Lecture 14. Advanced Plot Customization

R and Data Visualization

BIG2006, Hanyang University, Fall 2022

Handling the Graphics Device

Many users are first drawn to R because of its impressive graphical flexibility and the ease with which you can control and tailor the resulting visuals.

In this lecture, we will take a closer look at the base R graphic device, and at how you can tune the plots you are already familiar with, to get the most use out of your visualization.

Manually Opening a New Device

- It is possible to have multiple graphics device open, but only one will be deemed active at any given time.
- You can open new device windows via the `dev.new` function.
- The newest window will immediately become active, and any subsequent plotting commands will affect that particular device.

```
# generate a plot of the spatial locations in the quakes data frame
plot(quakes$long,quakes$lat)
# open a new plotting window
dev.new()
# generate a histogram on the new window (active device)
hist(quakes$stations)
```

Note: If you hadn't used `dev.new`, the histogram would've just overwritten the plot from `quakes`.

Switching Between Devices and Closing a Device

- To change something in the old device without closing new device, use `dev.set` followed by the device number you want to make active.
- To close a graphics device, either click the X with your mouse as you would to close any window or use the `dev.off` function.

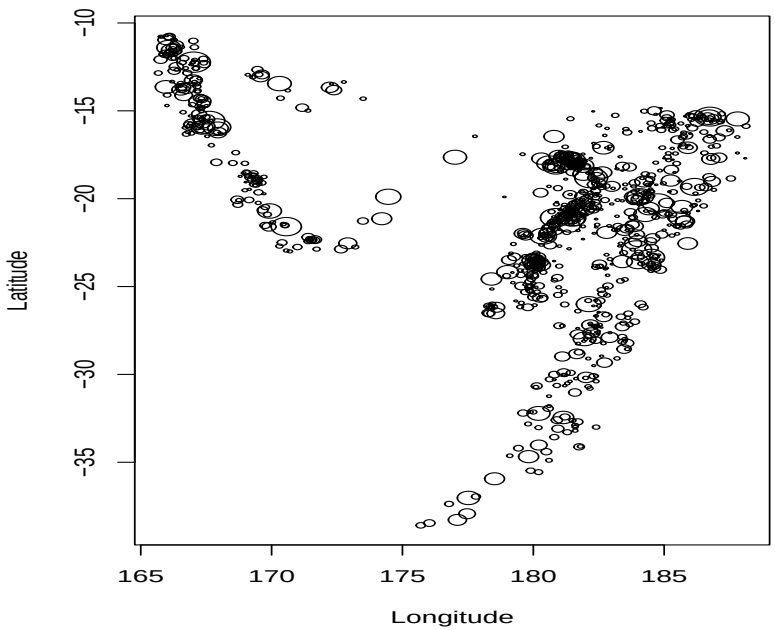
Note: Calling `dev.off()` with no arguments simply close the currently active device. Otherwise, you can specify the device number just as when using `dev.set`.

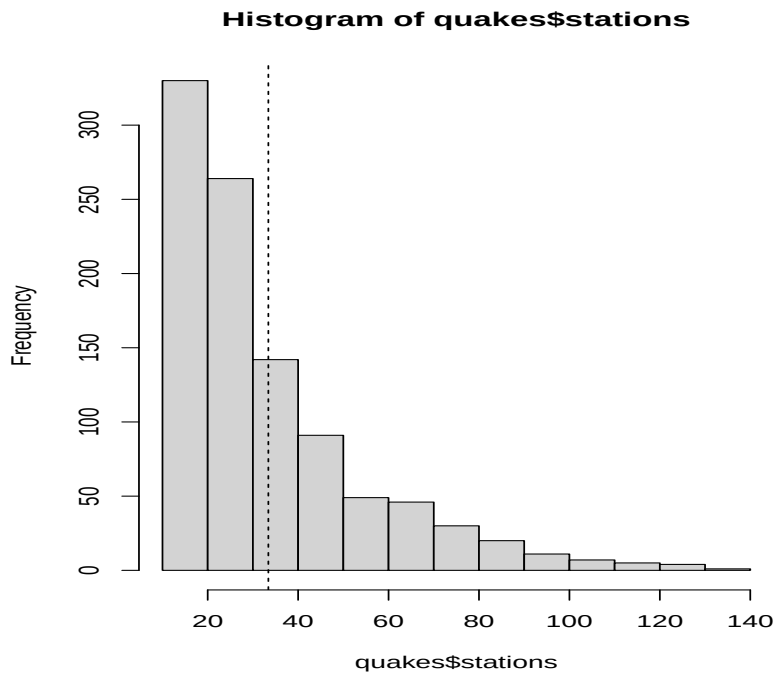
Multiple Plots in One Device

Setting the mfrow Parameter

- `mfrow`: instructs a new device to divide itself into a grid of the specified dimensions, with each cell holding one plot
- You pass the `mfrow` option a numeric integer vector of length 2, in the order of `c(rows,column)`

```r
#dev.new(width=8,height=4)
par(mfrow=c(1,2))
plot(quakes$long,quakes$lat,cex=0.02*quakes$stations,
     xlab="Longitude",ylab="Latitude")
hist(quakes$stations)
abline(v=mean(quakes$stations),lty=2)
```





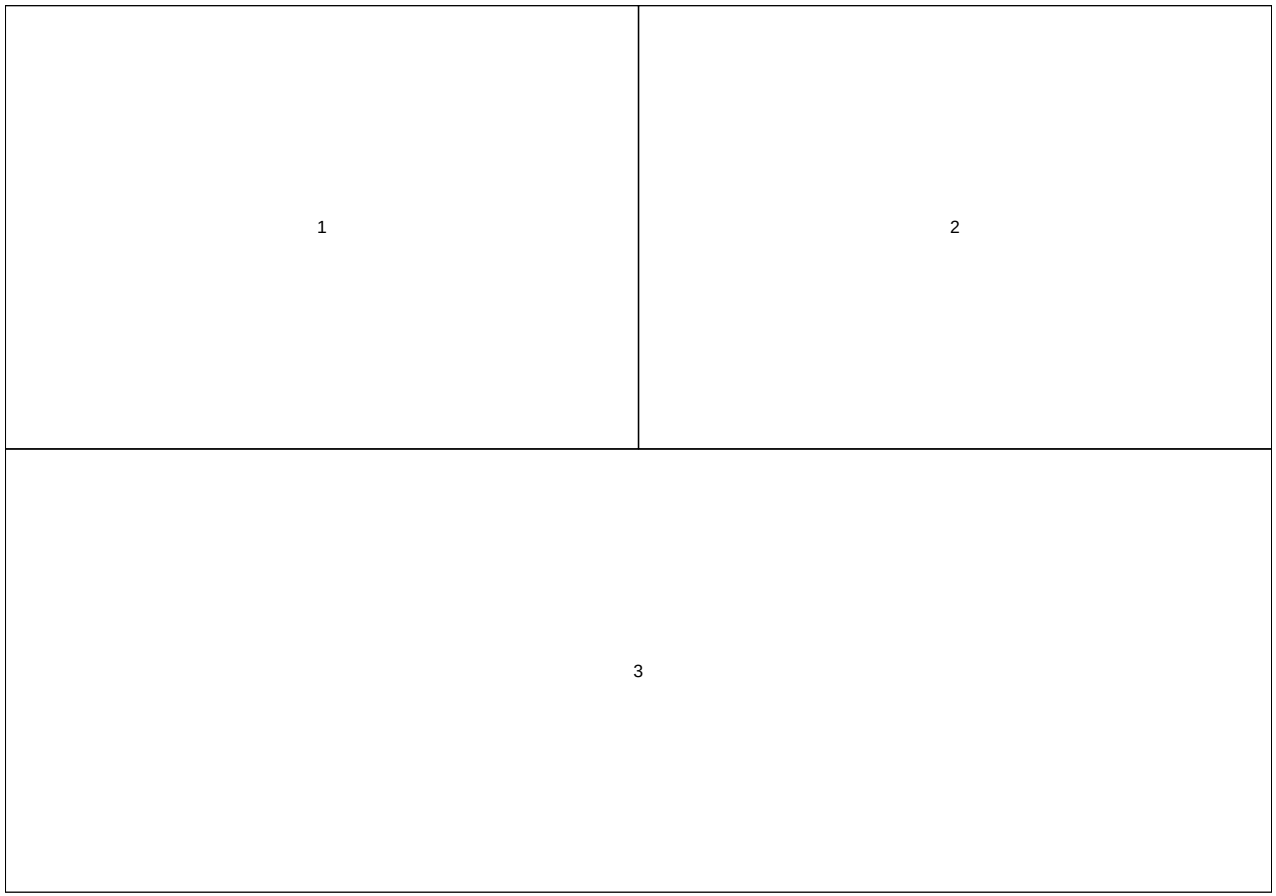
Defining a Particular Layout

- `layout`: refines the arrangement of plots in a single device
- When you use `layout`, you provide the dimensions in a matrix `mat` as the first argument; these govern an invisible rectangular grid, just like controling the `mfrow` option.

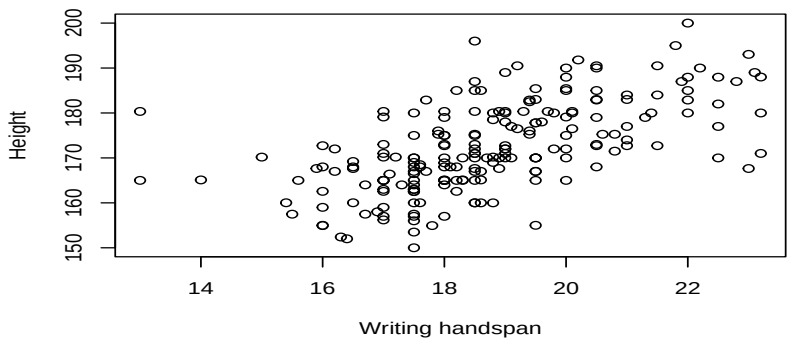
```
lay.mat <- matrix(c(1,3,2,3),2,2)
lay.mat
```

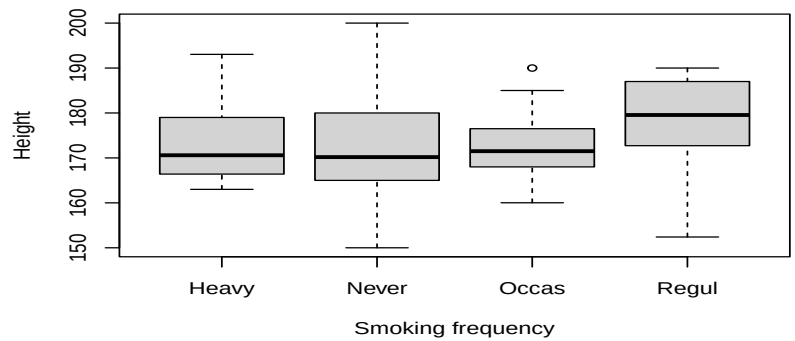
```
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    3
```

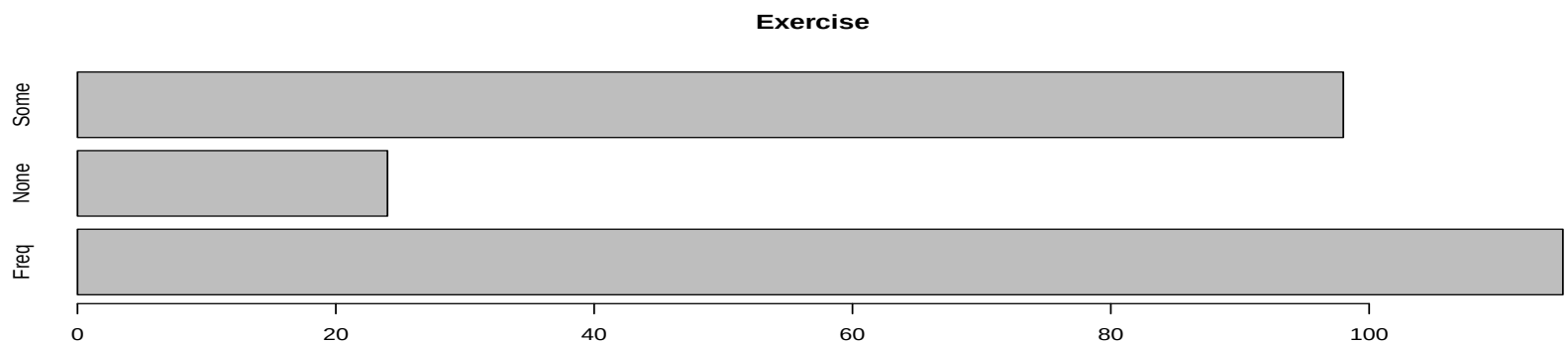
```
layout(mat=lay.mat)
layout.show(n=max(lay.mat))
```



```
layout(mat=lay.mat)
plot(survey$Wr.Hnd,survey$Height,
     xlab="Writing handspan",ylab="Height")
boxplot(survey$Height~survey$Smoke,
        xlab="Smoking frequency",ylab="Height")
barplot(table(survey$Exer),horiz=TRUE,main="Exercise")
```







Plotting Regions and Margins

There are three regions that make up the image.

- *plot region*: where your actual plot appears and where you will usually be drawing your points, lines, text, and so on. The plot region uses the *user coordinate system*.
- *figure region*: the area that contains the space for your axes, their labels, and any titles. These spaces are referred to as the *figure margins*.
- *outer region (margins)*: additional space around the figure region that is not included by default but can be specified if it is needed.

Default and Custom Spacing

- `oma=c(0,0,0,0)`: no outer margin set by default
- `mar=c(5.1,4.1,4.1,2.1)`: the default figure margin space is 5.1 lines of text on the bottom, 4.1 on the left and top, and 2.1 on the right.

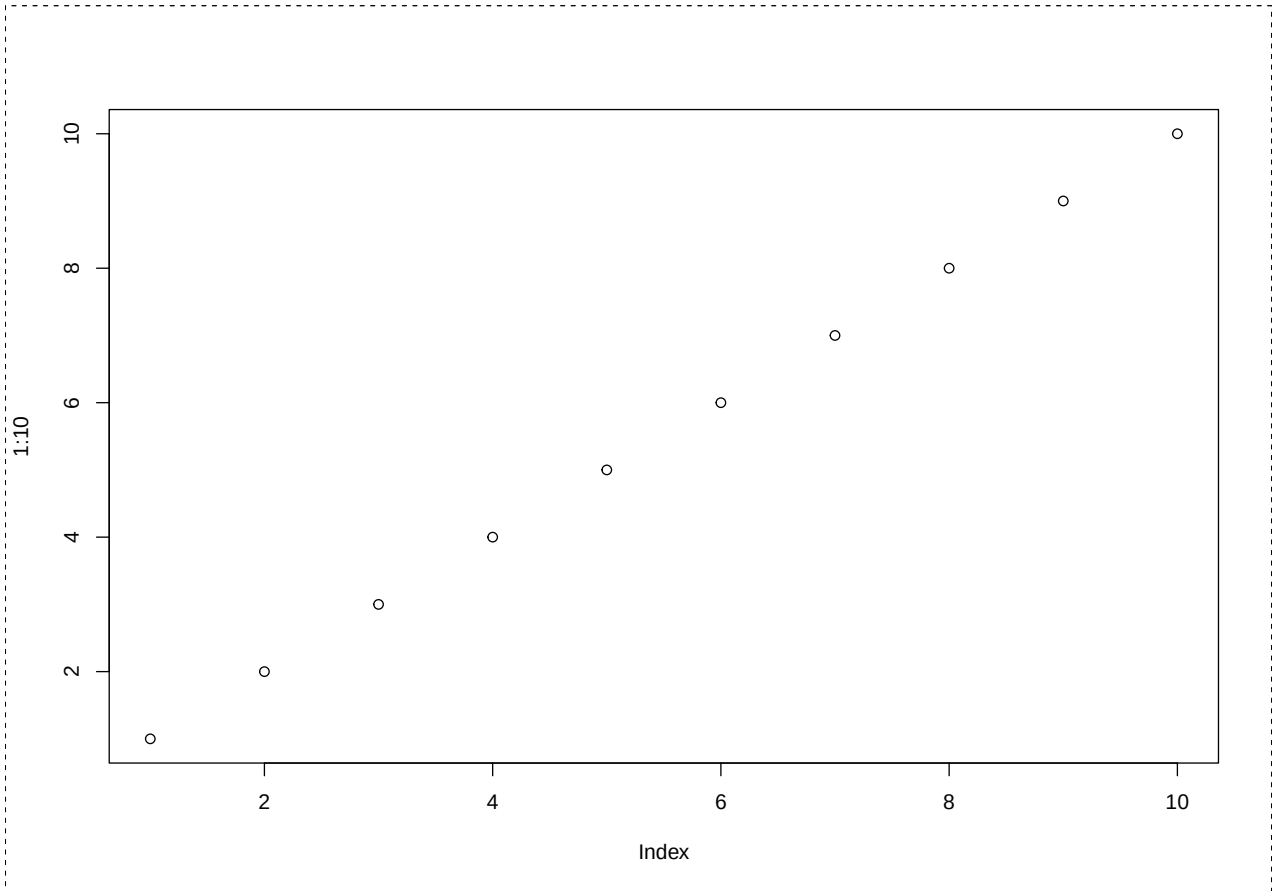
```
par()$oma
```

```
## [1] 0 0 0 0
```

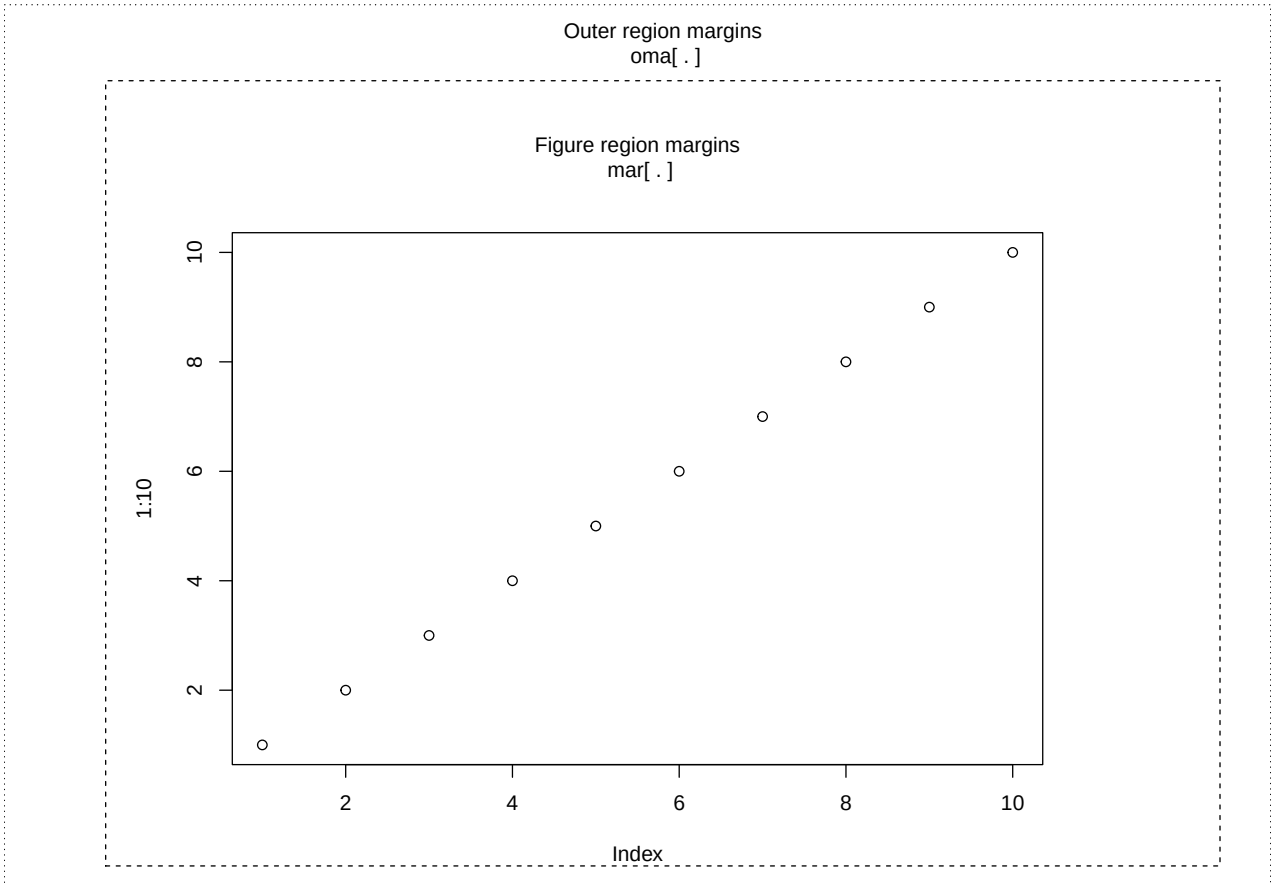
```
par()$mar
```

```
## [1] 5.1 4.1 4.1 2.1
```

```
plot(1:10)
box(which="figure",lty=2) # shows you the figure region
```



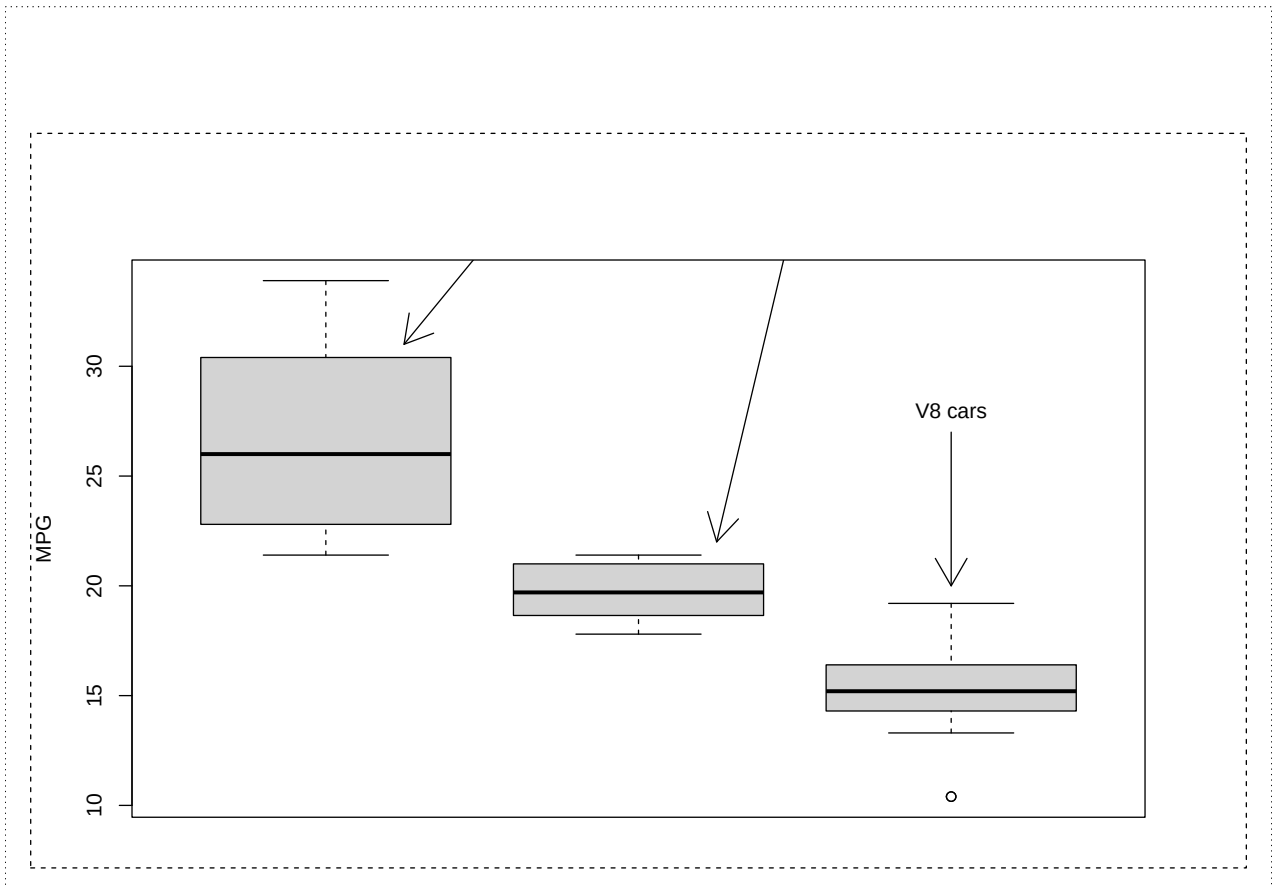
```
par(oma=c(1,4,3,2),mar=4:7)
plot(1:10)
box("figure",lty=2)
box("outer",lty=3)
mtext("Figure region margins\n mar[ . ]",line=2)
mtext("Outer region margins\n oma[ . ]",line=0.5,outer=TRUE)
```



Clipping

- Controlling *clipping* allows you to draw in or add elements to the margin regions with reference to the user coordinates of the plot itself.
- The graphical parameter `xpd` controls clipping in base R graphics.
- `xpd=FALSE`: all drawing is clipped to the available plot region only
- `xpd=TRUE`: allows you draw things outside the formally defined plot region into the figure margins
- `xpd=NA`: permit drawing in all three areas (plot region, figure margins, and the outer margins)

```
par(oma=c(1,1,5,1),mar=c(2,4,5,4))
boxplot(mtcars$mpg~mtcars$cyl,xaxt="n",ylab="MPG")
box("figure",lty=2); box("outer",lty=3)
arrows(x0=c(2,2.5,3),y0=c(44,37,27),x1=c(1.25,2.25,3),y1=c(31,22,20),xpd=FALSE)
text(x=c(2,2.5,3),y=c(45,38,28),c("V4 cars","V6 cars","V8 cars"),xpd=FALSE)
```

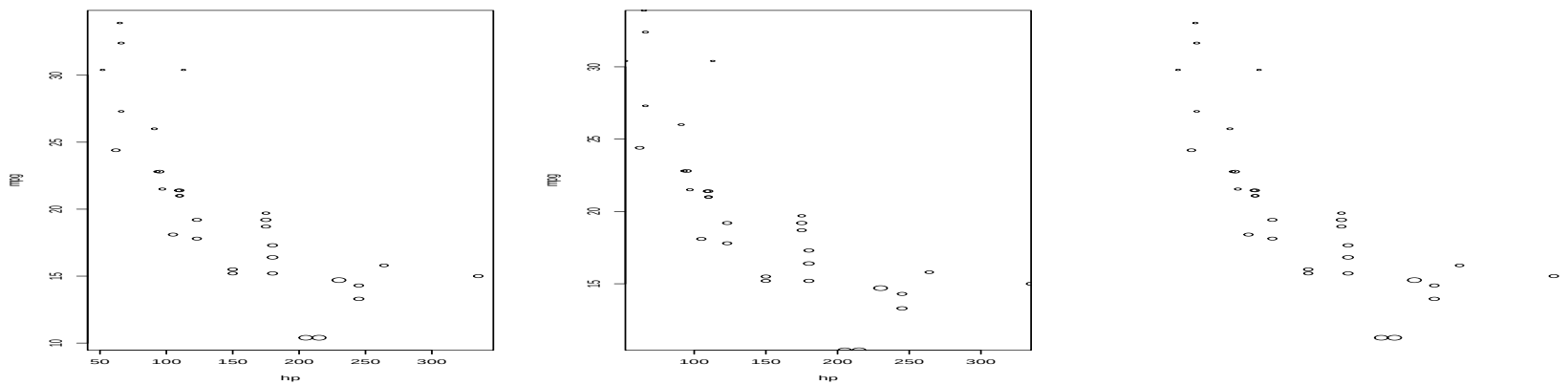


Customizaing Traditional R Plots

Graphical Parameters for Style and Suppression

- `xaxs, yaxs`: controls axis style, e.g., `xaxs="r"` includes the space, `xaxs=i` instructs the plot region to be strictly defined by the upper and lower limits of the data (no additional padding space)
- `xaxt="n", yaxt="n", bty="n"`: set the default axis labels to the empty string `""` and empty box
- `axes=FALSE`: suppresses all axes and the box
- `ann=FALSE`: suppresses any annotation

```r
hp <- mtcars$hp; mpg <- mtcars$mpg; wtcex <- mtcars$wt/mean(mtcars$wt)
par(mfrow=c(1,3))
plot(hp,mpg,cex=wtcex) # left
plot(hp,mpg,cex=wtcex,xaxs="i",yaxs="i") # middle
plot(hp,mpg,cex=wtcex,axes=FALSE,ann=FALSE) # right
```




```r
#plot(hp,mpg,cex=wtcex,axes=FALSE,ann=FALSE)
```

Customizing Boxes

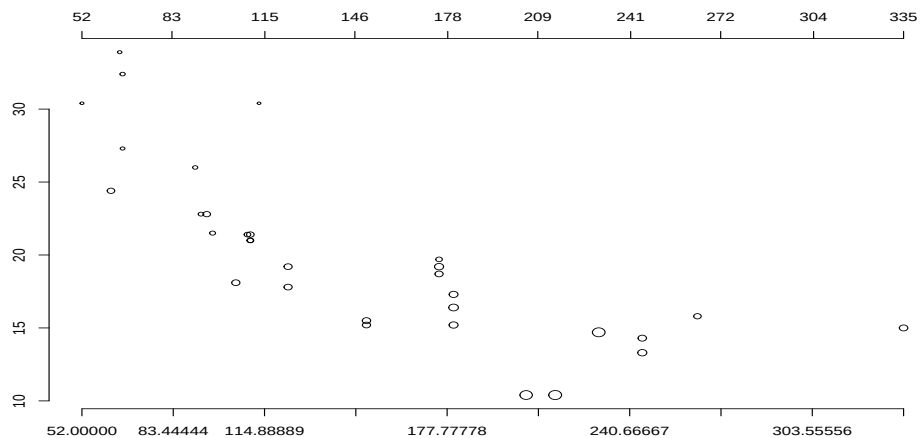
- The `bty` argument is supplied a single character: `"o"` (default), `"l"`, `"7"`, `"c"`, `"u"`, `"]"`, or `"n"`.

```
par(mfrow=c(1,2))
plot(hp,mpg,cex=wtcex,axes=FALSE,ann=FALSE)
box(bty="l",lty=3,lwd=2)
plot(hp,mpg,cex=wtcex,axes=FALSE,ann=FALSE)
box(bty="]",lty=2,col="gray")
```

Customizing Axes

- `axis`: allows you to control the addition and appearance of an axis on any of the four sides of the plot region. `side=1` (bottom), 2 (left), 3 (top), or 4 (right)

```
hpseq <- seq(min(hp),max(hp),length=10)
plot(hp,mpg,cex=wtcex,xaxt="n",bty="n",ann=FALSE)
axis(side=1,at=hpseq); axis(side=3,at=round(hpseq))
```



Specialized Text and Label Notation

Now we will investigate some immediately accessible tools for controlling fonts and displaying special notation, such as Greek symbols and mathematical expression.

Font

- The displayed font is controlled by two graphical parameters: `family` for the specific font family and `font`.

```
par(mar=c(3,3,3,3))
plot(1,1,type="n",xlim=c(-1,1),ylim=c(0,7),xaxt="n",yaxt="n",ann=FALSE)
text(0,6,label="sans text (default)\n family=\"sans\", font=1")
text(0,5,label="serif text\n family=\"serif\", font=1",family="serif",font=1)
text(0,4,label="mono text\n family=\"mono\", font=1",family="mono",font=1)
text(0,3,label="mono text (bold, italic)\n family=\"mono\", font=4",
     family="mono",font=4)
text(0,2,label="sans text (italic)\n family=\"sans\", font=3",
     family="sans",font=3)
text(0,1,label="serif text (bold)\n family=\"serif\", font=2",
     family="serif",font=2)
mtext("some",line=1,at=-0.5,cex=2,family="sans")
mtext("different",line=1,at=0,cex=2,family="serif")
mtext("fonts",line=1,at=0.5,cex=2,family="mono")
```

some different fonts

sans text (default)
family="sans", font=1

serif text
family="serif", font=1

mono text
family="mono", font=1

mono text (bold, italic)
family="mono", font=4

sans text (italic)
family="sans", font=3

serif text (bold)
family="serif", font=2

Greek Symbols

- You can display Greek symbols or mathematical markup.

```
par(mar=c(3,3,3,3))
plot(1,1,type="n",xlim=c(-1,1),ylim=c(0.5,4.5),xaxt="n",yaxt="n",ann=FALSE)
text(0,4,label=expression(alpha),cex=1.5)
text(0,3,label=expression(paste("sigma: ",sigma," Sigma: ",Sigma)),
     family="mono",cex=1.5)
text(0,2,label=expression(paste(beta," ",gamma," ",Phi)),cex=1.5)
text(0,1,label=expression(paste(Gamma,"(",tau,") = 24 when ",tau," = 5")),
     family="serif",cex=1.5)
title(main=expression(paste("Gr",epsilon,epsilon,"k")),cex.main=2)
```

Greek

α

`sigma:` σ `Sigma:` Σ

$\beta\ \gamma\ \Phi$

$\Gamma(\tau) = 24$ when $\tau = 5$

Mathematical Expressions

- Formatting entire mathematical expressions to appear in R plots is a bit more complicated and is reminiscent of using markup language like LaTeX

```
expr1 <- expression(c^2==a[1]^2+b[1]^2)
expr2 <- expression(paste(pi^{x[i]},(1-pi)^(n-x[i])))
expr3 <- expression(paste("Sample mean: ",italic(n)^{-1},
                    sum(italic(x)[italic(i)],italic(i)==1,
                    italic(n))==frac(italic(x)[1]+...+italic(x)[italic(n)],
                    italic(n))))
expr4 <- expression(paste("f(x","|",alpha,",",beta,")"==
                    frac(x^{alpha-1}~(1-x)^{beta-1},B(alpha,beta))))
par(mar=c(3,3,3,3))
plot(1,1,type="n",xlim=c(-1,1),ylim=c(0.5,4.5),xaxt="n",yaxt="n",ann=FALSE)
text(0,4:1,labels=c(expr1,expr2,expr3,expr4),cex=1.5)
title(main="Math",cex.main=2)
```

Math

$$c^2 = a_1^2 + b_1^2$$

$$\pi^{x_i}(1-\pi)^{(n-x_i)}$$

Sample mean: $n^{-1}\sum_{i=1}^{n} x_i = \frac{x_1 + \cdots + x_n}{n}$

$$f(x|\alpha,\beta) = \frac{x^{\alpha-1}\,(1-x)^{\beta-1}}{B(\alpha,\,\beta)}$$

A Fully Annotated Scatterplot

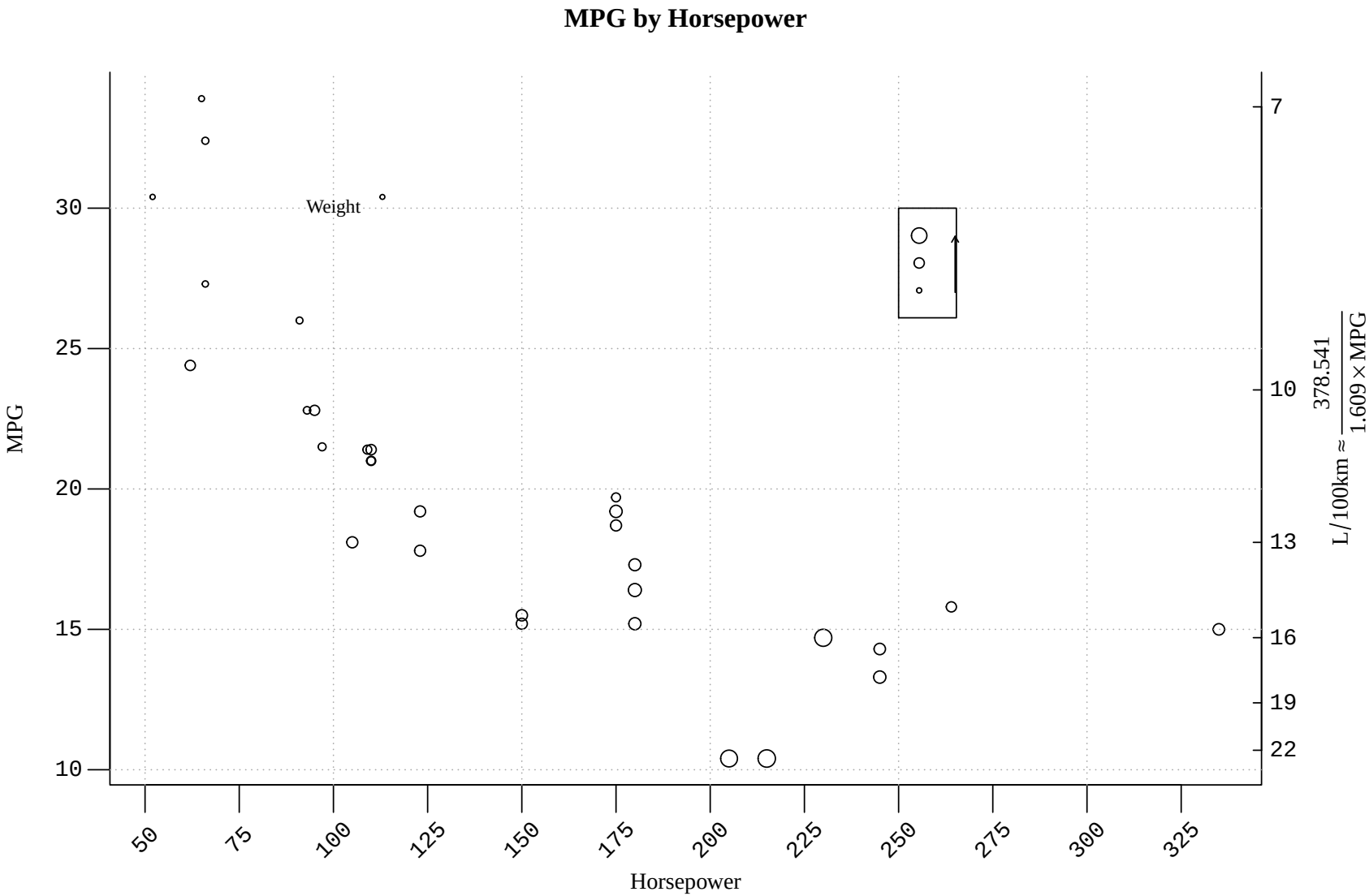
```
hp <- mtcars$hp; mpg <- mtcars$mpg; wtcex <- mtcars$wt/mean(mtcars$wt)
hpseq2 <- seq(50,325,by=25)

par(mar=c(5,4,4,4))
plot(hp,mpg,cex=wtcex,axes=FALSE,ann=FALSE); box(bty="u")
axis(2,las=1,tcl=-0.8,family="mono")
axis(1,at=hpseq2,labels=FALSE,tcl=-1)

L100 <- seq(22,7,by=-3); MPG.L100 <- (100/L100*3.78541)/1.609
axis(4,at=MPG.L100,labels=L100,las=1,tcl=0.3,mgp=c(3,0.3,0),family="mono")
express.L100 <- expression(paste(L/100,"km"%~~%frac(378.541,1.609%*%MPG)))

title(main="MPG by Horsepower",xlab="Horsepower",ylab="MPG",family="serif")
mtext(express.L100,side=4,line=3,family="serif")
text(hpseq2,rep(7.5,length(hpseq2)),labels=hpseq2,srt=45,xpd=TRUE,
     family="mono")

grid(col="darkgray")
legend(250,30,legend=rep(" ",3),pch=rep(1,3),pt.cex=c(1.5,1,0.5))
arrows(265,27,265,29,length=0.05)
text(locator(1),labels="Weight",cex=0.8,family="serif")
# locator(1) point-and-click coordinate interaction
```


MPG by Horsepower
Weight
MPG
Horsepower
10
15
20
25
30
50
75
100
125
150
175
200
225
250
275
300
325
7
10
13
16
19
22
L/100km ≈ 378.541 / (1.609 × MPG)

Reference

- Davies, T. M. The Book of R. No Starch Press. Chapter 23.