Portfolio 2.1, Methods 3, 2021, autumn semester

Sigurd Fyhn Sørensen

04/10/2021

Exercises and objectives

The objectives of the exercises of this assignment are:
1) Download and organise the data and model and plot staircase responses based on fits of logistic functions
2) Fit multilevel models for response times
3) Fit multilevel models for count data

REMEMBER: In your report, make sure to include code that can reproduce the answers requested in the exercises below (**MAKE A KNITTED VERSION**)
REMEMBER: This assignment will be part of your final portfolio

Exercise 1

Go to https://osf.io/ecxsj/files/ (https://osf.io/ecxsj/files/) and download the files associated with Experiment 2 (there should be 29).
The data is associated with Experiment 2 of the article at the following DOI https://doi.org/10.1016/j.concog.2019.03.007 (https://doi.org/10.1016/j.concog.2019.03.007)

1. Put the data from all subjects into a single data frame
2. Describe the data and construct extra variables from the existing variables
    i. add a variable to the data frame and call it *correct* (have it be a *logical* variable). Assign a 1 to each row where the subject indicated the correct answer and a 0 to each row where the subject indicated the incorrect answer (**Hint:** the variable *obj.resp* indicates whether the subject answered "even", *e* or "odd", *o*, and the variable *target_type* indicates what was actually presented.

```
df_exp <- df_exp %>%
  mutate(right_answer = if_else(target.type == "odd" & obj.resp == "o" | target.type
 == "even" & obj.resp == "e",1,0)) %>%
  mutate(right_answer = as.numeric(right_answer))

df_exp <- df_exp %>%
  mutate(right_answer = as.factor(right_answer)) %>%
  mutate(subject = as.factor(subject)) %>%
  mutate(task = as.factor(task)) %>%
  mutate(target.type = as.factor (target.type))

sum(is.na(df_exp) == TRUE)
```

```
## [1] 0
```

```
ii. describe what the following variables in the data frame contain, _trial.type_, _p
as_, _trial_, _target.contrast_, _cue_, _task_, _target_type_, _rt.subj_, _rt.obj_, _
obj.resp_, _subject_ and _correct_. (That means you can ignore the rest of the variab
les in your description). For each of them, indicate and argue for what `class` they
should be classified into, e.g. _factor_, _numeric_ etc.
```

** trial.type **: Contains two levels - staircase and experiment. The staircase trials are made as a way of adjusting the target.contrast. A 75% accuracy was aimed for. The experiment trials are what the actual experiment consists of (factor)** pas **: Perceptual awareness scale. Subjective rating of the experience of awareness of the stimulus.Ranging from 1 (no experience) to 4 (clear experience) (numeric)** trial **: Trial number, resets when trial.type changes (numeric)** target.contrast **: The contrast of the target stimulus relative to the background (numeric)** cue **: An indicator of which set of cue stimuli was used ranging from 0 to 35 (factor)** task **: Indication of if the cue was shown as singles, pairs or quadruplets (factor)** target_type **: Showing if the target variable is odd or even (factor)** rt.subj **: Reaction time on subjective rating (numeric)** rt.obj **: Task reaction time in milliseconds (numeric)** obj.resp **: The response given by the participant, e.g. even or odd (factor)** subject **: Participant ID (factor)** correct **: Indicating whether participants answered correctly (factor)

```
iii. for the staircasing part __only__, create a plot for each subject where you plot
the estimated function (on the _target.contrast_ range from 0-1) based on the fitted
values of a model (use `glm`) that models _correct_ as dependent on _target.contrast
_. These plots will be our _no-pooling_ model. Comment on the fits - do we have enoug
h data to plot the logistic functions?
```

```
m1 <- glm(right_answer ~ target.contrast*subject, data = filter(df_exp,trial.type ==
"staircase" ), family = binomial(link = "logit"))
```

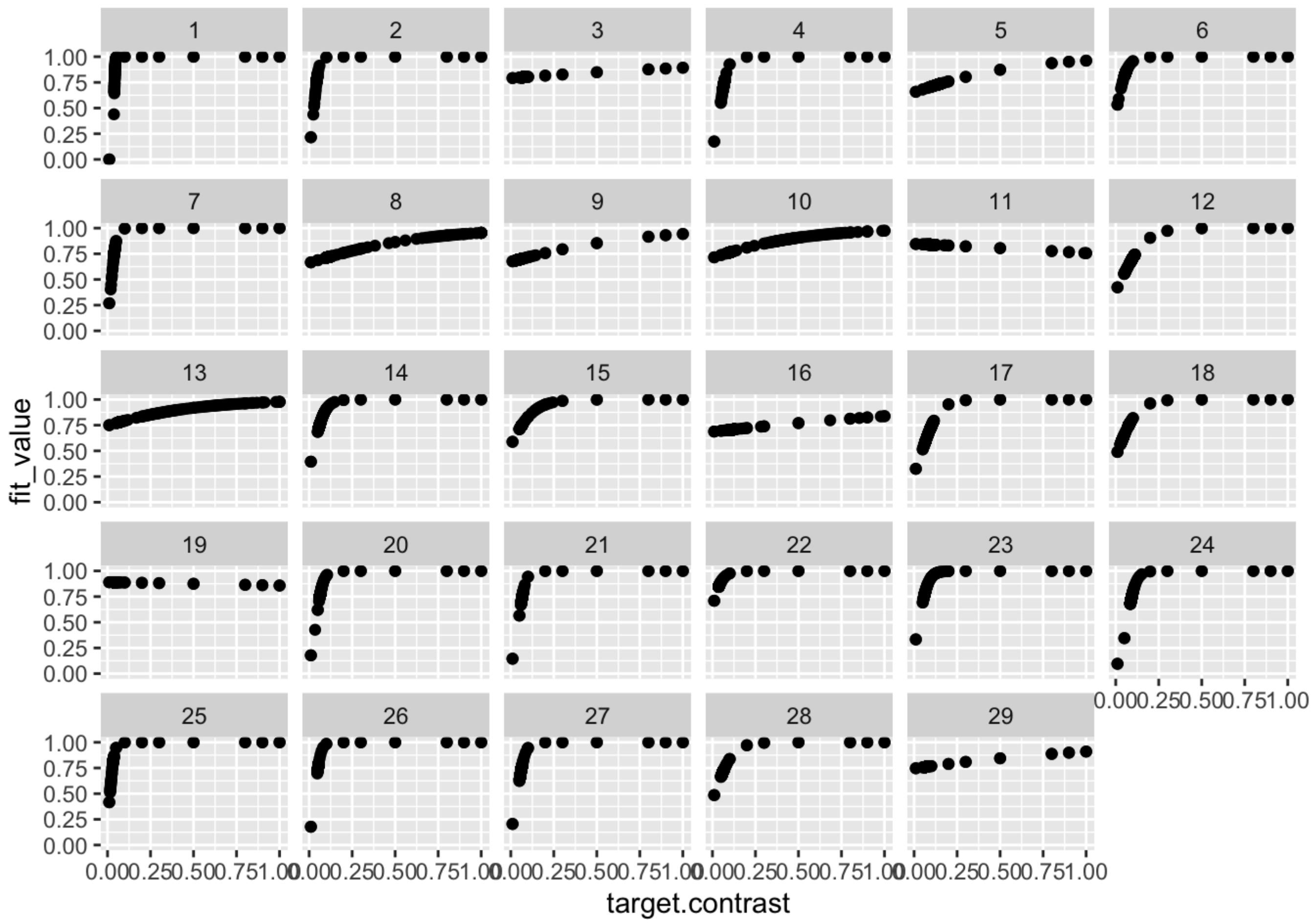
```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
invlogit(coef(m1))
```

```
##               (Intercept)           target.contrast                  subject2 
##              6.403916e-07              1.000000e+00              9.999952e-01 
##                  subject3                  subject4                  subject5 
##              9.999998e-01              9.999952e-01              9.999997e-01 
##                  subject6                  subject7                  subject8 
##              9.999992e-01              9.999964e-01              9.999997e-01 
##                  subject9                 subject10                 subject11 
##              9.999997e-01              9.999997e-01              9.999999e-01 
##                 subject12                 subject13                 subject14 
##              9.999990e-01              9.999998e-01              9.999987e-01 
##                 subject15                 subject16                 subject17 
##              9.999995e-01              9.999997e-01              9.999984e-01 
##                 subject18                 subject19                 subject20 
##              9.999992e-01              9.999999e-01              9.999951e-01 
##                 subject21                 subject22                 subject23 
##              9.999937e-01              9.999996e-01              9.999981e-01 
##                 subject24                 subject25                 subject26 
##              9.999910e-01              9.999980e-01              9.999944e-01 
##                 subject27                 subject28                 subject29 
##              9.999961e-01              9.999992e-01              9.999998e-01 
##  target.contrast:subject2  target.contrast:subject3  target.contrast:subject4 
##             1.145135e-132             6.241067e-164             1.346004e-144 
##  target.contrast:subject5  target.contrast:subject6  target.contrast:subject7 
##             3.786360e-163             4.602997e-150             2.372017e-132 
##  target.contrast:subject8  target.contrast:subject9 target.contrast:subject10 
##             3.012820e-163             2.293329e-163             4.537618e-163 
## target.contrast:subject11 target.contrast:subject12 target.contrast:subject13 
##             1.595643e-164             2.099260e-158             4.266843e-163 
## target.contrast:subject14 target.contrast:subject15 target.contrast:subject16 
##             3.497137e-151             2.442752e-158             6.674961e-164 
## target.contrast:subject17 target.contrast:subject18 target.contrast:subject19 
##             1.025591e-155             8.563314e-157             2.105787e-164 
## target.contrast:subject20 target.contrast:subject21 target.contrast:subject22 
##             2.541380e-142             3.584018e-142             9.595477e-151 
## target.contrast:subject23 target.contrast:subject24 target.contrast:subject25 
##             6.700431e-148             7.672048e-147             1.685355e-129 
## target.contrast:subject26 target.contrast:subject27 target.contrast:subject28 
##             7.626064e-137             4.700550e-144             4.378820e-156 
## target.contrast:subject29 
##             9.797932e-164
```

```
df_exp_stair <- df_exp %>%
  filter(trial.type == "staircase") %>%
  mutate(fit_value = fitted.values(m1))

df_exp_stair %>%
  ggplot(aes(x = target.contrast, y = fit_value)) + geom_point() + facet_wrap(~subjec
t)
```



As can be seen by the above plot we do not have something that resembles a sigmoid function.The “shape” is almost there but with a lot of holes. Having a data point for each x with a step of 0.01 should result in a perfect sigmoid fit.

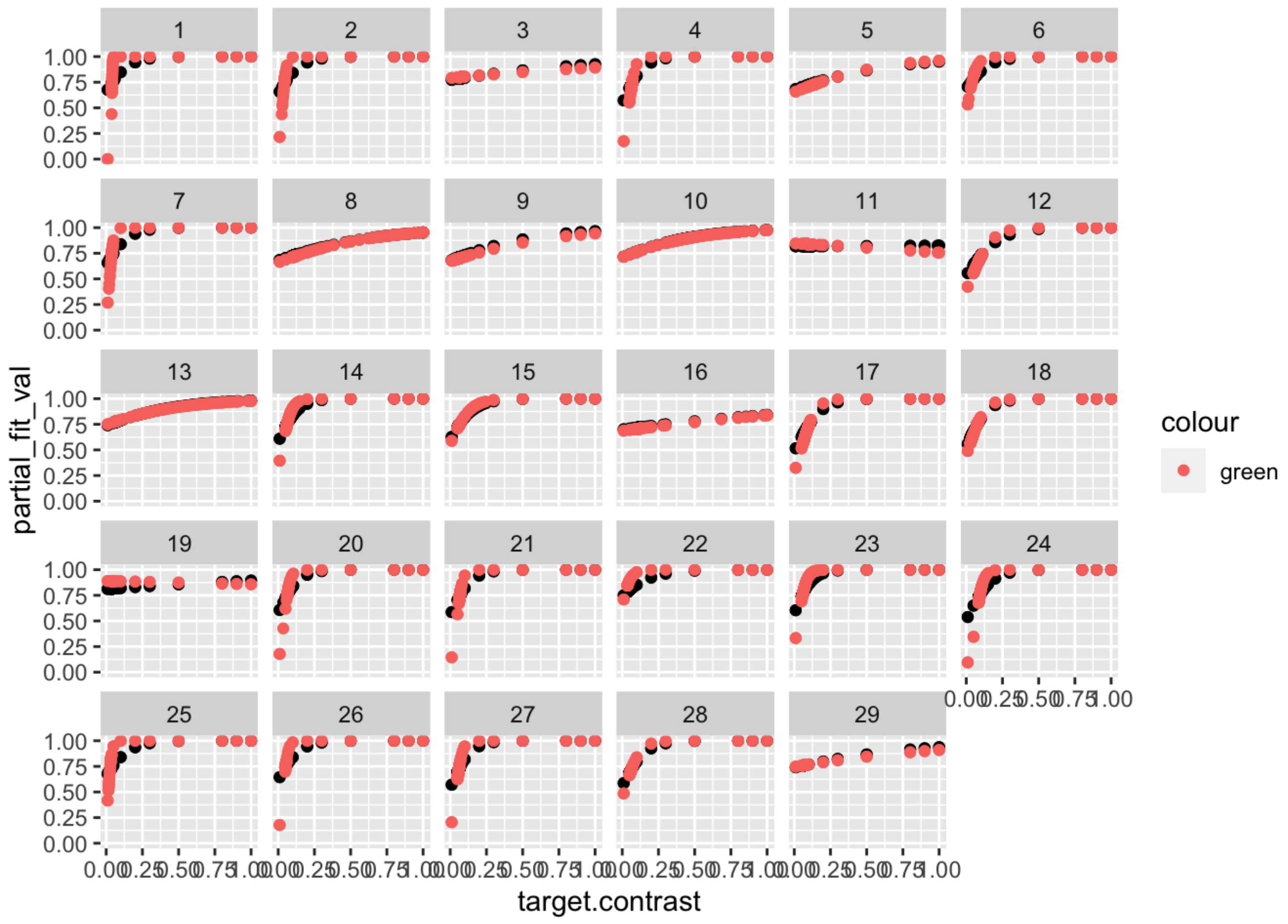
```
iv. on top of those plots, add the estimated functions (on the _target.contrast_ rang
e from 0-1) for each subject based on partial pooling model (use `glmer` from the pac
kage `lme4`) where unique intercepts and slopes for _target.contrast_ are modelled fo
r each _subject_
```

```
m2 <- glmer(right_answer ~ target.contrast + (1+target.contrast|subject), data = df_e
xp_stair, family = binomial(link = "logit"))
summary(m2)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: right_answer ~ target.contrast + (1 + target.contrast | subject)
##    Data: df_exp_stair
##
##      AIC      BIC   logLik deviance df.resid
##   5988.5   6021.6  -2989.2   5978.5     5598
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.7671  0.0068  0.5532  0.5915  0.9264
##
## Random effects:
##  Groups  Name            Variance Std.Dev. Corr
##  subject (Intercept)      0.2717  0.5213
##          target.contrast 42.7575  6.5389   -0.84
## Number of obs: 5603, groups:  subject, 29
##
## Fixed effects:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)       0.5619     0.1595   3.523 0.000427 ***
## target.contrast   8.7132     2.3879   3.649 0.000263 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## trgt.cntrst -0.909
```

```
df_exp_stair <- df_exp_stair %>%
  mutate(partial_fit_val = fitted.values(m2))

df_exp_stair %>%
  ggplot(aes(x = target.contrast, y = partial_fit_val)) + geom_point() + geom_point(a
es(x = target.contrast, y = fit_value, col = "green")) + facet_wrap(~subject)
```



```
v. in your own words, describe how the partial pooling model allows for a better fit
for each subject
```

Exercise 2

Now we **only** look at the *experiment* trials (*trial.type*)

```
df_exp_exp <- df_exp %>%
  filter(trial.type == "experiment")
```

1. Pick four subjects and plot their Quantile-Quantile (Q-Q) plots for the residuals of their objective response times (*rt.obj*) based on a model where only intercept is modelled.

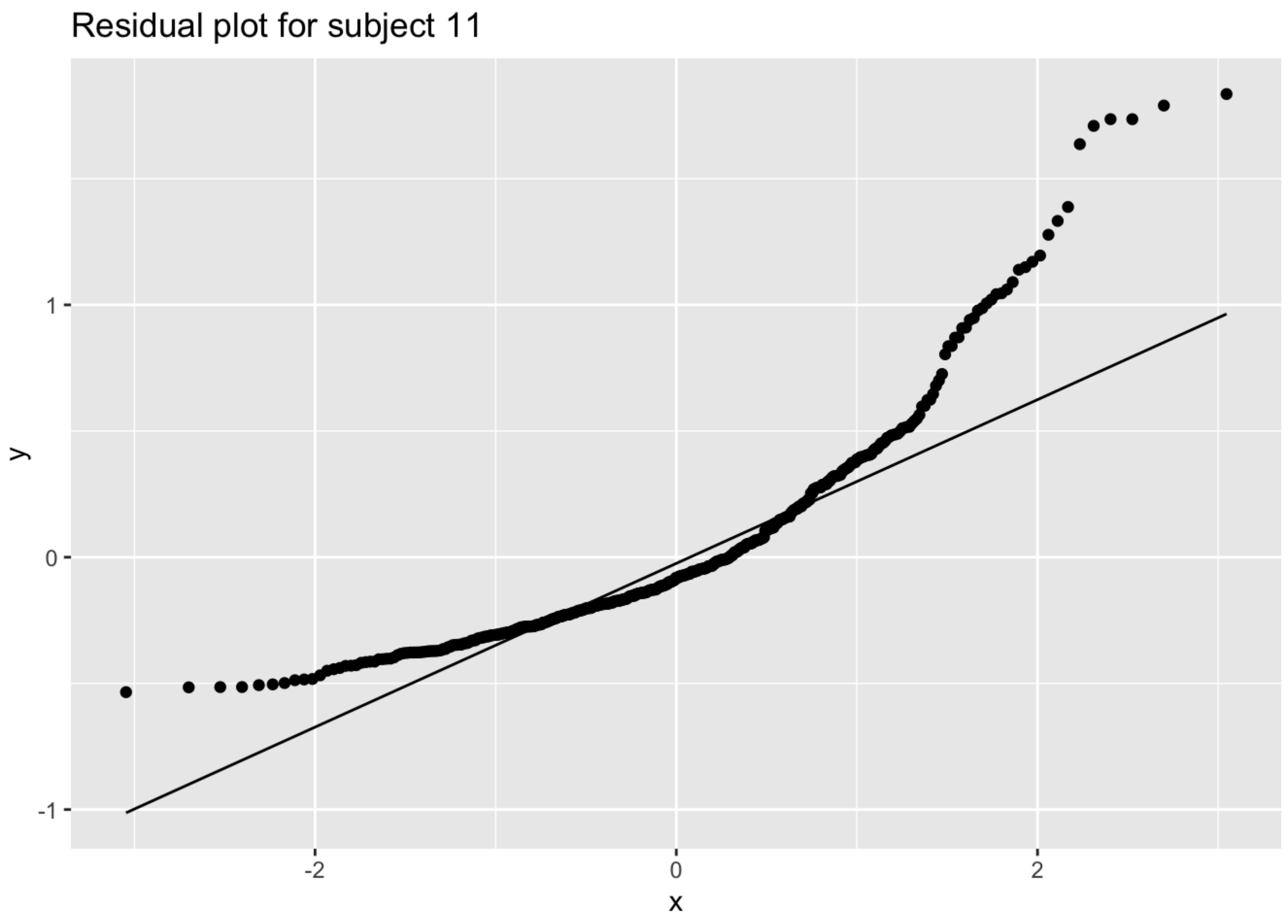
```
m3 <- lmer(rt.obj ~ (1|subject), data = df_exp_exp)

df_exp_exp <- df_exp_exp %>%
  mutate(rt.obj_fit_val = fitted(m3)) %>%
  mutate(rt.obj_resid = resid(m3))
```

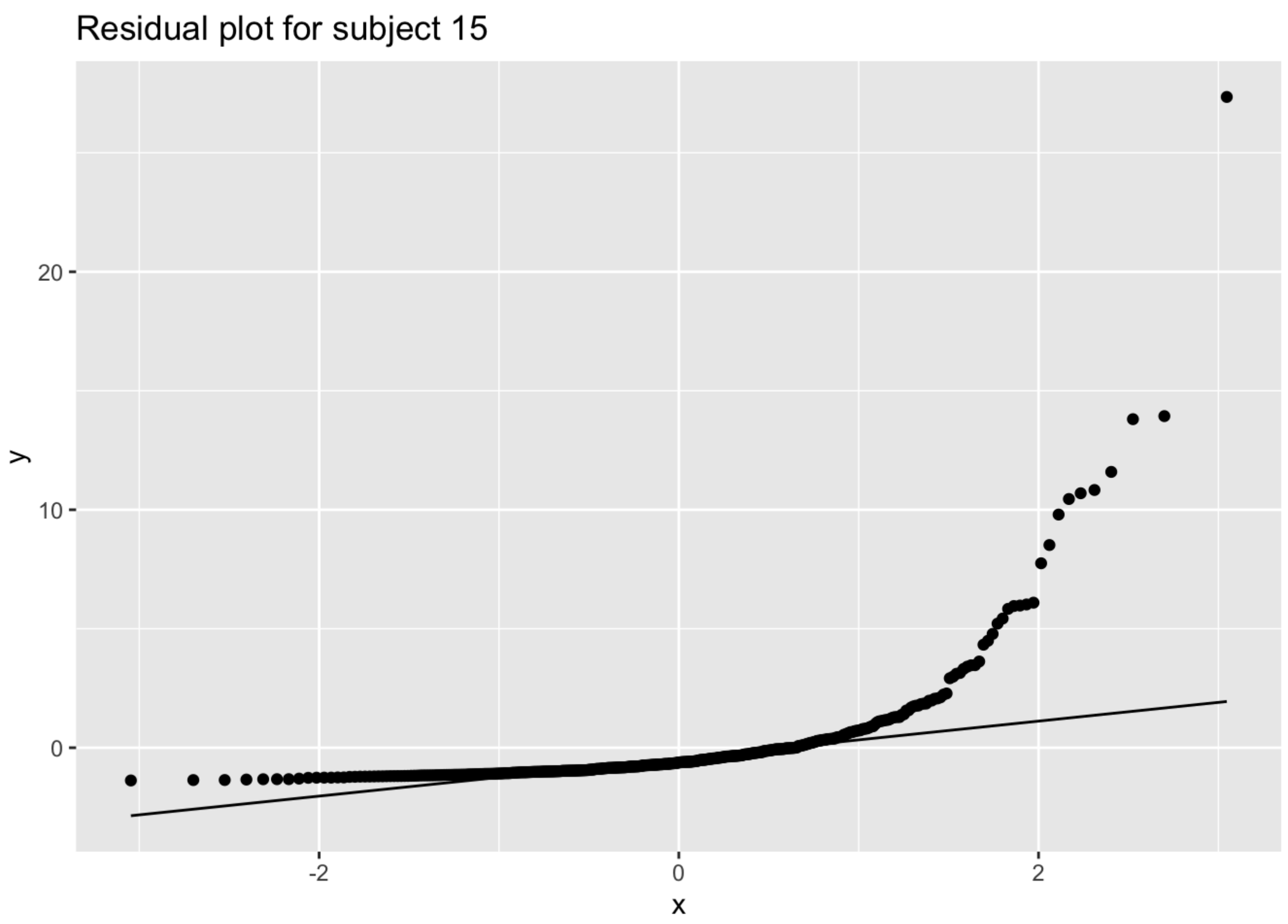
```
#For-loop not printing out plot.. Don't know why as I've done similair loops before.
for (i in sample(1:length(unique(df_exp_exp$subject)),4)){
  df_exp_exp %>%
    filter(subject == as.character(i)) %>%
    ggplot(aes(sample = rt.obj_resid)) + stat_qq()+stat_qq_line()
}
```

```
#Boring tedious way
subjects <- sample(1:length(unique(df_exp_exp$subject)),4)
```

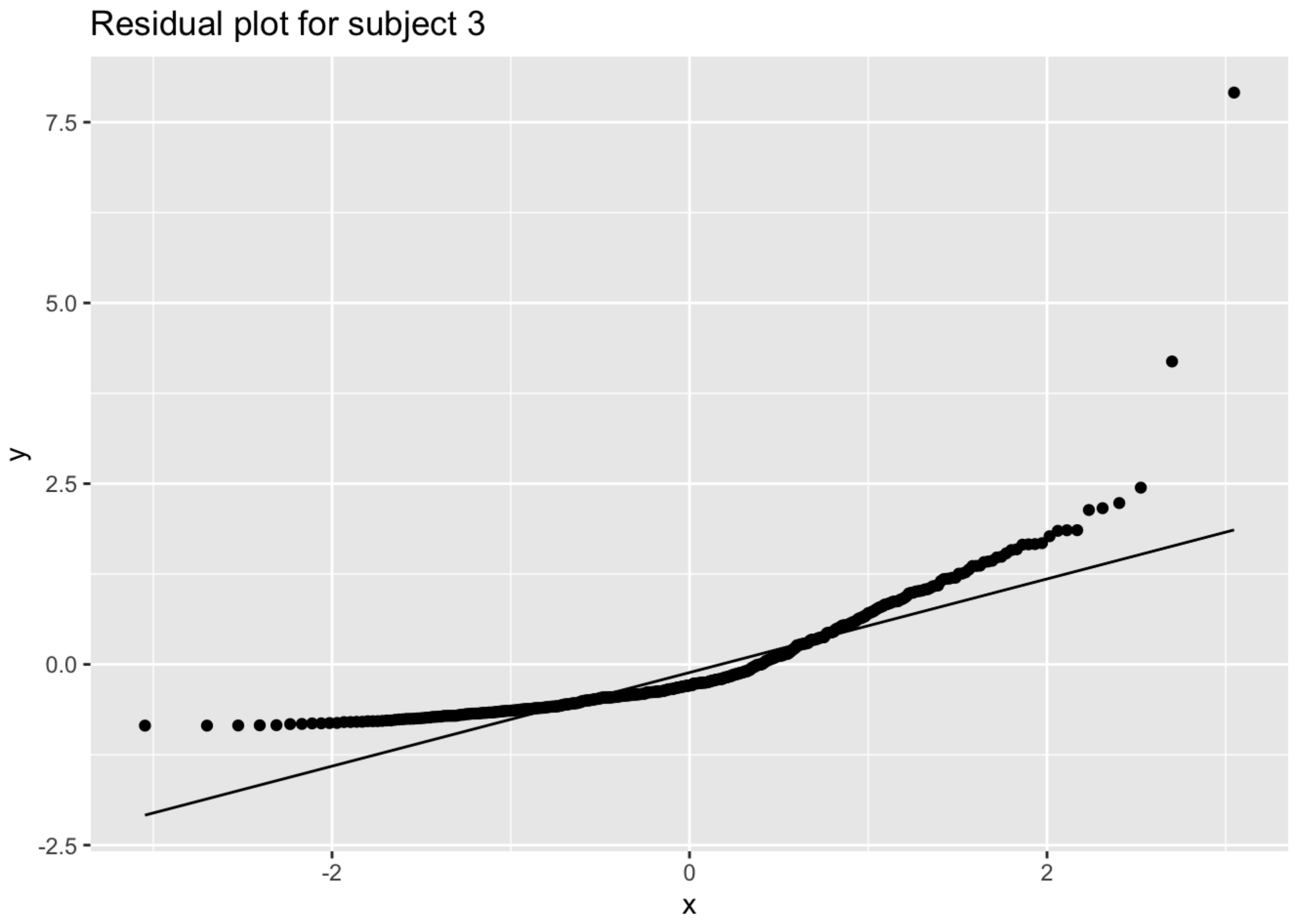
```
df_exp_exp %>%
  filter(subject == as.character(subjects[1])) %>%
  ggplot(aes(sample = rt.obj_resid)) + stat_qq() + stat_qq_line() + labs(title = past
e("Residual plot for subject", subjects[1],sep = " "))
```



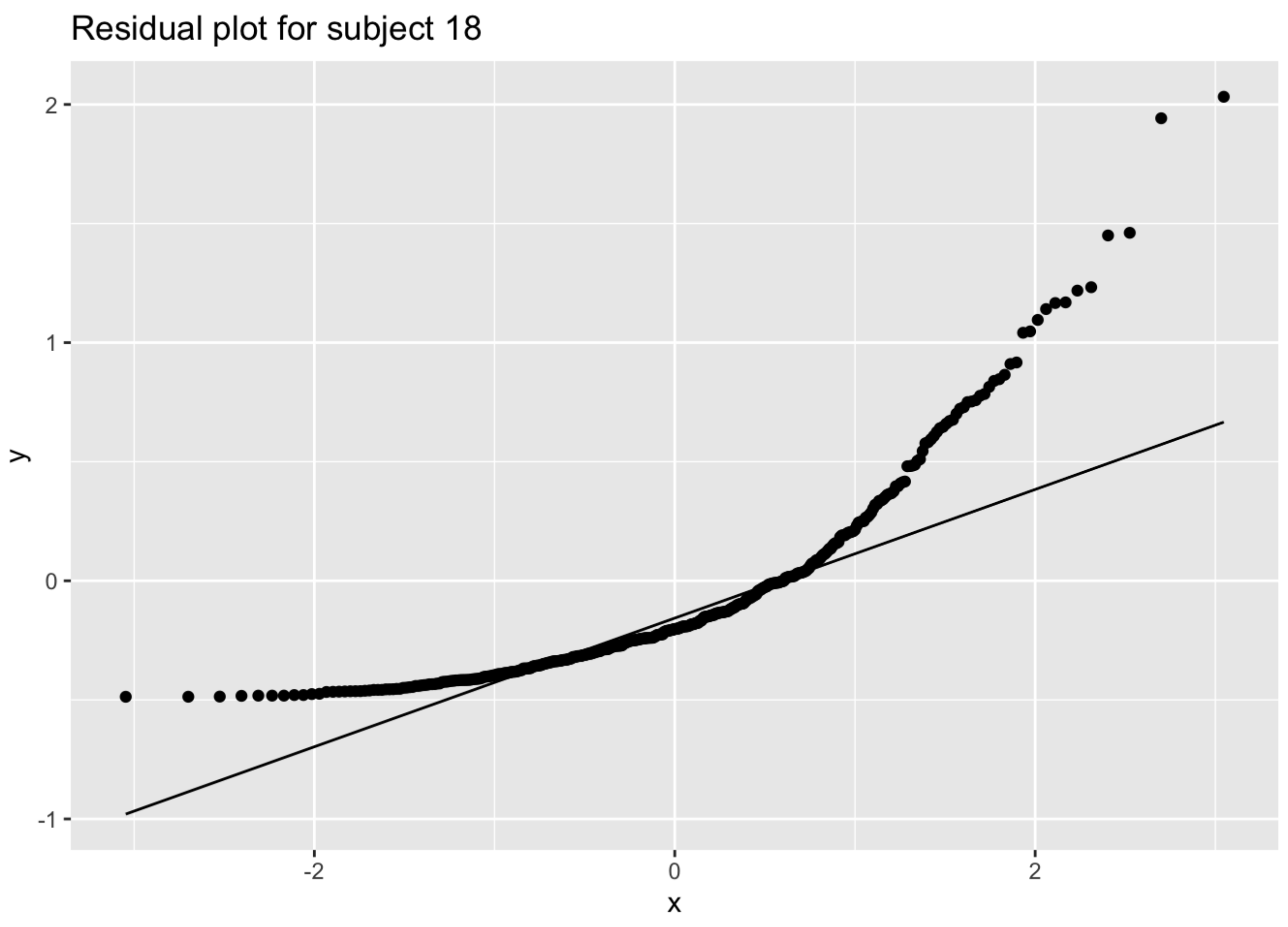
```
df_exp_exp %>%
  filter(subject == as.character(subjects[2])) %>%
  ggplot(aes(sample = rt.obj_resid)) + stat_qq() + stat_qq_line() + labs(title = past
e("Residual plot for subject", subjects[2],sep = " "))
```



```
df_exp_exp %>%
  filter(subject == as.character(subjects[3])) %>%
  ggplot(aes(sample = rt.obj_resid)) + stat_qq() + stat_qq_line() + labs(title = past
e("Residual plot for subject", subjects[3],sep = " "))
```



```
df_exp_exp %>%
  filter(subject == as.character(subjects[4])) %>%
  ggplot(aes(sample = rt.obj_resid)) + stat_qq() + stat_qq_line() + labs(title = past
e("Residual plot for subject", subjects[4],sep = " "))
```



```
i. comment on these
```

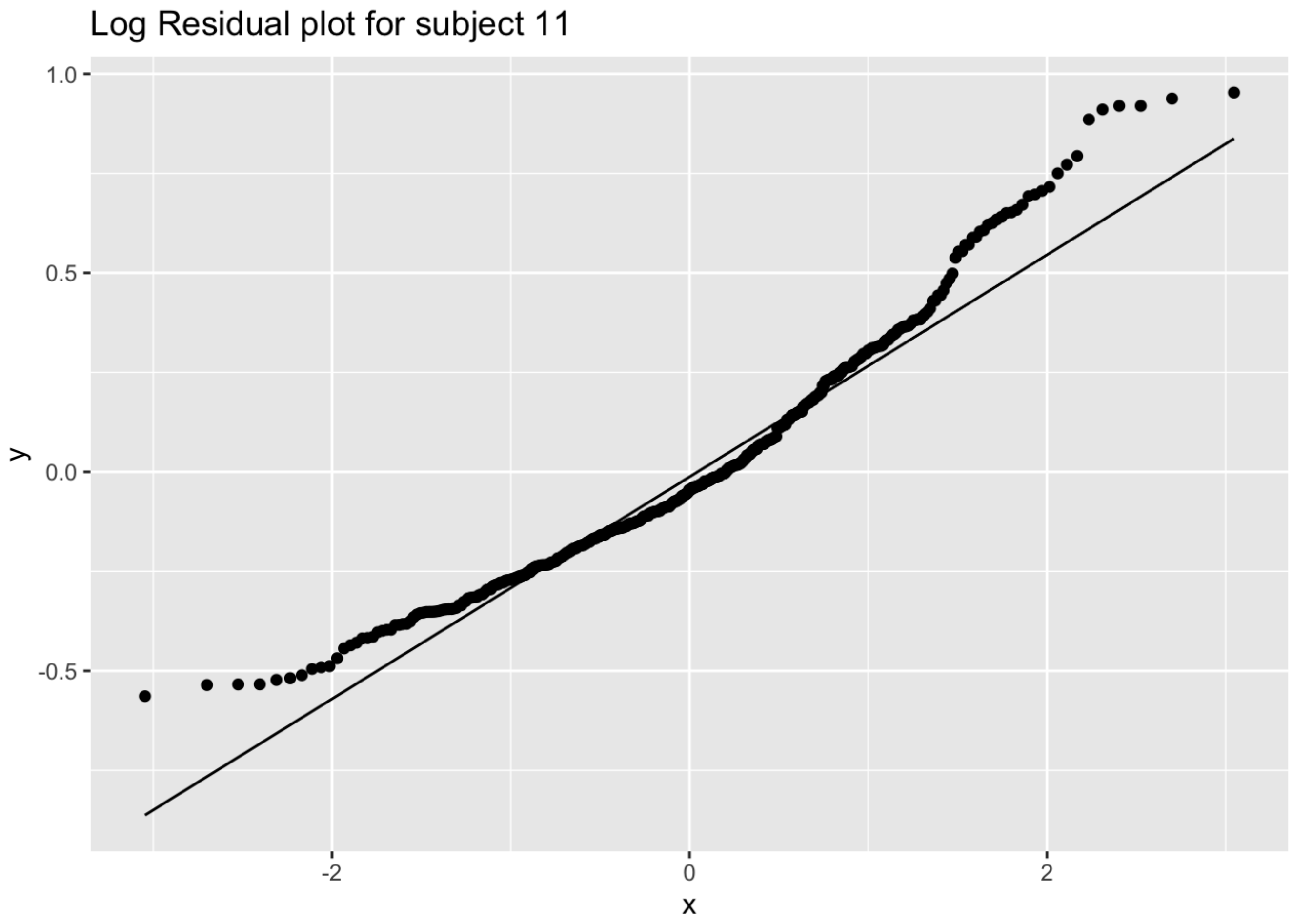
The residuals does not look normally distributed some worse than others. As this is an assumption of the linear mixed-effect models measures to counteract is required.

```
ii. does a log-transformation of the response time data improve the Q-Q-plots?
```

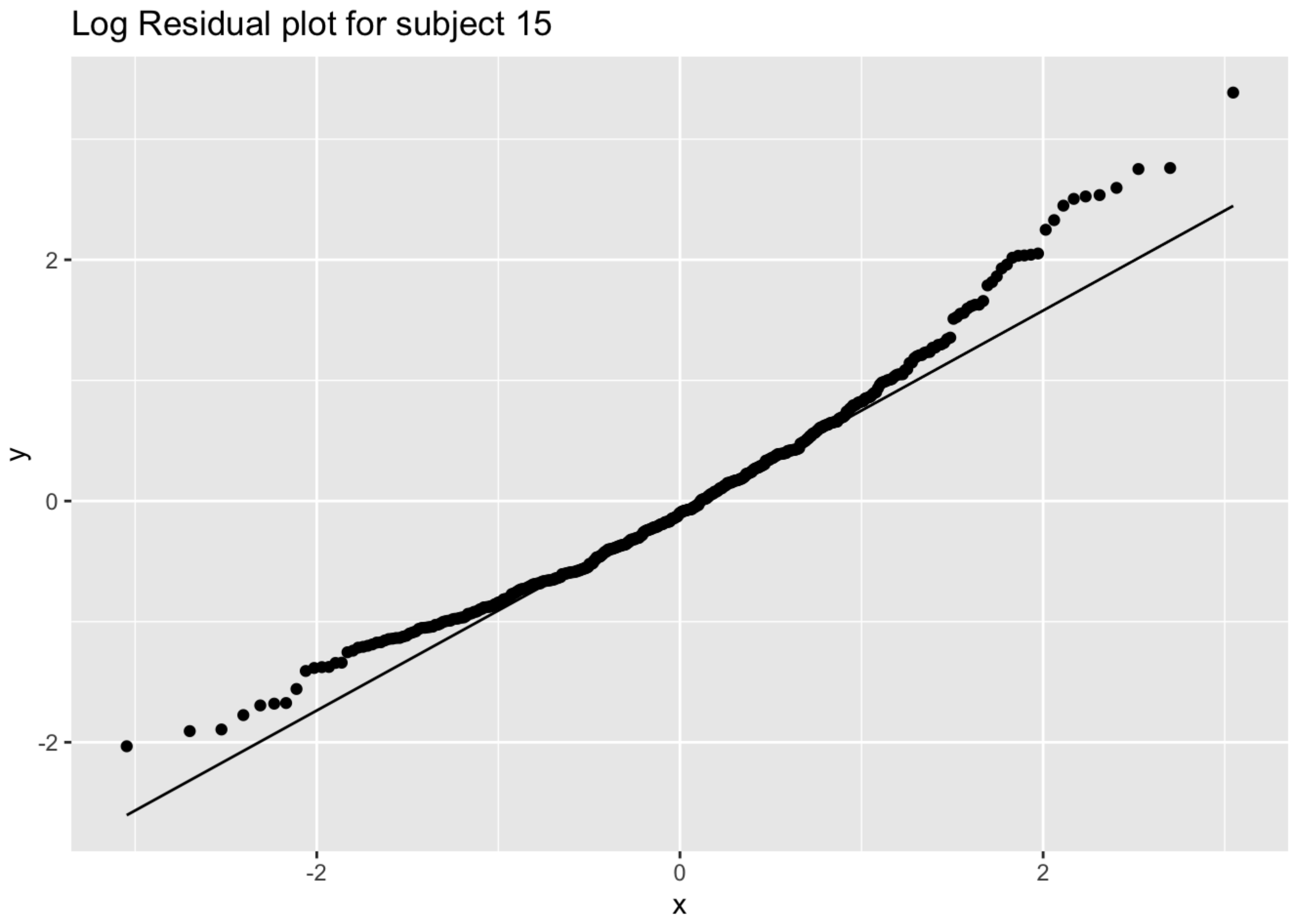
```
m4 <- lmer(log(rt.obj) ~ (1|subject), data = df_exp_exp)

df_exp_exp <- df_exp_exp %>%
  mutate(rt.obj_fit_val_log = fitted(m4)) %>%
  mutate(rt.obj_resid_log = resid(m4))
```

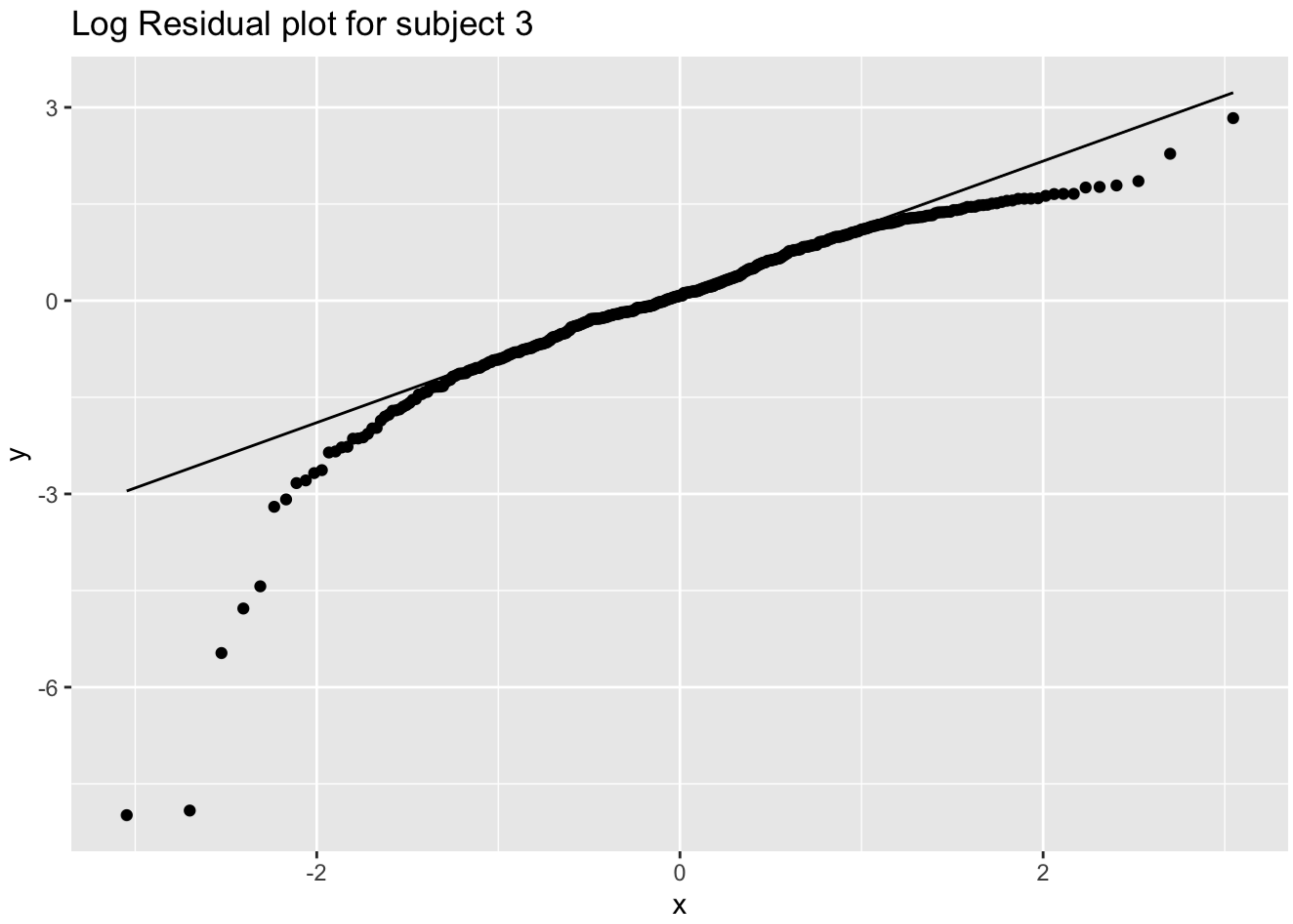
```
#Boring tedious way with log
df_exp_exp %>%
  filter(subject == as.character(subjects[1])) %>%
  ggplot(aes(sample = rt.obj_resid_log)) + stat_qq() + stat_qq_line() + labs(title =
 paste("Log Residual plot for subject", subjects[1],sep = " "))
```



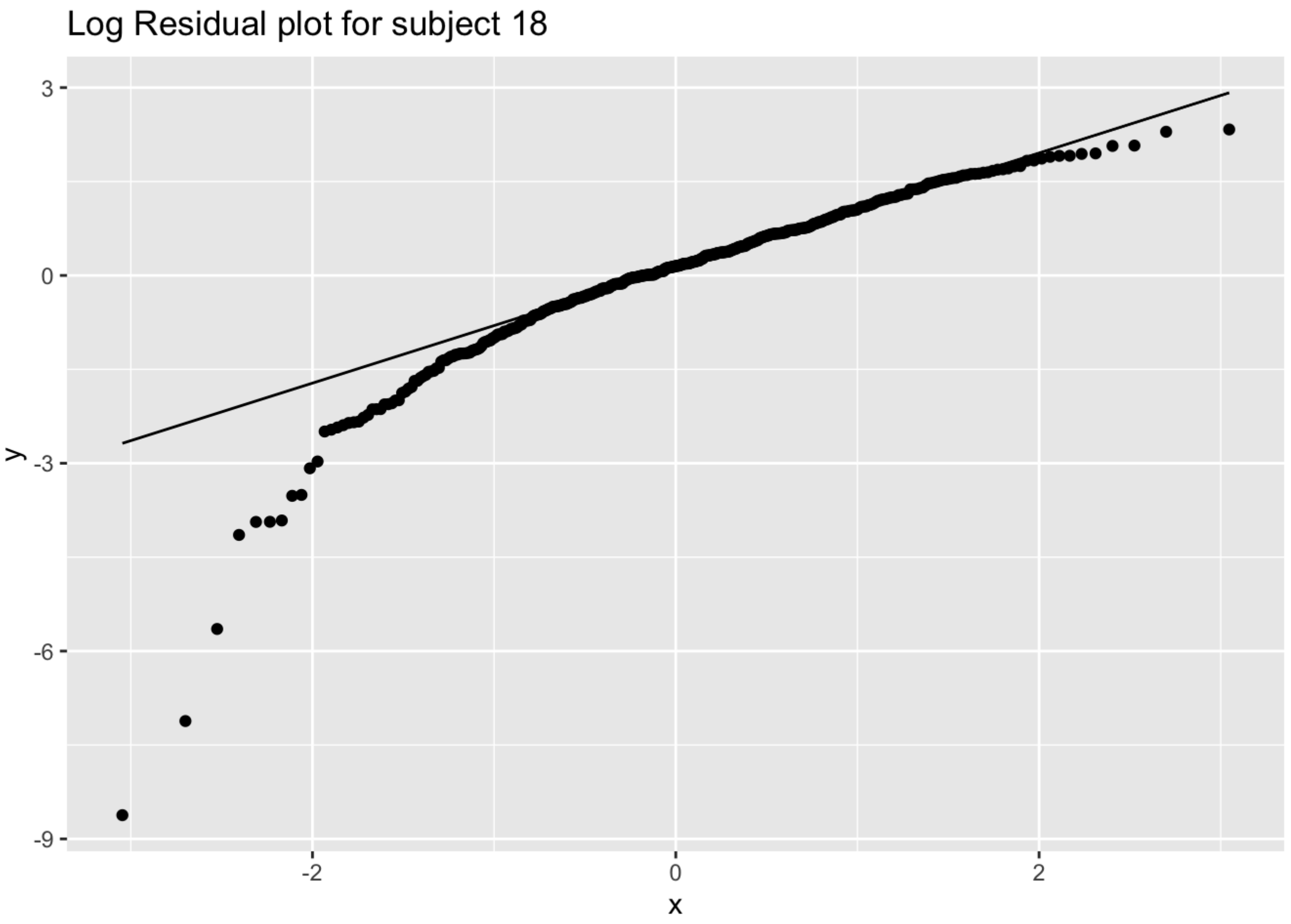
```
df_exp_exp %>%
  filter(subject == as.character(subjects[2])) %>%
  ggplot(aes(sample = rt.obj_resid_log)) + stat_qq() + stat_qq_line() + labs(title =
 paste("Log Residual plot for subject", subjects[2],sep = " "))
```



```
df_exp_exp %>%
  filter(subject == as.character(subjects[3])) %>%
  ggplot(aes(sample = rt.obj_resid_log)) + stat_qq() + stat_qq_line() + labs(title =
 paste("Log Residual plot for subject", subjects[3],sep = " "))
```



```
df_exp_exp %>%
  filter(subject == as.character(subjects[4])) %>%
  ggplot(aes(sample = rt.obj_resid_log)) + stat_qq() + stat_qq_line() + labs(title =
 paste("Log Residual plot for subject", subjects[4],sep = " "))
```



It generally generated some better QQ-plots for some of the subjects. But there are still some inconsistency leaving some of the QQ-plots still showing skewness.

2. Now do a partial pooling model modelling objective response times as dependent on *task*? (set `REML=FALSE` in your `lmer`-specification)

```
m5 <- lmer(rt.obj ~ task + (1|subject) + (1|pas), data = df_exp_exp, REML = FALSE)
summary(m5)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: rt.obj ~ task + (1 | subject) + (1 | pas)
##    Data: df_exp_exp
##
##      AIC      BIC   logLik deviance df.resid
##  61917.5  61962.1 -30952.7  61905.5    12522
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
##  -0.640  -0.153  -0.065   0.046 101.556
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  subject  (Intercept) 0.1013   0.3183
##  pas      (Intercept) 0.0342   0.1849
##  Residual             8.1542   2.8555
## Number of obs: 12528, groups:  subject, 29; pas, 4
##
## Fixed effects:
##                  Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)     1.085e+00  1.185e-01  8.312e+00   9.152 1.27e-05 ***
## taskquadruplet -1.611e-01  6.251e-02  1.250e+04  -2.576  0.00999 **
## tasksingles    -1.557e-01  6.283e-02  1.248e+04  -2.478  0.01321 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) tskqdr
## taskqudrplt -0.262
## tasksingles -0.267  0.495
```

```
MuMIn::r.squaredGLMM(m5)
```

```
## Warning: 'r.squaredGLMM' now calculates a revised statistic. See the help page.
```

```
##               R2m        R2c
## [1,] 0.0006726226 0.01701042
```

```
i. which would you include among your random effects and why? (support your choices w
ith relevant measures, taking into account variance explained and number of parameter
s going into the modelling)
```

```
partpoolm1.1 <- lmer(log(rt.obj) ~ task + (1 | subject), data = df_exp_exp, REML=FALS
E)
partpoolm1.2 <- lmer(log(rt.obj) ~ task + (1 | trial), data = df_exp_exp, REML=FALSE)
partpoolm1.3 <- lmer(log(rt.obj) ~ task + (1 | trial)+ (1|subject), data = df_exp_ex
p, REML=FALSE)
partpoolm1.4 <- lmer(log(rt.obj) ~ task + (1 + task | subject), data = df_exp_exp, RE
ML=FALSE)
```

```
## boundary (singular) fit: see ?isSingular
```

```
## Warning: Model failed to converge with 1 negative eigenvalue: -7.6e+02
```

```
partpoolm1.5 <- lmer(log(rt.obj) ~ task + (1 + task | subject) + (1 | trial), data =
 df_exp_exp, REML = F)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.00270478 (tol = 0.002, component 1)
```

```
anova(partpoolm1.1,partpoolm1.2,partpoolm1.3,partpoolm1.4,partpoolm1.5)
```

```
## Data: df_exp_exp
## Models:
## partpoolm1.1: log(rt.obj) ~ task + (1 | subject)
## partpoolm1.2: log(rt.obj) ~ task + (1 | trial)
## partpoolm1.3: log(rt.obj) ~ task + (1 | trial) + (1 | subject)
## partpoolm1.4: log(rt.obj) ~ task + (1 + task | subject)
## partpoolm1.5: log(rt.obj) ~ task + (1 + task | subject) + (1 | trial)
##              npar   AIC   BIC logLik deviance   Chisq Df Pr(>Chisq)
## partpoolm1.1    5 29685 29722 -14838    29675
## partpoolm1.2    5 32560 32597 -16275    32550    0.00  0
## partpoolm1.3    6 29460 29505 -14724    29448 3102.21  1     <2e-16 ***
## partpoolm1.4   10 29560 29635 -14770    29540    0.00  4          1
## partpoolm1.5   11 29328 29409 -14653    29306  234.65  1     <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

m1.3 and m1.5 both seem to perform well while this seem to be due to having random intercept for both trial and subject. Adding a random slope of task for subjects improves the model slightly following AIC.

A random intercept for both task and subject is theoretically warranted as well as the effect of task being different between individuals.

partpoolm1.5 is therefore selected.

```
ii. explain in your own words what your chosen models says about response times betwe
en the different tasks
```

```
summary(partpoolm1.5)
```

```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: log(rt.obj) ~ task + (1 + task | subject) + (1 | trial)
##    Data: df_exp_exp
## 
##      AIC      BIC   logLik deviance df.resid 
##  29327.7  29409.5 -14652.8  29305.7    12517 
## 
## Scaled residuals: 
##      Min       1Q   Median       3Q      Max 
## -10.5363  -0.4958  -0.0279   0.5119   8.0799 
## 
## Random effects:
##  Groups   Name           Variance Std.Dev. Corr       
##  trial    (Intercept)    0.02995  0.17307             
##  subject  (Intercept)    0.14916  0.38621             
##           taskquadruplet 0.00400  0.06325   0.35      
##           tasksingles    0.03227  0.17965   0.38 -0.58
##  Residual                0.57917  0.76103             
## Number of obs: 12528, groups:  trial, 432; subject, 29
## 
## Fixed effects:
##                Estimate Std. Error       df t value Pr(>|t|)    
## (Intercept)    -0.26407    0.07317 29.75926  -3.609  0.00111 ** 
## taskquadruplet -0.07225    0.02054 28.64575  -3.518  0.00147 ** 
## tasksingles    -0.17868    0.03740 29.11742  -4.778 4.66e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Correlation of Fixed Effects:
##             (Intr) tskqdr
## taskqudrplt  0.099       
## tasksingles  0.279 -0.113
## optimizer (nloptwrap) convergence code: 0 (OK)
## Model failed to converge with max|grad| = 0.00270478 (tol = 0.002, component 1)
```

The estimated effect sizes for quadruplet and singles are negative, which indicates that these tasks will result in lower reaction time compared to single_task.

3. Now add *pas* and its interaction with *task* to the fixed effects

    i. how many types of group intercepts (random effects) can you add without ending up with convergence issues or singular fits?

```
partpoom2.1 <- lmer(log(rt.obj) ~ task*pas + (1|subject), data = df_exp_exp, REML = F
)
partpoolm2.2 <- lmer(log(rt.obj) ~ task*pas + (1|trial) + (1|subject), data = df_exp_
exp, REML = F)
partpoolm2.3 <- lmer(log(rt.obj) ~ task*pas + (1|trial) + (1|subject) +  (1|cue), dat
a = df_exp_exp, REML = F)
partpoolm2.4 <- lmer(log(rt.obj) ~ task*pas + (1|trial) + (1|subject) +  (1|cue) + (1
|target.contrast), data = df_exp_exp, REML = F)
```

```
## boundary (singular) fit: see ?isSingular
```

When adding a fourth intercept an error occurs (singular fit)

```
ii. create a model by adding random intercepts (without modelling slopes) that result
s in a singular fit - then use `print(VarCorr(<your.model>), comp='Variance')` to ins
pect the variance vector - explain why the fit is singular (Hint: read the first para
graph under details in the help for `isSingular`)'
```

```
print(VarCorr(partpoolm2.4), comp='Variance')
```

```
##  Groups          Name        Variance
##  trial           (Intercept) 0.0268422
##  cue             (Intercept) 0.0040356
##  target.contrast (Intercept) 0.0000000
##  subject         (Intercept) 0.1628691
##  Residual                    0.5715602
```

Some of the variances approaches zero.

```
iii. in your own words - how could you explain why your model would result in a singu
lar fit?
```

The model will give an error of “is singular” when the random effect’s variance is nearly zero. Could be due to different things. Adding many random parameters will results in a model, where they each do NOT explain a lot of variance in the data. Or a certain selection of random effect is not warranted. It can also occur when the variables are highly correlated (i.e. correlation close to either -1 or 1).

Exercise 3

1. Initialise a new data frame, `data.count`. *count* should indicate the number of times they categorized their experience as *pas* 1-4 for each *task*. I.e. the data frame would have for subject 1: for task:singles, pas1 was used # times, pas2 was used # times, pas3 was used # times and pas4 was used # times. You would then do the same for task:pairs and task:quadruplet

```
## you can start from this if you want to, but you can also make your own from scratc
h
data.count <- data.frame(count = numeric(),
                         pas = numeric(), ## remember to make this into a factor afte
rwards
                         task = numeric(), ## and this too
                         subject = numeric()) ## and this too
```

```
df_count <- df_exp %>%
  group_by(subject, task, pas) %>%
  summarise(count = n()) %>%
  mutate(pas = as.factor(pas))
```

```
## `summarise()` has grouped output by 'subject', 'task'. You can override using the
`.groups` argument.
```

2. Now fit a multilevel model that models a unique “slope” for *pas* for each *subject* with the interaction between *pas* and *task* and their main effects being modelled.

```
m3.1 <- glmer(count ~ pas*task + (1+pas|subject), family = poisson(link = "log"), dat
a = df_count)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.00343245 (tol = 0.002, component 1)
```

```
summary(m3.1)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: count ~ pas * task + (1 + pas | subject)
##    Data: df_count
## 
##      AIC      BIC   logLik deviance df.resid 
##   3148.4   3232.7  -1552.2   3104.4      318 
## 
## Scaled residuals: 
##     Min      1Q  Median      3Q     Max 
## -4.3872 -0.7853 -0.0472  0.7552  6.5435 
## 
## Random effects:
##  Groups  Name        Variance Std.Dev. Corr             
##  subject (Intercept) 0.3322   0.5764                    
##          pas2        0.3801   0.6165   -0.75            
##          pas3        1.1956   1.0934   -0.84  0.63      
##          pas4        2.3731   1.5405   -0.86  0.42  0.72
## Number of obs: 340, groups:  subject, 29
## 
## Fixed effects:
##                     Estimate Std. Error z value Pr(>|z|)    
## (Intercept)          4.03550    0.10974  36.773  < 2e-16 ***
## pas2                -0.02364    0.11960  -0.198 0.843332    
## pas3                -0.51322    0.20715  -2.478 0.013230 *  
## pas4                -0.77251    0.29073  -2.657 0.007880 ** 
## taskquadruplet       0.11495    0.03127   3.676 0.000237 ***
## tasksingles         -0.23090    0.03418  -6.755 1.43e-11 ***
## pas2:taskquadruplet -0.11377    0.04605  -2.470 0.013500 *  
## pas3:taskquadruplet -0.20912    0.05287  -3.956 7.63e-05 ***
## pas4:taskquadruplet -0.21508    0.05230  -4.113 3.91e-05 ***
## pas2:tasksingles     0.19537    0.04830   4.045 5.23e-05 ***
## pas3:tasksingles     0.24291    0.05369   4.524 6.07e-06 ***
## pas4:tasksingles     0.56339    0.05101  11.044  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Correlation of Fixed Effects:
##             (Intr) pas2   pas3   pas4   tskqdr tsksng ps2:tskq ps3:tskq
## pas2        -0.742                                                     
## pas3        -0.829  0.613                                              
## pas4        -0.847  0.412  0.703                                       
## taskqudrplt -0.151  0.138  0.080  0.057                                
## tasksingles -0.138  0.126  0.073  0.052  0.484                         
## ps2:tskqdrp  0.102 -0.198 -0.054 -0.039 -0.679 -0.328                  
## ps3:tskqdrp  0.089 -0.082 -0.125 -0.034 -0.592 -0.286  0.402           
## ps4:tskqdrp  0.090 -0.083 -0.048 -0.093 -0.598 -0.289  0.406    0.354  
## ps2:tsksngl  0.098 -0.189 -0.052 -0.037 -0.342 -0.708  0.490    0.203  
## ps3:tsksngl  0.088 -0.081 -0.124 -0.033 -0.308 -0.637  0.209    0.486  
## ps4:tsksngl  0.092 -0.085 -0.049 -0.091 -0.324 -0.670  0.220    0.192  
##             ps4:tskq ps2:tsks ps3:tsks
## pas2                                  
## pas3                                  
## pas4                                  
## taskqudrplt
## tasksingles
## ps2:tskqdrp
## ps3:tskqdrp
## ps4:tskqdrp
## ps2:tsksngl  0.205
## ps3:tsksngl  0.184    0.451
## ps4:tsksngl  0.507    0.474    0.427
## optimizer (Nelder_Mead) convergence code: 0 (OK)
## Model failed to converge with max|grad| = 0.00343245 (tol = 0.002, component 1)
```

```
i. which family should be used?
```

Poisson which is basically a binomial logisitc regression where the boundaries approaches/goes to 0.

```
ii. why is a slope for _pas_ not really being modelled?
```

Because pas i being treated as a factor we’re not really modelling a slope but rather the differences between the levels of pas **compared to the first level.**

```
iii. if you get a convergence error, try another algorithm (the default is the _Nelde
r_Mead_) - try (_bobyqa_) for which the `dfoptim` package is needed. In `glmer`, you
can add the following for the `control` argument: `glmerControl(optimizer="bobyqa")`
(if you are interested, also have a look at the function `allFit`)
```

```
pacman::p_load(dfoptim)
```

Poison modeling

```
m3.2 <- glmer(count ~ pas*task + (1+pas|subject), family = poisson(link = "log"),cont
rol = glmerControl(optimizer="bobyqa"),data = df_count)
summary(m3.2)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: count ~ pas * task + (1 + pas | subject)
##    Data: df_count
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##   3148.4   3232.7  -1552.2   3104.4      318
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.3871 -0.7853 -0.0469  0.7550  6.5438
##
## Random effects:
##  Groups  Name        Variance Std.Dev. Corr
##  subject (Intercept) 0.3324   0.5765
##          pas2        0.3803   0.6167   -0.75
##          pas3        1.1960   1.0936   -0.84  0.63
##          pas4        2.3736   1.5407   -0.86  0.42  0.72
## Number of obs: 340, groups:  subject, 29
##
## Fixed effects:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)            4.03570    0.10976  36.769  < 2e-16 ***
## pas2                  -0.02378    0.11963  -0.199 0.842458
## pas3                  -0.51365    0.20718  -2.479 0.013166 *
## pas4                  -0.77292    0.29075  -2.658 0.007853 **
## taskquadruplet         0.11490    0.03127   3.674 0.000239 ***
## tasksingles           -0.23095    0.03418  -6.756 1.42e-11 ***
## pas2:taskquadruplet -0.11375    0.04605  -2.470 0.013508 *
## pas3:taskquadruplet -0.20901    0.05287  -3.954 7.70e-05 ***
## pas4:taskquadruplet -0.21500    0.05230  -4.111 3.94e-05 ***
## pas2:tasksingles      0.19536    0.04830   4.045 5.23e-05 ***
## pas3:tasksingles      0.24299    0.05369   4.526 6.02e-06 ***
## pas4:tasksingles      0.56346    0.05101  11.045  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) pas2   pas3   pas4   tskqdr tsksng ps2:tskq ps3:tskq
## pas2        -0.742
## pas3        -0.829  0.613
## pas4        -0.847  0.412  0.703
## taskqudrplt -0.151  0.138  0.080  0.057
## tasksingles -0.138  0.126  0.073  0.052  0.484
## ps2:tskqdrp  0.102 -0.198 -0.054 -0.039 -0.679 -0.328
## ps3:tskqdrp  0.089 -0.082 -0.125 -0.034 -0.592 -0.286  0.402
## ps4:tskqdrp  0.090 -0.083 -0.048 -0.093 -0.598 -0.289  0.406    0.354
## ps2:tsksngl  0.098 -0.188 -0.052 -0.037 -0.342 -0.708  0.490    0.203
## ps3:tsksngl  0.088 -0.080 -0.124 -0.033 -0.308 -0.637  0.209    0.486
## ps4:tsksngl  0.092 -0.085 -0.049 -0.091 -0.324 -0.670  0.220    0.192
##             ps4:tskq ps2:tsks ps3:tsks
## pas2
## pas3
## pas4
## taskqudrplt
## tasksingles
## ps2:tskqdrp
## ps3:tskqdrp
## ps4:tskqdrp
## ps2:tsksngl  0.205
## ps3:tsksngl  0.184    0.451
## ps4:tsksngl  0.507    0.474    0.427
```

```
iv. when you have a converging fit - fit a model with only the main effects of _pas_
and _task_. Compare this with the model that also includes the interaction
```

```
m3.3 <- glmer(count ~ pas+task + (1+pas|subject), family = poisson(link = "log"),cont
rol = glmerControl(optimizer="bobyqa"),data = df_count)
summary(m3.3)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: count ~ pas + task + (1 + pas | subject)
##    Data: df_count
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##   3398.5   3459.8  -1683.3   3366.5      324
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.5885 -0.9001 -0.0477  0.8253  6.5100
##
## Random effects:
##  Groups  Name        Variance Std.Dev. Corr
##  subject (Intercept) 0.3325   0.5766
##          pas2        0.3805   0.6169   -0.75
##          pas3        1.1895   1.0906   -0.84  0.63
##          pas4        2.4222   1.5563   -0.86  0.42  0.73
## Number of obs: 340, groups:  subject, 29
##
## Fixed effects:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     4.004589   0.108724  36.833   <2e-16 ***
## pas2           -0.006528   0.116616  -0.056   0.9554
## pas3           -0.509918   0.204452  -2.494   0.0126 *
## pas4           -0.663832   0.291962  -2.274   0.0230 *
## taskquadruplet  0.003294   0.018188   0.181   0.8563
## tasksingles     0.004307   0.018177   0.237   0.8127
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) pas2   pas3   pas4   tskqdr
## pas2        -0.742
## pas3        -0.832  0.623
## pas4        -0.850  0.412  0.723
## taskqudrplt -0.084  0.000  0.001 -0.002
## tasksingles -0.084  0.000  0.000  0.000  0.501
```

```
v. indicate which of the two models, you would choose and why
```

First we will add another column containing the total number of counts in each grouping of participant and task. The log() verison of that column will work as our **offset** variable in future models.

```
df_count <- df_count %>%
  group_by(task,subject) %>%
  mutate(total_count = sum(count))
```

Modeling with offset

Poission regression can either model count or rate data. So far we have modelled count data. But our count data is a fraction of larger grouping or time interval. In our case the frequency will be estimated within the grouping of task and subject. Subject 1 & task pairs has 170 data points where 4 of them is in pas = 4. The frequency is therefore 4/170.

Disclaimer I am not quite sure whether it is best practise to divide by subject in the grouping or if should only be done by task as we account for indivudial variance with the random effect of subject???

```
m3.4 <- glmer(count ~ pas*task + (1+pas|subject), family = poisson(link = "log"),cont
rol = glmerControl(optimizer="bobyqa"),data = df_count, offset = log(total_count))
summary(m3.4)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: count ~ pas * task + (1 + pas | subject)
##    Data: df_count
##  Offset: log(total_count)
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##   3132.5   3216.7  -1544.2   3088.5      318
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.3832 -0.7914 -0.0361  0.8026  6.4078
##
## Random effects:
##  Groups  Name        Variance Std.Dev. Corr
##  subject (Intercept) 0.3017   0.5493
##          pas2        0.3786   0.6153   -0.72
##          pas3        1.1754   1.0842   -0.86  0.62
##          pas4        2.2269   1.4923   -0.90  0.43  0.72
## Number of obs: 340, groups:  subject, 29
##
## Fixed effects:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -1.29022    0.10482 -12.309  < 2e-16 ***
## pas2                -0.02788    0.11938  -0.234 0.815350
## pas3                -0.51782    0.20545  -2.520 0.011722 *
## pas4                -0.75192    0.28146  -2.672 0.007551 **
## taskquadruplet       0.10589    0.03128   3.385 0.000711 ***
## tasksingles         -0.23847    0.03419  -6.975 3.06e-12 ***
## pas2:taskquadruplet -0.11152    0.04606  -2.421 0.015468 *
## pas3:taskquadruplet -0.20335    0.05287  -3.846 0.000120 ***
## pas4:taskquadruplet -0.20713    0.05231  -3.960 7.51e-05 ***
## pas2:tasksingles     0.19916    0.04831   4.123 3.74e-05 ***
## pas3:tasksingles     0.24851    0.05370   4.628 3.70e-06 ***
## pas4:tasksingles     0.56857    0.05102  11.143  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) pas2   pas3   pas4   tskqdr tsksng ps2:tskq ps3:tskq
## pas2        -0.714
## pas3        -0.847  0.610
## pas4        -0.880  0.419  0.704
## taskqudrplt -0.158  0.138  0.080  0.059
## tasksingles -0.144  0.127  0.074  0.054  0.484
## ps2:tskqdrp  0.107 -0.198 -0.055 -0.040 -0.679 -0.328
## ps3:tskqdrp  0.093 -0.082 -0.126 -0.035 -0.592 -0.286  0.402
## ps4:tskqdrp  0.094 -0.083 -0.048 -0.097 -0.598 -0.289  0.406    0.354
## ps2:tsksngl  0.102 -0.189 -0.052 -0.038 -0.342 -0.708  0.490    0.203
## ps3:tsksngl  0.092 -0.081 -0.125 -0.034 -0.308 -0.637  0.209    0.486
## ps4:tsksngl  0.097 -0.085 -0.049 -0.094 -0.324 -0.670  0.220    0.192
##             ps4:tskq ps2:tsks ps3:tsks
## pas2
## pas3
## pas4
## taskqudrplt
## tasksingles
## ps2:tskqdrp
## ps3:tskqdrp
## ps4:tskqdrp
## ps2:tsksngl  0.205
## ps3:tsksngl  0.184    0.451
## ps4:tsksngl  0.507    0.474    0.427
```

```
m3.5 <- glmer(count ~ pas+task + (1+pas|subject), family = poisson(link = "log"),cont
rol = glmerControl(optimizer="bobyqa"),data = df_count, offset = log(total_count))
summary(m3.5)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: count ~ pas + task + (1 + pas | subject)
##    Data: df_count
##  Offset: log(total_count)
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##   3381.9   3443.1  -1674.9   3349.9      324
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.5004 -0.9089 -0.0148  0.8785  6.2895
##
## Random effects:
##  Groups  Name        Variance Std.Dev. Corr
##  subject (Intercept) 0.3020   0.5495
##          pas2        0.3789   0.6155   -0.72
##          pas3        1.1695   1.0814   -0.86  0.63
##          pas4        2.2744   1.5081   -0.90  0.42  0.73
## Number of obs: 340, groups:  subject, 29
##
## Fixed effects:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.324e+00  1.038e-01 -12.754   <2e-16 ***
## pas2           -8.617e-03  1.164e-01  -0.074   0.9410
## pas3           -5.104e-01  2.027e-01  -2.517   0.0118 *
## pas4           -6.387e-01  2.827e-01  -2.260   0.0238 *
## taskquadruplet -2.290e-03  1.819e-02  -0.126   0.8998
## tasksingles     3.191e-06  1.818e-02   0.000   0.9999
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) pas2   pas3   pas4   tskqdr
## pas2        -0.712
## pas3        -0.850  0.621
## pas4        -0.884  0.419  0.724
## taskqudrplt -0.088  0.000  0.001 -0.002
## tasksingles -0.088  0.000  0.000  0.000  0.501
```

Comparing interaction and offset models

```
#check R^2
MuMIn::r.squaredGLMM(m3.2)
```

```
##                 R2m       R2c
## delta     0.1577062 0.9750530
## lognormal 0.1577375 0.9752462
## trigamma  0.1576745 0.9748567
```

```
MuMIn::r.squaredGLMM(m3.3)
```

```
##                 R2m       R2c
## delta     0.1375957 0.9747420
## lognormal 0.1376233 0.9749376
## trigamma  0.1375677 0.9745434
```

```
MuMIn::r.squaredGLMM(m3.4)
```

```
##                  R2m        R2c
## delta     0.02414973 0.14650923
## lognormal 0.04667814 0.28318238
## trigamma  0.00720645 0.04371939
```

```
MuMIn::r.squaredGLMM(m3.5)
```

```
##                   R2m        R2c
## delta     0.020714560 0.14477964
## lognormal 0.040114255 0.28036933
## trigamma  0.006172595 0.04314193
```

```
#check with anova
anova(m3.2,m3.3,m3.4,m3.5)
```

```
## Data: df_count
## Models:
## m3.3: count ~ pas + task + (1 + pas | subject)
## m3.5: count ~ pas + task + (1 + pas | subject)
## m3.2: count ~ pas * task + (1 + pas | subject)
## m3.4: count ~ pas * task + (1 + pas | subject)
##      npar    AIC    BIC  logLik deviance   Chisq Df Pr(>Chisq)
## m3.3   16 3398.5 3459.8 -1683.3   3366.5
## m3.5   16 3381.9 3443.1 -1674.9   3349.9  16.683  0
## m3.2   22 3148.4 3232.7 -1552.2   3104.4 245.425  6  < 2.2e-16 ***
## m3.4   22 3132.5 3216.7 -1544.2   3088.5  15.945  0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The interaction effects are significant in both our model with and without offset. As showed by our culminated R2 the interaction effect also adds a tiny bit of explained variance while also performing better following AIC and BIC which punishes unceccesary model complexity.

The inclusion of an offset variable is theoretical warranted when working with frequencies in poission regression as argued by Gelman and Hill in (Data Analysis Using Regression and Multilevel/Hierarchical Models, 2007). The offset variable also shows a marginal improvement in AIC/BIC but with a enormous reduction in culminated R2.

My model of choice will be m3.4 as it contains the needed interaction effect and offset variable which also performs the best following the AIC and BIC.

```
mean(df_count$count)
```

```
## [1] 53.32647
```

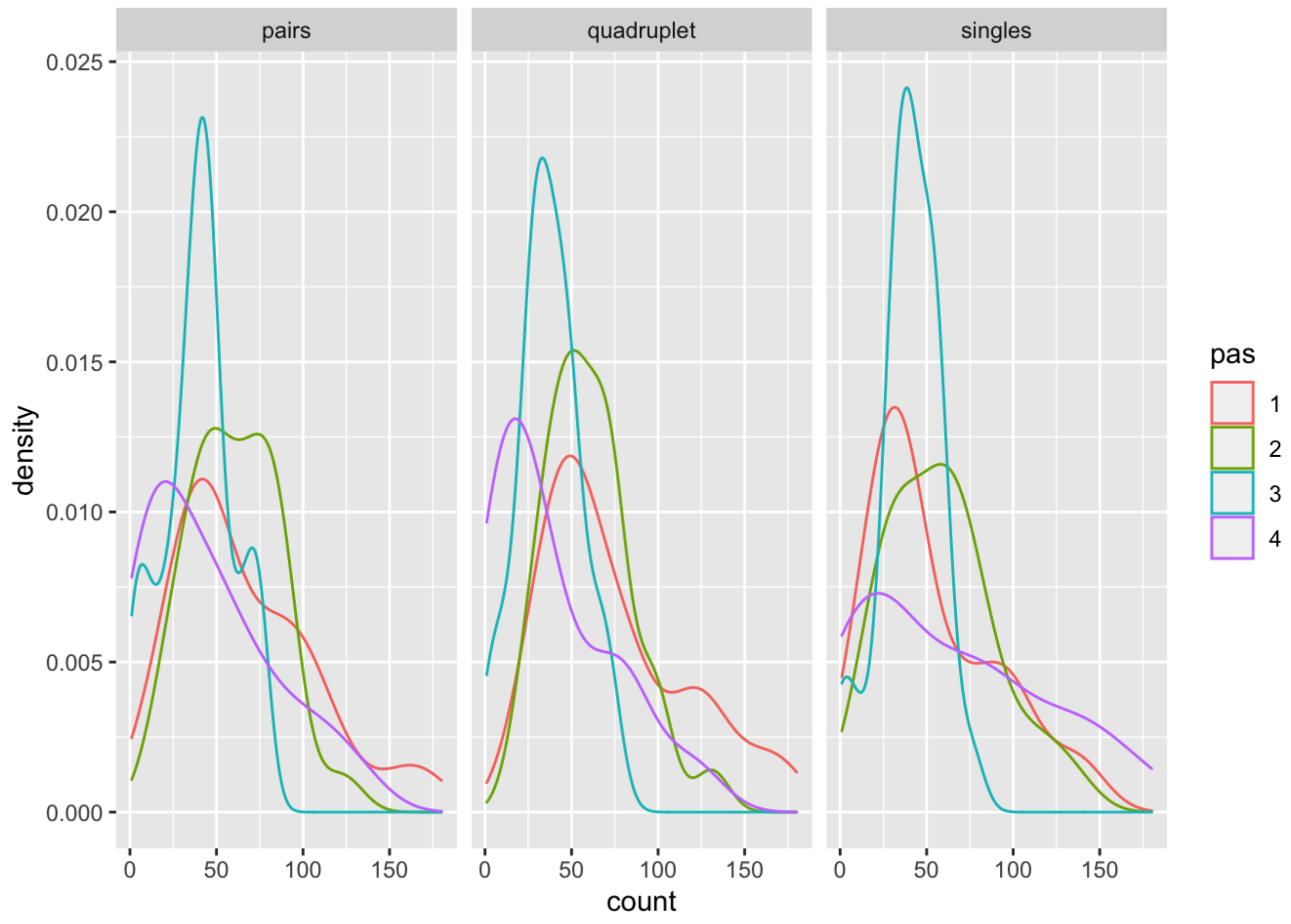
```
var(df_count$count)
```

```
## [1] 1214.002
```

as $mean var $ this is an indication of that there will be overdispersion in all our models. But as non of my known dispersiontest (If the Residual Deviance is greater than the degrees of freedom, then over-dispersion exists) and quasipoission distributions work on mixed effect modelling I don’t know what can be done about it. Ideally something would be done like a quasipoission regression.

```
vi. based on your chosen model - write a short report on what this says about the dis
tribution of ratings as dependent on _pas_ and _task_
```

```
ggplot(df_count, aes(col = pas, x = count)) + geom_density() + facet_wrap(~task)
```



```
exp(fixef(m3.4))
```

```
##         (Intercept)                pas2                pas3                pas4 
##           0.2752108           0.9725073           0.5958187           0.4714597 
##      taskquadruplet         tasksingles pas2:taskquadruplet pas3:taskquadruplet 
##           1.1117039           0.7878326           0.8944698           0.8159922 
## pas4:taskquadruplet    pas2:tasksingles    pas3:tasksingles    pas4:tasksingles 
##           0.8129117           1.2203738           1.2821197           1.7657324 
```

- exp(α)= effect on the mean μ, when X = 0
- exp(β) = with every unit increase in X, the predictor variable has multiplicative effect of exp(β) on the mean of Y, that is μ
- If β = 0, then exp(β) = 1, and the expected count is exp(α) and, Y and X are not related.
- If β > 0, then exp(β) > 1, and the expected count is exp(β) times larger than when X = 0
- If β < 0, then exp(β) < 1, and the expected count is exp(β) times smaller than when X = 0

Pas fix

Pas2 will reduce the frequency of count by *(1-0.9725)* percentage compared to pas1. As shown by the exp(fixef(m3.4)) pas3 and pas4 will result in an even more drastic reduction compared to pas1.

Task fix

Quadruplet will increase count frequency with *(1-1.11)* percentage compared to task_pairs. Though task_singles decrease the frequency with *(1-0.78)* percentage compared to task_pairs.

Interaction effect

The interaction effects show that task_singles interaction with pas level 2:4 increase our frequency. While the interaction between task_quadruplet and pas level 2:4 will significantly decrease our frequency. In a percentage wise interpretation the effect of some of the interactions are even larger than the fixed effects alone. This is seems kinda odd as adding the interaction effect didn't improve the models with a big margin.

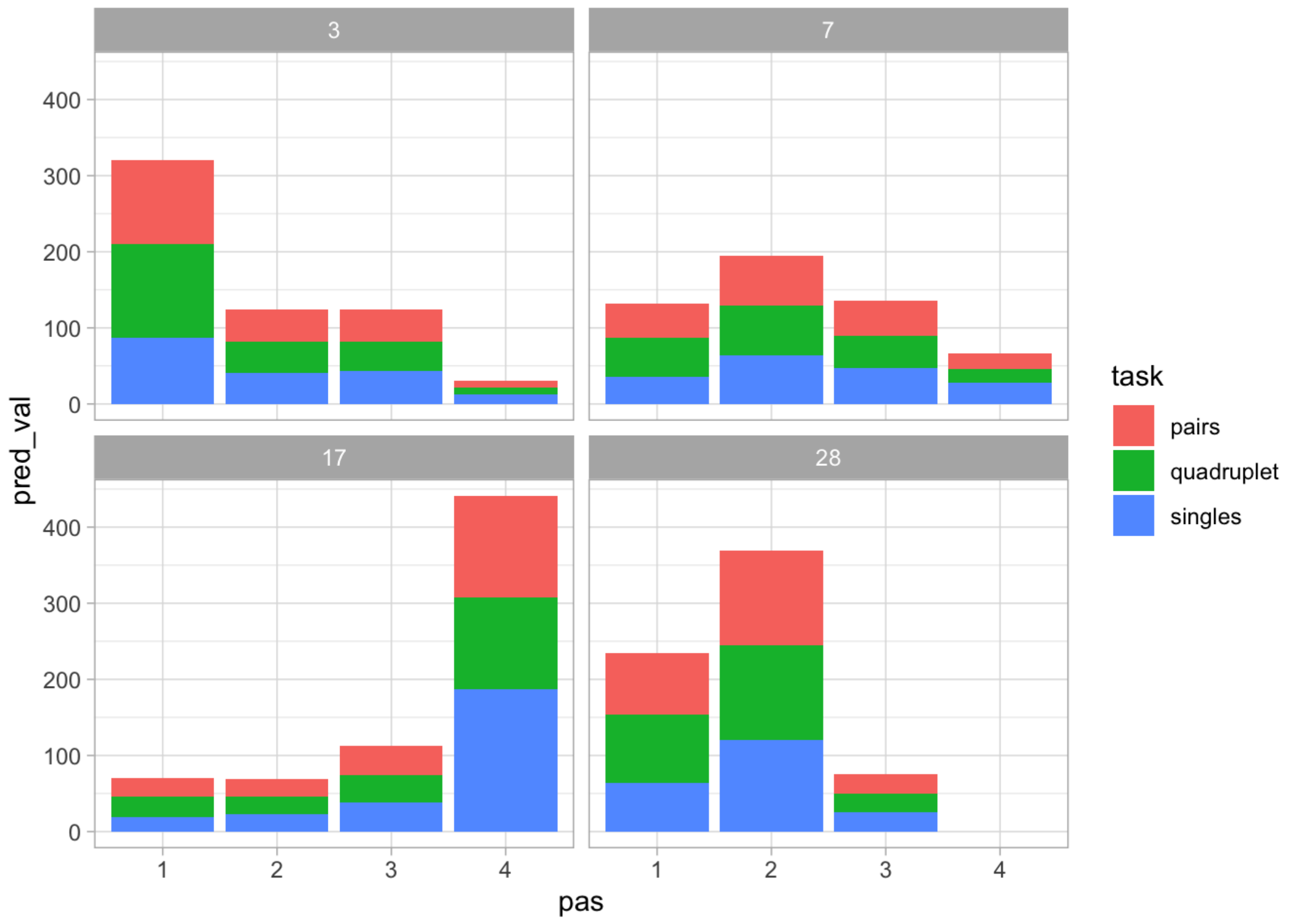
```
vii. include a plot that shows the estimated amount of ratings for four subjects of y
our choosing
```

As our current selected model is a model of frequency it wouldn't be fitting for plotting "amount". There the similair count model m3.2 without the offset is selected for this assigment.

```
df_count_fil <- df_count %>%
  filter(subject == '3' | subject == '7' | subject == '17' | subject == '28')


df_count_fil_pred <- cbind(df_count_fil, pred_val = exp(predict(m3.2, newdata = df_co
unt_fil)))

ggplot(df_count_fil_pred, aes(x = pas, y = pred_val, fill = task)) +
  geom_bar(stat = 'identity') +
  facet_wrap(~ subject) +
  theme_light()
```



3. Finally, fit a multilevel model that models *correct* as dependent on *task* with a unique intercept for each *subject* Two options:
4. I could calculate the amount of corrects and do a poission regression on the count or frequency of correct responses.

```
#Something like this where I also add some of
df_exp_count_correct <- df_exp %>%
  group_by(subject,task,pas) %>%
  summarise(n_correct = sum(right_answer == 1))
```

```
## `summarise()` has grouped output by 'subject', 'task'. You can override using the
`.groups` argument.
```

```
df_exp_count_correct <- df_exp_count_correct %>%
  group_by(subject) %>%
  mutate(n_answers = sum(n_correct)) %>%
  mutate(pas = as.factor(pas))
```

```
model1_pois <- glmer(n_correct ~ task + (1|subject), data = df_exp_count_correct,
              offset = log(n_answers), family = poisson(link = "log"))
summary(model1_pois)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: n_correct ~ task + (1 | subject)
##    Data: df_exp_count_correct
##  Offset: log(n_answers)
## 
##      AIC      BIC   logLik deviance df.resid 
##   7720.9   7736.2  -3856.4   7712.9      336 
## 
## Scaled residuals: 
##     Min      1Q  Median      3Q     Max 
## -6.8366 -2.7035 -0.8973  2.2792 17.5566 
## 
## Random effects:
##  Groups  Name        Variance Std.Dev.
##  subject (Intercept) 0.003766 0.06137 
## Number of obs: 340, groups:  subject, 29
## 
## Fixed effects:
##                Estimate Std. Error  z value Pr(>|z|)    
## (Intercept)    -2.46639    0.01882 -131.033   <2e-16 ***
## taskquadruplet -0.02958    0.02122   -1.394   0.1633    
## tasksingles     0.04724    0.02087    2.264   0.0236 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Correlation of Fixed Effects:
##             (Intr) tskqdr
## taskqudrplt -0.558       
## tasksingles -0.567  0.503
```

```
exp(fixef(model1_pois))
```

```
##    (Intercept) taskquadruplet    tasksingles 
##     0.08489079     0.97085114     1.04837246
```

```
MuMIn::r.squaredGLMM(model1_pois)
```

```
## Warning: The null model is correct only if all variables used by the original
## model remain unchanged.
```

```
##                    R2m          R2c
## delta     8.594200e-05 4.081108e-04
## lognormal 3.947110e-04 1.874355e-03
## trigamma  7.280081e-06 3.457075e-05
```

2. I could just do a normal binomial regression trying to predict correct.

```
model1_binom <- glmer(right_answer ~ task + (1|subject), data = df_exp,
                      family = binomial(link = "logit"))
summary(model1_binom)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: right_answer ~ task + (1 | subject)
##    Data: df_exp
## 
##      AIC      BIC   logLik deviance df.resid 
##  19927.2  19958.4  -9959.6  19919.2    18127 
## 
## Scaled residuals: 
##     Min      1Q  Median      3Q     Max 
## -2.7426 -1.0976  0.5098  0.6101  0.9111 
## 
## Random effects:
##  Groups  Name        Variance Std.Dev.
##  subject (Intercept) 0.1775   0.4214  
## Number of obs: 18131, groups:  subject, 29
## 
## Fixed effects:
##                Estimate Std. Error z value Pr(>|z|)    
## (Intercept)     1.10071    0.08387  13.124  < 2e-16 ***
## taskquadruplet -0.09825    0.04190  -2.345    0.019 *  
## tasksingles     0.18542    0.04336   4.276  1.9e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Correlation of Fixed Effects:
##             (Intr) tskqdr
## taskqudrplt -0.256       
## tasksingles -0.247  0.495
```

```
invlogit(fixef(model1_binom))
```

```
##    (Intercept) taskquadruplet    tasksingles 
##      0.7503932      0.4754570      0.5462226
```

```
MuMIn::r.squaredGLMM(model1_binom)
```

```
## Warning: The null model is correct only if all variables used by the original
## model remain unchanged.
```

```
##                     R2m        R2c
## theoretical 0.003979585 0.05497746
## delta       0.002526101 0.03489776
```

```
i. does _task_ explain performance?
```

Both models shows task significantly predicts right_answer. Binomial models shows a rather lower R2c = 0.055 but the poisson model is attrosious.

```
ii. add _pas_ as a main effect on top of _task_ - what are the consequences of that?
```

Due to my time and your reading time I will only continue using the binomial approach. ;)

```
model2_binom <- glmer(right_answer ~ task + pas + (1|subject), data = df_exp,
                      family = binomial(link = "logit"))
summary(model2_binom)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: right_answer ~ task + pas + (1 | subject)
##    Data: df_exp
## 
##      AIC      BIC   logLik deviance df.resid 
##  17425.0  17464.0  -8707.5  17415.0    18126 
## 
## Scaled residuals: 
##     Min      1Q  Median      3Q     Max 
## -8.1096 -0.6101  0.3181  0.5653  1.6476 
## 
## Random effects:
##  Groups  Name        Variance Std.Dev.
##  subject (Intercept) 0.2004   0.4477  
## Number of obs: 18131, groups:  subject, 29
## 
## Fixed effects:
##                 Estimate Std. Error z value Pr(>|z|)    
## (Intercept)    -0.950104   0.098399  -9.656   <2e-16 ***
## taskquadruplet -0.029418   0.045016  -0.653    0.513    
## tasksingles    -0.008914   0.046889  -0.190    0.849    
## pas             1.014031   0.022900  44.281   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Correlation of Fixed Effects:
##             (Intr) tskqdr tsksng
## taskqudrplt -0.247              
## tasksingles -0.189  0.489       
## pas         -0.421  0.030 -0.083
```

```
invlogit(fixef(model2_binom))
```

```
##    (Intercept) taskquadruplet    tasksingles            pas 
##      0.2788640      0.4926460      0.4977716      0.7338084
```

```
MuMIn::r.squaredGLMM(model2_binom)
```

```
## Warning: The null model is correct only if all variables used by the original
## model remain unchanged.
```

```
##                    R2m       R2c
## theoretical 0.2728668 0.3146160
## delta       0.1925254 0.2219821
```

R2c has gotten a boost up to 0.31 from 0.055. The effect size of task_quadruplet and task_singles has almost remained the same but the is now no longer signifcant.

```
anova(model1_binom, model2_binom)
```

```
## Data: df_exp
## Models:
## model1_binom: right_answer ~ task + (1 | subject)
## model2_binom: right_answer ~ task + pas + (1 | subject)
##              npar   AIC   BIC  logLik deviance  Chisq Df Pr(>Chisq)
## model1_binom    4 19927 19958 -9959.6    19919
## model2_binom    5 17425 17464 -8707.5    17415 2504.2  1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model 2 appears to be the best model.

```
iii. now fit a multilevel model that models _correct_ as dependent on _pas_ with a un
ique intercept for each _subject_
```

```
model3_binom <- glmer(right_answer ~ pas + (1|subject), data = df_exp,
                      family = binomial(link = "logit"))
```

```
iv. finally, fit a model that models the interaction between _task_ and _pas_  and th
eir main effects
```

```
model4_binom <- glmer(right_answer ~ pas*task + (1|subject), data = df_exp,
                      family = binomial(link = "logit"))
```

```
v. describe in your words which model is the best in explaining the variance in accur
acy
```

```
anova(model1_binom,model2_binom,model3_binom,model4_binom)
```

```
## Data: df_exp
## Models:
## model3_binom: right_answer ~ pas + (1 | subject)
## model1_binom: right_answer ~ task + (1 | subject)
## model2_binom: right_answer ~ task + pas + (1 | subject)
## model4_binom: right_answer ~ pas * task + (1 | subject)
##              npar   AIC   BIC  logLik deviance     Chisq Df Pr(>Chisq)
## model3_binom    3 17422 17445 -8707.7    17416
## model1_binom    4 19927 19958 -9959.6    19919    0.0000  1    1.00000
## model2_binom    5 17425 17464 -8707.5    17415 2504.2020  1    < 2e-16 ***
## model4_binom    7 17423 17478 -8704.4    17409    6.1866  2    0.04535 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model 1 has a significantly higher BIC and AIC than the rest. We can therefore conclude including pas as predictor of right_answer is key for explaining the variance in the data.

Currently the choice is between model2, model3, model4. So I suggest we do a cross-validation test to test the generalisability of the models.

Cross-Validation

```
pacman::p_load(caret)
#sample
rand_sample <- createDataPartition(df_exp $ right_answer, p = 0.8, list = FALSE)
#training
train_df <- df_exp[rand_sample,]
#test
test_df <- df_exp[-rand_sample,]
```

```
#Train on train_df
cv_model2 <- glmer(right_answer ~ task + pas + (1|subject), data = train_df,
                      family = binomial(link = "logit"))
cv_model3 <- glmer(right_answer ~ pas + (1|subject), data = train_df,
                      family = binomial(link = "logit"))
cv_model4 <- glmer(right_answer ~ pas*task + (1|subject), data = train_df,
                      family = binomial(link = "logit"))
```

```
#Predict
predictions2 <- predict(cv_model2, test_df, type = "response")
predictions3 <- predict(cv_model3, test_df, type = "response")
predictions4 <- predict(cv_model4, test_df, type = "response")
#Turn into binary response
prediction_bin2 <- if_else(predictions2 > 0.5, 1, 0)
prediction_bin3 <- if_else(predictions3 > 0.5, 1, 0)
prediction_bin4 <- if_else(predictions4 > 0.5, 1, 0)

#append predicted values to test_df
test_df <- test_df %>%
  mutate(pred_val2 = as.factor(prediction_bin2), pred_val3 = as.factor(prediction_bin
3), pred_val4 = as.factor(prediction_bin4))
```

```
#Confusion Matrix for model2
confusionMatrix(data=test_df$pred_val2, reference = test_df$right_answer)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    0    1
##          0  194  208
##          1  715 2509
##                                           
##                Accuracy : 0.7454          
##                  95% CI : (0.7309, 0.7596)
##     No Information Rate : 0.7493          
##     P-Value [Acc > NIR] : 0.7115          
##                                           
##                   Kappa : 0.1681          
##                                           
##  Mcnemar's Test P-Value : <2e-16          
##                                           
##             Sensitivity : 0.2134          
##             Specificity : 0.9234          
##          Pos Pred Value : 0.4826          
##          Neg Pred Value : 0.7782          
##              Prevalence : 0.2507          
##          Detection Rate : 0.0535          
##    Detection Prevalence : 0.1109          
##       Balanced Accuracy : 0.5684          
##                                           
##        'Positive' Class : 0               
## 
```

```
#Confusion Matrix for model3
confusionMatrix(data=test_df$pred_val3, reference = test_df$right_answer)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    0    1
##          0  194  209
##          1  715 2508
##                                           
##                Accuracy : 0.7452          
##                  95% CI : (0.7307, 0.7593)
##     No Information Rate : 0.7493          
##     P-Value [Acc > NIR] : 0.7244          
##                                           
##                   Kappa : 0.1675          
##                                           
##  Mcnemar's Test P-Value : <2e-16          
##                                           
##             Sensitivity : 0.2134          
##             Specificity : 0.9231          
##          Pos Pred Value : 0.4814          
##          Neg Pred Value : 0.7782          
##              Prevalence : 0.2507          
##          Detection Rate : 0.0535          
##    Detection Prevalence : 0.1111          
##       Balanced Accuracy : 0.5682          
##                                           
##        'Positive' Class : 0               
## 
```

```
#Confusion Matrix for model4
confusionMatrix(data=test_df$pred_val4, reference = test_df$right_answer)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  183  200
##          1  726 2517
##
##                Accuracy : 0.7446
##                  95% CI : (0.7301, 0.7588)
##     No Information Rate : 0.7493
##     P-Value [Acc > NIR] : 0.7493
##
##                   Kappa : 0.1582
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.20132
##             Specificity : 0.92639
##          Pos Pred Value : 0.47781
##          Neg Pred Value : 0.77613
##              Prevalence : 0.25069
##          Detection Rate : 0.05047
##    Detection Prevalence : 0.10563
##       Balanced Accuracy : 0.56385
##
##        'Positive' Class : 0
##
```

sum up As all models have exactly the same accuracy we could continue with doing k-fold cross-validation. But for now I'll just argue that we care more about true-negative rate. Correctly classifying a person with a tumor is not fun, but it would be even worse to miss diagnose someone as a false-negative which could then result in death due to lack of treatment. I know the analogy doesn't quite fit in this experiment but I need some way to choose. ;)

model4 shows the best true-negative-rate and will therefore be chosen as the final model.