez, compilez et déboguez en un seul et même emplacement. L'inconvénient est qu'il est payant (mais gratuit pour les étudiants).

Si vous n'êtes pas certain que la dépense en vaille la peine, téléchargez la version d'évaluation gratuite et prenez votre décision ultérieurement. Pour l'exemple du graphique en couches empilées, j'expliquerai les étapes que vous devez suivre dans Flex Builder.

> Au moment où j'écris ces lignes, Adobe a changé le nom Flex Builder en Flash Builder. Bien que similaires, les deux produits présentent néanmoins quelques différences. Même si les étapes suivantes s'appuient sur Flex Builder, elles peuvent tout à fait être réalisées avec Flash Builder.
>
> Pour télécharger Flash Builder, rendez-vous à l'adresse : *http://www.adobe.com/products/flashbuilder/*. Si vous êtes concerné, n'hésitez pas à profiter de la remise accordée aux étudiants, vous donnant droit à une licence gratuite. Pour cela, il vous suffit de fournir un justificatif mentionnant votre numéro d'étudiant. Une autre solution consiste à vous procurer à bas prix une ancienne version de Flex Builder.

Une fois Flex Builder téléchargé et installé sur votre ordinateur, lancez-le. La fenêtre de la figure 5-27 apparaît alors.

Figure 5-27 *À l'ouverture de Flex Builder, cette fenêtre apparaît.*

Cliquez avec le bouton droit de la souris sur le Flex Navigator (barre de gauche), puis sur Import. La fenêtre de la figure 5-28 s'affiche.

Sélectionnez Existing Projects into Workspace (Projets existants de l'espace de travail) et cliquez sur Next (Suivant). Accédez à l'emplacement où vous stockez les fichiers Flare. Sélectionnez le répertoire Flare, puis veillez à ce que la case située en regard de Flare soit cochée dans le champ Projects, comme illustré à la figure 5-29.

Figure 5-28 *Fenêtre d'importation dans Flex Builder*

Figure 5-29 *Fenêtre des projets existants*

Procédez de même avec le dossier `flare.apps`. La fenêtre principale de Flex Builder se présentera ensuite comme illustrée à la figure 5-30, après que vous avez développé le dossier `flare.apps/flare/apps` et cliqué sur `JobVoyager.as`.

Figure 5-30 *Code de JobVoyager*

Si, maintenant, vous cliquez sur le bouton d'exécution situé à droite (bouton vert avec le triangle blanc de lecture en haut à gauche), vous devriez voir JobVoyager à l'œuvre, comme illustré à la figure 5-31. Vous en avez fini avec la partie la plus difficile : l'installation. Il vous reste maintenant à relier vos propres données et à les personnaliser comme bon vous semble. Cela ne vous rappelle rien ?

La figure 5-32 illustre le résultat attendu : un navigateur à travers les dépenses de consommation de 1984 à 2008, telles que recensées par l'U.S. Census Bureau. L'axe horizontal correspond aux années, et les catégories sont ici liées aux dépenses, telles que la maison et l'alimentation.

Nous allons à présent modifier la source de données, comme spécifiée à la ligne 57 du fichier `JobVoyager.as`.

```
private var _url:String = "http://flare.prefuse.org/data/jobs.txt";
```

Figure 5-31 *Application JobVoyager*

Le graphique final est disponible à l'adresse suivante : *http://datafl.ws/16r* ; allez-y pour tester la visualisation finale et voir comment le navigateur procède avec les dépenses de consommation.

Modifiez `_url` pour pointer vers les données disponibles à l'adresse *http://datasets.flowingdata.com/expenditures.txt*. De même que pour le fichier `jobs.txt` de l'exemple précédent, les données se présentent sous forme de valeurs séparées par une tabulation. La première colonne correspond à l'année, la deuxième à la catégorie et la troisième aux dépenses.

```
private var _url:String =
    "http://datasets.flowingdata.com/expenditures.txt";
```

Au lieu de données sur les emplois de l'exemple précédent, le fichier propose ici les données relatives aux dépenses. Rien que de très difficile jusque-là.

Les deux lignes suivantes, les lignes 58 et 59, correspondent aux noms de colonne ou, dans le cas présent, les années pour lesquelles les données sur les emplois étaient disponibles. Elles sont classées par décennie, de 1850 à 2000. Vous pourriez créer un graphique plus élaboré en recherchant les années dans les données chargées, mais comme celles-ci ne varient pas, vous pouvez gagner du temps et spécifier explicitement les années.

Figure 5-32 *Navigateur interactif pour les dépenses de consommation*

Comme les données sur les dépenses sont disponibles année après année de 1984 à 2008, modifiez les lignes 58 à 59 en conséquence.

```
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];
```

Modifiez ensuite les références aux en-têtes de données. Le fichier original de données (`jobs.txt`) possède quatre colonnes : l'année, le poste, la personne et le sexe. Les données de consommation n'ont que trois colonnes : l'année, la catégorie et les dépenses. Vous devez adapter le code à cette nouvelle structure de données.

Ceci ne présente aucune difficulté. La colonne de l'année ne change pas ; vous n'avez donc qu'à modifier les références aux personnes associées aux dépenses (axe vertical) et les références aux postes associées aux catégories (les

couches). Il vous faut également supprimer toutes les utilisations de la quatrième colonne.

À la ligne 74, les données sont refaçonnées et préparées en vue d'être utilisées dans le graphique en couches empilées. Cette ligne de code spécifie que les catégories (autrement dit, les couches) sont représentées par le poste et le sexe, tandis que l'axe des *x* correspond aux années et l'axe des *y* aux personnes.

```
var dr:Array = reshape(ds.nodes.data, ["occupation","sex"],
    "year", "people", _cols);
```

Modifions la ligne de la façon suivante :

```
var dr:Array = reshape(ds.nodes.data, ["category"],
    "year", "expenditure", _cols);
```

Vous n'avez plus qu'une seule catégorie, intitulée `category`. L'axe des *x* demeure celui des années et l'axe des *y* est celui des dépenses.

La ligne 84 trie les données par poste (alphabétiquement), puis par genre (numériquement). Trions à notre tour par catégorie :

```
data.nodes.sortBy("data.category");
```

Commencez-vous à bien comprendre ? La plus grande part du travail est effectuée. Vous avez simplement à ajuster les variables pour les adapter aux données.

La visualisation est l'objet de nombreux développements open source et, même si l'écriture du code peut être déroutante au début, vous pouvez très souvent utiliser le code existant avec vos propres données en modifiant uniquement les variables. Le seul défi est de savoir lire le code et de découvrir comment il fonctionne.

La ligne 92 colore les couches en fonction du sexe, mais comme vous n'avez pas une telle répartition dans vos données, vous pouvez supprimer toute la ligne :

```
data.nodes.setProperty("fillHue", iff(eq("data.sex",1), 0.7, 0));
```

Nous reviendrons un peu plus tard sur la personnalisation des couleurs des couches. La ligne 103 ajoute les libellés basés sur le poste :

```
_vis.operators.add(new StackedAreaLabeler("data.occupation"));
```

Comme vous voulez que les libellés soient liés aux catégories de dépense, modifiez la ligne en conséquence :

```
_vis.operators.add(new StackedAreaLabeler("data.category"));
```

Les lignes 213 à 231 gèrent les filtres de JobVoyager : d'abord le filtre masculin/féminin, puis le filtre par poste. Comme vous n'avez nul besoin du premier filtre, supprimez les lignes 215 à 218 et transformez la ligne 219 en instruction `if` ordinaire.

De même, les lignes 264 à 293 créent les boutons qui permettent de déclencher le filtre masculin/féminin. Nous pouvons également les supprimer.

Vous avez presque fini de personnaliser le navigateur pour l'adapter aux données sur les dépenses. Retournez à la fonction `filter()` de la ligne 213. Mettez à jour la fonction de façon à pouvoir trier sur la catégorie de dépense au lieu de l'emploi occupé.

La ligne 22 initiale se présente ainsi :

```
var s:String = String(d.data["occupation"]).toLowerCase();
```

Remplacez `occupation` par `category` :

```
var s:String = String(d.data["category"]).toLowerCase();
```

L'étape suivante de la personnalisation consiste à modifier les couleurs. Si vous compiliez le code à ce stade et que vous l'exécutiez, vous obtiendriez un graphique en couches empilées en nuances de rouge, comme illustré à la figure 5-33. Cependant, vous souhaitez un contraste plus marqué.

La couleur est spécifiée à deux emplacements. D'abord, les lignes 86 à 89 indiquent la couleur du pinceau et colorient la totalité en rouge :

```
shape: Shapes.POLYGON,
lineColor: 0,
fillValue: 1,
fillSaturation: 0.5
```

Ensuite, la ligne 105 met à jour la saturation (niveau de rouge). Le code `SaturationEncoder()` se trouve aux lignes 360 à 383. Vous n'utiliserez pas la saturation ; en lieu et place, spécifiez explicitement le jeu de couleurs.

Commencez par mettre à jour les lignes 86 à 89 :

```
shape: Shapes.POLYGON,
lineColor: 0xFFFFFFFF
```

Maintenant, spécifiez une couleur de trait blanche avec `lineColor`. S'il y avait d'autres catégories de dépense, vous ne procèderiez probablement pas ainsi, car le graphique serait alors surchargé. Comme vous n'avez pas beaucoup de catégories, la lecture sera ainsi plus aisée.

Créez ensuite un tableau des couleurs que vous voulez utiliser, classées par niveaux. Insérez-le aux alentours de la ligne 50 :

```
private var _reds:Array = [0xFFFEF0D9, 0xFFFDD49E, 0xFFFDBB84,
➥0xFFFC8D59, 0xFFE34A33, 0xFFB30000];
```

Figure 5-33 *Graphe en couches empilées avec couleurs de base*

Pour ces couleurs, j'ai utilisé ColorBrewer, qui suggère des jeux de couleurs basés sur les critères que vous avez définis. L'outil est destiné à choisir les couleurs des cartes, mais il fonctionne aussi parfaitement pour les applications de visualisation.

À présent, ajoutez une nouvelle fonction `ColorEncoder` vers la ligne 110 :

```
var colorPalette:ColorPalette = new ColorPalette(_reds);
vis.operators.add(new ColorEncoder("data.max", "nodes",
     "fillColor", null, colorPalette));
```

Si vous obtenez une erreur lors de la compilation du code, vérifiez la présence en haut du fichier `JobVoyager.as` des deux lignes de code suivantes. Elles permettent d'importer les objets `ColorPallete` et `Encoder`. Si elles ne figurent pas en haut du fichier, ajoutez-les.

```
import flare.util.palette.*;
import flare.vis.operator.encoder.*;
```

Vous devriez maintenant avoir quelque chose qui ressemble à ce que nous recherchions (figure 5-32). Bien sûr, vous n'avez pas à vous arrêter là. Vous pouvez encore faire beaucoup de choses, comme appliquer ce que nous venons de faire à vos propres données, utiliser un autre jeu de couleurs ou accentuer la personnalisation pour répondre à vos besoins. Vous pouvez également modifier la police ou la mise en forme de l'infobulle, ajouter davantage de modifications en recourant à d'autres outils ou insérer d'autres instructions ActionScript, et ainsi de suite.

Point par point

L'un des inconvénients du graphique en couches empilées est qu'il peut être difficile d'y dégager des tendances pour chaque groupe, car l'emplacement de chaque point est affecté par les points qui sont situés en dessous. Aussi, il est parfois préférable de tracer les proportions sous la forme d'une série de données temporelles, comme expliqué au chapitre précédent.

Heureusement, vous pouvez passer sans peine d'une solution à l'autre dans Illustrator. Les données en entrée restant les mêmes, il suffit de changer de type de graphique. Sélectionnez le graphique linéaire au lieu du graphe en couches, vous obtiendrez le résultat illustré à la figure 5-34.

Figure 5-34 *Graphique linéaire par défaut*

Procédez à la mise en forme souhaitée, de la même façon qu'avec les exemples précédents sur les séries de données temporelles. Vous obtiendrez les mêmes données, mais vues sous un angle différent (figure 5-35).

Grâce au tracé chronologique, il est plus facile de distinguer les tendances de chaque groupe d'âges. En revanche, vous perdez bel et bien la notion de totalité et de distributions. Le graphique que vous choisissez doit refléter le point de vue que vous souhaitez transmettre ou les éléments que vous souhaitez mettre en valeur dans les données. Vous pouvez même afficher les deux vues si vous disposez de l'espace nécessaire.

Figure 5-35 *Graphique linéaire avec libellés*

Pour résumer

Le principal point qui distingue les proportions des autres types de données est qu'elles représentent les parties d'un tout. Chaque valeur individuelle a une signification particulière, tout comme la somme de toutes les parties ou d'un seul sous-ensemble. La visualisation que vous créez doit illustrer ces idées.

Vous n'avez que quelques valeurs ? Le graphique en camembert est peut-être votre meilleur allié. N'utilisez les graphiques en anneau qu'avec précaution. Si vous avez plusieurs valeurs et plusieurs catégories, envisagez de créer un graphique en couches empilées plutôt qu'un graphique en barres empilées. Si vous recherchez des modèles au fil du temps, envisagez le graphique en couches empilées ou les séries de données temporelles traditionnelles. Grâce à ces principes de base, vos proportions seront parfaitement rendues.

Quand vient le moment de la conception et de l'implémentation, demandez-vous ce que vous souhaitez savoir sur vos données, et commencez à partir de là. Un graphique statique permet-il de traduire intégralement votre histoire ? Très souvent, la réponse sera oui. Cependant, si vous décidez de recourir à un graphique interactif, tracez sur papier ce qui doit se passer (et ce qui ne doit pas se passer) quand vous cliquez sur tel ou tel objet. Si vous ajoutez un trop grand nombre de fonctionnalités, votre graphique risque de devenir rapidement compliqué ; aussi, restez aussi simple que possible. Demandez à différentes personnes d'interagir avec les graphiques pour vérifier qu'elles comprennent bien ce qui se passe.

Enfin, lorsque vous programmez, surtout si vous êtes débutant, vous atteindrez inévitablement un point où vous ne serez pas sûr de ce que vous devez faire ensuite. Cela m'arrive en permanence. Lorsque vous êtes coincé, rien ne vaut le Web pour trouver une solution. Consultez la documentation si elle est disponible ou étudiez les exemples similaires à ce que vous essayez de faire. Ne regardez pas seulement la syntaxe. Apprenez aussi la logique, car c'est ce qui vous aidera le plus. Heureusement, il existe des bibliothèques telles que D3 et Flare qui proposent de nombreux exemples et une excellente documentation.

Dans le chapitre suivant, nous allons passer à une analyse et une interprétation plus approfondies des données et nous retrouverons notre bon ami statistique. Vous avez fait un bon usage de R lorsque vous avez étudié les relations entre les ensembles de données et les variables.

Chapitre 6

Visualisation des relations

Les statistiques cherchent à déterminer les relations qui unissent les données. Quelles sont les similarités entre groupes ? Au sein des groupes ? Au sein des sous-groupes ? La relation qui nous est la plus familière en matière de statistiques est la corrélation. Par exemple, si la hauteur moyenne de la population augmente, il est fort probable que le poids moyen augmente lui aussi. Il s'agit d'une simple corrélation positive. Les relations entre les données, tout comme dans la vie réelle, deviennent beaucoup plus complexes si vous prenez en compte un plus grand nombre de facteurs ou envisagez des modèles qui ne sont pas aussi linéaires. Ce chapitre traite de l'utilisation de la visualisation pour rechercher de telles relations et les mettre en évidence dans l'histoire racontée par vos graphiques.

Au fur et à mesure que nous aborderons des graphiques statistiques plus complexes, nous utiliserons plus intensivement l'application R, que ce soit dans ce chapitre ou le suivant. C'est là qu'excellent les logiciels open source. Comme nous l'avons vu dans les chapitres précédents, R accomplit le travail le plus difficile. Il vous reste ensuite à personnaliser vos graphiques afin de les rendre plus lisibles, ce pour quoi vous utiliserez Illustrator.

Quelles relations rechercher ?

Jusqu'à présent, nous nous sommes intéressés à des relations élémentaires constituées de modèles temporels et de proportions. Vous avez observé les tendances temporelles et vous avez comparé proprortions et pourcentages pour déterminer les valeurs maximales, minimales et intermédiaires. L'étape suivante consiste à rechercher les relations entre les différentes variables. Si quelque chose augmente, une autre chose diminue-t-elle ? S'agit-il d'une relation causale ou d'une corrélation ? Le premier cas est généralement assez difficile à prouver quantativement et de ce fait, il est peu probable que vous parveniez à l'illustrer d'un graphique. Cependant, vous pouvez montrer aisément une corrélation, et aboutir ainsi à une exploration plus approfondie.

Vous pouvez aussi prendre un peu de recul et examiner la vue d'ensemble ou, plus précisément, la distribution de vos données. Sont-elles espacées ou regroupées ? De telles comparaisons peuvent conduire à des histoires sur les citoyens d'un pays ou à la façon de comparer ceux qui vous entourent. Vous pouvez voir de quelle façon différents pays peuvent être comparés entre eux ou comment le développement général progresse à travers le monde, ce qui peut contribuer à la prise de décisions sur les pays qui ont besoin d'une aide particulière.

Il est également possible de comparer plusieurs distributions afin d'avoir une vue plus élargie des données. De quelle façon la composition d'une population s'est-elle modifiée au fil du temps ? Dans quelle mesure est-elle restée identique ?

Le plus important, finalement, est que vous vous demandiez, une fois que vous avez tous les graphiques devant vous, ce que les résultats signifient. Sont-ils conformes à ce que vous attendiez ? Quelque chose vous surprend-il ? Comme cela peut paraître abstrait et quelque peu magique, passons à des exemples plus concrets sur la façon de regarder les données.

Corrélation

La corrélation est probablement ce à quoi vous pensez quand est évoquée une relation entre données. La deuxième chose est probablement la causalité. Les deux notions ne sont pas équivalentes. La corrélation signifie qu'une chose tend à changer d'une certaine façon lorsqu'une autre chose change également. Par exemple, le prix du litre de lait et le prix du litre d'essence sont positivement corrélés. Les deux ont tendance à augmenter au fil des ans. Mais si le prix de l'essence augmente, le prix du lait augmentera-t-il par défaut ? Plus important, si le prix du lait a augmenté, était-ce en raison de l'augmentation du prix de l'essence ou d'un facteur externe, tel qu'une grève des producteurs laitiers ? Ainsi corrélation et causalité sont deux choses bien différentes.

Il est difficile de se représenter chaque facteur extérieur, ou facteur de confusion, et de ce fait, de prouver la causalité. Les chercheurs passent des années à essayer de trouver de telles causalités. En revanche, vous pouvez facilement rechercher et trouver une corrélation, ce qui peut toujours être utile, comme vous le verrez dans les prochaines sections.

La corrélation peut vous aider à prédire une mesure en en connaissant une autre. Pour voir cette relation, revenez au nuage de points et aux multiples nuages de points.

Plus de points

Dans le chapitre 4, vous avez utilisé un nuage de points comme graphique de mesures au fil du temps, où l'axe horizontal correspondait au temps et l'axe vertical aux mesures. Il était ainsi possible de repérer les modifications (ou les

non-modifications) intervenues au cours du temps. La relation existait entre le temps et un autre facteur, ou une autre variable. Cependant, comme illustré à la figure 6-1, vous pouvez employer le nuage de points pour des variables autres que le temps ; il peut servir, par exemple, à rechercher les relations entre deux variables.

Si deux mesures sont corrélées positivement (figure 6-2, à gauche), les points se déplacent vers le haut lorsque vous lisez le graphique de gauche à droite. Inversement, s'il existe une corrélation négative, les points se déplacent vers le bas, de gauche à droite (figure 6-2, au centre).

Parfois, la relation est simple, à l'image de la corrélation entre la hauteur et le poids des individus. Généralement, quand la hauteur croît, le poids augmente aussi. D'autres fois, la corrélation n'est pas aussi manifeste, comme celle entre la santé et l'IMC (indice de masse corporelle). Un IMC élevé est signe, généralement, d'un surpoids. Cependant, les athlètes peuvent par exemple avoir un IMC élevé et ne pas être en surpoids. Qu'en serait-il si les échantillons de population étudiés étaient des culturistes ou des rugbymen ? À quoi ressemblerait alors la relation entre santé et IMC ?

Figure 6-1 *Comparaison de deux variables avec un nuage de points*

Figure 6-2 *Corrélations illustrées par des nuages de points*

N'oubliez pas que le graphique n'est qu'une partie de l'histoire. C'est à vous qu'il appartient d'interpréter les résultats. Ceci est particulièrement important dans le cas des relations. Vous pourriez être tenté de présumer une relation de cause à effet, mais la plupart du temps ce n'est pas le cas. Le seul fait que le litre d'essence et la population mondiale aient tous deux augmenté au fil des ans ne signifie pas qu'il faille baisser le prix de l'essence pour ralentir la croissance de la population.

Créer un nuage de points

Dans cet exemple, nous nous intéressons à la criminalité aux États-Unis en 2005, au niveau de chaque État. Les chiffres dont nous disposons correspondent au nombre de crimes (meurtres, vols et agressions à main armée) par État pour une poulation de 100 000 personnes, et tels que rapportés par le Census Bureau. Il existe sept types de crimes en tout. Commençons par deux d'entre eux : les cambriolages et les meurtres. Quelle relation existe-t-il entre eux ? Est-ce que les États avec des taux de meurtres relativement élevés présentent aussi des taux de cambriolage élevés ? Tournons-nous vers R pour mener l'enquête.

Comme toujours, la première chose à faire est de charger les données dans R avec `read.csv()`. Le fichier CSV peut être téléchargé à l'adresse *http://datasets.flowingdata.com/crimeRatesByState2005.csv*, mais chargez-le directement dans R via l'URL.

```
# Charger les données
crime <-
  read.csv('http://datasets.flowingdata.com/crimeRatesByState2005.
       csv',
  sep=",", header=TRUE)
```

Consultez les premières lignes des données en saisissant la variable `crime` suivie des lignes que vous souhaitez voir.

```
crime[1:3,]
```

Les trois premières lignes se présentent comme suit :

```
     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   295753151
2       2650.0        288.3    4545049
3       2599.1        391.0    669488
```

La première colonne affiche le nom de l'État, tandis que les autres colonnes correspondent aux taux relevés pour les différents crimes. Par exemple, le taux de vol moyen pour les États-Unis en 2005 était de 140,7 pour 100 000 personnes. Avec `plot()`, créez le nuage de points par défaut du nombre de meurtres comparé au nombre de vols, comme illustré à la figure 6-3.

```
plot(crime$murder, crime$burglary)
```


Figure 6-3 *Nuage de points par défaut du nombre de meurtres par rapport au nombre de cambriolages*

Il semble qu'il existe ici une corrélation positive. Les États ayant des taux de meurtre plus élevés tendent à avoir des taux de vol plus importants, mais la relation n'est pas facile à voir en raison du point isolé situé à l'extrémité droite du graphique. Cette observation aberrante nécessite que l'axe horizontal soit beaucoup plus étendu. Le point isolé correspond de fait à l'État de Washington, DC, qui présente un taux de meurtre beaucoup plus élevé de 35,4. Les États qui arrivent en deuxième et troisième places quant au nombre de meurtres sont la Louisiane et le Maryland, avec 9,9.

Pour obtenir un graphique plus clair et plus utile, retirez Washington (DC) et, par la même occasion, les moyennes des États-Unis, puis mettez entièrement l'accent sur chaque État individuellement.

```
crime2 <- crime[crime$state != "District of Columbia",]
crime2 <- crime2[crime2$state != "United States",]
```

La première ligne stocke toutes les rangées qui ne concernent pas le `District of Columbia` dans `crime2`. De même, la deuxième ligne filtre les moyennes des États-Unis.

Lorsque vous générez le graphique, vous obtenez à présent un résultat plus clair (figure 6-4).

```
plot(crime2$murder, crime2$burglary)
```

Figure 6-4 *Nuage de points après filtrage des données*

Cependant, le graphique aurait meilleure allure si les axes démarraient à zéro : l'axe des `x` s'étend de 0 à 10 et l'axe des `y` de 0 à 1 200. Les points se trouvent ainsi décalés vers le haut et vers la droite, comme illustré à la figure 6-5.

```
plot(crime2$murder, crime2$burglary, xlim=c(0,10), ylim=c(0, 1200))
```

Savez-vous ce qui pourrait rendre ce tracé encore plus utile ? Une courbe LOESS telle que nous l'avons vue au chapitre 4. Elle aiderait à voir la relation entre le nombre de cambriolages et le nombre de meurtres. Utilisez à cette fin `scatter.smooth()`. Le résultat est illustré à la figure 6-6.

```
scatter.smooth(crime2$murder, crime2$burglary,
    xlim=c(0,10), ylim=c(0, 1200))
```

Figure 6-5 *Nuage de points avec axes commençant à zéro*

Pour des raisons de simplicité, nous avons retiré Washington, DC du jeu de données afin de mieux visualiser les données. Toutefois, il est important de tenir compte de l'importance des observations aberrantes dans vos données (voir chapitre 7, « Identification des différences »).

Figure 6-6 *Utilisation d'une courbe pour estimer la relation*

Le résultat est acceptable pour un graphique de base ; s'il était simplement destiné à l'analyse, vous pourriez vous arrêter là. Cependant, il est possible d'en améliorer grandement la lisibité avec quelques modifications simples, comme vous le voyez à la figure 6-7.

Je retire d'abord la bordure et dirige plutôt l'attention du lecteur sur la courbe en la rendant plus épaisse et plus sombre que les points.

Figure 6-7 *Graphique revu et corrigé*

Exploration de variables supplémentaires

Maintenant que vous avez tracé deux variables l'une par rapport à l'autre, la prochaine étape consiste à comparer d'autres variables. Vous pouvez choisir celles que vous voulez comparer et créer un nuage de points pour chaque paire de variables. Cependant, cela peut facilement conduire à manquer des opportunités et à ignorer des points intéressants dans les données. Tel n'est pas votre souhait. Par conséquent, vous pouvez à la place tracer toutes les paires possibles sous forme de matrice de nuages, (figure 6-8).

Figure 6-8 *Structure d'une matrice de nuages de points*

Cette méthode est particulièrement utile pendant les phases d'exploration des données. Vous pourriez avoir un ensemble de données sous les yeux, sans savoir ce dont il s'agit vraiment ou par où commencer. Si vous-même ne savez pas sur quoi portent les données, vos lecteurs ne le sauront pas non plus.

Pour raconter une histoire complète, vous devez comprendre vos données. Plus vous en savez sur elles, meilleure sera votre histoire.

La matrice de nuages de points se lit comme vous l'imaginez. Il s'agit généralement d'une grille carrée avec l'ensemble des variables disposées verticalement et horizontalement. Chaque colonne représente une variable sur l'axe horizontal

et chaque ligne représente une variable sur l'axe vertical. Vous obtenez ainsi toutes les paires possibles, tandis que la diagonale est réservée aux libellés, car il n'y aucun sens à comparer une variable à elle-même.

Créer une matrice de nuages de points

À present, revenez aux données sur les crimes. Vous disposez de sept variables, ou taux pour les types de crimes. Dans l'exemple précédent, vous n'en avez comparé que deux : les meurtres et les cambriolages. Avec une matrice de nuages de points, vous pouvez comparer tous les types de crimes (figure 6-9).

Figure 6-9 *Matrice de nuages de points des taux de criminalité*

Comme vous pouvez l'imaginer, il existe un grand nombre de corrélations positives. Par exemple, la corrélation entre les cambriolages et les agressions à main armée semble relativement élevée. Quand les premiers augmentent, le nombre

d'agressions tend aussi à croître, et inversement. Cependant, la relation entre meurtre et vol simple n'est pas aussi évidente. Vous ne devriez faire aucune hypothèse, mais il doit être facile de voir en quoi la matrice de nuages de points peut être utile. À première vue, elle peut paraître déroutante avec toutes les lignes et tous les tracés, mais si vous la lisez de gauche à droite et de haut en bas, vous pouvez en retirer beaucoup d'informations.

Le logiciel R permet de créer facilement des matrices de nuages de points, aussi simplement que des nuages de points. Utilisez la fonction `plot()`, mais au lieu de transmettre deux colonnes, passez la totalité du cadre des données, à l'exception de la première colonne, car il s'agit uniquement du nom des États.

```
plot(crime2[,2:9])
```

L'argument `panel` de `pairs()` accepte une fonction de *x* et de *y*. Dans cet exemple, vous utilisez `panel.smooth()`, qui est une fonction native de R et dont le résultat est une courbe LOESS. Vous pouvez aussi spécifier votre propre fonction.

Vous obtenez ainsi une matrice telle que celle représentée à la figure 6-10, qui correspond pratiquement à ce que vous voulez. Il faut encore ajouter les courbes ajustées afin de mieux distinguer les relations.

Pour créer une matrice de nuages de points avec des courbes LOESS ajustées, vous pouvez utiliser la fonction `pairs()`. Le résultat est illustré à la figure 6-11.

```
pairs(crime2[,2:9], panel=panel.smooth)
```

Vous disposez maintenant d'un excellent cadre de travail, que vous pourrez modifier ultérieurement pour le rendre plus lisible (figure 6-9). Enregistrez le graphique au format PDF et ouvrez le fichier dans Illustrator.

Pour l'essentiel, vous devez éviter tout encombrement et permettre de voir facilement ce qui est le plus important. Mettez l'accent sur les types de crimes et les courbes de tendance, puis sur les points et enfin sur les axes. Cet ordre d'importance doit se refléter dans le choix des couleurs et des tailles. Les tailles des libellés de la diagonale augmentent et les cases sont grisées. Les courbes de tendance sont plus épaisses afin de fournir un meilleur contraste entre les courbes elles-mêmes et les points. Pour finir, les bordures et les lignes du quadrillage sont rendues presque imperceptibles grâce à une largeur de trait plus fine et un gris plus clair. Comparez les figures 6-9 et 6-11 : la première offre une moins grande vibration visuelle, n'est-ce pas ?

Décidez de la partie de l'histoire que vous voulez raconter et concevez le graphique de façon à accentuer ces zones, tout en prenant soin de ne pas embrouiller les faits.

Figure 6-10 *Matrice de nuages de points par défaut dans R*

Figure 6-11 *Matrice de nuages de points avec courbes LOESS*

Bulles

Hans Rosling est professeur de santé internationale au Karolinska Institute et président de la fondation Gapminder. Pour illustrer son propos sur la richesse et la santé des nations, il a utilisé un graphique en bulles animé, ce qui a suscité un véritable engouement pour les bulles proportionnelles sur les axes des *x* et des *y*. Même si l'animation du graphique a pour but d'illustrer les modifications au fil du temps, vous pouvez également créer une version statique : le graphique en bulles.

Les célèbres conférences de Hans Rosling sont disponibles sur le site de la fondation Gapminder, à l'adresse suivante : *http://www.gapminder.org*. Vous y trouverez notamment un documentaire de la BBC sur les statistiques.

Un graphique en bulles peut aussi être composé de bulles directement dimensionnées de façon proportionnelle. Il s'apparente au nuage de points, mais diffère de celui-ci par la présence d'une troisième dimension… extrêmement pétillante !

Figure 6-12 *Structure du graphique en bulles*

L'avantage de ce type de graphique est qu'il permet de comparer trois variables à la fois, comme illustré à la figure 6-12. La première variable se trouve sur l'axe des *x*, la deuxième sur l'axe des *y* et la troisième est représentée par l'aire des bulles.

Notez bien le dernier point, car souvent, les bulles ne sont pas correctement dimensionnées. Comme nous l'avons évoqué au premier chapitre, la taille des bulles doit être fonction de la surface, et non du rayon, du diamètre ou de la circonférence. Dans le cas contraire, vous vous retrouverez immanquablement avec des cercles trop grands ou trop petits.

Illustrons ce point par un exemple. Imaginez que vous ayez en charge les ventes publicitaires de votre entreprise et que vous testiez deux bandeaux promotionnels sur un site. Vous souhaitez savoir quelle publicité obtient les meilleurs résultats. Sur un mois, les deux publicités s'exécutent le même nombre de fois ; cependant, le nombre de clics n'est pas le même. Il y a eu 150 clics sur la première bannière et 100 clics sur la seconde. Autrement dit, la première bannière obtient

des résultats 50 % meilleurs que la seconde. La figure 6-13 représente un cercle, dimensionné selon la surface, pour chaque bannière. Le cercle de la première bannière est 50 % plus grand que celui de la seconde bannière.

Figure 6-13 *Bulles dimensionnées selon la surface*

À la figure 6-14, les deux bulles sont comparées non pas en fonction de leur surface, mais en fonction de leur rayon.

Figure 6-14 *Bulles dimensionnées selon le rayon*

Le rayon du premier cercle (première bannière) est 50 % plus grand que celui du second (seconde bannière). De ce fait, la surface du premier cercle correspond au double de celle du second cercle. Même si cela ne semble pas poser un réel problème dans le cas de deux points de données seulement, le défi est beaucoup plus réel quand vous utilisez plusieurs données.

Créer un graphique en bulles

Le graphique final que vous souhaitez obtenir est représenté à la figure 6-15. Les données utilisées sont les mêmes que précédemment, elles concernent les taux de meurtre et les taux de cambriolage par État. La seule différence ici est que la population est ajoutée comme troisième dimension. Est-ce que les États avec le plus grand nombre d'habitants ont des taux de criminalité plus élevés ? Ce n'est pas si simple (comme c'est généralement le cas). Les grands États comme la Californie, la Floride et le Texas se situent dans le quart supérieur droit, alors que New York et la Pennsylvanie ont des taux de cambriolage relativement bas. De même, la Louisiane et le Maryland, qui ont des populations plus réduites se trouvent à l'extrémité droite.

Figure 6-15 *Graphique en bulles illustrant la criminalité aux États-Unis*

Pour commencer, chargez les données dans R avec `read.csv()`. Ce sont les mêmes données que celles que vous venez d'utiliser, si ce n'est que nous avons ajouté une colonne pour la population et que l'État de Washington, DC a été supprimé. Les valeurs sont séparées par des tabulations et non par des virgules. Rien de très compliqué. Modifiez simplement l'argument `sep` de la fonction.

```
crime <-
  read.csv("http://datasets.flowingdata.com/crimeRatesByState2005.
                tsv",header=TRUE, sep="\t")
```

Dessinons ensuite quelques bulles à l'aide de la fonction `symbols()`. L'axe des *x* correspond au taux de meurtre, l'axe des *y* au taux de cambriolage et le rayon

des bulles est déterminé proportionnellement à la population. La figure 6-16 illustre le résultat. Nous verrons bientôt comment utiliser autrement la fonction `symbols()`.

```
symbols(crime$murder, crime$burglary, circles=crime$population)
```


Figure 6-16 *Graphique en bulles par défaut*

Vous croyez avoir fini ? Eh bien, non. C'était un simple test. Dans l'exemple précédent, la taille des cercles est déterminée de telle sorte que la population soit proportionelle au rayon. Vous devez dimensionner les cercles proportionnellement à la surface. Les proportions relatives sont totalement erronées si vous déterminez la taille des cercles en fonction du rayon. La population de Californie, représentée par le grand cercle au milieu, est-elle à ce point plus élevée que celle de tous les autres États américains ?

Pour dimensionner correctement les rayons, revenons à l'équation permettant de calculer l'aire d'un cercle.

$$\text{Aire du cercle} = \pi r^2$$

L'aire de chaque bulle représente la population. Ce que vous devez trouver est comment dimensionner le rayon. À cette fin, isolez le rayon et constatez qu'il est proportionnel à la racine carrée de l'aire.

$$r = \text{(aire du cercle / } \pi\text{)}$$

En réalité, vous pouvez vous passer de π, car il s'agit d'une constante, mais conservez-le pour des raisons de clarté. À présent, au lieu d'utiliser `crime$population` pour dimensionner les rayons des cercles, recherchez la racine carrée et passez-la à `symbols()`.

```
radius <- sqrt(crime$population/pi)
symbols(crime$murder, crime$burglary, circles=radius)
```

La première ligne de code crée simplement un nouveau vecteur de valeurs racine carrée stockées dans `radius`. La figure 6-17 illustre le tracé en bulles avec les rayons correctement dimensionnés, mais ça n'est pas très clair car les États ayant une population moins importante que celle de la Californie apparaissent désormais plus grands.

Figure 6-17 *Graphique en bulles par défaut avec les cercles correctement dimensionnés*

Vous devez réduire tous les cercles afin de voir ce qui se passe. L'argument `inches` de `symbols()` définit la taille en pouces du plus grand cercle. La valeur est de 1 pouce par défaut. Sur la figure 6-17, la taille de la bulle correspondant à la Californie est de 1 pouce et tous les autres cercles sont mis à l'échelle en conséquence. Il est possible de réduire le maximum jusqu'à 0,35 pouces, tout en conservant les bonnes proportions. Vous pouvez aussi modifier la couleur avec `fg` et `bg` afin de changer respectivement la couleur du trait et la couleur du remplissage. Vous pouvez aussi ajouter vos propres libellés aux axes. La figure 6-18 illustre le résultat obtenu.

```
symbols(crime$murder, crime$burglary, circles=radius, inches=0.35,
   fg="white", bg="red", xlab="Murder Rate", ylab="Burglary Rate")
```

Figure 6-18 *Graphique en bulles pour lequel les cercles ont été réduits.*

Il est aussi possible de créer un graphique avec d'autres formes à l'aide de `symbols()`. Vous pouvez ainsi créer des carrés, des rectangles, des thermomètres, des diagrammes à deux dimensions et des étoiles. Les arguments sont différents de ceux du cercle. Par exemple, les carrés utilisent la longueur du côté. Mais comme pour les bulles, vous voulez que la mesure des carrés soit fonction de la surface. Autrement dit, vous devez déterminer les côtés au carré en fonction de la racine carrée de l'aire.

La figure 6-19 illustre l'aspect des carrés à partir de la ligne de code suivante :

```
symbols(crime$murder, crime$burglary,
squares=sqrt(crime$population), inches=0.5)
```

Figure 6-19 *Utilisation de carrés à la place de cercles*

Tenons-nous en aux cercles pour l'heure. Le graphique de la figure 6-18 illustre une certaine forme de distribution, mais vous ne savez pas quel cercle représente quel État. Aussi, ajoutons les libellés, avec `text()`, dont les arguments sont les coordonnées *x*, les coordonnées *y* et le texte à afficher. Les valeurs *x* correspondent aux meurtres et les valeurs *y* aux cambriolages. Les libellés réels sont les noms des États, première colonne du cadre des données.

```
text(crime$murder, crime$burglary, crime$state, cex=0.5)
```

L'argument `cex` contrôle la taille du texte, qui est égale à 1 par défaut. Les valeurs supérieures à 1 permettent d'agrandir les libellés et les valeurs inférieures à 1 de les réduire. Les libellés peuvent être centrés sur les coordonnées *x* et *y*, comme illustré à la figure 6-20.

À ce stade, peu de modifications sont nécessaires pour que votre graphique corresponde au graphique final de la figure 6-15. Enregistrez le graphique créé

dans R en tant que fichier PDF, puis ouvrez-le dans votre logiciel d'illustration préféré pour affiner le graphique à votre guise. Pour l'alléger, vous pouvez affiner les lignes des axes et supprimer le contour. Vous pouvez aussi déplacer certains libellés, particulièrement en bas et à gauche, afin de lire aisément les noms des États. Amenez ensuite au premier plan la bulle correspondant à l'État de Georgie, qui était jusqu'à présent masquée par la plus grande bulle du Texas.

Pour configurer davantage d'options de tracé, saisissez `?symbols` dans R.

Figure 6-20 *Graphique en bulles avec libellés*

Distribution

Vous avez probablement entendu parler de moyenne arithmétique, de valeur médiane et de mode, et peut-être même les avez-vous étudiés au lycée ou à l'université. La moyenne arithmétique est la somme de tous les points de données divisée par le nombre de points. Pour trouver la valeur médiane, vous classez les données de la plus petite valeur à la plus grande et identifiez celle qui est située à mi-chemin entre les deux. Le mode désigne le nombre le plus souvent pré-

sent. Ces valeurs sont parfaites et très faciles à trouver, mais ne disent pas tout. Elles décrivent de quelle façon les différentes parties des données sont distribuées. Et si vous visualisez la totalité, vous pouvez appréhender la distribution complète.

Une inclinaison vers la gauche signifie que la plupart de vos données sont regroupées dans le côté inférieur de la plage complète. Une inclinaison vers la droite signifie l'inverse. Une ligne plate exprime une distribution uniforme, tandis que la courbe en cloche classique illustre un regroupement au niveau de la moyenne arithmétique et une diminution progressive dans les deux directions.

Examinons un tracé classique, principalement pour avoir un aperçu de la distribution, puis passons à l'histogramme, plus pratique, et au tracé de densité.

Distribution à l'ancienne

Dans les années 1970, quand les ordinateurs existaient à peine, la plupart des graphiques de données étaient dessinés à la main. Certains conseils du célèbre statisticien John Tukey, extraits de son livre intitulé *Exploratory Data Analysis*, étaient centrés autour de la plume et du crayon pour varier l'intensité des traits et des ombres. Il était aussi possible d'utiliser des modèles de hachage comme remplissage pour différencier les variables.

Le diagramme à tiges et à feuilles fut conçu d'une façon identique. Tout ce que vous avez à faire est d'écrire les nombres à l'aide d'une méthode ordonnée, le résultat final étant une vue approximative de la distribution. La méthode était particulièrement répandue dans les années 1980 (époque où l'utilisation des graphiques statistiques pour l'analyse montait en puissance), car il était aisé d'inclure le graphique – même si vous écriviez avec une machine à écrire.

Il existe aujourd'hui des solutions plus simples et plus rapides pour examiner les distributions. Mais il est utile de s'y intéresser parce que vous pourrez toujours appliquer les mêmes principes en créant un diagramme à tiges et à feuilles de la même façon que vous le feriez avec un histogramme.

Créer un diagramme à tiges et à feuilles

Vous êtes entièrement libre, bien sûr, de tracer votre diagramme à tiges et à feuilles avec un crayon et un papier, mais vous pouvez en créer un beaucoup plus rapidement dans R. La figure 6-21 illustre un diagramme à tiges et à feuilles pour les taux de naissance dans le monde entier en 2008, selon les estimations de la Banque mondiale.

Comme vous le constatez, ce diagramme est rudimentaire. Les nombres de base se trouvent à gauche et les chiffres après la virgule à droite. Dans ce cas, le point décimal se situe à la barre (|), et par conséquent, l'intervalle qui contient le plus grand nombre de pays est celui qui s'étend de 10 à 12 naissances pour 1 000 personnes. Le Niger, se distingue avec un taux de naissance compris entre 52 et 54.

Figure 6-21 *Diagramme à tiges et à feuilles illustrant les taux de natalité à travers le monde*

Voici comment vous traceriez ce diagramme à la main. Écrivez les nombres de 8 à 52 par intervalles de 2, de haut en bas. Tracez une ligne à droite de la colonne des nombres. Puis, parcourez chaque rangée de données et ajoutez les nombres correspondants. Si un pays a un taux de natalité de 8,2, vous ajoutez un 2 à la droite du 8. Si un pays a un taux de 9,9, il va aussi sur la ligne 8 : vous écrivez 9.

Comme cela peut devenir fastidieux si vous avez beaucoup de données, voici comment créer un diagramme à tiges et à feuilles dans R. Après avoir chargé les données, vous utilisez simplement la fonction `stem()`.

```
birth <- read.csv("http://datasets.flowingdata.com/birth-rate.csv")
stem(birth$X2008)
```

Si vous souhaitez donner plus d'allure à votre diagramme (figure 6-22), vous pouvez copier le texte dans R et le coller à un autre emplacement. Cependant, cette méthode est désuète et probablement mieux adaptée aux histogrammes, qui correspondent à une version plus graphique des diagrammes à tiges et à feuilles.

Figure 6-22 *Diagramme à tiges et à feuilles revu et corrigé*

Barres de distribution

Si vous observez le diagramme à tiges et à feuilles de la figure 6-22, vous pouvez repérer une fréquence dans des plages spécifiques. Plus une plage comporte de pays avec un taux de natalité, plus il y a de nombres tracés et plus longue est la ligne. Maintenant, basculez le tracé sur son côté de telle sorte que les lignes deviennent des colonnes. Plus la colonne est haute, plus la plage comporte de pays. Transformez la colonne des nombres en une simple barre ou un simple rectangle. Vous obtenez un histogramme, tel que celui illustré à la figure 6-23.

Figure 6-23 *Structure d'un histogramme*

La hauteur des barres représente la fréquence, leur largeur ne correspond à aucune valeur. Les axes horizontaux et verticaux sont continus. En revanche, l'axe horizontal d'un graphique en barres est discret. Lorsque vous utilisez un graphique en barres, vous avez défini des catégories et, généralement, l'espace qui sépare les barres.

Il arrive fréquemment que les personnes qui ne sont pas habituées à lire les graphiques de données confondent l'axe horizontal et le temps. Ce peut l'être, mais il n'y a aucune contrainte. Ce point est particulièrement important lorsque vous prenez en compte votre auditoire. Si votre graphique s'adresse à un public large, vous devez expliquer comment lire le graphique et souligner les points importants à noter. Gardez aussi à l'esprit que beaucoup de personnes ne sont pas familières du concept de distribution. Aussi concevez votre graphique de façon claire et vous pourrez le leur apprendre.

Créer un histogramme

De même que le diagramme à tiges et à feuilles, l'histogramme est très simple à créer dans R. À l'aide de la fonction `hist()`, tracez à nouveau la distribution des taux de natalité à travers le monde, comme vous le voyez à la figure 6-24. Avez-vous remarqué que sa forme était similaire à celle du diagramme à tiges et à feuilles de la figure 6-22 ?

Figure 6-24 *Distribution des taux de natalité à travers le monde*

Sous réserve que vous ayez déjà chargé les données de l'exemple précédent, exécutez la fonction `hist()` avec les mêmes nombres depuis 2008.

```
hist(birth$X2008)
```

Il s'agit de l'histogramme par défaut illustré à la figure 6-25. Il comporte dix barres, mais vous pouvez modifier ce nombre à l'aide de l'argument `breaks`. Par exemple, vous pourriez avoir moins de barres, plus larges (figure 6-26). L'histogramme ne contient que cinq classes.

```
hist(birth$X2008, breaks=5)
```

Vous pouvez aussi choisir l'autre solution et créer un histogramme avec plus de barres (par exemple 20, figure 6-27), celles-ci étant alors plus étroites.

```
hist(birth$X2008, breaks=20)
```

Figure 6-25 *Histogramme par défaut*

Figure 6-26 *Histogramme avec cinq classes*

Figure 6-27 *Histogramme avec vingt classes*

Le nombre de classes par défaut ne constitue pas toujours le meilleur choix pour votre histogramme. Essayez les différentes options et décidez du nombre le plus adapté à votre ensemble particulier de données.

Le nombre de classes que vous devez choisir dépend des données que vous visualisez. Si la plupart de vos données sont regroupées au début, vous souhaiterez peut-être avoir un plus grand nombre de classes afin d'apprécier les variations, au lieu de n'avoir qu'une seule barre élevée. D'un autre côté, si vous n'avez pas trop de données ou que les nombres sont également répartis, des barres plus larges pourraient être plus appropriées. La bonne nouvelle est que vous pouvez modifier et expérimenter sans aucune difficulté.

Dans le cas de nos données sur les taux de natalité, le nombre de classes par défaut convient parfaitement. Vous constatez qu'il y a certains pays avec un taux de natalité inférieur à 10, mais que la plupart ont un taux compris entre 10 et 25 pour 1 000 personnes. Un certain nombre de pays sont aussi au-dessus de 25, mais relativement peu si nous les comparons aux groupes inférieurs.

Dans R, vous pouvez calculer facilement la moyenne arithmétique, la valeur médiane, la valeur maximale et les quartiles avec la fonction `summary()`.

À ce stade, vous pouvez enregistrer le résultat obtenu en tant que fichier PDF, puis modifier le graphique dans Illustrator. La plupart des modifications seront similaires à celles apportées aux graphiques en barres du chapitre 4, mais certaines sont spécifiques et permettent de rendre le graphique plus lisible et d'expliquer aux lecteurs quel est son propos.

Dans le graphique final de la figure 6-24, vous pouvez voir quelques facettes importantes de la répartition, à savoir la valeur médiane, la valeur maximale et la valeur minimale. Le paragraphe d'introduction, bien sûr, constitue une autre

opportunité d'explication. Enfin, vous pouvez ajouter de la couleur pour rendre votre histogramme plus clair et agréable à lire.

Densité continue

Même si l'axe des valeurs est continu, la répartition est toujours ventilée en un nombre discret de barres. Chaque barre représente une collection d'éléments, ou dans le cas des précédents exemples, de pays. Quelle sorte de variation se produit au sein de chaque intervalle ? Avec le diagramme à tiges et à feuilles, vous pouvez voir chaque nombre, mais il reste difficile d'évaluer l'ampleur des différences, ce qui est similaire à la façon dont vous aviez utilisé les courbes LOESS de Cleveland et Devlin au chapitre 4 afin de mieux voir les tendances. Vous pouvez vous servir d'un tracé de densité pour visualiser les plus petites variations au sein d'une distribution.

Figure 6-28 *Structure d'un tracé de densité*

La figure 6-28 illustre l'emploi d'une courbe à la place de barres. L'aire totale sous la courbe est égale à 1 et l'axe vertical représente la probabilité ou la proportion d'une valeur dans un échantillon de la population.

Créer un tracé de densité

Si l'on revient aux données sur les taux de natalité, une étape supplémentaire est nécessaire pour obtenir un tracé de densité. Vous devez utiliser la fonction `density()` afin d'estimer les points de la courbe ; cependant, il ne peut y avoir de

valeurs manquantes dans les données. Pour l'instant, 15 lignes des données de 2008 ne comportent aucune valeur.

Dans R, ces emplacements avec valeurs manquantes sont libellés `NA`. Il est heureusement facile de les filtrer.

```
birth2008 <- birth$X2008[!is.na(birth$X2008)]
```

Cette ligne de code extrait la colonne `2008` des données sur les taux de natalité. Vous demandez ensuite uniquement les lignes qui n'ont pas de valeurs et les stockez dans `birth2008`. Plus techniquement parlant, `is.na()` vérifie chaque élément du vecteur `birth$X2008` et retourne un vecteur de longueur égale de valeurs `true` et `false`, appelées *booléens*. Lorsque vous passez un vecteur de valeurs booléennes à l'index d'un vecteur, seuls les éléments qui correspondent aux valeurs `true` sont retournés. Ne vous inquiétez pas si cela vous semble un peu compliqué. Vous n'avez pas besoin de connaître les détails techniques pour que cela fonctionne. Si, cependant, vous prévoyez d'écrire vos propres fonctions, il est utile de maîtriser le langage. Cette connaissance peut rendre la documentation plus facile à lire, mais, avec la pratique, vous vous y ferez peu à peu.

Maintenant que les taux de natalité sont stockés dans `birth2008`, vous pouvez les passer à la fonction `density()` pour estimer une courbe et stocker les résultats dans `d2008`.

```
d2008 <- density(birth2008)
```

Dans cet exemple, les valeurs manquantes sont retirées pour des raisons de simplicité. Lorsque vous visualisez et explorez vos propres données, observez plus attentivement les valeurs manquantes. Pourquoi sont-elles absentes ? Doivent-elles être définies à zéro ou totalement supprimées ?

Nous obtenons ainsi les coordonnées *x* et *y* de la courbe. Vous pouvez les enregistrer dans un fichier texte si vous souhaitez utiliser un autre programme pour le tracé. Saisissez `d2008` dans la console R pour afficher le contenu de la variable. Voici ce que vous obtenez :

La fonction `write.table()` enregistre les nouveaux fichiers dans votre répertoire de travail actuel. Si vous souhaitez le modifier, utilisez le menu principal ou la fonction `setwd()`.

```
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
```

Si vous souhaitez effectuer le tracé avec un autre logiciel que R, mais que vous souhaitez continuer à utiliser les fonctions de calcul de R, vous pouvez enregistrer tout ou partie de vos résultats avec`write.table()`.

La principale chose dont vous devez vous préoccuper sont les coordonnées *x* et *y*. Le résultat obtenu illustre la ventilation des coordonnées. Pour accéder à l'ensemble des coordonnées, entrez les lignes de code suivantes :

```
d2008$x
d2008$y
```

Pour stocker les coordonnées dans un fichier texte, utilisez `write.table()`. La fonction accepte comme arguments les données que vous voulez enregistrer, le nom du fichier dans lequel elles sont sauvegardées, le séparateur à choisir (virgule ou tabulation, par exemple), et quelques autres paramètres. Pour enregistrer les données comme fichier texte dont les valeurs sont séparées par des tabulations, entrez les instructions suivantes.

```
d2008frame <- data.frame(d2008$x, d2008$y)
write.table(d2008frame, "birthdensity.txt", sep="\t")
```

Le fichier `birthdensity.txt` doit être maintenant disponible dans votre répertoire de travail. Si vous ne voulez pas que les lignes soient numérotées et que la virgule remplace la tabulation comme séparateur, vous pouvez y parvenir sans peine.

```
write.table(d2008frame, "birthdensity.txt", sep=",", row.names=FALSE)
```

Vous pouvez maintenant charger ces données dans Excel, Tableau, D3 ou toute autre application qui accepte le texte délimité, soit la majorité d'entre elles.

Revenons à présent au tracé de la densité. Vous disposez déjà des coordonnées du tracé de densité. Il vous suffit de les placer sous forme graphique avec, naturellement, la fonction `plot()`. La figure 6-29 illustre le résultat obtenu.

```
plot(d2008)
```

Vous pouvez également créer un tracé de densité rempli, à l'aide de `plot()` et de `polygon()`, comme illustré à la figure 6-30. Vous utilisez `plot()` pour définir les axes, mais avec l'argument `type` ayant pour valeur `n` (pour *no plotting*). Avec `polygon()`, vous dessinez la forme ; définissez la couleur de remplissage en rouge foncé et la bordure en gris clair.

```
plot(d2008, type="n")
polygon(d2008, col="#821122", border="#cccccc")
```

Figure 6-29 *Tracé de densité pour les taux de naissance*

Figure 6-30 *Tracé de densité rempli*

Vous pouvez alors dessiner conjointement l'histogramme et le tracé de densité pour obtenir les fréquences exactes représentées par les barres et les proportions estimées de la courbe (figure 6-31). Utilisez les fonctions `histogram()` et `lines()` du package `lattice`. La fonction `histogram()` crée un nouveau tracé, tandis que la fonction `lines()` ajoute des lignes au tracé existant.

```
library(lattice)
histogram(birth$X2008, breaks=10)
lines(d2008)
```

Figure 6-31 *Histogramme et tracé de densité combinés*

Vous pouvez donc faire une multitude de choses et procéder à un très grand nombre de variations, mais les principes mathématiques et géométriques sont les mêmes que ceux du bon vieux diagramme à tiges et à feuilles. Vous comptez, agrégez et regroupez. La meilleure variation peut diverger en fonction de vos données. La figure 6-32 illustre un graphique plus abouti. J'ai désaccentué les lignes des axes, réorganisé les libellés et ajouté un pointeur pour la valeur médiane. L'axe vertical, qui correspond à la densité, n'est pas particulièrement utile dans ce graphique, mais je l'ai conservé par simple souci d'exhaustivité.

Figure 6-32 *Tracé de densité pour les taux de naissance à travers le monde en 2008*

Comparaison

Il est souvent utile de comparer plusieurs répartitions plutôt que les seules moyennes arithmétiques, valeurs médianes ou modes. Ces statistiques récapitulatives sont après tout des éléments descriptifs de la vue d'ensemble. Elles ne racontent qu'une partie de l'histoire.

Par exemple, je pourrais dire que le taux de natalité moyen pour le monde en 2008 était de 19,98 naissances pour 1 000 personnes et de 32,87 en 1960, soit un taux inférieur de 39 % en 2008 par rapport à 1960. Cependant, cela ne vous renseigne que sur ce qui se passe au centre de la distribution. Ou est-ce bien même le centre ? N'y a-t-il que quelques pays qui avaient un taux de natalité élevé en 1960, accroissant ainsi la moyenne ? Les différences en termes de taux de natalité ont-elles augmenté ou diminué au cours des toutes dernières décennies ?

Vous pouvez procéder à des comparaisons de multiples façons. Une méthode consiste à recourir exclusivement à l'analyse et à ne pas utiliser du tout la visualisation. (J'ai passé une année à apprendre les méthodes statistiques pendant mes études supérieures, et ce n'était que la partie visible de l'iceberg.) À l'inverse, vous pouvez ne choisir que la visualisation. Vos résultats ne seront pas équivalents à une réponse exacte offerte par une analyse statistique détaillée, mais ils suffiront pour permettre de prendre une décision avisée par rapport au domaine concerné. De toute évidence, vous vous apprêtez à privilégier la voie de la visualisation.

Répartitions multiples

Jusqu'à présent, nous ne nous sommes intéressés qu'aux distributions uniques, à savoir les taux de natalité pour 2008. Mais si vous avez examiné le fichier de données ou la structure des données dans R, vous savez déjà qu'y sont présents les taux de natalité annuels à partir de 1960. Comme je l'ai dit précédemment, le taux de natalité mondial a diminué de façon significative, mais de quelle manière la distribution complète a-t-elle évolué ?

Empruntons maintenant la voie directe afin de créer un histogramme pour chaque année et disposons ces derniers de telle façon qu'ils composent une matrice. L'idée est similaire à celle de la matrice de nuages de points conçue au début du chapitre.

Créer une matrice d'histogrammes

Le package `lattice` de R permet de créer facilement tout un ensemble d'histogrammes à partir d'une seule ligne de code, mais il existe un petit piège. Vous devez fournir les données au format attendu par la fonction. Vous trouverez ci-après un extrait du fichier texte initialement chargé.

```
Country,1960,1961,1962,1963...
Aruba,36.4,35.179,33.863,32.459...
Afghanistan,52.201,52.206,52.208,52.204...
...
```

Il y a une ligne pour chaque pays. La première colonne correspond au nom du pays, puis il y a une colonne par année, soit 30 colonnes et 234 lignes de données, plus l'en-tête. Vous avez toutefois besoin que les données soient en deux colonnes, l'une pour l'année et l'autre pour le taux de natalité. Comme vous n'avez pas véritablement besoin ici des noms des pays, les toutes premières lignes doivent se présenter de la façon suivante :

```
year,rate
1960,36.4
1961,35.179
1962,33.863
1963,32.459
```

```
1964,30.994
1965,29.513
...
```

Si vous comparez l'extrait qu'il vous faut à l'extrait actuel, vous remarquez que les valeurs du second extrait correspondent aux valeurs d'Aruba. Par conséquent, il existe une ligne pour chaque valeur de taux de natalité qui est accompagnée de l'année. Il en résulte 9 870 lignes de données, plus un en-tête.

Comment obtenir les données au format souhaité ? Souvenez-vous de ce que vous aviez fait au chapitre 2 avec Python. Vous aviez chargé le fichier CSV dans Python, puis parcouru chaque ligne l'une après l'autre, en affichant les valeurs selon le format souhaité. Vous pouvez procéder de même ici. Ouvrez un nouveau fichier dans votre éditeur de texte et nommez-le `transform-birth-rate.py`. Assurez-vous que le fichier se trouve bien dans le même répertoire que le fichier `birth-rate.csv`. Saisissez ensuite le script suivant :

```
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
```

Le code doit vous paraître familier, mais décomposons-le. Vous importez le package `csv` et chargez le fichier `birth-rate.csv`. Vous affichez ensuite l'en-tête et parcourez chaque ligne et chaque colonne de telle sorte que le script produise les données au format désiré. Exécutez le script sur votre console, puis enregistrez la sortie dans un nouveau fichier CSV, que vous nommerez `birth-rate-yearly.csv`.

```
python transform-birth-rate.py > birth-rate-yearly.csv
```

Parfait. Maintenant utilisez la fonction `histogram()` pour la matrice. Revenez dans R et chargez le nouveau fichier de données avec `read.csv()`. Au cas où vous ignoreriez toute la mise en forme des données (en vue d'un enregistrement

ultérieur), le nouveau fichier de données se trouve en ligne de telle sorte que vous puissiez le charger depuis une URL.

```
birth_yearly <-
  read.csv("http://datasets.flowingdata.com/birth-rate-yearly.csv")
```

Passons à présent les données à la fonction `histogram()` pour créer une matrice de 10 sur 5, les taux étant classés par année. La figure 6-33 illustre le résultat obtenu.

```
histogram(~ rate | year, data=birth_yearly, layout=c(10,5))
```

Si vous voulez conserver dans R toutes vos lignes de code, vous pouvez essayer d'utiliser le package de refaçonnage de Hadley Wickham. Il permet de basculer les structures de données au format de votre choix.

Figure 6-33 *Matrice d'histogrammes par défaut*

Ce résultat est satisfaisant mais il est possible de l'améliorer. Tout d'abord, on remarque une observation aberrante à l'extrémité droite qui pousse toutes les barres vers la gauche. Ensuite, une barre passe de gauche à droite selon l'année de chaque cellule de la matrice, mais la lecture serait plus simple si nous avions des libellés indiquant précisément l'année. Enfin, difficile d'apprendre quelque chose de la matrice, car l'ordre des histogrammes ne fonctionne pas.

La première année, 1960, se trouve en bas à gauche, et 1969 en bas à droite. La cellule située au-dessus de 1960 est celle de 1970. Ainsi, l'ordre se déplace de bas en haut et de gauche à droite. Étrange.

Pour trouver l'observation aberrante, utilisez à nouveau `summary()` sur `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
```

Le taux maximal est 132. La valeur paraît anormale. Aucun autre taux ne s'approche même de 100. Que se passe-t-il ? Il s'avère que le taux enregistré pour Palau en 1999 est 132. Il s'agit vraisemblablement d'une coquille, car les taux pour Palau avant et après 1999 ne sont pas supérieurs à 20. Probablement le taux est-il censé être de 13,2, mais il faudrait y regarder plus en détail. Pour l'instant, supprimez temporairement cette erreur.

```
birth_yearly.new <- birth_yearly[birth_yearly$rate < 132,]
```

Passons aux libellés des années. Quand les valeurs utilisées pour les libellés sont stockées comme nombres, la fonction `lattice` emploie automatiquement la barre orange pour indiquer une valeur. Cependant, si les libellés sont des caractères, la fonction utilise des chaînes. Ce que nous allons faire maintenant.

```
birth_yearly.new$year <- as.character(birth_yearly.new$year)
```

Il vous reste toujours à mettre à jour l'ordre, mais créez d'abord la matrice des histogrammes et stockez-la dans une variable.

```
h <- histogram(~ rate | year, data=birth_yearly.new, layout=c(10,5))
```

Utilisez maintenant la fonction `update()` pour modifier l'ordre des histogrammes.

```
update(h, index.cond=list(c(41:50, 31:40, 21:30, 11:20, 1:10)))
```

Cette instruction inverse l'ordre de toutes les lignes. Comme illustré à la figure 6-34, vous obtenez une matrice aux jolis libellés, ainsi qu'un meilleur aperçu des distributions, une fois la coquille supprimée. De plus, comme les histogrammes sont disposés plus logiquement, vous pouvez lire de gauche à droite et de haut en bas. Lisez simplement une cellule de chaque ligne, et déplacez les yeux de haut en bas afin de voir de quelle façon la distribution change par décennie.

Figure 6-34 *Matrice des histogrammes modifiée*

À ce stade, la disposition est bonne. Si vous aviez à affiner le graphique dans Illustrator, vous pourriez réduire certains libellés, modifier la couleur des bordures et la couleur de remplissage, et procéder à quelque nettoyage général, comme vous le voyez à la figure 6-35. Le résultat est plus lisible de cette façon. Pour qu'il soit même plus clair, vous pouvez aussi ajouter une introduction, inclure la source et souligner de quelle façon la distribution glisse de la gauche vers les taux de natalité inférieurs à travers le monde. Le graphique pourrait être trop complexe comme graphique autonome. Vous devez fournir beaucoup de contexte aux lecteurs pour qu'ils puissent apprécier réellement les données.

Ce n'est bien sûr pas la seule façon d'agir. Vous pourriez créer la même matrice avec Processing, D3, PHP… Il existe même de multiples façons de créer le même type de matrice dans R. Par exemple, j'ai créé un graphique pour FlowingData sur l'évolution de la distribution des tailles des télévisions de 2002 à 2009 (figure 6-36).

Figure 6-35 *Matrice d'histogrammes affinée dans Illustrator*

Figure 6-36 *Distribution de la taille des télévisions de 2002 à 2009*

Le code paraît différent de ce que vous venez juste d'écrire, mais la logique est la même. J'ai chargé les données, j'ai appliqué certains filtres pour les données aberrantes et j'ai tracé un ensemble d'histogrammes. La différence ici est que je n'ai pas utilisé `histogram()` du package `lattice`. Au lieu de cela, j'ai spécifié la mise en page avec `par()`, qui permet de définir des paramètres universels dans R, puis j'ai tracé chaque histogramme à l'aide de `hist()`.

```
# Charger les données
tvs <- read.table('http://datasets.flowingdata.com/tv_sizes.txt',
     sep="\t", header=TRUE)

# Filtrer les observations aberrantes
tvs <- tvs[tvs$size < 80, ]
tvs <- tvs[tvs$size > 10, ]

# Définir les classes des histogrammes
breaks = seq(10, 80, by=5)

# Définir la disposition
par(mfrow=c(4,2))

# Dessiner les histogrammes, un par un
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)
```

Le résultat correspondant est illustré à la figure 6-37. Le graphique se compose de quatre lignes et de deux colonnes, comme spécifié dans l'argument `mfrow` de `par()`. Pour le graphique final, j'ai regroupé l'ensemble en une seule colonne, mais le point important est que je n'ai pas eu à saisir beaucoup de données dans Illustrator ou Excel pour créer manuellement huit graphiques.

Petits multiples

La technique qui consiste à regrouper plusieurs graphiques en un seul est communément appelée « technique des petits multiples ». Elle invite le lecteur à effectuer des comparaisons entre différents groupes ou catégories, ou à l'intérieur des groupes ou catégories eux-mêmes. De plus, si votre graphique est organisé, vous pouvez placer de nombreuses informations au sein d'un seul et même espace.

Figure 6-37 *Disposition en grille pour les histogrammes*

Par exemple, je me suis intéressé aux notes attribuées aux films sur le site Rotten Tomatoes pour les trilogies. Au cas où vous ne le sauriez pas, Rotten Tomatoes regroupe les critiques des films et les marque comme positives ou négatives. Quand au moins 60 % des critiques sont positives, le film est considéré comme « frais » (*fresh*), sinon il est classé comme « pourri » (*rotten*). Je souhaitais savoir comment les suites étaient comparées aux originaux en termes de « fraîcheur ».

Le résultat n'est pas très bon, comme illustré à la figure 6-38. L'évaluation moyenne du troisième volet de la trilogie était inférieure de 37 % à la valeur médiane des originaux. Autrement dit, la majorité des originaux était « fraîche » et la majorité des troisièmes volets « pourrie ».

Figure 6-38 *Évaluation des trois volets d'une trilogie*

Figure 6-39 *Histogrammes originaux*

La figure 6-39 illustre les histogrammes originaux dans R. Je les ai légèrement améliorés dans Illustrator. Quoi qu'il en soit, les lecteurs de FlowingData comprirent le graphique dans leur grande majorité. Cependant, le graphique fut par la suite mis en lien depuis IMDB (*Internet Movie Database*). IMDB a un auditoire

beaucoup plus général et, à en juger par les commentaires, les lecteurs les moins experts en données eurent quelques difficultés à interpréter les distributions.

Cependant, la seconde partie du graphique (figure 6-40) semblait beaucoup plus facile à comprendre. Elle constitue un emploi des petits multiples où chaque barre représente l'évaluation d'un film. Les barres étaient rouges pour les films « pourris » et vertes pour les films « frais ».

Figure 6-40 *Petits multiples pour les évaluations des trilogies*

Au cas où vous vous demanderiez comment faire pour obtenir ce résultat, il suffit de regrouper plusieurs graphiques en barres. Ainsi, vous pourriez modifier le paramètre `mfrow` comme vous l'aviez fait précédemment et utiliser la fonction `plot()` ou `polygon()`. Je me suis toutefois servi de l'outil Graphe à barres verticales d'Illustrator, parce qu'il s'est avéré que je l'avais déjà ouvert.

J'ai appris un certain nombre de choses après avoir publié le graphique. Le plus important est que les agrégats et les distributions ne sont pas quelque chose que chacun rencontre tous les jours. Aussi, vous devez faire votre possible pour expliquer les données et raconter au mieux votre histoire. Une autre chose que j'ai apprise est que les spectateurs n'apprécient pas qu'on leur dise que les films qu'ils apprécient sont mauvais.

Pour résumer

Parfois, la recherche de relations dans les données peut constituer un défi et nécessite une plus grande réflexion critique que le fait de tracer aveuglément des nombres. Cependant, elle peut aussi être beaucoup plus gratifiante et révéler davantage d'informations. C'est la façon dont vos données, ou plutôt, la façon dont les « choses » que vos données représentent s'associent et interagissent entre elles qui est intéressant, et qui donne les meilleures histoires.

Ce chapitre a traité de la recherche de corrélations entre plusieurs variables et expliqué aussi les relations en termes plus généraux. Regardez comment chaque chose se rapporte à une autre comme un tout au travers des distributions. Cherchez au sein des distributions les observations aberrantes ou les modèles, puis réfléchissez au contexte qui entoure ce que vous voyez. Si vous trouvez quelque chose d'intéressant, demandez-vous pourquoi. Méditez sur le contexte des données et les explications possibles.

Telle est la meilleure partie dans la manipulation des données, parce que vous êtes conduit à explorer ce à quoi les données se réfèrent et, peut-être, à exhumer quelque chose d'intéressant. Puis, si vous creusez assez, vous pourrez expliquer aux lecteurs ce que vous avez trouvé. N'oubliez pas que tout le monde ne parle pas la langue des chiffres et, par conséquent, restez d'abord à un niveau compréhensible par le public le plus large possible. Et, si vous avez un auditoire composé de cracks, alors n'hésitez pas à passer au niveau supérieur !

Chapitre 7

Identification des différences

Les commentateurs sportifs aiment à considérer certains athlètes comme une élite et dont leurs adversaires ne seraient que de simples participants. Ces classifications ne proviennent pas tant de statistiques sportives que de l'observation des matchs eux-mêmes. Cela revient à ne pouvoir reconnaître un grand sportif que lorsque l'on se retrouve face à lui. En soi, rien de faux. Les commentateurs, en principe, savent de quoi ils parlent et prennent toujours en compte le contexte des résultats. J'apprécie toujours quand des analystes sportifs examinent les performances et, quasi immanquablement, déclarent : « Regarder les chiffres ne suffit pas. Ce sont des éléments insaisissables qui font la grandeur ». C'est là que les statistiques entrent en scène.

De toute évidence, cette remarque ne s'applique pas qu'aux sportifs. Peut-être souhaitez-vous savoir quels sont les meilleurs restaurants d'une ville ou identifier les clients fidèles ? Plutôt que de classer les unités par catégorie, vous pourriez rechercher la personne ou l'élément qui tranche sur le reste. Ce chapitre s'intéresse à la détection de groupes au sein d'une population, à partir de plusieurs critères, et à l'identification des observations aberrantes.

Que rechercher ?

La comparaison à l'aide d'une seule variable est aisée. Une maison possède une surface plus grande qu'une autre ou un chat est plus lourd qu'un autre. Comparer à partir de deux variables est un peu plus difficile, mais toujours possible. La première maison est plus grande, mais la seconde possède plus de salles de bains. Le premier chat a un poids supérieur et un poil plus court, mais le second pèse moins et a un poil plus long.

Qu'en est-il si vous avez une centaine de maisons ou de chats à classer ? Qu'en est-il si, pour chaque maison, vous devez considérer plusieurs variables, comme le nombre de chambres, la superficie du jardin et le montant des impôts locaux ?

Vous vous retrouverez avec le nombre d'unités multiplié par le nombre de variables. La comparaison devient plus complexe et c'est sur cette difficulté que nous allons porter notre attention.

Si vos données comportent un certain nombre de variables, peut-être désirez-vous classer ou regrouper les unités (des personnes ou des lieux, par exemple) par catégories et identifier les observations aberrantes. Vous voulez rechercher les différences de chaque variable, mais vous souhaitez aussi étudier les différences à travers l'ensemble des variables. Deux joueurs de basketball peuvent avoir des moyennes de points marqués complètement différentes, mais avoir pratiquement les mêmes nombres de rebonds, interceptions et minutes par match. Vous devez repérer les différences mais ne pas oublier les similitudes et les relations, tout comme les commentateurs sportifs !

Comparaison entre plusieurs variables

Lors du traitement de plusieurs variables, l'un des principaux défis consiste à déterminer le point de départ. Vous êtes confronté à tant de variantes et de sous-ensembles que vous risquez de vous retrouver écrasé sous le poids de la tâche si vous ne cessez de vous inquiéter des données en votre possession. Parfois, il est préférable d'analyser toutes les données en même temps, et quelques points intéressants pourraient vous montrer la bonne direction à emprunter.

Les cartes chaudes

Pour visualiser un tableau de données, l'un des moyens les plus simples est de le montrer dans sa totalité. À la place des nombres, vous pouvez recourir aux couleurs pour indiquer les valeurs, comme illustré à la figure 7-1.

Vous vous retrouvez avec une grille de la même taille que le tableau original, mais vous pouvez aisément identifier les valeurs élevées ou basses en fonction de la couleur. Généralement, plus la valeur est élevée, plus la couleur est sombre, et inversement. Ce comportement peut être modifié sans peine en fonction de votre application.

Lisez la carte chaude comme s'il s'agissait d'un tableau. Vous pouvez lire une ligne de gauche à droite pour voir les valeurs de toutes les variables d'une seule unité, ou comparer toutes les unités à partir d'une seule variable.

Cette disposition peut prêter à confusion, notamment si le tableau de données est volumineux. Aidez-vous d'un jeu de couleurs adapté et d'une opération de tri pour créer un graphique lisible et séduisant.

Figure 7-1 *Structure d'une carte chaude*

Créer une carte chaude

La création de cartes chaudes dans R ne présente aucune difficulté car celui-ci propose une fonction `heatmap()` qui effectue tous les calculs. Vous n'avez qu'à choisir les couleurs les plus appropriées aux données et à organiser les libellés afin qu'ils demeurent lisibles, même si vous avez un grand nombre de lignes et de colonnes. Autrement dit, R configure le cadre et vous gérez la présentation d'ensemble. Cette tâche doit vous être familière désormais.

Dans cet exemple, nous nous intéressons aux statistiques de basketball de la NBA pour 2008. Vous pouvez télécharger les données au format CSV à l'adresse suivante : *http://datasets.flowingdata.com/ppg2008.csv*. On compte 22 colonnes, la première correspondant aux noms des joueurs et les autres aux différentes statistiques telles que le nombre de points par match ou le pourcentage de lancers. Vous pouvez utiliser `read.csv()` pour charger les données dans R. Examinons les cinq premières lignes pour obtenir un aperçu de la structure des données (figure 7-2).

```
bball <-
    read.csv("http://datasets.flowingdata.com/ppg2008.csv",
    header=TRUE)
bball[1:5,]
```

Figure 7-2 *Structure des cinq premières lignes de données*

Les joueurs sont triés en fonction du nombre de points par match, du plus élevé au plus faible. Cependant, vous pouvez les trier sur la colonne de votre choix, comme le nombre de rebonds par match ou le pourcentage de paniers réussis, à l'aide de la fonction `order()`.

```
bball_byfgp <- bball[order(bball$FGP, decreasing=TRUE),]
```

L'argument `decreasing` dans `order()` indique si vous voulez que les données soient triées par ordre croissant ou décroissant.

À présent, si vous examinez les cinq premières lignes de `bball_byfgp`, vous constatez que les joueurs qui se retrouvent en tête sont Shaquille O'Neal, Dwight Howard et Pau Gasol, à la place de Dwyane Wade, Lebron James et Kobe Bryant. Pour cet exemple, inversons l'ordre du nombre de points par match.

```
bball <- bball[order(bball$PTS, decreasing=FALSE),]
```

En l'état, les noms des colonnes correspondent à l'en-tête du fichier CSV. C'est ce que vous souhaitez. Mais vous voulez aussi libeller les lignes en fonction du nom du joueur et pas du numéro de ligne. Par conséquent, superposez la première colonne aux noms des lignes.

```
row.names(bball) <- bball$Name
bball <- bball[,2:20]
```

La première ligne de ce code remplace les noms de ligne par la première colonne du tableau de données. La seconde ligne de code sélectionne les colonnes 2 à 20 et redéfinit le sous-ensemble des données en `bball`.

Les données doivent aussi être dans un format matriciel, plutôt que tabulaire. Vous obtiendriez une erreur si vous essayiez d'utiliser un tableau de données avec la fonction `heatmap()`. Généralement, un tableau de données s'apparente à une collection de vecteurs où chaque colonne représente une mesure distincte.

Une grande part du travail de visualisation nécessite la collecte et la préparation des données. Comme il est rare d'obtenir les données au format souhaité, vous devrez certainement les manipuler quelque peu avant de commencer le travail de visualisation.

Chaque colonne peut avoir différents formats, tels que numérique ou chaîne. En revanche, une matrice permet traditionnellement de représenter un espace bidimensionnel et le type de données doit être uniforme d'une cellule à l'autre.

```
bball_matrix <- data.matrix(bball)
```

Les données sont triées et mises en forme conformément à vos choix ; vous pouvez les connecter à `heatmap()` pour recueillir le fruit de vos efforts. En définissant l'argument `scale` avec la valeur `column`, vous demandez à R d'utiliser les valeurs minimale et maximale de chaque colonne pour déterminer les dégradés de couleurs à la place des valeurs minimale et maximale de la totalité de la matrice.

```
bball_heatmap <- heatmap(bball_matrix, Rowv=NA,
    Colv=NA, col = cm.colors(256), scale="column",
    margins=c(5,10))
```

Le résultat obtenu doit ressembler à la figure 7-3. Avec `cm.colors()`, vous avez spécifié un éventail de couleurs du cyan au magenta. La fonction crée un vecteur de couleurs hexadécimales avec, par défaut, la plage du cyan au magenta, et `n` nuances intermédiaires (256, dans le cas présent). Notez que la troisième colonne, qui correspond au nombre de points par match, commence à la couleur magenta, soit les valeurs les plus élevées pour Dwyane Wade et Lebron James, puis évolue vers une teinte cyan plus sombre pour Allen Iverson et Nate Robinson. Vous pouvez aussi trouver rapidement d'autres présences de la couleur magenta, correspondant à Dwight Howard, premier au rebond, ou à Chris Paul, son coéquipier.

Peut-être souhaitez-vous un autre jeu de couleurs. Pour cela, modifiez simplement l'argument `col`, à savoir `cm.colors(256)` dans la ligne de code que vous venez juste d'exécuter. Tapez `?cm.colors` pour obtenir de l'aide sur les couleurs proposées par R. Par exemple, vous pourriez utiliser des couleurs plus chaudes, comme illustré à la figure 7-4.

```
bball_heatmap <- heatmap(bball_matrix,
    Rowv=NA, Colv=NA, col = heat.colors(256), scale="column",
    margins=c(5,10))
```

Figure 7-3 *Carte chaude triée par défaut sur le nombre de points par match*

Figure 7-4 *Carte chaude avec un dégradé de couleurs du rouge au jaune*

Si vous tapez `cm.colors(10)` dans la console R, vous obtennez un tableau de dix couleurs s'échelonnant du cyan au magenta. La fonction `heatmap()` choisit ensuite automatiquement la couleur qui correspond à chaque valeur à partir d'une échelle linéaire.

```
 [1] "#80FFFFFF" "#99FFFFFF" "#B3FFFFFF" "#CCFFFFFF" "#E6FFFFFF"
 [6] "#FFE6FFFF" "#FFCCFFFF" "#FFB3FFFF" "#FF99FFFF" "#FF80FFFF"
```

Parfait, car vous pouvez facilement créer votre propre échelle de couleurs. Par exemple, rendez-vous sur le site *0to255.com*, choisissez une couleur de base et démarrez votre dégradé à partir d'elle. La figure 7-5 illustre un dégradé dont la couleur de base est le rouge. Vous pouvez choisir quelques couleurs, allant des plus claires aux plus foncées, puis les associer facilement à `heatmap()`, comme vous pouvez le voir à la figure 7-6. Au lieu de créer un vecteur de couleurs à l'aide de R, vous définissez le vôtre avec la variable `red_colors`.

```
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))
```

Choisissez intelligemment les couleurs, car elles définissent aussi la tonalité du contexte de votre histoire. Par exemple, si vous traitez un thème sombre, il est préférable que vous choisissiez des couleurs neutres et plus sourdes. Les couleurs vives, quant à elles, conviendront mieux à des sujets ordinaires ou plus exaltants.

Si vous ne voulez pas sélectionner vos propres couleurs, vous pouvez utiliser le package `RColorBrewer`. Celui-ci n'est pas installé par défaut, vous devrez donc le télécharger et l'installer si ce n'est pas déjà fait. ColorBrewer a été conçu par la cartographe Cynthia Brewer et était destiné à l'origine aux cartes. Cependant, ce package peut vous aider à créer des graphiques de données. Vous avez le choix entre une grande variété d'options, comme la palette de couleurs séquentielles ou divergentes, ainsi que le nombre de nuances. Dans le cadre de cet exemple, optez pour une simple palette de bleus. Saisissez `?brewer.pal` dans la console R pour accéder à un plus grand nombre d'options. En présumant que vous avez installé RColorBrewer, saisissez les lignes suivantes pour obtenir une carte chaude utilisant une palette de bleus et neuf nuances. La figure 7-7 illustre le résultat obtenu.

La version interactive de ColorBrewer est disponible à l'adresse *http://colorbrewer2.com*. Vous pouvez sélectionner les options à partir des menus déroulants pour visualiser le rendu des jeux de couleurs sur un exemple de carte.

```
library(RColorBrewer)
bball_heatmap <- heatmap(bball_matrix, Rowv=NA,
    Colv=NA, col = brewer.pal(9, "Blues"),
    scale="column", margins=c(5,10))
```

Figure 7-5 *Dégradé de rouges extrait du site 0to255.com*

Figure 7-6 *Carte chaude utilisant un dégradé de rouges personnalisé*

Importez la figure 7-7 dans Illustrator afin de l'améliorer. Peu de modifications sont nécessaires. Vous allez ici modifier les libellés pour les rendre plus lisibles et adoucir les couleurs de manière à ce que le graphique soit plus facilement analysable.

Figure 7-7 *Carte chaude utilisant RColorBrewer pour la palette des couleurs*

Commençons par les libellés, que nous allons remplacer par des descriptions complètes. En tant que fan de basketball, je sais ce que représente chaque abréviation, mais une personne qui n'est pas familière de ce sport risque d'être perdue.

Adoucissez ensuite les couleurs en atténuant le contraste grâce à la fonction de transparence, disponible dans la boîte de dialogue Couleur d'Illustrator. Les contours des cellules peuvent aussi enrichir la définition de chaque cellule de telle sorte que le graphique soit plus facile à analyser de gauche à droite et de haut en bas. La figure 7-8 illustre le résultat final.

Figure 7-8 *Carte chaude montrant les performances par match de la NBA pour les 50 meilleurs marqueurs durant la saison 2008-2009*

Lire sur le visage

L'intérêt des cartes chaudes est qu'elles permettent de visualiser toutes les données simultanément. Cependant, l'accent est mis sur les points individuels. Vous pouvez facilement identifier les scores élevés ou faibles concernant les nombres de points ou de rebonds par match, mais il est plus intéressant de comparer un joueur à un autre.

Souvent, vous souhaitez voir chaque unité comme un tout, au lieu de l'éclater en plusieurs mesures. Les visages de Chernoff sont un moyen d'y parvenir. Cependant, la méthode n'est pas parfaitement exacte et il se peut que le public en soit dérouté. Ceci dit, les visages de Chernoff peuvent être utiles de temps à autre et ce peut être un exercice fort divertissant pour les passionnés des données.

L'objet des visages de Chernoff est d'afficher plusieurs variables à la fois en positionnant les parties du visage humain (par exemple les oreilles, les yeux et le nez) en fonction des valeurs numériques d'un ensemble de données (figure 7-9). L'hypothèse est que vous pouvez lire facilement les visages des individus dans la vie réelle et que, par conséquent, vous devriez être à même de reconnaître les légères différences quand ces visages représentent des données. L'hypothèse est peut-être audacieuse, mais faisons-la nôtre.

Comme vous le verrez dans l'exemple suivant, les valeurs supérieures ressortent sous la forme d'une chevelure épaisse ou de grands yeux, tandis que les valeurs inférieures tendent à réduire les caractéristiques faciales. En dehors de la taille, vous pouvez aussi ajuster des caractéristiques comme la courbe des lèvres ou la forme du visage.

Figure 7-9 *Structure des visages de Chernoff*

Créer les visages de Chernoff

Revenons aux données sur les 50 meilleurs marqueurs de la NBA, pendant la saison 2008-2009. Nous allons créer un visage par joueur. Ne vous inquiétez pas, vous n'avez pas à créer chaque visage manuellement ! Dans R, le package `aplpack` de Hans Peter Wolf propose la fonction `faces()` qui vous sera d'une grande utilité.

Si vous ne disposez pas du package `aplpack`, installez-le grâce à la commande `install.packages()` ou via son programme d'installation. Lors de l'installation, le package est normalement automatiquement chargé. Si tel n'est pas le cas, il vous appartient de le faire.

```
library(aplpack)
```

Vous devez avoir également déjà chargé les données lors de la création d'une carte chaude. Dans le cas contraire, utilisez à nouveau `read.csv()` pour charger directement les données depuis une URL.

```
bball <- read.csv("http://datasets.flowingdata.com/ppg2008.csv",
header=TRUE)
```

Une fois que le package et les données ont été chargés, il ne reste plus qu'à créer les visages de Chernoff avec la fonction `faces()`, comme illustré à la figure 7-10.

```
faces(bball[,2:16], ncolors=0)
```

Votre jeu de données comporte plus de 20 variables, ainsi que les noms des joueurs. Cependant, la fonction `faces()` proposée par `aplpack` ne permet qu'un maximum de 15 variables, parce que vous ne pouvez modifier qu'un certain nombre de caractéristiques faciales. Voilà pourquoi vous répartissez le sous-ensemble des données sur les colonnes 2 à 16.

Que représente chaque visage ? La fonction `faces()` modifie les caractéristiques dans l'ordre suivant, conformément à l'ordre des colonnes de données.

1. Longueur du visage
2. Largeur du visage
3. Forme du visage
4. Longueur de la bouche
5. Largeur de la bouche
6. Courbe du sourire
7. Longueur des yeux
8. Largeur des yeux
9. Longueur des cheveux
10. Largeur des cheveux

11. Coupe de cheveux
12. Longueur du nez
13. Largeur du nez
14. Largeur des oreilles
15. Longueur des oreilles

La version la plus récente du package `aplpack` permet d'ajouter une couleur à l'aide de la fonction `faces()`. Dans cet exemple, vous définissez `ncolors` avec la valeur 0 afin d'utiliser uniquement le noir et blanc. Saisissez `?faces` pour voir comment se servir des vecteurs de couleurs de la même façon que dans l'exemple précédent de carte chaude.

Figure 7-10 *Visages de Chernoff par défaut*

Quand il y a plusieurs personnes, il peut être utile de les regrouper par catégorie afin que les visages soient plus faciles à analyser. Dans le cas présent, vous pourriez distinguer les visages en fonction du poste : arrière, ailier ou pivot.

Par exemple, la longueur du visage représente le nombre de rencontres disputées, et celle de la bouche les paniers réalisés par match. Ces informations sont quelque peu inutiles en l'état, car vous n'avez aucun nom en regard des visages, mais vous pouvez remarquer que les tout premiers montrent plus de réussite que les autres, que le joueur 7 possède une chevelure relativement large qui correspond aux paniers à trois points, etc.

Utilisez l'argument `labels` de `faces()` pour ajouter les noms aux visages, comme vous pouvez le voir à la figure 7-11.

```
faces(bball[,2:16], labels=bball$Name)
```

Figure 7-11 *Visages de Chernoff avec les noms des joueurs*

Maintenant, vous savez à quel visage correspond tel joueur. Pour identifier les meneurs de jeu, vous pouvez commencer par Chris Paul, par exemple, et rechercher les visages similaires, tels que celui de Devin Harris ou Deron Williams. Chauncey Billups, dans le coin inférieur droit, est également meneur de jeu, mais son visage paraît différent de celui des autres. Les cheveux sont plus longs et la bouche plus étroite, ce qui correspond respectivement à un pourcentage de lancers-francs et un nombre de tentatives de paniers élevés.

Pour améliorer la lisibilité du graphique, vous pouvez agrandir l'espace entre les lignes et fournir une description de la correspondance entre la caractéristique faciale et les données sportives prises en compte (figure 7-12). En temps normal, j'utiliserais une légende graphique, mais comme nous nous sommes servis de chaque caractéristique faciale, proposer autant d'indications pour un même visage constitue un réel défi.

L'utilité des visages de Chernoff peut varier selon le jeu de données et le lectorat concerné ; par conséquent, il vous appartient de décider d'utiliser ou non la méthode. Notez toutefois que les personnes qui ne sont pas accoutumées aux visages de Chernoff tendent à les prendre un peu trop au pied de la lettre, et pensent que les visages représentés sont ceux des joueurs !

Figure 7-12 *Visages de Chernoff correspondant aux meilleurs marqueurs de la NBA pendant la saison 2008-2009*

Mettez-vous à la place du lecteur lorsque vous concevez votre graphique. Le lecteur ne sera pas toujours aussi familier que vous l'êtes des méthodes de visualisation ou n'aura pas le même niveau de connaissance des données. Par conséquent, c'est à vous de fournir les explications nécessaires.

De manière similaire, j'ai conçu un graphique pour le crime aux États-Unis (figure 7-13) à l'aide des visages de Chernoff. Quelqu'un m'a fait la remarque qu'il s'agissait d'une représentation quelque peu raciste, en raison de l'aspect du visage qui correspondaient aux États présentant des taux de criminalité élevés. Cela ne m'a jamais traversé l'esprit, parce que, pour moi, changer une caractéristique faciale était comme changer la longueur d'une barre sur un graphique, mais sans doute est-ce là une réflexion à méditer.

Figure 7-13 *Crime aux États-Unis avec un visage pour chaque État*

Nuit étoilée

Au lieu d'utiliser les visages pour illustrer les données à plusieurs variables, vous pouvez choisir une autre représentation. Vous ne changez pas les caractéristiques faciales, mais la forme pour qu'elle corresponde aux valeurs des données. Telle est l'idée des graphiques en étoile, appelés aussi graphiques en radar ou graphiques en toile d'araignée.

Comme illustré à la figure 7-14, vous pouvez tracer plusieurs axes, un pour chaque variable, en commençant par le milieu et en les espaçant de façon égale dans un cercle. Le centre constitue la valeur minimale de chaque variable et les extrémités représentent les valeurs maximales. Par conséquent, si vous créez un graphique pour une seule unité, commencez par une variable et tracez une ligne continue jusqu'au point correspondant de l'axe suivant. Vous vous retrouvez alors avec une figure qui évoque une étoile (ou un radar ou une toile d'araignée).

Figure 7-14 *Structure du graphique en étoile*

Il est possible de représenter plusieurs unités sur un seul graphique, mais celui-ci deviendra très vite inutile et il en résultera une histoire médiocrement racontée. Aussi tenez-vous en à des graphiques en étoile distincts et comparez-les.

Créer un graphique en étoile

Utilisez les mêmes données sur la criminalité que celles de la figure 7-13 pour vérifier si l'emploi d'un graphique en étoile entraîne une différence. Tout d'abord, chargez les données dans R.

```
crime <- read.csv("http://datasets.flowingdata.com/
crimeRatesByState-formatted.csv")
```

La création de graphiques en étoile est aussi simple que celle des visages de Chernoff. Utilisez la fonction `stars()`, fournie avec le logiciel R.

```
stars(crime)
```

Les graphiques par défaut sont illustrés à la figure 7-15. Nous espérons qu'ils ne choqueront personne. Bien sûr, vous aurez toujours besoin des libellés des États, mais également d'une légende explicative sur les branches.

Figure 7-15 *Graphiques en étoile par défaut illustrant la criminalité par État*

Un certain ordre est respecté, comme avec `faces()`, mais vous ne savez pas où commence la première variable. Notez que vous pouvez remplacer les noms des lignes par la première colonne, tout comme vous l'aviez fait avec la carte chaude. Vous pouvez aussi définir `flip.labels` sur la valeur `FALSE`, parce que vous ne voulez pas que les libellés changent de hauteur. La figure 7-16 illustre les résultats obtenus.

```
row.names(crime) <- crime$state
crime <- crime[,2:7]
stars(crime, flip.labels=FALSE, key.loc = c(15, 1.5))
```

Désormais, il est relativement facile d'identifier les différences et les ressemblances. Dans la version des visages de Chernoff, le District of Columbia ressemblait à un clown au regard perdu, si on le compare à tous les autres États, mais dans la version du graphique en étoile, vous voyez qu'il présente des taux élevés de criminalité dans certaines catégories, mais des taux relativement bas de viol avec violence et de cambriolage. Il est également aisé de trouver des États avec des taux de criminalité relativement bas, comme le New Hampshire et Rhode Island. Ensuite, il y a des États comme la Caroline du Nord qui présentent un taux élevé dans une seule catégorie.

Pour cet ensemble de données, le format me convient, mais il y a deux variantes que vous pourriez vouloir essayer avec vos propres données. La première restreint toutes les données à la moitié supérieure du cercle, comme illustré à la figure 7-17.

```
stars(crime, flip.labels=FALSE, key.loc = c(15, 1.5), full=FALSE)
```

La seconde variante utilise la longueur des segments à la place du placement des points (figure 7-18). Il s'agit de fait des graphiques de Nightingale (aussi appelés diagrammes polaires) plus que des graphiques en étoile, mais tel est le résultat. Si vous retenez cette option, vous voudrez peut-être tester un autre jeu de couleurs que celui proposé par défaut.

```
stars(crime, flip.labels=FALSE, key.loc = c(15, 1.5), draw.segments=TRUE)
```

Le format original de la figure 7-16 me convient ; aussi, vous pouvez l'importer dans Illustrator pour effectuer quelques opérations. Il n'est pas nécessaire de procéder à de nombreuses modifications. Un plus grand espace entre les lignes pourrait rendre les libellés moins ambigus ; de même, vous pouvez placer la clé en haut afin que les lecteurs sachent ce qu'ils regardent (figure 7-19). En dehors de ces deux points, vous pouvez poursuivre avec le graphique en l'état.

Figure 7-16 *Graphiques en étoile avec libellés et clé explicative*

Figure 7-17 *Graphiques en étoile limités à la moitié supérieure du cercle*

Figure 7-18 *Criminalité illustrée sous forme de graphiques de Nightingale*

Figure 7-19 *Séries de graphiques en étoile illustrant la criminalité par État*

Exécution en parallèle

Même si les graphiques en étoile et les visages de Chernoff facilitent la détection des unités différentes du reste du pack, l'identification de groupes ou de relations entre variables peut être un défi. Les coordonnées parallèles, inventées en 1885 par Maurice d'Ocagne, peuvent y aider.

Comme illustré à la figure 7-20, vous placez plusieurs axes parallèles les uns aux autres. Le haut de chaque axe représente le maximum d'une variable, et le bas son minimum. Pour chaque unité, une ligne est tracée de la gauche vers la droite, montante ou descendante, selon les valeurs de l'unité.

Figure 7-20 *Structure à coordonnées parallèles*

Par exemple, imaginez que vous créiez un tracé à l'aide des données précédentes raltives au basketball. Pour des raisons de simplicité, vous ne tracez que les points, les rebonds et les fautes personnelles, dans cet ordre. Maintenant, imaginez qu'un joueur ait un score élevé, qu'il soit faible au rebond et qu'il ait commis beaucoup de fautes. La ligne tracée pour ce joueur selon le principe des coordonnées parallèles s'élèverait, descendrait et remonterait à nouveau.

Lorsque vous tracez plusieurs unités, la méthode aide à identifier les groupes et les tendances. Dans l'exemple suivant, vous pouvez appliquer les coordonnées parallèles sur les données extraites du National Center for Education Statistics.

Créer un tracé de coordonnées parallèles

Il existe plusieurs options interactives pour les coordonnées parallèles. Vous pouvez les créer dans D3 pour obtenir un graphique personnalisé, ou associer vos données à un outil exploratoire tel que GGobi (*http://ggobi.org*, le téléchargement est gratuit). Ces implémentations vous permettent de filtrer et de privilégier les points de données qui vous intéressent. Pour ma part, je continue à aimer les tracés de coordonnées parallèles statiques, notamment parce qu'ils permettent de comparer simultanément différents filtres. Avec les versions interactives, vous n'avez qu'un seul tracé et il est difficile de bien comprendre ce que l'on regarde quand on a sous les yeux de multiples données mises en évidence.

Vous connaissez la première étape. Avant de procéder à la moindre visualisation, vous avez besoin de données. Chargez les données relatives à l'enseignement dans R avec `read.csv()`.

```
education <- read.csv("http://datasets.flowingdata.com/education.csv",
header=TRUE)
education[1:10,]
```

Il y a sept colonnes. La première concerne le nom de l'État, y compris « États-Unis » pour la moyenne nationale. Les trois suivantes correspondent aux notes moyennes du SAT (test d'entrée à l'université) obtenues en lecture critique, mathématiques et aptitude à la rédaction. La cinquième colonne correspond au pourcentage d'étudiants qui passent le baccalauréat, tandis que les deux dernières colonnes expriment le rapport étudiants/enseignants et le taux d'étudiants qui abandonnent leurs études. Nous cherchons ici à savoir si certaines de ces variables sont reliées et s'il existe des regroupements évidents. Par exemple, les États présentant des taux d'abandon d'études élevés tendent à avoir en moyenne de mauvais résultats au SAT.

La répartition de base de R n'offre pas un moyen simple d'utiliser les coordonnées parallèles, contrairement au package `lattice`. Chargez donc ce package (ou installez-le) afin de poursuivre.

```
library(lattice)
```

Notre tâche va être bien plus facile maintenant. Le package `lattice` propose la fonction `parallel()` que vous pouvez utiliser immédiatement.

```
parallel(education)
```

Vous obtenez le tracé de la figure, ce qui est relativement inutile, j'en conviens. Nous avons tout un tas de lignes et les variables se lisent de haut en bas, au lieu de gauche à droite. On dirait un peu des spaghettis aux couleurs de l'arc-en-ciel.

Figure 7-21 *Tracé des coordonnées parallèles, par défaut avec le package lattice*

Comment modifier ce tracé afin de pouvoir réellement en extraire des informations ? Pour commencer, basculez-le sur le côté. Il s'agit plus d'une préférence personnelle que d'une règle, mais les coordonnées parallèles ont plus de sens quand elles se lisent de gauche à droite, comme vous pouvez le voir à la figure 7-22.

```
parallel(education, horizontal.axis=FALSE)
```

Vous n'avez pas besoin d'inclure la colonne `state`, car il s'agit d'une catégorie et de plus, chaque État possède un nom différent. Modifiez à présent la couleur des lignes en noir – je n'ai rien contre la couleur, mais, là, il y en a trop. Exécutez la ligne de code suivante, vous obtenez alors le graphique de la figure 7-23.

```
parallel(education[,2:7], horizontal.axis=FALSE, col="#000000")
```

Figure 7-22 *Coordonnées parallèles horizontales*

Figure 7-23 *Coordonnées parallèles simplifiées*

Le résultat est meilleur. Les lignes de lecture critique, de mathématiques et d'aptitude à la rédaction s'entrecroisent à peine et courent presque de façon parallèle. Cela signifie que les États ayant un score élevé en lecture ont aussi des scores élevés en mathématiques et en aptitude à la rédaction. De même, les États avec des scores faibles en lecture tendent à avoir des notes basses en mathématiques et à l'écrit.

Il se produit ensuite quelque chose d'intéressant quand vous passez des résultats du SAT au pourcentage d'élèves qui s'y présentent. Il semble que les États avec des résultats élevés tendent à avoir un pourcentage inférieur d'étudiants qui passent le test. C'est la situation inverse des États ayant des résultats au SAT plus bas. Je ne suis pas spécialiste de l'enseignement, mais je parierai que, dans certains États, tout le monde passe le SAT, tandis que dans d'autres États, les individus ne passent pas le test s'ils n'envisagent pas d'aller à l'université. Ainsi, la moyenne baisse quand vous demandez à des personnes qui ne se soucient pas du résultat de passer le test.

Ce point peut être rendu encore plus évident à l'aide de couleurs de contraste. La fonction `parallel()` offre un contrôle complet sur les couleurs grâce à l'argument `col`. Précédemment, vous n'aviez utilisé qu'une seule couleur (`#000000`), mais vous pouvez aussi recourir à un tableau de couleurs, avec une valeur de couleur pour chaque ligne de données. Maintenant, définissez les États situés dans le 50e percentile des scores de lecture en noir et la moitié inférieure en gris. Utilisez `summary()`pour déterminer les valeurs médianes. Entrez simplement `summary(education)` dans la console. Vous obtenez ainsi les statistiques récapitulatives de toutes les colonnes, mais il s'agit d'un moyen rapide de découvrir que la valeur médiane pour l'épreuve de lecture est 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
```

À présent, parcourez chaque ligne de données, vérifiez si elle se situe au-dessus ou en dessous, et spécifiez les couleurs en conséquence. La directive `c()` crée un vecteur vide, que vous incrémentez à chaque itération.

```
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)
}
```

Transmettez ensuite le tableau `reading_colors` en parallèle, en lieu et place de la seule valeur `#000000`. Vous obtenez alors le graphique de la figure 7-24, où il est beaucoup plus aisé de voir les pics et les creux.

```
parallel(education[,2:7], horizontal.axis=FALSE, col=reading_colors)
```

Figure 7-24 *États avec les meilleurs scores à l'épreuve de lecture mis en évidence*

Qu'en est-il des taux d'abandon ? Que se passe-t-il si vous procédez avec les taux d'abandon comme avec les résultats en lecture, si ce n'est que vous utilisez le

troisième quartile à la place de la valeur médiane ? Le quartile est égal à 5,3 %. Une fois encore, vous parcourez chaque ligne de données, mais cette fois en vous intéressant au taux d'abandon, et pas au résultat à l'épreuve de lecture.

```
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)
```

La figure 7-25 illustre le résultat obtenu, lequel n'est pas aussi convaincant que le graphique précédent. Visuellement parlant, il n'y a aucun regroupement manifeste à travers l'ensemble des variables.

Figure 7-25 *États avec les plus hauts taux d'abandon mis en évidence*

Vous pouvez aller encore plus loin en explorant par vous-même. Revenez à présent à la figure 7-24 et améliorez-la. Il serait souhaitable que les libellés soient mieux présentés et plus clairs. Peut-être serait-il préférable d'ajouter un peu

de couleur à la place de ce dégradé de gris ? Et pourquoi pas un bref texte de présentation expliquant pourquoi la moitié supérieure des États est mise en évidence ? Le résultat obtenu est représenté à la figure 7-26.

Figure 7-26 *Tracé de coordonnées parallèles autonomes sur les résultats du SAT*

Réduction des dimensions

Lorsque vous utilisez les visages de Chernoff ou les coordonnées parallèles, l'objectif principal est de parvenir à réduire les dimensions. Vous voulez trouver des groupes au sein de l'ensemble de données ou de la population. La difficulté est que vous ne savez pas toujours si vous devez commencer par regarder les visages OU les lignes continues. Par conséquent, il serait appréciable que vous puissiez regrouper les objets, à partir de différents critères. Tel est l'un des objectifs de la méthode de l'échelonnement multidimensionnel (MDS). Prenez tous les éléments en compte et placez à proximité sur le tracé les unités les plus similaires.

Comme l'attestent les nombreux ouvrages consacrés au sujet, les explications peuvent être assez techniques ; aussi, pour des raisons de simplicité, je

me consacrerai à l'essentiel en reportant à une autre occasion l'aspect mathématique. Cela dit, MDS est l'une des premières méthodes qui m'aient été enseignées quand j'étais étudiant et elle mérite d'être examinée de très près, si le cœur vous en dit.

Imaginez que vous vous trouviez dans une salle carrée et vide, en présence de deux autres personnes. Il vous appartient de dire à celles-ci où elles doivent se tenir dans la pièce, en fonction de leur taille. Plus leurs tailles sont proches, plus les deux personnes doivent se rapprocher et inversement, plus leurs tailles sont différentes, plus elles doivent se tenir à distance l'une de l'autre. L'une des personnes est vraiment petite et l'autre réellement grande. Où doivent-elles se mettre ? Les deux personnes doivent se tenir dans des coins opposés, car elles ont des tailles totalement différentes l'une de l'autre.

Une troisième personne se présente ensuite, de taille moyenne. Si l'on s'en tient au modèle d'organisation retenu, la personne doit se tenir au centre de la salle, entre les deux autres personnes, à égale distance de chacune d'elles. Dans le même temps, les deux premières personnes demeurent à une distance maximale l'une de l'autre.

Introduisons à présent une autre variable : le poids. Nous connaissons la taille et le poids des trois personnes. La personne de petite taille et celle de taille moyenne font exactement le même poids, tandis que la personne de grande taille est environ un tiers plus lourde. Comment disposer les trois personnes dans la pièce, en fonction de leur taille et de leur poids ? Si nous maintenons les deux premières (la petite et la grande) à leurs emplacements opposés respectifs, la troisième personne doit se trouver plus proche de la plus petite, car leurs poids sont identiques.

Comprenez-vous ce qui se passe ? Plus deux personnes sont semblables, plus elles doivent se tenir près l'une de l'autre. Dans ce cas simple, vous n'avez que trois personnes et deux variables, le problème est donc facile à résoudre manuellement. Mais imaginez que vous ayez 50 personnes et que vous deviez les placer dans une pièce selon cinq critères. Ce serait plus délicat. Et c'est à cela que sert l'échelonnement multidimensionnel.

Pour plus d'informations sur la méthode, effectuez une recherche sur Internet en saisissant les mots-clés « échelonnement multidimensionnel » ou « analyse des composants principaux ».

Utilisation de l'échelonnement multidimensionnel

L'échelonnement multidimensionnel sera beaucoup plus aisé à comprendre avec un exemple concret. Reprenons les données sur l'enseignement ; si vous ne les avez pas déjà chargées dans R, commencez par cela.

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

Souvenez-vous : il y a une ligne par État, District of Columbia inclus, et d'autres lignes pour les moyennes des États-Unis. Il y a six variables pour chaque État :

scores aux épreuves de lecture, de mathématiques et d'aptitude à la rédaction du SAT ; pourcentage d'étudiants ayant passé le SAT, rapport élèves/enseignants et taux d'abandon.

Le principe est ici le même que celui appliqué à la métaphore de la chambre, mais au lieu d'une pièce carrée, il s'agit d'un graphique carré. Les personnes sont remplacées par les États et les variables de taille et de poids, sont remplacées par des mesures relatives à l'éducation. L'objectif est le même. Vous voulez placer les États sur un graphique XY, de telle sorte que les États similaires soient les plus proches les uns des autres.

Première étape : découvrir à quelle distance chaque État doit se trouver par rapport aux autres États. Pour cela, vous allez recourir à la fonction `dist()`. Vous n'utilisez que les colonnes 2 à 7, car la première colonne correspond aux noms des États, et que ceux-ci sont tous différents les uns des autres.

```
ed.dis <- dist(education[,2:7])
```

Si vous tapez `ed.dis` dans la console, vous voyez une série de matrices. Chaque cellule représente la distance à laquelle un État doit se trouver par rapport à un autre (distance euclidienne en pixels). Par exemple, la valeur de la deuxième ligne, deuxième colonne correspond à la distance qui doit exister entre l'Alabama et l'Alaska. Les unités ne sont pas si importantes à ce stade. Ce sont plutôt les différences relatives qui importent.

Comment représenter ceci sur une matrice de 51 × 51 sur un graphique XY ? Vous ne pouvez pas encore, à moins que vous n'ayez les coordonnées XY de chaque État. Tel est le but de la fonction `cmdscale()` qui accepte en entrée une matrice de distances et retourne un ensemble de points de telle sorte que les différences entre ces points soient identiques à celles spécifiées dans la matrice.

```
ed.mds <- cmdscale(ed.dis)
```

Tapez `ed.mds` dans la console. Comme vous pouvez le constater, vous disposez maintenant des coordonnées XY pour chaque ligne de données. Stockez-les dans les variables `x` et `y`, et passez-les à `plot()` pour voir le résultat obtenu (figure 7-27).

```
x <- ed.mds[,1]
y <- ed.mds[,2]
plot(x,y)
```

C'est assez satisfaisant, chaque point correspond à un État. Toutefois, il y a un problème dans la mesure où vous ne savez pas de quel État il s'agit. Comme vous avez besoin de libellés, utilisez `text()` pour remplacer les points par les noms des États (figure 7-28).

```
plot(x, y, type="n")
text(x, y, labels=education$state)
```

Figure 7-27 *Graphique en points illustrant les résultats de l'échelonnement multidimensionnel*

Figure 7-28 *Utilisation des noms des États à la place des points pour visualiser l'emplacement de chaque État*

Vous constatez que deux regroupements émergent, l'un à gauche et l'autre à droite. Le point correspondant aux États-Unis se trouve en bas du regroupement droit, vers le milieu, ce qui semble correct. À ce stade, il vous appartient d'établir ce que les regroupements signifient.

Vous pourriez, par exemple, colorier les États avec `dropout_colors` comme vous l'avez fait pour les coordonnées parallèles (figure 7-29). Mais cela ne vous apprend pas grand-chose de plus, cela confirme simplement ce que vous avez observé à la figure 7-25.

Figure 7-29 *États coloriés en fonction des taux d'abandon*

Que se passerait-il si nous colorions les États en fonction des résultats à l'épreuve de lecture ? Bien sûr, il est possible de procéder ainsi (figure 7-30). Il semble qu'un modèle clair se dégage ici. Les notes supérieures se retrouvent-elles à gauche et les notes inférieures à droite ? Qu'est-ce qui différencie Washington des autres États ? Réfléchissez à ces questions et vous pourrez me répondre plus tard.

Figure 7-30 *États coloriés en fonction des résultats à l'épreuve de lecture*

Si vous voulez faire preuve de fantaisie, vous pouvez essayer ce que l'on appelle le clustering à base de modèle. Je ne vais pas rentrer dans les détails, mais juste vous montrer comment procéder (croyez-moi, il n'y aura aucune part de magie). Quelques mathématiques sont néanmoins nécessaires. Pour résumer, utilisez le package `mclust` pour identifier les regroupements de votre tracé MDS. Installez `mclust` si besoin, puis exécutez le code suivant pour les graphiques de la figure 7-31.

```
library(mclust)
ed.mclust <- Mclust(ed.mds)
plot(ed.mclust, data=ed.mds)
```

Le premier tracé en haut à gauche illustre les résultats obtenus par l'utilisation d'un algorithme de recherche du nombre idéal de clusters dans les données. Les trois autres tracés illustrent les regroupements. Assez impressionnant. Vous avez les deux regroupements, bien mieux définis maintenant et qui représentent les États avec des notes élevées et les États avec des notes basses.

Figure 7-31 *Résultats du clustering à base de modèle*

Nous arrivons au moment où, normalement, je vous demande d'importer vos fichiers PDF dans Illustrator et de leur appliquer quelques retouches ; mais je ne suis pas certain que je destinerais ces graphiques au grand public. Ils sont trop abstraits pour qu'une personne insuffisamment formée sur le plan technique puisse comprendre de quoi il retourne. Ils sont adaptés à l'exploration des données ; cependant, si vous y êtes enclin, tous les principes standards de design s'appliquent. Déterminez ce dont vous avez besoin pour que votre propos soit clair et supprimez le reste.

Recherche des observations aberrantes

Plutôt que d'examiner comment des unités de données appartiennent à certains groupes, vous devriez vous intéresser à l'inverse, à savoir en quoi elles ne font pas partie de tel ou tel groupe. Plus précisément, il existe souvent des points de données qui se démarquent du reste et que l'on appelle, vous l'aurez deviné, observations aberrantes. Il s'agit de points de données qui diffèrent du reste de la population. Parfois, ils peuvent constituer la part la plus intéressante de l'histoire, mais ce peut être aussi une simple erreur typographique avec un zéro en moins, par exemple. Quoi qu'il en soit, vous devez les vérifier pour savoir ce qui se passe. Vous n'allez pas créer un graphique exceptionnel en supposant la présence d'une observation aberrante pour découvrir plus tard, suite à la remarque d'un lecteur attentif, que tout votre travail n'a aucun sens.

Des types de graphiques ont été conçus tout particulièrement pour mettre en évidence les observations aberrantes, mais, d'après mon expérience, rien ne vaut les tracés de base et une dose de bon sens. Enquêtez sur le contexte de vos données, faites votre travail et renseignez-vous auprès des experts en cas de doute sur les données. Une fois que vous avez identifié les observations aberrantes, vous pouvez utiliser les mêmes techniques graphiques que celles employées jusqu'à présent pour les souligner à l'attention des lecteurs : choisissez des couleurs variées, fournissez des pointeurs ou recourez à des bordures plus épaisses.

Étudions maintenant un exemple simple. La figure 7-32 représente un tracé chronologique illustrant les données extraites du site Weather Underground (comme au chapitre 2), de 1980 à 2005. Il y a des cycles saisonniers comme vous pouvez vous en douter, mais que se passe-t-il au milieu ? Les données y semblent inhabituellement lisses (régulières), alors que le reste des données présente un certain bruit. Rien de quoi devenir fou, mais s'il arrive que vous exécutiez des modèles météorologiques sur ces données, vous souhaiterez peut-être connaître d'une part les estimations, et d'autre part les données réelles.

Figure 7-32 *Données météorologiques estimées extraites du site Weather Underground*

De même, si vous observez les graphiques en étoile créés pour illustrer les taux de criminalité, vous remarquez que le District of Columbia se distingue. Vous auriez pu tout aussi bien l'observer avec un graphique en barres élémentaire, comme illustré à la figure 7-33. Est-il raisonnable de comparer Washington, DC aux autres États, sachant que sa composition s'apparente plus à une ville ? Vous êtes le seul juge.

Figure 7-33 *Meurtres par arme à feu aux États-Unis*

Et qu'en est-il des nombres d'abonnés évoqués au chapitre 3 et illustrés à la figure 7-34 ? Vous remarquez la présence d'un creux au milieu, qui semble indiquer la perte de plus de la moitié du lectorat de FlowingData.

Vous pouvez aussi considérer la répartition comme un tout, via un histogramme, tel que celui de la figure 7-35. Tous les nombres sont regroupés à droite, à l'exception d'un ou deux qui se trouvent à gauche, et rien ne figure entre les deux côtés.

Figure 7-34 *Nombres d'abonnés au site FlowingData au fil du temps*

Figure 7-35 *Histogramme illustrant la répartition du nombre d'abonnés*

Plus concrètement, vous pouvez utiliser un diagramme à surfaces, qui représente les quartiles d'une distribution. Les diagrammes à surfaces générés dans R à l'aide de la fonction `boxplot()` peuvent automatiquement mettre en évidence les points qui équivalent respectivement à 1,5 fois plus ou moins les quartiles supérieurs et les quartiles inférieurs (figure 7-36).

Figure 7-36 *Diagramme à surfaces illustrant la répartition des nombres d'abonnés*

Si je n'avais qu'un petit nombre d'abonnés se comptant sur les dix doigts de la main, une diminution si importante en pourcentage serait possible, mais il est peu probable que j'ai tenu des propos si choquants qu'ils aient entraîné la défection de dizaines de milliers de lecteurs (revenus, en outre, quelques jours plus tard). Il est beaucoup plus vraisemblable que Feedburner, le service de gestion de flux que j'utilise, ait fait une erreur de reporting.

Ces observations évidentes au sein de tels ensembles de données coulent de source parce que vous connaissez quelque peu les données. Elles seraient beaucoup moins manifestes si vous utilisiez des jeux de données dont vous ne seriez pas familier. Si tel est le cas, il peut être utile d'accéder directement à la source et de demander à parler à la personne responsable des données. Généralement, celle-ci acceptera volontiers de vous faire part de conseils ou d'avis. Si jamais vous ne pouviez obtenir des informations supplémentaires, à tout le moins vous

aurez essayé et pourrez ajouter une note relative à l'ambiguïté des données dans votre explication du graphique.

Un quartile est l'une des trois valeurs qui divisent les données en quatre parts égales. Le deuxième quartile correspond à la médiane, le quartile supérieur identifie les 25 % des données dont la valeur est supérieure à cette donnée et le quartile inférieur correspond aux 25 % dont la valeur est inférieure.

Pour résumer

Pour les débutants, l'un des aspects les plus difficiles en matière de création de graphiques de données est de savoir par où commencer. Vous vous retrouvez face à une multitude de données sans le moindre indice quant à leur nature ou à ce que vous devez en attendre. Généralement, vous démarrerez par une question sur les données, mais qu'en est-il si vous ne savez pas quoi demander ? Les méthodes décrites dans ce chapitre peuvent être une aide précieuse en la matière. Elles vous permettront de considérer toutes les données dans leur ensemble, ce qui facilitera l'identification de la partie des données que vous devez alors explorer en priorité.

Cependant, ne vous arrêtez pas en si bon chemin. Utilisez ces points comme tremplin pour affiner ceux qui semblent intéressants. Les explications de ce chapitre, en complément de celles vues précédemment, doivent suffire pour vous aider à explorer plus avant vos données, quel que soit leur type. Ceci est valable sauf pour un type, qui fait l'objet du prochain chapitre : les données spatiales. Soyez alors prêts à créer quelques cartes supplémentaires.

Chapitre 8

Visualisation des relations spatiales

Les cartes représentent une sous-catégorie de visualisation qui offre l'avantage d'être incroyablement intuitive. Même enfant, je pouvais les lire. Je me souviens qu'assis à l'arrière de la voiture familiale, je criais les directions tout en lisant l'immense carte dépliée sur mes genoux. Aujourd'hui, une voix féminine artificielle, mais rassurante, énonce les directions à partir d'une petite boîte fixée sur le tableau de bord.

Quoi qu'il en soit, les cartes sont un excellent moyen de comprendre les données. Ce sont des versions réduites du monde réel, et elles sont omniprésentes. Dans ce chapitre, nous allons explorer plusieurs jeux de données en quête de modèles spatiaux et temporels. Nous créerons quelques cartes élémentaires en R et passerons à une cartographie plus avancée avec Python et SVG. Et nous terminerons avec des cartes interactives et animées conçues avec ActionScript et Flash.

Que chercher ?

Les cartes se lisent de la même façon que les graphiques statistiques. Quand vous regardez un emplacement spécifique sur une carte, vous cherchez une agglomération dans une zone donnée ou, par exemple, désirez comparer une région au reste du pays. La différence est qu'au lieu de coordonnées x et y, vous vous occupez de latitude et de longitude. Les coordonnées d'une carte sont de fait reliées l'une à l'autre de la même façon que deux villes entre elles. Le point A et le point B, par exemple, sont éloignés d'un nombre spécifique de kilomètres et il faut un temps donné pour se rendre de l'un à l'autre. Par comparaison, la distance sur un nuages de points est abstraite et (généralement) ne possède pas d'unités.

Cette différence se traduit par un certain nombre de subtilités en termes de cartes et de cartographie. Il existe une raison pour laquelle le *New York Times*

possède au sein de son département graphique quelques personnes travaillant exclusivement à la conception de cartes. Vous devez vous assurer que tous les lieux sont correctement placés, que les couleurs ont un sens, que les libellés ne masquent pas les lieux et que la projection utilisée est la bonne.

Ce chapitre ne traite que de quelques aspects des notions élémentaires. Bien sûr, elles vous permettront d'être relativement exhaustif en termes de narration, mais n'oubliez pas que vous attend, si vous le désirez, un niveau bien plus élaboré que vous vous efforcerez peut-être d'atteindre.

La situation peut devenir particulièrement intéressante si vous introduisez la notion de temps. Une carte représente une tranche de temps, mais il est possible de représenter différentes tranches de temps à l'aide de plusieurs cartes. Vous pouvez aussi animer les modifications pour observer le développement ou le déclin d'une activité dans une région géographique, par exemple. Les essors de zones spécifiques deviennent manifestes et, si la carte est interactive, le lecteur peut aisément se concentrer sur un secteur et évaluer son degré d'évolution. Vous n'obtenez pas le même résultat avec les graphiques en barres ou les nuages de points, mais, avec les cartes, les données peuvent devenir instantanément personnelles.

Emplacements spécifiques

Une liste de lieux constitue le type le plus simple de données spatiales que vous puissiez rencontrer. Vous disposez de la latitude et de la longitude pour tout un tas de lieux que vous désirez cartographier. Peut-être désirez-vous montrer où certains événements, tels que crimes ou autres, se sont produits, ou rechercher les zones où les points sont particulièrement rassemblés. Ceci est simple à faire, et il existe de multiples façons d'y parvenir.

Sur le Web, le moyen le plus aisé de cartographier des points consiste à utiliser Google ou Microsoft Maps. Grâce à leurs API de cartographie, vous disposez d'une carte active sur laquelle vous pouvez effectuer un zoom avant ou arrière en un rien de temps, avec quelques lignes à peine en JavaScript. Vous trouverez sur Internet des multitudes de didacticiels et une excellente documentation sur l'emploi de ces API.

Il y a un inconvénient, toutefois. La personnalisation des cartes se heurte à certaines limites et, à la fin, vous vous retrouvez presque invariablement avec quelque chose qui ressemble à une carte Google ou une carte Microsoft. Je ne dis pas qu'elles sont laides, mais, lorsque vous développez une application ou concevez un graphique destiné à une publication, il est préférable d'avoir une carte correspondant à votre schéma de conception. Il est possible de contourner cet obstacle, mais l'effort ne se justifie guère si vous pouvez obtenir le même résultat, et en mieux, avec un autre outil.

Recherche de la latitude et de la longitude

Avant toute chose, considérez les données disponibles et celles dont vous avez réellement besoin. Si vous n'avez pas les données nécessaires, vous n'aurez rien à visualiser, n'est-ce pas ? Dans la plupart des applications pratiques, vous avez besoin des données de latitude et de longitude pour cartographier les points, et la plupart des jeux de données ne se présentent pas ainsi, mais plus généralement sous forme d'une liste d'adresses.

Pour autant que vous le vouliez, vous ne pouvez pas juste associer noms de rue et codes postaux et espérer en retour une jolie carte. Il faut d'abord que vous ayez la latitude et la longitude, et à cette fin, que vous vous tourniez vers le *géocodage*. En gros, vous avez une adresse, la confiez à un service, lequel interroge sa base de données pour obtenir l'adresse correspondante, et vous récupérez ainsi la latitude et la longitude de l'adresse recherchée.

Quant à savoir quel service utiliser, il en existe plusieurs. Si vous ne souhaitez appliquer le géocodage qu'à quelques emplacements, il est aisé de se rendre sur un site web et d'entrer les coordonnées manuellement – Geocoder.us constitue un bon choix en ce sens. Si vous n'avez pas besoin que les emplacements soient exactement situés, vous pouvez essayer *Google Maps Latitude Longitude Popup* de Pierre Gorissen. Il s'agit d'une simple interface Google Maps qui communique la latitude et longitude de n'importe quel emplacement de la carte sur lequel vous cliquez.

Si, toutefois, vous avez de multiples emplacements à géocoder, vous devriez recourir à la programmation. Inutile que vous perdiez votre temps à copier et à coller. Google, Yahoo!, Geocoder.us et Mediawiki proposent tous des API de géocodage ; et Geopy, une boîte à outils de géocodage pour Python, les regroupe toutes dans un seul et même package.

Google et Microsoft proposent des didacticiels extrêmement simples pour démarrer à l'aide de leurs API de cartographie ; aussi pensez à bien les consulter si vous souhaitez tirer parti des quelques fonctionnalités élémentaires de cartographie.

Outils de géocodage

Geocoder.us, *http://geocoder.us* – Fournit une interface simple pour copier et coller un emplacement et obtenir ainsi sa latitude et sa longitude. Propose aussi une API.

Latitude Longitude Popup, *www.gorissen.info/Pierre/maps/* – Mélange de Google Maps. Cliquez sur un emplacement de la carte pour connaître sa latitude et sa longitude.

Geopy, *http://code.google.com/p/geopy/* – Outil de géocodage pour Python. Inclut plusieurs API de géocodage en un seul package.

Visitez la page Geopy pour obtenir des instructions sur l'installation du package. Vous y trouverez également de nombreux exemples simples sur la façon de démarrer. L'exemple suivant présume que vous avez déjà installé le package sur votre ordinateur.

Une fois que vous avez installé Geopy, téléchargez les données à l'adresse *http://book.flowingdata.com/ch08/geocode/costcos-limited.csv*. Il s'agit d'un fichier CSV contenant l'adresse de chaque entrepôt Costco aux États-Unis, mais sans les coordonnées de latitude ou de longitude. Il vous appartient de les obtenir.

Ouvrez un nouveau fichier et enregistrez-le sous le nom geocode-locations.py. Comme à l'ordinaire, importez les packages dont vous avez besoin pour le reste du script.

```
from geopy import geocoders
import csv
```

Vous avez aussi besoin d'une clé API pour chaque service que vous voulez utiliser. Dans le cadre de cet exemple, vous n'avez besoin que de celle de Google. Stockez la clé API dans une variable nommée `g_api_key`, puis utilisez-la quand vous instanciez le géocodeur.

```
g_api_key = 'INSERT_YOUR_API_KEY_HERE'
g = geocoders.Google(g_api_key)
```

Visitez le site *http://code.google.com/apis/maps/signup.html* pour vous inscrire à l'obtention d'une clé gratuite pour l'API Google Maps. L'inscription est simple et ne nécessite que quelques minutes.

Chargez le fichier de données `costcos-limited.csv`, et exécutez une boucle. Pour chaque ligne, vous reconstituez l'adresse complète et la connectez en vue de son géocodage.

```
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)
```

Exécutez le script Python et enregistrez la sortie sous le nom costcos-geocoded .csv. Les toutes premières lignes des données se présentent de la façon suivante :

```
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
...
```

Par chance, les coordonnées en latitude et longitude sont disponibles pour chaque adresse. Généralement, tel n'est pas le cas. Si vous vous heurtez réellement

à ce problème, vous devez ajouter un contrôle d'erreur à la dernière ligne du précédent script.

```
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"
```

Le code essaie de trouver la latitude et la longitude, et s'il échoue, imprime l'adresse avec les coordonnées ayant comme valeurs « null ». Exécutez le script Python, enregistrez la sortie dans un fichier, puis revenez en arrière et recherchez les valeurs « null ». Pour les adresses manquantes, vous pouvez essayer un autre service via Geopy, ou simplement entrer vous-même les adresses dans Geocoder.us.

Des points, simplement

Maintenant que vous disposez de points avec la latitude et la longitude, vous pouvez les cartographier. La voie la plus simple à suivre est l'équivalent informatique du placement de punaises sur une carte en papier fixée sur un panneau. Comme vous pouvez le voir à la figure 8-1, vous placez un repère sur la carte pour chaque emplacement.

Figure 8-1 *Cartographie des points*

Même si le concept est simple, vous pouvez identifier certaines caractéristiques des données, comme le regroupement, la dispersion et les cas aberrants.

Carte avec points

R, bien que limité en termes de fonctionnalité de cartographie, permet de placer facilement des points sur une carte. Le package `maps` effectue l'essentiel du travail. Installez-le via Package Installer ou utilisez `install.packages()` dans la console. Une fois le package installé, chargez-le dans l'espace de travail.

```
library(maps)
```

Étape suivante : chargement des données. Sentez-vous libre d'utiliser les emplacements Costco que vous venez juste de géocoder ; pour des raisons pratiques, j'ai placé le jeu de données traité en ligne et, par conséquent, vous pouvez le charger directement depuis l'URL.

```
costcos <-
  read.csv("http://book.flowingdata.com/ch08/geocode/costcos-
  geocoded.csv", sep=",")
```

Passons à la cartographie. Lorsque vous créez une carte, il est utile de la considérer comme un calque (indépendamment du logiciel utilisé). Le calque inférieur correspond généralement à la carte de base qui représente les frontières géographiques, puis, par-dessus, vous placez les calques des données. Dans le cas présent, le calque inférieur est une carte des États-Unis, et le second calque correspond aux emplacements Costco. Voici comment créer le premier calque, tel qu'illustré à la figure 8-2.

```
map(database="state")
```

Figure 8-2 *Carte simple des États-Unis*

Le second calque, celui de Costco, est ensuite cartographié à l'aide de la fonction `symbols()`. Il s'agit de la même fonction que celle que vous avez utilisée pour créer les graphiques en bulles du chapitre 6 ; vous l'employez de la même façon, à la différence que vous passez la latitude et la longitude au lieu des coordonnées *x* et *y*. De même, définissez `add` avec la valeur `TRUE` pour préciser que vous voulez ajouter les symboles à la carte, et non pas créer un nouveau graphique.

```
symbols(costcos$Longitude, costcos$Latitude,
  circles=rep(1, length(costcos$Longitude)), inches=0.05, add=TRUE)
```

La figure 8-3 illustre les résultats. Tous les cercles sont de la même taille, parce vous définissez `circles` comme tableau dont la longueur est égale au nombre d'emplacements. Vous définissez également `inches` avec la valeur 0,05, qui correspond à la dimension des cercles. Pour obtenir des repères plus petits, il suffit de diminuer cette valeur.

Figure 8-3 *Carte des emplacements Costco*

Comme précédemment, vous pouvez modifier les couleurs de la carte et des cercles de telle sorte que les emplacements se détachent et que les lignes des frontières reposent à l'arrière-plan, comme présenté à la figure 8-4. Maintenant, coloriez les points en un joli rouge Costco et les frontières des États en gris clair.

```
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)
```

Figure 8-4 *Utilisation de la couleur avec les emplacements cartographiés*

Dans la figure 8-3, les cercles vides et la carte offraient la même couleur et la même épaisseur de trait, si bien que tout semblait se confondre. Avec les couleurs appropriées, il devient possible de faire ressortir les données à l'avant.

Le résultat est plutôt positif pour quelques lignes de code. Costco a clairement privilégié l'ouverture d'emplacements sur le littoral avec des concentrations en Californie, dans la partie nord-ouest de l'État de Washington, et au nord-est du pays.

Figure 8-5 *Carte mondiale des emplacements Costco*

Cependant, il y a une omission flagrante ici. En vérité, deux omissions. Où sont les États de l'Alaska et d'Hawaï ? Ils font partie des États-Unis, aussi, mais ne peuvent être trouvés nulle part même si vous utilisez la base de données « state » avec `map()`. Comme les deux États figurent bel et bien dans la base de données « world », pour visualiser les emplacements Costco en Alaska et à Hawaï, vous devez cartographier l'univers entier, comme illustré à la figure 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)
```

C'est une perte d'espace, je sais. Il existe des options avec lesquelles vous pouvez « bidouiller », et que vous trouverez dans la documentation, mais vous pouvez effectuer les modifications dans Illustrator pour zoomer sur les États-Unis ou retirer de la carte les autres pays.

Dans R, en cas de doute, pour accéder à la documentation sur une fonction ou un package, faites précéder son nom d'un point d'interrogation.

Abordons la carte dans la direction opposée et imaginons que vous ne souhaitiez représenter les emplacements Costco que pour quelques États. Utilisez à cette fin l'argument `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)
```

Comme le montre la figure 8-6, vous créez un calque inférieur avec la Californie, le Nevada, l'Oregon et l'État de Washington. Puis, vous lui superposez le calque des données. Certains points ne sont dans aucun de ces États, mais continuent d'apparaître. Une fois encore, rien de plus simple que de les supprimer à l'aide de votre logiciel de retouche préféré.

Carte avec lignes

Dans certains cas, il peut être utile de connecter les points de la carte, si leur ordre offre une quelconque pertinence. Grâce aux services de localisation en ligne, tels que Foursquare dont la popularité ne cesse de croître, les suivis de déplacement sont fréquents. Un moyen simple de tracer des lignes consiste à utiliser la fonction `lines()`. À titre d'illustration, regardez les lieux que j'ai parcourus en tant qu'espion du gouvernement fictif de Fakesville. Commencez, comme à l'ordinaire, par charger les données et par dessiner une carte du monde élémentaire.

```
faketrace <-
   read.csv("http://book.flowingdata.com/ch08/points/fake-
     trace.txt",sep="\t")
map(database="world", col="#cccccc")
```

Examinez le tableau de données en entrant `faketrace` dans la console R. Vous remarquez que le tableau se compose de deux colonnes, l'une pour la latitude et l'autre pour la longitude, et de huit points de données. Vous pouvez considérer que les points se trouvent déjà dans l'ordre de mes voyages durant ces sept longues journées.

```
   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
```

Figure 8-6 *Emplacements Costco dans les États sélectionnés*

Tracez les lignes en connectant simplement les deux colonnes avec `lines()`. Spécifiez aussi la couleur (`col`) et l'épaisseur de la ligne (`lwd`).

```
lines(faketrace$longitude, faketrace$latitude, col="#bb4cd4", lwd=2)
```

Maintenant ajoutez les points, exactement comme vous le feriez avec les emplacements Costco (figure 8-7).

```
symbols(faketrace$longitude, faketrace$latitude, lwd=1,
➥bg="#bb4cd4", fg="#ffffff", circles=rep(1,
length(faketrace$longitude)),
➥inches=0.05,
add=TRUE)
```


Figure 8-7 *Dessin d'un suivi d'emplacement*

Après sept jours et sept nuits passés comme espion au service du gouvernement de Fakesville, j'ai décidé que cette mission n'était pas pour moi. Elle était loin d'être aussi glamour que celle de James Bond. Cependant, j'ai bel et bien créé des liens dans tous les pays visités. Il pourrait être intéressant de tracer des lignes entre mon emplacement et tous les autres, comme illustré à la figure 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)
}
```

Une fois que vous avez créé la carte de base, parcourez chaque point et tracez une ligne entre le dernier point du tableau de données et tous les autres emplacements. Le résultat ne fournit pas beaucoup d'informations, mais peut-être en

trouverez-vous une utilisation judicieuse. Le fait est ici que vous pouvez tracer une carte, puis employer les autres fonctions graphiques de R pour représenter ce que vous souhaitez à l'aide des coordonnées de latitude et de longitude. (À propos, je n'ai jamais été espion pour Fakesville. Je plaisantais bien sûr.)

Figure 8-8 *Tracé de connexions à travers le monde*

Points à l'échelle

Si l'on revient à des données réelles et plus intéressantes que mes activités d'espion fictif, il arrivera très souvent que vous n'ayez pas seulement des emplacements. Il existe d'autres valeurs associées aux emplacements comme les ventes liées à une activité ou à une entreprise, ou la population d'une ville. Vous pouvez toujours élaborer une carte à partir de points, mais vous pouvez aussi reprendre le principe du graphique en bulles et l'appliquer à une carte. Je pense que je n'ai pas à expliquer de nouveau que la dimension des bulles doit être déterminée en fonction de l'aire et non pas du rayon, n'est-ce pas ?

Carte avec bulles

Dans cet exemple, nous allons nous intéresser au taux de natalité chez les adolescentes, tel qu'établi dans le Rapport des Nations Unies sur le développement humain – à savoir, le nombre de naissances pour 1 000 femmes âgées de 15 à 19 ans en 2008. Les géocoordonnées ont été fournies par GeoCommons. Le but est de dimensionner les bulles en fonction des taux.

Le code est pratiquement identique à celui utilisé pour les emplacements Costco, mais souvenez-vous que vous aviez alors transmis un vecteur comme

taille des cercles à la fonction `symbols()`. Cette fois, pour la taille, nous utiliserons la fonction `sqrt()`.

```
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")
```

Figure 8-9 *Taux de fécondité chez les adolescentes à travers le monde*

La figure 8-9 illustre le résultat. Aussitôt vous voyez que les taux de fécondité chez les adolescentes sont les plus élevés dans les pays africains, tandis que les pays européens ont des taux relativement bas. Au vu du seul graphique, il n'est pas évident de déduire la valeur de chaque cercle, car il n'y a aucune légende. Un rapide examen avec `summary()` dans R permet d'en apprendre un peu plus.

```
summary(fertility$ad_fert_rate)

    Min.  1st Qu.  Median   Mean   3rd Qu.    Max.   NA's
    3.20   16.20    39.00  52.89    78.20   201.40   1.00
```

Parfait pour nous, mais vous devrez nécessairement fournir des explications supplémentaires si vous désirez que celles et ceux qui n'ont pas eu connaissance des données comprennent le graphique. Vous pouvez ajouter des annotations pour mettre en évidence les pays avec les taux de fécondité les plus élevés et les plus bas, indiquer le pays d'où proviennent les lecteurs (ici, les États-Unis)

et proposer une introduction pour préparer les lecteurs à ce qu'ils vont regarder. La figure 8-10 illustre ces modifications.

Figure 8-10 *Taux mieux explicités pour une audience plus large*

Régions

La cartographie de points ne saurait vous conduire au-delà, car ils ne correspondent qu'à des emplacements uniques. Les départements, les provinces, les États et les continents sont des régions à part entière avec des frontières et les données géographiques sont généralement regroupées ainsi. Par exemple, il est beaucoup plus aisé de retrouver les données sanitaires d'un État ou d'un pays que celles d'un patient ou d'un hôpital. Cela s'explique généralement pour des raisons de confidentialité, alors que, parfois, il est plus simple de distribuer les données globales. Quoi qu'il en soit, comme c'est généralement ainsi que vous allez utiliser les données spatiales, il est temps d'apprendre à les visualiser.

Couleur par données

Les cartes choroplèthes (ou cartes planes) sont la solution la plus commune pour représenter les données régionales. Les régions y sont coloriées suivant

une gamme de couleurs que vous définissez vous-même, comme on le voit à la figure 8-11. Comme les zones et les emplacements sont déjà définis, il ne vous reste qu'à déterminer les gammes de couleur à utiliser.

Figure 8-11 *Structure d'une carte choroplèthe*

Comme évoqué dans un précédent chapitre, ColorBrewer de Cynthia Brewer constitue un excellent moyen de sélectionner des couleurs, ou à tout le moins de se créer une palette de couleurs. Si vos données sont continues, vous souhaiterez peut-être une gamme s'échelonnant du clair au sombre, mais avec la même teinte (ou plusieurs teintes similaires), comme illustré à la figure 8-12. Un autre jeu de couleurs, tel que celui de la figure 8-13, peut convenir si vos données sont « doubles », à savoir, par exemple, qu'elles peuvent présenter un aspect positif ou un aspect négatif, ou se situer au-dessus ou sous un seuil donné.

Enfin, si vos données sont d'ordre qualitatif avec des classes et des catégories, peut-être souhaiterez-vous une couleur unique pour chacune d'elles (figure 8-14).

Une fois que vous avez votre jeu de couleurs, vous avez encore deux choses à faire. La première est de décider comment les couleurs que vous avez sélec-

tionnées concordent avec la plage de données, et la seconde est d'attribuer des couleurs à chaque région en fonction de votre choix. Vous accomplirez les deux avec Python et SVG (*Scalable Vector Graphics*) dans les exemples suivants.

Figure 8-12 *Jeu de couleurs séquentielles avec ColorBrewer*

Figure 8-13 *Jeu de couleurs divergentes avec ColorBrewer*

Figure 8-14 *Jeu de couleurs qualitatives avec ColorBrewer*

Carte des comtés

Chaque mois, l'U.S. Bureau of Labor Statistics fournit les données du chômage au niveau comté. Vous pouvez télécharger les taux les plus récents ou remonter plusieurs années en arrière à partir du site. Cependant, le navigateur proposé est quelque peu obsolète et alambiqué ; aussi, pour des raisons de simplicité, et au cas où le site changerait, vous pouvez télécharger les données à l'adresse *http://book.flowingdata.com/ch08/regions/unemployment-aug2010.txt*. Les données se composent de six colonnes. La première est un code spécifique au Bureau of Labor Statistics. Les deux suivantes forment ensemble l'identifiant unique de chaque comté. Les quatrième et cinquième colonnes correspondent respectivement au nom du comté et au mois de l'estimation du taux. La dernière colonne représente le pourcentage estimé du nombre de personnes sans emploi dans le comté. Dans le cadre de cet exemple, vous ne vous préoccuperez que de l'identifiant du comté (à savoir, les codes FIPS) et du taux.

Passons à la carte. Dans les exemples précédents, vous avez généré les cartes de base en R, mais, désormais, vous pouvez utiliser à la place Python et SVG. La première application permet de traiter les données et la seconde la carte elle-même. Néanmoins, vous n'avez pas besoin de démarrer de zéro. Vous pouvez vous procurer une carte vide auprès de Wikimedia Commons, à l'adresse *http://commons.wikimedia.org/wiki/File:USA_Counties_with_FIPS_and_names.svg*, comme illustré à la figure 8-15. La page propose des liens vers quatre tailles de carte différentes au format PNG et vers une carte au format SVG. C'est ce dernier

qu'il vous faut. Téléchargez le fichier et enregistrez-le sous counties.svg, dans le même répertoire que celui où se trouvent les données sur le chômage.

Figure 8-15 *Carte des comtés des États-Unis fournie par Wikimedia Commons*

Le point important, si vous n'êtes pas habitué à SVG, est qu'il s'agit de fait d'un fichier XML. C'est un fichier texte avec des balises et vous pouvez le modifier dans un éditeur de texte comme vous le feriez avec un fichier HTML. Le navigateur ou la visionneuse lit le code XML, lequel indique au navigateur ce qui doit être montré, comme les couleurs à utiliser et les formes à tracer.

Concrètement, ouvrez le fichier SVG dans un éditeur de texte. Il s'agit principalement de déclarations et autres instructions, que vous pouvez ignorer pour l'heure. Faites défiler le fichier jusqu'à la première balise `<path>`, comme vous le voyez à la figure 8-16. Tous les nombres entre ces balises spécifient les frontières d'un comté. Vous n'allez pas y toucher, car ce qui nous intéresse ici est de modifier la couleur de remplissage de chaque comté pour qu'elle corresponde au taux de chômage. À cette fin, vous devez changer le `style` dans la balise `path`.

Figure 8-16 *Balises « path » d'un fichier SVG*

Les fichiers SVG sont des fichiers XML, faciles à modifier dans un éditeur de texte. Cela signifie aussi que vous pouvez analyser le code SVG pour procéder à des modifications par programme.

Vous avez remarqué comment chaque balise `<path>` démarre avec le mot-clé `style` ? Ceux qui ont déjà créé des feuilles de style CSS l'auront vu immédiatement. Comme l'attribut `fill` est suivi d'une couleur hexadécimale, si vous modifiez celle-ci dans le fichier SVG, vous modifiez la couleur de l'image de sortie. Vous pouvez modifier chacune d'elles manuellement, mais il y a plus de 3 000 comtés. L'opération serait fastidieuse. À la place, retrouvez Beautiful Soup, le package Python qui facilite l'analyse du code XML ou HTML.

Ouvrez un fichier vide dans le même répertoire que celui où se trouvent votre carte SVG et vos données sur le chômage. Enregistrez-le sous le nom `colorize_svg.py`. Comme vous devez importer le fichier de données CSV et décoder le fichier SVG avec Beautiful Soup, commencez par importer les packages nécessaires.

```
import csv
from BeautifulSoup import BeautifulSoup
```

Puis, ouvrez le fichier CSV et stockez-le de telle sorte que vous puissiez parcourir toutes les lignes à l'aide de `csv.reader()`. Notez que le « r » de la fonction `open()` signifie simplement que vous voulez ouvrir le fichier pour lire son contenu, et non pour lui ajouter de nouvelles lignes.

```
reader = csv.reader(open('unemployment-aug2010.txt', 'r'), delimiter=",")
```

Maintenant, chargez aussi la carte SVG des comtés.

```
svg = open('counties.svg', 'r').read()
```

Vous avez chargé tout ce qu'il vous faut pour créer une carte choroplèthe. À ce stade, la difficulté est qu'il vous faut d'une façon ou d'une autre lier les données au fichier SVG. Quel est leur point commun ? Indice : il a partie liée à chaque identifiant unique de comté, et je l'ai déjà mentionné plus tôt. Si vous avez deviné qu'il s'agit des codes FIPS, vous avez trouvé !

Dans les fichiers SVG, les balises `<path>`, notamment les balises géographiques, possèdent généralement un seul identifiant unique. Ce n'est pas toujours un code FIPS, mais les mêmes règles s'appliquent.

Chaque balise `path` du fichier SVG possède un identifiant unique, combinaison du code FIPS de l'État et du code FIPS du comté. Chaque ligne des données du chômage possède les codes FIPS de l'État et du comté, également, mais ils sont distincts. Par exemple, le code FIPS de l'État pour Autauga County, Alabama, est 01, et le code FIPS du comté est 001. L'identifiant « path » dans le fichier SVG est le résultat de leur combinaison : 01001.

Vous devez stocker les données sur le chômage de telle façon que vous puissiez extraire le taux de chaque comté par code FIPS tandis que vous parcourez chaque balise « path ». Si vous commencez à vous sentir quelque peu perdu, rassurez-vous, les choses seront plus claires avec le code réel. Mais, ici, le point principal est que les codes FIPS constituent le lien commun entre vos fichiers SVG et CSV, et que vous pouvez utiliser ce lien à votre profit.

Pour stocker les données sur le chômage de telle sorte qu'elles soient par la suite facilement accessibles par code FIPS, utilisez une construction Python appelée « dictionnaire » (*dictionary*). Cette construction permet de stocker les valeurs et de les extraire par mot-clé. Dans le cas présent, votre mot-clé est une combinaison du code FIPS de l'État et du code FIPS du comté, comme illustré ci-après.

```
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
```

Analysez ensuite le fichier SVG avec BeautifulSoup. La plupart des balises se composent d'une balise d'ouverture et d'une balise de fermeture, mais il y a ici quelques balises à fermeture automatique que vous devez spécifier. Utilisez enfin la fonction `findAll()` pour extraire tous les chemins de la carte.

```
soup = BeautifulSoup(svg, selfClosingTags=['defs','sodipodi:namedview'])
paths = soup.findAll('path')
```

Puis, stockez les couleurs, obtenues auprès de ColorBrewer, dans une liste Python. Il s'agit d'un jeu de couleurs séquentielles composé de plusieurs nuances s'échelonnant du violet au rouge.

```
colors = ["#F1EEF6", "#D4B9DA", "#C994C7", "#DF65B0", "#DD1C77",
➥"#980043"]
```

Vous n'êtes plus très loin de la fin. Comme je l'ai dit précédemment, vous allez modifier l'attribut de style de chaque balise `path` du fichier SVG. Seule la couleur de remplissage vous intéresse, mais pour simplifier les choses, vous pouvez remplacer la totalité du style au lieu de procéder à une analyse et de ne remplacer qu'une seule couleur. J'ai modifié la valeur hexadécimale après `stroke` en `#ffffff`, à savoir blanc. Les bordures grises deviennent blanches.

```
path_style = 'font-size:12px;fill-rule:nonzero;stroke:#ffffff;
➥stroke-opacity:
1;stroke-width:0.1;stroke-miterlimit:4;stroke-dasharray:
none;stroke-linecap:butt;marker-start:none;stroke-linejoin:
bevel;fill:'
```

J'ai aussi déplacé `fill` vers la fin et laissé la valeur vide parce que c'est la partie qui dépend du taux de chômage de chaque comté.

Enfin, vous êtes prêt à modifier certaines couleurs ! Vous pouvez parcourir chaque balise `path` (à l'exception des lignes des frontières des États et du séparateur pour Hawaï et l'Alaska) et appliquer les couleurs en conséquence. Si le taux de chômage est supérieur à 10, utilisez une nuance plus sombre et s'il est inférieur à 2 utilisez la nuance la plus claire.

```
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
```

La dernière étape consiste à afficher le fichier SVG avec `prettify()`. La fonction convertit en une chaîne que votre navigateur peut interpréter.

```
print soup.prettify()
```

Il ne vous reste plus qu'à exécuter le script Python et à enregistrer la sortie comme nouveau fichier SVG, sous le nom, par exemple, de `colored_map.svg` (figure 8-17).

Figure 8-17 *Exécution du script Python et enregistrement de la sortie comme nouveau fichier SVG*

Ouvrez votre carte choroplèthe flambant neuve dans Illustrator ou dans un navigateur moderne tel que Firefox, Safari ou Chrome, pour admirer le résultat de votre travail, comme illustré à la figure 8-18. Il est maintenant facile de voir où étaient les taux de chômage les plus élevés en août 2010. Manifestement, une grande partie de la côte ouest et du sud-est avait des taux très élevés, à l'image de l'Alaska et du Michigan. Au centre de l'Amérique se situent un certain nombre de comtés dont le taux de chômage est relativement bas.

Vous pouvez obtenir l'intégralité du script à l'adresse *http://book.flowingdata.com/ch08/regions/colorize_svg.py.txt*.

Une fois terminée la partie la plus difficile de l'exercice, vous pouvez personnaliser la carte à votre guise. Vous pouvez modifier le fichier SVG dans Illustrator, modifier les couleurs et les tailles des bordures, et ajouter une annotation pour que le graphique soit complet et compréhensible par le plus grand nombre.

Figure 8-18 *Carte choroplèthe illustrant les taux de chômage*

L'aspect le plus positif est que le code est réutilisable et que vous pouvez l'appliquer à d'autres jeux de données qui emploient le code FIPS. Avec ce même jeu, vous pouvez aussi bricoler la palette de couleurs et concevoir une carte adaptée au thème de vos données.

En fonction des données, vous pouvez aussi modifier les seuils des couleurs applicables à chaque région. Jusque-là, les exemples s'appuyaient sur des seuils égaux où les régions étaient coloriées à l'aide de six nuances et où tout écart de 2 % donnait lieu à une nouvelle classe : les comtés dont le taux de chômage était supérieur à 10 constituaient une classe, puis ceux dont le taux était compris entre 8 et 10 formaient une classe à leur tour, et ainsi de suite. Une autre solution courante pour définir les seuils consiste à utiliser les quartiles : vous choisissez quatre couleurs et chacune d'elles représente un quart des régions.

Par exemple, les quartiles inférieur, intermédiaire et supérieur, des taux de chômage sont respectivement 6,9, 8,7 et 10,8 %. Cela signifie qu'un quart des comtés possède un taux inférieur à 6,9 %, un quart un taux compris entre 6,9 et 8,7 %, un quart un taux compris entre 8,7 et 10,8 %, et un dernier quart un taux supérieur à 10,8 %. À cette fin, modifiez la liste des couleurs de votre script en une palette de violets, par exemple, avec une nuance par quart.

```
colors = ["#f2f0f7", "#cbc9e2", "#9e9ac8", "#6a51a3"]
```

Puis, modifiez les conditions de couleur dans la boucle `for`, à l'aide des quartiles précédents.

```
    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
```

Exécutez le script et enregistrez-le comme précédemment ; vous obtenez la figure 8-19. Notez le grand nombre de comtés colorés de façon claire.

Pour accroître l'efficacité de votre code, vous pouvez calculer les quartiles par programme au lieu de les coder en dur. Cette opération est simple dans Python. Vous stockez une liste de vos valeurs, les triez de la plus petite à la plus grande, et rechercher les valeurs au quart, au demi et aux trois-quarts. Plus concrètement, comme dans cet exemple, vous pouvez modifier la première boucle de `colorize_svg.py` pour ne stocker que les taux de chômage.

```
unemployment = {}
rates_only = [] # Calculer les 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
```

Puis, triez le tableau et déterminez les quartiles.

```
# 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
```

Figure 8-19 *Taux de chômage divisés par quartiles*

Au lieu d'inscrire en dur les valeurs 6,9, 8,7 et 10,8 dans le code, vous pouvez les remplacer respectivement par `q1`, `q2` et `q3`. L'avantage du calcul des valeurs par programme est que vous pouvez réutiliser le code avec un autre jeu de données en modifiant simplement le fichier CSV.

Le choix de la gamme des couleurs dépend des données en votre possession et du message que vous voulez communiquer. Pour ce jeu de données particulier, je préfère une gamme linéaire parce qu'elle représente mieux la distribution et qu'elle met en évidence les taux de chômage relativement élevés à travers le pays. À partir de la figure 8-18, vous pouvez ajouter une légende, un titre et un paragraphe d'introduction afin d'obtenir un graphique plus finalisé, comme celui de la figure 8-20.

Figure 8-20 *Carte avec titre, introduction et légende*

Cartographier les pays

La Banque mondiale est l'une des ressources les plus complètes pour les données démographiques. Elle est généralement ma première source d'informations.

Le processus qui consiste à colorier les comtés dans l'exemple précédent n'est pas propre à ces régions. Vous pouvez recourir aux mêmes étapes pour colorier les États ou les nations. Tout ce dont vous avez besoin est un fichier SVG avec un identifiant unique pour les zones à colorier (facilement accessibles sur Wikipédia) et les identifiants de données correspondants. Appliquez maintenant cette solution à l'aide des données de la Banque mondiale.

Regardez les pourcentages de populations urbaines ayant accès à une source d'eau de qualité, par pays, en 2008. Vous pouvez télécharger le fichier Excel à partir du site de la Banque mondiale, à l'adresse *http://data.worldbank.org/indicator/SH.H2O.SAFE.UR.ZS/countries*. Pour des raisons de commodité, vous pouvez aussi télécharger les données brutes comme fichier CSV à l'adresse *http://book.flowingdata.com/ch08/worldmap/water-source1.txt*. Pour certains pays, les données manquent, ce qui est fréquent dans ce type de données. J'ai supprimé les lignes correspondantes du fichier CSV.

Il y a sept colonnes. La première contient le nom du pays, la deuxième son code (pourrait-il servir d'identifiant unique ?) et les cinq dernières colonnes les pourcentages de 1990 à 2008.

Pour la carte de base, tournons-nous à nouveau vers Wikipédia. Vous pouvez trouver différentes versions, mais utilisez de préférence celle-ci : *http://en.wikipedia.org/wiki/File:BlankMap-World6.svg*. Téléchargez le fichier SVG en pleine résolution et enregistrez-le dans le même répertoire que vos données. Comme illustré à la figure 8-21, il s'agit d'une carte du monde vide, de couleur grise et dont les bordures sont blanches.

Figure 8-21 *Carte du monde*

Ouvrez le fichier SVG dans un éditeur de texte. Il s'agit bien sûr d'un texte formaté en XML, mais quelque peu différemment de l'exemple des comtés. Les balises `path` n'ont pas d'identifiant utile et l'attribut `style` n'est pas utilisé. Les balises `path`, cependant, possèdent des classes s'apparentant aux codes pays. Elles n'ont cependant que deux lettres. Les codes pays employés dans les données de la Banque mondiale ont trois lettres.

Selon la documentation, la Banque mondiale utilise les codes ISO 3166-1 alpha 3. Le fichier SVG de Wikipédia, en revanche, utilise les codes ISO 3166-1 alpha 2. Les noms sont monstrueux, je le reconnais, mais pas de souci, vous n'aurez pas à vous en souvenir. Tout ce que vous devez savoir est que Wikipédia propose un graphique de conversion à l'adresse *http://en.wikipedia.org/wiki/ISO_3166-1*. J'ai copié et collé le tableau dans Excel, puis copié les informations importantes comme fichier texte. Il comporte une colonne pour le code alpha 2 et une autre pour le

code alpha 3. Téléchargez le tableau à l'adresse *http://book.flowingdata.com/ch08/worldmap/country-codes.txt*. Utilisez ce graphique pour passer d'un code à l'autre.

Quant à l'attribution d'un style à chaque pays, adoptez à cette fin une autre voie. Au lieu de modifier les attributs directement dans les balises `path`, utilisez CSS en dehors des balises `path` pour colorier les régions. En route ! Créez un fichier nommé `generate_css.py` dans le même répertoire que les fichiers SVG et CSV. À nouveau, importez le package CSV pour charger les données des fichiers CSV avec les codes pays et les pourcentages d'accès à l'eau.

```
import csv
codereader = csv.reader(open('country-codes.txt', 'r'), delimiter="\t")
waterreader = csv.reader(open('water-source1.txt', 'r'), delimiter="\t")
```

Puis, stockez les codes pays de telle sorte que l'on puisse passer aisément d'alpha 3 à alpha 2.

```
alpha3to2 = {}
i = 0
next(codereader)
for row in codereader:

      alpha3to2[row[1]] = row[0]
```

Ces instructions stockent les codes dans un dictionnaire Python où alpha 3 est la clé et alpha 2 la valeur.

Maintenant, comme dans notre exemple précédent, parcourez chaque ligne des données sur l'eau et attribuez une couleur en fonction de la valeur du pays en cours.

```
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
```

Cette partie du script exécute les étapes suivantes :

1. Elle ignore l'en-tête du fichier CSV.
2. Elle démarre la boucle pour parcourir les données sur l'eau.
3. S'il existe un code alpha 2 correspondant au code alpha 3 du fichier CSV et qu'il existe des données disponibles pour le pays en 2008, le script recherche le code alpha 2 correspondant.
4. La couleur appropriée de remplissage est choisie en fonction du pourcentage.
5. Une ligne CSS est imprimée pour chaque rang de données.

Exécutez `generate_css.py` et enregistrez la sortie sous le nom `style.css`. Les toutes premières lignes de la feuille CSS se présentent ainsi :

```
.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 }
...
```

Il s'agit de code CSS standard. La première ligne, par exemple, modifie la couleur de remplissage de toutes les balises `path` dont la classe est `.af` en `#7BCCC 4`.

Ouvrez `style.css` dans votre éditeur de texte et copiez la totalité du contenu. Puis, ouvrez la carte SVG et collez le contenu à la ligne 135 approximativement, sous les crochets de `.oceanxx`. Vous venez juste de créer une carte choroplèthe du monde, coloriée en fonction du pourcentage de la population ayant accès à une source d'eau améliorée, comme présenté à la figure 8-22. Le bleu le plus sombre indique 100 % et les nuances les plus claires de vert correspondent aux pourcentages les plus bas. Les pays qui demeurent en gris correspondent à ceux où les données ne sont pas disponibles.

Maintenant, vous pouvez télécharger pratiquement n'importe quel jeu de données de la Banque mondiale (et ils sont nombreux), puis créer assez rapidement une carte choroplèthe simplement en modifiant quelques lignes de code. Pour

obtenir le graphique de la figure 8-22, une fois encore, vous pouvez ouvrir le fichier SVG dans Illustrator et le retoucher. Pour l'essentiel, la carte a besoin d'un titre et d'une légende pour expliquer la signification de chaque nuance, comme illustré à la figure 8-23.

Figure 8-22 *Carte choroplèthe du monde illustrant l'accès aux sources d'eau améliorées*

Figure 8-23 *Carte complète*

Au fil de l'espace et du temps

Les exemples ont permis jusqu'à présent de visualiser un grand nombre de types de données, d'ordre qualitatif ou quantitatif. Vous pouvez varier les couleurs, les catégories et les symboles pour les adapter à l'histoire que vous racontez, annoter les cartes pour mettre en évidence des régions ou des caractéristiques spécifiques ou agréger les données pour effectuer un zoom avant sur un comté ou un pays. Mais, ce n'est pas tout ! Si vous intégrez une autre dimension des données, vous pouvez voir leur évolution au fil du temps et de l'espace.

Au chapitre 4, « Visualisation des modèles temporels », vous avez visualisé le temps de façon plus abstraite sous forme de graphiques et de tracés, ce qui est utile, mais si vous associez un emplacement aux données, il peut être plus intuitif de visualiser modèles et modifications sous forme de cartes. Il est plus aisé de voir des regroupements ou des groupes de régions proches en termes de distance physique. Le grand intérêt est de pouvoir intégrer ce que vous avez déjà appris pour visualiser vos données au fil de l'espace et du temps.

Petits multiples

Vous avez déjà vu cette technique au chapitre 6, « Visualisation des relations », pour visualiser les relations entre catégories et elle peut s'appliquer aux données spatiales également, comme présenté à la figure 8-24. Au lieu de petits graphiques en barres, vous pouvez utiliser des petites cartes, une pour chaque tranche de temps. Alignez-les de gauche à droite ou empilez-les de haut en bas, et vous suivrez aisément les modifications intervenues.

Figure 8-24 *Petits multiples avec cartes*

Par exemple, fin 2009, j'ai conçu un graphique qui montrait les taux de chômage par comté (figure 8-25). De fait, j'ai utilisé une variante du code de la section précédente, mais je l'ai appliquée à plusieurs tranches de temps.

Figure 8-25 *Taux de chômage de 2004 à 2009*

Il est aisé de voir les modifications, ou leur absence, par année, de 2004 à 2006, comme illustré à la figure 8-26. La moyenne nationale a réellement baissé durant cette période.

Figure 8-26 *Taux de chômage de 2004 à 2006*

Puis, l'année 2008 marque une évolution (figure 8-27) et vous commencez à voir certaines hausses du taux de chômage, notamment en Californie, dans l'Oregon et dans le Michigan, ainsi que dans certains comtés du sud-est.

Avançons jusqu'en 2009 et nous remarquons une claire différence, comme illustré à la figure 8-28. La moyenne nationale a augmenté de 4 % et les couleurs des comtés deviennent très sombres.

Figure 8-27 *Taux de chômage en 2008*

Figure 8-28 *Taux de chômage durant septembre 2009*

Si les images haute résolution sont trop grandes pour être affichées sur un seul écran, il peut être utile de placer l'image dans la visionneuse OpenZoom (*http://openzoom.org*), de telle sorte que vous puissiez voir l'image et effectuer un zoom avant sur les détails.

Ce fut l'un des graphiques les plus populaires que j'ai publié sur FlowingData, parce qu'il est facile de voir une évolution spectaculaire après plusieurs années de relative stabilité. J'ai également utilisé la visionneuse OpenZoom, qui permet d'effectuer un zoom avant sur les images haute résolution et de se concentrer ainsi sur votre propre zone pour voir comment elle a évolué.

J'aurais aussi pu visualiser les données comme graphique de séries temporelles, où chaque ligne représentait un comté ; cependant, il existe plus de 3 000 comtés américains. Le graphique aurait été encombré, et à moins qu'il n'ait été interactif, vous n'auriez pas pu dire à quel tracé correspondait tel comté.

Voir la différence

Il n'est pas toujours nécessaire de créer plusieurs cartes pour illustrer les changements. Parfois, il est plus judicieux de visualiser les différences réelles dans une seule carte. Cela permet d'économiser de la place et de mettre en évidence les modifications à la place de simples tranches de temps, comme vous pouvez le voir à la figure 8-29.

Figure 8-29 *Accent mis sur le changement*

Si vous deviez télécharger le nombre d'habitants des villes à partir de la Banque mondiale, vous auriez des données similaires au précédent exemple sur l'utilisation de l'accès à l'eau améliorée. Chaque ligne correspond à un pays et chaque colonne à une année. Cependant, les données de population urbaine sont les chiffres bruts du nombre estimé de personnes du pays vivant dans les zones urbaines. Une carte choroplèthe de ces chiffres mettrait inévitablement en évidence les pays les plus grands, parce que, bien sûr, d'une façon générale, ils ont plus d'habitants. Deux cartes pour illustrer les différences de population urbaine entre 2005 et 2009 ne seraient pas utiles à moins que vous ne transformiez les valeurs en proportions. Pour ce faire, vous devriez télécharger les données sur la population de tous les pays pour 2005 et 2009, puis vous livrer à quelques calculs. Rien de vraiment difficile à cela, mais il s'agit d'une étape supplémentaire. En outre, si les modifications sont subtiles, elles seront difficiles à visualiser sur plusieurs cartes.

À la place, vous pouvez ne prendre en compte que les différences et les représenter sur une seule carte. Vous pouvez facilement calculer ces différences dans Excel ou modifier le précédent script Python, puis créer une carte unique, comme illustré à la figure 8-30.

Il est aisé de voir quels sont les pays qui ont le plus changé et ceux qui ont le moins changé, quand vous visualisez les différences. Par contraste, la figure 8-31 illustre la proportion de la population totale de chaque pays qui vivait en zone urbaine en 2005.

Figure 8-30 *Évolution de la population urbaine entre 2005 et 2009*

Figure 8-31 *Proportion de personnes vivant en zone urbaine en 2005*

La figure 8-32 illustre les mêmes données pour 2009. Elle paraît similaire à la figure 8-31 et vous pouvez à peine voir la différence. Dans le cas de cet exemple particulier, il est clair que la carte seule est plus informative. Vous avez à fournir un moins grand effort mental pour déchiffrer les modifications. Il est évident que même si plusieurs pays d'Afrique, comparés au reste du monde, ont un pourcentage relativement inférieur de leur population vivant en zone urbaine, ce sont aussi ceux qui ont le plus évolué au cours des dernières années.

Figure 8-32 *Proportion de personnes vivant dans une zone urbaine en 2009*

N'oubliez pas d'ajouter une légende, la source et un titre si vous destinez votre graphique à une audience plus large, comme vous pouvez le voir à la figure 8-33.

Animation

Affichez la carte Walmart dans son intégralité à l'adresse *http://datafl.ws/197*.

L'une des solutions les plus évidentes pour visualiser les modifications au fil de l'espace et du temps consiste à animer les données. Au lieu d'afficher les tranches de temps sous forme de cartes individuelles, vous pouvez représenter les modifications telles qu'elles interviennent sur une seule et même carte interactive. Cela permet de préserver l'aspect intuitif de la carte, tout en autorisant les lecteurs à explorer les données par eux-mêmes.

Il y a quelques années, j'ai créé une carte détaillant la croissance de Walmart à travers les États-Unis, comme illustré à la figure 8-34. L'animation démarre par l'ouverture du premier magasin en 1962 à Rogers, Arkansas, et se poursuit jusqu'en 2010. Pour chaque nouveau magasin qui se créait, un autre point apparaît sur la carte. La croissance est d'abord lente, puis Walmart « s'étend » à travers le pays, un peu à la façon d'un virus. La croissance ne cesse de se poursuivre et l'entreprise procède à de larges acquisitions. Avant même que vous ne le sachiez, Walmart est omniprésente.

Figure 8-33 *Carte annotée des différences*

Figure 8-34 *Carte animée illustrant la croissance des magasins Walmart*

Vous pouvez observer la croissance des magasins Target à l'adresse *http://datafl.ws/198*.

À l'époque, j'essayais juste d'apprendre Flash et ActionScript, mais la carte était partagée sur le Web et avait été vue des millions de fois. Plus tard, j'ai créé une carte similaire illustrant la croissance de Target (figure 8-35) et elle fut également bien diffusée.

Figure 8-35 *Carte animée illustrant la croissance des magasins Target*

Les personnes se sont montrées très intéressées pour deux raisons essentielles. La première est que la carte animée permet de voir des modèles que vous ne verriez pas avec un graphique de séries temporelles. Un graphique normal afficherait uniquement le nombre d'ouvertures de magasins par année, ce qui est parfait si c'est l'histoire que vous voulez raconter, mais les cartes animées illustrent une croissance plus organique, comme celle de Walmart, notamment.

La deuxième raison est que la carte est immédiatement compréhensible par le grand public. Quand l'animation démarre, vous savez ce que vous voyez. Je ne suis pas en train de dire qu'une visualisation nécessitant un certain temps pour être correctement interprétée n'a pas de valeur – c'est même souvent l'inverse. Cependant, il existe un seuil temporel sur le Web, et par conséquent le fait que la carte soit intuitive (et que les utilisateurs puissent zoomer sur leurs propres zones d'habitation) a certainement favorisé le succès de la carte.

Créer une carte de croissance animée

Téléchargez Modest Maps à l'adresse *http://modestmaps.com*.

Dans cet exemple, vous créez la carte de croissance Walmart avec ActionScript. Vous utilisez Modest Maps, bibliothèque de cartographie ActionScript, pour créer l'interaction et la carte de base. Le reste, vous le codez vous-même.

Téléchargez la totalité du code source à l'adresse *http://book.flowingdata.com/ch08/Openings_src.zip*. Au lieu d'examiner chaque ligne et chaque fichier, nous nous concentrerons sur les aspects importants.

Comme au chapitre5, « Visualisation des proportions », lorsque vous créez un graphique en couches empilées avec ActionScript la boîte à outils de visualisation Flare, je vous recommande vivement d'utiliser Flex Builder. Il rend l'emploi d'ActionScript plus simple et permet que le code soit organisé. Bien sûr, vous pouvez continuer à écrire le code dans un éditeur de texte standard, mais Flex Builder regroupe l'éditeur, le débogueur et le compilateur au sein d'un même package. Cet exemple présume que vous avez bien Flex Builder, mais, bien sûr, vous êtes libre de vous procurer un compilateur ActionScript 3 sur le site Adobe.

Flex Builder s'appelle désormais Flash Builder. Il existe de petites différences entre les deux, mais vous pouvez utiliser l'un ou l'autre.

Pour commencer, ouvrez Flex Builder 3 et cliquez avec le bouton droit sur la barre de gauche, où s'affiche la liste active des projets. Sélectionnez Import (Importer), comme illustré à la figure 8-36.

Figure 8-36 *Importer le projet ActionScript*

Sélectionnez Existing Projects Into Workspace (Projets existants dans l'espace de travail), comme illustré à la figure 8-37. Puis, comme à la figure 8-38, accédez au répertoire dans lequel vous avez enregistré le code. Le projet Openings doit apparaître après que vous avez sélectionné le répertoire racine.

Téléchargez le code de la carte de croissance dans son intégralité à l'adresse *http://book.flowingdata.com/ch08/Openings_src.zip* afin de suivre le présent exemple.

Figure 8-37 *Projet existant*

Figure 8-38 *Importation du projet Openings*

Votre espace de travail dans Flex Builder doit être similaire à la figure 8-39. La totalité du code se trouve dans le dossier src. Ceci inclut Modest Maps dans le dossier com et TweenFilterLite dans le dossier gs, ce qui facilite les transitions. Avec le projet Openings importé, vous êtes prêt à démarrer la construction de la

carte. Procédez en deux parties. Dans la première partie, créez une carte de base interactive. Dans la deuxième, ajoutez les marqueurs.

Figure 8-39 *Espace de travail après l'importation d'un projet*

Ajouter la carte de base interactive

Dans Openings.as, les premières lignes du code importent les packages nécessaires.

```
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.*;
```

La première section importe les classes du package Modest Maps, tandis que la deuxième section importe les objets `display` et les classes `event` fournies par Flash. Le nom de chaque classe n'importe pas pour l'heure. Ils deviendront clairs au fur et à mesure de leur utilisation. Cependant, la convention de dénomination de la première section correspond à la structure du répertoire, en commençant par `com`, en se poursuivant avec `modestmaps` et en se finissant par `Map`. C'est ainsi que vous importez les classes la plupart du temps quand vous écrivez votre propre code ActionScript.

Au-dessus de `public class Openings extends Sprite`, plusieurs variables – `width`, `height`, `background color` et `frame rate` – du fichier compilé Flash sont initialisées.

```
    [SWF(width="900", height="450", backgroundColor="#ffffff",
frameRate="32")]
```

Puis, après la déclaration de classe, vous devez spécifier certaines variables et initialiser un objet Map.

```
private var stageWidth:Number = 900;
private var stageHeight:Number = 450;
private var map:Map;
private var mapWidth:Number = stageWidth;
private var mapHeight:Number = stageHeight;
```

Entre les crochets de la fonction `Openings()`, vous pouvez maintenant créer votre première carte interactive avec Modest Maps.

```
stage.scaleMode = StageScaleMode.NO_SCALE;
stage.align = StageAlign.TOP_LEFT;

// Initialiser la carte
map = new TweenMap(mapWidth, mapHeight, true, new
OpenStreetMapProvider());
map.setExtent(new MapExtent(50.259381, 24.324408, -128.320313,
-59.941406));
addChild(map);
```

Comme dans Illustrator, vous pouvez assimiler la totalité de l'interactivité à un ensemble de calques. Dans ActionScript et Flash, le premier calque établit le

décor. Vous le définissez pour ne pas réduire les objets lorsque vous effectuez un zoom avant et vous l'alignez sur le coin supérieur gauche. Ensuite, vous initialisez la carte avec les valeurs des variables `mapWidth` et `mapHeight`, activez l'interaction et utilisez les pavés de carte d'OpenStreetMap. En définissant l'étendue de la carte au code précédent, vous délimitez la carte autour des États-Unis.

Les coordonnées de `MapExtent()` sont la latitude et la longitude qui définissent le cadre de délimitation des zones du monde à montrer. Les premier et troisième nombres correspondent à la latitude et à la longitude du coin supérieur gauche, tandis que les deuxième et quatrième nombres correspondent à celles du coin inférieur droit.

Pour finir, ajoutez la carte (avec `addChild()`) au décor. La figure 8-40 illustre le résultat obtenu lorsque vous compilez le code sans ajouter aucun filtre à la carte. Vous pouvez appuyer sur le bouton Play (Lire) du coin supérieur gauche de Flex Builder, ou, dans le menu principal, sélectionner Run ⇨ Run Openings (Exécuter ⇨ Exécuter Openings).

Figure 8-40 *Carte simple utilisant les pavés OpenStreetMap*

Lorsque vous exécutez Openings, le résultat doit s'afficher dans votre navigateur par défaut. Rien n'apparaît encore, mais vous pouvez au moins vous amuser à cliquer et à déplacer. De même, si vous préférez un autre ensemble de pavés de carte, vous pouvez utiliser la carte routière proposée par Microsoft (figure 8-41) ou la carte hybride de Yahoo! (figure 8-42).

Vous pouvez aussi utiliser vos propres pavés si vous le désirez. Vous trouverez un excellent didacticiel sur le site Modest Maps.

Consultez les manuels de référence Adobe pour plus d'informations sur les matrices de couleur et la personnalisation des objets dans ActionScript à l'adresse *http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/flash/filters/ColorMatrixFilter.html*.

Figure 8-41 *Carte simple avec la carte routière de Microsoft*

Figure 8-42 *Carte simple avec la carte hybride de Yahoo!*

Vous pouvez aussi appliquer différentes couleurs à la carte à l'aide de filtres. Par exemple, vous pouvez modifier la carte en nuances de gris en ajoutant le code ci-après à celui que vous venez d'écrire. Le tableau `mat` comporte 20 éléments dont les valeurs sont comprises entre 0 et 1. Chaque valeur indique la quantité de rouge, de vert, de bleu et d'alpha de chaque pixel.

```
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];
```

Comme vous pouvez le voir à la figure 8-43, la carte est entièrement grise ce qui peut être utile pour mettre en évidence les données que vous prévoyez de déposer sur la carte. Celle-ci sert d'arrière-plan au lieu d'attirer l'attention.

Figure 8-43 *Carte en échelle de gris après application du filtre*

Vous pouvez aussi inverser les couleurs à l'aide d'une transformation de couleurs.

```
map.grid.transform.colorTransform =
    new ColorTransform(-1,-1,-1,1,255,255,255,0);
```

Le blanc est inversé en noir et le noir en blanc, comme illustré à la figure 8-44.

Figure 8-44 *Carte en noir et blanc après inversion des couleurs à l'aide d'une opération de transformation*

Pour créer les boutons de zoom, écrivez d'abord une fonction. Peut-être pensez-vous qu'il existe un moyen rapide par défaut de procéder, mais en réalité cette opération nécessite une quantité de code importante. La définition de la fonction `makeButton()` se trouve en bas de la classe 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;
}
```

Puis, créez une autre fonction qui se sert de la fonction et dessine les boutons souhaités. Le code suivant crée deux boutons utilisant `makeButton()`, l'un pour le zoom avant et l'autre pour le zoom arrière. Les boutons prennent place dans le coin inférieur gauche de la carte.

```
// Dessiner les boutons de navigation
private function drawNavigation():void
{
    // Boutons de navigation (zoom)
    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;
}
```

Cependant, comme il s'agit d'une fonction, le code ne s'exécutera que lorsque vous l'appellerez. Dans la fonction `Openings()`, également appelée constructeur, sous les filtres, ajoutez `drawNavigation()`. Maintenant, vous pouvez exécuter un zoom sur les emplacements qui vous intéressent, comme vous pouvez le voir à la figure 8-45.

Figure 8-45 *Carte avec fonction zoom activée*

Voilà tout ce dont vous avez besoin pour votre carte de base. Vous choisissez les pavés, définissez les variables et activez l'interaction.

Ajouter les marqueurs

Les étapes suivantes consistent à charger les données sur les emplacements Walmart et à créer un marqueur pour chaque ouverture de magasin. Dans le constructeur, le code suivant charge un fichier XML à partir d'une URL. Quand le fichier a terminé le chargement, la fonction intitulée `onLoadLocations()` est appelée.

```
    var urlRequest:URLRequest =
    new URLRequest('http://projects.flowingdata.com/walmart/
    ➥walmarts_new.xml');
    urlLoader = new URLLoader();
    urlLoader.addEventListener(Event.COMPLETE, onLoadLocations);
    urlLoader.load(urlRequest);
```

L'étape suivante repose sur la création de la fonction `onLoadLocations()`. Elle lit le fichier XML et stocke les données en tableaux en vue d'une utilisation ultérieure plus aisée. Au préalable, cependant, vous devez initialiser quelques variables supplémentaires après `navButtons`.

```
private var urlLoader:URLLoader;
private var locations:Array = new Array();
private var openingDates:Array = new Array();
```

Ces variables sont utilisées dans `onLoadLocations()`. La latitude et la longitude sont stockées dans `locations`, et les dates d'ouverture, au format an, dans `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);
}
```

À l'étape suivante, vous allez créer la classe MarkersClip. En suivant la même structure de répertoire que celle déjà présentée, il existe un répertoire baptisé flowingdata dans le répertoire com. Un répertoire gps se trouve dans le répertoire flowingdata. Enfin, dans com ➪ flowingdata ➪ gps, vous trouverez la classe MarkersClip. Il s'agit du conteneur de tous les marqueurs Walmart, ou plus précisément, de la couche de données de votre carte interactive.

Comme précédemment, vous devez importer les classes que vous utiliserez. Généralement, vous les ajoutez au fur et à mesure que vous en avez besoin dans le code, mais pour des raisons de simplicité, vous pouvez les ajouter toutes à la fois.

```
    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;
```

Les deux premières proviennent de Modest Maps, tandis que les quatre dernières sont des classes natives. Puis, vous définissez les variables juste avant la fonction `MarkersClip()`. Une fois encore, vous devriez les ajouter au fur et à mesure de vos besoins, mais, vous pouvez le faire maintenant pour accéder au cœur de cette classe, à savoir les fonctions.

```
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;
```

Dans le constructeur `MarkersClip()`, stockez les variables qui seront transmises à la classe et procédez à quelques calculs comme le nombre d'ouvertures par an et les coordonnées des magasins. Vous pouvez assimiler ces opérations à une configuration.

La variable `storesPerYear` stocke le nombre de magasins ouverts pendant une année donnée. Par exemple, un magasin a ouvert la première année et aucun autre l'année suivante. Quand vous utilisez ce code avec vos propres données, vous devez mettre à jour `storesPerYear` en conséquence. Vous pouvez aussi écrire une fonction qui calcule les magasins ou les ouvertures d'emplacement par an afin d'accroître la réutilisabilité du code. Pour des raisons de simplicité, dans cet exemple, nous avons préféré un tableau aux valeurs codées en dur.

```
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;
```

Il y a deux autres fonctions dans la classe MarkersClip : `setPoints()` et `setMarkers()`. La première traduit les coordonnées de latitude et de longitude en coordonnées *x* et *y*, et la seconde fonction place les marqueurs sur la carte sans réellement les afficher. Ci-après figure la définition de `setPoints()`.

La classe utilise une fonction intégrée fournie par Modest Maps pour calculer *x* et *y*, puis stocke les nouvelles coordonnées dans `xpoints` et `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;
          }
    }
```

La seconde fonction, `setMarkers()`, utilise les points stockés par `setPoints()` et place les marqueurs en conséquence.

```
        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);
        }
    }
```

La fonction utilise aussi une classe personnalisée `Marker`, que vous trouverez dans com ➪ flowingdata ➪ gps ➪ Marker.as, en présumant que vous ayez téléchargé la totalité du code source. Il s'agit pour l'essentiel d'un support qui « s'éclaire » quand vous appelez la fonction `play()`.

Maintenant les emplacements et les marqueurs sont chargés sur la carte. Cependant, si vous compiliez le code et lisiez le fichier, vous verriez toujours une carte vide. L'étape suivante consiste à parcourir les marqueurs pour qu'ils s'éclairent au moment approprié.

La fonction `playNextStore()` appelle simplement la fonction `play()` du marqueur suivant et se prépare à lire le marqueur suivant. Les fonctions `startAnimation()` et `onNextYear()` utilisent des minuteurs pour afficher de façon incrémentielle chaque magasin.

```
private function playNextStore(e:TimerEvent):void
{
  Marker(markers[markerIndex]).play();
  markerIndex++;
}
```

Si vous deviez compiler le code et exécuter l'animation maintenant, vous verriez les points, mais les fonctions zoom avant et zoom arrière ne fonctionneraient pas, comme illustré à la figure 8-46. Tandis que vous déplacez la carte d'avant en arrière et procédez à des zooms avant ou arrière, les bulles de chaque magasin restent fixes.

Figure 8-46 *Carte de croissance avec fonctions panoramique et zoom incorrectes*

Des *listeners* sont ajoutés au constructeur de telle sorte que les points se déplacent quand la carte se déplace. Chaque fois qu'un événement `MapEvent` est déclenché par Modest Maps, la fonction correspondante définie dans `MarkersClip.as` est appelée. Par exemple dans la première ligne ci-après, `onMapStartZooming()` est appelée chaque fois qu'un utilisateur clique sur le bouton de zoom de la carte.

```
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);
```

Vous obtenez alors la carte finale, telle qu'illustrée à la figure 8-47.

Figure 8-47 *Carte entièrement interactive illustrant les ouvertures de magasin WalMart*

L'histoire des ouvertures de magasin Walmart constitue une croissance organique. La compagnie a démarré dans un emplacement unique et s'est étendue lentement. Manifestement, ce n'est pas toujours le cas. Par exemple, la croissance de Target ne semble pas autant calculée. La croissance de Costco est moins spectaculaire parce qu'il y a moins d'emplacements, mais sa stratégie semble être de croître sur les côtes, puis d'opérer un mouvement vers l'intérieur.

Quoi qu'il en soit, c'est un moyen intéressant et divertissant d'afficher les données. Les cartes de croissance semblent éveiller l'imagination des individus, qui

peuvent alors réfléchir à l'extension de McDonald ou de Starbucks. Maintenant que vous disposez du code, il est beaucoup plus facile de l'implémenter. La difficulté reste la recherche des données.

Pour résumer

Les cartes sont un type de visualisation délicat, parce que, en plus de vos propres données, vous devez gérer l'aspect géographique. Cependant, en raison de leur degré d'intuitivité, les cartes peuvent aussi être gratifiantes, aussi bien dans la façon de présenter les données à autrui que dans l'exploration plus approfondie des données que ne le permet un graphique statistique.

Comme l'ont montré les exemples de ce chapitre, vous avez une multitude de possibilités à votre disposition pour traiter les données spatiales. Avec simplement quelques compétences de base, vous pouvez visualiser un grand nombre de jeux de données et raconter toutes sortes d'histoires intéressantes. Ce n'est là que la pointe émergée de l'iceberg. Ce que je veux dire, c'est que si des personnes se spécialisent en cartographie et géographie, c'est que les possibilités doivent être nombreuses. Vous pouvez manipuler des cartogrammes, qui dimensionnent les régions géographiques, ajouter plus d'interaction dans Flash ou combiner des cartes avec des graphiques pour des vues plus détaillées et exploratoires de vos données.

Les cartes en ligne sont devenues spécialement courantes et leur popularité ne fera que croître au fur et à mesure de l'évolution des navigateurs et des outils. Pour l'exemple de la carte de croissance, nous avons utilisé ActionScript et Flash, mais nous aurions pu tout aussi bien implémenter la carte en JavaScript. L'outil à utiliser dépend de l'objectif fixé. Si l'outil ne revêt pas une réelle importance, choisissez celui qui vous est le plus familier. Le principal point, en dehors du logiciel, est la logique. La syntaxe peut changer, mais vous faites la même chose avec vos données et recherchez le même flux dans votre narration.

Chapitre 9

Concevoir avec un objectif

Lorsque vous explorez vos propres données, vous n'avez pas grand-chose à faire en termes de narration. Après tout, c'est vous le narrateur. Cependant, lorsque vous présentez vos informations, que ce soit à un seul individu ou à plusieurs milliers ou millions de personnes, un graphique autonome ne suffit pas.

Bien sûr, vous voulez que d'autres interprètent les résultats et créent peut-être leurs propres histoires, mais il est difficile pour les lecteurs de savoir quelles questions poser quand ils ignorent tout des données qu'ils ont sous les yeux. Il est de votre devoir et de votre responsabilité de préparer le terrain. La façon dont vous concevez les graphiques influe sur celle dont les lecteurs interprètent les données sous-jacentes.

Se préparer

Vous devez connaître les documents source pour pouvoir raconter une histoire intéressante avec les données dont vous disposez. Il s'agit d'un aspect souvent ignoré dans la conception des graphiques. Au début, il est aisé de se passionner pour le résultat final. Vous voulez un beau graphique, à la fois étonnant et intéressant à regarder ; rien à redire à cela, mais vous n'y parviendrez pas si vous n'avez aucune idée de ce que vous visualisez. Vous vous retrouverez avec un graphique tel que celui illustré à la figure 9-1. Comment réussir à expliquer les points intéressants d'un jeu de données si vous n'en maîtrisez pas le contenu ?

Visualiser consiste à communiquer des données. Par conséquent, prenez le temps de découvrir ce qui forme la base de votre graphique, sans quoi vous vous retrouverez avec des chiffres dans tous les sens.

Intéressez-vous aux chiffres et aux mesures, à leur provenance et à leur méthode d'estimation. Demandez-vous même s'ils ont un sens. Cette première étape de recueil des données est ce qui fait la qualité des graphiques du *The New York Times*. Vous lisez le résultat final dans le journal et sur le Web, mais vous ignorez tout le temps passé à la conception du graphique avant qu'une forme ne se dessine. Généralement, la mise en ordre des données se révèle plus longue que la création du graphique.

Figure 9-1 *Projet de graphique. Faire bien ou ne pas faire.*

Aussi, la prochaine fois que vous vous retrouverez avec un jeu de données, ne passez pas directement à la conception du graphique. C'est la solution des paresseux et cela finit toujours par se voir. Prenez le temps de découvrir les données et leur contexte.

Placez quelques chiffres dans R, lisez la documentation associée pour connaître le système de mesure utilisé, et vérifiez que certains éléments ne paraissent pas étranges. Si tel est le cas et que vous ne comprenez pas pourquoi, vous pouvez toujours contacter les auteurs de votre source. Ceux-ci aiment découvrir que l'on s'intéresse aux données qu'ils publient et procéderont volontiers à d'éventuelles corrections.

Une fois que vous maîtrisez bien vos données, vous voilà prêt à créer le graphique. Prenons un exemple dans le monde du cinéma. Vous vous souvenez de cette scène dans *The Karate Kid* où Daniel commence à apprendre les arts martiaux ? Mister Miyagi lui demande de lustrer un flot de véhicules, de poncer un parquet et de refaire une clôture. Daniel éprouve un sentiment de frustration, parce qu'à ses yeux, ce sont là des tâches inutiles. Puis, il s'avère que, tout à coup, les gestes du karaté lui viennent naturellement, parce que ce sont ceux-là qu'il a travaillés sans le savoir. C'est la même chose avec les données. Apprenez tout ce qu'il vous est possible d'apprendre sur les données et la narration viendra naturellement.

Préparer le public

Votre tâche en tant que concepteur de données est de communiquer à votre public ce que vous savez. Il est vraisemblable que vos lecteurs n'auront pas regardé les données et, de ce fait, ils ne verront pas la même chose que vous, si vous ne leur fournissez aucune explication ou introduction. Je pars du principe que les lecteurs tombent sur mes graphiques par hasard, et avec le partage d'informations via Facebook, Twitter ou les liens d'autres blogs, je ne suis sans doute pas très loin de la vérité.

Par exemple, la figure 9-2 illustre la capture d'écran d'une carte animée créée par mes soins. Si vous n'avez jamais vu ce graphique auparavant, vous n'avez probablement aucun indice sur ce que vous regardez. En vous appuyant sur les exemples du chapitre 8, « Visualisation des relations spatiales », votre meilleure supposition pourrait concerner les ouvertures de magasin d'une grande enseigne.

En réalité, la carte illustre les tweets géolocalisés publiés lors de l'investiture du président Barack Obama, le mardi 20 janvier 2009. L'animation démarre le lundi matin et, au fur et à mesure de la journée, un plus grand nombre de personnes échange des messages à un rythme régulier.

Consultez l'animation dans sa totalité à l'adresse suivante : *http://datafl.ws/19n*.

Figure 9-2 *Carte sans aucun titre ou contexte*

Le nombre de tweets par heure augmente à l'approche de l'événement, et l'Europe elle-même prend le relais tandis que les Américains dorment encore. Puis, le mardi matin, la cérémonie donne lieu à une grande excitation. Vous pouvez facilement voir cette progression à la figure 9-3. Si j'avais fourni ce contexte à la figure 9-2, le graphique aurait eu plus de sens pour les lecteurs.

Figure 9-3 *Tweets échangés lors de l'intronisation du président Barack Obama*

Inutile d'écrire un essai pour accompagner chaque graphique, mais un titre et quelques explications via un texte d'introduction sont toujours utiles. Il est souvent judicieux d'inclure un lien dans le graphique de telle sorte que les lecteurs puissent toujours accéder à vos explications, même si le graphique est partagé sur un autre site. Sinon, et avant même que vous ne le sachiez, le graphique que vous avez minutieusement conçu risque de se retrouver revêtu d'une signification inverse à celle de votre propos initial. Le Web fourmille de bizarreries de ce genre.

Autre exemple, le graphique de la figure 9-4 est une simple chronologie qui illustre les dix principales violations de données depuis 2000. Le graphique est simple, avec dix points de données seulement, mais lorsque je l'ai publié sur FlowingData, j'ai évoqué le fait que les violations avaient augmenté en fréquence entre 2000 et 2008. Le graphique s'est retrouvé abondamment partagé et même le magazine *Forbes* en a publié une variante. Je ne suis pas certain que les lecteurs auraient prêté autant d'attention au graphique si je n'avais pas fourni cette simple observation.

Figure 9-4 *Principales violations de données depuis 2000*

Un enseignement à tirer : ne présumez pas que le lecteur sait tout ou qu'il est à même d'identifier les particularités de votre graphique. Cela est particulièrement vrai du Web, où les utilisateurs sont habitués à toujours cliquer sur l'information suivante. Je ne dis pas, bien sûr, que les lecteurs ne passent pas de temps à regarder les données. Le blog OkCupid a proposé des billets relativement longs présentant les résultats des analyses détaillés de ses données de rencontre en ligne. Parmi les titres figuraient « Les meilleures questions pour une première rencontre » et « Les mathématiques de la beauté ».

Les billets du blog ont été vus des millions de fois et les gens aiment ce que les membres d'OkCupid ont à raconter. En dehors du contexte du billet proprement dit, les lecteurs accèdent au blog avec leur propre contexte. Comme il s'agit de données et de conclusions sur le sexe opposé, les lecteurs peuvent facilement y

associer leur propre expérience. La figure 9-5, par exemple, est un graphique qui illustre ce que les Asiatiques aiment généralement, informations provenant d'un billet publié par OkCupid sur ce que les gens aiment, en fonction de leur origine ethnique et de leur sexe. Hé ! je suis d'origine asiatique *et* je suis un homme. Connexion immédiate !

Figure 9-5 *Ce qu'aiment les Asiatiques, à partir des profils de rencontres en ligne du site OkCupid.*

En revanche, si votre graphique traite des niveaux de pollution ou de la dette mondiale, il est beaucoup plus difficile de le présenter à un large public sans quelques bonnes explications.

Parfois, peu importe le nombre d'explications que vous donnez, car les utilisateurs ne les liront pas et ne feront que les parcourir rapidement. J'ai publié, par exemple, une carte de FloatingSheep comparant le nombre de bars et le nombre de supermarchés aux États-Unis, comme illustré à la figure 9-6. La couleur rouge indique les zones où les bars sont plus nombreux que les supermarchés, et la couleur orange illustre la situation inverse. Les membres de FloatingSheep ont baptisé la carte, « Le ventre à bière de l'Amérique ».

Vers la fin du billet, je m'interrogeais sur la précision de la carte et je terminais par la question suivante : « Y a-t-il quelqu'un dans la région qui puisse confirmer ? Je m'attends à ce que vos commentaires soient pleins de fautes d'orthographe et n'aient guère de sens. Avec, peut-être, beaucoup de bêtises ». La

leçon ? L'humour pince-sans-rire et l'ironie ne sont pas très faciles à traduire en ligne, notamment quand les lecteurs ne sont pas habitués à vous lire. Je ne m'attendais pas réellement à ce que les commentaires contiennent beaucoup de bêtises. La plupart ont compris la plaisanterie, mais j'ai aussi reçu un grand nombre de commentaires de la part d'habitants du Wisconsin qui se sont sentis outragés. Comme je l'ai dit, le Web est vraiment un endroit intéressant (dans le bon sens du terme).

Figure 9-6 *Lieux où on trouve plus de bars que de supermarchés aux États-Unis.*

Indices visuels

Au chapitre 1, nous avons vu comment fonctionnait le codage. Pour résumer, vous possédez les données et celles-ci sont codées sous forme de figure géométrique, de couleur ou d'animation. Les lecteurs décodent alors les formes, les nuances ou les mouvements, et les mettent en correspondance avec les chiffres.

Tel est le fondement de la visualisation. Le codage constitue une traduction visuelle. Le décodage permet de voir les données sous un autre angle et de découvrir des modèles que vous n'auriez pas repérés si vous vous étiez contenté de regarder les données dans un tableau ou une feuille de calcul.

Ces opérations sont habituellement simples, car elles reposent sur des règles mathématiques. Les barres les plus hautes correspondent aux valeurs les plus hautes et les cercles les plus petits aux valeurs les plus basses. Même si votre ordinateur prend un grand nombre de décisions pendant ce processus, il vous appartient toujours de choisir les codages adaptés au jeu de données à votre disposition.

À travers tous les exemples des précédents chapitres, vous avez vu qu'une bonne conception contribuait non seulement à l'esthétique des graphiques, mais les rendait également plus faciles à lire et modifiait la perception des lecteurs quant aux données ou à l'histoire que vous leur narrez. Les graphiques créés avec les paramètres par défaut de R ou d'Excel présentent un aspect brut et mécanique. Ce n'est pas nécessairement une mauvaise chose. Peut-être cela correspond-il précisément à ce que vous voulez montrer dans le cas d'un rapport universitaire. Ou si votre graphique constitue un simple complément d'un texte plus important, il peut être préférable de ne pas détourner l'attention du lecteur des points sur lesquels vous souhaitez qu'il se concentre. La figure 9-7 illustre un graphique en barres aussi simple que l'on puisse l'imaginer.

Cependant, si vous voulez mettre en évidence votre graphique, un rapide changement de couleur peut faire toute la différence. La figure 9-8 reprend simplement la figure 9-7 avec des couleurs de premier plan et d'arrière-plan différentes. Pour un sujet plus grave, il est possible d'utiliser un jeu de couleurs plus sombres, tandis que des couleurs plus vives conviendront à un thème plus léger (figure 9-9). Bien sûr, vous n'avez pas toujours besoin d'un thème. Vous pouvez utiliser une palette de couleurs neutre si vous le désirez, comme à la figure 9-10.

Figure 9-7 *Graphique en barres simple*

Figure 9-8 *Graphique par défaut avec un jeu de couleurs sombres*

Figure 9-9 *Graphique par défaut avec un jeu de couleurs claires*

Figure 9-10 *Graphique par défaut avec un jeu de couleurs neutres*

Le point essentiel est que le choix de la couleur peut jouer un rôle déterminant dans les graphiques de données. La couleur peut susciter des émotions (ou pas) et aider à fournir un contexte. Il est de votre responsabilité de choisir des couleurs qui traduisent un message précis. Elles doivent aussi correspondre à l'histoire que vous voulez raconter. Comme illustré à la figure 9-11, un simple changement de couleur peut modifier complètement la signification des données. Le graphique conçu par David McCandless et par Always With Honor explore la signification des couleurs selon les différentes cultures. Par exemple, le noir et le blanc sont souvent utilisés pour représenter la mort ; cependant, le bleu et le vert sont plus couramment employés, respectivement dans la culture musulmane et la culture latino-américaine.

De même, vous pouvez modifier une forme géométrique pour exprimer une autre émotion ou une signification différente. Par exemple, la figure 9-12 illustre un graphique en barres empilées généré de façon aléatoire à l'aide des documents D3 (*Data-Driven Documents*) du spécialiste en visualisation Mike Bostock. Il possède des bords droits et des points distincts, ainsi que des pics et des creux.

Figure 9-11 *Les couleurs selon les cultures par David McCandless et Always With Honor*

Pour plus d'informations sur les graphiques de flux, consultez l'article de Lee Byron et Martin Wattenberg intitulé *Stacked Graphs – Geometry and Aesthetics*. Il existe aussi plusieurs packages qui vous permettront de créer votre propre graphique.

Figure 9-12 *Graphique en barres empilées généré de façon aléatoire*

Si, à la place, vous aviez utilisé un graphique de flux pour afficher les mêmes données (figure 9-13), vous auriez obtenu un tout autre aspect. Il est plus fluide et continu, et au lieu de pics et de creux, nous avons des gonflements et des resserrements. Toutefois, la géométrie entre les deux types de graphiques est similaire. Le graphique de flux consiste en un graphique en barres empilées lissé et dont l'axe horizontal se trouve au centre et non en bas.

Figure 9-13 *Graphique de flux généré de façon aléatoire*

Parfois, le contexte peut simplement provenir de la façon dont vous organisez formes et couleurs. La figure 9-14 illustre un graphique que j'ai créé pour les fêtes de Noël. La partie supérieure montre les ingrédients qui composent la

marinade de la dinde de Noël et la partie inférieure ce qu'il faut y mettre quand vous rôtissez l'animal.

Figure 9-14 *Recette de la dinde de Noël*

Ainsi, à son niveau le plus élémentaire, la visualisation permet de convertir les données, qu'il s'agisse de nombres, de texte, de catégories ou de toute autre chose, en éléments visuels. Certains indices visuels fonctionnent mieux que d'autres, mais leur applicabilité varie en fonction du jeu de données. Une méthode qui semble totalement erronée pour un jeu de données peut parfaitement convenir à un autre. Avec de l'entraînement, vous pourrez rapidement décider laquelle correspond le mieux à votre objectif.

Bonne visualisation

Même si l'on représente les données sous forme de graphiques depuis des années, ce n'est qu'au cours des toutes dernières décennies que les chercheurs se sont intéressés à ce qui fonctionnait et à ce qui ne fonctionnait pas. À cet égard, la visualisation est un domaine relativement nouveau. Cependant, il n'existe pas de consensus sur ce qu'est réellement la visualisation. S'agit-il de quelque chose généré par un ordinateur à partir d'un ensemble de règles ? Si une personne est partiellement responsable du processus de conception, est-ce que cela n'en fait plus de la visualisation ? Les graphiques d'informations relèvent-ils de la visualisation ou de leur propre catégorie ?

Si vous consultez le Web, vous trouverez de nombreuses discussions sur les différences et les similitudes entre graphiques d'informations et visualisation, ainsi que des tentatives de définition de ce qu'est la visualisation. Le résultat est toujours un incessant va-et-vient, sans solution tranchée. Ces opinions contradictoires mènent à différents critères pour décider si un graphique de données est bon ou mauvais.

Les statisticiens et les analystes, par exemple, considèrent généralement la visualisation comme un graphique statistique traditionnel qu'ils peuvent utiliser dans leurs analyses. Si un graphique ou une animation interactive n'aide pas à l'analyse, il n'a aucune utilité. C'est un échec. En revanche, si vous parlez aux concepteurs graphiques du même graphique, ils peuvent considérer que le résultat est une réussite, car il affiche les données pertinentes équitablement et présente les données d'une manière intéressante.

Ce que vous devez réussir à faire c'est réunir ces avis opposés dans le même espace plus souvent ! Celui qui est intéressé par l'analyse peut apprendre beaucoup des concepteurs pour rendre les données plus fiables et plus compréhensibles, tandis que celui qui s'intéresse à la conception peut apprendre à explorer plus profondément les données de ses homologues analystes.

Je n'essaie pas de définir ce qu'est la visualisation, parce que sa définition n'affecte pas la façon dont je travaille. Je considère le public et les données que j'ai sous les yeux, et je me demande si le graphique final a un sens. Me dit-il ce que je veux savoir ? Si la réponse est oui, alors très bien. Si la réponse est non, je retourne sur mon ordinateur et cherche à améliorer le graphique pour qu'il

réponde aux questions que je me pose sur les données. En fin de compte, tout dépend de vos objectifs pour le graphique, de l'histoire que vous voulez raconter et des personnes auxquelles vous voulez vous adresser. Prenez en compte tous les éléments ci-dessus et la chance sera avec vous !

Pour résumer

Beaucoup de personnes qui travaillent sur les données considèrent la conception comme un simple moyen d'embellir les graphiques. C'est en partie vrai, mais il convient d'ajouter qu'elle permet de rendre les graphiques lisibles, compréhensibles et utilisables. Vous pouvez aider le lecteur à mieux comprendre les données que s'il regardait un graphique ordinaire. Vous pouvez libérer de la place, mettre en évidence les points importants des données ou même susciter une réponse de nature émotionnelle. Les graphiques de données peuvent être divertissants et informatifs. Parfois, ils ne seront qu'amusants, selon votre objectif, mais peu importe ce que vous concevez – visualisation, graphique d'informations ou art des données – laissez les données guider votre travail !

Lorsque vous avez un jeu de données volumineux et que vous ne savez pas par où commencer, la meilleure solution pour démarrer consiste à poser une question. Que voulez-vous savoir ? Recherchez-vous un modèle saisonnier ? Des relations entre plusieurs variables ? Des observations aberrantes ? Des relations spatiales ? Puis retournez à vos données et voyez si vous pouvez répondre à la question. Si vous ne disposez pas des données nécessaires, cherchez-en de nouvelles.

Une fois que vous avez vos données, vous pouvez utiliser les compétences acquises grâce aux exemples de ce livre pour raconter une histoire intéressante. Cependant, ne vous arrêtez pas là. Considérez les éléments sur lesquels vous avez travaillés comme un fondement. Au cœur de tous vos graphiques de données favoris se trouvent un type de données et une méthode de visualisation avec laquelle vous savez maintenant travailler. Vous pouvez vous appuyer sur ces fondations pour élaborer des graphiques plus avancés et plus complexes. Ajoutez des interactions, associez des graphiques ou complétez ceux-ci à l'aide de photos et de texte pour offrir un contexte enrichi.

N'oubliez pas : les données ne sont qu'une représentation de la vie réelle. Lorsque vous visualisez les données, vous visualisez ce qui se passe autour de vous et dans le monde. Vous pouvez voir ce qui se passe à un micro niveau avec les individus ou à une échelle plus large en couvrant l'univers. Découvrez les données et vous pourrez raconter des histoires que la plupart des personnes ne connaissent pas, mais sont impatientes d'entendre. Vous avez plus de données que vous n'en avez jamais eues auparavant et le public veut savoir ce qu'elles signifient. Maintenant, vous pouvez le lui dire. Amusez-vous bien.

Index