R: character manipulation

Daniel E. Weeks

University of Pittsburgh

Acknowledgment

- Many of these slides were originally created by Stephen Eglen, and are used by permission.
- Reference: Eglen SJ. A quick guide to teaching R programming to computational biology students. PLoS Comput Biol (2009) vol. 5 (8) pp. e1000482
- `http://www.ploscompbiol.org/doi/pcbi.1000482`

Packages

- R has a packaging system for external code.
- A package is loaded from a library using library(pkg.name).
- Loading a package makes the functions and commands in it available.

```
library()  ## view available packages
library(help=cluster) ## what's in this?
library(cluster) ## load package
```

Modified from original slide by Eglen (2009).

CRAN: Comprehensive R Archive Network

- CRAN: Site(s) for downloading R, and also its many contributed *packages*.
- Mac/Win have a GUI for installing packages, or it can be done on the command line:

```
install.packages("genetics")
R CMD INSTALL package.tar.gz  ## from shell
```

Modified from original slide by Eglen (2009).

Characters in R

- Excellent link for understanding characters in R:
  - Character Data in R (PDF slides by R.M. Ripley)
- Link to the Spector book:
  - `http://site.ebrary.com/lib/pitt/Doc?id=10223421`
- The 'stringr' package (related to the 'tidyverse' package):
  - `http://r4ds.had.co.nz/strings.html`

Character vectors

Character vectors are vectors of strings.

- Use single (') or double (") quotes to mark strings, but don't mix:

```
> x <- 'good'
> x
[1] "good"
> z <- "it's working"
> z
[1] "it's working"
> y <- c(x,z)
> y
[1] "good"          "it's working"
```

Modified from original slide by Eglen (2009).

Character vectors

The length() function returns the number of character elements in the object:

```
> length("")
[1] 1
> x <- c("ABCD","def","g")
> length(x)
[1] 3
> length(x[1])
[1] 1
```

Modified from original slide by Ripley (2008/9).

Character vectors

The nchar() function returns the number of characters in a character value:

```
> x <- c("ABCD","def","g")
> nchar(x)
[1] 4 3 1
> nchar(x[1])
[1] 4
```

Modified from original slide by Ripley (2008/9).

Character vectors

The str_length() function from the 'stringr' package returns the number of character elements in the object:

```
> library(stringr)
> x <- c("ABCD","def",NA)
> nchar(x)
[1]  4  3 NA
> str_length(x)
[1]  4  3 NA
```

Strings/character arrays

- Within a script, easy way to generate output:

```
> cat("Now computing the steady-state\n")
Now computing the steady-state
> x <- 134
> cat("sqrt of", x, "is", sqrt(x),"\n")
sqrt of 134 is 11.57584
> cat("sqrt of", x, "is", sqrt(x),"\n",sep='_')
sqrt of_134_is_11.57584_
```

Modified from original slide by Eglen (2009).

Strings/character arrays

- Backslash characters allow you to generate control characters, importantly: newline: $\backslash n$, tab: $\backslash t$.
- paste() returns string, e.g. for assignment.

```
> x <- 1:2
> exp.dir <- '/home/dw'
> file <- paste(exp.dir,'/results',x,'.dat',sep="")
> file
[1] "/home/dw/results1.dat" "/home/dw/results2.dat"
```

Modified from original slide by Eglen (2009).

paste, paste0 and str_c

- str_c from the stringr package

```
> paste("X","Y","Z")
[1] "X Y Z"
> paste0("X","Y","Z")
[1] "XYZ"
> str_c("X","Y","Z")
[1] "XYZ"
> str_c("X","Y","Z",sep=" ")
[1] "X Y Z"
```

Character functions

paste joins together multiple arguments element by element. Can collapse the result to a single vector using the argument collapse

```
> paste(c("X","Y"),1:4,sep="")
[1] "X1" "Y2" "X3" "Y4"
> paste(c("X","Y"),1:4,sep="",collapse=" + ")
[1] "X1 + Y2 + X3 + Y4"
```

Modified from original slide by Ripley (2008/9).

Character functions

Question: Given two vectors of alleles, how would you combine them into a single vector of genotypes?

```
> a1 <- c("A","T","A")
> a2 <- c("A","A","T")
> a1
[1] "A" "T" "A"
> a2
[1] "A" "A" "T"
```

Character functions

Answer: Given two vectors of alleles, you could combine them into a single vector of genotypes using the paste command, setting the separator to "/":

```
> paste(a1,a2)
[1] "A A" "T A" "A T"
> paste(a1,a2,sep="/")
[1] "A/A" "T/A" "A/T"
> str_c(a1,a2,sep="/")
[1] "A/A" "T/A" "A/T"
```

Strings

- Just as R stores vectors of numbers, it also stores vectors of strings.
- Cannot use ordinary subscripting to access individual characters of strings.

```
> s <- c('apple', 'bee', 'cars', 'Danish', 'egg')
> s
[1] "apple"  "bee"    "cars"   "Danish" "egg"
> nchar(s)
[1] 5 3 4 6 3
> substr(s, 2,3)
[1] "pp" "ee" "ar" "an" "gg"
```

Modified from original slide by Eglen (2009).

Strings

```
> substr(s, 2,3)
[1] "pp" "ee" "ar" "an" "gg"
> str_sub(s,2,3)
[1] "pp" "ee" "ar" "an" "gg"
```

Strings

- Pattern matching facilities are available, based on Unix terms (grep, regular expressions). These are worth learning:

```
> s
[1] "apple"  "bee"    "cars"   "Danish" "egg"
> grep('e',s)
[1] 1 2 5
> grep('^e',s)
[1] 5
> sub('e','_',s)
[1] "appl_"  "b_e"    "cars"   "Danish" "_gg"
> gsub('e','_',s)
[1] "appl_"  "b__"    "cars"   "Danish" "_gg"
```

Modified from original slide by Eglen (2009).

The "taq_cohort1_snp1.txt" file is a tab-delimited text file containing the results from a TaqMan genotyping experiment. Open it up and examine it to get a sense of its content. In this file, there were some control samples that were either from the CEPH collection of samples or water samples.

Question: How would you use the grep command to remove the "WATER" and "CEPH" lines from the "taq_cohort1_snp1.txt" file?

Answer: Using the grep command to remove the "WATER" and "CEPH" lines from the "taq_cohort1_snp1.txt" file

```
> a <- read.table("taq_cohort1_snp1.txt",skip=7,quote="",hea
> grep("CEPH",a$Lib)
 [1]   8   9  10  11 120 121 122 123 124 125 126 127 471 472
[20] 478
> head(a[grep("CEPH",a$Lib),c(1:5)],3)
   Project  Lib     Ind Indcode Flag
8  WTAC-15 CEPH 1347-02 B002RTM    1
9  WTAC-15 CEPH 1347-02 B002RTM    1
10 WTAC-15 CEPH  884-15 B002RTG    1
> b <- a[-1* grep("CEPH",a$Lib),]
> dim(a)
[1] 831  12
> dim(b)
[1] 811  12
```

Answer: Using the grep command to remove the "WATER" and "CEPH" lines from the "taq_cohort1_snp1.txt" file

```
> head(b[grep("WATER",b$Indcode),c(1:5)])
    Project          Lib   Ind Indcode Flag
1   WTAC-15 NEG_CONTROL water   WATER    0
65  WTAC-15 NEG_CONTROL water   WATER    0
66  WTAC-15 NEG_CONTROL water   WATER    0
67  WTAC-15 NEG_CONTROL water   WATER    0
68  WTAC-15 NEG_CONTROL water   WATER    0
449 WTAC-15 NEG_CONTROL water   WATER    0
> d <- b[-1*grep("WATER",b$Indcode),]
> dim(b)
[1] 811  12
> dim(d)
[1] 802  12
```

Strings

```
> s
[1] "apple"  "bee"    "cars"   "Danish" "egg"
> toupper(s)
[1] "APPLE"  "BEE"    "CARS"   "DANISH" "EGG"
> sprintf('name %s has length %d',s,nchar(s))
[1] "name apple has length 5"  "name bee has length 3"
[3] "name cars has length 4"   "name Danish has length 6"
[5] "name egg has length 3"
```

Modified from original slide by Eglen (2009).

Splitting strings

Use the strsplit function to divide up a character string into smaller parts:

```
> s <- "This is a sentence."
> parts <- strsplit(s,' ')
> parts
[[1]]
[1] "This"     "is"       "a"        "sentence."
> parts[[1]][2]
[1] "is"
```

Results are returned as a list.

Splitting strings

The unlist() function can be used to simplify the results into a vector:

```
> s <- "This is a sentence."
> parts <- strsplit(s,' ')
> allparts <- unlist(parts)
> allparts
[1] "This"     "is"       "a"        "sentence."
> allparts[2]
[1] "is"
```

Splitting strings with stringr

```
> s <- "This is a sentence."
> str_split_fixed(s,' ',n=4)
     [,1]   [,2] [,3] [,4]
[1,] "This" "is" "a"  "sentence."
> str_split(s,' ',simplify=TRUE)
     [,1]   [,2] [,3] [,4]
[1,] "This" "is" "a"  "sentence."
```

Results are returned as a character matrix (with 'n' columns).

Splitting strings

Watch out for multiple spaces - may need to split on one or more spaces:

```
> s <- "This  is a sentence."
> strsplit(s,' ')
[[1]]
[1] "This"      ""          "is"        "a"         "senten
> strsplit(s,' +')
[[1]]
[1] "This"      "is"        "a"         "sentence."
```

Splitting strings

Question: What if you have a vector of genotypes like this:

```
> g <- c("A/A","A/T","A/T","T/T","A/A")
> g
[1] "A/A" "A/T" "A/T" "T/T" "A/A"
```

How would you change this into a vector of alleles?

Splitting strings

Answer: One way to change g into a vector of alleles:

```
> g
[1] "A/A" "A/T" "A/T" "T/T" "A/A"
> al <- unlist(strsplit(gsub("/"," ",g)," "))
> al
 [1] "A" "A" "A" "T" "A" "T" "T" "T" "A" "A"
> table(al)
al
A T
6 4
```

What might be a simpler way to do this?

Splitting strings

Answer: A simpler way to change g into a vector of alleles:

```
> g
[1] "A/A" "A/T" "A/T" "T/T" "A/A"
> a2 <- unlist(strsplit(g,"/"))
> a2
 [1] "A" "A" "A" "T" "A" "T" "T" "T" "A" "A"
```

Splitting strings

Remember if 'simplify=TRUE', a character matrix is returned by 'str_split':

```
> g
[1] "A/A" "A/T" "A/T" "T/T" "A/A"
> str_split(g,"/",simplify=TRUE)
     [,1] [,2]
[1,] "A"  "A"
[2,] "A"  "T"
[3,] "A"  "T"
[4,] "T"  "T"
[5,] "A"  "A"
```

Duplicating strings

The 'stringr' package provides a useful function of duplicating strings:

```
> str_dup("AG",times=4)
[1] "AGAGAGAG"
```

Removing whitespace

The 'stringr' package provides a useful function for trimming off leading and trailing whitespace:

```
> str_trim(" AG ")
[1] "AG"
```

Regular expresssions

- Regular expressions provide a system for searching for patterns in character strings.
  - literal characters
  - character classes, in square brackets
  - modifiers: see Table 7.1 of Spector's book
- Metacharacters: . ^ $ + ? * ( ) [ ] { } | \
- Precede metacharacters by a backslash if you want to use them literally.
- Interactive web site for trying out regular expressions: https://regexr.com/

Regular expresssions

From: http://imgs.xkcd.com/comics/regular_expressions.png

Modifiers for Regular Expressions

Modifier	Meaning
^	beginning of target
$	end of target
.	matches any single character except newline
*	matches 0 or more of preceeding entity
?	matches 0 or 1 occurences of preceeding entity
+	matches 1 or more occurences of preceeding entity
{n}	matches exactly n occurences of preceeding entity
{n,}	matches at least n occurences of preceeding entity
{n,m}	matches between n and m occurences
\b	word boundary
\<	matches the start of a word
\>	matches the end of a word

Modified from Table 7.1 of Spector (2008).

Regular expressions examples

```
> x <- "My pattern string 32144"
> gsub("n.","_",x)
[1] "My patter_stri_ 32144"
> gsub("[1-3]","_",x)
[1] "My pattern string ___44"
> gsub("n\\>","_",x)
[1] "My patter_ string 32144"
> gsub("\\<p","_",x)
[1] "My _attern string 32144"
> gsub("t{2}","_",x)
[1] "My pa_ern string 32144"
```

Modified from original slide by Ripley (2008/9).

Regular expressions examples

grep(pattern, x) returns the indices of the elements that match the pattern:

```
> x <- "My pattern string 32144"
> grep("pa", c(x,tolower(x)))
[1] 1 2
> grep("PA", c(x,toupper(x)))
[1] 2
> grep("PA",c(x,toupper(x)),value=TRUE)
[1] "MY PATTERN STRING 32144"
```

Modified from original slide by Ripley (2008/9).

Regular expressions examples

sub: replaces first match
gsub: replaces all matches ("g" = global)

```
> x <- "My pattern string 32144"
> sub("PA","pa", c(x,toupper(x)))
[1] "My pattern string 32144" "MY paTTERN STRING 32144"
> sub("n.","XX", c(x,toupper(x)))
[1] "My patterXXstring 32144" "MY PATTERN STRING 32144"
> gsub("n.","XX",c(x,toupper(x)))
[1] "My patterXXstriXX 32144" "MY PATTERN STRING 32144"
```

Modified from original slide by Ripley (2008/9).

Regular expressions examples

```
> x <- "My pattern string 32144"
> sub("n.","XX", c(x,toupper(x)))
[1] "My patterXXstring 32144" "MY PATTERN STRING 32144"
> str_replace(c(x,toupper(x)),"n.","XX")
[1] "My patterXXstring 32144" "MY PATTERN STRING 32144"
> gsub("n.","XX",c(x,toupper(x)))
[1] "My patterXXstriXX 32144" "MY PATTERN STRING 32144"
> str_replace_all(c(x,toupper(x)),"n.","XX")
[1] "My patterXXstriXX 32144" "MY PATTERN STRING 32144"
```

Example: impact factor data

Question: How many journals are there in the 'Ranking_tab_delimited.txt' list that **start** with the phrase "GENET"?

Modifiers for Regular Expressions

Modifier	Meaning
^	beginning of target
$	end of target
.	matches any single character except newline
*	matches 0 or more of preceeding entity
?	matches 0 or 1 occurences of preceeding entity
+	matches 1 or more occurences of preceeding entity
{n}	matches exactly n occurences of preceeding entity
{n,}	matches at least n occurences of preceeding entity
{n,m}	matches between n and m occurences
\b	word boundary
\<	matches the start of a word
\>	matches the end of a word

Modified from Table 7.1 of Spector (2008).

Example: impact factor data

Answer: How many journal names **start** with the phrase "GENET"?

```
> a <- read.table("Ranking_tab_delimited.txt",sep="\t")
> names(a) <- c("Journal","IF")
> head(a[grep("^GENET",a$Journal),])
```

```
                   Journal   IF
2054  GENET ANAL-BIOMOL E 0.96
2055 GENET ANAL-TECH APPL 1.36
2056        GENET COUNSEL 0.38
2057       GENET ENG NEWS 0.09
2058      GENET EPIDEMIOL 1.51
2059           GENET IBER 0.10
```

```
> dim(a[grep("^GENET",a$Journal),])
```

```
[1] 14  2
```

Example: impact factor data

Question: How many journal names **end** with the phrase "GENET"?

Modifiers for Regular Expressions

Modifier	Meaning
^	beginning of target
$	end of target
.	matches any single character except newline
*	matches 0 or more of preceeding entity
?	matches 0 or 1 occurences of preceeding entity
+	matches 1 or more occurences of preceeding entity
{n}	matches exactly n occurences of preceeding entity
{n,}	matches at least n occurences of preceeding entity
{n,m}	matches between n and m occurences
\b	word boundary
\<	matches the start of a word
\>	matches the end of a word

Modified from Table 7.1 of Spector (2008).

Example: impact factor data

Answer: How many journal names **end** with the phrase "GENET"?

```
> head(a[grep("GENET$",a$Journal),])
```

```
                Journal   IF
187          ADV GENET 4.79
191       ADV HUM GENET 5.22
261 AGR HORTIQUE GENET 0.20
313      AM J HUM GENET 5.94
316      AM J MED GENET 1.62
375          ANIM GENET 1.32
```

```
> dim(a[grep("GENET$",a$Journal),])
```

```
[1] 37  2
```

Example: impact factor data

Question: How many journal names **contain** the phrase "GENET"?

- NOTE: We originally asked this question, which is not straightforward to answer using regular expressions: *"Question: How many journal names **contain** a single copy of the phrase "GENET"?"*

Modifiers for Regular Expressions

Modifier	Meaning
^	beginning of target
$	end of target
.	matches any single character except newline
*	matches 0 or more of preceeding entity
?	matches 0 or 1 occurences of preceeding entity
+	matches 1 or more occurences of preceeding entity
{n}	matches exactly n occurences of preceeding entity
{n,}	matches at least n occurences of preceeding entity
{n,m}	matches between n and m occurences
\b	word boundary
\<	matches the start of a word
\>	matches the end of a word

Modified from Table 7.1 of Spector (2008).

Example: impact factor data

Answer: How many journal names **contain** the phrase "GENET"?

```
> head(a[grep("GENET",a$Journal),])
```

```
                    Journal   IF
58   ACTA GENET MED GEMEL 0.29
187              ADV GENET 4.79
191          ADV HUM GENET 5.22
261    AGR HORTIQUE GENET 0.20
313         AM J HUM GENET 5.94
316         AM J MED GENET 1.62
```

```
> dim(a[grep("GENET",a$Journal),])
```

```
[1] 71  2
```

```
> dim(a[grep("\\bGENET\\b",a$Journal),])
```

```
[1] 59  2
```

Example: impact factor data

Answer: How many journal names **contain** the word "GENET"?

```
> head(a[grep("\\bGENET\\b",a$Journal),])
```

```
                  Journal   IF
58  ACTA GENET MED GEMEL 0.29
187             ADV GENET 4.79
191       ADV HUM GENET 5.22
261   AGR HORTIQUE GENET 0.20
313       AM J HUM GENET 5.94
316       AM J MED GENET 1.62
```

```
> dim(a[grep("\\bGENET\\b",a$Journal),])
```

```
[1] 59  2
```

Does not find journals like "IMMUNO**GENET**ICS".

Example: impact factor data

Question: How many journal names **contain** the phrase "GENET" **internally** (neither at the beginning nor the end of the journal name)?

Example: impact factor data

Answer: How many journal names **contain** the phrase "GENET" **internally**?

```
> head(a[grep(".GENET.",a$Journal),])
```

```
                    Journal   IF
58    ACTA GENET MED GEMEL 0.29
398     ANN GENET SEL ANIM 0.26
399        ANN GENET-PARIS 0.94
681    ATTI ASSOC GENET IT 0.45
950   BIOTECHNOL GENET ENG 0.71
1045     CAN J GENET CYTOL 0.96
```

```
> dim(a[grep(".GENET.",a$Journal),])
```

```
[1] 21  2
```

This will also match CYTO**GENET** CELL **GENET**".

Example: impact factor data

Task: Create a new unique journal ID 'jid' that consists of a 'J' followed by the row number.

Example: impact factor data

Task: Create a new unique journal ID 'jid' that consists of a 'J' followed by the row number.
Solution: use the paste() command

```
> a$jid <- paste("J",row.names(a),sep="")
> head(a)
```

	Journal	IF	jid
1	A VAN LEEUW J MICROB	0.81	J1
2	AAPG BULL	1.43	J2
3	ABDOM IMAG	0.62	J3
4	ABH MATH SEM HAMBURG	0.17	J4
5	ABSTR PAP AM CHEM S	5.22	J5
6	ACAD MED	0.94	J6

Example: impact factor data

Task: Change unique journal ID 'jid' so that it starts with 'Journal_' followed by the row number.

Example: impact factor data

Task: Change unique journal ID 'jid' so that it starts with 'Journal_' followed by the row number.
Solution: use the sub() command

```
> a$jid <- sub("J","Journal_",a$jid)
> head(a)
```

	Journal	IF	jid
1	A VAN LEEUW J MICROB	0.81	Journal_1
2	AAPG BULL	1.43	Journal_2
3	ABDOM IMAG	0.62	Journal_3
4	ABH MATH SEM HAMBURG	0.17	Journal_4
5	ABSTR PAP AM CHEM S	5.22	Journal_5
6	ACAD MED	0.94	Journal_6

Useful RStudio cheatsheet

See the "String manipulation with stringr cheatsheet" at https://rstudio.github.io/cheatsheets/html/strings.html

The End

What questions do you have?