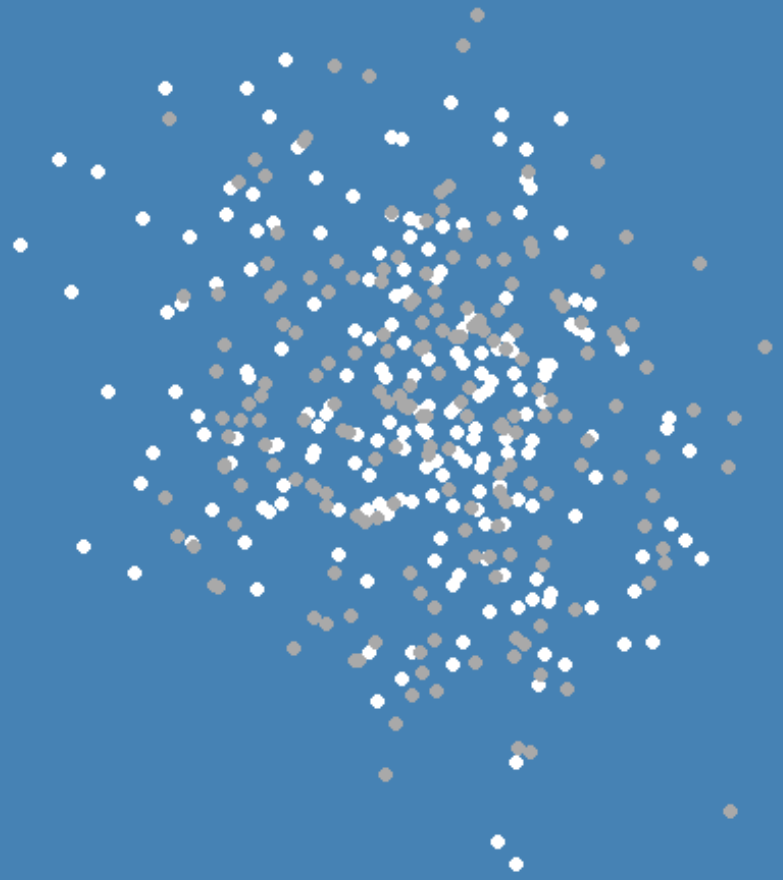
Introduction to R

2.3 Re-Naming and Re-Ordering Rows and Variables

rename(), relocate(), arrange()

Lion Behrens, M.Sc.



UNIVERSITY OF MANNHEIM

University of Mannheim
Chair of Social Data Science and Methodology
Chair of Quantitative Methods in the Social Sciences

Data Import

```
library(haven)
ess10 <- haven::read_dta("./dat/ESS10.dta")
dim(ess10) # check dimensionality of data frame
```

```
## [1] 18060   513
```

Data Import

```
library(haven)
ess10 <- haven::read_dta("./dat/ESS10.dta")
colnames(ess10)[1:50]
```

```
##  [1] "name"     "essround" "edition"  "proddate" "idno"     "cntry"
##  [7] "dweight"  "pweight"  "nwspol"   "netusoft" "netustm"  "ppltrst"
## [13] "pplfair"  "pplhlp"   "polintr"  "psppsgva" "actrolga" "psppipla"
## [19] "cptppola" "trstprl"  "trstlgl"  "trstplc"  "trstplt"  "trstprt"
## [25] "trstep"   "trstun"   "trstsci"  "vote"     "prtvtebg" "prtvtbhr"
## [31] "prtvtecz" "prtvthee" "prtvtefi" "prtvtefr" "prtvtghu" "prtvclt1"
## [37] "prtvclt2" "prtvclt3" "prtvtfsi" "prtvtesk" "contplt"  "donprty"
## [43] "badge"    "sgnptit"  "pbldmna"  "bctprd"   "pstplonl" "volunfp"
## [49] "clsprty"  "prtclebg"
```

Data Import

```
print(ess10[1:15, 1:10])
```

```
## # A tibble: 15 × 10
##    name        essro…¹ edition prodd…²  idno cntry dweight pweight nwspol netus…³
##    <chr>         <dbl> <chr>   <chr>   <dbl> <chr>   <dbl>   <dbl> <dbl+> <dbl+l>
##  1 ESS10e01_2       10 1.2     28.06.… 10002 BG      1.03    0.218  80    1 [Nev…
##  2 ESS10e01_2       10 1.2     28.06.… 10006 BG      0.879   0.218  63    5 [Eve…
##  3 ESS10e01_2       10 1.2     28.06.… 10009 BG      1.01    0.218 390    5 [Eve…
##  4 ESS10e01_2       10 1.2     28.06.… 10024 BG      0.955   0.218  60    5 [Eve…
##  5 ESS10e01_2       10 1.2     28.06.… 10027 BG      0.841   0.218 120    5 [Eve…
##  6 ESS10e01_2       10 1.2     28.06.… 10048 BG      0.946   0.218  60    5 [Eve…
##  7 ESS10e01_2       10 1.2     28.06.… 10053 BG      1.01    0.218  30    5 [Eve…
##  8 ESS10e01_2       10 1.2     28.06.… 10055 BG      1.03    0.218  70    5 [Eve…
##  9 ESS10e01_2       10 1.2     28.06.… 10059 BG      0.991   0.218  60    1 [Nev…
## 10 ESS10e01_2       10 1.2     28.06.… 10061 BG      1.05    0.218  60    1 [Nev…
## 11 ESS10e01_2       10 1.2     28.06.… 10064 BG      1.00    0.218 300    5 [Eve…
## 12 ESS10e01_2       10 1.2     28.06.… 10068 BG      1.03    0.218   0    1 [Nev…
## 13 ESS10e01_2       10 1.2     28.06.… 10071 BG      0.931   0.218  30    5 [Eve…
## 14 ESS10e01_2       10 1.2     28.06.… 10077 BG      0.991   0.218  30    1 [Nev…
## 15 ESS10e01_2       10 1.2     28.06.… 10078 BG      0.990   0.218  60    1 [Nev…
## # … with abbreviated variable names ¹essround, ²proddate, ³netusoft
```

Renaming Variables

Renaming Variables

We often want to assign more meaningful names to individual columns/variables within our dataset. We aim at short and informative variable names.

This is how you inspect current variable names:

```
names(ess10)[20:27]
colnames(ess10)[20:27] # equivalent
```

```
## [1] "trstprl" "trstlgl" "trstplc" "trstplt" "trstprt" "trstep"  "trstun"
## [8] "trstsci"
```

And here is how you can assign new variable names:

base R

```
names(ess10)[20:27] <- c("trst_parliament", "trst_legalsys", "trst_police", "trst_politicians",
                         "trst_parties", "trst_EP", "trst_UN", "trst_scientists")

colnames(ess10)[20:27] <- c("trst_parliament", "trst_legalsys", "trst_police", "trst_politicians",
                         "trst_parties", "trst_EP", "trst_UN", "trst_scientists") # equivalent
```

Renaming Variables

We often want to assign more meaningful names to individual columns/variables within our dataset. We aim at short and informative variable names.

This is how you inspect current variable names:

```
names(ess10)[20:27]
colnames(ess10)[20:27] # equivalent
```

```
## [1] "trstprl" "trstlgl" "trstplc" "trstplt" "trstprt" "trstep"  "trstun"
## [8] "trstsci"
```

And here is how you can assign new variable names:

dplyr

```
ess10 <- ess10 %>%
  rename(trst_parliament = trstprl,
         trst_legalsys = trstlgl,
         trst_police = trstplc,
         trst_politicians = trstplt)
```

Moving the Position of Variables

Relocating variables

The positions of variables in your dataset does not matter for statistical modeling or visualization. Still, you might want to change them.

In the European Social Survey Wave 10, there is a variable called ppltrust:

"Generally speaking, would you say that most people can be trusted or that you can't be too careful in dealing with people?".

Relocating variables

The positions of variables in your dataset does not matter for statistical modeling or visualization. Still, you might want to change them.

In the European Social Survey Wave 10, there is a variable called ppltrust:

"Generally speaking, would you say that most people can be trusted or that you can't be too careful in dealing with people?".

- currently, this variable is stored at a different place in the dataset than the variables measuring trust in political institutions

```
which(names(ess10) == "ppltrst") # position of variable ppltrust
```

```
## [1] 12
```

```
which(substr(names(ess10), 1, 4) == "trst") # position of other trust variables
```

```
## [1] 20 21 22 23 24 25 26 27
```

Relocating variables

The positions of variables in your dataset does not matter for statistical modeling or visualization. Still, you might want to change them.

In the European Social Survey Wave 10, there is a variable called ppltrust:

"Generally speaking, would you say that most people can be trusted or that you can't be too careful in dealing with people?".

- let's say we want to move this variable right behind the other trust variables: relocate()

```
ess10 <- ess10 %>%
  relocate(ppltrst, .after = trstsci)

names(ess10)[19:27]
```

```
## [1] "trstprl" "trstlgl" "trstplc" "trstplt" "trstprt" "trstep"  "trstun"
## [8] "trstsci" "ppltrst"
```

Relocating variables

The positions of variables in your dataset does not matter for statistical modeling or visualization. Still, you might want to change them.

In the European Social Survey Wave 10, there is a variable called ppltrust:

"Generally speaking, would you say that most people can be trusted or that you can't be too careful in dealing with people?".

- or before the other trust variables: relocate()

```
ess10 <- ess10 %>%
  relocate(ppltrst, .before = trstprl)

names(ess10)[19:27]
```

```
## [1] "ppltrst" "trstprl" "trstlgl" "trstplc" "trstplt" "trstprt" "trstep"
## [8] "trstun"  "trstsci"
```

Ordering Rows

Re-Arranging Order of Rows - base R

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police:

```
# only sort by trust in police
ess10 <- ess10[order(ess10$trstplc),]
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

Re-Arranging Order of Rows - base R

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police:

```
# only sort by trust in police
ess10 <- ess10[order(ess10$trstplc),]
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

```
## # A tibble: 10 × 7
##    trstplc              agea      trstprt          trstep  trstun  trstsci vote
##    <dbl+lbl>            <dbl+lbl> <dbl+lbl>       <dbl+l> <dbl+l> <dbl+l> <dbl+l>
##  1 0 [No trust at all] 70        0 [No trust at… 0 [No … 0 [No … 3 [3]   1 [Yes]
##  2 0 [No trust at all] 31        0 [No trust at… 5 [5]   3 [3]   5 [5]   2 [No]
##  3 0 [No trust at all] 52        1 [1]           1 [1]   0 [No … 1 [1]   2 [No]
##  4 0 [No trust at all] 78        4 [4]           0 [No … 0 [No … 1 [1]   1 [Yes]
##  5 0 [No trust at all] 65        0 [No trust at… 0 [No … 0 [No … 4 [4]   1 [Yes]
##  6 0 [No trust at all] 21        0 [No trust at… 0 [No … 0 [No … 0 [No … 2 [No]
##  7 0 [No trust at all] 40        0 [No trust at… 7 [7]   7 [7]   7 [7]   1 [Yes]
##  8 0 [No trust at all] 47        0 [No trust at… 0 [No … 0 [No … 0 [No … 1 [Yes]
##  9 0 [No trust at all] 49        3 [3]           4 [4]   4 [4]   5 [5]   2 [No]
## 10 0 [No trust at all] 52        0 [No trust at… 5 [5]   5 [5]   5 [5]   1 [Yes]
```

Re-Arranging Order of Rows - base R

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police and age:

```
# sort by trust in police and age
ess10 <- ess10[order(ess10$trstplc, ess10$agea),]
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

Re-Arranging Order of Rows - base R

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police and age:

```
# sort by trust in police and age
ess10 <- ess10[order(ess10$trstplc, ess10$agea),]
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

```
## # A tibble: 10 × 7
##    trstplc              agea  trstprt trstep      trstun      trstsci     vote
##    <dbl+lbl>            <dbl> <dbl+l> <dbl+lbl>   <dbl+lbl>   <dbl+lbl>   <dbl+l>
##  1 0 [No trust at all] 15    0 [No … NA(b)       NA(b)           3 [3]   3 [Not…
##  2 0 [No trust at all] 15    0 [No …     0 [No …     0 [No …     0 [No … 3 [Not…
##  3 0 [No trust at all] 15    0 [No …     0 [No …     4 [4]       0 [No … 3 [Not…
##  4 0 [No trust at all] 16    2 [2]       2 [2]       8 [8]       9 [9]   2 [No]
##  5 0 [No trust at all] 16    0 [No … NA(b)       NA(b)       NA(b)       2 [No]
##  6 0 [No trust at all] 16    0 [No …     5 [5]   NA(b)           5 [5]   2 [No]
##  7 0 [No trust at all] 17    0 [No …     0 [No …     0 [No …     0 [No … 3 [Not…
##  8 0 [No trust at all] 17    0 [No …     0 [No …     0 [No … NA(c)       3 [Not…
##  9 0 [No trust at all] 17    1 [1]       3 [3]       5 [5]       4 [4]   2 [No]
## 10 0 [No trust at all] 18    0 [No …     4 [4]       7 [7]       8 [8]   2 [No]
```

Re-Arranging Order of Rows - dplyr

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police:

```
# only sort by trust in police
ess10 <- ess10 %>%
  arrange(trstplc)
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

Re-Arranging Order of Rows - dplyr

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police:

```
# only sort by trust in police
ess10 <- ess10 %>%
  arrange(trstplc)
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

```
## # A tibble: 10 × 7
##    trstplc              agea      trstprt          trstep  trstun  trstsci vote
##    <dbl+lbl>            <dbl+lbl> <dbl+lbl>       <dbl+l> <dbl+l> <dbl+l> <dbl+l>
##  1 0 [No trust at all] 70        0 [No trust at… 0 [No … 0 [No … 3 [3]   1 [Yes]
##  2 0 [No trust at all] 31        0 [No trust at… 5 [5]   3 [3]   5 [5]   2 [No]
##  3 0 [No trust at all] 52        1 [1]           1 [1]   0 [No … 1 [1]   2 [No]
##  4 0 [No trust at all] 78        4 [4]           0 [No … 0 [No … 1 [1]   1 [Yes]
##  5 0 [No trust at all] 65        0 [No trust at… 0 [No … 0 [No … 4 [4]   1 [Yes]
##  6 0 [No trust at all] 21        0 [No trust at… 0 [No … 0 [No … 0 [No … 2 [No]
##  7 0 [No trust at all] 40        0 [No trust at… 7 [7]   7 [7]   7 [7]   1 [Yes]
##  8 0 [No trust at all] 47        0 [No trust at… 0 [No … 0 [No … 0 [No … 1 [Yes]
##  9 0 [No trust at all] 49        3 [3]           4 [4]   4 [4]   5 [5]   2 [No]
## 10 0 [No trust at all] 52        0 [No trust at… 5 [5]   5 [5]   5 [5]   1 [Yes]
```

Re-Arranging Order of Rows - dplyr

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police and age:

```
# only sort by trust in police and age
ess10 <- ess10 %>%
  arrange(trstplc, age)
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

Re-Arranging Order of Rows - dplyr

Re-arranging the order of rows (or columns) does not change the outcome of your statistical analyses or your visualizations. Still, you can re-arrange the order of rows in a dataset.

Sort rows of ess10 by individuals' level of trust in the police and age:

```
# only sort by trust in police and age
ess10 <- ess10 %>%
  arrange(trstplc, agea)
print(ess10[1:10, c("trstplc", "agea", "trstprt", "trstep", "trstun", "trstsci", "vote")])
```

```
## # A tibble: 10 × 7
##    trstplc              agea  trstprt trstep      trstun      trstsci     vote
##    <dbl+lbl>            <dbl> <dbl+l> <dbl+lbl>   <dbl+lbl>   <dbl+lbl>   <dbl+l>
##  1 0 [No trust at all] 15    0 [No … NA(b)       NA(b)           3 [3]   3 [Not…
##  2 0 [No trust at all] 15    0 [No …     0 [No …     0 [No …     0 [No … 3 [Not…
##  3 0 [No trust at all] 15    0 [No …     0 [No …     4 [4]       0 [No … 3 [Not…
##  4 0 [No trust at all] 16    2 [2]       2 [2]       8 [8]       9 [9]   2 [No]
##  5 0 [No trust at all] 16    0 [No … NA(b)       NA(b)       NA(b)       2 [No]
##  6 0 [No trust at all] 16    0 [No …     5 [5]   NA(b)           5 [5]   2 [No]
##  7 0 [No trust at all] 17    0 [No …     0 [No …     0 [No …     0 [No … 3 [Not…
##  8 0 [No trust at all] 17    0 [No …     0 [No …     0 [No … NA(c)       3 [Not…
##  9 0 [No trust at all] 17    1 [1]       3 [3]       5 [5]       4 [4]   2 [No]
## 10 0 [No trust at all] 18    0 [No …     4 [4]       7 [7]       8 [8]   2 [No]
```

References

Parts of this course are inspired by the following resources:

- Wickham, Hadley and Garrett Grolemund, 2017. *R for Data Science - Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly.
- Bahnsen, Oke and Guido Ropers, 2022. *Introduction to R for Quantitative Social Science*. Course held as part of the GESIS Workshop Series.
- Breuer, Johannes and Stefan Jünger, 2021. *Introduction to R for Data Analysis*. Course held as part of the GESIS Summer School in Survey Methodology.
- Teaching material developed by Verena Kunz, David Weyrauch, Oliver Rittmann and Viktoriia Semenova.