

---

```
title: "Dendrochronology in R"
author: "Jeff Atkins"
date: 'r format(Sys.time(), "%B %d, %Y)"
output:
word_document: default
pdf_document: default
html_document: default
```

---

## Working with ring-width data

dplR version 1.7.2 is available on CRAN

Here we show how to import tree-ring data.

We are using data downloaded from the NOAA XXXX, specifically the Cook - Kelsey Tract TSCA-ITRDB NC005 [url - <https://www.ncdc.noaa.gov/paleo-search/study/2987?siteId=15463>]

### Installation of dplR

```
# install the package if not already installed and call it via library()
if(!require(dplR)){install.packages("dplR")}

## Loading required package: dplR

library(dplR)
library(tidyverse)

## -- Attaching packages ----- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr    1.0.7
## v tidyr   1.1.3     v stringr  1.4.0
## v readr   2.0.0     vforcats 0.5.1

## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()
```

### Importing ring-width data

The vignette included with dplR provides a thorough walkthrough (<https://cran.r-project.org/web/packages/dplR/vignettes/intro-dplR.pdf>) here we show some highlights based on the independent dataset we have downloaded.

```
# the primary function in dplR is read.rwl()
rw <- dplR::read.rwl("../inst/extdata/dendro/nc005.rwl")
```

```

## Attempting to automatically detect format.
## Assuming a Tucson format file.
## There does not appear to be a header in the rw file
## There are 31 series
## 1      102061    1653  1983  0.01
## 2      102062    1649  1983  0.01
## 3      102081    1707  1983  0.01
## 4      102101    1634  1983  0.01
## 5      102102    1637  1983  0.01
## 6      102121    1675  1976  0.01
## 7      102141    1638  1983  0.01
## 8      102142    1665  1983  0.01
## 9      102171    1623  1983  0.01
## 10     102172    1654  1983  0.01
## 11     102181    1695  1983  0.01
## 12     102182    1684  1983  0.01
## 13     102241    1654  1983  0.01
## 14     102261    1694  1983  0.01
## 15     102271    1737  1983  0.01
## 16     102272    1678  1983  0.01
## 17     102291    1594  1983  0.01
## 18     102292    1594  1983  0.01
## 19     102301    1673  1932  0.01
## 20     102302    1628  1977  0.01
## 21     102311    1623  1983  0.01
## 22     102312    1645  1983  0.01
## 23     102321    1689  1983  0.01
## 24     102322    1738  1983  0.01
## 25     102341    1560  1983  0.01
## 26     102342    1573  1983  0.01
## 27     102431    1692  1983  0.01
## 28     102432    1663  1983  0.01
## 29     102451    1617  1983  0.01
## 30     102452    1581  1983  0.01
## 31     102491    1612  1983  0.01

```

```

# lets look at the dimension of our dataset
dim(rw) # 424 years and 31 series

```

```

## [1] 424 31

```

```

# and you can see that the data are both class "rw" and "data.frame"
class(rw)

```

```

## [1] "rw"           "data.frame"

```

## Data descriptions

dplR has a function, `rw.report()` that provides a fairly detailed description of the data set(s) you are working with.

```

## Number of dated series: 31

```

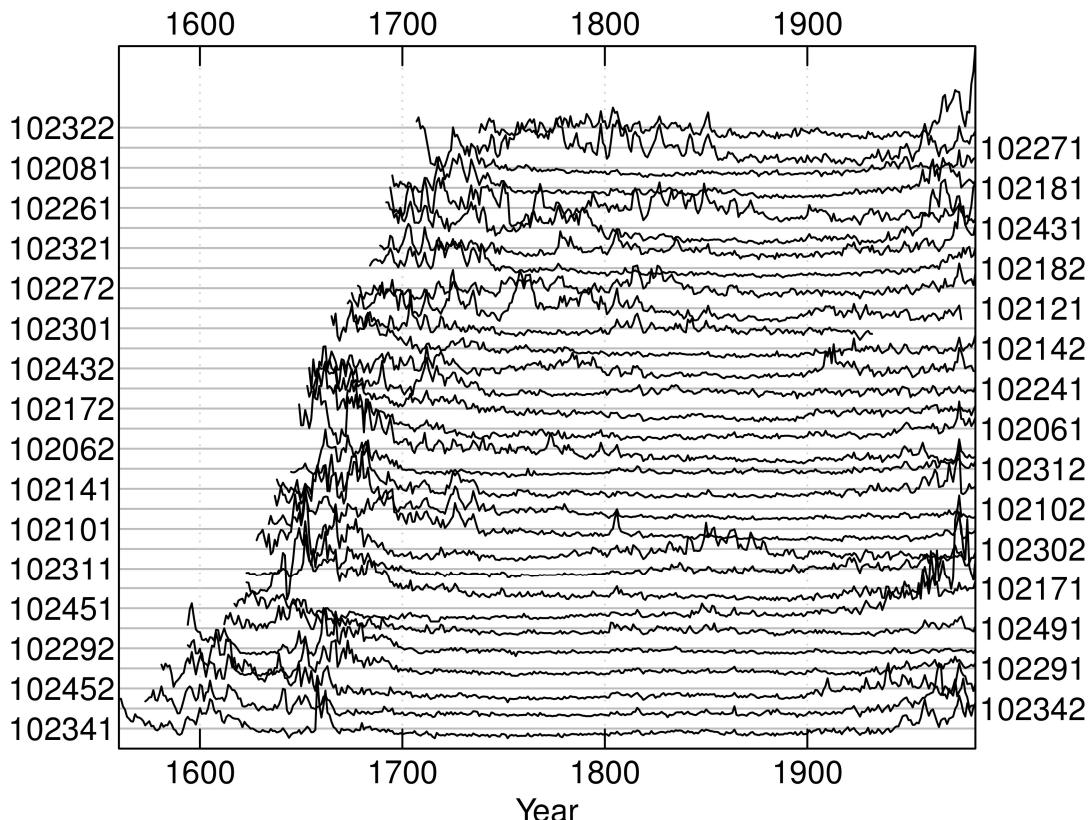
```

## Number of measurements: 10281
## Avg series length: 331.6452
## Range: 424
## Span: 1560 - 1983
## Mean (Std dev) series intercorrelation: 0.5316996 (0.05975429)
## Mean (Std dev) AR1: 0.8697419 (0.04867715)
## -----
## Years with absent rings listed by series
##   Series 102182 -- 1877 1888
##   Series 102291 -- 1644
##   Series 102292 -- 1756
##   Series 102311 -- 1752 1753
##   Series 102312 -- 1752 1753
##   Series 102321 -- 1876
##   Series 102341 -- 1716 1752
##   Series 102451 -- 1714 1715 1716 1717 1718
##   Series 102452 -- 1752
## 17 absent rings (0.165%)
## -----
## Years with internal NA values listed by series
##   None

```

## Plotting data (Spaghetti Plots)

We can now plot these data.



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

## Detrending

```
## Detrending example, albeit brief.
rw.detrend <- detrend(rwl = rw, method = "ModNegExp", verbose = TRUE)

# we can look at the col names of the new object
names(rw.detrend)

## [1] "102061" "102062" "102081" "102101" "102102" "102121" "102141" "102142"
## [9] "102171" "102172" "102181" "102182" "102241" "102261" "102271" "102272"
## [17] "102291" "102292" "102301" "102302" "102311" "102312" "102321" "102322"
## [25] "102341" "102342" "102431" "102432" "102451" "102452" "102491"

# and we can look at the means of those individuals
colMeans(rw.detrend, na.rm=TRUE)

##    102061    102062    102081    102101    102102    102121    102141    102142
## 1.0046278 1.0021507 0.9977224 1.0407909 1.0610862 1.0023233 0.9996141 1.0014092
##    102171    102172    102181    102182    102241    102261    102271    102272
## 1.0108265 1.0001056 0.9977949 1.0038460 0.9997856 1.0000540 0.9998278 1.0000000
##    102291    102292    102301    102302    102311    102312    102321    102322
## 0.9979854 0.9990962 0.9996780 1.0112870 1.0000000 1.0007899 0.9996691 1.0000000
##    102341    102342    102431    102432    102451    102452    102491
## 0.9979342 0.9979640 0.9982441 0.9994659 0.9992464 1.0024558 1.0383337
```

## Building a Chronology

A chronology can be built from the detrended data.

```
rw.crn <- chron(rw.detrend, prefix = "CAM")

# now we can plot these data
plot(rw.crn, add.spline=TRUE, nyrs=20)
```

