R programming for beginners

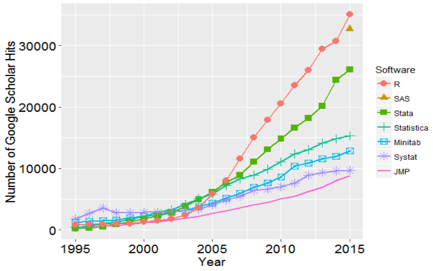
Ni Shuai

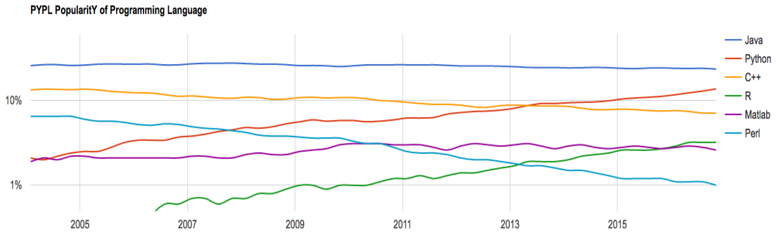
Computational Genome Biology
German Cancer Research Center (DKFZ)

November, 2016

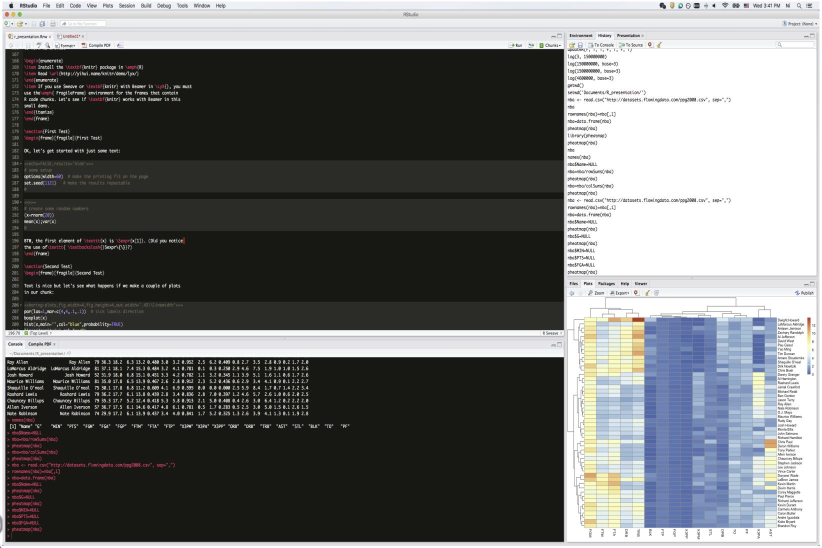
Why use R

- Interactive, analysis your data on the fly
- Free & Open source, a strong community
- 8000+ R packages for various methods, still growing





Take control of your R code - Rstudio



```
\begin{enumerate}
\item Install the \textbf{knitr} package in \emph{R}
\item Read \url{http://yihui.name/knitr/demo/lyx/}
\end{enumerate}
\item If you use Sweave or \textbf{knitr} with Beamer in \LyX{}, you must
use the\emph{ FragileFrame} environment for the frames that contain
R code chunks. Let's see if \textbf{knitr} works with Beamer in this
small demo.
\end{itemize}
\end{frame}

\section{First Test}
\begin{frame}[fragile]{First Test}

OK, let's get started with just some text:

<<echo=FALSE,results='hide'>>=
# some setup
options(width=60)  # make the printing fit on the page
set.seed(1121)   # make the results repeatable
@

<<>>=
# create some random numbers
(x=rnorm(20))
mean(x);var(x)
@

BTW, the first element of \texttt{x} is \Sexpr{x[1]}. (Did you notice
the use of\texttt{ \textbackslash{}Sexpr\{\}}?)
\end{frame}

\section{Second Test}
\begin{frame}[fragile]{Second Test}

Text is nice but let's see what happens if we make a couple of plots
in our chunk:

<<boring-plots,fig.width=4,fig.height=4,out.width='.45\\linewidth'>>=
par(las=1,mar=c(4,4,.1,.1))  # tick labels direction
boxplot(x)
hist(x,main='',col="blue",probability=TRUE)
```

```
log(3, 150000000)
log(150000000, base=3)
log(1500000000, base=3)
log(4600000, base=3)
getwd()
setwd('Documents/R_presentation/')
nba <- read.csv("http://datasets.flowingdata.com/ppg2008.csv", sep=",")
nba
rownames(nba)=nba[,1]
nba=data.frame(nba)
pheatmap(nba)
library(pheatmap)
pheatmap(nba)
nba
names(nba)
nba$Name=NULL
nba=nba/rowSums(nba)
pheatmap(nba)
nba=nba/colSums(nba)
pheatmap(nba)
nba <- read.csv("http://datasets.flowingdata.com/ppg2008.csv", sep=",")
rownames(nba)=nba[,1]
nba=data.frame(nba)
nba$Name=NULL
pheatmap(nba)
nba$G=NULL
pheatmap(nba)
nba$MIN=NULL
nba$PTS=NULL
nba$FGA=NULL
pheatmap(nba)
```

```
~/Documents/R_presentation/
Ray Allen                  Ray Allen  79 36.3 18.2  6.3 13.2 0.480 3.0  3.2 0.952  2.5  6.2 0.409 0.8 2.7  3.5  2.8 0.9 0.2 1.7 2.0
LaMarcus Aldridge  LaMarcus Aldridge  81 37.1 18.1  7.4 15.3 0.484 3.2  4.1 0.781  0.1  0.3 0.250 2.9 4.6  7.5  1.9 1.0 1.0 1.5 2.6
Josh Howard              Josh Howard  52 31.9 18.0  6.8 15.1 0.451 3.3  4.2 0.782  1.1  3.2 0.345 1.1 3.9  5.1  1.6 1.1 0.6 1.7 2.6
Maurice Williams    Maurice Williams  81 35.0 17.8  6.5 13.9 0.467 2.6  2.8 0.912  2.3  5.2 0.436 0.6 2.9  3.4  4.1 0.9 0.1 2.2 2.7
Shaquille O'neal    Shaquille O'neal  75 30.1 17.8  6.8 11.2 0.609 4.1  6.9 0.595  0.0  0.0 0.000 2.5 5.9  8.4  1.7 0.7 1.4 2.2 3.4
Rashard Lewis          Rashard Lewis  79 36.2 17.7  6.1 13.8 0.439 2.8  3.4 0.836  2.8  7.0 0.397 1.2 4.6  5.7  2.6 1.0 0.6 2.0 2.5
Chauncey Billups    Chauncey Billups  79 35.3 17.7  5.2 12.4 0.418 5.3  5.8 0.913  2.1  5.0 0.408 0.4 2.6  3.0  6.4 1.2 0.2 2.2 2.0
Allen Iverson          Allen Iverson  57 36.7 17.5  6.1 14.6 0.417 4.8  6.1 0.781  0.5  1.7 0.283 0.5 2.5  3.0  5.0 1.5 0.1 2.6 1.5
Nate Robinson          Nate Robinson  74 29.9 17.2  6.1 13.9 0.437 3.4  4.0 0.841  1.7  5.2 0.325 1.3 2.6  3.9  4.1 1.3 0.1 1.9 2.8
> names(nba)
 [1] "Name" "G"    "MIN"  "PTS"  "FGM"  "FGA"  "FGP"  "FTM"  "FTA"  "FTP"  "X3PM" "X3PA" "X3PP" "ORB"  "DRB"  "TRB"  "AST"  "STL"  "BLK"  "TO"   "PF"
> nba$Name=NULL
> nba=nba/rowSums(nba)
> pheatmap(nba)
> nba=nba/colSums(nba)
> pheatmap(nba)
> nba <- read.csv("http://datasets.flowingdata.com/ppg2008.csv", sep=",")
> rownames(nba)=nba[,1]
> nba=data.frame(nba)
> nba$Name=NULL
> pheatmap(nba)
> nba$G=NULL
> pheatmap(nba)
> nba$MIN=NULL
> nba$PTS=NULL
> nba$FGA=NULL
> pheatmap(nba)
>
```

Contents

History

- R is an implementation of the S programming language which is created by John Chambers while at Bell Labs, it was first created for teaching purpose, because of the high fees for SâĂȘPLUS licenses.
- R is names after its two authors ***Ross Ihaka*** and ***Robert Gentlemar*** who cheated it at the University of Auckland, and also because ... it's close to S.
- The stable beta verson of R is released in 2000, it's currently developed by the *R Develop Core Team*, of which Chambers is a member.

R as a calculator

Lets try it in R!

```
3+3      # This starts a comment (where R recoginze but ignores them)
2*8
2^10     # 2 to the power of 10
0/Inf
0/0      # the expression has no meaning
log(10)
sqrt(3) # the square root of 3
sin(pi) # R kinda knows pi already
```

Basic calculations

- PEMDAS (Please Excuse My Dear Aunt Sally) rule applies, i.e.: Parentheses, Exponents, Multiplication & Division, Addition & Subtraction.
- Just use additional parentheses to clarify evaluation order!

Basic calculations

Operators	+, -, *, /, ^
Integer division, modulo	%/%, %%
Extremes	max(), min(), range()
Square root	sqrt()
Rounding	round(), floor(), ceiling()
trigonometric functions	sin(), cos(), tan(), asin(), acos(), atan()
Logarithms	log(), log10(), log2(), exp()
Sum, product	sum(), prod()
π	pi
Infinity	Inf, -Inf (infinity)
Not defined	NaN (Not a number)
Missing values	NA (Not available)
Empty set	NULL

How can I get help?

help.start()	Start the help system in a browser
help(something)	Get help about something
?(something)	does the same as help()
apropos('foo')	list all functions containing string 'foo'
example(foo)	show an example of function 'foo'

(Examples:)

help.start()	The browser is open
?abs	Help on abs()
apropos("help")	Is something similar to help()?
example(min)	shows an example of function min, which returns the minimum of all the values present in their arguments

Assignment

X	<-	10
Variable		Value

Lets try to calculate something out of x:

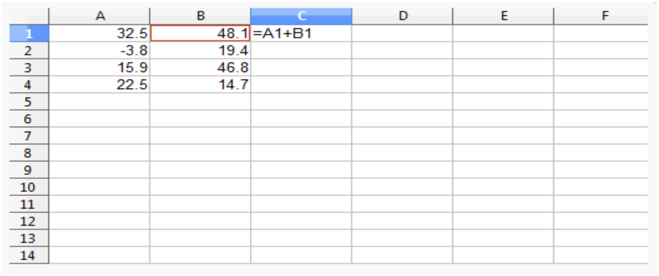
```
x*3

## [1] 30

100-x

## [1] 90

y = x+10
```

Conclusions:

- Insert blanks in order to improve readability.
- Variable names should not start with a number!

Vectorize your thinking: doing things the "R" way

Task: Adding two columns in a spreadsheet.



	A	B	C	D	E	F
1	32.5	48.1	=A1+B1			
2	-3.8	19.4				
3	15.9	46.8				
4	22.5	14.7				
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						

It is much easier to do the same thing in R, we just need to add them up like they are two numbers.

Vectorize your thinking: doing things the "R" way

One of the most useful features in R is vectors, a vector is a sequence of data of the same type.

```
a=c(1,2,3,4,5,6)

a*3

## [1]  3  6  9 12 15 18

100-a

## [1] 99 98 97 96 95 94

b=a+10
b

## [1] 11 12 13 14 15 16
```

Logical operations

Comparisons	==, !=, >, <, >=, <=
Constants	TRUE, FALSE
test if X is TRUE	isTRUE(X)
Operations	! (negation)
	xor (exclusive or)
	&, &&, |, || (and, or)

Examples:

4<3		FALSE
(3 + 1) != 3		TRUE
(3 >= 2) & (4 == (3 + 1))		TRUE
-3<-2	wrong - assignment!	Error
-3 < -2		TRUE
F = 80	never assign values to T or F in R	

Logical operations

The operators && and || are NOT working vector wise, they return always a single logical value (make sometimes sense, but also dangerous). For efficiency reasons, the logical expression is only evaluated up to the point where the result is already known.
The operators & and | are working vector wise.
Examples:

FALSE && TRUE	right hand side will NOT be evaluated
TRUE && TRUE	right hand side will be evaluated
TRUE || (x <- 3)	
FALSE || TRUE	right hand side will be evaluated
FALSE || (x <- 3)	
c(TRUE, TRUE) & c(FALSE, TRUE)	vector wise
c(TRUE, TRUE) && c(FALSE, TRUE)	NOT vector wise

Logical operations

Given a set of logical vectors, is at least one of the elements TRUE?

any(A):	Is any element in vector A TRUE?
all(A):	Are all elements in vector A TRUE?

Examples:

```
a1 <-  c(FALSE, FALSE); a2 <-  c(FALSE, TRUE); a3 <-  c(TRUE, TRUE);
any(a1);  any(a2)

## [1] FALSE
## [1] TRUE

all(a1);  all(a2)

## [1] FALSE
## [1] FALSE

!(a1); !(a2)

## [1] TRUE TRUE
## [1]  TRUE FALSE
```

Exercises:

- Compare the value of 3 to the power of 5 and 2 to the power of 8
- create a vector with numbers from 1 to 10 and assign it as 'A'
- create a vector 'B' with number from 1001 to 1010
- create a vector 'C' with values from 0 to 50 but only count in intervals of 5
- Add vector A, and C together to create a vector D