R-bloggers

R news and tutorials contributed by (573) R bloggers

- Home
- About
- RSS
- add your blog!
- Learn R
- R jobs���
- Contact us

Welcome!

Follow @rbloggers 30K fo

Here you will find daily **news and tutorials about R**, contributed by over 573 bloggers. There are many ways to **follow us -**

By e-mail:

Your e-mail h

Subscribe

27023 readers BY FEEDBURNER

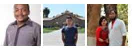
On Facebook:

R blog…

32K likes

Like Page

1 friend likes this



If you are an R blogger yourself you are invited to add your own R content feed to this site (**Non-English** R bloggers should add themselves- here)

Jobs for R-users

- Data Scientist
- Survey Associate @ Chicago
- Statistical Programmer @ Portland

- Research Manager @ Chicago
- R Programmer @ NJ (USA)

Search & Hit Enter

Popular Searches

- web scraping
- heatmap
- maps
- twitter
- shiny
- boxplot
- Time Series
- animation
- hadoop
- ggplot2
- how to import image file to R
- ggplot
- trading
- latex
- finance
- excel
- pca
- eclipse
- sql
- RSTUDIO
- googlevis
- quantmod
- market research
- rattle
- knitr
- tutorial
- coplot
- map
- rcmdr
- regression

Recent Posts

- Free data science video lecture: debugging in R
- Try’in to 3D network: Quest (shiny + plotly)
- RcppNumerical: Numerical integration and optimization with Rcpp
- Safe Loading of RData Files

- Half off Win-Vector data science books and video training!
- Mapping Twitter Followers in R
- geom_lollipop() by the Chartettes
- The new and improved pirateplot()! Now with themes!
- Who Came to Vote in Utah's Caucuses?
- In case you missed it: March 2016 roundup
- Election analysis contest entry part 3 – interactive exploration of voting locations with leaflet and Shiny
- Shiny module design patterns: Pass a single input to multiple modules
- d3/R Chord Diagram of White House Petition Data
- One-sided F-tests and halving p-values
- RcppArmadillo 0.6.700.3.0

Other sites

- Statistics of Israel
- Jobs for R-users
- SAS blogs

How to Pimp Your .Rprofile

December 30, 2015
By Kris Eberwein

Like Share 0 Tweet Share

(This article was first published on **R Tricks – Data Science Riot!**, and kindly contributed to R-bloggers)

After you’ve been using R for a little bit, you start to notice people talking about their .Rprofile as if it’s some mythical being. Nothing magical about it, but it can be a big time-saver if you find yourself typing things like, “summary()” or, the ever-hated, “stringasfactors=FALSE” ad nauseam.

Where is my .Rprofile?

The simple answer is, if you don’t know, then you probably don’t have one. R-Studio doesn’t include one unless you tell it to. In Mac and Linux the .Rprofile is usually a hidden file in your user’s home directory. In Windows the most common place is C:Program FilesRRx.xetc.

Check to see if I have an .Rprofile

Before creating a new profile, fire up R and check to see if you have an existing .Rprofile lying around. Like I said, it’s usually a hidden file.

```
c(Sys.getenv("R_PROFILE_USER"), file.path(getwd(),".Rprofile"))
```

How to create an .Rprofile

Assuming you don’t already have one, these files are easy to create. Open a text editor and name your blank file .Rprofile with no trailing extension and place it in the appropriate directory. After populating the file, you’ll have to restart R for the settings to take affect.

Sample .Rprofile

Below is a snapshot of mine. Of coarse, you can make this as simple or as complex as you like.

```
## Print this on start so I know it's loaded.
cat("Loading custom .Rprofile")

## A little gem from Hadley Wickam that will set your CRAN mirror and automatically load devtools in interactive sessions.
.First <- function() {
  options(
    repos = c(CRAN = "https://cran.rstudio.com/"),
    setwd("~/Documents/R"),
    deparse.max.lines = 2)
}

if (interactive()) {
  suppressMessages(require(devtools))
}

## Nice option for local work. I keep it commented out so my code can remain portable.
## options(stringsAsFactors=FALSE)

## Increase the size of my Rhistory file, becasue I like to use the up arrow!
Sys.setenv(R_HISTSIZE='100000')

## Create invisible environment ot hold all your custom functions
.env <- new.env()

## Single character shortcuts for summary() and head().
.env$s <- base::summary
.env$h <- utils::head

#ht==headtail, i.e., show the first and last 10 items of an object.
.env$ht <- function(d) rbind(head(d,10),tail(d,10))

## Read data on clipboard.
.env$read.cb <- function(...) {
  ismac <- Sys.info()[1]=="Darwin"
  if (!ismac) read.table(file="clipboard", ...)
  else read.table(pipe("pbpaste"), ...)
}
```

```
## List objects and classes.
.env$lsa <- function() {
{
    obj_type <- function(x) class(get(x, envir = .GlobalEnv)) # define environment
    foo = data.frame(sapply(ls(envir = .GlobalEnv), obj_type))
    foo$object_name = rownames(foo)
    names(foo)[1] = "class"
    names(foo)[2] = "object"
    return(unrowname(foo))
}

## List all functions in a package.
.env$lsp <-function(package, all.names = FALSE, pattern) {
    package <- deparse(substitute(package))
    ls(
        pos = paste("package", package, sep = ":"),
        all.names = all.names,
        pattern = pattern
    )
}

## Open Finder to the current directory. Mac Only!
.env$macopen <- function(...) if(Sys.info()[1]=="Darwin") system("open .")
.env$o       <- function(...) if(Sys.info()[1]=="Darwin") system("open .")


## Attach all the variables above
attach(.env)

## Finally, a function to print out all the functions you have defined in the .Rprofile.
print.functions <- function(){
        cat("s() - shortcut for summaryn",sep="")
        cat("h() - shortcut for headn",sep="")
        cat("read.cb() - read from clipboardn",sep="")
        cat("lsa() - list objects and classesn",sep="")
        cat("lsp() - list all functions in a packagen",sep="")
        cat("macopen() - open finder to current working directoryn",sep="")
}
```

Limitations and gotchas

The major disadvantage to all this is code portability. For example, if you set your .Rprofile to load `dplyr` on every session, when someone else tries to run your code, it won't work. For this reason, I'm a little picky about my settings, opting for functions that will only be used in interactive sessions.

Related

Sync Your Rprofile Across Multiple R Installations
Sync Your Rprofile Across Multiple R Installations
In "R bloggers"

Customize your .Rprofile and Keep Your Workspace Clean
Like your .bashrc, .vimrc, or many other dotfiles you may have in your home directory,
In "R bloggers"

Fun with .Rprofile and customizing R startup
In "R bloggers"

page: 2

Using segmented regression to analyse world record running times

December 30, 2015
By Andrie de Vries

(This article was first published on **Revolutions**, and kindly contributed to R-bloggers)

by Andrie de Vries

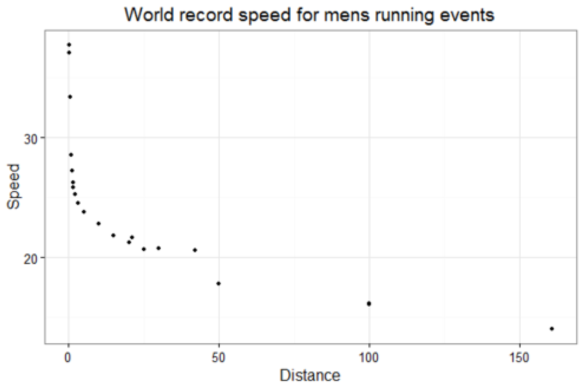
A week ago my high school friend, @XLRunner, sent me a link to the article "How Zach Bitter Ran 100 Miles in Less Than 12 Hours". Zach's effort was rewarded with the American record for the 100 mile event.

Zach Bitter holds the American record for the 100 mile

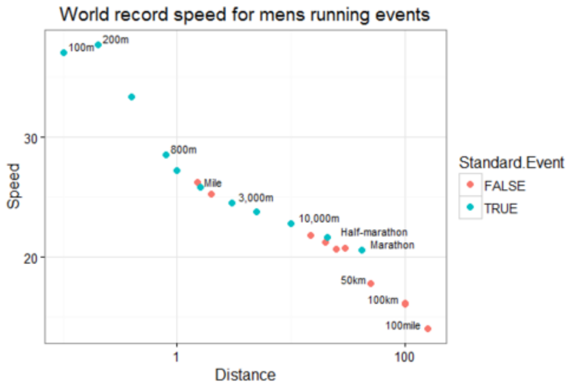
This reminded me of some analysis I did, many years ago, of the world record speeds for various running distances. The International Amateur Athletics Federation (IAAF) keeps track of world records for distances from 100m up to the marathon (42km). The distances longer than 42km do not fall in the IAAF event list, but these are also tracked by various other organisations.

You can find a list of IAAF world records at Wikipedia, and a list of ultramarathon world best times at Wikepedia.

I extracted only the mens running events from these lists, and used R to plot the average running speeds for these records:



You can immediately see that the speed declines very rapidly from the sprint events. Perhaps it would be better to plot this using a logarithmic x-scale, adding some labels at the same time. I also added some colour for what I call standard events – where "standard" is the type of distance you would see regularly at a world championships or olympic games. Thus the mile is "standard", but the 2,000m race is not.



Now our data points are in somewhat more of a straight line, meaning we could consider fitting a linear regression.

However, it seems that there might be two kinks in the line:

- The first kink occurs somewhere between the 800m distance and the mile. It seems that the sprinting distances (and the 800m is sometimes called a long sprint) has different dynamics from the events up to the marathon.
- And then there is another kink for the ultra-marathon distances. The standard marathon is 42.2km, and distances longer than this are called ultramarathons.

Also, note that the speed for the 100m is actually slower than for the 200m. This indicates the transition effect of getting started from a standing start – clearly this plays a large role in the very short sprint distance.

Subsetting the data

For the analysis below, I exlcuded the data for:

- The 100m sprint (transition effects play too large a

role)
- The ultramarahon distances (they get raced less frequently, thus something strange seems to be happening in the data for the 50km race in particular).

Using the segmented package

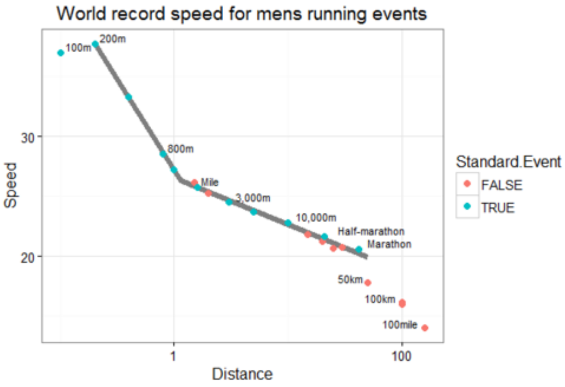
To fit a regression line with kinks, more properly known as a segmented regression (or sometimes called piecewise regression), you can use the `segmented` package, available on CRAN.

The `segmented()` function allows you to modify a fitted object of class `lm` or `glm`, specifying which of the independent variables should have segments (kinks). In my case, I fitted a linear model with a single variable (log of distance), and allowed `segmented()` to find a single kink point.

My analysis indicates that there is a kink point at 1.13km (10^0.055 = 1.13), i.e. between the 800m event and the 1,000m event.

```
> summary(sfit)

***Regression Model with Segmented Relationship(s)***

Call:
segmented.lm(obj = lfit, seg.Z = ~logDistance)

Estimated Break-Point(s):
Est. St.Err
0.055 0.021

Meaningful coefficients of the linear terms:
              Estimate Std. Error  t value Pr(>|t|)
(Intercept)    27.2064     0.1755   155.04 < 2e-16 ***
logDistance -  15.1305     0.4332   -34.93 1.94e-13 ***
U1.logDistance 11.2046     0.4536    24.70 NA
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’
1

Residual standard error: 0.2373 on 12 degrees of freedom
Multiple R-Squared: 0.9981, Adjusted R-squared: 0.9976

Convergence attained in 4 iterations with relative change
-4.922372e-16
```

The final plot shows the same data, but this time with the segmented regression line also displayed.



Conclusion

I conlude:

1. It is really easy to fit a segmented linear regression model using the segmented package
2. There seems to be a different physiological process for the sprint events and the middle distance events. The segmented regression finds this kink point between the 800m event and the 1,000m event
3. The ultramarathon distances have a completely different dynamic. However, it's not clear to me whether this is due to inherent physiological constraints, or vastly reduced competition in these "non-standard" events.
4. The 50km world record seems too "slow". Perhaps the competition for this event is less intense than for the marathon?

Dennis Kimetto holds the world record for the marathon

The code

Here is my code for the analysis:

Comments are closed.

Search & Hit Enter

Recent popular posts

- Try'in to 3D network: Quest (shiny + plotly)
- Free data science video lecture: debugging in R

Most visited articles of the week

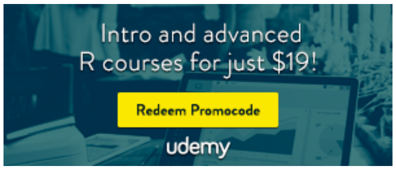
1. Installing R packages
2. R tutorials
3. In-depth introduction to machine learning in 15 hours of expert videos
4. Using apply, sapply, lapply in R
5. How to write the first for loop in R
6. How to perform a Logistic Regression in R
7. Computing and visualizing PCA in R
8. A quick introduction to machine learning in R with caret
9. AirbnB uses R to scale data science

Sponsors

R training
R consulting
QUANTIDE
knowledge from data

New in RStudio Server Pro
Collaboration | Multiple Sessions | R Versions
R Studio
Free Evaluation

Highland Statistics Ltd
Zero Inflated Models & GLMM
Beginner's Guide to GAM
Beginner's Guide to GLM & GLMM
Beginner's Guide to GAMM

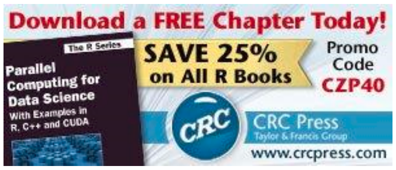

Intro and advanced
R courses for just $19!
Redeem Promocode
udemy

Werden Sie zum Expe[R]ten mit der
R-Akademie von
eoda
daten ▸ wissen ▸ nutzen
Beratung I Software
Training I Lösungen

STATISTICS
VIEWS
Bringing Statistics Together

BECOME A DATA SCIENTIST
NYC DATA SCIENCE ACADEMY
12 Weeks to A New Career

#ODSC
BOSTON MAY 20-22
TICKETS ON SALE NOW


Download a FREE Chapter Today!
The R Series
Parallel Computing for Data Science With Examples in R, C++ and CUDA
SAVE 25% on All R Books
Promo Code CZP40
CRC Press Taylor & Francis Group
www.crcpress.com

Jobs for R users

- Data Scientist
- Survey Associate @ Chicago
- Statistical Programmer @ Portland
- Research Manager @ Chicago
- R Programmer @ NJ (USA)
- Research Fellow in Spatial Epidemiology
- R programmer with knowledge of S4 class structure

Search & Hit Enter

Full list of contributing R-bloggers

R-bloggers was founded by Tal Galili, with gratitude to the R community.
Is powered by WordPress using a bavotasan.com design.