Seminar III: R/Bioconductor

Leonardo Collado Torres
lcollado@lcg.unam.mx
Bachelor in Genomic Sciences
www.lcg.unam.mx/~lcollado/

August - December, 2009

Advanced Plotting

intro

lattice

plotrix

ggplot2

car

R is strong in plots

- As you might recall, R is very strong for making plots, and it does so fast.
- We've seen how to make barplots, qqplots, mosaicplots, and many other ones.
- After all, plotting is very important for doing exploratory data analysis.
- However, all of them just make a small part.

Install some packages

To gain some time, please install these packages:

```
> install.packages("lattice")
> install.packages("mlmRev")
> install.packages("plotrix")
> install.packages("ggplot2")
> install.packages("car")
> install.packages("DAAG")
```

Task Views

- First of all, remember the CRAN Task Views.
- `http://cran.r-project.org/web/views/Graphics.html`
- From there, go to the `plotrix` page.
- What two functions did they introduce on version 2.5-3?

tools

- You might decide to check the reference manual and test out the examples, but that's quite time consuming.
- I found out on the R Journal about a new function on the `tools` package.

  ```
  > library(tools)
  > testInstalledPackage(pkgname)
  ```
- Its very easy to create pdf files with all the example plots of a given package!

Remember to check the help

- Remember to use:

```
> help.start()
> help(package = pkgname)
```

- What is the replacement of the hist function on the lattice package?

Intro

- It's an *implementation* of Trellis graphics and created by Deepayan Sarkar.
- `http://dsarkar.fhcrc.org/`
- Basically, its great for plotting multivariate data!

  ```
  > `?`(Lattice)
  ```
- How are the lattice high level functions special?

Data

- We'll use a data set from the mlmRev package and in general we'll follow the BioC2008 lattice lab.

  ```
  > library(lattice)
  > data(Chem97, package = "mlmRev")
  ```

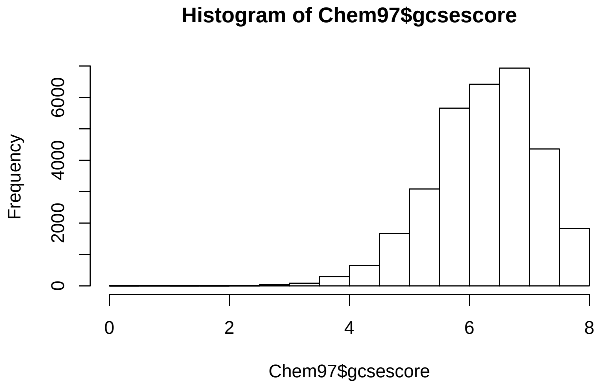
- What is the class of Chem97?
- How many variables does it have? *You might want to use length*

Formula syntax

- We'll mostly use three variables: score, gcsescore and gender.
- Now, lattice uses the formula syntax.
- Basically its $y\ x|g1$ where x is the variable with the numeric values and $g1$ is a factor.

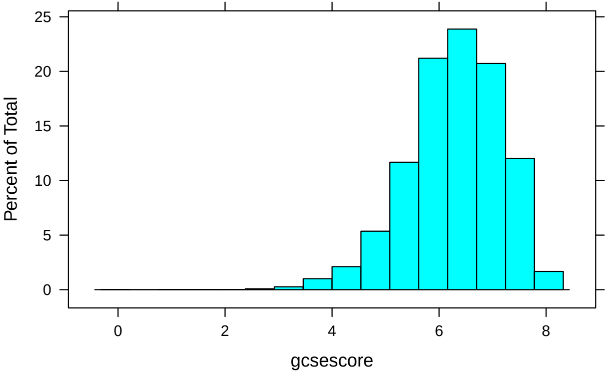
Comparing histograms

```
> hist(Chem97$gcsescore)
```



Comparing histograms: part II

```
> print(histogram(~gcsescore, data = Chem97))
```



A grouping var

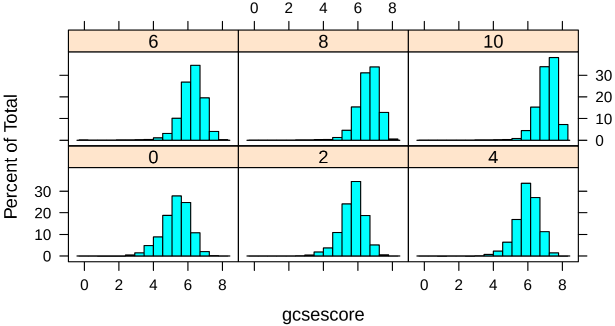
- The variable score only has values 0, 2, 4, 6, 8 and 10.

```
> head(Chem97$score)
[1]  4 10 10 10  8 10
> class(Chem97$score)
[1] "numeric"
```

- We can use this variable as a factor!
- Lets make a more interesting plot :)

Multiple hist

```
> print(histogram(~gcsescore | factor(score),
+       data = Chem97))
```



And gender?

- But we want to use our third variable: gender

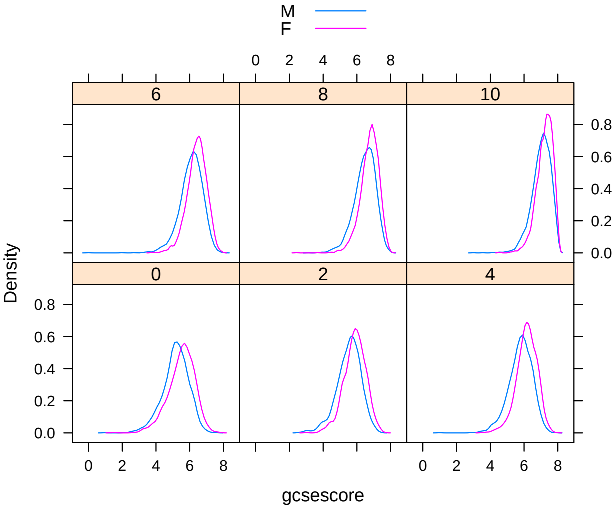
  ```
  > class(Chem97$gender)

  [1] "factor"
  ```

- Its difficult to plot two histograms on the same panel, but that's not the case with density lines!

Densities

```
> print(densityplot(~gcsescore |
+     factor(score), Chem97, groups = gender,
+     plot.points = FALSE, auto.key = TRUE))
```

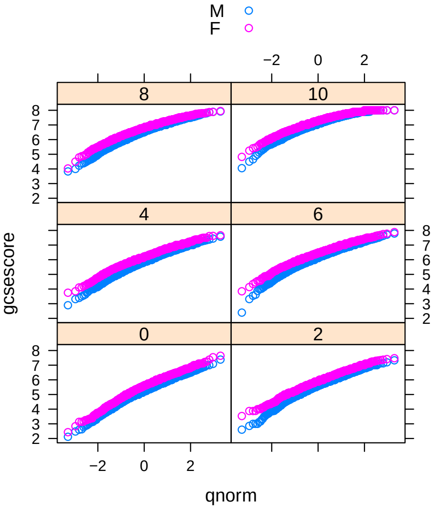
Densities



QQ norm too!

```
> print(qqmath(~gcsescore | factor(score),
+     Chem97, groups = gender, auto.key = TRUE,
+     aspect = "xy", f.value = ppoints(1000)))
```

QQ norm too!



Compare QQ norm

- Re-do the above QQ norm plot with the following arguments:

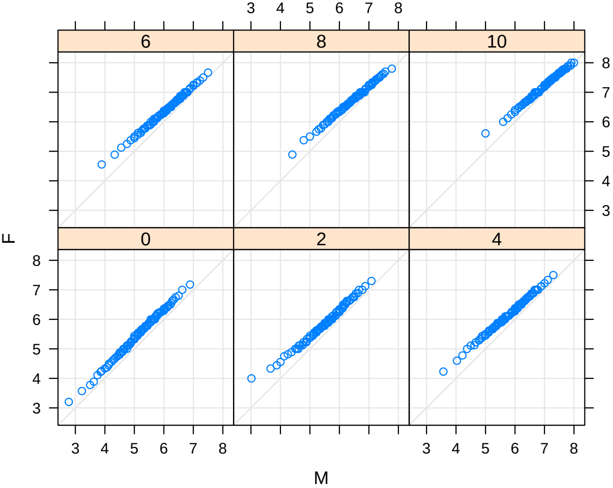
  ```
  > f.value = ppoints(100)
  > type = c("p", "g")
  ```

- Which of the two QQ norm plots is clearer?

QQ plots

```
> print(qq(gender ~ gcsescore | factor(score),
+     Chem97, f.value = ppoints(100),
+     type = c("p", "g"), aspect = 1))
```

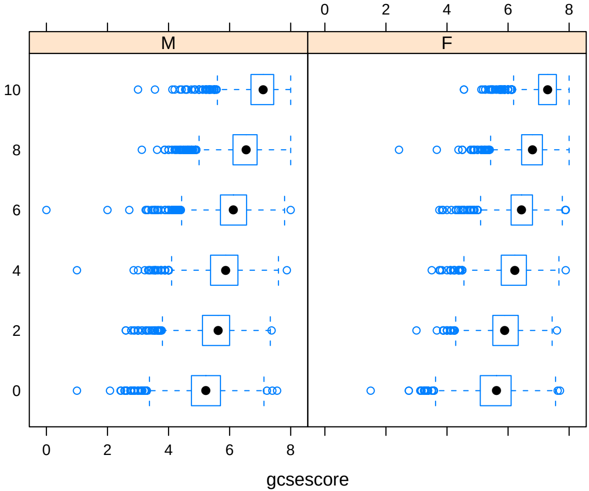
QQ plots



Boxplots

```
> print(bwplot(factor(score) ~ gcsescore |
+     gender, Chem97))
```

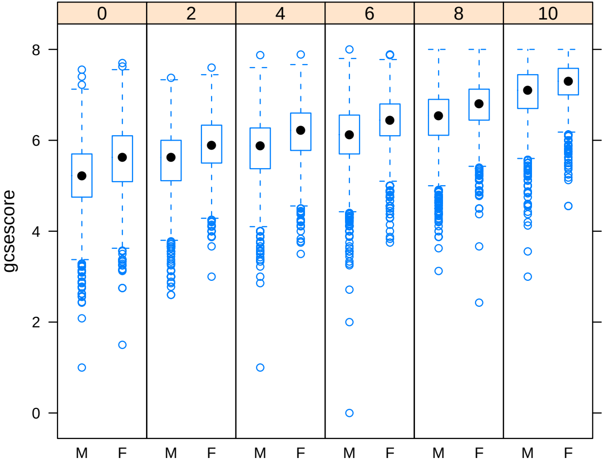
Boxplots



Boxplots II

```
> print(bwplot(gcsescore ~ gender |
+     factor(score), Chem97, layout = c(6,
+     1)))
```

Boxplots II

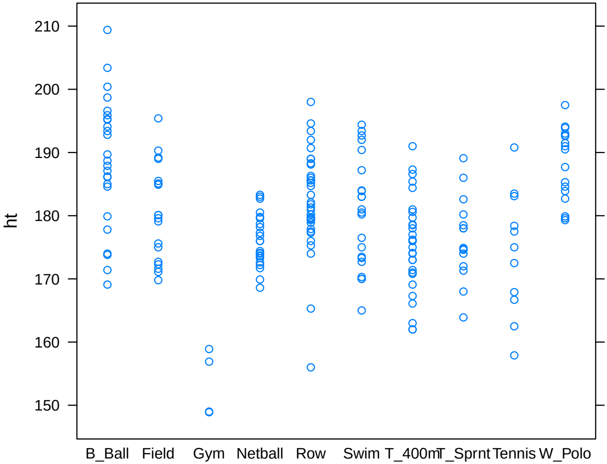


Stripplot

- Useful for small data sets :)

```
> library(DAAG)
> print(stripplot(ht ~ factor(sport),
+     data = ais))
```

Stripplot

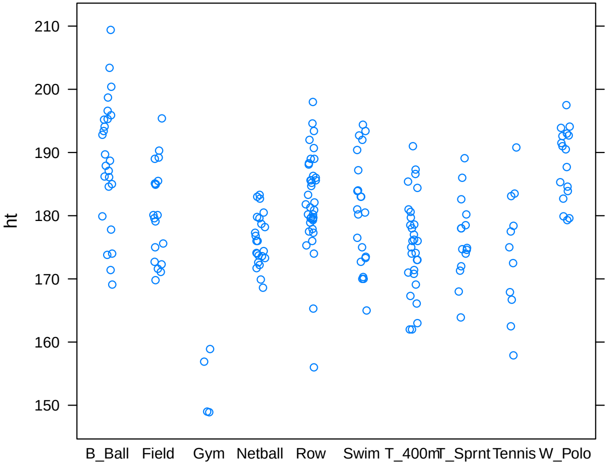

210
200
190
180
170
160
150
ht
B_Ball
Field
Gym
Netball
Row
Swim
T_400m
T_Sprnt
Tennis
W_Polo

Stripplot II

- The jitter argument saves the day!
- Plus points in lattice are partially transparent

```
> print(stripplot(ht ~ factor(sport),
+     data = ais, jitter = T))
```

Stripplot II

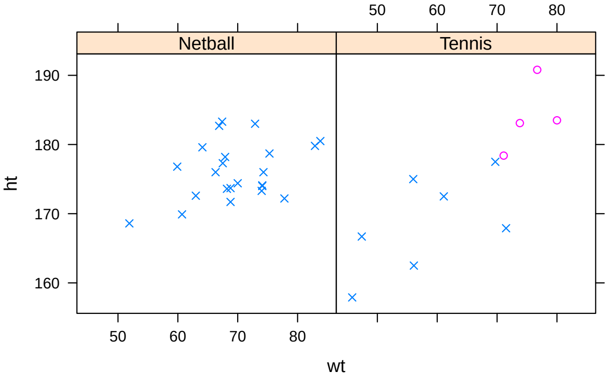


xyplot

- With lattice, we can also make something similar to plot
- But first, lets create a subset of the type of sports.

```
> subset <- ais$sport %in% c("Netball",
+     "Tennis")
> print(xyplot(ht ~ wt | sport, groups = sex,
+     pch = c(4, 1), aspect = 1,
+     subset = subset, data = ais))
```

xyplot

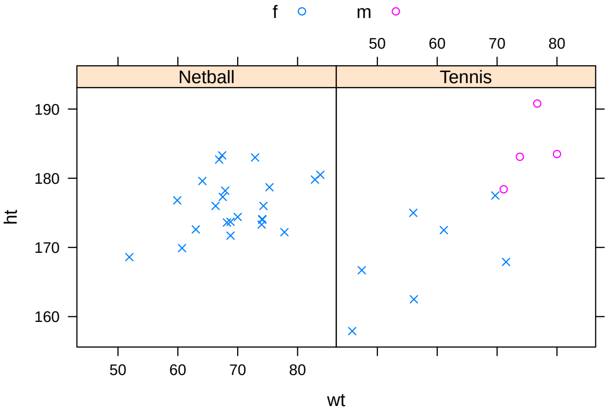


xyplot II

- What will happen if we say `auto.key=TRUE`?
- On this plot, we are visualizing data from how many variables?

```
> print(xyplot(ht ~ wt | sport, groups = sex,
+      pch = c(4, 1), aspect = 1,
+      auto.key = list(columns = 2),
+      subset = subset, data = ais))
```

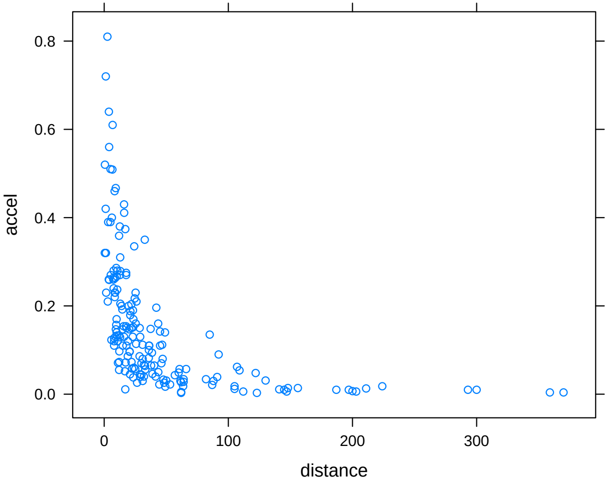
xyplot II



xyplot B

```
> data(Earthquake, package = "nlme")
> print(xyplot(accel ~ distance,
+     data = Earthquake))
```

xyplot B

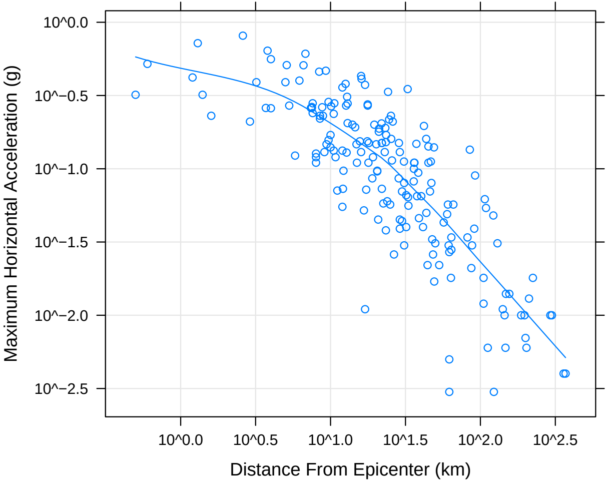


xyplot B II

- What does the `scales` argument control?
- What would happen if you delete `smooth` from the type argument?

```
> print(xyplot(accel ~ distance,
+     data = Earthquake, scales = list(log = TRUE),
+     type = c("p", "g", "smooth"),
+     xlab = "Distance From Epicenter (km)",
+     ylab = "Maximum Horizontal Acceleration (g)"))
```

xyplot B II

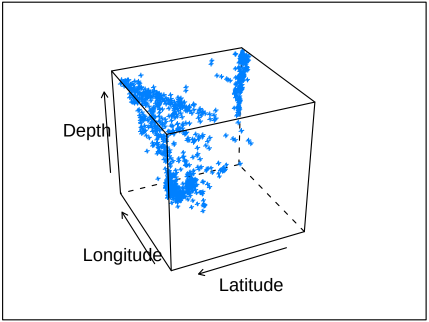

10^0.0
10^−0.5
10^−1.0
10^−1.5
10^−2.0
10^−2.5
Maximum Horizontal Acceleration (g)
10^0.0
10^0.5
10^1.0
10^1.5
10^2.0
10^2.5
Distance From Epicenter (km)

3D!

- With the cloud function its possible to create 3D plots.
- To rotate it, you need to re-make it with different values for the x, y and z.

```
> print(cloud(depth ~ lat * long,
+     data = quakes, zlim = rev(range(quakes$depth)),
+     screen = list(z = 115, x = -60),
+     panel.aspect = 0.75, xlab = "Longitude",
+     ylab = "Latitude", zlab = "Depth"))
```

3D!



That's it for lattice

- Lattice has more plot functions such as barchart and dotplot which we won't cover today, but feel free to check them.
- There is also a book available on `lattice`: `http://lmdvr.r-forge.r-project.org/`
- As I said at the beginning, use the `tools` package to explore `lattice` and `latticeExtra`.

Intro

- It contains loads of enhanced R functions.
- The reference manual has 139 pages!!!
- Functions such as adding a table, standard deviation bars, cutting axes, etc.

Barplot with table

- First, we'll create a data.frame with some data
- Then we'll use the barp function to create a barplot
- Finally, we'll add the table to our plot

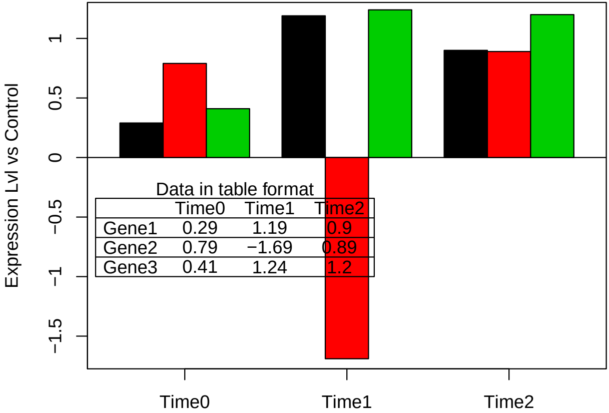
```
> set.seed(123)
> df <- data.frame(Time0 = runif(3),
+     Time1 = rnorm(3), Time2 = rlnorm(3))
> df <- round(df, digits = 2)
> rownames(df) <- c("Gene1", "Gene2",
+     "Gene3")
> df
```

Barplot with table

```
      Time0 Time1 Time2
Gene1  0.29  1.19  0.90
Gene2  0.79 -1.69  0.89
Gene3  0.41  1.24  1.20
```

```
> library(plotrix)
> barp(df, ylab = "Expression Lvl vs Control",
+     names.arg = colnames(df), col = 1:3)
> addtable2plot(0.45, -1, df, bty = "o",
+     display.rownames = TRUE, hlines = TRUE,
+     title = "Data in table format")
```

Barplot with table



Data in table format

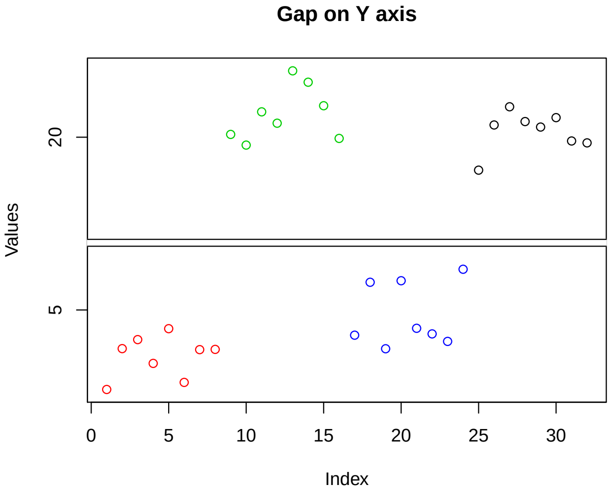
	Time0	Time1	Time2
Gene1	0.29	1.19	0.9
Gene2	0.79	−1.69	0.89
Gene3	0.41	1.24	1.2

Plot with gaps

- With Plotrix we can make plots that have a gap on one axis.
- For example, a normal `plot` with a gap on the Y axis.

```
> data <- c(rnorm(8) + 3, rnorm(8) +
+     21, rnorm(8) + 4.5, rnorm(8) +
+     20)
> color <- c(rep(2, 8), rep(3, 8),
+     rep(4, 8), rep(1, 8))
> gap.plot(data, gap = c(8, 16),
+     xlab = "Index", ylab = "Values",
+     main = "Gap on Y axis", col = color)
```

Plot with gaps

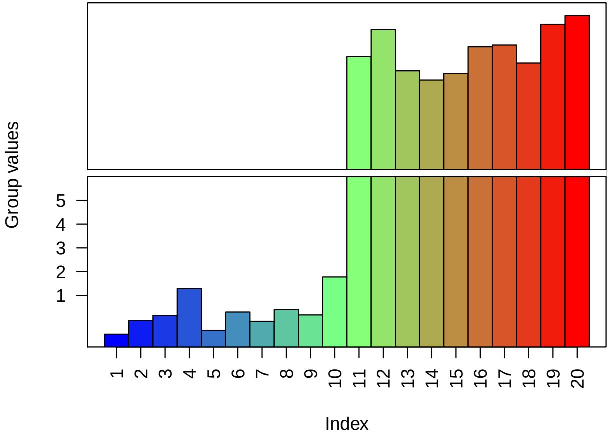


Gap on a barplot

- Or a barplot with a gap.
- Very helpful to visualize all your data.
- However, there is an issue with the labels on the Y axis T_T so be careful when using this kind of plot.

```
> data <- c(rnorm(10), rnorm(10) +
+     30)
> gap.barplot(data, gap = c(6, 25),
+     xlab = "Index", ytics = c(1:30),
+     ylab = "Group values", las = 2)
```

Gap on a barplot



Error bars

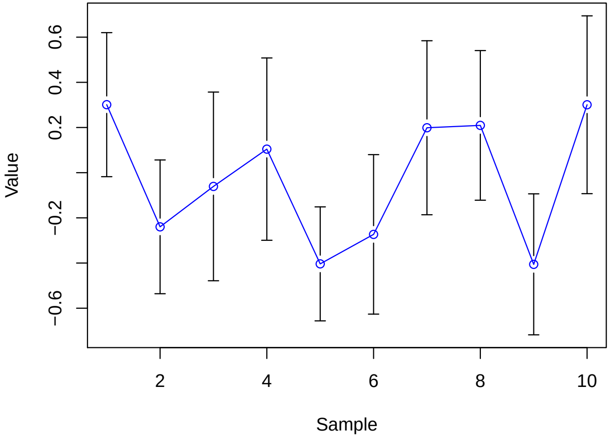
- Nowadays you get to see lots of graphs with the error bars.
- Experimental papers generally have 3 to 5 repeats of the same experiment.
- The dispersion function will be helpful to make this kind of plot.

```
> data <- matrix(rnorm(100), 10,
+     10)
> a <- colMeans(data)
> b <- std.error(data)
> plot(a, ylim = c(min(a - b), max(a +
+     b)), xlab = "Sample", ylab = "Value",
+     col = 4, type = "o")
```

Error bars

```
> dispersion(1:10, colMeans(data),
+     b)
```

Error bars



Some real data

- For the next plots, we'll use data from this article where they sequenced a Korean individual.
- I already saved as csv files two tables for easy import. We'll load them into R with the read.csv function.

```
> t1 <- read.csv("http://www.lcg.unam.mx/~lcollado/B/data/SuppTable01_kogen
+     header = T)
> t2 <- read.csv("http://www.lcg.unam.mx/~lcollado/B/data/SuppTable06_nsSnp
+     header = T)
```

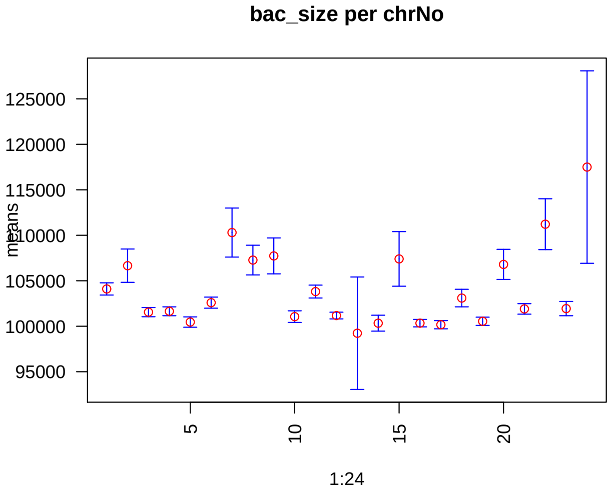
- Use head, dim, class to find out more about the data.

plotCI

- Plotrix has another function that plots error bars.
- We'll use our first table and get the data we need using tapply.

```
> means <- tapply(t1$bac_size, t1$chrNo,
+     mean)
> err <- tapply(t1$bac_size, t1$chrNo,
+     std.error)
> plotCI(1:24, means, err, col = "red",
+     scol = "blue", las = 2, main = "bac_size per chrNo")
```

plotCI

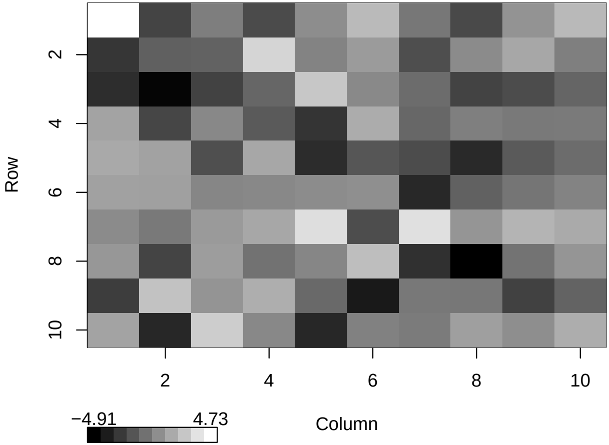

bac_size per chrNo
means
125000
120000
115000
110000
105000
100000
95000
5
10
15
20
1:24

One similar to image

- With color2D.matplot we can make plots very similar to image
- What differences do you notice vs image?

```
> mat <- matrix(rnorm(100, 0, 2),
+      10, 10)
> color2D.matplot(mat, show.legend = T)
```

One similar to image



Hierobarp

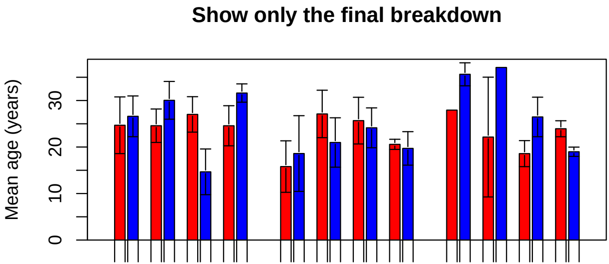
- We'll use the default example for this powerful plot.

```
> test.df <- data.frame(Age = rnorm(100,
+      25, 10), Sex = sample(c("M",
+      "F"), 100, TRUE), Marital = sample(c("D",
+      "M", "S", "W"), 100, TRUE),
+      Employ = sample(c("Full Time",
+          "Part Time", "Unemployed"),
+          100, TRUE))
> test.col <- list(Overall = "green",
+      Employ = c("purple", "orange",
+          "brown"), Marital = c("#1affd8",
+          "#caeecc", "#f7b3cc", "#94ebff"),
+      Sex = c(2, 4))
```

Hierobarp

```
> hierobarp(formula = Age ~ Sex +
+     Marital + Employ, data = test.df,
+     ylab = "Mean age (years)",
+     main = "Show only the final breakdown",
+     errbars = TRUE, col = test.col$Sex)
```

Hierobarp

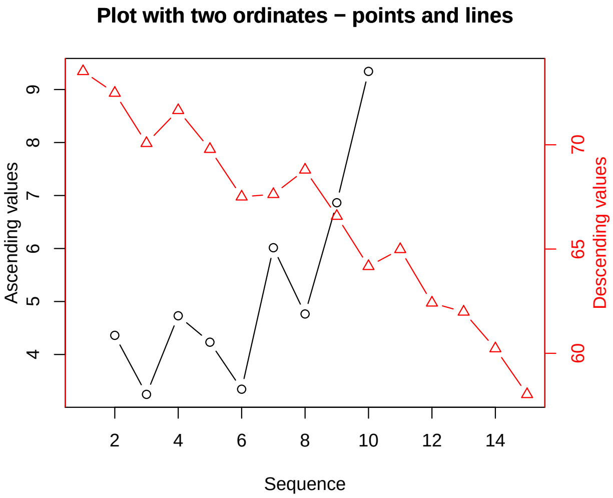


Two scales

- Sometimes you want two lines with different scales on the same plot.
- twoord.plot is the solution :)

```
> twoord.plot(2:10, seq(3, 7, by = 0.5) +
+     rnorm(9), 1:15, rev(60:74) +
+     rnorm(15), xlab = "Sequence",
+     ylab = "Ascending values",
+     rylab = "Descending values",
+     main = "Plot with two ordinates - points and lines")
```

Two scales

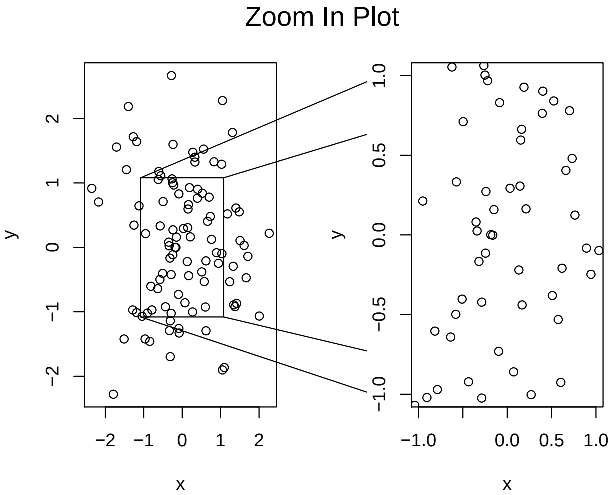

Plot with two ordinates − points and lines
Ascending values
4
5
6
7
8
9
Descending values
60
65
70
2
4
6
8
10
12
14
Sequence

Zoom

- The final plot I'll show you from `plotrix` enables us to zoom into a section of the plot.

```
> zoomInPlot(rnorm(100), rnorm(100),
+     rxlim = c(-1, 1), rylim = c(-1,
+         1), zoomtitle = "Zoom In Plot")
```

Zoom



Intro

- ggplot2 is a much more sophisticated plotting package.
- 199 pages long ref manual!!!
- Lets take a look at some examples.

Plotmatrix

- We'll use the iris data set which is used quite frequently to exemplify scatterplots.
- Meaning that you are using 3 or more variables.
- Explore `iris` with head and other similar functions.

```
> plotmatrix(iris[, 1:4])
```

Plotmatrix II

- If we combine plotmatrix with geom_smooth we can get a much better graph.

```
> plotmatrix(iris[, 1:4]) + geom_smooth(method = "lm")
```

We'll be back

- On the class where we'll learn about linear regressions, we'll be back and make plots like this one:

```
> mod <- lm(mpg ~ wt, data = mtcars)
> qplot(.fitted, .resid, data = mod) +
+     geom_hline() + geom_smooth(se = FALSE)
```

Intro

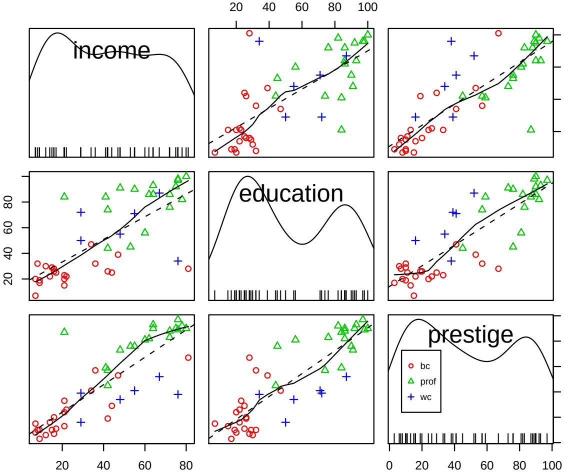
- While this package has quite a lot of functions too (105 page ref man), one special plot caught my eye.
- Feel free to check all the examples later if you want :D

scatterplot.matrix

- Quite similar plot to some we made before with automatic colors

```
> library(car)
> scatterplot.matrix(~income + education +
+     prestige | type, data = Duncan)
```

scatterplot.matrix

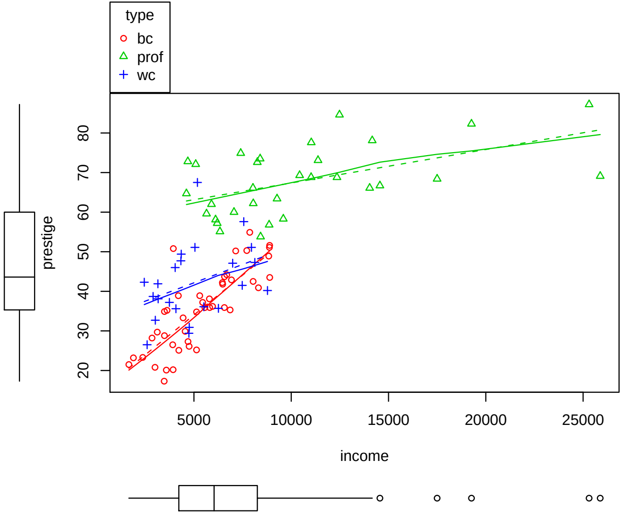


scatterplot

- With scatterplot we can create boxplots on our axis!!

```
> scatterplot(prestige ~ income |
+      type, data = Prestige, span = 1)
```

scatterplot



Session Info

```
> sessionInfo()

R version 2.10.0 Under development (unstable) (2009-07-21 r48968)
i386-pc-mingw32

locale:
[1] LC_COLLATE=English_United States.1252
[2] LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
[1] stats     graphics  grDevices
[4] utils     datasets  methods
[7] base

other attached packages:
[1] car_1.2-15
[2] plotrix_2.6-4
```

Session Info

```
[3] DAAG_1.00
[4] randomForest_4.5-30
[5] rpart_3.1-44
[6] MASS_7.3-0
[7] lattice_0.17-25

loaded via a namespace (and not attached):
[1] grid_2.10.0  tools_2.10.0
```