Read fixed width text file

82

41

I'm trying to load this ugly-formatted data-set into my R session:
http://www.cpc.ncep.noaa.gov/data/indices/wksst8110.for

```
Weekly SST data starts week centered on 3Jan1990

Nino1+2      Nino3        Nino34        Nino4
Week          SST SSTA     SST SSTA     SST SSTA     SST SSTA
03JAN1990     23.4-0.4     25.1-0.3     26.6 0.0     28.6 0.3
10JAN1990     23.4-0.8     25.2-0.3     26.6 0.1     28.6 0.3
17JAN1990     24.2-0.3     25.3-0.3     26.5-0.1     28.6 0.3
```

So far, i can read the lines with

```
x = readLines(path)
```

But the file mixes 'white space' with '-' as separators, and i'm not a regex expert. I Appreciate any help on turning this into a nice and clean R data-frame. thanks!

r fixed-width

edited Jan 17 '13 at 16:48
Andrie
140k 29 369 443

asked Jan 17 '13 at 16:33
Fernando
5,313 4 35 69

5 And take a look at `read.fwf` to read read fixed width formatted data. – Paul Hiemstra Jan 17 '13 at 16:37

1 I think it's a better idea to process each row. It mixes '-' with ' ' characters. – Fernando Jan 17 '13 at 16:38

Alternatively, you could say white-space or - is just one character, so first replace all multiple occurrences of a space with a tab character, then split all tab-separated entry's on - or white space. – GitaarLAB Jan 17 '13 at 16:45

Fixed width = no separators. That means the "-" is a minus sign and the spaces are not separators either, they just occur when the number doesn't fill the entire available width – Eusebio Rufian-Zilbermann Feb 15 '15 at 5:53

6 Answers

170

This is a fixed width file. Use `read.fwf()` to read it:

```
x <- read.fwf(
  file=url("http://www.cpc.ncep.noaa.gov/data/indices/wksst8110.for"),
  skip=4,
  widths=c(12, 7, 4, 9, 4, 9, 4, 9, 4))

head(x)

           V1   V2   V3   V4   V5   V6   V7   V8  V9
1  03JAN1990   23.4 -0.4 25.1 -0.3 26.6  0.0 28.6 0.3
2  10JAN1990   23.4 -0.8 25.2 -0.3 26.6  0.1 28.6 0.3
3  17JAN1990   24.2 -0.3 25.3 -0.3 26.5 -0.1 28.6 0.3
```

Update

The package `readr` (released April, 2015) provides a simple and fast alternative.

```
library(readr)

x <- read_fwf(
  file="http://www.cpc.ncep.noaa.gov/data/indices/wksst8110.for",
  skip=4,
  fwf_widths(c(12, 7, 4, 9, 4, 9, 4, 9, 4)))
```

Speed comparison: `readr::read_fwf()` was ~2x faster than `utils::read.fwf()`.

edited Nov 15 '15 at 23:01
Megatron
6,356 6 51 68

answered Jan 17 '13 at 16:45
Andrie
140k 29 369 443

7 @Andrie how did you know what were the widths and skips? – Koba Apr 20 '14 at 19:55

12 @Koba: I copied and pasted one of the lines into a text editor that had a column count and I manually counted the widths for each column (including whitespace when required). Also you can tell that you need to skip 4 whole lines before you get to the raw data. – rayryeng Apr 21 '14 at 1:57

5 @Pavithra's answer below with negative column widths for skipping unwanted whitespace might be better suited for the accepted answer. – Marius Butuc Nov 19 '14 at 20:49

1 @Andrie How did you get the fwf_widths values? – BICube Jul 28 '17 at 23:16

2 @Ala I believe `readr::fwf_empty` will attempt to guess the widths for you. The examples for `readr::read_fwf` shows the usage for `readr::fwf_empty`. – Jake Fisher Jul 30 '17 at 19:56

53

Another way to determine widths...

```
df <- read.fwf(
  file=url("http://www.cpc.ncep.noaa.gov/data/indices/wksst8110.for"),
  widths=c(-1, 9, -5, 4, 4, -5, 4, 4, -5, 4, 4, -5, 4, 4),
  skip=4
)
```

The -1 in the widths argument says there is a one-character column that should be ignored,the -5 in the widths argument says there is a five-character column that should be ignored, likewise...

ref : https://www.inkling.com/read/r-cookbook-paul-teetor-1st/chapter-4/recipe-4-6

edited Dec 27 '17 at 16:04
Claus Wilke
8,462 4 30 56

answered Jun 12 '14 at 21:42
GunaPa
2,126 6 29 43

First off, that question is directly from a the Coursera "Get Data and Clean It" course by Leeks. While there is another part of the question, the tough part is reading the file.

I hate R's fixed width procedure. It is slow and for large number of variables, it very quickly becomes a pain to negate certain columns, etc.

I think its easier to use `readLines()` and then from that use `substr()` to make your variables

```
x <- readLines(con=url("http://www.cpc.ncep.noaa.gov/data/indices/wksst8110.for"))

# Skip 4 lines
x <- x[-(1:4)]

mydata <- data.frame(var1 = substr(x, 1, 10),
                     var2 = substr(x, 16, 19),
                     var3 = substr(x, 20, 23),
                     var4 = substr(x, 29, 32)  # and so on and so on
                     )
```

edited Dec 27 '17 at 16:02

Claus Wilke
8,462 4 30 56

answered Oct 17 '14 at 18:35

James Holland
640 7 14

This approach worked for me. Two additional tips: 1) you can define mydata to be only the data you need. So it could be as simple as `mydata <- data.frame(var4 = substr(x,29,32))` if you only need the fourth column of data. Also, for Windows users, Notepad++ with the TextFX plugin will get you a plain and simple, counted character ruler so you can figure out what to put in the start and stop values in `substr` . Note, however that the stop value is one more than the position of the last character you want to preserve. – globalSchmidt Mar 19 '17 at 11:22

10

You can now use the `read_fwf()` function in Hadley Wickham's `readr` package.

- Annoucement: http://blog.rstudio.org/2015/04/09/readr-0-1-0/
- Development page: https://github.com/hadley/readr
- CRAN page: http://cran.r-project.org/web/packages/readr/index.html

A huge performance improvement is to be expected, compared to base `read.fwf()` .

answered Apr 10 '15 at 10:20

Lionel Henry
3,352 17 22

5

I document here the list of alternatives for reading fixed-width files in R, as well as providing some benchmarks for which is fastest.

My preferred approach is to combine `fread` with `stringi` ; it's competitive as the fastest approach, and has the added benefit (IMO) of storing your data as a `data.table` :

```
library(data.table)
library(stringi)

col_ends <-
  list(beg = c(1, 10, 15, 19, 23, 28, 32, 36,
               41, 45, 49, 54, 58),
       end = c(9, 14, 18, 22, 27, 31, 35,
               40, 44, 48, 53, 57, 61))

data = fread(
  "http://www.cpc.ncep.noaa.gov/data/indices/wksst8110.for",
  header = FALSE, skip = 4L, sep = NULL
  )[, lapply(1:(length(col_ends$beg)),
             function(ii)
               stri_sub(V1, col_ends$beg[ii], col_ends$end[ii]))
```

```
#    2: 10JAN1990 23.4 -0.8 25.2 -0.3 26.6  0.1 28.6  0.3
#    3: 17JAN1990 24.2 -0.3 25.3 -0.3 26.5 -0.1 28.6  0.3
#    4: 24JAN1990 24.4 -0.5 25.5 -0.4 26.5 -0.1 28.4  0.2
#    5: 31JAN1990 25.1 -0.2 25.8 -0.2 26.7  0.1 28.4  0.2
#   ---
# 1365: 24FEB2016 27.1  0.9 28.4  1.8 29.0  2.1 29.5  1.4
# 1366: 02MAR2016 27.3  1.0 28.6  1.8 28.9  1.9 29.5  1.4
# 1367: 09MAR2016 27.7  1.2 28.6  1.6 28.9  1.8 29.6  1.5
# 1368: 16MAR2016 27.5  1.0 28.8  1.7 28.9  1.7 29.6  1.4
# 1369: 23MAR2016 27.2  0.9 28.6  1.4 28.8  1.5 29.5  1.2
```

Note that `fread` automatically strips leading and trailing whitespace -- sometimes, this is undesirable, in which case set `strip.white = FALSE`.

We could also have started with a vector of column widths `ww` by doing:

```
ww <- c(9, 5, 4, 4, 5, 4, 4, 5, 4, 4, 5, 4, 4)
nd <- cumsum(ww)

col_ends <-
  list(beg = c(1, nd[-length(nd)]+1L),
       end = nd)
```

And we could have picked which columns to exclude more robustly by using negative indices like:

```
col_ends <-
  list(beg = c(1, -10, 15, 19, -23, 28, 32, -36,
               41, 45, -49, 54, 58),
       end = c(9, 14, 18, 22, 27, 31, 35,
               40, 44, 48, 53, 57, 61))
```

Then replace `col_ends$beg[ii]` with `abs(col_ends$beg[ii])` and in the next line:

```
paste0("V", which(col_ends$beg < 0))
```

Lastly, if you want the column names to be read programatically as well, you could clean up with `readLines`:

```
cols <-
  gsub("\\s", "",
       sapply(1:(length(col_ends$beg)),
              function(ii)
                stri_sub(readLines(URL, n = 4L)[4L],
                         col_ends$beg[ii]+1L,
                         col_ends$end[ii]+1L)))

cols <- cols[cols != ""]
```

(note that combining this step with `fread` would require creating a copy of the table in order to remove the header row, and would thus be inefficient for large data sets)

edited Apr 24 '18 at 9:17

answered Mar 31 '16 at 14:34

MichaelChirico

21.1k 8 64 121

I don't know a thing about R, but I can provide you with a regex that will match such lines:

edited Dec 27 '17 at 16:03

Claus Wilke
8,462 4 30 56

answered Jan 17 '13 at 16:43

11684
5,379 9 40 69

protected by Andrie Nov 17 '15 at 12:38

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?