Michael Friendly

Visualizing Multivariate Data and Models in R

Here is where the dedication goes ...

Table of contents

Preface

TODO: Make this a more general introduction

This book is about graphical methods developed recently for multivariate data, and their uses in understanding relationships when there are several aspects to be considered together. Data visualization methods for statistical analysis are well-developed for simple linear models with a single outcome variable. However, with applied research in the social and behavioral sciences, it is often the case that the phenomena of interest (e.g., depression, job satisfaction, academic achievement, childhood ADHD disorders, etc.) can be measured in several different ways or related aspects.

For example, if academic achievement can be measured for adolescents by reading, mathematics, science and history scores, how do predictors such as parental encouragement, school environment and socioeconomic status affect all these outcomes? In a similar way? In different ways? In such cases, much more can be understood from a multivariate approach that considers the correlations among the outcomes. Yet, sadly, researchers typically examine the outcomes one by one which often only tells part of the data story.

However, to do this it is useful to set the stage for multivariate thinking, with a grand scheme for statistics and data visualization, a parable, and an example of multivariate discovery.

ONE, TWO, MANY

There is an old and helpful idea I learned from John Hartigan in my graduate days at Princeton:

> In statistics and data visualization *all* methods can be classified by the number of dimensions contemplated, on a scale of **ONE**, **TWO**, **MANY**.

By this, he meant that, at a global level, all data, statistical summaries, and graphical displays could be classified as:

- **univariate**: a single variable, considered in isolation (age, COVID cases, pizzas ordered). Univariate numerical summaries are means, medians, measures of variablilty, and so forth. Univariate displays include dot plots, boxplots, histograms and density estimates.
- **bivariate**: two variables, considered jointly. Numerical summaries include correlations, covariances and two-way tables of frequencies or measures of association for categorical variables. Bivariate displays include scatterplots and mosaic plots.
- **multivariate**: three or more variables, considered jointly. Numerical summaries include correlation and covariance matrices, consisting of all pairwise values, but also derived measures from the analysis of these matrices (eigenvalues, eigenvectors). Graphical displays of multivariate data can sometimes be shown in 3D, but often involve multiple views of the data projected into 2D plots.

As a quasi-numerical scale, I refer to these as **1D**, **2D** and **nD**. This admits the possibility of half-integer cases, such as **1.5D**, where the main focus is on a single variable, but that is classified by a simple factor (e.g., gender), or **2.5D** where a 2D scatterplot can show other variables using color, shape or other visual attributes His point in this classification was that once you've reached three variables, all higher dimensions involve similar summaries and data displays.

Univariate and bivariate methods and displays are well-known. This book is about how these ideas can be

extended to an n-dimensional world. Three-dimensional data displays are now fairly easy to produce, even if they are sometimes difficult to understand. But how can we even think about four or more dimensions? The difficulty can be appreciated by considering the tale of *Flatland*.

Flatland

> To comport oneself with perfect propriety in Polygonal society, one ought to be a Polygon oneself. — Edwin A. Abbott, *Flatland*

In 1884, an English schoolmaster, Edwin Abbott Abbott, shook the world of Victorian culture with a slim volume, *Flatland: A Romance of Many Dimensions* (Abbott, 1884). He described a two-dimensional world, *Flatland*, inhabited entirely by geometric figures in the plane. His purpose was satirical, to poke fun at the social and gender class system at the time: Women were mere line segments, while men were represented as polygons with varying numbers of sides— a triangle was a working man, but acute isosceles were soldiers or criminals of very small angle; gentlemen and professionals had more sides. Abbot published this under the pseudonym, "A Square", suggesting his place in the hierarchy.

> True, said the Sphere; it appears to you a Plane, because you are not accustomed to light and shade and perspective; just as in Flatland a Hexagon would appear a Straight Line to one who has not the Art of Sight Recognition. But in reality it is a Solid, as you shall learn by the sense of Feeling. — Edwin A. Abbott, *Flatland*

But how did it feel to be a member of a flatland society? How could a point (a newborn child?) understand a line (a woman)? How does a Triangle "see" a Hexagon or even a infinitely-sided Circle? Abbott introduces the very idea of different dimensions of existence through dreams and visions:

- A Square dreams of visiting a one-dimensional *Lineland* where men appear as lines, and women are merely "illustrious points", but the inhabitants can only see the Square as lines.
- In a vision, the Square is visited by a Sphere, to illustrate what a 2D Flatlander could understand from a 3D sphere (Figure 1) that passes through the plane he inhabits. It is a large circle when seen at the moment of its' greatest extent. As the Spehere rises, it becomes progressively smaller, until it becomes a point, and then vanishes.

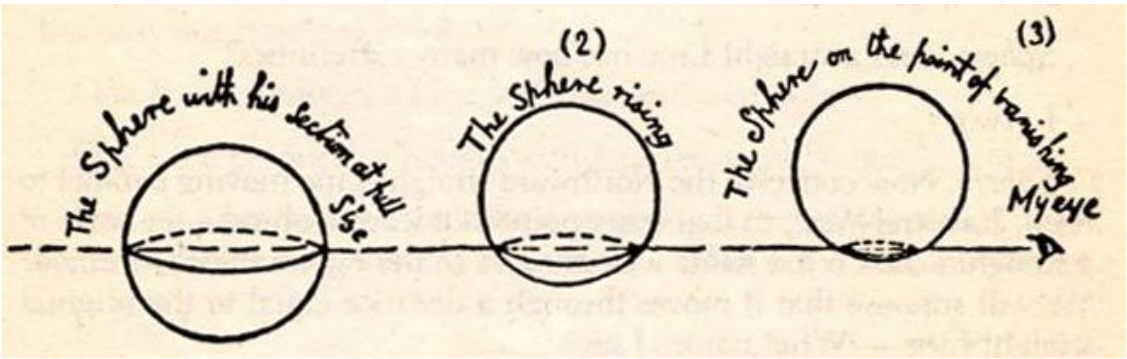


Figure 1: A 2D Flatlander seeing a sphere as it passes through Flatland. The line, labeled 'My Eye' indicates what the Flatlander would see. Source: Abbott (1884)

Abbott goes on to state what could be considered as a demonstration (or proof) by induction of the difficulties of seeing in 1, 2, 3 dimensions, and how the idea motion over time (one more dimension) could allow citizens of any 1D, 2D, 3D world to contemplate one more dimension.

> In One Dimensions, did not a moving Point produce a Line with two terminal points? In two

> Dimensions, did not a moving Line produce a Square with four terminal points? In Three Dimensions, did not a moving Square produce - did not the eyes of mine behold it - that blessed being, a Cube, with eight terminal points? And in Four Dimensions, shall not a moving Cube - alas, for Analogy, and alas for the Progress of Truth if it be not so - shall not, I say the motion of a divine Cube result in a still more divine organization with sixteen terminal points? — Edwin A. Abbott

For Abbot, the way for a citizen of any world to imagine one more dimension was to consider how a higher-dimensional object would change over time.[1] A line moved over time could produce a rectangle as shown in Figure 2; that rectangle moving in another direction over time would produce a 3D figure, and so forth.

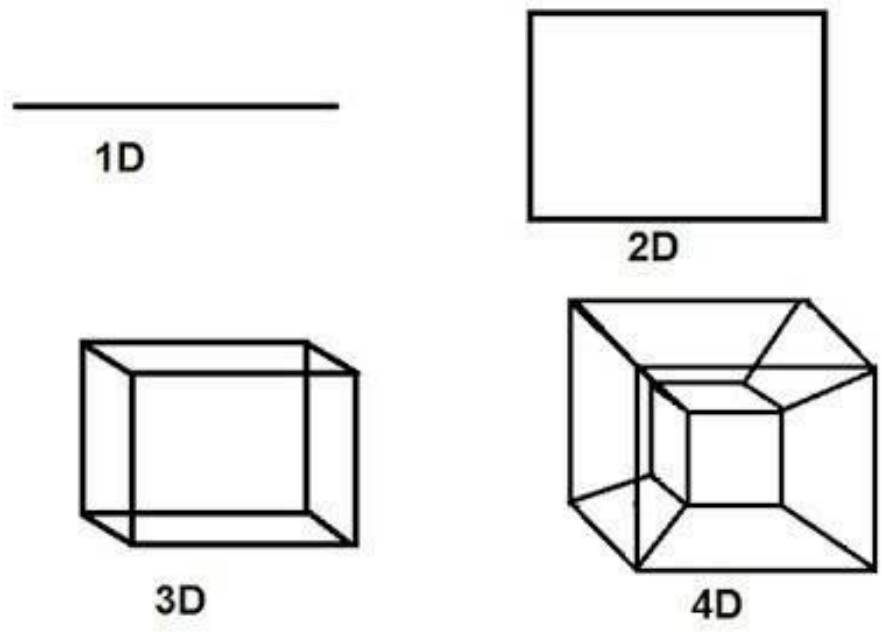


Figure 2: Geometrical objects in 1 to 4 dimensions. One more dimension can be thought of as the trace of movement over time.

But wait! Where does that 4D thing (a *tesseract*) come from? To really see a tesseract it helps to view it in an animation over time (**?@fig-tesseract**). But like the Square, contemplating 3D from a 2D world, it takes some imagination.

Yet the deep mathematics of more than three dimensions only emerged in the 19th century. In Newtonian mechanics, space and time were always considered independent of each other. Our familiar three-dimensional space, of length, width, and height had formed the backbone of Euclidean geometry for millenea. However, the idea that space and time are indeed interwoven was first proposed by German mathematician Hermann Minkowski (1864–1909) in 1908. This was a powerful idea. It bore fruit when Albert Einstein revolutionized the Newtonian conceptions of gravity in 1915 when he presented a theory of general relativity which was based primarily on the fact that mass and energy warp the fabric of four-dimensional spacetime.

The parable of *Flatland* can provide inspiration for statistical thinking and data visualization. Once we go beyond bivariate statistics and 2D plots, we are in a multivariate world of possibly MANY dimensions. It takes only some imagination and suitable methods to get there.

[1] In his famous TV series, *Cosmos*, Carl Sagan provides an intriguing video presentation Flatland and the 4th dimension. However, as far back as 1754 (Cajori, 1926), the idea of adding a fourth dimension appears in Jean le Rond d'Alembert's "Dimensions", and one realization of a four-dimensional object is a *tesseract*, shown in Figure 2.

Like Abbott's *Flatland*, this book is a romance, in many dimensions, of what we can learn from modern methods of data visualization.

EUREKA!

Even modest sized multivariate data can have secrets that can be revealed in the right view. As an example, David Coleman at RCA Laboratories in Princeton, N.J. generated a dataset of five (fictitious) measurements of grains of pollen for the 1986 Data Exposition at the Joint statistical Meetings. The first three variables are the lengths of geometric features 3848 observed sampled pollen grains – in the x, y, and z dimensions: a `ridge` along x, a `nub` in the y direction, and a `crack` in along the z dimension. The fourth variable is pollen grain `weight`, and the fifth is `density`. The challenge was to "find something interesting" in this dataset, now available as `animation::pollen`.

Those who solved the puzzle were able to find an orientation of this 5-dimensional dataset, such that zooming in revealed a magic word, "EUREKA" spelled in points, as in the following figure.

The following code transforms this data to long format and calculates some summary statistics for each `dataset`.

```
anscombe_long <- anscombe |>
  pivot_longer(everything(),
               names_to = c(".value", "dataset"),
               names_pattern = "(.)(.)"
  ) |>
  arrange(dataset)

anscombe_long |>
  group_by(dataset) |>
  summarise(xbar      = mean(x),
            ybar      = mean(y),
            r         = cor(x, y),
            intercept = coef(lm(y ~ x))[1],
            slope     = coef(lm(y ~ x))[2]
          )
#> # A tibble: 4 x 6
#>   dataset  xbar  ybar     r intercept slope
#>   <chr>   <dbl> <dbl> <dbl>     <dbl> <dbl>
#> 1 1           9  7.50 0.816      3.00 0.500
#> 2 2           9  7.50 0.816      3.00 0.5
#> 3 3           9  7.5  0.816      3.00 0.500
#> 4 4           9  7.50 0.817      3.00 0.500
```

As we can see, all four datasets have nearly identical univariate and bivariate statistical measures. You can only see how they differ in graphs, which show their true natures to be vastly different.

Figure 4 is an enhanced version of Anscombe's plot of these data, adding helpful annotations to show visually the underlying statistical summaries.

This figure is produced as follows, using a single call to `ggplot()`, faceted by `dataset`. As we will see later (Section 0.1.4), the data ellipse (produced by `stat_ellipse()`) reflects the correlation between the variables.

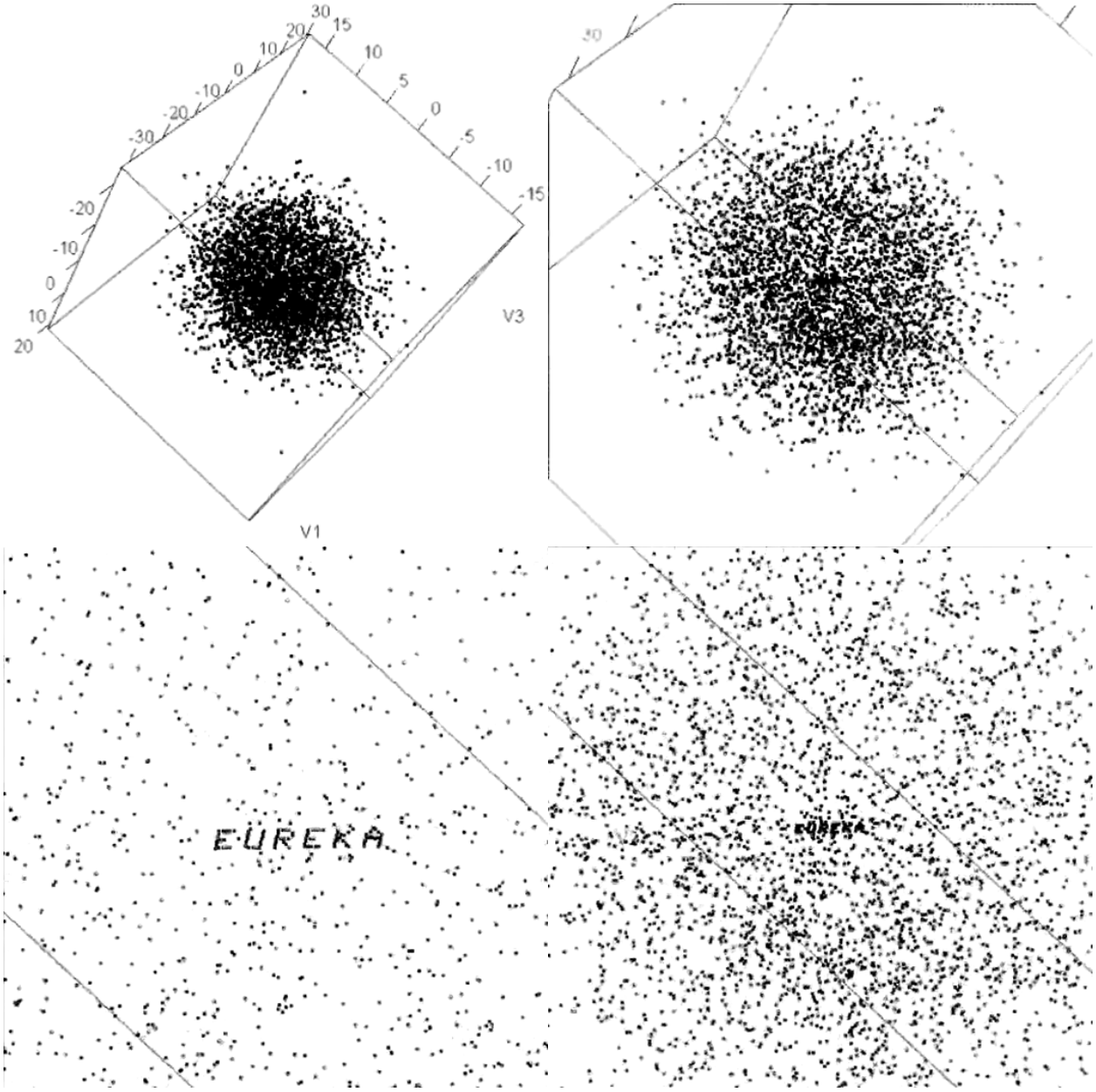


Figure 3: Four views of the `pollen` data, zooming in, clockwise from the upper left to discover the word "EUREKA".

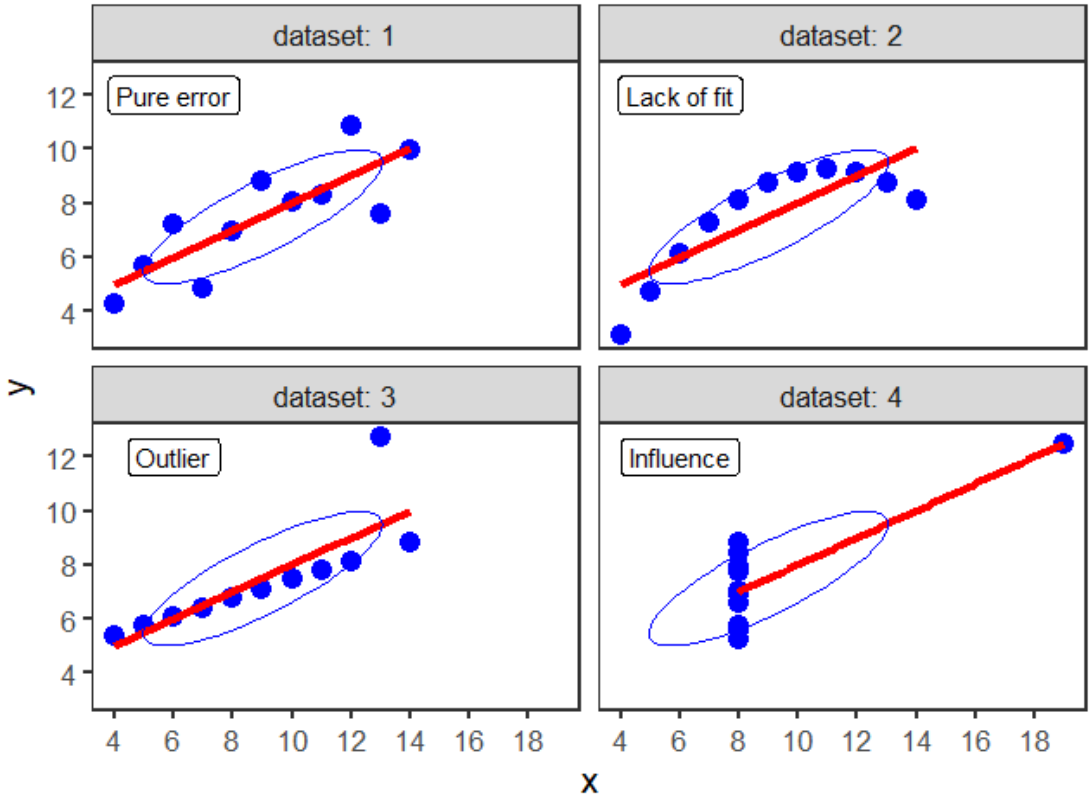


Figure 4: Scatterplots of Anscombe's Quartet. Each plot shows the fitted regression line and a 68% data ellipse representing the correlation between x and y.

```
desc <- tibble(
  dataset = 1:4,
  label = c("Pure error", "Lack of fit", "Outlier", "Influence")
)

ggplot(anscombe_long, aes(x = x, y = y)) +
  geom_point(color = "blue", size = 4) +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE,
              color = "red", linewidth = 1.5) +
  scale_x_continuous(breaks = seq(0,20,2)) +
  scale_y_continuous(breaks = seq(0,12,2)) +
  stat_ellipse(level = 0.5, color=col, type="norm") +
  geom_label(data=desc, aes(label = label), x=6, y=12) +
  facet_wrap(~dataset, labeller = label_both)
```

The subplots are labeled with the statistical idea they reflect:

- dataset 1: **Pure error**. This is the typical case with well-behaved data. Variation of the points around the line reflect only measurement error or unreliability in the response, y.
- dataset 2: **Lack of fit**. The data is clearly curvilinear, and would be very well described by a quadratic, `y ~ poly(x, 2)`. This violates the assumption of linear regression that the fitted model has the correct form.

- dataset 3: **Outlier**. One point, second from the right, has a very large residual. Because this point is near the extreme of x, it pulls the regression line towards it, as you can see by imagining a line through the remaining points.
- dataset 4: **Influence**. All but one of the points have the same x value. The one unusual point has sufficient influence to force the regression line to fit it **exactly**.

One moral from this example:

> **Linear regression only "sees" a line. It does its' best when the data are really linear. Because the line is fit by least squares, it pulls the line toward discrepant points to minimize the sum of squared residuals.**

i Datasaurus Dozen

The method Anscombe used to compose his quartet is unknown, but it turns out that that there is a method to construct a wider collection of datasets with identical statistical properties. After all, in a bivariate dataset with n observations, the correlation has $(n-2)$ degrees of freedom, so it is possible to choose $n-2$ of the (x, y) pairs to yield any given value. As it happens, it is also possible to create any number of datasets with the same means, standard deviations and correlations with nearly any shape you like — even a dinosaur!
The *Datasaurus Dozen* was first publicized by Alberto Cairo in a blog post and are available in the **datasauRus** package Davies et al. (2022). As shown in **?@fig-datasaurus**, the sets include a star, cross, circle, bullseye, horizontal and vertical lines, and, of course the "dino". The method (Matejka & Fitzmaurice, 2017) uses *simulated annealing*, an iterative process that perturbs the points in a scatterplot, moving them towards a given shape while keeping the statistical summaries close to the fixed target value.
The **datasauRus** package just contains the datasets, but a general method, called *statistical metamers*, for producing such datasets has been described by Elio Campitelli and implemented in the **metamer** package.

i Quartets

The essential idea of a statistical "quartet" is to illustrate four quite different datasets or circumstances that seem superficially the same, but yet are paradoxically very different when you look behind the scenes. For example, in the context of causal analysis Gelman et al. (2023), illustrated sets of four graphs, within each of which all four represent the same average (latent) causal effect but with much different patterns of individual effects; McGowan et al. (2023) provide another illustration with four seemingly identical data sets each generated by a different causal mechanism. As an example of machine learning models, Biecek et al. (2023), introduced the "Rashamon Quartet", a synthetic dataset for which four models from different classes (linear model, regression tree, random forest, neural network) have practically identical predictive performance. In all cases, the paradox is solved when their visualization reveals the distinct ways of understanding structure in the data. The **quartets** package contains these and other variations on this theme.

A real example

In the mid 1980s, a consulting client had a strange problem. She was conducting a study of the relation between body image and weight preoccupation in exercising and non-exercising people (Davis, 1990). As part of the design, the researcher wanted to know if self-reported weight could be taken as a reliable indicator of true weight measured on a scale. It was expected that the correlations between reported and measured weight should be close to 1.0, and the slope of the regression lines for men and women should also be close to 1.0. The dataset is `car::Davis`.

She was therefore very surprise to see the following numerical results: For men, the correlation was nearly perfect, but not so for women.

```
data(Davis, package="carData")
Davis <- Davis |>
  drop_na()          # drop missing cases
Davis |>
  group_by(sex) |>
  select(sex, weight, repwt) |>
  summarise(r = cor(weight, repwt))
#> # A tibble: 2 x 2
#>   sex       r
#>   <fct> <dbl>
#> 1 F     0.501
#> 2 M     0.979
```

Similarly, the regression lines showed the expected slope for men, but that for women was only 0.26.

```
Davis |>
  nest(data = -sex) |>
  mutate(model = map(data, ~ lm(repwt ~ weight, data = .)),
         tidied = map(model, tidy)) |>
  unnest(tidied) |>
  filter(term == "weight") |>
  select(sex, term, estimate, std.error)
#> # A tibble: 2 x 4
#>   sex   term   estimate std.error
#>   <fct> <chr>     <dbl>     <dbl>
#> 1 M     weight    0.990    0.0229
#> 2 F     weight    0.262    0.0459
```

What could be wrong here?, the client asked. The consultant replied with the obvious question:

> *Did you plot your data?*

The answer turned out to be one discrepant point, a female, whose measured weight was 166 kg (366 lbs!). This single point exerted so much influence that it pulled the fitted regression line down to a slope of only 0.26.

```
Davis |>
  ggplot(aes(x = weight, y = repwt,
             color = sex, shape=sex)) +
  geom_point(size = ifelse(Davis$weight==166, 6, 2)) +
  labs(y = "Measured weight (kg)",
       x = "Reported weight (kg)") +
  geom_smooth(method = "lm", formula = y~x, se = FALSE) +
  theme(legend.position = c(.8, .8))
```

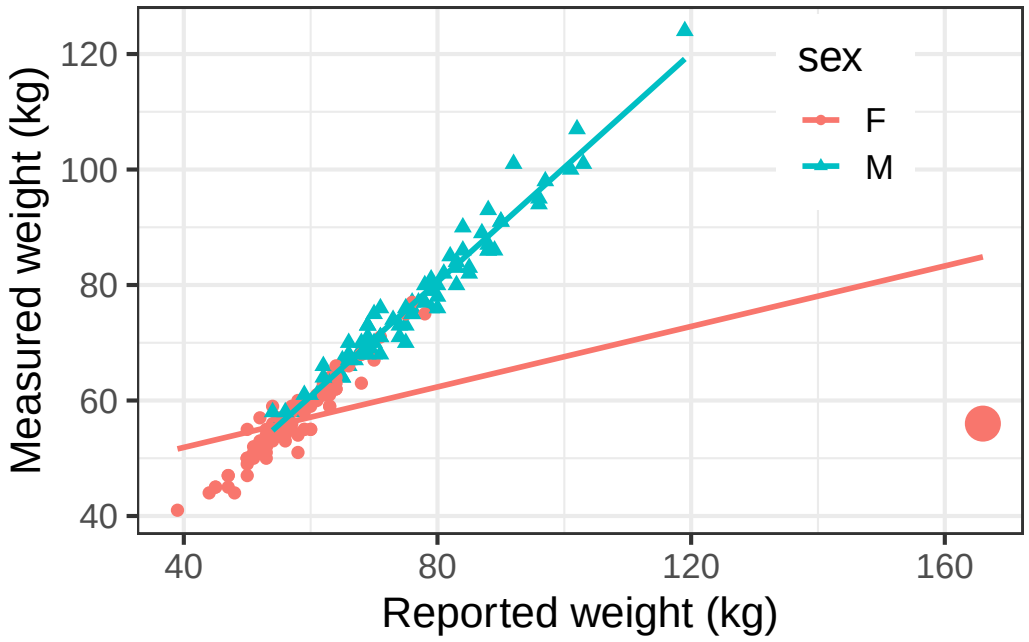


Figure 5: Regression for Davis' data on reported weight and measures weight for men and women. Separate regression lines, predicting reported weight from measured weight are shown for males and females. One highly unusual point is highlighted.

In this example, it was arguable that x and y axes should be reversed, to determine how well measured weight can be predicted from reported weight. In `ggplot` this can easily be done by reversing the `x` and `y` aesthetics.

```
Davis |>
  ggplot(aes(y = weight, x = repwt, color = sex, shape=sex)) +
  geom_point(size = ifelse(Davis$weight==166, 6, 2)) +
  labs(y = "Measured weight (kg)", x = "Reported weight (kg)") +
  geom_smooth(method = "lm", formula = y~x, se = FALSE) +
  theme(legend.position = c(.8, .8))
```

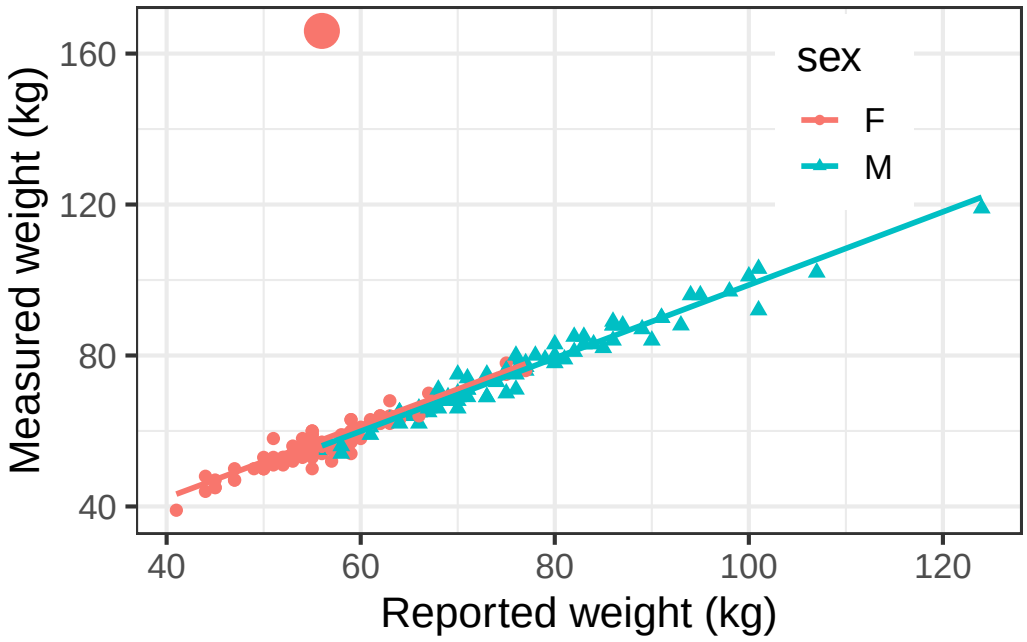


Figure 6: Regression for Davis' data on reported weight and measures weight for men and women. Separate regression lines, predicting measured weight from re[ported] weight are shown for males and females. The highly unusual point no longer has an effect on the fitted lines.

In Figure 6, this discrepant observation again stands out like a sore thumb, but it makes very little difference in the fitted line for females. The reason is that this point is well within the range of the x variable (`repwt`). To impact the slope of the regression line, an observation must be unusual in_both_ x and y. We take up the topic of how to detect influential observations and what to do about them in Chapter XX.

The value of such plots is not only that they can reveal possible problems with an analysis, but also help identify their reasons and suggest corrective action. What went wrong here? Examination of the original data showed that this person switched the values, recording her reported weight in the box for measured weight and vice versa.

Plots for data analysis

Visualization methods take an enormous variety of forms, but it is useful to distinguish several broad categories according to their use in data analysis:

- **data plots** : primarily plot the raw data, often with annotations to aid interpretation (regression lines and smooths, data ellipses, marginal distributions)
- **reconnaissance plots** : with more than a few variables, reconnaissance plots provide a high-level, bird's-eye overview of the data, allowing you to see patterns that might not be visible in a set of separate plots. Some examples are scatterplot matrices, showing all bivariate plots of variables in a dataset; correlation diagrams, using visual glyphs to represent the correlations between all pairs of variables and "trellis" or faceted plots that show how a focal relation of one or more variables differs across values of other variables.
- **model plots** : plot the results of a fitted model, such as a regression line or curve to show uncertainty, or a regression surface in 3D, or a plot of coefficients in model together with confidence intervals. Other

model plots try to take into account that a fitted model may involve more variables than can be shown in a static 2D plot. Some examples of these are added variable plots, and marginal effect plots, both of which attempt to show the net relation of two focal variables, controlling or adjusting for other variables in a model.

- **diagnostic plots** : indicating potential problems with the fitted model. These include residual plots, influence plots, plots for testing homogeneity of variance and so forth.
- **dimension reduction plots** : plot representations of the data into a space of fewer dimensions than the number of variables in the dataset. Simple examples include principal components analysis (PCA) and the related biplots, and multidimensional scaling (MDS) methods.

We give some more details and a few examples in the sections that follow.

Data plots

Data plots portray the data in a space where the coordinate axes are the observed variables.

- 1D plots include line plots, histograms and density estimates.
- 2D plots are most often scatterplots, but contour plots or hex-binned plots are also useful when the sample size is large.

Model plots

Model plots show the fitted or predicted values from a statistical model and provide visual summaries...

Diagnostic plots

Principles of graphical display

[This could be a separate chapter]

- Criteria for assessing graphs: communication goals
- Effective data display:
  - Make the data stand out
  - Make graphical comparison easy
  - Effect ordering: For variables and unordered factors, arrange them according to the effects to be seen
- Visual thinning: As the data becomes more complex, focus more on impactful summaries

Packages used here: 19 packages used here: base, broom, datasets, dplyr, forcats, ggplot2, graphics, grDevices, knitr, lubridate, methods, purrr, readr, stats, stringr, tibble, tidyr, tidyverse, utils

0

Plots of Multivariate Data

> There is no excuse for failing to plot and look.
>
> The greatest value of a picture is when it forces us to notice what we never expected to see. — John W. Tukey, *Exploratory Data Analysis*, 1977

These quotes from John Tukey remind us that data analysis should nearly always start with graphs to help us understand the main features of our data. It is important to understand the general *patterns* and *trends*: Are relationships increasing or decreasing? Are they approximately linear or non-linear? But it is also important to spot *anomalies*: "unusual" observations, groups of points that seem to differ from the rest, and so forth. As we saw with Anscombe's quartet (**?@sec-anscombe**) numerical summaries hide features that are immediately apparent in a plot.

This chapter introduces a toolbox of basic graphical methods for visualizing multivariate datasets. It starts with some simple techniques to enhance the basic scatterplot with graphical *annotations* such as fitted lines, curves and data ellipses to *summarize* the relation between two variables.

To visualize more than two variables, we can view all pairs of variables in a scatterplot matrix or shift gears entirely to show multiple variables along a set of parallel axes. As the number of variables increases, we may need to suppress details with stronger summaries for a high-level reconnaissance of our data terrain, as we do by zooming out on a map. For example, we can simply remove the data points or make them nearly transparent to focus on the visual summaries provided by fitted lines or other graphical summaries.

Packages

In this chapter I use the following packages. Load them now:

```
library(car)
library(ggplot2)
library(dplyr)
library(tidyr)
library(corrplot)
library(corrgram)
library(GGally)
library(ggdensity)
library(patchwork)
library(ggpcp)
```

0.1 Bivariate summaries

The basic scatterplot is the workhorse of multivariate data visualization, showing how one variable, y, often an outcome to be explained by or varies with another, x. It is a building block for many useful techniques, so it is helpful to understand how it can be used as a tool for thinking in a wider, multivariate context.

The essential idea is that we can start with a simple version of the scatterplot and add annotations to show interesting features more clearly. We consider the following here:

- **Smoothers**: Showing overall trends, perhaps in several forms, as visual summaries such as fitted regression lines or curves and nonparametric smoothers.
- **Stratifiers**: Using color, shape or other features to identify subgroups; more generally, *conditioning* on other variables in multi-panel displays;
- **Data ellipses**: A compact 2D visual summary of bivariate linear relations and uncertainty assuming normality; more generally, contour plots of bivariate density.

Example: Academic salaries

Let's start with data on the academic salaries of faculty members collected at a U.S. college for the purpose of assessing salary differences between male and female faculty members, and perhaps address anomalies in compensation. The dataset `carData::Salaries` gives data on nine-month salaries and other variables for 397 faculty members in the 2008-2009 academic year.

```
data(Salaries, package = "carData")
str(Salaries)
#> 'data.frame':    397 obs. of  6 variables:
#>  $ rank         : Factor w/ 3 levels "AsstProf","AssocProf",..: 3 3 1 3 3 2 3 3 3 3 ...
#>  $ discipline   : Factor w/ 2 levels "A","B": 2 2 2 2 2 2 2 2 2 2 ...
#>  $ yrs.since.phd: int  19 20 4 45 40 6 30 45 21 18 ...
#>  $ yrs.service  : int  18 16 3 39 41 6 23 45 20 18 ...
#>  $ sex          : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 1 ...
#>  $ salary       : int  139750 173200 79750 115000 141500 97000 175000 147765 119250 129000 ...
```

The most obvious, but perhaps naive, predictor of `salary` is `years.since.phd`. For simplicity, I'll refer to this as years of "experience." Before looking at differences between males and females, we would want consider faculty `rank` (related also to `yrs.service`) and `discipline`, recorded here as `A` ("theoretical" departments) or `B` ("applied" departments). But, for a basic plot, we will ignore these for now to focus on what can be learned from plot annotations.

```
library(ggplot2)
gg1 <- ggplot(Salaries,
       aes(x = yrs.since.phd, y = salary)) +
  geom_jitter(size = 2) +
  scale_y_continuous(labels = scales::dollar_format(
    prefix="$", scale = 0.001, suffix = "K")) +
  labs(x = "Years since PhD",
       y = "Salary")

gg1
```

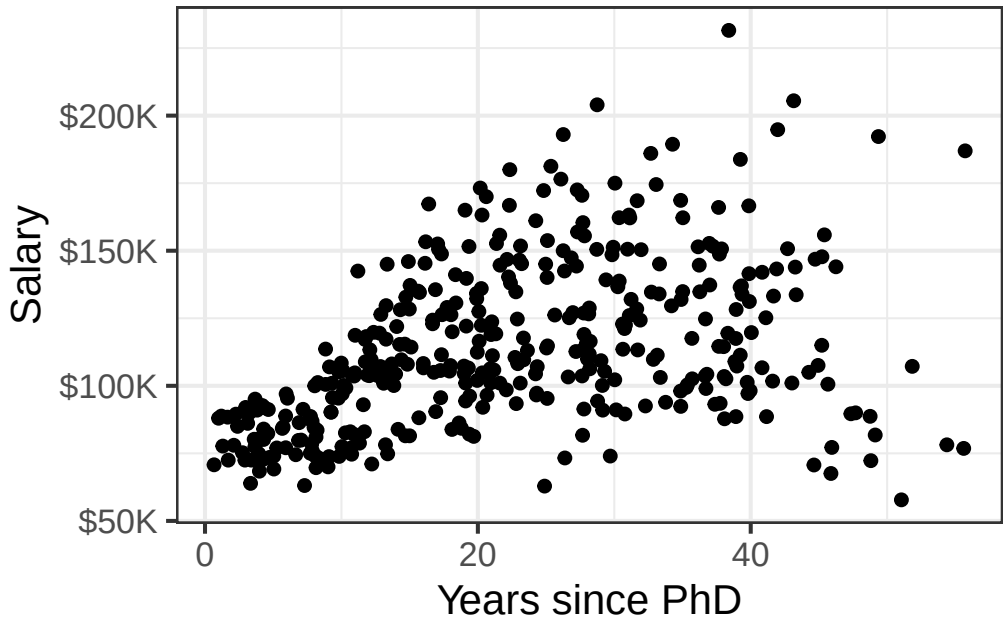


Figure 7: Naive scatterplot of Salary vs. years since PhD, ignoring other variables, and without graphical annotations.

There is quite a lot we can see "just by looking" at this simple plot, but the main things are:

- Salary increases generally from 0 - 40 years since the PhD;
- Variability in salary increases among those with the same experience, a "fan-shaped" pattern that signals a violation of homogeneity of variance in simple regression;
- Data beyond 50 years is thin, but there are some quite low salaries there.

0.1.1 Smoothers

Smoothers are among the most useful graphical annotations you can add to such plots, giving a visual summary of how y changes with x. The most common smoother is a line showing the linear regression for y given x, expressed in math notation as $\mathbb{E}(y|x) = b_0 + b_1 x$. If there is doubt that a linear relation is an adequate summary, you can try a quadratic or other polynomial smoothers.

In **ggplot2**, these are easily added to a plot using `geom_smooth()` with `method = "lm"`, and a model `formula`, which (by default) is `y ~ x` for a linear relation or `y ~ poly(x, k)` for a polynomial of degree k.

```
gg1 +
  geom_smooth(method = "lm", formula = "y ~ x",
              color = "red", fill= "red",
              linewidth = 2) +
  geom_smooth(method = "lm", formula = "y ~ poly(x,2)",
              color = "darkgreen", fill = "darkgreen",
              linewidth = 2)
```

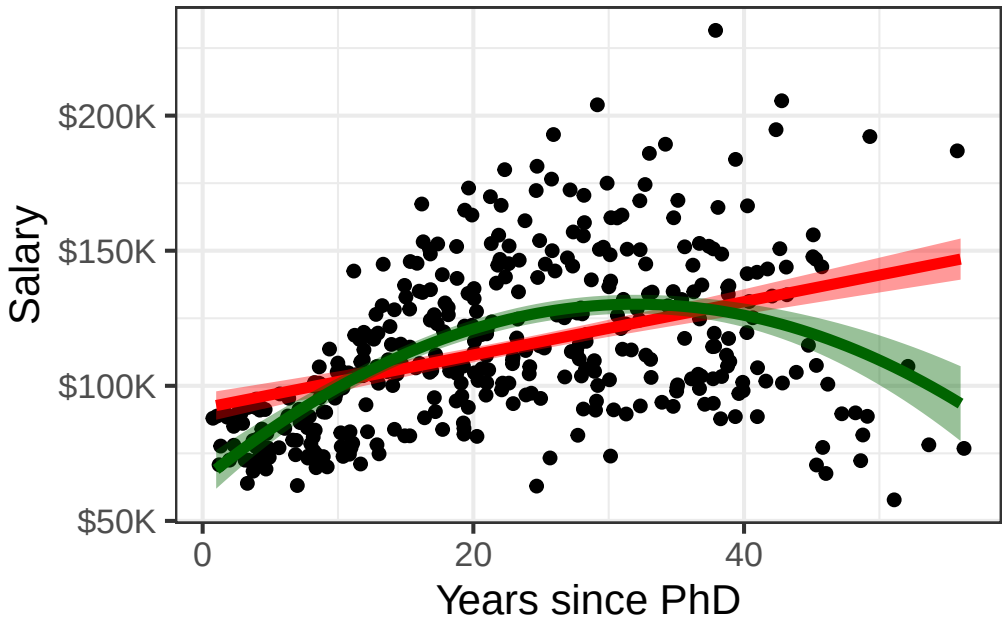


Figure 8: Scatterplot of Salary vs. years since PhD, showing linear and quadratic smooths with 95% confidence bands.

This serves to highlight some of our impressions from the basic scatterplot shown in Figure 7, making them more apparent. And that's precisely the point: the regression smoother draws attention to a possible pattern that we can consider as a visual summary of the data. You can think of this as showing what a linear (or quadratic) regression "sees" in the data. Statistical tests can help you decide if there is more evidence for a quadratic fit compared to the simpler linear relation.

It is useful to also show some indication of *uncertainty* (or inversely, *precision*) associated with the predicted values. Both the linear and quadratic trends are shown in Figure 8 with 95% pointwise confidence bands. These are necessarily narrower in the center of the range of x where there is typically more data; they get wider toward the highest values of experience where the data are thinner.

0.1.1.1 Non-parametric smoothers

The most generally useful idea is a smoother that tracks an average value, $\mathbb{E}(y|x)$, of y as x varies across its' range *without* assuming any particular functional form, and so avoiding the necessity to choose among `y ~ poly(x, 1)`, or `y ~ poly(x, 2)`, or `y ~ poly(x, 3)`, etc.

Non-parametric smoothers attempt to estimate $\mathbb{E}(y|x) = f(x)$ where $f(x)$ is some smooth function. These typically use a collection of weighted *local regressions* for each x_i within a window centered at that value. In the method called *lowess* or *loess* (Cleveland, 1979; Cleveland & Devlin, 1988), a weight function is applied, giving greatest weight to x_i and a weight of 0 outside a window containing a certain fraction, s, called *span*, of the nearest neighbors of x_i. The fraction, s, is usually within the range $1/3 \leq s \leq 2/3$, and it determines the smoothness of the resulting curve; smaller values produce a wigglier curve and larger values giving a smoother fit (an optimal span can be determined by k-fold cross-validation to minimize a measure of overall error of approximation).

Non-parametric regression is a broad topic; see Fox (2016), Ch. 18 for a more general treatment and Wood (2006) for generalized additive models, fit using `method = "gam"` in **ggplot2**, which is the default when the largest group has more than 1,000 observations.

Figure 9 shows the addition of a loess smooth to the plot in Figure 8, suppressing the confidence band for the

linear regression. The loess fit is nearly coincident with the quadratic fit, but has a slightly wider confidence band.

```
gg1 +
  geom_smooth(method = "loess", formula = "y ~ x",
              color = "blue", fill = scales::muted("blue"),
              linewidth = 2) +
  geom_smooth(method = "lm", formula = "y ~ x", se = FALSE,
              color = "red",
              linewidth = 2) +
  geom_smooth(method = "lm", formula = "y ~ poly(x,2)",
              color = "darkgreen", fill = "lightgreen",
              linewidth = 2)
```

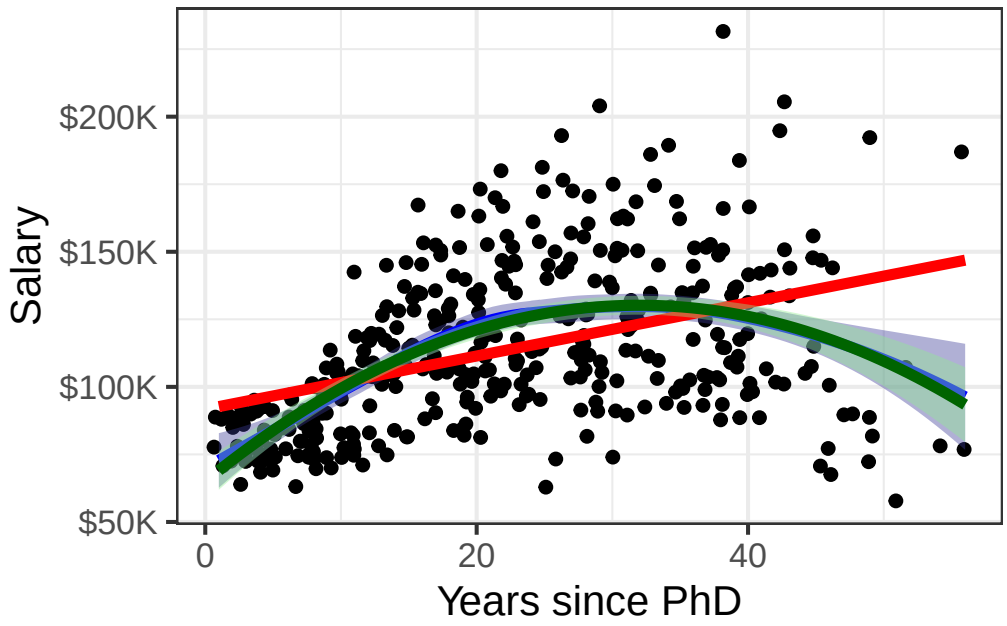


Figure 9: Scatterplot of Salary vs. years since PhD, adding the loess smooth.

But now comes an important question: is it reasonable that academic salary should increase up to about 40 years since the PhD degree and then decline? The predicted salary for someone still working 50 years after earning their degree is about the same as a person at 15 years. What else is going on here?

0.1.2 Stratifiers

Very often, we have a main relationship of interest, but various groups in the data are identified by discrete factors (like faculty `rank` and `sex`, their type of `discipline` here), or there are quantitative predictors for which the main relation might vary. In the language of statistical models such effects are *interaction* terms, as in `y ~ group + x + group:x`, where the term `group:x` fits a different slope for each group and the grouping variable is often called a *moderator* variable. Common moderator variables are ethnicity, health status, social class and level of education. Moderators can also be continuous variables as in `y ~ x1 + x2 + x1:x2`.

I call these *stratifiers*, recognizing that we should consider breaking down the overall relation to see whether and how it changes over such "other" variables. Such variables are most often factors, but we can cut a continuous variable into ranges (*shingles*) and do the same graphically. There are two general stratifying graphical techniques:

- **Grouping**: Identify subgroups in the data by assigning different visual attributes, such as color, shape, line style, etc. within a single plot. This is quite natural for factors; quantitative predictors can be accommodated by cutting their range into ordered intervals. Grouping has the advantage that the levels of a grouping variable can be shown within the same plot, facilitating direct comparison.
- **Conditioning**: Showing subgroups in different plot panels. This has the advantages that relations for the individual groups more easily discerned and one can easily stratify by two (or more) other variables jointly, but visual comparison is more difficult because the eye must scan from one panel to another.

> **i** History Corner
>
> Recognition of the roles of visual grouping by factors within a panel and conditioning in multi-panel displays was an important advance in the development of modern statistical graphics. It began at A.T.&T. Bell Labs in Murray Hill, NJ in conjunction with the **S** language, the mother of R.
> Conditioning displays (originally called *coplots* (Chambers & Hastie, 1991)) are simply a collection of 1D, 2D or 3D plots separate panels for subsets of the data broken down by one or more factors, or, for quantitative variables, subdivided into a factor with several overlapping intervals (*shingles*). The first implementation was in *Trellis* plots Becker et al. (1996);Cleveland (1985).
> Trellis displays were extended in the **lattice** package (Sarkar, 2023), which offered:
> - A **graphing syntax** similar to that used in statistical model formulas: `y ~ x | g` conditions the data by the levels of `g`, with `|` read as "given"; two or more conditioning are specified as `y ~ x | g1 + g2 + ...`, with `+` read as "and".
> - **Panel functions** define what is plotted in a given panel. `panel.xyplot()` is the default for scatterplots, plotting points, but you can add `panel.lmline()` for regression lines, `latticeExtra::panel.smoother()` for loess smooths and a wide variety of others.
>
> The **car** package (Fox et al., 2023) supports this graphing syntax in many of its functions. **ggplot2** does not; it uses aesthetics (`aes()`), which map variables in the data to visual characteristics in displays.

The most obvious variable that affects academic salary is `rank`, because faculty typically get an increase in salary with a promotion that carries through in their future salary. What can we see if we group by `rank` and fit a separate smoothed curve for each?

In `ggplot2` thinking, grouping is accomplished simply by adding an aesthetic, such as `color = rank`. What happens then is that points, lines, smooths and other `geom_*()` inherit the feature that they are differentiated by color. In the case of `geom_smooth()`, we get a separate fit for each subset of the data, according to `rank`.

```
# make some re-useable pieces to avoid repetitions
scale_salary <-   scale_y_continuous(
  labels = scales::dollar_format(prefix="$",
                                 scale = 0.001,
                                 suffix = "K"))
# position the legend inside the plot
legend_pos <- theme(legend.position = c(.1, 0.95),
                    legend.justification = c(0, 1))

ggplot(Salaries,
       aes(x = yrs.since.phd, y = salary,
           color = rank)) +
  geom_point() +
  scale_salary +
  labs(x = "Years since PhD",
       y = "Salary") +
  geom_smooth(aes(fill = rank),
```

```
                  method = "loess", formula = "y ~ x",
                  linewidth = 2)  +
  legend_pos
```

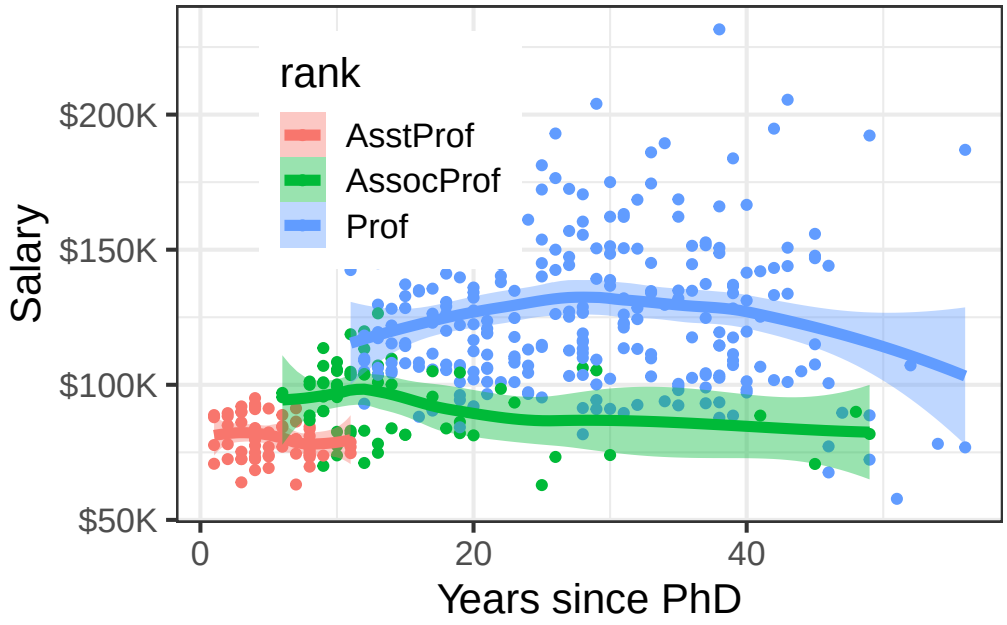


Figure 10: Scatterplot of Salary vs. years since PhD, grouped by rank.

Well, there is a different story here. Salaries generally occupy separate levels, increasing with academic rank. The horizontal extents of the smoothed curves show their ranges. Within each rank there is some initial increase after promotion, and then some tendency to decline with increasing years. But, by and large, years since the PhD doesn't make that much difference once we've taken academic rank into account.

What about the `discipline` which is classified, perhaps peculiarly, as "theoretical" vs. "applied"? The values are just `"A"` and `"B"`, so I map these to more meaningful labels before making the plot.

```
Salaries <- Salaries |>
  mutate(discipline = factor(discipline,
                             labels = c("A: Theoretical", "B: Applied")))

Salaries |>
  ggplot(aes(x = yrs.since.phd, y = salary, color = discipline)) +
    geom_point() +
  scale_salary +
  geom_smooth(aes(fill = discipline ),
              method = "loess", formula = "y ~ x",
              linewidth = 2) +
  labs(x = "Years since PhD",
       y = "Salary") +
  legend_pos
```

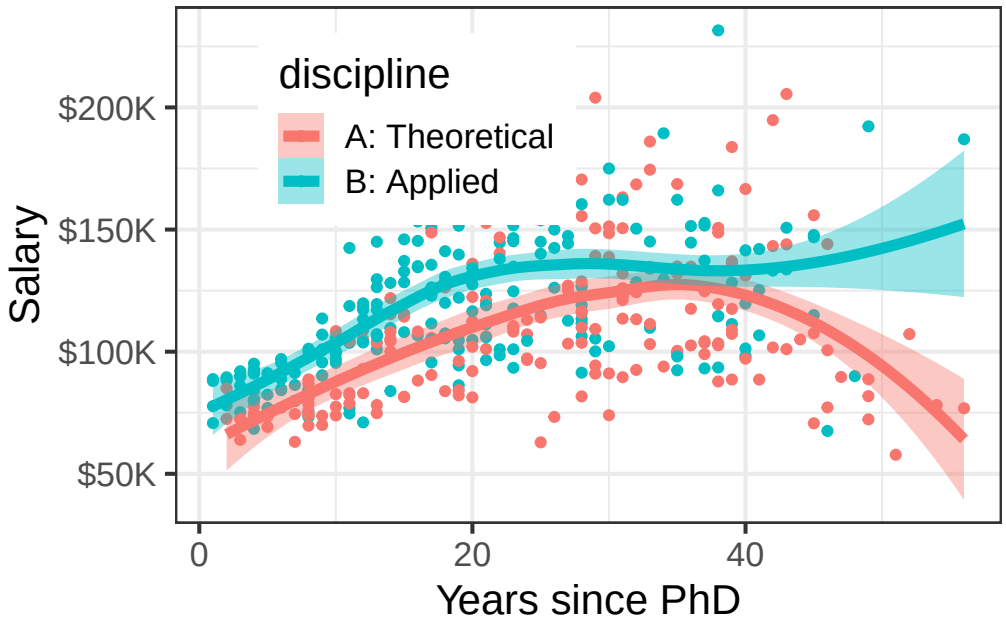


Figure 11: Scatterplot of Salary vs. years since PhD, grouped by discipline.

The story in Figure 11 is again different. Faculty in applied disciplines on average earn about 10,000$ more per year on average than their theoretical colleagues.

```
Salaries |>
  group_by(discipline) |>
  summarize(mean = mean(salary))
#> # A tibble: 2 x 2
#>   discipline        mean
#>   <fct>            <dbl>
#> 1 A: Theoretical 108548.
#> 2 B: Applied     118029.
```

For both groups, there is an approximately linear relation up to about 30–40 years, but the smoothed curves then diverge into the region where the data is thinner.

0.1.3 Conditioning

The previous plots use grouping by color to plot the data for different subsets inside the same plot window, making comparison among groups easier, because they can be directly compared along a common vertical scale [2]. This gets messy, however, when there are more than just a few levels, or worse—when there are two (or more) variables for which we want to show separate effects. In such cases, we can plot separate panels using the `ggplot2` concept of *faceting*. There are two options: `facet_wrap()` takes one or more conditioning variables and produces a ribbon of plots for each combination of levels; `facet_grid(row ~ col)` takes two or more conditioning variables and arranges the plots in a 2D array identified by the `row` and `col` variables.

Let's look at salary broken down by the combinations of discipline and rank. Here, I chose to stratify using color by rank within each of panels faceting by discipline. Because there is more going on in this plot, a linear smooth is used to represent the trend.

[2]The classic study by Cleveland & McGill (1984);Cleveland & McGill (1985) shows that judgements of magnitude along a common scale are more accurate than those along separate, aligned scales.

```
Salaries |>
  ggplot(aes(x = yrs.since.phd, y = salary, color = rank)) +
  geom_point() +
  scale_salary +
  labs(x = "Years since PhD",
       y = "Salary") +
  geom_smooth(aes(fill = rank),
              method = "lm", formula = "y ~ x",
              linewidth = 2) +
  facet_wrap(~ discipline) +
  legend_pos
```

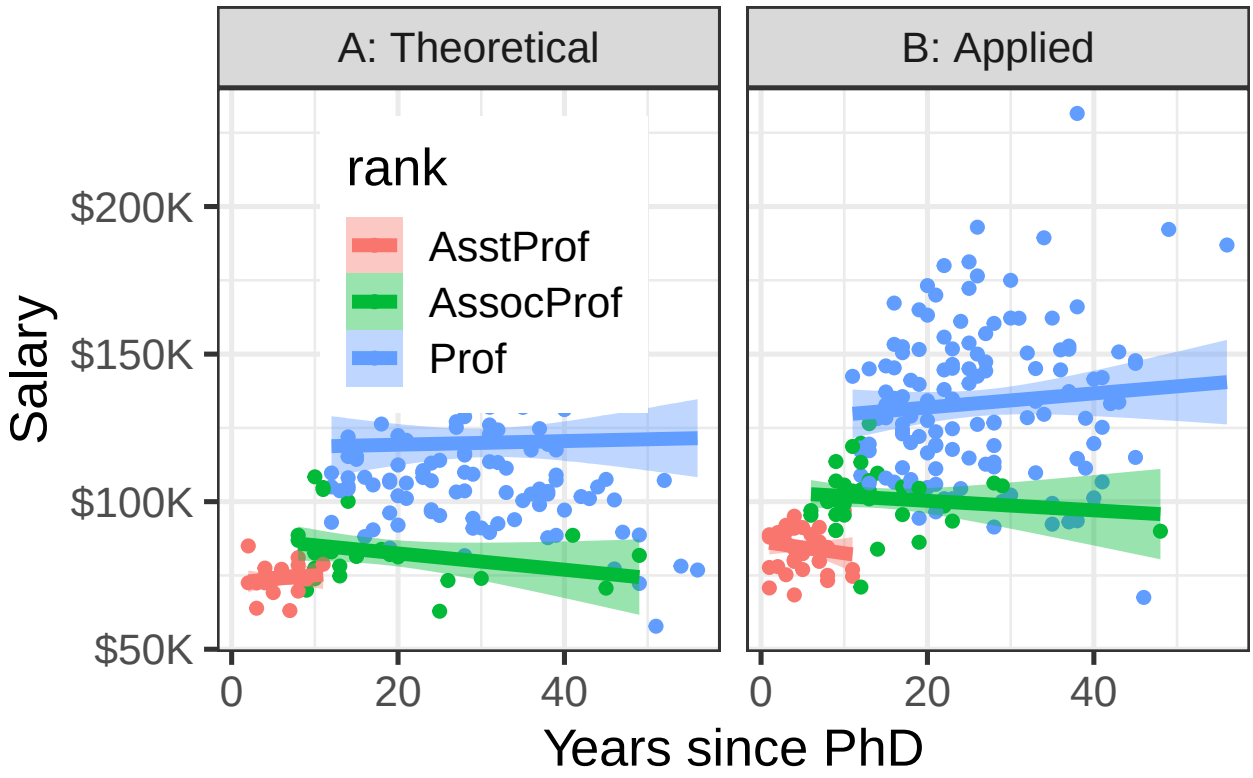


Figure 12: Scatterplot of Salary vs. years since PhD, grouped by rank, with separate panels for discipline.

Once both of these factors are taken into account, there does not seem to be much impact of years of service. Salaries in theoretical disciplines are noticeably greater than those in applied disciplines at all ranks, and there are even greater differences among ranks.

Finally, to shed light on the question that motivated this example— are there anomalous differences in salary for men and women— we can look at differences in salary according to sex, when discipline and rank are taken into account. To do this graphically, condition by both variables, but use `facet_grid(discipline ~ rank)` to arrange their combinations in a grid whose rows are the levels of `discipline` and columns are those of `rank`. I want to make the comparison of males and females most direct, so I use `color = sex` to stratify the panels. The smoothed regression lines and error bands are calculated separately for each combination of discipline, rank and sex.

```
Salaries |>
  ggplot(aes(x = yrs.since.phd, y = salary, color = sex)) +
  geom_point() +
  scale_salary +
  labs(x = "Years since PhD",
       y = "Salary") +
  geom_smooth(aes(fill = sex),
              method = "lm", formula = "y ~ x",
              linewidth = 2) +
  facet_grid(discipline ~ rank) +
  theme_bw(base_size = 14) +
  legend_pos
```

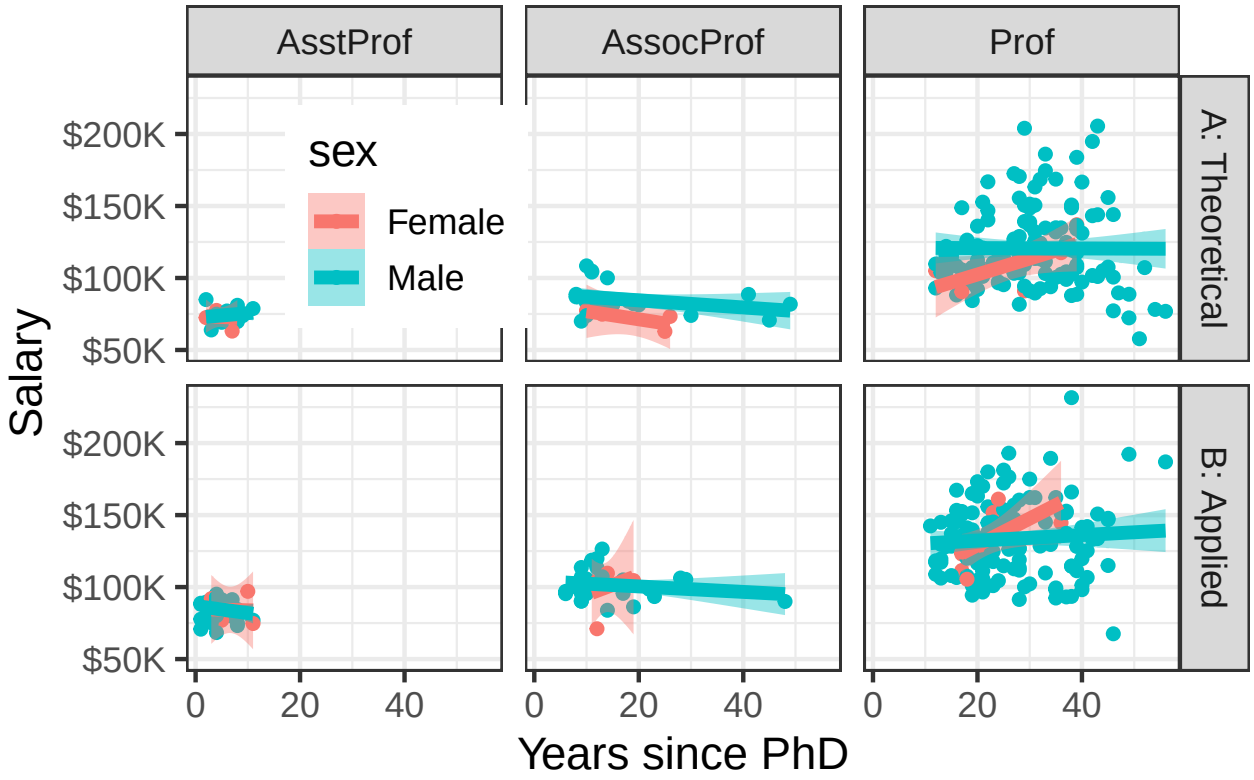


Figure 13: Scatterplot of Salary vs. years since PhD, grouped by sex, faceted by discipline and rank.

0.1.4 Data Ellipses

The *data ellipse* (Monette, 1990), or *concentration ellipse* (Dempster, 1969) is a remarkably simple and effective display for viewing and understanding bivariate relationships in multivariate data. The data ellipse is typically used to add a visual summary to a scatterplot, that shows all together the means, standard deviations, correlation, and slope of the regression line for two variables, perhaps stratified by another variable. Under the classical assumption that the data are bivariate normally distributed, the data ellipse is also a **sufficient** visual summary, in the sense that it captures **all** relevant features of the data. See Friendly et al. (2013) for a complete discussion of the role of ellipsoids in statistical data visualization.

It is based on the idea that in a bivariate normal distribution, the contours of equal probability form a series of concentric ellipses. If the variables were uncorrelated and had the same variances, these would be circles,

and Euclidean distance would measure the distance of each observation from the mean. When the variables are correlated, a different measure, *Mahalanobis distance* is the proper measure of how far a point is from the mean, taking the correlation into account.

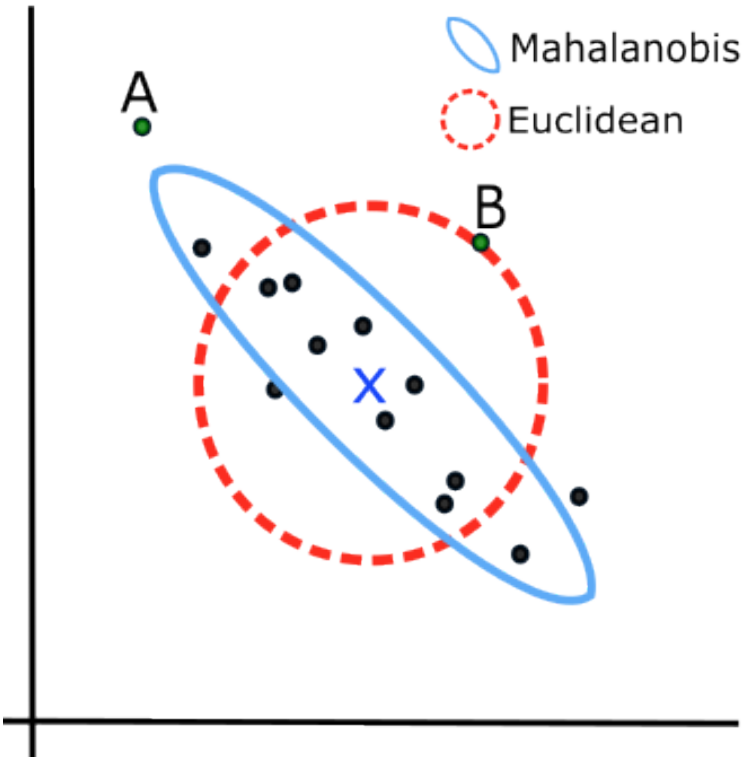


Figure 14: 2D data with curves of constant distance from the centroid. The blue solid ellipse shows a contour of constant Mahalanobis distance, taking the correlation into account; the dashed red circle is a contour of equal Euclidean distance. Given the data points, Which of the points **A** and **B** is further from the mean (X)? *Source*: Re-drawn from Ou Zhang

To illustrate, Figure 14 shows a scatterplot with labels for two points, "A" and "B". Which is further from the mean, "X"? A contour of constant Euclidean distance, shown by the red dashed circle, ignores the apparent negative correlation, so point "A" is further. The blue ellipse for Mahalanobis distance takes the correlation into account, so point "B" has a greater distance from the mean.

Mathematically, Euclidean (squared) distance for p variables, $j = 1, 2, \ldots, p$, is just a generalization of the square of a univariate standardized (z) score, $z^2 = [(y - \bar{y})/s]^2$,

$$D^2_E(\mathbf{y}) = \sum_j^p z_j^2 = \mathbf{z}^T\mathbf{z} = (\mathbf{y} - \bar{\mathbf{y}})^T \operatorname{diag}(\mathbf{S})^{-1}(\mathbf{y} - \bar{\mathbf{y}}) \; ,$$

where $\mathbf{S}$ is the sample variance-covariance matrix, $\mathbf{S} = (n-1)^{-1}\sum_{i=1}^n (\mathbf{y}_i - \bar{\mathbf{y}})^T(\mathbf{y}_i - \bar{\mathbf{y}})$.

Mahalanobis' distance takes the correlations into account simply by using the covariances as well as the variances,

$$D^2_M(\mathbf{y}) = (\mathbf{y} - \bar{\mathbf{y}})^T S^{-1}(\mathbf{y} - \bar{\mathbf{y}}) \; .$$

For p variables, the data *ellipsoid* $\mathcal{E}_c$ of size c is a p-dimensional ellipse, defined as the set of points $\mathbf{y} = (y_1, y_2, \ldots y_p)$ whose squared Mahalanobis distance, $D^2_M(\mathbf{y})$ is less than or equal to c^2.

When $\mathbf{y}$ is (at least approximately) bivariate normal, $D^2_M(\mathbf{y})$ has a large-sample χ^2_2 distribution (χ^2 with 2 df), so taking $c^2 = \chi^2_2(0.68) = 2.28$ gives a "1 standard deviation bivariate ellipse," an analog of the standard interval $\bar{y} \pm 1s$, while $c^2 = \chi^2_2(0.95) = 5.99 \approx 6$ gives a data ellipse of 95% coverage.

0.1.4.1 Properties

The essential ideas of correlation and regression and their relation to ellipses go back to Galton (1886). Galton's goal was to predict (or explain) how a heritable trait, Y, (e.g., height) of children was related to that of their parents, X. He made a semi-graphic table of the frequencies of 928 observations of the average height of father and mother versus the height of their child, shown in Figure 15. He then drew smoothed contour lines of equal frequencies and had the wonderful visual insight that these formed concentric shapes that were tolerably close to ellipses. He then calculated summaries, Ave$(Y|X)$, and, for symmetry, Ave$(X|Y)$, and plotted these as lines of means on his diagram. Lo and behold, he had a second visual insight: the lines of means of $(Y|X)$ and $(X|Y)$ corresponded approximately to the loci of horizontal and vertical tangents to the concentric ellipses. To complete the picture, he added lines showing the major and minor axes of the family of ellipses (which turned out to be the principal components) with the result shown in Figure 15.

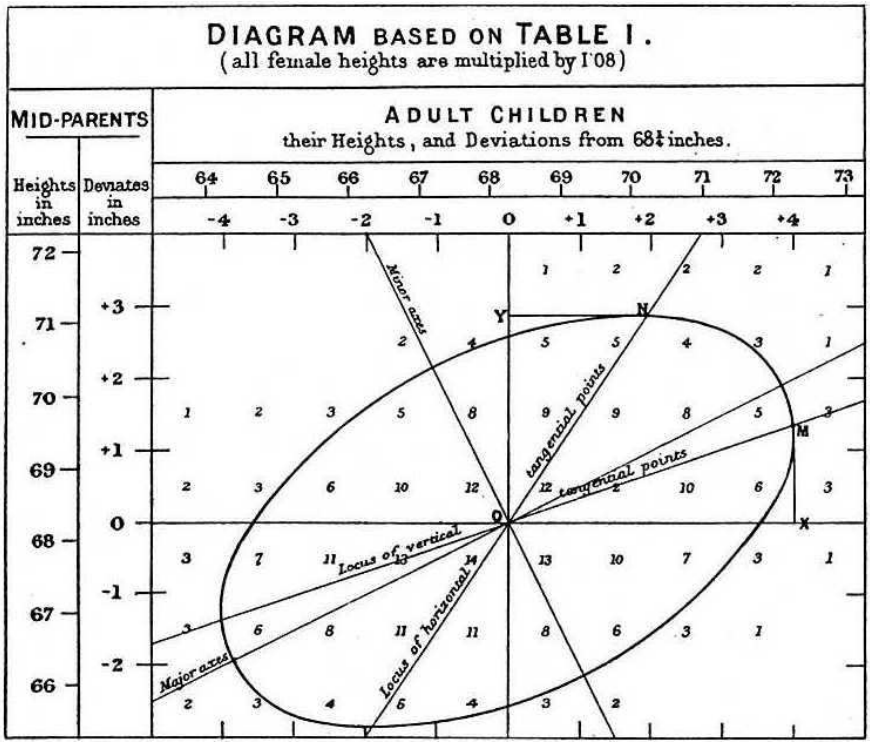


Figure 15: Galton's 1886 diagram, showing the relationship of height of children to the average of their parents' height. The diagram is essentially an overlay of a geometrical interpretation on a bivariate grouped frequency distribution, shown as numbers.

For two variables, x and y, the remarkable properties of the data ellipse are illustrated in Figure 16, a modern reconstruction of Galton's diagram.

- The ellipses have the mean vector $(\bar{x}, \bar{y})$ as their center.
- The lengths of arms of the central cross show the standard deviations of the variables, which correspond to the shadows of the ellipse covering 40% of the data. These are the bivariate analogs of the standard intervals $\bar{x} \pm 1 s_x$ and $\bar{y} \pm 1 s_y$.
- More generally, shadows (projections) on the coordinate axes, or any linear combination of them, give any standard interval, $\bar{x} \pm k s_x$ and $\bar{y} \pm k s_y$. Those with $k = 1, 1.5, 2.45$, have bivariate coverage 40%, 68% and 95% respectively, corresponding to these quantiles of the χ^2 distribution with 2 degrees of freedom, i.e., $\chi^2_2(.40) \approx 1^2$, $\chi^2_2(.68) \approx 1.5^2$, and $\chi^2_2(.95) \approx 2.45$. The shadows of the 68% ellipse are the bivariate analog of a univariate $\bar{x} \pm 1 s_x$ interval.

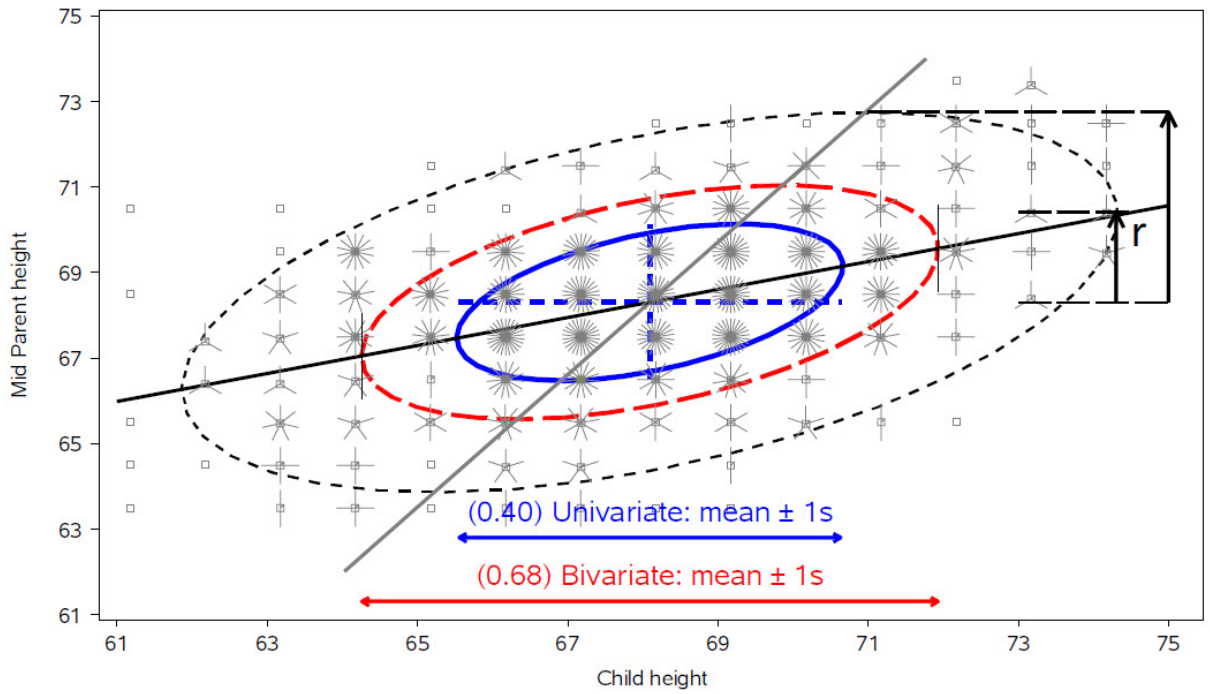


Figure 16: Sunflower plot of Galton's data on heights of parents and their children (in.), with 40%, 68% and 95% data ellipses and the regression lines of y on x (black) and x on y (grey).

- The regression line predicting y from x goes through the points where the ellipses have vertical tangents. The *other* regression line, predicting x from y goes through the points of horizontal tangency.
- The correlation $r(x, y)$ is the ratio of the vertical segment from the mean of y to the regression line to the vertical segment going to the top of the ellipse as shown at the right of the figure. It is $r = 0.46$ in this example.
- The residual standard deviation, $s_e = \sqrt{MSE} = \sqrt{\Sigma(y - \bar{y})^2/n - 2}$, is the half-length of the ellipse at the mean $\bar{x}$.

Because Galton's values of `parent` and `child` height were recorded in class intervals of 1 in., they are shown as sunflower symbols in Figure 16, with multiple 'petals' reflecting the number of observations at each location. This plot (except for annotations) is constructed using `sunflowerplot()` and `car::dataEllipse()` for the ellipses.

```
data(Galton, package = "HistData")

sunflowerplot(parent ~ child, data=Galton,
      xlim=c(61,75),
      ylim=c(61,75),
      seg.col="black",
        xlab="Child height",
      ylab="Mid Parent height")

y.x <- lm(parent ~ child, data=Galton)     # regression of y on x
abline(y.x, lwd=2)
x.y <- lm(child ~ parent, data=Galton)     # regression of x on y
```

```
cc <- coef(x.y)
abline(-cc[1]/cc[2], 1/cc[2], lwd=2, col="gray")

with(Galton,
     car::dataEllipse(child, parent,
         plot.points=FALSE,
         levels=c(0.40, 0.68, 0.95),
         lty=1:3)
    )
```

0.1.4.2 R functions for data ellipses

A number of packages provide functions for drawing data ellipses in a scatterplot, with various features.

- `car::scatterplot()`: uses base R graphics to draw 2D scatterplots, with a wide variety of plot enhancements including linear and non-parametric smoothers (loess, gam), a formula method, e.g., `y ~ x | group`, and marking points and lines using symbol shape, color, etc. Importantly, the **car** package generally allows automatic identification of "noteworthy" points by their labels in the plot using a variety of methods. For example, `method = "mahal"` labels cases with the most extreme Mahalanobis distances; `method = "r"` selects points according to their value of `abs(y)`, which is appropriate in residual plots.
- `car::dataEllipse()`: plots classical or robust data (using `MASS::cov/trob()`) ellipses for one or more groups, with the same facilities for point identification.
- `heplots::covEllipses()`: draws classical or robust data ellipses for one or more groups in a one-way design and optionally for the pooled total sample, where the focus is on homogeneity of within-group covariance matrices.
- `ggplot2::stat_ellipse()`: uses the calculation methods of `car::dataEllipse()` to add unfilled (`geom = "path"`) or filled (`geom = polygon"`) data ellipses in a `ggplot` scatterplot, using inherited aesthetics.

0.1.4.3 Example: Canadian occupational prestige

These examples use the data on the prestige of 102 occupational categories and other measures from the 1971 Canadian Census, recorded in `carData::Prestige`. Our interest is in understanding how `prestige` (the Pineo-Porter (Pineo & Porter, 2008) prestige score for an occupational category, derived from a social survey) is related to census measures of the average education, income, percent women of incumbents in those occupations. Occupation `type` is a factor with levels `"bc"` (blue collar), `"wc"` (white collar) and `"prof"` (professional).

```
data(Prestige, package="carData")
# `type` is really an ordered factor. Make it so.
Prestige$type <- ordered(Prestige$type,
                         levels=c("bc", "wc", "prof"))
str(Prestige)
#> 'data.frame':    102 obs. of  6 variables:
#>  $ education: num  13.1 12.3 12.8 11.4 14.6 ...
#>  $ income   : int  12351 25879 9271 8865 8403 11030 8258 14163 11377 11023 ...
#>  $ women    : num  11.16 4.02 15.7 9.11 11.68 ...
#>  $ prestige : num  68.8 69.1 63.4 56.8 73.5 77.6 72.6 78.1 73.1 68.8 ...
#>  $ census   : int  1113 1130 1171 1175 2111 2113 2133 2141 2143 2153 ...
#>  $ type     : Ord.factor w/ 3 levels "bc"<"wc"<"prof": 3 3 3 3 3 3 3 3 3 3 ...
```

I first illustrate the relation between `income` and `prestige` using `car::scatterplot()` with many of its bells and whistles, including marginal boxplots for the variables, the linear regression line, loess smooth and the 68% data ellipse.

```
scatterplot(prestige ~ income, data=Prestige,
  pch = 16, cex.lab = 1.25,
  regLine = list(col = "red", lwd=3),
  smooth = list(smoother=loessLine,
                lty.smooth = 1, lwd.smooth=3,
                col.smooth = "darkgreen", col.var = "darkgreen"),
  ellipse = list(levels = 0.68),
  id = list(n=4, method = "mahal", col="black", cex=1.2))
#> general.managers          lawyers        ministers       physicians
#>                2                17                20               24
```

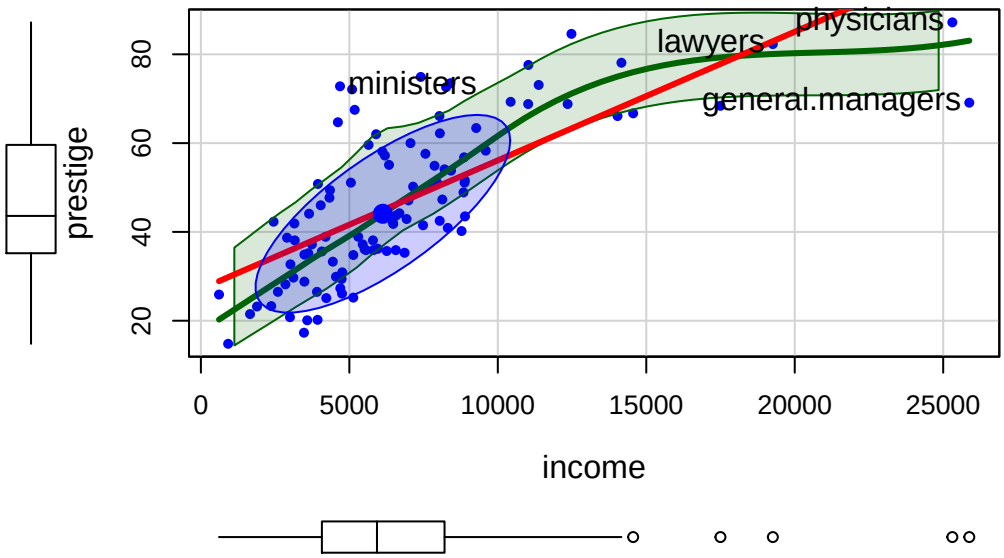


Figure 17: Scatterplot of prestige vs. income, showing the linear regression line (red), the loess smooth with a confidence envelope (darkgreen) and a 68% data ellipse.

There is a lot that can be seen here:

- `income` is positively skewed, as is often the case.
- The loess smooth, on the scale of income, shows `prestige` increasing up to $15,000 (these are 1971 incomes), and then leveling off.
- The data ellipse, centered at the means encloses approximately 68% of the data points. It adds visual information about the correlation and precision of the linear regression; but here, the non-linear trend for higher incomes strongly suggests a different approach.
- The four points identified by their labels are those with the largest Mahalanobis distances. `scatterplot()` prints their labels to the console.

Figure 18 shows a similar plot for education, which from the boxplot appears to be reasonably symmetric. The smoothed curve is quite close to the linear regression, according to which `prestige` increases on average `coef(lm(prestige ~ education, data=Prestige))[2]` = 5.361 with each year of education.

```
scatterplot(prestige ~ education, data=Prestige,
  pch = 16, cex.lab = 1.25,
  regLine = list(col = "red", lwd=3),
  smooth = list(smoother=loessLine,
                lty.smooth = 1, lwd.smooth=3,
```

```
                col.smooth = "darkgreen", col.var = "darkgreen"),
  ellipse = list(levels = 0.68),
  id = list(n=4, method = "mahal", col="black", cex=1.2))
#>  physicians file.clerks    newsboys     farmers
#>          24          41          53          67
```

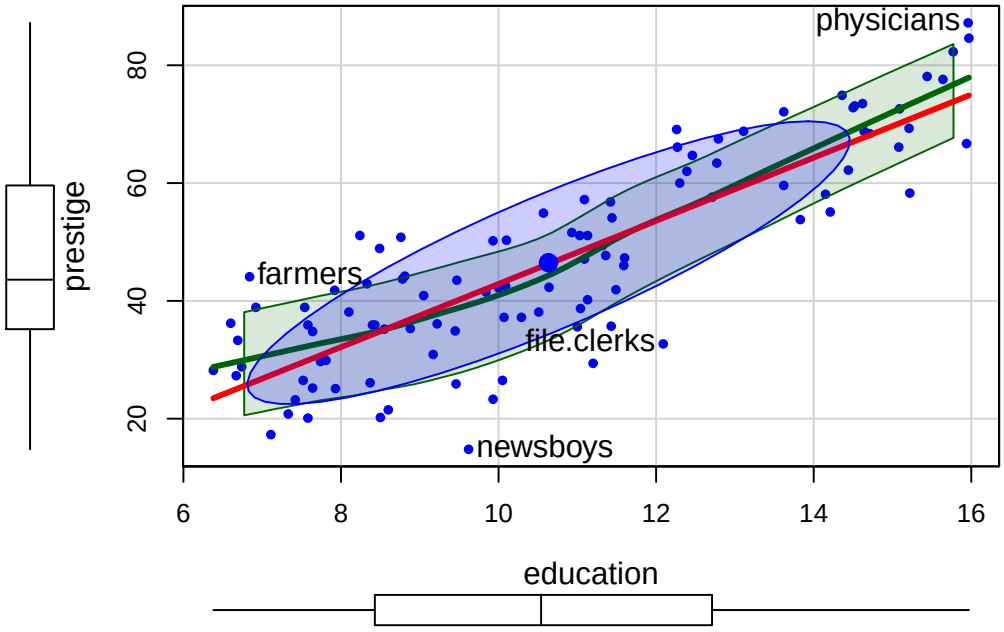


Figure 18: Scatterplot of prestige vs. education, showing the linear regression line (red), the loess smooth with a confidence envelope (darkgreen) and a 68% data ellipse.

In this plot, farmers, newsboys, file.clerks and physicians are identified as noteworthy, for being furthest from the mean by Mahalanobis distance. In relation to their typical level of education, these are mostly understandable, but it is nice that farmers are rated of higher prestige than their level of education would predict.

Note that the `method` argument for point identification can take a vector of case numbers indicating the points to be labeled. So, to label the observations with large absolute standardized residuals in the linear model `m`, you can use `method = which(abs(rstandard(m)) > 2)`.

```
m <- lm(prestige ~ education, data=Prestige)
scatterplot(prestige ~ education, data=Prestige,
            pch = 16, cex.lab = 1.25,
            boxplots = FALSE,
            regLine = list(col = "red", lwd=3),
            smooth = list(smoother=loessLine,
                          lty.smooth = 1, col.smooth = "black", lwd.smooth=3,
                          col.var = "darkgreen"),
            ellipse = list(levels = 0.68),
            id = list(n=4, method = which(abs(rstandard(m))>2),
                      col="black", cex=1.2)) |> invisible()
```

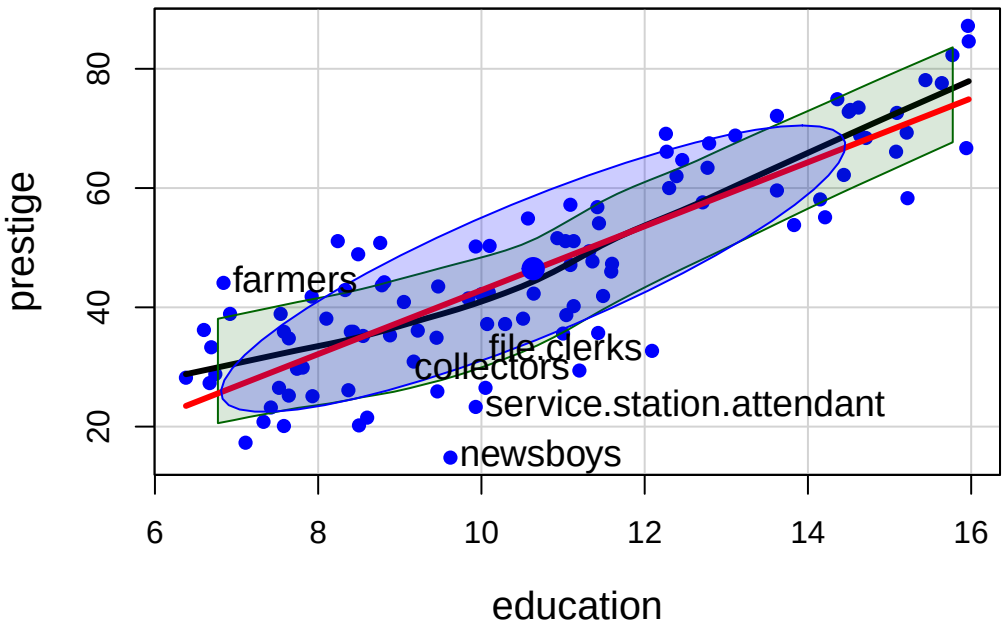


Figure 19: Scatterplot of prestige vs. education, labeling points whose absolute stanardized residual is > 2.

0.1.4.4 Plotting on a log scale

A typical remedy for the non-linear relationship of income to prestige is to plot income on a log scale. This usually makes sense, and expresses a belief that a **multiple** of or **percentage increase** in income has a constant impact on prestige, as opposed to the **additive** interpretation for income itself.

For example, the slope of the linear regression line in Figure 17 is given by `coef(lm(prestige ~ income, data=Prestige))[2]` $= 0.003$. Multiplying this by 1000 says that a \$1000 increase in `income` is associated with with an average increase of `prestige` of 2.9.

In the plot below, `scatterplot(..., log = "x")` re-scales the x-axis to the $\log_e()$ scale. The slope, `coef(lm(prestige ~ log(income), data=Prestige))[2]` $= 21.556$ says that a 1% increase in salary is associated with an average change of 21.55 / 100 in prestige.

```
scatterplot(prestige ~ income, data=Prestige,
  log = "x",
  pch = 16, cex.lab = 1.25,
  regLine = list(col = "red", lwd=3),
  smooth = list(smoother=loessLine,
                lty.smooth = 1, lwd.smooth=3,
                col.smooth = "darkgreen", col.var = "darkgreen"),
  ellipse = list(levels = 0.68),
  id = list(n=4, method = "mahal", col="black", cex=1.2))
#> general.managers        ministers         newsboys      babysitters
#>                2               20               53               63
```

The smoothed curve in Figure 20 exhibits a slight tendency to bend upwards, but a linear relation is a reasonable approximation.

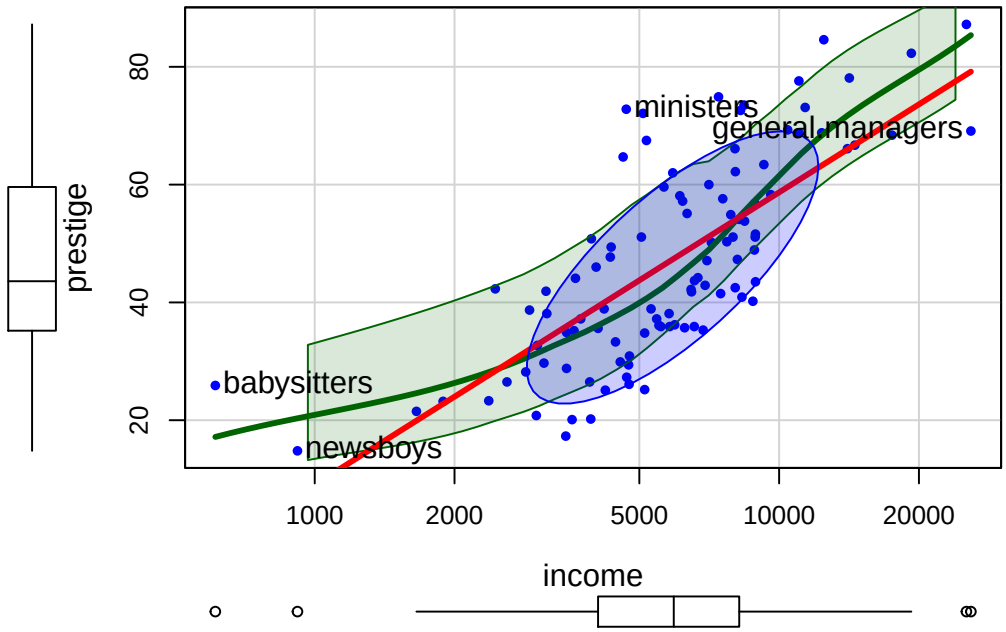


Figure 20: Scatterplot of prestige vs. log(income).

0.1.4.5 Stratifying

Before going further, it is instructive to ask what we could see in the relationship between income and prestige if we stratified by type of occupation, fitting separate regressions and smooths for blue collar, white collar and professional incumbents in these occupations.

The formula `prestige ~ income | type` is a natural way to specify grouping by `type`; separate linear regressions and smooths are calculated for each group, applying the color and point shapes specified by the `col` and `pch` arguments.

```
scatterplot(prestige ~ income | type, data=Prestige,
  col = c("blue", "red", "darkgreen"),
  pch = 15:17,
  grid = FALSE,
  legend = list(coords="bottomright"),
  regLine = list(lwd=3),
  smooth=list(smoother=loessLine,
              var=FALSE, lwd.smooth=2, lty.smooth=1))
```

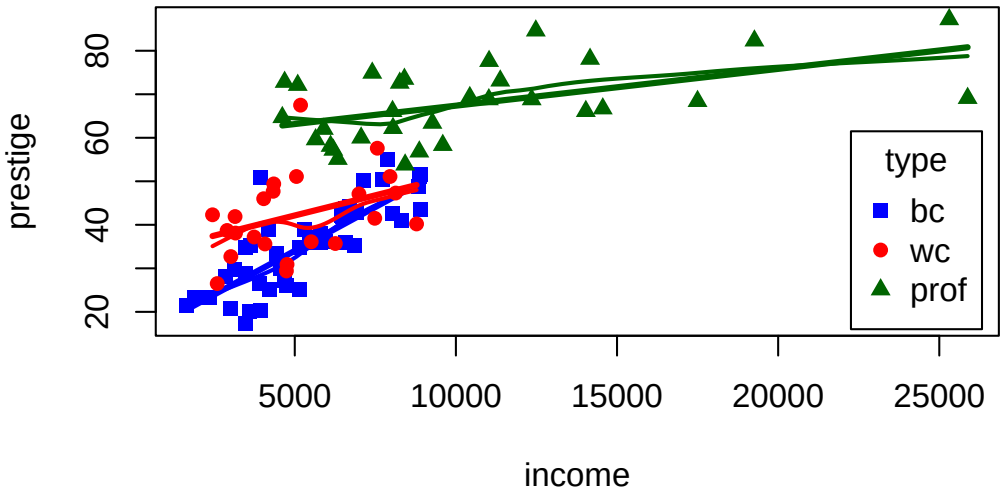


Figure 21: Scatterplot of prestige vs. income, stratified by occupational type. This implies a different interpretation, where occupation type is a moderator variable.

This visual analysis offers a different interpretation of the dependence of prestige on income, which appeared to be non-linear when occupation type was ignored. Instead, Figure 21 suggests an *interaction* of income by type. In a model formula this would be expressed as one of:

```
lm(prestige ~ income + type + income:type, data = Prestige)
lm(prestige ~ income * type, data = Prestige)
```

These models signify that there are different slopes (and intercepts) for the three occupational types. In this interpretation, `type` is a moderator variable, with a different story. The slopes of the fitted lines suggest that among blue collar workers, prestige increases sharply with their income. For white collar and professional workers, there is still an increasing relation of prestige with income, but the effect of income (slope) diminishes with higher occupational category. A different plot entails a different story.

0.1.4.6 Example: Penguins data

The `penguins` dataset from the **palmerpenguins** package (Horst et al., 2022) provides further instructive examples of plots and analyses of multivariate data. The data consists of measurements of body size (flipper length, body mass, bill length and depth) of 344 penguins collected at the Palmer Research Station in Antarctica.

There were three different species of penguins (Adélie, Chinstrap & Gentoo) collected from 3 islands in the Palmer Archipelago between 2007–2009 (Gorman et al., 2014). The purpose was to examine differences in size or appearance of these species, particularly differences among the sexes (sexual dimorphism) in relation to foraging and habitat.

Here, I use a slightly altered version of the dataset, `peng`, renaming variables to remove the units, making factors of character variables and deleting a few cases with missing data.

```
data(penguins, package = "palmerpenguins")
peng <- penguins |>
  rename(
    bill_length = bill_length_mm,
    bill_depth = bill_depth_mm,
    flipper_length = flipper_length_mm,
```

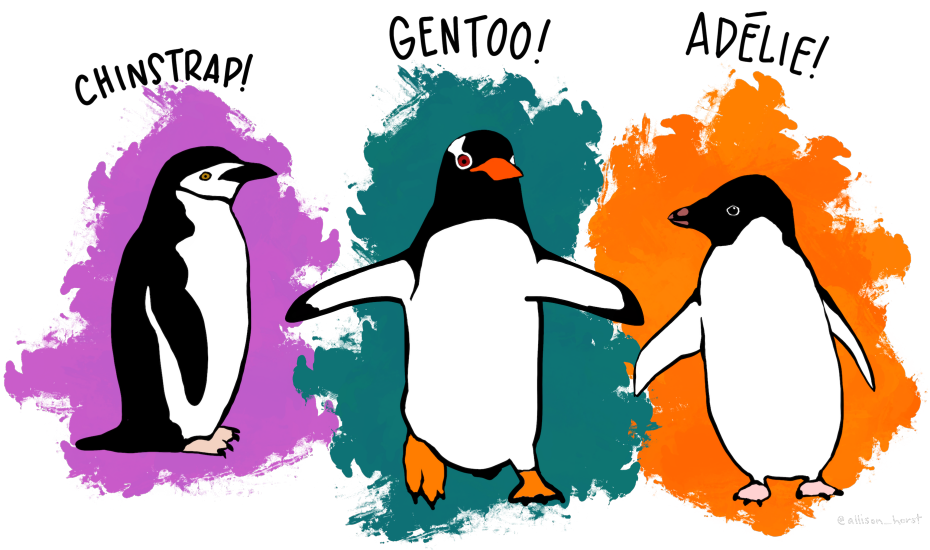


Figure 22: Penguin species observed in the Palmer Archipelago. This is a cartoon, but it illustrates some features of penguin body size measurements, and the colors typically used for species. Image: Allison Horst

```
    body_mass = body_mass_g
  ) |>
  mutate(species = as.factor(species),
         island = as.factor(island),
         sex = as.factor(substr(sex,1,1))) |>
  tidyr::drop_na()

str(peng)
#> tibble [333 x 8] (S3: tbl_df/tbl/data.frame)
#>  $ species       : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
#>  $ island        : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
#>  $ bill_length   : num [1:333] 39.1 39.5 40.3 36.7 39.3 38.9 39.2 41.1 38.6 34.6 ...
#>  $ bill_depth    : num [1:333] 18.7 17.4 18 19.3 20.6 17.8 19.6 17.6 21.2 21.1 ...
#>  $ flipper_length: int [1:333] 181 186 195 193 190 181 195 182 191 198 ...
#>  $ body_mass     : int [1:333] 3750 3800 3250 3450 3650 3625 4675 3200 3800 4400 ...
#>  $ sex           : Factor w/ 2 levels "f","m": 2 1 1 1 2 1 2 1 2 2 ...
#>  $ year          : int [1:333] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

There are quite a few variables to choose for illustrating data ellipses in scatterplots. Here I focus on the measures of their bills, `bill_length` and `bill_depth` (indicating curvature) and show how to use `ggplot2` for these plots.

I'll be using the penguins data quite a lot, so it is useful to set up custom colors like those used in Figure 22, and shown in Figure 23 with their color codes. These are shades of:

- Adelie: orange,
- Chinstrap: purple, and
- Gentoo: green.

To use these in `ggplot2` I define a function `peng.colors()` that allows shades of light, medium and dark and then functions `scale_*_penguins()` for color and fill.

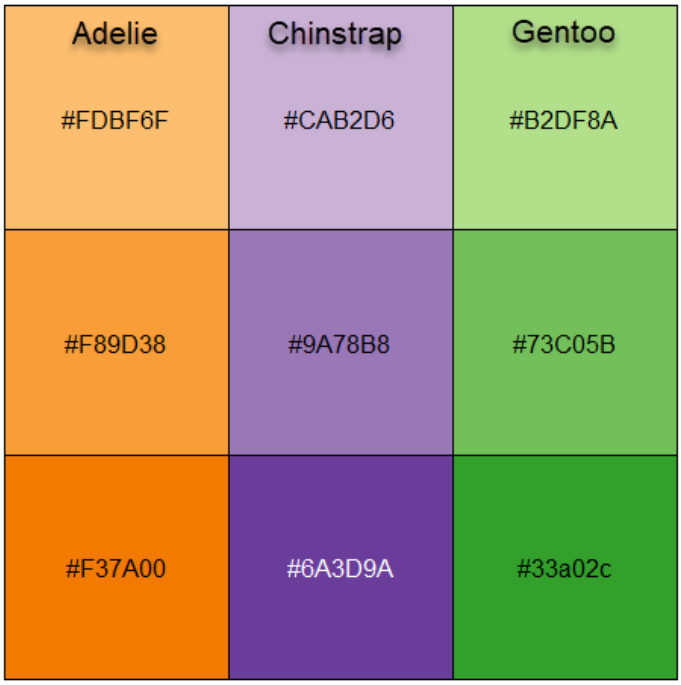


Figure 23: Color palettes used for penguin species.

```
peng.colors <- function(shade=c("medium", "light", "dark")) {
  shade = match.arg(shade)
  #             light      medium     dark
  oranges <- c("#FDBF6F", "#F89D38", "#F37A00")  # Adelie
  purples <- c("#CAB2D6", "#9A78B8", "#6A3D9A")  # Chinstrap
  greens <-  c("#B2DF8A", "#73C05B", "#33a02c")  # Gentoo

  cols.vec <- c(oranges, purples, greens)
  cols.mat <-
    matrix(cols.vec, 3, 3,
           byrow = TRUE,
           dimnames = list(species = c("Adelie", "Chinstrap", "Gentoo"),
                           shade = c("light", "medium", "dark")))
  # get shaded colors
  cols.mat[, shade ]
}

# define color and fill scales
scale_fill_penguins <- function(shade=c("medium", "light", "dark"), ...){
  shade = match.arg(shade)
  ggplot2::discrete_scale(
    "fill","penguins",
     scales:::manual_pal(values = peng.colors(shade)), ...)
}

scale_colour_penguins <- function(shade=c("medium", "light", "dark"), ...){
```

```
  shade = match.arg(shade)
  ggplot2::discrete_scale(
    "colour","penguins",
    scales:::manual_pal(values = peng.colors(shade)), ...)
}
scale_color_penguins <- scale_colour_penguins
```

This is used to define a `theme_penguins()` function that I use to simply change the color and fill scales for plots below.

```
theme_penguins <- function(shade=c("medium", "light", "dark"), ...) {
  shade = match.arg(shade)
  list(scale_color_penguins(shade=shade),
       scale_fill_penguins(shade=shade)
      )
}
```

An initial plot using `ggplot2` shown in Figure 24 uses color and point shape to distinguish the three penguin species. I annotate the plot of points using the linear regression lines, loess smooths to check for non-linearity and 95% data ellipses to show precision of the linear relation.

```
ggplot(peng,
       aes(x = bill_length, y = bill_depth,
           color = species, shape = species, fill=species)) +
  geom_point(size=2) +
  geom_smooth(method = "lm", formula = y ~ x,
              se=FALSE, linewidth=2) +
  geom_smooth(method = "loess",  formula = y ~ x,
              linewidth = 1.5, se = FALSE, alpha=0.1) +
  stat_ellipse(geom = "polygon", level = 0.95, alpha = 0.2) +
  theme_penguins("dark") +
  theme(legend.position = c(0.85, 0.15))
```

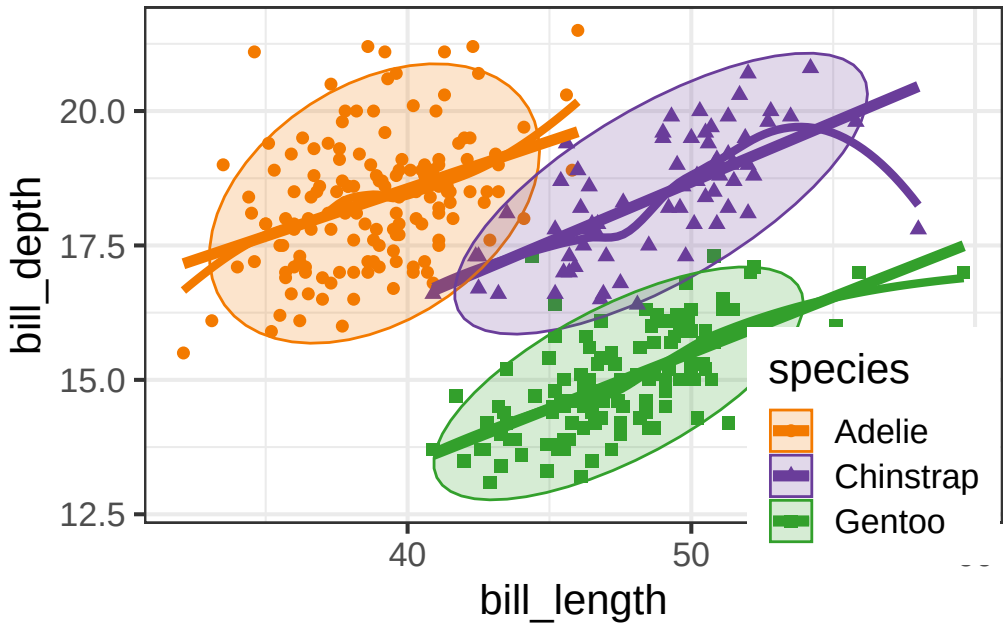


Figure 24: Penguin bill length and bill depth according to species.

Overall, the three species occupy different regions of this 2D space and for each species the relation between bill length and depth appears reasonably linear. Given this, we can suppress plotting the data points to get a visual summary of the data using the fitted regression lines and data ellipses, as shown in Figure 25.

This idea, of **visual thinning** a graph to focus on what should be seen, becomes increasingly useful as the data becomes more complex. The `ggplot2` framework encourages this, because we can think of various components as layers, to be included or not. Here I chose to include only the regression line and add data ellipses of 40%, 68% and 95% coverage to highlight the increasing bivariate density around the group means.

```
ggplot(peng,
       aes(x = bill_length, y = bill_depth,
           color = species, shape = species, fill=species)) +
  geom_smooth(method = "lm",  se=FALSE, linewidth=2) +
  stat_ellipse(geom = "polygon", level = 0.95, alpha = 0.2) +
  stat_ellipse(geom = "polygon", level = 0.68, alpha = 0.2) +
  stat_ellipse(geom = "polygon", level = 0.40, alpha = 0.2) +
  theme_penguins("dark") +
  theme(legend.position = c(0.85, 0.15))
```

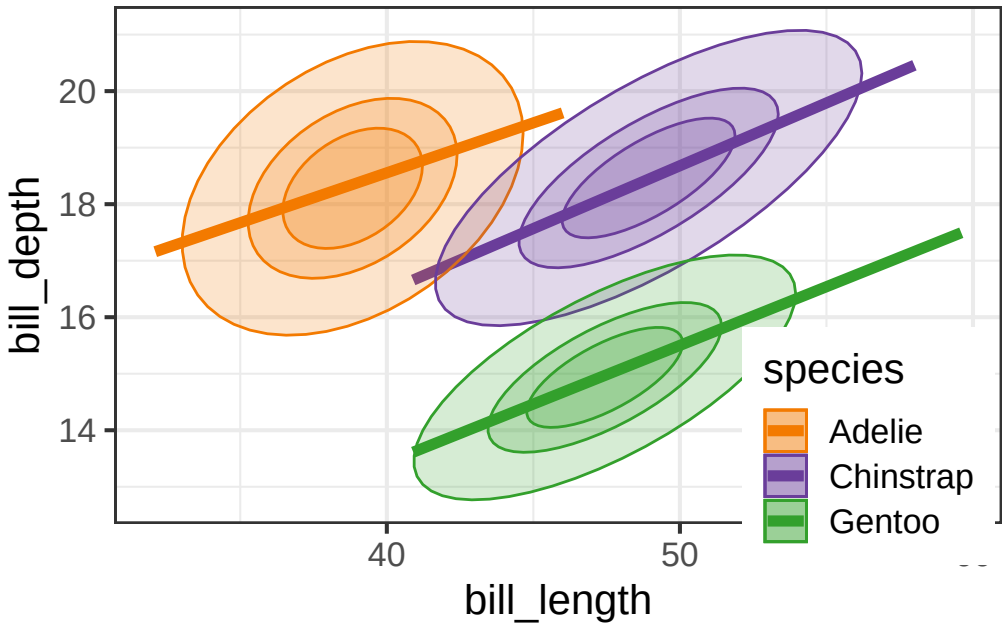


Figure 25: Visual thinning: Suppressing the data points gives a visual summary of the relation between bill length and bill depth using the regression line and data ellipses.

0.1.4.7 Nonparamtric bivariate density plots

While I emphasize data ellipses (because I like their beautiful geometry), other visual summaries of the bivariate density are possible and often useful.

For a single variable, `stats::density()` and `ggplot2::geom_density()` calculate a smoothed estimate of the density using nonparametric kernel methods (Silverman, 1986) whose smoothness is controlled by a bandwidth parameter, analogous to the span in a loess smoother. This idea extends to two (and more) variables (Scott, 1992). For bivariate data, `MASS::kde2d()` estimates the density on a square $n \times n$ grid over the ranges of the variables.

`ggplot2` provides `geom_density_2d()` which uses `MASS::kde2d()` and displays these as contours— horizontal slices of the 3D surface at equally-spaced heights and projects these onto the 2D plane. The **ggdensity** package (Otto & Kahle, 2023) extends this with `geom_hdr()`, computing the high density regions that bound given levels of probability and maps these to the `alpha` transparency aesthetic. A `method` argument allows you to specify various nonparametric (`method ="kde"` is the default) and parametric (`method ="mvnorm"` gives normal data ellipses) ways to estimate the underlying bivariate distribution.

Figure 26 shows these side-by-side for comparison. With `geom_density_2d()` you can specify either the number of contour `bins` or the width of these bins (`binwidth`). For `geom_hdr()`, the `probs` argument gives a result that is easier to understand.

```
library(ggdensity)
library(patchwork)
p1 <- ggplot(peng,
       aes(x = bill_length, y = bill_depth,
           color = species)) +
  geom_smooth(method = "lm",  se=FALSE, linewidth=2) +
  geom_density_2d(linewidth = 1.1, bins = 8) +
  ggtitle("geom_density_2d") +
  theme_bw(base_size = 14) +
```

```
  theme_penguins() +
  theme(legend.position = c(0.85, 0.15))

p2 <- ggplot(peng,
       aes(x = bill_length, y = bill_depth,
           color = species, fill = species)) +
  geom_smooth(method = "lm",  se=FALSE, linewidth=2) +
  geom_hdr(probs = c(0.95, 0.68, 0.4), show.legend = FALSE) +
  ggtitle("ggdensity::geom_hdr") +
  theme_bw(base_size = 14) +
  theme_penguins() +
  theme(legend.position = "none")

p1 + p2
```

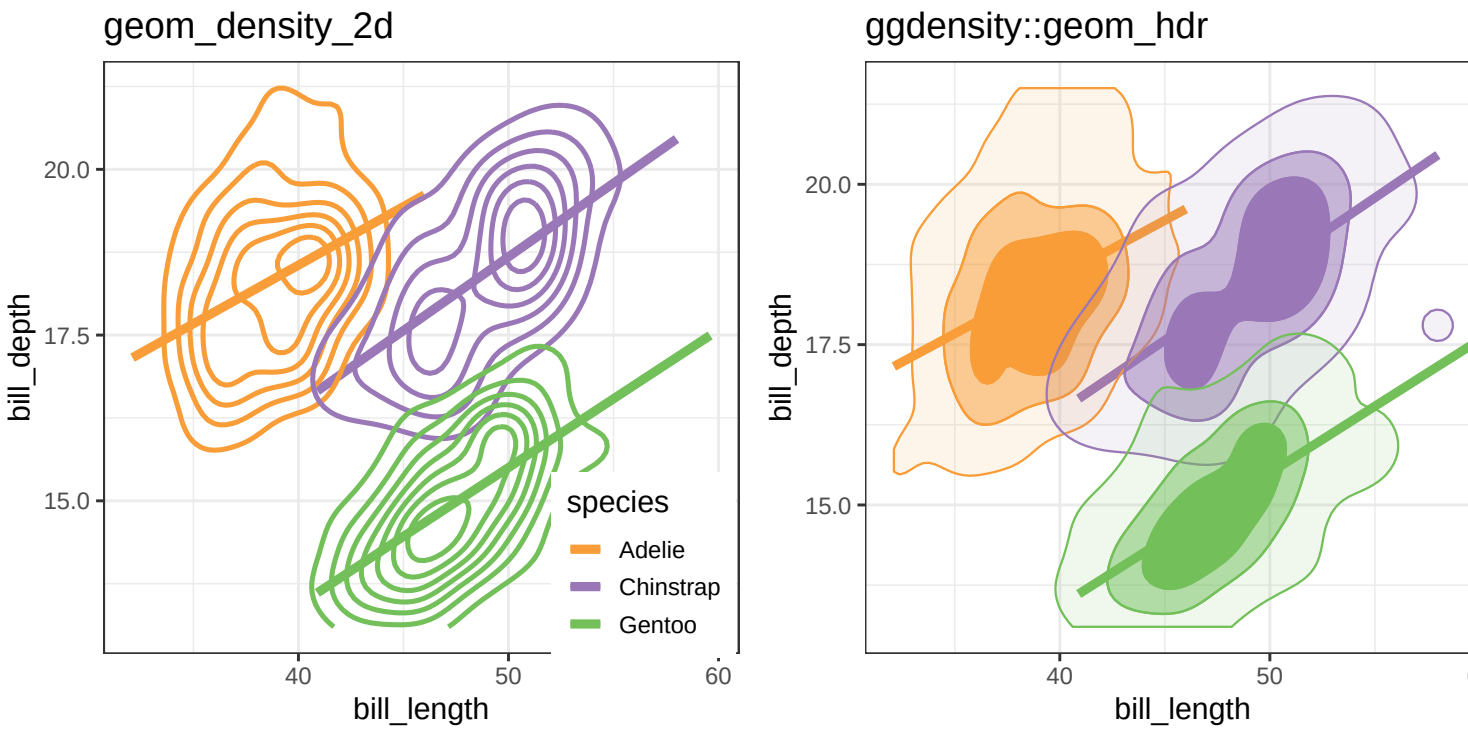


Figure 26: Bivariate densities show the contours of the 3D surface representing the frequency in the joint distribution of bill length and bill depth.

0.2 Scatterplot matrices

Going beyond bivariate scatterplots, a *pairs* plot (or *scatterplot matrix*) displays all possible $p \times p$ pairs of p variables in a matrix-like display where variables (x_i, x_j) are shown in a plot for row i, column j. This idea, due to Hartigan (1975b), uses small multiple plots, so that the eye can easily scan across a row or down a column to see how a given variable is related to all the others.

The most basic version is provided by `pairs()` in base R. When one variable is considered as an outcome or

response, it is usually helpful to put this in the first row and column. For the `Prestige` data, in addition to income and education, we also have a measure of % women in each occupational category.

Plotting these together gives Figure 27. In such plots, the diagonal cells give labels for the variables, but they are also a guide to interpreting what is shown. In each row, say row 2 for `income`, income is the vertical y variable in plots against other variables. In each column, say column 3 for `education`, education is the horizontal x variable.

```
pairs(~ prestige + income + education + women,
      data=Prestige)
```

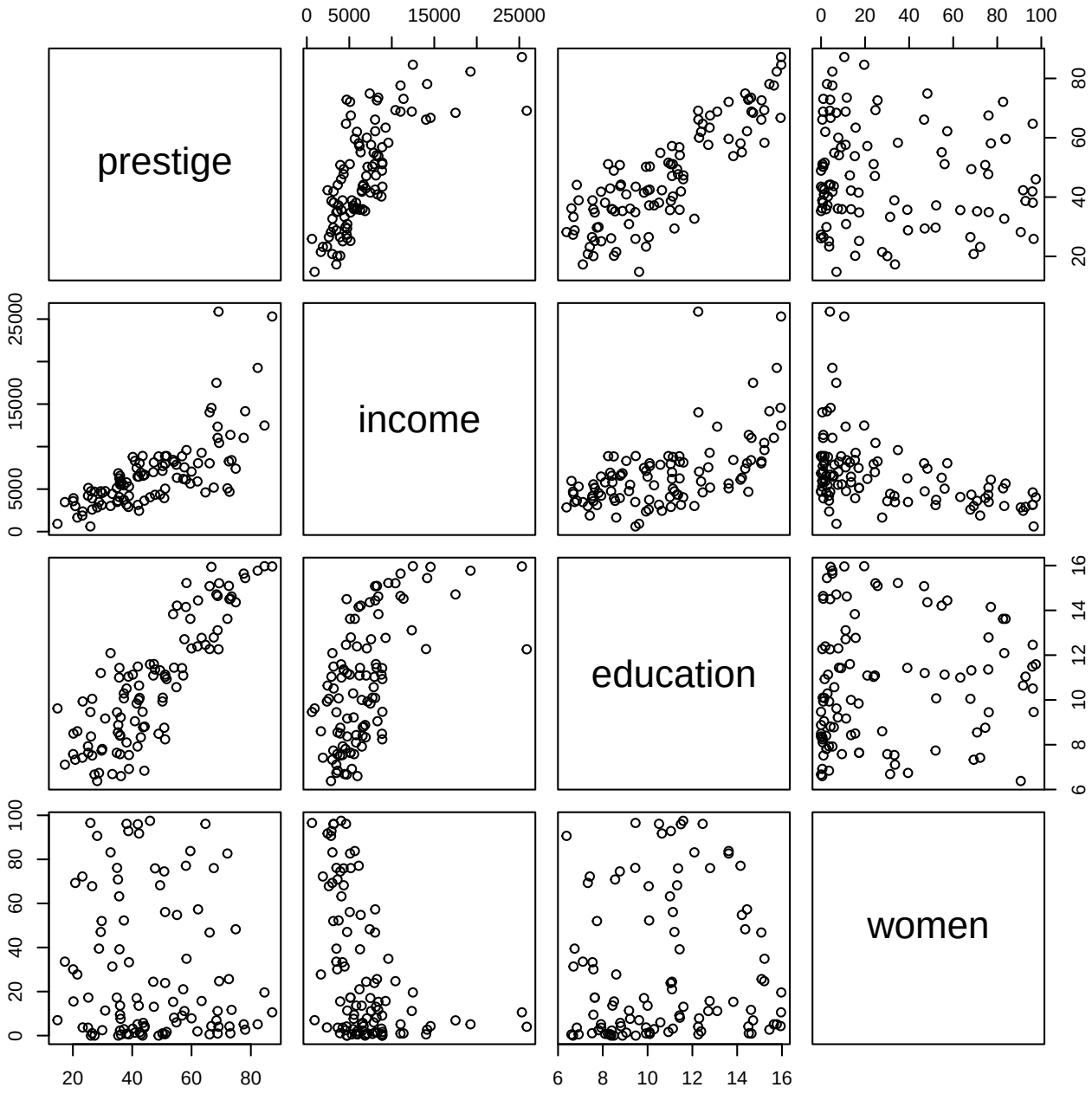


Figure 27: Scatterplot matrix of the variables in the Prestige dataset produced by `pairs()`

The plots in the first row show what we have seen before for the relations between prestige and income and education, adding to those the plot of prestige vs. % women. Plots in the first column show the same data, but with x and y interchanged.

But this basic `pairs()` plot is very limited. A more feature-rich version is provided by `car::scatterplotMatrix()` which can add the regression lines, loess smooths and data ellipses for each pair, as shown in Figure 28.

The diagonal panels show density curves for the distribution of each variable; for example, the distribution of `education` appears to be multi-modal and that of `women` shows that most of the occupations have a low percentage of women.

The combination of the regression line with the loess smoothed curve, but without their confidence envelopes, provides about the right amount of detail to take in at a glance where the relations are non-linear. We've already seen (Figure 17) the non-linear relation between prestige and income (row 1, column 2) when occupational type is ignored. But all relations with income in column 2 are non-linear, reinforcing our idea (Section 0.1.4.4) that effects of income should be assessed on a log scale.

```
scatterplotMatrix(~ prestige + income + education + women,
  data=Prestige,
  regLine = list(method=lm, lty=1, lwd=2, col="black"),
  smooth=list(smoother=loessLine, spread=FALSE,
              lty.smooth=1, lwd.smooth=3, col.smooth="red"),
  ellipse=list(levels=0.68, fill.alpha=0.1))
```

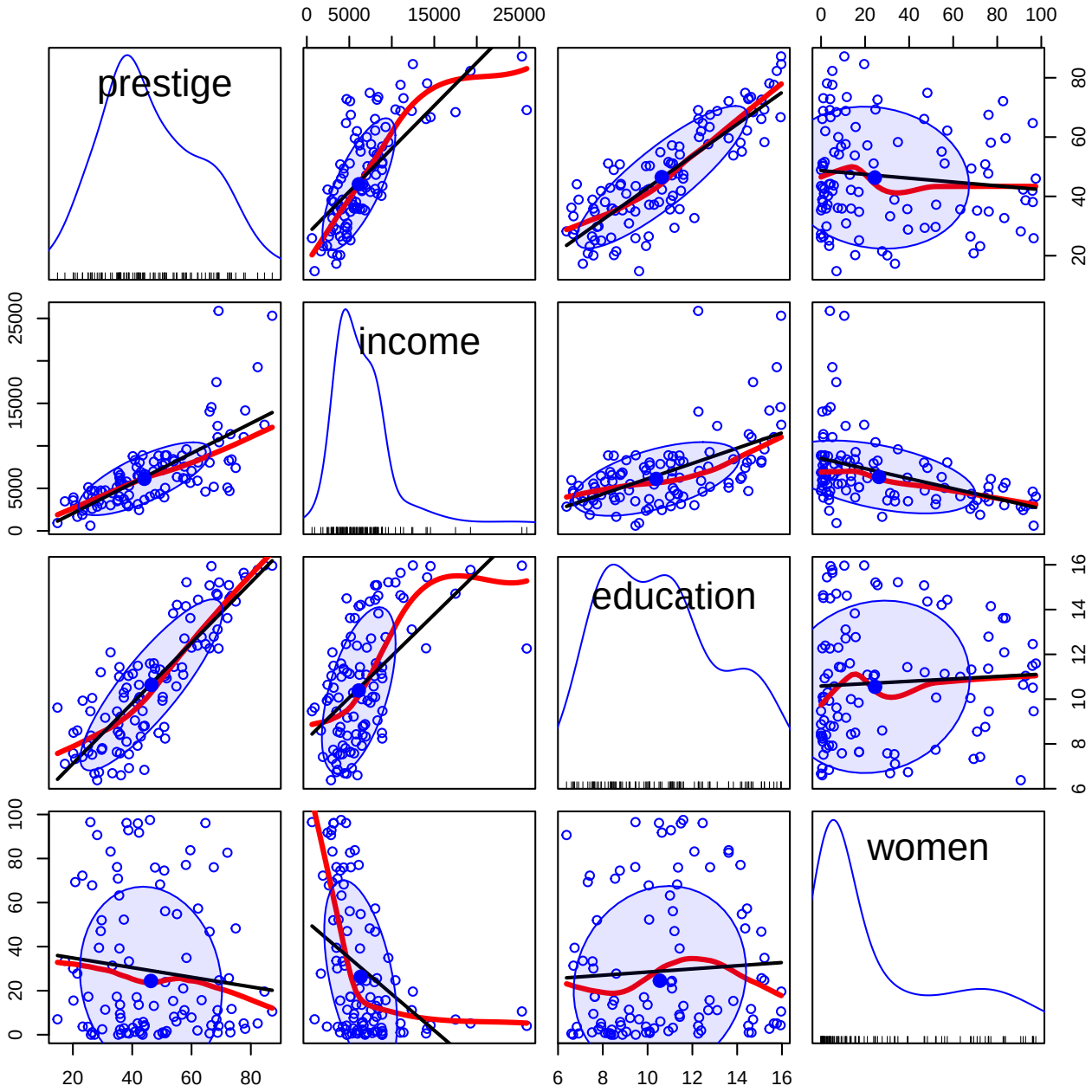


Figure 28: Scatterplot matrix of the variables in the Prestige dataset from `car::scatterplotMatrix()`.

`scatterplotMatrix()` can also label points using the `id =` argument (though this can get messy) and can stratify the observations by a grouping variable with different symbols and colors. For example, Figure 29 uses the syntax `~ prestige + education + income + women | type` to provide separate regression lines, smoothed curves and data ellipses for the three types of occupations. (The default colors are somewhat garish, so I use `scales::hue_pal()` to mimic the discrete color scale used in `ggplot2`).

```
scatterplotMatrix(~ prestige + income + education + women | type,
  data = Prestige,
  col = scales::hue_pal()(3),
  pch = 15:17,
```

```
    smooth=list(smoother=loessLine, spread=FALSE,
                lty.smooth=1, lwd.smooth=3, col.smooth="black"),
    ellipse=list(levels=0.68, fill.alpha=0.1))
```

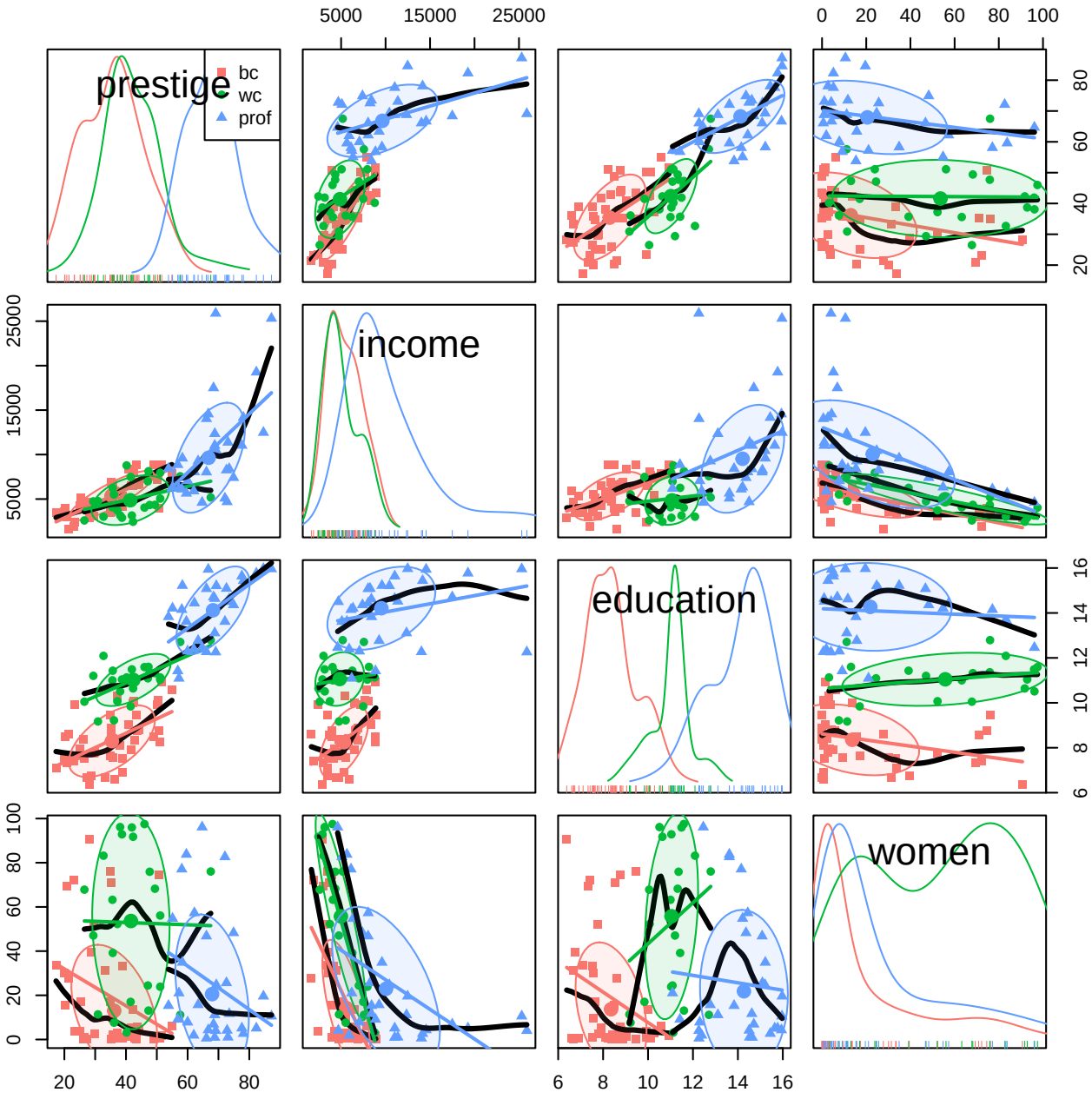


Figure 29: Scatterplot matrix of the variables in the Prestige dataset from `car::scatterplotMatrix()`, stratified by type of occupation.

It is now easy to see why education is multi-modal: blue collar, white collar and professional occupations have largely non-overlapping years of education. As well, the distribution of % women is much higher in the white collar category.

For the `penguins` data, given what we've seen before in Figure 24 and Figure 25, we may wish to suppress

details of the points (`plot.points = FALSE`) and loess smooths (`smooth = FALSE`) to focus attention on the similarity of regression lines and data ellipses for the three penguin species. In Figure 30, I've chosen to show boxplots rather than density curves in the diagonal panels in order to highlight differences in the means and interquartile ranges of the species, and to show 68% and 95% data ellipses in the off-diagonal panels.

```
scatterplotMatrix(~ bill_length + bill_depth + flipper_length + body_mass | species,
  data = peng,
  col = peng.colors("medium"),
  legend=FALSE,
  ellipse = list(levels = c(0.68, 0.95),
                 fill.alpha = 0.1),
  regLine = list(lwd=3),
  diagonal = list(method = "boxplot"),
  smooth = FALSE,
  plot.points = FALSE,
  cex.labels=1)
```

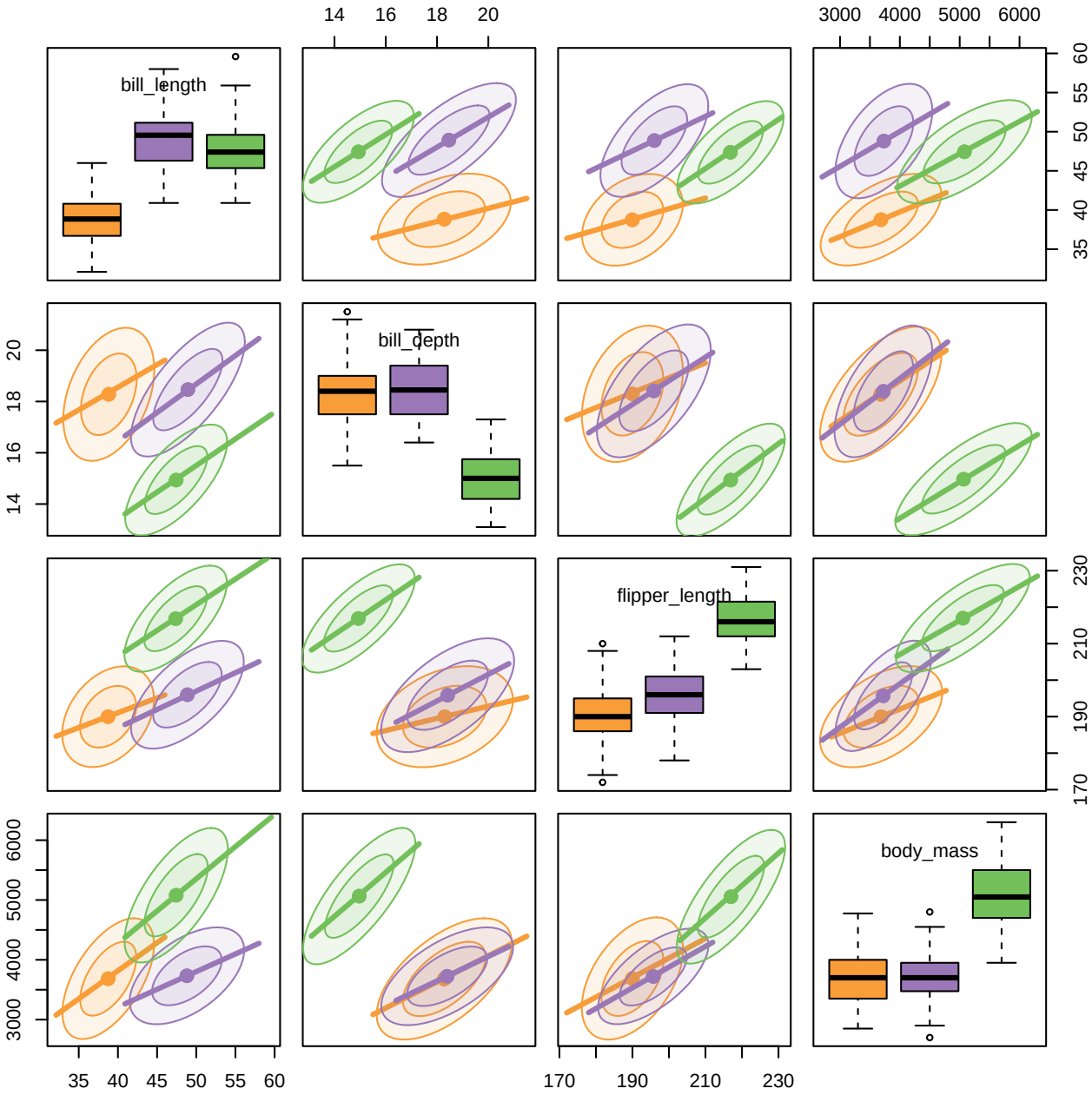


Figure 30: Scatterplot matrix of the variables in the penguins dataset, stratified by species.

It can be seen that the species are widely separated in most of the bivariate plots. As well, the regression lines for species have similar slopes and the data ellipses have similar size and shape in most of the plots. From the boxplots, we can also see that Adelie penguins have shorter bill lengths than the others, while Gentoo penguins have smaller bill depth, but longer flippers and are heavier than Chinstrap and Adelie penguins.

> **Looking ahead**
>
> Figure 30 provides a reasonably complete visual summary of the data in relation to multivariate models that ask "do the species differ in their means on these body size measures?" This corresponds to the MANOVA model,

```
peng.mod <- lm(cbind(bill_length, bill_depth, flipper_length, body_mass) ~ species,
               data=peng)
```

Hypothesis-error (HE) plots, described in Chapter 0.32 provide a better summary of the evidence for the MANOVA test of differences among means on all variables together. These give an **H** ellipse reflecting the differences among means, to be compared with an **E** ellipse reflecting within-group variation and a visual test of significance.
A related question is "how well are the penguin species distinguished by these body size measures?" Here, the relevant model is linear discriminant analysis (LDA), where `species` plays the role of the response in the model,

```
peng.lda <- MASS:lda( species ~ cbind(bill_length, bill_depth, flipper_length, body_mass),
               data=peng)
```

Both MANOVA and LDA depend on the assumption that the variances and correlations between the variables are the same for all groups. This assumption can be tested and visualized using the methods in Chapter 0.41.2.

0.2.1 Visual thinning

What can you do if there are even more variables than in these examples? If what you want is a high-level, zoomed-out display summarizing the pairwise relations more strongly, you can apply the idea of visual thinning to show only the most important features.

This example uses data on the rate of various crimes in the 50 U.S. states from the United States Statistical Abstracts, 1970, used by Hartigan (1975a) and Friendly (1991). These are ordered in the dataset roughly by seriousness of crime or from crimes of violence to property crimes.

```
data(crime, package = "ggbiplot")
str(crime)
#> 'data.frame':    50 obs. of  10 variables:
#>  $ state   : chr  "Alabama" "Alaska" "Arizona" "Arkansas" ...
#>  $ murder  : num  14.2 10.8 9.5 8.8 11.5 6.3 4.2 6 10.2 11.7 ...
#>  $ rape    : num  25.2 51.6 34.2 27.6 49.4 42 16.8 24.9 39.6 31.1 ...
#>  $ robbery : num  96.8 96.8 138.2 83.2 287 ...
#>  $ assault : num  278 284 312 203 358 ...
#>  $ burglary: num  1136 1332 2346 973 2139 ...
#>  $ larceny : num  1882 3370 4467 1862 3500 ...
#>  $ auto    : num  281 753 440 183 664 ...
#>  $ st      : chr  "AL" "AK" "AZ" "AR" ...
#>  $ region  : Factor w/ 4 levels "Northeast","South",..: 2 4 4 2 4 4 1 2 2 2 ...
```

Figure 31 displays the scatterplot matrix for these seven variables, using only the regression line and data ellipse to show the linear relation and the loess smooth to show potential non-linearity. To make this even more schematic, the axis tick marks and labels are also removed using the `par()` settings `xaxt = "n", yaxt = "n"`.

```
crime |>
  select(where(is.numeric)) |>
  scatterplotMatrix(
```

```
    plot.points = FALSE,
    ellipse = list(levels = 0.68, fill=FALSE),
    smooth = list(spread = FALSE,
                  lwd.smooth=2, lty.smooth = 1, col.smooth = "red"),
    cex.labels = 2,
    xaxt = "n", yaxt = "n")
```

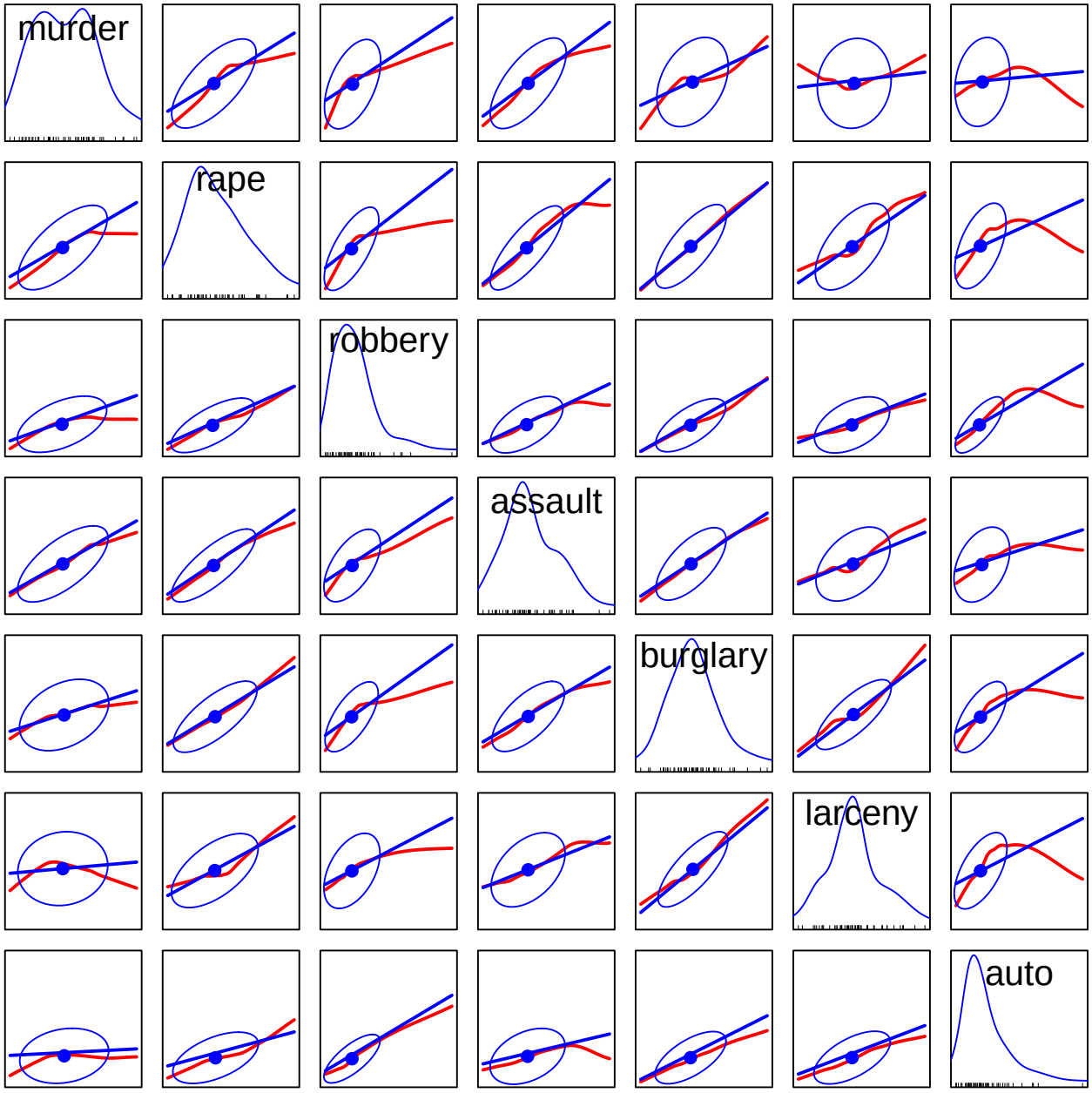


Figure 31: Visual thinning: Scatterplot matrix of the crime data, showing only high-level summaries of the linear and nonlinear relations betgween each pair of variables.

We can see that all pairwise correlations are positive, pairs closer to the main diagonal tend to be more highly

correlated and in most cases the nonparametric smooth doesn't differ much from the linear regression line. Exceptions to this appear mainly in the columns for `robbery` and `auto` (auto theft).

0.2.2 Corrgrams

What if you want to summarize the data even further, for example to show only the value of the correlation for each pair of variables? A **corrgram** (Friendly, 2002) is a visual display of a correlation matrix, where the correlation can be rendered in a variety of ways to show the direction and magnitude: circular "pac-man" (or pie) symbols, ellipses, colored vars or shaded rectangles, as shown in Figure 32.

Another aspect is that of **effect ordering** (Friendly & Kwan, 2003), ordering the levels of factors and variables in graphic displays to make important features most apparent. For variables, this means that we can arrange the variables in a matrix-like display in such a way as to make the pattern of relationships easiest to see. Methods to achieve this include using principal components and cluster analysis to put the most related variables together as described in Chapter 0.4.

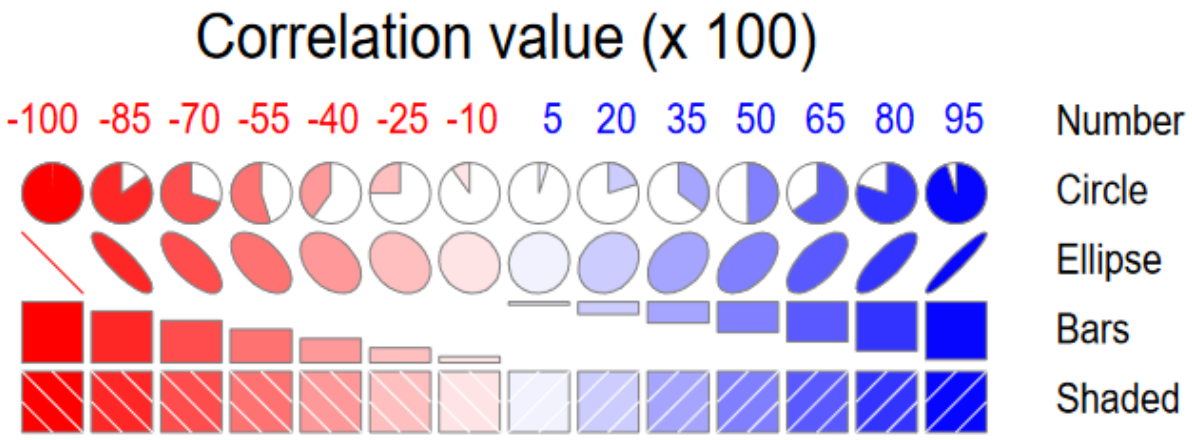


Figure 32: Corrgrams: Some renderings for the value of a correlation in a corrgram display, conveying sign and magnitude in different ways.

In R, these diagrams can be created using the **corrgram** (Wright, 2021) and **corrplot** (Wei & Simko, 2021) packages, with different features. `corrgram::corrgram()` is closest to Friendly (2002), in that it allows different rendering functions for the lower, upper and diagonal panels as illustrated in Figure 32. For example, a corrgram similar to Figure 31 can be produced as follows (not shown here):

```
crime |>
  select(where(is.numeric)) |>
  corrgram(lower.panel = panel.ellipse,
           upper.panel = panel.ellipse,
           diag.panel = panel.density)
```

`corrplot::corrplot()` provides the rendering methods `c("circle", "square", "ellipse", "number", "shade", "color", "pie")`, but only one can be used at a time. The function `corrplot::corrplot.mixed()` allows different options to be selected for the lower and upper triangles. The iconic shape is colored with a gradient in relation to the correlation value.

```
crime |>
  select(where(is.numeric)) |>
```

```
cor() |>
corrplot.mixed(
         lower = "ellipse",
         upper = "pie",
         tl.col = "black",
         tl.srt = 0,
         addCoef.col = "black",
         addCoefasPercent = TRUE)
```

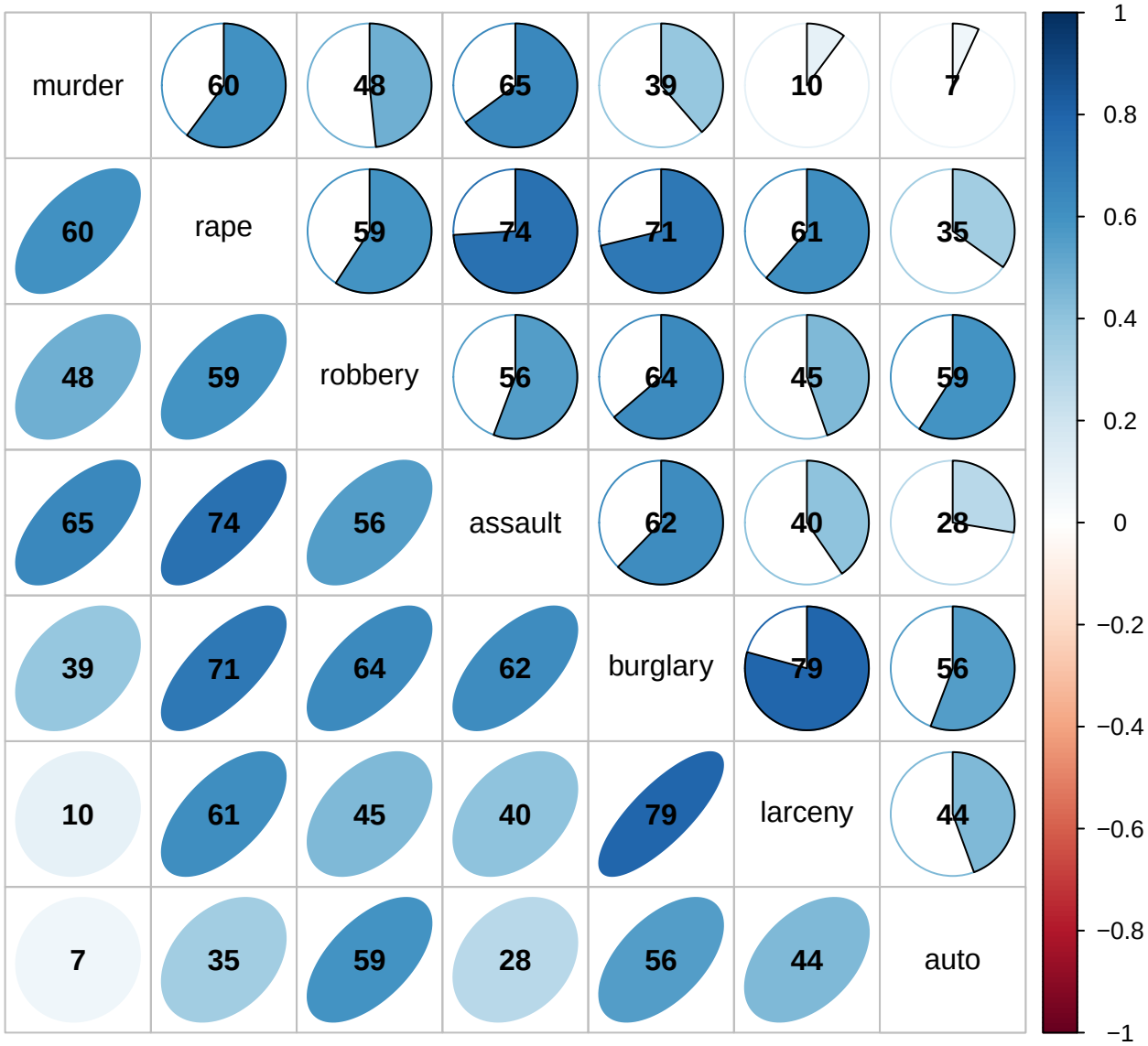


Figure 33: Corrplot of the `crime` data, showing the correlation between each pair of variables with an ellipse (lower) and a pie chart symbol (upper), all shaded in proportion to the correlation value, also shown numerically.

The combination of renderings shown in Figure 33 is instructive. Small differences among correlation values are easier to see with the pie symbols than with the ellipses; for example, compare the values for murder with

larceny and auto theft in row 1, columns 6-7 with those in column 1, rows 6-7, where the former are easier to distinguish. The shading color adds another visual cue.

Variations of corrgrams are worthy replacements for a numeric table of correlations, which are often presented in publications only for archival value. Including the numeric value (rounded here, for presentation purposes), makes this an attractive alternative to boring tables of correlations.

TODO: Add example showing correlation ordering – e.g., `mtcars` data.

0.3 Generalized pairs plots

When a dataset contains one or more discrete variables, the traditional pairs plot cannot cope, using only color and/or point symbols to represent categorical variables. In the context of mosaic displays and loglinear models, representing n-way frequency tables by rectangular tiles depicting cell frequencies, I (Friendly, 1994) proposed an analog of the scatterplot matrix using mosaic plots for each pair of variables. The **vcd** package (Meyer et al., 2023) implements very general `pairs()` methods for `"table"` objects. See my book *Discrete Data Analysis with R* (Friendly & Meyer, 2016) and the **vcdExtra** (Friendly, 2023) package for mosaic plots and mosaic matrices.

For example, we can tabulate the distributions of penguin species by sex and the island where they were observed using `xtabs()`. `ftable()` prints this three-way table more compactly. (In this example, and what follows in the chapter, I've changed the labels for sex from ("f", "m") to ("Female", "Male")).

```
# use better labels for sex
peng <- peng |>
  mutate(sex = factor(sex, labels = c("Female", "Male")))
peng.table <- xtabs(~ species + sex + island, data = peng)

ftable(peng.table)
#>                  island Biscoe Dream Torgersen
#> species   sex
#> Adelie    Female            22    27        24
#>           Male              22    28        23
#> Chinstrap Female             0    34         0
#>           Male               0    34         0
#> Gentoo    Female            58     0         0
#>           Male              61     0         0
```

We can see immediately that the penguin species differ by island: only Adelie were observed on all three islands; Biscoe Island had no Chinstraps and Dream Island had no Gentoos.

`vcd::pairs()` produces all pairwise mosaic plots, as shown in Figure 34. The diagonal panels show the one-way frequencies by width of the divided bars. Each off-diagonal panel shows the bivariate counts, breaking down each column variable by splitting the bars in proportion to a second variable. Consequently, the frequency of each cell is represented by its' area. The purpose is to show the **pattern of association** between each pair, and so, the tiles in the mosaic are shaded according to the signed standardized residual, $d_{ij} = (n_{ij} - \hat{n}_{ij})/\sqrt{\hat{n}_{ij}}$ in a simple $\chi^2 = \Sigma_{ij}\, d_{ij}^2$ test for association— blue where the observed frequency n_{ij} is significantly greater than expected $\hat{n}_{ij}$ under independence, and red where it is less than expected. The tiles are unshaded when $|d_{ij}| < 2$.

```
library(vcd)
pairs(peng.table, shade = TRUE,
      lower_panel_args = list(labeling = labeling_values()),
      upper_panel_args = list(labeling = labeling_values()))
```

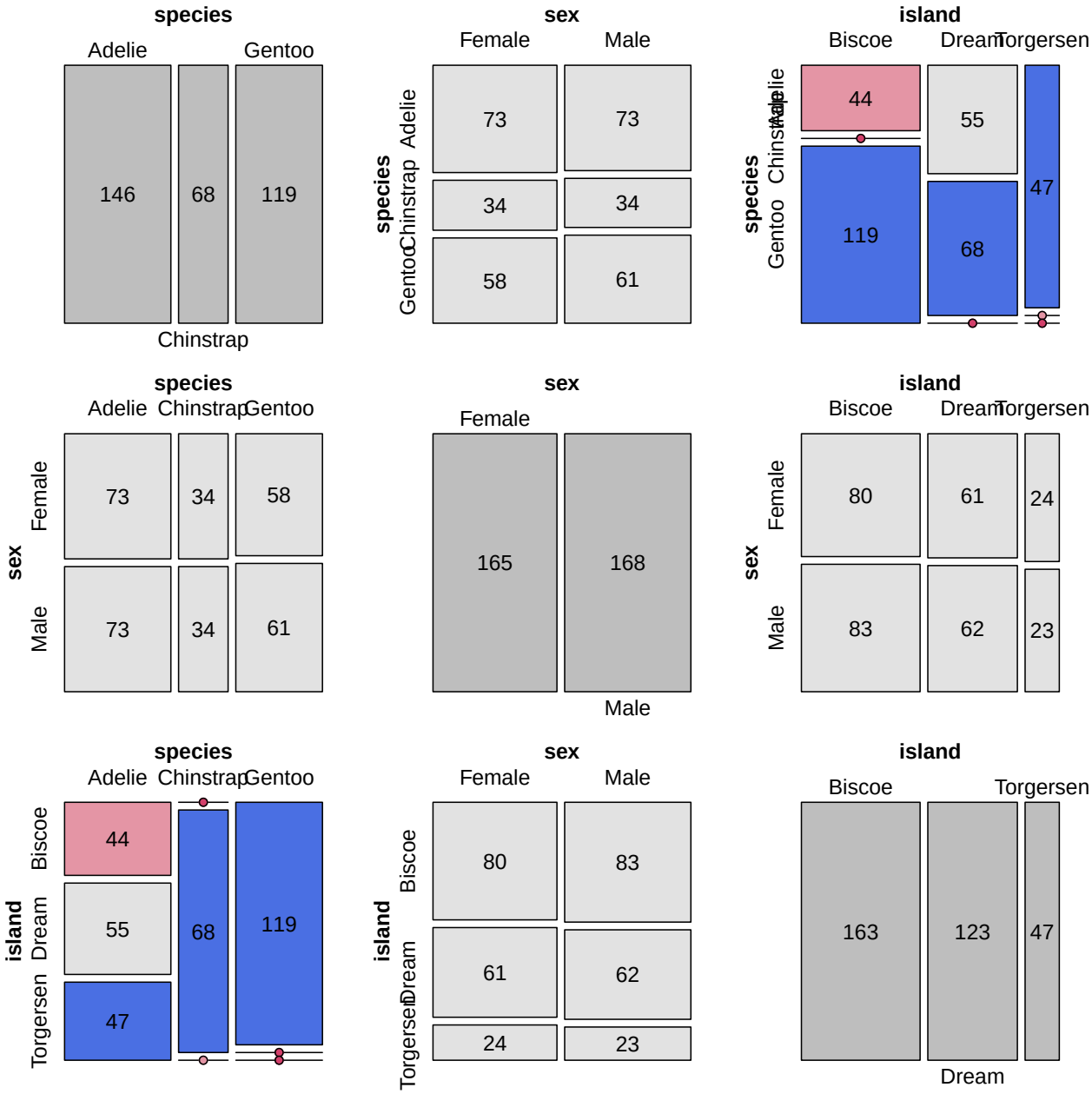


Figure 34: Mosaic pairs plot for the combinations of species, sex and island. Diagnonal plots show the marginal frequency of each variable by the width of each rectangle. Off-diagonal mosaic plots subdivide by the conditional frequency of the second variable, shown numerically in the tiles.

The shading patterns in cells (1,3) and (3,1) of Figure 34 show what we've seen before in the table of frequencies: The distribution of species varies across island because on each island one or more species did not occur. Row 2 and column 2 show that sex is nearly exactly proportional among species and islands,

indicating independence, sex $\perp$ {species, island}. More importantly, mosaic pairs plots can show, at a glance, all (bivariate) associations among multivariate categorical variables.

The next step, by John Emerson and others (Emerson et al., 2013) was to recognize that combinations of continuous and discrete, categorical variables could be plotted in different ways.

- Two continuous variables can be shown as a standard scatterplot of points and/or bivariate density contours, or simply by numeric summaries such as a correlation value;
- A pair of one continuous and one categorical variable can be shown as side-by-side boxplots or violin plots, histograms or density plots;
- Two categorical variables could be shown in a mosaic plot or by grouped bar plots.

In the **ggplot2** framework, these displays are implemented using the `ggpairs()` function from the **GGally** package (Schloerke et al., 2023). This allows different plot types to be shown in the lower and upper triangles and in the diagonal cells of the plot matrix. As well, aesthetics such as color and shape can be used within the plots to distinguish groups directly. As illustrated below, you can define custom functions to control exactly what is plotted in any panel.

The basic, default plot shows scatterplots for pairs of continuous variables in the lower triangle and the values of correlations in the upper triangle. A combination of a discrete and continuous variables is plotted as histograms in the lower triangle and boxplots in the upper triangle. Figure 35 includes `sex` to illustrate the combinations.

```
ggpairs(peng, columns=c(3:6, 7),
        aes(color=species, alpha=0.5),
        progress = FALSE) +
  theme_penguins() +
  theme(axis.text.x = element_text(angle = -45))
```

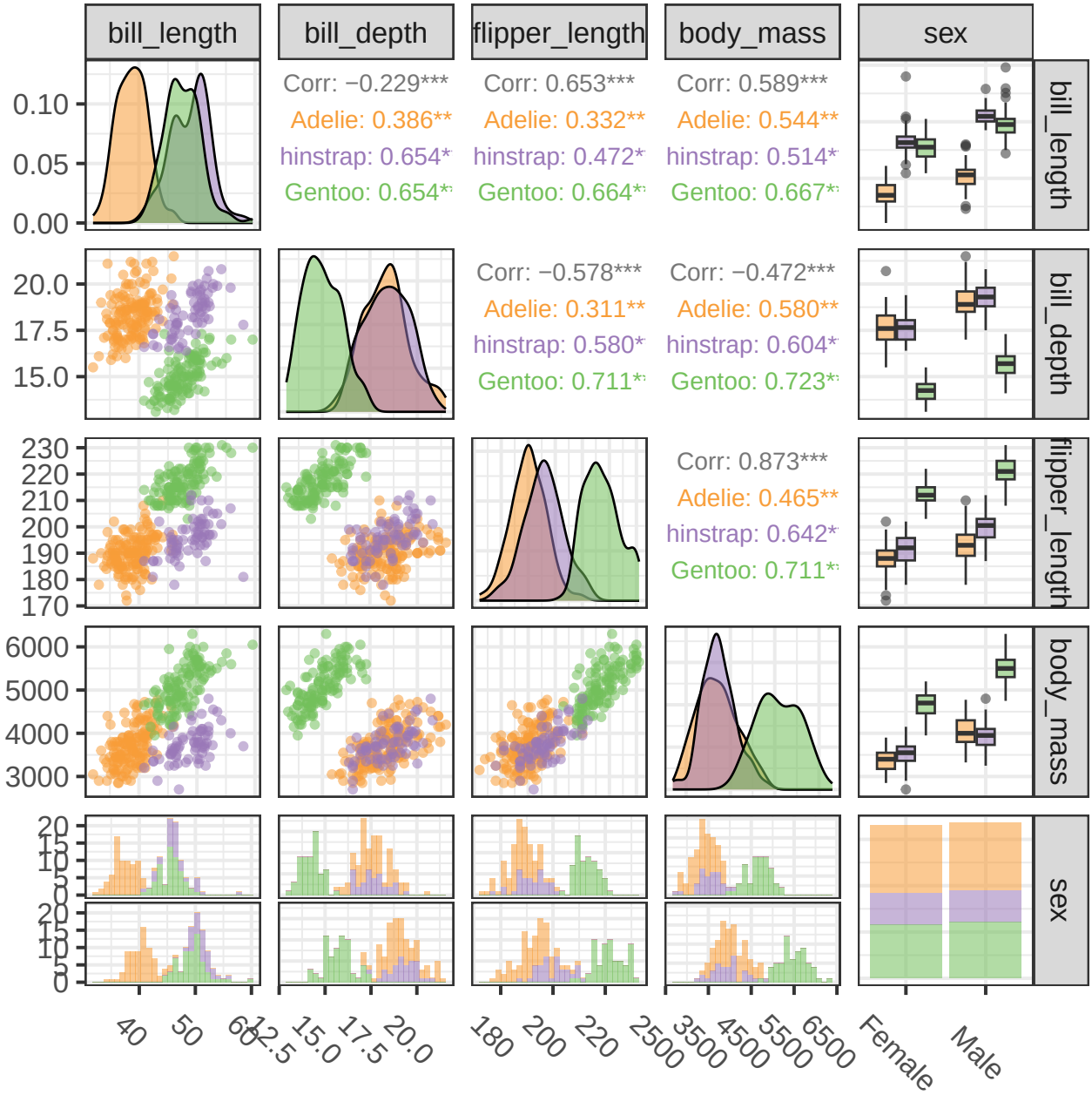


Figure 35: Basic `ggpairs()` plot of penguin size variables and sex, stratified by species.

To my eye, printing the values of correlations in the upper triangle is often a waste of graphic space. But this example shows something peculiar and interesting if you look closely: In all pairs among the penguin size measurements, there are positive correlations within each species, as we can see in Figure 30. Yet, in three of these panels, the overall correlation ignoring species is negative. For example, the overall correlation between bill depth and flipper length is $r = -0.579$ in row 2, column 3; the scatterplot in the diagonally opposite cell, row 3, column 2 shows the data. These cases, of differing signs for an overall correlation, ignoring a group variable and the within group correlations are examples of **Simpson's Paradox**, explored later in Chapter XX.

The last row and column, for `sex` in Figure 35, provides an initial glance at the issue of sex differences among penguin species that motivated the collection of these data. We can go further by also examining differences

among species and island, but first we need to understand how to display exactly what we want for each pairwise plot.

`ggpairs()` is extremely general in that for each of the `lower`, `upper` and `diag` sections you can assign any of a large number of built-in functions (of the form `ggally_NAME`), or your own custom function for what is plotted, depending on the types of variables in each plot.

- `continuous`: both X and Y are continuous variables, supply this as the `NAME` part of a `ggally_NAME()` function or the name of a custom function;
- `combo`: one X of and Y variable is discrete while the other is continuous, using the same convention;
- `discrete`: both X and Y are discrete variables.

The defaults, which were used in Figure 35, are:

```
upper = list(continuous = "cor",          # correlation values
             combo = "box_no_facet",      # boxplots
             discrete = "count")          # rectangles ~ count
lower = list(continuous = "points",       # just data points
             combo = "facethist",         # faceted histograms
             discrete = "facetbar")       # faceted bar plots
diag  = list(continuous = "densityDiag",  # density plots
             discrete = "barDiag")        # bar plots
```

Thus, `ggpairs()` uses `ggally_cor()` to print the correlation values for pairs of continuous variables in the upper triangle, and uses `ggally_points()` to plot scatterplots of points in the lower portion. The diagonal panels as shown as density plots (`ggally_densityDiag()`) for continuous variables but as bar plots (`ggally_barDiag()`) for discrete factors.

See the vignette, ggally_plots for an illustrated list of available high-level plots. For our purpose here, which is to illustrate enhanced displays, note that for scatterplots of continuous variables, there are two functions which plot the points and also add a smoother, `_lm` or `_loess`.

```
ls(getNamespace("GGally")) |> stringr::str_subset("^ggally_smooth_")
#> [1] "ggally_smooth_lm"    "ggally_smooth_loess"
```

A customized display for scatterplots of continuous variables can be any function that takes `data` and `mapping` arguments and returns a `"ggplot"` object. The `mapping` argument supplies the aesthetics, e.g., `aes(color=species, alpha=0.5)`, but only if you wish to override what is supplied in the `ggpairs()` call.

Here is a function, `my_panel()` that plots the data points, regression line and loess smooth:

```
my_panel <- function(data, mapping, ...){
  p <- ggplot(data = data, mapping = mapping) +
    geom_point() +
    geom_smooth(method=lm, formula = y ~ x, se = FALSE, ...) +
    geom_smooth(method=loess, formula = y ~ x, se = FALSE, ...)
  p
}
```

For this example, I want only simple summaries of for the scatterplots, so I don't want to plot the data points, but do want to add the regression line and a data ellipse.

```
my_panel1 <- function(data, mapping, ...){
  p <- ggplot(data = data, mapping = mapping) +
```

```
    geom_smooth(method=lm, formula = y ~ x, se = FALSE, ...) +
    stat_ellipse(geom = "polygon", level = 0.68, ...)
  p
}
```

Then, to show what can be done, Figure 36 uses `my_panel1()` for the scatterplots in the 4 x 4 block of plots in the upper left. The combination of the continuous body size measures and the discrete factors `species`, `island` and `sex` are shown in upper triangle by boxplots but by faceted histograms in the lower portion. The factors are shown as rectangles with area proportional to count (poor-man's mosaic plots) above the diagonal and as faceted bar plots below.

```
ggpairs(peng, columns=c(3:6, 1, 2, 7),
        mapping = aes(color=species, fill = species, alpha=0.2),
        lower = list(continuous = my_panel1),
        upper = list(continuous = my_panel1),
        progress = FALSE) +
  theme_penguins() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank()) +
  theme(axis.text.x = element_text(angle = -45))
```

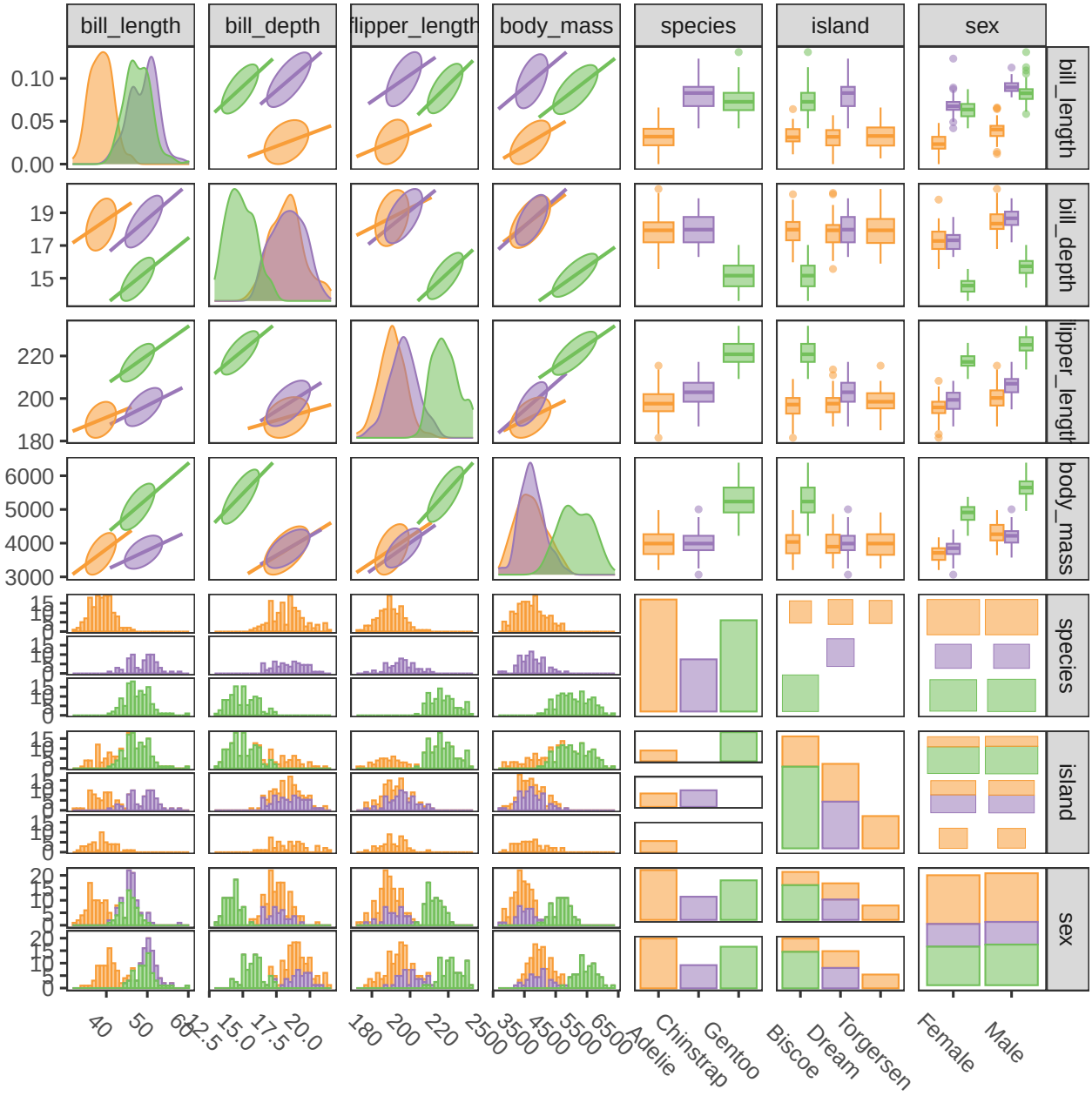


Figure 36: Customized `ggpairs()` plot of penguin size variables, together with species, island and sex.

There is certainly a lot going on in Figure 36, but it does show a high-level overview of all the variables (except `year`) in the penguins dataset.

0.4 Parallel coordinate plots

As we have seen above, scatterplot matrices and generalized pairs plots extend data visualization to multivariate data, but these variables share one 2D space, so resolution decreases as the number of variable increase. You need a very large screen or sheet of paper to see more than, say 5-6 variables with any clarity.

Parallel coordinate plots are an attractive alternative, with which we can visualize an arbitrary number of variables to get a visual summary of a potentially high-dimensional dataset, and perhaps recognize outliers and clusters in the data in a different way. In these plots, each variable is shown on a separate, parallel axis. A multivariate observation is then plotted by connecting their respective values on each axis with lines across all the axes.

The geometry of parallel coordinates is interesting, because what is a point in n-dimensional (Euclidean) *data* space becomes a line in the *projective* parallel coordinate space with n axes, and vice-versa: lines in parallel coordinate space correspond to points in data space. Thus, a collection of points in data space map to lines that intersect in a point in projective space. What this does is to map n-dimensional relations into 2D patterns we can see in a parallel coordinates plot.

i History Corner

Those who don't know history are doomed to plagarize it —The author

The theory of projective geometry originated with the French mathematician Maurice d'Ocagne (1885) who sought a way to provide graphic calculation of mathematical functions with alignment diagrams or *nomograms* using parallel axes with different scales. A three-variable equation, for example, could be solved using three parallel axes, where known values could be marked on their scales, a line drawn between them, and an unknown read on its scale at the point where the line intersects that scale.

Henry Gannet (1880), in work preceding the *Statistical Atlas of the United States* for the 1890 Census (Gannett, 1898), is widely credited with being the first to use parallel coordinates plots to show data, in his case, to show the rank ordering of US states by 10 measures including population, occupations, wealth, manufacturing, agriculture and so on.

However, both d'Ocagne and Gannet were far preceded in this by Andre-Michel Guerry (1833) who used this method to show how the rank order of various crimes changed with age of the accused. See Friendly (2022), Figure 7 for his version and for an appreciation of the remarkable contributions of this amateur statistician to the history of data visualization.

The use of parallel coordinates for display of multidimensional data was rediscovered by Alfred Inselberg (1985) and extended by Edward Wegman (1990), neither of whom recognized the earlier history. Somewhat earlier, David Andrews (1972) proposed mapping multivariate observations to smooth Fourrier functions composed of alternating sin() and cos() terms. And in my book, *SAS System for Statistical Graphics* (Friendly, 1991), I implemented what I called *profile plots* without knowing their earlier history as parallel coordinate plots.

Parallel coordinate plots present a challenge for graphic developers, in that they require a different way to think about plot construction for multiple variables, which can be quantitative, as in the original idea, or categorical factors, all to be shown along parallel axes.

Here, I use the **ggpcp** package (Hofmann et al., 2022), best described in VanderPlas et al. (2023), who also review the modern history. This takes some getting used to, because they develop `pcp_*()` extensions of the `ggplot2` grammar of graphics framework to allow:

- `pcp_select()`: selections of the variables to be plotted and their horizontal order on parallel axes,
- `pcp_scale()`: methods for scaling of the variables to each axis,
- `pcp_arrange()`: methods for breaking ties in factor variables to space them out.

Then, it provides `geom_pcp_*()` functions to control the display of axes with appropriate aesthetics, labels for categorical factors and so forth. **?@fig-peng-ggpcp1** illustrates this type of display, using sex and species in addition to the quantitative variables for the penguin data.

```
peng |>
  pcp_select(bill_length:body_mass, sex, species) |>
  pcp_scale(method = "uniminmax") |>
```

```
  pcp_arrange() |>
  ggplot(aes_pcp()) +
  geom_pcp_axes() +
  geom_pcp(aes(colour = species), alpha = 0.8, overplot = "none") +
  geom_pcp_labels() +
  scale_colour_manual(values = peng.colors()) +
  labs(x = "", y = "") +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(),
        axis.ticks.y = element_blank(), legend.position = "none")
```

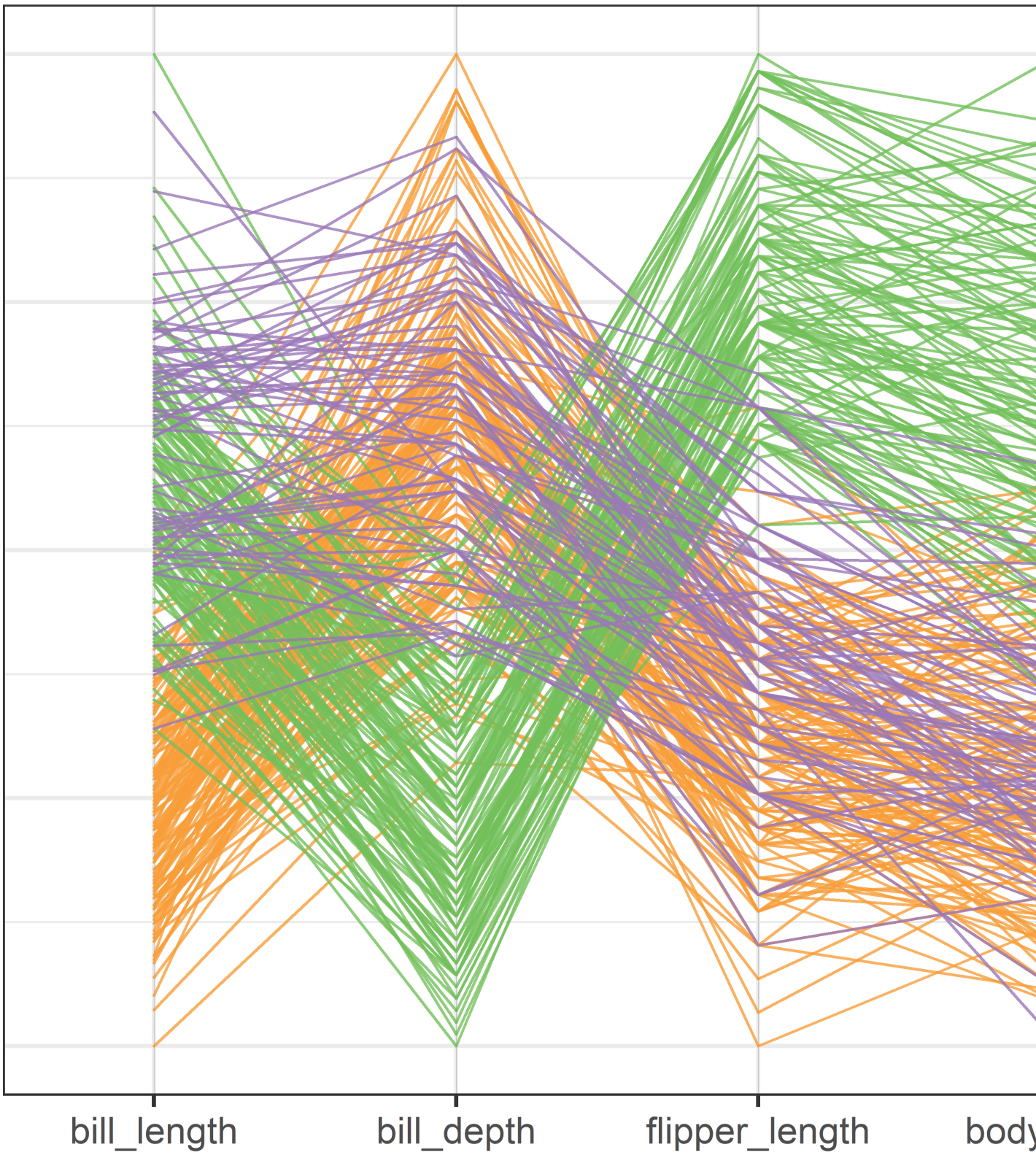
Rearranging the order of variables and the ordering of factor levels can make a difference in what we can see in such plots. For a simple example (following VanderPlas et al. (2023)), we reorder the levels of species and islands to make it clearer which species occur on each island.

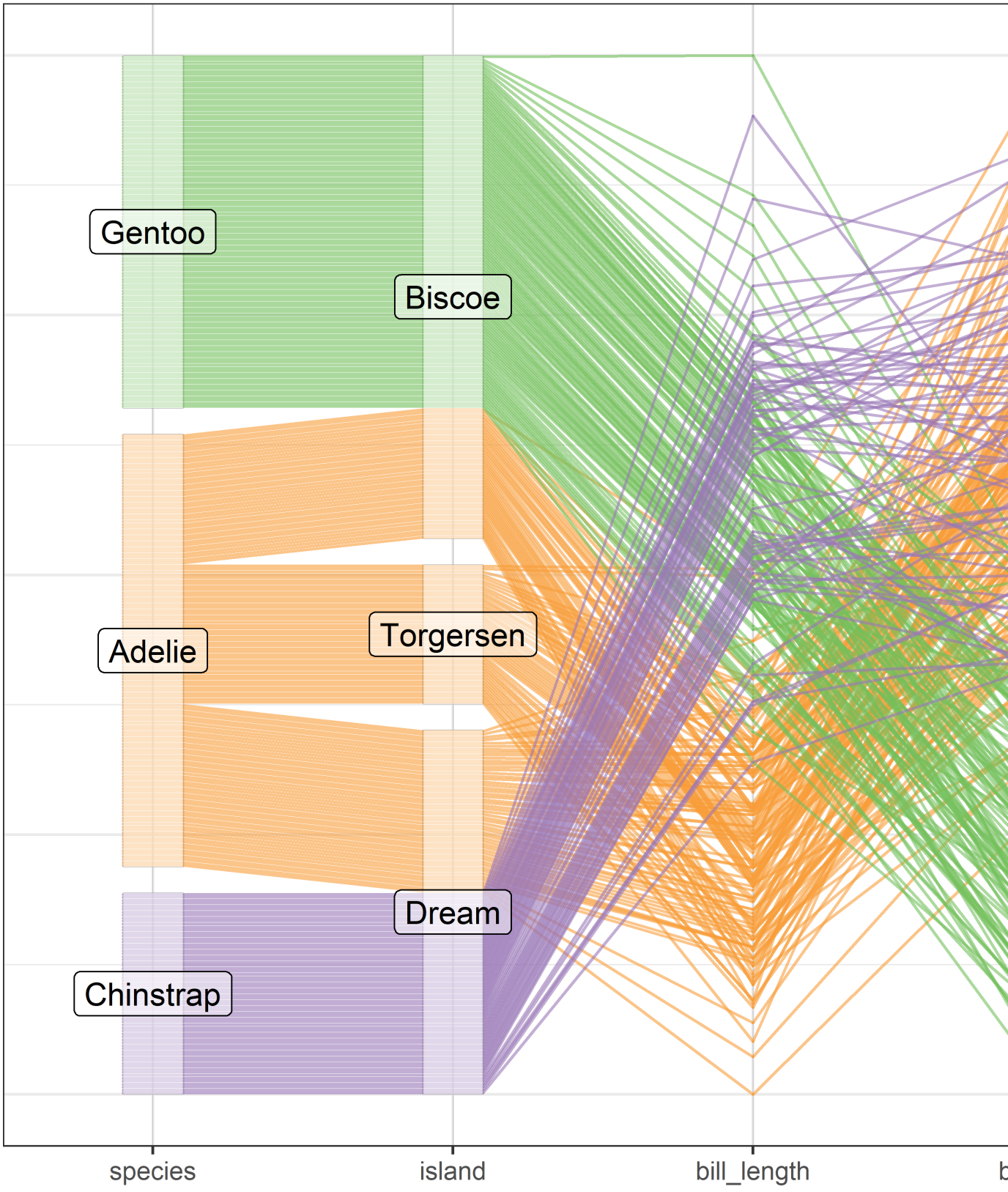
```
peng1 <- peng |>
  mutate(species = factor(species, levels = c("Chinstrap", "Adelie", "Gentoo"))) |>
  mutate(island = factor(island, levels = c("Dream", "Torgersen", "Biscoe")))

peng1 |>
  pcp_select(species, island, bill_length:body_mass) |>
  pcp_scale() |>
  pcp_arrange(method = "from-left") |>
  ggplot(aes_pcp()) +
  geom_pcp_axes() +
  geom_pcp(aes(colour = species), alpha = 0.6, overplot = "none") +
  geom_pcp_boxes(fill = "white", alpha = 0.5) +
  geom_pcp_labels() +
  scale_colour_manual(values = peng.colors()[c(2,1,3)]) +
  theme_bw() +
  labs(x = "", y = "") +
  theme(axis.text.y = element_blank(), axis.ticks.y = element_blank(),
        legend.position = "none")
```

This plot emphasizes the relation between penguin species and the island where they were observed and then shows the values of the quantitative body size measurements.

Packages used here: 21 packages used here: base, car, carData, corrgram, corrplot, datasets, dplyr, GGally, ggdensity, ggpcp, ggplot2, graphics, grDevices, grid, knitr, methods, patchwork, stats, tidyr, utils, vcd


bill_length
bill_depth
flipper_length
body


Gentoo
Biscoe
Adelie
Torgersen
Dream
Chinstrap
species
island
bill_length

0

PCA and Biplots

0.5 *Flatland* and *Spaceland*

> It is high time that I should pass from these brief and discursive notes about Flatland to the central event of this book, my initiation into the mysteries of Space. THAT is my subject; all that has gone before is merely preface — Edwin Abbott, *Flatland*, p. 57.

There was a cloud in the sky above *Flatland* one day. But it was a huge, multidimensional cloud of sparkly points that might contain some important message, perhaps like the hidden EUREKA (**?@fig-pollen-eureka-fig**), or perhaps forecasting the upcoming harvest, if only Flatlanders could appreciate it.

A leading citizen, A SQUARE, who had traveled once to Spaceland and therefore had an inkling of its majesty beyond the simple world of his life in the plane looked at that cloud and had a brilliant thought, an OMG moment:

> "Oh, can I, in my imagination, rotate that cloud and squeeze its' juice so that it rains down on Flatland with greatest joy?"

As it happened, our Square friend, although he could never really *see* in three dimensions, he could now at least *think* of a world described by **height** as well as breadth and width, and think of the **shadow** cast by a cloud as something mutable, changing size and shape depending on its' orientation over Flatland.

And what a world it was, inhabited by Pryamids, Cubes and wondrous creatures called Polyhedrons with many *C*orners, *F*aces and *E*dges. Not only that, but all those Polyhedra were forced in Spaceland to obey a magic formula: $C + F - E = 2$.[3] How cool was that!

Indeed, there were even exalted Spheres, having so many faces that its surface became as smooth as a baby's bottom with no need for pointed corners or edges, just as Circles were the smoothest occupants of his world with far too many sides to count. It was his dream of a Sphere passing through Flatland (Figure 1) that first awakened him to a third dimension.

He also marveled at Ellipsoids, as smooth as Spheres, but in Spaceland having three natural axes of different extent and capable of being appearing fatter or slimmer when rotated from different views. An Ellipsoid had magical properties: it could appear as so thin in one or more dimensions that it became a simple 2D ellipse, or a 1D line, or even a 0D point (Friendly et al., 2013).

All of these now arose in Square's richer 3D imagination. And, all of this came from just one more dimension than his life in Flatland.

[3] This is Euler's (1758) formula, which states that any convex polyheron must obey the formula $V + F - E = 2$ where V is the number of vertexes (corners), F is the number of faces and E is the number of edges. For example, a tetrahedron or pyramid has $(V, F, E) = (4, 4, 6)$ and a cube has $(V, F, E) = (8, 6, 12)$. Stated in words, for all solid bodies confined by planes, the sum of the number of vertexes and the number of faces is two less than the number of edges.

0.5.1 Multivariate juicers

Up to now, we have also been living in Flatland. We have been trying to understand data in **data space** of possibly many dimensions, but confined to the 2D plane of a graph window. Scatterplot matrices and parallel coordinate plots provided some relief. The former did so by **projecting** the data into sets of 2D views in the coordinates of data space; the latter did so by providing multiple axes in a 2D space along which we could trace the paths of individual observations.

This chapter is about seeing data in a different projection, a low-dimensional, usually 2D, space that which squeezes out the most juice from multidimensional data for a particular purpose (Figure 37), where what we want to understand can be more easily seen.

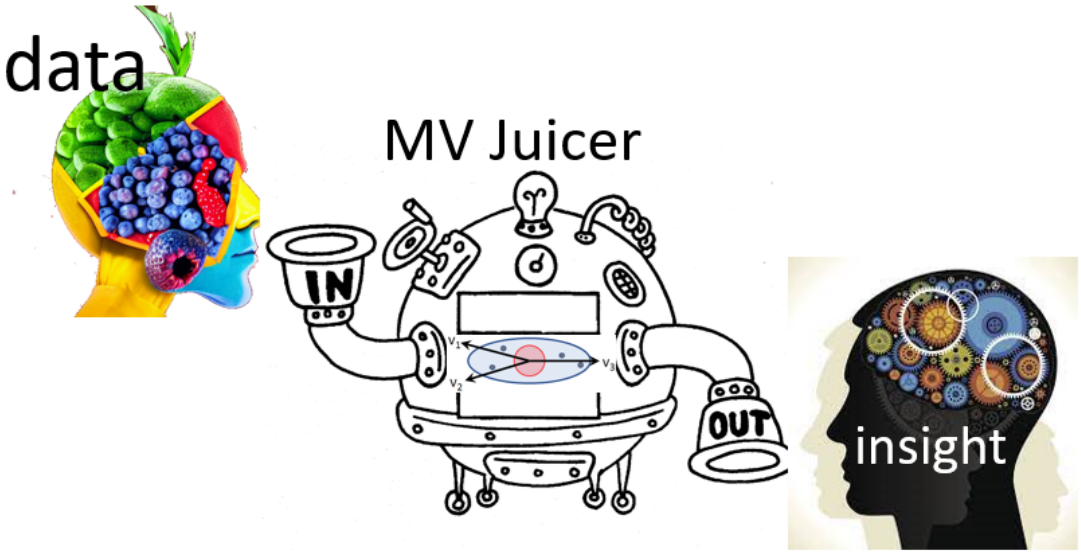


Figure 37: A multivariate juicer takes data from possibly high-dimensional data space and transforms it to a lower-dimenional space in which important effects can be more easily seen.

Here, I concentrate on **principal components analysis** (PCA), whose goal reflects A Square's desire to see that sparkly cloud of points in nD space in the plane showing the greatest variation (squeezing the most juice) among all other possible views. This appealed to his sense of geometry, but left him wondering how the variables in that high-D cloud were related to the dimensions he could see in a best-fitting plane.

The idea of a **biplot**, showing the data points in the plane, together with thick pointed arrows—variable vectors— in one view is the other topic explained in this chapter (Section 0.7). The biplot is the simplest example of a multivariate juicer. The essential idea is to project the cloud of data points in n dimensions into the 2D space of principal components and simultaneously show how the original variables relate to this space. For exploratory analysis to get an initial, incisive view of a multivariate dataset, a biplot is often my first choice.

> **i** Looking ahead
>
> I'm using the term *multivariate juicer* here to refer the wider class of **dimension reduction** techniques, used for various purposes in data analysis and visualization. Principal components analysis is the simplest example and illustrates the general ideas.
> The key point is that these methods are designed to transform the data into a low-dimensional space for a particular goal or purpose. In PCA, the goal is to extract the greatest amount of total variability in the data. In the context of univariate multiple regression, the goal is often to reduce the number

of predictors necessary to account for an outcome variable, called *feature extraction* in the machine learning literature.
When the goal is to best distinguish among groups **discriminant analysis** finds uncorrelated weighted sums of predictors on which the means of groups are most widely separated in a reduced space of hopefully fewer dimensions.
The methods I cover in this book are all linear methods, but there is also a wide variety of non-linear dimension reduction techniques ...

Packages

In this chapter I use the following packages. Load them now:

```
library(ggplot2)
library(dplyr)
library(tidyr)
library(patchwork)
library(ggbiplot)
library(FactoMineR)
library(factoextra)
```

0.6 Principal components analysis

When Francis Galton (1886) first discovered the idea of regression toward the mean and presented his famous diagram (Figure 15), he had little thought that he had provided a window to a higher-dimensional world, beyond what even A Square could imagine. His friend, Karl Pearson (1896) took that idea and developed it into a theory of regression and a measure of correlation that would bear his name, Pearson's r.

But then Pearson (1901) had a further inspiration, akin to that of A Square. If he also had a cloud of sparkly points in $2, 3, 4, ..., p$ dimensions, could he find a point ($0D$), or line ($1D$), or plane ($2D$), or even a hyperplane (nD) that best summarized — squeezed out the most juice—from multivariate data? This was the first trully multivariate problem in the history of statistics (Friendly & Wainer, 2021, p. 186).

The best $0D$ point was easy— it was simply the centroid, the means of each of the variables in the data, $(\bar{x}_1, \bar{x}_2, ..., \bar{x}_p)$, because that was "closest" to the data in the sense of minimizing the sum of squared differences, $\Sigma_i \Sigma_j (x_{ij} - \bar{x}_j)^2$. In higher dimensions, his solution was also an application of the method of least squares, but he argued it geometrically and visually as shown in Figure 38.

For a $1D$ summary, the line of best fit to the points $P_1, P_2, \dots P_n$ is the line that goes through the centroid and made the average squared length of the *perpendicular* segments from those points to a line as small as possible. This was different from the case in linear regression, for fitting y from x, where the average squared length of the *vertical* segments, $\Sigma_i (y_i - \hat{y}_i)^2$ was minimized by least squares.

He went on to prove the visual insights from simple smoothing of Galton (1886) (shown in Figure 15) regarding the regression lines of `y ~ x` and `x ~ y`. More importantly, he proved that the cloud of points is captured, for the purpose of finding a best line, plane or hyperplane, by the ellipsoid that encloses it, as seen in his diagram, Figure 39. The major axis of the 2D ellipse is the line of best fit, along which the data points have the smallest average squared distance from the line. The axis at right angles to that—the minor axis— is labeled "line of worst fit" with the largest average squared distance.

Even more importantly— and this is the basis for what we call **principal components analysis** (PCA)— he recognized that the two orthogonal axes of the ellipse gave new coordinates for the data which were uncorrelated, whatever the correlation of x and y.

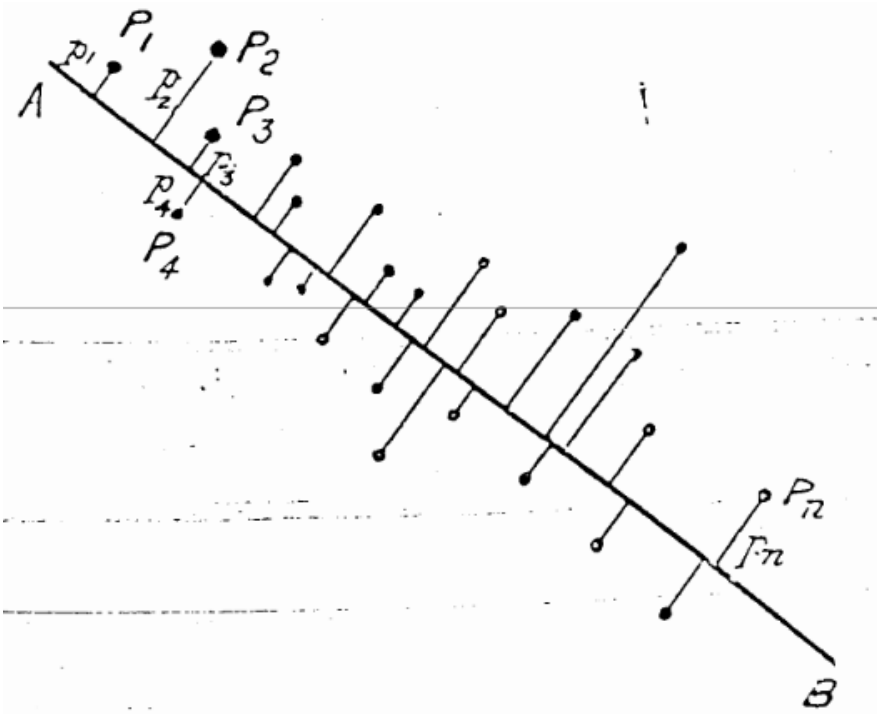


Figure 38: Karl Pearson's (1901) geometric, visual argument for finding the line or plane of closest fit to a collection of points, P1, P2, P3, ...

> Physically, the axes of the correlation type-ellipse are the directions of independent and uncorrelated variation. — Pearson (1901), p. 566.

It was but a small step to recognize that for two variables, x and y:

- the line of best fit, the major axis (PC1) had the greatest variance of points projected onto it;
- the line of worst fit, the minor axis (PC2), had the least variance;
- these could be seen as a rotation of the data space of (x, y) to a new space (PC1, PC2) with uncorrelated variables;
- the total variation of the points in data space, $\text{Var}(x) + \text{Var}(y)$, being unchanged by rotation, was equally well expressed as the total variation $\text{Var}(PC1) + \text{Var}(PC2)$ of the scores on what are now called the principal component axes.

It would have appealed to Pearson (and also to A Square) to see these observations demonstrated in a 3D video. **?@fig-pca-animation** shows a 3D plot of the variables `Sepal.Length`, `Sepal.Width` and `Petal.Length` in Edgar Anderson's `iris` data, with points colored by species and the 95% data ellipsoid. This is rotated smoothly by interpolation until the first two principal axes, PC1 and PC2 are aligned with the horizontal and vertical dimensions. Because this is a rigid rotation of the cloud of points, the total variability is obviously unchanged.

0.6.1 PCA by springs

Before delving into the mathematics of PCA, it is useful to see how Pearson's problem, and fitting by least squares generally, could be solved in a physical realization.

From elementary statistics, you may be familiar with a physical demonstration that the mean, $\bar{x}$, of a sample is the value for which the sum of deviations, $\Sigma_i (x_i - \bar{x})$ is zero, so the mean can be visualized as the point of balance on a line where those differences $(x_i - \bar{x})$ are placed. Equally well, there is a physical realization of

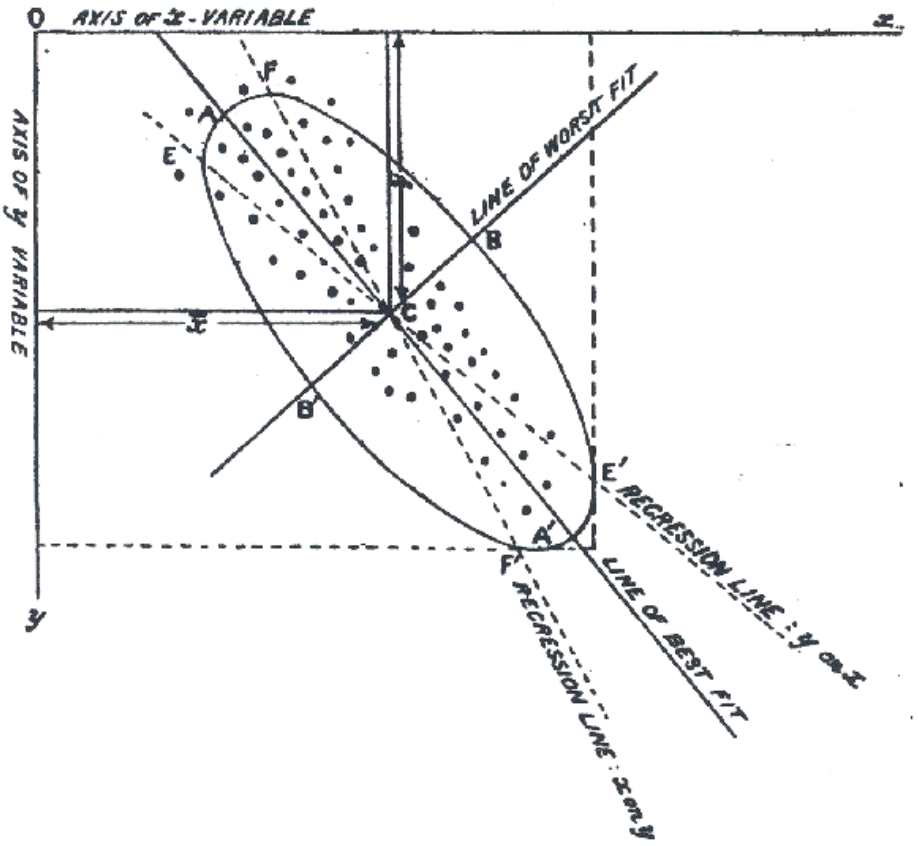


Figure 39: Karl Pearson's diagram showing the elliptical geometry of regression and principal components analysis ... *Source*: Pearson (1901), p. 566.

the mean as the point along an axis where weights connected by springs will minimize the sum of squared differences, because springs with a constant stiffness, k, exert forces proportional to $k(x_i - \bar{x})^2$. That's the reason is useful as a measure of central tendency: it minimizes the average squared error.

In two dimensions, imagine that we have points, (x_i, y_i) and these are attached by springs of equal stiffness k, to a line anchored at the centroid, $(\bar{x}, \bar{y})$ as shown in Figure 40. If we rotate the line to some initial position and release it, the springs will pull the line clockwise or counterclockwise and the line will bounce around until the forces, proportional to the squares of the lengths of the springs, will eventually balance out at the position (shown by the red fixed line segments at the ends). This is the position that minimizes the the sum of squared lengths of the connecting springs, and also minimizes the kinetic energy in the system.

If you look closely at Figure 40 you will see something else: When the line is at its' final position of minimum squared length and energy, the positions of the red points on this line are spread out furthest, i.e., have **maximum** variance. Conversely, when the line line is at right angles to its' final position (shown by the black line at 90^o) the projected points have the smallest possible variance.

Figure 40: Animation of PCA fitted by springs. The blue data points are connected to their projections on the red line by springs perpendicular to that line. From an initial position, the springs pull that line in proportion to their squared distances, until the line finally settles down to the position where the forces are balanced and the minimum is achieved. *Source*: Amoeba, https://bit.ly/46tAicu.

TODO: Simple PCA example using workers data

0.6.2 Mathematics and geometry of PCA

As the ideas of principal components developed, there was a happy marriage of Galton's geometrical intuition and Pearson's mathematical analysis. The best men at the wedding were ellipses and higher-dimensional ellipsoids. The brides maids were eigenvectors, pointing in as many different directions as space would allow, each sized according to their associated eigenvalues. Attending the wedding were the ghosts of uncles, Leonhard Euler, Jean-Louis Lagrange, Augustin-Louis Cauchy and others who had earlier discovered the mathematical properties of ellipses and quadratic forms in relation to problems in physics.

The key idea in the statistical application was that, for a set of variables $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p$, the $p \times p$ covariance matrix $\mathbf{S}$ could be expressed **exactly** as a matrix product involving a matrix $\mathbf{V}$, whose columns are *eigenvectors* ($\mathbf{v}_i$) and a diagonal matrix $\mathbf{\Lambda}$, whose diagonal elements (λ_i) are the corresponding *eigenvalues*.

To explain this, it is helpful to use a bit of matrix math:

$$
\begin{aligned}
\mathbf{S}_{p\times p} &= \mathbf{V}_{p\times p} \quad \mathbf{\Lambda}_{p\times p} \quad \mathbf{V}^T_{p\times p} \\
&= (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_p)
\begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_p \end{pmatrix}
\begin{pmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_p^T \end{pmatrix} \\
&= \lambda_1 \mathbf{v}_1 \mathbf{v}_1^T + \lambda_2 \mathbf{v}_2 \mathbf{v}_2^T + \cdots + \lambda_p \mathbf{v}_p \mathbf{v}_p^T
\end{aligned}
$$

In this equation,

- The last line follows because $\mathbf{\Lambda}$ is a diagonal matrix, so $\mathbf{S}$ is expressed as a sum of outer products of each $\mathbf{v}_i$ with itself.

- The columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{S}$. They are orthogonal and of unit length, so $\mathbf{V}^T\mathbf{V} = \mathbf{I}$ and thus they represent orthogonal (uncorrelated) directions in data space.
- The columns $\mathbf{v}_i$ are the weights applied to the variables to produce the scores on the principal components. For example, the first principal component is the weighted sum:

$$PC1 = v_{11}\mathbf{x}_1 + v_{12}\mathbf{x}_2 + \cdots + v_{1p}\mathbf{x}_p$$

- The eigenvalues, $\lambda_1, \lambda_2, \ldots, \lambda_p$ are the variances of the the components, because $\mathbf{v}_i^T\ \mathbf{S}\ \mathbf{v}_i = \lambda_i$.
- It is usually the case that the variables $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p$ are linearly independent, which means that none of these is an exact linear combination of the others. In this case, all eigenvalues λ_i are positive and the covariance matrix $\mathbf{S}$ is said to have **rank** p.
- Here is the key fact: If the eigenvalues are arranged in order, so that $\lambda_1 > \lambda_2 > \cdots > \lambda_p$, then the first d components give a d-dimensional approximation to $\mathbf{S}$, which accounts for $\Sigma_i^d \lambda_i / \Sigma_i^p \lambda_i$ of the total variance.

For the case of two variables, $\mathbf{x}_1$ and $\mathbf{x}_2$ Figure 41 shows the transformation from data space to component space. The eigenvectors, $\mathbf{v}_1, \mathbf{v}_2$ are the major and minor axes of the data ellipse, whose lengths are the square roots $\sqrt{\lambda_1}, \sqrt{\lambda_2}$ of the eigenvalues.

0.6.3 Finding principal components

In R, principal components analysis is most easily carried out using `stats::prcomp()` or `stats::princomp()` or similar functions in other packages such as `FactomineR::PCA()`. The **FactoMineR** package (Husson et al., 2017, 2023) has extensive capabilities for exploratory analysis of multivariate data (PCA, correspondence analysis, cluster analysis).

A particular strength of FactoMineR for PCA is that it allows the inclusion of *supplementary variables* (which

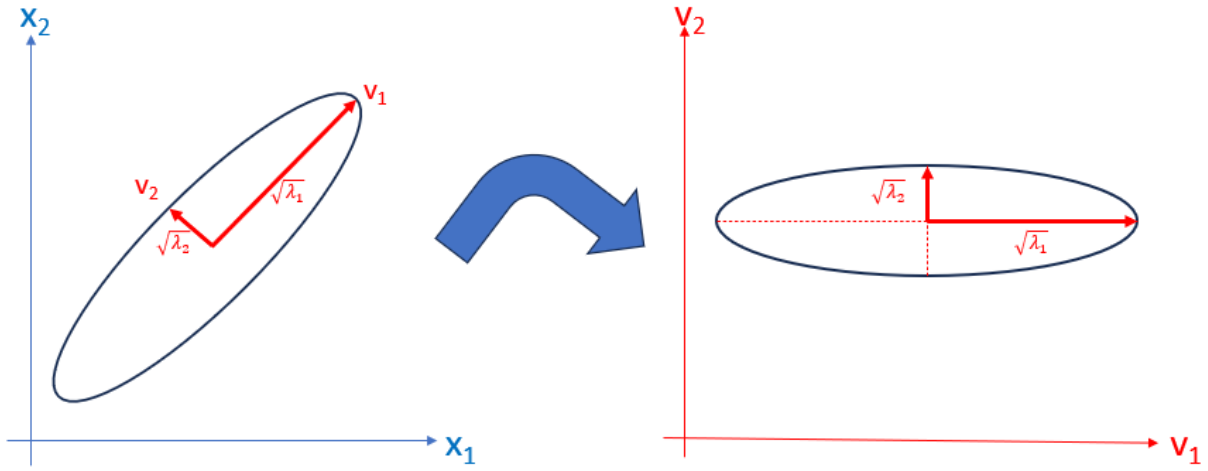


Figure 41: Geometry of PCA as a rotation from data space to principal component space, defined by the eigenvectors v1 and v2 of a covariance matrix

can be categorical or quantitative) and *supplementary points* for individuals. These are not used in the analysis, but are projected into the plots to facilitate interpretation. For example, in the analysis of the `crime` data described below, it would be useful to have measures of other characteristics of the U.S. states, such as poverty and average level of education (Section 0.7.5).

Unfortunately, although all of these functions perform similar calculations, the options for analysis and the details of the result they return differ.

The important options for analysis include:

- whether or not the data variables are **centered**, to a mean of $\bar{x}_j = 0$
- whether or not the data variables are **scaled**, to a variance of $\text{Var}(x_j) = 1$.

It nearly always makes sense to center the variables. The choice of scaling determines whether the correlation matrix is analyzed, so that each variable contributes equally to the total variance that is to be accounted for versus analysis of the covariance matrix, where each variable contributes its own variance to the total. Analysis of the covariance matrix makes little sense when the variables are measured on different scales.[4]

Example: Crime data

The dataset `crime`, analysed in Section 0.2.2, showed all positive correlations among the rates of various crimes in the corrgram, Figure 33. What can we see from a principal components analysis? Is it possible that a few dimensions can account for most of the juice in this data?

In this example, you can easily find the PCA solution using `prcomp()` in a single line in base-R. You need to specify the numeric variables to analyze by their columns in the data frame. The most important option here is `scale. = TRUE` ...

```
data(crime, package = "ggbiplot")
crime.pca <- prcomp(crime[, 2:8], scale. = TRUE)
```

The tidy equivalent is more verbose, but also more expressive about what is being done. It selects the

[4] For example, if two variables in the analysis are height and weight, changing the unit of height from inches to centimeters would multiply its' variance by 2.54^2; changing weight from pounds to kilograms would divide its' variance by 2.2^2.

variables to analyze by a function, `is.numeric()` applied to each of the columns and feeds the result to `prcomp()`.

```
crime.pca <-
  crime |>
  dplyr::select(where(is.numeric)) |>
  prcomp(scale. = TRUE)
```

As is typical with models in R, the result, `crime.pca` of `prcomp()` is an object of class `"prcomp"`, a list of components, and there are a variety of methods for `"prcomp"` objects. Among the simplest is `summary()`, which gives the contributions of each component to the total variance in the dataset.

```
summary(crime.pca) |> print(digits=2)
#> Importance of components:
#>                          PC1  PC2  PC3   PC4   PC5   PC6   PC7
#> Standard deviation     2.03 1.11 0.85 0.563 0.508 0.471 0.352
#> Proportion of Variance 0.59 0.18 0.10 0.045 0.037 0.032 0.018
#> Cumulative Proportion  0.59 0.76 0.87 0.914 0.951 0.982 1.000
```

The object, `crime.pca` returned by `prcomp()` is a list of the following the following elements:

```
names(crime.pca)
#> [1] "sdev"     "rotation" "center"   "scale"    "x"
```

Of these, for n observations and p variables,

- `sdev` is the length p vector of the standard deviations of the principal components (i.e., the square roots $\sqrt{\lambda_i}$ of the eigenvalues of the covariance/correlation matrix). When the variables are standardized, the sum of squares of the eigenvalues is equal to p.
- `rotation` is the $p \times p$ matrix of weights or **loadings** of the variables on the components; the columns are the eigenvectors of the covariance or correlation matrix of the data;
- `x` is the $n \times p$ matrix of **scores** for the observations on the components, the result of multiplying (rotating) the data matrix by the loadings. These are uncorrelated, so `cov(x)` is a $p \times p$ diagonal matrix whose diagonal elements are the eigenvalues $\lambda_i =$ `sdev^2`.

0.6.4 Visualizing variance proportions: screeplots

For a high-D dataset, such as the crime data in seven dimensions, a natural question is how much of the variation in the data can be captured in 1D, 2D, 3D, ... summaries and views. This is answered by considering the proportions of variance accounted by each of the dimensions, or their cumulative values. The components returned by various PCA methods have (confusingly) different names, so `broom::tidy()` provides methods to unify extraction of these values.

```
(crime.eig <- crime.pca |>
  broom::tidy(matrix = "eigenvalues"))
#> # A tibble: 7 x 4
#>      PC std.dev percent cumulative
#>   <dbl>   <dbl>   <dbl>      <dbl>
#> 1     1   2.03   0.588       0.588
#> 2     2   1.11   0.177       0.765
#> 3     3   0.852  0.104       0.868
#> 4     4   0.563  0.0452      0.914
```

```
#> 5     5   0.508  0.0368      0.951
#> 6     6   0.471  0.0317      0.982
#> 7     7   0.352  0.0177      1
```

Then, a simple visualization is a plot of the proportion of variance for each component (or cumulative proportion) against the component number, usually called a **screeplot**. The idea, introduced by Cattell (1966), is that after the largest, dominant components, the remainder should resemble the rubble, or scree formed by rocks falling from a cliff.

From this plot, imagine drawing a straight line through the plotted eigenvalues, starting with the largest one. The typical rough guidance is that the last point to fall on this line represents the last component to extract, the idea being that beyond this, the amount of additional variance explained is non-meaningful. Another rule of thumb is to choose the number of components to extract a desired proportion of total variance, usually in the range of 80 - 90%.

`stats::plot(crime.pca)` would give a bar plot of the variances of the components, however `ggbiplot::ggscreeplot()` gives nicer and more flexible displays as shown in Figure 42.

```
p1 <- ggscreeplot(crime.pca) +
  stat_smooth(data = crime.eig |> filter(PC>=4),
              aes(x=PC, y=percent), method = "lm",
              se = FALSE,
              fullrange = TRUE) +
  theme_bw(base_size = 14)

p2 <- ggscreeplot(crime.pca, type = "cev") +
  geom_hline(yintercept = c(0.8, 0.9), color = "blue") +
  theme_bw(base_size = 14)

p1 + p2
```

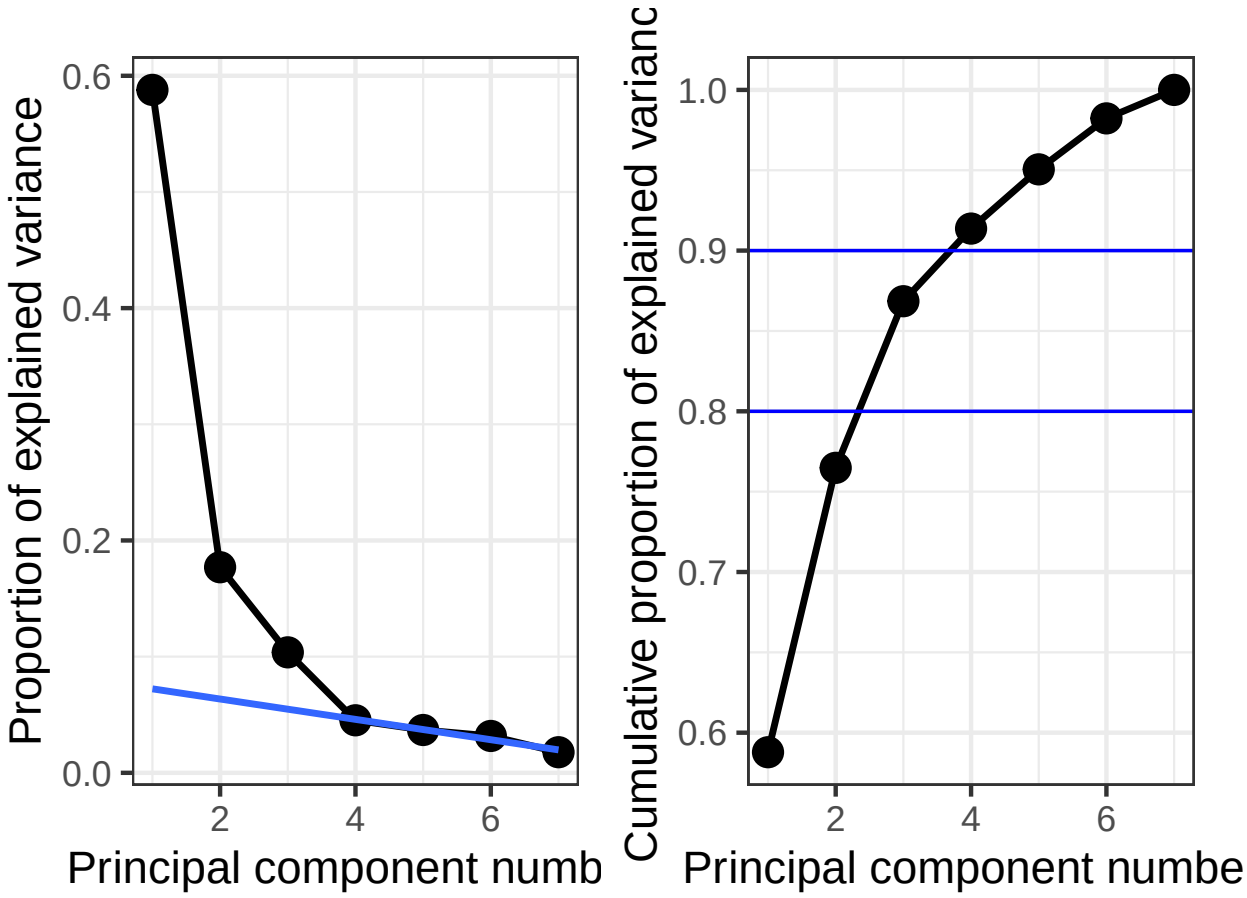


Figure 42: Screeplots for the PCA of the crime data. The left panel shows the traditional version, plotting variance proportions against component number, with linear guideline for the scree rule of thumb. The right panel plots cumulative proportions, showing cutoffs of 80%, 90%.

From this we might conclude that four components are necessary to satisfy the scree criterion or to account for 90% of the total variation in these crime statistics. However two components, giving 76.5%, might be enough juice to tell a reasonable story.

0.6.5 Visualizing PCA scores and variable vectors

To see and attempt to understand PCA results, it is useful to plot both the scores for the observations on a few of the largest components and also the loadings or variable vectors that give the weights for the variables in determining the principal components.

In Section 0.7 I discuss the biplot technique that plots both in a single display. However, I do this directly here, using tidy processing to explain what is going on in PCA and in these graphical displays.

Scores

The (uncorrelated) principal component scores can be extracted as `crime.pca$x` or using `purrr::pluck("x")`. As noted above, these are uncorrelated and have variances equal to the eigenvalues of the correlation matrix.

```
scores <- crime.pca |> purrr::pluck("x")
cov(scores) |> zapsmall()
```

```
#>       PC1  PC2  PC3  PC4  PC5  PC6  PC7
#> PC1 4.11 0.00 0.00 0.00 0.00 0.00 0.00
#> PC2 0.00 1.24 0.00 0.00 0.00 0.00 0.00
#> PC3 0.00 0.00 0.73 0.00 0.00 0.00 0.00
#> PC4 0.00 0.00 0.00 0.32 0.00 0.00 0.00
#> PC5 0.00 0.00 0.00 0.00 0.26 0.00 0.00
#> PC6 0.00 0.00 0.00 0.00 0.00 0.22 0.00
#> PC7 0.00 0.00 0.00 0.00 0.00 0.00 0.12
```

For plotting, it is more convenient to use `broom::augment()` which extracts the scores (named `.fittedPC*`) and appends these to the variables in the dataset.

```
crime.pca |>
  broom::augment(crime) |> head()
#> # A tibble: 6 x 18
#>   .rownames state      murder  rape robbery assault burglary larceny
#>   <chr>     <chr>       <dbl> <dbl>   <dbl>   <dbl>    <dbl>   <dbl>
#> 1 1         Alabama      14.2  25.2    96.8    278.    1136.   1882.
#> 2 2         Alaska       10.8  51.6    96.8    284     1332.   3370.
#> 3 3         Arizona       9.5  34.2   138.     312.    2346.   4467.
#> 4 4         Arkansas      8.8  27.6    83.2    203.     973.   1862.
#> 5 5         California   11.5  49.4   287      358     2139.   3500.
#> 6 6         Colorado      6.3  42     171.     293.    1935.   3903.
#> # i 10 more variables: auto <dbl>, st <chr>, region <fct>,
#> #   .fittedPC1 <dbl>, .fittedPC2 <dbl>, .fittedPC3 <dbl>,
#> #   .fittedPC4 <dbl>, .fittedPC5 <dbl>, .fittedPC6 <dbl>,
#> #   .fittedPC7 <dbl>
```

Then, we can use `ggplot()` to plot any pair of components. To aid interpretation, I label the points by their state abbreviation and color them by `region` of the U.S.. A geometric interpretation of the plot requires an aspect ratio of 1.0 (via `coord_fixed()`) so that a unit distance on the horizontal axis is the same length as a unit distance on the vertical. To demonstrate that the components are uncorrelated, I also added their data ellipse.

```
crime.pca |>
  broom::augment(crime) |> # add original dataset back in
  ggplot(aes(.fittedPC1, .fittedPC2, color = region)) +
  geom_hline(yintercept = 0) +
  geom_vline(xintercept = 0) +
  geom_point(size = 1.5) +
  geom_text(aes(label = st), nudge_x = 0.2) +
  stat_ellipse(color = "grey") +
  coord_fixed() +
  labs(x = "PC Dimension 1", y = "PC Dimnension 2") +
  theme_minimal(base_size = 14) +
  theme(legend.position = "top")
```

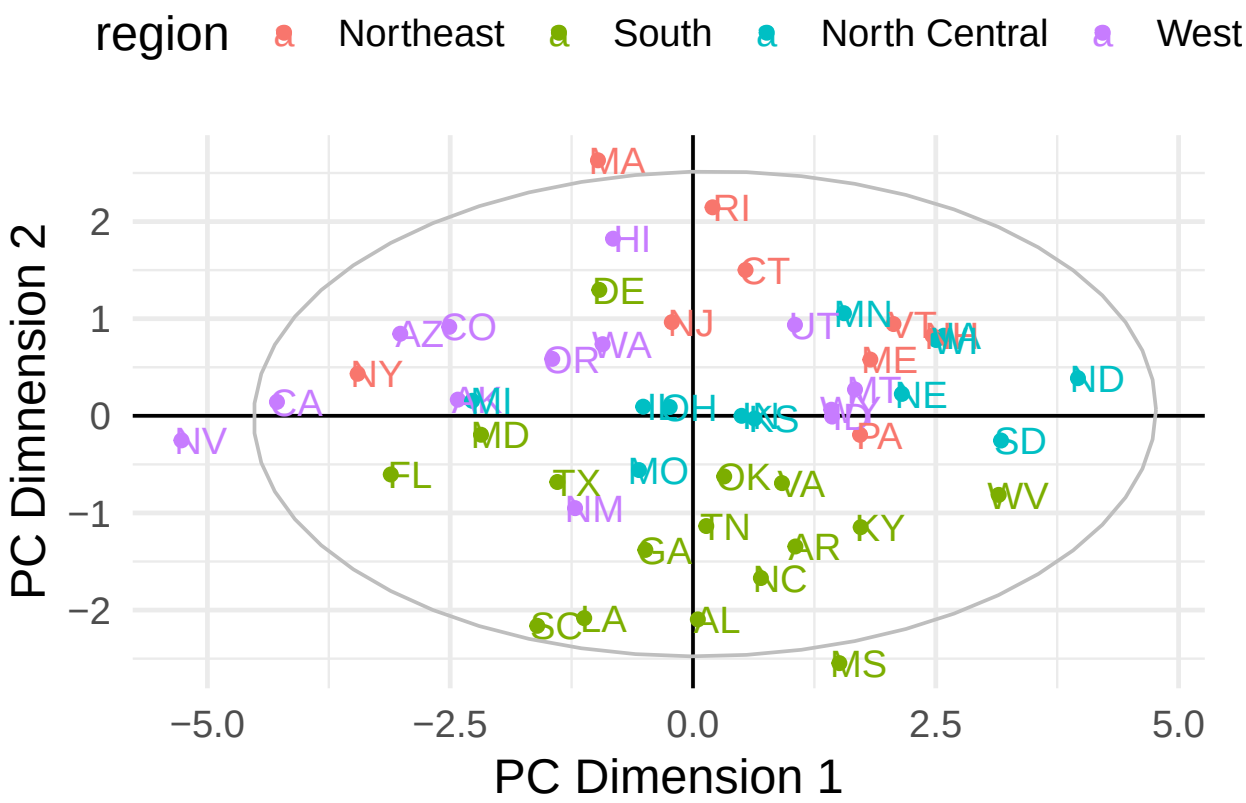


Figure 43: Plot of component scores on the first two principal components for the `crime` data. States are colored by `region`.

To interpret such plots, it is useful consider the observations that are a high and low on each of the axes as well as other information, such as region here, and ask how these differ on the crime statistics. The first component, PC1, contrasts Nevada and California with North Dakota, South Dakota and West Virginia. The second component has most of the southern states on the low end and Massachusetts, Rhode Island and Hawaii on the high end. However, interpretation is easier when we also consider how the various crimes contribute to these dimensions.

When, as here, there are more than two components that seem important in the scree plot, we could obviously go further and plot other pairs.

Variable vectors

You can extract the variable loadings using either `crime.pca$rotation` or `purrr::pluck("rotation")`, similar to what I did with the scores.

```
crime.pca |> purrr::pluck("rotation")
#>             PC1     PC2     PC3     PC4     PC5     PC6     PC7
#> murder   -0.300 -0.6292  0.1782 -0.2321  0.5381  0.2591  0.2676
#> rape     -0.432 -0.1694 -0.2442  0.0622  0.1885 -0.7733 -0.2965
#> robbery  -0.397  0.0422  0.4959 -0.5580 -0.5200 -0.1144 -0.0039
#> assault  -0.397 -0.3435 -0.0695  0.6298 -0.5067  0.1724  0.1917
#> burglary -0.440  0.2033 -0.2099 -0.0576  0.1010  0.5360 -0.6481
#> larceny  -0.357  0.4023 -0.5392 -0.2349  0.0301  0.0394  0.6017
#> auto     -0.295  0.5024  0.5684  0.4192  0.3698 -0.0573  0.1470
```

But note something important in this output: All of the weights for the first component are negative. In PCA, the directions of the eigenvectors are completely arbitrary, in the sense that the vector $-\mathbf{v}_i$ gives the same linear combination as $\mathbf{v}_i$, but with its' sign reversed. For interpretation, it is useful (and usually recommended) to reflect the loadings to a positive orientation by multiplying them by -1.

To reflect the PCA loadings and get them into a convenient format for plotting with `ggplot()`, it is necessary to do a bit of processing, including making the `row.names()` into an explicit variable for the purpose of labeling.

TODO: Should this be a callout-warning or a footnote??

> **`rownames` in R**
>
> R software evolved over many years, particularly in conventions for labeling cases in printed output and graphics. In base-R, the convention was that the `row.names()` of a matrix or data.frame served as observation labels in all printed output and plots, with a default to use numbers `1:n` if there were no rownames. In `ggplot2` and the `tidyverse` framework, the decision was made that observation labels had to be an **explicit** variable in a "tidy" dataset, so it could be used as a variable in constructs like `geom_text(aes(label = label))` as in this example. This change often requires extra steps in software that uses the rownames convention.

```
vectors <- crime.pca |>
  purrr::pluck("rotation") |>
  as.data.frame() |>
  mutate(PC1 = -1 * PC1, PC2 = -1 * PC2) |>      # reflect axes
  tibble::rownames_to_column(var = "label")

vectors[, 1:3]
#>      label   PC1     PC2
#> 1   murder 0.300  0.6292
#> 2     rape 0.432  0.1694
#> 3  robbery 0.397 -0.0422
#> 4  assault 0.397  0.3435
#> 5 burglary 0.440 -0.2033
#> 6  larceny 0.357 -0.4023
#> 7     auto 0.295 -0.5024
```

Then, I plot these using `geom_segment()`, taking some care to use arrows from the origin with a nice shape and add `geom_text()` labels for the variables positioned slightly to the right. Again, `coord_fixed()` ensures equal scales for the axes, which is important because we want to interpret the angles between the variable vectors and the PCA coordinate axes.

```
arrow_style <- arrow(
  angle = 20, ends = "first", type = "closed",
  length = grid::unit(8, "pt")
)

vectors |>
  ggplot(aes(PC1, PC2)) +
  geom_hline(yintercept = 0) +
  geom_vline(xintercept = 0) +
  geom_segment(xend = 0, yend = 0,
               linewidth = 1,
```

```
              arrow = arrow_style,
              color = "brown") +
  geom_text(aes(label = label),
            size = 5,
            hjust = "outward",
            nudge_x = 0.05,
            color = "brown") +
  xlim(-0.4, 0.9) +
  ylim(-0.8, 0.8) +
  coord_fixed() +
  theme_minimal(base_size = 14)
```

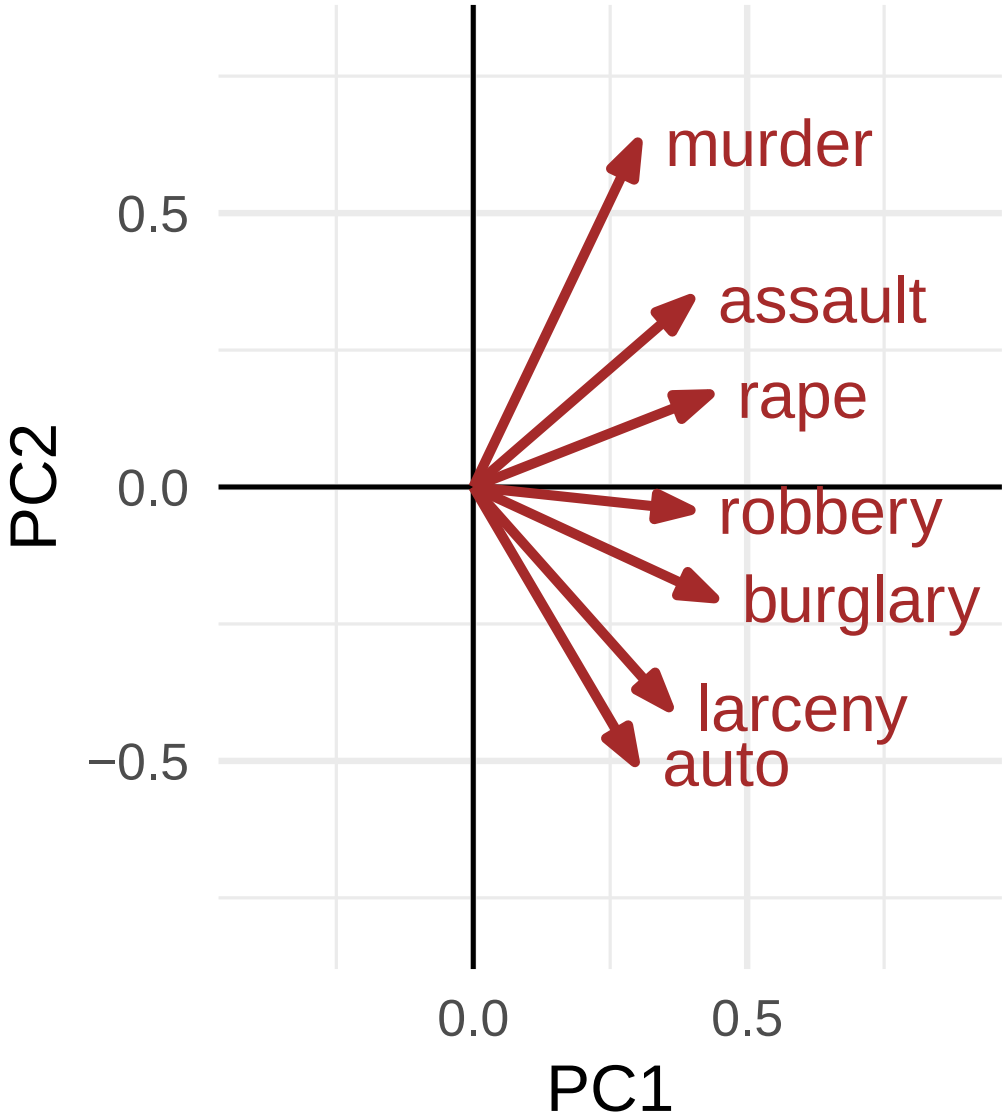


Figure 44: Plot of component loadings the first two principal components for the `crime` data. These are interpreted as the contributions of the variables to the components.

What is shown in Figure 44 has the following interpretations:

(1) the lengths of the variable vectors, $||\mathbf{v}_i|| = \sqrt{\Sigma_{j=1}^{j=2} v_{ij}^2}$ give the proportion of variance of each variable accounted for in a two-dimensional display.

(2) the value, v_{ij}, of the vector for variable $\mathbf{x}_i$ on component j gives the correlation of that variable with the jth principal component.

(3) the cosine of the angle between two variable vectors, $\mathbf{v}_i$ and $\mathbf{v}_j$ gives the approximation of the correlation between $\mathbf{x}_i$ and $\mathbf{x}_j$ that is shown in this space. This means that two variable vectors that point in the same direction are highly correlated, while variable vectors at right angles are approximately uncorrelated.

To illustrate point (1), the following indicates that almost 70% of the variance of `murder` is represented in the the 2D plot shown in Figure 43, but only 40% of the variance of `robbery` is captured.

```
vectors |> select(label, PC1, PC2) |>
  mutate(length = sqrt(PC1^2 + PC2^2))
#>      label   PC1     PC2 length
#> 1   murder 0.300  0.6292  0.697
#> 2     rape 0.432  0.1694  0.464
#> 3  robbery 0.397 -0.0422  0.399
#> 4  assault 0.397  0.3435  0.525
#> 5 burglary 0.440 -0.2033  0.485
#> 6  larceny 0.357 -0.4023  0.538
#> 7     auto 0.295 -0.5024  0.583
```

0.7 Biplots

The biplot is a simple and powerful idea that came from the recognition that you can overlay a plot of observation scores in a principal components analysis with the information of the variable loadings (weights) to give a simultaneous display that is easy to interpret. In this sense, a biplot is generalization of a scatterplot, projecting from data space to PCA space, where the observations are shown by points, as in the plots of component scores in Figure 43, but with the variables also shown by vectors (or scaled linear axes aligned with those vectors).

The idea of the biplot was introduced by Ruben Gabriel (1971, 1981) and later expanded in scope by Gower & Hand (1996). The book by Greenacre (2010) gives a practical overview of the many variety of biplots and Gower et al. (2011) provide a full treatment ...

Biplot methodolgy is far more general than I cover here. Categorical variables can be incorporated in PCA using category level points. Two-way frequency tables of categorical variables can be analysed using *correspondence analysis*, which is similar to PCA, but designed to account for the maximum amount of the χ^2 statistic for association; *multiple correspondence analysis* extends this to method to multi-way tables (Friendly & Meyer, 2016; Greenacre, 1984).

0.7.1 Constructing a biplot

The biplot is constructed by using the singular value decomposition (SVD) to obtain a low-rank approximation to the data matrix $\mathbf{X}_{n \times p}$ (centered, and optionally scaled to unit variances) whose n rows are the observations and whose p columns are the variables.

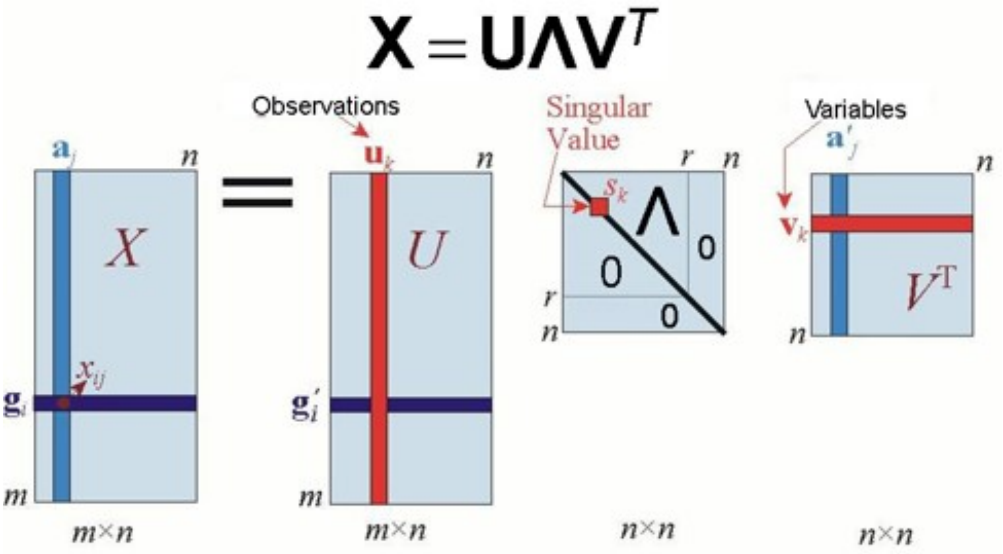


Figure 45: The singular value decomposition expresses a data matrix **X** as the product of a matrix **U** of observation scores, a diagonal matrix **Lambda** of singular values and a matrix **V** of variable weights. **TODO**: Re-draw to fix notation.

Using the SVD, the matrix $\mathbf{X}$, of rank $r \leq p$ can be expressed *exactly* as:

$$\mathbf{X} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}' = \sum_{i}^{r} \lambda_i \mathbf{u}_i \mathbf{v}_i' \; , \tag{0.1}$$

where

- $\mathbf{U}$ is an $n \times r$ orthonormal matrix of uncorrelated observation scores; these are also the eigenvectors of $\mathbf{X}\mathbf{X}'$,
- $\mathbf{\Lambda}$ is an $r \times r$ diagonal matrix of singular values, $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_r$, which are also the square roots of the eigenvalues of $\mathbf{X}\mathbf{X}'$.
- $\mathbf{V}$ is an $r \times p$ orthonormal matrix of variable weights and also the eigenvectors of $\mathbf{X}'\mathbf{X}$.

Then, a rank 2 (or 3) PCA approximation $\widehat{\mathbf{X}}$ to the data matrix used in the biplot can be obtained from the first 2 (or 3) singular values λ_i and the corresponding $\mathbf{u}_i, \mathbf{v}_i$ as:

$$\mathbf{X} \approx \widehat{\mathbf{X}} = \lambda_1 \mathbf{u}_1 \mathbf{v}_1' + \lambda_2 \mathbf{u}_2 \mathbf{v}_2' \; .$$

The variance of $\mathbf{X}$ accounted for by each term is λ_i^2.

A biplot is then obtained by overlaying two scatterplots that share a common set of axes and have a between-set scalar product interpretation. Typically, the observations (rows of $\mathbf{X}$) are represented as points and the variables (columns of $\mathbf{X}$) are represented as vectors from the origin.

The `scale` factor, α allows the variances of the components to be apportioned between the row points and column vectors, with different interpretations, by representing the approximation $\widehat{\mathbf{X}}$ as the product of two matrices,

$$\widehat{\mathbf{X}} = (\mathbf{U}\mathbf{\Lambda}^{\alpha})(\mathbf{\Lambda}^{1-\alpha}\mathbf{V}') = \mathbf{A}\mathbf{B}'$$

This notation uses a little math trick involving a power, $0 \leq \alpha \leq 1$: When $\alpha = 1$, $\mathbf{\Lambda}^{\alpha} = \mathbf{\Lambda}^1 = \mathbf{\Lambda}$, and $\mathbf{\Lambda}^{1-\alpha} = \mathbf{\Lambda}^0 = \mathbf{I}$. $\alpha = 1/2$ gives the diagonal matrix $\mathbf{\Lambda}^{1/2}$ whose elements are the square roots of the singular values.

The choice $\alpha = 1$ assigns the singular values totally to the left factor; then, the angle between two variable

vectors, reflecting the inner product $\mathbf{x}_j^T, \mathbf{x}_{j'}$ approximates their correlation or covariance, and the distance between the points approximates their Mahalanobis distances. $\alpha = 0$ gives a distance interpretation to the column display. $\alpha = 1/2$ gives a symmetrically scaled biplot. *TODO**: Explain this better.

When the singular values are assigned totally to the left or to the right factor, the resultant coordinates are called *principal coordinates* and the sum of squared coordinates on each dimension equal the corresponding singular value. The other matrix, to which no part of the singular values is assigned, contains the so-called *standard coordinates* and have sum of squared values equal to 1.0.

0.7.2 Biplots in R

There are a large number of R packages providing biplots. The most basic, `stats::biplot()`, provides methods for `"prcomp"` and `"princomp"` objects. Among other packages, **factoextra** package (Kassambara & Mundt, 2020), an extension of **FactoMineR** (Husson et al., 2023), is perhaps the most comprehensive and provides `ggplot2` graphics. In addition to biplot methods for quantitative data using PCA (`fviz_pca()`), it offers biplots for categorical data using correspondence analysis (`fviz_ca()`) and multiple correspondence analysis (`fviz_mca()`); factor analysis with mixed quantitative and categorical variables (`fviz_famd()`) and cluster analysis (`fviz_cluster()`). **TODO**: Also mention **adegraphics** package

Here, I use the **ggbiplot** package, which aims to provide a simple interface to biplots within the `ggplot2` framework. I also use some convenient utility functions from **factoextra**.

0.7.3 Example

A basic biplot of the `crime` data, using standardized principal components and labeling the observation by their state abbreviation is shown in Figure 46. The correlation circle indicates that these components are uncorrelated and have equal variance in the display.

```
crime.pca <- reflect(crime.pca) # reflect the axes

ggbiplot(crime.pca,
   obs.scale = 1, var.scale = 1,
   labels = crime$st ,
   circle = TRUE,
   varname.size = 4,
   varname.color = "brown") +
  theme_minimal(base_size = 14)
```

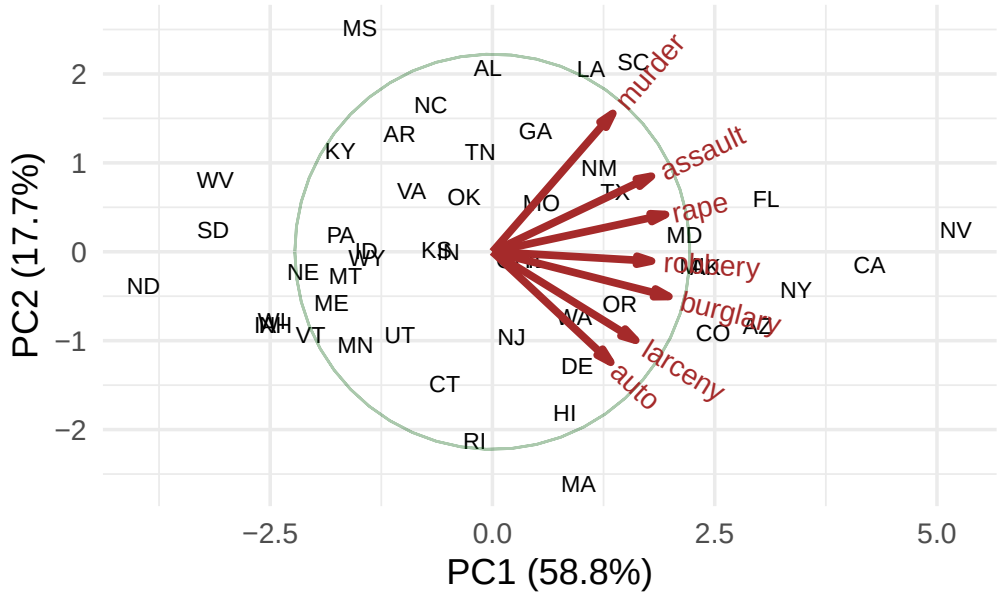


Figure 46: Basic biplot of the crime data. ...

In this dataset the states are grouped by region and we saw some differences among regions in the plot (Figure 43) of component scores. `ggbiplot()` provides options to include a `groups =` variable, used to color the observation points and also to draw their data ellipses, facilitating interpretation.

```
ggbiplot(crime.pca,
   obs.scale = 1, var.scale = 1,
   groups = crime$region,
   labels = crime$st,
   labels.size = 4,
   var.factor = 1.4,
   ellipse = TRUE, ellipse.level = 0.5, ellipse.alpha = 0.1,
   circle = TRUE,
   varname.size = 4,
   varname.color = "black") +
  labs(fill = "Region", color = "Region") +
  theme_minimal(base_size = 14) +
  theme(legend.direction = 'horizontal', legend.position = 'top')
```

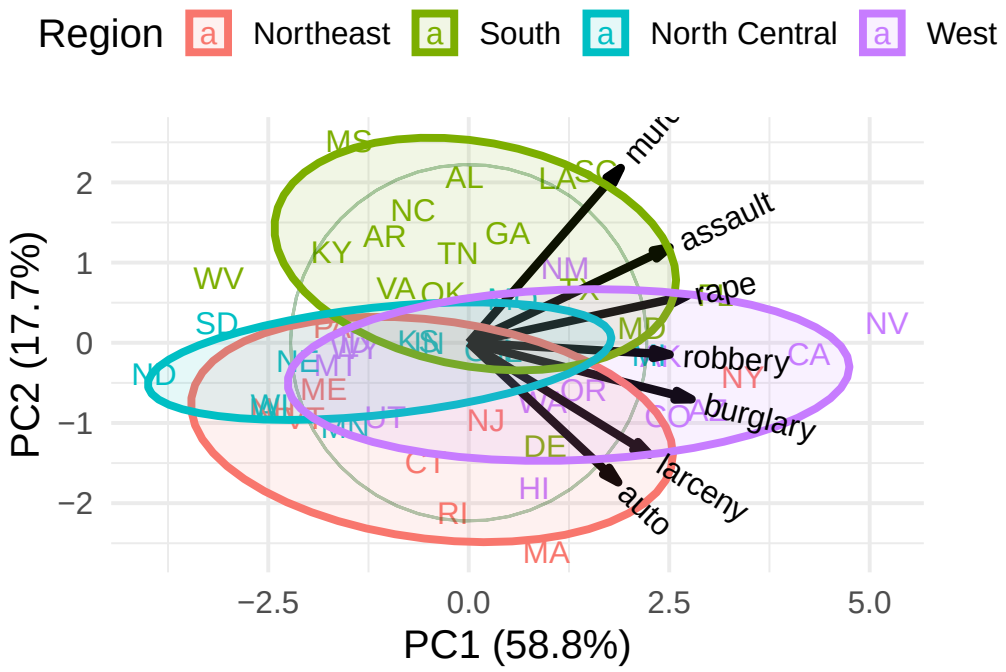


Figure 47: Enhanced biplot of the crime data, grouping the states by region and adding data ellipses.

This plot provides what is necessary to interpret the nature of the components and also the variation of the states in relation to these. In this, the data ellipses for the regions provide a visual summary that aids interpretation.

- From the variable vectors, it seems that PC1, having all positive and nearly equal loadings, reflects a total or overall index of crimes. Nevada, California, New York and Florida are highest on this, while North Dakota, South Dakota and West Virginia are lowest.
- The second component, PC2, shows a contrast between crimes against persons (murder, assault, rape) at the top and property crimes (auto theft, larceny) at the bottom. Nearly all the Southern states are high on personal crimes; states in the North East are generally higher on property crimes.
- Western states tend to be somewhat higher on overall crime rate, while North Central are lower on average. In these states there is not much variation in the relative proportions of personal vs. property crimes.

Moreover, in this biplot you can interpret the the value for a particular state on a given crime by considering its projection on the variable vector, where the origin corresponds to the mean, positions along the vector have greater than average values on that crime, and the opposite direction have lower than average values. For example, Massachusetts has the highest value on auto theft, but a value less than the mean. Louisiana and South Carolina on the other hand are highest in the rate of murder and slightly less than average on auto theft.

These 2D plots account for only 76.5% of the total variance of crimes, so it is useful to also examine the third principal component, which accounts for an additional 10.4%. The `choices =` option controls which dimensions are plotted.

```
ggbiplot(crime.pca,
         choices = c(1,3),
         obs.scale = 1, var.scale = 1,
         groups = crime$region,
         labels = crime$st,
```

```
        labels.size = 4,
        var.factor = 2,
        ellipse = TRUE, ellipse.level = 0.5, ellipse.alpha = 0.1,
        circle = TRUE,
        varname.size = 4,
        varname.color = "black") +
  labs(fill = "Region", color = "Region") +
  theme_minimal(base_size = 14) +
  theme(legend.direction = 'horizontal', legend.position = 'top')
```

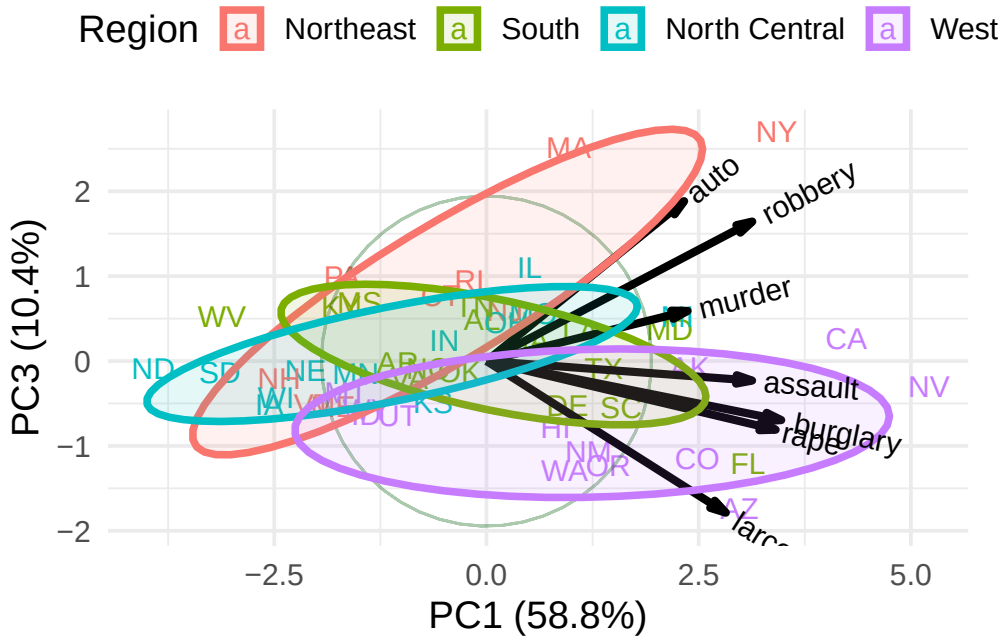


Figure 48: Biplot of dimensions 1 & 3 of the crime data, with data ellipses for the regions.

Dimension 3 in Figure 48 is more subtle. One interpretation is a contrast between larceny, which is a larceny (simple theft) and robbery, which involves stealing something from a person and is considered a more serious crime with an element of possible violence. In this plot, murder has a relatively short variable vector, so does not contribute very much to differences among the states.

0.7.4 Biplot contributions and quality

To better understand how much each variable contributes to the biplot dimensions, it is helpful to see information about the variance of variables along each dimension. Graphically, this is nothing more than a measure of the lengths of projections of the variables on each of the dimensions. `factoextra::get_pca_var()` calculates a number of tables from a `"prcomp"` or similar object.

```
var_info <- factoextra::get_pca_var(crime.pca)
names(var_info)
#> [1] "coord"   "cor"     "cos2"    "contrib"
```

The component `cor` gives correlations of the variables with the dimensions and `contrib` gives their variance contributions as percents, where each row and column sums to 100.

```
contrib <- var_info$contrib
cbind(contrib, Total = rowSums(contrib)) |>
  rbind(Total = c(colSums(contrib), NA)) |>
  round(digits=2)
#>           Dim.1  Dim.2  Dim.3  Dim.4  Dim.5  Dim.6  Dim.7 Total
#> murder     9.02  39.59   3.18   5.39  28.96   6.71   7.16   100
#> rape      18.64   2.87   5.96   0.39   3.55  59.79   8.79   100
#> robbery   15.75   0.18  24.59  31.14  27.04   1.31   0.00   100
#> assault   15.73  11.80   0.48  39.67  25.67   2.97   3.68   100
#> burglary  19.37   4.13   4.41   0.33   1.02  28.73  42.01   100
#> larceny   12.77  16.19  29.08   5.52   0.09   0.16  36.20   100
#> auto       8.71  25.24  32.31  17.58  13.67   0.33   2.16   100
#> Total    100.00 100.00 100.00 100.00 100.00 100.00 100.00    NA
```

These contributions can be visualized as sorted barcharts for a given axis using `factoextra::fviz_contrib()`. The dashed horizontal lines are at the average value for each dimension.

```
p1 <- fviz_contrib(crime.pca, choice = "var", axes = 1,
                   fill = "lightgreen", color = "black")
p2 <- fviz_contrib(crime.pca, choice = "var", axes = 2,
                   fill = "lightgreen", color = "black")
p1 + p2
```

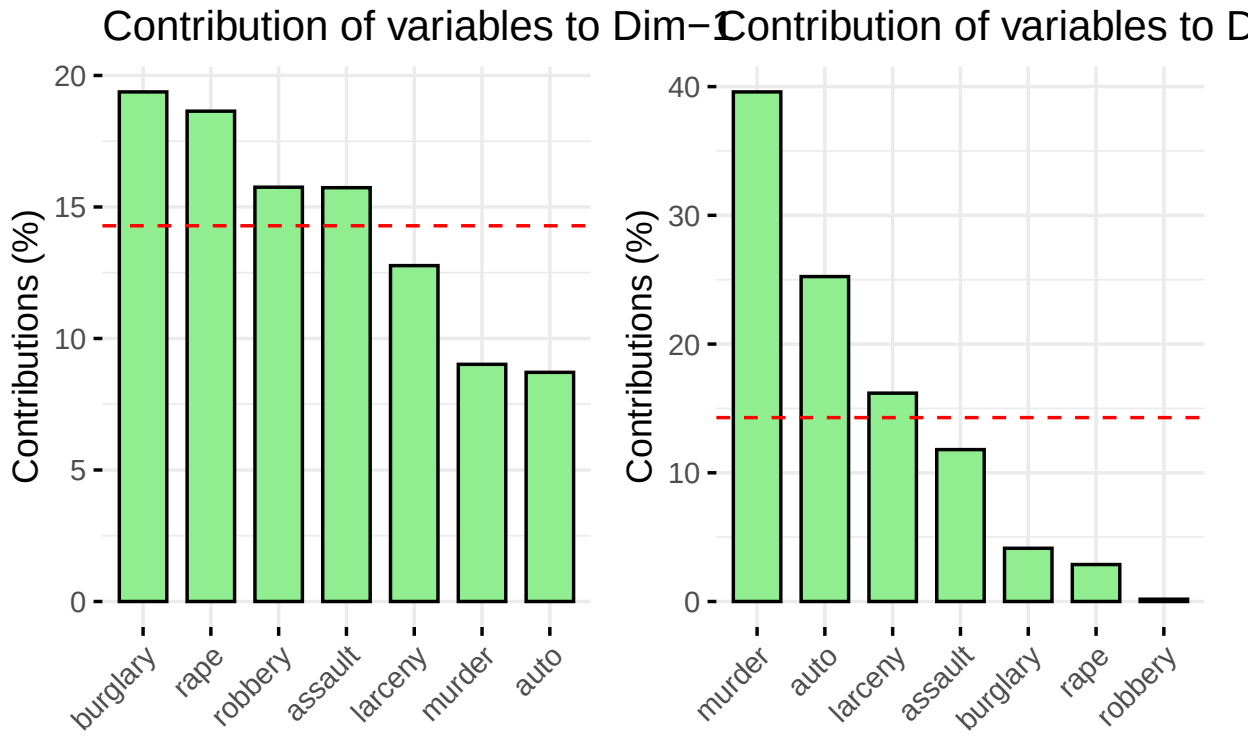


Figure 49: Contributions of the crime variables to dimensions 1 (left) & 2 (right) of the PCA solution

A simple rubric for interpreting the dimensions in terms of the variable contributions is to mention those that are largest or above average on each dimension. So, burglary and rape contribute most to the first dimension, while murder and auto theft contribute most to the second.

Another useful measure is called `cos2`, the *quality* of representation, meaning how much of a variable is represented in a given component. The columns sum to the eigenvalue for each dimension. The rows each sum to 1.0, meaning each variable is completely represented on all components, but we can find the quality of a k-D solution by summing the values in the first k columns. These can be plotted in a style similar to Figure 49 using `factoextra::fviz_cos2()`.

```
quality <- var_info$cos2
rowSums(quality)
#>   murder     rape  robbery  assault burglary  larceny     auto
#>        1        1        1        1        1        1        1

colSums(quality)
#> Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6 Dim.7
#> 4.115 1.239 0.726 0.316 0.258 0.222 0.124

cbind(quality[, 1:2],
      Total = rowSums(quality[, 1:2])) |>
  round(digits = 2)
#>          Dim.1 Dim.2 Total
#> murder    0.37  0.49  0.86
#> rape      0.77  0.04  0.80
#> robbery   0.65  0.00  0.65
#> assault   0.65  0.15  0.79
#> burglary  0.80  0.05  0.85
#> larceny   0.53  0.20  0.73
#> auto      0.36  0.31  0.67
```

In two dimensions, murder and burglary are best represented; robbery and larceny are the worst, but as we saw above (Figure 48), these crimes are implicated in the third dimension.

0.7.5 Supplementary variables

An important feature of biplot methodology is that once you have a reduced-rank display of the relations among a set of variables, you can use other available data to help interpret what what is shown in the biplot. In a sense, this is what I did above in Figure 47 and Figure 48 using `region` as a grouping variable and summarizing the variability in the scores for states with their data ellipses by region.

When we have other quantitative variables on the same observations, these can be represented as supplementary variables in the same space, by what amounts to regressions of these new variables on the scores for the principal component dimensions. For example, the left panel of Figure 50 depicts the vector geometry of a regression of a variable $\mathbf{y}$ on two predictors, $\mathbf{x}_1$ and $\mathbf{x}_2$. The fitted vector, $\widehat{\mathbf{y}}$, is the perpendicular projection of $\mathbf{y}$ onto the plane of $\mathbf{x}_1$ and $\mathbf{x}_2$. In the same way, in the right panel the supplementary variable is projected into the plane of two principal component axes. The fitted vector shows how that additional variable relates to the biplot dimensions.

For this example, it happens that some suitable supplementary variables to aid interpretation of crime rates are available in the dataset `datsets::state.x77`, which was obtained from the U.S. Bureau of the Census *Statistical Abstract of the United States* for 1977. I select a few of these below and make the state name a column variable so it can be merged with the `crime` data.

```
supp_data <- state.x77 |>
  as.data.frame() |>
  tibble::rownames_to_column(var = "state") |>
```

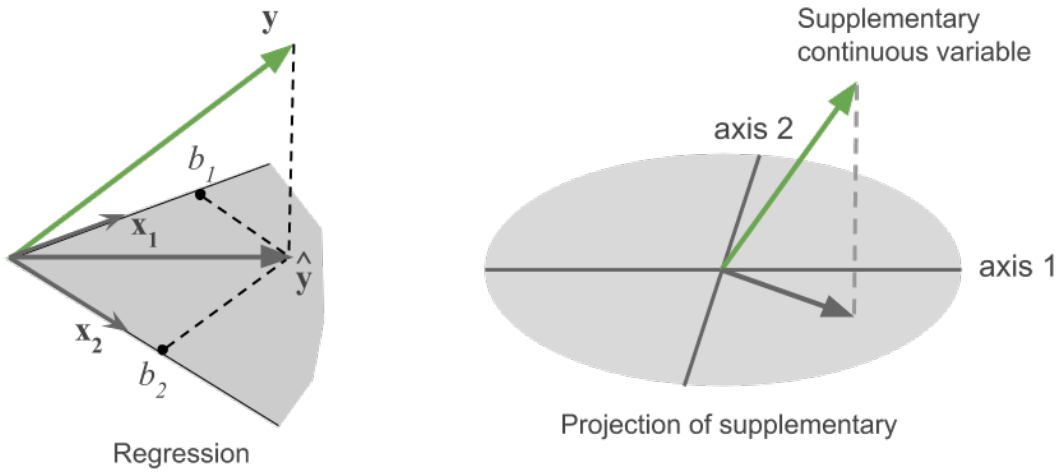


Figure 50: Fitting supplementary variables in a biplot is analogous to regression on the principal component dimensions. *Source*: Aluja et al. (2018), Figure 2.11

```
  select(state, Income:`Life Exp`, `HS Grad`) |>
  rename(Life_Exp = `Life Exp`,
         HS_Grad = `HS Grad`)

head(supp_data)
#>        state Income Illiteracy Life_Exp HS_Grad
#> 1    Alabama   3624        2.1     69.0    41.3
#> 2     Alaska   6315        1.5     69.3    66.7
#> 3    Arizona   4530        1.8     70.5    58.1
#> 4   Arkansas   3378        1.9     70.7    39.9
#> 5 California   5114        1.1     71.7    62.6
#> 6   Colorado   4884        0.7     72.1    63.9
```

Then, we can merge the `crime` data with the `supp_data` dataset to produce something suitable for analysis using `factoMineR::PCA()`.

```
crime_joined <-
  dplyr::left_join(crime[, 1:8], supp_data, by = "state")
names(crime_joined)
#>  [1] "state"      "murder"     "rape"       "robbery"
#>  [5] "assault"    "burglary"   "larceny"    "auto"
#>  [9] "Income"     "Illiteracy" "Life_Exp"   "HS_Grad"
```

`PCA()` can only get the labels for the observations from the `row.names()` of the dataset, so I assign them explicitly. The supplementary variables are specified by the argument `quanti.sup` as the indices of the columns in what is passed as the data argument.

```
row.names(crime_joined) <- crime$st
crime.PCA_sup <- PCA(crime_joined[,c(2:8, 9:12)],
                     quanti.sup = 8:11,
                     scale.unit=TRUE,
```

```
                      ncp=3,
                      graph = FALSE)
```

The essential difference between the result of `prcomp()` used earlier to get the `crime.pca` object and the result of `PCA()` with supplementary variables is that the `crime.PCA_sup` object now contains a `quanti.sup` component containing the coordinates for the supplementary variables in PCA space.

These can be calculated directly as a the coefficients of a multivariate regression of the standardized supplementary variables on the PCA scores for the dimensions, with no intercept to force the fitted vectors to go through the origin. For example, in the plot below (Figure 51), the vector for Income has coordinates (0.192, .530) on the first two PCA dimensions.

```
reg.data <- cbind(scale(supp_data[, -1]),
                  crime.PCA_sup$ind$coord) |>
  as.data.frame()

sup.mod <- lm(cbind(Income, Illiteracy, Life_Exp, HS_Grad) ~
                   0 + Dim.1 + Dim.2 + Dim.3,
              data = reg.data )

(coefs <- t(coef(sup.mod)))
#>             Dim.1  Dim.2   Dim.3
#> Income      0.192  0.530  0.0482
#> Illiteracy  0.112 -0.536  0.1689
#> Life_Exp   -0.131  0.649 -0.2158
#> HS_Grad     0.103  0.610 -0.4095
```

The `PCA()` result can be plotted using `FactoMiner::plot()` or various `factoextra` functions like `fviz_pca_var()` for a plot of the variable vectors or `fviz_pca_biplot()` for a biplot. When a `quanti.sup` component is present, supplementary variables are also shown in the displays.

For simplicity I use `FactoMiner::plot()` here and only show the variable vectors. For consistency with earlier plots, I first reflect the orientation of the 2nd PCA dimension so that crimes of personal violence are at the top, as in Figure 44.

```
# reverse coordinates of Dim 2
crime.PCA_sup <- ggbiplot::reflect(crime.PCA_sup, columns = 2)
# also reverse the orientation of coordinates for supplementary vars on Dim 2
# crime.PCA_sup$quanti.sup$coord[, 2] <- -crime.PCA_sup$quanti.sup$coord[, 2]
plot(crime.PCA_sup, choix = "var")
```

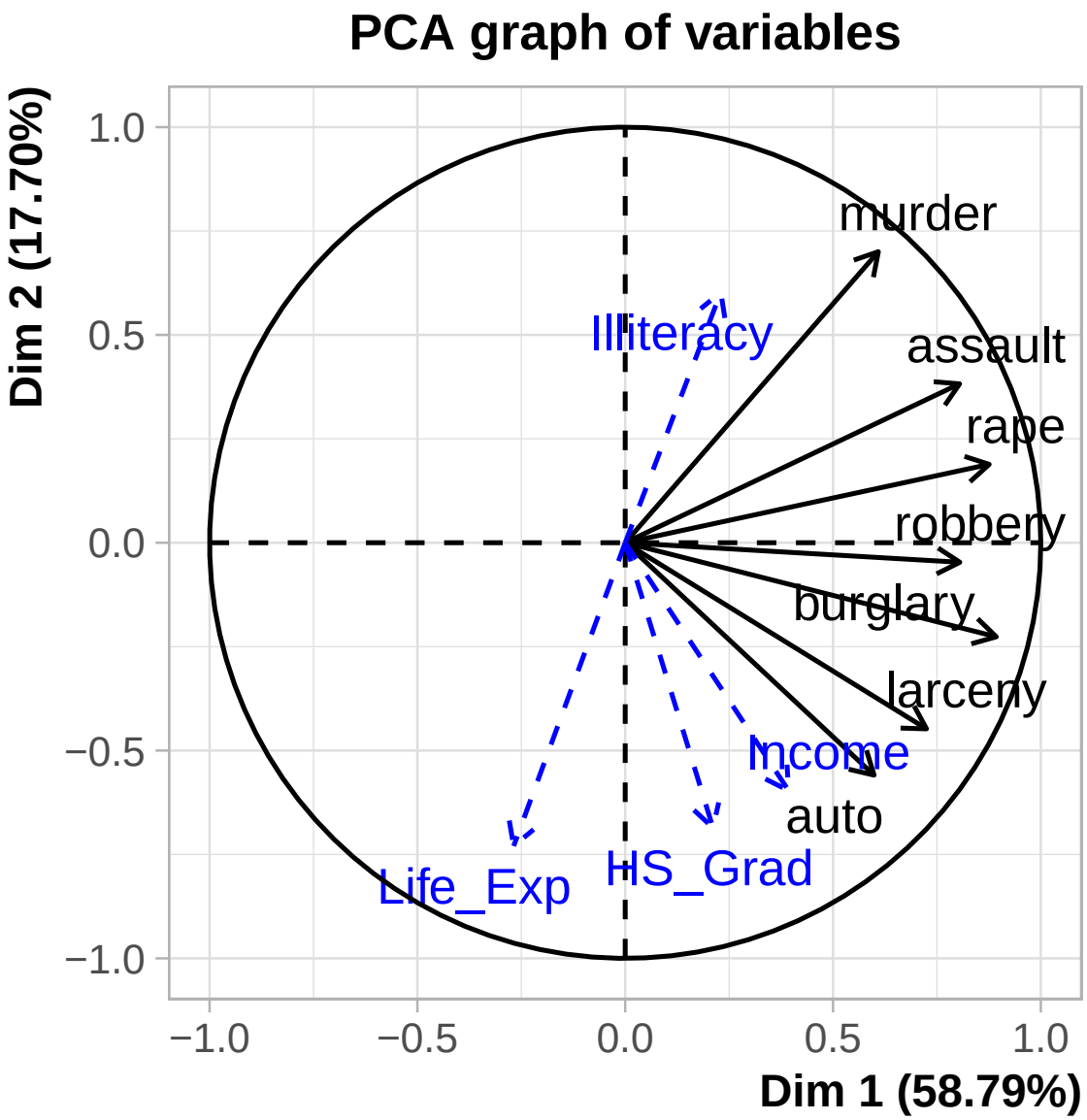


Figure 51: PCA plot of variables for the crime data, with vectors for the supplementary variables showing their association with the principal component dimensions.

Recall that from earlier analyses, I interpreted the the dominant PC1 dimension as reflecting overall rate of crime. The contributions to this dimension, which are the projections of the variable vectors on the horizontal axis in Figure 44 and Figure 47 were shown graphically by barcharts in the left panel of Figure 49.

But now in Figure 51, with the addition of variable vectors for the supplementary variables, you can see how income, rate of illiteracy, life expectancy and proportion of high school graduates are related to the variation in rates of crimes for the U.S. states.

On dimension 1, what stands out is that life expectancy is associated with lower overall crime, while other supplementary variable have positive associations. On dimension 2, crimes against persons (murder, assault, rape) are associated with greater rates of illiteracy among the states, which as we earlier saw (Figure 47) were more often Southern states. Crimes against property (auto theft, larceny) at the bottom of this dimension are associated with higher levels of income and high school graduates

0.8 Application: Variable ordering for data displays

In many multivariate data displays, such as scatterplot matrices, parallel coordinate plots and others reviewed in Chapter , the order of different variables might seem arbitrary. They might appear in alphabetic order, or more often in the order they appear in your dataset, for example when you use `pairs(mydata)`. Yet, the principle of *effect ordering* (Friendly & Kwan (2003)) for variables says you should try to arrange the variables so that adjacent ones are as similar as possible.[5]

For example, the `mtcars` dataset contains data on 32 automobiles from the 1974 U.S. magazine *Motor Trend* and consists of fuel comsumption (`mpg`) and 10 aspects of automobile design (`cyl`: number of cyliners; `hp`: horsepower, `wt`: weight) and performance (`qsec`: time to drive a quarter-mile). What can we see from a simple `corrplot()` of their correlations?

```
data(mtcars)
library(corrplot)
R <- cor(mtcars)
corrplot(R,
         method = 'ellipse',
         title = "Dataset variable order",
         tl.srt = 0, tl.col = "black", tl.pos = 'd',
         mar = c(0,0,1,0))
```

[5]The general topic of arranging items (variables, factor values) in an orderly sequence is called *seriation*, and stems from methods of dating in archaeology, used to arrange stone tools, pottery fragments, and other artifacts in time order. In R, the **seriation** package (Hahsler et al., 2023) provides a wide range of techniques. ...

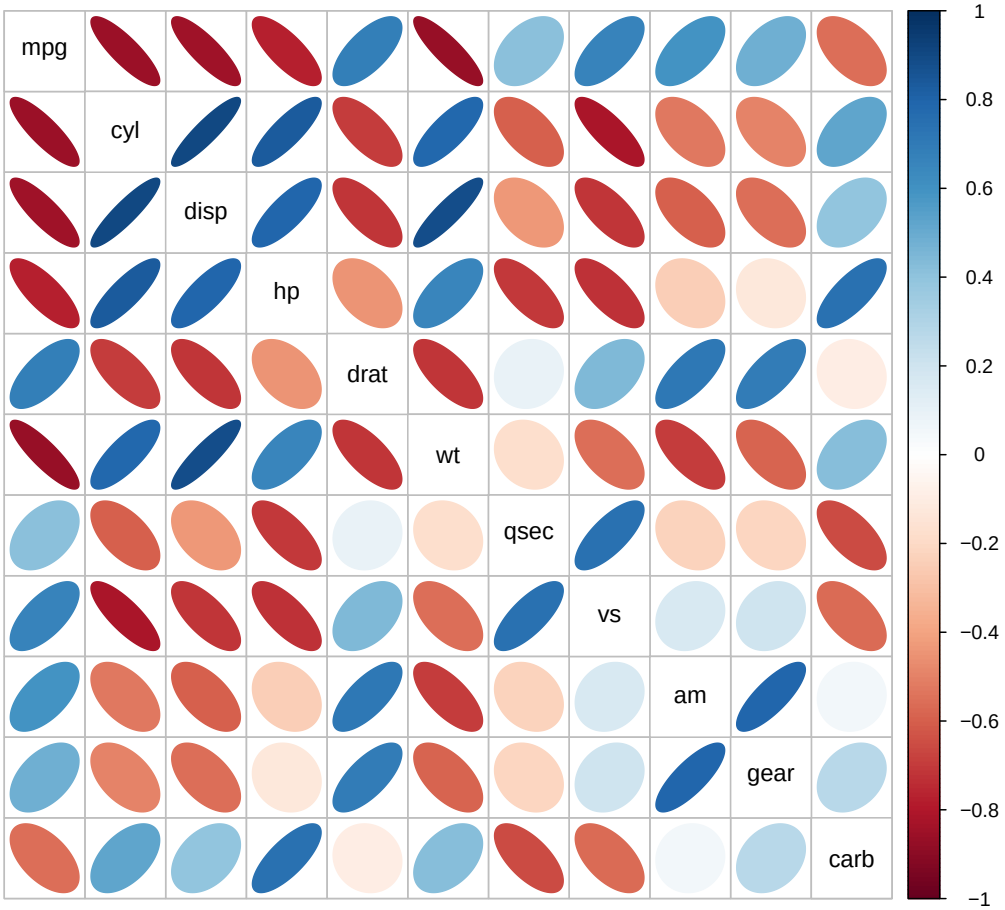


Figure 52: Corrplot of `mtcars` data, with the variables in the order they appear in the dataset.

In this display you can scan the rows and columns to "look up" the sign and approximate magnitude of a given correlation; for example, the correlation between `mpg` and `cyl` appears to be about -0.9, while that between `mpg` and `gear` is about 0.5. Of course, you could print the correlation matrix to find the actual values (-0.86 and 0.48 respectively):

```
print(floor(100*R))
#>      mpg cyl disp  hp drat  wt qsec  vs  am gear carb
#> mpg  100 -86  -85 -78   68 -87   41  66  59   48  -56
#> cyl  -86 100   90  83  -70  78  -60 -82 -53  -50   52
#> disp -85  90  100  79  -72  88  -44 -72 -60  -56   39
#> hp   -78  83   79 100  -45  65  -71 -73 -25  -13   74
#> drat  68 -70  -72 -45  100 -72    9  44  71   69  -10
#> wt   -87  78   88  65  -72 100  -18 -56 -70  -59   42
#> qsec  41 -60  -44 -71    9 -18  100  74 -23  -22  -66
#> vs    66 -82  -72 -73   44 -56   74 100  16   20  -57
#> am    59 -53  -60 -25   71 -70  -23  16 100   79    5
```

```
#> gear  48 -50  -56 -13   69 -59  -22  20  79  100   27
#> carb -56  52   39  74  -10  42  -66 -57   5   27  100
```

Because the angles between variable vectors in the biplot reflect their correlations, Friendly & Kwan (2003) defined **principal component variable ordering** as the order of angles, a_i of the first two eigenvectors, $\mathbf{v}_1, \mathbf{v}_2$ around the unit circle. These values are calculated going counter-clockwise from the 12:00 position as:

$$a_i = \begin{cases} \tan^{-1}(v_{i2}/v_{i1}), & \text{if } v_{i1} > 0; \\ \tan^{-1}(v_{i2}/v_{i1}) + \pi, & \text{otherwise.} \end{cases}$$

(read $\tan^{-1}(x)$ as "the angle whose tangent is x".)

TODO: Make a diagram of this

For the `mtcars` data the biplot in Figure 53 accounts for 84% of the total variance so a 2D representation is fairly good. The plot shows the variables as widely dispersed. There is a collection at the left of positively correlated variables and another positively correlated set at the right.

```
mtcars.pca <- prcomp(mtcars, scale. = TRUE)
ggbiplot(mtcars.pca,
         circle = TRUE,
         point.size = 2.5,
         varname.size = 6,
         varname.color = "brown") +
  theme_minimal(base_size = 14)
```

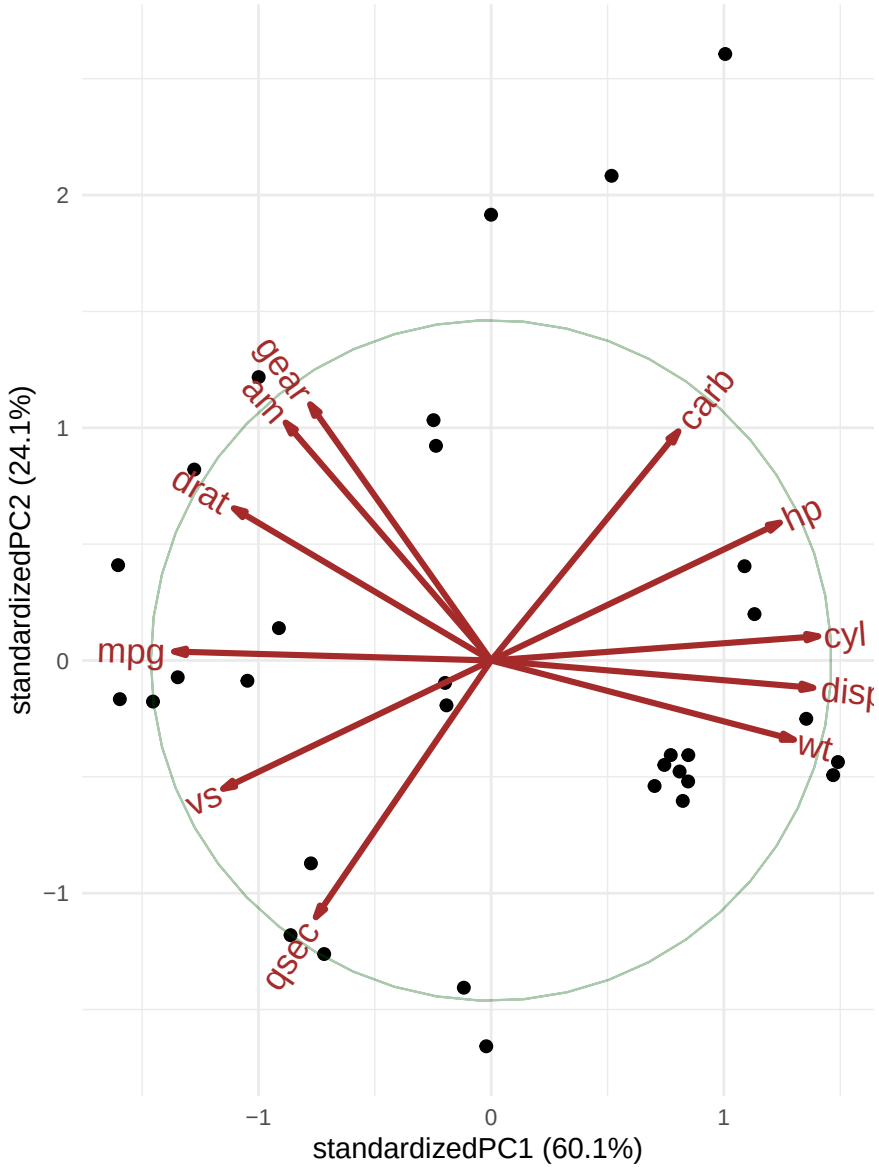


Figure 53: Biplot of the `mtcars` data ...

In `corrplot()` principal component variable ordering is implemented using the `order = "AOE"` option. There are a variety of other methods based on hierarchical cluster analysis described in the package vignette.

Figure 54 shows the result. A nice feature of `corrplot()` is the ability to manually highlight blocks of variables that have a similar pattern of signs by outlining them with rectangles. From the biplot, the two main clusters of positively correlated variables seemed clear, and are outlined in the plot using `corrplot::corrRect()`. What became clear in the corrplot is that `qsec`, the time to drive a quarter-mile from a dead start didn't fit this pattern, so I highlighted it separately.

```
corrplot(R,
         method = 'ellipse',
         order = "AOE",
         title = "PCA variable order",
```

```
        tl.srt = 0, tl.col = "black", tl.pos = 'd',
        mar = c(0,0,1,0)) |>
  corrRect(c(1, 6, 7, 11))
```

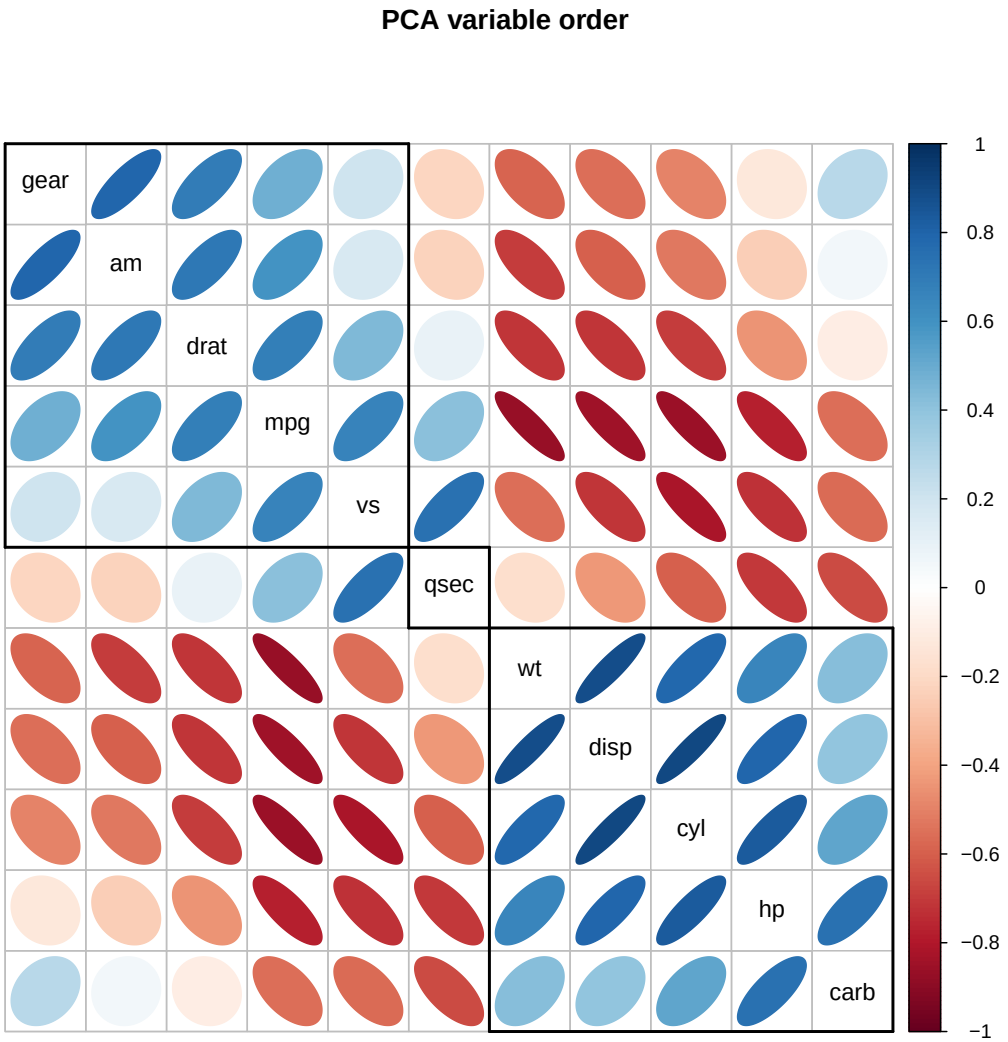


Figure 54: Corrplot of `mtcars` data, with the variables ordered according to the variable vectors in the biplot.

But wait, there is something else to be seen in Figure 54. Can you see one cell that doesn't fit with the rest?

Yes, the correlation of number of forward gears (`gear`) and number of carburators (`carb`) in the upper left and lower right corners stands out as moderately positive (0.27) while all the others in their off-diagonal blocks are negative. This is another benefit of effect ordering: when you arrange the variables so that the most highly related variable are together, features that deviate from dominant pattern become visible.

0.9 Application: Eigenfaces

There are many applications of principal components analysis beyond the use for visualization for multivariate data covered here, that rely on its' ability as a **dimension reduction** technique, that is, to find a low-dimensional approximation to a high-dimensional dataset.

> **i** Machine learning uses
>
> In machine learning, for example, PCA is a method used to reduce model complexity and avoid overfitting by *feature extraction*, which amounts to fitting a response variable in a low-D space of the predictors. This is just another name for *principal components regression*, where, instead of regressing the dependent variable on all the explanatory variables directly, a smaller number principal components of the explanatory variables is used as predictors. This has the added benefit that it avoids problems of collinearity (section-ref) due to high correlations of the predictors, because the principal component scores are necessarily uncorrelated. When the goal is model explanation rather than pure prediction, it has the disadvantage that the components may be hard to interpret.

An interesting class of problems have to do with image processing, where an image of size width $\times$ height in pixels can be represented by a $w \times h$ array of greyscale values x_{ij} in the range of [0, 1] or $h \times w \times 3$ array x_{ijk} of (red, green, blue) color values. For example a single 640×640 photo is comprised of about 400K pixels in B/W and 1200K pixels in color.

The uses here include

- **Image compression**: a process applied to a graphics file to minimize its size in bytes for storage or transmission, without degrading image quality below an acceptable threshold
- **image enhancement**: improving the quality of an image, with applications in Computer Vision tasks, remote sensing, and satellite imagery.
- **facial recognition**: classifying or matching a facial image against a large corpus of stored images.

When PCA is used on facial images, you can think of the process as generating **eigenfaces**, a representation of the pixels in the image in terms of an eigenvalue decomposition. Dimension reduction means that a facial image can be considerably compressed by removing the components associated with small dimensions.

As an example, consider the black and white version of the Mona Lisa shown in Figure 55. The idea and code for this example is adapted from this blog post by Kieran Healy.[6]

TODO: ~~Web links like this should be footnotes for PDF~~

It would take too long to explain the entire method, so I'll just sketch the essential parts here. The complete script for this example is contained in PCA-MonaLisa.R. ...

TODO: Show the necessary parts, including the screeplot.

An image can be imported using `imager::load.image()` which creates a `"cimg"` object, a 4-dimensional array with dimensions named `x,y,z,c`. `x` and `y` are the usual spatial dimensions, `z` is a depth dimension (which would correspond to time in a movie), and `c` is a color dimension containing R, G, B values.

```
library(imager)
img <- imager::load.image(here::here("images", "MonaLisa-BW.jpg"))
dim(img)
#> [1] 640 954   1   1
```

[6] https://kieranhealy.org/blog/archives/2019/10/27/reconstructing-images-using-pca/

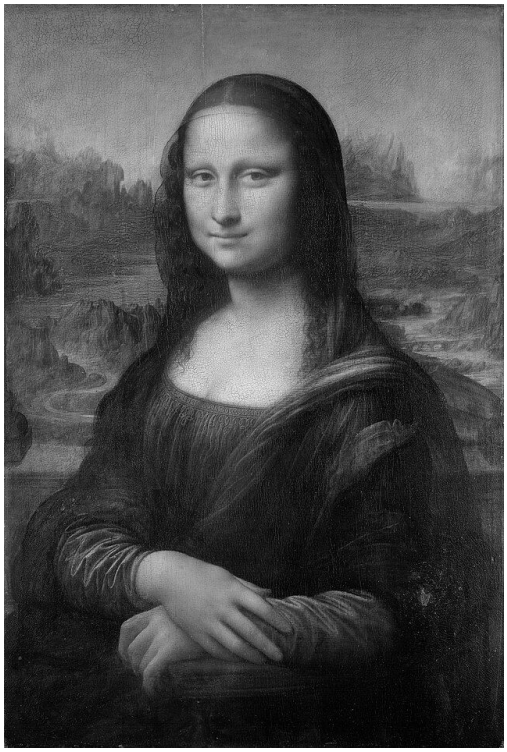


Figure 55: 640 x 954 black and white image of the *Mona Lisa*. Source: Wikipedia

An `as.data.frame()` method converts this to a data frame with `x` and `y` coordinates. Each x-y pair is a location in the 640 by 954 pixel grid, and the `value` is a grayscale value ranging from zero to one.

```
img_df_long <- as.data.frame(img)
head(img_df_long)
#>   x y value
#> 1 1 1 0.431
#> 2 2 1 0.337
#> 3 3 1 0.467
#> 4 4 1 0.337
#> 5 5 1 0.376
#> 6 6 1 0.361
```

However, to do a PCA we will need a matrix of data in wide format containing the grayscale pixel values. We can do this using `tidyr::pivot_wider()`, giving a result with 640 rows and 954 columns.

```
img_df <- pivot_wider(img_df_long,
                      names_from = y,
                      values_from = value) |>
  select(-x)
dim(img_df)
#> [1] 640 954
```

Mona's PCA is produced from this `img_df` with `prcomp()`:

```
img_pca <- img_df |>
  prcomp(scale = TRUE, center = TRUE)
```

With 955 columns, the PCA comprises 955 eigenvalue/eigenvector pairs. However, the rank of a matrix is the smaller of the number of rows and columns, so only 640 eigenvalues can be non-zero. Printing the first 10 shows that the first three dimensions account for 46% of the variance and we only get to 63% with 10 components.

```
img_pca |>
  broom::tidy(matrix = "eigenvalues") |> head(10)
#> # A tibble: 10 x 4
#>       PC std.dev percent cumulative
#>    <dbl>   <dbl>   <dbl>      <dbl>
#>  1     1   14.1  0.209        0.209
#>  2     2   11.6  0.141        0.350
#>  3     3   10.1  0.107        0.457
#>  4     4    7.83 0.0643       0.522
#>  5     5    6.11 0.0392       0.561
#>  6     6    4.75 0.0237       0.585
#>  7     7    3.70 0.0143       0.599
#>  8     8    3.52 0.0130       0.612
#>  9     9    3.12 0.0102       0.622
#> 10    10    2.86 0.00855      0.631
```

Figure 56 shows a screeplot of proportions of variance. Because there are so many components and most of the information is concentrated in the largest dimensions, I've used a $\log_{10}()$ scale on the horizontal axis. Beyond 10 or so dimensions, the variance of additional components looks quite tiny.

```
ggscreeplot(img_pca) +
  scale_x_log10()
```

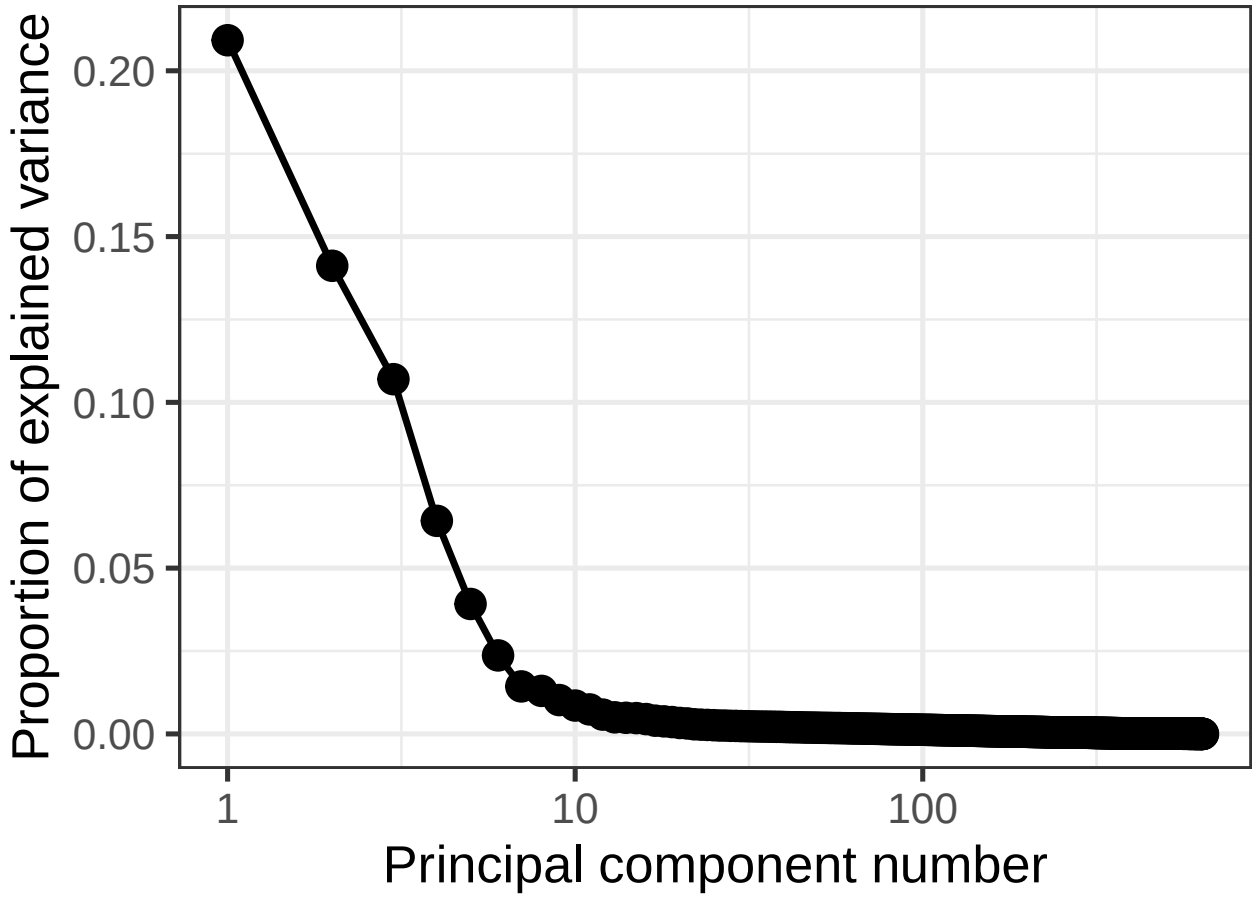


Figure 56: Screeplot of the variance proportions in the Mona Lisa PCA.

Then, if $\mathbf{M}$ is the 640×955 matrix of pixel values, a best approximation $\widehat{\mathbf{M}}_k$ using k dimensions can be obtained as $\widehat{\mathbf{M}}_k = \mathbf{X}_k \, \mathbf{V}_k^T$ where $\mathbf{X}_k$ are the principal component scores and $\mathbf{V}_k$ are the eigenvectors corresponding to the k largest eigenvalues. The function `approx_pca()` does this, and also undoes the scaling and centering carried out in PCA.

TODO: ~~Also, separate approximation from the pivot_longer code...~~

```
approx_pca <- function(n_comp = 20, pca_object = img_pca){
  ## Multiply the matrix of rotated data (component scores) by the transpose of
  ## the matrix of eigenvectors (i.e. the component loadings) to get back to a
  ## matrix of original data values

  recon <- pca_object$x[, 1:n_comp] %*% t(pca_object$rotation[, 1:n_comp])

  ## Reverse any scaling and centering that was done by prcomp()
  if(all(pca_object$scale != FALSE)){
    ## Rescale by the reciprocal of the scaling factor, i.e. back to
    ## original range.
    recon <- scale(recon, center = FALSE, scale = 1/pca_object$scale)
  }
  if(all(pca_object$center != FALSE)){
```

```
    ## Remove any mean centering by adding the subtracted mean back in
    recon <- scale(recon, scale = FALSE, center = -1 * pca_object$center)
  }

  ## Make it a data frame that we can easily pivot to long format
  ## for drawing with ggplot
  recon_df <- data.frame(cbind(1:nrow(recon), recon))
  colnames(recon_df) <- c("x", 1:(ncol(recon_df)-1))

  ## Return the data to long form
  recon_df_long <- recon_df |>
    tidyr::pivot_longer(cols = -x,
                        names_to = "y",
                        values_to = "value") |>
    mutate(y = as.numeric(y)) |>
    arrange(y) |>
    as.data.frame()

  recon_df_long
}
```

Finally, the recovered images, using 2, 3 , 4, 5, 10, 15, 20, 50, and 100 principal components can be plotted using ggplot. In the code below, the `approx_pca()` function is run for each of the 9 values specified by `n_pcs` giving a data frame `recovered_imgs` containing all reconstructed images, with variables `x`, `y` and `value` (the greyscale pixel value).

```
n_pcs <- c(2:5, 10, 15, 20, 50, 100)
names(n_pcs) <- paste("First", n_pcs, "Components", sep = "_")

recovered_imgs <- map_dfr(n_pcs,
                          approx_pca,
                          .id = "pcs") |>
  mutate(pcs = stringr::str_replace_all(pcs, "_", " "),
         pcs = factor(pcs, levels = unique(pcs), ordered = TRUE))
```

In `ggplot()`, each is plotted using `geom_raster()`, using `value` to as the fill color. A quirk of images imported to R is that origin is taken as the upper left corner, so the Y axis scale needs to be reversed. The 9 images are then plotted together using `facet_wrap()`.

```
p <- ggplot(data = recovered_imgs,
            mapping = aes(x = x, y = y, fill = value))
p_out <- p + geom_raster() +
  scale_y_reverse() +
  scale_fill_gradient(low = "black", high = "white") +
  facet_wrap(~ pcs, ncol = 3) +
  guides(fill = "none") +
  labs(title = "Recovering Mona Lisa from PCA of her pixels") +
  theme(strip.text = element_text(face = "bold", size = rel(1.2)),
        plot.title = element_text(size = rel(1.5)))

p_out
```

The result, in Figure 57 is instructive about how much visual information is contained in lower-dimensional reconstructions, or conversely, how much the image can be compressed by omitting the many small dimensions.

In this figure, with 4-5 components most people would recognize this as a blury image of the world's most famous portrait. It is certainly clear that this is the Mona Lisa with 10–15 components. Details of the portrait and backgound features become recognizable with 20–50 components, and with 100 components it compares favorably with the original in Figure 55. In numbers, the original 640×955) image is of size 600 Kb. The 100 component version is only 93 Kb, 15.6% of this.

0.10 Elliptical insights: Outlier detection

The data ellipse (Section 0.1.4), or ellipsoid in more than 2D is fundamental in regression. But also, as Pearson showed, it is key to understanding principal components analysis, where the principal component directions are simply the axes of the ellipsoid of the data. As such, observations that are unusual in data space may not stand out in univariate views of the variables, but will stand out in principal component space, usually on the *smallest* dimension.

As an illustration, I created a dataset of $n = 100$ observations with a linear relation, $y = x + \mathcal{N}(0, 1)$ and then added two discrepant points at (1.5, -1.5), (-1.5, 1.5).

```
set.seed(123345)
x <- c(rnorm(100),             1.5, -1.5)
y <- c(x[1:100] + rnorm(100), -1.5,  1.5)
```

When these are plotted with a data ellipse in Figure 58 (left), you can see the discrepant points labeled 101 and 102, but they do not stand out as unusual on either x or y. The transformation to from data space to principal components space, shown in Figure 58 (right), is simply a rotation of (x, y) to a space whose coordinate axes are the major and minor axes of the data ellipse, (PC_1, PC_2). In this view, the additional points appear a univariate outliers on the smallest dimension, PC_2.

To see this more clearly, Figure 59 shows an animation of the rotation from data space to PCA space. This uses `heplots::interpPlot()` ...

Packages used here: 18 packages used here: base, corrplot, datasets, dplyr, factoextra, FactoMineR, ggbiplot, ggplot2, graphics, grDevices, imager, knitr, magrittr, methods, patchwork, stats, tidyr, utils

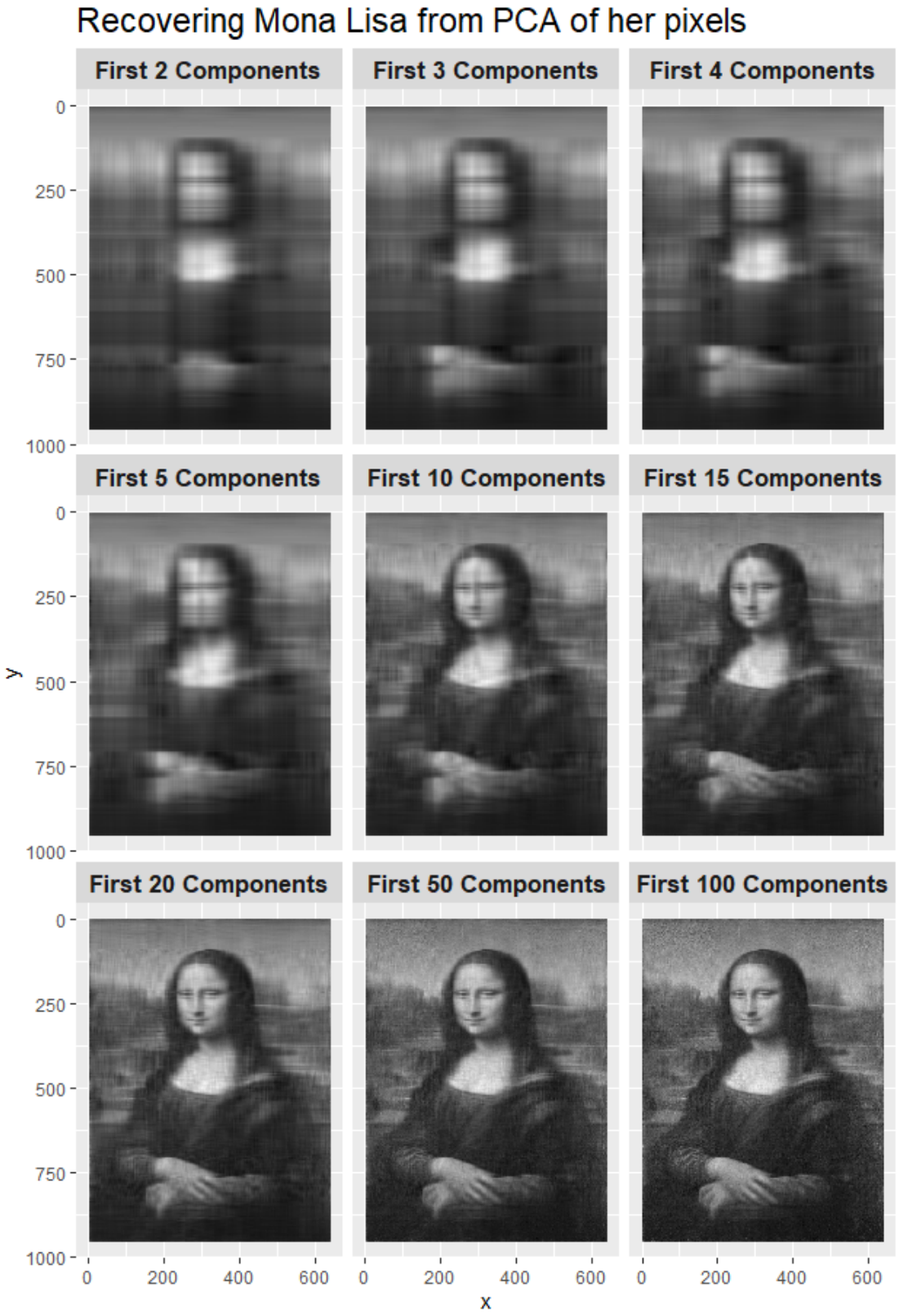


Figure 57: Re-construction of the Mona Lisa using 2, 3 , 4, 5, 10, 15, 20, 50, and 100 principal components.

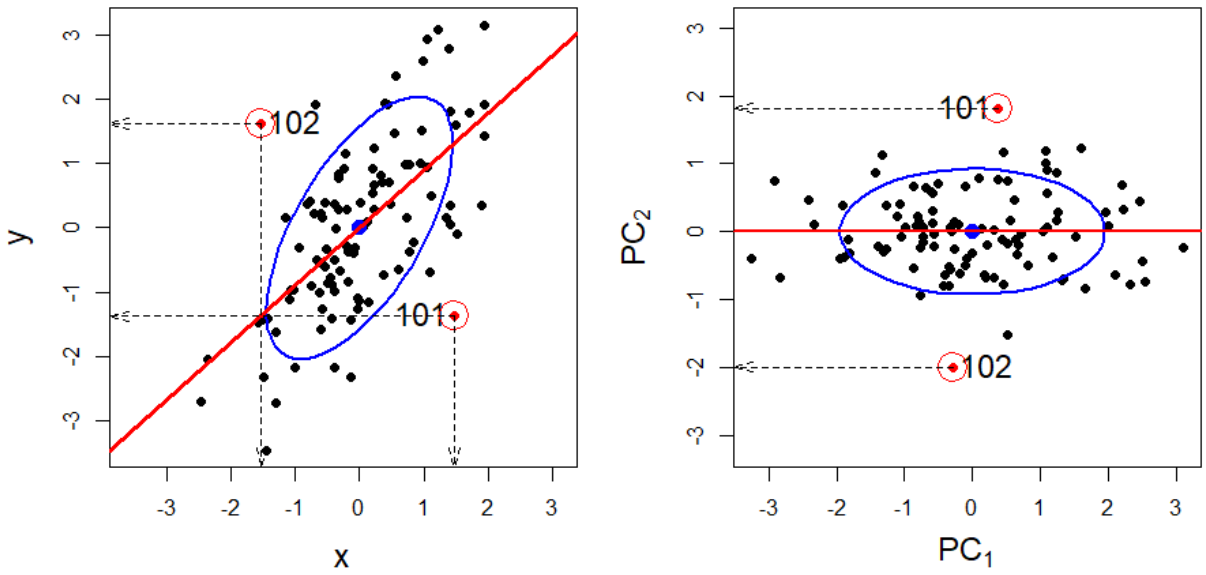


Figure 58: Outlier demonstration: The left panel shows the original data and highlights the two discrepant points, which do not appear to be unusual on either x or y. The right panel shows the data rotated to principal components, where the labeled points stand out on the smallest PCA dimension.

Figure 59: Animation of rotation from data space to PCA space.

0

Overview of Linear models

Although this book is primarily about multivariate models, it is useful to have an overview of the range of available techniques to see their relations and to appreciate how easily univariate response models generalize to multivariate ones. Hence, this chapter reviews the characteristics of the standard univariate methods for explaining or predicting a single outcome variable from a set of predictors.

The key ideas are:

- For a single quantitative outcome variable $\mathbf{y}$, methods of linear regression and analysis of variance (ANOVA) are comprised within a single framework of the **general linear model** (GLM). Regression and ANOVA differ only in that the predictors in the former are quantitative, while those in the latter are discrete factors. They can all be fit using `lm()`.
- These models extend directly to the multivariate case of $q > 1$ outcomes, $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_q)$, and are also fit using `lm()`.
- A binary outcome $y = (0, 1)$ and categorical outcomes like marital status ("never married", "married", "separated", "divorced") can be handled within a different extension, the **generalized** linear model as logistic regression or multinomial regression, fit using `glm()`.
- All of these models involve linear combinations of predictors (weighted sums) fit to optimize some criterion, for example minimizing some function of the residuals or maximizing some measure of fit.
- Models and data can be more easily understood with graphics, and many statistical ideas have a visual representation in geometry.

Figure 60 summarizes a variety of methods for linear models, classified by number of predictors and number of response variables, and whether these are quantitative vs. discrete. For the present purposes, the key columns are the first two, for the case of one or more quantitative outcome variables.

when the predictors are also quantitative, simple regression ($p = 1$) generalizes to multivariate regression with two or more outcomes, for example predicting weight and body mass index jointly from a person's height. The situation is more interesting when there are $p > 1$ predictors. The most common multivariate generalization is multivariate multiple regression (MMRA), where each outcome is regressed on the predictors, as if done separately for each outcome, but using multivariate tests that take correlations among the predictors into account. Other methods for this case include canonical correlation analysis, which tries to explain all relations between $\mathbf{Y}$ and a set of $\mathbf{x}$s through maximally correlated linear combinations of each.

When the predictor variables are all categorical, like gender or level of education, methods like the simple t-test, one-way ANOVA and factorial ANOVA with $q = 1$ outcome measures all have simple extensions to the case of $q > 1$ outcomes.

Not shown in Figure 60 are rows for cases where predictor variables include both quantitative and discrete variables -> ANCOVA, Homogeneity of regression ...

::: {.callout-note title = "History Corner"} Why are there so many different names for regression vs. ANOVA techniques? ... :::

Response variables: $\mathbf{Y} = (y_1, \ldots y_q)$

Predictor variables: $\mathbf{X} = (x_1, \ldots x_p)$

		Quantitative		Discrete	
		q=1	q>1	q=1	q>1
Quantitative	p=1	Simple regression	Multivariate regression	Simple logistic regression	
	p>1	Multiple regression	Multivariate regression Canonical corr. Partial corr.	Mult. logistic regression Discriminant analysis	Multivariate logistic regression
Discrete	p=1	t-test 1-way ANOVA	Hotelling T^2 1-way MANOVA	Simple χ^2	Loglinear models
	p>1	Factorial ANOVA	Factorial MANOVA	Logit models Loglinear models	

Figure 60: Techniques for linear models classified by number of predictors and number of response variables, and whether these are quantitative vs. discrete

0.11 Linear combinations

All methods of multivariate statistics involve a simple idea: Finding weighted sums—*linear combinations*— of observed variables to optimize some criterion—maximizing a measure of goodness-of-fit, like R^2 or minimizing a measure of badness-of-fit like sums of squares of residuals. Methods differ according to whether:

- All variables belong to **one set** (say, $\mathbf{X}$), not distinguished as to whether they are responses or predictors, as in PCA and factor analysis, vs. **two sets** where one set is considered outcome, dependent variables, to be explained by predictors, independent variables ($\mathbf{X}$), as in multiple regression, multivariate analysis of variance, discriminant analysis and canonical correlation analysis.

- The variables in $\mathbf{X}$ and $\mathbf{Y}$ are discrete, **categorical factors** like sex and level of education or **quantitative** variables like salary and number of years of experience.

For example, Figure 61 illustrates PCA (as we saw in Chapter 0.4) as finding weights to maximize the variance of linear combinations, $v_1, v_2, \ldots$,

$$\begin{aligned} \mathbf{v}_1 &= a_1\mathbf{x}_1 + a_2\mathbf{x}_2 + a_3\mathbf{x}_3 + a_4\mathbf{x}_4 \\ \mathbf{v}_2 &= b_1\mathbf{x}_1 + b_2\mathbf{x}_2 + b_3\mathbf{x}_3 + b_4\mathbf{x}_4 \\ \ldots &\quad \ldots \ , \end{aligned}$$

subject to all $\mathbf{v}_i, \mathbf{v}_j$ being uncorrelated, $\mathbf{v}_i \perp \mathbf{v}_j$.

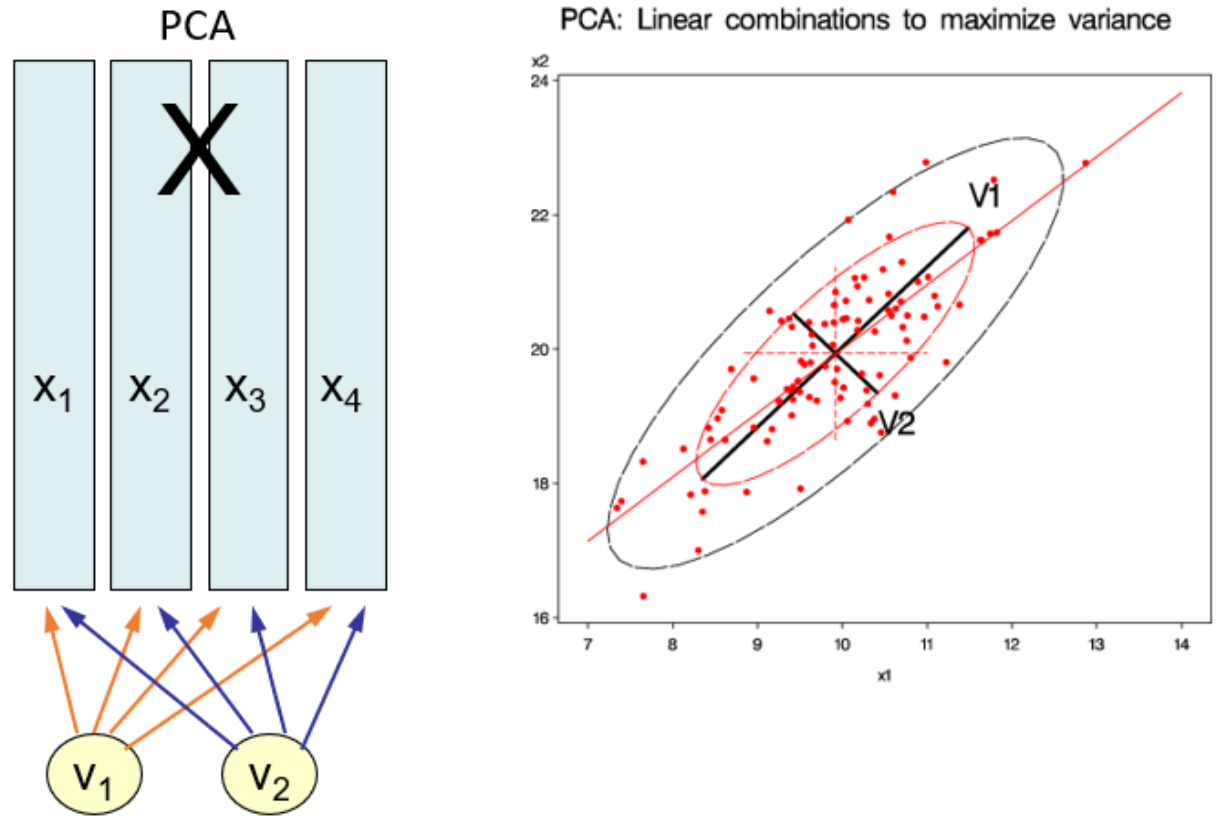


Figure 61: Principal components analysis as linear combinations to maximize variance accounted for. Left: diagram of PCA showing two uncorrelated linear combinations, v1 and v2. Right: Geometry of PCA.

0.11.1 Multiple regression

0.11.2 Multivariate regression

0.11.3 Canonical correlation analysis

0.11.4 The General Linear Model

0.11.5 Model formulas

0.12 Regression

0.13 ANOVA

0.14 ANCOVA

0.15 Regression trees

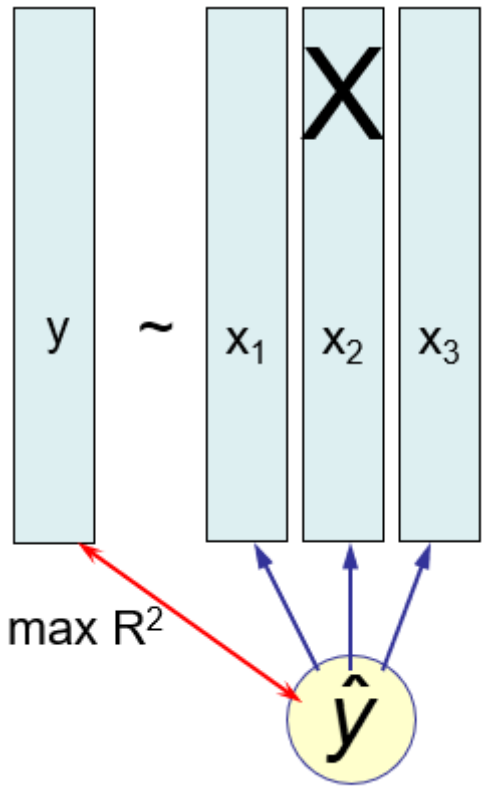


Figure 62: Multiple regression as linear combinations to maximize R squared. ... **TODO** Add vector diagram

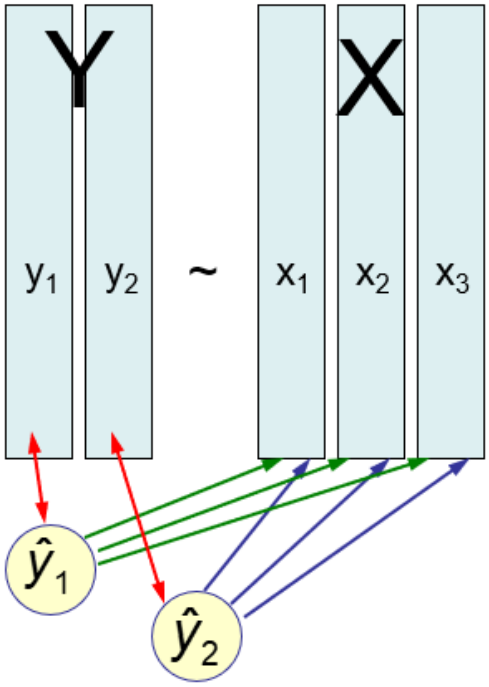


Figure 63: Multivariate multiple regression as linear combinations to maximize R squared. ... **TODO** Add vector diagram

0

Plots for univariate response models

For a univariate linear model fit using `lm()`, `glm()` and similar functions, the standard `plot()` method gives basic versions of *diagnostic* plots of residuals and other calculated quantities for assessing possible violations of the model assumptions. Some of these can be considerably enhanced using other packages.

Beyond this,

- tables of model coefficients, standard errors and test statistics can often be usefully supplemented or even replaced by suitable plots providing essentially the same information.
- when there are two or more predictors, you can more easily understand their separate impact on the response by plotting the *marginal* effects of one or more focal variables, averaging over other variables not shown in a given plot.
- when there are highly correlated predictors, some specialized plots are useful to understand the nature of *multicolinearity*.

The classic reference on regression diagnostics is Belsley et al. (1980). My favorite modern texts are the brief Fox (2020) and the more complete Fox & Weisberg (2018), both of which are supported by the **car** package (Fox et al., 2023).

0.16 The "regression quartet"

For a fitted model, plotting the model object with `plot(model)` provides for any of six basic plots, of which four are produced by default, giving rise to the term *regression quartet* for this collection. These are:

- **Residuals vs. Fitted**: For well-behaved data, the points should hover around a horizontal line at residual = 0, with no obvious pattern or trend.
- **Normal Q-Q plot**: A plot of sorted standardized residuals e_i (obtained from`rstudent(model)`) against the theoretical values those values would have in a standard normal $\mathcal{N}(0, 1)$ distribution.
- **Scale-Location**: Plots the square-root of the absolute values of the standardized residuals $\sqrt{|e_i|}$ as a measure of "scale" against the fitted values $\hat{y}_i$ as a measure of "location". This provides an assessment of homogeneity of variance, which appears as a tendency for scale to vary with location.
- **Residuals vs. Leverage**: Plots standardized residuals against leverage to help identify possibly influential observations. Leverage, or "hat" values (given by `hat(model)`) are proportional to the squared Mahalanobis distances of the predictor values $\mathbf{x}_i$ from the means, and measure the potential of an observation to change the fitted coefficients if that observation was deleted. Actual influence is measured by Cooks's distance (`cooks.distance(model)`) and is proportional to the product of residual times leverage. Contours of constant Cook's D are added to the plot.

One key feature of these plots is providing **reference** lines or smoothed curves for ease of judging the extent to which a plot conforms to the expected pattern; another is the **labeling** of observations which deviate from an assumption.

The base-R `plot(model)` plots are done much better in a variety of packages. I illustrate some versions from the **car** (Fox et al., 2023) and **performance** (Lüdecke et al., 2021) packages, part of the **easystats** (Lüdecke et al., 2022) suite of packages.

Packages:

```
library(car)
library(easystats)
```

Example: Duncan's occupational prestige

In a classic study in sociology, Duncan (1961) used data from the U.S. Census in 1950 to study how one could predict the prestige of occupational categories — which is hard to measure — from available information in the census for those occupations. His data is available in `carData:Duncan`, and contains

- `type`: the category of occupation, one of `prof` (professional), `wc` (white collar) or `bc` (blue collar);
- `income`: the percentage of occupational incumbents with a reported income > $3500 (about $40,000 in current dollars);
- `education`: the percentage of occupational incumbents who were high school graduates;
- `prestige`: the percentage of respondents in a social survey who rated the occupation as "good" or better in prestige.

These variables are a bit quirky in they are measured in percents, 0-100, rather dollars for `income` and years for `education`, but this common scale permitted Duncan to ask an interesting sociological question: Assuming that both income and education predict prestige, are they equally important, as might be assessed by testing the hypothesis $H_0 : \beta_{\text{income}} = \beta_{\text{education}}$.

A quick look at the data shows the variables and a selection of the occupational categories, which are the `row.names()` of the dataset.

```
data(Duncan, package = "carData")
set.seed(42)
car::some(Duncan)
#>                  type income education prestige
#> accountant       prof     62        86       82
#> professor        prof     64        93       93
#> engineer         prof     72        86       88
#> factory.owner    prof     60        56       81
#> store.clerk        wc     29        50       16
#> carpenter          bc     21        23       33
#> machine.operator   bc     21        20       24
#> barber             bc     16        26       20
#> soda.clerk         bc     12        30        6
#> janitor            bc      7        20        8
```

Let's start by fitting a simple model using just income and education as predictors. The results look very good! Both `income` and `education` are highly significant and the $R^2 = 0.828$ for the model indicates that `prestige` is very well predicted by just these variables.

```
duncan.mod <- lm(prestige ~ income + education, data=Duncan)
summary(duncan.mod)
#>
#> Call:
#> lm(formula = prestige ~ income + education, data = Duncan)
```

```
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -29.54  -6.42   0.65   6.61  34.64
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)  -6.0647     4.2719   -1.42     0.16
#> income        0.5987     0.1197    5.00  1.1e-05 ***
#> education     0.5458     0.0983    5.56  1.7e-06 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 13.4 on 42 degrees of freedom
#> Multiple R-squared:  0.828,  Adjusted R-squared:  0.82
#> F-statistic:  101 on 2 and 42 DF,  p-value: <2e-16
```

Beyond this, Duncan was interested in the coefficients and whether income and education could be said to have equal impacts on predicting occupational prestige. A nice display of model coefficients with confidence intervals is provided by `parameters::model_parameters()` and we can test Duncan's hypothesis with `car::linearHypothesis()`. The latter is constructed as a test of a restricted model in which the two coefficients are forced to be equal against the unrestricted model. Duncan was very happy with this result.

```
parameters::model_parameters(duncan.mod)
#> Parameter   | Coefficient |   SE |         95% CI | t(42) |      p
#> ------------------------------------------------------------------
#> (Intercept) |       -6.06 | 4.27 | [-14.69, 2.56] | -1.42 | 0.163
#> income      |        0.60 | 0.12 | [  0.36, 0.84] |  5.00 | < .001
#> education   |        0.55 | 0.10 | [  0.35, 0.74] |  5.56 | < .001

car::linearHypothesis(duncan.mod, "income = education")
#> Linear hypothesis test
#>
#> Hypothesis:
#> income - education = 0
#>
#> Model 1: restricted model
#> Model 2: prestige ~ income + education
#>
#>   Res.Df  RSS Df Sum of Sq    F Pr(>F)
#> 1     43 7519
#> 2     42 7507  1      12.2 0.07    0.8
```

But, should Duncan be **so** happy? It is unlikely that he ran any model diagnostics or plotted his model; we do so now. Here is the regression quartet for this model. Each plot shows some trend lines, and importantly, labels some observations that stand out and might deserve attention.

```
op <- par(mfrow = c(2,2),
          mar = c(4,4,3,1)+.1)
plot(duncan.mod, lwd=2, pch=16)
par(op)
```

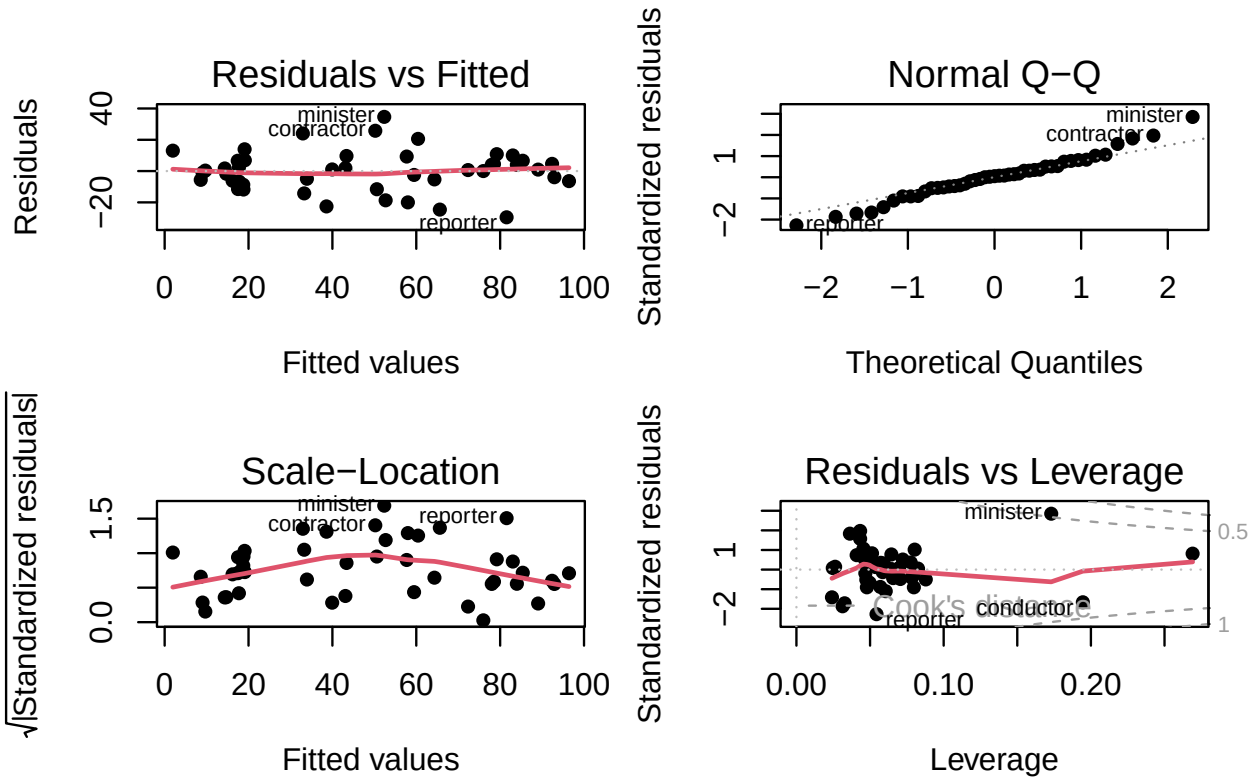


Figure 64: Regression quartet of diagnostic plots for the `Duncan` data. Several possibly unusual observations are labeled.

Example: Occupational prestige

CUT THIS EXAMPLE

These examples use the data on the prestige of 102 occupational categories and other measures from the 1971 Canadian Census, recorded in `carData::Prestige`. Our interest is in understanding how `prestige` (the Pineo-Ported prestige score, from a social survey) is related to census measures of the average education, income, percent women of incumbents in those occupations. Occupation type is a factor with levels `"bc"` (blue collar), `"wc"` (white collar) and `"prof"` (professional). **TODO**: These data should be introduced earlier with descriptive plots, scatterplots, ...

```
data(Prestige, package="carData")
# `type` is really an ordered factor. Make it so.
Prestige$type <- ordered(Prestige$type,
                         levels=c("bc", "wc", "prof"))
str(Prestige)
#> 'data.frame':    102 obs. of  6 variables:
#>  $ education: num  13.1 12.3 12.8 11.4 14.6 ...
#>  $ income   : int  12351 25879 9271 8865 8403 11030 8258 14163 11377 11023 ...
#>  $ women    : num  11.16 4.02 15.7 9.11 11.68 ...
#>  $ prestige : num  68.8 69.1 63.4 56.8 73.5 77.6 72.6 78.1 73.1 68.8 ...
#>  $ census   : int  1113 1130 1171 1175 2111 2113 2133 2141 2143 2153 ...
#>  $ type     : Ord.factor w/ 3 levels "bc"<"wc"<"prof": 3 3 3 3 3 3 3 3 3 3 ...
```

We fit a main-effects model using all predictors (ignoring `census`, the Canadian Census occupational code):

```
prestige.mod <- lm(prestige ~ education + income + women + type,
                   data=Prestige)
```

`plot(model)` produces four separate plots. For a quick look, I like to arrange them in a single 2x2 figure.

```
op <- par(mfrow = c(2,2),
          mar=c(4,4,3,1)+.1)
plot(prestige.mod, lwd=2, cex.lab=1.4)
par(op)
```

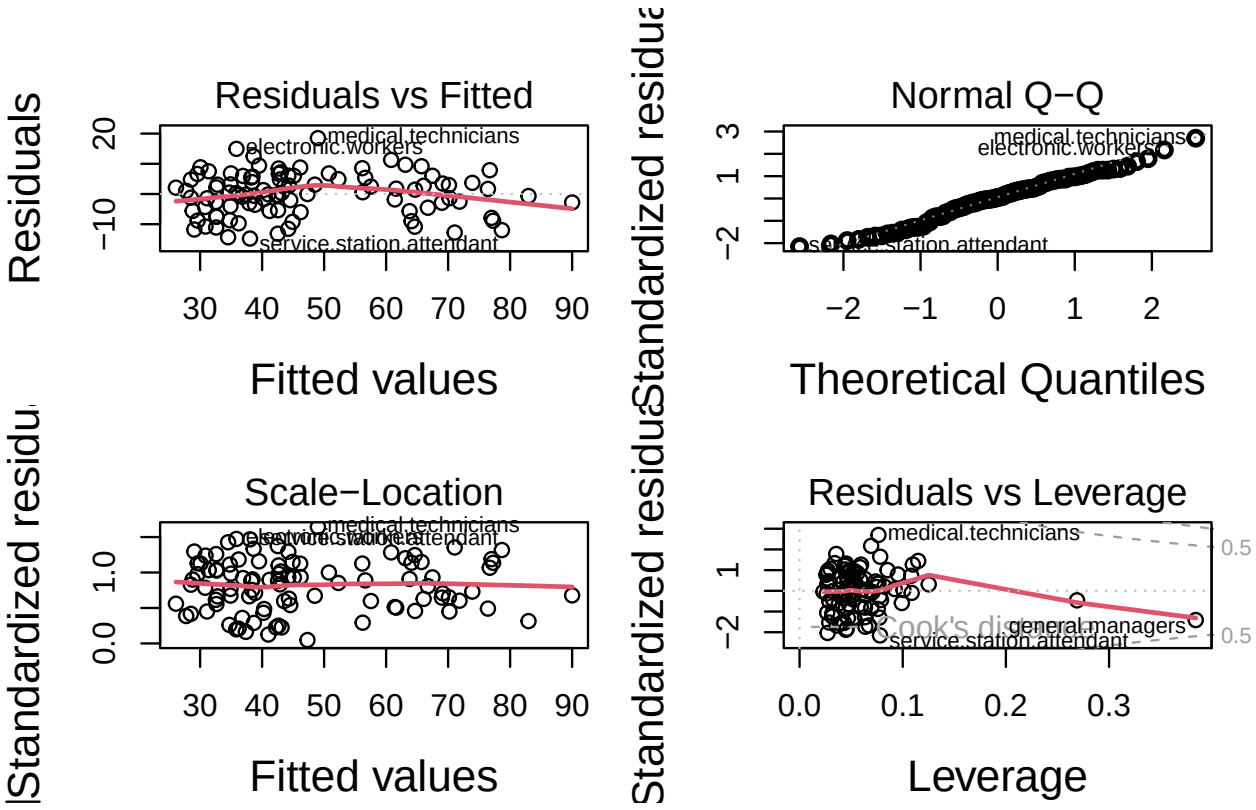


Figure 65: Regression quartet of diagnostic plots for the `Prestige` data. Several possibly unusual observations are labeled.

0.17 Other Diagnostic plots

0.17.1 Spread-level plot

0.18 Coefficient plots

0.19 Added-variable plots

0.20 Marginal plots

0.21 Outliers, leverage and influence

In small to moderate samples, "unusual" observations can have dramatic effects on a fitted regression model, as we saw in the analysis of Davis's data on reported and measured weight (**?@sec-davis**) where one erroneous observations hugely altered the fitted line.

An observation can be unusual in three archetypal ways, with different consequences:

- Unusual in the response y, but typical in the predictor(s), $\mathbf{x}$ — a badly fitted case with a large absolute residual, but with x not far from the mean, as in Figure 6. This case does not do much harm to the fitted model.
- Unusual in the predictor(s) $\mathbf{x}$, but typical in y — an otherwise well-fitted point. This case also does litle harm, and in fact can be considered to improve precision, a "good leverage" point.
- Unusual in **both** $\mathbf{x}$ and y — This is the case, a "bad leverage" point, revealed in the analysis of Davis's data, Figure 5, where the one erroneous point for women was highly influential, pulling the regression line towards it and affecting the estimated coefficient as well as all the fitted values.

Influential cases are the ones that matter most. As suggested above, to be influential an observation must be unusual in **both** $\mathbf{x}$ and y, and affects the estimated coefficients, thereby also altering the predicted values for all observations. A heuristic formula capturing the relations among leverage, "outlyingness" on y and influence is

$$\text{Influence}_{\text{coefficients}} \;=\; X_{\text{leverage}} \;\times\; Y_{\text{residual}}$$

As described below, leverage is proportional to the squared distance $(x_i - \bar{x})^2$ of an observation x_i from its mean in simple regression and to the squared Mahalanobis distance in the general case. The Y_{residual} is best measured by a *studentized* residual, obtained by omitting each case i in turn and calculating its residual from the coefficients obtained from the remaining cases.

0.21.1 The leverage-influence quartet

These ideas can be illustrated in the "leverage-influence quartet" by considering a standard simple linear regression for a sample and then adding one additional point reflecting the three situations described above. Below, I generate a sample of $N = 15$ points with x uniformly distributed between (40, 60) and $y \sim 10 + 0.75x + \mathcal{N}(0, 1.25^2)$, duplicated four times.

```
library(tidyverse)
library(car)
set.seed(42)
N <- 15
case_labels <- paste(1:4, c("OK", "Outlier", "Leverage", "Influence"))
levdemo <- tibble(
    case = rep(case_labels,
               each = N),
    x = rep(round(40 + 20 * runif(N), 1), 4),
    y = rep(round(10 + .75 * x + rnorm(N, 0, 1.25), 4)),
    id = " "
)

mod <- lm(y ~ x, data=levdemo)
coef(mod)
#> (Intercept)           x
#>      13.332       0.697
```

The additional points, one for each situation are set to the values below.

- `Outlier`: (52, 60) a low leverage point, but an outlier (`O`) with a large residual
- `Leverage`: (75, 65) a "good" high leverage point (`L`) that fits well with the regression line
- `Influence`: (70, 40) a "bad" high leverage point (`OL`) with a large residual.

```
extra <- tibble(
  case = case_labels,
  x  = c(65, 52, 75, 70),
  y  = c(NA, 65, 65, 40),
  id = c("  ", "O", "L", "OL")
)

#' Join these to the data
both <- bind_rows(levdemo, extra) |>
  mutate(case = factor(case))
```

We can plot these four situations with `ggplot2` in panels faceted by `case` as shown below. The standard version of this plot shows the regression line for the original data and that for the ammended data with the additional point. Note that we use the `levdemo` dataset in `geom_smooth()` for the regression line with the original data, but specify `data = both` for that with the additional point.

```
ggplot(levdemo, aes(x = x, y = y)) +
  geom_point(color = "blue", size = 2) +
  geom_smooth(data = both,
              method = "lm", formula = y ~ x, se = FALSE,
              color = "red", linewidth = 1.3, linetype = 1) +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE,
              color = "blue", linewidth = 1, linetype = "longdash" ) +
  stat_ellipse(data = both, level = 0.5, color="blue", type="norm", linewidth = 1.4) +
  geom_point(data=extra, color = "red", size = 4) +
  geom_text(data=extra, aes(label = id), nudge_x = -2, size = 5) +
  facet_wrap(~case, labeller = label_both) +
  theme_bw(base_size = 14)
```

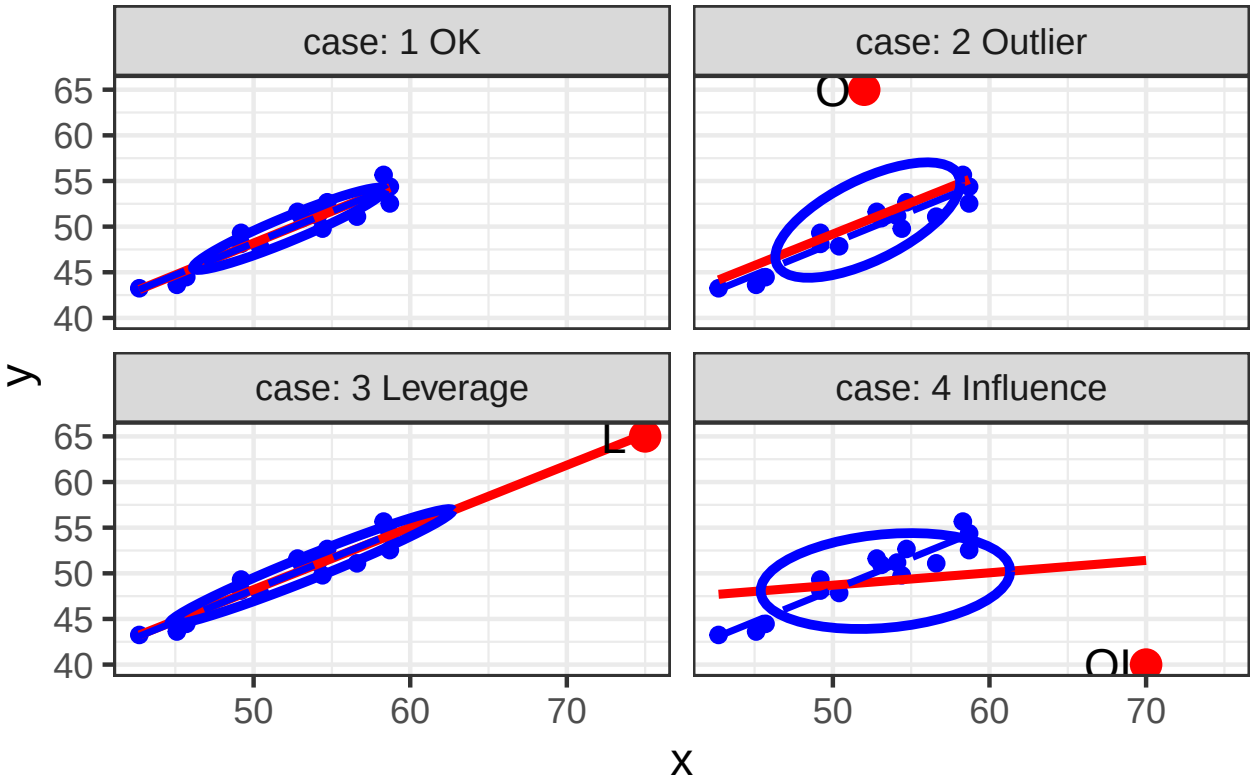


Figure 66: Leverage influence quartet with data 50% ellipses. Case (1) original data; (2) adding one low-leverage outlier, "O"; (3) adding one "good" leverage point, "L"; (4) adding one "bad" leverage point, "OL". The dashed line is the fitted line for the original data, while the solid line reflects the additional point. The data ellipses show the effect of the additional point on precision.

The standard version of this graph shows only the fitted regression lines in each panel. As can be seen, the fitted line doesn't change very much in panels (2) and (3); only the bad leverage point, "OL" in panel (4) is harmful. Adding data ellipses to each panel immediately makes it clear that there is another part to this story— the effect of the unusual point on *precision* (standard errors) of our estimates of the coefficients.

Now, we see *directly* that there is a big difference in impact between the low-leverage outlier [panel (2)] and the high-leverage, small-residual case [panel (3)], even though their effect on coefficient estimates is negligible. In panel (2), the single outlier inflates the estimate of residual variance (the size of the vertical slice of the data ellipse at $\bar{x}$), while in panel (3) this is decreased.

To allow direct comparison and make the added value of the data ellipse more apparent, we overlay the data ellipses from Figure 66 in a single graph, shown in Figure 67.
Here, we can also see why the high-leverage point "L" added in panel (c) of 66 is called a "good leverage" point. By increasing the standard deviation of x, it makes the data ellipse somewhat more elongated, giving increased precision of our estimates of $\vec{\beta}$.

```
colors <- c("black", "blue", "darkgreen", "red")
with(both,
     {dataEllipse(x, y, groups = case,
          levels = 0.68,
          plot.points = FALSE, add = FALSE,
          center.pch = "+",
          col = colors,
```

```
            fill = TRUE, fill.alpha = 0.1)
      })

case1 <- both |> filter(case == "1 OK")
points(case1[, c("x", "y")], cex=1)

points(extra[, c("x", "y")],
       col = colors,
       pch = 16, cex = 2)

text(extra[, c("x", "y")],
     labels = extra$id,
     col = colors, pos = 2, offset = 0.5)
```

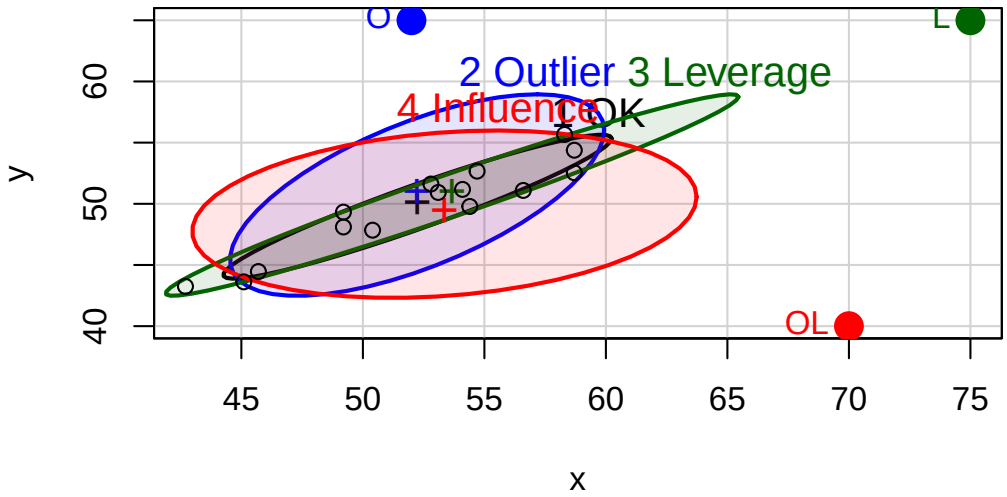


Figure 67: Data ellipses in the Leverage-influence quartet. This graph overlays the data ellipses and additional points from the four panels of Figure 67. It can be seen that only the OL point affects the slope, while the O and L points affect precision of the estimates in opposite directions.

0.21.2 Measuring leverage

Leverage is thus an index of the *potential* impact of an observation on the model due to its' atypical value in the X space of the predictor(s). It is commonly measured by the "hat" value, h_i, so called because it puts the hat ($\hat{\bullet}$) on $\mathbf{y}$, i.e., the vector of fitted values can be expressed as

$$\begin{aligned} \hat{\mathbf{y}} &= \mathbf{H}\mathbf{y} \\ &= [\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T]\,\mathbf{y}\,, \end{aligned}$$

where $h_i \equiv h_{ii}$ are the diagonal elements of the Hat matrix $\mathbf{H}$. In simple regression, hat values are proportional to the squared distance of the observation x_i from the mean, $h_i \propto (x_i - \bar{x})^2$,

$$h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\Sigma_i (x_i - \bar{x})^2}\,,$$

and range from $1/n$ to 1, with an average value $\bar{h} = 2/n$. Consequently, observations with h_i greater than $2\bar{h}$ or $3\bar{h}$ are commonly considered to be of high leverage.

With $p \geq 2$ predictors, it is demonstrated below that $h_i \propto D^2(\mathbf{x} - \bar{\mathbf{x}})$, the squared distance of $\mathbf{x}$ from the centroid $\bar{\mathbf{x}}$[7]. [8]: See this Stats StackExchange discussion for a proof. The analogous formula is

$$h_i = \frac{1}{n} + \frac{1}{n-1} D^2(\mathbf{x} - \bar{\mathbf{x}}) \ ,$$

where $D^2(\mathbf{x} - \bar{\mathbf{x}}) = (\mathbf{x} - \bar{\mathbf{x}})^T \mathbf{S}_X^{-1} (\mathbf{x} - \bar{\mathbf{x}})$. From Section 0.1.4, it follows that contours of constant leverage correspond to data ellipses or ellipsoids of the predictors in $\mathbf{x}$, whose boundaries, assuming normality, correspond to quantiles of the χ^2_p distribution

Example:

To illustrate, I generate $N = 100$ points from a bivariate normal distribution with means $\mu = (30, 30)$, variances = 10, and a correlation $\rho = 0.7$ and add two noteworthy points that show an apparently paradoxical result.

```
set.seed(421)
N <- 100
r <- 0.7
mu <- c(30, 30)
cov <- matrix(c(10,   10*r,
                10*r, 10), ncol=2)

X <- MASS::mvrnorm(N, mu, cov) |> as.data.frame()
colnames(X) <- c("x1", "x2")

# add 2 points
X <- rbind(X,
           data.frame(x1 = c(28, 38),
                      x2 = c(42, 35)))
```

The Mahalanobis squared distances of these points can be calculated using `heplots::Mahalanobis()`, and their corresponding hatvalues found using `hatvalues()` for any linear model using both `x1` and `x2`.

```
X <- X |>
  mutate(Dsq = heplots::Mahalanobis(X)) |>
  mutate(y = 2*x1 + 3*x2 + rnorm(nrow(X), 0, 5),
         hat = hatvalues(lm(y ~ x1 + x2)))
```

Plotting `x1` and `x2` with data ellipses shows the relation of leverage to squared distance from the mean. The blue point looks to be farther from the mean, but the red point is actually very much further by Mahalanobis squared distance, which takes the correlation into account; it thus has much greater leverage.

```
par(mar = c(4, 4, 1, 1) + 0.1)
dataEllipse(X$x1, X$x2,
            levels = c(0.40, 0.68, 0.95),
            fill = TRUE, fill.alpha = 0.05,
            col = "darkgreen",
```

[7] If group sizes are greatly unequal **and** homogeneity of variance is violated, then the F statistic is too liberal (p values too large) when large sample variances are associated with small group sizes. Conversely, the F statistic is too conservative if large variances are associated with large group sizes.

[8] If group sizes are greatly unequal **and** homogeneity of variance is violated, then the F statistic is too liberal (p values too large) when large sample variances are associated with small group sizes. Conversely, the F statistic is too conservative if large variances are associated with large group sizes.

```
            xlab = "X1", ylab = "X2")
points(X[1:nrow(X) > N, 1:2], pch = 16, col=c("red", "blue"), cex = 2)
X |> slice_tail(n = 2) |>      # last two rows
  points(pch = 16, col=c("red", "blue"), cex = 2)
```

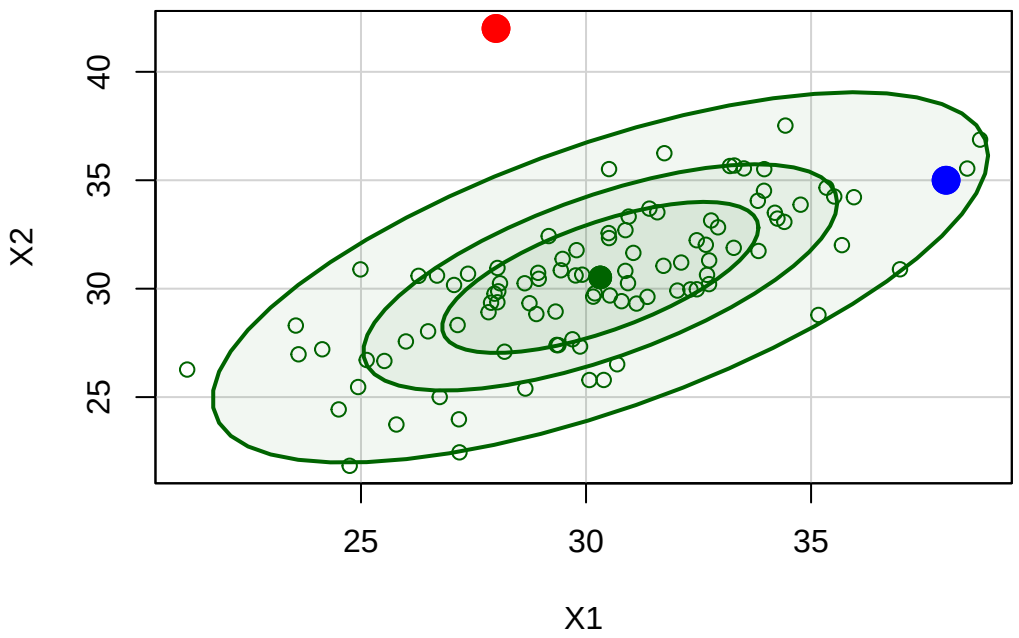


Figure 68: Data ellipses for a bivariate normal sample with correlation 0.7, and two additional noteworthy points. The blue point looks to be farther from the mean, but the red point is actually more than 5 times further by Mahalanobis squared distance, and thus has much greater leverage.

The fact that hatvalues are proportional to leverage can be seen by plotting one against the other. I highlight the two noteworthy points in their colors from Figure 68 to illustrate how much greater leverage the red point has compared to the blue point.

```
plot(hat ~ Dsq, data = X,
     cex = c(rep(1, N), rep(2, 2)),
     col = c(rep("black", N), "red", "blue"),
     pch = 16,
     ylab = "Hatvalue",
     xlab = "Mahalanobis Dsq")
```

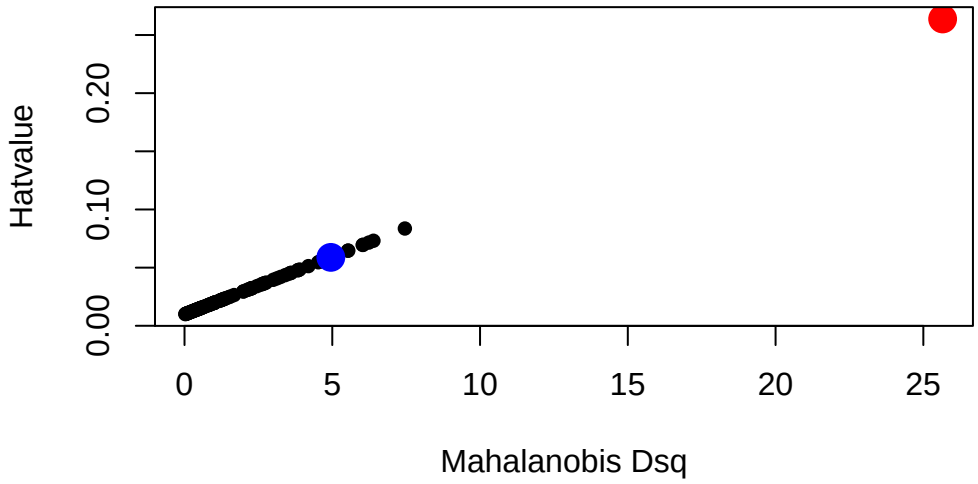


Figure 69: Hat values are proportional to squared Mahalanobis distances from the mean.

Look back at these two points in Figure 68. Can you guess how much further the red point is from the mean than the blue point? You might be surprised that its' D^2 and leverage are about five times as great!

```
X |> slice_tail(n=2)
#>   x1 x2   Dsq   y    hat
#> 1 28 42 25.65 179 0.2638
#> 2 38 35  4.95 175 0.0588
```

0.21.3 Outliers: Measuring residuals

From the discussion in Section 0.21, outliers for the response y are those observations for which the residual $e_i = y_i - \hat{y}_i$ are unusually large in magnitude. However, as demonstrated in Figure 66, a high-leverage point will pull the fitted line towards it, reducing its' residual and thus making them look less unusual.

The standard approach (Cook & Weisberg, 1982; Hoaglin & Welsch, 1978) is to consider a *deleted residual* $e_{(-i)}$, conceptually as that obtained by re-fitting the model with observation i omitted and obtaining the fitted value $\hat{y}_{(-i)}$ from the remaining $n-1$ observations,

$$e_{(-i)} = y_i - \hat{y}_{(-i)} \; .$$

The (externally) *studentized residual* is then obtained by dividing $e_{(-i)}$ by it's estimated standard error, giving

$$e^{\star}_{(-i)} = \frac{e_{(-i)}}{\text{sd}(e_{(-i)})} = \frac{e_i}{\sqrt{\text{MSE}_{(-i)}\,(1-h_i)}} \; .$$

This is just the ordinary residual e_i divided by a factor that increases with the residual variance but decreases with leverage. It can be shown that these studentized residuals follow a t distribution with $n-p-2$ degrees of freedom, so a value $|e^{\star}_{(-i)}| > 2$ can be considered large enough to pay attention to.
In practice for classical linear models, it is unnecessary to actually re-fit the model n times ...

0.21.4 Measuring influence

As described at the start of this section, the actual influence of a given case depends multiplicatively on its' leverage and residual. But how can we measure it?

The essential idea introduced above, is to delete the observations one at a time, each time refitting the regression model on the remaining n˘1 observations. Then, for observation i compare the results using all n

observations to those with the i^{th} observation deleted to see how much influence the observation has on the analysis.

The simplest such measure, called DFFITS, compares the predicted value for case i with what would be obtained when that observation is excluded.

$$\begin{aligned} \text{DFFITS}_i &= \frac{\hat{y}_i - \hat{y}_{(-i)}}{\sqrt{\text{MSE}_{(-i)}\, h_i}} \\ &= e^{\star}_{(-i)} \times \sqrt{\frac{h_i}{1-h_i}} \ . \end{aligned}$$

The first equation gives the signed difference in fitted values in units of the standard deviation of that difference weighted by leverage; the second version (Belsley et al., 1980) represents that as a product of residual and leverage. A rule of thumb is that an observation is deemed to be influential if $|\text{DFFITS}_i| > 2\sqrt{(p+1)/n}$.

```
#> Writing packages to  C:/R/Projects/Vis-MLM-book/bib/pkgs.txt
#> 31  packages used here:
#>  base, bayestestR, car, carData, correlation, datasets, datawizard, dplyr, easystats, effectsize, forc
```

0

Collinearity & Ridge Regression

> Some of my collinearity diagnostics have large values, or small values, or whatever they are not supposed to be * What is bad? * If bad, what can I do about it?

In univariate multiple regression models, we usually hope to have high correlations between the outcome y and each of the predictors, $x_1, x_2, \ldots$, but high correlations *among* the predictors can cause problems in estimating and testing their effects. The quote above shows the a typical quandary of some researchers in trying do understand these problems and and take steps to resolve them. This chapter illustrates the problems of collinearity, describes diagnostic measures to asses its effects, and presents some novel visual tools for these purposes using the **VisCollin** package.

One class of solutions for collinearity involves *regularization methods* such as ridge regression. Another collection of graphical methods, generalized ridge trace plots, implemented in the **genridge** package, sheds further light on what is accomplished by this technique.

Packages

In this chapter we use the following packages. Load them now.

```
library(car)
library(VisCollin)
library(genridge)
library(MASS)
library(dplyr)
library(factoextra)
library(ggrepel)
library(patchwork)
```

0.22 What is collinearity?

The chapter quote above is not untypical of researchers who have read standard treatments of linear models (eg.: ???) and yet are still confused about what collinearity is, how to find its sources and how to correct them. In Friendly & Kwan (2009), we liken this problem to that of the reader of Martin Hansford's successful series of books, *Where's Waldo*. These consist of a series of full-page illustrations of hundreds of people and things and a few Waldos— a character wearing a red and white striped shirt and hat, glasses, and carrying a walking stick or other paraphernalia. Waldo was never disguised, yet the complex arrangement of misleading visual cues in the pictures made him very hard to find. Collinearity diagnostics often provide a similar puzzle.

Recall the standard classical linear model for a response variable y with a collection of predictors in $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_p)$

$$\begin{aligned} \mathbf{y} &= \beta_0 + \beta_1 \mathbf{x}_1 + \beta_2 \mathbf{x}_2 + \cdots + \beta_p \mathbf{x}_p + \epsilon \\ &= \mathbf{X}\beta + \epsilon \, , \end{aligned}$$

for which the ordinary least squares solution is:

$$\widehat{\mathbf{b}} = (\mathbf{X}^T\mathbf{X})^{-1}\,\mathbf{X}^T\mathbf{y}\;,$$

with sampling variances and covariances $\text{Var}(\widehat{\mathbf{b}}) = \sigma^2 \times (\mathbf{X}^T\mathbf{X})^{-1}$ and σ^2 is the variance of the residuals ϵ, estimated by the mean squared error (MSE).

In the limiting case, when one x_i is *perfectly* predictable from the other xs, i.e., $R^2(x_i|\text{other } x) = 1$,

- there is no *unique* solution for the regression coefficients $\mathbf{b} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}\mathbf{y}$;
- the standard errors $s(b_i)$ of the estimated coefficients are infinite and t statistics $t_i = b_i/s(b_i)$ are 0.

This extreme case reflects a situation when one or more predictors are effectively redundant, for example when you include two variables x and y and their sum $z = x + y$ in a model, or use *ipsatized* scores that sum to a constant. More generally, collinearity refers to the case when there are very high **multiple correlations** among the predictors, such as $R^2(x_i|\text{other } x) \geq 0.9$. Note that you can't tell simply by looking at the simple correlations. A large correlation r_{ij} is *sufficient* for collinearity, but not *necessary* — you can have variables x_1, x_2, x_3 for which the pairwise correlation are low, but the multiple correlation is high.

The consequences are:

- The estimated coefficients have large standard errors, $s(\hat{b_j})$. They are multiplied by the square root of the variance inflation factor, $\sqrt{\text{VIF}}$, discussed below.
- This deflates the t-statistics, $t = \hat{b_j}/s(\hat{b_j})$ by the same factor.
- Thus you may find a situation where an overall model is highly significant (large F-statistic), while no (or few) of the individual predictors are. This is a puzzlement!
- Beyond this, the least squares solution may have poor numerical accurracy (Longley, 1967), because the solution depends on the determinant $|\,\mathbf{X}^T\mathbf{X}\,|$, which approaches 0 as multiple correlations increase.
- As well, recall that the coefficients $\hat{b}$ are **partial coefficients**, meaning the estimated change Δy in y when x changes by one unit Δx, but **holding all other variables constant**. Then, the model may be trying to estimate something that does not occur in the data.

0.22.1 Visualizing collinearity

Collinearity can be illustrated in data space for two predictors in terms of the stability of the regression plane for a linear model `Y = X1 + X2`. In Figure 70 (adapted from Fox (2016), Fig. 13.2):

(a) shows a case where X_1 and X_2 are uncorrelated as can be seen in their scatter in the horizontal plane (`+` symbols). The regression plane is well-supported; a small change in Y for one observation won't make much difference.

(b) In panel (b), X_1 and X_2 have a perfect correlation, $r(x_1, x_2) = 1.0$. The regression plane is not unique; in fact there are an infinite number of planes that fit the data equally well. Note that, if all we care about is prediction (not the coefficients), we could use X_1 or X_2, or both, or any weighted sum of them in a model and get the same predicted values.

(c) Shows a typical case where there is a strong correlation between X_1 and X_2. The regression plane here is unique, but is not well determined. A small change in Y **can** make quite a difference in the fitted value or coefficients, depending on the values of X_1 and X_2. Where X_1 and X_2 are far from their near linear relation in the botom plane, you can imagine that it is easy to tilt the plane substantially by a small change in Y.

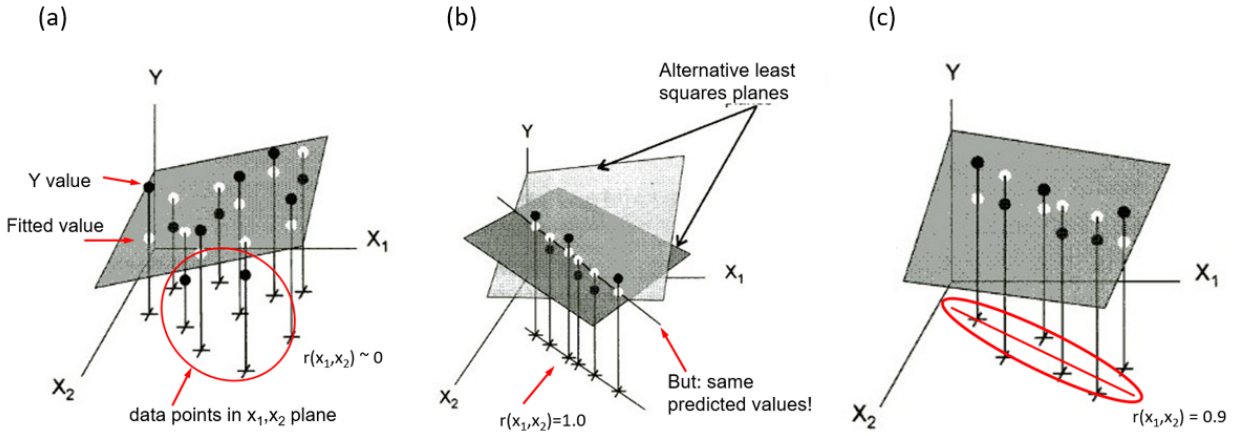


Figure 70: Effect of collinearity on the least squares regression plane. (a) Small correlation between predictors; (b) Perfect correlation ; (c) Very strong correlation. The black points show the data Y values, white points are the fitted values in the regression plane, and + signs represent the values of X1 and X2. *Source*: Adapted from Fox (2016), Fig. 13.2

0.22.2 Data space and β space

It is also useful to visualize collinearity by comparing the representation in **data space** with the analogous view of the confidence ellipses for coefficients in **beta space**. To do so, we generate data from a known model $y = 3x_1 + 3x_2 + \epsilon$ with $\epsilon \sim \mathcal{N}(0, 100)$ and various true correlations between x_1 and x_2, $\rho_{12} = (0, 0.8, 0.97)$ [9].

First, we use `MASS:mvrnorm()` to construct a list of data frames `XY` with specified values for the means and covariance matrices and a corresponding list of models, `mods`.

Working file:
`R/collin-data-beta.R`

```
library(MASS)
library(car)

set.seed(421)             # reproducibility
N <- 200                  # sample size
mu <- c(0, 0)             # means
s <- c(1, 1)              # standard deviations
rho <- c(0, 0.8, 0.97)    # correlations
beta <- c(3, 3)           # true coefficients

# Specify a covariance matrix, with standard deviations s[1], s[2] and correlation r
Cov <- function(s, r){
  matrix(c(s[1],    r * prod(s),
         r * prod(s), s[2]), nrow = 2, ncol = 2)
}

# Generate a dataframe of X, y for each rho
# Fit the model for each
XY <- vector(mode ="list", length = length(rho))
mods <- vector(mode ="list", length = length(rho))
for (i in seq_along(rho)) {
```

[9] If group sizes are greatly unequal **and** homogeneity of variance is violated, then the F statistic is too liberal (p values too large) when large sample variances are associated with small group sizes. Conversely, the F statistic is too conservative if large variances are associated with large group sizes.

```
  r <- rho[i]
  X <- mvrnorm(N, mu, Sigma = Cov(s, r))
  colnames(X) <- c("x1", "x2")
  y <- beta[1] * X[,1] + beta[2] * X[,2] + rnorm(N, 0, 10)

  XY[[i]] <- data.frame(X, y=y)
  mods[[i]] <- lm(y ~ x1 + x2, data=XY[[i]])
}
```

The estimated coefficients in these models are:

```
coefs <- sapply(mods, coef)
colnames(coefs) <- c("Intercept", "b1", "b2")
coefs
#>             Intercept      b1    b2
#> (Intercept)      1.01 -0.0535 0.141
#> x1               3.18  3.4719 3.053
#> x2               1.68  2.9734 2.059
```

Then, we define a function to plot the data ellipse (`car::dataEllipse()`) for each data frame and confidence ellipse (`car::dataEllipse()`) in the corresponding fitted model. In this figure, I specify the x, y limits for each plot so that the relative sizes of these ellipses are comparable, so that variance inflation can be assesed visually.

```
do_plots <- function(XY, mod, r) {
  X <- as.matrix(XY[, 1:2])
  dataEllipse(X,
              levels= 0.95,
              col = "darkgreen",
              fill = TRUE, fill.alpha = 0.05,
              xlim = c(-3, 3),
              ylim = c(-3, 3), asp = 1)
  text(0, 3, bquote(rho == .(r)), cex = 2, pos = NULL)

  confidenceEllipse(mod,
                    col = "red",
                    fill = TRUE, fill.alpha = 0.1,
                    xlab = "x1 coefficient",
                    ylab = "x2 coefficient",
                    xlim = c(-5, 10),
                    ylim = c(-5, 10),
                    asp = 1)
  points(beta[1], beta[2], pch = "+", cex=2)
  abline(v=0, h=0, lwd=2)
}

op <- par(mar = c(4,4,1,1)+0.1,
          mfcol = c(2, 3),
          cex.lab = 1.5)

for (i in seq_along(rho)) {
  do_plots(XY[[i]], mods[[i]], rho[i])
```

```
}
par(op)
```

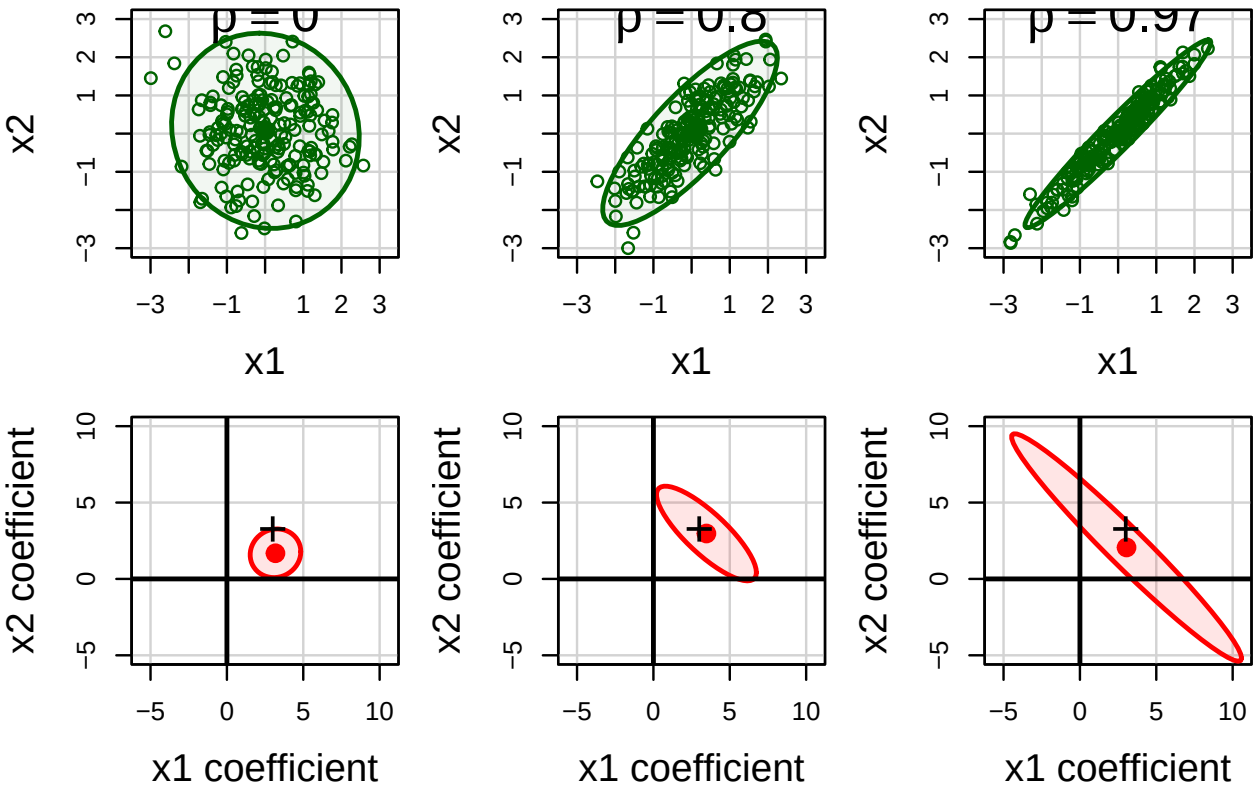


Figure 71: 95% Ddata ellipses for x1, x2 and the corresponding 95% confidence ellipses for their coefficients. In the confidence ellipse plots, reference lines show the value (0,0) for the null hypothesis and "+" marks the true values for the coefficients. This figure adapts an example by John Fox (2022).

Recall (Section #sec-data-beta) that the confidence ellipse for (β_1, β_2) is just a 90 degree rotation (and rescaling) of the data ellipse for (x_1, x_2): it is wide (more variance) in any direction where the data ellipse is narrow.

The shadows of the confidence ellipses on the coordinate axes in Figure 71 represent the standard errors of the coefficients, and get larger with increasing ρ. This is the effect of variance inflation, described in the following section.

0.23 Measuring collinearity

0.23.1 Variance inflation factors

How can we measure the effect of collinearity? The essential idea is to compare, for each predictor the variance $s^2(\widehat{b_j})$ that the coefficient that x_j would have if it was totally unrelated to the other predictors to the actual variance it has in the given model.

For two predictors such as shown in Figure 71 the sampling variance of x_1 can be expressed as

$$s^2(\widehat{b_1}) = \frac{MSE}{(n-1)\ s^2(x_1)} \times \left[\frac{1}{1-r_{12}^2}\right]$$

The first term here is the variance of b_1 when the two predictors are uncorrelated. The term in brackets represents the **variance inflation factor** (Marquardt, 1970), the amount by which the variance of the coefficient is multiplied as a consequence of the correlation r_{12} of the predictors. As $r_{12} \to 1$, the variances approaches infinity.

More generally, with any number of predictors, this relation has a similar form, replacing the simple correlation r_{12} with the multiple correlation predicting x_j from all others,

$$s^2(\widehat{b_j}) = \frac{MSE}{(n-1)\ s^2(x_j)} \times \left[\frac{1}{1-R^2_{j|\text{others}}}\right]$$

So, we have that the variance inflation factors are:

$$\text{VIF}_j = \frac{1}{1-R^2_{j\,|\,\text{others}}}$$

In practice, it is often easier to think in terms of the square root, $\sqrt{\text{VIF}_j}$ as the multiplier of the standard errors. The denominator, $1-R^2_{j|\text{others}}$ is sometimes called **tolerance**, a term I don't find particularly useful.

For the cases shown in Figure 71 the VIFs and their square roots are:

```
vifs <- sapply(mods, car::vif)
colnames(vifs) <- paste("rho:", rho)
vifs
#>    rho: 0 rho: 0.8 rho: 0.97
#> x1      1     3.09      18.6
#> x2      1     3.09      18.6

sqrt(vifs)
#>    rho: 0 rho: 0.8 rho: 0.97
#> x1      1     1.76      4.31
#> x2      1     1.76      4.31
```

Note that when there are terms in the model with more than one df, such as education with four levels (and hence 3 df) or a polynomial term specified as `poly(x, 3)`, the standard VIF calculation gives results that vary with how those terms are coded in the model. Fox & Monette (1992) define *generalized*, GVIFs as the inflation in the squared area of the confidence ellipse for the coefficients of such terms, relative to what would be obtained with uncorrelated data. Visually, this can be seen by comparing the areas of the ellipses in the bottom row of Figure 71. Because the magnitude of the GVIF increases with the number of degrees of freedom for the set of parameters, Fox & Monette suggest the analog $\sqrt{\text{GVIF}^{1/2\text{df}}}$ as the measure of impact on standard errors.

Example: This example uses the `cars` dataset in the `VisCollin` package containing various measures of size and performance on 406 models of automobiles from 1982. Interest is focused on predicting gas mileage, `mpg`.

```
data(cars, package = "VisCollin")
str(cars)
#> 'data.frame':    406 obs. of  10 variables:
#>  $ make    : Factor w/ 30 levels "amc","audi","bmw",..: 6 4 22 1 12 12 6 22 23 1 ...
#>  $ model   : chr  "chevelle" "skylark" "satellite" "rebel" ...
#>  $ mpg     : num  18 15 18 16 17 15 14 14 14 15 ...
#>  $ cylinder: int  8 8 8 8 8 8 8 8 8 8 ...
```

```
#>  $ engine  : num  307 350 318 304 302 429 454 440 455 390 ...
#>  $ horse   : int  130 165 150 150 140 198 220 215 225 190 ...
#>  $ weight  : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
#>  $ accel   : num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
#>  $ year    : int  70 70 70 70 70 70 70 70 70 70 ...
#>  $ origin  : Factor w/ 3 levels "Amer","Eur","Japan": 1 1 1 1 1 1 1 1 1 1 ...
```

We fit a model predicting gas mileage (`mpg`) from the number of cylinders, engine displacement, horsepower, weight, time to accelerate from 0 – 60 mph and model year (1970–1982). Perhaps surprisingly, only `weight` and `year` appear to significantly predict gas mileage. What's going on here?

```
cars.mod <- lm (mpg ~ cylinder + engine + horse + weight + accel + year,
                data=cars)
Anova(cars.mod)
#> Anova Table (Type II tests)
#>
#> Response: mpg
#>           Sum Sq  Df F value Pr(>F)
#> cylinder      12   1    0.99   0.32
#> engine        13   1    1.09   0.30
#> horse          0   1    0.00   0.98
#> weight      1214   1  102.84 <2e-16 ***
#> accel          8   1    0.70   0.40
#> year        2419   1  204.99 <2e-16 ***
#> Residuals   4543 385
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We check the variance inflation factors, using `car::vif()`. We see that most predictors have very high VIFs, indicating moderately severe multicollinearity.

```
vif(cars.mod)
#> cylinder   engine    horse   weight    accel     year
#>    10.63    19.64     9.40    10.73     2.63     1.24

sqrt(vif(cars.mod))
#> cylinder   engine    horse   weight    accel     year
#>     3.26     4.43     3.07     3.28     1.62     1.12
```

According to $\sqrt{\text{VIF}}$, the standard error of `cylinder` has been multiplied by 3.26 and it's t-value divided by this number, compared with the case when all predictors are uncorrelated. `engine`, `horse` and `weight` suffer a similar fate.

Connection with inverse of correlation matrix

In the linear regression model with standardized predictors, the covariance matrix of the estimated intercept-excluding parameter vector $\mathbf{b}^\star$ has the simpler form,

$$\mathcal{V}(\mathbf{b}^\star) = \frac{\sigma^2}{n-1}\mathbf{R}_X^{-1} \; .$$

where $\mathbf{R}_X$ is the correlation matrix among the predictors. It can then be seen that the VIF_j are just the diagonal entries of $\mathbf{R}_X^{-1}$.

More generally, the matrix $\mathbf{R}_X^{-1} = (r^{ij})$, when standardized to a correlation matrix as $-r^{ij}/\sqrt{r^{ii}\, r^{jj}}$ gives the matrix of all partial correlations, $r_{ij\,|\,\text{others}}$. }

0.23.2 Collinearity diagnostics

OK, we now know that large VIF_j indicate predictor coefficients whose estimation is degraded due to large $R^2_{j\,|\,\text{others}}$. But for this to be useful, we need to determine:

- how many dimensions in the space of the predictors are associated with nearly collinear relations?
- which predictors are most strongly implicated in each of these?

Answers to these questions are provided using measures developed by Belsley and colleagues (Belsley et al., 1980; Belsley, 1991). These measures are based on the eigenvalues $\lambda_1, \lambda_2, \ldots \lambda_p$ of the correlation matrix R_X of the predictors (preferably centered and scaled, and not including the constant term for the intercept), and the corresponding eigenvectors in the columns of $\mathbf{V}_{p\times p}$, given by the the eigen decomposition

$$\mathbf{R}_X = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$$

By elementary matrix algebra, the eigen decomposition of $\mathbf{R}_{XX}^{-1}$ is then

$$\mathbf{R}_X^{-1} = \mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{V}^T, \tag{0.2}$$

so, $\mathbf{R}_X$ and $\mathbf{R}_{XX}^{-1}$ have the same eigenvectors, and the eigenvalues of $\mathbf{R}_X^{-1}$ are just λ_i^{-1}. Using Equation 0.2, the variance inflation factors may be expressed as

$$\text{VIF}_j = \sum_{k=1}^{p} \frac{V_{jk}^2}{\lambda_k} \;,$$

which shows that only the *small* eigenvalues contribute to variance inflation, but only for those predictors that have large eigenvector coefficients on those small components. These facts lead to the following diagnostic statistics for collinearity:

- **Condition indices**: The smallest of the eigenvalues, those for which $\lambda_j \approx 0$, indicate collinearity and the number of small values indicates the number of near collinear relations. Because the sum of the eigenvalues, $\Sigma\lambda_i = p$ increases with the number of predictors p, it is useful to scale them all in relation to the largest. This leads to *condition indices*, defined as $\kappa_j = \sqrt{\lambda_1/\lambda_j}$. These have the property that the resulting numbers have common interpretations regardless of the number of predictors.
  - For completely uncorrelated predictors, all $\kappa_j = 1$.
  - $\kappa_j \to \infty$ as any $\lambda_k \to 0$.
- **Variance decomposition proportions**: Large VIFs indicate variables that are involved in *some* nearly collinear relations, but they don't indicate *which* other variable(s) each is involved with. For this purpose, Belsley et al. (1980) and Belsley (1991) proposed calculation of the proportions of variance of each variable associated with each principal component as a decomposition of the coefficient variance for each dimension.

These measures can be calculated using `VisCollin::colldiag()`. For the current model, the usual display contains both the condition indices and variance proportions. However, even for a small example, it is often difficult to know what numbers to pay attention to.

```
(cd <- colldiag(cars.mod, center=TRUE))
#> Condition
#> Index    Variance Decomposition Proportions
#>           cylinder engine horse weight accel year
#> 1   1.000 0.005    0.003  0.005 0.004  0.009 0.010
#> 2   2.252 0.004    0.002  0.000 0.007  0.022 0.787
#> 3   2.515 0.004    0.001  0.002 0.010  0.423 0.142
#> 4   5.660 0.309    0.014  0.306 0.087  0.063 0.005
```

```
#> 5   8.342 0.115    0.000  0.654 0.715  0.469 0.052
#> 6  10.818 0.563    0.981  0.032 0.176  0.013 0.004
```

Belsley (1991) recommends that the sources of collinearity be diagnosed (a) only for those components with large κ_j, and (b) for those components for which the variance proportion is large (say, ≥ 0.5) on *two* or more predictors. The print method for `"colldiag"` objects has a `fuzz` argument controlling this.

```
print(cd, fuzz = 0.5)
#> Condition
#> Index    Variance Decomposition Proportions
#>           cylinder engine horse weight accel year
#> 1   1.000  .        .      .     .      .     .
#> 2   2.252  .        .      .     .      .    0.787
#> 3   2.515  .        .      .     .      .     .
#> 4   5.660  .        .      .     .      .     .
#> 5   8.342  .        .    0.654 0.715   .     .
#> 6  10.818 0.563    0.981   .     .      .     .
```

The mystery is solved, if you can read that table with these recommendations in mind. There are two nearly collinear relations among the predictors, corresponding to the two smallest dimensions.

- Dimension 5 reflects the high correlation between horsepower and weight,
- Dimension 6 reflects the high correlation between number of cylinders and engine displacement.

Note that the high variance proportion for `year` (0.787) on the second component creates no problem and should be ignored because (a) the condition index is low and (b) it shares nothing with other predictors.

0.23.3 Tableplots

The default tabular display of condition indices and variance proportions from `colldiag()` is what triggered the comparison to "Where's Waldo". It suffers from the fact that the important information — (a) how many Waldos? (b) where are they hiding — is disguised by being embedded in a sea of mostly irrelevant numbers. The simple option of using a principled `fuzz` factor helps considerably, but not entirely.

The simplified tabular display above can be improved to make the patterns of collinearity more visually apparent and to signify warnings directly to the eyes. A **tableplot** (Kwan et al., 2009) is a semi-graphic display that presents numerical information in a table using shapes proportional to the value in a cell and other visual attributes (shape type, color fill, and so forth) to encode other information.

For collinearity diagnostics, these show:

- the condition indices, using using *squares* whose background color is red for condition indices > 10, green for values > 5 and green otherwise, reflecting danger, warning and OK respectively. The value of the condition index is encoded within this using a white square whose side is proportional to the value (up to some maximum value, `cond.max` that fills the cell).
- Variance decomposition proportions are shown by filled *circles* whose radius is proportional to those values and are filled (by default) with shades ranging from white through pink to red. Rounded values of those diagnostics are printed in the cells.

The tableplot below (Figure 72) encodes all the information from the values of `colldiag()` printed above (but using `prop.col` color breaks such that variance proportions < 0.3 are shaded white). The visual message is that one should attend to collinearities with large condition indices **and** large variance proportions implicating two or more predictors.

```
tableplot(cd, title = "Tableplot of cars data", cond.max = 30 )
```

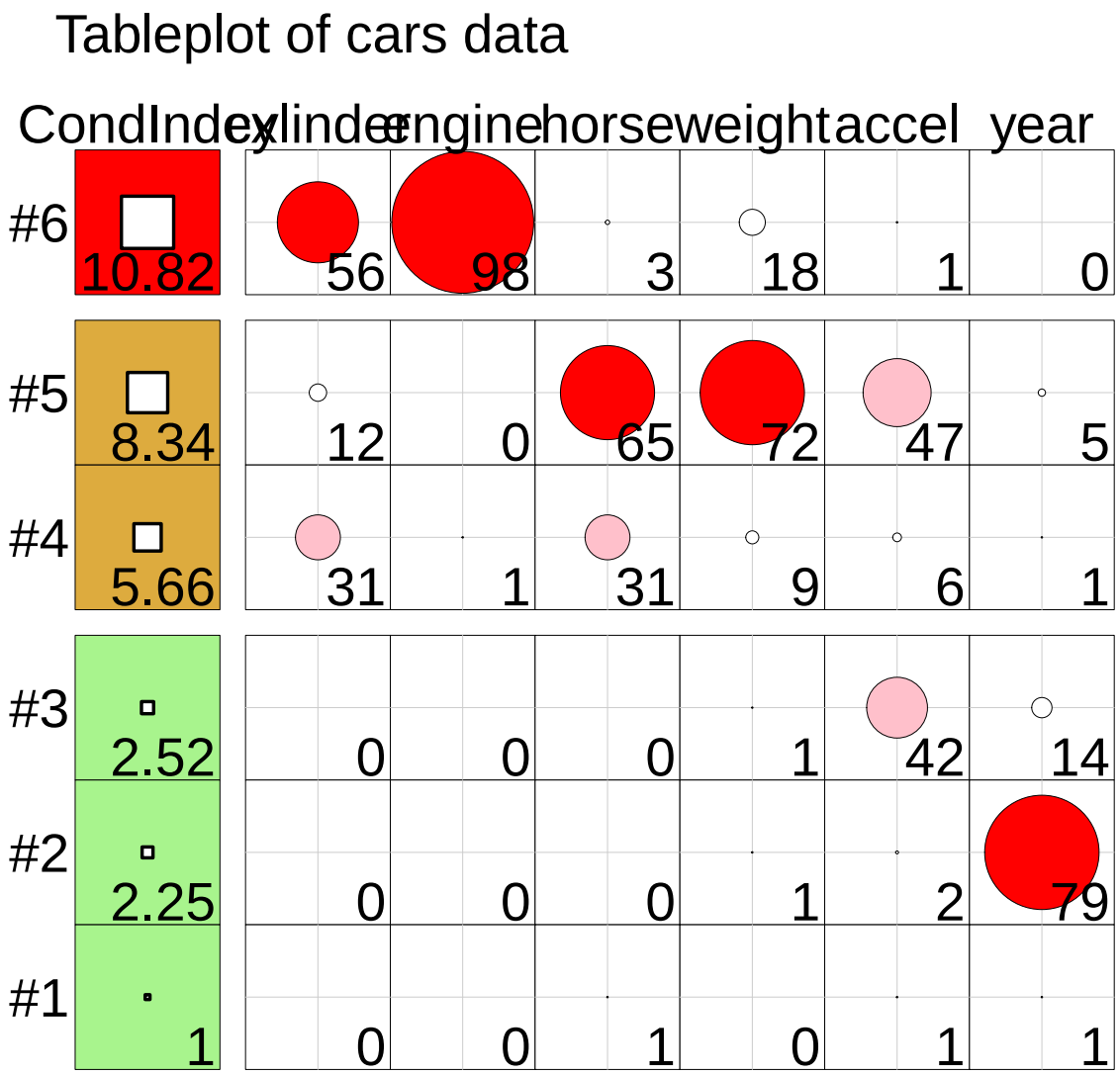


Figure 72: Tableplot of condition indices and variance proportions for the Cars data. In column 1, the square symbols are scaled relative to a maximum condition index of 30. In the remaining columns, variance proportions (times 100) are shown as circles scaled relative to a maximum of 100.

0.23.4 Collinearity biplots

As we have seen, the collinearity diagnostics are all functions of the eigenvalues and eigenvectors of the correlation matrix of the predictors in the regression model, or alternatively, the SVD of the $\mathbf{X}$ matrix in the linear model (excluding the constant). The standard biplot (Gabriel, 1971; Gower & Hand, 1996) (see: Section 0.7) can be regarded as a multivariate analog of a scatterplot, obtained by projecting a multivariate sample into a low-dimensional space (typically of 2 or 3 dimensions) accounting for the greatest variance in the data.

However the standard biplot is less useful for visualizing the relations among the predictors that lead to nearly collinear relations. Instead, biplots of the **smallest dimensions** show these relations directly, and can show

other features of the data as well, such as outliers and leverage points. We use `prcomp(X, scale.=TRUE)` to obtain the PCA of the correlation matrix of the predictors:

```
cars.X <- cars |>
  select(where(is.numeric)) |>
  select(-mpg) |>
  tidyr::drop_na()
cars.pca <- prcomp(cars.X, scale. = TRUE)
cars.pca
#> Standard deviations (1, .., p=6):
#> [1] 2.070 0.911 0.809 0.367 0.245 0.189
#>
#> Rotation (n x k) = (6 x 6):
#>             PC1    PC2    PC3    PC4     PC5     PC6
#> cylinder -0.454 0.1869 -0.168  0.659 -0.2711  0.4725
#> engine   -0.467 0.1628 -0.134  0.193 -0.0109 -0.8364
#> horse    -0.462 0.0177  0.123 -0.620 -0.6123  0.1067
#> weight   -0.444 0.2598 -0.278 -0.350  0.6860  0.2539
#> accel     0.330 0.2098 -0.865 -0.143 -0.2774 -0.0337
#> year      0.237 0.9092  0.335 -0.025 -0.0624 -0.0142
```

The standard deviations above are the square roots $\sqrt{\lambda_j}$ of the eigenvalues of the correlation matrix, and are returned in the `sdev` component of the `"prcomp"` object. The eigenvectors are returned in the `rotation` component, whose directions are arbitrary. Because we are interested in seeing the relative magnitude of variable vectors, we are free to multiply them by any constant to make them more visible in relation to the scores for the cars.

```
cars.pca$rotation <- -2.5 * cars.pca$rotation    # reflect & scale the variable vectors

ggp <- fviz_pca_biplot(
  cars.pca,
  axes = 6:5,
  geom = "point",
  col.var = "blue",
  labelsize = 5,
  pointsize = 1.5,
  arrowsize = 1.5,
  addEllipses = TRUE,
  ggtheme = ggplot2::theme_bw(base_size = 14),
  title = "Collinearity biplot for cars data")

# add point labels for outlying points
dsq <- heplots::Mahalanobis(cars.pca$x[, 6:5])
scores <- as.data.frame(cars.pca$x[, 6:5])
scores$name <- rownames(scores)

ggp + geom_text_repel(data = scores[dsq > qchisq(0.95, df = 6),],
                aes(x = PC6,
                    y = PC5,
                    label = name),
                vjust = -0.5,
                size = 5)
```

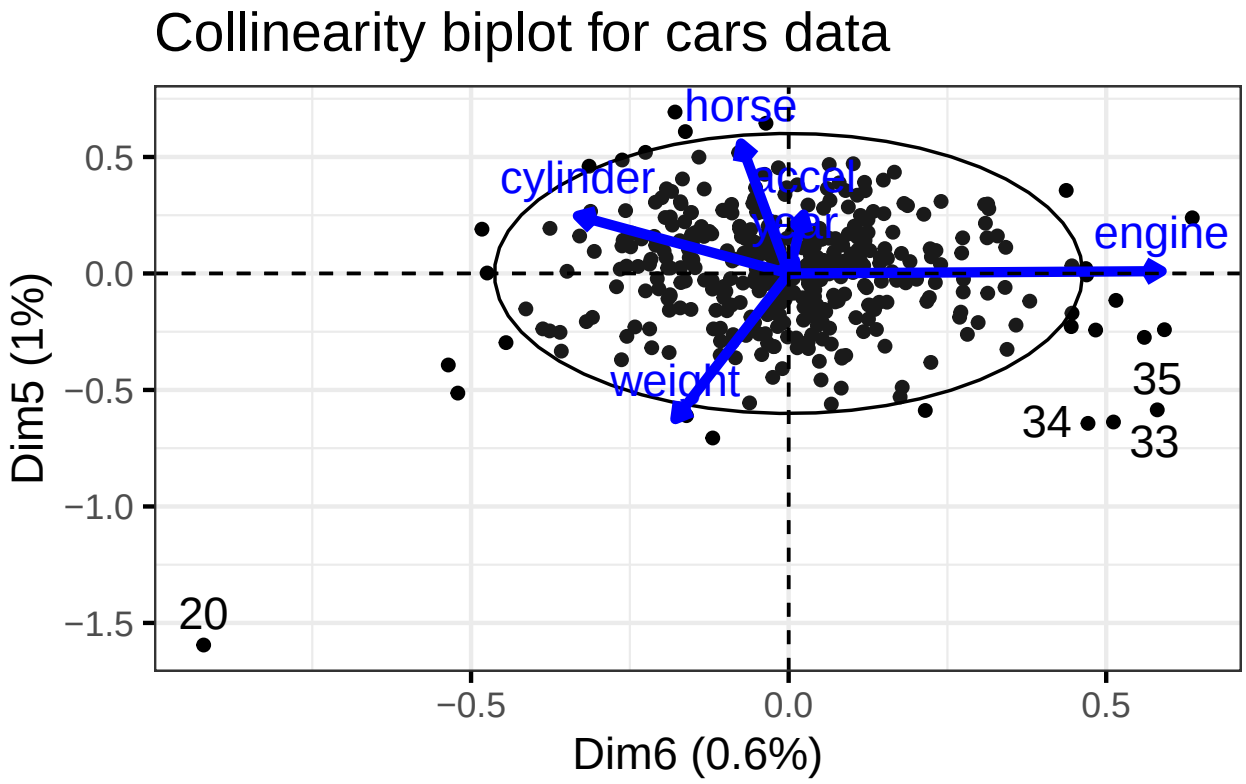


Figure 73: Collinearity biplot of the Cars data, showing the last two dimensions. The projections of the variable vectors on the coordinate axes are proportional to their variance proportions. To reduce graphic clutter, only the most outlying observations in predictor space are identified by case labels. An extreme outlier (case 20) appears in the lower left corner.

As with the tabular display of variance proportions, Waldo is hiding in the dimensions associated with the smallest eigenvalues (largest condition indices).
As well, it turns out that outliers in the predictor space (also high leverage observations) can often be seen as observations far from the centroid in the space of the smallest principal components.

The projections of the variable vectors in Figure 73 on the Dimension 5 and Dimension 6 axes are proportional to their variance proportions shown above. The relative lengths of these variable vectors can be considered to indicate the extent to which each variable contributes to collinearity for these two near-singular dimensions.

Thus, we see again that Dimension 6 is largely determined by `engine` size, with a substantial (negative) relation to `cylinder`. Dimension 5 has its' strongest relations to `weight` and `horse`.

Moreover, there is one observation, #20, that stands out as an outlier in predictor space, far from the centroid. It turns out that this vehicle, a Buick Estate wagon, is an early-year (1970) American behemoth, with an 8-cylinder, 455 cu. in, 225 horse-power engine, and able to go from 0 to 60 mph in 10 sec. (Its MPG is only slightly under-predicted from the regression model, however.)

0.24 Remedies for collinearity: What can I do?

Collinearity is often a **data** problem, for which there is no magic cure. Nevertheless there are some general guidelines and useful techniques to address this problem.

- **Pure prediction**: If we are only interested in predicting / explaining an outcome, and not the model coefficients or which are "significant", collinearity can be largely ignored. The fitted values are unaffected by collinearity, even in the case of perfect collinearity as shown in Figure 70 (b).
- **structural collinearity**: Sometimes collinearity results from structural relations among the variables that relate to how they have been defined.
  - For example, polynomial terms, like x, x^2, x^3 or interaction terms like $x_1, x_2, x_1 * x_2$ are necessarily correlated. A simple cure is to *center* the predictors at their means, using $x - \bar{x}, (x - \bar{x})^2, (x - \bar{x})^3$ or $(x_1 - \bar{x}_1), (x_2 - \bar{x}_2), (x_1 - \bar{x}_1) * (x_2 - \bar{x}_2)$. Centering removes the spurious ill-conditioning, thus reducing the VIFs. Note that in polynomial models, using `y ~ poly(x, 3)` to specify a cubic model generates *orthogonal* (uncorrelated) regressors, whereas in `y ~ x + I(x^2) + I(x^3)` the terms have built-in correlations.
  - When some predictors share a common cause, as in GNP or population in time-series or cross-national data, you can reduce collinearity by re-defining predictors to reflect *per capita measures*. In a related example with sports data, when you have cumulative totals (e.g., runs, hits, homeruns in baseball) for players over years, expressing these measures as *per year* will reduce the common effect of longevity on these measures.
- **Model re-specification**:
  - Drop one or more regressors that have a high VIF if they are not deemed to be essential
  - Replace highly correlated regressors with linear combination(s) of them. For example, two related variables, x_1 and x_2 can be replaced without any loss of information by replacing them with their sum and difference, $z_1 = x_1 + x_2$ and $z_2 = x_1 - x_2$. For example, in a dataset on fitness, we may have correlated predictors of resting pulse rate and pulse rate while running. Transforming these to average pulse rate and their difference gives new variables which are interpretable and less correlated.
- **Statistical remedies**:
  - Transform the predictors $\mathbf{X}$ to uncorrelated principal component scores $\mathbf{Z} = \mathbf{XV}$, and regress $\mathbf{y}$ on $\mathbf{Z}$. These will have the identical overall model fit without loss of information. A related technique is *incomplete* principal components regression, where some of the smallest dimensions (those causing collinearity) are omitted from the model. The trade-off is that it may be more difficult to interpret what the model means, but this can be countered with a biplot, showing the projections of the original variables into the reduced space of the principal components.
  - use **regularization methods** such as ridge regression and lasso, which correct for collinearity by introducing shrinking coefficients towards 0, introducing a small amount of bias, . See the genridge package and its `pkgdown` documentation for visualization methods.
  - use Bayesian regression; if multicollinearity prevents a regression coefficient from being estimated precisely, then a prior on that coefficient will help to reduce its posterior variance.

Example: Centering

To illustrate the effect of centering a predictor in a polynomial model, we generate a perfect quadratic relationship, $y = x^2$ and consider the correlations of y with x and with $(x - \bar{x})^2$. The correlation of y with x is 0.97, while the correlation of y with $(x - \bar{x})^2$ is zero.

```
x <- 1:20
y1 <- x^2
y2 <- (x - mean(x))^2
XY <- data.frame(x, y1, y2)

(R <- cor(XY))
```

```
#>        x    y1    y2
#> x  1.000 0.971 0.000
#> y1 0.971 1.000 0.238
#> y2 0.000 0.238 1.000
```

The effect of centering here is remove the linear association in what is a purely quadratic relationship, as can be seen by plotting `y1` and `y2` against `x`.

```
r1 <- R[1, 2]
r2 <- R[1, 3]

gg1 <-
ggplot(XY, aes(x = x, y = y1)) +
  geom_point(size = 3) +
  geom_smooth(method = "lm", formula = y~x, linewidth = 2, se = FALSE) +
  labs(x = "X", y = "Y") +
  theme_bw(base_size = 16) +
  annotate("text", x = 5, y = 350, size = 6,
           label = paste("X Uncentered\nr =", round(r1, 3)))

gg2 <-
  ggplot(XY, aes(x = x, y = y2)) +
  geom_point(size = 3) +
  geom_smooth(method = "lm", formula = y~x, linewidth = 2, se = FALSE) +
  labs(x = "X", y = "Y") +
  theme_bw(base_size = 16) +
  annotate("text", x = 5, y = 80, size = 6,
           label = paste("X Centered\nr =", round(r2, 3)))

gg1 + gg2         # show plots side-by-side
```

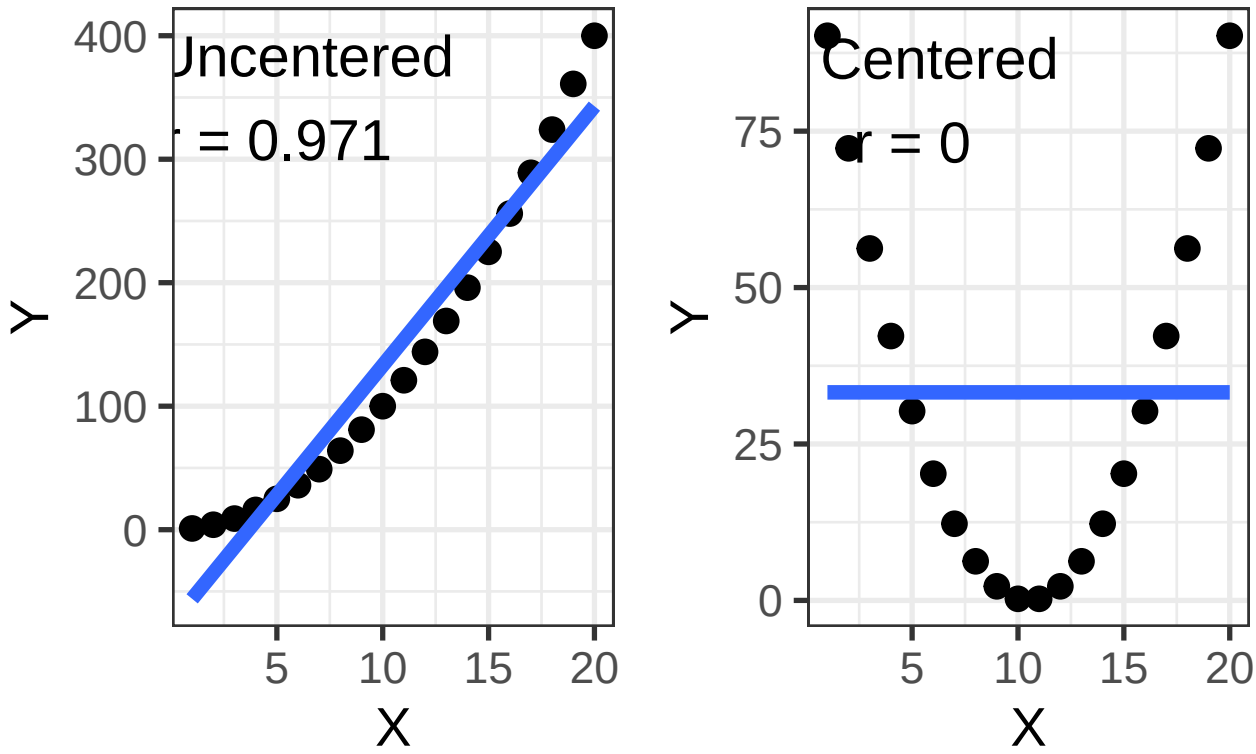


Figure 74: Centering a predictor removes the nessessary correlation in a quadratic regression

Example: Interactions

The dataset `genridge::Acetylene` gives data from Marquardt & Snee (1975) on the `yield` of a chemical manufacturing process to produce acetylene in relation to reactor temperature (`temp`), the `ratio` of two components and the contact `time` in the reactor. A naive response surface model might suggest that yield is quadratic in time and there are potential interactions among all pairs of predictors.

```
data(Acetylene, package = "genridge")
acetyl.mod0 <- lm(yield ~ temp + ratio + time + I(time^2) +
                          temp:time + temp:ratio + time:ratio,
                  data=Acetylene)

(acetyl.vif0 <- vif(acetyl.mod0))
#>       temp      ratio       time  I(time^2)  temp:time temp:ratio
#>        383      10555      18080        564       9719       9693
#> ratio:time
#>        225
```

These results are horrible! How much does centering help? I first center all three predictors and then use `update()` to re-fit the same model using the centered data.

```
Acetylene.centered <-
  Acetylene |>
  mutate(temp = temp - mean(temp),
         time = time - mean(time),
         ratio = ratio - mean(ratio))
```

```
acetyl.mod1 <- update(acetyl.mod0, data=Acetylene.centered)
(acetyl.vif1 <- vif(acetyl.mod1))
#>       temp      ratio       time  I(time^2)  temp:time temp:ratio
#>      57.09       1.09      81.57      51.49      44.67      30.69
#> ratio:time
#>      33.33
```

This is far better, although still not great in terms of VIF. But, how much have we improved the situation by the simple act of centering the predictors? The square roots of the ratios of VIFs tell us the impact of centering on the standard errors.

```
sqrt(acetyl.vif0 / acetyl.vif1)
#>       temp      ratio       time  I(time^2)  temp:time temp:ratio
#>       2.59      98.24      14.89       3.31      14.75      17.77
#> ratio:time
#>       2.60
```

Finally, we use `poly(time, 2)` in the model for the centered data. Because there are multiple degree of freedom terms in the model, `car::vif()` calculates GVIFs here. The final column gives $\sqrt{\text{GVIF}^{1/2\text{df}}}$, the remaining effect of collinearity on the standard errors of terms in this model.

```
acetyl.mod2 <- lm(yield ~ temp + ratio + poly(time, 2) +
                          temp:time + temp:ratio + time:ratio,
                  data=Acetylene.centered)

vif(acetyl.mod2, type = "term")
#>                   GVIF Df GVIF^(1/(2*Df))
#> temp             57.09  1            7.56
#> ratio             1.09  1            1.05
#> poly(time, 2) 1733.56  2            6.45
#> temp:time        44.67  1            6.68
#> temp:ratio       30.69  1            5.54
#> ratio:time       33.33  1            5.77
```

0.25 Ridge regression

0.25.1 What is ridge regression?

0.25.2 Univariate ridge trace plots

0.25.3 Bivariate ridge trace plots

```
#> Writing packages to  C:/R/Projects/Vis-MLM-book/bib/pkgs.txt
#> 18  packages used here:
#>  base, car, carData, datasets, dplyr, factoextra, genridge, ggplot2, ggrepel, graphics, grDevices, kni
```

0

Hotelling's T^2

Just as the one- and two- sample univariate t-test is the gateway drug for understanding analysis of variance, so too Hotelling's T^2 test provides an entry point to multivariate analysis of variance. This simple case provides an entry point to understanding the collection of methods I call the **HE plot framework** for visualizing effects in multivariate linear models, which are a main focus of this book.

The essential idea is that Hotelling's T^2 provides a test of the difference in means between two groups on a *collection* of variables, $\mathbf{x} = x_1, x_2, \ldots x_p$ *simultaneously*, rather than one by one. This has the advantages that it:

- does not require p-value corrections for multiple tests (e.g., Bonferroni);
- combines the evidence from the multiple response variables, and *pools strength*, accumulating support across measures;
- clarifies how the multiple response are *jointly* related to the group effect along a single dimension, the *discriminant axis*;
- in so doing, it reduces the problem of testing differences for two (and potentially more) response variables to testing the difference on a single variable that best captures the multivariable relations.

After describing it's features, I use an example of a two-group T^2 test to illustrate the basic ideas behind multivariate tests and hypothesis error plots.

Packages

In this chapter we use the following packages. Load them now.

```
library(car)
library(heplots)
library(Hotelling)
library(ggplot2)
library(dplyr)
library(tidyr)
library(ggbiplot)
```

0.26 T^2 as a generalized t-test

Hotelling's T^2 (Hotelling, 1931) is an analog the square of a univariate t statistic, extended to the Consider the basic one-sample t-test, where we wish to test the hypothesis that the mean $\bar{x}$ of a set of N measures on a test of basic math, with standard deviation s does not differ from an assumed mean $\mu_0 = 150$ for a population. The t statistic for testing $H_0 : \mu = \mu_0$ against the two-sided alternative, $H_0 : \mu \neq \mu_0$ is

$$t = \frac{(\bar{x} - \mu_0)}{s/\sqrt{N}} = \frac{(\bar{x} - \mu_0)\sqrt{N}}{s}$$

Squaring this gives

$$t^2 = \frac{N(\bar{x} - \mu_0)^2}{s} = N(\bar{x} - \mu_0)(s^2)^{-1}(\bar{x} - \mu_0)$$

Now consider we also have measures on a test of solving word problems for the same sample. Then, a hypothesis test for the means on basic math (BM) and word problems (WP) is the test of the means of these two variables jointly equal their separate values, say, $(150, 100)$.

$$H_0 : \mu = \mu_{\mathbf{0}} = \begin{pmatrix} \mu_{0,BM} \\ \mu_{0,WP} \end{pmatrix} = \begin{pmatrix} 150 \\ 100 \end{pmatrix}$$

Hotelling's T^2 is then the analog of t^2, with the variance-covariance matrix $\mathbf{S}$ of the scores on (BM, WP) replacing the variance of a single score. This is nothing more than the squared Mahalanobis distance between the sample mean vector $(\bar{x}_{BM}, \bar{x}_{WP})^T$ and the hypothesized means μ_0, in the metric of $\mathbf{S}$, as shown in Figure 75.

$$\begin{aligned} T^2 &= N(\bar{\mathbf{x}} - \mu_0)^T \, \mathbf{S}^{-1} \, (\bar{\mathbf{x}} - \mu_0) \\ &= N D_M^2(\bar{\mathbf{x}}, \mu_0) \end{aligned}$$

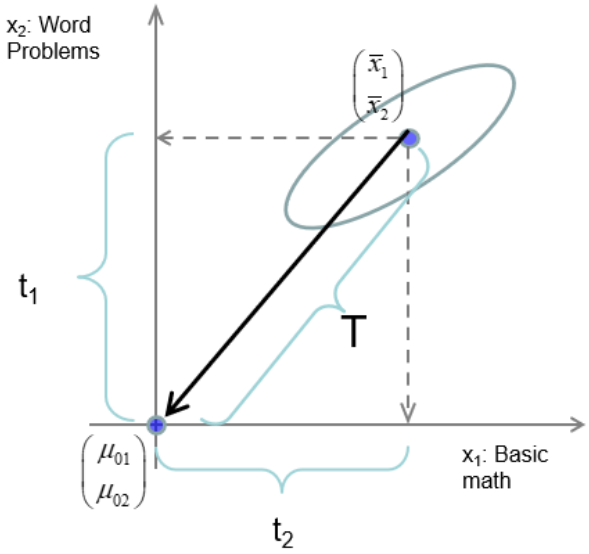


Figure 75: Hotelling's T^2 statistic as the squared distance between the sample means and hypothesized means relative to the variance-covariance matrix. *Source*: Author

0.27 T^2 properties

Aside from it's elegant geometric interpretation Hotelling's T^2 has simple properties that aid in understanding the extension to more complex multivariate tests.

- **Maximum** t^2 : Consider constructing a new variable w as a linear combination of the scores in a matrix $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_p}]$ with weights $\mathbf{a}$,
$$w = a_1\mathbf{x_1} + a_2\mathbf{x_2} + \cdots + a_p\mathbf{x_p} = \mathbf{Xa}$$
Hotelling's T^2 is then the maximum value of a univariate $t^2(\mathbf{a})$ over all possible choices of the weights in $\mathbf{a}$. In this way, Hotellings test reduces a multivariate problem to a univariate one.

- **Eigenvalue** : Hotelling showed that T^2 is the one non-zero eigenvalue (latent root) λ of the matrix $\mathbf{Q}_H = N(\bar{\mathbf{x}} - \mu_0)^T(\bar{\mathbf{x}} - \mu_0)$ relative to $\mathbf{Q}_E = \mathbf{S}$ that solves the equation
$$(\mathbf{Q}_H - \lambda \mathbf{Q}_E)\mathbf{a} = 0 \tag{0.3}$$
In more complex MANOVA problems, there are more than one non-zero latent roots, $\lambda_1, \lambda_2, \ldots \lambda_s$, and test statistics (Wilks' Λ, Pillai and Hotelling-Lawley trace criteria, Roy's maximum root test) are functions of these.

- **Eigenvector** : The corresponding eigenvector is $\mathbf{a} = \mathbf{S}^{-1}(\bar{\mathbf{x}} - \mu_0)$. These are the (raw) *discriminant coefficients*, giving the relative contribution of each variable to T^2.

- **Critical values** : For a single response, the square of a t statistic with $N-1$ degrees of freedom is an $F(1, N-1)$ statistic. But we chose $\mathbf{a}$ to give the *maximum* $t^2(\mathbf{a})$; this can be taken into account with a transformation of T^2 to give an **exact** F test with the correct sampling distribution:
$$F^* = \frac{N-p}{p(N-1)} T^2 \;\sim\; F(p, N-p) \tag{0.4}$$

- **Invariance under linear transformation** : Just as a univariate t-test is unchanged if we apply a linear transformation to the variable, $x \to ax + b$, T^2 is invariant under all linear (*affine*) transformations,
$$\mathbf{x}_{p\times 1} \to \mathbf{C}_{p\times p}\mathbf{x} + \mathbf{b}$$
So, you get the same results if you convert penguins flipper lengths from millimeters to centimeters or inches. The same is true for all MANOVA tests.

- **Two-sample tests** : With minor variations in notation, everything above applies to the more usual test of equality of multivariate means in a two sample test of $H_0: \mu_1 = \mu_2$.
$$T^2 = N(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T \mathbf{S}_p^{-1} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)$$
where $\mathbf{S}_p$ is the pooled within-sample variance covariance matrix.

Example

The data set `heplots::mathscore` gives (fictitious) scores on a test of basic math skills (BM) and solving word problems (WP) for two groups of $N = 6$ students in an algebra course, each taught by different instructors.

```
data(mathscore, package = "heplots")
str(mathscore)
#> 'data.frame':    12 obs. of  3 variables:
#>  $ group: Factor w/ 2 levels "1","2": 1 1 1 1 1 1 2 2 2 2 ...
#>  $ BM   : int  190 170 180 200 150 180 160 190 150 160 ...
#>  $ WP   : int  90 80 80 120 60 70 120 150 90 130 ...
```

You can carry out the test that the means for both variables are jointly equal using either `Hotelling::hotelling.test()` (Curran & Hersh, 2021) or `car::Anova()`,

```
hotelling.test(cbind(BM, WP) ~ group, data=mathscore) |> print()
#> Test stat:  64.174
#> Numerator df:  2
#> Denominator df:  9
#> P-value:  0.0001213

math.mod <- lm(cbind(BM, WP) ~ group, data=mathscore)
Anova(math.mod)
#>
#> Type II MANOVA Tests: Pillai test statistic
#>       Df test stat approx F num Df den Df  Pr(>F)
```

```
#> group  1     0.865     28.9      2      9 0.00012 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

What's wrong with just doing the two t-tests (or equivalent F-test with `lm()`)?

```
Anova(mod1 <- lm(BM ~ group, data=mathscore))
#> Anova Table (Type II tests)
#>
#> Response: BM
#>           Sum Sq Df F value Pr(>F)
#> group       1302  1    4.24  0.066 .
#> Residuals   3071 10
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Anova(mod2 <- lm(WP ~ group, data=mathscore))
#> Anova Table (Type II tests)
#>
#> Response: WP
#>           Sum Sq Df F value Pr(>F)
#> group       4408  1    10.4  0.009 **
#> Residuals   4217 10
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From this, we might conclude that the two groups do *not* differ significantly on Basic Math but strongly differ on Word problems. But the two univariate tests do not take the correlation among the mean differences into account.

To see the differences between the groups on both variables together, we draw their data (68%) ellipses, using `heplots::covEllpses()`

```
colors <- c("darkgreen", "blue")
covEllipses(mathscore[,c("BM", "WP")], mathscore$group,
            pooled=FALSE,
            col = colors,
            fill = TRUE,
            fill.alpha = 0.05,
            cex = 2, cex.lab = 1.5,
            asp = 1,
            xlab="Basic math", ylab="Word problems")
# plot points
pch <- ifelse(mathscore$group==1, 15, 16)
col <- ifelse(mathscore$group==1, colors[1], colors[2])
points(mathscore[,2:3], pch=pch, col=col, cex=1.25)
```

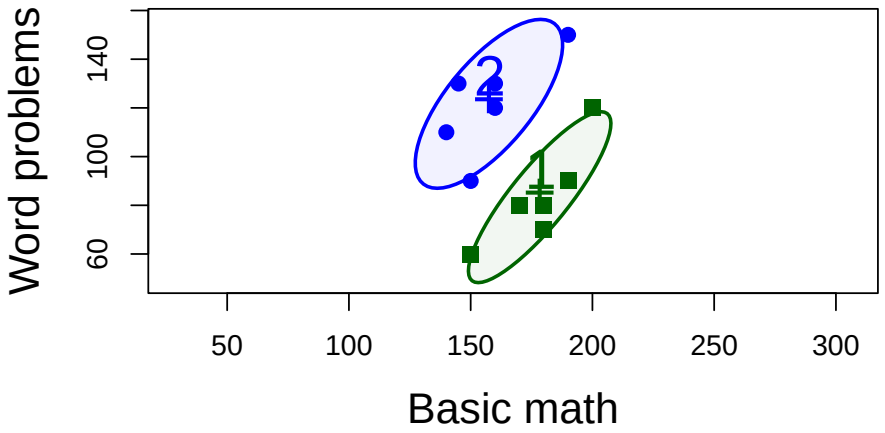


Figure 76: Data ellipses for the `mathscore` data, enclosing approximately 68% of the observations in each group

We can see that:

- Group 1 > Group 2 on Basic Math, but worse on Word Problems
- Group 2 > Group 1 on Word Problems, but worse on Basic Math
- Within each group, those who do better on Basic Math also do better on Word Problems

We can also see why the univariate test, at least for Basic math is non-significant: the scores for the two groups overlap considerably on the horizontal axis. They are slightly better separated along the vertical axis for word problems. The plot also reveals why Hotelling's T^2 reveals such a strongly significant result: the two groups are very widely separated along an approximately 45^o line between them.

A relatively simple interpretation is that the groups don't really differ in overall math ability, but perhaps the instructor in Group 1 put more focus on basic math skills, while the instructor for Group 2 placed greater emphasis on solving word problems.

In Hotelling's T^2, the "size" of the difference between the means (labeled "1" and "2") is assessed relative to the pooled within-group covariance matrix $\mathbf{S}_p$, which is just a size-weighted average of the two within-sample matrices, $\mathbf{S}_1$ and $\mathbf{S}_2$,

$$\mathbf{S}_p = [(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2]/(n_1 + n_2 - 2)$$

Visually, imagine sliding the the separate data ellipses to the grand mean, $(\bar{x}_{\text{BM}}, \bar{x}_{\text{WP}})$ and finding their combined data ellipse. This is just the data ellipse of the sample of deviations of the scores from their group means, or that of the residuals from the model `lm(cbind(BM, WP) ~ group, data=mathscore)`

To see this, we plot $\mathbf{S}_1$, $\mathbf{S}_2$ and $\mathbf{S}_p$ together,

```
covEllipses(mathscore[,c("BM", "WP")], mathscore$group,
            col = c(colors, "red"),
            fill = c(FALSE, FALSE, TRUE),
            fill.alpha = 0.3,
            cex = 2, cex.lab = 1.5,
            asp = 1,
            xlab="Basic math", ylab="Word problems")
```

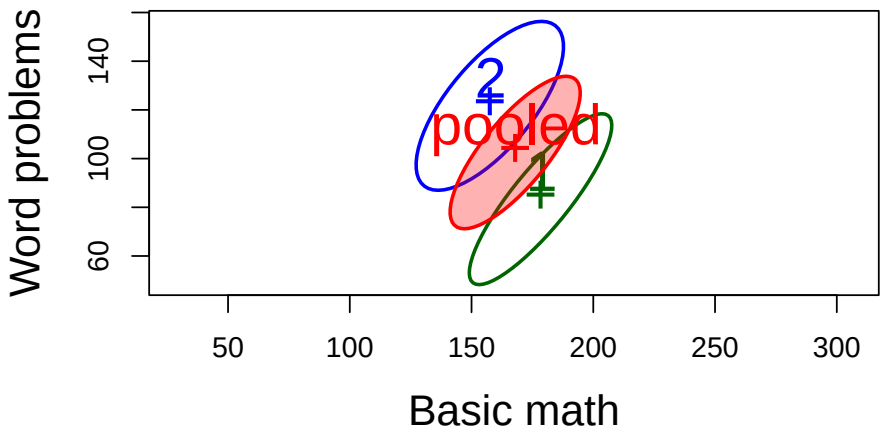


Figure 77: Data ellipses and the pooled covariance matrix `mathscore` data.

One of the assumptions of the T^2 test (and of MANOVA) is that the within-group variance covariance matrices, $\mathbf{S}_1$ and $\mathbf{S}_2$, are the same. In Figure 77, you can see how the shapes of $\mathbf{S}_1$ and $\mathbf{S}_2$ are very similar, differing in that the variance of word Problems is slightly greater for group 2. In Chapter XX we take of the topic of visualizing tests of this assumption, based on Box's M-test.

0.28 HE plot and discriminant axis

As we describe in detail in Chapter XX, all the information relevant to the T^2 test and MANOVA can be captured in the remarkably simple *Hypothesis Error* plot, which shows the relative size of two data ellipses,

- **H**: the data ellipse of the *fitted* values, which are just the group means on the two variables, $\bar{\mathbf{x}}$, corresponding to $\mathbf{Q}_H$ in Equation 0.3. In case of T^2, the **H** matrix is of rank 1, so the "ellipse" plots as a line.

```
# calculate H directly
fit <- fitted(math.mod)
xbar <- colMeans(mathscore[,2:3])
N <- nrow(mathscore)
crossprod(fit) - N * outer(xbar, xbar)
#>       BM    WP
#> BM  1302 -2396
#> WP -2396  4408

# same as: SSP for group effect from Anova
math.aov <- Anova(math.mod)
(H <- math.aov$SSP)
#> $group
#>       BM    WP
#> BM  1302 -2396
#> WP -2396  4408
```

- **E**: the data ellipse of the *residuals*, the deviations of the scores from the group means, $\mathbf{x} - \bar{\mathbf{x}}$, corresponding to $\mathbf{Q}_E$.

```
# calculate E directly
resids <- residuals(math.mod)
crossprod(resids)
#>      BM   WP
#> BM 3071 2808
#> WP 2808 4217

# same as: SSPE from Anova
(E <- math.aov$SSPE)
#>      BM   WP
#> BM 3071 2808
#> WP 2808 4217
```

0.28.1 `heplot()`

`heplots::heplot()` takes the model object, extracts the **H** and **E** matrices (from `summary(Anova(math.mod))`) and plots them. There are many options to control the details.

```
heplot(math.mod,
       fill=TRUE, lwd = 3,
       asp = 1,
       cex=2, cex.lab=1.8,
       xlab="Basic math", ylab="Word problems")
```

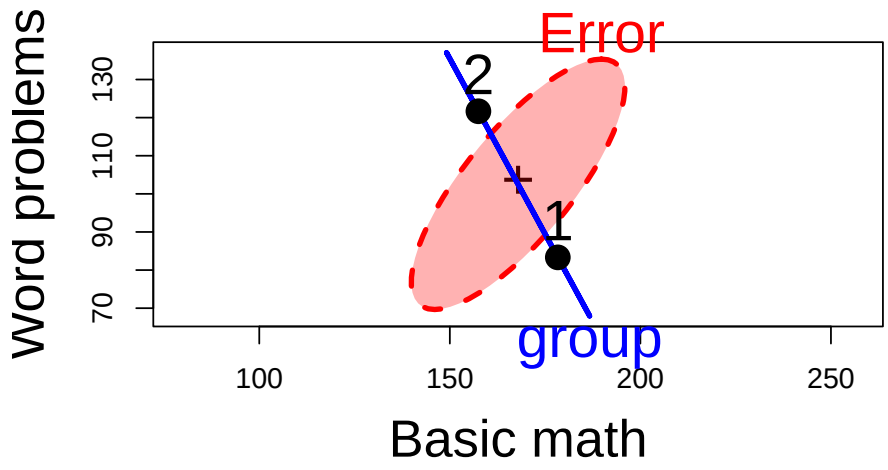


Figure 78: Hypothesis error plot of the `mathscore` data. The line through the group means is the H ellipse, which plots as a line here. The red ellipse labeled 'Error' represents the pooled within-group covariance matrix.

But the HE plot offers more:

- A visual test of significance: the **H** ellipse is scaled so that it projects *anywhere* outside the **E** ellipse, if and only if the test is significant at a given α level ($\alpha = 0.05$ by default)
- The **H** ellipse, which appears as a line, goes through the means of the two groups. This is also the *discriminant axis*, the direction in the space of the variables which maximally discriminates between the groups. That is, if we project the data points onto this line, we get the linear combination w which has the maximum possible univariate t^2.

You can see how the HE plot relates to the plots of the separate data ellipses by overlaying them in a single figure. We also plot the scores on the discriminant axis, by using this small function to find the orthogonal projection of a point $\mathbf{a}$ on the line joining two points, $\mathbf{p}_1$ and $\mathbf{p}_2$, which in math is $\mathbf{p}_1 + \frac{\mathbf{d}^T(\mathbf{a}-\mathbf{p}_1)}{\mathbf{d}^T\mathbf{d}}$, letting $\mathbf{d} = \mathbf{p}_1 - \mathbf{p}_2$.

```
dot <- function(x, y) sum(x*y)
project_on <- function(a, p1, p2) {
    a <- as.numeric(a)
    p1 <- as.numeric(p1)
    p2 <- as.numeric(p2)
    dot <- function(x,y) sum( x * y)
    t <- dot(p2-p1, a-p1) / dot(p2-p1, p2-p1)
    C <- p1 + t*(p2-p1)
    C
}
```

Then, we run the same code as before to plot the data ellipses, and follow this with a call to `heplot()` using the option `add=TRUE` which adds to an existing plot. Following this, we find the group means and draw lines projecting the points on the line between them.

```
covEllipses(mathscore[,c("BM", "WP")], mathscore$group,
            pooled=FALSE,
            col = colors,
            cex=2, cex.lab=1.5,
            asp=1,
            xlab="Basic math", ylab="Word problems"
            )
pch <- ifelse(mathscore$group==1, 15, 16)
col <- ifelse(mathscore$group==1, "red", "blue")
points(mathscore[,2:3], pch=pch, col=col, cex=1.25)

# overlay with HEplot (add = TRUE)
heplot(math.mod,
       fill=TRUE,
       cex=2, cex.lab=1.8,
       fill.alpha=0.2, lwd=c(1,3),
       add = TRUE,
       error.ellipse=TRUE)

# find group means
means <- mathscore |>
  group_by(group) |>
  summarize(BM = mean(BM), WP = mean(WP))

for(i in 1:nrow(mathscore)) {
    gp <- mathscore$group[i]
    pt <- project_on( mathscore[i, 2:3], means[1, 2:3], means[2, 2:3])
    segments(mathscore[i, "BM"], mathscore[i, "WP"], pt[1], pt[2], lwd = 1.2)
}
```

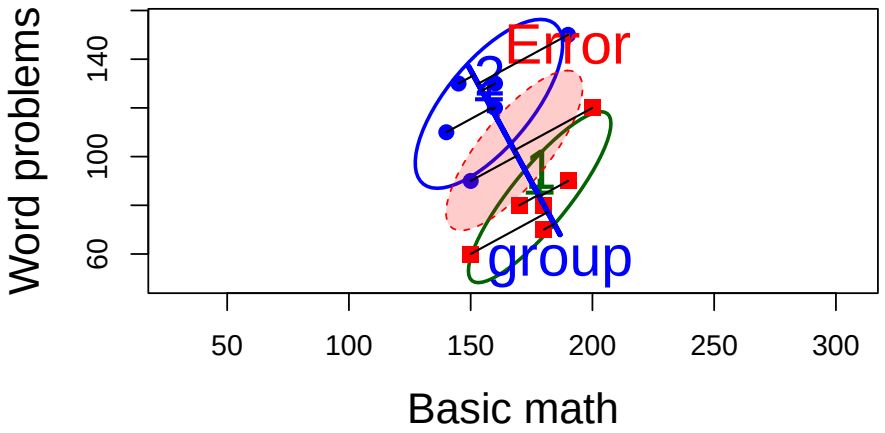


Figure 79: HE plot overlaid on top of the within-group data ellipses, with lines showing the projection of each point on the discriminant axis.

0.29 Discriminant analysis

Discriminant analysis for two-group designs or for one-way MANOVA essentially turns the problem around: Instead of asking whether the mean vectors for two or more groups are equal, discriminant analysis tries to find the linear combination w of the response variables that has the greatest separation among the groups, allowing cases to be best classified. It was developed by Fisher (1936) as a solution to the biological taxonomy problem of developing a rule to classify instances of flowers—in his famous case, Iris flowers—into known species (*I. setosa*, *I. versicolor*, *I. virginica*) on the basis of multiple measurements (length and width of their sepals and petals).

```
(math.lda <- MASS::lda(group ~ ., data=mathscore))
#> Call:
#> lda(group ~ ., data = mathscore)
#>
#> Prior probabilities of groups:
#>   1   2
#> 0.5 0.5
#>
#> Group means:
#>    BM    WP
#> 1 178  83.3
#> 2 158 121.7
#>
#> Coefficients of linear discriminants:
#>        LD1
#> BM -0.0835
#> WP  0.0753
```

The coefficients give $w = -0.84\text{BM} + 0.75\text{WP}$. This is exactly the direction given by the line for the **H** ellipse in Figure 79.

To round this out, we can calculate the discriminant scores by multiplying the matrix $\mathbf{X}$ by the vector $\mathbf{a} = \mathbf{S}^{-1}(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)$ of the discriminant weights.

```
math.lda$scaling
#>        LD1
#> BM -0.0835
#> WP  0.0753

scores <- cbind(group = mathscore$group,
                as.matrix(mathscore[, 2:3]) %*% math.lda$scaling) |>
  as.data.frame()
scores |>
  group_by(group) |>
  slice(1:3)
#> # A tibble: 6 x 2
#> # Groups:   group [2]
#>   group   LD1
#>   <dbl> <dbl>
#> 1     1 -9.09
#> 2     1 -8.17
#> 3     1 -9.01
#> 4     2 -4.33
#> 5     2 -4.58
#> 6     2 -5.75
```

Then a t-test on these scores gives Hotelling's T, accessed via the `statistic` component of `t.test()`

```
t <- t.test(LD1 ~ group, data=scores)$statistic
c(t, T2 =t^2)
#>     t  T2.t
#> -8.01 64.17
```

Finally, it is instructive to compare violin plots for the three measures, `BM`, `WP` and `LD1`. To do this with `ggplot2` requires reshaping the data from wide to long format so the plots can be faceted.

```
scores <- mathscore |>
  bind_cols(LD1 = scores[, "LD1"])

scores |>
  tidyr::gather(key = "measure", value ="Score", BM:LD1) |>
  mutate(measure = factor(measure, levels = c("BM", "WP", "LD1"))) |>
  ggplot(aes(x = group, y = Score, color = group, fill = group)) +
    geom_violin(alpha = 0.2) +
    geom_jitter(width = .2, size = 2) +
    facet_wrap( ~ measure, scales = "free", labeller = label_both) +
    scale_fill_manual(values = c("darkgreen", "blue")) +
    scale_color_manual(values = c("darkgreen", "blue")) +
    theme_bw(base_size = 14) +
    theme(legend.position = "none")
```

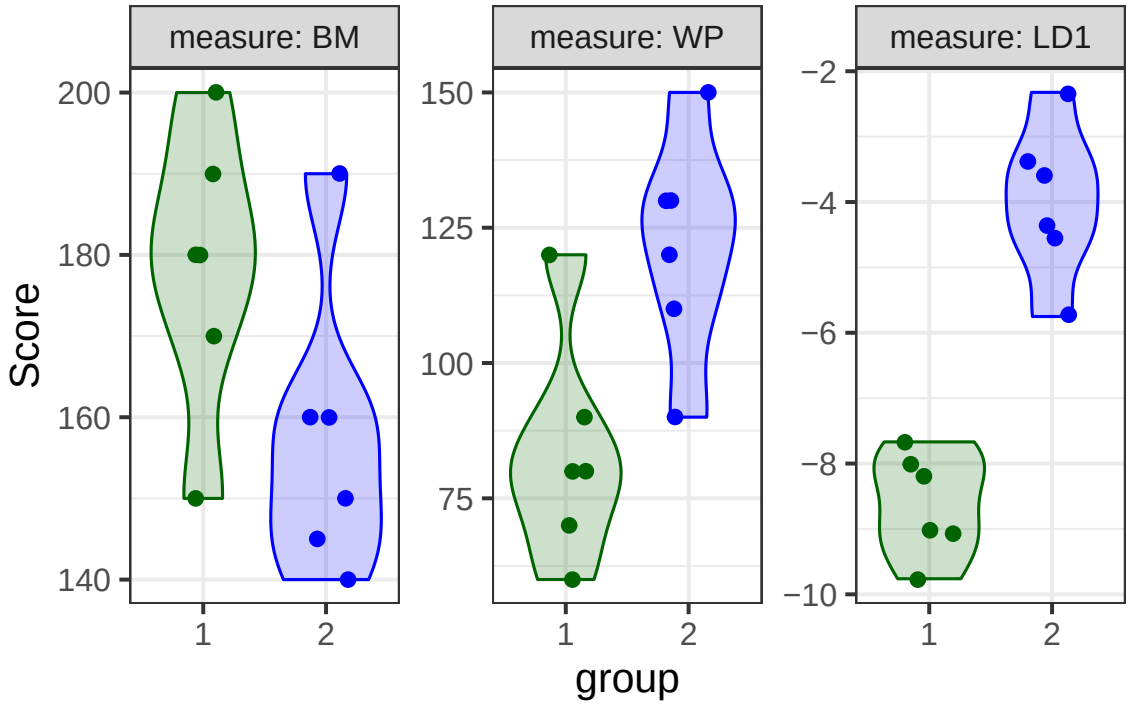


Figure 80: Violin plots comparing group 1 and 2 for the two observed measures and the linear discriminant score.

You can readily see how well the groups are separated on the discriminant axes, relative to the two individual variables.

0.30 More variables

The `mathscore` data gave a simple example with two outcomes to explain the essential ideas behind Hotelling's T^2 and multivariate tests. Multivariate methods become increasingly useful as the number of response variables increases because it is harder to show them all together and see how they relate to differences between groups.

A classic example is the dataset `mbclust::banknote`, containing six size measures made on 100 genuine and 100 counterfeit old-Swiss 1000-franc bank notes (Flury & Riedwyl, 1988). The goal is to see how well the real and fake banknotes can be distinguished. The measures are the `Length` and `Diagonal` lengths of a banknote and the `Left`, `Right`, `Top` and `Bottom` edge margins in mm.

Before considering hypothesis tests, let's look at some exploratory graphics. Figure 81 shows univariate violin and boxplots of each of the measures. To make this plot, faceted by measure, I first reshape the data from wide to long and make `measure` a factor with levels in the order of the variables in the data set.

```
data(banknote, package= "mclust")
banknote |>
  tidyr::gather(key = "measure",
                value = "Size",
                Length:Diagonal) |>
```

```
  mutate(measure = factor(measure,
                          levels = c(names(banknote)[-1]))) |>

  ggplot(aes(x = Status, y = Size, color = Status)) +
  geom_violin(aes(fill = Status), alpha = 0.2) +           # (1)
  geom_jitter(width = .2, size = 1.2) +                    # (2)
  geom_boxplot(width = 0.25,                               # (3)
               linewidth = 1.1,
               color = "black",
               alpha = 0.5) +
  labs(y = "Size (mm)") +
  facet_wrap( ~ measure, scales = "free", labeller = label_both) +
  theme_bw(base_size = 14) +
  theme(legend.position = "top")
```

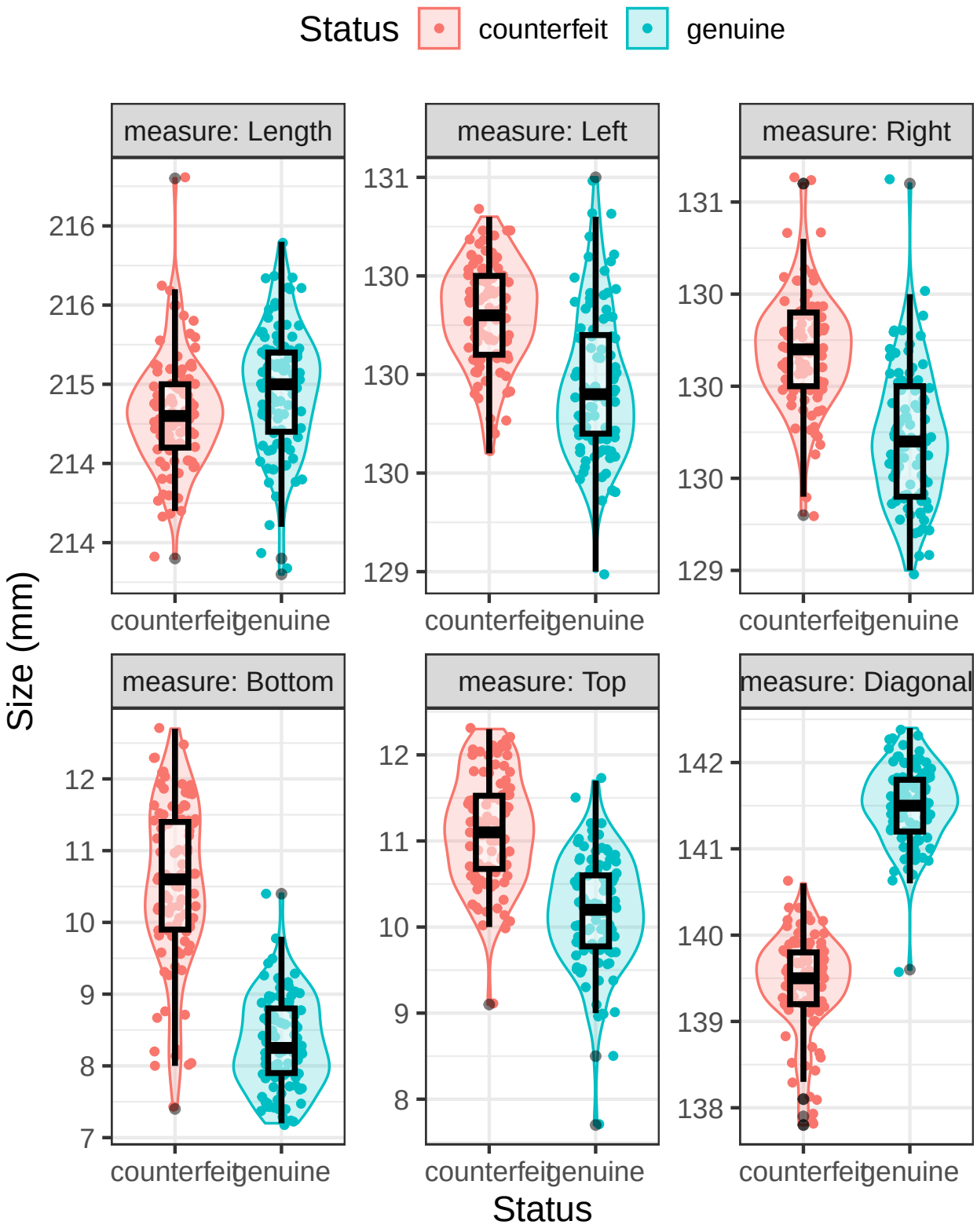


Figure 81: Overlaid violin and boxplots of the banknote variables. The violin plots give a sense of the shapes of the distributions, while the boxplots highlight the center and spread.

A quick glance at Figure 81 shows that the counterfeit and genuine bills differ in their means on most of the measures, with the counterfeit ones slightly larger on Left, Right, Bottom and Top margins. But univariate plots don't give an overall sense of how these variables are related to one another.

Graph craft: Layers and transparency

Figure 81 is somewhat complex, so it is useful to understand the steps needed to make this figure show what I wanted. The plot in each panel contains three layers:

(1) the violin plot based on a density estimate, showing the shape of each distribution;
(2) the data points, but they are jittered horizontally using `geom_jiter()` because otherwise they would all overlap on the X axis;
(3) the boxplot, showing the center (median) and spread (IQR) of each distribution.

In composing graphs with layers, order matters, and also does the `alpha` transparency, because each layer adds data ink on top of earlier ones. I plotted these in the order shown because I wanted the violin plot to provide the background, and the boxplot to show a simple univariate summary, not obscured by the other layers. The `alpha` values allow the data ink to be blended for each layer, and in this case, `alpha = 0.5` for the boxplot let the earlier layers show through.

0.30.1 Biplots

Multivariate relations among these six variables could be explored in data space using scatterplots or other methods, but I turn to my trusty multivariate juicer, a biplot, to give a 2D summary. Two dimensions account for 70% of the total variance of all the banknotes, while three would give 85%.

```
banknote.pca <- prcomp(banknote[, -1], scale = TRUE)
summary(banknote.pca)
#> Importance of components:
#>                          PC1   PC2   PC3   PC4    PC5    PC6
#> Standard deviation     1.716 1.131 0.932 0.671 0.5183 0.4346
#> Proportion of Variance 0.491 0.213 0.145 0.075 0.0448 0.0315
#> Cumulative Proportion  0.491 0.704 0.849 0.924 0.9685 1.0000
```

The biplot in Figure 82 gives a nicely coherent overview, at least in two dimensions. The first component shows the positive correlations among the measures of the margins, where the counterfeit bills are larger than the real ones and a negative correlation of the Diagonal with the other measures. The length of bills only distinguishes the types of banknotes on the second dimension.

```
banknote.pca <- reflect(banknote.pca)
ggbiplot(banknote.pca,
   obs.scale = 1, var.scale = 1,
   groups = banknote$Status,
   ellipse = TRUE,
   ellipse.level = 0.5,
   ellipse.alpha = 0.1,
   ellipse.linewidth = 0,
   varname.size = 4,
   varname.color = "black") +
  labs(fill = "Status",
       color = "Status") +
```

```
    theme_minimal(base_size = 14) +
    theme(legend.position = 'top')
```

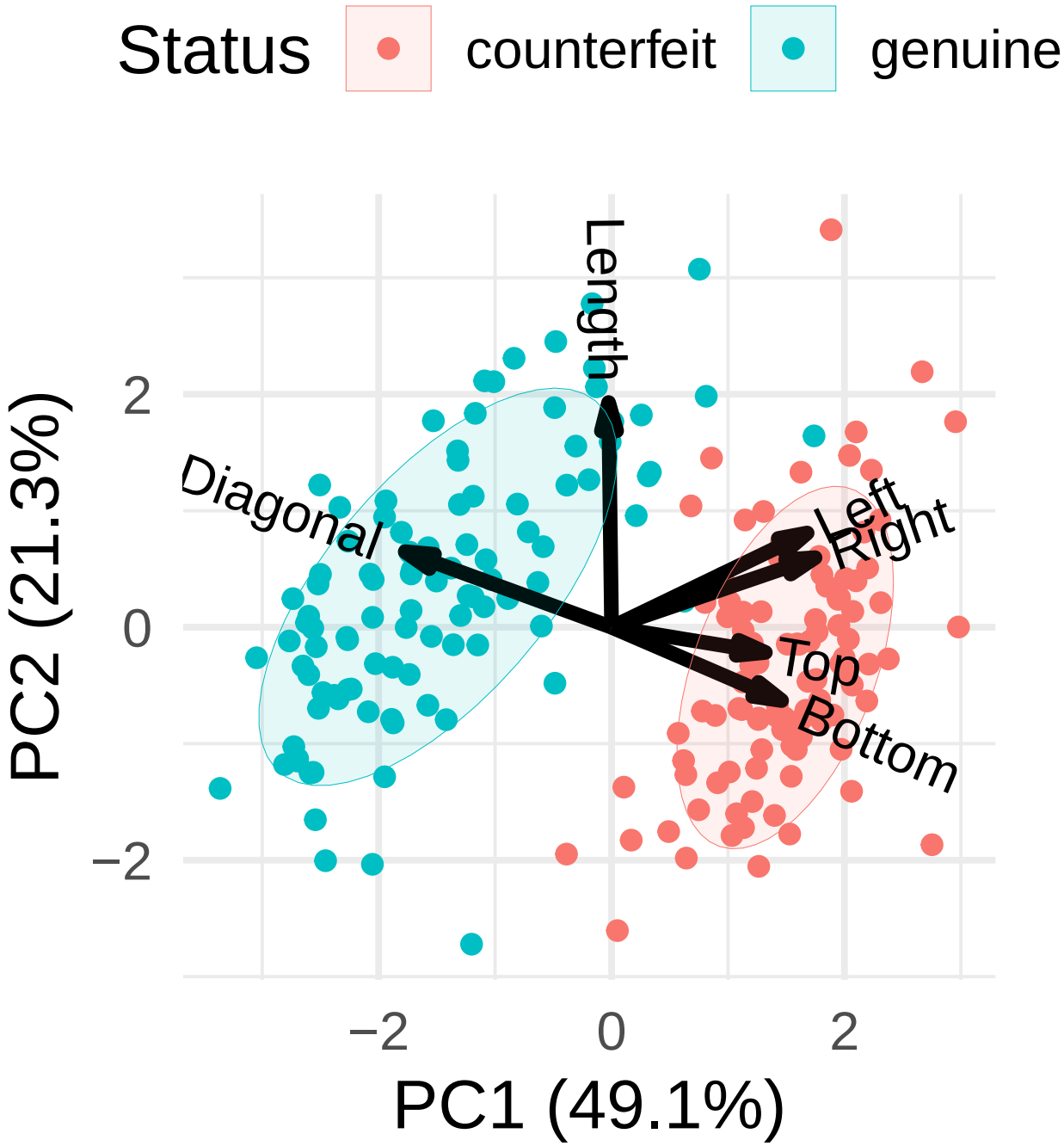


Figure 82: Biplot of the banknote variables, showing how the size measurements are related to each other. The points and data ellipses for the component scores are colored by Status, showing how the counterfeit and genuine bills are distinguished by these measures.

0.30.2 Testing mean differences

As noted above, Hotelling's T^2 is equivalent to a one-way MANOVA, fitting the size measures to the `Status` of the banknotes. `Anova()` reports only the F-statistic based on Pillai's trace criterion.

```
banknote.mlm <- lm(cbind(Length, Left, Right, Bottom, Top, Diagonal) ~ Status,
                   data = banknote)
Anova(banknote.mlm)
#>
#> Type II MANOVA Tests: Pillai test statistic
#>        Df test stat approx F num Df den Df Pr(>F)
#> Status  1     0.924      392      6    193 <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

You can see all the multivariate test statistics with the `summary()` method for `"Anova.mlm"` objects. With two groups, and hence a 1 df test, these all translate into identical F-statistics.

```
summary(Anova(banknote.mlm)) |> print(SSP = FALSE)
#>
#> Type II MANOVA Tests:
#>
#> ----------------------------------------
#>
#> Term: Status
#>
#> Multivariate Tests: Status
#>                  Df test stat approx F num Df den Df Pr(>F)
#> Pillai            1      0.92      392      6    193 <2e-16 ***
#> Wilks             1      0.08      392      6    193 <2e-16 ***
#> Hotelling-Lawley  1     12.18      392      6    193 <2e-16 ***
#> Roy               1     12.18      392      6    193 <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

If you wish, you can extract the univariate t-tests or equivalent $F = t^2$ statistics from the `"mlm"` object using `broom::tidy.mlm()`. What is given as the `estimate` is the difference in the mean for the genuine banknotes relative to the counterfeit ones.

```
broom::tidy(banknote.mlm) |>
  filter(term != "(Intercept)") |>
  dplyr::select(-term) |>
  rename(t = statistic) |>
  mutate(F = t^2) |>
  relocate(F, .after = t)
#> # A tibble: 6 x 6
#>   response estimate std.error      t      F  p.value
#>   <chr>       <dbl>     <dbl>  <dbl>  <dbl>    <dbl>
#> 1 Length      0.146    0.0524   2.79   7.77 5.82e- 3
#> 2 Left       -0.357    0.0445  -8.03  64.5  8.50e-14
#> 3 Right      -0.473    0.0464 -10.2  104.   6.84e-20
#> 4 Bottom     -2.22     0.130  -17.1  292.   7.78e-41
```

```
#> 5 Top        -0.965    0.0909 -10.6  113.   3.85e-21
#> 6 Diagonal    2.07     0.0715  28.9  836.   5.35e-73
```

The individual $F_{(1,198)}$ statistics can be compared to the $F_{(6,193)} = 392$ value for the overall multivariate test. While all of the individual tests are highly significant, the average of the univariate Fs is only 236. The multivariate test gains power by taking the correlations of the size measures into account.

0.31 Variance accounted for: Eta square (η^2)

In a univariate multiple regression model, the coefficient of determination $R^2 = \text{SS}_H/\text{SS}_{\text{Total}}$ gives the proportion of variance accounted for by hypothesized terms in H relative to the total variance. An analog for ANOVA-type models with categorical, group factors as predictors is η^2 (Pearson, 1903), defined as

$$\eta^2 = \frac{\text{SS}_{\text{Between groups}}}{\text{SS}_{\text{Total}}}$$

For multivariate response models, the generalization of η^2 uses multivariate analogs of these sums of squares, $\mathbf{Q}_H$ and $\mathbf{Q}_T = \mathbf{Q}_H + \mathbf{Q}_E$, and there are different different calculations for a single measure corresponding to the various test statistics (Wilks' Λ, etc.), as described in **Chapter XX**.

Let's calculate the η^2 for the multivariate model `banknote.mlm` with `Status` as the only predictor, giving $\eta^2 = 0.92$, or 92% of the total variance.

```
heplots::etasq(banknote.mlm)
#>        eta^2
#> Status 0.924
```

This can be compared (unfavorably) to the principal components analysis and the biplot in Figure 82, where two components accounted for 70% of total variance and it took four PCA dimensions to account for over 90%. The goals of PCA and MANOVA are different, of course, but they are both concerned with accounting for variance of multivariate data. We will meet another multivariate juicer, **canonical discriminant analysis** in **Chapter XX**.

0.32 Exercises

1. The value of Hotelling's T^2 found by `hotelling.test()` is 64.17. The value of the equivalent F statistic found by `Anova()` is 28.9. Verify that Equation 0.4 gives this result.

Packages

```
#> 18 packages used here:
#> base, broom, car, carData, corpcor, datasets, dplyr,
ggbiplot, ggplot2, graphics, grDevices, heplots, Hotelling,
knitr, methods, stats, tidyr, utils
```

0

Visualizing Multivariate Models

Tests of multivariate models, including multivariate analysis of variance (MANOVA) for group differences and multivariate multiple regression (MMRA) can be easily visualized by plots of a hypothesis ("H") data ellipse for the fitted values relative to the corresponding plot of the error ellipse ("E") of the residuals, which I call the HE plot framework.

For more than a few response variables, these result can be projected onto a lower-dimensional "canonical discriminant" space providing an even simpler description.

Packages

In this chapter we use the following packages. Load them now

```
library(car)
library(heplots)
library(ggplot2)
library(dplyr)
library(tidyr)
```

0.33 HE plot framework

Chapter 0.25.3 illustrated the basic ideas of the framework for visualizing multivariate linear models in the context of a simple two group design, using Hotelling's T^2. These are illustrated in Figure 83.

- In data space, each group is summarized by its **data ellipse**, representing the means and covariances.
- Variation against the hypothesis of equal means can be seen by the **H** ellipse in the **HE plot**, representing the data ellipse of the fitted values. Error variance is shown in the **E** ellipse, representing the pooled within-group covariance matrix, $\mathbf{S}_p$ and the data ellipse of the residuals from the model.
- The MANOVA (or Hotelling's T^2) is formally equivalent to a **discriminant analysis**, predicting group membership from the response variables which can be seen in data space.
- This effectively projects the p-dimensional space of the predictors into the smaller **canonical space** that shows the greatest differences among the groups.

For more complex models such as MANOVA with multiple factors or multivariate multivariate regression, there is one **H** ellipse for each term in the model. ...

0.33.1 HE plot details

0.34 Canonical discriminant analysis

```
#> Writing packages to  C:/R/Projects/Vis-MLM-book/bib/pkgs.txt
```

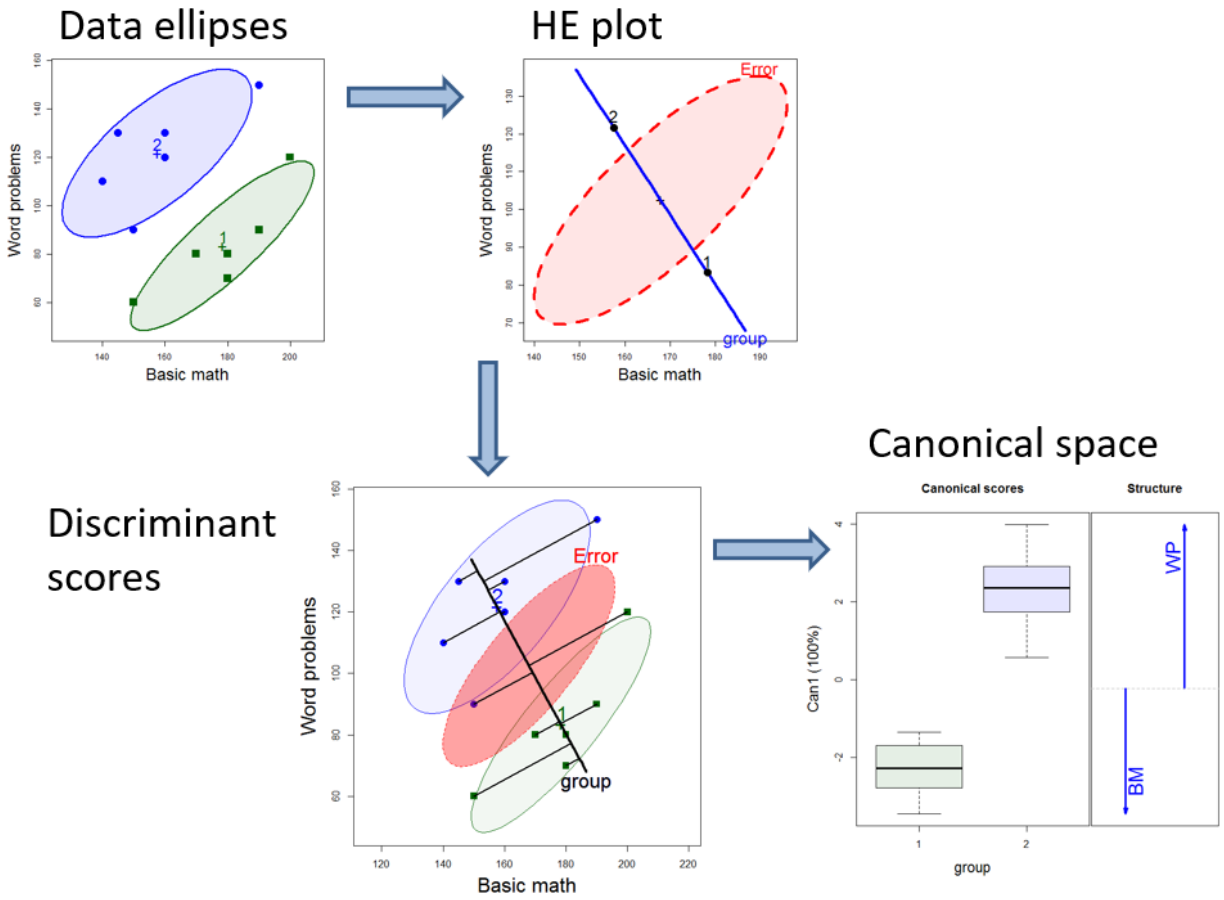


Figure 83: The Hypothesis Error plot framework. *Source*: author

```
#> 15  packages used here:
#>  base, broom, car, carData, datasets, dplyr, ggplot2, graphics, grDevices, heplots, knitr, methods, st
```

0

Brief review of the multivariate linear model

The general multivariate linear model (MLM) can be understood as an extension of the univariate linear model, with the main difference being that there are multiple response variables instead of just one.

In this context, there are multiple techniques that can be applied depending on the structure of the variables at hand. For instance, with one or more continuous predictors and multiple response variables, one could use multivariate regression to obtain estimates useful for prediction. Instead, if the predictors are categorical, multivariate analysis of variance (MANOVA) can be applied to test for differences between groups. Again, this is akin to multiple regression and ANOVA in the univariate context – the same underlying model is utilized, but the tests for terms in the model are multivariate ones for the collection of all response variables, rather than univariate ones for a single response.

In each of these cases, the underlying MLM is given most compactly using the matrix equation,

$$\mathbf{Y}_{n\times p} = \mathbf{X}_{n\times q}\,\mathbf{B}_{q\times p} + \mathbf{U}_{n\times p} \ ,$$

where

- $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_p)$ is the matrix of n observations on p responses;
- $\mathbf{X}$ is the model matrix with columns for q regressors, which typically includes an initial column of 1s for the intercept;
- $\mathbf{B}$ is a matrix of regression coefficients, one column for each response variable; and $\mathbf{U}$ is a matrix of errors.

The structure of the model matrix $\mathbf{X}$ is the same as the univariate linear model, and may contain, therefore,

- quantitative predictors, such as `age`, `income`, years of `education`
- transformed predictors like $\sqrt{\text{age}}$ or $\log$ income
- polynomial terms: age^2, age^3, ... (using `poly(age, k)` in R)
- categorical predictors ("factors"), such as treatment (Control, Drug A, drug B), or sex; internally a factor with `k` levels is transformed to `k-1` dummy (0, 1) variables, representing comparisons with a reference level, typically the first.
- interaction terms, involving either quantitative or categorical predictors, e.g., `age * sex`, `treatment * sex`.

0.35 ANOVA -> MANOVA

0.36 MRA -> MMRA

0.37 ANCOVA -> MANCOVA

0.38 Repeated measures designs

0

Case studies

This chapter presents some complete analyses of datasets that will be prominent in the book. Some of this material may later be moved to earlier chapters.

Packages

In this chapter we use the following packages. Load them now

```
library(car)
library(heplots)
library(candisc)
library(ggplot2)
library(dplyr)
library(tidyr)
library(corrgram)
```

0.39 Neuro- and Social-cognitive measures in psychiatric groups

A Ph.D. dissertation by Laura Hartman (2016) at York University was designed to evaluate whether and how clinical patients diagnosed (on the DSM-IV) as schizophrenic or with schizoaffective disorder could be distinguished from each other and from a normal, control sample on collections of standardized tests in the following domains:

- **Neuro-cognitive**: processing speed, attention, verbal learning, visual learning and problem solving;
- **Social-cognitive**: managing emotions, theory of mind, externalizing, personalizing bias.

The study is an important contribution to clinical research because the two diagnostic categories are subtly different and their symptoms often overlap. Yet, they're very different and often require different treatments. A key difference between schizoaffective disorder and schizophrenia is the prominence of mood disorder involving bipolar, manic and depressive moods. With schizoaffective disorder, mood disorders are front and center. With schizophrenia, that is not a dominant part of the disorder, but psychotic ideation (hearing voices, seeing imaginary people) is.

0.39.1 Research questions

This example is concerned with the following substantitive questions:

- To what extent can patients diagnosed as schizophrenic or with schizoaffective disorder be distinguished from each other and from a normal control sample using a well-validated, comprehensive neurocognitive battery specifically designed for individuals with psychosis (Heinrichs et al., 2015) ?
- If the groups differ, do any of the cognitive domains show particularly larger or smaller differences among these groups?

- Do the neurocognitive measures discriminate among the in the same or different ways? If different, how many separate aspects or dimensions are distinguished?

Apart from the research interest, it could aid diagnosis and treatment if these similar mental disorders could be distinguished by tests in the cognitive domain.

0.39.2 Data

The clinical sample comprised 116 male and female patients who met the following criteria: 1) a diagnosis of schizophrenia ($n = 70$) or schizoaffective disorder ($n = 46$) confirmed by the Structured Clinical Interview for DSM-IV-TR Axis I Disorders; 2) were outpatients; 3) a history free of developmental or learning disability; 4) age 18-65; 5) a history free of neurological or endocrine disorder; and 6) no concurrent diagnosis of substance use disorder. Non-psychiatric control participants ($n = 146$) were screened for medical and psychiatric illness and history of substance abuse and were recruited from three outpatient clinics.

```
data(NeuroCog, package="heplots")
glimpse(NeuroCog)
#> Rows: 242
#> Columns: 10
#> $ Dx        <fct> Schizophrenia, Schizophrenia, Schizophrenia, Sch~
#> $ Speed     <int> 19, 8, 14, 7, 21, 31, -1, 17, 7, 37, 30, 26, 32,~
#> $ Attention <int> 9, 25, 23, 18, 9, 10, 8, 20, 30, 15, 27, 20, 23,~
#> $ Memory    <int> 19, 15, 15, 14, 35, 26, 3, 27, 26, 17, 28, 22, 2~
#> $ Verbal    <int> 33, 28, 20, 34, 28, 29, 20, 30, 26, 33, 34, 33, ~
#> $ Visual    <int> 24, 24, 13, 16, 29, 21, 12, 32, 27, 21, 19, 18, ~
#> $ ProbSolv  <int> 39, 40, 32, 31, 45, 33, 29, 29, 30, 33, 30, 39, ~
#> $ SocialCog <int> 28, 37, 24, 36, 28, 28, 28, 44, 39, 24, 32, 36, ~
#> $ Age       <int> 44, 26, 55, 53, 51, 21, 53, 56, 48, 46, 48, 31, ~
#> $ Sex       <fct> Female, Male, Female, Male, Male, Male, Male, Fe~
```

The diagnostic classification variable is called `Dx` in the dataset. To facilitate answering questions regarding group differences, the following contrasts were applied: the first column compares the control group to the average of the diagnosed groups, the second compares the schizophrenia group against the schizoaffective group.

```
contrasts(NeuroCog$Dx)
#>                 [,1] [,2]
#> Schizophrenia   -0.5    1
#> Schizoaffective -0.5   -1
#> Control          1.0    0
```

In this analysis, we ignore the `SocialCog` variable. The primary focus is on the variables `Attention : ProbSolv`.

0.39.3 A first look

As always, plot the data first! We want an overview of the distributions of the variables to see the centers, spread, shape and possible outliers for each group on each variable.

The plot below combines the use of boxplots and violin plots to give an informative display. As we saw earlier (Chapter XXX), doing this with `ggplot2` requires reshaping the data to long format.

```
# Reshape from wide to long
NC_long <- NeuroCog |>
  dplyr::select(-SocialCog, -Age, -Sex) |>
  tidyr::gather(key = response, value = "value", Speed:ProbSolv)
# view 3 observations per group and measure
NC_long |>
  group_by(Dx) |>
  sample_n(3) |> ungroup()
#> # A tibble: 9 x 3
#>   Dx              response  value
#>   <fct>           <chr>     <int>
#> 1 Schizophrenia   Speed        39
#> 2 Schizophrenia   Visual       21
#> 3 Schizophrenia   Memory       40
#> 4 Schizoaffective ProbSolv     40
#> 5 Schizoaffective Speed        25
#> 6 Schizoaffective Verbal       48
#> 7 Control         Speed        33
#> 8 Control         ProbSolv     43
#> 9 Control         Attention    37
```

In the plot, we take care to adjust the transparency (`alpha`) values for the points, violin plots and boxplots so that all can be seen. Options for `geom_boxplot()` are used to give these greater visual prominence.

```
ggplot(NC_long, aes(x=Dx, y=value, fill=Dx)) +
  geom_jitter(shape=16, alpha=0.8, size=1, width=0.2) +
  geom_violin(alpha = 0.1) +
  geom_boxplot(width=0.5, alpha=0.4,
               outlier.alpha=1, outlier.size = 3, outlier.color = "red") +
  scale_x_discrete(labels = c("Schizo", "SchizAff", "Control")) +
  facet_wrap(~response, scales = "free_y", as.table = FALSE) +
  theme_bw() +
  theme(legend.position="bottom",
        axis.title = element_text(size = rel(1.2)),
        axis.text  = element_text(face = "bold"),
        strip.text = element_text(size = rel(1.2)))
```

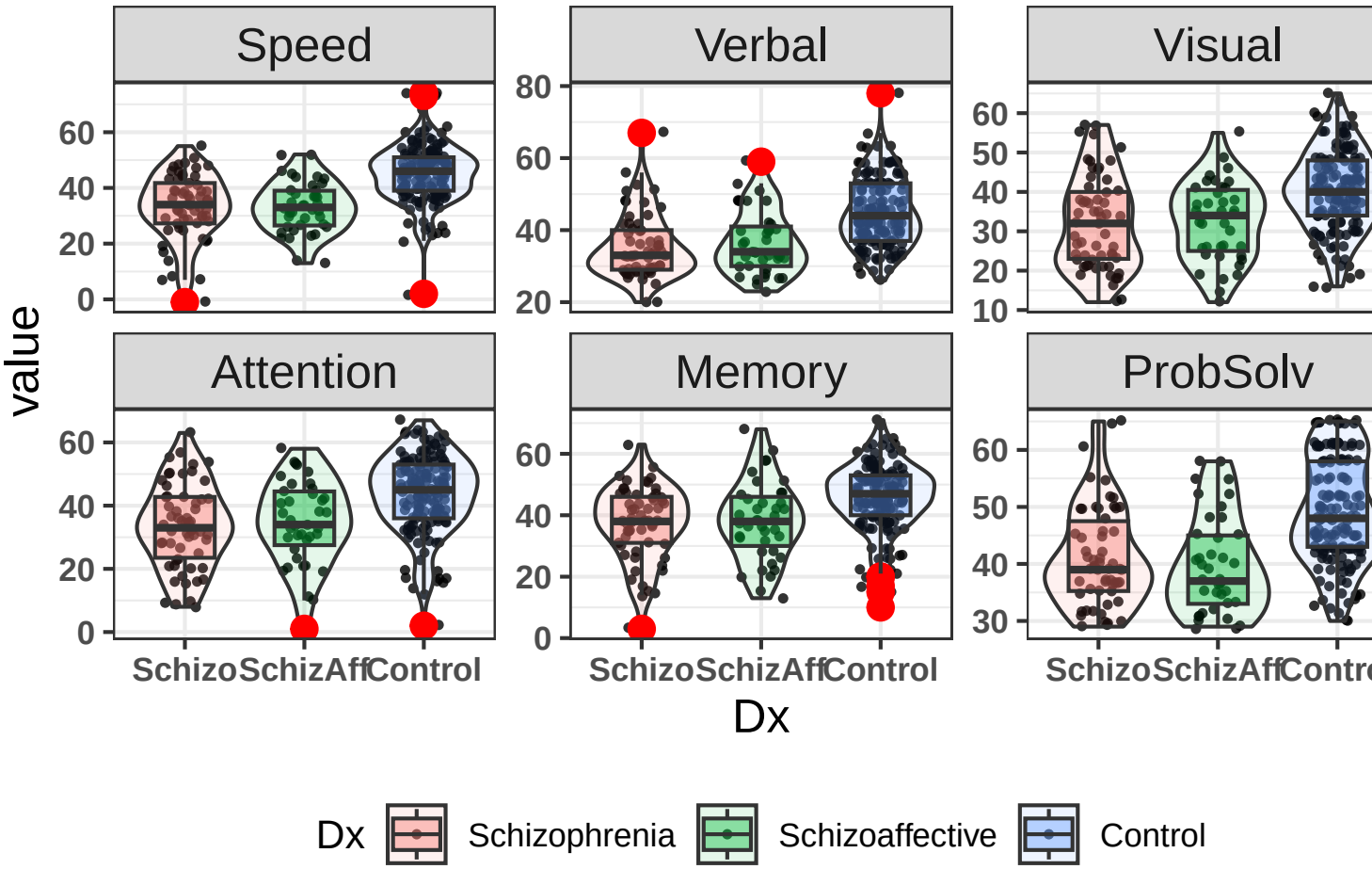


Figure 84: Boxplots and violin plots of the `NeuroCog` data.

We can see that the control participants score higher on all measures, but there is no consistent pattern of medians for the two patient groups. But these univariate summaries do not inform about the relations among variables.

0.39.4 Bivariate views

Corrgram

A corrgram (Friendly, 2002) provides a useful reconnaisance plot of the bivariate correlations in the dataset. It suppresses details, and allows focus on the overall pattern. The `corrgram::corrgram()` function has the ability to enhance perception by permuting the variables in the order of their variable vectors in a biplot, so more highly correlated variables are adjacent in the plot, and example of *effect ordering* for data displays (Friendly & Kwan, 2003).

The plot below includes all variables except for `Dx` group. There are a number of `panel.*` functions for choosing how the correlation for each pair is rendered.

```
NeuroCog |>
  select(-Dx) |>
  corrgram(order = TRUE,
```

```
        diag.panel = panel.density,
        upper.panel = panel.pie)
```

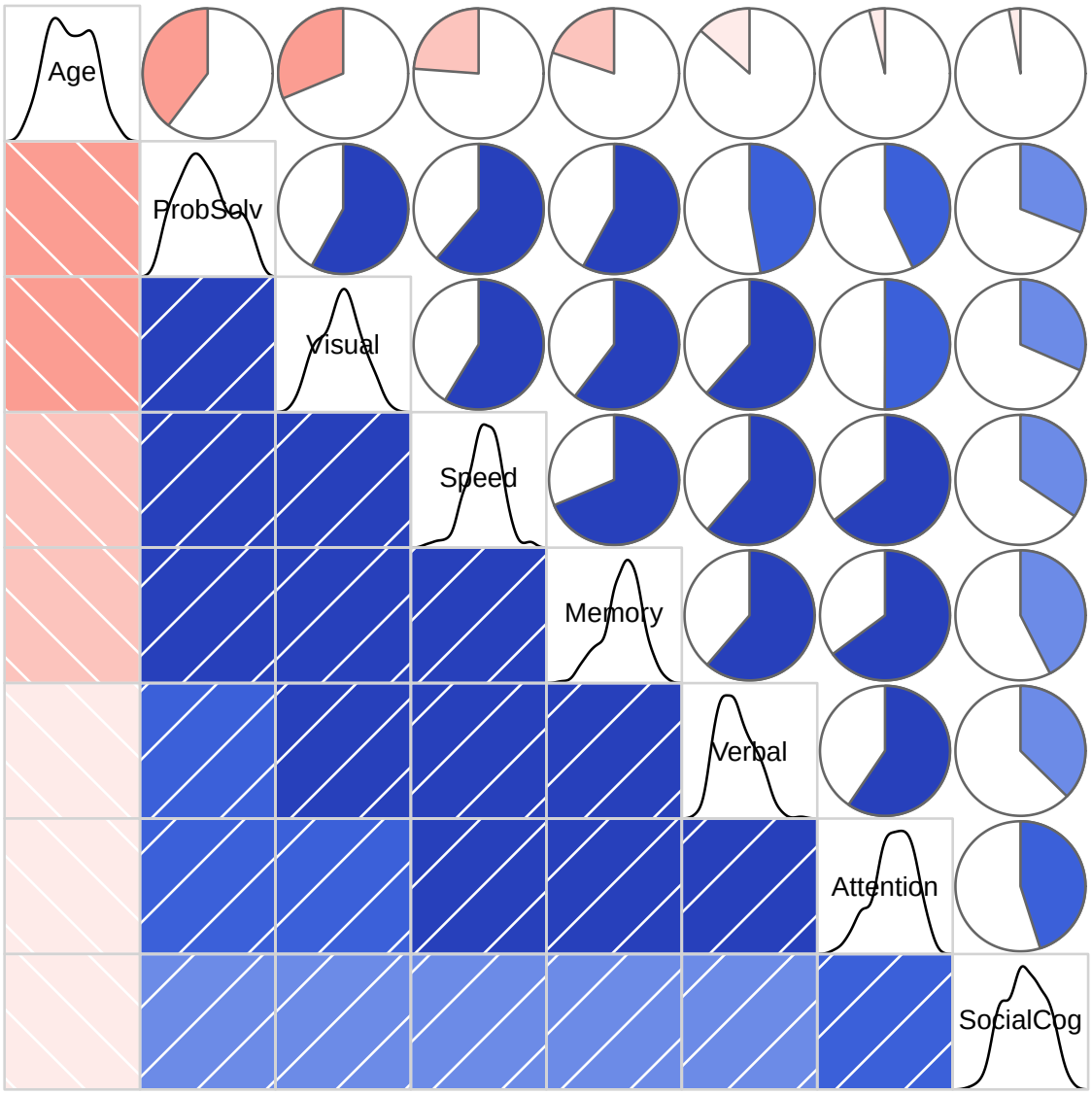


Figure 85: corrgram of the `NeuroCog` data. The upper and lower triangles use two different ways of encoding the value of the correlation for each pair of variables.

In this plot you can see that adjacent variables are more highly correlated than those more widely separated. The diagonal panels show that most variables are reasonably symmetric in their distributions. `Age`, not included in this analysis is negatively correlated with the others: older participants tend to do less well on these tests.

Scatterplot matrix

A scatterplot matrix gives a more detailed overview of all pairwise relations. The plot below suppresses the data points and summarizes the relation using data ellipses and regression lines. The model syntax, `~ Speed + ... |Dx`, treats `Dx` as a conditioning variable (similar to the use of the `color` aestheic in `ggplot2`) giving

a separate data ellipse and regression line for each group. (The legend is suppressed here. The groups are Schizophrenic, SchizoAffective, Normal.)

```
scatterplotMatrix(~ Speed + Attention + Memory + Verbal + Visual + ProbSolv | Dx,
  data=NeuroCog,
  plot.points = FALSE,
  smooth = FALSE,
  legend = FALSE,
  col = scales::hue_pal()(3),
  ellipse=list(levels=0.68))
```

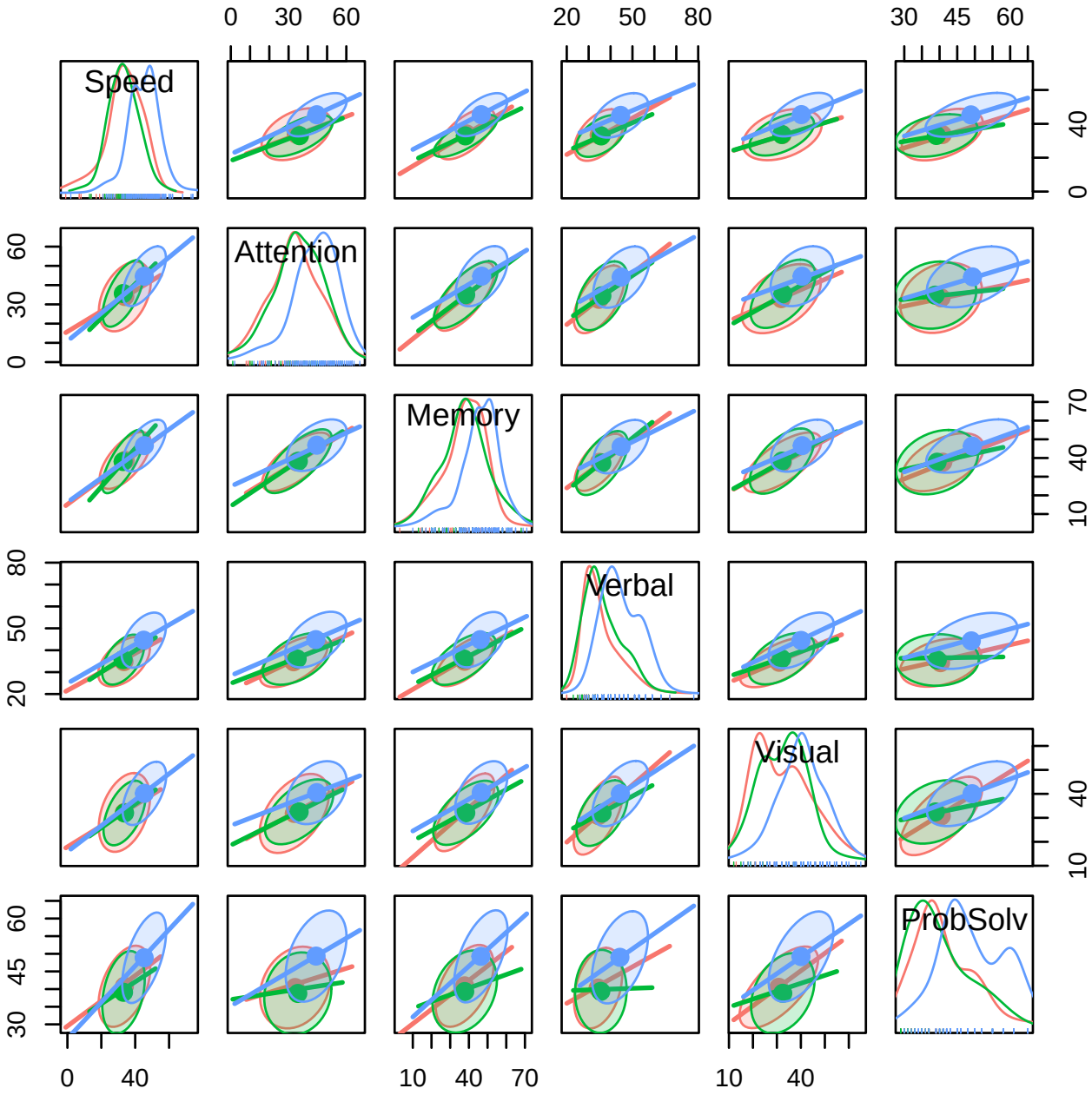


Figure 86: Scatterplot matrix of the `NeuroCog` data. Points are suppressed here, focusing on the data ellipses and regression lines. Colors for the groups: Schizophrenic (red), SchizoAffective (green), Normal (blue)

In this figure, we can see that the regression lines have similar slopes and similar data ellipses for the groups, though with a few exceptions.

TODO: Should we add biplot here?

0.40 Fitting the MLM

We proceed to fit the one-way MANOVA model.

```
NC.mlm <- lm(cbind(Speed, Attention, Memory, Verbal, Visual, ProbSolv) ~ Dx,
             data=NeuroCog)
Anova(NC.mlm)
#>
#> Type II MANOVA Tests: Pillai test statistic
#>    Df test stat approx F num Df den Df  Pr(>F)
#> Dx  2     0.299     6.89     12    470 1.6e-11 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The first research question is captured by the contrasts for the `Dx` factor shown above. We can test these with `car::linearHypothesis()`. The contrast `Dx1` for control vs. the diagnosed groups is highly significant,

```
# control vs. patients
print(linearHypothesis(NC.mlm, "Dx1"), SSP=FALSE)
#>
#> Multivariate Tests:
#>                  Df test stat approx F num Df den Df  Pr(>F)
#> Pillai            1     0.289     15.9      6    234 2.8e-15 ***
#> Wilks             1     0.711     15.9      6    234 2.8e-15 ***
#> Hotelling-Lawley  1     0.407     15.9      6    234 2.8e-15 ***
#> Roy               1     0.407     15.9      6    234 2.8e-15 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

but the second contrast, `Dx2`, comparing the schizophrenic and schizoaffective group, is not.

```
# Schizo vs SchizAff
print(linearHypothesis(NC.mlm, "Dx2"), SSP=FALSE)
#>
#> Multivariate Tests:
#>                  Df test stat approx F num Df den Df Pr(>F)
#> Pillai            1     0.006    0.249      6    234   0.96
#> Wilks             1     0.994    0.249      6    234   0.96
#> Hotelling-Lawley  1     0.006    0.249      6    234   0.96
#> Roy               1     0.006    0.249      6    234   0.96
```

0.40.1 HE plot

So the question becomes: how to understand these results.
`heplot()` shows the visualization of the multivariate model in the space of two response variables (the first two by default). The result (Figure 87) tells a very simple story: The control group performs higher on higher measures than the diagnosed groups, which do not differ between themselves.

(For technical reasons, to abbreviate the group labels in the plot, we need to `update()` the MLM model after the labels are reassigned.)

```
# abbreviate levels for plots
NeuroCog$Dx <- factor(NeuroCog$Dx,
                      labels = c("Schiz", "SchAff", "Contr"))
NC.mlm <- update(NC.mlm)
```

```
op <- par(mar=c(5,4,1,1)+.1)
heplot(NC.mlm,
       fill=TRUE, fill.alpha=0.1,
       cex.lab=1.3, cex=1.25)
par(op)
```

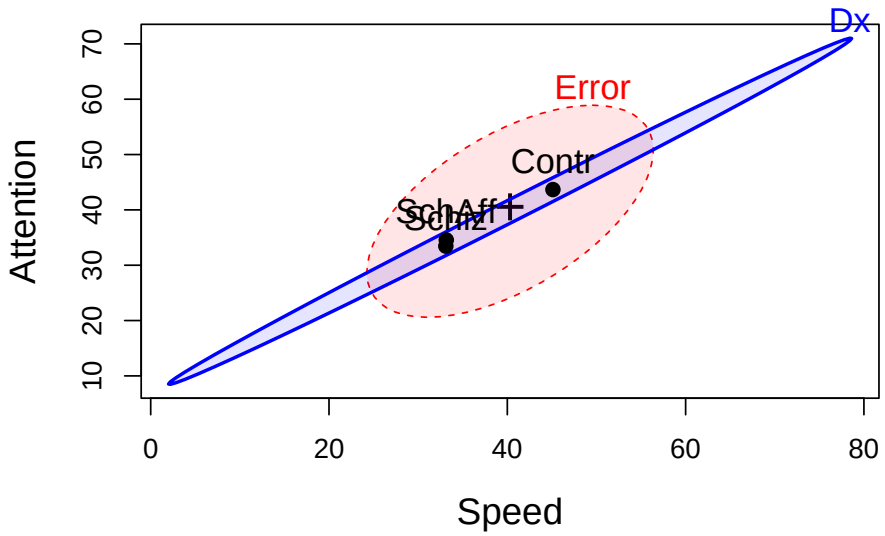


Figure 87: HE plot of Speed and Attention in the MLM for the `NeuroCog` data. The labeled points show the means of the groups on the two variables. The blue H ellipse for groups indicates the strong positive correlation of the group means.

This pattern is consistent across all of the response variables, as we see from a plot of `pairs(NC.mlm)`:

```
pairs(NC.mlm,
      fill=TRUE, fill.alpha=0.1,
      var.cex=2)
```

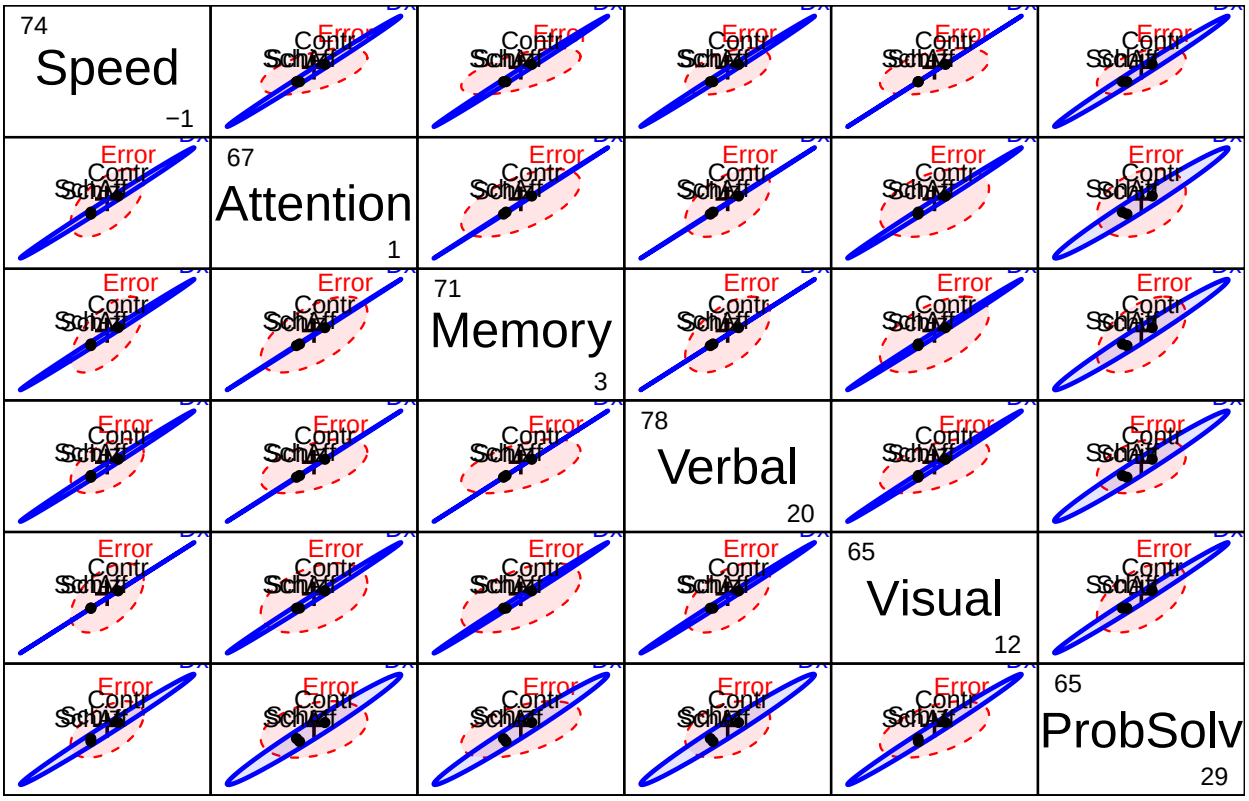


Figure 88: HE plot matrix of the MLM for `NeuroCog` data.

0.40.2 Canonical space

We can gain further insight, and a simplified plot showing all the response variables by projecting the MANOVA into the canonical space, which is entirely 2-dimensional (because $df_h = 2$). However, the output from `candisc()` shows that 98.5% of the mean differences among groups can be accounted for in one canonical dimension. ::: {.cell layout-align="center"}

```
NC.can <- candisc(NC.mlm)
NC.can
#>
#> Canonical Discriminant Analysis for Dx:
#>
#>    CanRsq Eigenvalue Difference Percent Cumulative
#> 1 0.29295    0.41433      0.408    98.5       98.5
#> 2 0.00625    0.00629      0.408     1.5      100.0
#>
#> Test of H0: The canonical correlations in the
#> current row and all that follow are zero
#>
#>   LR test stat approx F numDF denDF Pr(> F)
#> 1        0.703     7.53    12   468   9e-13 ***
#> 2        0.994     0.30     5   235    0.91
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

:::

Figure 89 is the result of the `plot()` method for class `"candisc"` objects, that is, the result of calling `plot(NC.can, ...)`. It plots the two canonical scores, $\mathbf{Z}_{n\times 2}$ for the subjects, together with data ellipses for each of the three groups.

```
pos <- c(4, 1, 4, 4, 1, 3)
col <- c("red", "darkgreen", "blue")
op <- par(mar=c(5,4,1,1)+.1)
plot(NC.can,
     ellipse=TRUE,
     rev.axes=c(TRUE,FALSE),
     pch=c(7,9,10),
     var.cex=1.2, cex.lab=1.5, var.lwd=2,  scale=4.5,
     col=col,
       var.col="black", var.pos=pos,
     prefix="Canonical dimension ")
par(op)
```

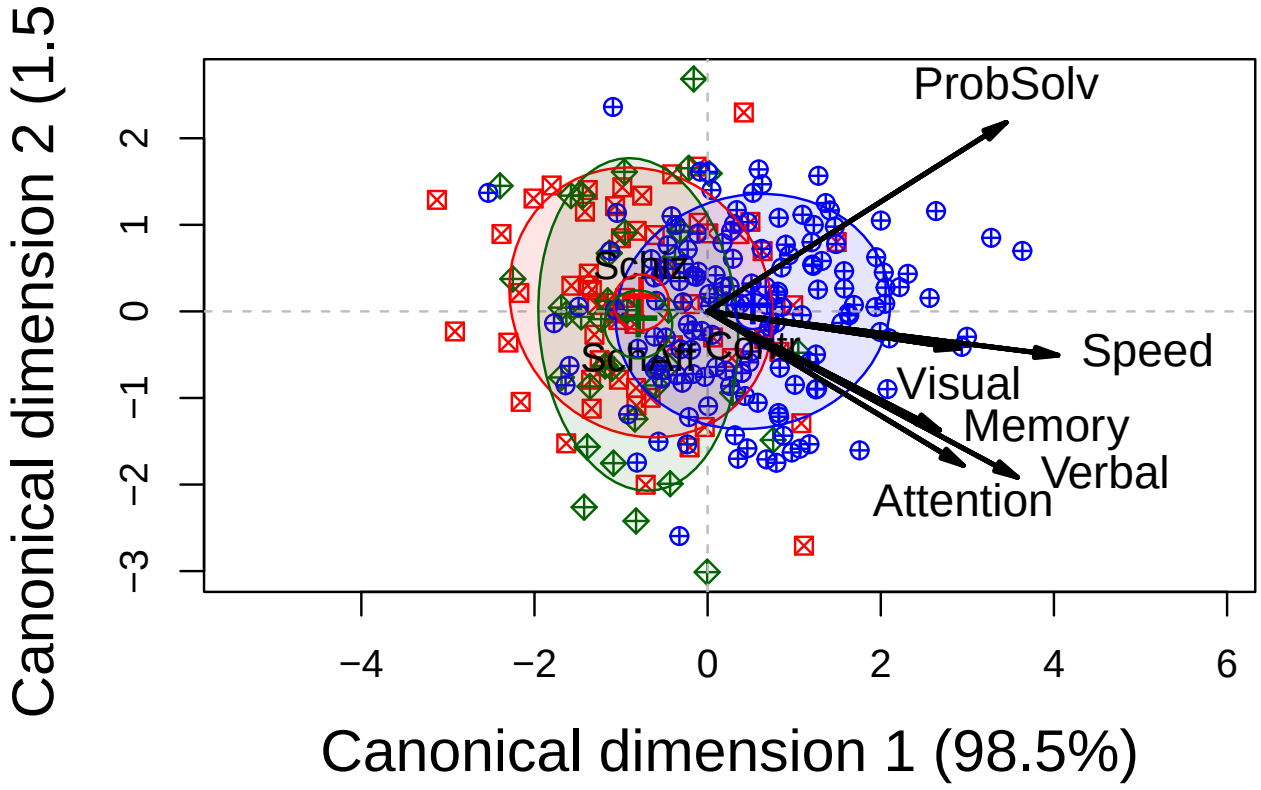


Figure 89: Canonical discriminant plot for the `NeuroCog` data MANOVA. Scores on the two canonical dimensions are plotted, together with 68% data ellipses for each group.

The interpretation of Figure 89 is again fairly straightforward. As noted earlier (REF???), the projections of the variable vectors in this plot on the coordinate axes are proportional to the correlations of the responses with the canonical scores. From this, we see that the normal group differs from the two patient groups, having higher scores on all the neurocognitive variables, most of which are highyl correlated. The problem

solving measure is slightly different, and this, compared to the cluster of memory, verbal and attention, is what distinguishes the schizophrenic group from the schizoaffectives.

The separation of the groups is essentially one-dimensional, with the control group higher on all measures. Moreover, the variables processing speed and visual memory are the purest measures of this dimension, but all variables contribute positively. The second canonical dimension accounts for only 1.5% of group mean differences and is non-significant (by a likelihood ratio test). Yet, if we were to interpret it, we would note that the schizophrenia group is slightly higher on this dimension, scoring better in problem solving and slightly worse on working memory, attention, and verbal learning tasks.

Summary

This analysis gives a very simple description of the data, in relation to the research questions posed earlier:

- On the basis of these neurocognitive tests, the schizophrenic and schizoaffective groups do not differ significantly overall, but these groups differ greatly from the normal controls.
- All cognitive domains distinguish the groups in the same direction, with the greatest differences shown for the variables most closely aligned with the horizontal axis in Figure 89.

0.41 Social cognitive measures

The social cognitive measures were designed to tap various aspects of the perception and cognitive processing of emotions of others. Emotion perception was assessed using a Managing Emotions score from the MCCB. A "theory of mind" (`ToM`) score assessed ability to read the emotions of others from photographs of the eye region of male and female faces. Two other measures, externalizing bias (`ExtBias`) and personalizing bias (`PersBias`) were calculated from a scale measuring the degree to which individuals attribute internal, personal or situational causal attributions to positive and negative social events.

The analysis of the `SocialCog` data proceeds in a similar way: first we fit the MANOVA model, then test the overall differences among groups using `Anova()`. We find that the overall multivariate test is again significant,

```
data(SocialCog, package="heplots")
SC.mlm <-  lm(cbind(MgeEmotions,ToM, ExtBias, PersBias) ~ Dx,
              data=SocialCog)
Anova(SC.mlm)
#>
#> Type II MANOVA Tests: Pillai test statistic
#>    Df test stat approx F num Df den Df  Pr(>F)
#> Dx  2     0.212     3.97      8    268 0.00018 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Testing the same two contrasts using `linearHypothesis()` (results not shown), w e find that the overall multivariate test is again significant, but now *both* contrasts are significant (Dx1: $F(4, 133) = 5.21, p < 0.001$; Dx2: $F(4, 133) = 2.49, p = 0.0461$), the test for `Dx2` just barely.

```
# control vs. patients
print(linearHypothesis(SC.mlm, "Dx1"), SSP=FALSE)
# Schizo vs. SchizAff
print(linearHypothesis(SC.mlm, "Dx2"), SSP=FALSE)
```

These results are important, because, if they are reliable and make sense substantively, they imply that

patients with schizophrenia and schizoaffective diagnoses *can* be distinguished by their performance on tasks assessing social perception and cognition. This was potentially a new finding in the literature on schizophrenia.

As we did above, it is useful to visualize the nature of these differences among groups with HE plots for the `SC.mlm` model. Each contrast has a corresponding **H** ellipse, which we can show in the plot using the `hypotheses` argument. With a single degree of freedom, these degenerate ellipses plot as lines.

```
op <- par(mar=c(5,4,1,1)+.1)
heplot(SC.mlm,
       hypotheses=list("Dx1"="Dx1", "Dx2"="Dx2"),
       fill=TRUE, fill.alpha=.1,
       cex.lab=1.5, cex=1.2)
par(op)
```

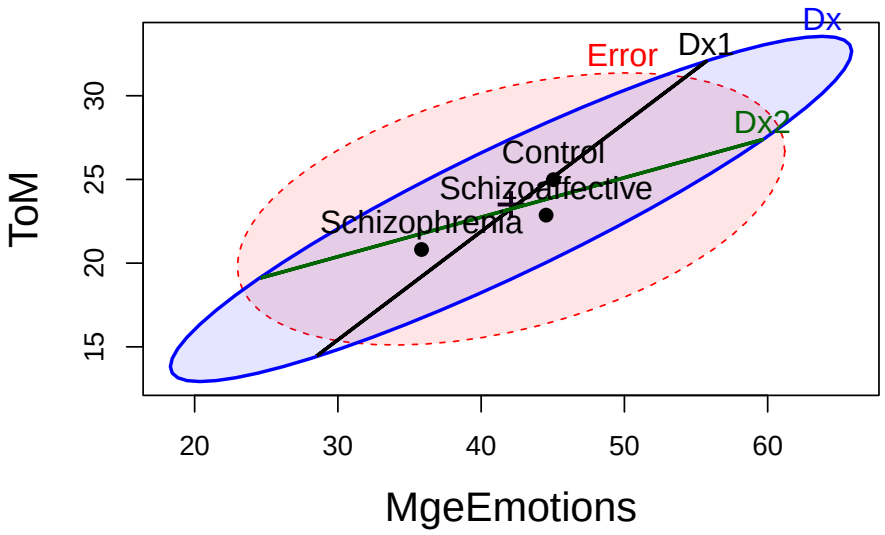


Figure 90: HE plot of Speed and Attention in the MLM for the `SocialCog` data. The labeled points show the means of the groups on the two variables. The lines for Dx1 and Dx2 show the tests of the contrasts among groups.

It can be seen that the three group means are approximately equally spaced on the `ToM` measure, whereas for `MgeEmotions`, the control and schizoaffective groups are quite similar, and both are higher than the schizophrenic group. This ordering of the three groups was somewhat similar for the other responses, as we could see in a `pairs(SC.mlm)` plot.

0.41.1 Model checking

Normally, we would continue this analysis, and consider other HE and canonical discriminant plots to further interpret the results, in particular the relations of the cognitive measures to group differences, or perhaps an analysis of the relationships between the neuro- and social-cognitive measures. We don't pursue this here for reasons of length, but this example actually has a more important lesson to demonstrate.

Before beginning the MANOVA analyses, extensive data screening was done by the client using SPSS, in which all the response *and* predictor variables were checked for univariate normality and multivariate normality

(MVN) for both sets. This traditional approach yielded a huge amount of tabular output and no graphs, and did not indicate any major violation of assumptions.[10]

A simple visual test of MVN and the possible presence of multivariate outliers is related to the theory of the data ellipse: Under MVN, the squared Mahalanobis distances $D^2_M(\mathbf{y}) = (\mathbf{y} - \bar{\mathbf{y}})' \, \mathbf{S}^{-1} \, (\mathbf{y} - \bar{\mathbf{y}})$ should follow a χ^2_p distribution. Thus, a quantile-quantile plot of the ordered D^2_M values vs. corresponding quantiles of the χ^2 distribution should approximate a straight line (Cox, 1968; Healy, 1968). Note that this should be applied to the *residuals* from the model – `residuals(SC.mlm)` – and not to the response variables directly.

`heplots::cqplot()` implements this for `"mlm"` objects Calling this function for the model `SC.mlm` produces Figure 91. It is immediately apparent that there is one extreme multivariate outlier; three other points are identified, but the remaining observations are nearly within the 95% confidence envelope (using a robust MVE estimate of $\mathbf{S}$).

```
op <- par(mar=c(5,4,1,1)+.1)
cqplot(SC.mlm, method="mve",
       id.n=4,
       main="",
       cex.lab=1.25)
par(op)
```

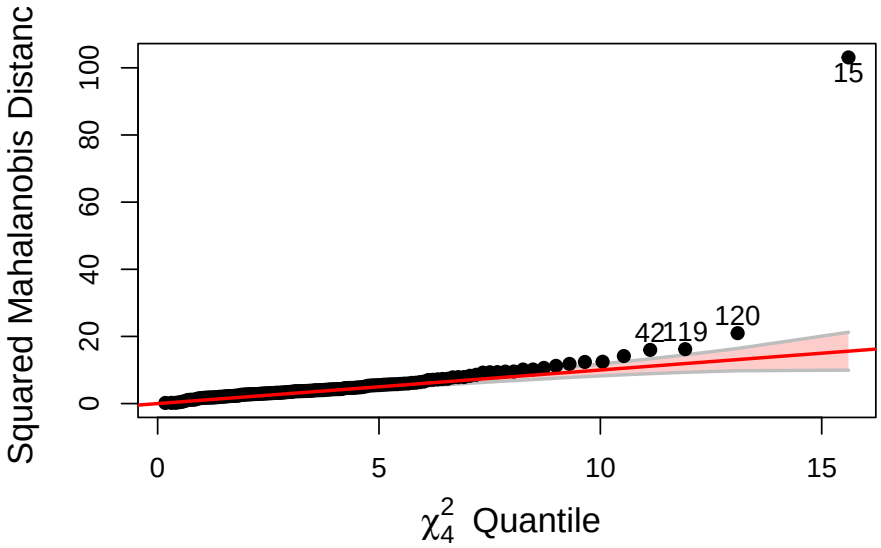


Figure 91: Chi-square quantile-quantile plot for residuals from the model `SC.mlm`. The confidence band gives a point-wise 95% envelope, providing information about uncertainty. One extreme multivariate outlier is highlighted.

Further checking revealed that this was a data entry error where one case (15) in the schizophrenia group had a score of -33 recorded on the `ExtBias` measure, whose valid range was (-10, +10). In R, it is very easy to re-fit a model to a subset of observations (rather than modifying the dataset itself) using `update()`. The result of the overall Anova and the test of `Dx1` were unchanged; however, the multivariate test for the most interesting contrast `Dx2` comparing the schizophrenia and schizoaffective groups became non-significant at the $\alpha = 0.05$ level ($F(4, 133) = 2.18, p = 0.0742$).

[10] Actually, multivariate normality of the predictors in $\mathbf{X}$ is not required in the MLM. This assumption applies only to the conditional values $\mathbf{Y} \mid \mathbf{X}$, i.e., that the errors $\mathbf{u}'_i \sim \mathcal{N}_p(\mathbf{0}, \mathbf{\Sigma})$ with constant covariance matrix. Moreover, the widely used MVN test statistics, such as Mardia's (1970) test based on multivariate skewness and kurtosis are known to be quite sensitive to mild departures in kurtosis (Mardia, 1974) which do not threaten the validity of the multivariate tests.

```
SC.mlm1 <- update(SC.mlm,
                  subset=rownames(SocialCog)!="15")

Anova(SC.mlm1)
print(linearHypothesis(SC.mlm1, "Dx1"), SSP=FALSE)
print(linearHypothesis(SC.mlm1, "Dx2"), SSP=FALSE)
```

0.41.2 Canonical HE plot

This outcome creates a bit of a quandry for further analysis (do univariate follow-up tests? try a robust model?) and reporting (what to claim about the `Dx2` contrast?) that we don't explore here. Rather, we proceed to attempt to interpret the MLM with the aid of canonical analysis and a canonical HE plot. The canonical analysis of the model `SC.mlm1` now shows that both canonical dimensions are significant, and account for 83.9% and 16.1% of between group mean differences respectively.

```
SC.can1 <- candisc(SC.mlm1)
SC.can1
#>
#> Canonical Discriminant Analysis for Dx:
#>
#>   CanRsq Eigenvalue Difference Percent Cumulative
#> 1 0.1645     0.1969      0.159    83.9       83.9
#> 2 0.0364     0.0378      0.159    16.1      100.0
#>
#> Test of H0: The canonical correlations in the
#> current row and all that follow are zero
#>
#>   LR test stat approx F numDF denDF Pr(> F)
#> 1        0.805     3.78     8   264 0.00032 ***
#> 2        0.964     1.68     3   133 0.17537
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
op <- par(mar=c(5,4,1,1)+.1)
heplot(SC.can1,
    fill=TRUE, fill.alpha=.1,
    hypotheses=list("Dx1"="Dx1", "Dx2"="Dx2"),
    lwd = c(1, 2, 3, 3),
    col=c("red", "blue", "darkgreen", "darkgreen"),
    var.lwd=2,
    var.col="black",
    label.pos=c(3,1),
    var.cex=1.2,
    cex=1.25, cex.lab=1.2,
    scale=2.8,
    prefix="Canonical dimension ")
par(op)
```

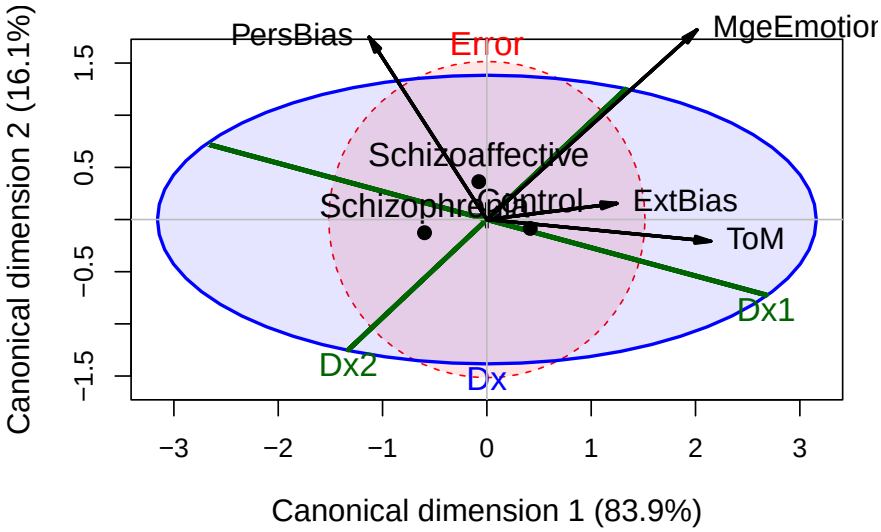


Figure 92: Canonical HE plot for the corrected `SocialCog` MANOVA. The variable vectors show the correlations of the responses with the canonical variables. The embedded green lines show the projections of the **H** ellipses for the contrasts `Dx1` and `Dx2` in canonical space.

The HE plot version of this canonical plot is shown in Figure 92. Because the `heplot()` method for a `"candisc"` object refits the original model to the **Z** canonical scores, it is easy to also project other linear hypotheses into this space. Note that in this view, both the `Dx1` and `Dx2` contrasts project outside **E** ellipse.[11].

This canonical HE plot has a very simple description:

- Dimension 1 orders the groups from control to schizoaffective to schizophrenia, while dimension 2 separates the schizoaffective group from the others;
- Externalizing bias and theory of mind contributes most to the first dimension, while personal bias and managing emotions are more aligned with the second; and,
- The relations of the two contrasts to group differences and to the response variables can be easily read from this plot.

```
#cat("Packages used here:\n")
write_pkgs(file = .pkg_file)
#> 17  packages used here:
#>  base, broom, candisc, car, carData, corrgram, datasets, dplyr, ggplot2, graphics, grDevices, heplo
```

[11]The direct application of significance tests to canonical scores probably requires some adjustment because these are computed to have the optimal between-group discrimination.

0

Visualizing Tests for Equality of Covariance Matrices

> *To make the preliminary test on variances is rather like putting to sea in a rowing boat to find out whether conditions are sufficiently calm for an ocean liner to leave port.* — G. E. P. Box (1953)

This chapter concerns the extension of tests of homogeneity of variance from the classical univariate ANOVA setting to the analogous multivariate (MANOVA) setting. Such tests are a routine but important aspect of data analysis, as particular violations can drastically impact model estimates (Lix & Keselman, 1996).

We provide some answers to the following questions:

- **Visualization**: How can we visualize differences among group variances and covariance matrices, perhaps in a way that is analogous to what is done to visualize differences among group means? As will be illustrated, differences among covariance matrices can be comprised of spread in overall size ("scatter") and shape ("orientation"). These can be seen in data space with data ellipses, particularly if the data is centered by shifting all groups to the grand mean,

- **Low-D views**: When there are more than a few response variables, what low-dimensional views can show the most interesting properties related to the equality of covariance matrices? Projecting the data into the space of the principal components serves well again here. Surprisingly, we will see that the small dimensions contain useful information about differences among the group covariance matrices.

- **Other statistics**: Box's M-test is most widely used. Are there other worthwhile test statistics? We will see that graphics methods suggest alternatives.

The following subsections provide a capsule summary of the issues in this topic. Most of the discussion is couched in terms of a one-way design for simplicity, but the same ideas can apply to two-way (and higher) designs, where a "group" factor is defined as the product combination (interaction) of two or more factor variables. When there are also numeric covariates, this topic can also be extended to the multivariate analysis of covaraiance (MANCOVA) setting. This can be accomplished by applying these techniques to the residuals from predictions by the covariates alone.

Packages

In this chapter we use the following packages. Load them now

```
library(car)
library(heplots)
library(candisc)
library(ggplot2)
library(dplyr)
library(tidyr)
```

0.42 Homogeneity of Variance in Univariate ANOVA

In classical (Gaussian) univariate ANOVA models, the main interest is typically on tests of mean differences in a response y according to one or more factors. The validity of the typical F test, however, relies on the assumption of *homogeneity of variance*: all groups have the same (or similar) variance,

$$\sigma_1^2 = \sigma_2^2 = \cdots = \sigma_g^2 \ .$$

It turns out that the F test for differences in means is relatively robust to violation of this assumption (Harwell et al., 1992), as long as the group sizes are roughly equal.[12]

A variety of classical test statistics for homogeneity of variance are available, including Hartley's F_{max} (Hartley, 1950), Cochran's C (Cochran, 1941),and Bartlett's test (Bartlett, 1937), but these have been found to have terrible statistical properties (Rogan & Keselman, 1977), which prompted Box's famous quote.

Levene (1960) introduced a different form of test, based on the simple idea that when variances are equal across groups, the average absolute values of differences between the observations and group means will also be equal, i.e., substituting an L_1 norm for the L_2 norm of variance. In a one-way design, this is equivalent to a test of group differences in the means of the auxilliary variable $z_{ij} = |y_{ij} - \bar{y}_i|$.

More robust versions of this test were proposed by Brown & Forsythe (1974). These tests substitute the group mean by either the group median or a trimmed mean in the ANOVA of the absolute deviations, and should be almost always preferred to Levene's version. See Conover et al. (1981) for an early review and Gastwirth et al. (2009) for a general discussion of these tests. In what follows, we refer to this class of tests as "Levene-type" tests and suggest a multivariate extension described below (Section 0.43).

0.43 Homogeneity of variance in ANOVA

0.44 Homogeneity of variance in MANOVA

In the MANOVA context, the main emphasis, of course, is on differences among mean vectors, testing

$$H_0 : \mu_1 = \mu_2 = \cdots = \mu_g \ .$$

However, the standard test statistics (Wilks' Lambda, Hotelling-Lawley trace, Pillai-Bartlett trace, Roy's maximum root) rely upon the analogous assumption that the within-group covariance matrices for all groups are equal,

$$\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \cdots = \mathbf{\Sigma}_g \ .$$

Insert pairs covEllipses for penguins data

To preview the main example, Figure 93 shows data ellipses for the main size variables in the `palmerpenguins::penguins` data.

They covariance ellipses look pretty similar in size, shape and orientation. But what does Box's M test (described below) say? As you can see, it concludes strongly against the null hypothesis.

```
boxM(cbind(bill_length, bill_depth, flipper_length, body_mass) ~ species, data=peng)
#>
```

[12] If group sizes are greatly unequal **and** homogeneity of variance is violated, then the F statistic is too liberal (p values too large) when large sample variances are associated with small group sizes. Conversely, the F statistic is too conservative if large variances are associated with large group sizes.

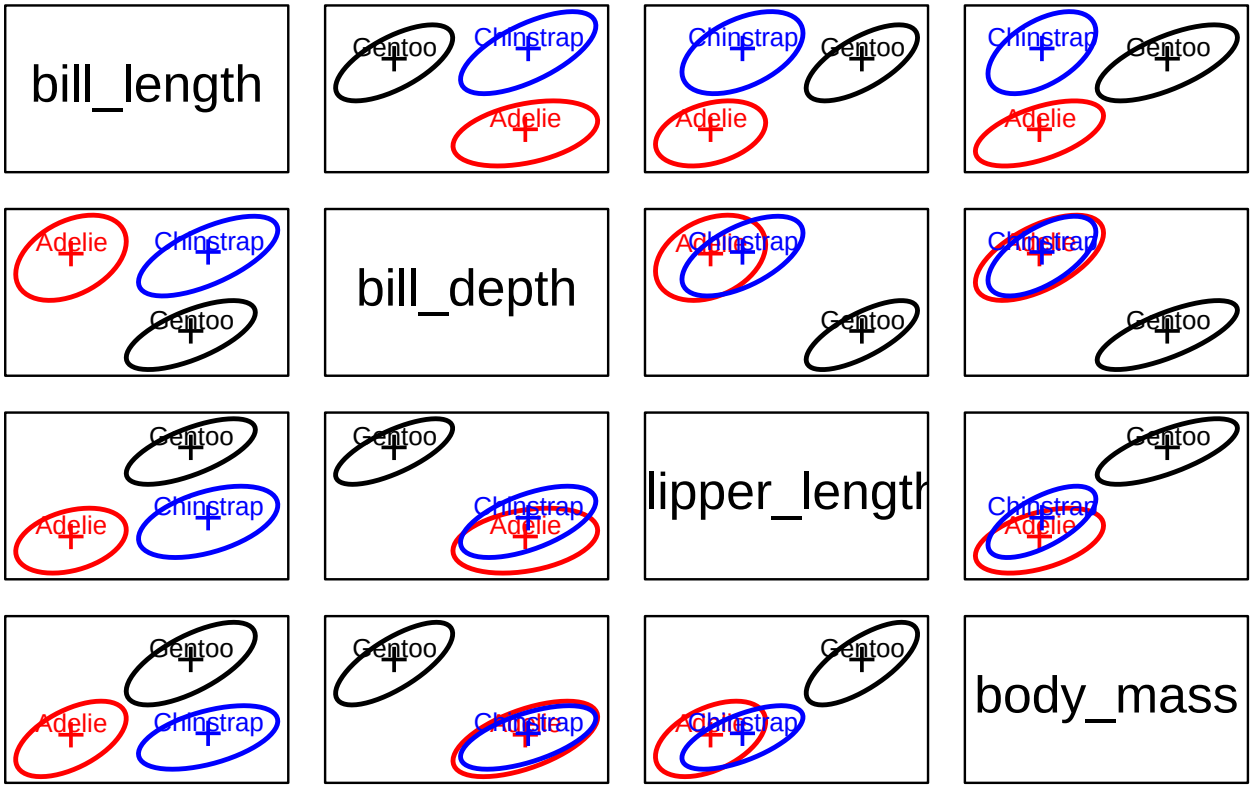


Figure 93: Data ellipses for the penguins data.

```
#>  Box's M-test for Homogeneity of Covariance Matrices
#>
#> data:  Y
#> Chi-Sq (approx.) = 75, df = 20, p-value = 3e-08
```

0.45 Assessing heterogeneity of covariance matrices: Box's M test

Box (1949) proposed the following likelihood-ratio test (LRT) statistic for testing the hypothesis of equal covariance matrices,

$$M = (N - g) \ln \mid \mathbf{S}_p \mid \; - \sum_{i=1}^{g} (n_i - 1) \ln \mid \mathbf{S}_i \mid \; ,$$

{eq-boxm}

where $N = \sum n_i$ is the total sample size and $\mathbf{S}_p = (N - g)^{-1} \sum_{i=1}^{g} (n_i - 1)\mathbf{S}_i$ is the pooled covariance matrix. M can thus be thought of as a ratio of the determinant of the pooled $\mathbf{S}_p$ to the geometric mean of the determinants of the separate $\mathbf{S}_i$.

In practice, there are various transformations of the value of M to yield a test statistic with an approximately known distribution (Timm, 1975). Roughly speaking, when each $n_i > 20$, a χ^2 approximation is often used; otherwise an F approximation is known to be more accurate.

Asymptotically, $-2 \ln(M)$ has a χ^2 distribution. The χ^2 approximation due to Box (1949, 1950) is that

$$X^2 = -2(1 - c_1) \ln(M) \quad \sim \quad \chi^2_{df}$$

with $df = (g-1)p(p+1)/2$ degrees of freedom, and a bias correction constant:

$$c_1 = \left(\sum_i \frac{1}{n_i - 1} - \frac{1}{N-g} \right) \frac{2p^2 + 3p - 1}{6(p+1)(g-1)} \; .$$

In this form, Bartlett's test for equality of variances in the univariate case is the special case of Box's M when there is only one response variable, so Bartlett's test is sometimes used as univariate follow-up to determine which response variables show heterogeneity of variance.

Yet, like its univariate counterpart, Box's test is well-known to be highly sensitive to violation of (multivariate) normality and the presence of outliers. For example, Tiku & Balakrishnan (1984) concluded from simulation studies that the normal-theory LRT provides poor control of Type I error under even modest departures from normality. O'Brien (1992) proposed some robust alternatives, and showed that Box's normal theory approximation suffered both in controlling the null size of the test and in power. Zhang & Boos (1992) also carried out simulation studies with similar conclusions and used bootstrap methods to obtain corrected critical values.

0.46 Visualizing heterogeneity

The goal of this chapter is to use the above background as a platform for discussing approaches to visualizing and testing the heterogeneity of covariance matrices in multivariate designs. While researchers often rely on a single number to determine if their data have met a particular threshold, such compression will often obscure interesting information, particularly when a test concludes that differences exist, and one is left to wonder "why?' '. It is within this context where, again, visualizations often reign supreme. In fact, we find it somewhat surprising that this issue has not been addressed before graphically in any systematic way.
TODO: cut this down

In what follows, we propose three visualization-based approaches to questions of heterogeneity of covariance in MANOVA designs:

(a) direct visualization of the information in the $\mathbf{S}_i$ and $\mathbf{S}_p$ using *data ellipsoids* to show size and shape as minimal schematic summaries;

(b) a simple dotplot of the components of Box's M test: the log determinants of the $\mathbf{S}_i$ together with that of the pooled $\mathbf{S}_p$. Extensions of these simple plots raise the question of whether measures of heterogeneity other than that captured in Box's test might also be useful; and,

(c) the connection between Levene-type tests and an ANOVA (of centered absolute differences) suggests a parallel with a multivariate extension of Levene-type tests and a MANOVA. We explore this with a version of Hypothesis-Error (HE) plots we have found useful for visualizing mean differences in MANOVA designs.

```
#> Writing packages to  C:/R/Projects/Vis-MLM-book/bib/pkgs.txt
#> 16  packages used here:
#>  base, broom, candisc, car, carData, datasets, dplyr, ggplot2, graphics, grDevices, heplots, knitr, me
```

References

Abbott, E. A. (1884). *Flatland: A romance of many dimensions*. Buccaneer Books.

Aluja, T., Morineau, A., & Sanchez, G. (2018). *Principal component analysis for data science*. https://pca4ds.github.io/

Andrews, D. F. (1972). Plots of high dimensional data. *Biometrics*, *28*, 123–136.

Bartlett, M. S. (1937). Properties of sufficiency and statistical tests. *Proceedings of the Royal Society of London. Series A*, *160*(901), 268–282. https://doi.org/10.2307/96803

Becker, R. A., Cleveland, W. S., & Shyu, M.-J. (1996). The visual design and control of trellis display. *Journal of Computational and Graphical Statistics*, *5*(2), 123–155.

Belsley, D. A. (1991). *Conditioning diagnostics: Collinearity and weak data in regression*. Wiley.

Belsley, D. A., Kuh, E., & Welsch, R. E. (1980). *Regression diagnostics: Identifying influential data and sources of collinearity*. John Wiley; Sons.

Biecek, P., Baniecki, H., Krzyzinski, M., & Cook, D. (2023). *Performance is not enough: A story of the rashomon's quartet*. https://arxiv.org/abs/2302.13356

Box, G. E. P. (1949). A general distribution theory for a class of likelihood criteria. *Biometrika*, *36*(3-4), 317–346. https://doi.org/10.1093/biomet/36.3-4.317

Box, G. E. P. (1950). Problems in the analysis of growth and wear curves. *Biometrics*, *6*, 362–389.

Box, G. E. P. (1953). Non-normality and tests on variances. *Biometrika*, *40*(3/4), 318–335. https://doi.org/10.2307/2333350

Brown, M. B., & Forsythe, A. B. (1974). Robust tests for equality of variances. *Journal of the American Statistical Association*, *69*(346), 364–367. https://doi.org/10.1080/01621459.1974.10482955

cagne, M. (1885). *Coordonnées parallèles et axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèlles*. Gauthier-Villars. http://historical.library.cornell.edu/cgi-bin/cul.math/docviewer?did=00620001&seq=3

Cajori, F. (1926). Origins of fourth dimension concepts. *The American Mathematical Monthly*, *33*(8), 397–406. https://doi.org/10.1080/00029890.1926.11986607

Cattell, R. B. (1966). The scree test for the number of factors. *Multivariate Behavioral Research*, *1*(2), 245–276. https://doi.org/10.1207/s15327906mbr0102_10

Chambers, J. M., & Hastie, T. J. (1991). *Statistical models in s* (p. 624). Chapman & Hall/CRC.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, *74*, 829–836.

Cleveland, W. S. (1985). *The elements of graphing data*. Wadsworth Advanced Books.

Cleveland, W. S., & Devlin, S. J. (1988). Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, *83*, 596–610.

Cleveland, W. S., & McGill, R. (1984). Graphical perception: Theory, experimentation and application to the development of graphical methods. *Journal of the American Statistical Association*, *79*, 531–554.

Cleveland, W. S., & McGill, R. (1985). Graphical perception and graphical methods for analyzing scientific data. *Science*, *229*, 828–833.

Cochran, W. G. (1941). The distribution of the largest of a set of estimated variances as a fraction of their total. *Annals of Eugenics*, *11*(1), 47–52. https://doi.org/10.1111/j.1469-1809.1941.tb02271.x

Conover, W. J., Johnson, M. E., & Johnson, M. M. (1981). A comparative study of tests for homogeneity of variances, with applications to the outer continental shelf bidding data. *Technometrics*, *23*(4), 351–361. https://doi.org/10.1080/00401706.1981.10487680

Cook, R. D., & Weisberg, S. (1982). *Residuals and influence in regression*. Chapman; Hall.

Cox, D. R. (1968). Notes on some aspects of regression analysis. *Journal of the Royal Statistical Society Series A*, *131*, 265–279.

Curran, J., & Hersh, T. (2021). *Hotelling: Hotelling's t^2 test and variants*. https://CRAN.R-project.org/package=Hotelling

Davies, R., Locke, S., & D'Agostino McGowan, L. (2022). *datasauRus: Datasets from the datasaurus dozen*. https://CRAN.R-project.org/package=datasauRus

Davis, C. (1990). Body image and weight preoccupation: A comparison between exercising and non-exercising women. *Appetite*, *16*(1), 84. https://doi.org/10.1016/0195-6663(91)90115-9

Dempster, A. P. (1969). *Elements of continuous multivariate analysis*. Addison-Wesley.

Duncan, O. D. (1961). A socioeconomic index for all occupations. In Jr. A. J. Reiss, P. K. H. O. D. Duncan, & C. C. North (Eds.), *Occupations and social status*. The Free Press.

Emerson, J. W., Green, W. A., Schloerke, B., Crowley, J., Cook, D., Hofmann, H., & Wickham, H. (2013). The generalized pairs plot. *Journal of Computational and Graphical Statistics*, *22*(1), 79–91. http://www.tandfonline.com/doi/ref/10.1080/10618600.2012.694762

Euler, L. (1758). Elementa doctrinae solidorum. *Novi Commentarii Academiae Scientiarum Petropolitanae*, *4*, 109–140. https://scholarlycommons.pacific.edu/euler-works/230/

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, *7*(2), 179–188. https://doi.org/10.1111/j.1469-1809.1936.tb02137.x

Flury, B., & Riedwyl, H. (1988). *Multivariate statistics: A practical approach*. Chapman & Hall.

Fox, J. (2016). *Applied regression analysis and generalized linear models* (Third edition.). SAGE.

Fox, J. (2020). *Regression diagnostics* (2nd ed.). SAGE Publications, Inc. https://doi.org/10.4135/9781071878651

Fox, J., & Monette, G. (1992). Generalized collinearity diagnostics. *Journal of the American Statistical Association*, *87*(417), 178–183.

Fox, J., & Weisberg, S. (2018). *An r companion to applied regression* (Third). SAGE Publications. https://books.google.ca/books?id=uPNrDwAAQBAJ

Fox, J., Weisberg, S., & Price, B. (2023). *Car: Companion to applied regression*. https://CRAN.R-project.org/package=car

Friendly, M. (1991). *SAS System for statistical graphics* (1st ed.). SAS Institute. http://www.sas.com/service/doc/pubcat/uspubcat/ind_files/56143.html

Friendly, M. (1994). Mosaic displays for multi-way contingency tables. *Journal of the American Statistical Association*, *89*, 190–200. http://www.jstor.org/stable/2291215

Friendly, M. (2002). Corrgrams: Exploratory displays for correlation matrices. *The American Statistician*, *56*(4), 316–324. https://doi.org/10.1198/000313002533

Friendly, M. (2022). The life and works of andré-michel guerry, revisited. *Sociological Spectrum*, *42*(4-6), 233–259. https://doi.org/10.1080/02732173.2022.2078450

Friendly, M. (2023). *vcdExtra: Vcd extensions and additions*. https://friendly.github.io/vcdExtra/

Friendly, M., & Kwan, E. (2003). Effect ordering for data displays. *Computational Statistics and Data Analysis*, *43*(4), 509–539. https://doi.org/10.1016/S0167-9473(02)00290-6

Friendly, M., & Kwan, E. (2009). Where's Waldo: Visualizing collinearity diagnostics. *The American Statistician*, *63*(1), 56–65. https://doi.org/10.1198/tast.2009.0012

Friendly, M., & Meyer, D. (2016). *Discrete data analysis with R: Visualization and modeling techniques for categorical and count data*. Chapman & Hall/CRC.

Friendly, M., Monette, G., & Fox, J. (2013). Elliptical insights: Understanding statistical methods through elliptical geometry. *Statistical Science*, *28*(1), 1–39. https://doi.org/10.1214/12-STS402

Friendly, M., & Wainer, H. (2021). *A history of data visualization and graphic communication*. Harvard University Press. https://doi.org/10.4159/9780674259034

Gabriel, K. R. (1971). The biplot graphic display of matrices with application to principal components analysis. *Biometrics*, *58*(3), 453–467. https://doi.org/10.2307/2334381

Gabriel, K. R. (1981). Biplot display of multivariate matrices for inspection of data and diagnosis. In V. Barnett (Ed.), *Interpreting multivariate data* (pp. 147–173). John Wiley; Sons.

Galton, F. (1886). Regression towards mediocrity in hereditary stature. *Journal of the Anthropological Institute*, *15*, 246–263. http://www.jstor.org/cgi-bin/jstor/viewitem/09595295/dm995266/99p0374f/0

Gannett, H. (1898). *Statistical atlas of the united states, eleventh (1890) census*. U.S. Government Printing Office.

Gastwirth, J. L., Gel, Y. R., & Miao, W. (2009). The impact of Levene's test of equality of variances on statistical theory and practice. *Statistical Science*, *24*(3), 343–360. https://doi.org/10.1214/09-STS301

Gelman, A., Hullman, J., & Kennedy, L. (2023). *Causal quartets: Different ways to attain the same average treatment effect*. http://www.stat.columbia.edu/~gelman/research/unpublished/causal_quartets.pdf

Gorman, K. B., Williams, T. D., & Fraser, W. R. (2014). Ecological sexual dimorphism and environmental variability within a community of antarctic penguins (genus pygoscelis). *PLoS ONE*, *9*(3), e90081. https://doi.org/10.1371/journal.pone.0090081

Gower, J. C., & Hand, D. J. (1996). *Biplots*. Chapman & Hall.

Gower, J. C., Lubbe, S. G., & Roux, N. J. L. (2011). *Understanding biplots*. Wiley. http://books.google.ca/books?id=66gQCi5JOKYC

Greenacre, M. (1984). *Theory and applications of correspondence analysis*. Academic Press.

Greenacre, M. (2010). *Biplots in practice*. Fundación BBVA. https://books.google.ca/books?id=dv4LrFP7U/_EC

Guerry, A.-M. (1833). *Essai sur la statistique morale de la France*. Crochard.

Hahsler, M., Buchta, C., & Hornik, K. (2023). *Seriation: Infrastructure for ordering objects using seriation*. https://github.com/mhahsler/seriation

Hartigan, J. A. (1975a). *Clustering algorithms*. John Wiley; Sons.

Hartigan, J. A. (1975b). Printer graphics for clustering. *Journal of Statistical Computing and Simulation*, *4*, 187–213.

Hartley, H. O. (1950). The use of range in analysis of variance. *Biometrika*, *37*(3–4), 271–280. https://doi.org/10.1093/biomet/37.3-4.271

Hartman, L. I. (2016). *Schizophrenia and schizoaffective disorder: One condition or two?* [PhD dissertation]. York University.

Harwell, M. R., Rubinstein, E. N., Hayes, W. S., & Olds, C. C. (1992). Summarizing monte carlo results in methodological research: The one- and two-factor fixed effects ANOVA cases. *Journal of Educational and Behavioral Statistics*, *17*(4), 315–339. https://doi.org/10.3102/10769986017004315

Healy, M. J. R. (1968). Multivariate normal plotting. *Journal of the Royal Statistical Society Series C*, *17*(2), 157–161.

Heinrichs, R. W., Pinnock, F., Muharib, E., Hartman, L., Goldberg, J., & McDermid Vaz, S. (2015). Neurocognitive normality in schizophrenia revisited. *Schizophrenia Research: Cognition*, *2*(4), 227–232. https://doi.org/10.1016/j.scog.2015.09.001

Hoaglin, D. C., & Welsch, R. E. (1978). The hat matrix in regression and ANOVA. *The American Statistician*, *32*(1), 17–22. https://doi.org/10.1080/00031305.1978.10479237

Hofmann, H., VanderPlas, S., & Ge, Y. (2022). *Ggpcp: Parallel coordinate plots in the ggplot2 framework*. https://github.com/heike/ggpcp

Horst, A., Hill, A., & Gorman, K. (2022). *Palmerpenguins: Palmer archipelago (antarctica) penguin data*. https://allisonhorst.github.io/palmerpenguins/

Hotelling, H. (1931). The generalization of Student's ratio. *The Annals of Mathematical Statistics*, *2*(3), 360–378. https://doi.org/10.1214/aoms/1177732979

Husson, F., Josse, J., Le, S., & Mazet, J. (2023). *FactoMineR: Multivariate exploratory data analysis and data mining*. http://factominer.free.fr

Husson, F., Le, S., & Pagès, J. (2017). *Exploratory multivariate analysis by example using r*. Chapman & Hall. https://doi.org/10.1201/b21874

Inselberg, A. (1985). The plane with parallel coordinates. *The Visual Computer*, *1*, 69–91.

Kassambara, A., & Mundt, F. (2020). *Factoextra: Extract and visualize the results of multivariate data analyses*. http://www.sthda.com/english/rpkgs/factoextra

Kwan, E., Lu, I. R. R., & Friendly, M. (2009). Tableplot: A new tool for assessing precise predictions. *Zeitschrift für Psychologie / Journal of Psychology*, *217*(1), 38–48. https://doi.org/10.1027/0044-3409.217.1.38

Levene, H. (1960). Robust tests for equality of variances. In I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, & H. B. Mann (Eds.), *Contributions to probability and statistics: Essays in honor of Harold Hotelling* (pp. 278–292). Stanford University Press.

Lix, J. M., L. M. Keselman, & Keselman, H. J. (1996). Consequences of assumption violations revisited:

A quantitative review of alternatives to the one-way analysis of variance F test. *Review of Educational Research*, *66*(4), 579–619. https://doi.org/10.3102/00346543066004579

Longley, J. W. (1967). An appraisal of least squares programs for the electronic computer from the point of view of the user. *Journal of the American Statistical Association*, *62*, 819–841. https://doi.org/https://www.tandfonline.com/doi/abs/10.1080/01621459.1967.10500896

Lüdecke, D., Ben-Shachar, M. S., Patil, I., Waggoner, P., & Makowski, D. (2021). performance: An R package for assessment, comparison and testing of statistical models. *Journal of Open Source Software*, *6*(60), 3139. https://doi.org/10.21105/joss.03139

Lüdecke, D., Ben-Shachar, M. S., Patil, I., Wiernik, B. M., & Makowski, D. (2022). Easystats: Framework for easy statistical modeling, visualization, and reporting. In *CRAN*. https://easystats.github.io/easystats/

Mardia, K. V. (1970). Measures of multivariate skewness and kurtosis with applications. *Biometrika*, *57*(3), 519–530. https://doi.org/http://dx.doi.org/10.2307/2334770

Mardia, K. V. (1974). Applications of some measures of multivariate skewness and kurtosis in testing normality and robustness studies. *Sankhya: The Indian Journal of Statistics, Series B*, *36*(2), 115–128. http://www.jstor.org/stable/25051892

Marquardt, D. W. (1970). Generalized inverses, ridge regression, biased linear estimation, and nonlinear estimation. *Technometrics*, *12*, 591–612.

Marquardt, D. W., & Snee, R. D. (1975). Ridge regression in practice. *The American Statistician*, *29*(1), 3–20. https://doi.org/10.1080/00031305.1975.10479105

Matejka, J., & Fitzmaurice, G. (2017, May). Same stats, different graphs. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3025453.3025912

McGowan, L. D., Gerke, T., & Barrett, M. (2023). Causal inference is not just a statistics problem. *Journal of Statistics and Data Science Education*, 1–9. https://doi.org/10.1080/26939169.2023.2276446

Meyer, D., Zeileis, A., & Hornik, K. (2023). *Vcd: Visualizing categorical data*. https://CRAN.R-project.org/package=vcd

Monette, G. (1990). Geometry of multiple regression and interactive 3-D graphics. In J. Fox & S. Long (Eds.), *Modern methods of data analysis* (pp. 209–256). SAGE Publications.

O'Brien, P. C. (1992). Robust procedures for testing equality of covariance matrices. *Biometrics*, *48*(3), 819–827. http://www.jstor.org/stable/2532347

Otto, J., & Kahle, D. (2023). *Ggdensity: Interpretable bivariate density visualization with ggplot2*. https://jamesotto852.github.io/ggdensity/

Pearson, K. (1896). Contributions to the mathematical theory of evolution—III, regression, heredity and panmixia. *Philosophical Transactions of the Royal Society of London*, *187*, 253–318.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, *6*(2), 559–572.

Pearson, K. (1903). I. Mathematical contributions to the theory of evolution. —XI. On the influence of natural selection on the variability and correlation of organs. *Philosophical Transactions of the Royal Society of London*, *200*(321–330), 1–66. https://doi.org/10.1098/rsta.1903.0001

Pineo, P. O., & Porter, J. (2008). Occupational prestige in canada. *Canadian Review of Sociology*, *4*(1), 24–40. https://doi.org/10.1111/j.1755-618x.1967.tb00472.x

Rogan, J. C., & Keselman, H. J. (1977). Is the ANOVA f-test robust to variance heterogeneity when sample sizes are equal?: An investigation via a coefficient of variation. *American Educational Research Journal*, *14*(4), 493–498. https://doi.org/10.3102/00028312014004493

Sarkar, D. (2023). *Lattice: Trellis graphics for r*. https://lattice.r-forge.r-project.org/

Schloerke, B., Cook, D., Larmarange, J., Briatte, F., Marbach, M., Thoen, E., Elberg, A., & Crowley, J. (2023). *GGally: Extension to ggplot2*. https://ggobi.github.io/ggally/

Scott, D. W. (1992). *Multivariate density estimation: Theory, practice, and visualization*. Wiley.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Chapman & Hall.

Tiku, M. L., & Balakrishnan, N. (1984). Testing equality of population variances the robust way. *Communications in Statistics - Theory and Methods*, *13*(17), 2143–2159. https://doi.org/10.1080/03610928408828818

Timm, N. H. (1975). *Multivariate analysis with applications in education and psychology*. Wadsworth (Brooks/Cole).

VanderPlas, S., Ge, Y., Unwin, A., & Hofmann, H. (2023). Penguins go parallel: A grammar of graphics

framework for generalized parallel coordinate plots. *Journal of Computational and Graphical Statistics*, 1–16. https://doi.org/10.1080/10618600.2023.2195462

Wegman, E. J. (1990). Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, *85*(411), 664–675.

Wei, T., & Simko, V. (2021). *Corrplot: Visualization of a correlation matrix*. https://github.com/taiyun/corrplot

Wood, S. N. (2006). *Generalized additive models: An introduction with r*. Chapman; Hall/CRC Press.

Wright, K. (2021). *Corrgram: Plot a correlogram*. https://kwstat.github.io/corrgram/

Zhang, J., & Boos, D. D. (1992). Bootstrap critical values for testing homogeneity of covariance matrices. *Journal of the American Statistical Association*, *87*(418), 425–429. http://www.jstor.org/stable/2290273

Package used

Colophon

This book was produced using R version 4.2.3 (2023-03-15 ucrt). Fundamental to this was the framework for reproducible documents provided by Yihui Xie's knitr package.

Quarto was used to compile and render the book in HTML and PDF formats.

```
[>] Checking versions of quarto binary dependencies...
      Pandoc version 3.1.1: OK
      Dart Sass version 1.55.0: OK
[>] Checking versions of quarto dependencies......OK
[>] Checking Quarto installation......OK
      Version: 1.3.450
      CodePage: 1252
[>] Checking basic markdown render....OK
[>] Checking Python 3 installation....(None)
      Unable to locate an installed version of Python 3.
      Install Python 3 from https://www.python.org/downloads/
[>] Checking R installation...........OK
      Version: 4.2.3
      LibPaths:
        - C:/R/R-4.2.3/library
      knitr: 1.45
      rmarkdown: 2.25
[>] Checking Knitr engine render......OK
```

Package versions

The principal R package versions used in examples and illustrations are listed below. At the time of writing, most of these were current on CRAN repositories but some development versions are indicated in the `source` column.

package * version date (UTC) lib source bayestestR 0.13.1 2023-04-07 [1] CRAN broom 1.0.5 2023-06-09 [1] CRAN candisc 0.8-6 2021-10-07 [1] CRAN car 3.1-4 2023-07-02 [1] R-Forge carData 3.0-5 2022-01-06 [1] CRAN corpcor 1.6.10 2021-09-16 [1] CRAN correlation 0.8.4 2023-04-06 [1] CRAN corrgram 1.14 2021-04-29 [1] CRAN corrplot 0.92 2021-11-18 [1] CRAN datawizard 0.9.1 2023-12-21 [1] CRAN dplyr * 1.1.4 2023-11-17 [1] CRAN easystats 0.7.0 2023-11-05 [1] CRAN effectsize 0.8.6 2023-09-14 [1] CRAN factoextra 1.0.7 2020-04-01 [1] CRAN FactoMineR 2.9 2023-10-12 [1] CRAN forcats 1.0.0 2023-01-29 [1] CRAN genridge 0.7.0 2023-07-31 [1] local GGally 2.2.0 2023-11-22 [1] CRAN ggbiplot 0.6.2 2024-01-08 [1] CRAN ggdensity 1.0.0 2023-02-09 [1] CRAN ggpcp 0.2.0 2023-05-09 [1] Github (heike/ggpcp@bdb2fea) ggplot2 3.4.4 2023-10-12 [1] CRAN ggrepel 0.9.4 2023-10-13 [1] CRAN heplots 1.6.1 2023-12-09 [1] local Hotelling 1.0-8 2021-09-09 [1] CRAN imager 0.45.2 2023-05-10 [1] CRAN insight 0.19.7 2023-11-26 [1] CRAN knitr * 1.45 2023-10-30 [1] CRAN lubridate 1.9.3 2023-09-27 [1] CRAN magrittr 2.0.3 2022-03-30 [1] CRAN MASS 7.3-60 2023-05-04 [1] CRAN modelbased 0.8.6 2023-01-13 [1] CRAN parameters 0.21.3 2023-11-02 [1] CRAN patchwork 1.2.0 2024-01-08 [1] CRAN performance 0.10.8 2023-10-30 [1] CRAN purrr 1.0.2 2023-08-10 [1] CRAN readr 2.1.4 2023-02-10

[1] CRAN report 0.5.8 2023-12-07 [1] CRAN see 0.8.1 2023-11-03 [1] CRAN stringr 1.5.1 2023-11-14 [1] CRAN tibble 3.2.1 2023-03-20 [1] CRAN tidyr 1.3.0 2023-01-24 [1] CRAN tidyverse 2.0.0 2023-02-22 [1] CRAN vcd 1.4-12 2023-12-29 [1] CRAN VisCollin 0.1.2 2023-09-05 [1] local

[1] C:/R/R-4.2.3/library