Seminar 3

I dag skal vi fortsette med OLS og databehandling: 1. Hvordan plotter vi resultater fra OLS? 2. Hvordan bruker vi R til å sjekke om forutsetningene for OLS holder? 3. Hvordan slår vi sammen flere datasett?

Først: er det noen spørsmål til det vi gikk gjennom i går? Dersom du synes manipulering av data er vanskelig så kan det hjelpe å ta en titt på kapittel seks i **Lær deg R**. Dersom du er nysgjerrig på flere måter å omkode variabler på så kan du kikke på kapittel 5 i **R for Data Science**. Og ikke glem: internett er din venn når du skal lære R.

Hvordan plotte resutlater fra OLS?

I dag skal vi plotte resultatene og gjøre regresjonsdiagnostikk på modellen fra **Burnside and Dollar** - samme artikkel som vi har brukt tidligere i uka og samme som dere repliserte i oppgaven i går. Først laster vi inn pakker, data og kjører modellen.

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

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

```
# Laster inn data
load("../../data/aid.RData")

# Gjør de nødvendige omkodingene som dere gjorde i oppgaven
aid <- aid %>%
  mutate(log_gdp_pr_capita = log(gdp_pr_capita),
         period_fac = as.factor(period),
         region = ifelse(fast_growing_east_asia == 1, "East Asia",
                         ifelse(sub_saharan_africa == 1, "Sub-Saharan Africa", "Other")),
         region = factor(region, levels = c("Other", "Sub-Saharan Africa", "East Asia")))

# Kjører modellen og bevarer informasjon om missing med na.action = "na.exclude"
m5 <- lm(data = aid,
         gdp_growth ~ log_gdp_pr_capita + ethnic_frac*assasinations +
           institutional_quality + m2_gdp_lagged + region + policy*aid +
           period_fac,
         na.action = "na.exclude")

# Printer resultatene i en tabell
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```r
stargazer(m5, type = "text")
```

```
##
## ===================================================
##                              Dependent variable:
##                          -------------------------
##                                  gdp_growth
## ---------------------------------------------------
## log_gdp_pr_capita                 -0.600
##                                   (0.393)
##
## ethnic_frac                       -0.424
##                                   (0.810)
##
## assasinations                     -0.449
##                                   (0.301)
##
## institutional_quality            0.687***
##                                   (0.175)
##
## m2_gdp_lagged                      0.012
##                                   (0.016)
##
## regionSub-Saharan Africa         -1.872***
##                                   (0.681)
##
## regionEast Asia                    1.307*
##                                   (0.731)
##
## policy                           0.712***
##                                   (0.244)
##
## aid                               -0.021
##                                   (0.178)
##
## period_fac3                       -0.013
##                                   (0.620)
##
## period_fac4                      -1.414**
##                                   (0.629)
##
## period_fac5                      -3.470***
##                                   (0.641)
##
## period_fac6                      -2.010***
##                                   (0.661)
##
## period_fac7                      -2.256***
##                                   (0.708)
##
## ethnic_frac:assasinations          0.792
```

```
##                                    (0.620)
##
## policy:aid                         0.186*
##                                    (0.101)
##
## Constant                            3.391
##                                    (2.945)
##
## ---------------------------------------------------
## Observations                         270
## R2                                  0.394
## Adjusted R2                         0.356
## Residual Std. Error           2.873 (df = 253)
## F Statistic                10.297*** (df = 16; 253)
## ===================================================
## Note:                     *p<0.1; **p<0.05; ***p<0.01
```

Så plotter vi effekten av institusjonell kvalitet på vekst i BNP (GDP). Vi går ikke veldig nøye inn på dette nå, men les gjerne denne guiden til regresjonsplot. For å plotte en regresjonslinje så oppretter vi først et datasett der vi holder alle uavhengige variabler bortsett fra den vi vil plotte effekten til konstante. Her velger jeg å la `institutional_quality` variere fra minimums- til maksimumsverdien og setter resten av variablene til gjennomsnitt eller modusverdi. Neste steg er å predikere verdier for det nye datasettet basert på modellen vår ved hjelp av `predict()`. `predict()` tar datasettet vi har laget og gir oss blant annet predikerte verdier og konfidensintervaller basert på modellen vår. For å få datasettet vi skal bruke til plotting så binder vi resultatet av `predict` sammen med datasettet vi lagde. For at `predict()` skal gi likt antall observasjoner som vi har i datasettet vårt så er det viktig å bevare informasjon om de observasjonene som har missing. Dette gjør vi med argumentet `na.action = "na.exclude` i `lm()`.

```r
# Lager datasettet
snitt_data <- data.frame(log_gdp_pr_capita = mean(aid$log_gdp_pr_capita, na.rm = TRUE),
                         ethnic_frac = mean(aid$ethnic_frac, na.rm = TRUE),
                         assasinations = mean(aid$assasinations, na.rm = TRUE),
                         institutional_quality = c(seq(min(aid$institutional_quality, na.rm = TRUE),
                                                       max(aid$institutional_quality, na.rm =TRUE), by = 0.5
                         m2_gdp_lagged = mean(aid$m2_gdp_lagged, na.rm = TRUE),
                         region = "Other",
                         policy = mean(aid$policy, na.rm = TRUE),
                         aid = mean(aid$aid, na.rm = TRUE),
                         period_fac = "4")

# Bruker predict
predict(m5, newdata = snitt_data, se = TRUE)
```

```
## $fit
##         1         2         3         4         5         6         7         8
## 0.3620244 0.7054454 1.0488664 1.3922874 1.7357084 2.0791294 2.4225504 2.7659714
##         9        10
## 3.1093924 3.4528134
##
## $se.fit
##         1         2         3         4         5         6         7         8
## 0.5780223 0.5366767 0.5071121 0.4914594 0.4910508 0.5059230 0.5348029 0.5755858
##         9        10
## 0.6259494 0.6837801
##
```

```
## $df
## [1] 253
##
## $residual.scale
## [1] 2.872583
```

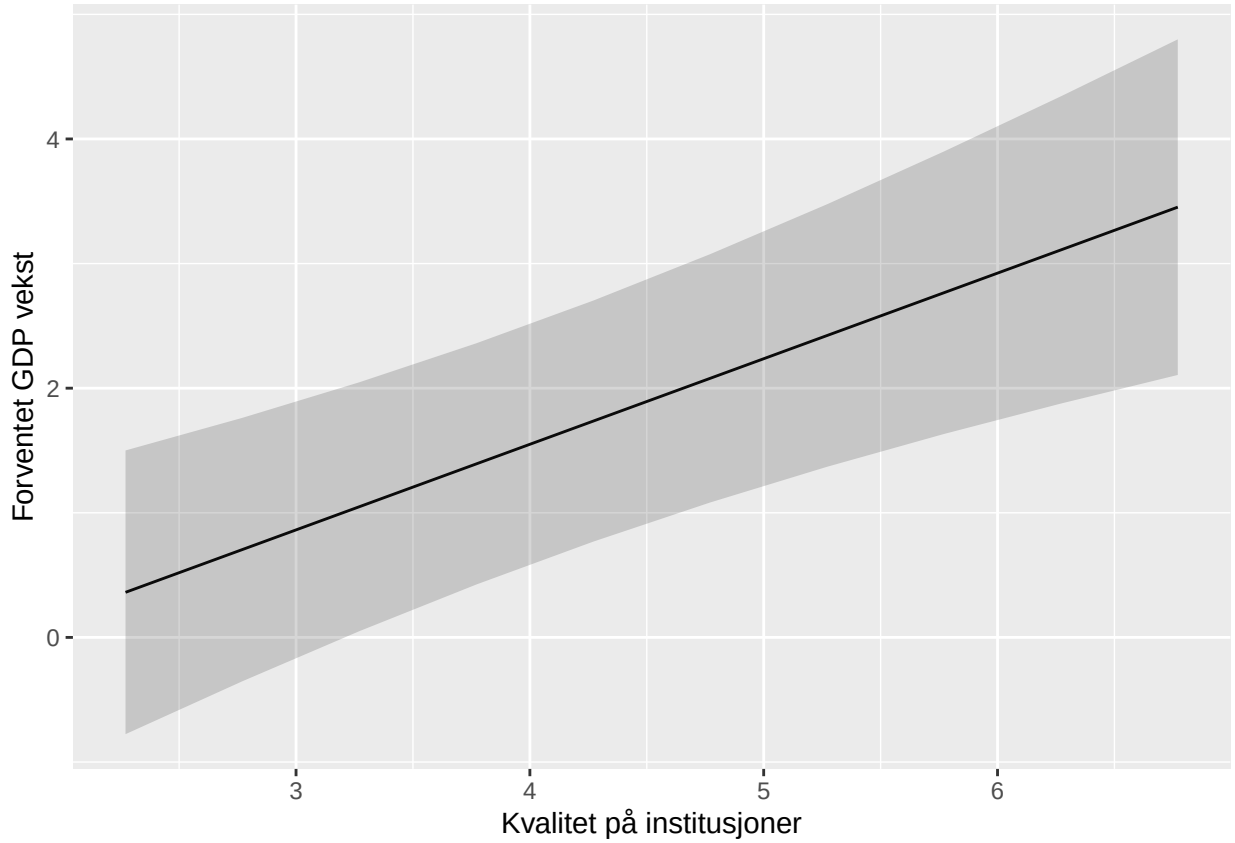
```
# Legger predikerte verdier inn i snitt_data
snitt_data <- cbind(snitt_data, predict(m5, newdata = snitt_data, se = TRUE, interval = "confidence"))
snitt_data
```

```
##    log_gdp_pr_capita ethnic_frac assasinations institutional_quality
## 1           7.440955   0.4738369     0.3974164              2.270833
## 2           7.440955   0.4738369     0.3974164              2.770833
## 3           7.440955   0.4738369     0.3974164              3.270833
## 4           7.440955   0.4738369     0.3974164              3.770833
## 5           7.440955   0.4738369     0.3974164              4.270833
## 6           7.440955   0.4738369     0.3974164              4.770833
## 7           7.440955   0.4738369     0.3974164              5.270833
## 8           7.440955   0.4738369     0.3974164              5.770833
## 9           7.440955   0.4738369     0.3974164              6.270833
## 10          7.440955   0.4738369     0.3974164              6.770833
##    m2_gdp_lagged region   policy     aid period_fac   fit.fit     fit.lwr
## 1         28.415  Other 1.160546 1.75757          4 0.3620244 -0.77632391
## 2         28.415  Other 1.160546 1.75757          4 0.7054454 -0.35147747
## 3         28.415  Other 1.160546 1.75757          4 1.0488664  0.05016746
## 4         28.415  Other 1.160546 1.75757          4 1.3922874  0.42441462
## 5         28.415  Other 1.160546 1.75757          4 1.7357084  0.76864047
## 6         28.415  Other 1.160546 1.75757          4 2.0791294  1.08277224
## 7         28.415  Other 1.160546 1.75757          4 2.4225504  1.36931761
## 8         28.415  Other 1.160546 1.75757          4 2.7659714  1.63242141
## 9         28.415  Other 1.160546 1.75757          4 3.1093924  1.87665708
## 10        28.415  Other 1.160546 1.75757          4 3.4528134  2.10618737
##     fit.upr    se.fit  df residual.scale
## 1  1.500373 0.5780223 253       2.872583
## 2  1.762368 0.5366767 253       2.872583
## 3  2.047565 0.5071121 253       2.872583
## 4  2.360160 0.4914594 253       2.872583
## 5  2.702776 0.4910508 253       2.872583
## 6  3.075487 0.5059230 253       2.872583
## 7  3.475783 0.5348029 253       2.872583
## 8  3.899521 0.5755858 253       2.872583
## 9  4.342128 0.6259494 253       2.872583
## 10 4.799439 0.6837801 253       2.872583
```

Variabelen som heter `fit.fit` er de predikerte verdiene. `fit.lwr` og `fit.upr` er nedre og øvre grense for et 95 % konfidensintervall. `se.fit` er standardfeilen.

Lager plot:

```
library(ggplot2)
ggplot(snitt_data, aes(x = institutional_quality, y = fit.fit)) + # Setter institusjonell kvalitet på x
  geom_line() +                                                   # Sier at jeg vil ha et linjediagram
  scale_y_continuous(breaks = seq(-12, 12, 2)) +                  # Bestemmer verdier og mellomrom på y
  geom_ribbon(aes(ymin = fit.lwr, ymax = fit.upr, color = NULL), alpha = .2) + # Legger til konfidensin
  labs(x = "Kvalitet på institusjoner", y = "Forventet GDP vekst", color = "Policy", fill = "Policy") #
```



Dette kan, og bør, også gjøres når det er samspill i modellen. Samspill er vanskelig å tolke i en tabell og jeg synes derfor det er fint å plotte disse. Når vi skal plotte samspill så lar vi begge variablene som er en del av samspillsleddet variere, mens resten er konstante. Vi lar den ene variabelen være `x`, mens vi bruker den andre til å fylle ut argumentet `color`. I tilfellet med to kontinuerlige variabler må en gjøre den ene om til en faktorvariabel slik jeg gjør med policy under.

```
# Lager plot data
snitt_data_sam <- data.frame(log_gdp_pr_capita = mean(aid$log_gdp_pr_capita, na.rm = TRUE),
                         ethnic_frac = mean(aid$ethnic_frac, na.rm = TRUE),
                         assasinations = mean(aid$assasinations, na.rm = TRUE),
                         institutional_quality = mean(aid$institutional_quality, na.rm = TRUE),
                         m2_gdp_lagged = mean(aid$m2_gdp_lagged, na.rm = TRUE),
                         region = "Other",
                         policy = c(rep(-1, 9), rep(0, 9), rep(1, 9)),
                         aid = rep(0:8, 3),
                         period_fac = "4")

# Predikerer verdier (løser likningen for modellen)
predict(m5, newdata = snitt_data_sam, se = TRUE)
```

```
## $fit
##           1           2           3           4           5           6
##  0.08409744 -0.12290215 -0.32990173 -0.53690132 -0.74390091 -0.95090049
##           7           8           9          10          11          12
## -1.15790008 -1.36489967 -1.57189925  0.79654817  0.77576457  0.75498097
##          13          14          15          16          17          18
##  0.73419737  0.71341377  0.69263017  0.67184656  0.65106296  0.63027936
```

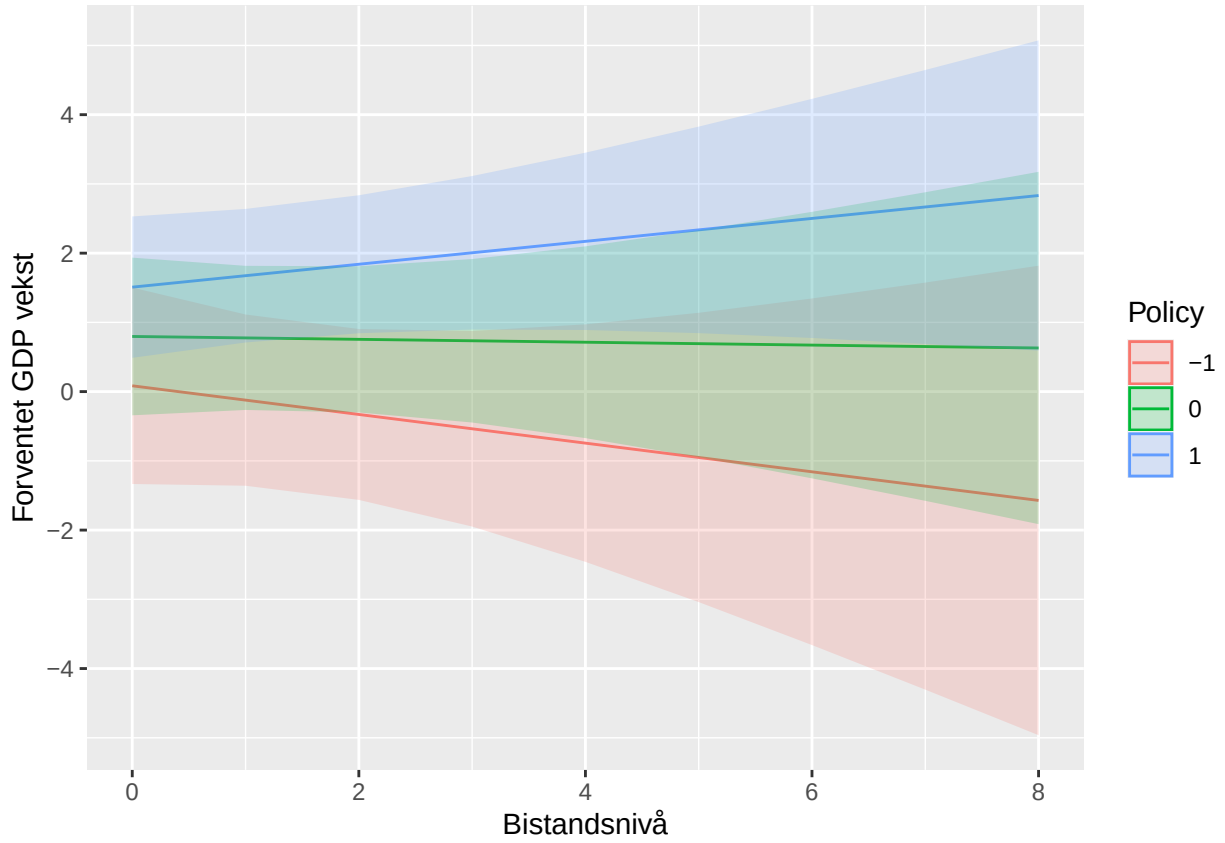
```
##          19          20          21          22          23          24
##  1.50899890  1.67443129  1.83986367  2.00529606  2.17072844  2.33616083
##          25          26          27
##  2.50159321  2.66702559  2.83245798
##
## $se.fit
##         1         2         3         4         5         6         7         8
## 0.7202650 0.6279388 0.6266786 0.7169646 0.8707719 1.0608217 1.2709582 1.4927225
##         9        10        11        12        13        14        15        16
## 1.7216269 0.5782802 0.5284831 0.5362879 0.5994488 0.7032047 0.8325135 0.9772843
##        17        18        19        20        21        22        23        24
## 1.1315980 1.2920401 0.5183695 0.4898167 0.5059652 0.5629815 0.6502018 0.7572607
##        25        26        27
## 0.8769218 1.0046925 1.1378441
##
## $df
## [1] 253
##
## $residual.scale
## [1] 2.872583
```

```
# Lagrer predikerte verdier i plot datasettet
snitt_data_sam <- cbind(snitt_data_sam, predict(m5, newdata = snitt_data_sam, se = TRUE, interval = "co
snitt_data_sam
```

```
##    log_gdp_pr_capita ethnic_frac assasinations institutional_quality
## 1           7.440955   0.4738369     0.3974164              4.607119
## 2           7.440955   0.4738369     0.3974164              4.607119
## 3           7.440955   0.4738369     0.3974164              4.607119
## 4           7.440955   0.4738369     0.3974164              4.607119
## 5           7.440955   0.4738369     0.3974164              4.607119
## 6           7.440955   0.4738369     0.3974164              4.607119
## 7           7.440955   0.4738369     0.3974164              4.607119
## 8           7.440955   0.4738369     0.3974164              4.607119
## 9           7.440955   0.4738369     0.3974164              4.607119
## 10          7.440955   0.4738369     0.3974164              4.607119
## 11          7.440955   0.4738369     0.3974164              4.607119
## 12          7.440955   0.4738369     0.3974164              4.607119
## 13          7.440955   0.4738369     0.3974164              4.607119
## 14          7.440955   0.4738369     0.3974164              4.607119
## 15          7.440955   0.4738369     0.3974164              4.607119
## 16          7.440955   0.4738369     0.3974164              4.607119
## 17          7.440955   0.4738369     0.3974164              4.607119
## 18          7.440955   0.4738369     0.3974164              4.607119
## 19          7.440955   0.4738369     0.3974164              4.607119
## 20          7.440955   0.4738369     0.3974164              4.607119
## 21          7.440955   0.4738369     0.3974164              4.607119
## 22          7.440955   0.4738369     0.3974164              4.607119
## 23          7.440955   0.4738369     0.3974164              4.607119
## 24          7.440955   0.4738369     0.3974164              4.607119
## 25          7.440955   0.4738369     0.3974164              4.607119
## 26          7.440955   0.4738369     0.3974164              4.607119
## 27          7.440955   0.4738369     0.3974164              4.607119
##    m2_gdp_lagged region policy aid period_fac     fit.fit    fit.lwr   fit.upr
## 1         28.415  Other     -1   0          4  0.08409744 -1.3343814 1.5025763
```

```
## 2         28.415  Other     -1   1          4 -0.12290215 -1.3595553 1.1137510
## 3         28.415  Other     -1   2          4 -0.32990173 -1.5640730 0.9042695
## 4         28.415  Other     -1   3          4 -0.53690132 -1.9488805 0.8750779
## 5         28.415  Other     -1   4          4 -0.74390091 -2.4587859 0.9709841
## 6         28.415  Other     -1   5          4 -0.95090049 -3.0400666 1.1382656
## 7         28.415  Other     -1   6          4 -1.15790008 -3.6609059 1.3451058
## 8         28.415  Other     -1   7          4 -1.36489967 -4.3046446 1.5748453
## 9         28.415  Other     -1   8          4 -1.57189925 -4.9624451 1.8186466
## 10        28.415  Other      0   0          4  0.79654817 -0.3423081 1.9354044
## 11        28.415  Other      0   1          4  0.77576457 -0.2650221 1.8165512
## 12        28.415  Other      0   2          4  0.75498097 -0.3011763 1.8111382
## 13        28.415  Other      0   3          4  0.73419737 -0.4463480 1.9147427
## 14        28.415  Other      0   4          4  0.71341377 -0.6714669 2.0982945
## 15        28.415  Other      0   5          4  0.69263017 -0.9469093 2.3321696
## 16        28.415  Other      0   6          4  0.67184656 -1.2528022 2.5964953
## 17        28.415  Other      0   7          4  0.65106296 -1.5774890 2.8796149
## 18        28.415  Other      0   8          4  0.63027936 -1.9142448 3.1748035
## 19        28.415  Other      1   0          4  1.50899890  0.4881299 2.5298679
## 20        28.415  Other      1   1          4  1.67443129  0.7097938 2.6390688
## 21        28.415  Other      1   2          4  1.83986367  0.8434235 2.8363038
## 22        28.415  Other      1   3          4  2.00529606  0.8965689 3.1140232
## 23        28.415  Other      1   4          4  2.17072844  0.8902308 3.4512261
## 24        28.415  Other      1   5          4  2.33616083  0.8448232 3.8274984
## 25        28.415  Other      1   6          4  2.50159321  0.7745967 4.2285898
## 26        28.415  Other      1   7          4  2.66702559  0.6883994 4.6456518
## 27        28.415  Other      1   8          4  2.83245798  0.5916051 5.0733109
##       se.fit  df residual.scale
## 1  0.7202650 253       2.872583
## 2  0.6279388 253       2.872583
## 3  0.6266786 253       2.872583
## 4  0.7169646 253       2.872583
## 5  0.8707719 253       2.872583
## 6  1.0608217 253       2.872583
## 7  1.2709582 253       2.872583
## 8  1.4927225 253       2.872583
## 9  1.7216269 253       2.872583
## 10 0.5782802 253       2.872583
## 11 0.5284831 253       2.872583
## 12 0.5362879 253       2.872583
## 13 0.5994488 253       2.872583
## 14 0.7032047 253       2.872583
## 15 0.8325135 253       2.872583
## 16 0.9772843 253       2.872583
## 17 1.1315980 253       2.872583
## 18 1.2920401 253       2.872583
## 19 0.5183695 253       2.872583
## 20 0.4898167 253       2.872583
## 21 0.5059652 253       2.872583
## 22 0.5629815 253       2.872583
## 23 0.6502018 253       2.872583
## 24 0.7572607 253       2.872583
## 25 0.8769218 253       2.872583
## 26 1.0046925 253       2.872583
## 27 1.1378441 253       2.872583
```

```r
# Plotter
ggplot(snitt_data_sam, aes(x = aid, y = fit.fit,
                       group = factor(policy),
                       color = factor(policy),
                       fill = factor(policy))) +
  geom_line() +
  scale_y_continuous(breaks = seq(-12, 12, 2)) +
  geom_ribbon(aes(ymin = fit.lwr, ymax = fit.upr, color = NULL), alpha = .2) +
  labs(x = "Bistandsnivå", y = "Forventet GDP vekst", color = "Policy", fill = "Policy")
```



Vi skal ikke bruke snitt_data mer så jeg fjerner objektene fra environment:

```r
rm(snitt_data, snitt_data_sam)
```

Forutsetninger for regresjon

Dere vil se forutsetningene for OLS formulert på litt forskjellige måter i ulike metodetekster. Det er blant annet forskjell på forutsetninger for at OLS skal være forventningsrett og konsistent, og at OLS skal være BLUE. Det er også mulig å formulere de samme forutsetningene i ulik språkdrakt, selv når forutsetningene bygger på de samme matematiske formuleringene. Noen ganger vil dere også se at forutsetningene om restledd er utelatt, andre ganger vil dere kunne se en antagelse om at kurtosen ikke er uendelig stor. Noen vil kategorisere ingen innflytelsesrike observasjoner og ikke perfekt multikolinearitet som antagelser, mens andre vil kategorisere det som problemer/trusler. Dere forholder dere til pensum, jeg følger Cristophersens forelesning her. Det bør forøvrig nevnes at **Lær deg R** gir en ypperlig gjennomgang av regresjonsdiagnostikk.

Kritiske aspekter i modellvurdering - OLS:

1. Ingen utelatt variabelskjevhet

2. Lineær sammenheng mellom variablene
3. Ingen autokorrelasjon/Uavhengige observasjoner
4. Normalfordelte residualer
5. Homoskedastiske residualer
6. Ingen perfekt multikollinearitet
7. Manglende opplysninger(missing values)

Regresjonsdiagnostikk i R

Jeg anbefaler `car` pakken til John Fox til regresjonsdiagnostikk. Den gir ikke like vakre figurer som `ggplot`, men er veldig lett å bruke for nybegynnere, og inneholder alle slags funksjoner man trenger for regresjonsdiagnostikk. På sikt kan dere lære dere å konstruere disse plottene selv med `ggplot`. Pass imidlertid på at dere forstår hva plotet dere bruker faktisk innebærer (det er lov å spørre om hjelp). I kapittel 6 av boken *An R Companion to Applied Regression* (Fox og Weisberg), gjennomgås diagnostikk med `car` i detalj. En annen pakke som er god, er `lmtest`. Til testing av autokorrelasjon på paneldata er det lettest å bruke pakken `plm`.

I tillegg til diagnostikken som vi gjør i seminaret, er det fint å se på deskriptiv statistikk, effektplot, samt diskusjon av data. I tillegg til å teste antagelsene over (med unntak av antagelse 1), skal vi også se på innflytelsesrike observasjoner og multikolinearitet.

Ingen utelatt variabelskjevhet

Hva innebærer denne antagelsen?

- Dersom vi vil tolke alle variablene i modellen vår substansielt, må alle variabler som påvirker vår avhengige variabel, og som er korrelert med en uavhengig variabel inkluderes i modellen.
- Dersom vi vil tolke en uavhengig variabel, kan vi tenke på de resterende variablene som kontrollvariabler, som er korrelert med uavhengig variabel og påvirker avhengig variabel.

Merk: korrelasjon er lineær sammenheng mellom to variabler, ikke et årsaksforhold. Så lenge to variabler påvirker den avhengige variabelen, og de er korrelert (selv om de ikke påvirker hverandre på noe vis), får vi utelatt variabelskjevhet dersom vi ikke kontrollerer for den andre variabelen.

Denne antagelsen kan vi ikke teste, dersom vi ikke har data for alle variabler. Det betyr at dette først og fremst må begrunnes teoretisk. Det finnes imidlertid metoder for å estimere effekten av utelatte variabler med ulike egenskaper. Denne formen for robusthetstesting kalles *sensivity analysis*, men det er ikke noe vi kommer til å gå inn på her.

Lineær sammenheng mellom variablene

Metoden vi bruker for å regne ut lineær regresjon tar blant annet utgangspunkt i kovarians mellom uavhengige variabler og avhengige variabler. I likhet med korrelasjon, er kovarians et mål på lineær sammenheng mellom to variabler. Derfor forutsetter lineær regresjon en lineær sammenheng mellom uavhengig av avhengig variabel. Brudd på denne forutsetningen kan potensielt gi svært missvisende resultater, f.eks. ved å gjøre en U-formet sammenheng om til *ingen lineær sammenheng*.

Huskregel: Hver gang vi opphøyer en uavhengig variabel, tillater vi en ekstra *sving* i sammenhengen mellom den avhengige og uavhengige variabelen.

Dersom hypotesen vår er at det er en positiv sammenheng mellom to variabler, står vi fritt til å legge til andregradsledd og tredjegradsledd, osv, fordi vi ikke påstår at sammenhengen er perfekt lineær, bare at den er positiv. Dette er det vanligste. Vi står dermed fritt til å slenge inn andregrads og tredjegradsledd. Vær imidlertid forsiktig med å opphøye en uavhengig variabel for mye. Da står man i fare for **overfitting**, dvs. å finne en svært spesifikk sammenheng i datasettet ditt, som du ikke finner dersom du samler inn samme type data på nytt.

I noen tilfeller er hypotesen vår mer spesifikk, for eksempel at en sammenheng er U-formet (konveks), da må vi teste om:

1. Vi får en U-formet sammenheng når vi legger inn et annengradsledd.
2. Om regresjonen med et andregradsledd passer til data.

Viktig: Dersom dere legger inn andregradsledd eller andre polynomer, husk på å tolke alle leddene for den variabelen sammen. Det er lettest å gjøre dette ved hjelp av plot.

Sjekke linearitet i R Det finnes flere måter å teste linearitetsantagelsen på. Man kan gjøre en grafisk test, ved å plotte residualene til den avhengige variabelen mot residualene til den uavhengige variabelen vi er interessert i. Jeg viser en annen test som gjør samme nytten, men som har noen fordeler.

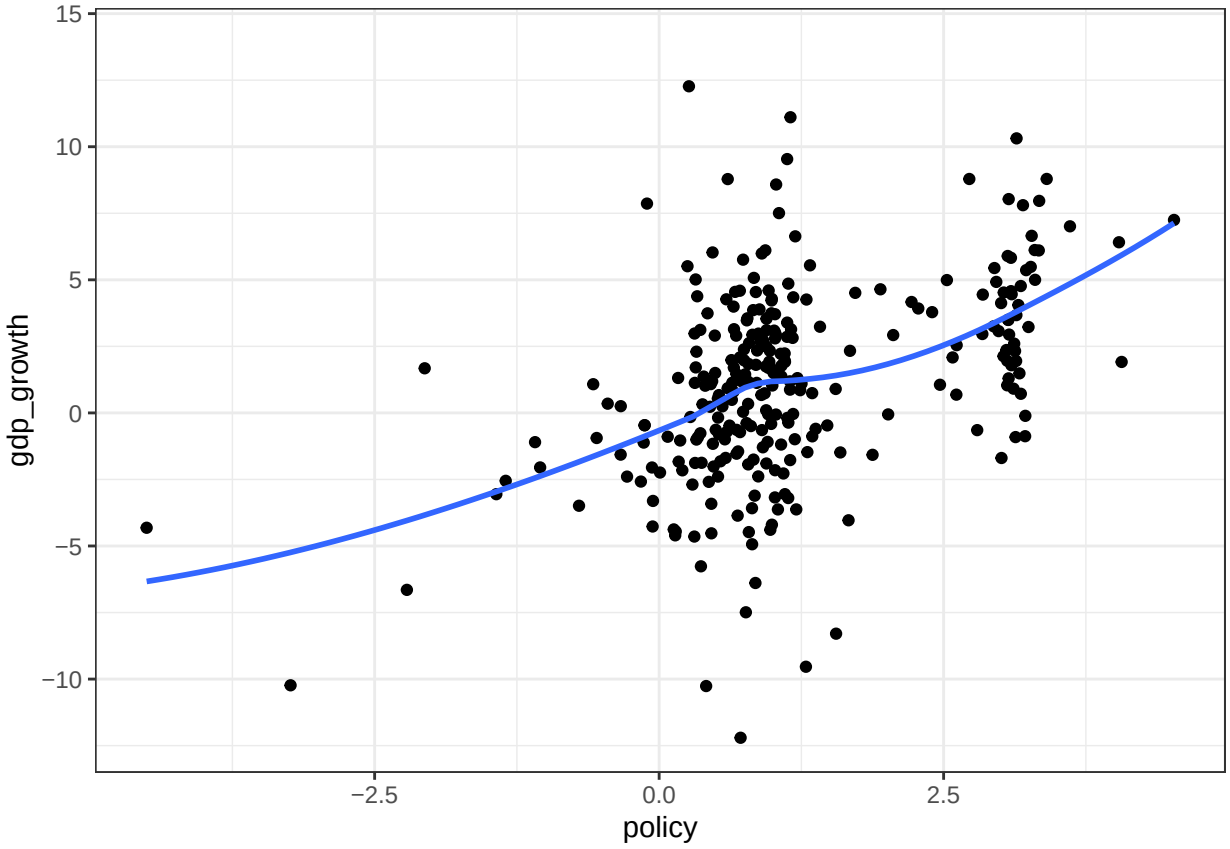
Vi kan bruke `ggplot()` til å undersøke om sammenhengen mellom en avhengig og en uavhengig variabel er lineær. Ved å lage en spredningsdiagram kan vi undersøke formen på sammenhengen (Introduksjon til statistisk analyse, Christophersen (2013)). Dette kan vi gjøre før vi kjører modellen.

```
ggplot(aid) +
  geom_point(aes(y = gdp_growth, x = policy)) +
  geom_smooth(aes(y = gdp_growth, x = policy),
              se = FALSE) +
  theme_bw()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## Warning: Removed 47 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 47 rows containing missing values (geom_point).
```



Vi kan også bruke funksjonen `ceresPlot()` fra pakken `car` til å teste om sammenhengen mellom en uavhengig og en avhengig variabel er lineær, men da må vi kjøre modellen først. Denne funksjonen fungerer både for

lineær regresjon, og for logistisk regresjon (`glm`). Denne funksjonen fungerer imidlertid ikke for regresjon med samspill.

Det denne funksjonen gjør, er å legge sammen residualene fra en regresjon med parameterestimatet til en variabel (på y-aksen), og plotte mot variabelens verdi. Deretter tegnes det en rosa linje som passer data.

Dersom sammenhengen ikke er lineær, kan man prøve en transformasjon eller et polynom.

Uavhengighet/Ingen autokorrelasjon

Denne antagelsen holder dersom vi har et tilfeldig utvalg fra en populasjon, på et tidspunkt. Da vil observasjonene være statistisk uavhengige (alle observasjonene er trukket tilfeldig), og likt distribuert (alle observasjonene er trukket fra samme populasjon). Dersom vi ikke har et slikt utvalg, vil det kunne være sammenhenger mellom observasjoner. Dersom vi f.eks. har data for statsbudsjettet over tid, vil vi trolig se **autokorrelasjon** fra ett år til det neste fordi budsjettet endres inkrementelt. Andre typer avhengighet enn autokorrelasjon er også mulig, som geografisk avhengighet eller tidsperioder.

Sjekke uavhengighet i R Man kan teste for autkorrelasjon med Durbin-Watson testen. En funksjon for dette er `pdwtest()` fra pakken `plm` - denne fungerer både på tidsserier og paneldata, men krever at du bruker funksjonen `plm()` til å kjøre OLS-modellen din (bruk `?plm` for å se hvordan du kan gjøre dette eller kom på fordypningsseminarene neste uke).`durbinWatsonTest()` fra `car` virker bare på tidsserier, men her kan du bruke `lm()`-objekter.

```
#install.packages("plm")
# library(plm)

# Kjører modellen på ny uten å bevare missingverdier
m5b <- lm(gdp_growth ~ log_gdp_pr_capita + ethnic_frac * assasinations +
              institutional_quality + m2_gdp_lagged + region + policy * aid +
              period_fac,
            data = aid, na.action = "na.omit")
# Her blir det problemer om vi bevarer na med na.exclude.

car::durbinWatsonTest(m5b)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1      0.02981496      1.939984   0.398
##  Alternative hypothesis: rho != 0
```

For model 5 er Durbin-Watson testen ikke signifikant på konvensjonelle nivåer (p-verdien er 0.336). Samtidig ser vi at Durbin-Watson verdien er rett under 2. Durbin-Watson verdier i nærheten av 2 indikerer ingen autokorrelasjon (se Christophersen s. 78).

Normalfordelte residualer:

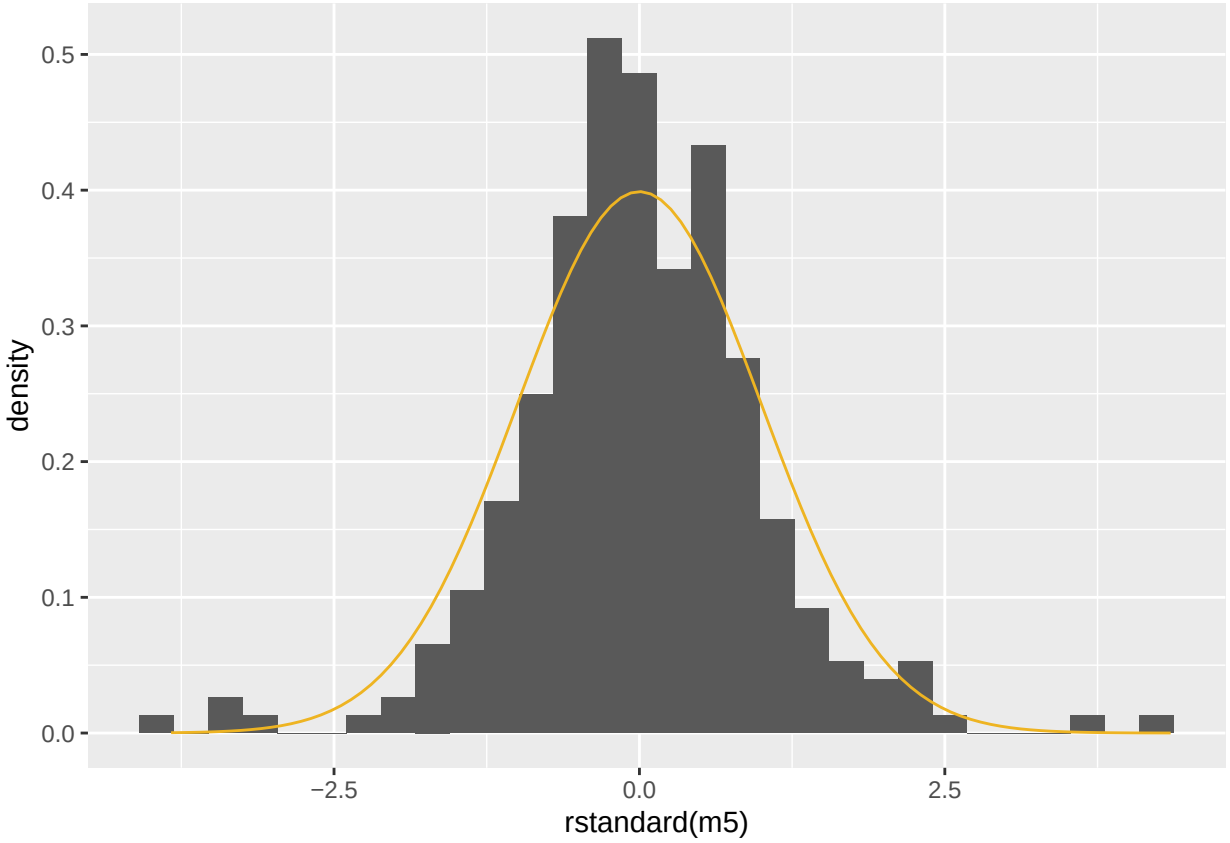
Residualene fra modellen er normalfordelt og har gjennomsnitt tilnærmet lik 0.

Sjekke normalfordelte residualer i R:

Det er flere måter å gjøre dette på. Først kan vi plotte fordelingene til residualene våre og sammenligne med en normalfordeling ved hjelp av `ggplot()`.

```
ggplot() +
  geom_histogram(aes(x = rstandard(m5),
                     y = ..density..)) +
  stat_function(fun = dnorm,
                color = "goldenrod2") # Plotter inn en normalfordeling
```

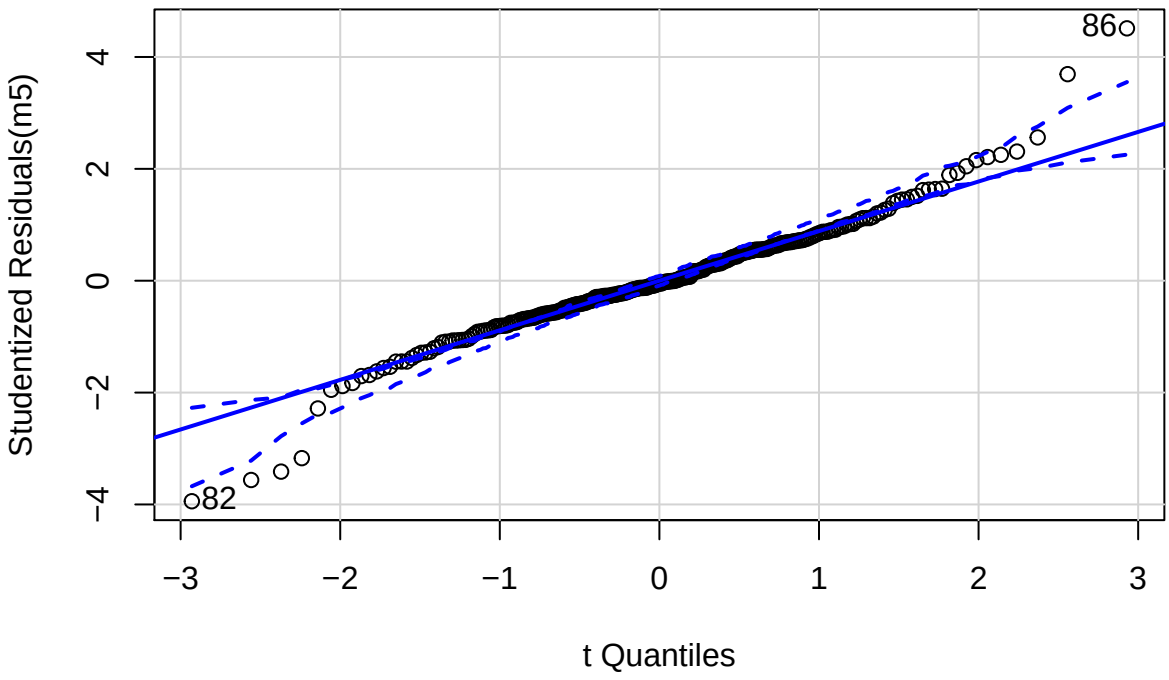
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 61 rows containing non-finite values (stat_bin).
```



Vi kan teste for normalfordelte residualer ved å plotte studentiserte residualer fra regresjonen vår mot kvantiler fra den kummulative normalfordelingen. Dette kalles qq-plot, og kan kjøres i R med `qqPlot()`.

Studentiserte residualer: Alternativ måte å standardisere på, i beregning av varians for hver enkelt observasjon, fjerner man observasjonen. Formålet med dette er at vi får statistisk uavhengihet mellom teller og nevner, noe som lar oss bruke residualene til statistiske tester.

```
car::qqPlot(m5)
```




```
## [1] 82 86
```

Vi kan også sjekke skjevhet og kurtose til standardavvikene ved hjelp av funksjonene`kurtosis()` og `skewness()` i pakken moments.

```
#install.packages("moments")
library(moments)
kurtosis(rstandard(m5), na.rm = TRUE)
```

```
## [1] 5.74082
```

```
skewness(rstandard(m5), na.rm = TRUE)
```

```
## [1] 0.04282164
```

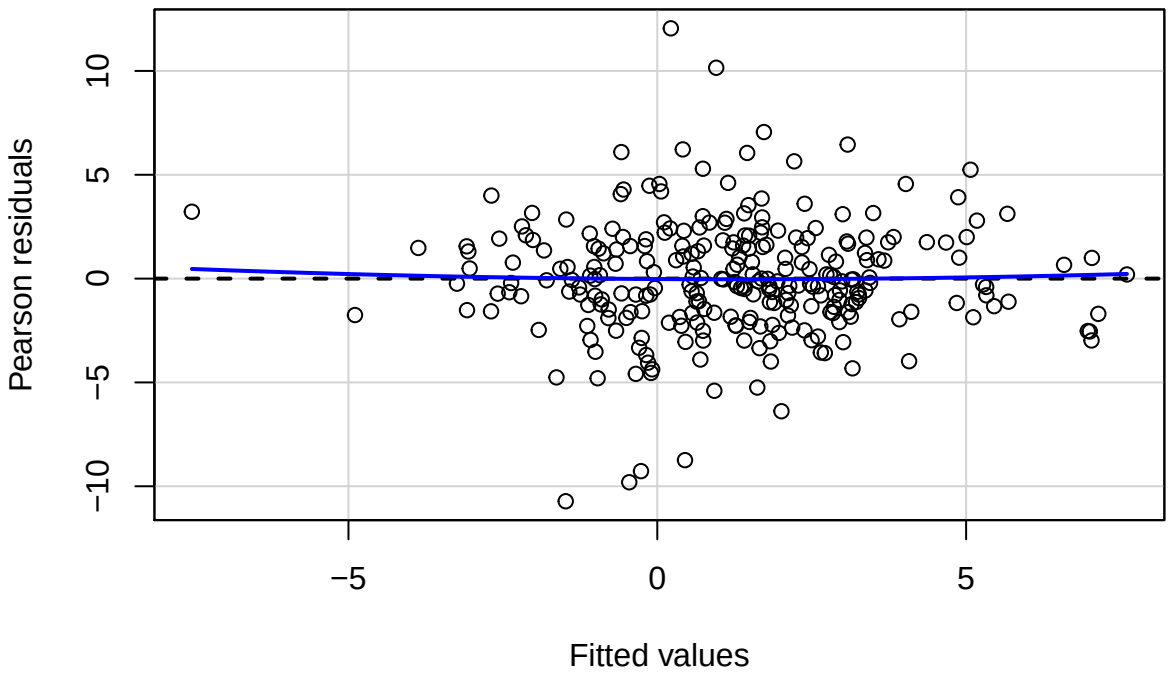
Homoskedastiske residualer:

Variansen til residualene skal være konstant for ulike nivå av uavhengig variabel.

Sjekke om vi har homoskedastiske residualer i R:

Vi kan teste for heteroskedastisitet ved hjelp av plot av studentiserte residualer mot standardiserte predikerte verdier fra modellen. Dette kan gjøres med `residualPlot()` i car. Dere kan også lage deres egen versjon med `ggplot()` i stedet.

```
car::residualPlot(m5)
```



Ingen perfekt multikolinearitet:

Det skal ikke være en perfekt lineær sammenheng mellom et sett av de uavhengige variablene. Dette fører til at regresjonen ikke lar seg estimere, og skyldes som regel at man har lagt inn dummyvariabler for alle kategorier av en variabel, som en dummy for mann og en for kvinne. Høy multikolinearitet kan også være problematisk, men er ikke en forutsetning for at regresjon ikke skal fungere.

Sjekke om vi har multikolinearitet i R:

Vi kan teste for multikolinearitet ved hjelp av en vif-test. Funksjonen for dette er `vif()`. Med vif tester vi om det er en sterk lineær sammenheng mellom uavhengige variabler, dersom dette er tilfellet er det gjerne nødvendig med store mengder data for å skille effektene av ulike variabler fra hverandre/få presise estimater (små standardfeil), men bortsett fra å samle mer data er det ikke så mye vi gjøre dersom vi mener begge variablene må være med i modellen.

```
car::vif(m5)
```

```
##                              GVIF Df GVIF^(1/(2*Df))
## log_gdp_pr_capita        2.487685  1        1.577240
## ethnic_frac              1.944429  1        1.394428
## assasinations            4.588600  1        2.142102
## institutional_quality    1.509069  1        1.228442
## m2_gdp_lagged            1.345847  1        1.160107
## region                   5.006125  2        1.495807
## policy                   2.712507  1        1.646969
## aid                      3.169260  1        1.780241
## period_fac               1.514257  5        1.042365
```

```
## ethnic_frac:assasinations 4.663442  1        2.159500
## policy:aid                2.633899  1        1.622929
```

Du kan lese mer om hvilke verdier som er problematiske i kapittel 7 i **Introduksjon til statistisk analyse** eller kapittel 9 i **Lær deg R**.

Outliers, leverage og innflytelsesrike observasjoner

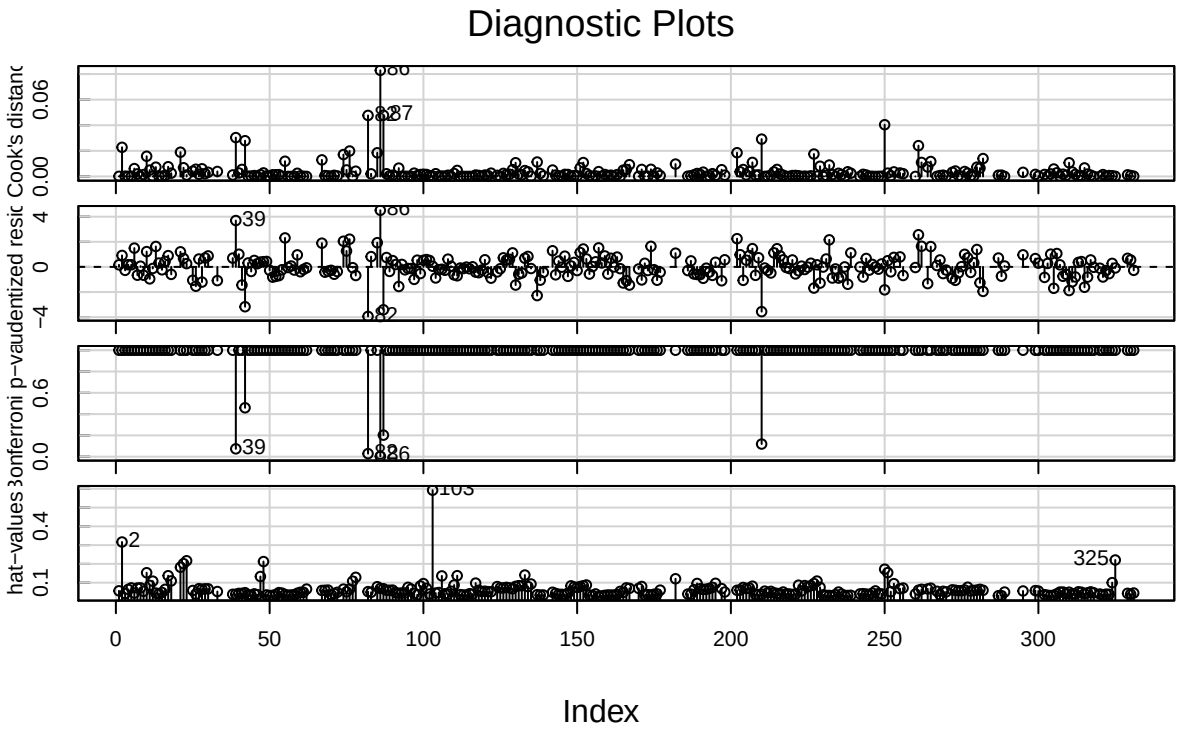
Observasjoner med uvanlige/ekstreme verdier på de uvahengige variablene (når man tar høyde for korrelasjonsmønstre), har høy leverage (Vi bruker gjerne hatte-verdier som mål på leverage observasjoner i lineær regresjon). Observasjoner med høy leverage vil ha stor innflytelse på regresjonslinjen, hvis modellen predikerer slike observasjoner dårlig. Observasjoner som blir predikert dårlig av en modell får store residualer. Vi kaller gjerne slike observasjoner "regression outliers" (Studentiserte residualer brukes ofte som mål på "regression outliers"). Innflytelsesrike observasjoner har dermed høy leverage/er dårlig predikert av modellen, og "trekker" regresjonslinjen mot seg med stor kraft.

Det er ofte lurt å se nærmere på innflytelsesrike enheter og uteliggere, vi kan bruke `influenceIndexPlot()` til å identifisere slike observasjoner. Spesifiser hvor mange observasjoner du vil ha nummerert med argumentet `id = list(n="antall")`. Deretter kan vi se nærmere på disse observasjonene ved hjelp av indeksering. En form for robusthetstesting er å kjøre regresjonen på nytt uten uteliggere og innflytelsesrike observasjoner, for å sjekke om man får samme resultat. Dersom man ikke gjør det, er ikke resultatene dine særlig robuste.

Vi kan også se på Cook's distance, som kombinerer informasjon om uteliggere, leverage og innflytelsesrike observasjoner. `influenceIndexPlot()` gir oss alle disse målene. Disse målene er godt beskrevet i kapittel 9 i **Lær deg R**.

Dersom du kun er interessert i observasjoners innflytelse på en enkeltvariabel, kan du bruke funksjonen `dfbetas()`, som gir deg hver observasjons innflytelse på koeffisientene til alle variablene i en modell.

```
car::influenceIndexPlot(m5,
                        id = list(n=3))
```




```
# Bruker indeksering til å se nærmere på noen av observasjonene
aid[c(39,86), ]
```

```
## # A tibble: 2 x 22
##   country period periodstart periodend code  gdp_growth gdp_pr_capita
##   <chr>    <dbl>       <dbl>     <dbl> <chr>       <dbl>         <dbl>
## 1 CMR          4        1978      1981 CMR4         11.1           972
## 2 GAB          3        1974      1977 GAB3         12.3          5030
## # ... with 15 more variables: economic_open <dbl>, budget_balance <dbl>,
## #   inflation <dbl>, ethnic_frac <dbl>, assasinations <dbl>, aid <dbl>,
## #   fast_growing_east_asia <dbl>, sub_saharan_africa <dbl>,
## #   central_america <dbl>, policy <dbl>, m2_gdp_lagged <dbl>,
## #   institutional_quality <dbl>, log_gdp_pr_capita <dbl>, period_fac <fct>,
## #   region <fct>
```

Observasjoner med manglende informasjon (Missing)

Mye kan sies om manglende informasjon (missing) - her viser jeg måter du kan bruke R til å identifisere missing. Jeg viser også noen enkle måter du kan bruke R til å få et inntrykk av konsekvensene av missing på.

I R kan missing være kodet på flere måter. Dersom missing er eksplisitt definert i R, vil vi se missing som `NA` når vi ser på datasettet. Noen ganger leses ikke missing inn som `NA`. Missing på variabler i datasett fra andre statistikkprogramm kan f.eks. leses som `character` med verdi `" "`, eller som `numeric` med verdi `-99`. For å sjekke dette, bør du lese kodebok. Det er ikke sikkert at `" "` bør omkodes til missing. Du kan også se på en tabell, for å identifisere suspekte verdier:

```
table(aid$country) # ingen suspekte verdier
```

```
##
## ARG BOL BRA BWA CHL CIV CMR COL CRI DOM DZA ECU EGY ETH GAB GHA GMB GTM GUY HND
##   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6   4   6   5   6
## HTI IDN IND JAM KEN KOR LKA MAR MDG MEX MLI MWI MYS NER NGA NIC PAK PER PHL PRY
##   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6   4   6   6   6   6
## SEN SLE SLV SOM SYR TGO THA TTO TUN TUR TZA URY VEN ZAR ZMB ZWE
##   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6
```

Moral: **alltid sjekk kodeboken**, og se på verdiene til data med tabell for å identifisere missing.

Når du kjører en lineær regresjonsanalyse i R, kastes observasjoner som har manglende informasjon (missing, angitt som `NA` i R) på en eller flere av variablene du legger inn i modellen din ut. Men dersom missing er kodet som f.eks.`999` så vil ikke R automatisk oppdage at dette er en missing verdi. Derfor er det viktig å sjekke hvilke observasjoner som faktisk blir kastet ut av analysen din pga. missing, og hva slags informasjon du faktisk sitter igjen med i analysen din.

Her er noen nyttige funksjoner for å jobbe missing:

```
aid$reg_miss <- aid %>%
  select(gdp_growth, aid, policy) %>%
  complete.cases()

# Lager variabel som viser hvilke observasjoner som forsvinner i regresjon med de sentrale variablene
# gdp_growth, aid og policy - fin å bruke i plot for å få et inntrykk av hva slags informasjon du miste
table(aid$reg_miss) # 47 observasjoner har missing på en eller flere av de tre variablene
```

```
##
## FALSE  TRUE
##    47   284
```

Vi kan bruke variabelen `reg_miss` til plot. Både spredningsplot og boxplot kan gi god innsikt i hvordan observasjoner med missing skiller seg fra andre. Et annet alternativ, er å se på en logistisk regresjon, med den nye dummyen som avhengig variabel. Her fjerner jeg de variablene som fører til flest missing:

Dersom det er mange observasjoner som kastes ut pga missing, som i eksempelet over, er det lurt å danne seg et inntrykk av konsekvense dette får for analysen din. Under skisserer jeg noen måter dere kan bruke R på for å lære mer om missingstruktur:

Metode 1: korrelasjonsmatriser

Korrelasjonsmatriser viser korrelasjoner mellom variabler av klassene `numeric` og `integer`. Dersom vi vil få et raskt inntrykk av konsekvensene av missing i en modell, kan vi lage en korrelasjonsmatrise med variablene som inngår i modellen, og varierer hvordan vi håndterer missing i korrelasjonsmatrisen. Her er et eksempel:

```
# Kjører en modell med litt færre variabler
m1 <- lm(gdp_growth ~ aid*policy + as.factor(period) + ethnic_frac*assasinations, data = aid )
summary(m1) # output viser at 48 observasjoner fjernes pga. missing
```

```
##
## Call:
## lm(formula = gdp_growth ~ aid * policy + as.factor(period) +
##     ethnic_frac * assasinations, data = aid)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.9551  -1.6037   0.0254   1.7550  10.8668
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)                1.73042    0.59379   2.914 0.003864 **
## aid                       -0.09609    0.14747  -0.652 0.515205
## policy                     1.33907    0.20270   6.606 2.08e-10 ***
## as.factor(period)3         0.25055    0.62976   0.398 0.691053
## as.factor(period)4        -1.17377    0.63080  -1.861 0.063861 .
## as.factor(period)5        -2.99437    0.63576  -4.710 3.96e-06 ***
## as.factor(period)6        -1.68657    0.63950  -2.637 0.008839 **
## as.factor(period)7        -2.55448    0.66951  -3.815 0.000168 ***
## ethnic_frac               -1.18048    0.66609  -1.772 0.077474 .
## assasinations             -0.71171    0.30425  -2.339 0.020050 *
## aid:policy                 0.04253    0.09587   0.444 0.657643
## ethnic_frac:assasinations  1.21982    0.62998   1.936 0.053874 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3 on 271 degrees of freedom
##   (48 observations deleted due to missingness)
## Multiple R-squared:  0.343,  Adjusted R-squared:  0.3164
## F-statistic: 12.86 on 11 and 271 DF,  p-value: < 2.2e-16
```

Lager korrelasjonsmatrise med variablene som inngår:

```
# Siden as.factor(period) lager en dummvariabel for alle perioder unntatt periode 1, må vi gjøre dette

aid$period2 <- ifelse(aid$period==2, 1, 0)
aid$period3 <- ifelse(aid$period==3, 1, 0)
aid$period4 <- ifelse(aid$period==4, 1, 0)
aid$period5 <- ifelse(aid$period==5, 1, 0)
aid$period6 <- ifelse(aid$period==6, 1, 0)
aid$period7 <- ifelse(aid$period==7, 1, 0)
aid$period8 <- ifelse(aid$period==8, 1, 0)

aid %>%
  select(gdp_growth,aid,policy, ethnic_frac,assasinations,period2,period3,period4,period5,period6,perio
  cor(, use = "pairwise.complete.obs")
```

```
##                 gdp_growth         aid      policy   ethnic_frac assasinations
## gdp_growth     1.000000000 -0.15872840  0.45346637 -0.1246418798   -0.06381011
## aid           -0.158728399  1.00000000 -0.14758229  0.2857390539   -0.15372098
## policy         0.453466367 -0.14758229  1.00000000 -0.0537569244   -0.01143047
## ethnic_frac   -0.124641880  0.28573905 -0.05375692  1.0000000000   -0.08658713
## assasinations -0.063810109 -0.15372098 -0.01143047 -0.0865871299    1.00000000
## period2        0.212474829 -0.08413737  0.02233928  0.0001371324   -0.06405721
## period3        0.186924657 -0.07696828 -0.06220243  0.0001371324   -0.01306690
## period4        0.006079669 -0.01445413 -0.06367073  0.0001371324    0.05931683
## period5       -0.279642298  0.02653048 -0.16002539  0.0001371324   -0.04193407
## period6       -0.022571291  0.07886175  0.03692845  0.0011598015   -0.04754414
## period7       -0.099934000  0.07290130  0.24028406 -0.0017293741    0.10841360
##                     period2       period3       period4       period5
## gdp_growth     0.2124748287  0.1869246573  0.0060796687 -0.2796422980
## aid           -0.0841373735 -0.0769682790 -0.0144541308  0.0265304812
## policy         0.0223392800 -0.0622024310 -0.0636707325 -0.1600253868
## ethnic_frac    0.0001371324  0.0001371324  0.0001371324  0.0001371324
## assasinations -0.0640572060 -0.0130669004  0.0593168341 -0.0419340675
## period2        1.0000000000 -0.2036363636 -0.2036363636 -0.2036363636
```

```
## period3       -0.2036363636  1.0000000000 -0.2036363636 -0.2036363636
## period4       -0.2036363636 -0.2036363636  1.0000000000 -0.2036363636
## period5       -0.2036363636 -0.2036363636 -0.2036363636  1.0000000000
## period6       -0.1992437293 -0.1992437293 -0.1992437293 -0.1992437293
## period7       -0.1970349207 -0.1970349207 -0.1970349207 -0.1970349207
##                     period6      period7
## gdp_growth    -0.022571291 -0.099934000
## aid            0.078861754  0.072901299
## policy         0.036928446  0.240284058
## ethnic_frac    0.001159801 -0.001729374
## assasinations -0.047544141  0.108413597
## period2       -0.199243729 -0.197034921
## period3       -0.199243729 -0.197034921
## period4       -0.199243729 -0.197034921
## period5       -0.199243729 -0.197034921
## period6        1.000000000 -0.192784686
## period7       -0.192784686  1.000000000
```

```
# Alternativet "pairwise.complete.obs" fjerner bare missing for de enkelte bivariate korrelasjonene
aid %>%
  select(gdp_growth,aid,policy, ethnic_frac,assasinations,period2,period3,period4,period5,period6,perio
  cor(, use = "complete.obs")
```

```
##                 gdp_growth          aid      policy  ethnic_frac assasinations
## gdp_growth     1.000000000 -0.163268793  0.45356785 -0.098074858   -0.07791173
## aid           -0.163268793  1.000000000 -0.14342838  0.266263842   -0.13871882
## policy         0.453567853 -0.143428382  1.00000000 -0.049420321   -0.01217812
## ethnic_frac   -0.098074858  0.266263842 -0.04942032  1.000000000   -0.08864022
## assasinations -0.077911729 -0.138718823 -0.01217812 -0.088640216    1.00000000
## period2        0.184487425 -0.149985596  0.02060803 -0.027435361   -0.05245223
## period3        0.178520491 -0.067445477 -0.06208268 -0.021115228   -0.03014763
## period4       -0.007052255  0.007802611 -0.05839454  0.012015522    0.05882137
## period5       -0.275963915  0.048299871 -0.16229948  0.023063278   -0.04151071
## period6       -0.020471898  0.093558859  0.03511158  0.018297092   -0.04992095
## period7       -0.052385079  0.063462589  0.23898094 -0.006452538    0.11657710
##                    period2     period3      period4     period5     period6
## gdp_growth     0.18448743  0.17852049 -0.007052255 -0.27596391 -0.02047190
## aid           -0.14998560 -0.06744548  0.007802611  0.04829987  0.09355886
## policy         0.02060803 -0.06208268 -0.058394538 -0.16229948  0.03511158
## ethnic_frac   -0.02743536 -0.02111523  0.012015522  0.02306328  0.01829709
## assasinations -0.05245223 -0.03014763  0.058821372 -0.04151071 -0.04992095
## period2        1.00000000 -0.19369514 -0.196081061 -0.19369514 -0.19130014
## period3       -0.19369514  1.00000000 -0.211981106 -0.20940171 -0.20681251
## period4       -0.19608106 -0.21198111  1.000000000 -0.21198111 -0.20936001
## period5       -0.19369514 -0.20940171 -0.211981106  1.00000000 -0.20681251
## period6       -0.19130014 -0.20681251 -0.209360009 -0.20681251  1.00000000
## period7       -0.18161679 -0.19634394 -0.198762490 -0.19634394 -0.19391619
##                    period7
## gdp_growth    -0.052385079
## aid            0.063462589
## policy         0.238980938
## ethnic_frac   -0.006452538
## assasinations  0.116577101
## period2       -0.181616790
## period3       -0.196343939
```

```
## period4       -0.198762490
## period5       -0.196343939
## period6       -0.193916193
## period7        1.000000000
```

```
# Alternativet "complete.obs" fjerner alle observasjoner som har missing på en av variablene som inngår
```

Ved å sammenligne disse korrelasjonsmatrisene, kan vi få et inntrykk av konsekvensene av å fjerne missing med listwise deletion.

Metode 2: Analyse av dummy-variabler for missing

En alternativ metode å utforske missing i en analyse på, er med funksjonen `complete.cases()`, som gjør en logisk test av om en observasjon har missing. Det var denne vi brukte til å lage variabelen `reg_miss` i stad og kjøre en binomisk logistisk modell. Vi skal snakke mer om logistisk regresjon i morgen så jeg går ikke nærmere inn på dette i dag.

```
miss_mod <- glm(reg_miss ~ aid*policy + as.factor(period), data = aid)
summary(miss_mod) # ingen store forskjeller
```

```
##
## Call:
## glm(formula = reg_miss ~ aid * policy + as.factor(period), data = aid)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.96138  -0.00554  -0.00174   0.01552   0.07640
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.009066   0.014559  69.309   <2e-16 ***
## aid                 -0.008781   0.003778  -2.324   0.0208 *
## policy              -0.003314   0.005516  -0.601   0.5485
## as.factor(period)3   0.002482   0.017192   0.144   0.8853
## as.factor(period)4  -0.033326   0.017139  -1.944   0.0529 .
## as.factor(period)5   0.005212   0.017457   0.299   0.7655
## as.factor(period)6   0.005038   0.017557   0.287   0.7744
## as.factor(period)7   0.005878   0.018121   0.324   0.7459
## aid:policy           0.002875   0.002604   1.104   0.2706
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.006795418)
##
##     Null deviance: 1.9860  on 285  degrees of freedom
## Residual deviance: 1.8823  on 277  degrees of freedom
##   (45 observations deleted due to missingness)
## AIC: -605.08
##
## Number of Fisher Scoring iterations: 2
```

```
# I denne modellen ønsker du ikke signifikante uavhengige variabler
```

Koeffisienten til bistand er negativ og signifikant på 5 % signifikansnivå.Dette indikerer at land som fjernes pga missing, får mindre bistand enn land som ikke fjernes.

Vi kunne også definert dummy-variabler for missing på de enkeltvariablene vi er mest interessert i (her: `gdp_growth`, `aid` og `policy`), og gjennomført tilsvarende analyser, ved hjelp av funksjonen `is.na()`.

I de fleste tilfeller er `ifelse()` en fin funksjon til å definere missing. Statistiske R-funksjoner har stort sett et eller flere argumenter der du kan velge hvordan missing skal håndteres (se for eksempel `?cor`, og argumentene `use` og `na.rm`). Husk på det dere har lært på forelesning, og ta aktive valg om hvordan missing bør håndteres.

Vi skal ikke bruke modellelementene mer så derfor fjerner vi de fra environment

```
rm(m1, m5, m5b, miss_mod, model5_usam)
```

```
## Warning in rm(m1, m5, m5b, miss_mod, model5_usam): object 'model5_usam' not
## found
```

Hvordan slår vi sammen flere datasett?

Når vi skal slå sammen ulike datasett må vi først tenke gjennom hvordan vi kan få en felles nøkkel som lar oss knytte sammen informasjon om observasjonene fra de to datasettene. Dette kan gjøres på flere nivåer. Vi jobber videre med aid-datasettet.

```
aid
```

```
## # A tibble: 331 x 30
##    country period periodstart periodend code  gdp_growth gdp_pr_capita
##    <chr>    <dbl>       <dbl>     <dbl> <chr>      <dbl>         <dbl>
##  1 ARG          2        1970      1973 ARG2        1.70          5637
##  2 ARG          3        1974      1977 ARG3        1.08          6168
##  3 ARG          4        1978      1981 ARG4       -1.12          5849
##  4 ARG          5        1982      1985 ARG5       -2.55          5487
##  5 ARG          6        1986      1989 ARG6       -1.10          5624
##  6 ARG          7        1990      1993 ARG7        4.26          4706
##  7 BOL          2        1970      1973 BOL2        1.30          1661
##  8 BOL          3        1974      1977 BOL3        2.96          1838
##  9 BOL          4        1978      1981 BOL4       -1.49          2015
## 10 BOL          5        1982      1985 BOL5       -4.32          1864
## # ... with 321 more rows, and 23 more variables: economic_open <dbl>,
## #   budget_balance <dbl>, inflation <dbl>, ethnic_frac <dbl>,
## #   assasinations <dbl>, aid <dbl>, fast_growing_east_asia <dbl>,
## #   sub_saharan_africa <dbl>, central_america <dbl>, policy <dbl>,
## #   m2_gdp_lagged <dbl>, institutional_quality <dbl>, log_gdp_pr_capita <dbl>,
## #   period_fac <fct>, region <fct>, reg_miss <lgl>, period2 <dbl>,
## #   period3 <dbl>, period4 <dbl>, period5 <dbl>, period6 <dbl>, period7 <dbl>,
## #   period8 <dbl>
```

Ser dere noen variabler her vi kunne brukt som felles nøkkel?

Hvilken variabel vi bruker som nøkkel vil avhenge av variablene i det andre datasettet. Er variablene på landnivå, årnivå, land-år-nivå, region eller noe helt annet? Vi skal nå se på hvordan vi kan slå sammen aid-datasettet med et datasett om konflikt.

Jeg har lastet ned versjon tid av Varieties of democracy datasettet fra V-den sin nettside. I V-dem er det en variabel som heter `v2pepwrsoc`. Denne variabelen måler hvor jevnt makt er fordelt mellom sosiale grupper. Jeg har lastet opp en redusert versjon av V-dem datasettet på github. Det kan du lese inn direkte fra denne lenken.

```
equality <- read_csv("https://raw.githubusercontent.com/liserodland/stv4020aR/master/H20-seminarer/Innf
```

```
## Parsed with column specification:
## cols(
##   country_name = col_character(),
##   country_text_id = col_character(),
##   country_id = col_double(),
```

```
##   year = col_double(),
##   v2pepwrsoc = col_double()
## )
```

```
summary(equality$v2pepwrsoc)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.9150 -0.6210  0.2460  0.3182  1.2780  3.2060
```

```
equality
```

```
## # A tibble: 5,163 x 5
##    country_name country_text_id country_id  year v2pepwrsoc
##    <chr>        <chr>                <dbl> <dbl>      <dbl>
##  1 Mexico       MEX                      3  1966     -0.028
##  2 Mexico       MEX                      3  1967     -0.028
##  3 Mexico       MEX                      3  1968     -0.028
##  4 Mexico       MEX                      3  1969     -0.028
##  5 Mexico       MEX                      3  1970     -0.028
##  6 Mexico       MEX                      3  1971     -0.028
##  7 Mexico       MEX                      3  1972     -0.028
##  8 Mexico       MEX                      3  1973     -0.028
##  9 Mexico       MEX                      3  1974     -0.028
## 10 Mexico       MEX                      3  1975     -0.028
## # ... with 5,153 more rows
```

```
# Vi ser at V-dem har en variabel som heter country_text_id og year
# Kanskje vi kan bruke disse?

# Bruker en logisk test og %in% for å sjekke om det finnes en match for alle land i aid-datasettet:
table(aid$country %in% equality$country_text_id)
```

```
##
## FALSE  TRUE
##     6   325
```

```
# Ikke alle matcher.
```

Når ikke alle observasjonen har en match så kan dette kan enten løses manuelt eller ved hjelp av andre datasett eller R-pakker.

```
# For å løse det manuelt så kan du bruke denne koden til å identifisere de som ikke matcher:
aid %>%
  select(country) %>%  # Velger country variabelen i aid
  anti_join(equality, by = c("country" = "country_text_id")) %>% # Bevarer de verdiene i equality som i
  unique()
```

```
## # A tibble: 1 x 1
##   country
##   <chr>
## 1 ZAR
```

```
# En nyttig pakke dersom dere kommer over dette problemet kan være countrycode
```

Vi kommer ikke til å bruke tid i seminar på å rette opp i dette, men her finner dere et eksempel på hvordan det kunne vært løst. Vi går derfor videre vel vitende om at vi ikke klarte å matche alle observasjonen (dette anbefaler jeg **ikke** å gjøre i hjemmeoppgaven). Det er fortsatt en ting vi må gjøre før vi kan slå datasettene sammen. V-dem-datasettet inneholder land-år-observasjoner, mens aid-datasettet inneholder land-periode-observasjoner. Vi må derfor lage en periode-variabel i equality-datasettet.

```r
# Oppretter periode-variabel i V-dem datasettet, slik at jeg er klar til å merge. Verdiene til period-v
table(aid$periodstart, aid$period)
```

```
##
##         2  3  4  5  6  7
##   1970 56  0  0  0  0  0
##   1974  0 56  0  0  0  0
##   1978  0  0 56  0  0  0
##   1982  0  0  0 56  0  0
##   1986  0  0  0  0 54  0
##   1990  0  0  0  0  0 53
```

```r
table(aid$periodend, aid$period)
```

```
##
##         2  3  4  5  6  7
##   1973 56  0  0  0  0  0
##   1977  0 56  0  0  0  0
##   1981  0  0 56  0  0  0
##   1985  0  0  0 56  0  0
##   1989  0  0  0  0 54  0
##   1993  0  0  0  0  0 53
```

```r
# Det kommer ikke tydelig frem her, men datasettet gikk opprinnelig fra 1966-1998
# Dersom jeg bruker 1966, 1970, 1974, 1978, 1982, 1986, 1990 og 1994 som kuttpunkt,
# bør jeg få de samme gruppene i V-dem-datasettet som i aid

periodcutpoints <-  unique(c(aid$periodstart)) # henter ut ovennevnt årtsall med unique()
# Her buker jeg funksjonen cut(), jeg kunne også brukt ifelse(), men cut() er raskere her.
equality$period <- cut(equality$year, periodcutpoints)
table(equality$year, equality$period)
```

```
##
##        (1970,1974] (1974,1978] (1978,1982] (1982,1986] (1986,1990]
##   1966           0           0           0           0           0
##   1967           0           0           0           0           0
##   1968           0           0           0           0           0
##   1969           0           0           0           0           0
##   1970           0           0           0           0           0
##   1971         158           0           0           0           0
##   1972         158           0           0           0           0
##   1973         158           0           0           0           0
##   1974         158           0           0           0           0
##   1975           0         158           0           0           0
##   1976           0         157           0           0           0
##   1977           0         157           0           0           0
##   1978           0         157           0           0           0
##   1979           0           0         157           0           0
##   1980           0           0         157           0           0
##   1981           0           0         157           0           0
##   1982           0           0         157           0           0
##   1983           0           0           0         157           0
##   1984           0           0           0         157           0
##   1985           0           0           0         157           0
##   1986           0           0           0         157           0
```

```
##   1987           0           0           0           0         157
##   1988           0           0           0           0         157
##   1989           0           0           0           0         158
##   1990           0           0           0           0         172
##   1991           0           0           0           0           0
##   1992           0           0           0           0           0
##   1993           0           0           0           0           0
##   1994           0           0           0           0           0
##   1995           0           0           0           0           0
##   1996           0           0           0           0           0
##   1997           0           0           0           0           0
```

```
# Tabell viser at jeg må justere periodcutpoints for å få rett

periodcutpoints <- periodcutpoints - 1
table(periodcutpoints)
```

```
## periodcutpoints
## 1969 1973 1977 1981 1985 1989
##    1    1    1    1    1    1
```

```
periodcutpoints <- c(1965, periodcutpoints, 1993, 1997) # legger til tre kuttpunkt for å få med periode

# Forsøker på nytt:
equality$period <- cut(equality$year, periodcutpoints)
table(equality$year,equality$period)
```

```
##
##        (1965,1969] (1969,1973] (1973,1977] (1977,1981] (1981,1985] (1985,1989]
##   1966         156           0           0           0           0           0
##   1967         156           0           0           0           0           0
##   1968         156           0           0           0           0           0
##   1969         156           0           0           0           0           0
##   1970           0         156           0           0           0           0
##   1971           0         158           0           0           0           0
##   1972           0         158           0           0           0           0
##   1973           0         158           0           0           0           0
##   1974           0           0         158           0           0           0
##   1975           0           0         158           0           0           0
##   1976           0           0         157           0           0           0
##   1977           0           0         157           0           0           0
##   1978           0           0           0         157           0           0
##   1979           0           0           0         157           0           0
##   1980           0           0           0         157           0           0
##   1981           0           0           0         157           0           0
##   1982           0           0           0           0         157           0
##   1983           0           0           0           0         157           0
##   1984           0           0           0           0         157           0
##   1985           0           0           0           0         157           0
##   1986           0           0           0           0           0         157
##   1987           0           0           0           0           0         157
##   1988           0           0           0           0           0         157
##   1989           0           0           0           0           0         158
##   1990           0           0           0           0           0           0
##   1991           0           0           0           0           0           0
```

```
##   1992           0           0           0           0           0           0
##   1993           0           0           0           0           0           0
##   1994           0           0           0           0           0           0
##   1995           0           0           0           0           0           0
##   1996           0           0           0           0           0           0
##   1997           0           0           0           0           0           0
##
##        (1989,1993] (1993,1997]
##   1966           0           0
##   1967           0           0
##   1968           0           0
##   1969           0           0
##   1970           0           0
##   1971           0           0
##   1972           0           0
##   1973           0           0
##   1974           0           0
##   1975           0           0
##   1976           0           0
##   1977           0           0
##   1978           0           0
##   1979           0           0
##   1980           0           0
##   1981           0           0
##   1982           0           0
##   1983           0           0
##   1984           0           0
##   1985           0           0
##   1986           0           0
##   1987           0           0
##   1988           0           0
##   1989           0           0
##   1990         172           0
##   1991         173           0
##   1992         174           0
##   1993         175           0
##   1994           0         175
##   1995           0         175
##   1996           0         175
##   1997           0         175
```

```
equality$period <- as.numeric(as_factor(equality$period))

table(equality$year,equality$period)
```

```
##
##          1   2   3   4   5   6   7   8
##   1966 156   0   0   0   0   0   0   0
##   1967 156   0   0   0   0   0   0   0
##   1968 156   0   0   0   0   0   0   0
##   1969 156   0   0   0   0   0   0   0
##   1970   0 156   0   0   0   0   0   0
##   1971   0 158   0   0   0   0   0   0
##   1972   0 158   0   0   0   0   0   0
##   1973   0 158   0   0   0   0   0   0
```

```
##   1974   0   0 158   0   0   0   0   0
##   1975   0   0 158   0   0   0   0   0
##   1976   0   0 157   0   0   0   0   0
##   1977   0   0 157   0   0   0   0   0
##   1978   0   0   0 157   0   0   0   0
##   1979   0   0   0 157   0   0   0   0
##   1980   0   0   0 157   0   0   0   0
##   1981   0   0   0 157   0   0   0   0
##   1982   0   0   0   0 157   0   0   0
##   1983   0   0   0   0 157   0   0   0
##   1984   0   0   0   0 157   0   0   0
##   1985   0   0   0   0 157   0   0   0
##   1986   0   0   0   0   0 157   0   0
##   1987   0   0   0   0   0 157   0   0
##   1988   0   0   0   0   0 157   0   0
##   1989   0   0   0   0   0 158   0   0
##   1990   0   0   0   0   0   0 172   0
##   1991   0   0   0   0   0   0 173   0
##   1992   0   0   0   0   0   0 174   0
##   1993   0   0   0   0   0   0 175   0
##   1994   0   0   0   0   0   0   0 175
##   1995   0   0   0   0   0   0   0 175
##   1996   0   0   0   0   0   0   0 175
##   1997   0   0   0   0   0   0   0 175
```

```
# Ser fint ut
```

Da har vi forhåpentligvis variabler som kan fungere som nøkler i begge datasettene. Neste steg er å endre på datastrukturen i datasettet `equality`, slik at den blir lik som i `aid`. For å få til dette, må vi endre observasjonene i `equality` til land-perioder. Dette kan vi gjøre med `group_by` og `summarise()`. På dette stadiet, må vi tenke datastruktur, og gjøre metodologiske valg om hvordan vi skal operasjonalisere informasjonen om konflikter. Under viser jeg to muligheter. I hjemmeoppgaven er dette et punkt der jeg vil anbefale at du tenker grundig gjennom de metodologiske implikasjonene av valgene du tar - tenk gjennom hva som er best og skriv koden din etterpå - ikke fall i fellen kode først, metode etterpå.

```
agg_equality <- equality %>%
  group_by(country_text_id, period) %>%
  summarise(avg_eq = mean(v2pepwrsoc, na.rm = TRUE)) %>% # regner ut gjennomsnittet for perioden
  mutate(period_num = as.numeric(period))
```

```
## `summarise()` regrouping output by 'country_text_id' (override with `.groups` argument)
```

```
table(agg_equality$period, agg_equality$period_num)
```

```
##
##       1   2   3   4   5   6   7   8
##   1 157   0   0   0   0   0   0   0
##   2   0 158   0   0   0   0   0   0
##   3   0   0 158   0   0   0   0   0
##   4   0   0   0 157   0   0   0   0
##   5   0   0   0   0 157   0   0   0
##   6   0   0   0   0   0 158   0   0
##   7   0   0   0   0   0   0 177   0
##   8   0   0   0   0   0   0   0 175
```

```
agg_equality
```

```
## # A tibble: 1,297 x 4
## # Groups:   country_text_id [179]
##    country_text_id period avg_eq period_num
##    <chr>            <dbl>  <dbl>      <dbl>
##  1 AFG                  1  1.02           1
##  2 AFG                  2  1.02           2
##  3 AFG                  3  1.01           3
##  4 AFG                  4  0.845          4
##  5 AFG                  5  0.845          5
##  6 AFG                  6  0.845          6
##  7 AFG                  7  0.587          7
##  8 AFG                  8 -0.133          8
##  9 AGO                  1 -2.52           1
## 10 AGO                  2 -2.52           2
## # ... with 1,287 more rows
```

Nå som data fra `equality` er i samme format som i `aid`, er vi klare til å kombinere informasjonen med `left_join`:

```
# husk: ?left_join for å forstå funksjonen
aid2 <- left_join(aid, agg_equality,
                  by = c("country" = "country_text_id", "period" = "period_num")) # Spesifiserer nøkkel
# Sjekker missing:
table(is.na(aid2$avg_eq))
```

```
##
## FALSE  TRUE
##   325     6
```

```
# 6 missing pga observasjonen som mangler

summary(aid2$avg_eq)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## -2.2160 -0.6338  0.0530  0.1102  0.8920  2.3890       6
```