Data analysis pilots

Nathanael Larigaldie

2023-07-24

Contents

Dataset loading

```
datasets       <- list.files(path="./final/", pattern="*.csv", full.names = TRUE)
datasets       <- lapply(datasets, read_csv, show_col_types = FALSE)

d <- datasets[[1]]
for (i in seq(from=2, to=length(datasets)))
{
  d <- rbind(d, datasets[[i]])
}
```

Dataset preparation

First, we need to get participant numbers. 2 participants only had 70 stimuli instead of 100 in the pilot study, so we can't just make a repetition of 100 stimuli.

```
d <- d %>% mutate(Subject=c(rep(1, 70), rep(2, 100), rep(3, 100),
                            rep(4, 70), rep(5, 100), rep(6, 100),
                            rep(7, 100), rep(8, 100)))
```

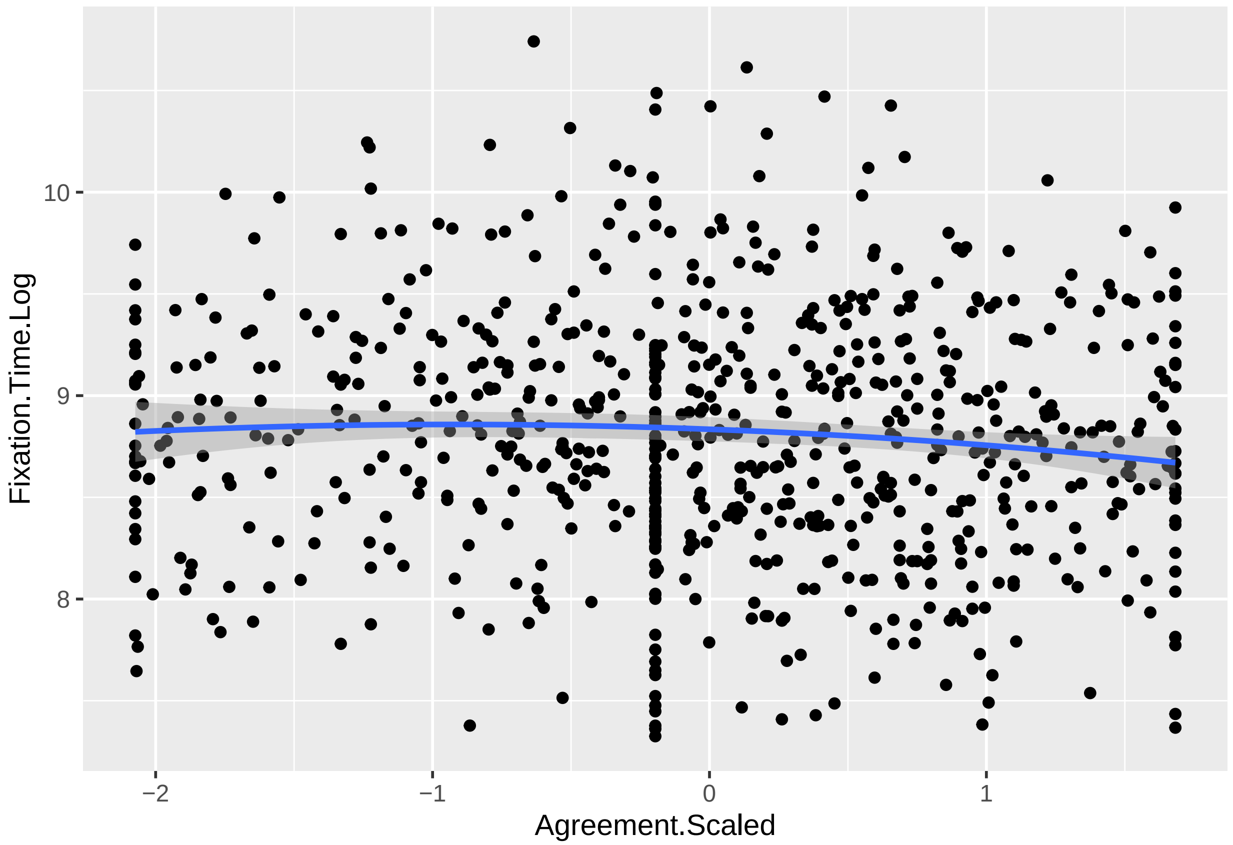
Then we remove fixation times <=1500ms (excludes 22 out of 740), scale the Agreement feature and take its square, and log-transform Fixation.Time

```
d_final <- d %>% filter(Fixation.Time > 1500) %>%
  # filter(New==1) %>%
  # filter(New==0) %>%
  mutate(Fixation.Time.Log = log(Fixation.Time),
         Agreement.Scaled = (Agreement-mean(Agreement))/sd(Agreement),
         Squared.Agreement.Scaled=Agreement.Scaled^2,
         Index=as.factor(Index), Subject=as.factor(Subject))
```

General plot and estimates (without random effects)

$$Fixation.Time = \alpha + \beta_1 Agreement + \beta_2 Agreement^2$$

```
ggplot(d_final, aes(x = Agreement.Scaled, y = Fixation.Time.Log)) +
  geom_point() + stat_smooth(method = "lm", formula = y ~ x + I(x^2))
```



```
res <- lm(Fixation.Time.Log ~ Agreement.Scaled + Squared.Agreement.Scaled,
          data=d_final)
anova(res)
```

```
## Analysis of Variance Table
##
## Response: Fixation.Time.Log
##                           Df  Sum Sq Mean Sq F value  Pr(>F)
## Agreement.Scaled           1   1.324 1.32419  3.5695 0.05925 .
## Squared.Agreement.Scaled   1   0.710 0.70980  1.9134 0.16702
## Residuals                715 265.243 0.37097
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coef(res)
```

```
##              (Intercept)         Agreement.Scaled Squared.Agreement.Scaled
##               8.83462577              -0.05105543              -0.02760266
```

Variables plot (violin & boxplot + density)

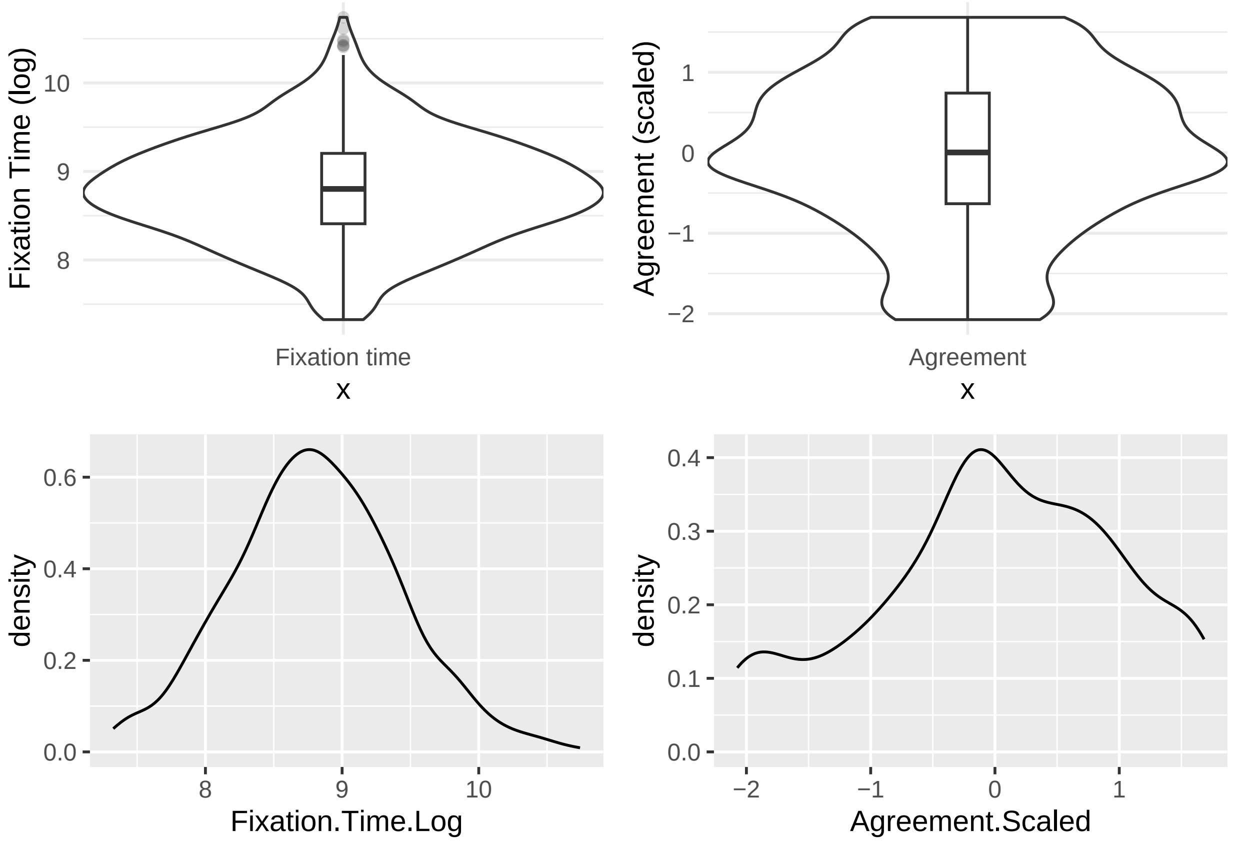
```
violin_fixation <- ggplot(d_final, aes(x="Fixation time", y=Fixation.Time.Log)) +
  ylab("Fixation Time (log)") + theme_minimal() +
  theme(legend.position = "none") +
  geom_violin(width=1.2) + geom_boxplot(width=0.1, alpha=0.2)


violin_agreement <- ggplot(d_final, aes(x="Agreement", y=Agreement.Scaled)) +
  ylab("Agreement (scaled)") + theme_minimal() +
  theme(legend.position = "none") +
  geom_violin(width=1.2) + geom_boxplot(width=0.1, alpha=0.2)

density_fixation <- ggplot(d_final, aes(x=Fixation.Time.Log)) + geom_density()

density_agreement <- ggplot(d_final, aes(x=Agreement.Scaled)) + geom_density()

grid.arrange(violin_fixation, violin_agreement, density_fixation,
             density_agreement, nrow=2)
```



Statistical model

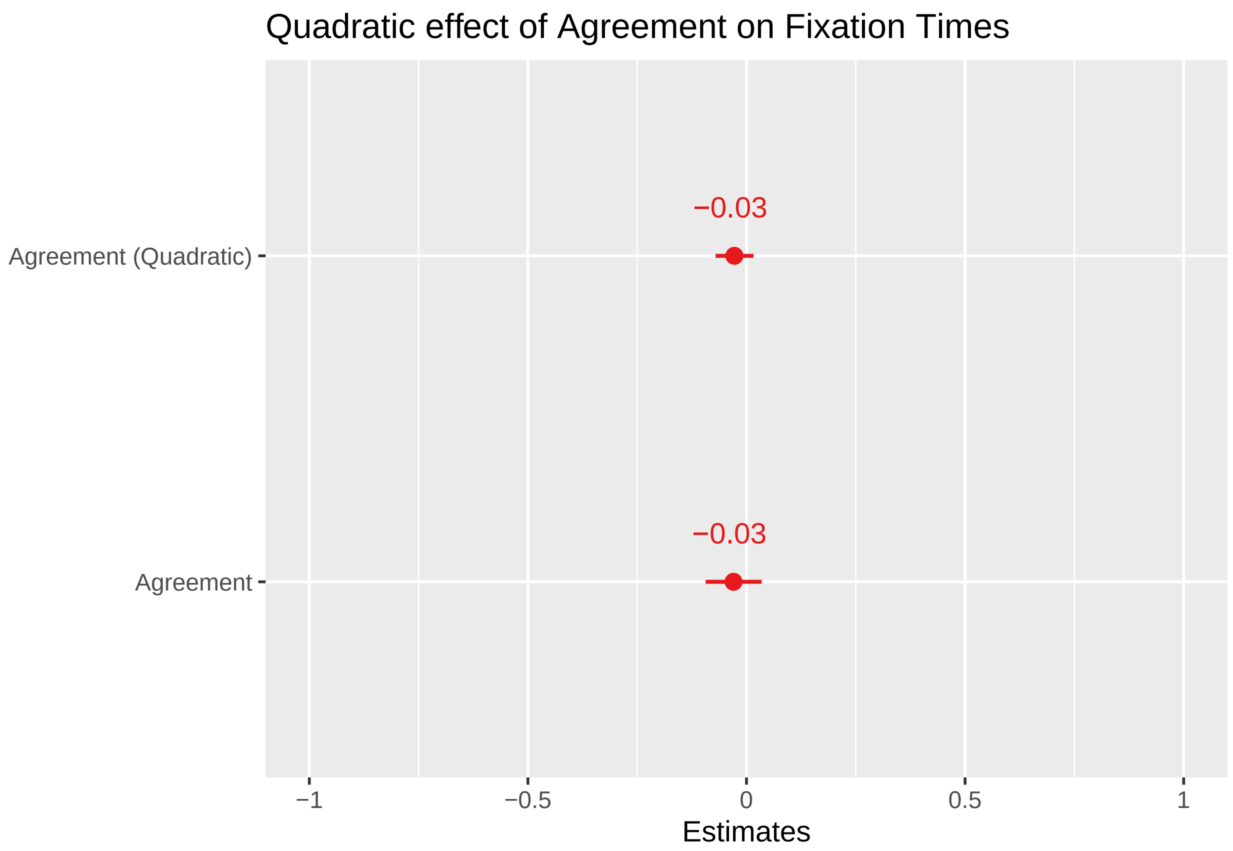
```
res_mixed <- lmer(Fixation.Time.Log ~ Agreement.Scaled +
                    Squared.Agreement.Scaled +
                    (1 | Index) +
                    (Agreement.Scaled + Squared.Agreement.Scaled | Subject),
                  data=d_final)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(res_mixed)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: Fixation.Time.Log ~ Agreement.Scaled + Squared.Agreement.Scaled +
##     (1 | Index) + (Agreement.Scaled + Squared.Agreement.Scaled |      Subject)
##    Data: d_final
##
## REML criterion at convergence: 828.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.7826 -0.6011 -0.0064  0.6106  3.2693
##
## Random effects:
##  Groups   Name                     Variance Std.Dev. Corr
##  Index    (Intercept)              0.021444 0.14644
##  Subject  (Intercept)              0.286592 0.53534
##           Agreement.Scaled         0.001519 0.03897  -0.99
##           Squared.Agreement.Scaled 0.006754 0.08218  -0.98  0.95
##  Residual                          0.157262 0.39656
## Number of obs: 718, groups:  Index, 100; Subject, 8
##
## Fixed effects:
##                          Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)               8.82038    0.19100  7.09903  46.179 4.59e-10 ***
## Agreement.Scaled         -0.02769    0.02172 12.32657  -1.275    0.226
## Squared.Agreement.Scaled -0.02962    0.03238  7.58461  -0.915    0.388
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Agrm.S
## Agrmnt.Scld -0.638
## Sqrd.Agrm.S -0.905  0.619
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
plot_model(res_mixed,
           axis.labels=c("Agreement", "Agreement (Quadratic)"),
           show.values=TRUE, show.p=TRUE,
           title="Quadratic effect of Agreement on Fixation Times")
```



```
tab_model(res_mixed,
          show.re.var=TRUE,
          pred.labels=c("(Intercept", "Agreement", "Agreement (Quadratic)"),
          dv.labels=c("Fixation Times"))
```

Fixation Times

Predictors

Estimates

CI

p

(Intercept

8.82

8.45 – 9.20

<0.001

Agreement

-0.03

-0.07 – 0.01

0.203

Agreement (Quadratic)

-0.03

-0.09 – 0.03

0.361

Random Effects

2

0.16

00 Index

0.02

00 Subject

0.29

11 Subject.Agreement.Scaled

0.00

11 Subject.Squared.Agreement.Scaled

0.01

01 Subject.Agreement.Scaled

-0.99

01 Subject.Squared.Agreement.Scaled

-0.98

ICC

0.60

N Index

100

N Subject

8

Observations

718

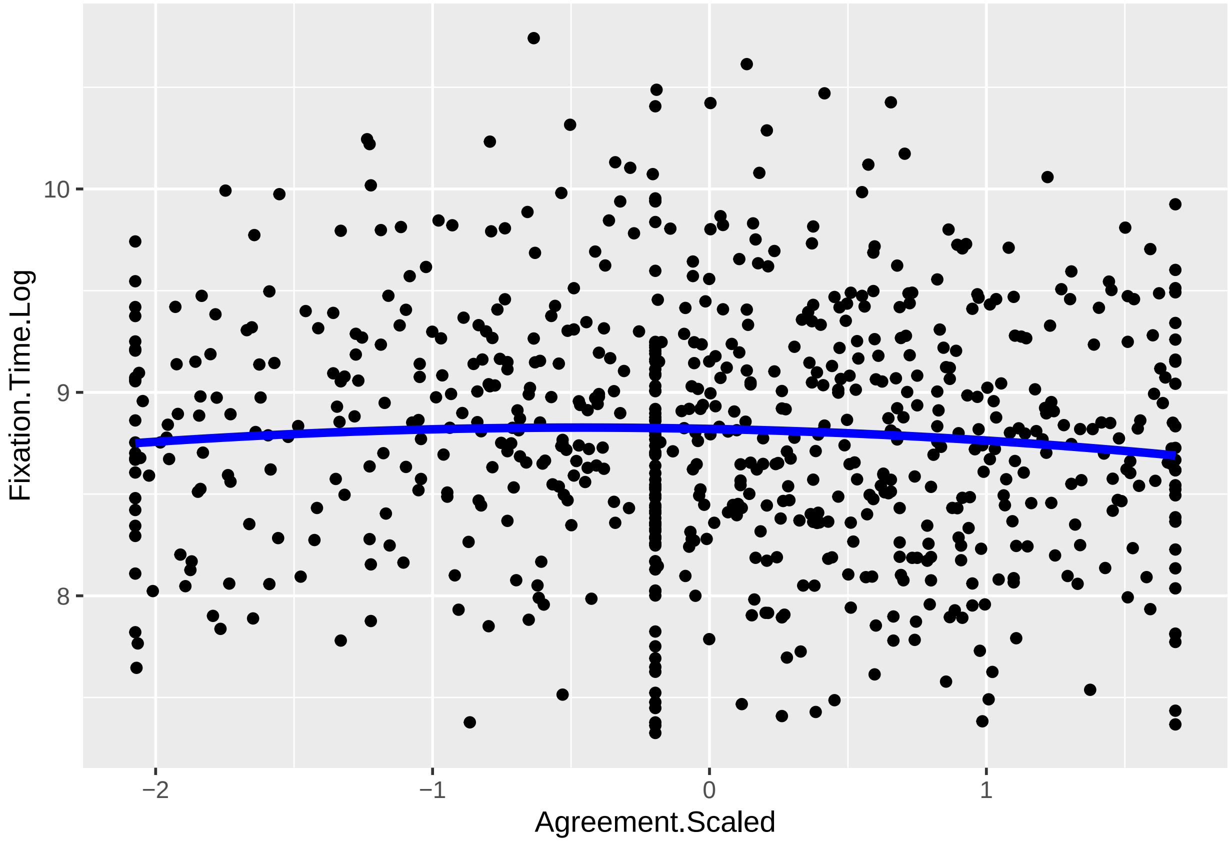
Marginal R2 / Conditional R2

0.004 / 0.603

```
quad_function <- function(x, alpha, beta1, beta2)
{
  return(alpha + beta1*x + beta2*x^2)
}
```

Random effect plot

```
ggplot() +
  geom_point(data=d_final, aes(Agreement.Scaled, Fixation.Time.Log)) +
  geom_function(fun=quad_function,
                args=list(alpha=summary(res_mixed)[["coefficients"]][1],
                                        beta1=summary(res_mixed)[["coefficients"]][2],
                                        beta2=summary(res_mixed)[["coefficients"]][3]),
                color="blue",
                linewidth=1.5)
```



Power analysis

```
res_mixed2 <- extend(res_mixed, along="Subject", n=45)
sim_treat <- powerSim(res_mixed2, nsim=150, progress=FALSE)
```

```
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
```

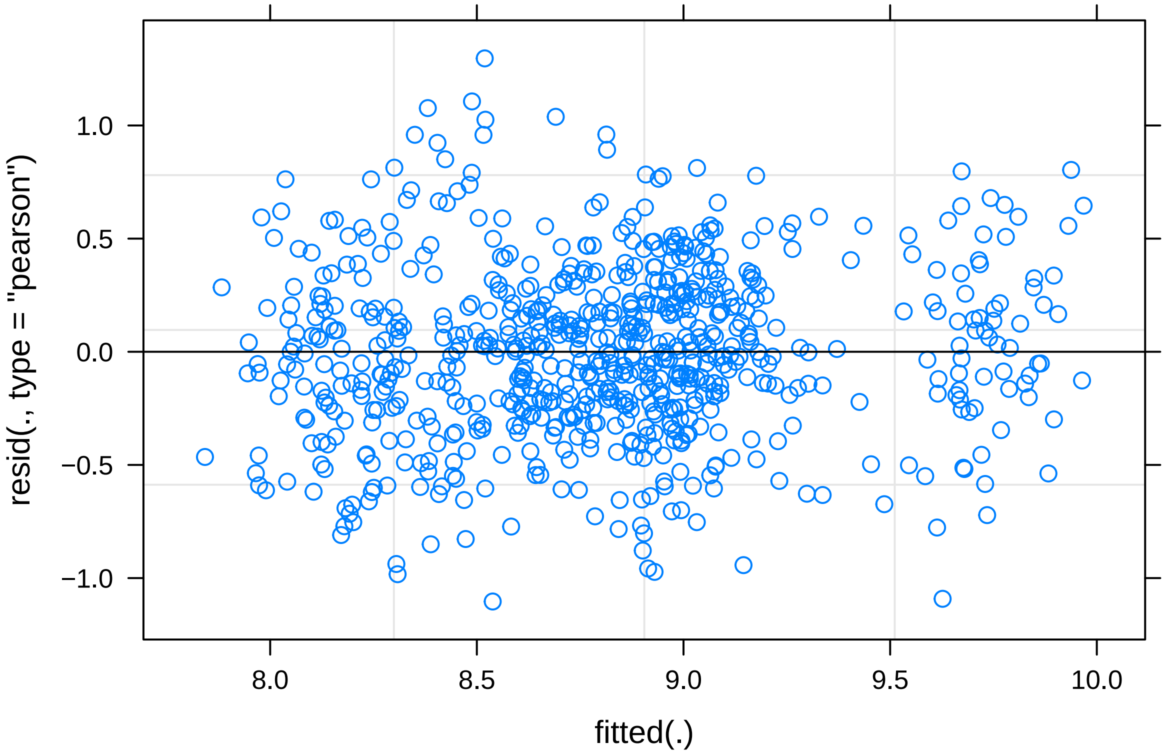
```
## Warning in observedPowerWarning(sim): This appears to be an "observed power"
## calculation
```

```
sim_treat
```

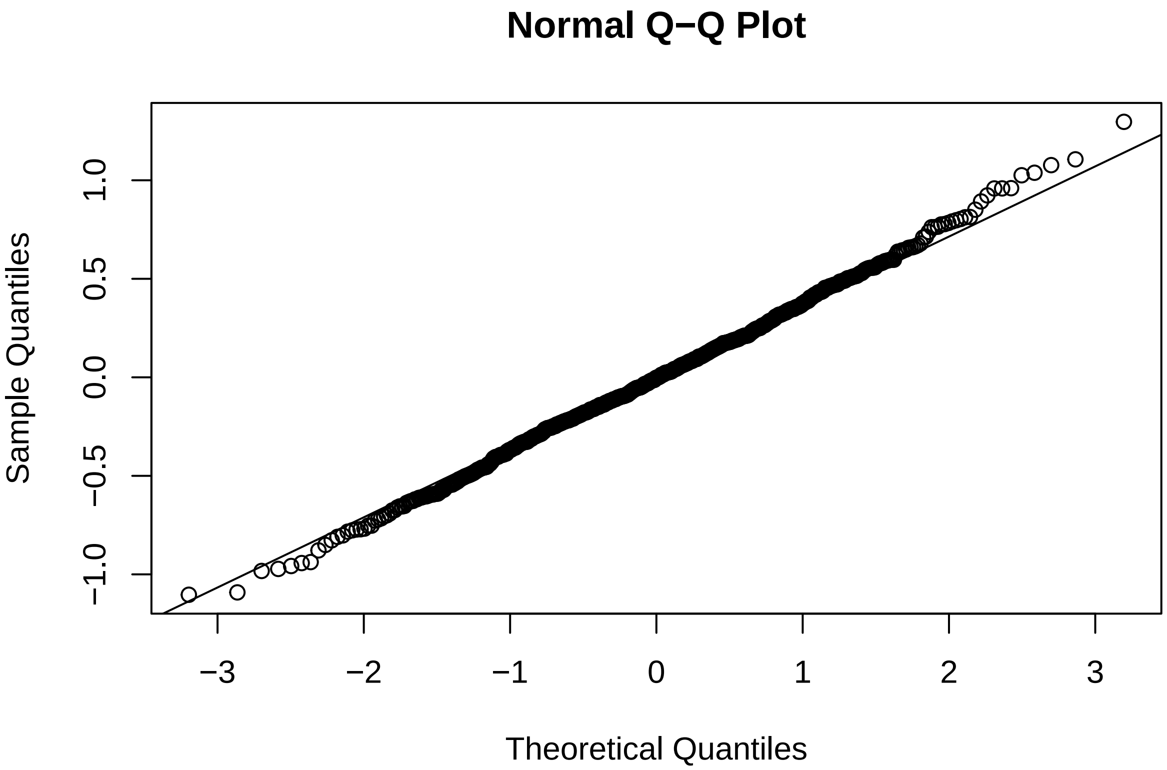
```
## Power for predictor 'Agreement.Scaled', (95% confidence interval):
##       79.33% (71.97, 85.51)
##
## Test: unknown test
##       Effect size for Agreement.Scaled is -0.028
##
## Based on 150 simulations, (65 warnings, 0 errors)
## alpha = 0.05, nrow = 4030
##
## Time elapsed: 9 h 3 m 28 s
##
## nb: result might be an observed power calculation
```

Residuals analysis

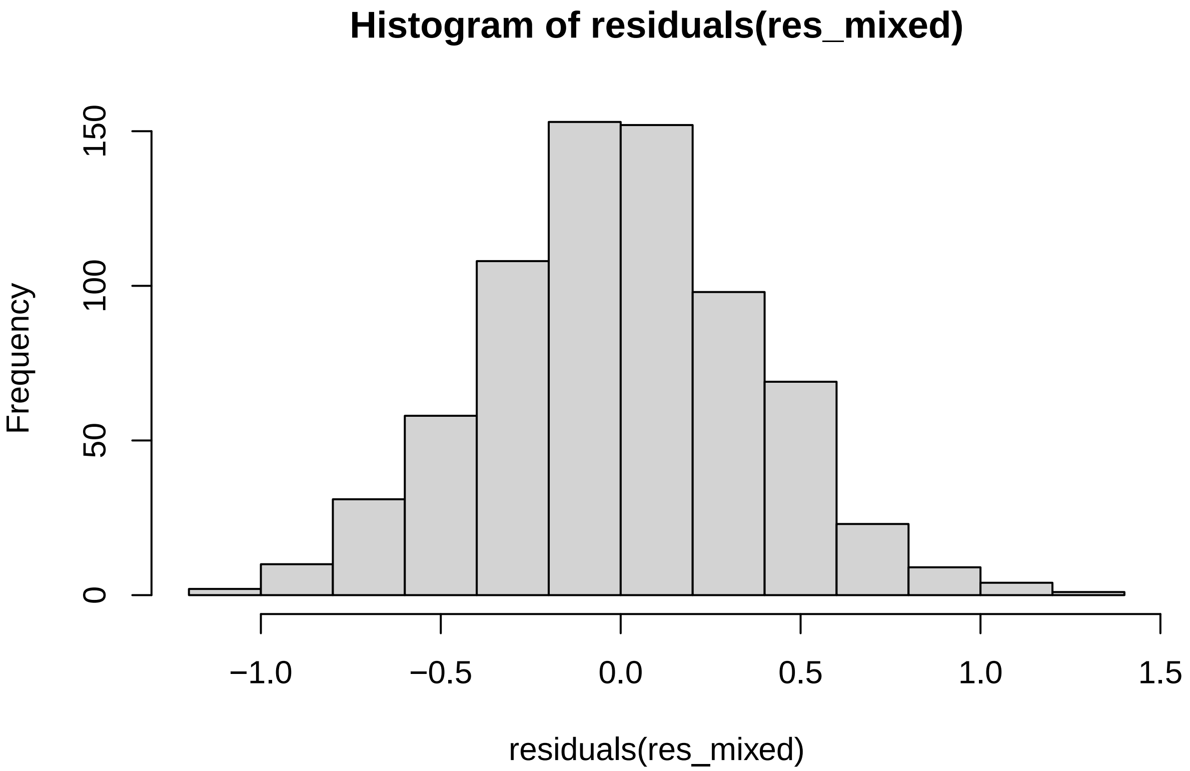
```
plot(res_mixed)
```



```
qqnorm(residuals(res_mixed))
qqline(residuals(res_mixed))
```



```
hist(residuals(res_mixed))
```



```
shapiro.test(residuals(res_mixed))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(res_mixed)
## W = 0.99859, p-value = 0.8486
```