Table 1: Control of ggplot2 graphics features. Note that functions such as `xlab()` and `scale_x_continuous()` all have counterparts with `y` in place of `x`. .

	Argument to `qplot()`	`ggplot()` or `qplot()`[1]
Title	`main="mytitle"`	`+ labs(title="mytitle")`
Axes	see `help(qplot)`	`+ scale_x_continuous()`[2] [or `scale_x_discrete()` or `scale_x_date()`]
Axis labels	e.g., `xlab="myxlab"`	`+ xlab("myxlab")`[3]
log axes	`log="x"`, (or `"y"`, or `"xy"`)	`+ scale_x_log10()`[4]
Facets[5]	`facets=sex ~ sport`	`+ facet_grid(sex ~ sport)`
Aspect ratio	e.g., `asp=1`	`+ coord_equal()`[6]
Theme	—	
Graph title	e.g., `main="maintitle"`	`+ ggtitle("mytitle")`

[1] Recall that `quickplot()` (or `qplot()`) returns a `ggplot` object. Functions such as `xlab()` or `scale_x_continuous()` can be used, just as for any other ggplot2 object, to update objects returned by `quickplot()`.

[2] Available arguments include `limits`, `breaks` (locations for the ticks), `labels` (labels for the breaks), and `trans` (e.g., `trans="log"`).

[3] This is an alternative to using `name` (e.g., `name="myxlab"`) as an argument to `scale_x_continuous()` or `scale_x_discrete()`.

[4] This is an alternative to using `trans="log10"` as an argument to `scale_x_continuous()` or `scale_x_discrete()`. Note also `trans="log"` and `trans="log2"`.

[5] Facets give Lattice style *conditioning*.

[6] By default (`ratio=1`), a given distance, e.g., 1cm, represents the same range along both $x-$ and $y-$axes.

[7] Themes control such graphical attributes as background color, gridlines, and size and color of fonts. See `help(ggtheme)` for details of other available themes.

Updates and Corrections (as of April 19, 2018)

Data Analysis and Graphics Using R – An Example-Based Approach, 3^{rd} edn, 2013 reprint

John Maindonald (email: john.maindonald@anu.edu.au) and John Braun

Webpage: `http://www.maths.anu.edu.au/~johnm/r-book/r-book.html`.

Changes are of four types:

- Changes made necessary, or desirable, arising from changes in R functions;
- Corrections to errors;
- Code changes, designed to improve coding style;
- Removal of some obscurities.

Note: Prospects for a 4th edition

An initial draft has been in place since late 2017. Cambridge University Press seems, at the moment, not to be interested in proceeding. New material will be posted on the web shortly.

Graphs that need to be replaced

Figure 2.14, page 64, should be replaced if the figure now given by the existing code is preferred.

Figure 7.10, page 237 (the function used for the smooth is from a package that has been archived; hence use a different package)

Figure 12.2, page 381 (the graph should be printed with `aspect="iso"`)

Figure 12.9, page 403 (the maximum number of features selected should be 23, as in the code that is in the text; the attempt to select more than 23 generates a collinearity warning.)

Chapter 1

page 27, Subsection 1.5.2

Replace the first paragraph, i.e.
The default palette, which can be changed, has eight colors including ‘ ‘white”.
by
The default color palette, shown in Plate 10, attaches the numbers 1, . . . , 8 to the eight colors that are included.

page 27, bullet points in lines 3 to 18 [addition]

Insert as the second to last bullet point:

- Aspect ratio: `asp=1` gives a plot with isometric scaling, i.e., x and y units occupy the same physical distances in both directions.

page 32, following line -9 [additions]

Add, following the first bullet point, the further bullet points:

- Most common types of parameter settings can be handled by using `simpleTheme()` to create a theme, which the argument `par.settings` then supplies to the function that creates the graph. See the code for Figure 15.3 in Subsection 15.5.1.
- Use the argument `aspect` to control aspect. If a number, this specifies the ratio of graph height to width. For isometric scales, specify `aspect="iso"`.

page 33, final 2 lines

Edit the final sentence to read: Use either `.RData` or `.rda` as the extension for such files.

Omit footnote 9.

page 39, line 2

Replace there is extreme value of area that by One extreme value for area

Chapter 2

page 45, line 1 [better coding style]

Replace `attach(fossum)`
by `ftotlngth <- fossum[, "totlngth"]`

page 45, lines 2,4, -17, -15, -12 [better coding style]

Replace `totlngth` by `ftotlngth`

Run together the two paragraphs in lines -9 to -4. Omit for details at the end of the final sentence.

p.473, line -6

Replace The function by The base graphics function.

p.474, *Color* [addition]

Add
The function `adjustcolor()` can be used to set the opacity `alpha` for the color that is returned. Thus with `alpha=0.4`, 60% of the background shows through. Two overlapping points have a combined opacity of 80%, so that 20% of the background shows through. Lattice and ggplot2 graphics functions accept the argument `alpha` directly.

p.481, lines 1 to 7

Delete lines 4 - 7.

Modify (add a comment at the end) line 2, so that it reads:

```
trellis.device(color=TRUE)  # Try also with color=TRUE
```

p.481, following the current line 7 [addition]

Insert
A theme `settings1` that has been created earlier, e.g., by a call to `simpleTheme()`, can be modified thus:

```
settings1[["fontsize"]] <- list(text = 16, points = 8)
## Then do, e.g.
grogplot1 <- update(grogplot0, par.settings=settings1)
```

p.491, Table 2

The current Table 15.2 has been overtaken by changes in *ggplot2*. Replace Table 15.2 by:

p.421, code on lines 3 to 12

The code assumes an updated version of the function `overlapDensity()`. This will be included in the next version of *DAAG* (>=1.21). It is provided as a separate file. Replace the current line 7 by the more explicit code:

```
overlapDensity(sc.rf[tnum==1], sc.rf[tnum==2], ratio=c(1/20, 50),
               ratio.number=TRUE, plotvalues="Density")
```

Replace the current lines 11 and 12 by the more explicit code:

```
overlapDensity(sc.lda[tnum==1], sc.lda[tnum==2], ratio=c(1/20, 50),
               ratio.number=TRUE, plotvalues="Density")
```

p.423, line starting `proba.rf`

Replace `probability` by `prob`

Chapter 14

p.429, line 2

Replace } by)

page 442, line -6:

Replace `scale` by `scales`

page 448, Subsection 14.9.7, line 6:

Replace `tsunits[,2]` by `jobts[,2]` or `jobts[,"Alberta"]`.

page 467, line 10:

Replace Packages can have their own namespaces, with
by
Each package has its own namespace. There can be

Chapter 15

p.473, lines 6-8 (2nd para)

Omit lines 6 - 8.

page 45, line-10, [better coding style]

Replace `par(mfrow=c(1,1)); detach(fossum)`
by `par(mfrow=c(1,1))`

page 47, line 7

Replace `horiz` by `horizontal`

page 51, line 9

Figure 2.7B would be better plotted with `asp=1`, i.e., line 9 becomes:

```
plot(log(brain) ~ log(body), data=Animals, asp=1) # Panel B
```

This ensures that equal changes on a logarithmic scale on the two axes (corresponding to equal relative changes) are the same physical distance apart. (The tick marks in the graph shown are a factor of 10 apart, on both axes.)

page 64, fnote 17, line 3

Edit line 3 to read:

```
## Panel function calls panel.dotplot(), then plots means
```

page 64, fnote 17, lines 6-7

Replace

```
panel.average(x, y, type="p", col="black",
              pch=3, cex=1.25)},
```

by

```
av <- sapply(split(x,y),mean)
ypos <- unique(y)
lpoints(ypos~av, pch=3, cex=1.25, col="black")},
```

[The code in the text still works, but uses a line to join up the means. There is in this instance, where there is an interest in comparing the pattern across the three panels, in leaving the code as is and replacing the graph.]

[There are equivalent replacements on page 123 (footnote 13) and page 261 (lines -4 and -3).]

page 75, exercise 5 [better coding style]

Replace the first line of code (`attach(cuckoohosts)`) by

```
with(cuckoohosts, {
```

Indent the next 5 lines by 5 spaces

Replace the final line of code (`detach(cuckoohosts)`) by

```
})
```

Chapter 3

page 105, line -11

Replace Figure 4.1B by Figure 4.1

Chapter 4

page 123, footnote 13, lines 6-7

Replace

```
            panel.average(x, y, type="p", col="black",
                          pch=3, cex=1.25)},
```

by

```
            av <- sapply(split(x,y),mean)
            ypos <- factor(levels(y), levels=levels(y))
            lpoints(ypos~av, pch=3, cex=1.25, col="black")},
```

[The code in the text still works, but uses a line to join up the means.]

page 131, lines -15 to -12

Replace

```
median.fun <- function(data, indices){median(data[indices])}
## Call boot(), with statistic=median.fun, R = # of resamples
set.seed(23)
(wren.boot <- boot(data = wren, statistic = median.fun, R = 999))
```

by

```
## Call the function boot.ci() , with boot.out=wren.boot
boot.ci(boot.out=wren.boot, type=c("perc"))
```


Replace the third sentence in the 2nd paragraph (halfway down the page) by:
The function `CVbinary()` returns the resubstitution estimate (other names are *internal* and *training* in the list element `training`, repeated in the list element `internal`.

p.395, lines -8 to -4

Replace with:

```
chooseCols <- with(golubInfo, tissue.mf=="PB:f"& cancer=="allB")
df.PBf <- data.frame(t(Golub[ord15, chooseCols, drop=FALSE]))
scores.PBf <- predict(dfB15.lda, newdata=df.PBf, dimen=2)$x
```

p.403, Figure 12.9

Replace with a figure in which the maximum number of features selected is 23, as in the text and in the code.

p.409, lines 8 and 9

Replace

```
 # Alternatively, get primates.dna from the DAAGbio package
primates.dist <- dist.dna(primates.dna, model="F84")
```

with

```
 # Alternatively, work with primateDNA from the DAAGbio package
library(DAAGbio)
primates.dna <- as.DNAbin(primateDNA)
primates.dist <- dist.dna(primates.dna, model="K80")
```

Chapter 13

p.413, footnote 2, line 1

Replace componenet by component

p.416, Figure 13.2

Code for Figure 13.2 is not given. See the separate file.

page 331, lines 8 and 13:

Replace `method="REML"` with `REML=TRUE`

pages 332-333:

Insert replacement pages, as supplied, in place of these.

page 340, fnote 13, line 4:

Replace `scale` by `scales`

page 349, exercise 1:

Replace the final three lines of code, i.e.

```
vars <- c("(block:plot)^2"=as.vector(vcov[["block:plot"]]),
          "sigma^2"=as.vector(attributes(vcov, "sigmaREML")$sc^2))
print(vars)
```

by

```
print(vcov)          # For variances, specify print(vcov^2)
```

Chapter 12

p.381, line -7

Add at end of line `aspect="iso",`, i.e. the line becomes:

```
xyplot(possum.prc$scores[, 2] ~ possum.prc$scores[, 1], aspect="iso",
```

[Also, replace Figure 12.2 with a figure that has been printed with `aspect="iso"`. (The vertical and horizontal scales are in the same units.)]

p.383, line -6 (3^{rd} last line of code)

Replace `value` by `values`

page 388, function `DAAG::CVbinary()`

In line 6, replace `leaf17.cv$cv` by `leaf17.cv$cvhat`. (It is not good practice to abbreviate the names of list elements.)

page 140, Exercise 14, final 2 lines

Replace

```
# See help(bootci.object). The 4th and 5th elements of
# the percent list element hold the interval endpoints.
```

by

```
# The 4th and 5th elements of the list element percent
# hold the interval endpoints. See ?boot.ci
```

Chapter 6

page 177, line -5

Replace The models `nihills.lmnihillsG.lm` by The models `nihills.lm` and `nihillsG.l`

page 186, lines of code preceding subsection 6.3.2

Replace lines 2 and 3 of the code with `influencePlot(allbacks.lm)`

[`influencePlots()` is not a direct replacement for `leverage.plots()`. It gives output that, while different from what is discussed in the text, is at the same more informative.]

page 203, line prior to 6.3 Errors in x

Replace remedy by) recourse

page 216, exercise 14, 1^{st} line of code

Replace } by)

Chapter 7

page 236, Section 7.5.3, line 7

Replace All by Most

page 237, final 13 lines

Replace with:
Monotone polynomials, as fitted using the function `monpol()` from the *MonoPoly* package, will often be satisfactory, as in Figure 7.10.

Code is:

```
library(MonoPoly)
u <- monpol(ohms~juice, data=fruitohms, degree=3)
plot(ohms ~ juice, data=fruitohms, xlab="Apparent juice content (%)",
     ylab="Resistance (ohms)", col="gray40")
ord <- with(fruitohms, order(juice))
lines(fitted(u)[ord] ~ juice[ord], data=fruitohms, col=2)
```

For fitting a monotonic spline curve, see `help(mono.con, package="mgcv")`

[Also, replace Figure 7.10 with a figure that has been generated using the above code. The *monoProc* package has been archived.]

page 238, line -2

Replace `lm` by `gam`

Chapter 8

Replace $15/60 = 0.25$ by $45/60 = 0.75$

page 249, footnote 4, line 2

Add, at the end of the line `asp=1`, i.e., the line becomes

```
plot(northing ~ easting, data=frogs, pch=c(1,16)[frogs$pres.abs+1], asp=1,
```

page 260, line -17

Replace is ("suggesting that is should be") by it.

page 261, line -12

Replace The `Number` is the total number by Here, `Number` is the number

page 318, footnote 3

Replace with

```
print(VarCorr(science2.lmer), comp="Variance", digits=3)
## The component of variance that is labeled 'Residual' is
## the estimate of the within class variance.
```

page 326, lines -10 to -1

Replace line 10 by: The following agree with results from the preceding section:

Replace the code chunk in lines -9 to -2 by:

```
> print(kiwishade.lmer, ranef.comp="Variance", digits=3)
. . . .
Random effects:
 Groups     Name        Variance
 block:plot (Intercept)  2.19
 block      (Intercept)  4.08
 Residual                12.18
Number of obs: 48, groups:  block:plot, 12; block, 3
. . . .
```

Delete the final line.

page 330, line 4:

Replace by:

```
library(R.utils)
levels(tinting$agegp) <- capitalize(levels(tinting$agegp))
```

page 330, lines 7, 11 and 14:

Replace `method="ML"` with `REML=FALSE`

page 330, line 15:

Replace
`method="ML"`
with
`REML=FALSE` (use maximum likelihood in place of REML)

The function `confint()`, as used here, returned confidence intervals for σ_L (row label `.sig01`, random), for σ (row label `.sigma`, random), and for `(Intercept)` (fixed). The `(Intercept)` is the intercept in the fitted model, which estimates the overall mean.

The profile likelihoods, scaled so that the lower 2.5% limit transforms to -1.96 and the upper lower 97.5% limit, can be plotted thus:

```
library(lattice)
print(xyplot(prof.lmer, conf=c(50, 80, 95, 99)/100, between=list(x=0.35)))
```

For variances, the horizontal scales show `Std.Dev.` $= \sqrt{\text{Variance}}$. On the vertical scale, the confidence interval limits are labeled according to the equivalent normal deviates. The 95% confidence interval limits are thus at -1.96 and 1.96.

For details of this and other displays that can be used for the output from the `profile()` method for `merMod` objects, see `help(xyplot.thpr)`.

See `help(confint.merMod)` for details of the `confint` method for `merMod` objects. Alternatives to `method="profile"` are `method="Wald"` or `method="boot"`. The `Wald` method is fast, but based on approximations that can be highly inaccurate. The `boot` method uses repeated fits to suitably constructed bootstrap samples, and can be time consuming. The trustworthiness of results from this method may be questioned if more than an occasional fit fails. See `help(bootMer)` and `help(simulate.merMod)` for further details of `method="boot"`, and for references.

page 316, line 7:

Replace `mcmcsamp()` by `confint()`

page 316, lines 9-18:

Replace by:

```
> ## Use profile likelihood
> pp <- profile(science1.lmer, which="theta_")
>  # which="theta_": all random parameters
>  # which="beta_": fixed effect parameters
> var95 <- confint(pp, level=0.95)^2
>  # Square to get variances in place of SDs
> rownames(var95) <- c("sigma_Class^2", "sigma^2")
> signif(var95, 3)
              2.5 % 97.5 %
sigma_Class^2 0.178  0.511
sigma^2       2.830  3.300
```


page 261, line -5 and -4

Replace

```
panel.average(x, y, pch=3, cex=1.25, type="p", col="gray45")
},
```

by

```
av <- sapply(split(x,y),mean)
ypos <- factor(levels(y), levels=levels(y))
lpoints(ypos~av, pch=3, cex=1.25, col="gray45")},
```

[The code in the text still works, but uses a line to join up the means.]

pages 262-266

Insert replacement pages, as supplied, in place of these.

page 282, exercise 7c, 3rd line of code

Replace `newdat` by `ndf`

page 285, Figures 9.2B & 9.2C

Figure 1 does a better job than Figures 9.2B & 9.2C.

page 287

Replace the final 4 lines of Subsection 9.1.3 (While the plot . . . the needed additional flexibility.) by
Is an AR(2) model adequate, or is something more needed? We now move to investigating use of an autoregressive moving average (ARMA) model, which adds one or more *moving average* terms, to examine whether this makes any worthwhile improvement.

Chapter 9

page 291, line 1

Replace `mfrow=c(3,2)` by `mfrow=c(2,2)`

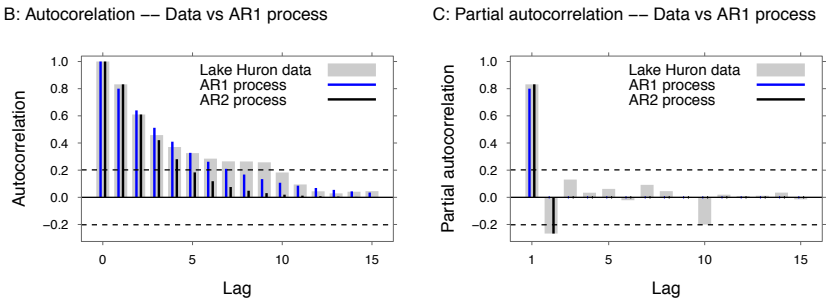

Figure 1: Panel B compares the estimated autocorrelations with the autocorrelations for an AR(1) process, and for an AR(2) process. Panel C repeats the comparison, now for the partial autocorrelations. The dashed horizontal lines are approximate pointwise 5% critical values for the autocorrelations for a pure noise process, i.e., for independent normal data with mean 0. Panel C shows the theoretical autocorrelations for an AR(1) process with $\alpha = 0.8$. For an AR(1) process, partial autocorrelations after the first are all 0 — hence the single lightblue spike at lag 1. For an AR(2) process, partial autocorrelations beyond lag 2 are all 0. — a positive spike at lag 1 is followed by a negative spike at lag 2, with no further spikes.

Chapter 10

The pdf for a rewrite of this chapter is available as a separate file.

General

This chapter is strongly affected by changes in the *lme4* package. The most important is that `msmcsamp()` is, because it did not work reliably, no longer available. The primary function for calculating confidence intervals is `confint()`, using `method="profile"`) or `method="boot"` in preference to the less reliable `method="Wald"`.

The following demonstrate the form of command that one might now expect to use, with objects returned by `lmer()`:

```
library(lme4)
library(DAAG)
science1.lmer <- lmer(like ~ sex + PrivPub + (1 | school:class),
                      data = science, na.action=na.exclude)
  # science1.lmer is then a merMod object
## Output from print()
print(science1.lmer, ranef.comp="Variance", digits=3)
## Output from use of summary() with a merMod object
print(summary(science1.lmer), ranef.comp="Variance", digits=3)
```



```
## Variance components informatioc
print(VarCorr(science1.lmer), comp="Variance", digits=3)
```

See `help(merMod)` and `help(VarCorr)` for further details.

More important Ch 10 changes, with explanatory comment

page 310, following final paragraph

Insert
The object `ant111b.lmer` has the class `merMod`. See `help(merMod)` for details of functions ("methods") that are available for use with this class.

page 312, subsubsection in lines 3 to 26

Replace with:

*Uncertainty in parameter estimates — profile likelihood & alternatives

The limits of acceptance of a likelihood ratio test for the null hypothesis of no change in a parameter value can be used as approximate 95% confidence limits for that parameter. Where the likelihood is a function of more than one parameter, the profile likelihood may be used. For any parameter ψ, the profile likelihood is the function of ψ that is obtained by maximizing the likelihood, for each value of ψ, over values of other parameters.[1]

The function `confint()` can be used to pull together the profile information, calculated using the profile method for `merMod` objects, to create approximate confidence intervals:

```
> prof.lmer <- profile(ant111b.lmer)
> CI95 <- confint(prof.lmer, level=0.95)
> rbind("sigmaL\^{}2"=CI95{[}1,{]}\^{}2, "sigma\^{}2"=CI95{[}2,{]}\^{}2)
            2.5 % 97.5 %
sigmaL^2    0.796   6.94
sigma^2     0.344   1.08
```

A 95% confidence interval for the intercept is:

```
> CI95[3,]
            2.5 % 97.5 %
(Intercept) 3.128   5.46
```

[1] Note that convergence problems will sometimes occur in the calculation of the profile likelihood, generating warning messages.