CHAPTER 4

GRAPHICS

In which we see a lot just by looking.

4.1 WHY IS SURVEY DATA DIFFERENT?

Historically, survey statistics has made relatively little use of graphics. This is only partly due to the genuine difficulties in visualizing large data sets with unequal sampling weights. Other contributing factors are the use of batch-mode computing on large, centralized computers rather than interactive data analysis on individual workstations, and the traditional focus on pre-specified population summaries rather than exploratory analysis.

In social sciences and health sciences there is increasing interest in regression analysis of large data sets that have often been collected in complex survey designs, for the reasons explained in Chapters 2 and 3. This will result in the same needs for data visualization that apply in traditional cohort studies.

In model-based statistics the uses of graphics can be divided into three categories: data exploration, model criticism, and communicating results. The first and last

Complex Surveys: A Guide to Analysis Using R. By Thomas Lumley

categories carry over directly into survey inference. The second category, model criticism, appears at first glance to be less useful since survey methods require no model assumptions for validity. Looking more carefully at graphical model diagnostics, however, reveals that they are less often concerned with technical validity and more often with whether the regression coefficients are meaningful summaries of the data. This question of whether a population estimate is a useful summary is still present in design-based inference, and graphical diagnostics are thus valuable.

One area in which there is a long history of visualization for survey data is cartography. There are important practical and design challenges in finding the necessary mapping data and in drawing useful maps. There are also statistical challenges in conveying uncertainty and reducing the visual bias from varying population densities. Section 4.6 gives some examples of maps based on complex surveys, but many of the more complex issues of cartographic design and small-area spatial estimation are beyond the scope of this book. Many of these issues can be addressed in R, and some references are given.

The principal difficulty in designing graphics for complex survey data is representing the sampling weights. The graphs in this chapter use three strategies

1. Base the graph on an estimated population distribution.

2. Explicitly indicate weights on the graph.

3. Draw a simple random sample from the estimated population distribution and graph this sample instead.

A secondary problem in visualizing survey data is that data sets are often large. If a graph includes a separate plotting symbol for each point in the sample, the symbols will tend to overlap and hide each other. Graphs with thousands of plotting symbols also tend to be slow to draw and to result in large graphics files. Figure 3.3 shows the overplotting problem.

It is often difficult to determine who first used a visualization technique, and which of the later uses were independent inventions. Korn and Graubard's book [83] and 1998 paper [82] describe a number of useful graphical techniques, and the evolutionary history for some of them goes back to at least Chambers et al. [33]. Modern methods for large data sets, not specifically complex surveys, are described by Unwin et al. [177]. For a general reference on R graphics see Murrell [111] or the R Graphics Gallery at `http://addictedtor.free.fr/graphiques/`.

4.2 PLOTTING A TABLE

Many analyses of survey data involve tables, either tables of summary statistics as created by `svyby()` or contingency tables. Visualization techniques for these tables are essentially the same as if the table had been created from a simple random sample. This section describes some of the facilities that have been implemented in the **survey** package. Other ideas for visualizing categorical data can be found in Friendly [50].

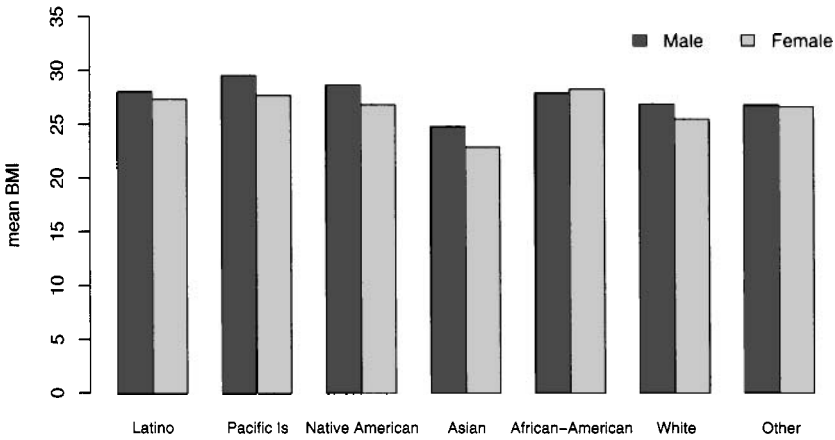

Figure 4.1 Mean BMI by race/ethnicity and gender, from CHIS

Many of these are implemented in the vcd package [108] and could easily be adapted to estimates from complex samples.

Interactive and dynamic graphics are especially valuable for displaying multivariate summaries of national or regional data. The website has links to examples of these at the UK Office of National Statistics, the SMARTCentre at the University of Durham, the Digital Government Quality Graphics project at Pennsylvania State University, and the "GapMinder" website.

Bar charts. Figures 4.1, 4.2, and 4.4 show the BMI subgroup means from CHIS 2005 that were computed with `svyby()` in Figure 2.11. The objects produced by most survey estimation functions have methods for the `barplot()` function, and so `barplot(bys)` will produce a barplot. The plot in Figure 4.1 has been enhanced by supplying a legend and more readable x-axis labels; the plot in Figure 4.2 is exactly as produced by `dotchart(bys)`. Because the bars in a barplot are tied to $y = 0$ and the scales in a dotplot are not, dotplots allow the informative part of the y-axis range to be clearly seen even if it does not include zero.

Forest plots. Figure 4.4 shows the uncertainties as well as the means, in a *forest plot* (Lewis and Clarke [94]). The estimated means are depicted by squares, with the area of the square proportional to the precision (or effective sample size) for the estimate, so that more precise estimates have more visual weight. The lines extend $\sqrt{2}$ standard errors in each direction, so that overlapping confidence intervals correspond roughly to a non-significant difference at a 0.05 level. By removing the need to include zero on the range of displayed values, the forest plot expands the scale and makes differences easier to see.

The `forestplot()` function is in the rmeta package [98]. The code to produce this forest plot is in Figure 4.3

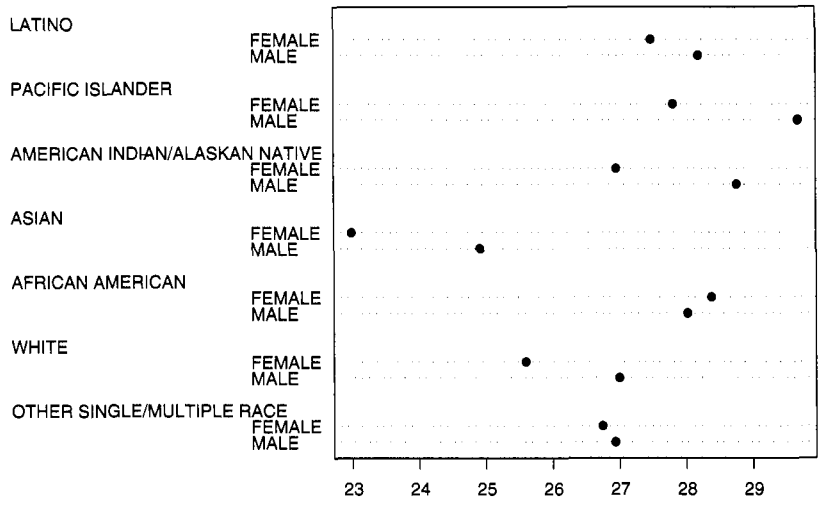

Figure 4.2 Mean BMI by race/ethnicity and gender, from CHIS

```
forestplot(ll, coef(bys), coef(bys)-sqrt(2)*SE(bys),
  coef(bys)+sqrt(2)*SE(bys), align=c("l","l"), zero=25,
  graphwidth=unit(3,"inches"), xticks=20+2*(1:5))
```

Figure 4.3 Code for forest plots of mean BMI, from CHIS

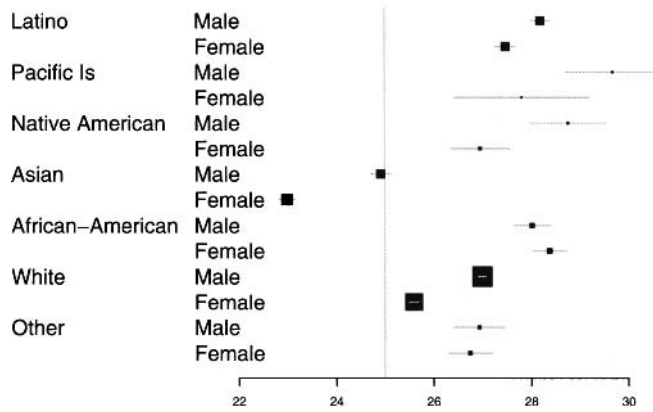

Figure 4.4 Mean BMI$\pm\sqrt{2}$ standard errors, by race/ethnicity and gender, from CHIS. The vertical line indicate the division between normal and overweight.

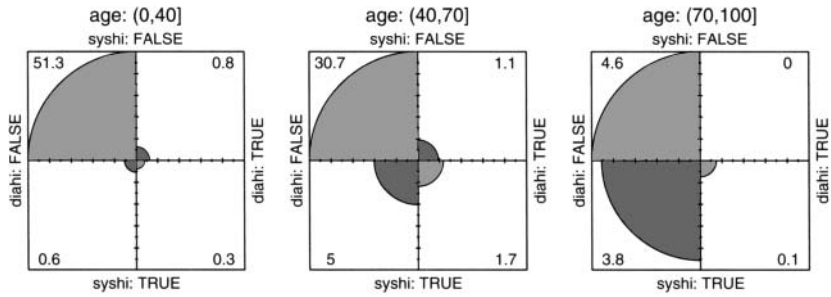

Figure 4.5 Systolic and diastolic hypertension by age group, from NHANES 2003–2004

Fourfold plots. Figure 4.5 is a fourfold plot (Friendly [49]), a summary of a set of 2×2 tables. In this example the tables were created with `svytable()` and show the proportions with elevated systolic (>140 mmHg) or diastolic (>90 mmHg) blood pressure, by age group. The quarters within each table have area proportional to the population counts, with the largest cell in each table standardized to fill its box. The numbers in each cell give the percentages of the total population. Code for loading the data is in Appendix B, in Figure B.1, and the survey design was set up in exercise 3.1.

The proportion with hypertension clearly increases with age. More interestingly, the form of hypertension also changes. In the middle age group there is a substantial fraction with elevated diastolic blood pressure, but the oldest group overwhelmingly has isolated systolic hypertension, that is, high systolic pressure with normal or low diastolic pressure. We can compute the kappa measure of agreement between systolic and diastolic hypertension for different ages

```
> svykappa(~hisys+hidia,
    subset(d, RIDAGEYR<50 &!is.na(hisys) & !is.na(hidia)))
        nlcon     SE
kappa 0.41659 0.0379
> svykappa(~hisys+hidia,
   subset(d, RIDAGEYR>50 &!is.na(hisys) & !is.na(hidia)))
        nlcon     SE
kappa 0.11824 0.0226
```

and see that agreement is moderately good for people under 50 but poor for people over 50.

4.3 ONE CONTINUOUS VARIABLE

4.3.1 Graphs based on the distribution function

The most straightforward graphical displays for a single continuous variable are boxplots, cumulative distribution function plots, and survival curves. These graphs are already all based on non-parametric estimates of population quantities: the emprirical CDF is the nonparametric maximum likelihood estimator of the population CDF, the Kaplan–Meier survival curve estimates the population survival curve, and the boxplot is based on quartiles that estimate population quartiles. Adapting these estimators to complex samples simply involves inserting sampling weights in the appropriate places. Computing standard error estimates can be more difficult (e.g., Figure 2.7) but is often not necessary for graphics.

The cumulative distribution function is computed by `svycdf()` and the survival curve for survival data is computed by `svykm()`. Both functions take a formula as their first argument, with the left-hand side of the formula giving the variable to be plotted and the right-hand side giving grouping variables, and both return an object that has a `plot()` method for drawing a graph and a `lines()` method for adding the curve to an existing plot. The analogous functions for unweighted estimates are `ecdf()` and `survfit()`.

Figure 4.6 is based on the stratified sample of schools from the API population defined in Figure 2.5 and shows the population CDF, the unweighted sample CDF, and the estimated CDF using sampling weights. Because the sampling design oversampled high schools, which tend to be larger, the unweighted sample CDF shows a higher proportion of large schools than the population. The correctly weighted CDF estimate is very close to the population CDF. The code to create this figure is in Figure 4.7. The call to `svycdf()` estimates three CDFs, the correct one can be specified

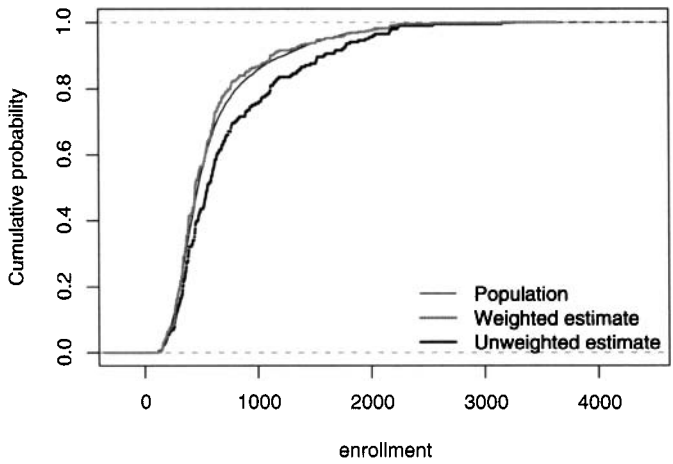

Figure 4.6 Cumulative distribution of California school size: population, weighted estimate, unweighted estimate.

by position, as in the code, or by name, e.g., `cdf.est[["enroll"]]`. The call to `par()` modifies the margins around the figure, reducing the white space at the top. The `do.points` argument to `plot()` and `lines()` specifies whether the individual data points should be plotted on top of the CDF line. The `legend()` function draws a legend, with the first argument indicating where it should be placed.

```
cdf.est<-svycdf(~enroll+api00+api99, dstrat)
cdf.pop<-ecdf(apipop$enroll)
cdf.samp<-ecdf(apistrat$enroll)
par(mar=c(4.1,4.1,1.1,2.1))
plot(cdf.pop,do.points=FALSE, xlab="enrollment",
    ylab="Cumulative probability",main="",lwd=1)
lines(cdf.samp, do.points=FALSE,lwd=2)
lines(cdf.est[[1]],lwd=2,col.vert="grey60",col.hor="grey60",
    do.points=FALSE)
legend("bottomright",lwd=c(1,2,2),bty="n",
    col=c("black","grey60","black"),
    legend=c("Population","Weighted estimate",
    "Unweighted estimate"))
```

Figure 4.7 Code for cumulative distribution function of California school size

Figure 4.8 shows the estimated survival curves for time to relapse in Wilms' tumor, using data from a two-phase case–cohort subsample of the National Wilms' Tumour

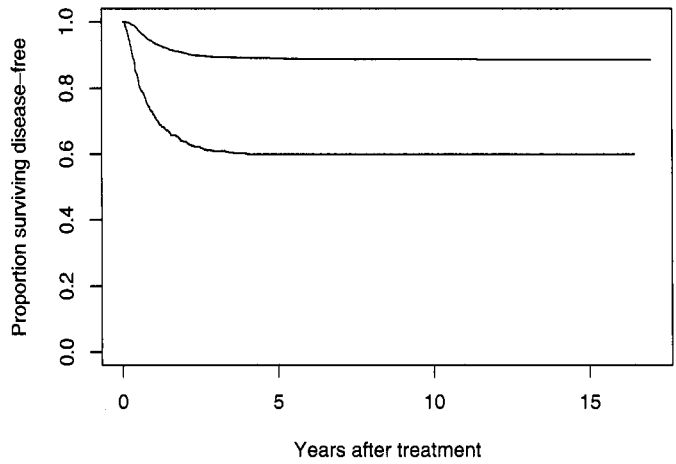

Figure 4.8 Disease-free survival in Wilms' tumor, for favorable and unfavorable histology

Study population, which is described in more detail in section 8.4.3. The code to estimate the survival curves is

```
scurves <- svykm(Surv(edrel/365.25,rel)~histol, dcchs),
```

where `dcchs` is name of the survey design object and the `Surv()` function, from the survival package wraps up the time to event and the event indicator into a single object. The graph is then drawn with `plot(scurves)`.

The survival curves show that relapse tends to happen within about three years, when it happens at all, and that prognosis is good for the 88% of children whose tumors have favorable histological classification. In this example the use of sampling weights makes a very large difference to the estimate: the case-cohort sample includes all relapses, and so the incorrect, unweighted estimates of relapse rates are much higher than the weighted estimates.

Boxplots. Traditional boxplots are based on quartiles of the data: the box shows the median and first and third quartiles, the whiskers extend out to the last observation within 1.5 interquartile ranges of the box ends, and all points beyond the whiskers are shown individually. `svyboxplot()` approximates this for a large, weighted sample. The box shows the estimated median and first and third quartiles, but the whiskers do not necessarily end at an observed point: they extend by 1.5 interquartile ranges or to the most extreme point, whichever is shorter. Only the most extreme point in each direction is shown explicitly, if it is outside the range of the whiskers.

Figure 4.9 uses the NHANES 2003–2004 demographic and blood pressure data that are loaded in section 1.4.2 and merged in section B.2.1. The survey design object uses the single-stage with-replacement approximation to the multistage NHANES design. The `agegp` variable is created by categorizing the reported age with `cut()`,

```
nhanes<-svydesign(id=~SDMVPSU, strat=~SDMVSTRA,
    weights=~WTMEC2YR, data=both, nest=TRUE)
nhanes<-update(nhanes,
    agegp=cut(RIDAGEYR,c(0,20,30,40,50,60,80,Inf), right=FALSE))
svyboxplot(BPXSAR~agegp, subset(nhanes, BPXSAR>0), col="gray80",
    varwidth=TRUE, ylab="Systolic BP (mmHg)",xlab="Age")
```

Figure 4.9 Drawing boxplots of systolic blood pressure by age

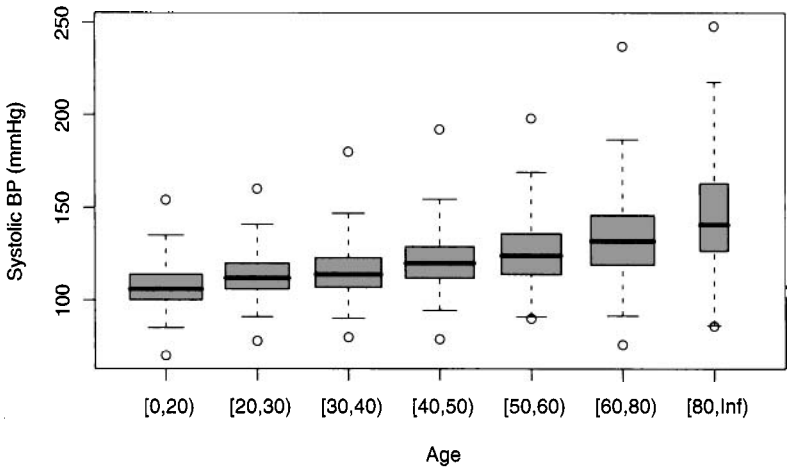

Figure 4.10 Systolic blood pressure by age in NHANES 2003–2004

and the boxplot uses only non-zero observations of blood pressure. Zero is a permitted value but implies a failure to hear the Korotkoff sounds that mark arterial flow changes and are used to define systolic and diastolic pressure. The boxplot in Figure 4.10 shows an increase in both typical level and variability of systolic blood pressure with age. The varying widths of the boxes indicate the varying population sizes for each group.

4.3.2 Graphs based on the density

Histograms and kernel density estimators in unweighted samples are designed to depict the probability density function that the data came from. Technically, when sampling from a finite population there is no smooth population density to estimate, but this is not a problem in practice. If we had complete population data we would still want to smooth or bin the data for display, and this is what sampling-weighted versions of the histogram and kernel density do.

The height of a bar in a histogram is the proportion of data in each bin divided by the width of the bin. Using sampling weights this is replaced by the estimated

```
svyhist(~BPXSAR, subset(nhanes,RIDAGEYR>20 & BPXSAR>0),main="",
    col="grey80", xlab="Systolic BP (mmHg)")
lines(svysmooth(~BPXSAR, bandwidth=5,
    subset(nhanes,RIDAGEYR>20 & BPXSAR>0)),lwd=2)
```

Figure 4.11 Drawing histograms and smooth density estimates of blood pressure.

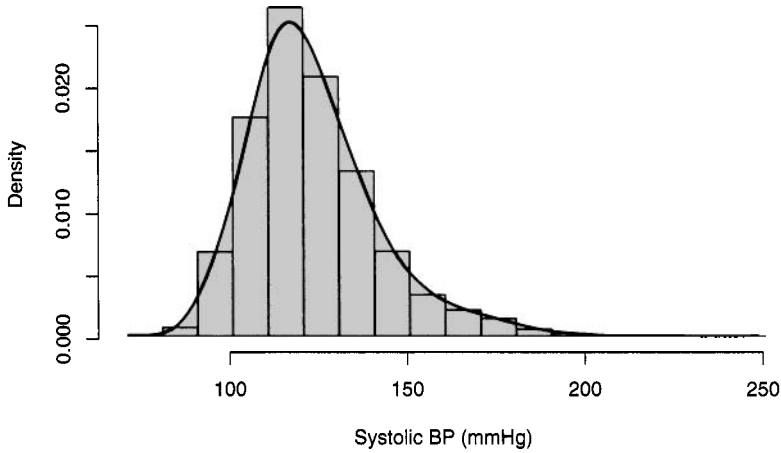

Figure 4.12 Distribution of systolic blood pressure in adults (NHANES 2003–2004)

population proportion of data in the bin divided by the width of the bin. The function `svyhist()` chooses the number of bins automatically, using the *ad hoc* but successful rule of choosing the same bins as for an unweighted histogram of the sample. The effectiveness of histograms is reduced by the discontinuities at the edges of the bins, and smooth density estimators can be more useful. `svysmooth()` fits a weighted density estimate using the local linear smoother from the KernSmooth package [182], as described by Wand and Jones [183].

Figure 4.11 shows code for a histogram and a density estimate for systolic blood pressure in adults, using the data from NHANES 2003–2004. Both `svyhist()` and `svysmooth()` take a one-sided formula as an argument to specify the variable, and a survey design object. `svysmooth()` also requires a bandwidth argument, analogous to the bin width for a histogram. Figure 4.12 shows the output from the code. The distribution appears to be better estimated by the smooth density, but the histogram does have the advantage that clinically important values such as 140mmHg are depicted exactly, as breaks between bins.

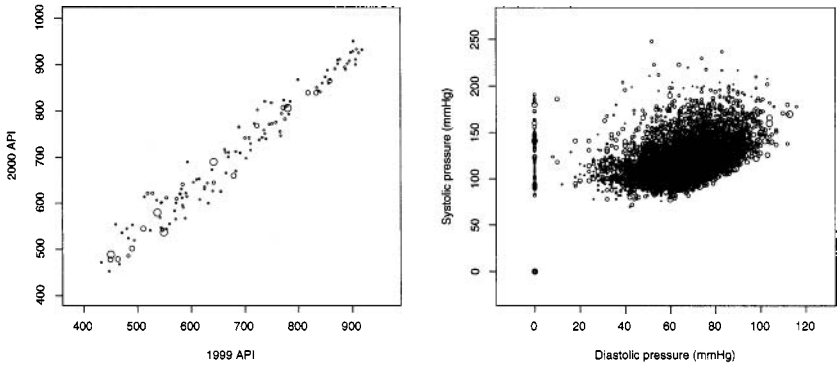

Figure 4.13 Representing sampling weights by glyph size: change in Academic Performance Index from 1999 to 2000, and relationship between systolic and diastolic blood pressure in NHANES 2003–2004.

4.4 TWO CONTINUOUS VARIABLES

The challenges of data visualization under complex sampling are clearest for scatterplots. A scatterplot consists of a single *glyph*, or plotting symbol, for each plotted point, so it is necessary to find some way to represent the sampling weights in that glyph, and to mitigate the problems of overplotting with large numbers of points. As far as I know, there has been no empirical evaluation of any of the techniques in this section in the context of complex samples, but they all appear useful in examples. My personal preference is bubble plots for small data sets, transparent scatterplots for large data sets where color is needed, and hexagonal binning for large data sets where color is not needed.

4.4.1 Scatterplots

A direct way to represent the sampling weights is with the size of the plotted glyph, leading to "bubble plots" such as Figure 4.13. Each point is plotted as a circle with area proportional to the sampling weight. The left-hand panel shows a graph of 1999 vs 2000 Academic Performance Index based on the two-stage cluster sample described in section 3.2. The right-hand panel shows systolic and diastolic blood pressure in the data from NHANES 2003–2004 used in section 1.4.

As Figure 4.13 illustrates, bubble plots can be useful for small data sets. In the left panel, it is useful to know that the outliers above the main trend do not have large sampling weights. In the right panel, however, the individual bubbles are not visible and the bubble plot does not give a clear picture of the trend.

An alternative is to vary the intensity of shading of the points to indicate sampling weight. Figure 4.14 shows the same two data sets drawn with shading proportional to sampling weight. This partial transparency allows superimposed points to be seen, showing where the data are more or less dense. In contrast to the bubble

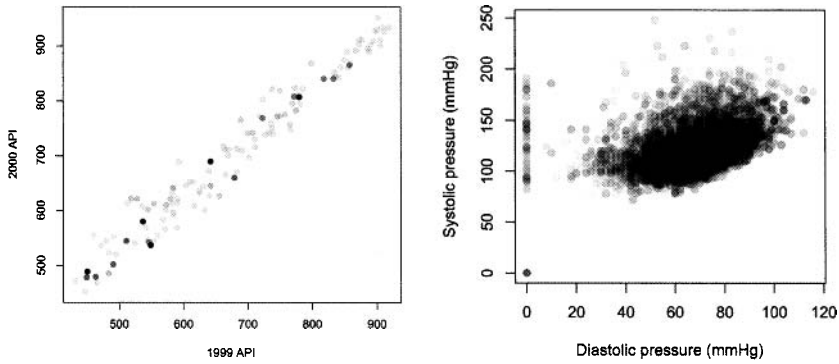

Figure 4.14 Representing sampling weights by shading: change in Academic Performance Index from 1999 to 2000, and relationship between systolic and diastolic blood pressure in NHANES 2003–2004

plot, the transparent scatterplot is more useful with large data sets. In the small sample in the left-hand panel the points with large sampling weight stand out as individuals, and the points with small weight are nearly invisible. In the right-hand panel, transparency reveals more structure in what was previously a solid blob. As Figure 4.14 unfortunately shows, transparency does not render as clearly on paper as on a computer screen.

Both bubble plots and transparent scatterplots are produced by `svyplot()`, which uses a model formula to specify graphs in the same way as the standard `plot()`, with the data coming from a survey design object rather than a data frame. The plotting style is specified by the `style` argument. Code for the two NHANES panels is in Figure 4.15.

```
svyplot(BPXSAR~BPXDAR,design=dhanes, style="bubble",
  xlab="Diastolic pressure (mmHg)",
  ylab="Systolic pressure (mmHg)")
svyplot(BPXSAR~BPXDAR,design=dhanes, style="transparent",
  pch=19,alpha=c(0,0.5), xlab="Diastolic pressure (mmHg)",
  ylab="Systolic pressure (mmHg)")
svyplot(BPXSAR~BPXDAR,design=dhanes, style="subsample",
   xlab="Diastolic pressure (mmHg)",
   ylab="Systolic pressure (mmHg)", sample.size=1000)
svyplot(BPXSAR~BPXDAR,dhanes,style="hex",legend=0,
   xlab="Diastolic pressure",ylab="Systolic pressure")
```

Figure 4.15 Scatterplots of systolic and diastolic blood pressure from NHANES 2003–2004

In the second line of code, which draws the transparent plot, `pch=19` requests a filled circle as the plotting symbol and the `alpha` argument give the range of opacity

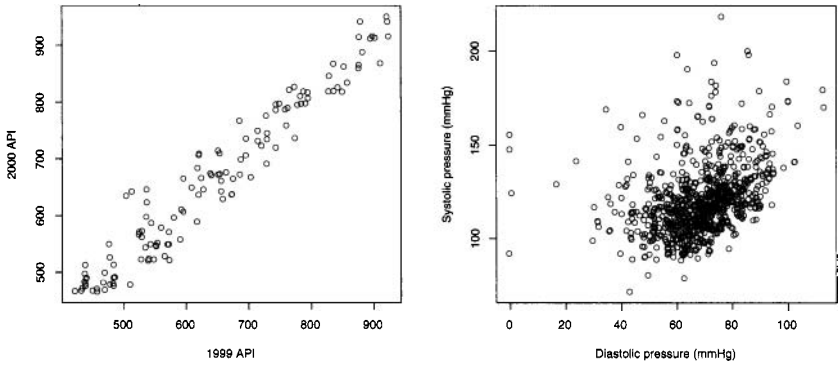

Figure 4.16 Resampling to create an unweighted scatterplot: change in Academic Performance from 1999 to 2000, and relationship between systolic and diastolic blood pressure in NHANES 2003–2004

to be used: 0 for a sampling weight of zero and 0.5 for the largest sampling weight. Bubbleplots and transparent scatterplots can be augmented by plotting the points in different colors to indicate a third, categorical variable. In R this is specified with the `col` argument to `svyplot()`.

A third approach to constructing a scatterplot that is not distorted by complex sampling is to convert the data into a simple random sample by *inverse sampling*. An independent sample of size m with replacement from the data set, with probability proportional to the sampling weight, is approximately a simple random sample from the population. Points that are under-represented in the sample will have higher sampling weight and so will be subsampled with higher probability. The approximation to a simple random sample becomes better as n increases and also as m/n decreases. It is reasonable to hope that a scatterplot of the subsample of size m will be easier to interpret, because it is effectively a familiar scatterplot of a simple random sample. If the variables are continuous, a better approximation to the population distribution would result from smoothing the sample, and this can be implemented by adding random error to the m sampled values. It is always advisable to repeat a subsampled scatterplot a few times to ensure that interesting visual features are not just chance findings with one particular subsample. This subsampling approach was described by Korn and Graubard [83] (without the addition of noise for smoothing).

Figure 4.16 shows plots of the same data sets as Figures 4.14 and 4.13, using resampling from the data set. The left-hand panel has $m = n$ and added noise of 25, which is the median year-to-year change in the Academic Performance Index, and so is a reasonable "measurement error" value. The right-hand panel is a 'thinned' plot, based on a subsampled with $m = 1000$ that is much smaller than the full sample. It has just enough noise to make overlapping points detectable. Code for the right-hand panel is in Figure 4.15. The `style` argument to `svyplot()` requests a subsampled plot and `sample.size` specifies the subsample size m.

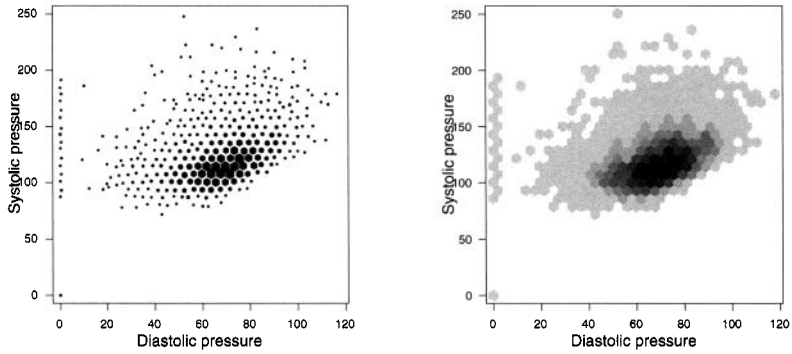

Figure 4.17 Hexagonal binned scatterplot of systolic and diastolic blood pressure from NHANES 2003–2004. The area or shading intensity is proportional to the total sampling weight in the hex.

4.4.2 Aggregation and smoothing

Plotting a symbol for each point in the data set leads to large graphics files. The transparent scatterplot of blood pressure in NHANES 2003–2004 gives a 5.5 Mb file even with only 9800 points. Screen display and printing of very large files is slow, and can even crash printer drivers or viewing software. The problems of large file size can be removed by aggregating the data or by replacing the scatterplot with a series of smooth curves giving trends in means or quantiles. These approaches scale efficiently to data sets of millions of points.

Hexagonal binning involves dividing the plotting surface into a grid of hexagons and combining all the points that fall in a grid cell into a single plotted hexagon whose shading or size indicates the number of points in the bin (Carr et al. [28]). The hexbin package [29] provides hexagonal binning and is used by the `"hex"` and `"grayhex"` styles of `svyplot()`. In the complex-sampling context the number of points in a hexagonal bin is replaced by the sum of the weights for points in the bin, i.e., the estimated number of individuals in the population that fall in the bin.

Figure 4.17 shows the scatterplot of blood pressures from NHANES 2003–2004 using hexagonal binning. The left-hand panel plots hexagons with area proportional to the sum of sampling weights in the bin. The right-hand panel plots hexagons of fixed size, with the shading intensity proportional to the sum of sampling weights. Both plots show the structure of the large blob of points well. In contrast to the transparent plot, individual outlying points are still visible. The graphics files for Figure 4.17 are 30–35kb in size, much smaller than those for the transparent scatterplot.

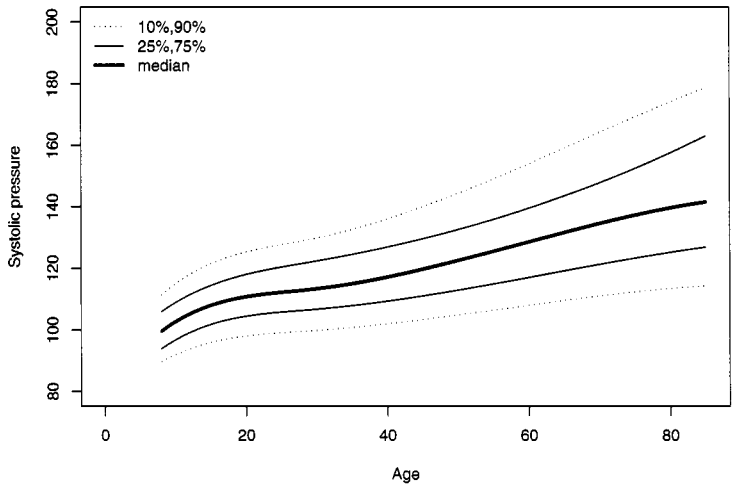

Figure 4.18 Quantiles of systolic blood pressure by age, using data from NHANES 2003–2004

4.4.3 Scatterplot smoothers

The boxplot in Figure 4.10 shows how the quartiles of systolic blood pressure vary with age in the NHANES 2003–2004 sample, but only for discrete categories of age. Figure 4.18 shows estimates of the 10th, 25th, 50th, 75th, and 90th percentiles of systolic blood pressure as smooth functions of age. As with the boxplots it is clear that blood pressure is increasing with age and that the distribution is more spread out at higher ages, presumably indicating different rates of increase for different individuals. Because the x-axis is numeric in this plot, in contrast to the discrete categories in Figure 4.10, it is easier to see the trend and to be confident that it is not being distorted by the way ages are categorized. The relationship between the smooth curves in Figure 4.18 and the boxplots in Figure 4.10 is analogous to the relationship between the histogram and smooth density estimate in Figure 4.12.

The curves are estimated by quantile regression (Koenker [79]), using the quantreg package [80]. They are similar in concept to the kernel estimators of conditional percentiles described by Korn and Graubard [83], but are quite different in computational details. The code is given in Figure 4.19. Each call to `svysmooth()` estimates one quantile curve, specified by the `quantile` argument. The flexibility of the curve is controlled by the `df` argument, which is the number of degrees of freedom for a natural cubic spline curve. The calls to `svysmooth()` differ from those for density estimation in Figure 4.11 in having `method="quantreg"` to request quantile regression and in having a two-sided model formula, with the left-hand side specifying the y-axis variable.

In addition to densities and smooth quantile curves, `svysmooth()` can also estimate smooth mean curves. These are requested with `method="locpoly"` and with the smoothness of the curve specified by the `bandwidth` argument (as in Fig-

```
adults<-subset(dhanes, !is.na(BPXSAR))

a25<-svysmooth(BPXSAR~RIDAGEYR,method="quantreg",design=adults,
  quantile=0.25,df=4)
a50<-svysmooth(BPXSAR~RIDAGEYR,method="quantreg",design=adults,
  quantile=0.50,df=4)
a75<-svysmooth(BPXSAR~RIDAGEYR,method="quantreg",design=adults,
  quantile=0.75,df=4)
a10<-svysmooth(BPXSAR~RIDAGEYR,method="quantreg",design=adults,
  quantile=0.10,df=4)
a90<-svysmooth(BPXSAR~RIDAGEYR,method="quantreg",design=adults,
  quantile=0.90,df=4)

plot(BPXSAR~RIDAGEYR,data=both,type="n",ylim=c(80,200),
  xlab="Age",ylab="Systolic pressure")

lines(a50,lwd=3)
lines(a25,lwd=1)
lines(a75,lwd=1)
lines(a10,lty=3)
lines(a90,lty=3)

legend("topleft",legend=c("10%,90%","25%,75%","median"),
  lwd=c(1,1,3),lty=c(3,1,1),bty="n")
```

Figure 4.19 Smooth quantile functions for systolic blood pressure by age

ure 4.11) rather than the `df` argument. Examples of smooth mean curves will be given in Chapter 5.

For both smooth quantile and smooth mean curves the output can be quite sensitive to the amount of smoothing, as explored in exercise 4.9. For independently sampled observations there are automated techniques to choose the amount of smoothing that gives the most accurate fit to an underlying true curve, but these do not seem to have been adapted to complex samples. When smooth curves are being used for exploratory or descriptive purposes, as in the examples in this book, the "Goldilocks" approach is often adequate. That is, fit the curves with different amounts of smoothness to find examples that are "too rough" and "too smooth", and hope that an intermediate amount of smoothness will be "just right."

4.5 CONDITIONING PLOTS

A third variable can be introduced into scatterplots (and other graphs) by conditioning or 'faceting' the plot. This involves creating an array of scatterplots with the same

x-axis and y-axis scales with each scatterplot showing a subset of the data based on the third variable. The conditioning variable can be discrete, in which case each plot represents a single value, or continuous, so that each plot represents a range of values. When the conditioning variable is continuous it turns out that using overlapping ranges, rather than disjoint ranges, gives more useful graphs. The use of conditioning was developed in detail by Cleveland and coworkers in the Trellis display system [4, 35], and implemented for R by Sarkar [147, 148].

Transparent scatterplots and hexagonally binned scatterplots can be produced by `svycoplot()`. Figure 4.20 shows hexagonally binned scatterplots conditioned on age, using the data from NHANES 2003–2004. The two plots differ in how the hexagon sizes are scaled: in the upper plot they are scaled separately for each panel, in the lower plot the scales are the same across all panels. The shaded strip above each panel shows the range of ages in that panel. Each panel has the same number of observations, and there is a 50% overlap between panels, so that most points appear twice. The overlap smooths the relationship between panels and makes it easier to see trends.

The code for creating the upper plot is

```
svycoplot(BPXSAR~BPXDAR|equal.count(RIDAGEMN), style="hex",
  design=subset(dhanes,BPXDAR>0), xbins=20,
  strip=strip.custom(var.name="AGE"),
  xlab="Diastolic pressure",ylab="Systolic pressure")
```

The conditioning variable or variables are indicated by the | symbol in the formula. In this example the conditioning variable is a *shingle*, a set of overlapping ranges, produced by the `equal.count()` function. The `xbins` arguments specifies the number of hexagonal bins, and the `strip` argument specifies a more readable name for the conditioning variable, to be used in the strips heading each panel. The lower plot has the additional argument `hexscale="absolute"` to make the hexagon scales comparable between panels.

Transparent scatterplots are specified by the `style="transparent"` argument to `svycoplot()`. They produce larger graphics files that do not print as attractively on paper, but they use the limited resolution of a computer screen more efficiently and have the advantage of being able to show an additional variable by using different colors for points. For example, in a transparent version of Figure 4.20, different colors could be used to indicate gender. The resulting plot, available on the web site, shows that blood pressure is initially lower for men, but that women catch up in the last two panels.

4.6 MAPS

4.6.1 Design and estimation issues

Many large complex surveys are national in scope, and include at least some geographic information on participants, so it is natural to compute estimates for geographic areas and display them on a map. Although specialized GIS (Geographical

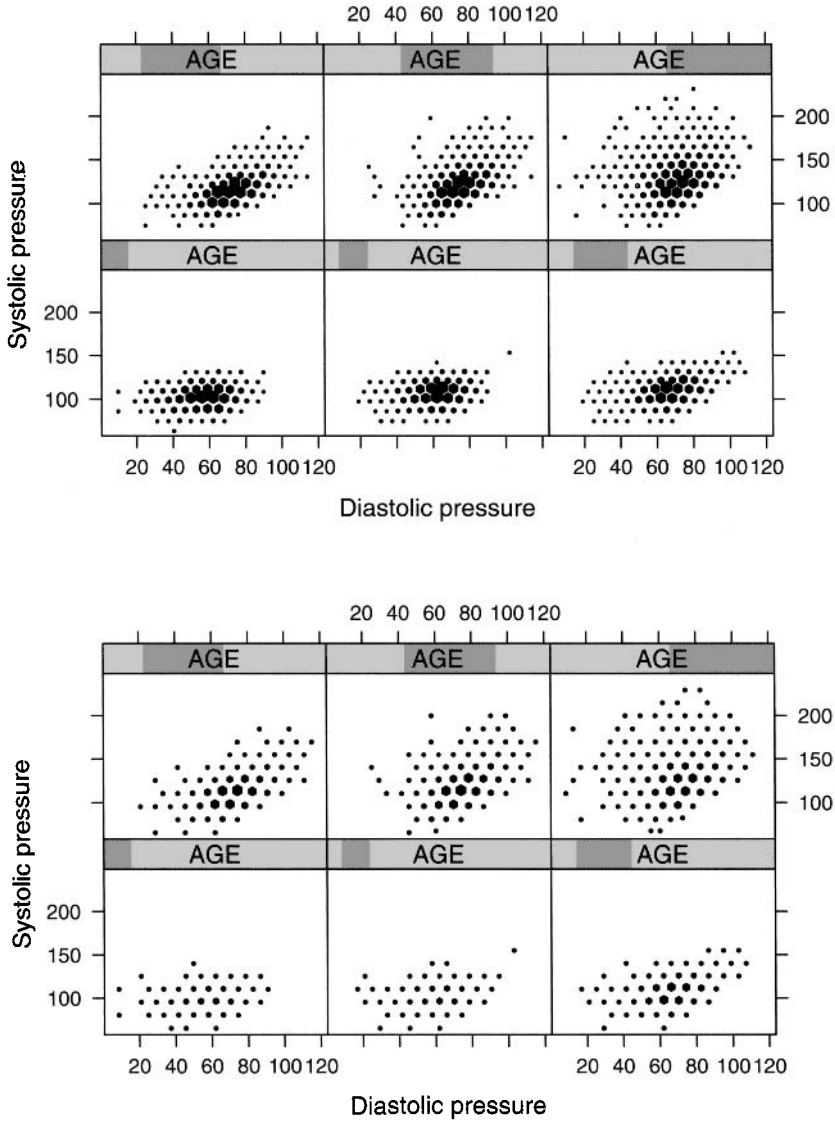

Figure 4.20 Relationship between systolic and diastolic blood pressure by age, using data from NHANES 2003–2004. In the upper plot the hexagons are scaled separately for each panel, in the lower plot the scales are the same across all panels

Information Systems) software is available, combining graphical, geographical, and database facilities, it is not always necessary and R is capable of producing useful statistical maps.

The primary operational difficulty in turning a table of regional estimates into a map is obtaining the geographical information needed to draw the map. In the United States the Census Bureau publishes the public-domain TIGER/Line database of states, counties, cities, postal codes, and streets, but in other countries such data are typically not free and may be expensive. The R packages maps [6] and mapdata [5] include US maps at the county level and world maps at the national level (but with national boundaries that are seriously outdated in areas such as Eastern Europe and Central Asia). The maptools package can read and write many popular data formats[93], making it possible to use R together with a GIS package when desirable.

A second difficulty, especially when dealing with geographical data from more than one source, is that the earth is not flat. Locations identified by latitude and longitude must be projected onto a flat page, and some distortion is inevitable. For single countries, not too close to the poles, there are many satisfactory projections, but it is still important to ensure that all data are using the same projection. When mapping small areas it becomes important that the earth is not even round: different approximations to the shape of the earth lead to differences of tens of meters in mapped locations. The R package mapproj performs a fairly wide variety of projections [106].

Figure 3.1 uses a rectangular projection for the US, which ensures that the vertical and horizontal distance scales match at the center of the map. There is definite distortion of distance at the top and bottom; a degree of longitude is nearly 10% shorter along the Canadian border than in south Florida. The US maps in this chapter use an Albers projection that accurately represents areas and has accurate horizontal scale along the 45.5N and 29.5N latitude lines.

When sample sizes are sufficiently large, computing the estimates for each geographic area is no different from any other subpopulation estimation. When sample sizes are small, however, the spatial structure provides extra opportunities to use data from adjacent regions to improve estimates. A wide range of model-based and design-based approaches have been proposed; references are Rao [129], Elliot et al. [45], and the review paper by Ghosh and Rao [52].

A further complication is that population density can be extremely variable. Regions such as census districts, zipcode regions, or counties, tend to be larger when they have smaller population, so rural areas tend to visually dominate a map. This can lead to misleading visual impressions either when rural areas are systematically different on the variables being estimated, or when the sampling uncertainty is large enough that some rural regions appear different due to chance.

A well-known example of systematic differences between rural and urban areas is the US electoral maps in which states are colored red or blue according to whether the Democratic or Republican candidate has a larger share of the vote. The "red" states have a lower average population density and so take up a much larger fraction of the map even when the voting proportions are about equal. There is no entirely satisfactory solution to this problem, but a Google search for "election 2004 maps"

will show a range of attempts. Two more are on the web site for this book, based on adding color to Figure 3.1.

Color scales are an important part of map design, though not one that can be addressed well in a black-and-white book. An important distinction is between *sequential* and *diverging* scales. A diverging scale measures distance from a midpoint, a sequential scale measures distance from the endpoint(s). For example, in a US election map the most important value is the 50% at the center of the scale, and useful color schemes diverge symmetrically from this center. If the endpoints are red and blue the center point could be purple or white. A red–white–blue scale makes it easier to see the directions of small deviations from 50%, a purple scale was used by Vanderbei [180] to de-emphasize small deviations. Sequential color scales are appropriate for most of the measurements collected by the Behavioral Risk Factor Surveillance System, which is the source of the examples in this section. For the risk factors targeted by BRFSS, less is always better and only unidirectional comparisons are needed.

A useful tool for choosing color scales for maps is ColorBrewer (Harrower & Brewer [58]), which demonstrates a range of color scales designed specifically for statistical maps. These color scales are conveniently available in the RColorBrewer package [119]. A range of grays makes an effective sequential scale, although there are perceptual advantages to a color scale of uniform brightness. Diverging color schemes, on the other hand, never render well in black-and-white.

As further resources, Monmonier [110] covers the use of maps to communciate social research (the same author has written several other books on maps), Brewer [21] discusses visual design based around examples, and Elliot et al [45] examine many issues of inference relevant to mapping data from surveys.

4.6.2 Drawing maps in R

The Behavioral Risk Factor Surveillance System has developed an online mapping tool (`http://apps.nccd.cdc.gov/gisbrfss/default.aspx`) to display estimates for states and for metropolitan areas that have a sufficiently large sample size. The spatial data used by this application is available for download, from the BRFSS website and consists of high-resolution state outlines in the standard "shapefile" format. These have already been projected to a flat space using the Albers projection and can be plotted without further transformation.

Figure 4.22 shows code to read in the shape files, to estimate average fruit consumption in each state from the BRFSS design constructed in Figure 3.7, merge this with the shapefile data, and create a plot. The "FIPS" in the variable name is Federal Information Processing Standard 6-4, which sets out numeric codes for US counties and states [121], and the `merge()` call uses `all=FALSE` to drop estimates for Guam, Puerto Rico, and the US Virgin Islands. It is necessary to ensure that the state shape data are in order by FIPS code before merging, otherwise the relationship between shape data and variables can be destroyed. The function `spplot()` from the sp package [125] draws the map using the shape data from the `states` object and fills in

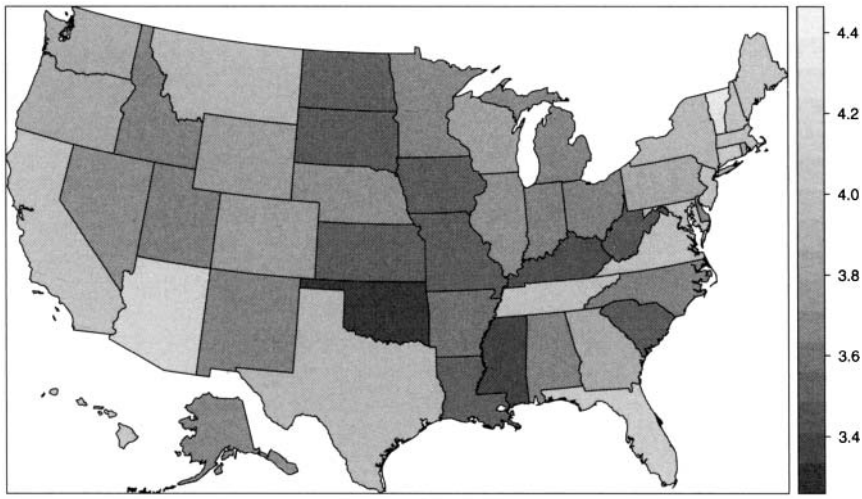

Figure 4.21 Servings of fruit per day, from BRFSS 2007

```
library(maptools)
states<-readShapePoly("brfss_state_2007_download")
states<-states[order(states$ST_FIPS),]

bys<-svyby(~X_FRTSERV,~X_STATE,svymean,
    design=subset(brfss,X_FRTSERV<99999))

states@data<-merge(states@data, bys, by.x="ST_FIPS",
    by.y="X_STATE",all=FALSE)
states$fruitday<-states$X_FRTSERV/100

spplot(states,"fruitday")
```

Figure 4.22 Drawing a map of average fruit consumption from BRFSS 2007

each state according to the variable `fruitday`, which is average number of servings of fruit per day.

Figure 4.23 creates a set of four linked maps showing health insurance coverage by age. The state map data and the BRFSS survey data are the same as in Figure 4.22, but the estimation step uses age groups $\leq$ 35, 35–50, 50–65, and $>$ 65 in addition to state to specify subpopulations. The `reshape()` function reshapes the data set of estimates from 4×54 records in one column to 54 records in four columns.

The data frame is merged with the state shape data in the same way as the previous example. Finally, the call to `spplot()` specifies that the variables `age1`–`age4` should

```
brfss<-update(brfss, agegp=cut(AGE, c(0,35,50,65,Inf)))
hlth<-svyby(~I(HLTHPLAN==1), ~agegp+X_STATE, svymean,
    design=brfss)

hlthdata<-reshape(hlth[,c(1,2,4)],idvar="X_STATE",
    direction="wide",timevar="agegp")
names(hlthdata)[2:5]<-paste("age",1:4,sep="")

states@data<-merge(states,hlthdata,
    by.x="ST_FIPS",by.y="X_STATE",all=FALSE)
spplot(states,c("age1","age2","age3","age4"),
    names.attr=c("<35","35-50","50-65","65+"))
```

Figure 4.23 Drawing a map of health insurance coverage by age with BRFSS 2007

be used to color the states, resulting in a set of four graphs with the same color scale. The `names.attr` option specifies names for each of the maps, corresponding to the age groups. Figure 4.24 shows the result. Health insurance coverage is very high for people over 65, because Medicare provides nearly universal coverage to citizens and permanent residents in this age group. Coverage is lower and more variable at younger ages. In these younger age groups coverage appears to be particularly low in Nevada and Texas and high in Wisconsin, Minnesota, and Massachusetts. Since Nevada has a fairly small population it is important to check the standard errors. The standard error for Nevada is only 2.6%, so the low insurance coverage is not just sampling variation.

BRFSS also provides summaries of some variables for smaller areas — cities and counties, called MMSAs — through the SMART system. These smaller areas are those where at least 500 responses were received, allowing sufficient precision in estimating local totals. Figure 4.25 shows insurance coverage for states and for the MMSAs on the same map. This requires more effort to ensure that the same color scale is used for the regions as for the points. The map was constructed with the ggplot2 package (Wickham[186]), which implements Wilkinson's [187] "Grammar of Graphics". The code is on the web site for the book.

Most of the MMSAs have similar insurance coverage rates to their surrounding states. There are some fairly obvious exceptions near the Texas–Mexico border with much lower coverage rates, and some less obvious ones in inland California, central Washington, and Florida.

The version of this map on the BRFSS web site uses a color scale with five discrete steps, in which all of Texas in the lowest step, so the MMSAs on the Texas–Mexico border do not show up, but those in California and Washington are clearly visible as being in a lower step than their surrounding states. Figure 4.26 shows the data using a five-level scale than puts about the same number of states into each level, which is close to the online BRFSS map. This example shows how large the impact of

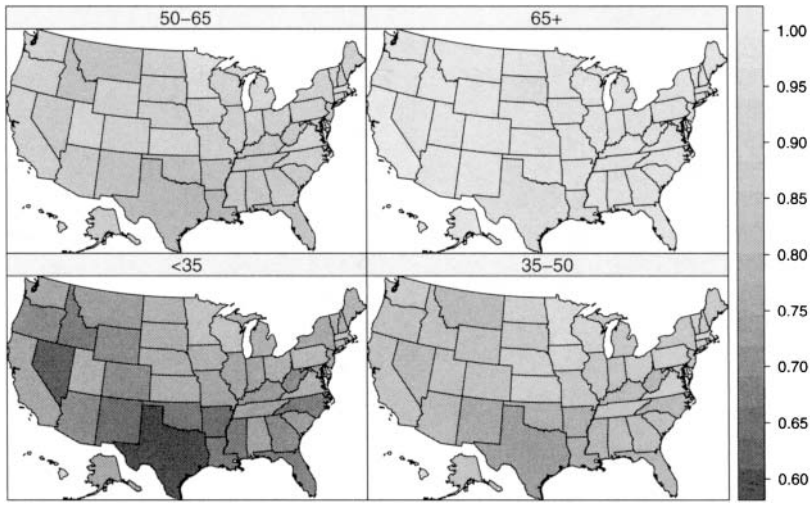

Figure 4.24 Health insurance by age, from BRFSS 2007

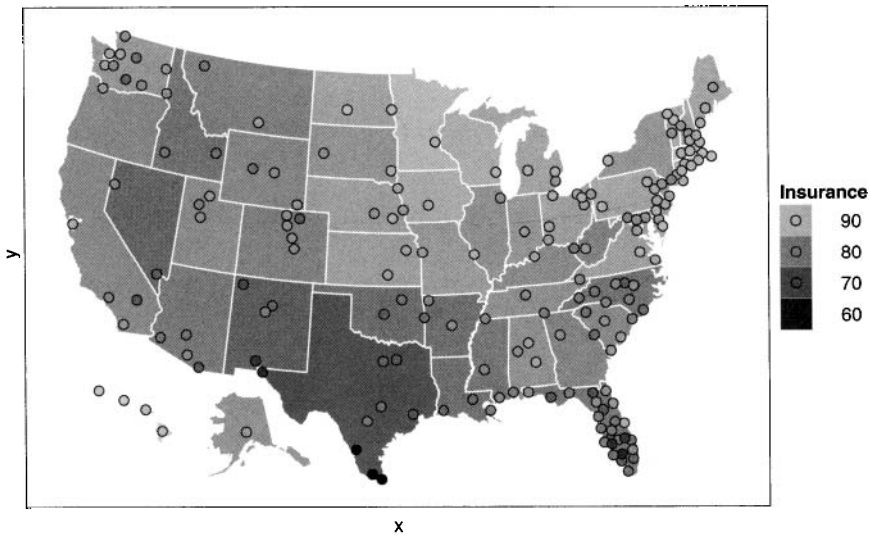

Figure 4.25 Health insurance for states and MMSAs, from BRFSS 2007

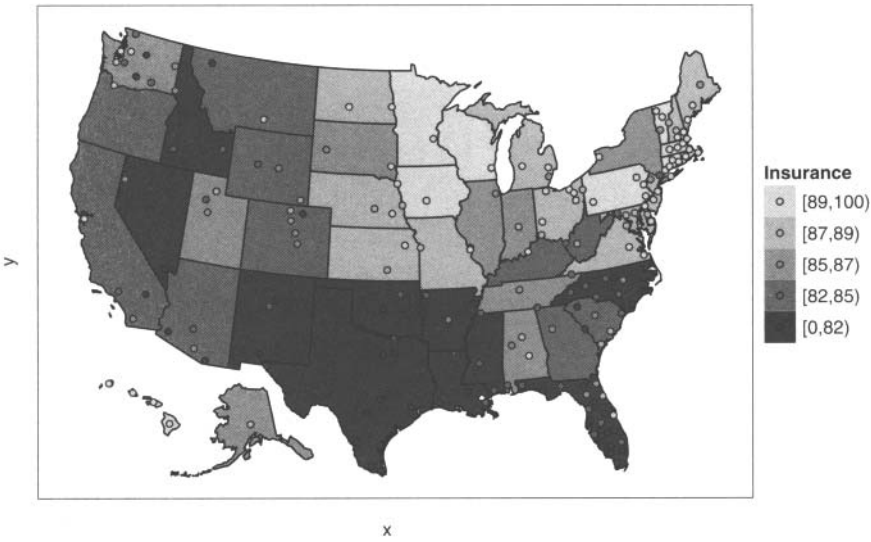

Figure 4.26 Health insurance for states and MMSAs, with a quantile-based color scale

different color scales can be. Because the low values are much lower than the rest of the country, a linear mapping of estimate to grayscale gives quite a different picture to a quantile-based mapping.

EXERCISES

4.1 Draw boxplots of body mass index by race/ethnicity and by sex using the CHIS 2005 data introduced in Chapter 2.

4.2 Using the code in Figure 3.8 draw a barplot of the quantiles of income and compare it to the dotchart in Figure 3.9. What are some advantages and disadvantages of each display?

4.3 Use `svysmooth()` to draw a graph showing change in systolic and diastolic blood pressure over time in the NHANES 2003–2004 data. Can you see the change to isolated systolic hypertension in old age that is shown in Figure 4.5?

4.4 With the data from the SIPP 1996 panel (see Figure 3.8) draw the cumulative distribution function, density function, a histogram, and a boxplot of total household income. Compare these graphs for their usefulness in showing the distribution of income.

4.5 With the data from the SIPP 1996 panel (see Figure 3.8) draw a graph showing amount of rent (`tmthrent`) and proportion of income paid in rent. You will want to exclude some outlying households that report much higher rent than income.

4.6 Using data from CHIS 2005 (see section 2.3.1) examine how body mass index varies with age as we did with blood pressure in this chapter.

4.7 The left-hand panel of Figure 3.3 shows an interesting two-lobed pattern. Can you find what makes these lobes?

4.8 Set up a survey design for the BRFSS 2007 data as in Figure 3.7. BRFSS measured annual income (`income2` in categories < $10k, $10–15k, $15–20k, $20–$25k, $25–35k, $25–50k, $50–75k, > $75k) and race (`orace`: white, black, asian, native hawaiian/pacific island, native american, other).

a) Draw a graph of income by race.

b) Draw maps showing the geographical distribution of income and of racial groups.

c) Draw a set of maps examining whether the geographical distribution of income differs by race.

4.9 Explore the impact on the graphs in Figure 4.18 of changes in the amount of smoothing, by altering the `df` argument to the code in Figure 4.19.