Open Reproducible Research in Empirical Science

Olivier Flores

UMR PVBMT, Université de La Réunion / CIRAD

Interests and motivation. I

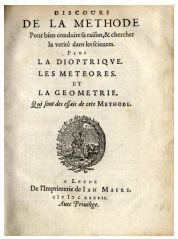
Wishful thinking or reality ?

Reproducibility

One of the main principles of the scientific method, refers to the ability of a test or experiment to be accurately reproduced, or replicated, by someone else working independently.

(Wikipedia)

Notion present in *Discours de la Méthode* (Descartes, 1637)


DISCOURS
DE LA METHODE
Pour bien conduire sa raison, & chercher la verité dans les Sciences.
Plus
LA DIOPTRIQVE.
LES METEORES.
ET
LA GEOMETRIE.
Qui sont des essais de cette METHODE.
A LEYDE
De l'Imprimerie de IAN MAIRE.
CIↃ IↃ CXXXVII.
Auec Priuilege.

Interests and motivation. II

Wishful thinking or reality ?

- "*Generating* **verifiable knowledge** *has long been* **scientific discovery's central goal**, *yet today it's impossible to verify most of the [computational] results that scientists present at conferences and in papers.*"
- About scientific articles: "*There is a leap of faith required by the reader; they must believe that the transformations and model fitting were done appropriately and without error.*"

NEWS

REPRODUCIBLE RESEARCH

ADDRESSING THE NEED FOR DATA AND CODE SHARING IN COMPUTATIONAL SCIENCE

By the Yale Law School Roundtable on Data and Code Sharing

Roundtable participants identified ways of making computational research details readily available, which is a crucial step in addressing the current credibility crisis.

Bioconductor Project
Bioconductor Project Working Papers

Year 2004 *Paper* 2

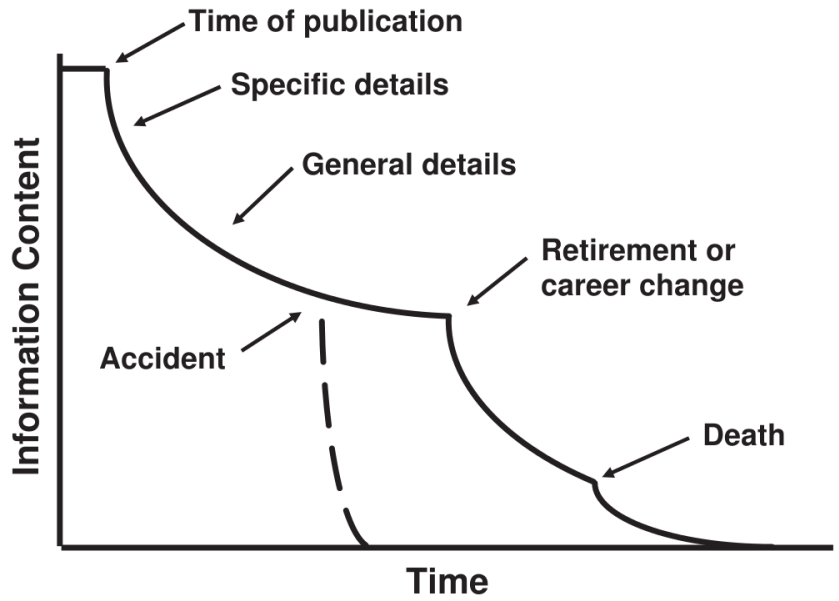
Statistical Analyses and Reproducible Research

Robert Gentleman* Duncan Temple Lang†

Interests and motivation. III

Wishful thinking or reality ?

- Essential in principle, difficult in general,
- Energy- and time-saving on a daily basis: to share and communicate with collaborators, colleagues, students,...



ECOLOGICAL INFORMATICS 1 (2006) 3–7

available at www.sciencedirect.com

SCIENCE DIRECT®

www.elsevier.com/locate/ecolinf

ELSEVIER

Meta-information concepts for ecological data management

William K. Michener

University of New Mexico, Department of Biology, Albuquerque, NM 87131-0001, United States

Interests and motivation. IV

Wishful thinking or reality ?

Objective

Propose a simple workflow based on open tools to easily share research results, if not with the World, at least within a collaborative group.

The basis: a distributable and executable unit

Components

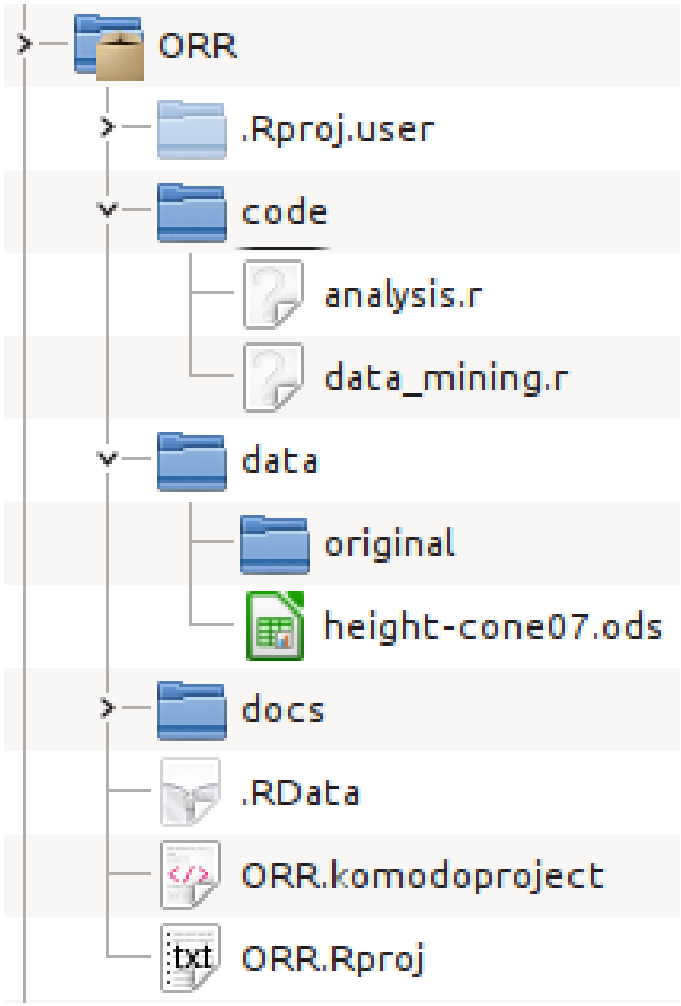
- Compendium: a special form of knowledge that "combines **text**, **data** and auxiliary software (**code**) into a *distributable* and *executable* unit"
- Dynamic documents:
  - the three unitary elements that can be "extracted and processed in various different ways by both the author and the reader"
  - Sequence of *text chunks* and *code chunks*
  - **Text** chunks: description for reading purpose
  - **Code** chunks: sequence of commands to be interpreted by general purpose software
  - **General purpose** software (R, Perl, ...) and **auxiliary** software (user's code)
  - Relations between chunks, not necessarily linear

The box contents

Folder structure

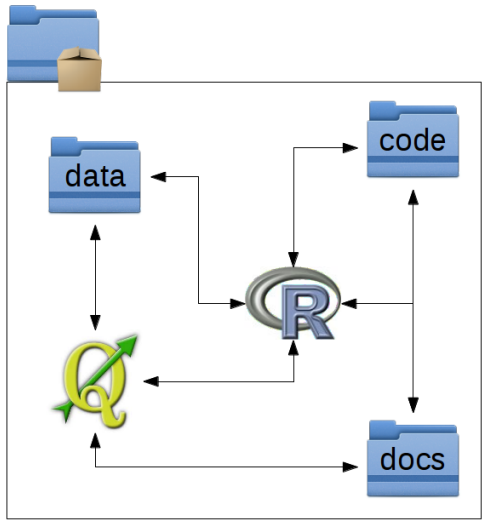
Box = Working folder

- Three primary subfolders: **code**, **data**, **text**
- Simple names
- Avoid multiple versions (versioning system)
- A simple (but helpful) basis one can easily develop !



Ineracting elements
A dynamic view

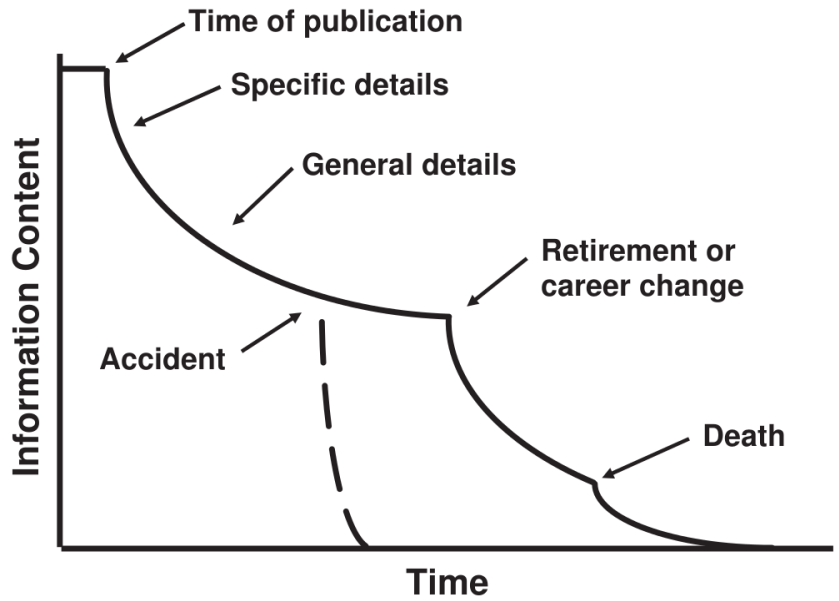
- Data first!
- All elements interact
- Interface between software and languages
(Python, Perl, C,. . .)



Handling, Analysis and Mining I

Format and import

- Keep copy of the original
- Working version ready for import
- Rectangular and simple
- Keep track (metadata)



ECOLOGICAL INFORMATICS 1 (2006) 3–7

available at www.sciencedirect.com

SCIENCE DIRECT®

www.elsevier.com/locate/ecolinf

ELSEVIER

Meta-information concepts for ecological data management

William K. Michener

University of New Mexico, Department of Biology, Albuquerque, NM 87131-0001, United States

Handling, Analysis and Mining II

Format and import

- Import working version **ROpenOffice**
- Transorm data

```
library(ROpenOffice)
tmp<-read.ods('../data/height-cone07.ods')
```

```
class(tmp)

[1] "list"

length(tmp)

[1] 6

head(tmp$Llaureolum)

  species height cones
1      Ll    150    96
2      Ll    190   270
3      Ll    120    20
4      Ll     80    13
5      Ll    120    11
6      Ll    100    10
```

Data transformation and visualizaion

```
dat<-do.call('rbind',tmp)
summary(dat)
```

```
##  species     height          cones
##  Ll:64   Min.   : 20   Min.   :   1.0
##  S :36   1st Qu.: 80   1st Qu.:  15.0
##  X :45   Median :120   Median :  32.5
##  R :47   Mean   :119   Mean   :  66.3
##  Lp:52   3rd Qu.:150   3rd Qu.:  73.8
##  C :34   Max.   :400   Max.   :1000.0
```

- Functions **melt** and **cast**: easy conversion between long and wide formats
- Data transformation: **reshape**, **plyr**,
- Data mining

```
library(plyr)
tab<-ddply(dat,.(species),.fun=function(df)
                 with(df,mean(cones^2/height)))
tab
```

```
  species      V1
1      Ll  34.704
2       S 206.706
3       X 147.961
4       R  81.245
5      Lp  41.191
6       C   3.548
```

Grammar of graphics

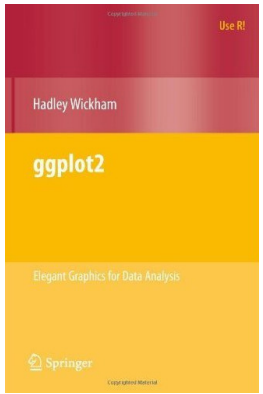
- **Mapping** graphical elements on variables
  $\neq$ setting elements by user (shape, size, colour,...)
- Multiple layer plots
- Flexible control on layers

The usual way

```
plot(x=cones,y=height,data=data,
pch=as.numeric(factor(dat[,'species'])))
```

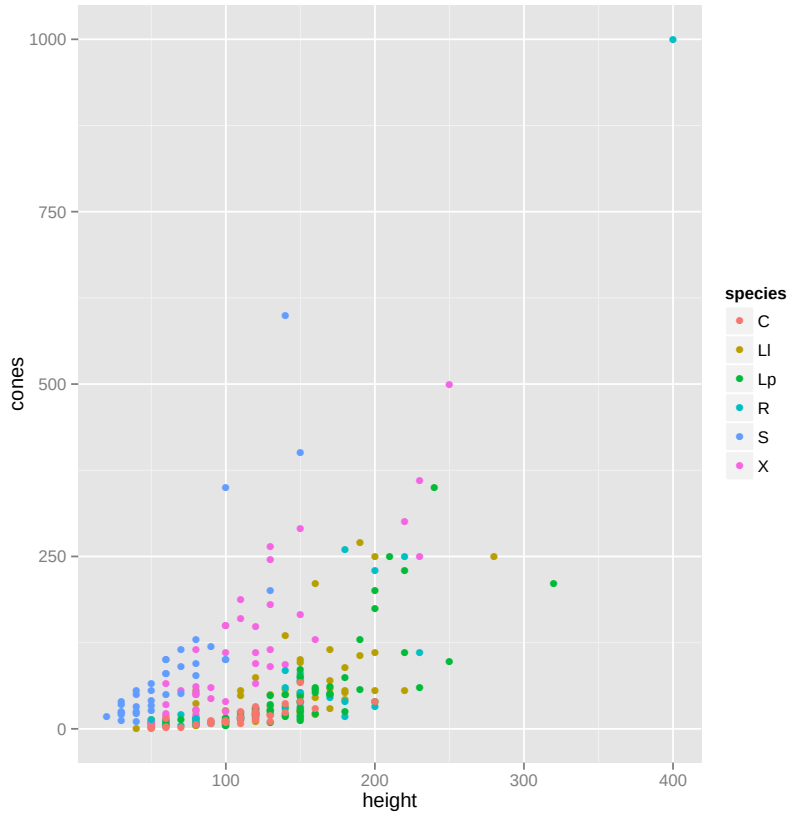
Grammar of graphics

- **Mapping** graphical elements on variables
  $\neq$ setting elements by user (shape, size, colour,...)
- Multiple layer plots
- Flexible control on layers



A simple example

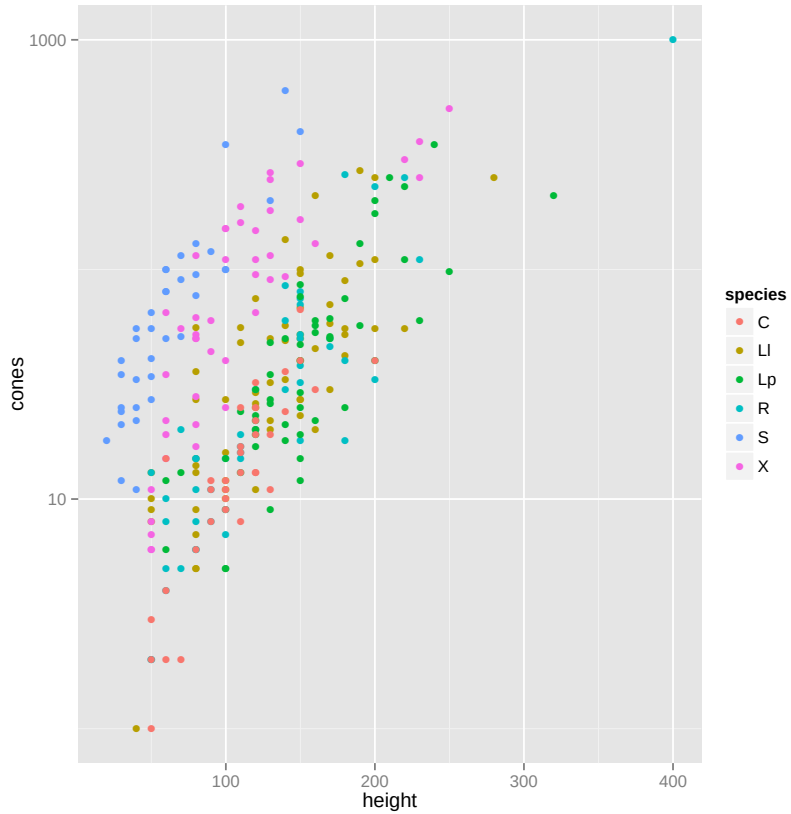
```
library(ggplot2)
qplot(data=dat,x=height,y=cones,
colour=species)
```



Grammar of graphics

- **Mapping** graphical elements on variables
  $\neq$ setting elements by user (shape, size, colour,...)
- Multiple layer plots
- Flexible control on layers

Change y-scale to logarithmic:

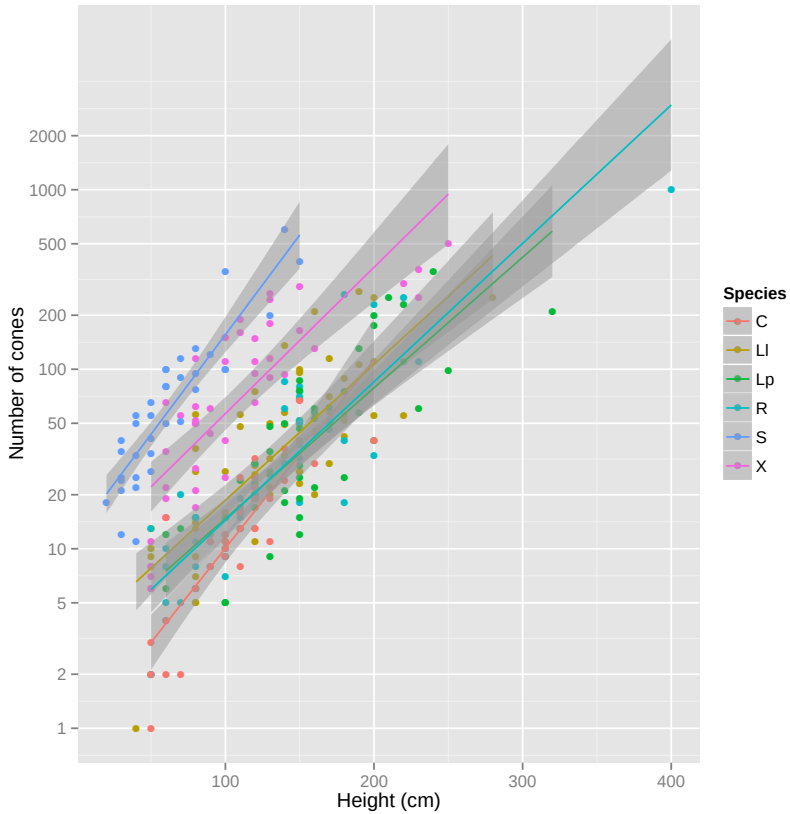


Grammar of graphics

- **Mapping** graphical elements on variables
  $\neq$ setting elements by user (shape, size, colour,...)
- Multiple layer plots
- Flexible control on layers

```
p<- ggplot(data = dat, aes(x = height, y =
cones, colour = species)) +
geom_point() +
geom_smooth(method ='lm', aes(group =
species), alpha = 0.5) +
scale_y_log10(breaks =
c(1,2,5,10,20,50,100,200,500,1000,2000)) +
labs(x = 'Height (cm)',y = 'Number of
cones',colour = 'Species')
```

Slightly more complex

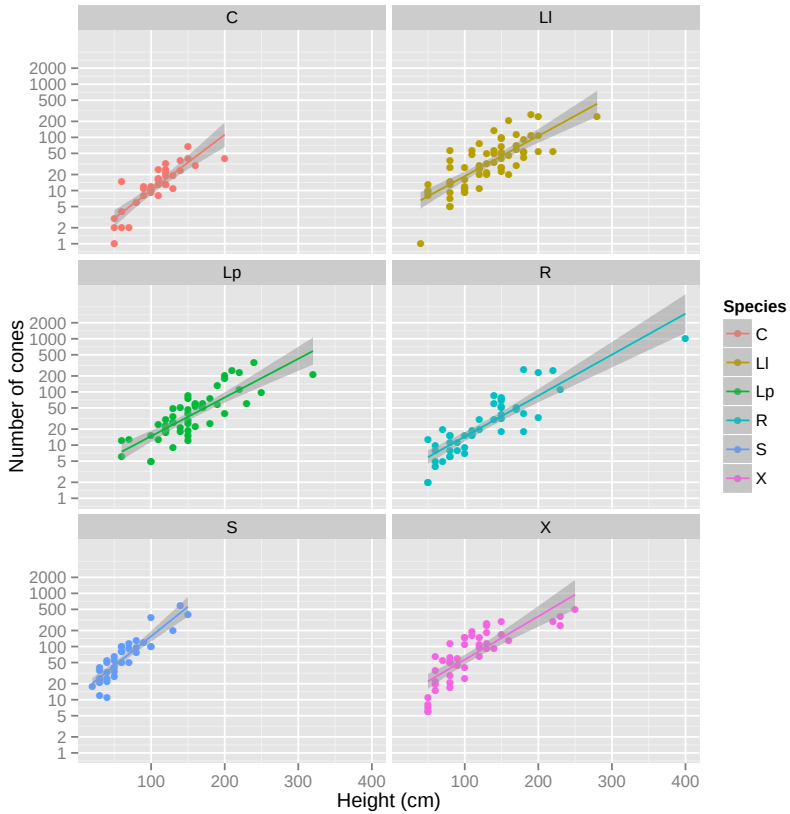


Grammar of graphics

- **Mapping** graphical elements on variables
  $\neq$ setting elements by user (shape, size, colour,...)
- Multiple layer plots
- Flexible control on layers

p + facet_wrap(~species,ncol=2)

Complex settings made easy

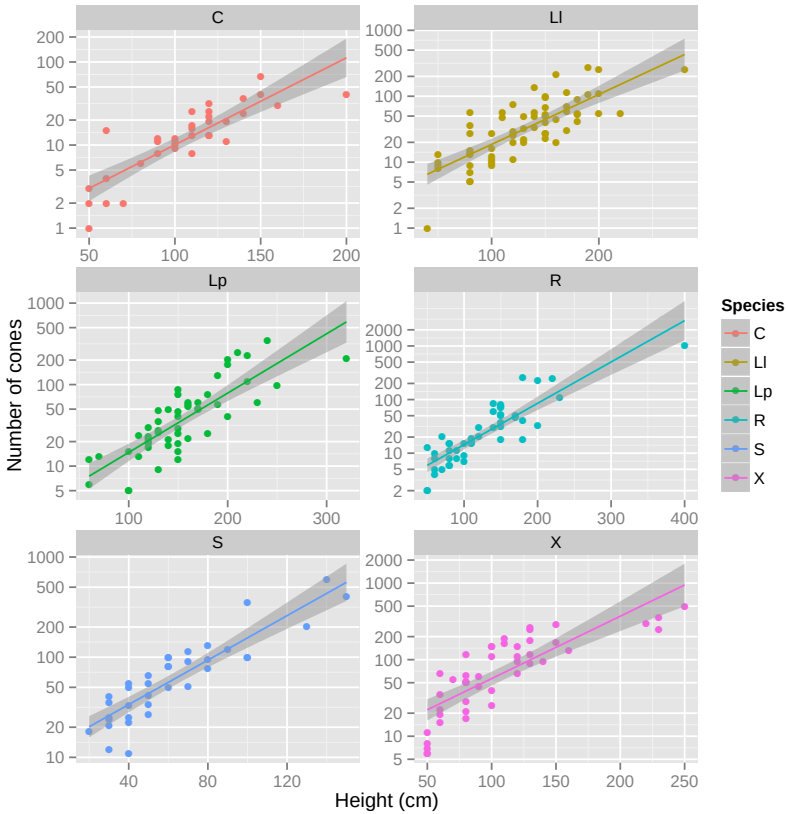


Grammar of graphics

- **Mapping** graphical elements on variables
  $\neq$ setting elements by user (shape, size, colour,...)
- Multiple layer plots
- Flexible control on layers

p + facet_wrap(~species,ncol = 2,scales = 'free')

Free scales

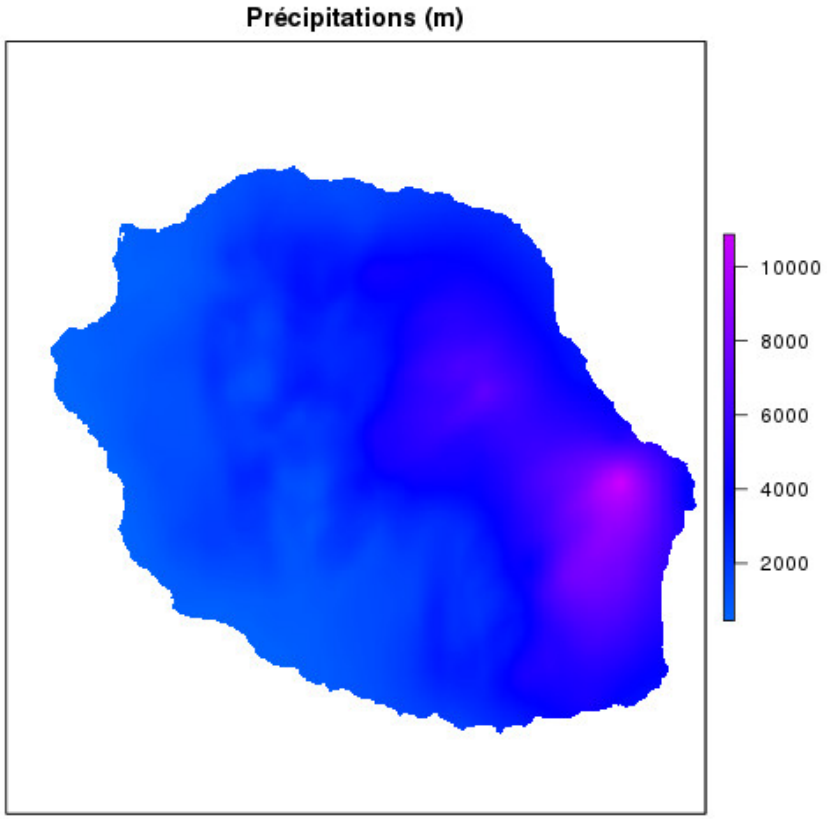


Integration. I
Example in GIS world

- **Open tools**
- Example in Geographic Information Systems (GIS)
- QGIS, GRASS, SAGA for analysis
- GDAL for format exchange and import
- Interface packages with R: **rgdal**, **RSAGA**, **rGrass**,
- Tools: **maptools**, **raster**, **rgeos**, **sp**, **spatial**,

Integration. II
Example in GIS world

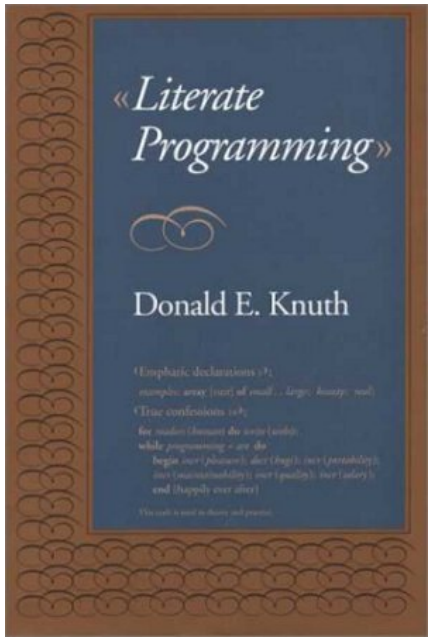
```
library(raster)
## read file names
RR<-dir(path='data/GIS/precip',
        full.names=TRUE)
## import files with GDAL and stack data
RR<-stack(RR)
## define Coordinate system
proj4string(RR)<-CRS('+init=epsg:2975')
## format data and sum
RR[RR>1e38]<-NA
RRm<-sum(RR)
```



Production
Literate programing

The key to reproducibility !!

- Literate programming
- "Instead of imagining that our task is to instruct a computer what to do, let us concentrate on explaining to human beings what we want a computer to do." (D. Knuth, 1984)
- Ordered mixture of **text** and **code** chunks
- Code chunks: produce figures and tables based on data
- Text chunks: explains the methods and results



Donald E. Knuth

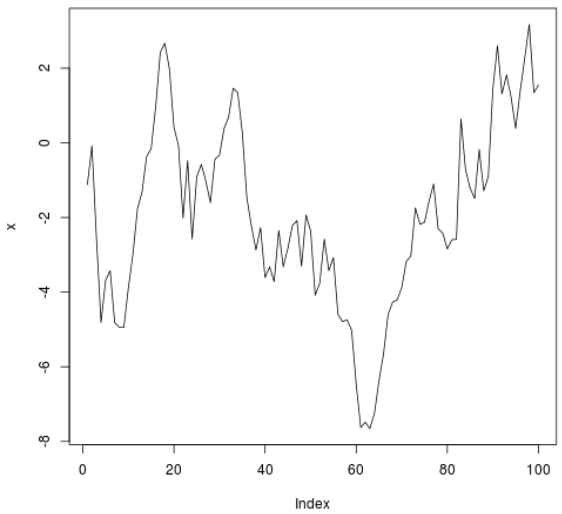
Production
Handling dynamic documents

- **Weaving**: process dynamics documents to produce human-readable versions in various formats (PDF, HTML, ODF,...)
- **Tangling**: extract code chunks from dynamic document,
- Multiple alternatives

Document format	(La)TeX, Markdown, ODF
Weaver/Tangler	**Sweave**, **knitr**, **pander**, **rmarddown**
Software	R, Perl, Python, C
Format	PDF, HTML, ODF

The Tex approach

- Benefits from the power of TeX to produce complex documents
- Rather steep learning curve...




```
Text without code, may contain inline commands,
to get the value of $\pi$ for instance: \Sexpr{pi}\\
Code chunk producing a plot :
<<label,eval=F,dev='png',size='scriptsize'>>=
set.seed(1213)  # for reproducibility
x <- cumsum(rnorm(100))
mean(x)  # mean of x
plot(x, type = 'l')  # Brownian motion
@
```

Text without code, may contain inline commands, to get the value of π for instance: 3.1416
Code chunk producing a plot :

```
set.seed(1213)  # for reproducibility
x <- cumsum(rnorm(100))
mean(x)  # mean of x
plot(x, type = 'l')  # Brownian motion
```

The markdown approach

- Simple mark-up language,
- Multiple output formats (PDF, HTML)
- Simple language for simple document (for now),

```
library(pander)
pander(tab)
```

species	V1
Ll	34.7
S	206.7
X	148
R	81.25
Lp	41.19
C	3.548

Thank you for your attention!

Questions, remarks ?