2: The Limits of Statistical Learning

John H Maindonald

August 20, 2015

Ideas and issues illustrated by the graphs in this vignette

In analyses in the traditions of 'data mining' and 'statistical learning', observations are typically assumed independent. There is a greater use of relatively automated approaches than is usual in many areas of statistical analysis. This limits the scope of models that are considered and rules out of consideration some very important types of analysis. Or, in order to fit the data to this type of analysis, some modest amount of preprocessing of the data may be required. This may be as simple as transforming data values. Or it may require the creation, from the data as it stands, of summary statistic values to which the methods can then be applied. Graphs that are shown here that are intended as starting points for discussing such issues.

1 R Functions for Creating Chapter 2 Figures

```
fig2.1 <-
function (form = speed ~ Year, data = subset(cvalues, Year >=
    1862), errors = TRUE, ...)
{
    if (!errors)
        plot(form, data = data, ...)
    else {
        ylim <- with(data, range(c(speed - error, speed + error),
            na.rm = TRUE))
        plot(form, data = data, ylim = ylim, ...)
        with(data, segments(Year, speed - error, Year, speed +
            error))
        with(data, segments(Year - 1.25, speed - error, Year +
            1.25, speed - error))
        with(data, segments(Year - 1.25, speed + error, Year +
            1.25, speed + error))
    }
    obj <- lm(form, data = data)
    abline(obj)
}
```

```
fig2.2 <-
function (seed = NULL, N = 10, parset = simpleTheme(pch = 1:N),
    fontsize = list(text = 12, points = 8))
{
    if (!is.null(parset))
        parset$fontsize <- fontsize
    if (!exists("Wages")) {
        if(!require("Ecdat", warn.conflicts=FALSE, quietly=TRUE))
    return("Dataset 'Wages' is not available; cannot show graph")
      Wages <- Ecdat::Wages
    }
    if (is.null(Wages$ID))
        Wages$ID <- rep(1:595, each = 7)
    if (!is.null(seed))
        set.seed(seed)
    chooseN <- sample(1:595, N)
    whichN <- Wages$ID %in% chooseN
    gph <- xyplot(lwage ~ exp, groups = ID, data = Wages, subset = whichN,
        xlab = "Years experience", ylab = "log(Wage)", par.settings = parset,
        type = c("p", "r"))
    gph
}
```

```
fig2.3 <-
function (parset = simpleTheme(pch = 16, alpha = 0.8, cex = 1.25),
    fontsize = list(text = 12, points = 8))
{
    if (!is.null(parset))
        parset$fontsize <- fontsize
    if(!require("lattice"))return("Package 'lattice' is not available; cannot show graph")
    if(!exists('ant111b')){
    if(!require("DAAG"))return("Dataset 'ant111b' is not available; cannot show graph")
      ant111b <- DAAG::ant111b
    }
    Site <- with(ant111b, reorder(site, harvwt, FUN = mean))
    gph <- stripplot(Site ~ harvwt, data = ant111b, par.settings = parset,
        xlab = "Harvest weight of corn")
    gph
}
```

```r
fig2.4 <-
function (parset = simpleTheme(pch = c(0, 1), cex = 1.2), fontsize = list(text = 12,
    points = 8), annotate = TRUE)
{
    if (!is.null(parset))
        parset$fontsize <- fontsize
    gph <- xyplot(Time ~ Distance, groups = roadORtrack, data = worldRecords,
        scales = list(log = 10, tck = -0.4, x = list(at = 10^c((-1):2)),
            y = list(at = 10^(0:3))))
    gph <- update(gph, xlab = "Distance (s, km)", ylab = "Time (t, min)",
        par.settings = parset, auto.key = list(columns = 2))
    gph1 <- xyplot(Time ~ Distance, data = worldRecords, scales = list(log = 10),
        type = "r")
    gph2 <- gph + as.layer(gph1)
    if (annotate) {
        layer3 <- layer(longd <- log10(290.2), longt <- log10(24 *
            60), panel.arrows(-1, -0.02, -1, -0.64, length = 0.1,
            col = "gray45"), panel.text(-1 + 0.125, -0.06, "100m",
            pos = 3, cex = 1.05, col = "gray45"), panel.arrows(longd,
            longt + 0.7, longd, longt + 0.15, length = 0.1, col = "gray45"),
            panel.text(longd + 0.18, longt + 0.65, "290km", pos = 3,
                cex = 1.05, col = "gray45"), panel.arrows(-1 -
                0.5, -0.79, -1 - 0.12, -0.79, length = 0.1, col = "gray45"),
            panel.text(-1 - 0.47, -0.79, "9.6sec", pos = 2, cex = 1.05,
                col = "gray45"), panel.arrows(longd - 0.5, longt,
                longd - 0.12, longt, length = 0.1, col = "gray45"),
            panel.text(longd - 0.48, longt, "24h", pos = 2, cex = 1.05,
                col = "gray45"))
        gph2 <- gph2 + layer3
    }
    gph2
}
```

```r
fig2.5 <-
function (parset = simpleTheme(lty = c(2, 1, 2), col.line = c("gray30",
    "black", "gray30"), pch = c(0, 1)), printit=TRUE)
{
    wr.lm <- lm(log(Time) ~ log(Distance), data = worldRecords)
    resid1 <- resid(wr.lm)
    msg <- "As 'mgcv::gam' is not available, unable to proceed."
    if(!require("mgcv", quietly=TRUE, warn.conflicts=FALSE))return(msg)
    wr.gam <- gam(resid1 ~ s(log(Distance)), data = worldRecords)
    hat.gam <- predict(wr.gam, se.fit = TRUE)
    wrgamdata <- with(worldRecords, data.frame(distance = Distance,
        roadORtrack = roadORtrack, resid1 = resid1, resid2 = resid(wr.gam),
        hat = hat.gam$fit, se = hat.gam$se.fit))
    ord <- with(wrgamdata, order(distance))
    wrgamdata <- wrgamdata[ord, ]
    msg <- "As 'lattice' is not available, cannot do graph."
    if(!require("lattice", quietly=TRUE))return(msg)
    gph0 <- lattice::xyplot(resid1 ~ distance, groups = roadORtrack,
                   ylim = c(-0.15, 0.175), xlab = "",
                   scales = list(x = list(log = 10, alternating = 0),
                   tck = -0.4), data = wrgamdata, type = "p",
                   par.settings = parset,
                   auto.key = list(columns = 2))
    gph01 <- lattice::xyplot(I(hat - 2 * se) + hat + I(hat + 2 * se) ~
        distance, outer = FALSE, ylim = c(-0.125, 0.175),
                    scales = list(tck = -0.4,
        x = list(log = 10, alternating = 2)), data = wrgamdata,
        type = "l", par.settings = parset)
    gph1 <- update(gph0 + as.layer(gph01),
                   ylab = expression(atop(Smooth %+-%
        2 * SE, "(resid1)")))
    gph2 <- lattice::xyplot(resid2 ~ distance, groups = roadORtrack,
                   scales = list(tck = -0.4,
        x = list(log = 10)), ylim = c(-0.125, 0.175),
                   ylab = expression(atop("Resids from smooth",
        "(resid2)")), data = wrgamdata, type = c("p"), par.settings = parset)
    if(printit){
      print(gph1, position=c(0, 0.425, 1, 1))
      print(gph2, position=c(0, 0, 1, 0.575) , newpage = FALSE)
    }
    invisible(list(upper = gph1, lower = gph2))
}
```

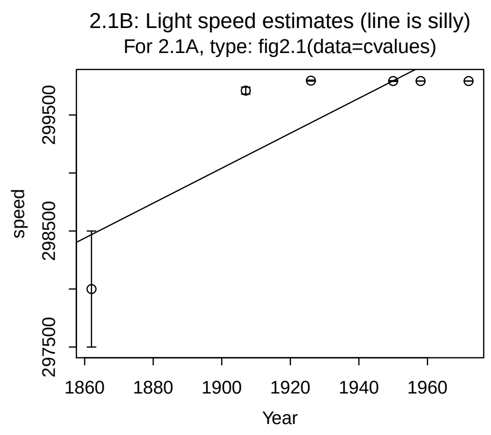
```
fig2.6 <-
function (data = loti)
{
    anom <- data[, "J.D"]
    num <- seq(along = anom)
    AVtodate <- cumsum(anom)/num
    yr <- data$Year
    plot(anom ~ yr, xlab = "", ylab = expression("Difference from 1951-1980 (" *
        degree * "C)"))
    lines(AVtodate ~ yr, col = "gray", lwd = 2)
    lastLessYr <- max(yr[anom < AVtodate])
    lastLessy <- data[as.character(lastLessYr), "J.D"]
    yarrow <- lastLessy - c(4, 0.75) * strheight("O")
    arrows(lastLessYr, yarrow[1], lastLessYr, yarrow[2], col = "gray",
        lwd = 2)
}
```

```
fig2.7 <-
function (statistics = c("airbagAvail", "airbagDeploy", "Restraint"),
    restrict = "!is.na(age)&age>=16&age<998")
{
    msg <- "As 'lattice' is not available; cannot show graph"
    if(!require("lattice"))return(msg)
    gph <- gamclass::plotFars(restrict = restrict)
    plotchars <- c(1:length(statistics))
    plotchars[1] <- 16
    gph <- update(gph, xlab = "", ylab = "Death rate ratio of ratios, w/wo",
        scales = list(tck = 0.5), par.settings = simpleTheme(pch = plotchars))
    gph
}
```

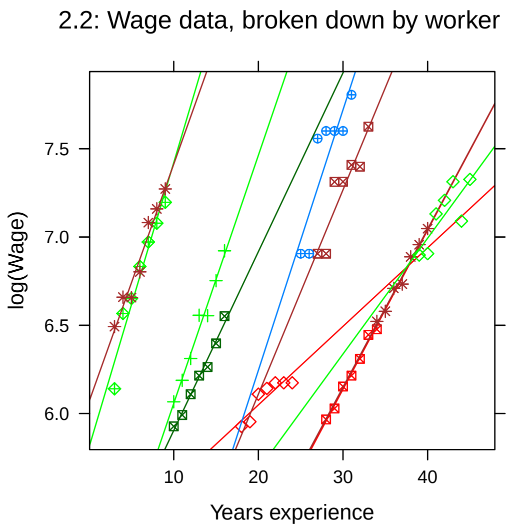
2 Show the Figures

```
pkgs <- c("gamclass","latticeExtra","DAAG", "mgcv")
z <- sapply(pkgs, require, character.only=TRUE, warn.conflicts=FALSE, quietly=TRUE)

This is mgcv 1.8-7.  For overview type 'help("mgcv-package")'.

if(any(!z)){
  notAvail <- paste(names(z)[!z], collapse=", ")
  print(paste("The following packages should be installed:", notAvail))
}
```

```
fig2.1()
title(main="2.1B: Light speed estimates (line is silly)",
      line=1.75, cex.main=1.1)
mtext(side=3, line=0.5, "For 2.1A, type: fig2.1(data=cvalues)")
```

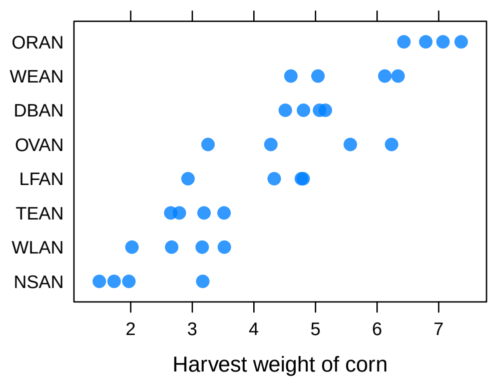


```
gph <- fig2.2()
update(gph, main = list("2.2: Wage data, broken down by worker",
                        fontface="plain", lineheight=0.25,
                        just=c("left","top"),
                        x = grid::unit(12, "mm")))
```



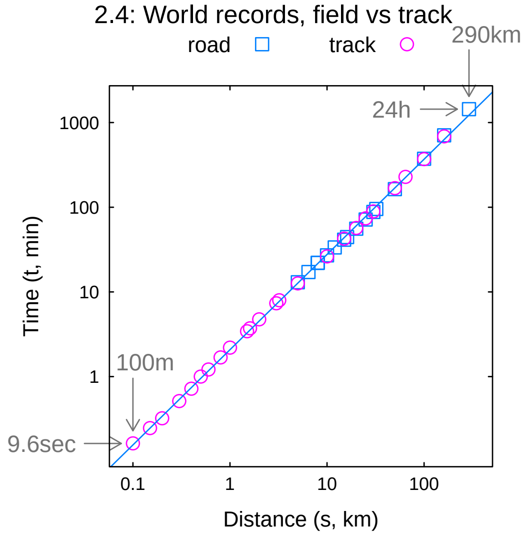
```
gph <- fig2.3()
update(gph, main=list("2.3: Corn harvest weight by site",
                      lineheight=0.75, fontface="plain"))
```

2.3: Corn harvest weight by site

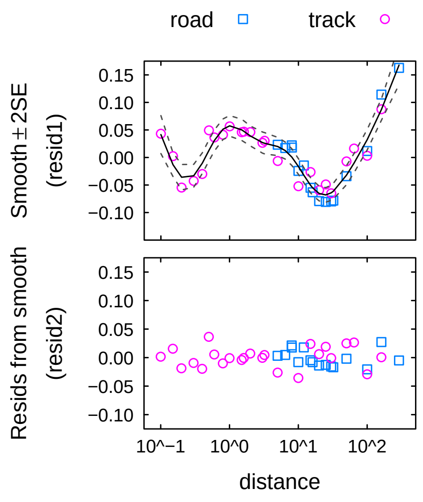


```
gph <- fig2.4()
trellis.par.set(clip=list(panel="off",strip="on"))
print(update(gph, main=list("2.4: World records, field vs track", fontface="plain")),
      position = c(0.05, 0, 1, 0.95))
trellis.par.set(clip=list(panel="on",strip="on"))
```

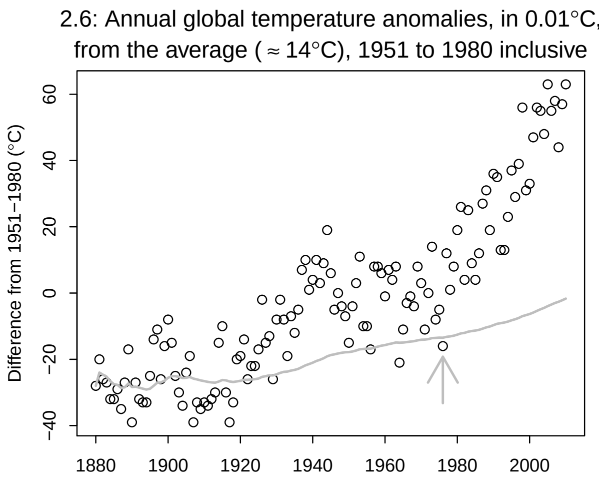
2.4: World records, field vs track



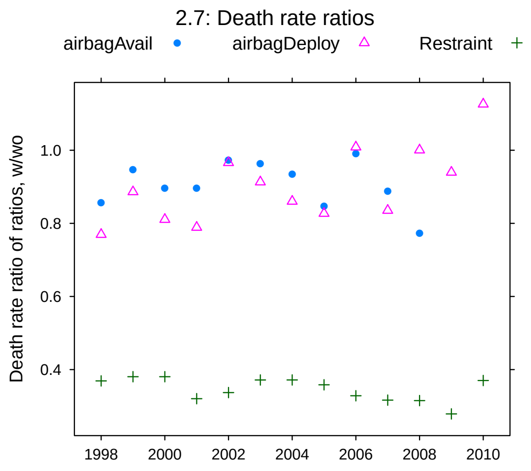
```
gphs <- fig2.5(printit=FALSE)
print(gphs[["upper"]], position=c(0, 0.415, 1,1))
print(gphs[["lower"]], position=c(0, 0, 1,0.585), newpage=FALSE)
```



```
fig2.6()
title1 <- expression("2.6: Annual global temperature anomalies, in 0.01" *
             degree * "C,")
title(main = title1, line = 2.1, cex=1.2)
title2 <- expression("from the average (" %~~% 14 * degree *
             "C), 1951 to 1980 inclusive")
title(main = title2, line = 0.8, cex=1.2)
```



```
gph <- fig2.7()
update(gph, main=list("2.7: Death rate ratios", fontface="plain"))
```


2.7: Death rate ratios
airbagAvail
airbagDeploy
Restraint
Death rate ratio of ratios, w/wo
1.0
0.8
0.6
0.4
1998
2000
2002
2004
2006
2008
2010