# RNA parsing

A. Blejec

March 16, 2010

**Abstract**

⋈

## Contents

## 1 Motivation

Tal Galili [tal.galili@gmail.com] writes:

```
For some work I am doing on RNA, I want to use R to do string parsing that
(I think) is like a simplistic HTML parsing.


For example, let's say we have the following two variables:

    Seq <-
"GCCTCGATAGCTCAGTTGGGAGAGCGTACGACTGAAGATCGTAAGGtCACCAGTTCGATCCTGGTTCGGGGCA"
    Str <-
">>>>>>>..>>>>........<<<<.>>>>>.......<<<<<.....>>>>>.......<<<<<<<<<<<<."

Say that I want to parse "Seq" According to "Str", by using the legend here

Seq: GCCTCGATAGCTCAGTTGGGAGAGCGTACGACTGAAGATCGTAAGGtCACCAGTTCGATCCTGGTTCGGGGCA
Str: >>>>>>>..>>>>........<<<<.>>>>>.......<<<<<.....>>>>>.......<<<<<<<<<<<<.

     |     |  |              | |               |     |               ||     |

     +-----+  +--------------+ +---------------+     +---------------++-----+
        |        Stem 1            Stem 2                 Stem 3         |

        |                                                                |

        +----------------------------------------------------------------+

                                Stem 0
```

Assume that we always have 4 stems (0 to 3), but that the length of letters before and after each of them can very.

The output should be something like the following list structure:

```
    list(
     "Stem 0 opening" = "GCCTCGA",
     "before Stem 1" = "TA",
     "Stem 1" = list(opening = "GCTC",
     inside = "AGTTGGGA",
     closing = "GAGC"
     ),
     "between Stem 1 and 2" = "G",
     "Stem 2" = list(opening = "TACGA",
     inside = "CTGAAGA",
     closing = "TCGTA"
     ),
     "between Stem 2 and 3" = "AGGtC",
     "Stem 3" = list(opening = "ACCAG",
     inside = "TTCGATC",
     closing = "CTGGT"
     ),
     "After Stem 3" = "",
     "Stem 0 closing" = "TCGGGGC"
    )
```

I don't have any experience with programming a parser, and would like advices as to what strategy to use when programming something like this (and any recommended R commands to use).

What I was thinking of is to first get rid of the "Stem 0", then go through the inner string with a recursive function (let's call it "seperate.stem") that each time will split the string into:

1. before stem
2. opening stem
3. inside stem
4. closing stem
5. after stem

```
Where the "after stem" will then be recursively entered into the same function ("sepe

The thing is that I am not sure how to try and do this coding without using a loop.
```

## 2 Data

```
> Seq <- "GCCTCGATAGCTCAGTTGGGAGAGCGTACGACTGAAGATCGTAAGGtCACCAGTTCGATCCTGGTTCG
> Str <- ">>>>>>>..>>>>........<<<<.>>>>>.......<<<<<.....>>>>>.......<<<<<<<<
```

## 3 Development of solution

This version uses regular expressions, functions `regexpr()` and `substr()`.

The following code extracts substrings according to the supporting pattern in Str:

```
> (starts <- gregexpr(">*", Str))

[[1]]
 [1]  1  8  9 10 14 15 16 17 18 19 20 21 22 23 24 25 26 27 32 33 34 35
[23] 36 37 38 39 40 41 42 43 44 45 46 47 48 49 54 55 56 57 58 59 60 61
[45] 62 63 64 65 66 67 68 69 70 71 72 73
attr(,"match.length")
 [1] 7 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[34] 0 0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

> X <- sapply(starts, FUN = function(x, Seq) substring(Seq,
+     x, x + attr(x, "match.length") - 1), Seq = Seq)
> X <- X[!X == ""]
> openings <- X
> openings

[1] "GCCTCGA" "GCTC"    "TACGA"   "ACCAG"


> (starts <- gregexpr(">[.]*<", Str))

[[1]]
[1] 13 31 53
attr(,"match.length")
[1] 10  9  9

> X <- sapply(starts, FUN = function(x, Seq) substring(Seq,
+     x + 1, x + attr(x, "match.length") - 2), Seq = Seq)
> X <- X[!X == ""]
> insides <- X
> insides

[1] "AGTTGGGA" "CTGAAGA"  "TTCGATC"


> (starts <- gregexpr("<*", Str))
```

```
[[1]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22
[23] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 44 45 46 47 48 49 50 51
[45] 52 53 54 55 56 57 58 59 60 61 73
attr(,"match.length")
 [1]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4
[23]  0  0  0  0  0  0  0  0  0  0  0  0  0  5  0  0  0  0  0  0  0  0
[45]  0  0  0  0  0  0  0  0  0 12  0
> X <- sapply(starts, FUN = function(x, Seq) substring(Seq,
+     x, x + attr(x, "match.length") - 1), Seq = Seq)
> X <- X[!X == ""]
> closings <- X
> closings
[1] "GAGC"         "TCGTA"        "CTGGTTCGGGGC"

> (id <- gregexpr("^[.]*", Str))
[[1]]
[1] 1
attr(,"match.length")
[1] 0
> X <- sapply(id, FUN = function(x, Seq) substring(Seq,
+     x, x + attr(x, "match.length") - 1), Seq = Seq)
> X <- X[!X == ""]
> before0 <- X
> before0
character(0)

> (id <- gregexpr("^[.]*>+", Str))
[[1]]
[1] 1
attr(,"match.length")
[1] 7
> X <- sapply(id, FUN = function(x, Seq) substring(Seq,
+     x, x + attr(x, "match.length") - 1), Seq = Seq)
> X <- X[!X == ""]
> stem0 <- X
> stem0
[1] "GCCTCGA"

> (id <- gregexpr(">+[.]+<+", Str))
[[1]]
[1] 10 27 49
attr(,"match.length")
[1] 16 17 24
> X <- sapply(id, FUN = function(x, Seq) substring(Seq,
+     x, x + attr(x, "match.length") - 1), Seq = Seq)
> X <- X[!X == ""]
> stems <- X
> stems
```

```
[1] "GCTCAGTTGGGAGAGC"         "TACGACTGAAGATCGTA"
[3] "ACCAGTTCGATCCTGGTTCGGGGC"

> getStem <- function(pattern, Seq = Seq, Str = Str) {
+     (id <- gregexpr(pattern, Str))
+     X <- sapply(id, FUN = function(x, Seq) substring(Seq,
+         x, x + attr(x, "match.length") - 1), Seq = Seq)
+     str <- sapply(id, FUN = function(x, Seq) substring(Seq,
+         x, x + attr(x, "match.length") - 1), Seq = Str)
+     str <- str[!X == ""]
+     X <- X[!X == ""]
+     if (length(X) == 0)
+         X <- str <- ""
+     return(cbind(X, str))
+ }

> stem0 <- getStem("^[.]*>{7}[.]*", Seq, Str)
> stem0
     X           str
[1,] "GCCTCGATA" ">>>>>>>.."
> stem4 <- getStem("[.]*<{7}[.]*$", Seq, Str)
> stem4
     X          str
[1,] "TCGGGGCA" "<<<<<<<."

> getStem("[.]*>+[.]+<+[.]*", Str, Str)
     X                           str
[1,] "..>>>>........<<<<."       "..>>>>........<<<<."
[2,] ">>>>>.......<<<<<....."    ">>>>>.......<<<<<....."
[3,] ">>>>>.......<<<<<<<<<<<<." ">>>>>.......<<<<<<<<<<<<."
> stems <- getStem("[.]*>+[.]+<+[.]*", Seq, Str)
> stems
     X                           str
[1,] "TAGCTCAGTTGGGAGAGCG"       "..>>>>........<<<<."
[2,] "TACGACTGAAGATCGTAAGGtC"    ">>>>>.......<<<<<....."
[3,] "ACCAGTTCGATCCTGGTTCGGGGCA" ">>>>>.......<<<<<<<<<<<<."

> splitStem <- function(x) {
+     str <- x[2]
+     X <- x[1]
+     (y <- getStem("^[.]+", X, str)[1])
+     (X <- substr(X, nchar(y) + 1, nchar(X)))
+     (str <- substr(str, nchar(y) + 1, nchar(str)))
+     before <- y
+     (y <- getStem("^>+", X, str)[1])
+     (X <- substr(X, nchar(y) + 1, nchar(X)))
+     (str <- substr(str, nchar(y) + 1, nchar(str)))
+     opening <- y
+     (y <- getStem("^[.]*", X, str))
+     (X <- substr(X, nchar(y) + 1, nchar(X)))
+     (str <- substr(str, nchar(y) + 1, nchar(str)))
+     inside <- y
+     (y <- getStem("^<+", X, str))
+     (X <- substr(X, nchar(y) + 1, nchar(X)))
+     (str <- substr(str, nchar(y) + 1, nchar(str)))
+     closing <- y
+     (y <- getStem("^[.]*$", X, str))
+     (X <- substr(X, nchar(y) + 1, nchar(X)))
+     (str <- substr(str, nchar(y) + 1, nchar(str)))
+     after <- y
+     return(c(before = before[1], opening = opening[1],
+         inside = inside[1], closing = closing[1], after = after[1]))
+ }
> stem0
     X           str
[1,] "GCCTCGATA" ">>>>>>>.."
> splitStem(stem0)
   before   opening    inside   closing     after 
       "" "GCCTCGA"      "TA"        ""        "" 

> stems <- rbind(stem0, stems, stem4)
> t(stems)
    [,1]        [,2]                  [,3]                    
X   "GCCTCGATA" "TAGCTCAGTTGGGAGAGCG" "TACGACTGAAGATCGTAAGGtC"
str ">>>>>>>.." "..>>>>........<<<<." ">>>>>.......<<<<<....."
    [,4]                        [,5]      
X   "ACCAGTTCGATCCTGGTTCGGGGCA" "TCGGGGCA"
str ">>>>>.......<<<<<<<<<<<<." "<<<<<<<."
> apply(stems, 1, splitStem)
        [,1]      [,2]       [,3]      [,4]           [,5]     
before  ""        "TA"       ""        ""             ""       
opening "GCCTCGA" "GCTC"     "TACGA"   "ACCAG"        ""       
inside  "TA"      "AGTTGGGA" "CTGAAGA" "TTCGATC"      ""       
closing ""        "GAGC"     "TCGTA"   "CTGGTTCGGGGC" "TCGGGGC"
after   ""        "G"        "AGGtC"   "A"            "A"      
```

## 4 All in one function: splitSeq()

```
> splitSeq <- function(Seq, Str) {
+     getStem <- function(pattern, Seq = Seq, Str = Str) {
+         (id <- gregexpr(pattern, Str))
+         X <- sapply(id, FUN = function(x, Seq) substring(Seq,
+             x, x + attr(x, "match.length") - 1), Seq = Seq)
+         str <- sapply(id, FUN = function(x, Seq) substring(Seq,
+             x, x + attr(x, "match.length") - 1), Seq = Str)
+         str <- str[!X == ""]
+         X <- X[!X == ""]
+         if (length(X) == 0)
+             X <- str <- ""
+         return(cbind(X, str))
+     }
+     splitStem <- function(x) {
+         str <- x[2]
+         X <- x[1]
+         (y <- getStem("^[.]+", X, str)[1])
+         (X <- substr(X, nchar(y) + 1, nchar(X)))
+         (str <- substr(str, nchar(y) + 1, nchar(str)))
+         before <- y
+         (y <- getStem("^>+", X, str)[1])
+         (X <- substr(X, nchar(y) + 1, nchar(X)))
+         (str <- substr(str, nchar(y) + 1, nchar(str)))
+         opening <- y
+         (y <- getStem("^[.]*", X, str))
+         (X <- substr(X, nchar(y) + 1, nchar(X)))
+         (str <- substr(str, nchar(y) + 1, nchar(str)))
+         inside <- y
+         (y <- getStem("^<+", X, str))
+         (X <- substr(X, nchar(y) + 1, nchar(X)))
+         (str <- substr(str, nchar(y) + 1, nchar(str)))
+         closing <- y
+         (y <- getStem("^[.]*$", X, str))
+         (X <- substr(X, nchar(y) + 1, nchar(X)))
+         (str <- substr(str, nchar(y) + 1, nchar(str)))
+         after <- y
+         return(c(before = before[1], opening = opening[1],
+             inside = inside[1], closing = closing[1],
+             after = after[1]))
+     }
+     (stem0 <- getStem("^[.]*>{7}[.]*", Seq, Str))
+     (stem4 <- getStem("[.]*<{7}[.]*$", Seq, Str))
+     (str <- substring(Str, nchar(stem0[1]) + 1, nchar(Str) -
+         nchar(stem4[1])))
+     (seq <- substring(Seq, nchar(stem0[1]) + 1, nchar(Seq) -
+         nchar(stem4[1])))
+     (stems <- getStem("[.]*>+[.]+<+[.]*", seq, str))
+     (stems <- rbind(stem0, stems, stem4))
+     (parts <- apply(stems, 1, splitStem))
+     dimnames(parts)[[2]] <- paste("stem", 0:4, sep = "")
+     parts["after", 1] <- parts["inside", 1]
+     parts["inside", 1] <- ""
+     return(parts)
+ }
```

```
> parsed <- splitSeq(Seq, Str)
> parsed
        stem0     stem1      stem2     stem3     stem4
before  ""        ""         ""        ""        ""
opening "GCCTCGA" "GCTC"     "TACGA"   "ACCAG"   ""
inside  ""        "AGTTGGGA" "CTGAAGA" "TTCGATC" ""
closing ""        "GAGC"     "TCGTA"   "CTGGT"   "TCGGGGC"
after   "TA"      "G"        "AGGtC"   ""        "A"
```

From here you can compose your list.

```
> x <- parsed
> x <- lapply(apply(x, 2, list), FUN = function(x) as.list(unlist(x)))
> str(x)
```

```
List of 5
 $ stem0:List of 5
  ..$ before : chr ""
  ..$ opening: chr "GCCTCGA"
  ..$ inside : chr ""
  ..$ closing: chr ""
  ..$ after  : chr "TA"
 $ stem1:List of 5
  ..$ before : chr ""
  ..$ opening: chr "GCTC"
  ..$ inside : chr "AGTTGGGA"
  ..$ closing: chr "GAGC"
  ..$ after  : chr "G"
 $ stem2:List of 5
  ..$ before : chr ""
  ..$ opening: chr "TACGA"
  ..$ inside : chr "CTGAAGA"
  ..$ closing: chr "TCGTA"
  ..$ after  : chr "AGGtC"
 $ stem3:List of 5
  ..$ before : chr ""
  ..$ opening: chr "ACCAG"
  ..$ inside : chr "TTCGATC"
  ..$ closing: chr "CTGGT"
  ..$ after  : chr ""
 $ stem4:List of 5
  ..$ before : chr ""
  ..$ opening: chr ""
  ..$ inside : chr ""
  ..$ closing: chr "TCGGGGC"
  ..$ after  : chr "A"
```

# SessionInfo

- R version 2.10.0 (2009-10-26), `i386-pc-mingw32`
- Locale: `LC_COLLATE=Slovenian_Slovenia.1250`, `LC_CTYPE=Slovenian_Slovenia.1250`, `LC_MONETARY=Slovenian_Slovenia.1250`, `LC_NUMERIC=C`, `LC_TIME=Slovenian_Slovenia.1250`
- Base packages: base, datasets, graphics, grDevices, methods, splines, stats, utils
- Other packages: Hmisc 3.7-0, survival 2.35-8
- Loaded via a namespace (and not attached): cluster 1.12.1, grid 2.10.0, lattice 0.18-3