The *first* argument to `plot` is the x-axis variable, and the *second* argument is the y-axis variable. The scatterplot in Figure 3.7 is a *summary graph* for the regression problem in which `prestige` is the response and `income` is the predictor. As our eye moves from left to right across the graph, we see how the distribution of `prestige` changes as `income` increases. In technical terms, we are visualizing the *conditional distributions* of `prestige` given values of `income`. The overall story here is that as `income` increases, so does `prestige`, at least up to about $10,000, after which the value of `prestige` stays more or less fixed on average at about 80.

We write E(`prestige|income`) to represent the mean value of `prestige` as the value of `income` varies and call this the *conditional mean function* or the *regression function*. The qualitative statements in the previous paragraph therefore concern the regression function. The *variance function*, Var(`prestige|income`), traces the conditional variability in `prestige` as `income` changes—that is, the spread of y in vertical strips in the plot. As in Figure 3.7, when the tilt of a scatterplot changes, it is difficult to judge changes in conditional variability from a simple scatterplot.

Scatterplots are useful for studying the mean and variance functions in the regression of the y-variable on the x-variable. In addition, scatterplots can help us identify *outliers*—points that have values of the response far different from the expected value—and *leverage points*—cases with extremely large or small values of the predictor. How these ideas relate to multiple regression is a topic discussed in Chapter 6.

PLOT ENHANCEMENTS

Scatterplots can be enhanced by adding curves to the graphs and by identifying unusual points. A *scatterplot smoother* provides a visual estimate of the regression function, either using a statistical model such as simple linear regression or nonparametrically, without specifying the shape of the regression curve explicitly.

The `scatterplot` function in the **car** package draws scatterplots with smoothers, as in Figure 3.8:

```
> scatterplot(prestige ~ income, span=0.6, lwd=3,
+     id.n=4, data=Prestige)

[1] "general.managers" "physicians"       "lawyers"
[4] "ministers"
```

The variables in the scatterplot are given in a *formula*, separated by a tilde (~), with the response variable on the left and the predictor on the right. If, as in this example, the `scatterplot` is specified via a formula, we can use the `data` argument to supply a data frame in which the variables in the formula are located; this is also true for the standard `plot` function.

There are two smoothers on the scatterplot in Figure 3.8. The first is a straight line fit by ordinary least squares (OLS); we can suppress this line with the argument `reg.line=FALSE`. The second, the solid curved line, is

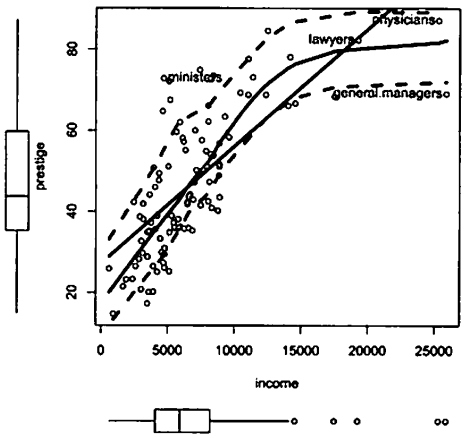

Figure 3.8 Enhanced scatterplot of `prestige` by `income`. Several points were identified automatically.

a nonparametric-regression smoother produced by the `lowess` function in R; the lowess smoother can be suppressed by the argument `smooth=FALSE`.

Lowess is an acronym for *lo*cally *we*ighted *s*catterplot *s*moother, which implements a form of *local linear regression*: Each point along the regression curve is produced by a weighted linear least-squares regression fit to data in the neighborhood of the focal point, with the weights declining as the x values get farther from the x value of the focal point. The degree of smoothness of the lowess line is controlled by a smoothing parameter, called the *span*, representing the fraction of the data included in each local-regression fit; larger spans produce a smoother regression line. The span is analogous to the bandwidth of the kernel-density estimator (described in Section 3.1.2). We generally want to use the smallest span that produces a reasonably smooth result, to achieve a favorable trade-off of bias against variance—balancing roughness against fidelity to the data. In the command above, the span was changed from the default value `span=0.5` to `span=0.6`. One advantage to the lowess smooth is that the fit is relatively insensitive to the choice of smoothing parameter, and values of the span from about 0.3 to about 0.7 generally give useful fitted lines. In very large samples, however, we can often use a smaller span than 0.3, and in very small samples, we may need a larger span than 0.7. In Figure 3.8, the least-squares line cannot match the obvious curve in the regression function that is apparent in the lowess fit.[8]

In addition to the least-squares and lowess lines, by default, `scatterplot` displays a nonparametric estimate of the variance function, showing how the conditional spread of the y-values changes with x. The broken lines in the graph are based on smoothing the residuals from the lowess line. By smoothing the positive and negative residuals separately, `scatterplot` also

[8]We use a smoother here and in most of this book as a plot enhancement, designed to help us derive information from a graph. Nonparametric regression, in which smoothers are substituted for more traditional regression models, is described in the online appendix to the book. *Kernel regression*, which is similar to lowess, is described in Section 7.2.

helps us detect asymmetry in the conditional distribution of y. In interpreting the conditional spread, however, we must be careful to focus on the *vertical* separation of the two broken lines. In Figure 3.8, the conditional spread of `prestige` remains quite constant as `income` increases, and the conditional distribution of `prestige` seems reasonably symmetric.

The graph in Figure 3.8 also includes marginal boxplots for the two variables, and these can be suppressed with `boxplots=FALSE`. As with the other graphical functions in the **car** package, point labeling is available either automatically or interactively (see Section 3.5); setting `id.n=4` identifies the four most extreme points, rather than the default `id.n=0`, which suppresses point identification.

For clarity, we set the line-width to `lwd=3` for the smoothers in Figure 3.8 to make the regression lines on the plot thicker than they would be by default (`lwd=1`). The default color of nonparametric-regression lines plotted by `scatterplot` is red. To get black points and lines, we can specify the argument `col="black"`. See `?scatterplot` for details.

CODED SCATTERPLOTS FOR GROUPED DATA

Using the categorical variable `type` (type of occupation) in the `Prestige` data set, we could simultaneously condition on both `type` and `income` by drawing a separate plot of `prestige` versus `income` for each level of `type`. When, as here, the categorical variable has only a few levels, we can achieve the same result in a single graph by using distinct colors or symbols for points in the different levels of the categorical variable, fitting separate smoothers to each group. We call the resulting graph a *coded scatterplot*. An example appears in Figure 3.9:

```
> scatterplot(prestige~income|type, data=Prestige, boxplots=FALSE,
+               span=0.75, col=gray(c(0, 0.5, 0.7)), id.n=0)
```

The variables for the coded scatterplot are given in a formula as `y ~ x | g`, which we read as plotting `y` on the vertical axis and `x` on the horizontal axis and marking points according to the value of `g` (or "`y` vs. `x` given `g`").

We selected a large span, `span=0.75`, for the lowess smoothers because of the small number of observations in the occupational groups. The legend for the graph, automatically generated by the `scatterplot` function, can be suppressed with `legend.plot=FALSE`. We set the colors for points and lines with the argument `col=gray(c(0, 0.5, 0.7))`, which generates three levels of gray for the three occupational types. If we omit the `col` argument, `scatterplot` will select the colors for us. The argument `id.n=0` was included as a reminder that we could have specified automatic point marking by setting `id.n` to the number of points to be labeled; this argument was unnecessary because `id.n=0` is the default value.

Figure 3.9 allows us to examine three regression functions simultaneously: E(`prestige`|`income`, `type` = `bc`), E(`prestige`|`income`, `type` =

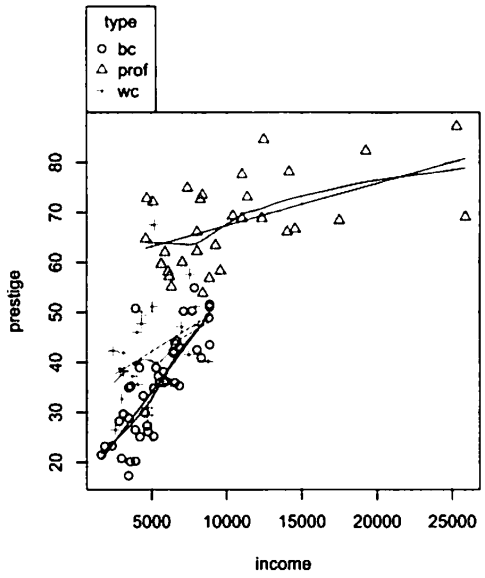

Figure 3.9 Scatterplot of `prestige` by `income`, coded by `type` of occupation.

wc), and E(`prestige`|`income`, `type` = `prof`). The nonlinear relationship in Figure 3.8 has disappeared, and we now have three reasonably linear regressions with different slopes. The slope of the relationship between `prestige` and `income` looks steepest for blue-collar occupations and looks least steep for professional and managerial occupations.

JITTERING SCATTERPLOTS

Discrete, quantitative variables typically result in uninformative scatterplots. The example in Figure 3.10a was produced by the `plot` command:

```
> head(Vocab)

         year    sex education vocabulary
20040001 2004 Female         9          3
20040002 2004 Female        14          6
20040003 2004   Male        14          9
20040005 2004 Female        17          8
20040008 2004   Male        14          1
20040010 2004   Male        14          7

> nrow(Vocab)

[1] 21638

> plot(vocabulary ~ education, data=Vocab)
```

The data for this illustration, from the `Vocab` data frame in the **car** package, come from the U.S. General Social Surveys, 1972–2004, conducted by the National Opinion Research Center. The two variables in the plot are `education` in years and the respondent's score on a 10-word `vocabulary` test.

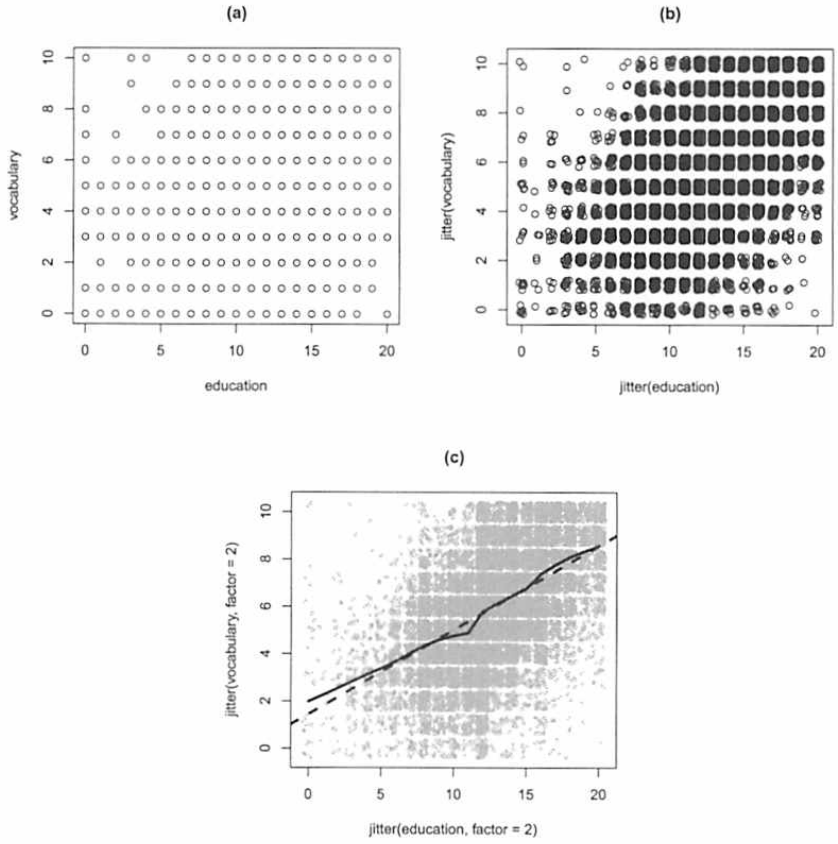

Figure 3.10 Scatterplots of vocabulary by education: (a) unjittered, (b) default jittering, and (c) twice default jittering, with least-squares and lowess lines.

Because `education` can take on only 21 distinct values and `vocabulary` only 11 distinct values, most of the nearly 22,000 observations in the data set are overplotted; indeed, almost all the possible $21 \times 11 = 231$ plotting positions are occupied, producing a meaningless rectangular grid of dots.

Jittering the data by adding a small random quantity to each coordinate serves to separate the overplotted points. We can use the `jitter` function in R for this purpose:

```
> plot(jitter(vocabulary) ~ jitter(education), data=Vocab)
```

The result is shown in Figure 3.10b. We can control the degree of jittering via the argument `factor`; for example, specifying `factor=2` doubles the jitter, yielding a more satisfactory result for the current example:

```
> plot(jitter(vocabulary, factor=2) ~ jitter(education, factor=2),
+     col="gray", cex=0.5, data=Vocab)
```

To render the individual points less prominent, we plot them in gray and use the argument `cex=0.5` to make the points half the default size. To complete the picture, we add least-squares and nonparametric-regression lines, using the original *unjittered* data for these computations, producing Figure 3.10c:

```
> with(Vocab, {
+     abline(lm(vocabulary ~ education), lwd=3, lty="dashed")
+     lines(lowess(education, vocabulary, f=0.2), lwd=3)
+     })
```

The least-squares line on the graph is computed by `lm` and drawn by `abline`; the argument `lwd` to `abline` sets the width of the regression line, while the line type `lty="dashed"` specifies a broken line. The `lowess` function returns the coordinates for the local-regression curve, which is drawn by `lines`; the span of the local regression is set by the argument `f` to `lowess`, and we take advantage of the very large data set by using a small span. The relationship between `vocabulary` and `education` appears nearly linear, and we can also discern other features of the data that previously were hidden by over plotting, such as the relatively large number of respondents with 12 years of education.

We could have more conveniently used the `jitter` argument to the `scatterplot` function in the **car** package to make the graphs in Figures 3.10b and c, but we wanted to demonstrate how to construct a simple plot from its components (a topic described in detail in Chapter 7).

3.2.2 PARALLEL BOXPLOTS

Parallel boxplots help us visualize the conditional distributions of a quantitative response for each of several values of a discrete predictor. We illustrate with data from Ornstein (1976) on interlocking directorates among 248 major Canadian corporations:

```
> some(Ornstein) # sample 10 rows

    assets sector nation interlocks
3   113230    BNK    CAN         94
29   11090    MAN     US         21
57    5021    MIN    OTH         27
123   1427    HLD    CAN          1
150    830    MIN     UK          1
153    802    MAN    CAN          0
154    798    AGR    CAN         11
156    789    MAN     US          6
160    761    WOD     US          1
211    376    MER    CAN          5

> nrow(Ornstein)

[1] 248
```

The variables in the data set include the `assets` of the corporation in millions of dollars; the corporation's `sector` of operation, a *factor* (i.e., categorical variable) with 10 *levels* (categories); the `nation` in which the firm is controlled, `CAN` (Canada), `OTH` (other), `UK`, and `US`; and the number of interlocking directorate and executive positions (`interlocks`) maintained between each company and others in the data set. Figure 3.11a shows a boxplot of the number of `interlocks` for each level of `nation`: