R - Datavisualisatie

J.J. van Nijnatten

Contents

contact: J.J.vanNijnatten@uva.nl

broncode: https://github.com/jonasvannijnatten/R_Data_Visualization

1 Software

1.1 Versies

software versies gebruikt voor deze handleiding:

- R version 3.5.1 (2018-07-02)
- ggplot2 versie: 3.0.0, 2018-07-02
- Hmisc versie: 4.1.1, 2018-01-03

1.2 Installatie

Benodigde packages downloaden & installeren:

```r
install.packages(pkgs="ggplot2", repos = "https://www.freestatistics.org/cran/")
install.packages(pkgs="Hmisc",   repos = "https://www.freestatistics.org/cran/")
```

Benodigde packages activeren:

```r
library(package=Hmisc)
library(package=ggplot2)
```

---

2 Het doel van datavisualisatie

2.1 Wat wil je weergeven?

Data beschrijven, samenvatten of statistiek weergeven. Beschrijven: patronen in de data weergeven, verdeling, verhoudingen, verschillen samenvatten: gemiddelde & maat van spreiding, regressielijn

2.2 Hoe kies je de juiste manier van weergeven? (exp. design)

De opmaak van je figuur wordt deels bepaalt door je experimental design. De soort en het aantal variabelen dat je onderzoekt in je experiment bepalen mee hoe het figuur er uit komt te zien. Hierbij is het belangrijk of het gaat om continue of categorische variabelen, en of de data is verzameld in onafhankelijke steekproeven (*independent samples*), of dat eenzelfde steekproef meerdere keren is gemeten in verschillende condities (*paired / repeated measures*). Hoewel er geen echte regels zijn voor het maken van figuren bestaat er wel enige consesus over hoe de data wordt weergegeven. de afhankelijke variabele (reponse variabele, outcome variable)

3 Introductie GGplot2 package

3.1 Opbouw van figuren

4 Voorbeelden

4.1 Data-inspectie

4.2 Normaliteit

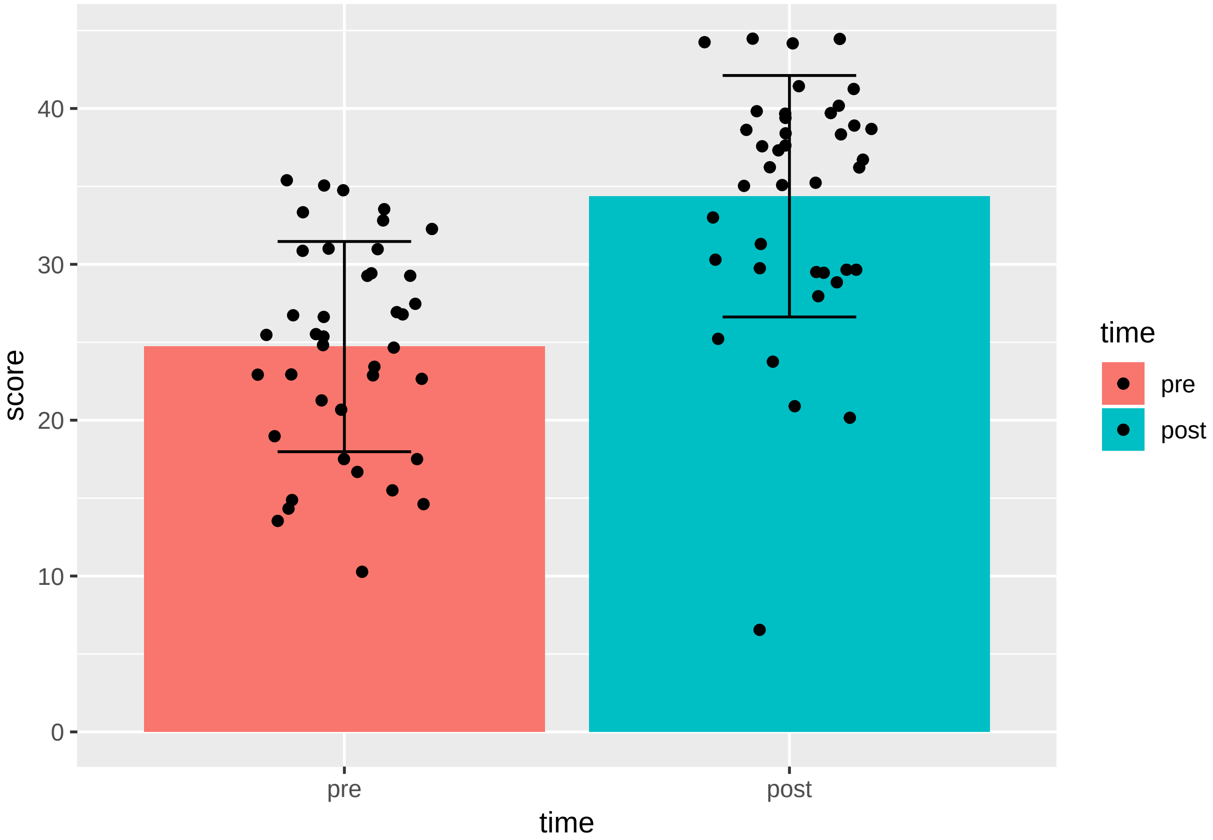
4.3 T-test

```
# generate data
group1 = rnorm(n = 40, mean = 25, sd = 6.5)
group2 = rnorm(n = 40, mean = 35, sd = 6.5)
data.wide = data.frame(group1, group2)
```

```
# reshape data
data.long = reshape(data = data.wide,
                    direction = "long"
                    , varying = c("group1", "group2")
                    , v.names = "score"
                    , times   = c('pre', 'post')
                    )

data.long$time = factor(data.long$time, levels = c("pre","post"))
```

```
# plot means and standard deviations
ggplot(data.long, aes(x=time, y=score, fill=time) ) +
  # plot de gemiddeldes voor iedere conditie als bargraph
  geom_bar      ( stat = "summary", fun.y    = "mean" ) +
  # voeg errobars toe aan de bargraph
  geom_errorbar( stat = "summary", fun.data = "mean_sdl", fun.args = 1, width = 0.3 ) +
  # plot alle individuele datapunten
  geom_point    ( position=position_jitter(width = .2, seed=1) )
```

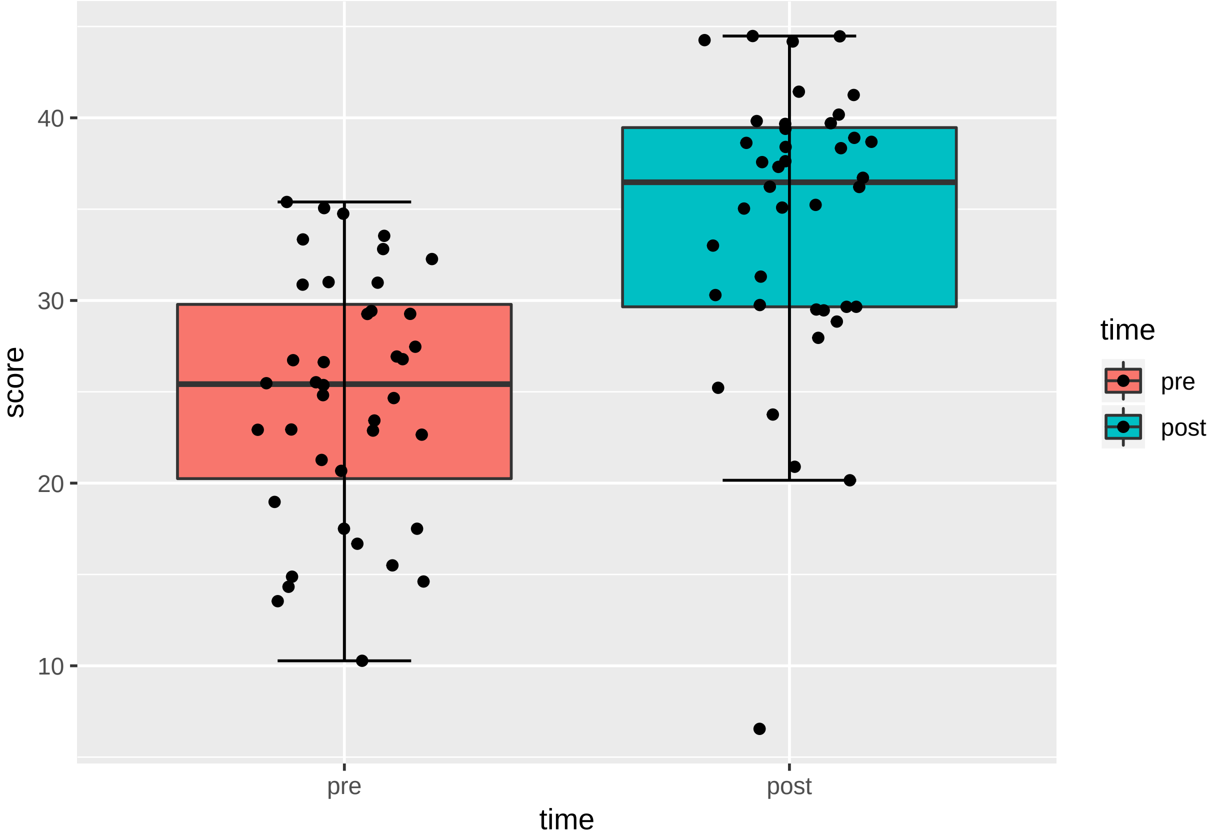


To plot standard errors instead of standard deviations replace "mean_sdl" with "mean_se", and it is common use to plot 2 (or 1.96) times the standard error to get an 95% confidence interval, so replace "fun.arg = 1" with "fun.arg = 2".

```
  geom_errorbar( stat = "summary", fun.data = "mean_se", fun.args = 2, width = 0.3 )
```

Wanneer de data niet normaal verdeeld is dan zijn de gemiddeldes en standaarddeviaties of standdaardfouten geen goede representiatie van de data. In dat geval is het gebruikelijker om de data in een boxplot weer te geven met mediaan en quantielafstanden.

```
ggplot(data.long, aes(x=time, y=score, fill=time) ) +
  # plot een boxplot, outlier.shape = NA onderdrukt het plotten van de outliers
  # (geom_points plot alle datapunten)
  geom_boxplot(outlier.shape = NA) +
  # voegt 'balkjes'errorbars' aan het einde van de whiskers toe
  stat_boxplot(geom="errorbar", width=.3) +
  # plot alle individuele datapunten
  geom_point(position=position_jitter(width=.2,seed = 1))
```



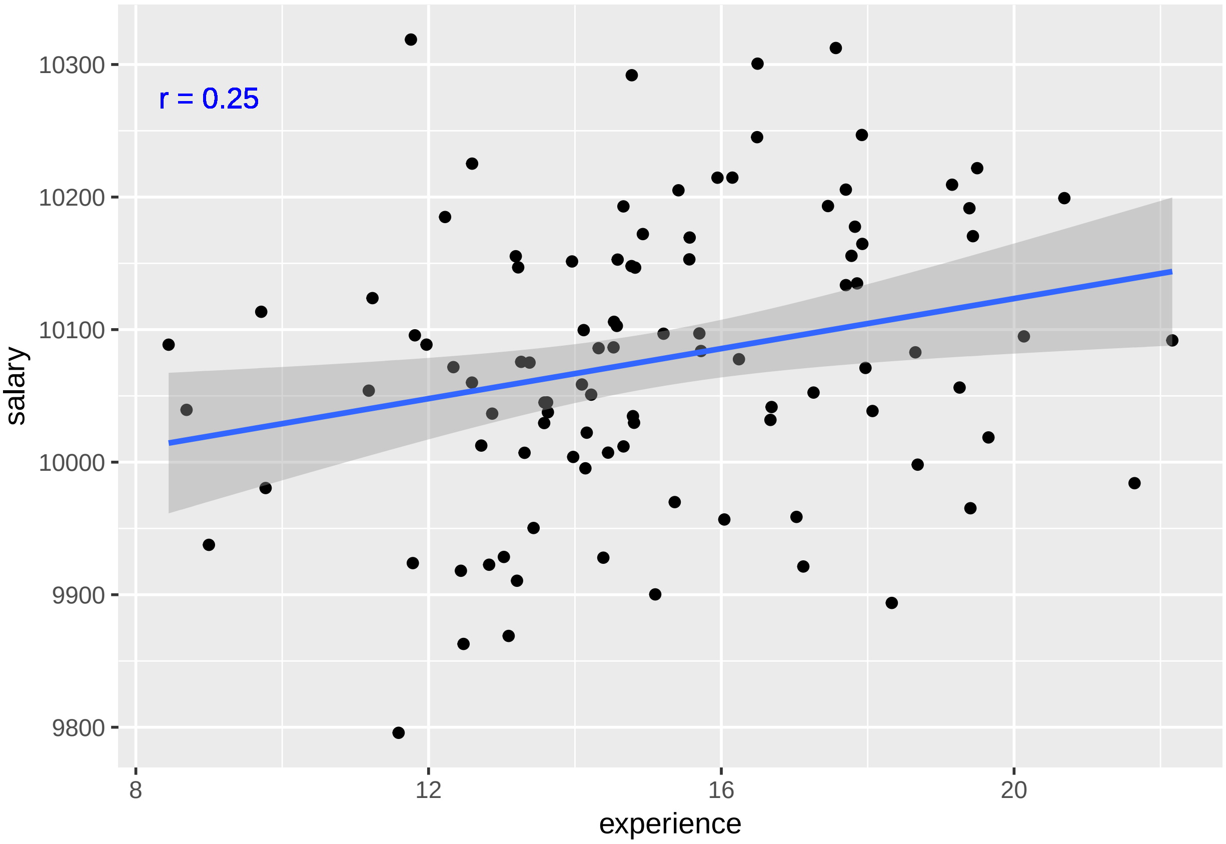
4.4 Correlatie & Regressie

Wanneer je de relatie tussen twee continue variabelen onderzoekt kun je een correlatie of lineaire regressie analyse gebruiken. Een correlatietoets geeft aan of beide variabelen samenhangen, terwijl een regressie een predictiemodel toetst. In de pratkijk geven beide statistisch gezien hetzelde resultaat maar verschillen ze in de hypothese die je toetst en de conclusies die je kunt trekken. Bij een correlatie is het daarom gebruikelijk om de correlatiecoefficient r te rapporteren, terwijl het bij een regressie gebruikelijk is om de regressie coefficienten (intercept & helling; *alpha* en *beta*) te rapporteren in de vorm van het predictiemodel.

```r
# generate data
set.seed(05)
nrobs = 100
experience = rnorm(n = nrobs, mean = 15, sd = 3)
salary     = 10000 + ( 5 * experience ) + rnorm(n = nrobs, mean = 0, sd = 100)
data = data.frame(experience, salary)
# calculate correlation coefficient r
corr_coef = cor(x = data$experience, y = data$salary)
```

Correlatieplot:

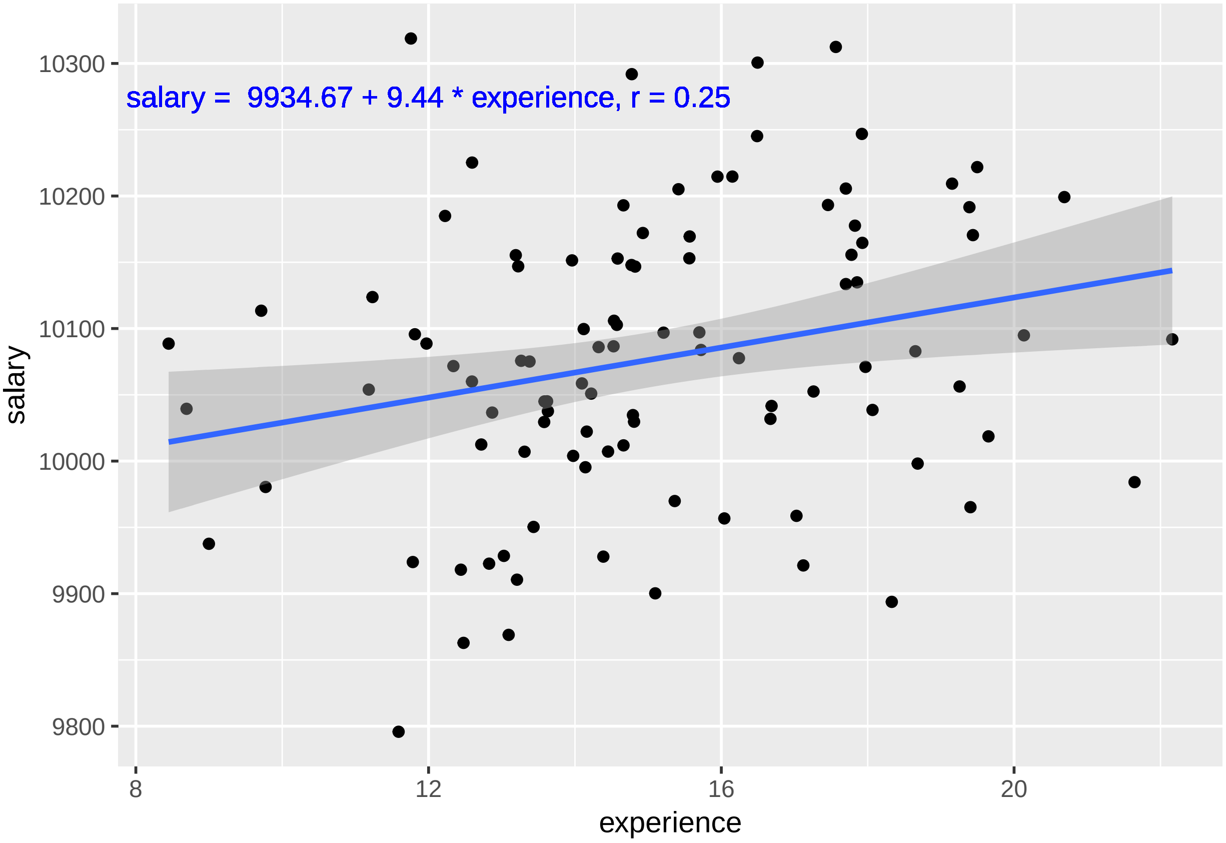
```r
ggplot(data = data, aes(x = experience, y = salary)) +
  geom_point() +  # plot the datapoints
  # add a linear regression line with 95% confidence interval
  geom_smooth(method='lm', se=TRUE, level=0.95) +
  # plot the correlation coefficient into to figure
  geom_text(x = 9, y = 10275, color="blue",
            label = paste("r =", as.character(round(corr_coef,3))))
```



Regressieplot:

```r
# apply linear regression and obtain regression coefficients
lin_mod = summary(lm(data = data, formula = salary~experience))
alpha = round(lin_mod$coefficients[1],2)
beta  = round(lin_mod$coefficients[2],2)
corr_coef = round(sqrt(lin_mod$r.squared),3)
```

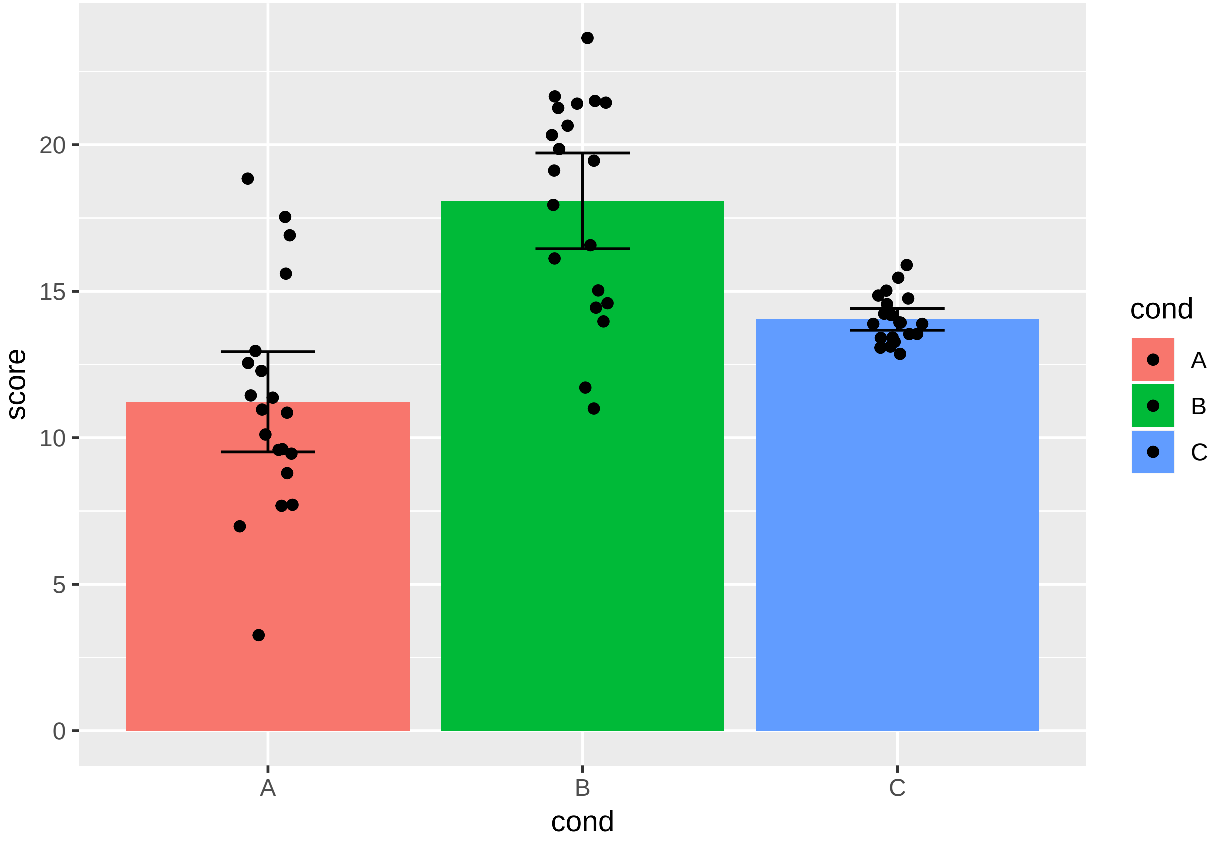
```
# create figure
ggplot(data = data, aes(x = experience, y = salary)) +
  geom_point() +  # plot the datapoints
  # add a linear regression line with 95% confidence interval
  geom_smooth(method='lm', se=TRUE, level=0.95) +
  # plot the correlation coefficient into to figure
  geom_text(x = 12, y = 10275, color="blue",
            label = paste("salary = ", alpha, "+", beta,"*","experience, r =",corr_coef))
```



4.5 One-way independent samples ANOVA

```r
set.seed(05)   # set seed
nrofconds = 3  # set number of conditions
nrofsubs  = 20 # set number of subjects
subj = as.factor(1:(nrofsubs*nrofconds))      # create array with subject IDs
cond = as.factor(rep(LETTERS[1:nrofconds],each=nrofsubs))   # create array with condition values
score = as.vector( replicate(
          nrofconds , rnorm(n = nrofsubs, mean = sample(8,1)+10 , sd = sample(5,1) )
        ) )                                     # create array with measurement values
data.long = data.frame(subj, cond, score);      # combine arrays into a data.frame
rm(list=setdiff(ls(), c("data.long", "nrofsubs","nrofconds"))) # delete arrays
```

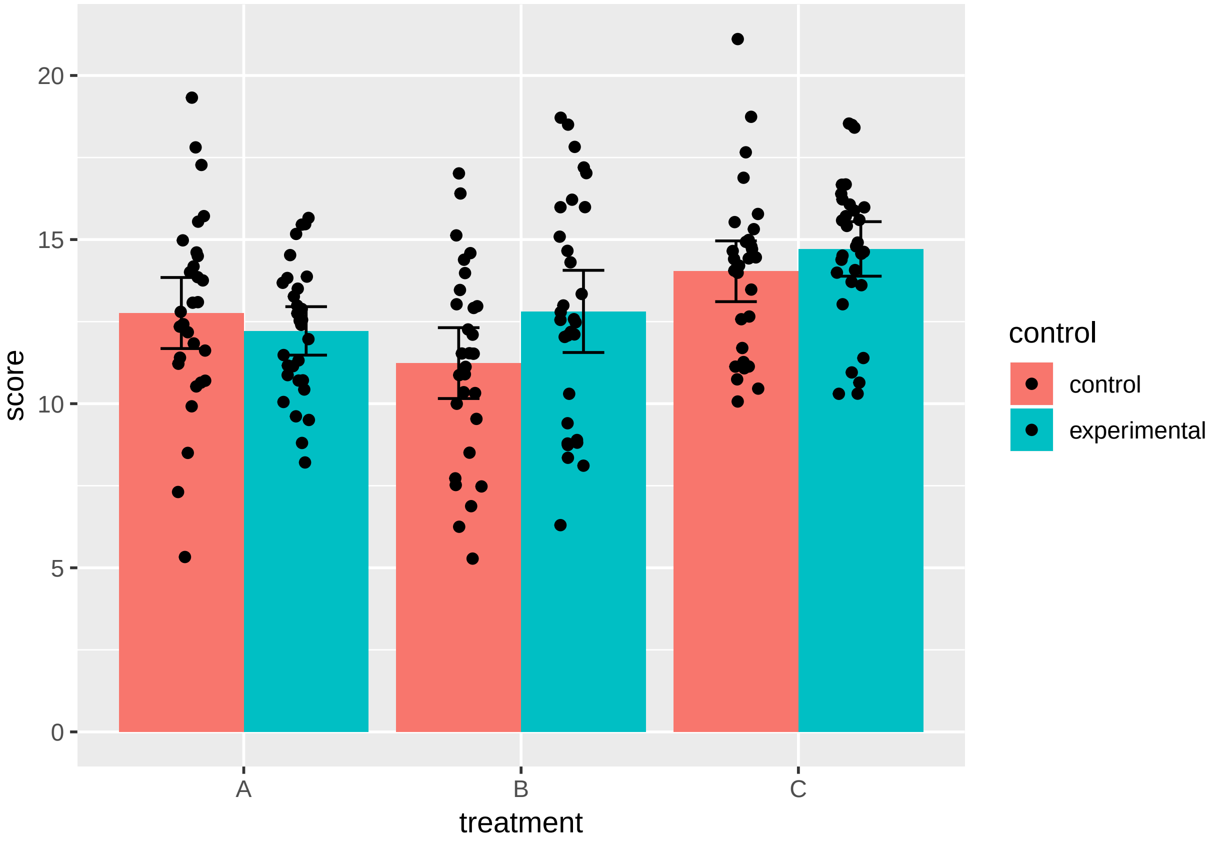
```r
ggplot(data.long, aes(x=cond, y=score, fill=cond) ) +
  geom_bar     ( stat = "summary", fun.y    = "mean" ) +
  geom_errorbar( stat = "summary", fun.data = "mean_se", fun.args = 2, width = 0.3 ) +
  geom_point   ( position=position_jitter(width = .1) )
```



4.6 Factorial independent samples ANOVA

```
set.seed(01)   # set seed
nrofcondsf1 = 3  # set number of conditions for factor 1
nrofcondsf2 = 2  # set number of conditions for factor 2
nrofsubs    = nrofcondsf1*nrofcondsf2*30 # set number of subjects per condition
subj = as.factor(1:(nrofsubs))       # create array with subject IDs
# create array witht treatment conditions
treatment = as.factor(rep(LETTERS[1:nrofcondsf1],each=nrofsubs/nrofcondsf1))
# create array with control / experimental
control   = as.factor(rep(c("control","experimental"),times=nrofsubs/nrofcondsf2))
# create array with measurement values
score = as.vector( replicate(nrofcondsf1, replicate (
          nrofcondsf2 , rnorm(
            n = (nrofsubs/(nrofcondsf1*nrofcondsf2)),
            mean = 0 , sd = sample(5,1) ) + sample(8,1)+10
        ) ) )
# combine arrays into a data.frame
data.long = data.frame(subj, score, treatment, control);
# delete unnecessary arrays
rm(list=setdiff(ls(), c("data.long", "nrofsubs","nrofconds")))
```

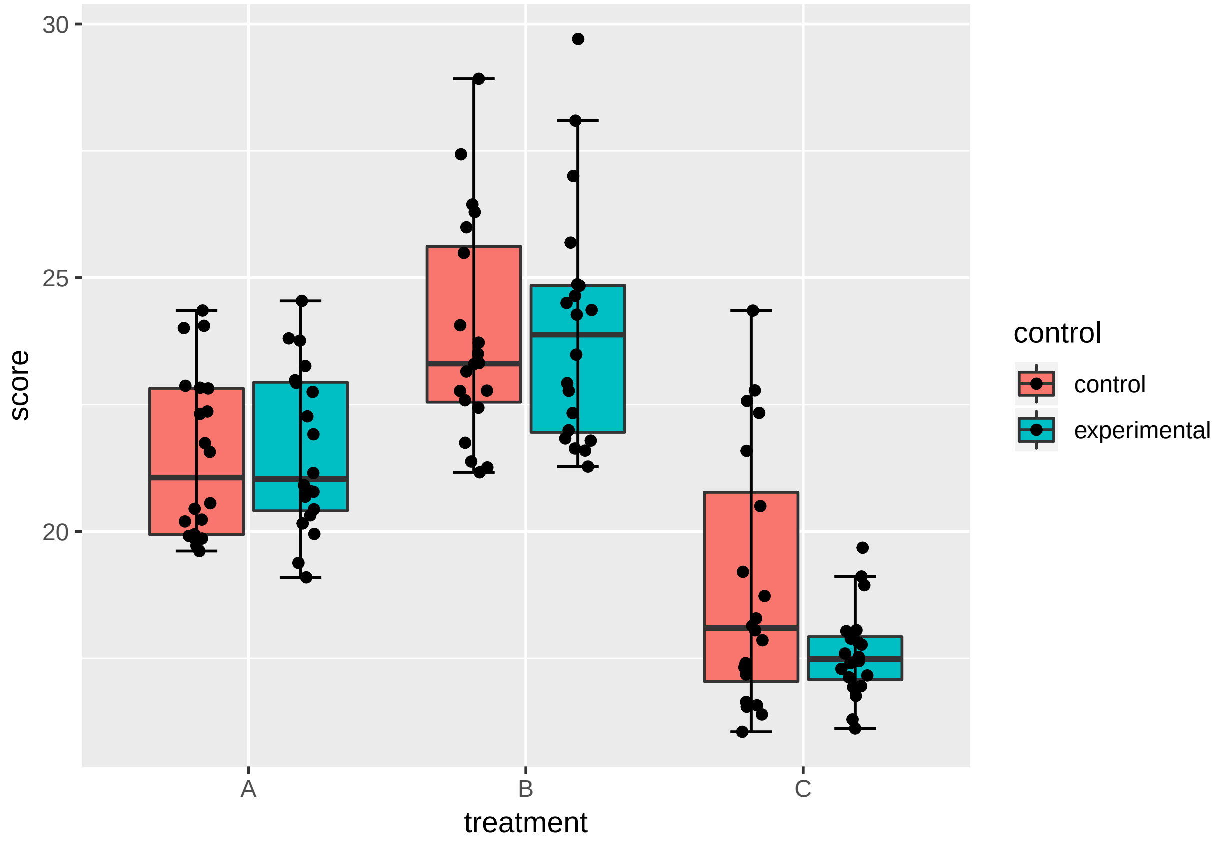
```
ggplot(data.long, aes(x=treatment, y=score, fill=control) ) +
  geom_bar     ( stat = "summary", fun.y     = "mean" , position = "dodge") +
  geom_errorbar( stat = "summary", fun.data = "mean_se", fun.args = 2, width = 0.3,
               position = position_dodge(width=.9) ) +
  geom_point  ( position = position_jitterdodge(jitter.width = .2) )
```



```
set.seed(01)   # set seed
nrofcondsf1 = 3  # set number of conditions for factor 1
nrofcondsf2 = 2  # set number of conditions for factor 2
nrofsubs    = nrofcondsf1*nrofcondsf2*20 # set number of subjects per condition
subj = as.factor(1:(nrofsubs))       # create array with subject IDs
```

```
# create array witht treatment conditions
treatment = as.factor(rep(LETTERS[1:nrofcondsf1],each=nrofsubs/nrofcondsf1))
# create array with control / experimental
control   = as.factor(rep(c("control","experimental"),times=nrofsubs/nrofcondsf2))
# create array with measurement values
score = as.vector( replicate(nrofcondsf1, replicate (
          nrofcondsf2 , rchisq(
            n = (nrofsubs/(nrofcondsf1*nrofcondsf2)),
            df = 3)
        )+  sample(14,1)+10  ) )
# combine arrays into a data.frame
data.long = data.frame(subj, score, treatment, control);
# delete unnecessary arrays
#rm(list=setdiff(ls(), c("data.long", "nrofsubs","nrofconds")))
```

```
ggplot(data.long, aes(x=treatment, y=score, fill=control) ) +
  geom_boxplot (outlier.shape = NA) +
  stat_boxplot(geom="errorbar", width=.3, position = position_dodge(.75) ) +
  geom_point  ( position = position_jitterdodge(jitter.width = .2) )
```

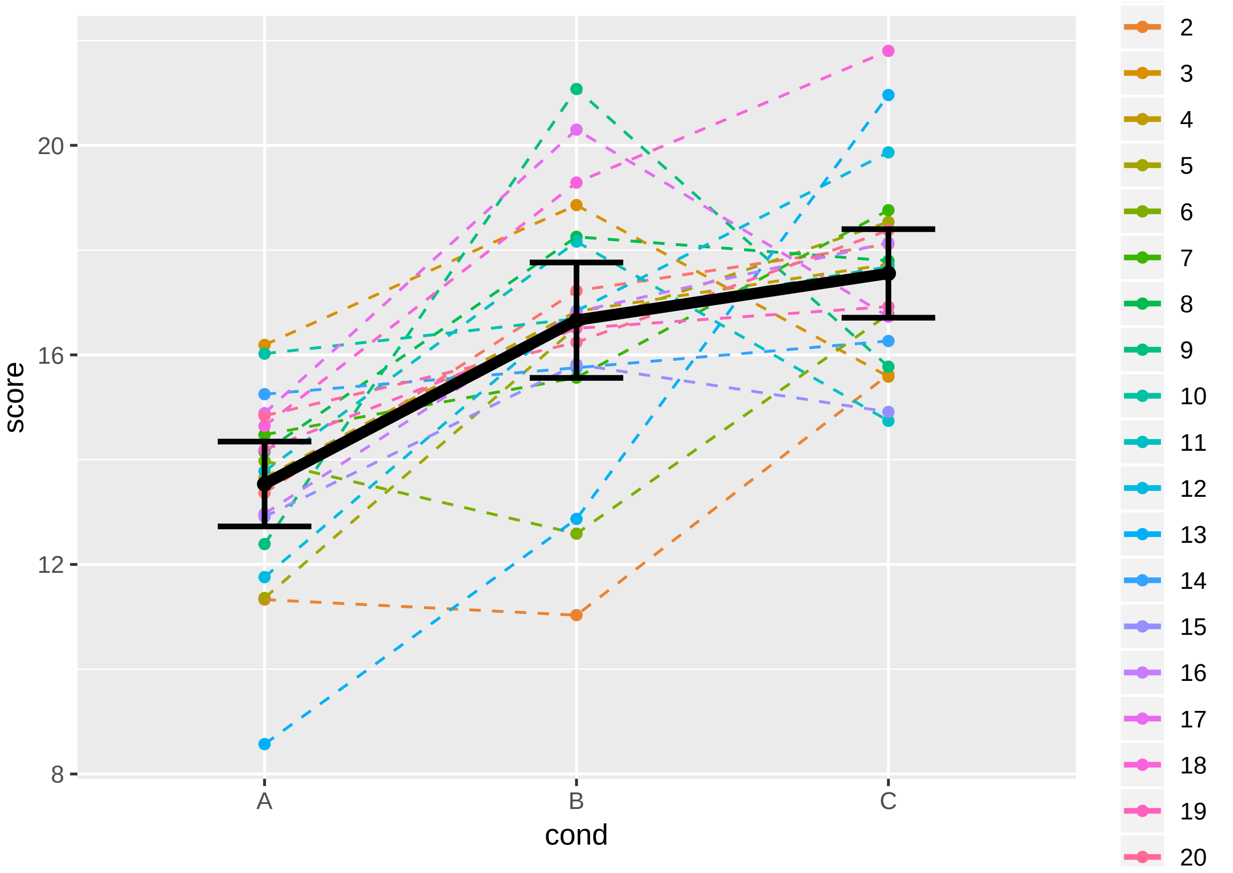


4.7 One-way repeated measures ANOVA

Generate dataset

```
set.seed(01)   # set seed
nrofsubs  = 20 # set number of subjects
nrofconds = 3  # set number of conditions
subj = as.factor(rep(1:nrofsubs,nrofconds))      # create array with subject IDs
cond = as.factor(rep(LETTERS[1:nrofconds],each=nrofsubs))   # create array with condition values
score = as.vector( replicate(
          nrofconds , rnorm(n = nrofsubs, mean = sample(8,1)+10 , sd = sample(5,1) )
        ) )                                     # create array with measurement values
data.long = data.frame(subj, cond, score);      # combine arrays into a data.frame
rm(list=setdiff(ls(), c("data.long", "nrofsubs","nrofconds"))) # delete arrays
```

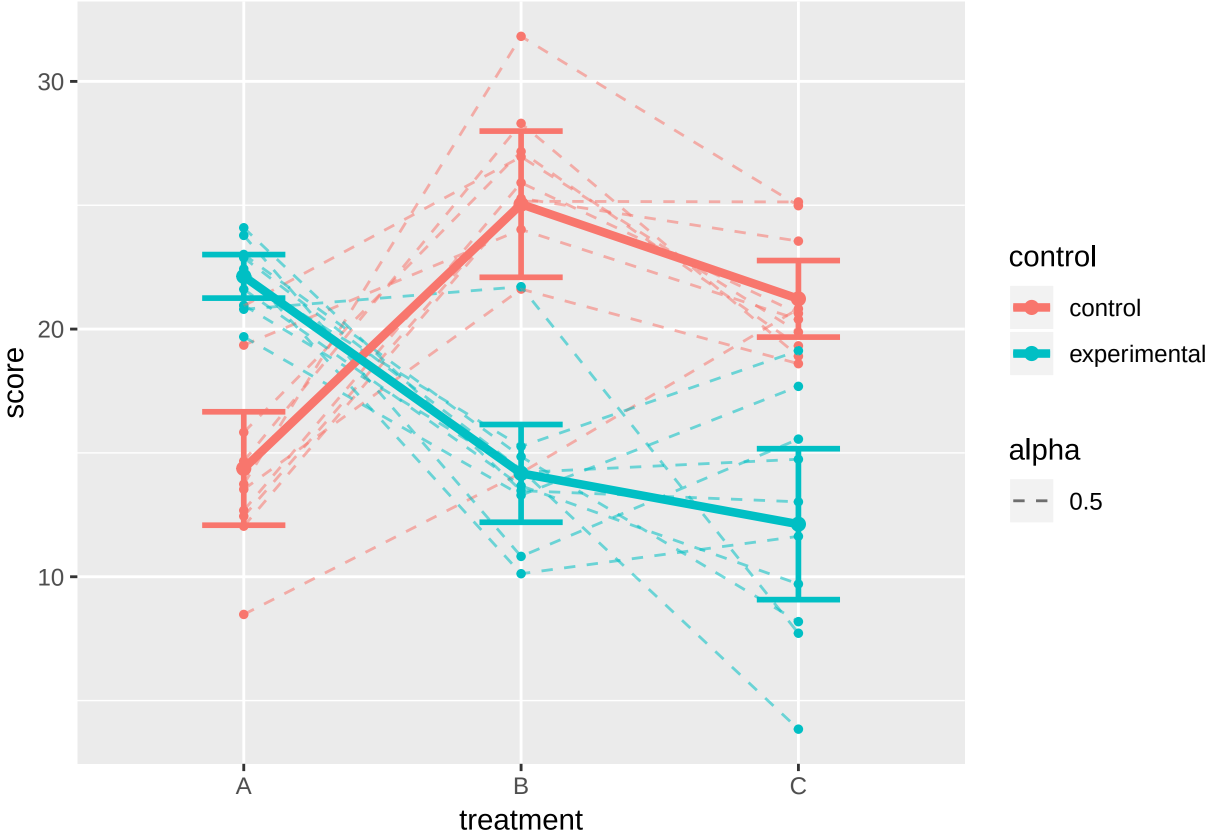
```
ggplot(data.long, aes(x=cond, y=score, group=1, colour=subj)) +
  geom_point   () +
  geom_line    ( linetype= "dashed", aes(group=subj) ) +
  geom_line    ( stat="summary", fun.y="mean",       size=2, colour="black", linetype="solid") +
  geom_point   ( stat="summary", fun.y="mean",       size=2, colour="black" ) +
  geom_errorbar( stat="summary", fun.data="mean_se", size=1, fun.args = 2, width = 0.3 )
```



4.8 Factorial repeated measures ANOVA

```
set.seed(02)  # set seed
nrofcondsf1 = 3  # set number of conditions for factor 1
nrofcondsf2 = 2  # set number of conditions for factor 2
nrofsubs    = 10 # set number of subjects
# create array with subject IDs
subj = as.factor(rep(1:(nrofsubs),times=nrofcondsf1*nrofcondsf2))
# create array witht treatment conditions
treatment = as.factor(rep(LETTERS[1:nrofcondsf1],each=nrofsubs*nrofcondsf2))
# create array with control / experimental
control   = as.factor(
  rep(rep(c("control","experimental"),each=nrofsubs),times=nrofcondsf1))
# create array with measurement values
score = as.vector( replicate(nrofcondsf1,
                   replicate(nrofcondsf2,
                   rnorm(n = (nrofsubs), mean = sample(14,1)+10 , sd = sample(5,1)
                                    ) ) ) )
# combine arrays into a data.frame
data.long = data.frame(subj, score, treatment, control);
# delete arrays
rm(list=setdiff(ls(), c("data.long", "nrofsubs","nrofconds")))
```

```
ggplot(data.long, aes(x=treatment, y=score, group=control, colour=control)) +
  geom_point   (size=1)  +
  geom_line    (linetype="dashed" ,aes(group=interaction(subj,control), alpha=.5)) +
  geom_line    ( stat="summary", fun.y   ="mean",    size=1.5 )  +
  geom_point   ( stat="summary", fun.y   ="mean",    size=2 ) +
  geom_errorbar( stat="summary", fun.data="mean_se", size=1, fun.args = 2, width = 0.3)
```


score
A
B
C
treatment
control
control
experimental
alpha