Title

Course

Christian Oppegård Moen

DD MM YYYY

Contents

Libraries & misc

```
library(Rmisc)
library(glmm)
library(ggplot2)
library(tidyr)
library(dplyr)
library(nlme)
library(lme4)
```

```
defaultMar = c(5.1, 4.1, 4.1, 2.1)
```

Data

Data format is of tab separated values regarding noise on campus. Some variables are shown in the printout below. The response is `spmTriv` and `spmEff` which are "trivsel" and "effektivitet", respectively. The response is evaluated for each building block.

Load data and reformat

The time is on a unfeasible format, so we format it to total seconds used.

```
pathData = "./data"
d = read.delim("./data/data-315297-2023-03-08-1434-utf.txt", header = T)

# Reformat the time to be total time in seconds
formatTime <- function(t) {
    tSplit = strsplit(t, " ")[[1]]
    s = 0
    for (i in seq(1, length(tSplit), 2)) {
        s = s + switch(tSplit[i + 1], dag = strtoi(tSplit[i]) * 24 * 3600, dager = strtoi(tSplit[i]) *
            24 * 3600, time = strtoi(tSplit[i]) * 3600, timer = strtoi(tSplit[i]) *
            3600, minutt = strtoi(tSplit[i]) * 60, minutter = strtoi(tSplit[i]) *
            60, sekund = strtoi(tSplit[i]), sekunder = strtoi(tSplit[i]), 0)
    }
    return(s)
}

ftimes = unlist(lapply(d$Svartid, formatTime))
head(cbind(old = d$Svartid, new = ftimes))
```

```
##      old                       new
## [1,] "2 minutter 32 sekunder" "152"
## [2,] "1 minutt 58 sekunder"   "118"
## [3,] "2 minutter 32 sekunder" "152"
## [4,] "4 minutter 28 sekunder" "268"
## [5,] "2 minutter 45 sekunder" "165"
## [6,] "2 minutter 41 sekunder" "161"
```

```
d$Svartid = ftimes
```

Initial values

In this section we take the initial look at the data by considering sample means, SDs, conf.ints and some graphs being plots of said values, histograms and more.

"How many" numbers

How many did something about the noise.

```
# head(d) head(d[,18:(18+6)])
dTiltak = d[, 18:(18 + 6)]
hasAns = function(x) {
    return(sum(x != ""))
}
dTiltakSum = apply(dTiltak, FUN = hasAns, MARGIN = 2)
dTiltakSum
```

```
## spmTiltak_1 spmTiltak_2 spmTiltak_3 spmTiltak_4 spmTiltak_5 spmTiltak_6
##         109          26          69          10          66          10
## spmTiltak_7
##           6
```

Total mean, SD, CI

We compute the sample mean, SD and confidence intervals of `effektivitet` and `trivsel` of the entire data set. We also take a look at their distribution by looking at the histograms.

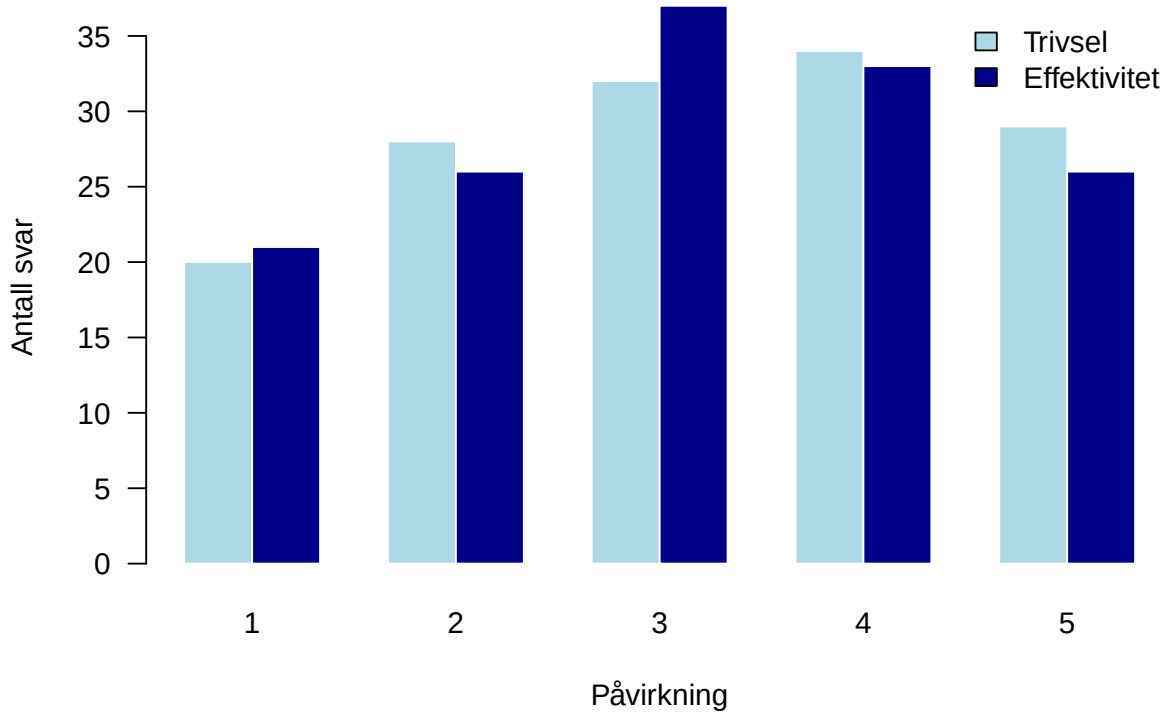
```
effTrivInit = cbind(t(apply(d[, c('spmEff', 'spmTriv')], 2, FUN = CI)), c(sd(d$spmEff), sd(d$spmTriv)))
colnames(effTrivInit) = c(colnames(effTrivInit)[1:3], 'SD')
effTrivInit
```

```
##           upper     mean    lower       SD
## spmEff  3.33601 3.118881 2.901752 1.313470
## spmTriv 3.38891 3.167832 2.946754 1.337361
```

```
# Histograms
histCounts = as.data.frame(rbind(hist(d$spmTriv, breaks = seq(0.5,5.5,1), plot = F)$counts,
              hist(d$spmEff, breaks = seq(0.5,5.5,1), plot = F)$counts))
rownames(histCounts) = c('trivsel', 'effektivitet')
colnames(histCounts) = 1:5

histCounts
```

```
##                1  2  3  4  5
## trivsel       20 28 32 34 29
## effektivitet 21 26 37 33 26
```

```
barplot(
  ((as.matrix(histCounts))),
  col = c("lightblue", 'darkblue'),
  border = "white",
  # main="Trivsel",
  ylab="Antall svar",
  xlab = "P\u00E5virkning",
  beside = T,
  las=1
  # ylim = c(0,5)
  # space=0.1
)
legend(
  "topright",
  legend = c("Trivsel", "Effektivitet"),
  fill = c("lightblue", 'darkblue'), bty = 'n')
```



Location Analysis

Let's look at some initial location based data. First, how many answers per building.

```r
# All locations/buildings
locs = unique(d$spmHvor) # all locations in this dataset
buildInit = data.frame(
  matrix(ncol=length(locs),
         nrow=5,
         dimnames=list(c('count', 'meanEff', 'SDEff', 'meanTriv', 'SDTriv'), locs))
)

for (loc in locs){
  buildInit[loc] = c(sum(d$spmHvor==loc),
              mean(d$spmEff[d$spmHvor==loc]),
               sd(d$spmEff[d$spmHvor==loc]),
              mean(d$spmTriv[d$spmHvor==loc]),
               sd(d$spmTriv[d$spmHvor==loc]))
}

# Count, mean and standard deviation per building
round(buildInit,1)
```

```
##          elD elB kjel  elE real sent2 verk bygg sent1 hand tapir berg gamEl
## count    4.0 7.0  6.0 20.0 30.0   4.0  9.0  4.0  11.0  4.0   8.0  3.0   8.0
## meanEff  3.5 3.9  3.0  4.0  2.7   2.2  3.2  2.8   2.9  4.0   2.2  1.3   3.9
## SDEff    1.0 1.2  1.4  1.1  1.3   1.0  1.3  1.3   1.4  1.2   0.9  0.6   1.4
## meanTriv 3.8 3.7  3.7  4.3  2.6   2.5  2.8  2.5   2.5  3.2   2.8  1.3   3.9
## SDTriv   0.5 0.8  1.8  0.8  1.3   1.3  0.8  1.3   1.3  1.3   1.0  0.6   1.2
##          hoved varme gruve gamFys itSyd kjemi1 kjemi2 elA kjemi5 ipd
## count      5.0   4.0   5.0    2.0     1    3.0      1 2.0      1   1
## meanEff    2.6   3.5   2.4    4.0     4    3.3      4 4.0      2   3
## SDEff      1.1   1.7   0.9    0.0    NA    1.5     NA 1.4     NA  NA
## meanTriv   2.8   3.5   3.2    3.5     2    3.0      5 4.5      2   4
## SDTriv     1.3   1.7   1.6    0.7    NA    2.0     NA 0.7     NA  NA
```

```r
# Sorted values for readability
sort(round(buildInit['meanTriv',],1))
```

```
##          berg itSyd kjemi5 sent2 bygg sent1 real verk tapir hoved kjemi1 hand
## meanTriv  1.3     2      2   2.5  2.5   2.5  2.6  2.8   2.8   2.8      3  3.2
##          gruve varme gamFys elB kjel elD gamEl ipd elE elA kjemi2
## meanTriv   3.2   3.5    3.5 3.7  3.7 3.8   3.9   4 4.3 4.5      5
```

```r
sort(round(buildInit['meanEff',],1))
```

```
##         berg kjemi5 sent2 tapir gruve hoved real bygg sent1 kjel ipd verk
## meanEff  1.3      2   2.2   2.2   2.4   2.6  2.7  2.8   2.9    3   3  3.2
##         kjemi1 elD varme elB gamEl elE hand gamFys itSyd kjemi2 elA
## meanEff    3.3 3.5   3.5 3.9   3.9   4    4      4     4      4   4
```

```r
sort(round(buildInit['SDTriv',],1))
```

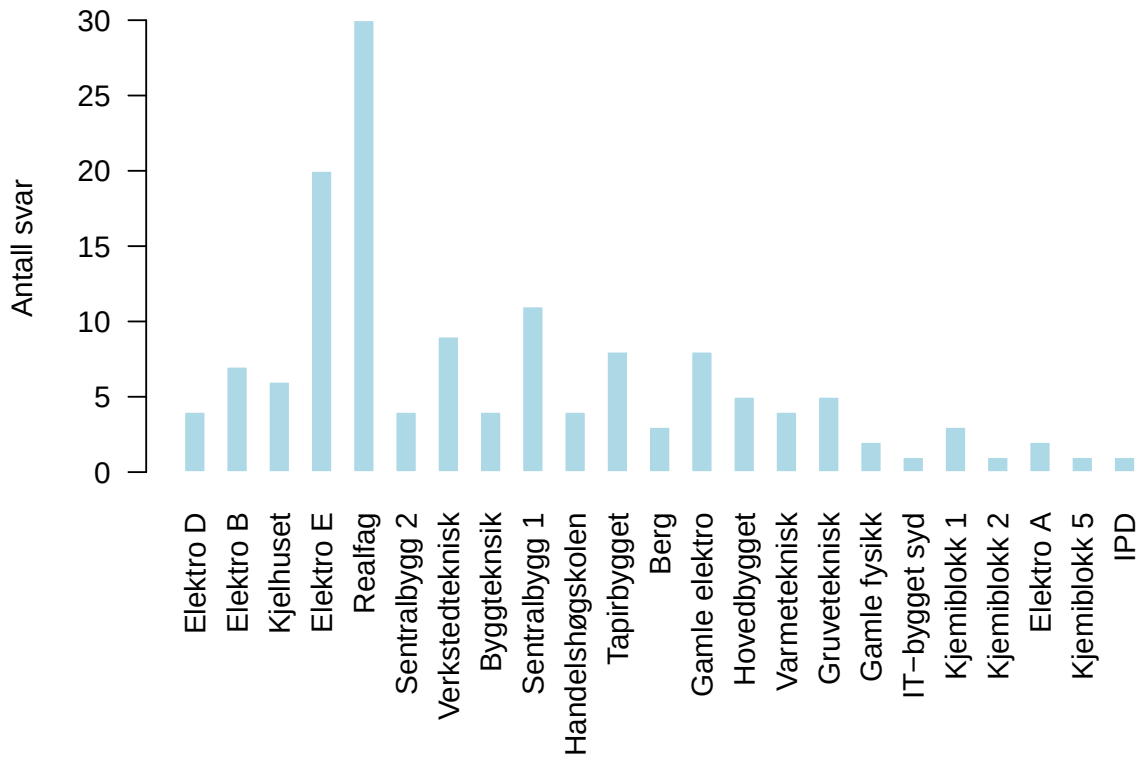
```
##        elD berg gamFys elA elB elE verk tapir gamEl real sent2 bygg sent1 hand
```

```
## SDTriv 0.5  0.6    0.7 0.7 0.8 0.8  0.8     1   1.2  1.3   1.3  1.3   1.3  1.3
##        hoved gruve varme kjel kjemi1
## SDTriv   1.3   1.6   1.7  1.8      2
```

```
sort(round(buildInit['SDEff',],1))
```

```
##       gamFys berg tapir gruve elD sent2 elE hoved elB hand real verk bygg kjel
## SDEff      0  0.6   0.9   0.9   1     1 1.1   1.1 1.2  1.2  1.3  1.3  1.3  1.4
##       sent1 gamEl elA kjemi1 varme
## SDEff   1.4   1.4 1.4    1.5   1.7
```

```
# Barplot of counts for readability
buildNames = c(
  'Elektro D',
  "Elektro B",
  "Kjelhuset",
  "Elektro E",
  "Realfag",
  "Sentralbygg 2",
  "Verkstedteknisk",
  "Byggteknsik",
  "Sentralbygg 1",
  "Handelsh\u00F8gskolen",
  "Tapirbygget",
  "Berg",
  "Gamle elektro",
  "Hovedbygget",
  "Varmeteknisk",
  "Gruveteknisk",
  "Gamle fysikk",
  "IT-bygget syd",
  "Kjemiblokk 1",
  "Kjemiblokk 2",
  "Elektro A",
  "Kjemiblokk 5",
  "IPD"
)
par(mar=defaultMar+c(3,0,0,0))
barplot(
  ((as.matrix(buildInit['count',]))),
  col = c("lightblue"),
  border = "white",
  # main="Trivsel",
  ylab="Antall svar",
  names.arg = buildNames,
  # xlab = "Lokasjon",
  beside = T,
  las=2
  # ylim = c(0,5)
  # space=0.1
)
```




```
par(mar=defaultMar)
```

Barplot of responses

Compute responses

```
# initiate per building empirical mean data frame
buildMeans = data.frame(matrix(ncol = length(locs), nrow = 2, dimnames = list(c("trivsel",
    "effektivitet"), locs)))
# initiate per building standard deviation data frame
buildSd = data.frame(matrix(ncol = length(locs), nrow = 2, dimnames = list(c("trivsel",
    "effektivitet"), locs)))
# Compute means and SDs
for (loc in locs) {
    buildMeans[loc] = c(mean(d$spmTriv[d$spmHvor == loc]), mean(d$spmEff[d$spmHvor ==
        loc]))
    buildSd[loc] = c(sd(d$spmTriv[d$spmHvor == loc]), sd(d$spmEff[d$spmHvor == loc]))
}

# Mean values per location
round(sort(buildMeans["trivsel", ]), 2)
```

```
##         berg itSyd kjemi5 sent2 bygg sent1 real tapir verk hoved kjemi1 gruve
```

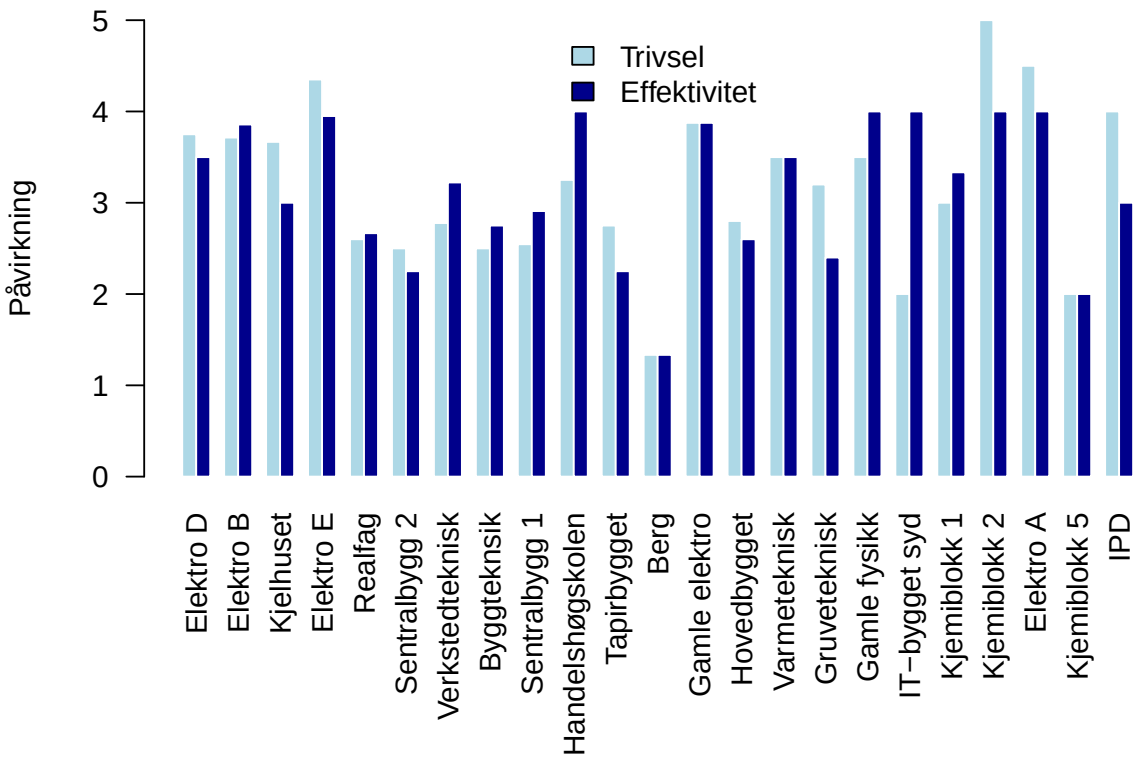
```
## trivsel 1.33     2      2   2.5  2.5  2.55  2.6  2.75 2.78   2.8      3   3.2
##         hand varme gamFys kjel  elB  elD gamEl ipd  elE elA kjemi2
## trivsel 3.25   3.5    3.5 3.67 3.71 3.75  3.88   4 4.35 4.5      5
```

```
round(sort(buildMeans["effektivitet", ]), 2)
```

```
##              berg kjemi5 sent2 tapir gruve hoved real bygg sent1 kjel ipd verk
## effektivitet 1.33      2  2.25  2.25   2.4   2.6 2.67 2.75  2.91    3   3 3.22
##              kjemi1 elD varme  elB gamEl  elE hand gamFys itSyd kjemi2 elA
## effektivitet   3.33 3.5   3.5 3.86  3.88 3.95    4      4     4      4   4
```

Plot responses

```
# æ is \u00E6
# ø is \u00F8
# å is \u00E5
par(mar=defaultMar+c(3,0,0,0))
barplot(
  ((as.matrix(buildMeans))),
  col = c("lightblue", 'darkblue'),
  border = "white",
  # main="Trivsel",
  ylab="P\u00E5virkning",
  beside = T,
  las=2,
  names.arg = buildNames,
  ylim = c(0,5)
  # space=0.1
)

legend(
  "top",
  legend = c("Trivsel", "Effektivitet"),
  fill = c("lightblue", 'darkblue'), bty = 'n')
```




```
d['Elektro_D.B2',]
```

```
##    NR Opprettet Endret spmFunk spmHvor spmTriv byggestoy1 personstoy1 trafikk1
## NA NA       <NA>   <NA>    <NA>    <NA>      NA          NA           NA       NA
##    vifte1 annet1 spmEff byggestoy2 personstoy2 trafikk2 vifte2 annet2
## NA     NA      NA      NA          NA           NA        NA     NA      NA
##    spmTiltak_1 spmTiltak_2 spmTiltak_3 spmTiltak_4 spmTiltak_5 spmTiltak_6
## NA        <NA>        <NA>        <NA>        <NA>        <NA>        <NA>
##    spmTiltak_7 spmTekst Svartid
## NA        <NA>     <NA>      NA
```

```
head(d)
```

```
##         NR        Opprettet           Endret spmFunk spmHvor spmTriv byggestoy1
## 1 25713468 08.02.2023 10:59 08.02.2023 10:59    stud     elD       4          4
## 2 25713476 08.02.2023 11:00 08.02.2023 11:00    stud     elB       3          4
## 3 25713498 08.02.2023 11:00 08.02.2023 11:00    stud    kjel       1          1
## 4 25713537 08.02.2023 11:02 08.02.2023 11:02    stud     elE       3          1
## 5 25713545 08.02.2023 11:03 08.02.2023 11:03    stud     elB       4          5
## 6 25714321 08.02.2023 11:44 08.02.2023 11:44    stud    real       3          1
##   personstoy1 trafikk1 vifte1 annet1 spmEff byggestoy2 personstoy2 trafikk2
## 1           2        1      2      5      4          5           3        1
## 2           1        1      3      1      2          3           1        1
## 3           2        1      1      1      1          1           3        1
```

```
## 4           2        1      1      1      1          1           1        1
## 5           2        1      1      1      3          5           2        1
## 6           4        1      3      1      4          4           3        4
##   vifte2 annet2 spmTiltak_1 spmTiltak_2 spmTiltak_3 spmTiltak_4 spmTiltak_5
## 1      1      5                                                  ingenting
## 2      3      1                                                  ingenting
## 3      2      1         mNc                                      ingenting
## 4      1      1         mNc                  flytt               ingenting
## 5      1      1                                                  ingenting
## 6      5      3         mNc                  flytt
##   spmTiltak_6 spmTiltak_7 spmTekst Svartid
## 1                                     152
## 2                                     118
## 3     paavirk                         152
## 4                                     268
## 5                                     165
## 6                                     161
```

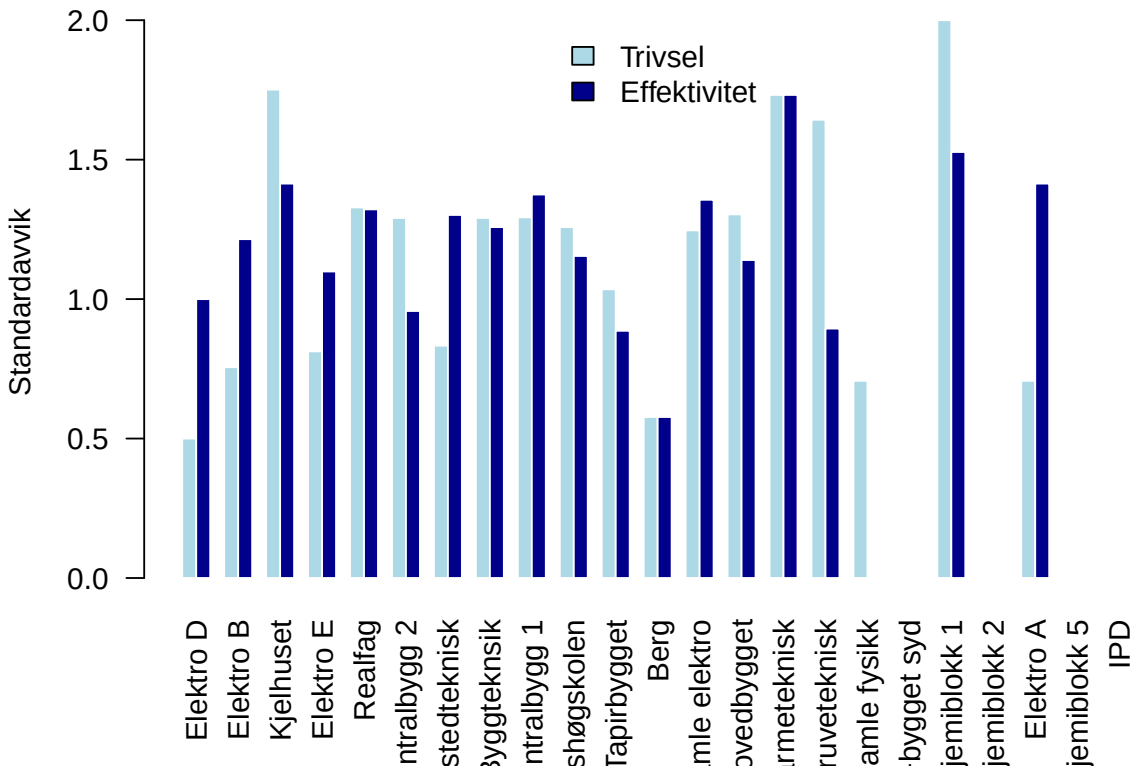
```
mTriv= max(buildMeans['trivsel',])
mEff = max(buildMeans['effektivitet',])
buildMeans['effektivitet',buildMeans['effektivitet',]==mEff]
```

```
##              hand gamFys itSyd kjemi2 elA
## effektivitet    4      4     4      4   4
```

```
buildMeans['trivsel',buildMeans['trivsel',]==mTriv]
```

```
## [1] 5
```

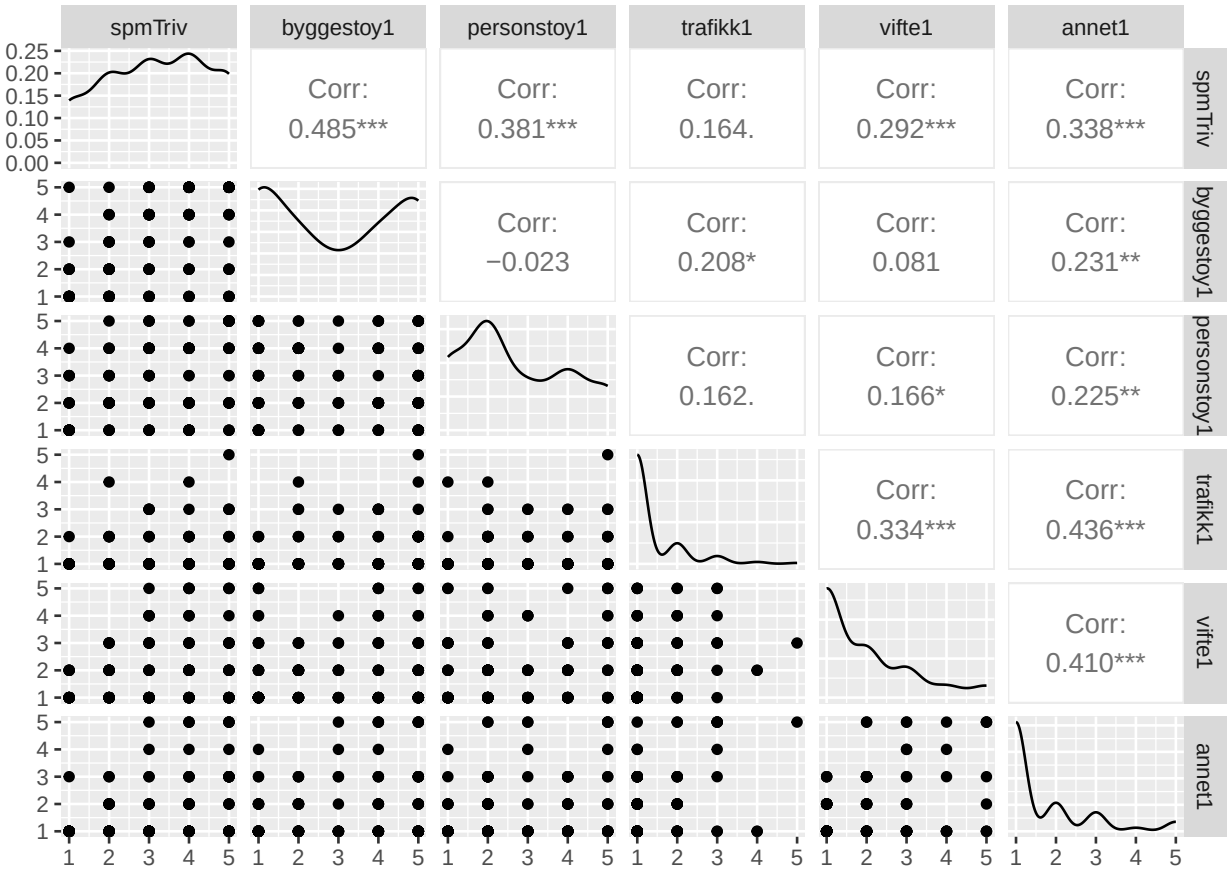
```
barplot(
  ((as.matrix(buildSd))),
  col = c("lightblue", 'darkblue'),
  border = "white",
  # main="Trivsel",
  ylab="Standardavvik",
  beside = T,
  las=2,
  names.arg = buildNames
  # ylim = c(0,5)
  # space=0.1
)
legend(
  "top",
  legend = c("Trivsel", "Effektivitet"),
  fill = c("lightblue", 'darkblue'), bty = 'n')
```



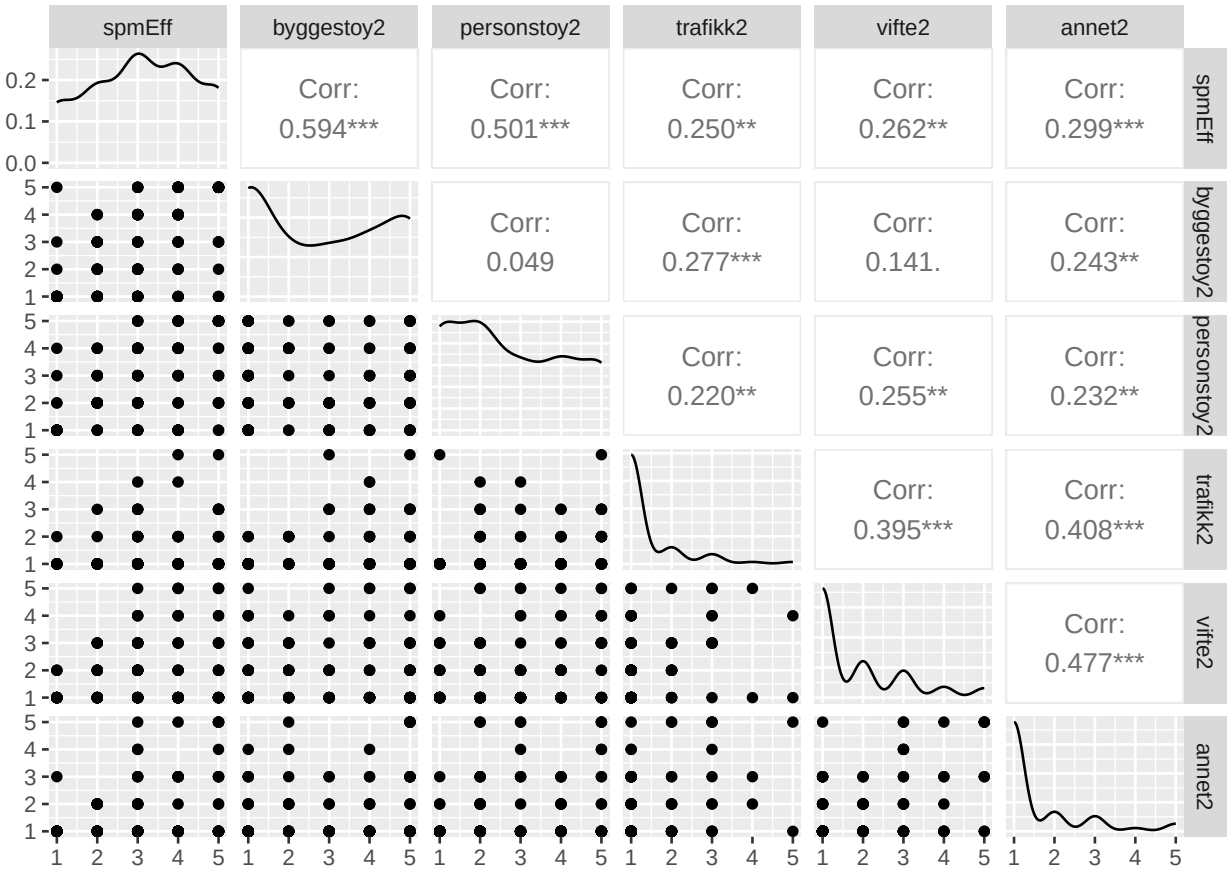
Pairs plots

Visualization of each data value compared to each other. Neat for initial data observations.

```
# pairs(dNumeric[,2:7])
library(GGally)
dNumeric = select_if(d, is.numeric)
ggpairs(dNumeric[, 2:7])  # Trivsel pairs
```




```
ggpairs(dNumeric[, 8:13])  # Effektivitet pairs
```



Noise type analysis

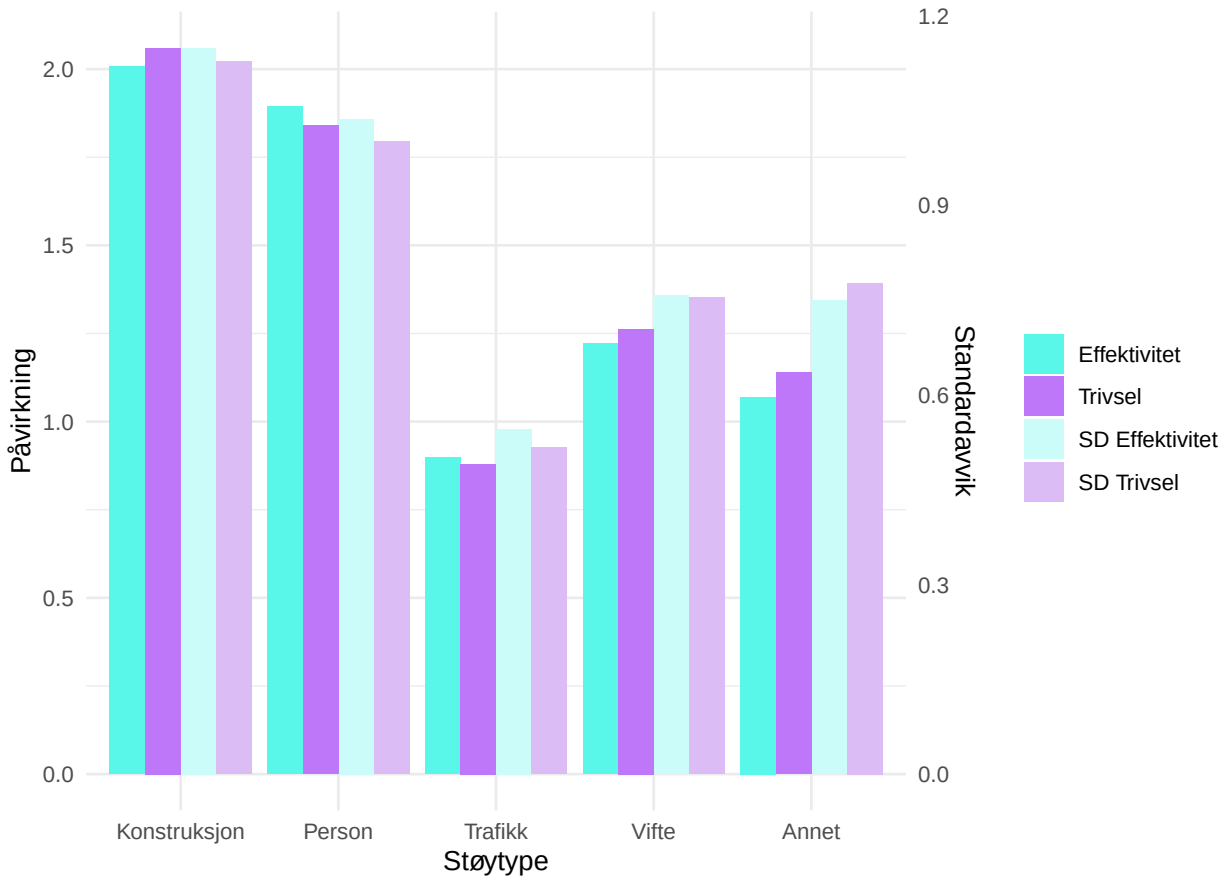
In this section we will take a look at each noise type and how much people feel like it influences their efficiency and well-being.

Noise type mean and SD barplot. (Discarded)

```
# setup
ntColnames = c("Konstruksjon", "Person", "Trafikk", "Vifte", "Annet")
nts = c("Konstruksjon", "Person", "Trafikk", "Vifte", "Annet")
ntTriv = colnames(d)[7:11]
ntEff = colnames(d)[13:17]
```

```
# compute mean and sd
ntDf = data.frame(cbind(nts = 1:length(nts), meanEff = apply(d[, ntEff], FUN = mean,
    MARGIN = 2), sdEff = apply(d[, ntEff], FUN = sd, MARGIN = 2), meanTriv = apply(d[,
    ntTriv], FUN = mean, MARGIN = 2), sdTriv = apply(d[, ntTriv], FUN = sd, MARGIN = 2)))

# compute mean and sd weighted
ntDfW = data.frame(cbind(nts = 1:length(nts), meanEff = apply(d[, ntEff] * (d$spmEff/5),
```

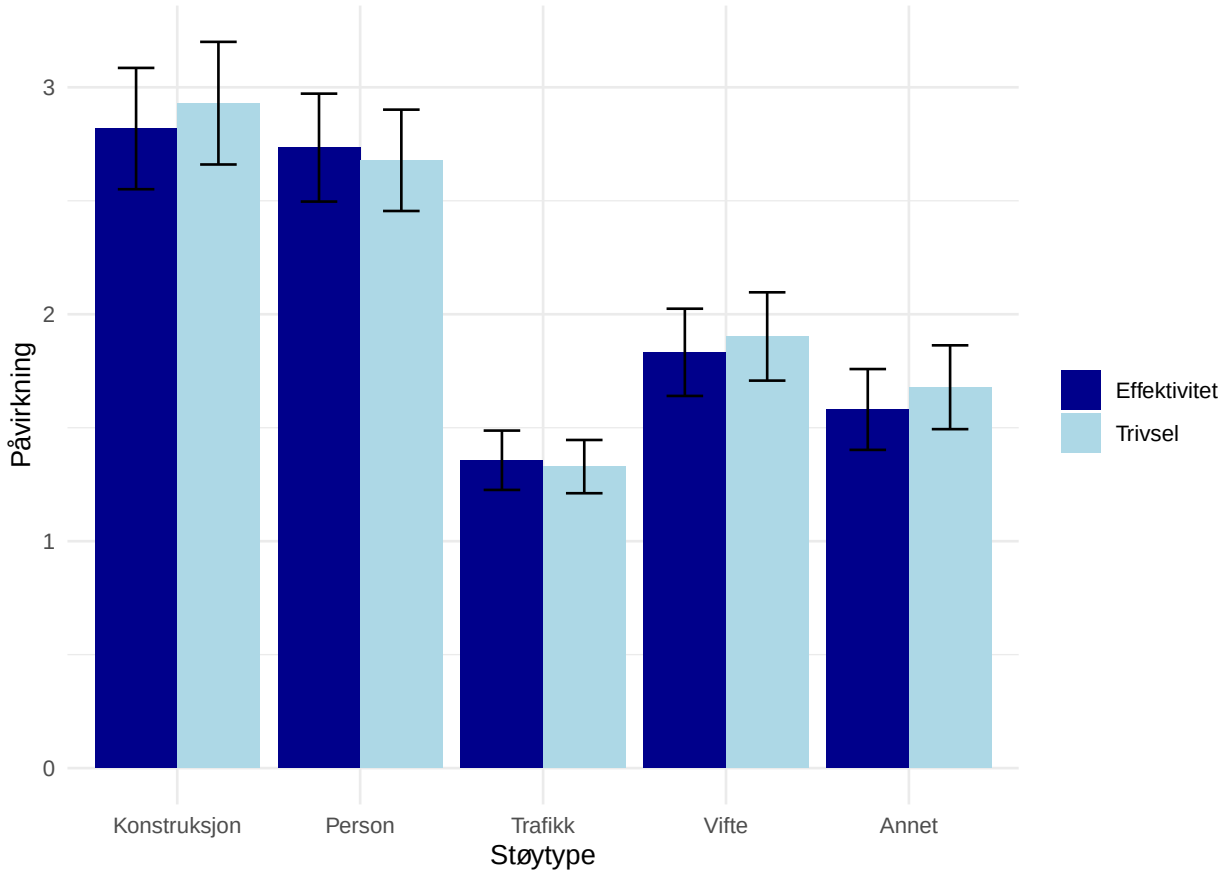
```
    FUN = mean, MARGIN = 2), sdEff = apply(d[, ntEff] * (d$spmEff/5), FUN = sd, MARGIN = 2),
    meanTriv = apply(d[, ntTriv] * (d$spmEff/5), FUN = mean, MARGIN = 2), sdTriv = apply(d[,
        ntTriv] * (d$spmEff/5), FUN = sd, MARGIN = 2)))

# Plot not weighted
ntDf_maxMean = max(ntDf[, c("meanEff", "meanTriv")])
ntDf_maxSD = max(ntDf[, c("sdEff", "sdTriv")])

ntDfLong <- ntDf |>
    pivot_longer(cols = -nts, names_to = "Type") |>
    mutate(scaled_value = ifelse(Type %in% c("meanEff", "meanTriv"), value, value/ntDf_maxSD *
        ntDf_maxMean))


# Plot Weighted
ntDfW_maxMean = max(ntDfW[, c("meanEff", "meanTriv")])
ntDfW_maxSD = max(ntDfW[, c("sdEff", "sdTriv")])

ntDfWLong <- ntDfW |>
    pivot_longer(cols = -nts, names_to = "Type") |>
    mutate(scaled_value = ifelse(Type %in% c("meanEff", "meanTriv"), value, value/ntDfW_maxSD *
        ntDfW_maxMean))

ggplot(ntDfWLong, aes(x = nts, y = scaled_value, fill = Type)) + geom_col(position = "dodge") +
    scale_y_continuous(sec.axis = sec_axis(~./ntDf_maxMean * ntDf_maxSD, name = "Standardavvik")) +
    scale_fill_manual(values = c("#59f7ea", "#be77f9", "#ccfcf9", "#dbbcf5"), labels = c("Effektivitet"
        "Trivsel", "SD Effektivitet", "SD Trivsel")) + labs(y = "Påvirkning", x = "Støytype") +
    scale_x_discrete(limit = nts) + theme_minimal() + theme(legend.title = element_blank())
```



Noise types with confidence intervals

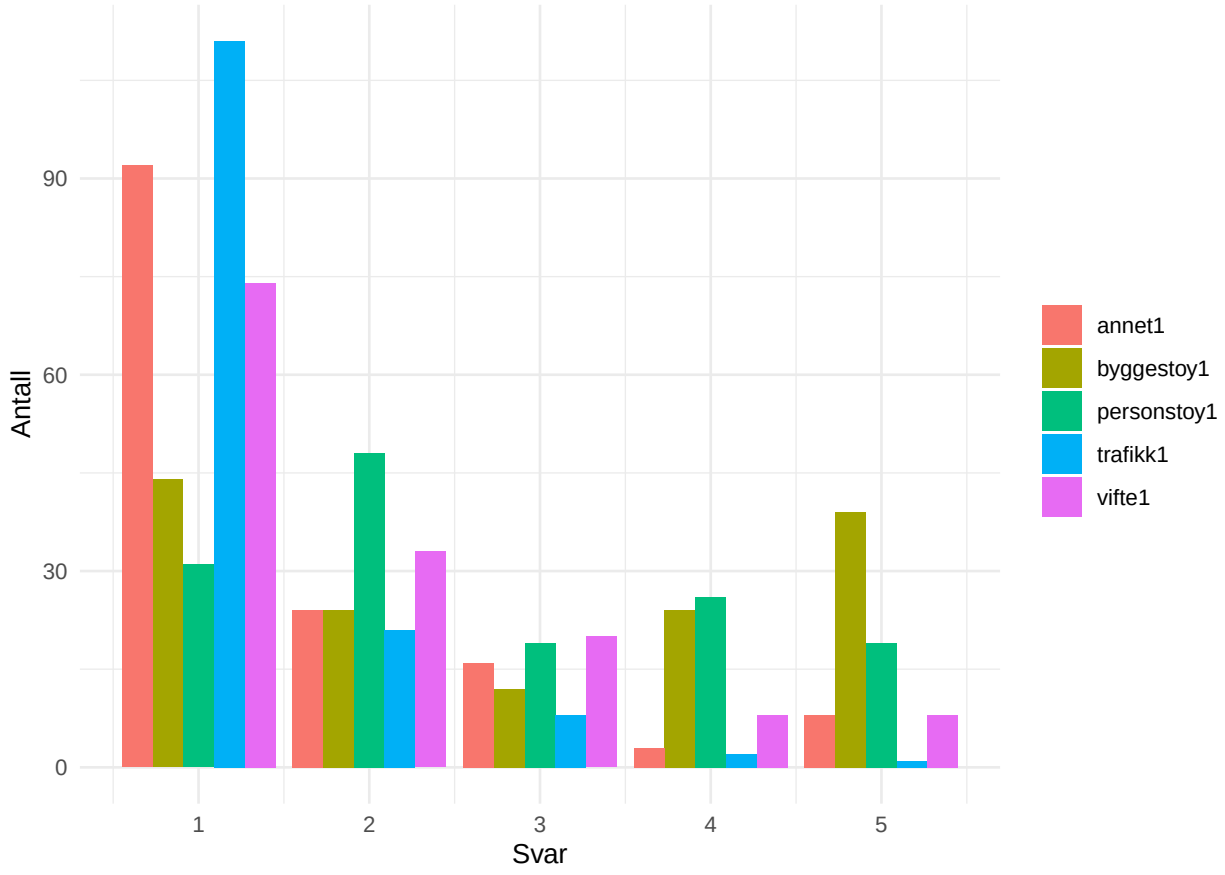
```
# Compute mean and confidence intervals
ntDfCi = data.frame(
  cbind(
    nts=1:length(nts),
    t(apply(d[,ntEff], FUN = CI, MARGIN=2)),
    t(apply(d[,ntTriv], FUN = CI, MARGIN=2))
    )
)

# colnames(ntDfCi)
meansTemp = c('nts','mean','mean.1')
upperTemp = c('nts','upper','upper.1')
lowerTemp = c('nts','lower','lower.1')
dfMTemp = ntDfCi[,meansTemp]
dfUTemp = ntDfCi[,upperTemp]
dfLTemp = ntDfCi[,lowerTemp]

ntDfCiLong <- dfMTemp|>
  pivot_longer(cols=-nts, names_to='Type')
ntDfULong = dfUTemp |>
```

```
  pivot_longer(cols=-nts, names_to='Type')
ntDfLLong = dfLTemp |>
  pivot_longer(cols=-nts, names_to='Type')

ntDfCiLong$upper=ntDfULong$value
ntDfCiLong$lower=ntDfLLong$value

ntDfCiLong$nts=nts[ntDfCiLong$nts]
# rep(c('eff','triv'), as.integer(length(nts)))
ntDfCiLong$Type=rep(c('eff','triv'), as.integer(length(nts)))
colnames(ntDfCiLong) = c('nts', 'Type', 'mean', 'upper', 'lower')

ggplot(ntDfCiLong, aes(x=nts, y = mean ,fill= Type)) +
  geom_col(position="dodge") +
  # scale_y_continuous(sec.axis = sec_axis(~ . /ntDf_maxMean*ntDf_maxSD , name = "Standardavvik"))+
  scale_fill_manual(
    values = c('darkblue', 'lightblue'),
    labels = c('Effektivitet', 'Trivsel')
  )+
  geom_errorbar(
    aes(
      # x=nts,
      ymin = lower,
      ymax = upper),
      color = "black",
    position=position_dodge(.9),
    width=.4
  ) +
  # facet_grid(cols = vars(Type))+
  labs(y="P\u00E5virkning", x="St\u00F8ytype") +
  scale_x_discrete(limit = nts)+
  theme_minimal() +
  theme(legend.title = element_blank())
```



NT Count (Discarded)

```
valueCount = function(x) {
    return(c(sum(x == 1), sum(x == 2), sum(x == 3), sum(x == 4), sum(x == 5)))
}

ntCount = data.frame(cbind(score = 1:length(nts), apply(d[, ntTriv], 2, FUN = valueCount)))

ntCountLong <- ntCount |>
    pivot_longer(cols = -score, names_to = "Type")

ggplot(ntCountLong, aes(x = score, y = value, fill = Type)) + geom_col(position = "dodge") +
    # scale_y_continuous(sec.axis = sec_axis(~ . /ntDf_maxMean*ntDf_maxSD ,
    # name = 'Standardavvik'))+ scale_fill_manual( values = c('darkblue',
    # 'lightblue'), labels = c('Effektivitet', 'Trivsel') )+ facet_grid(cols =
    # vars(Type))+
labs(y = "Antall", x = "Svar") + # scale_x_discrete(limit = nts)+ labs(y =
labs(y = "Antall", x = "Svar") + # scale_x_discrete(limit = nts)+ "Antall", x =
labs(y = "Antall", x = "Svar") + # scale_x_discrete(limit = nts)+ "Svar") + #
labs(y = "Antall", x = "Svar") + # scale_x_discrete(limit = nts)+ scale_x_discrete(limit
labs(y = "Antall", x = "Svar") + # scale_x_discrete(limit = nts)+ = nts)+
theme_minimal() + theme(legend.title = element_blank())
```




```
nTemp = length(d[, "personstoy1"])
CI(d[, "personstoy1"])
```

```
##    upper     mean    lower
## 2.901665 2.678322 2.454978
```

```
mTemp = mean(d[, "personstoy1"])
sTemp = sd(d[, "personstoy1"])
mTemp + qt(0.975, nTemp) * (sTemp/sqrt(nTemp))
```

```
## [1] 2.901651
```

Kjel and Varme

Combine `kjel` and `varme` because of struggle with polygons in "building colored" map.

```
nKjel = length(d[d$spmHvor == "kjel", 1])
nVarme = length(d[d$spmHvor == "varme", 1])
nKV = nKjel + nVarme
vkTriv = c(d[d$spmHvor == "kjel", "spmTriv"], d[d$spmHvor == "varme", "spmTriv"])
```

```
vkEff = c(d[d$spmHvor == "kjel", "spmEff"], d[d$spmHvor == "varme", "spmEff"])
mTriv = mean(vkTriv)
mEff = mean(vkEff)
sdTriv = sd(vkTriv)
sdEff = sd(vkEff)
kjVarme = cbind(mTriv, mEff, sdTriv, sdEff)
colnames(kjVarme) = c("meanTriv", "meanEff", "sdTriv", "sdEff")
round(kjVarme, 2)
```

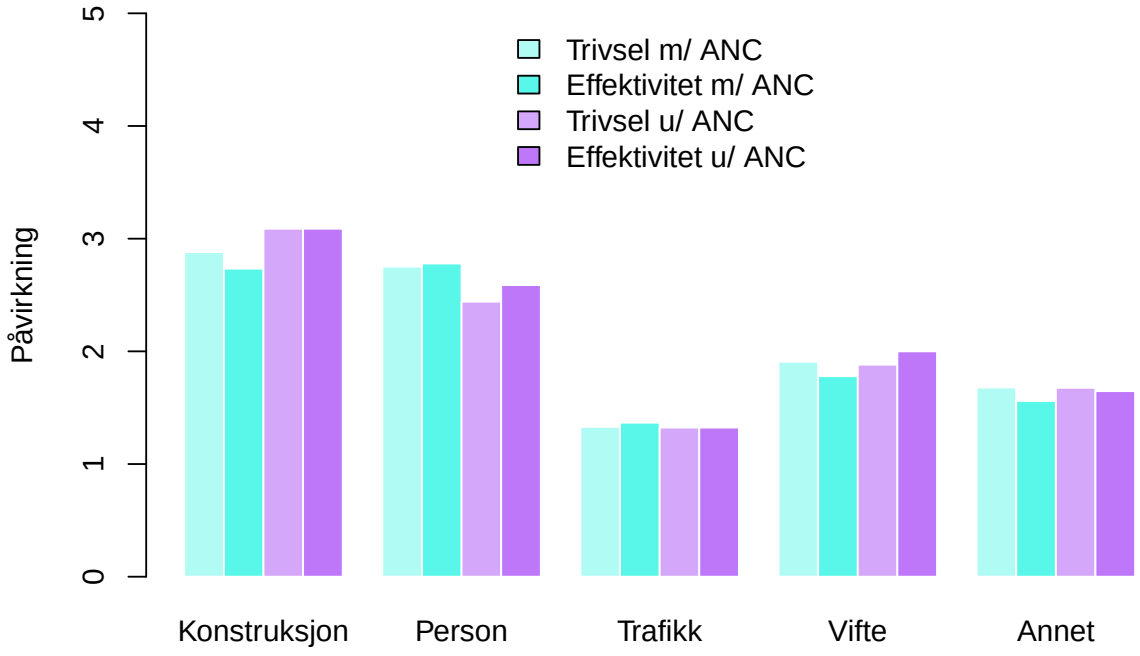
```
##      meanTriv meanEff sdTriv sdEff
## [1,]      3.6     3.2   1.65  1.48
```

ANC and noise type correlation

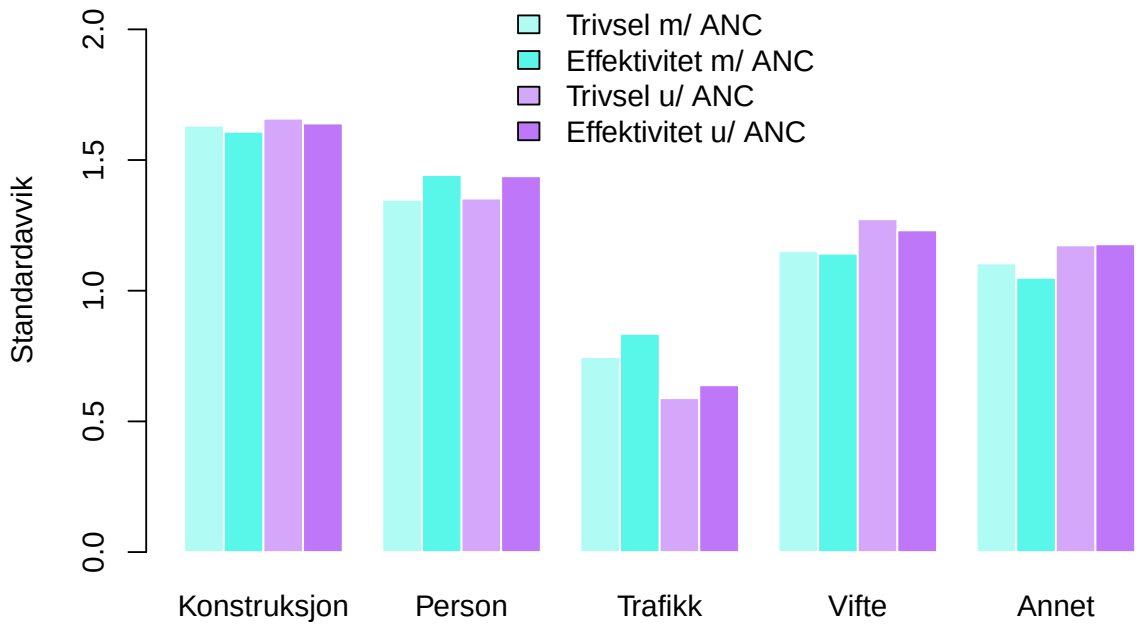
We will look at noise types and how much they disturb when the individual uses ANC compared to when the individual is not.

```
# ntTriv = colnames(d)[7:11]
# ntEff = colnames(d)[13:17]
ntColnames = c(
  'Konstruksjon', 'Person', 'Trafikk', 'Vifte', 'Annet')
ntRownames = c(
  'Trivsel m/ ANC', 'Trivsel u/ ANC', 'Effektivitet m/ ANC',
  'Effektivitet u/ ANC')
ntAnc = data.frame(
  matrix(
    ncol = length(ntTriv), nrow = 4,
    dimnames=list(ntRownames, ntColnames)
  )
)
ntAncSD = data.frame(
  matrix(
    ncol = length(ntTriv), nrow = 4,
    dimnames=list(ntRownames, ntColnames)
  )
)

# Find mean noise type disturbances
ancBool = d[,"spmTiltak_1"] == "mNc"
ntAnc[ntColnames] = rbind(
  apply(d[ancBool, ntTriv], FUN = mean, MARGIN = 2),
  apply(d[!ancBool, ntTriv], FUN = mean, MARGIN = 2),
  apply(d[ancBool,ntEff], FUN = mean, MARGIN = 2),
  apply(d[!ancBool,ntEff], FUN = mean, MARGIN = 2)
)
ntAncSD[ntColnames] = rbind(
  apply(d[ancBool, ntTriv], FUN = sd, MARGIN = 2),
  apply(d[!ancBool, ntTriv], FUN = sd, MARGIN = 2),
  apply(d[ancBool,ntEff], FUN = sd, MARGIN = 2),
  apply(d[!ancBool,ntEff], FUN = sd, MARGIN = 2)
)
```

```r
# Barplot
reOrder = c(1,3,2,4)
cols = c("#b1fbf5", '#d5a7fb', '#59f7ea', '#be77f9')

# Mean
barplot(
  ((as.matrix(ntAnc[reOrder,]))),
  col = cols[reOrder],
  border = "white",
  # main="Trivsel",
  ylab="P\u00E5virkning",
  beside = T,
  # las=2,
  ylim = c(0,5)
  # space=0.1
)
legend(
  "top",
  legend = ntRownames[reOrder],
  fill = cols[reOrder], bty = 'n')
```



```
# SD
ntAncSDrange = max(ntAncSD)
barplot(
  ((as.matrix(ntAncSD[reOrder,]))),
  col = cols[reOrder],
  border = "white",
  # main="Trivsel",
  ylab="Standardavvik",
  beside = T,
  # las=2,
  ylim = c(0,ntAncSDrange*(1+0.3))
  # space=0.1
)
legend(
  "top",
  legend = ntRownames[reOrder],
  fill = cols[reOrder], bty = 'n')
```



LME models

Here we look at the nested linear mixed models explained in the theory of the report.

```
# Setup
trivNames = names(d)[6:11]
effNames = names(d)[12:17]
tiltakNames = names(d)[18:24]
dt = d[, c(trivNames, tiltakNames, "spmHvor")]
de = d[, c(effNames, tiltakNames, "spmHvor")]

tiltakBool = function(x) {
    # Function for random effects
    return(x != "")
}
```

Model selection

First we fit the models.

```
# Trivsel ################################### Complex model
totMod_Triv = lmer(spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 + vifte1 +
    annet1 + as.factor(spmHvor) + (1 | spmTiltak_1) + (1 | spmTiltak_2) + (1 | spmTiltak_3) +
    (1 | spmTiltak_4) + (1 | spmTiltak_5) + (1 | spmTiltak_6) + (1 | spmTiltak_7),
    data = dt)

# without tiltak
ntWhereMod_Triv = lm(spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 + vifte1 +
    annet1 + as.factor(spmHvor), data = dt)


# Only noise types
ntMod_Triv = lm(spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 + vifte1 + annet1,
    data = dt)

# Additional model with only location as explanatory variable
whereMod_Triv = lm(spmTriv ~ as.factor(spmHvor), data = dt)


# Effektivitet ############################## Complex model
totMod_Eff = lmer(spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 + vifte2 + annet2 +
    as.factor(spmHvor) + (1 | spmTiltak_1) + (1 | spmTiltak_2) + (1 | spmTiltak_3) +
    (1 | spmTiltak_4) + (1 | spmTiltak_5) + (1 | spmTiltak_6) + (1 | spmTiltak_7),
    data = de)

# Without tiltak
ntWhereMod_Eff = lm(spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 + vifte2 +
    annet2 + as.factor(spmHvor), data = de)

# Only noise type
ntMod_Eff = lm(spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 + vifte2 + annet2,
    data = de)

# Additional model with only location as explanatory variable
whereMod_Eff = lm(spmEff ~ as.factor(spmHvor), data = de)
```

Then we make model selection using `anova`.

```
# Trivsel
anova(totMod_Triv, ntWhereMod_Triv, ntMod_Triv, test = "LRT")  # In report
```

```
## Data: dt
## Models:
## ntMod_Triv: spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 + vifte1 + annet1
## ntWhereMod_Triv: spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 + vifte1 + annet1 + as.factor(spr
## totMod_Triv: spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 + vifte1 + annet1 + as.factor(spmHvo
##                 npar    AIC    BIC  logLik deviance   Chisq Df Pr(>Chisq)
## ntMod_Triv         6 424.10 441.87 -206.05   412.10
## ntWhereMod_Triv   29 435.06 520.98 -188.53   377.06 35.0354 23     0.0516 .
## totMod_Triv       36 448.19 554.85 -188.09   376.19  0.8748  7     0.9966
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Effektivitet
anova(totMod_Eff, ntWhereMod_Eff, ntMod_Eff, test = "LRT")
```

```
## Data: de
## Models:
## ntMod_Eff: spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 + vifte2 + annet2
## ntWhereMod_Eff: spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 + vifte2 + annet2 + as.factor(spmH
## totMod_Eff: spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 + vifte2 + annet2 + as.factor(spmHvor)
##                 npar    AIC    BIC  logLik deviance   Chisq Df Pr(>Chisq)
## ntMod_Eff          6 377.11 394.89 -182.56   365.11
## ntWhereMod_Eff    29 381.55 467.47 -161.78   323.55 41.5615 23    0.01021 *
## totMod_Eff        36 391.39 498.06 -159.70   319.39  4.1569  7    0.76154
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model summaries trivsel

We skip showing the summaries of nested models without significant χ^2 on level $\alpha = 0.1$. That is, we skip the total model for `Triv`.

```
# Trivsel
summary(ntMod_Triv)
```

```
##
## Call:
## lm(formula = spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 +
##     vifte1 + annet1, data = dt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3250 -0.5627 -0.0132  0.5948  3.8080
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## byggestoy1   0.45379    0.04919   9.226 4.35e-16 ***
## personstoy1  0.43678    0.05741   7.608 3.92e-12 ***
```

```
## trafikk1    -0.10048    0.13560  -0.741  0.45994
## vifte1       0.25892    0.08053   3.215  0.00162 **
## annet1       0.14296    0.09381   1.524  0.12982
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.041 on 138 degrees of freedom
## Multiple R-squared:  0.9115, Adjusted R-squared:  0.9083
## F-statistic: 284.4 on 5 and 138 DF,  p-value: < 2.2e-16
```

```
summary(ntWhereMod_Triv)
```

```
##
## Call:
## lm(formula = spmTriv ~ -1 + byggestoy1 + personstoy1 + trafikk1 +
##     vifte1 + annet1 + as.factor(spmHvor), data = dt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2385 -0.6265  0.0000  0.4435  2.7856
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## byggestoy1                0.28727    0.07526   3.817 0.000219 ***
## personstoy1               0.33860    0.07292   4.644 9.16e-06 ***
## trafikk1                 -0.13685    0.16492  -0.830 0.408366
## vifte1                    0.18832    0.08515   2.212 0.028966 *
## annet1                    0.10068    0.10268   0.980 0.328915
## as.factor(spmHvor)berg   -0.65198    0.64670  -1.008 0.315488
## as.factor(spmHvor)bygg    0.94534    0.55721   1.697 0.092483 .
## as.factor(spmHvor)elA     2.10607    0.82123   2.565 0.011620 *
## as.factor(spmHvor)elB     1.30920    0.53656   2.440 0.016214 *
## as.factor(spmHvor)elD     1.19102    0.61930   1.923 0.056932 .
## as.factor(spmHvor)elE     1.62929    0.45491   3.582 0.000502 ***
## as.factor(spmHvor)gamEl   1.32542    0.51590   2.569 0.011475 *
## as.factor(spmHvor)gamFys  1.06497    0.81685   1.304 0.194920
## as.factor(spmHvor)gruve   1.43635    0.51546   2.787 0.006234 **
## as.factor(spmHvor)hand    1.07090    0.60414   1.773 0.078939 .
## as.factor(spmHvor)hoved   0.80264    0.53434   1.502 0.135809
## as.factor(spmHvor)ipd     1.96875    1.08016   1.823 0.070955 .
## as.factor(spmHvor)itSyd   1.15693    1.16738   0.991 0.323742
## as.factor(spmHvor)kjel    1.62808    0.49980   3.257 0.001478 **
## as.factor(spmHvor)kjemi1  1.09320    0.66685   1.639 0.103875
## as.factor(spmHvor)kjemi2  2.30259    1.09883   2.095 0.038322 *
## as.factor(spmHvor)kjemi5 -0.16678    1.06347  -0.157 0.875660
## as.factor(spmHvor)real    0.76277    0.31375   2.431 0.016595 *
## as.factor(spmHvor)sent1   0.79312    0.39941   1.986 0.049443 *
## as.factor(spmHvor)sent2   0.86727    0.58867   1.473 0.143407
## as.factor(spmHvor)tapir   0.89534    0.44569   2.009 0.046892 *
## as.factor(spmHvor)varme   1.15737    0.61456   1.883 0.062194 .
## as.factor(spmHvor)verk    0.84672    0.45745   1.851 0.066742 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.008 on 115 degrees of freedom
## Multiple R-squared:  0.9308, Adjusted R-squared:  0.9139
## F-statistic: 55.21 on 28 and 115 DF,  p-value: < 2.2e-16
```

```
# Effektivitet
summary(ntMod_Eff)
```

```
##
## Call:
## lm(formula = spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 +
##     vifte2 + annet2, data = de)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9177 -0.3255 -0.1274  0.4762  2.8186
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## byggestoy2   0.51463    0.04349  11.834   <2e-16 ***
## personstoy2  0.48649    0.04747  10.248   <2e-16 ***
## trafikk2    -0.02712    0.10662  -0.254    0.800
## vifte2       0.10859    0.07436   1.460    0.146
## annet2       0.06956    0.08248   0.843    0.401
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8829 on 138 degrees of freedom
## Multiple R-squared:  0.9342, Adjusted R-squared:  0.9319
## F-statistic: 392.1 on 5 and 138 DF,  p-value: < 2.2e-16
```

```
summary(ntWhereMod_Eff)
```

```
##
## Call:
## lm(formula = spmEff ~ -1 + byggestoy2 + personstoy2 + trafikk2 +
##     vifte2 + annet2 + as.factor(spmHvor), data = de)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.86256 -0.39101 -0.06636  0.49956  2.28961
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## byggestoy2                 0.447586   0.062537   7.157 8.29e-11 ***
## personstoy2                0.441493   0.057398   7.692 5.41e-12 ***
## trafikk2                  -0.268888   0.131484  -2.045  0.04314 *
## vifte2                     0.004305   0.079509   0.054  0.95692
## annet2                     0.174437   0.088262   1.976  0.05051 .
## as.factor(spmHvor)berg    -0.531972   0.514501  -1.034  0.30333
## as.factor(spmHvor)bygg     1.059205   0.452493   2.341  0.02096 *
## as.factor(spmHvor)elA      1.551117   0.655842   2.365  0.01970 *
## as.factor(spmHvor)elB      1.218916   0.415135   2.936  0.00401 **
```

```
## as.factor(spmHvor)elD    -0.033282   0.517185  -0.064  0.94880
## as.factor(spmHvor)elE     0.729011   0.337697   2.159  0.03295 *
## as.factor(spmHvor)gamEl   0.980450   0.396900   2.470  0.01497 *
## as.factor(spmHvor)gamFys  0.751529   0.657741   1.143  0.25558
## as.factor(spmHvor)gruve   0.468361   0.421042   1.112  0.26829
## as.factor(spmHvor)hand    1.426627   0.503061   2.836  0.00540 **
## as.factor(spmHvor)hoved   0.570720   0.416210   1.371  0.17297
## as.factor(spmHvor)ipd     0.338572   0.892560   0.379  0.70515
## as.factor(spmHvor)itSyd   3.381445   1.010518   3.346  0.00111 **
## as.factor(spmHvor)kjel    0.586410   0.400083   1.466  0.14545
## as.factor(spmHvor)kjemi1  1.175501   0.536147   2.192  0.03036 *
## as.factor(spmHvor)kjemi2  1.422178   0.908270   1.566  0.12014
## as.factor(spmHvor)kjemi5 -0.139904   0.862044  -0.162  0.87136
## as.factor(spmHvor)real    0.583627   0.248891   2.345  0.02075 *
## as.factor(spmHvor)sent1   0.825933   0.315808   2.615  0.01011 *
## as.factor(spmHvor)sent2   0.671235   0.460954   1.456  0.14807
## as.factor(spmHvor)tapir   0.276472   0.350057   0.790  0.43128
## as.factor(spmHvor)varme   0.739149   0.490428   1.507  0.13451
## as.factor(spmHvor)verk    1.016288   0.352592   2.882  0.00471 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8364 on 115 degrees of freedom
## Multiple R-squared:  0.9508, Adjusted R-squared:  0.9389
## F-statistic: 79.42 on 28 and 115 DF,  p-value: < 2.2e-16
```

Misc requests

These are some statistics requested by team members.

Anc data (old)

```
N = length(d[, 1])
ancPercentage = length(d$spmTiltak_1[d$spmTiltak_1 == "mNc"])/N
ancPercentage
```

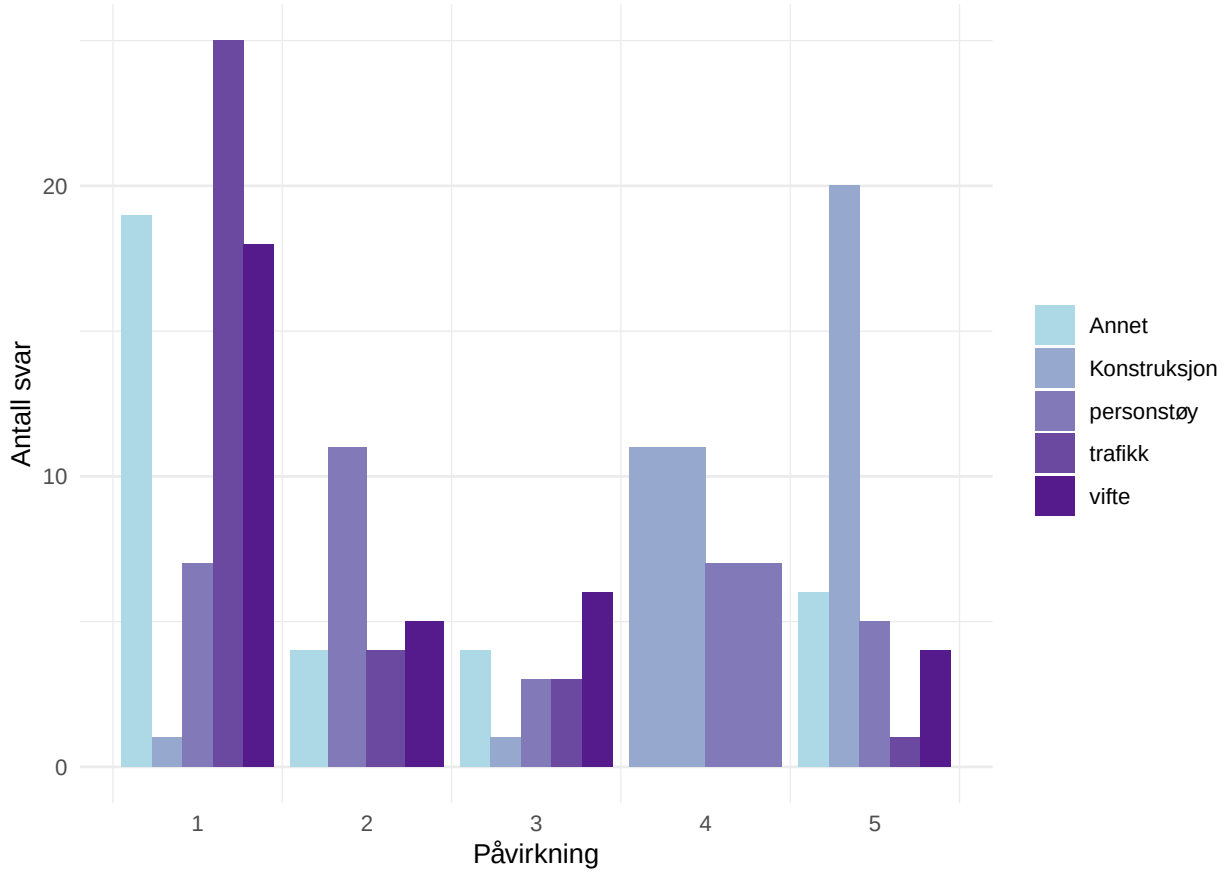
```
## [1] 0.7686567
```

Noise Type in Elektro (Sindre)

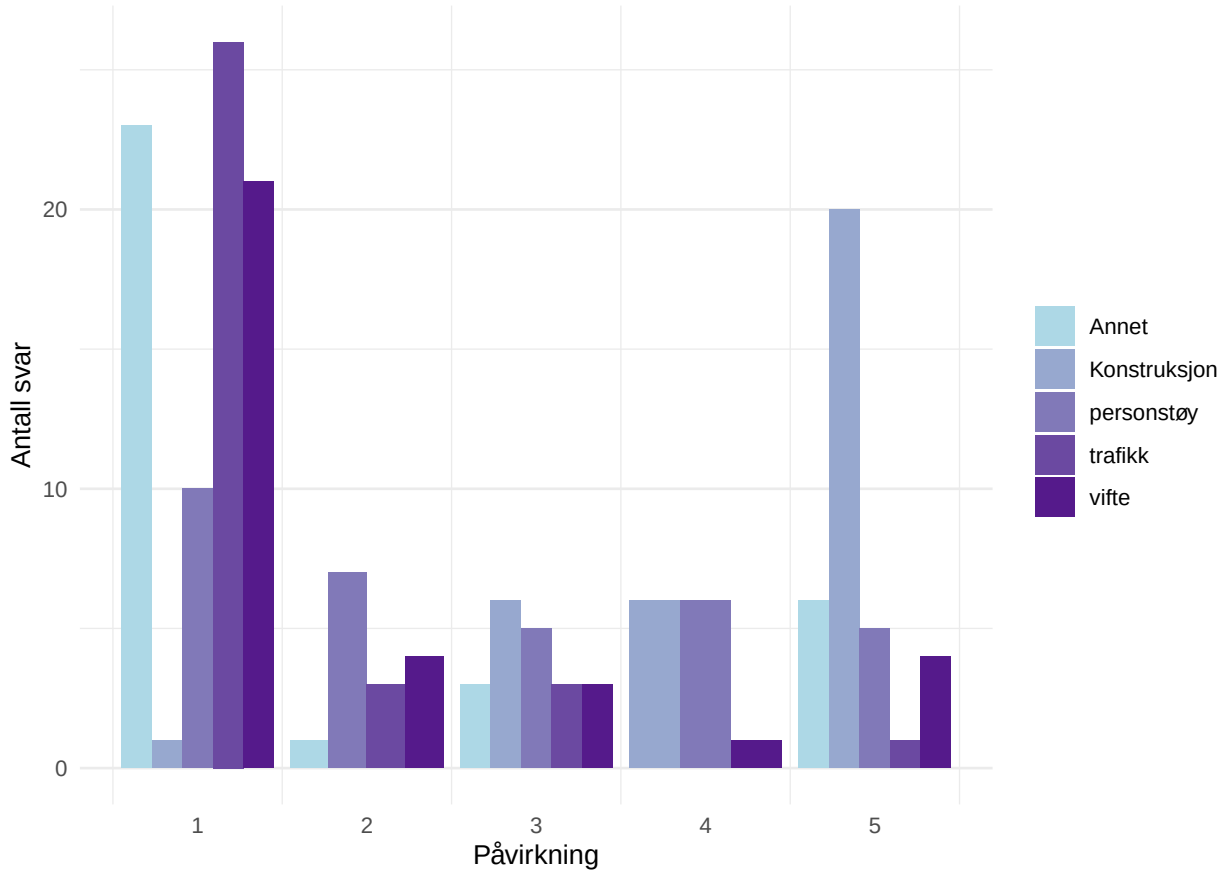
Closely consider the distribution of noise type answers on the scale 1-5 and each noise type mean and CI for location `elektro`.

```
elektro = c("elD", "elB", "elE", "elA")
elTriv = d[d$spmHvor %in% elektro, c("spmHvor", ntTriv)]
elEff = d[d$spmHvor %in% elektro, c("spmHvor", ntEff)]

elLong <- elTriv |>
    pivot_longer(cols = -spmHvor, names_to = "nt", values_to = "triv")
```

```
elEffLong = elEff |>
    pivot_longer(cols = -spmHvor, names_to = "nt", values_to = "eff")
elLong$eff = elEffLong$eff

# Plot
plotColors = colorRampPalette(c("lightblue", "purple4"))(5)
ggElektroTriv = ggplot(data = elLong, aes(x = triv, fill = nt)) + geom_bar(position = "dodge") +
    scale_fill_manual(values = plotColors, labels = c("Annet", "Konstruksjon", "personstøy",
        "trafikk", "vifte")) + labs(x = "Påvirkning", y = "Antall svar") + scale_y_continuous(breaks =
    20, 10)) + theme_minimal() + theme(panel.grid.major.x = element_blank(), legend.title = element_bla
ggElektroTriv
```



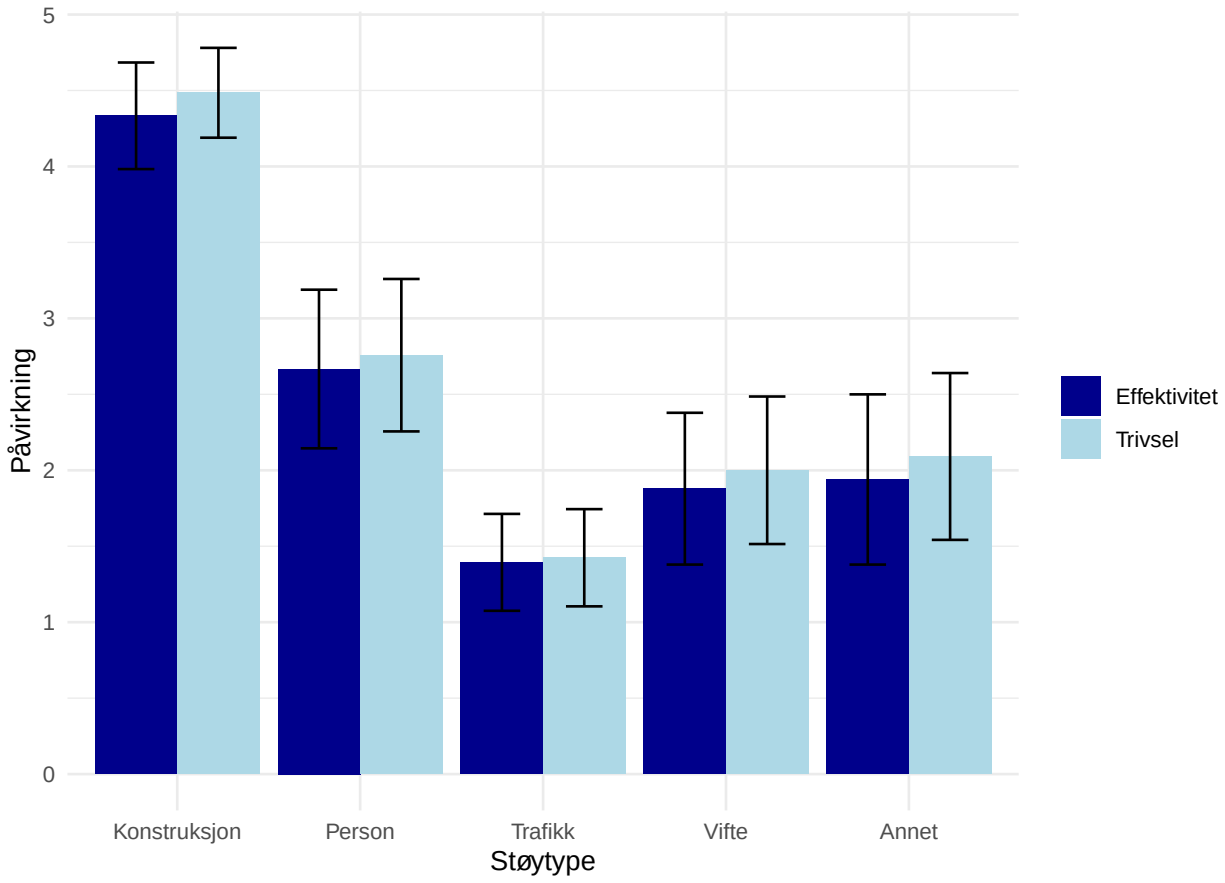

```
ggElektroEff = ggplot(data = elLong, aes(x = eff, fill = nt)) + geom_bar(position = "dodge") +
    scale_fill_manual(values = plotColors, labels = c("Annet", "Konstruksjon", "personstøy",
        "trafikk", "vifte")) + labs(x = "Påvirkning", y = "Antall svar") + theme_minimal() +
    theme(panel.grid.major.x = element_blank(), legend.title = element_blank())
ggElektroEff
```




```
# ggplot(data=elLong, aes(x=nt))+ geom_bar(position='dodge', aes(y = triv),
# stat='summary', fun='mean') + geom_bar(aes(y = eff), stat='summary',
# fun='mean')
```

```
elData = data.frame(
  cbind(
    nts=1:length(nts),
    t(apply(d[d$spmHvor %in% elektro, ntEff], FUN = CI, MARGIN=2)),
    t(apply(d[d$spmHvor %in% elektro, ntTriv], FUN = CI, MARGIN=2))
    )
)

meansTemp = c('nts','mean','mean.1')
upperTemp = c('nts','upper','upper.1')
lowerTemp = c('nts','lower','lower.1')
dfMTemp = elData[,meansTemp]
dfUTemp = elData[,upperTemp]
dfLTemp = elData[,lowerTemp]

ntDfCiLong <- dfMTemp|>
  pivot_longer(cols=-nts, names_to='Type')
ntDfULong = dfUTemp |>
  pivot_longer(cols=-nts, names_to='Type')
ntDfLLong = dfLTemp |>
```

```
  pivot_longer(cols=-nts, names_to='Type')

ntDfCiLong$upper=ntDfULong$value
ntDfCiLong$lower=ntDfLLong$value

ntDfCiLong$nts=nts[ntDfCiLong$nts]
ntDfCiLong$Type=rep(c('eff','triv'), as.integer(length(nts)))
colnames(ntDfCiLong) = c('nts', 'Type', 'mean', 'upper', 'lower')

ggElektro = ggplot(ntDfCiLong, aes(x=nts, y = mean ,fill= Type)) +
  geom_col(position="dodge") +
  scale_fill_manual(
    values = c('darkblue', 'lightblue'),
    labels = c('Effektivitet', 'Trivsel')
  )+
  geom_errorbar(
    aes(
      # x=nts,
      ymin = lower,
      ymax = upper),
      color = "black",
    position=position_dodge(.9),
    width=.4
  ) +
  labs(y="P\u00E5virkning", x="St\u00F8ytype") +
  scale_x_discrete(limit = nts)+
  theme_minimal() +
  theme(legend.title = element_blank())
ggElektro
```




```
pdf(file = "./figures/noiseTypeBarConfIntElektro.pdf", width = 6, height = 4)
ggElektro
dev.off()
```

```
## pdf
##   2
```