Exercise in Data Analytics and Causal Inference HS 2020, Part I: Descrptive statistics with R: an introduction

Professor Petyo Bonev

September 20, 2020

Contents

1 Organisation of the exercises

- Objectives: (i) learn concepts of data summary and visualization, (ii) learn how to perform causal analysis in the linear regression model and (iii) learn how to implement points (i) and (ii) in the statistical software R
- Preliminary structure of the exercises:
  - Exercise 1: (i) A very short introduction to the software R. (ii) Basic descriptive statistics with one variable.
  - Exercise 2: Basic descriptive statistics with one and two variables.
  - Exercise 3: Introduction to the linear model: basic concepts.
  - Exercise 4: Implementation of linear regression in R.
- There will be 2 problem sets on the topics in the exercises.
- You must solve these.
- Office hours: by appointment.
- Email: petyo.bonev@unisg.ch

2 One variable. The empirical distribution and its parameters.

2.1 Motivation

- Suppose there is a survey of households.
- We ask how many individuals live in the household.
- We get a list of **raw data**: $1, 3, 1, 2, \ldots$. This is a univariate case (one variable: number of individuals in a household).
- The total number of households surveyed, say n, is called sample size.
- How can we summarize the data?
- Two general ways : either through the **empirical distribution** or through some of its **parameters**.

2.2 The empirical distribution

- Empirical distribution describes the data in the sample.
- Theoretical distribution describes the data in the population.

- How is the data distributed: which values with which "probabilities".
- Population properties: probabilities, expectations, variances.
- Sample properties: frequencies, sample averages, sample variances.

2.2.1 Frequency tables

- A way to visualize the empirical distribution is through a frequency table.
- For each distinct (possible) value i we calculate the absolute frequency n_i and the relative frequency n_i/n.
- Here is an example in R.

```
> library(knitr)
> Survey.1=sample(c(1,2,3,4,5),
+ prob=c(0.3, 0.3, 0.2, 0.15,0.05), size=100, replace=T);
> Survey.1
```

```
  [1] 2 4 3 4 2 2 2 2 2 3 1 2 5 1 3 3 3 2 2 1 3 3 2 3 3 4 1
 [28] 1 2 4 4 2 3 3 1 2 2 1 4 4 4 4 4 2 2 4 3 4 4 4 2 2 1 2
 [55] 1 1 3 3 1 4 2 2 4 2 2 3 2 3 5 5 1 1 1 4 2 3 3 2 2 3 1
 [82] 1 3 2 2 1 3 1 5 2 1 1 5 1 3 2 2 3 1 4
```

- Look up values and absolute frequencies

```
> table(Survey.1)
```

```
Survey.1
 1  2  3  4  5
22 32 23 18  5
```

- To compute relative frequencies

```
> table(Survey.1)/length(Survey.1)
```

```
Survey.1
   1    2    3    4    5
0.22 0.32 0.23 0.18 0.05
```

- We can turn this table into a data array:

```
> Freq.table=table(Survey.1)
> Freq.table=as.data.frame(Freq.table)
> Freq.table
```

```
  Survey.1 Freq
1        1   22
2        2   32
3        3   23
4        4   18
5        5    5
```

```
> Rel.Freq=Freq.table[,2]/length(Survey.1)
> Rel.Freq
```

```
[1] 0.22 0.32 0.23 0.18 0.05
```

- This way makes sense only if only few possible/occured values: 1) the variable is discrete and 2) does not obtain two many values.
- What if there are many values of a discrete variable or the variable is continuous (can take any values between two different values)?
- Example: suppose we survey annual household income in 100 households.
- Simulate a new variable income:

```
> Income=sample(size=100, seq(from=20000, to=60000, by=1000), replac
```

- A table here makes no sense (too many values).

2.2.2 The empirical distribution function

- One way to summarize the data is with its **empirical distribution function**

$$\hat{F}(x) := \frac{\text{Number of observations} \leq x}{\text{Total number of observations}}.$$

- E.g. $\hat{F}(30000) \approx$ the probability that income is below 30000:

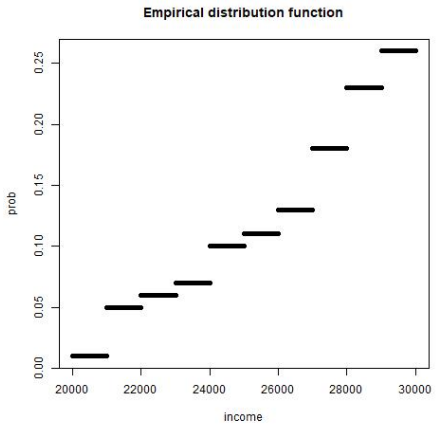
```
> sum(Income<30000)/length(Income)
```

```
[1] 0.23
```

- The intuition is clear: we try to approximate the $P\{X \leq x\}$.
- The empirical distribution function computes this probability for every value x (not only for 30000).
- In R:

```
> F.hat=function(x){sum(Income<x)/length(Income)}
> Emp.dist.income=vector(length=10000)
> for(i in 20001: 30000){Emp.dist.income[i-20000]=F.hat(i)}
> # Suppress the plot, sweave bug in option fig=T
> # jpeg("Emp_Dist_Function.jpg")
> #plot(20001:30000, Emp.dist.income, main="Empirical distribution fu
> # dev.off()
```

- An R plot of the empirical distribution function:



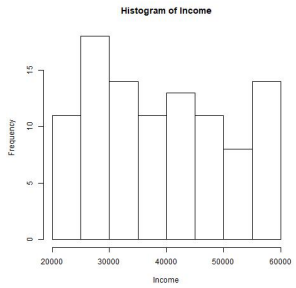
- It is a step-wise function.
- The jumps are at the observations.
- It can be seen as a (relative) counting function.
- The more observations we have, the closer we get to the true (i.e. population) probabilities.

2.2.3 The histogram

- An alternative way is through a histogram.
- The histogram counts frequencies for bins (classes).
- These frequencies can be both absolute and relative.
- When relative, then the surfaces (i) must add up to 1 and (2)correspond to the relative "weights".
- R Code:

```
> # hist(Income)    # runs a histogram
> #jpeg("Histogram_income.jpg")   # Save it as a picture
```

```
> #hist(Income)
> #dev.off()
```



Task: manipulate the histogram to show relative frequencies

2.3 Parameters of the empirical distribution

- There are many informative parameters.
- A parameter describes the distribution in a compact way
- It is a summary statistic/descriptive statistic.
- We look at the most common parameters
- The most important one is the sample average $\bar{X}$.
- Definition: $\bar{X} = n^{-1} \sum_{i=1}^{n} X_i$
- Example: what is the average wage in the university sector?
- Ex ante, it is a random variable.
- Once we learn the realisations $x_1, \ldots, x_n$, it is a constant.
- With R:

```
> mean(Income)  # calculate sample average

[1] 40100
```

- The sample average is useful when the distribution is close to symmetric.
- It can be misleading otherwise.
- Also, if there is a lot of variation, the mean is not very informative.
- Biggest problem: outlier in small samples.

- In addition: sample average might not be a value of the distribution itself (average gender?)
- Parameter robust to outliers: median.
- Median is defined as the middle value.
- Example 1: $1, 2, 2, 3, 4, 4, 1000$: Median is 3.
- Example 2: $1, 2, 3, 4, 5, 7, 8, 9$: Median is 4.5.
- In R:

```
> median(Income)

[1] 39500
```

- When no outliers, mean often better as a measure for centrality.
- Asymptotic theory for median more involved.
- Generally: the α-quantile has $(\alpha * 100)\%$ of the points behind.
- $q(\alpha) : P\{X \leq q(\alpha)\} = \alpha$
- Example with R:

```
> quantile(Income, probs=c(0,0.1, 0.25, 0.5, 0.75,1))

   0%   10%   25%   50%   75%  100%
20000 24900 30000 39500 50000 60000

> min(Income)

[1] 20000

> max(Income)

[1] 60000
```

- Summary: common quantiles (min, max, 0.25, 0.5,0.75, 1) and mean:

```
> summary(Income)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  20000   30000   39500   40100   50000   60000
```

- $q(0.25)$, $q(0.5)$ and $q(0.75)$ are called first, second and third quartiles.
- We can visualize a distribution in terms of quantiles.

- One common way: the boxplot.
- The box contains between 25% and 75% quantile.
- Define $D = q(0.75) - q(0.25)$.
- The highest(lowest) line:
  - Either $1.5D$-distnace from $q(0.75)(q(0.25))$: upper (lower) Whisker
  - Or maximum (minimum).
  - The first approach helps identify outliers.

- Task: plot a boxplot with $1.5D$ definition of whiskers. Difference to this one?
- Another parameter: the sample variance S_X^2
- Definition: $S_X^2 = n^{-1} \sum_{i=1}^{n} (X_i - \bar{X})^2$.
- It holds

$$n^{-1} \sum_{i=1}^{n} X_i^2 - (\bar{X})^2 \tag{1}$$

- Intuition behind S_X^2: average deviation around the sample average
- Sample mean and sample variance are good proxies for population expectation ($\mathbb{E}[X]=$ and variance ($\mathbb{V}[X]$), resp..
- Variance in R:

```
> Income=runif(20000, 100000) # generate vector income
> var(Income)
```

```
[1] NA
```

- Sample standard deviation: $S_X = \sqrt{\{S_X^2\}}$. In R:

```
> sd(Income)

[1] NA
```

- Sample covariance $S_{X,Y}$: how two variables co-move in the sample.

$$S_{X,Y} = n^{-1} \sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y}) \tag{2}$$

- Its realization $s_{x,y}$ is computed in R as:

```
> Men.intelligence=rnorm(4000 , 100 , 30)
> Women.intelligence=rnorm(4000 , 200 , 30)
> cov(Men.intelligence, Women.intelligence)

[1] 1.442447
```

- Standardized version: correlation:$\rho(x, y) = \frac{S_{X,Y}}{S_X S_Y}$
- Its values are between -1 and 1
- -1 (1): perfectly negatively (positively) correlated

```
> cor(Men.intelligence, Women.intelligence)

[1] 0.001623655
```

3 Two variables. Visualization of the empirical distribution and its parameters.

3.1 Scatter plots

- We now investigate several ways to visualize relationships
- We focus on: two variables, raw data.
- Let's use an inbuilt R dataset
- Check inbuilt datasets

```
> data(package='datasets')
```

- We choose a dataset on chicken weights
- Load dataset ChickWeight. Weights in time under 4 different diets.

```
> data(ChickWeight)
```

- Which variables do we have?

```
> names(ChickWeight)

[1] "weight" "Time"   "Chick"  "Diet"
```

- Let's check several observations:

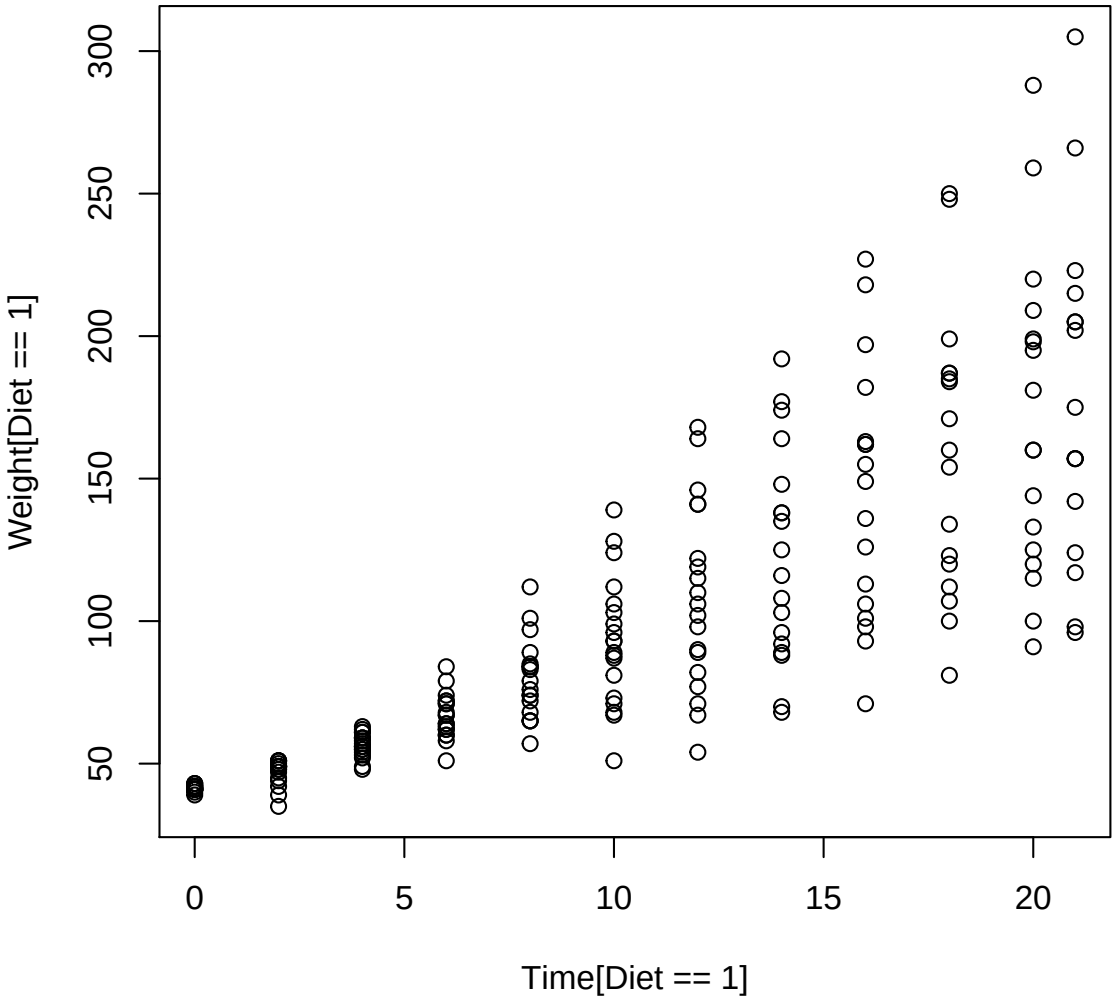
```
> head(ChickWeight)

  weight Time Chick Diet
1     42    0     1    1
2     51    2     1    1
3     59    4     1    1
4     64    6     1    1
5     76    8     1    1
6     93   10     1    1
```

- Let's define some variables:

```
> Weight=ChickWeight$weight
> Time=ChickWeight$Time
> Diet=ChickWeight$Diet
> summary(Weight)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   35.0    63.0   103.0   121.8   163.8   373.0
```

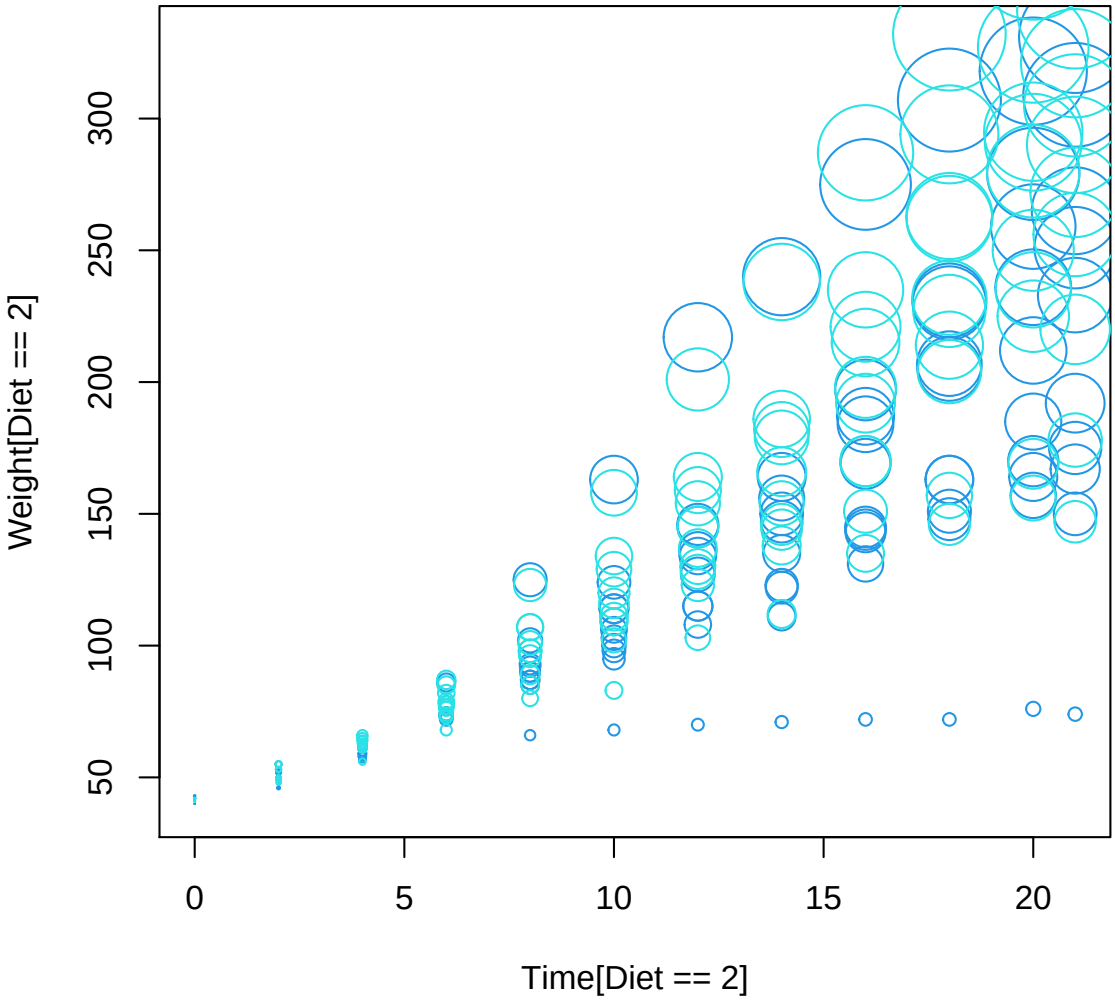
- The easiest scatter plot:

```
> plot(Time[Diet==1], Weight[Diet==1])
```



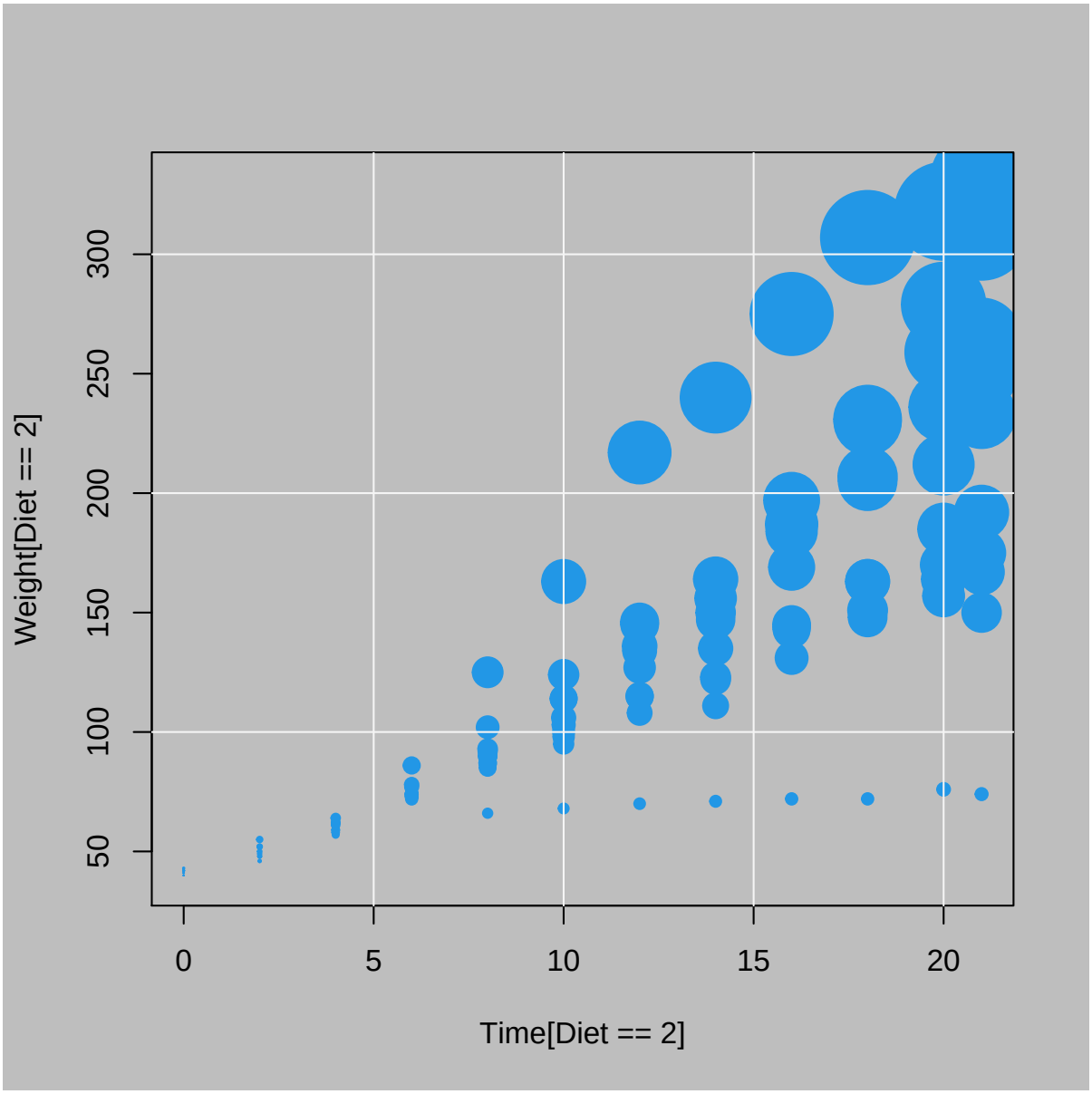
- Task: add proper axis labels and title. (Use help(plot)).
- Let the size of points be proportional to the weight values.
- Compare two different diets (choose diff. colors).

```
> Increase.2=(Weight[Diet==2]-min(Weight[Diet==2]))/+
+   min(Weight[Diet==2])
> Increase.3=(Weight[Diet==3]-min(Weight[Diet==3]))/+
+   min(Weight[Diet==3])
> plot(Time[Diet==2], Weight[Diet==2], cex=Increase.2, col=4)
> points(Time[Diet==2], Weight[Diet==3], cex=Increase.3, col=5)
```



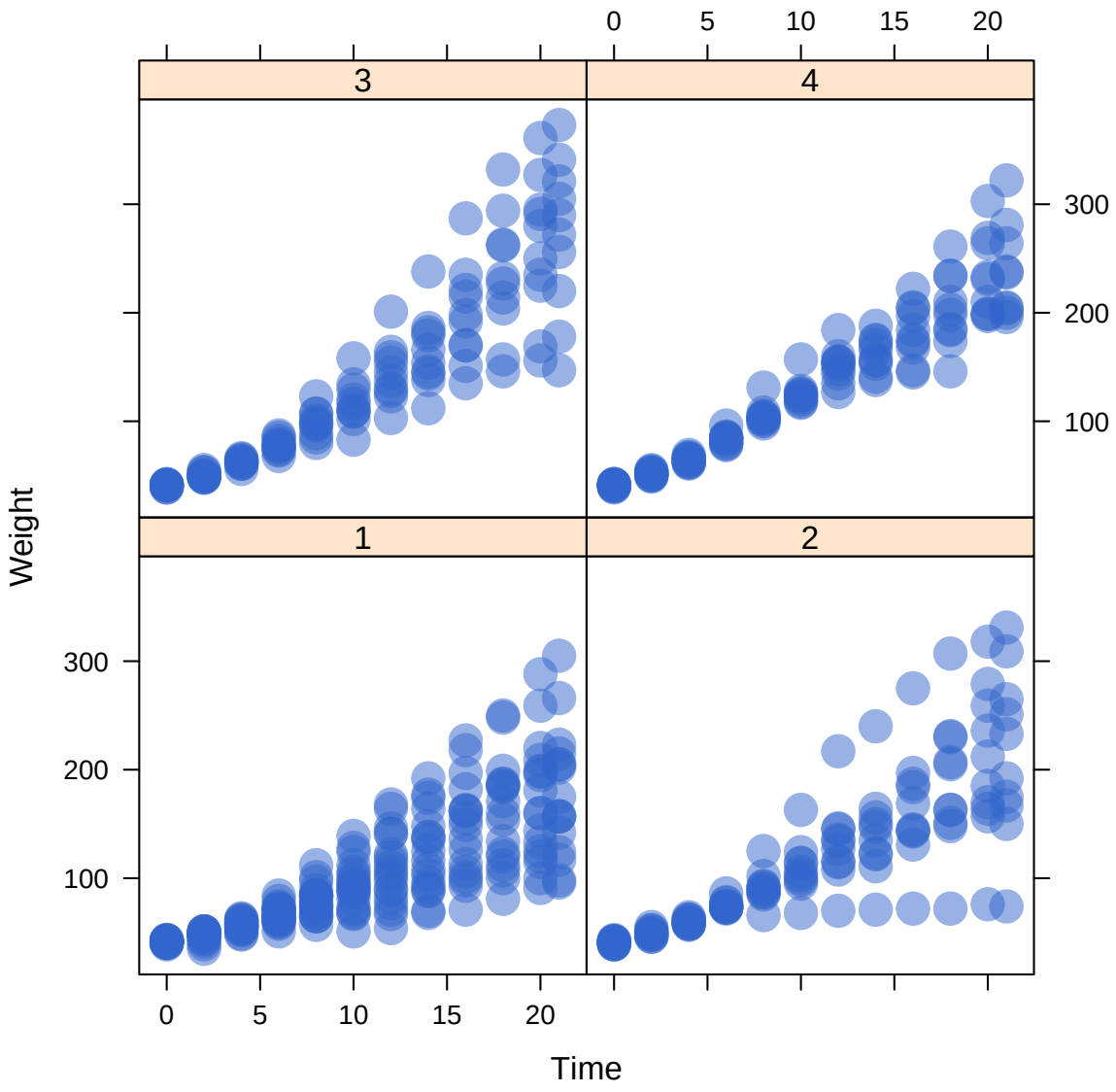
- See the "+" sign in the code? It is only for visualizations purposes. Codes stays in margins.
- When you run the code in R, delete these + signs.
- Task: how can we add a legend to this plot?
- Let's make background grey and add some location lines.

```
> par(bg = "grey")
> plot(Time[Diet == 2], Weight[Diet == 2], cex = Increase.2,
+      col = 4, pch = 19)
> abline(v = c(5, 10, 15, 20), h = c(100, 200, 300),
+      col = "whitesmoke")
```



- We might want to depict different graphics in different plots

```
> par(bg = "white")
> library(lattice)
> xyplot(Weight ~ Time | Diet, data = ChickWeight,
+     pch = 20, cex = 3, col = rgb(0.2, 0.4, 0.8,
+         0.5))
```



- Task: what is "rgb" in the code above?
- Finally, we might want to look up the value of the weight directly on the graph:

```
> ## RUN THIS CODE IN R
> # data(ChickWeight)
> # attach(ChickWeight)
> # library(plotly)      # this is a cool package
> #mygraph=plot_ly(x=Time, y=weight, mode="markers",+
> #size=weight, color=ifelse(Time>10, "blue", +
> #"red"), type="scatter")
> # mygraph;
```

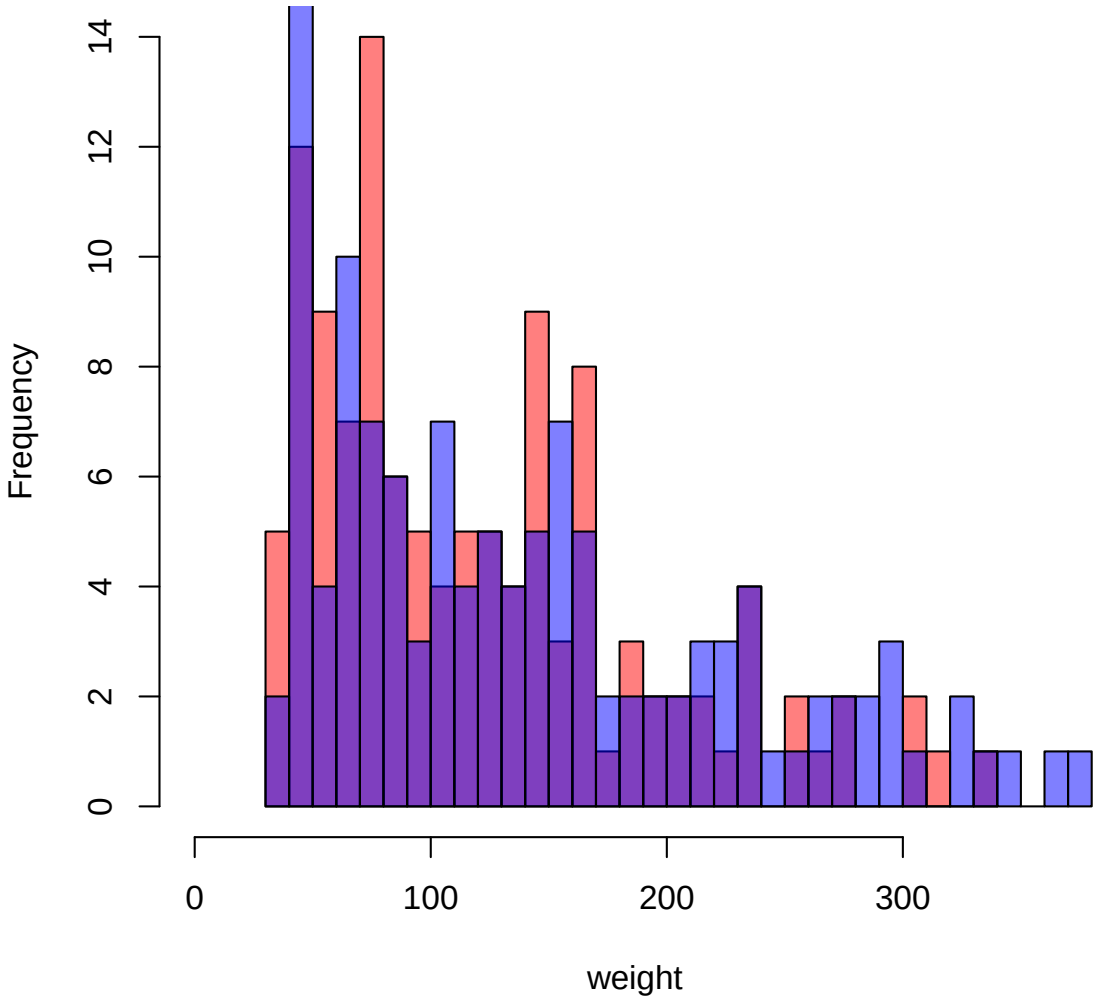
3.2 Further ways to visualize relationships between two or more variables

- Sofar scatter plots (plots of raw data)
- Now: plots of summaries of empirical distributions

3.2.1 Two or more histograms on the same plot

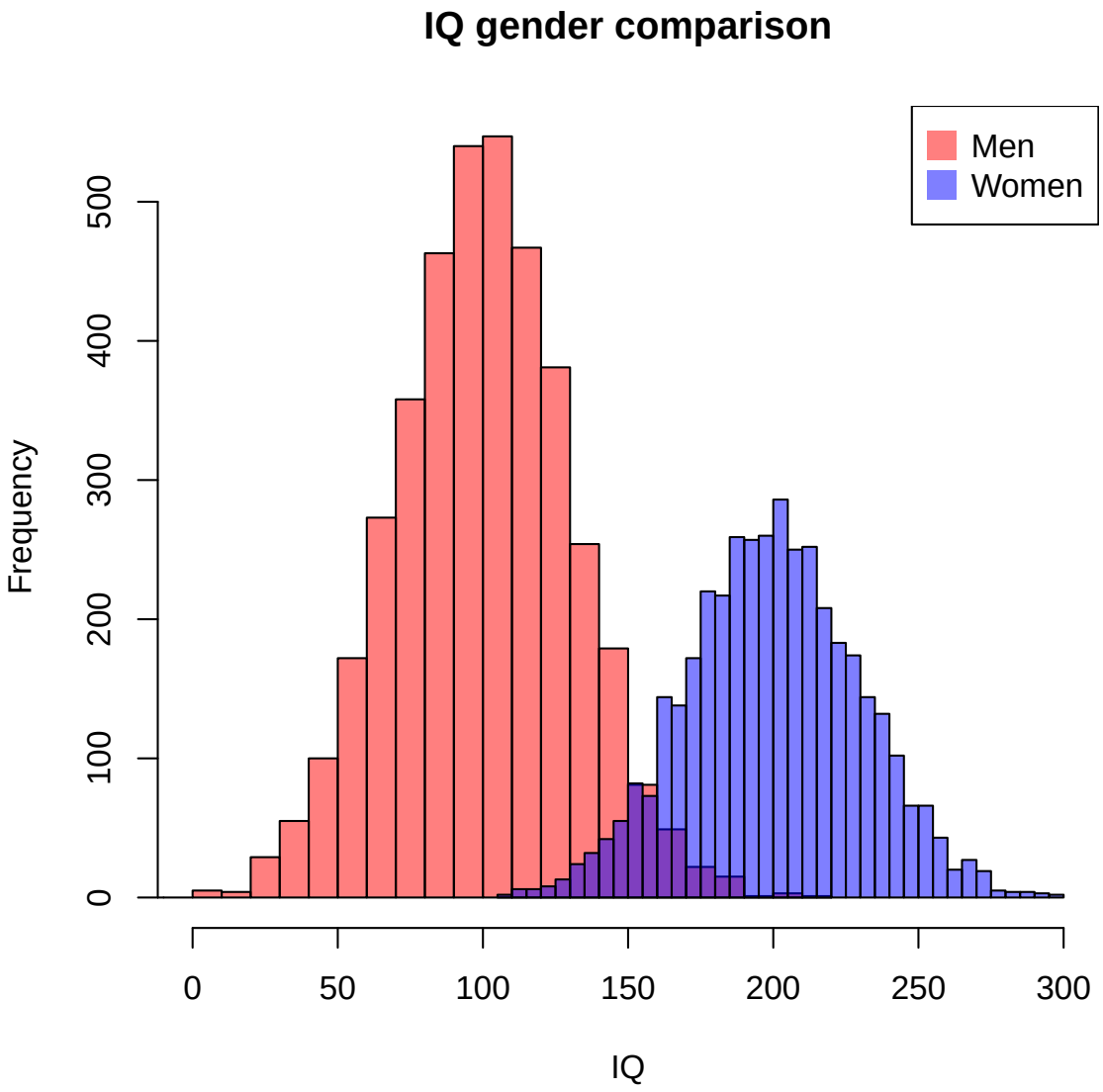
- First easy way: two (or more) histograms
- You need the option "add=T"

```
> hist(Weight[Diet == 2], breaks = 30, xlim = c(0,
+     max(Weight)), col = rgb(1, 0, 0, 0.5), xlab = "weight",
+     main = "Comparison of Diets")
> hist(Weight[Diet == 3], breaks = 30, xlim = c(0,
+     max(Weight)), col = rgb(0, 0, 1, 0.5), add = T)
```



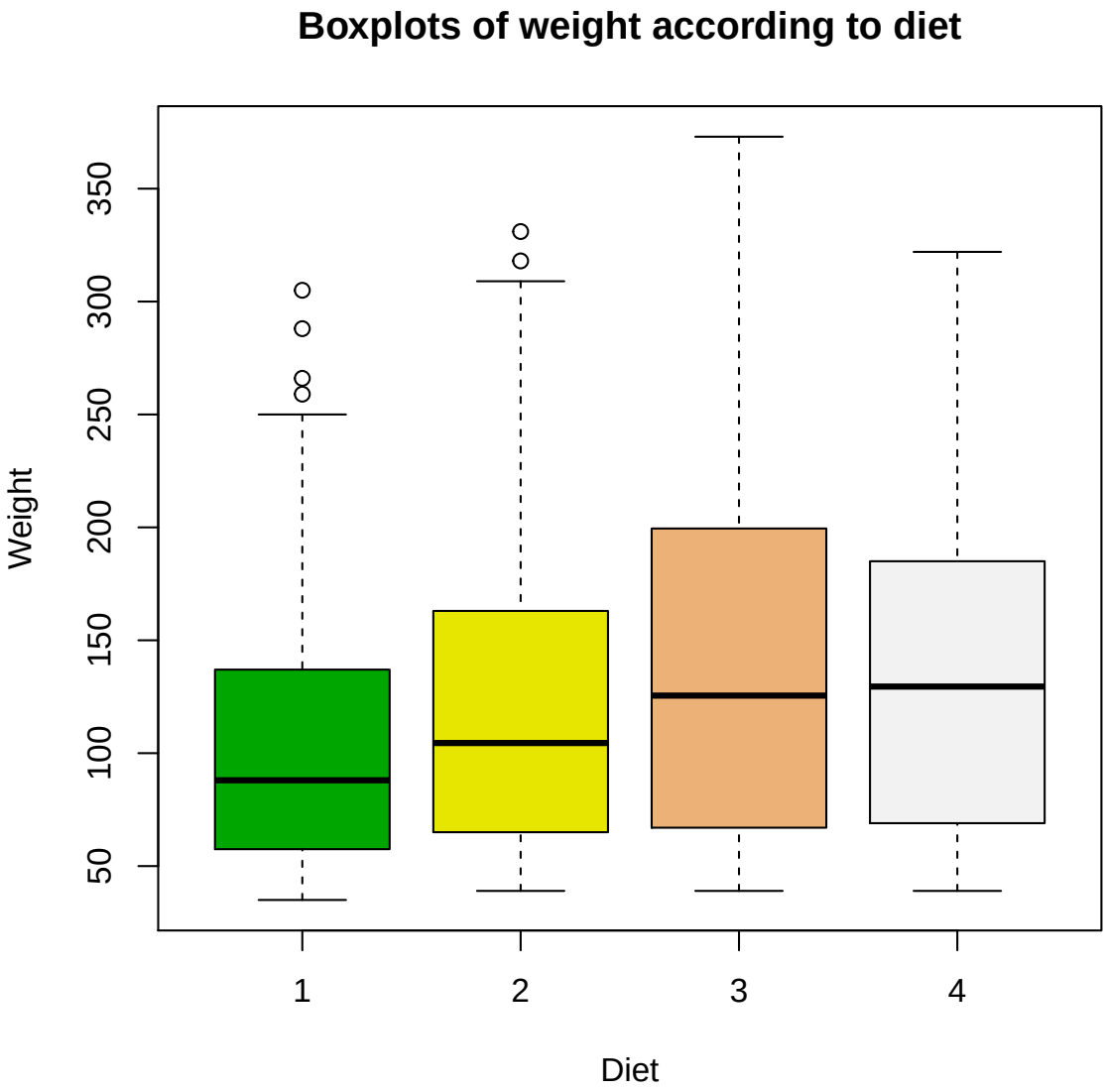
- In this case, not the best visualisation.
- Works better, if the (two) distributions are very different.
- Example:

```
> hist(Men.intelligence, breaks = 30, xlim = c(0,
+     300), col = rgb(1, 0, 0, 0.5), xlab = "IQ",
+     main = "IQ gender comparison")
> hist(Women.intelligence, breaks = 30, xlim = c(0,
+     300), col = rgb(0, 0, 1, 0.5), add = T)
> legend("topright", legend = c("Men", "Women"),
+     col = c(rgb(1, 0, 0, 0.5), rgb(0, 0, 1, 0.5)),
+     pt.cex = 2, pch = 15)
```



- Finally, we could decide to view the chicken weights separately (with no time reference) as boxplots

```
> boxplot(Weight ~ Diet, col = terrain.colors(4),
+     main = "Boxplots of weight according to diet")
```



- What can we say about the four diets according to this picture?

3.2.2 2D density

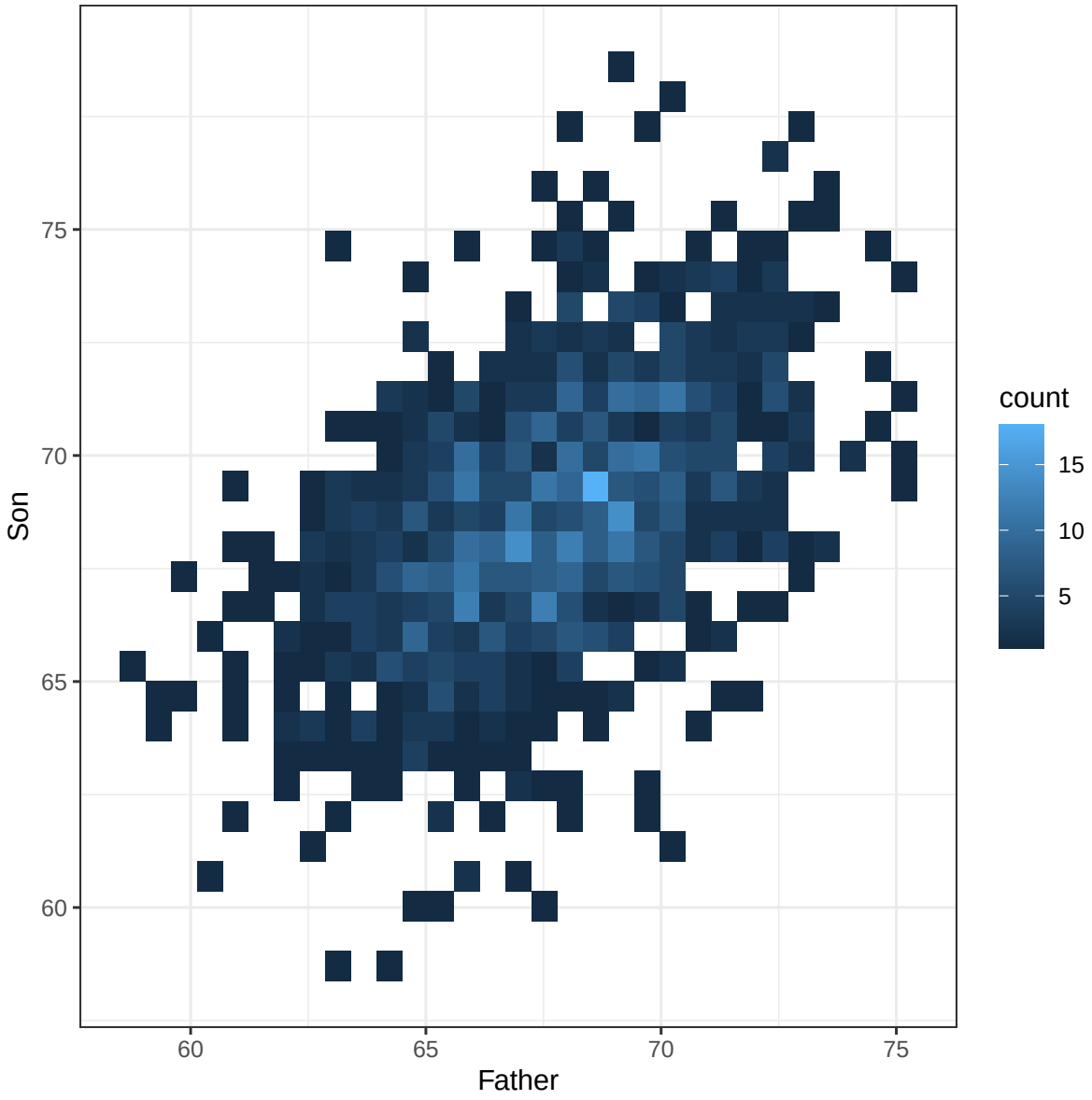
- Next visualization possibility: plot 2D density
- 2-D density: $f(x, y)$ with $\int \int f(x, y)dxdy = 1$
- In addition: $P(X < w, Y < z) = \int_{\infty}^{w} \int_{\infty}^{z} f(x, y)dydx$
- 2D density is therefore something like a 2D histogram
- Let's briefly study a famous dataset
- Pearson's heights of son and father
- Over 1000 "couples" of sons' and fathers'

- Downloaded: http://www.randomservices.org/random/data/Pearson.html
- Let's first prepare the data for work:

```
> Pearson.data=read.table("Pearson.txt", header=T)
> head(Pearson.data)        # look up the data

  Father  Son
1   65.0 59.8
2   63.3 63.2
3   65.0 63.3
4   65.8 62.8
5   61.1 64.3
6   63.0 64.2

> class(Pearson.data)        # What object do we deal with?

[1] "data.frame"

> attach(Pearson.data)
> summary(Father)                 # Check descr. stat Father

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  59.00   65.80   67.80   67.69   69.60   75.40
```

- Now let's visualize the 2D density

```
> library(tidyverse)                # need package to visualize
> # Make sure in the following code to include the
> # "+" signs on the right
> ggplot(Pearson.data, aes(x=Father, y=Son))+
+ geom_bin2d()+
+ theme_bw()
```



- Task: look up the help function: aes: you specify the variables for the plot
- with geombin2d: Divides the plane into rectangles, counts the number of cases in each rectangle, and then (by default) maps the number of cases to the rectangle's fil.

3.2.3 Covariance matrix

- Many (>2) variables ⇒ we can display covariances in a matrix.
- Covariance matrix:
  - If we have r.v. $X_1, \ldots X_n$
  - calculate sample covariances S_{X_i,X_j} for all $i, j \in 1, \ldots, n$.

– display them in a matrix Σ

– (i, j)-th el. $\Sigma_{i,j} = S_{X_i,X_j}$

- Example: Download from R the dataset on ability
- It is a covariance matrix with each cell the covariance between the average grades in a subject
- 6 tests (reading, voc, ect), 112 individuals

```
> data("ability.cov")
> names(ability.cov)

[1] "cov"    "center" "n.obs"

> # We need the cov object!
> dim(ability.cov$cov)                  # what are

[1] 6 6

> #the dimensions of this matrix?
> # display data
> ability.cov$cov

        general picture  blocks   maze reading   vocab
general  24.641   5.991  33.520  6.023  20.755  29.701
picture   5.991   6.700  18.137  1.782   4.936   7.204
blocks   33.520  18.137 149.831 19.424  31.430  50.753
maze      6.023   1.782  19.424 12.711   4.757   9.075
reading  20.755   4.936  31.430  4.757  52.604  66.762
vocab    29.701   7.204  50.753  9.075  66.762 135.292
```

- Not very informative (Cov-matrix not standardized)!
- Recompute it to get correlation matrix

```
> # Cor.int=matrix(nrow=6, ncol=6)
> # for(i in 1:6){
> # for(j in 1:6){
> # Cor.int[i,j]= ability.cov$cov[i,j]/
> # sqrt(ability.cov$cov[i,i]* ability.cov$cov[j,j])
> #   }
> #}
```

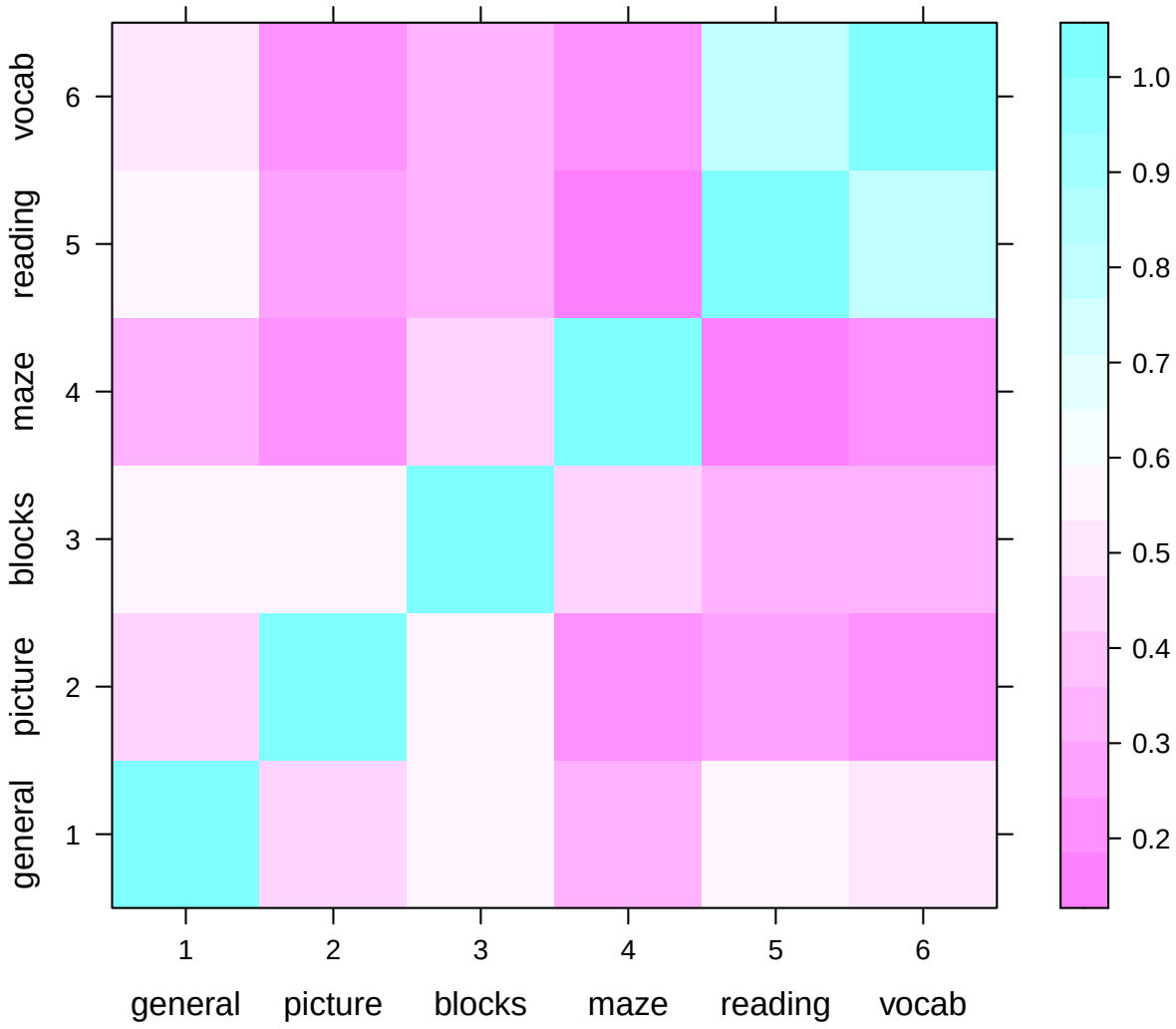
- Display now the matrix

```
> Cor.int
```

```
          [,1]      [,2]      [,3]      [,4]      [,5]
[1,] 1.0000000 0.4662649 0.5516632 0.3403250 0.5764799
[2,] 0.4662649 1.0000000 0.5724364 0.1930992 0.2629229
[3,] 0.5516632 0.5724364 1.0000000 0.4450901 0.3540252
[4,] 0.3403250 0.1930992 0.4450901 1.0000000 0.1839645
[5,] 0.5764799 0.2629229 0.3540252 0.1839645 1.0000000
[6,] 0.5144058 0.2392766 0.3564715 0.2188370 0.7913779
          [,6]
[1,] 0.5144058
[2,] 0.2392766
[3,] 0.3564715
[4,] 0.2188370
[5,] 0.7913779
[6,] 1.0000000
```

- Much better now. But still not very clear/obvious.
- Want to get a picture of it?
- We need a cool package

```
> require(lattice)
> levelplot(Cor.int, xlab = c("general", "picture",
+     "blocks", "maze", "reading", "vocab"), ylab = c("general",
+     "picture", "blocks", "maze", "reading", "vocab"))
```



- Such exercises in psychology are important.
- Researchers believe that several underlying intelligence factors make us smart in related fields.
- Ask a battery of questions and find which questions get correlated answers.
- This is the field of factor analysis.