Einführung in R - Housekeeping

Jan-Philipp Kolb

13 Juni, 2019

Pakete automatisch installieren

- Mit `installed.packages()` kann ich herausfinden, welche Pakete installiert sind.
- Bei mir sind es momentan 332 Pakete.

```
my_packages <- installed.packages()
mypack <- my_packages[,"Package"]
```

```
my_packages[,"Package"]
```

```
##              abind             acepack                 AER
##           "abind"           "acepack"               "AER"
##       AmesHousing            antiword             aplpack
##     "AmesHousing"          "antiword"           "aplpack"
##               arm             askpass          assertthat
##             "arm"           "askpass"        "assertthat"
##         backports                 base          base64enc
##       "backports"               "base"        "base64enc"
##        bayestestR             beanplot                  BH
##      "bayestestR"           "beanplot"                "BH"
##              bindr             bindrcpp               bitops
##            "bindr"           "bindrcpp"             "bitops"
```

Fehlende Pakete installieren

- Zunächst wird eine Liste (bspw. `packlist`) mit den benötigten Paketen erstellt.
- Dann wird geprüft ob diese Pakete unter den installierten Paketen sind.
- Zuletzt werden Pakete installiert, die noch nicht vorhanden sind.

```
packlist <- c("ggplot2", "Rcpp")
new.packages <- packlist[!(packlist %in% mypack)]
if(length(new.packages)) install.packages(new.packages)
```

Codeblöcke zusammenklappen

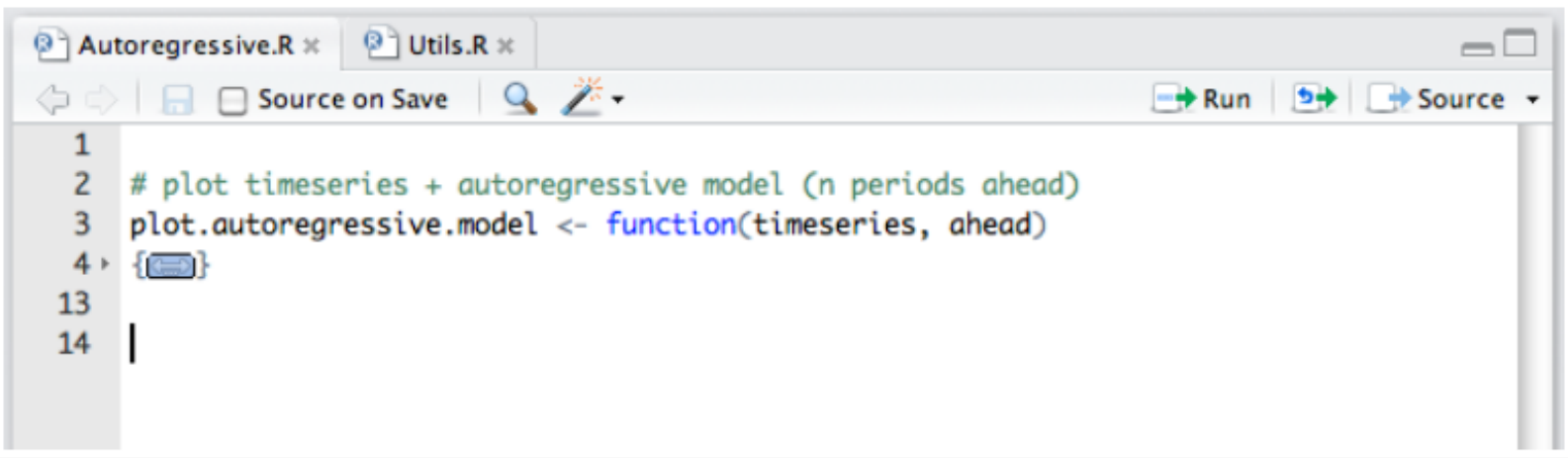
RStudio Support
April 10, 2019 09:51

Follow

Code Folding and Sections

Code Folding

RStudio supports both automatic and user-defined folding for regions of code. Code folding allows you to easily show and hide blocks of code to make it easier to navigate your source file and focus on the coding task at hand. For example, in the following source file the body of the `plot.autoregressive.model` has been folded:



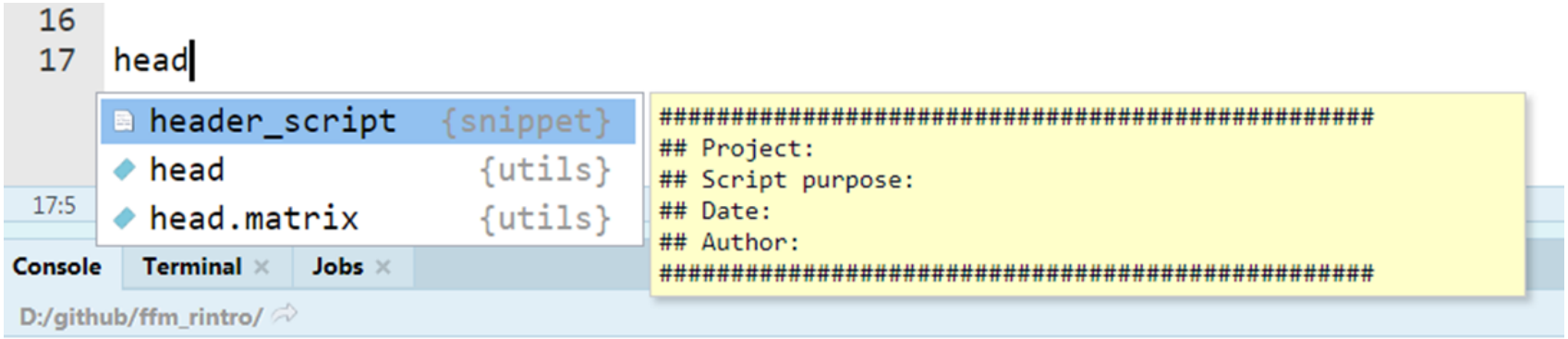
Die Funktion source

- Wenn das Argument `echo=T` gesetzt ist, sieht man auch was innerhalb des Skripts passiert.
- Die Funktion eignet sich, bspw. wenn immer wieder die gleichen Arbeitsschritte durchgeführt werden, die selten verändert werden müssen.

```
source("../rcode/load_packages.R",echo=T)
```

Snippets - für einen automatischen Header

Tools > Golbal Options



Shapefiles herunterladen

Shapefiles importieren

```
ffm_shp <- rgdal::readOGR(
  "../data/Stadtteile_Frankfurt_am_Main.shp")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "D:\github\ffm_rintro\data\Stadtteile_Frankfurt_am_Ma
## with 46 features
## It has 2 fields
```

Der Datenslot

```
head(ffm_shp@data)
```

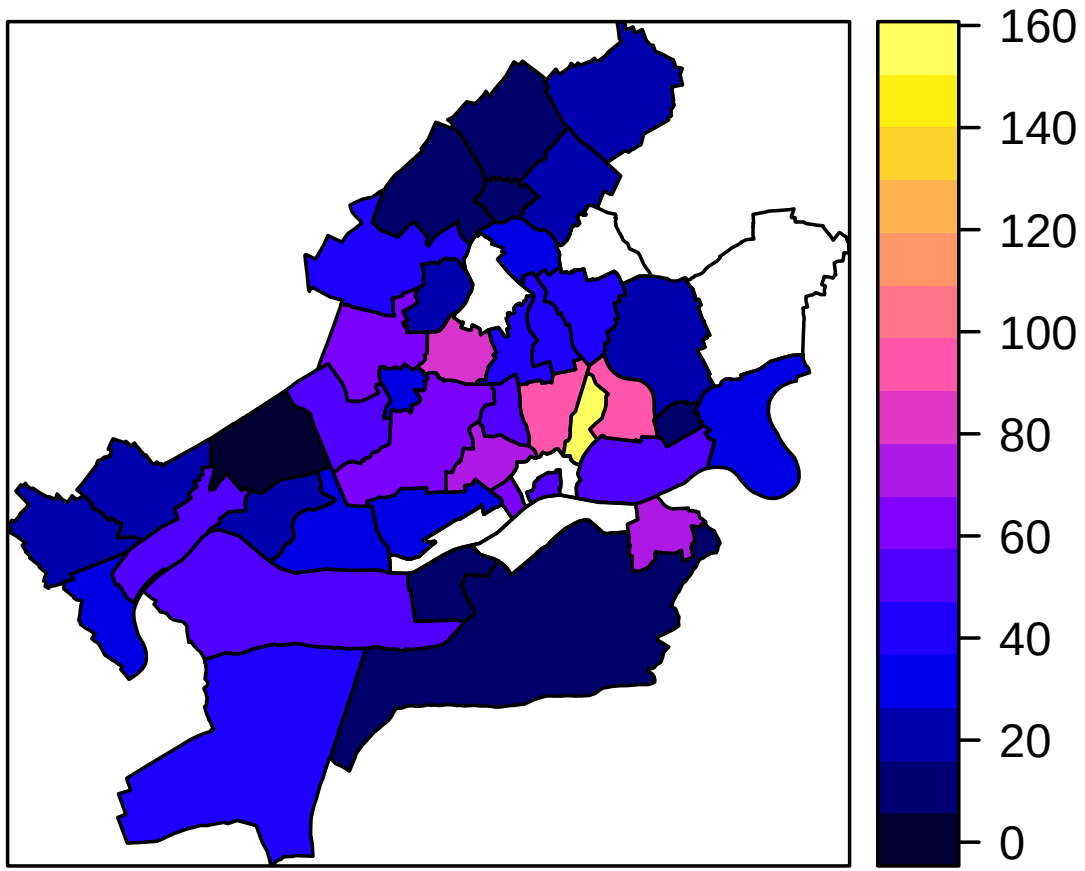
```
##   STTLNR          STTLNAME
## 0      1          Altstadt
## 1      2        Innenstadt
## 2      3 Bahnhofsviertel
## 3      4     Westend-SÃ¼d
## 4      5     Westend-Nord
## 5      6     Nordend-West
```

Inhaltliche Daten hinzufügen

```
dat <- read.csv2("../data/bauenwohnen.csv")
ffm_shp@data$Einwohnerdichte <-
  dat$Wohnumfeld...öffentlicher.Raum.Einwohnerdichte.je.ha.2012
```

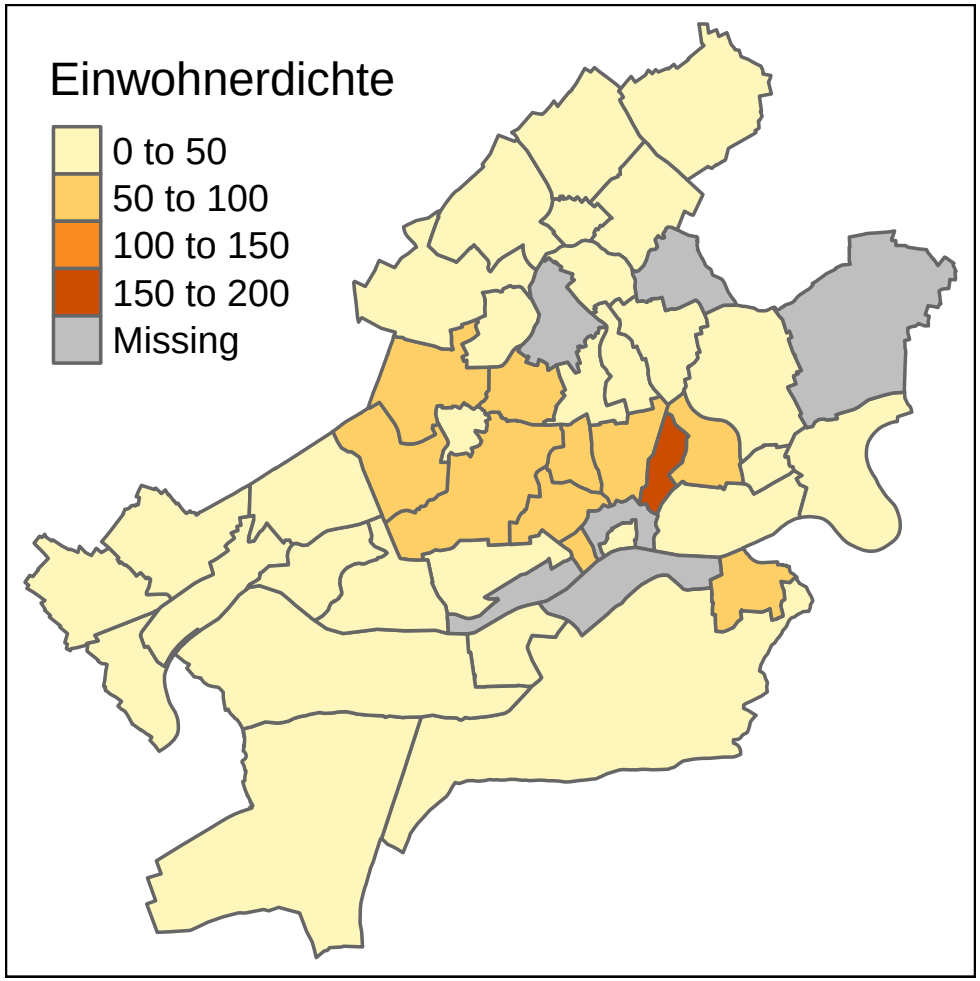
Eine thematische Karte plotten

```
sp::spplot(ffm_shp,"Einwohnerdichte")
```



Thematische Karte mit tmap

```
tmap::qtm(ffm_shp,"Einwohnerdichte")
```



Die Daten matchen

- Indikatoren matchen und hoffen, dass keine NA's auftauchen.

```
ind <- match(ffm_shp$STTLNAME,dat$Stadtteil)
ind
```

```
##  [1]  1  2  3 NA  5  6  7  8  9 10 11 12 NA NA NA 15 16 17 18
## [24] 23 24 25 26 27 28 29 30 31 32 33 NA 35 36 37 38 39 40 41
```

Wo hakt es?

```
dat_ffm <- data.frame(ffm_shp$STTLNAME,dat$Stadtteil)
dat_ffm[is.na(ind),]
```

```
##    ffm_shp.STTLNAME      dat.Stadtteil
## 4      Westend-SÃ¼d        Westend-Süd
## 13 Sachsenhausen-N. Sachsenhausen-Nord
## 14 Sachsenhausen-S.  Sachsenhausen-Süd
## 15        Flughafen            Oberrad
## 20       RÃ¶delheim             Hausen
## 35          HÃ¶chst               Nied
```

Werte manuell zuordnen

```
ffm_shp$STTLNAME[which(is.na(ind))]
```

```
## [1] Westend-SÃ¼d      Sachsenhausen-N. Sachsenhausen-S. Flugha
## [5] RÃ¶delheim       HÃ¶chst
## 46 Levels: Altstadt Bahnhofsviertel Bergen-Enkheim ... Zeilsh
```

```
nas1 <- ffm_shp$STTLNAME[which(is.na(ind))][1]
agrep(nas1,dat$Stadtteil)
```

```
## [1] 4
```

```
ind[which(is.na(ind))[1]] <- agrep(nas1,dat$Stadtteil)
```

Belabelter SPSS Datensatz

```
data_path <- "D:/gitlab/IntroDataAnalysis/data/"
gpdat <- foreign::read.spss(paste0(data_path,
            "ZA5666_v1-0-0.sav"),to.data.frame=TRUE)
att_dat <- attributes(gpdat)
names(att_dat)
att_dat$variable.labels
```

Destatis Daten

```
devtools::install_github("cutterkom/destatiscleanr")
```

```
library(destatiscleanr)
```

```
$codepage
[1] 65001

> names(att_dat)
[1] "names"
> destatiscleanr::
```

clean_header {destatiscleanr}
convert_columns_to_numeric {destatiscleanr}
delete_copyright {destatiscleanr}
destatiscleanr {destatiscleanr}
read_file {destatiscleanr}

```
read_file(file)
This functions reads the csv file by using German decimal marks
Press F1 for additional help
```

```
e.labels" "codepage"
```