Programmering i R

Pär Leijonhufvud

(October 12, 2022)

Översikt

- Varför skript?
- Några funktioner som är användbara i skriot
- Hur man skriver skript som
  - man kan förstå en vecka senare
  - uppdatera utan att börja om från början
- Hur man visar eller sparar resultatet

Fördelar med skript

- Man kan tänka igeom vad man vill göra, planera det steg gör steg
- Man kan återvända sina egna lösningar i andra projekt
- Det är enklare att t.ex. ta ut "samma" data varje månad
- Man har en dokumentation på vad man faktiskt gjorde.

Allmänt om programmering – goda vanor ger bra resultat!

- Kommentera din kod!
- Döp variabler, funktioner, tabeller, osv på ett strukturerat sätt
- Ha en struktur på dina skript som gör att du vet vad som kommer vart.

Namn

Dåligt df, df1, plot1, plot2, f1, f2, datum1, data.xlsx

Bra

- Tabeller:
  - acovid, acovid.ÖSJ, acovid.ÖSJ.2021...
  - DiagnoserSjukhusetInneliggande
- Grafer
  - plot.vårdtillfällen.vecka
  - plot_vårdplatser_2021
- Filer
  - vårdtillfällen_ÖSJ_2021Q2.xlsx

Hur kan ett skript vara uppdelat?

1. Inledande kommentar om varför, sammanhang, vem, när. TODO, uppdateringar.
2. libraries()
3. Skapa t.ex. namn för exportfiler, importfiler med förutsägbara namn (besökare-bräcke-2021Q3.csv)
4. Ladda in data: CSV, XLSX, webb. . .
5. Städa
6. utför analys, skapa grafer, osv
7. Spara filer, visa enkla data på skärmen

Om R-skript

- skriptnamn.R
- Kör med `source("skriptnamn.R, encoding="UTF-8")`
- Kommentarer: allt på en rad efter "#"
- Indentera din kod så blir den mer läslig

Ladda in data

CSV

- ```
  acov <-
  read.csv2("acov_GDPR.csv",
  fileEncoding = "UTF-8-BOM")
  ```
- ```
  readr::read_csv2(file =
  "acov_test.csv", col_names =
  TRUE, na=c("", "-"), skip = 2)
  ```

Excel

```
xlsx::read.xlsx2(file =
"foobar.xlsx", sheetIndex = 1)
```

URL

```
data <-
read.csv2("http://some.where.net/data/
```

IF-satser

Alternativa aktioner, kan bygga långa kedjor med `else if()`

```
if (Sys.Date() > "2022-10-01"){
        print("Efter")
} else {
        print("Före")
}
```

IF-satser

```
if(Sys.Date() > "2022-10-12"){
    print("Efter 2022-10-12")
} else if (Sys.Date() < "2022-10-12"){
    print("Efter 2022-10-12")
} else {
    print("Idag är det 2022-10-12")
}
.] "Idag är det 2022-10-12"
```

Funktioner: dina egna "kommandon"

- Skapa ett eget kommando som en funktion
- Bra om samma behandling skall göras med olika input flera gånger (manuellt eller i ett skript)

```
> DagarTillJul <- function(datum){as.Date("2022-12-24") -
> DagarTillJul("2022-10-12")
Time difference of 73 days
>
> HurMångaDagar <- function(datum1, datum2){as.Date(datum
as.Date(datum2)}
> HurMångaDagar("2022-10-10", "1969-07-20")
Time difference of 19440 days
```

Upprepa: for

När du vill upprepa något för t.ex. alla avdelningar, patientgrupper, osv.

```
> for (i in 1:5){
+ print(i)
+ }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

Upprepa: for

```
# ladda in datafilen, och snabbstäda
acov <- read.csv2("acov_GDPR.csv", fileEncoding = "UTF-8-BOM")
acov$Ålder <- as.numeric(gsub("," , ".",  acov$Ålder..År.), na.rm=TRUE)

## Warning: NAs introduced by coercion

for ( kundgrupp in unique(acov$Beställare)) {
print(
      paste0(
            "För ", kundgrupp, " var medianåldern ",
            median(filter(acov, Beställare == kundgrupp)$Ålder, na.rm=T
            " och i genomsnitt analysvärdet ",
            round(mean(filter(acov, Beställare == kundgrupp)$Mätvärde,
na.rm=TRUE ),1))

      )
}

## [1] "För ÖSJ var medianåldern 60.28 och i genomsnitt analysvärdet 5.2
## [1] "För VC.GLESBYGD var medianåldern 52.3 och i genomsnitt analysvär
10.1"
## [1] "För VC.Östersund var medianåldern 51.41 och i genomsnitt analysv
6.1"
## [1] "För NA var medianåldern NA och i genomsnitt analysvärdet NaN"
```

Utför tills: while

```
> räknare <- 0
> while(räknare < 5) {
+   print(räknare)
+   räknare <- räknare + 1
+ }
[1] 0
[1] 1
[1] 2
[1] 3
[1] 4
```

Utför tills: while, exempel 2

```
> räknare <- 1
> while(räknare > 0) {
+   print(räknare)
+   räknare <- räknare + 1
+ }
```

Ser ni problemet?

Utför tills: while, exempel 2

```
> räknare <- 1
> while(räknare > 0) {
+   print(räknare)
+   räknare <- räknare + 1
+ }
[1] 0
[1] 1
[1] 2
[1] 3
[1] 4
...
[1] 33698
>
```

Spara data som CSV eller XLSX

Spara en CSV med `write.csv2(objekt , file="filnamn.csv")` eller med `readr::write_csv2(x=acov.ÖSJ, file = "foobar.csv", append = TRUE, progre TRUE)`

Spara en excel-fil med `xlsx::write.xlsx2(acov.ÖSJ, file = "foobar.xlsx", sheetName = "ÖSJ", append=TRUE)`.

RMarkdown – skriptet

```
---
title: "Ledtid, sammafattning"
output:
  #html_document:
   # toc: true
  word_document:
    #toc: true
  #md_document: default
---


Sammanfattning av svarstider för analyser på CobasPro under perioden
2021-01-01 -- 2022-09-30. Q90, Q95 samt Q99 är respektive kvartiler.
Alla tider är i minuter.



```{r echo = FALSE, results='asis'}

library(knitr)
knitr::kable(head(TAT.summary.cobas[,  c(1,2,3, 4,9,10,11)]), "pipe")
```

Mitt förslag är därför att vi anger förväntat svarstid enligt följande:

* C-analyser
`r ceiling(TAT.summary.cobas[1,9][[1]]/60) `h
* E-analyser
`r ceiling(TAT.summary.cobas[2,9][[1]]/60)`h.
* För virusanalyserna
föreslår jag istället `r ceiling(TAT.summary.cobas[3,9][[1]]/(24*60))` dygn.

Vi kan ange att vi historisk lämnat ut över 90% av svaren innom dessa
tidsgränser.
```

RMarkdown – resultatet

Ledtid, sammafattning

Sammanfattning av svarstider för analyser på CobasPro under perioden 2021-01-01 – 2022-09-30. Q90, Q95 samt Q99 är respektive kvartiler. Alla tider är i minuter.

Typ	Antal	Medel	Median	Q90	Q95	Q99
C-analys	1821738	130	61	225	312	1489
E-analys	260794	166	94	281	366	1304
E-Virus	2276	1035	340	2770	4188	8364

Mitt förslag är därför att vi anger förväntat svarstid enligt följande:

- C-analyser 4h
- E-analyser 5h.
- För virusanalyserna föreslår jag istället 2 dygn.

Vi kan ange att vi historisk lämnat ut över 90% av svaren innom dessa tidsgränser.

Ett exempel

Förväntad-ledtid.R + (P:\Statistik\förväntad-ledtid) - GVIM6

```
# Sammanställa våra verkliga svarstider (TAT, Turn Around Time) på
# samtliga analyser.
#
# Drugt 6 miljoner rader initialt, körtid mindre än 1 minut (mest
# importen)
#
# Pär Leijonhufvud/2022-10-05
#
# Lade till sortering på CobasPro-modul PL/2022-10-10


library(tidyverse)
library(xlsx)
library(knitr)
library(rmarkdown)
```

```r
## Import data
files <- list.files(path = ".", pattern = "20[0-9]+-allt.csv" )

allt.21.22 <- read_csv2(files)

# Lista över analyser på C- resp E-modul på Cobas
analyser.kodade <- read.csv2("analyser-kodade.csv", fileEncoding = "UTF-8-BOM")


## Data cleaning

# ledtid till period
allt.21.22$Ledtid2 <-  lubridate::hms(allt.21.22$Ledtid)

# Ankomsttid som datum
allt.21.22$Ankomsttid <-   lubridate::ymd_hm(allt.21.22$Ankomsttid)

# ledtid till sekunder och minuter
allt.21.22$Ledtid.s <- lubridate::period_to_seconds(allt.21.22$Ledtid2)
allt.21.22$Ledtid.m <- round(lubridate::period_to_seconds(allt.21.22$Ledtid2)/60
, 0)

# Städa bort skräp: beställa eller ankomna mer än en månad före perioden
allt.21.22 <- filter(allt.21.22,
                     Registreringstid > "2020-12-01" &
                     Ankomsttid > "2020-12-01" &
                     !is.na(Mätvärde) &
                     !is.na(Ledtid) &
                     !is.na(Ledtid.m) &
                     Ledtid != "-"
)
```

```
# Bara Cobas, men Typ == modul på Cobas, eller "E-Virus" för
# virusanalyser
allt.21.22.cobas <- merge(allt.21.22, analyser.kodade, by = "Analys")
```

```r
## Summera
# Med avrundade värden: ingen bryr sig om den 5:e decimalen på
# 200+minuter
TAT.summary <- allt.21.22.cobas %>%
        group_by(Analys) %>%
        summarise(
                Antal = length(Ledtid.s),
                Medel = round(mean(Ledtid.m, na.rm=TRUE), 0),
                Median = round(median(Ledtid.m, na.rm=TRUE), 0),
                Min = min(Ledtid.m, na.rm=TRUE),
                Max = max(Ledtid.m, na.rm=TRUE),
                Q75 = round(quantile(Ledtid.m, 0.75, na.rm=TRUE), 0),
                Q90 = round(quantile(Ledtid.m, 0.90, na.rm=TRUE), 0),
                Q95 = round(quantile(Ledtid.m, 0.95, na.rm=TRUE), 0),
                Q99 = round(quantile(Ledtid.m, 0.99, na.rm=TRUE), 0)
        )


TAT.summary.modul <- allt.21.22.cobas %>%
        group_by(Typ) %>%
        summarise(
                Antal = length(Ledtid.s),
                Medel = round(mean(Ledtid.m, na.rm=TRUE), 0),
                Median = round(median(Ledtid.m, na.rm=TRUE), 0),
                Min = min(Ledtid.m, na.rm=TRUE),
                Max = max(Ledtid.m, na.rm=TRUE),
                Q75 = round(quantile(Ledtid.m, 0.75, na.rm=TRUE), 0),
                Q90 = round(quantile(Ledtid.m, 0.90, na.rm=TRUE), 0),
                Q95 = round(quantile(Ledtid.m, 0.95, na.rm=TRUE), 0),
                Q99 = round(quantile(Ledtid.m, 0.99, na.rm=TRUE), 0)
        )
```

```r
# Exportera: NB: kommer att skriva över filen!
write.xlsx2(TAT.summary.modul, file="Tat-tider_21-22_minuter_typ2.xlsx",
            sheetName="TAT-tider 2021-22 Modul")

write.xlsx2(as.data.frame(TAT.summary), file="Tat-tider_21-22_minuter_typ2.xlsx"
            sheetName="TAT-tider 2021-22 analyser", append=TRUE)

write.xlsx2(analyser.kodade, file="Tat-tider_21-22_minuter_typ2.xlsx",
            sheetName="CobasPro-analyser moduler", append=TRUE)
```

Tabell på skärmen

```
> tab1 <- ftable(acov$Beställare, acov$Kön)
> print(tab1)
                   K    M

VC.GLESBYGD     3201 1368
VC.Östersund    2881 1185
ÖSJ             5308 3993
```