Regular Expressions in R

Biostatistics 140.776

Regular Expression Functions

The primary R functions for dealing with regular expressions are

- `grep`, `grepl`: Search for matches of a regular expression/pattern in a character vector; either return the indices into the character vector that match, the strings that happen to match, or a TRUE/FALSE vector indicating which elements match
- `regexpr`, ‘gregexpr: Search a character vector for regular expression matches and return the indices of the string where the match begins and the length of the match
- `sub`, `gsub`: Search a character vector for regular expression matches and replace that match with another string
- `regexec`: Easier to explain through demonstration.

grep

```
library(readr)
commits <- read_lines("commit_logs_strip.txt.bz2")
head(commits)
```

```
[1] "commit ce5b08adc101aac0370800a230cedb24a93be679"
[2] "Merge: 7f6ef08 210649f"
[3] "Author: ZchMr <zc@.>"
[4] "Date:   Wed Oct 1 16:59:21 2014 -0400"
[5] ""
[6] "    Merge branch 'develop' of https://github.com/btsc
```

grep

How many commits are there?

```
g <- grep("^commit", commits)
head(g)
```

```
[1]   1  12  75  82  89 204
```

```
length(g)
```

```
[1] 3259
```

grep

Sometimes you want `grep()` to return the value instead of the index

```
g <- grep("^commit", commits, value = TRUE)
head(g)
```

```
[1] "commit ce5b08adc101aac0370800a230cedb24a93be679"
[2] "commit 7f6ef08e80191712a5eb0d75c42931466e7bbe73"
[3] "commit 210649f927d9233131a0b98c8db4d3c2ef9aa26a"
[4] "commit ccee79f90c2937ea902a26d0d39af3ec03542ad7"
[5] "commit 6fe5d43383d50c698993c9b46b33f08f4897c70f"
[6] "commit f24bed325a5a5a3989187edf704410d63e055efb"
```

grep

Who are the authors of these commits?

```
g <- grep("^Author", commits, value = TRUE, perl = TRUE)
head(g)
```

```
[1] "Author: ZchMr <zc@.>" "Author: ZchMr <zc@.>"
[3] "Author: DvdG <wh@.>"  "Author: ZchMr <zc@.>"
[5] "Author: ZchMr <zc@.>" "Author: DvdG <wh@.>"
```

```
length(unique(g))
```

```
[1] 20
```

grep, grepl

By default, `grep()` returns the *indices* into the character vector where the regex pattern matches.

```
head(state.name)
```

```
[1] "Alabama"    "Alaska"     "Arizona"
[4] "Arkansas"   "California" "Colorado"
```

```
grep("^New", state.name)
```

```
[1] 29 30 31 32
```

`grepl()` returns a logical vector indicating which element matches.

```
i <- grepl("^New", state.name)
head(i, 10)
```

```
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [8] FALSE FALSE FALSE
```

grep

Some limitations of `grep()`:

- The `grep()` function tells you which strings in a character vector match a certain pattern but it doesn't tell you exactly where the match occurs or what the match is (for a more complicated regex).
- The `regexpr()` function gives you the index into each string where the match begins and the length of the match for that string.
- `regexpr()` only gives you the first match of the string (reading left to right). `gregexpr()` will give you all of the matches in a given string.

regexpr

How can we obtain the email addresses of the authors?

```
commits[12:24]
```

```
 [1] "commit 7f6ef08e80191712a5eb0d75c42931466e7bbe73"
 [2] "Author: ZchMr <zc@.>"
 [3] "Date:   Wed Oct 1 16:55:12 2014 -0400"
 [4] ""
 [5] "    date changes to pages/tickets"
 [6] ""
 [7] "diff --git a/workbench/bts/boxoffice/src/views/reviev
 [8] "index 0967709..6900b84 100644"
 [9] "--- a/workbench/bts/boxoffice/src/views/review.blade.
[10] "+++ b/workbench/bts/boxoffice/src/views/review.blade.
[11] "@@ -29,7 +29,7 @@"
[12] "                  <div class=\"event-listing\" ng-repeat
[13] "                    <h2>@{{ parent[0].parent.name }} <a
```

What if we use the regex `<(.*)>` and search for that?

regexpr

We need to search the Author line for a pattern. We can first grep the Author lines and then search for a pattern.

```
author <- grep("^Author:", commits, value = TRUE)
head(author, 3)
```

```
[1] "Author: ZchMr <zc@.>" "Author: ZchMr <zc@.>"
[3] "Author: DvdG <wh@.>"
```

```
r <- regexpr("<.*>", author)
str(r)
```

```
 atomic [1:3259] 15 15 14 15 15 14 14 14 14 14 ...
 - attr(*, "match.length")= int [1:3259] 6 6 6 6 6 6 6 6 6
 - attr(*, "useBytes")= logi TRUE
```

regexpr

- `regexpr()` returns a vector of integers indicating where the match starts
- The attribute `match.length` indicates how long the match is
- If there's no match, `regexpr()` returns `-1` with a `match.length` of `-1`.

The obvious way to select out a match is to use the indices and the `substr()` function.

```
substr(author[1], 15, 15 + 6 - 1)
```

```
[1] "<zc@.>"
```

```
substr(author[3], 14, 14 + 6 - 1)
```

```
[1] "<wh@.>"
```

regmatches

We can also use the `regmatches()` function to just grab all of the matches at once.

```
r <- regexpr("<.*>", author)
m <- regmatches(author, r)
head(m)
```

```
[1] "<zc@.>" "<zc@.>" "<wh@.>" "<zc@.>" "<zc@.>"
[6] "<wh@.>"
```

sub/gsub

But we still don't have actual email addresses. We need to remove the < and > characters. We can use the `sub()` function for that.

```
sub("<", "", m[1:5])
```

```
[1] "zc@.>" "zc@.>" "wh@.>" "zc@.>" "zc@.>"
```

```
sub(">", "", m[1:5])
```

```
[1] "<zc@." "<zc@." "<wh@." "<zc@." "<zc@."
```

But we want to remove *both* < and >!

sub/gsub

We can use a regular expression in `sub()`.

```
sub("<|>", "", m[1:5])
```

```
[1] "zc@.>" "zc@.>" "wh@.>" "zc@.>" "zc@.>"
```

`gsub()` substitutes all occurrences of the regex (g is for "global").

```
gsub("<|>", "", m[1:5])
```

```
[1] "zc@." "zc@." "wh@." "zc@." "zc@."
```

regexec

The `regexec()` function can make the previous task a bit simpler by using *parenthesized sub-expressions*.

```
author[1]
```

```
[1] "Author: ZchMr <zc@.>"
```

We can capture the email address portion of the line with parentheses.

```
regexec("^Author: [^ ]+ <(.*)>", author[1])
```

```
[[1]]
[1]  1 16
attr(,"match.length")
[1] 20  4
attr(,"useBytes")
[1] TRUE
```

regexec

```
r <- regexec("^Author: [^ ]+ <(.*)>", author[1])
regmatches(author[1], r)

[[1]]
[1] "Author: ZchMr <zc@.>" "zc@."
```

regexec

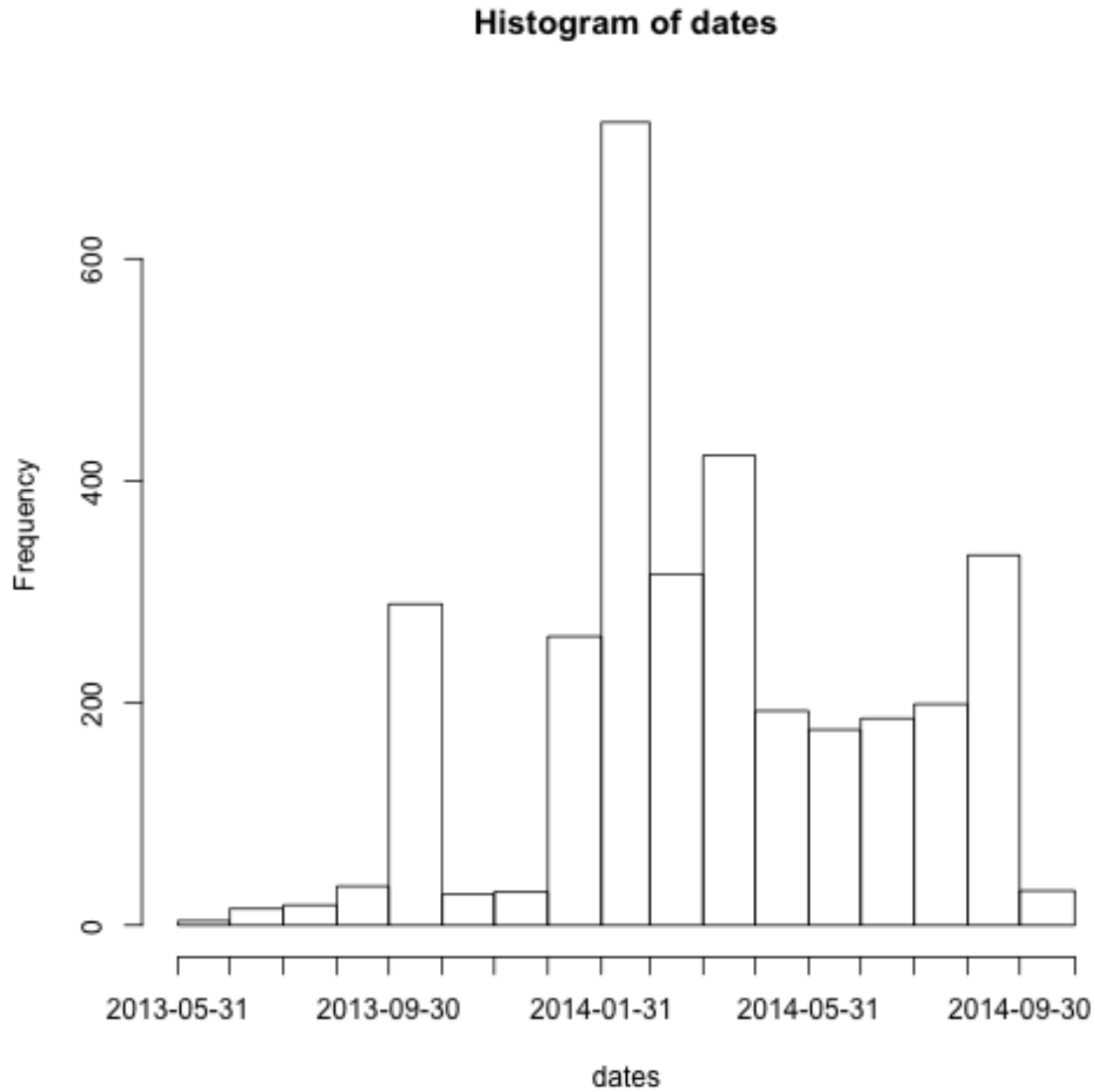
When were all of the commits made?

```
r <- regexec("^Date: +(.*)$", commits)
m <- regmatches(commits, r)
u <- sapply(m, length) > 0
dates <- sapply(m[u], function(x) x[2])
dates <- strptime(dates, "%a %b %d %H:%M:%S %Y",
                  tz = "America/New_York")
```

Histogram

You can make a histogram of the dates

```
hist(dates, "month", freq = TRUE)
```



Summary

The primary R functions for dealing with regular expressions are

- `grep`, `grepl`: Search for matches of a regular expression/pattern in a character vector
- `regexpr`, `gregexpr`: Search a character vector for regular expression matches and return the indices where the match begins; useful in conjunction with regmatches
- `sub`, `gsub`: Search a character vector for regular expression matches and replace that match with another string
- `regexec`: Gives you indices of parethensized sub-expressions.