Presentation of Scientific Results

Arnaud Legrand

MOSIG/PDES, Performance Evaluation Workshop,
Grenoble, November 3, 2014

Outline

1. Data Visualization
   - Motivation
   - Jain, Chapter 10
2. R Crash Course
   - General Introduction
   - Reproducible Documents: knitR
   - Introduction to R
   - Needful Packages by Hadley Wickam

Outline

Why do we need to visualise ? The Anscombe's Quartet

$X^{(1)}$	$Y^{(1)}$
10.00	8.04
8.00	6.95
13.00	7.58
9.00	8.81
11.00	8.33
14.00	9.96
6.00	7.24
4.00	4.26
12.00	10.24
7.00	4.82
5.00	5.68

$N = 11$ samples
Mean of $X = 9.0$
Mean of $Y = 7.5$

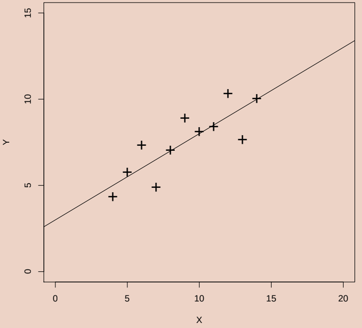
Correlation $= 0.816$

Why do we need to visualise ? The Anscombe's Quartet

$X^{(1)}$	$Y^{(1)}$
10.00	8.04
8.00	6.95
13.00	7.58
9.00	8.81
11.00	8.33
14.00	9.96
6.00	7.24
4.00	4.26
12.00	10.24
7.00	4.82
5.00	5.68

$N = 11$ samples
Mean of $X = 9$
Mean of $Y = 7$
Intercept = 3
Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

Scatter plot

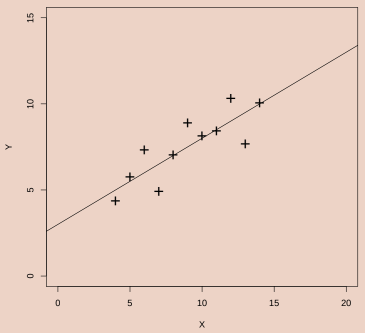


Why do we need to visualise ? The Anscombe's Quartet

$X^{(1)}$	$Y^{(1)}$
10.00	8.04
8.00	6.95
13.00	7.58
9.00	8.81
11.00	8.33
14.00	9.96
6.00	7.24
4.00	4.26
12.00	10.24
7.00	4.82
5.00	5.68

$N = 11$ samples
Mean of $X = 9$
Mean of $Y = 7$
Intercept $= 3$
Slope $= 0.5$
Res. stdev $= 1.237$
Correlation $= 0.816$

Scatter plot



1. The data set "behaves like" a linear curve with some scatter;
2. There is no justification for a more complicated model (e.g., quadratic);
3. There are no outliers;
4. The vertical spread of the data appears to be of equal height irrespective of the X-value; this indicates that the data are equally-precise throughout and so a "regular" (that is, equi-weighted) fit is appropriate.

Why do we need to visualise ? The Anscombe's Quartet

$X^{(1)}$	$Y^{(1)}$
10.00	8.04
8.00	6.95
13.00	7.58
9.00	8.81
11.00	8.33
14.00	9.96
6.00	7.24
4.00	4.26
12.00	10.24
7.00	4.82
5.00	5.68

$N = 11$ samples
Mean of $X = 9.0$
Mean of $Y = 7.5$
Intercept = 3
Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

$X^{(2)}$	$Y^{(2)}$
10.00	9.14
8.00	8.14
13.00	8.74
9.00	8.77
11.00	9.26
14.00	8.10
6.00	6.13
4.00	3.10
12.00	9.13
7.00	7.26
5.00	4.74

$N = 11$ samples
Mean of $X = 9.0$
Mean of $Y = 7.5$
Intercept = 3
Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

$X^{(3)}$	$Y^{(3)}$
10.00	7.46
8.00	6.77
13.00	12.74
9.00	7.11
11.00	7.81
14.00	8.84
6.00	6.08
4.00	5.39
12.00	8.15
7.00	6.42
5.00	5.73

$N = 11$ samples
Mean of $X = 9.0$
Mean of $Y = 7.5$
Intercept = 3
Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

$X^{(4)}$	$Y^{(4)}$
8.00	6.58
8.00	5.76
8.00	7.71
8.00	8.84
8.00	8.47
8.00	7.04
8.00	5.25
19.00	12.50
8.00	5.56
8.00	7.91
8.00	6.89

$N = 11$ samples
Mean of $X = 9.0$
Mean of $Y = 7.5$
Intercept = 3
Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

Why do we need to visualise ? The Anscombe's Quartet

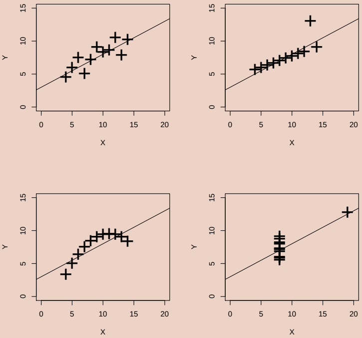
$X^{(1)}$	$Y^{(1)}$
10.00	8.04
8.00	6.95
13.00	7.58
9.00	8.81
11.00	8.33
14.00	9.96
6.00	7.24
4.00	4.26
12.00	10.24
7.00	4.82
5.00	5.68

$X^{(2)}$	$Y^{(2)}$

$X^{(3)}$	$Y^{(3)}$

$X^{(4)}$	$Y^{(4)}$

Scatter plot



$N = 11$ samples
Mean of $X = 9$
Mean of $Y = 7$
Intercept = 3
Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

Slope = 0.5
Res. stdev = 1.237
Correlation = 0.816

Why do we need to visualise ? The Anscombe's Quartet

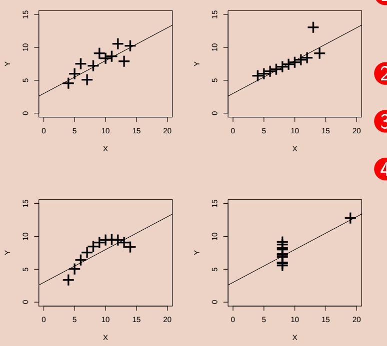
$X^{(1)}$	$Y^{(1)}$
10.00	8.04
8.00	6.95
13.00	7.58
9.00	8.81
11.00	8.33
14.00	9.96
6.00	7.24
4.00	4.26
12.00	10.24
7.00	4.82
5.00	5.68

$X^{(2)}$	$Y^{(2)}$

$X^{(3)}$	$Y^{(3)}$

$X^{(4)}$	$Y^{(4)}$

Scatter plot



1. data set 1 is clearly linear with some scatter.
2. data set 2 is clearly quadratic.
3. data set 3 clearly has an outlier.
4. data set 4 is obviously the victim of a poor experimental design with a single point far removed from the bulk of the data "wagging the dog".

$N = 11$ samples
Mean of $X = 9$
Mean of $Y = 7$
Intercept $= 3$
Slope $= 0.5$
Res. stdev $= 1.237$
Correlation $= 0.816$

Slope $= 0.5$
Res. stdev $= 1.237$
Correlation $= 0.816$

Slope $= 0.5$
Res. stdev $= 1.237$
Correlation $= 0.816$

Slope $= 0.5$
Res. stdev $= 1.237$
Correlation $= 0.816$

Problem statement

- All analysis we perform rely on (sometimes implicit) assumptions. If these assumptions do not hold, the analysis will be a complete non-sense.
- Checking these assumptions is not always easy and sometimes, it may even be difficult to list all these assumptions and formally state them.

A visualization can help to check these assumptions.

- Visual representation resort to our cognitive faculties to check properties.
  The visualization is meant to let us detect expected and unexpected behavior with respect to a given model.

Using the “right” representations

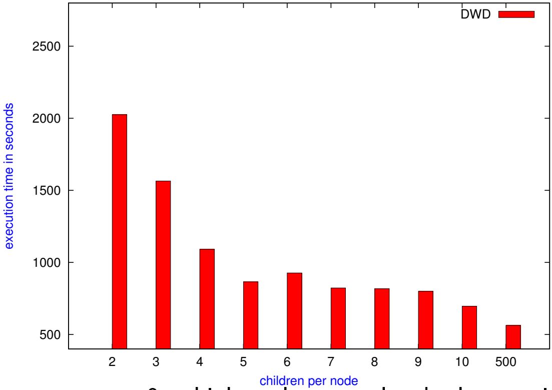
- The problem is to represent on a limited space, typically a screen with a fixed resolution, a meaningful information about the behavior of an application or system.
- $\leadsto$ need to aggregate data and be aware of what information loss this incurs.
- Every visualization emphasizes some characteristics and hides others. Being aware of the underlying models helps choosing the right representation.

Visualization and intuition

- Visualization can also be used to guide your intuition.
  Sometimes, you do not know exactly what you are looking for and looking at the data just helps.
- Some techniques (Exploratory Data Analysis) even build on this and propose to summarize main characteristics in easy-to-understand form, often with visual graphs, without using a statistical model or having formulated a hypothesis.
- **Use with care**, visualizations always have underlying models: when visualization is not adapted, what you may observe may be meaningless. Such approaches may help formulating hypothesis but these hypothesis have then to be tested upon new data-sets.

A "simple" graphical check for investigating speedup/scalability

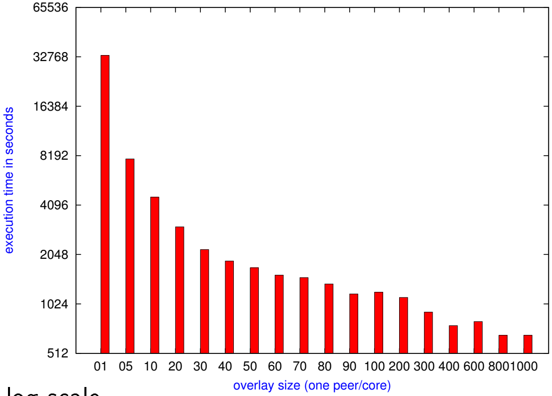
Plotting T_p versus p.



- y-axis does not start at 0, which makes speedup look more impressive
- x-axis is linear with an outlier.

A "simple" graphical check for investigating speedup/scalability

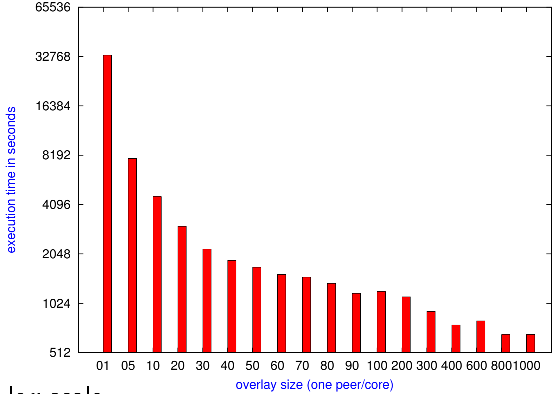
Plotting T_p versus p.



- y-axis uses log-scale
- x-axis is neither linear nor logarithmic so we cannot reason about the shape of the curve

A “simple” graphical check for investigating speedup/scalability

Plotting T_p versus p.

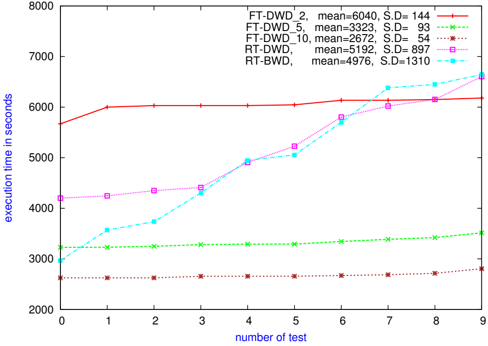


- y-axis uses log-scale
- x-axis is neither linear nor logarithmic so we cannot reason about the shape of the curve

Say, we want to test for Amhdal’s law. Propose a better representation.

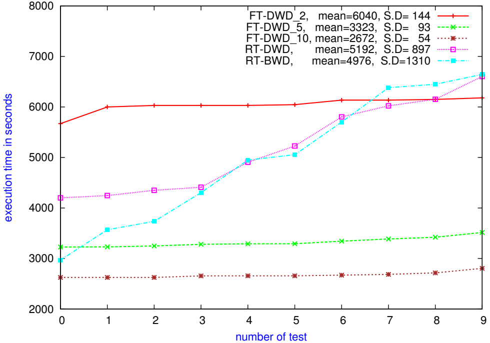
Graphically checking which alternative is better ?

5 different alternatives (`FT-DWD_2`, `FT-DWD_5`, `FT-DWD_10`, `RT-DWD`, `RT-BWD`), each tested 10 times.



Graphically checking which alternative is better ?

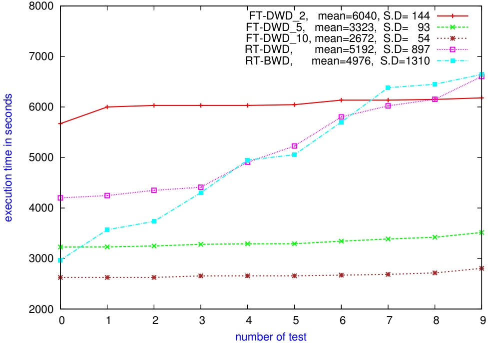
5 different alternatives (FT-DWD_2, FT-DWD_5, FT-DWD_10, RT-DWD, RT-BWD), each tested 10 times.



- Outcomes have been sorted by increasing value for each alternative and are then linked together.
- The line does not make any sense.
- Experiment order does not make any sense and makes it look like alternatives have been evaluated in 10 different settings (, which means they can be compared with each others for each setting).

Graphically checking which alternative is better ?

5 different alternatives (FT-DWD_2, FT-DWD_5, FT-DWD_10, RT-DWD, RT-BWD), each tested 10 times.



- Outcomes have been sorted by increasing value for each alternative and are then linked together.
- The line does not make any sense.
- Experiment order does not make any sense and makes it look like alternatives have been evaluated in 10 different settings (, which means they can be compared with each others for each setting).

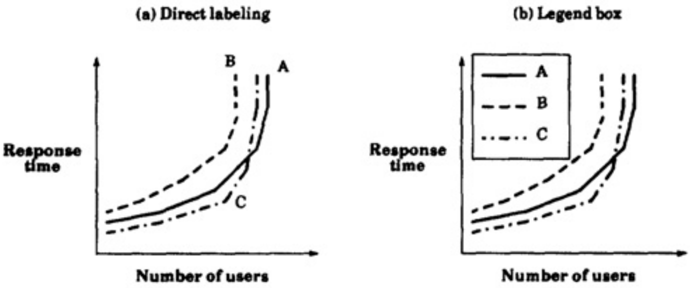
Outline

Read the basics

- For all such kind of "general" graphs where you summarize the results of several experiments, the very least you need to read is Jain's book.
- It has check lists for "Good graphics", which I made more or less available on the lecture's webpage.
- It presents the most common pitfalls in data representation
- It will teach how to cheat with your figures...
- ... and how to detect cheaters. ;)

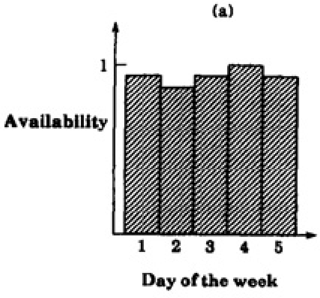
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



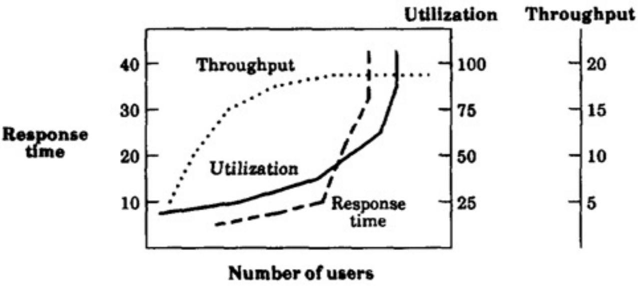
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



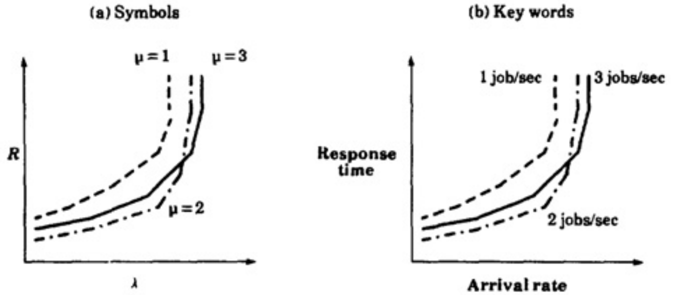
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, ...)
2. Maximize information (self-sufficient, clear labels, units, ...)
3. Minimize Ink (avoid cluttered information...)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, ...)



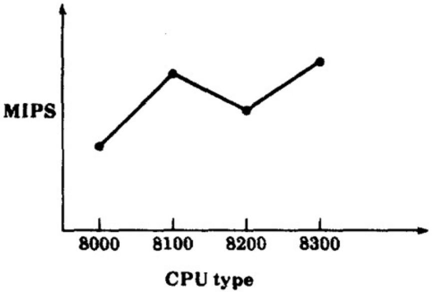
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



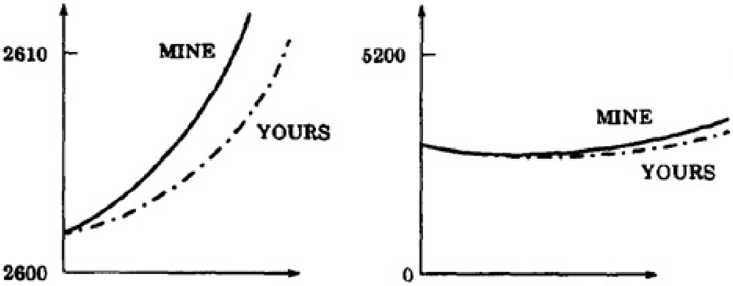
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



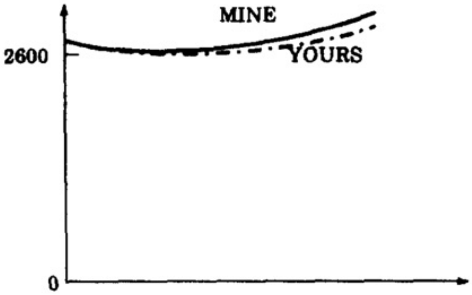
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



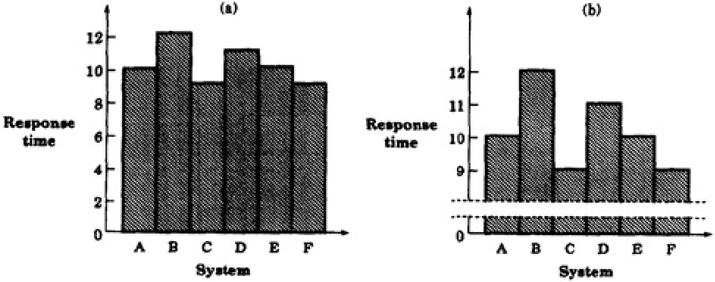
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



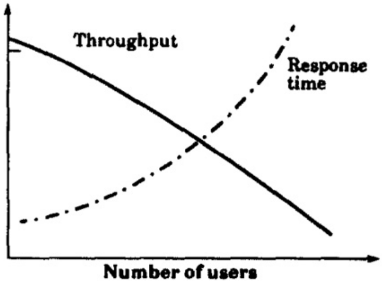
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



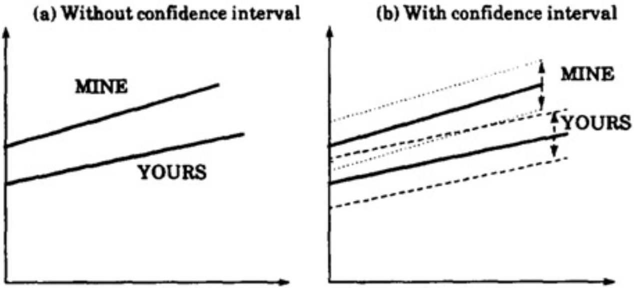
Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)

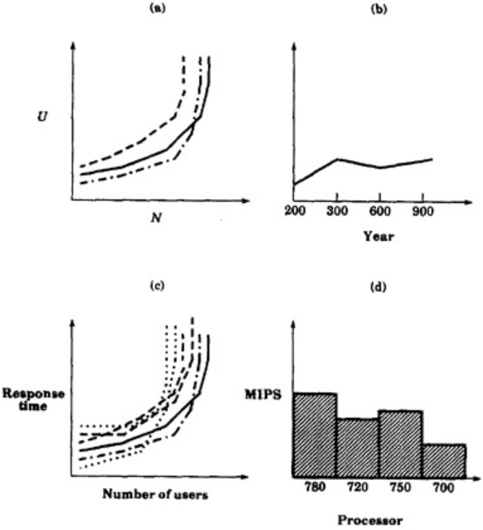


Guidelines

1. Require minimum effort to the reader: get the message (legends, labels, trends, annotations, . . .)
2. Maximize information (self-sufficient, clear labels, units, . . .)
3. Minimize Ink (avoid cluttered information. . .)
4. Use commonly accepted practices (axis, effect along the y-axis, scales)
5. Avoid Ambiguity (coordinates, scales, colors, only one variable, . . .)



What about these ones ?



Use the right tools

R is a system for statistical computation and graphics.

- Avoid programming with R. Most things can be done with one liners.
- Excellent graphic support with **ggplot2**.
- `knitr` allows to mix R with LaTeX or Markdown. Litterate programming to ease reproducible research.

Rstudio is an IDE a system for statistical computation and graphics. It is easy to use and allows publishing on **rpubs**.

Org-mode Allows to mix sh, perl, R, . . . within plain text documents and export to LaTeX, HTML, . . .

Outline

Why R?

R is a great language for data analysis and statistics

- Open-source and multi-platform
- Very expressive with high-level constructs
- Excellent graphics
- Widely used in academia and business
- Very active community
  - Documentation, FAQ on http://stackoverflow.com/questions/tagged/r
- Great integration with other tools

Why is R a pain for computer scientists?

- R is not really a programming language
- Documentation is for statisticians
- Default plots are ~~cumbersome~~ (meaningful)
- Summaries are ~~cryptic~~ (precise)
- Steep learning curve even for us, computer scientists whereas we generally switch seamlessly from a language to another! That's frustrating! ;)

Do's and dont's

~~R is high level, I'll do everything myself~~

- CTAN comprises 4,334 TEX, LATEX, and related packages and tools. Most of you do not use plain TEX.
- Currently, the CRAN package repository features 4,030 available packages.
- How do you know which one to use??? Many of them are highly exotic (not to say useless to you).

  I learnt with http://www.r-bloggers.com/

- Lots of introductions but not necessarily what you're looking for so I'll give you a short tour.

  You should quickly realize though that you need proper training in statistics and data analysis if you do not want tell nonsense.
- Again, you should read Jain's book on The Art of Computer Systems Performance Analysis
- You may want to follow online courses:
  - https://www.coursera.org/course/compdata
  - https://www.coursera.org/course/repdata

Install and run R on debian

```
apt-cache search r
```

Err, that's not very useful :) It's the same when searching on google but once the filter bubble is set up, it gets better...

```
sudo apt-get install r-base
```

```
R
```

```
R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>
```

Install a few cool packages

R has it's own package management mechanism so just run R and type the following commands:

- ddply, reshape and ggplot2 by Hadley Wickham (http://had.co.nz/)

```r
install.packages("plyr")
install.packages("reshape")
install.packages("ggplot2")
```

- knitR by (Yihui Xie) http://yihui.name/knitr/

```r
install.packages("knitr")
```

IDE

Using R interactively is nice but quickly becomes painful so at some point, you'll want an IDE.

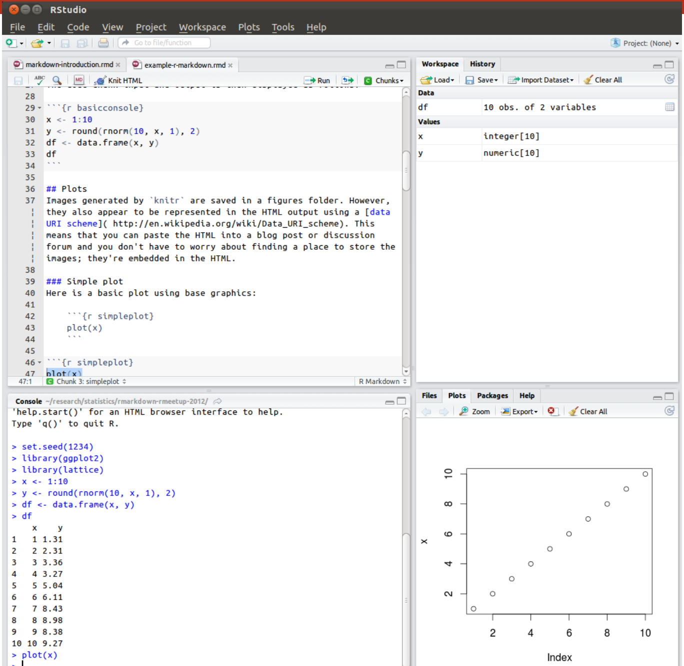
Emacs is great but you'll need *Emacs Speaks Statistics*

```
sudo apt-get install ess
```

In this tutorial, I will briefly show you rstudio (https://www.rstudio.com/) and later how to use `org-mode`

Outline

Rstudio screenshot



Reproducible analysis in Markdown + R

- Create a new R Markdown document (Rmd) in rstudio
- R chunks are interspersed with ```{r} and ```
- Inline R code: `r sin(2+2)`
- You can knit the document and share it via rpubs
- R chunks can be sent to the top-level with `Alt-Ctrl-c`
- I usually work mostly with the current environment and only knit in the end
- Other engines can be used (use rstudio completion)

````
```{r engine='sh'}
ls /tmp/
```
````

- Makes reproducible analysis as simple as one click
- Great tool for quick analysis for self and colleagues, homeworks, ...

Reproducible articles with LaTeX + R

- Create a new R Sweave document (Rnw) in rstudio
- R chunks are interspersed with <<>>= and @
- You can knit the document to produce a pdf
- You'll probably quickly want to change default behavior (activate the cache, hide code, ...). In the preembule:

```
<<echo=FALSE>>=
opts_chunk$set(cache=TRUE,dpi=300,echo=FALSE,fig.width=7,
                    warning=FALSE,message=FALSE)
@
```

- Great for journal articles, theses, books, ...

Outline

Data frames

A data frame is a data tables (with columns and rows). `mtcars` is a built-in data frame that we will use in the sequel

```
head(mtcars);
```

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear car
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3
```

You can also load a data frame from a CSV file:

```
df <- read.csv("http://foo.org/mydata.csv", header=T,
       strip.white=TRUE);
```

You will get help by using ?:

```
?data.frame
?rbind
?cbind
```

Exploring Content (1)

```
names(mtcars);
```

```
 [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"
[11] "carb"
```

```
str(mtcars);
```

```
'data.frame': 32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
 $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

Exploring Content (2)

```
dim(mtcars);
length(mtcars);
```

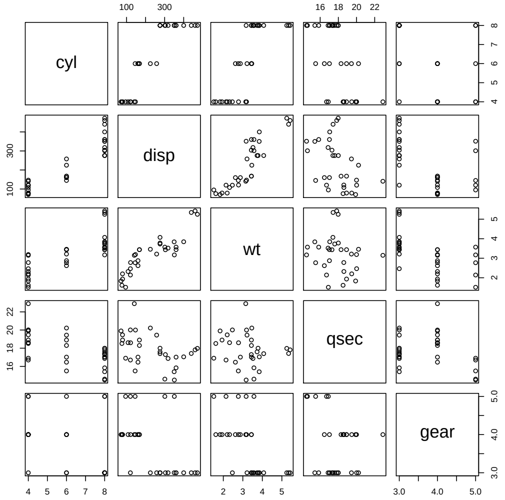
```
[1] 32 11
[1] 11
```

```
summary(mtcars);
```

```
      mpg             cyl             disp             hp
 Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0
 1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
 Median :19.20   Median :6.000   Median :196.3   Median :123.0
 Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7
 3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
 Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0
      drat             wt             qsec             vs
 Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000
 1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
 Median :3.695   Median :3.325   Median :17.71   Median :0.0000
 Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375
 3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000
```

Exploring Content (3)

```
plot(mtcars[names(mtcars) %in% c("cyl","wt","disp","qsec","gear")])
```



Accessing Content

```
mtcars$mpg
```

```
 [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17
[16] 10.4 14.7 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30
[31] 15.0 21.4
```

```
mtcars[2:5,]$mpg
```

```
[1] 21.0 22.8 21.4 18.7
```

```
mtcars[mtcars$mpg == 21.0,]
```

```
              mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4      21   6  160 110  3.9 2.620 16.46  0  1    4    4
Mazda RX4 Wag  21   6  160 110  3.9 2.875 17.02  0  1    4    4
```

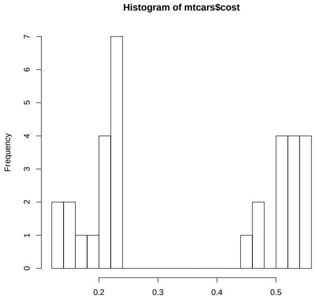
```
mtcars[mtcars$mpg == 21.0 & mtcars$wt > 2.7,]
```

```
              mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4 Wag  21   6  160 110  3.9 2.875 17.02  0  1    4    4
```

```
mtcars$cost = log(mtcars$hp)*atan(mtcars$disp)/
 sqrt(mtcars$gear**5);
mean(mtcars$cost);
summary(mtcars$cost);
```

```
[1] 0.345994
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.1261  0.2038  0.2353  0.3460  0.5202  0.5534
```

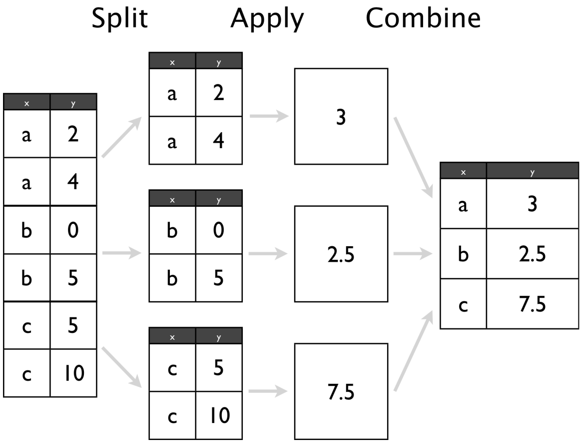
```
hist(mtcars$cost,breaks=20);
```



Outline

plyr: the Split-Apply-Combine Strategy

Have a look at http://plyr.had.co.nz/09-user/ for a more detailed introduction.



plyr: Powerfull One-liners

```
library(plyr)
mtcars_summarized = ddply(mtcars,c("cyl","carb"), summarize,
      num = length(wt), wt_mean = mean(wt), wt_sd = sd(wt),
      qsec_mean = mean(qsec), qsec_sd = sd(qsec));
mtcars_summarized
```

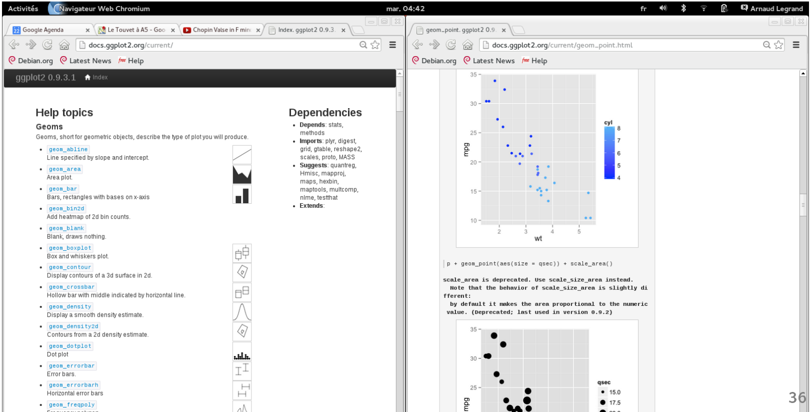
```
  cyl carb num  wt_mean     wt_sd qsec_mean   qsec_sd
1   4    1   5 2.151000 0.2627118  19.37800 0.6121029
2   4    2   6 2.398000 0.7485412  18.93667 2.2924368
3   6    1   2 3.337500 0.1732412  19.83000 0.5515433
4   6    4   4 3.093750 0.4131460  17.67000 1.1249296
5   6    6   1 2.770000        NA  15.50000        NA
6   8    2   4 3.560000 0.1939502  17.06000 0.1783255
7   8    3   3 3.860000 0.1835756  17.66667 0.3055050
8   8    4   6 4.433167 1.0171431  16.49500 1.4424112
9   8    8   1 3.570000        NA  14.60000        NA
```

If your data is not in the right form give a try to reshapeP/melt.

plyr next generation = dplyr

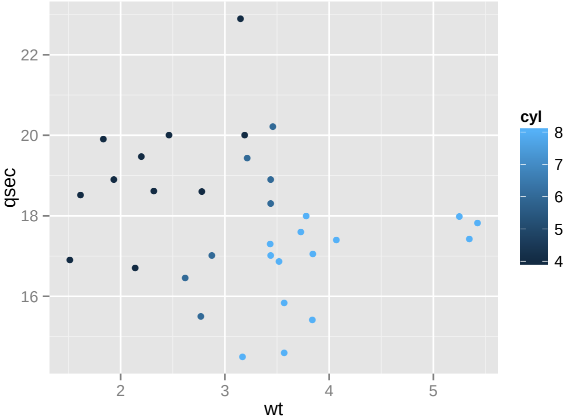
ggplot2: Modularity in Action

- ggplot2 builds on plyr and on a modular grammar of graphics
- ~~obnoxious function with dozens of arguments~~
- combine small functions using layers and transformations
- aesthetic mapping between observation characteristics (data frame column names) and graphical object variables
- an incredible documentation: http://docs.ggplot2.org/current/



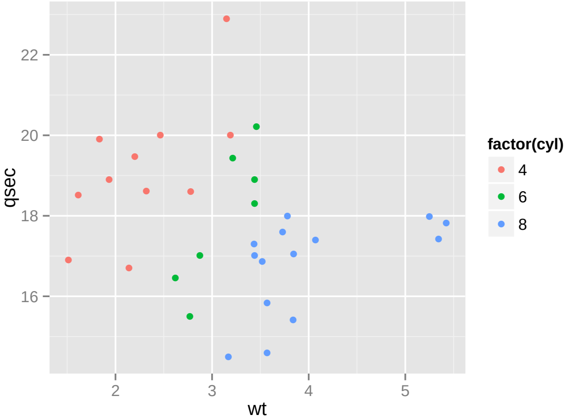
ggplot2: Illustration (1)

```
ggplot(data = mtcars, aes(x=wt, y=qsec, color=cyl)) +
        geom_point();
```



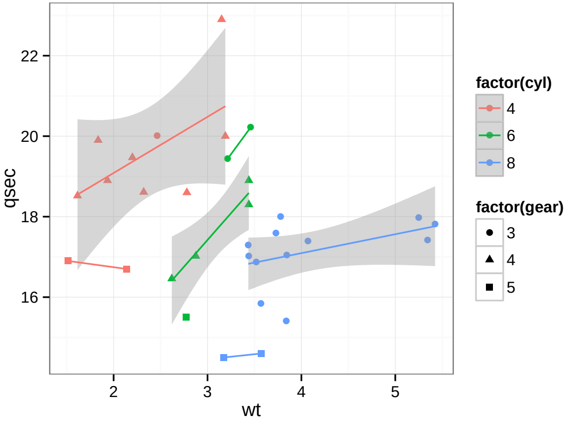
ggplot2: Illustration (2)

```
ggplot(data = mtcars, aes(x=wt, y=qsec, color=factor(cyl))) +
      geom_point();
```



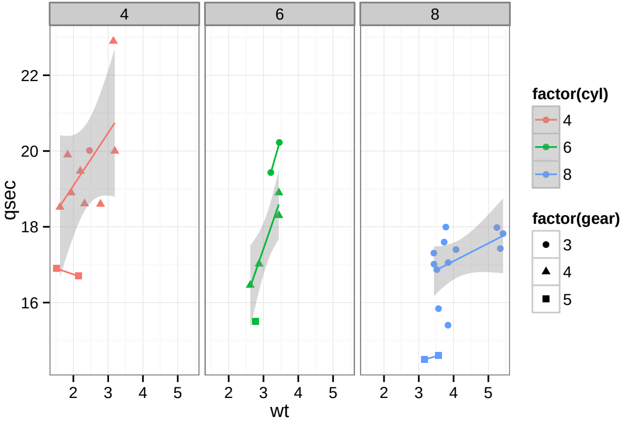
ggplot2: Illustration (3)

```
ggplot(data = mtcars, aes(x=wt, y=qsec, color=factor(cyl),
        shape = factor(gear))) +  geom_point() + theme_bw() +
        geom_smooth(method="lm");
```



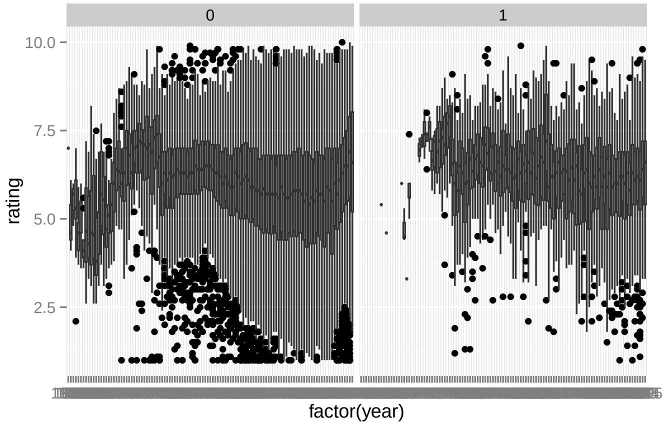
ggplot2: Illustration (4)

```
ggplot(data = mtcars, aes(x=wt, y=qsec, color=factor(cyl),
        shape = factor(gear))) + geom_point() + theme_bw() +
        geom_smooth(method="lm") + facet_wrap(~ cyl);
```



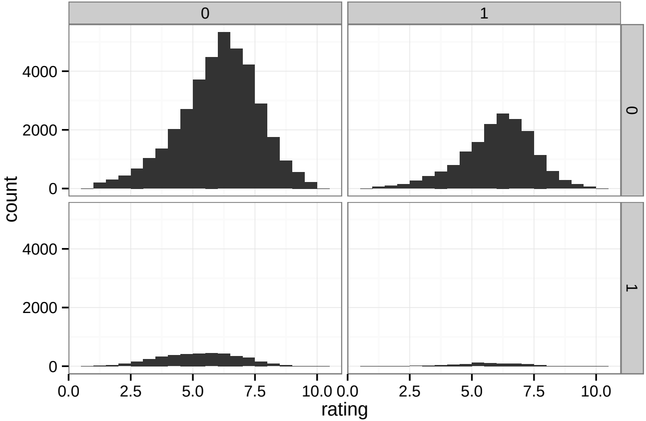
ggplot2: Illustration (5)

```
ggplot(data = movies, aes(x=factor(year),y=rating)) +
        geom_boxplot() + facet_wrap(~Romance)
```



ggplot2: Illustration (6)

```
ggplot(movies, aes(x = rating)) + geom_histogram(binwidth = 0.5)+
        facet_grid(Action ~ Comedy) + theme_bw();
```



Take away Message

- R is a great tool but is only a tool. There is no magic. You need to understand what you are doing and get a minimal training in statistics
- It is one of the building block of reproducible research (the *reproducible analysis* block) and will save you a lot of time
- Read at least Jain's book: The Art of Computer Systems Performance Analysis