Supplement 9

Experiment 2 – Stimulus-response characteristics of the SPARS

Peter Kamerman

17 Jun 2018

Contents

This analysis examines the stimulus-response characteristics of the SPARS.

Unlike Trial A, where participants were exposed to a prescribed range of stimulus intensities (1 to 4J, at 0.25J intervals), in Trial B, all participants were exposed to 9 stimulus intensities (0.25J interval), but the range of stimuli intensities were calibrated against the sensitivity of each participant. For example, a more sensitive participant may have been exposed to 9 stimuli from 1.75J to 3.75J, another participant may be exposed to stimuli from 2.5J to 4.5J.

This design makes performing group-level analyses difficult (the extremes will have fewer observations), and so we transformed the exposure intensities into an relative scale by ranking (1 to 9) an ordered list of 9 stimulus intensities for each participant. This brought everyone onto the same 1 to 9 scale.

For transparency, we have performed exploratory plots using the raw stimulus intensity data and the relative intensity data. However, the regression analysis was performed using the relative intensity data only.

Import and inspect data

```
# Import
data <- read_rds('./data-cleaned/SPARS_B.rds')

# Inspect
glimpse(data)
```

```
## Observations: 2,256
## Variables: 10
## $ PID                <chr> "ID01", "ID01", "ID01", "ID01", "ID01", "ID0...
## $ scale              <chr> "SPARS", "SPARS", "SPARS", "SPARS", "SPARS",...
```

```
## $ block_number      <int> 2, 2, 2, 4, 4, 4, 6, 6, 6, 8, 8, 8, 11, 11, ...
## $ trial_number      <int> 9, 15, 23, 7, 20, 25, 9, 18, 22, 3, 17, 23, ...
## $ intensity         <dbl> 2.25, 2.25, 2.25, 2.25, 2.25, 2.25, 2.25, 2....
## $ intensity_char    <chr> "2.25", "2.25", "2.25", "2.25", "2.25", "2.2...
## $ intensity_rank    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ rating            <dbl> -31, -20, -48, -48, -21, -23, -48, -45, -47,...
## $ rating_positive   <dbl> 19, 30, 2, 2, 29, 27, 2, 5, 3, 0, 1, 3, 50, ...
## $ rating_equivalent <dbl> -31, -20, -48, -48, -21, -23, -48, -45, -47,...
```

Clean and process data

We performed a basic clean-up of the data, and then calculated *Tukey trimean* at each stimulus intensity for each participant (participant average), and finally the *median* of the trimeans at each stimulus intensity across participants (group average).

```
############################################################
#                                                          #
#                           Clean                          #
#                                                          #
############################################################
data %<>%
    # Rename block_number
    rename(block = block_number) %>%
    # Select SPARS scale
    filter(scale == 'SPARS') %>%
    ungroup() %>%
    arrange(PID)

############################################################
#                                                          #
#                Calculate 'Tukey trimean'                 #
#                                                          #
############################################################
# Define tri.mean function
tri.mean <- function(x) {
  # Calculate quantiles
  q1 <- quantile(x, probs = 0.25, na.rm = TRUE)[[1]]
  q2 <- median(x, na.rm = TRUE)
  q3 <- quantile(x, probs = 0.75, na.rm = TRUE)[[1]]
  # Calculate trimean
  tm <- (q2 + ((q1 + q3) / 2)) / 2
  # Convert to integer
  tm <- as.integer(round(tm))
  return(tm)
}

# Calculate the participant average based on 'raw' intensity
data_tm <- data %>%
  group_by(PID, intensity) %>%
  summarise(tri_mean = tri.mean(rating)) %>%
```

```r
  ungroup()

# Calculate the group average based on 'raw' intensity
data_group <- data_tm %>%
  group_by(intensity) %>%
  summarise(median = median(tri_mean)) %>%
  ungroup()

# Calculate the participant average based on 'relative' intensity
data_tmR <- data %>%
  group_by(PID, intensity_rank) %>%
  summarise(tri_mean = tri.mean(rating)) %>%
  ungroup()

# Calculate the group average based on 'relative' intensity
data_groupR <- data_tmR %>%
  group_by(intensity_rank) %>%
  summarise(median = median(tri_mean)) %>%
  ungroup()
```

Stimulus exposure ranges

```r
knitr::kable(data %>%
                 group_by(PID) %>%
                 summarise(`Minimum stimulus intensity` = min(intensity),
                           `Maximum stimulus intensity` = max(intensity)),
             caption = 'Range of stimulus intensities covered in each participant')
```

Table 1: Range of stimulus intensities covered in each participant

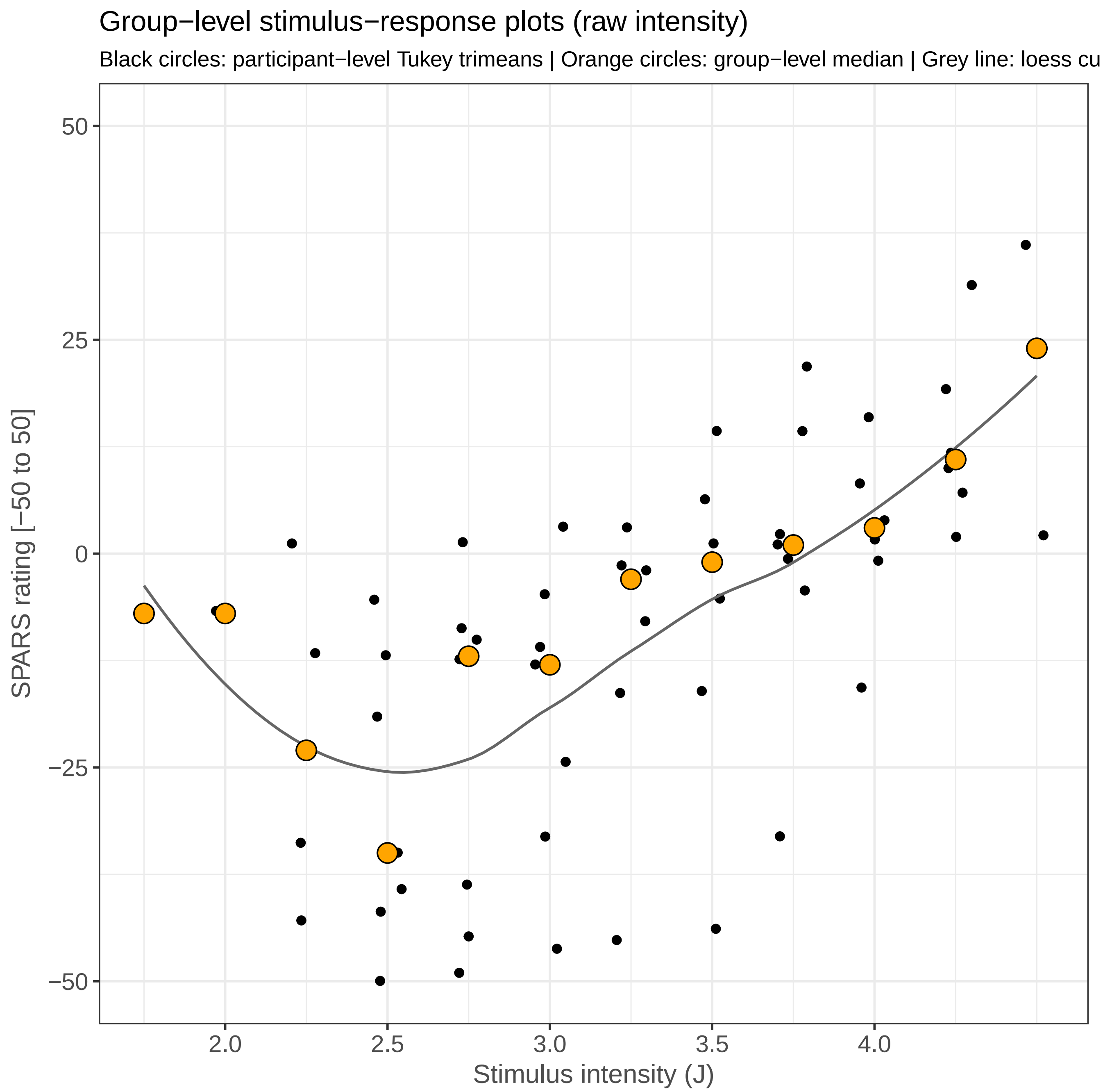
PID	Minimum stimulus intensity	Maximum stimulus intensity
ID01	2.25	4.25
ID02	2.25	4.25
ID03	2.50	4.50
ID04	2.50	4.50
ID05	2.50	4.50
ID06	1.75	3.75
ID07	2.25	4.25

Exploratory plots

Group-level stimulus response curve

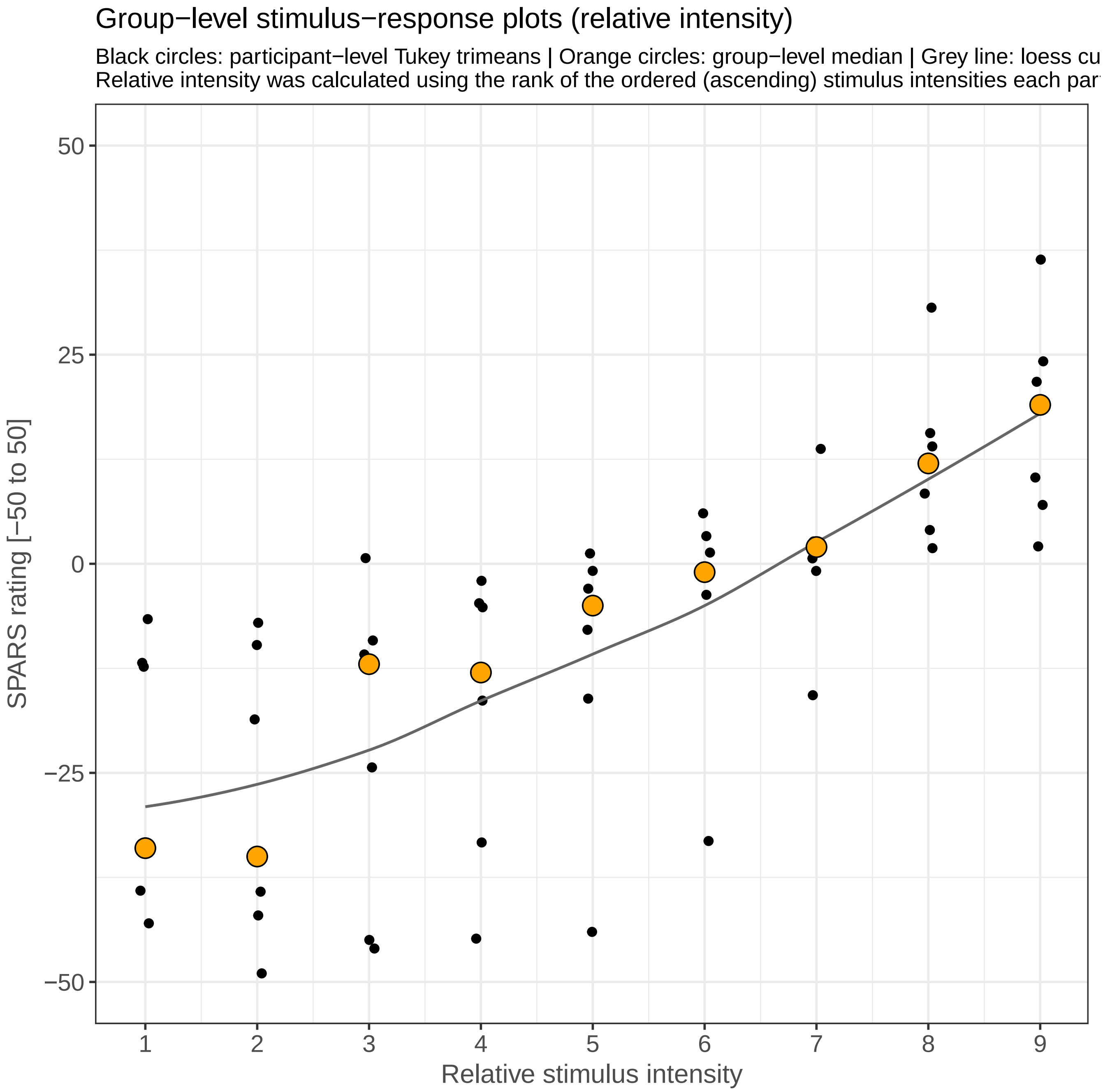
```r
# Plot (y.axis = raw stimulus intensity)
data_tm %>%
```

```
ggplot(data = .) +
aes(x = intensity,
    y = tri_mean) +
geom_point(position = position_jitter(width = 0.05)) +
geom_smooth(method = 'loess',
            se = FALSE,
            colour = '#666666',
            size = 0.6) +
geom_point(data = data_group,
           aes(y = median),
           shape = 21,
           size = 4,
           fill = '#FFA500') +
labs(title = 'Group-level stimulus-response plots (raw intensity)',
     subtitle = 'Black circles: participant-level Tukey trimeans | Orange circles: group
     x = 'Stimulus intensity (J)',
     y = 'SPARS rating [-50 to 50]') +
scale_y_continuous(limits = c(-50, 50)) +
scale_x_continuous(breaks = seq(from = 1, to = 4, by = 0.5))
```




```
# Plot (y.axis = relative stimulus intensity)
data_tmR %>%
  ggplot(data = .) +
  aes(x = intensity_rank,
      y = tri_mean) +
  geom_point(position = position_jitter(width = 0.05)) +
  geom_smooth(method = 'loess',
              se = FALSE,
              colour = '#666666',
              size = 0.6) +
  geom_point(data = data_groupR,
             aes(y = median),
             shape = 21,
             size = 4,
             fill = '#FFA500') +
  labs(title = 'Group-level stimulus-response plots (relative intensity)',
       subtitle = 'Black circles: participant-level Tukey trimeans | Orange circles: group
```

```
      x = 'Relative stimulus intensity',
      y = 'SPARS rating [-50 to 50]') +
  scale_y_continuous(limits = c(-50, 50)) +
  scale_x_continuous(breaks = seq(from = 1, to = 9, by = 1))
```

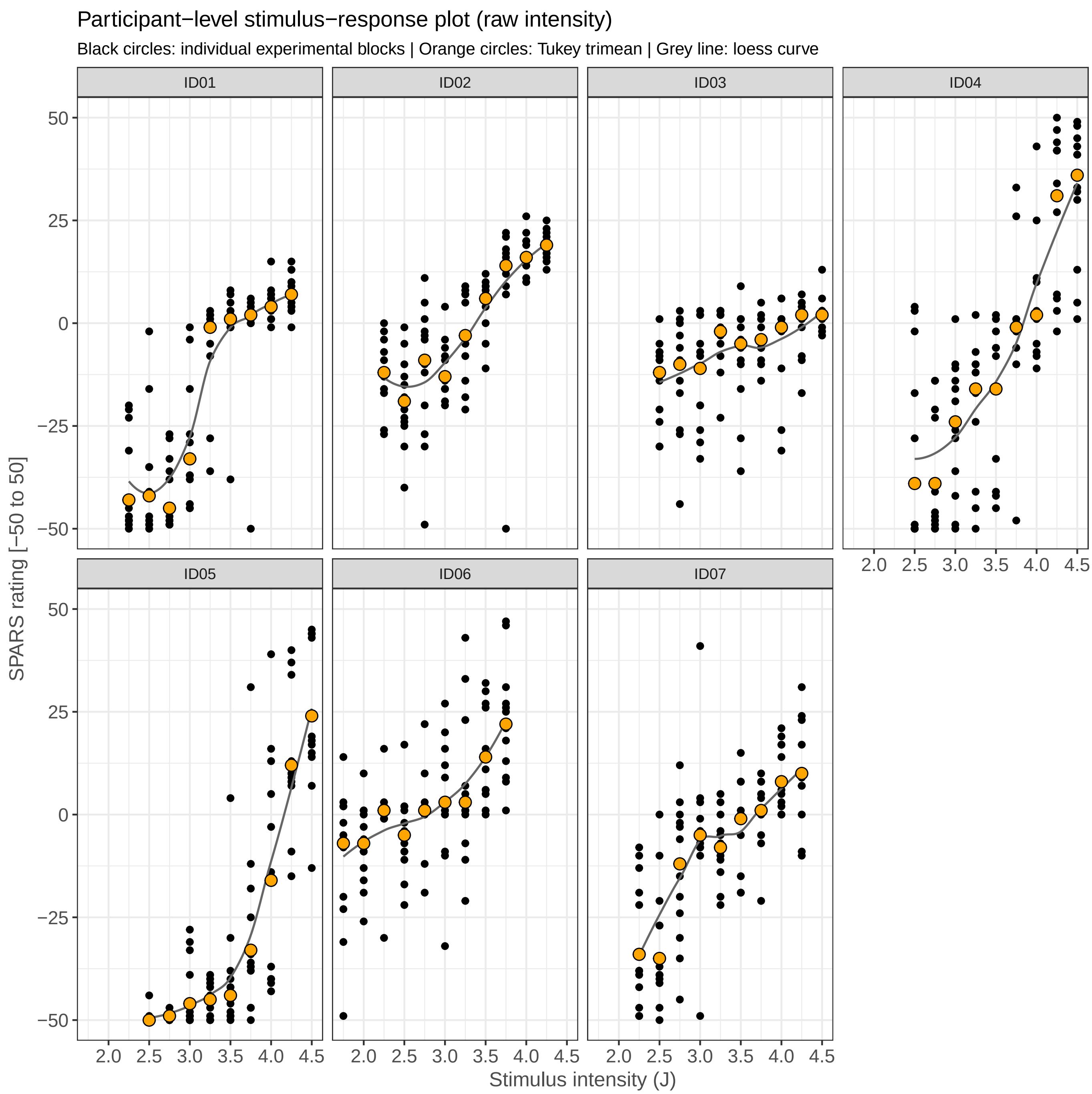


Participant-level stimulus response curves

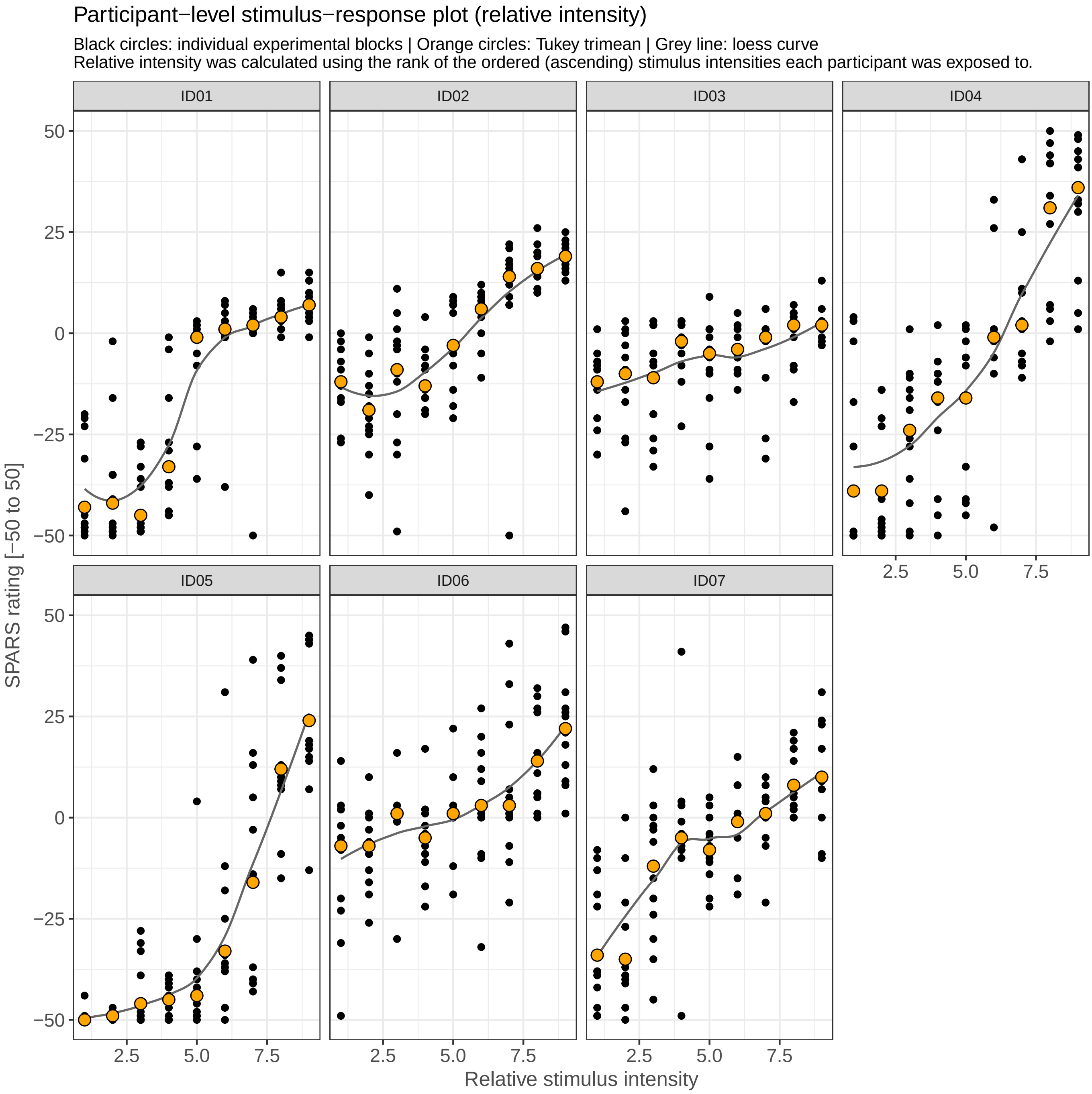
All trials

```
# Plot (y.axis = raw stimulus intensity)
data %>%
  ggplot(data = .) +
  aes(x = intensity,
      y = rating) +
  geom_point() +
  geom_smooth(method = 'loess',
```

```
              se = FALSE,
              colour = '#666666',
              size = 0.6) +
  geom_point(data = data_tm,
             aes(y = tri_mean),
             shape = 21,
             size = 3,
             fill = '#FFA500') +
  labs(title = 'Participant-level stimulus-response plot (raw intensity)',
       subtitle = 'Black circles: individual experimental blocks | Orange circles: Tukey t
       x = 'Stimulus intensity (J)',
       y = 'SPARS rating [-50 to 50]') +
  scale_y_continuous(limits = c(-50, 50)) +
  facet_wrap(~ PID, ncol = 4)
```



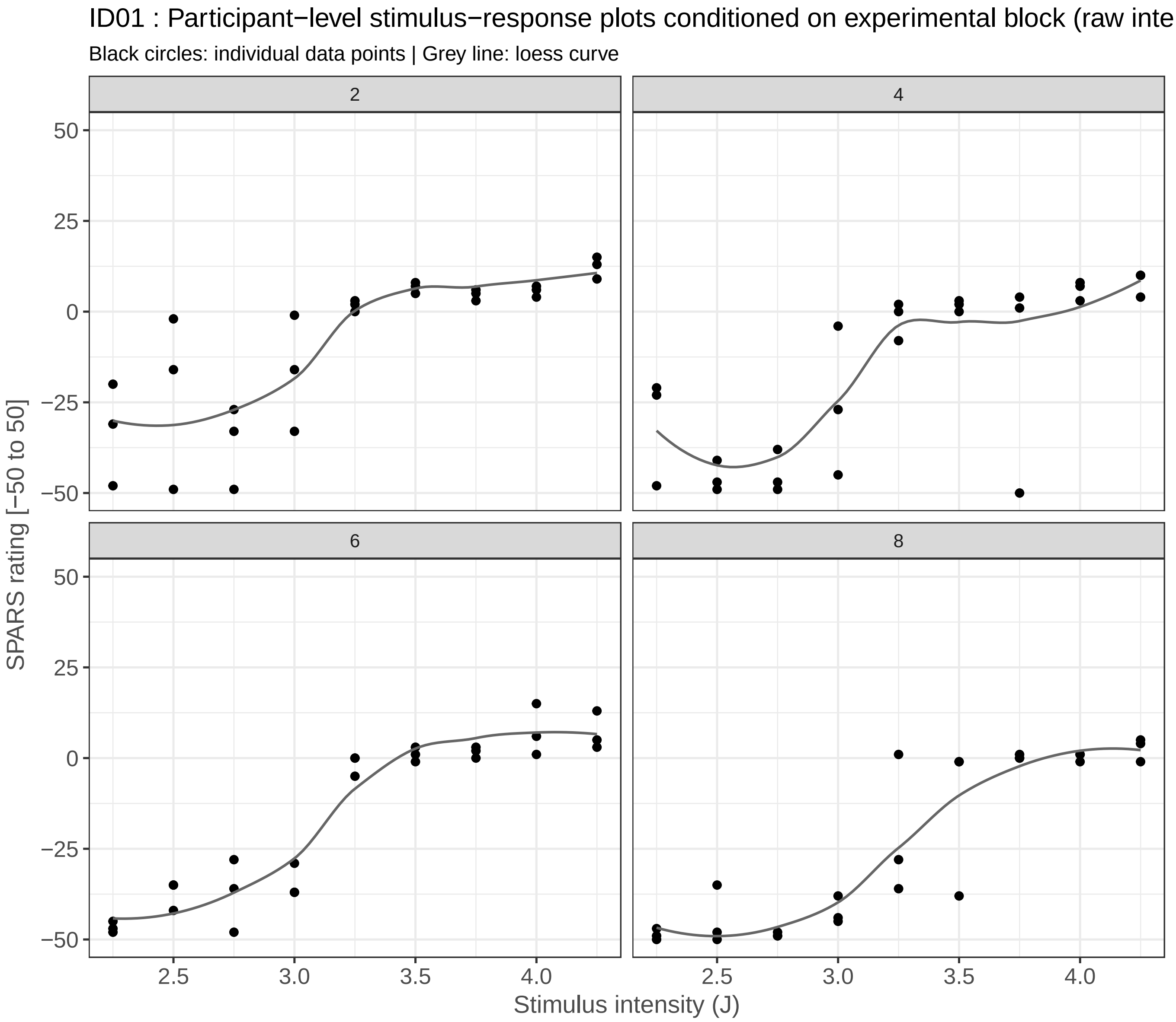
```r
# Plot (y.axis = rank stimulus intensity)
data %>%
  ggplot(data = .) +
  aes(x = intensity_rank,
      y = rating) +
  geom_point() +
  geom_smooth(method = 'loess',
              se = FALSE,
              colour = '#666666',
              size = 0.6) +
  geom_point(data = data_tmR,
             aes(y = tri_mean),
             shape = 21,
             size = 3,
             fill = '#FFA500') +
  labs(title = 'Participant-level stimulus-response plot (relative intensity)',
       subtitle = 'Black circles: individual experimental blocks | Orange circles: Tukey t
       x = 'Relative stimulus intensity',
       y = 'SPARS rating [-50 to 50]') +
  scale_y_continuous(limits = c(-50, 50)) +
  facet_wrap(~ PID, ncol = 4)
```



Trials by experimental block

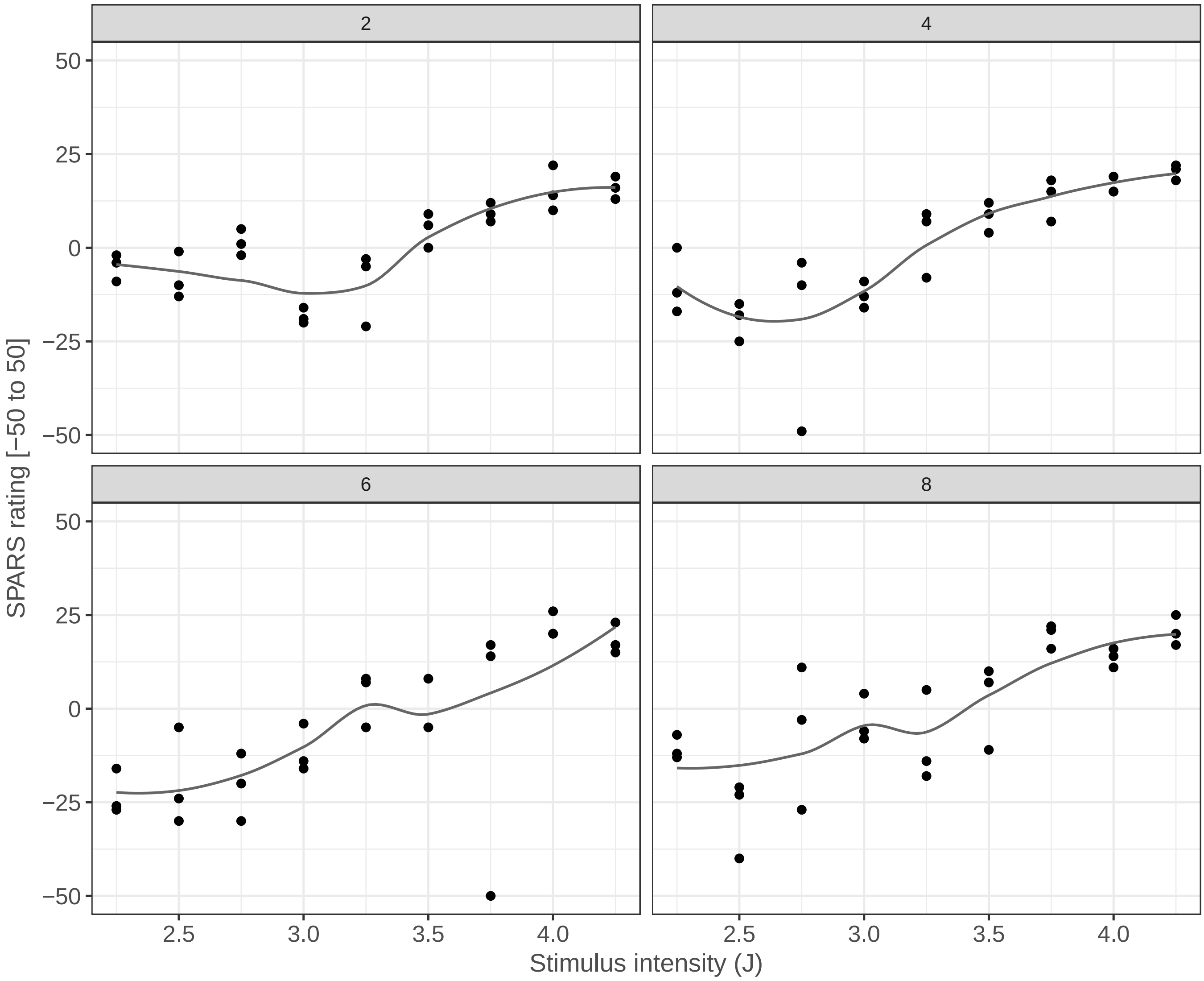
```
# Process data (raw stimulus intensity)
data_block <- data %>%
  # Rename blocks
  #mutate(block = sprintf('Block: %s (order: %i)', block, block_order)) %>%
  # Nest by PID
  group_by(PID) %>%
  nest() %>%
  # Generate plots
  mutate(plots = map2(.x = data,
                      .y = unique(PID),
                      ~ ggplot(data = .x) +
                        aes(x = intensity,
                            y = rating) +
                        geom_point() +
```

```r
                                  geom_smooth(method = 'loess',
                                              se = FALSE,
                                              colour = '#666666',
                                              size = 0.6) +
                                  labs(title = paste(.y, ': Participant-level stimulus-response plot
                                       subtitle = 'Black circles: individual data points | Grey line
                                       x = 'Stimulus intensity (J)',
                                       y = 'SPARS rating [-50 to 50]') +
                                  scale_y_continuous(limits = c(-50, 50)) +
                                  facet_wrap(~ block, ncol = 2)))

# Print plots
walk(.x = data_block$plots, ~ print(.x))
```



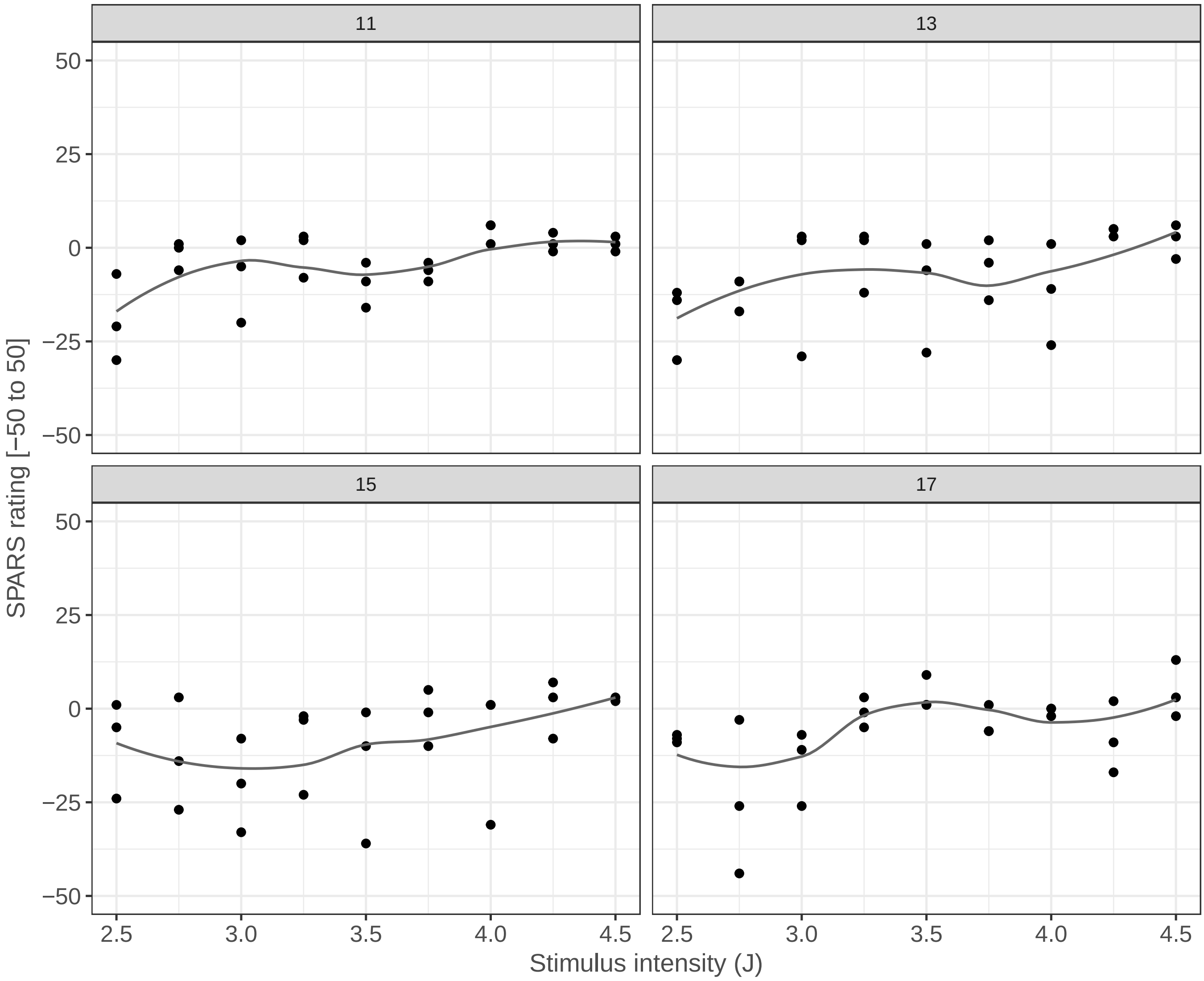
ID02 : Participant−level stimulus−response plots conditioned on experimental block (raw inte

Black circles: individual data points | Grey line: loess curve



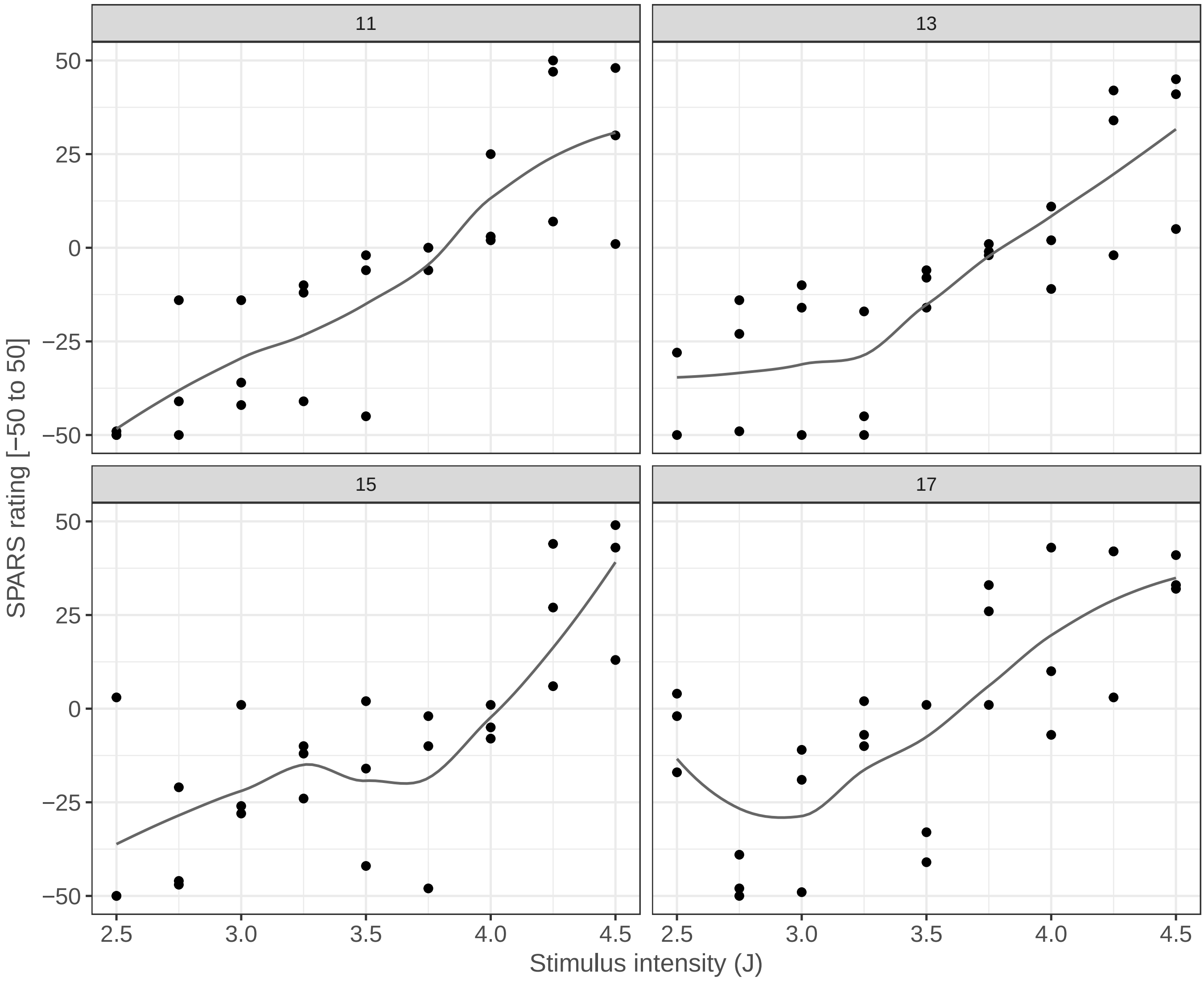
ID03 : Participant−level stimulus−response plots conditioned on experimental block (raw inte

Black circles: individual data points | Grey line: loess curve



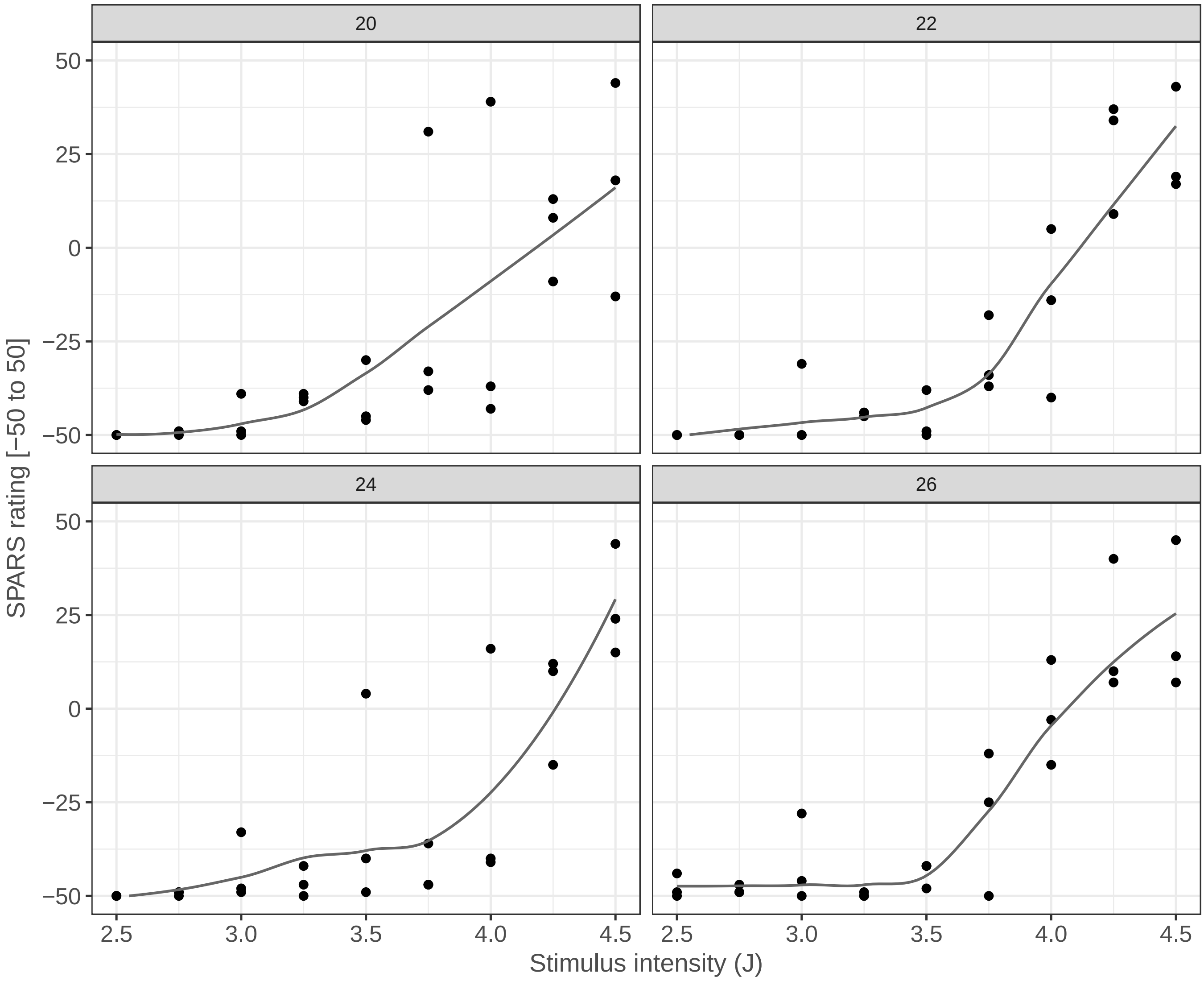
ID04 : Participant−level stimulus−response plots conditioned on experimental block (raw inte

Black circles: individual data points | Grey line: loess curve



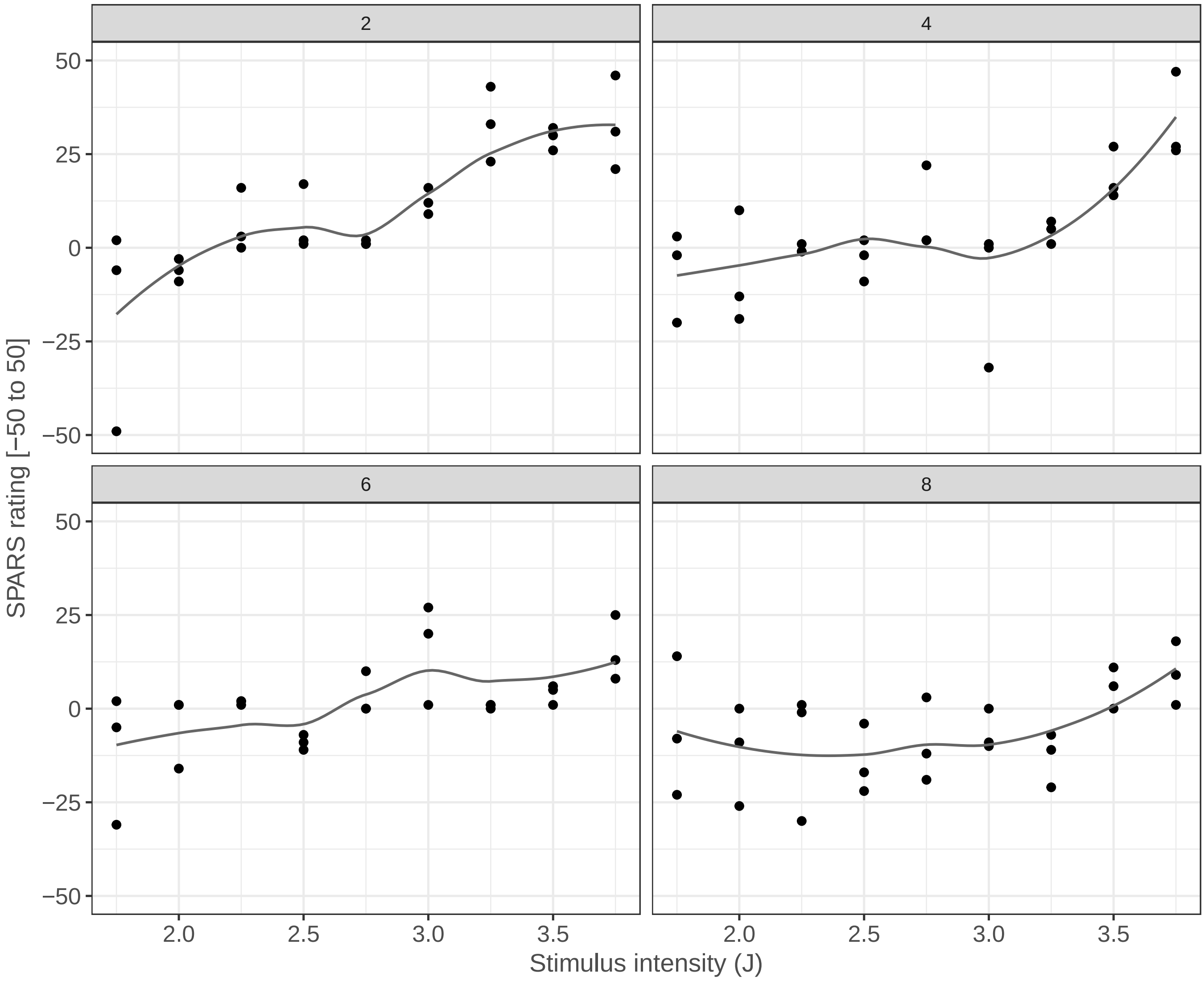
ID05 : Participant−level stimulus−response plots conditioned on experimental block (raw inte

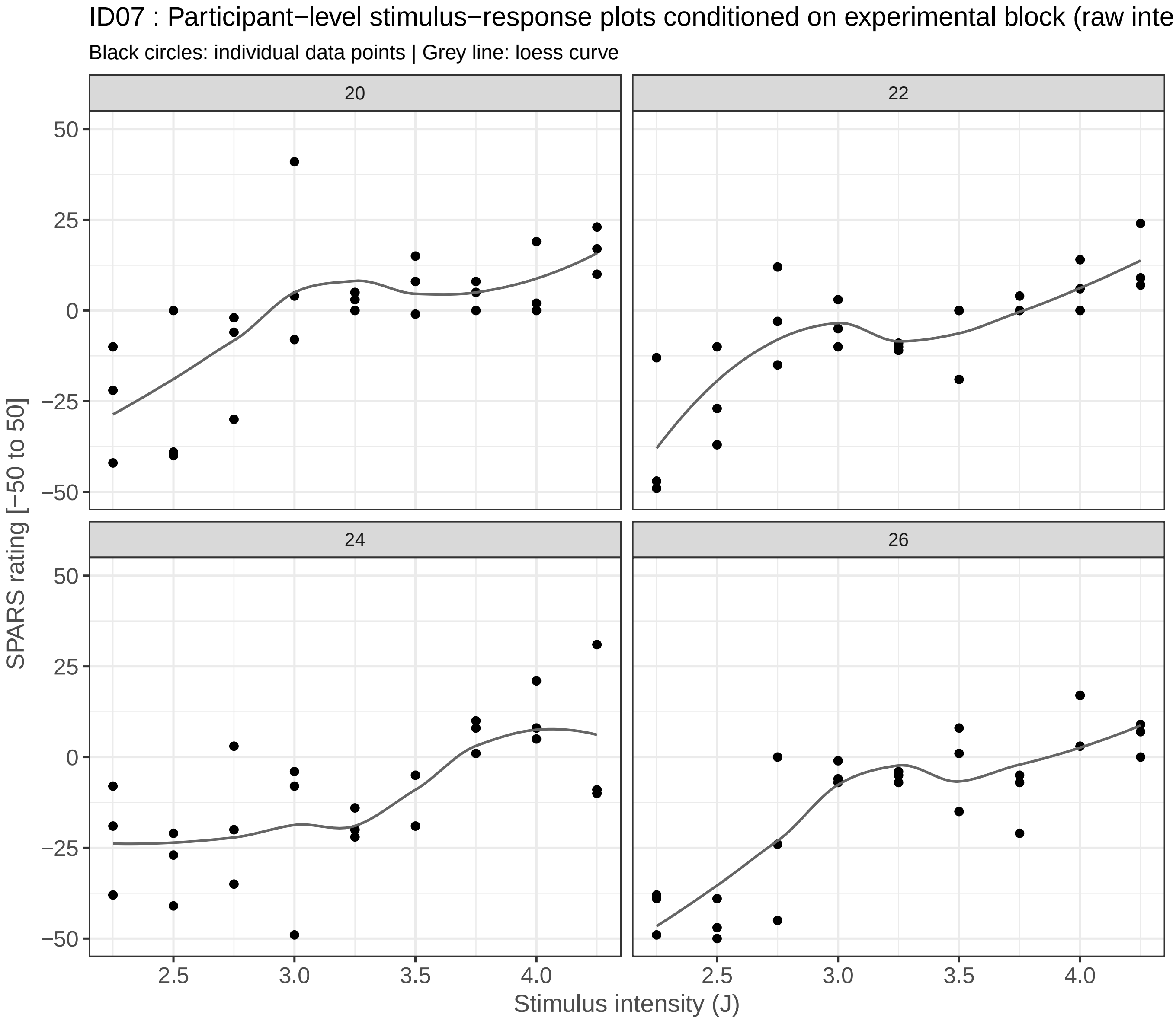
Black circles: individual data points | Grey line: loess curve



ID06 : Participant−level stimulus−response plots conditioned on experimental block (raw inte

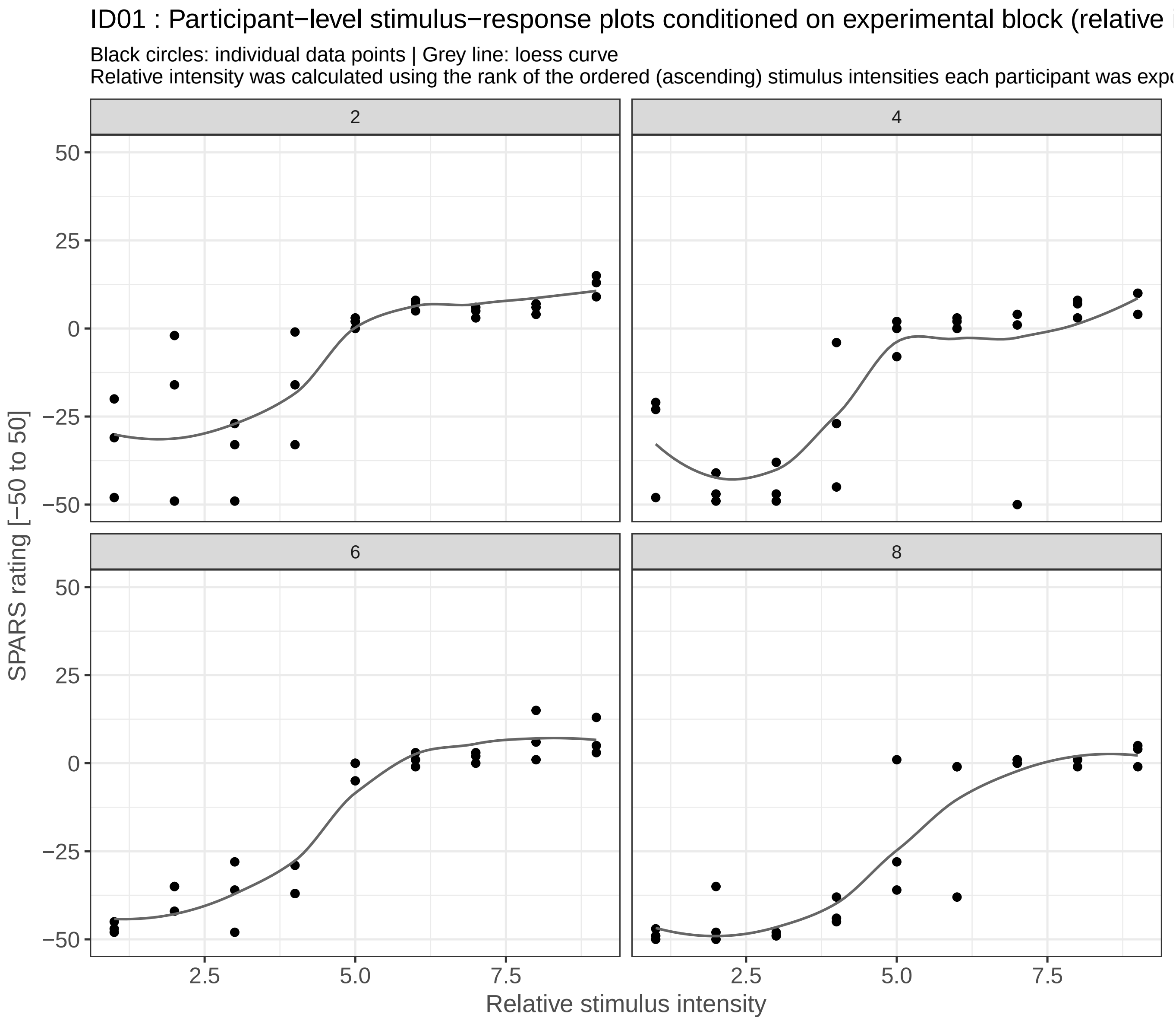
Black circles: individual data points | Grey line: loess curve

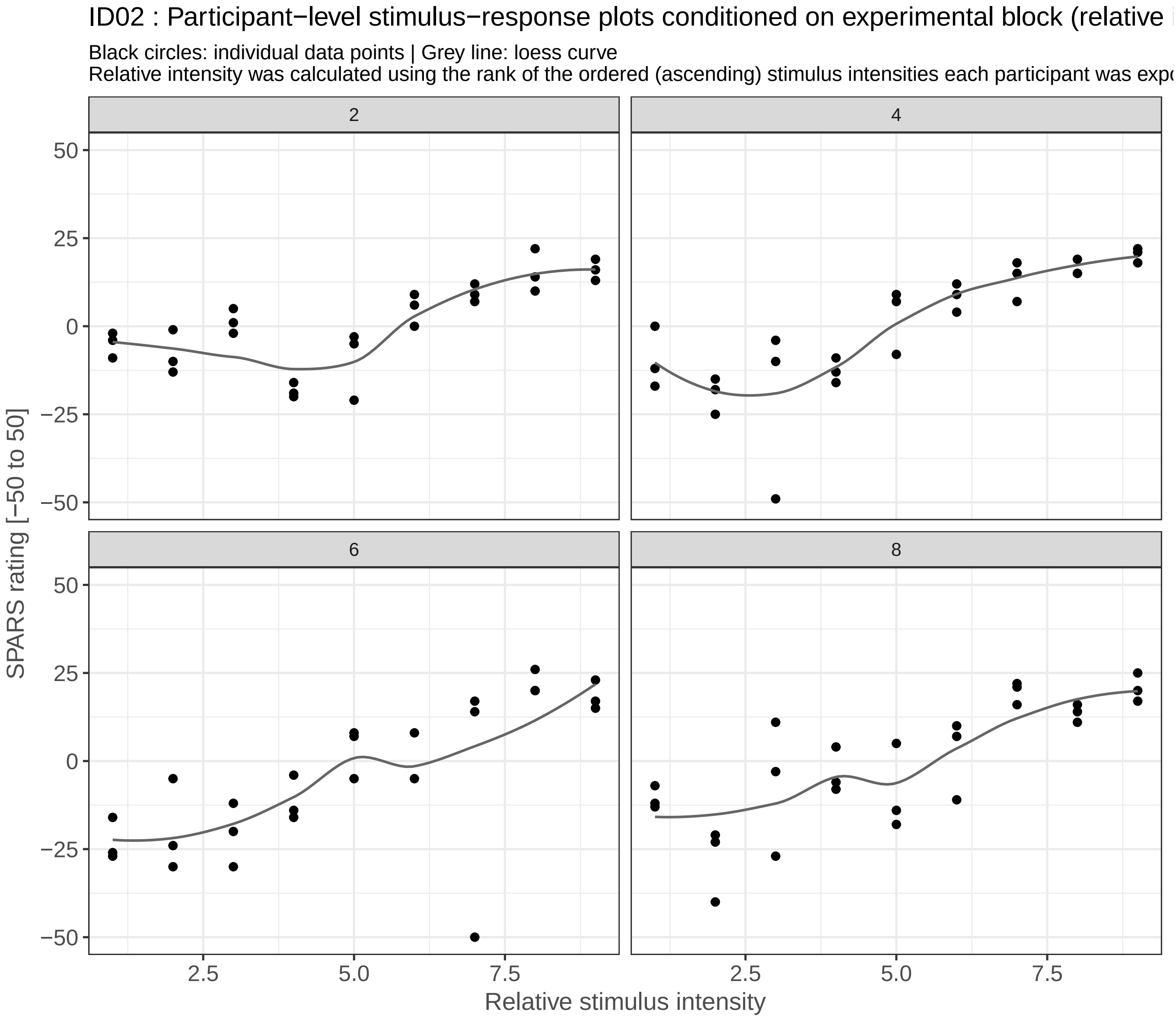





```
# Process data (relative stimulus intensity)
data_blockR <- data %>%
  # Rename blocks
  #mutate(block = sprintf('Block: %s (order: %i)', block, block_order)) %>%
  # Nest by PID
  group_by(PID) %>%
  nest() %>%
  # Generate plots
  mutate(plots = map2(.x = data,
                      .y = unique(PID),
                      ~ ggplot(data = .x) +
                        aes(x = intensity_rank,
                            y = rating) +
                        geom_point() +
                        geom_smooth(method = 'loess',
                                    se = FALSE,
                                    colour = '#666666',
                                    size = 0.6) +
                        labs(title = paste(.y, ': Participant-level stimulus-response plot
                             subtitle = 'Black circles: individual data points | Grey line
                             x = 'Relative stimulus intensity',
```

```
                        y = 'SPARS rating [-50 to 50]') +
                    scale_y_continuous(limits = c(-50, 50)) +
                    facet_wrap(~ block, ncol = 2)))

# Print plots
walk(.x = data_blockR$plots, ~ print(.x))
```

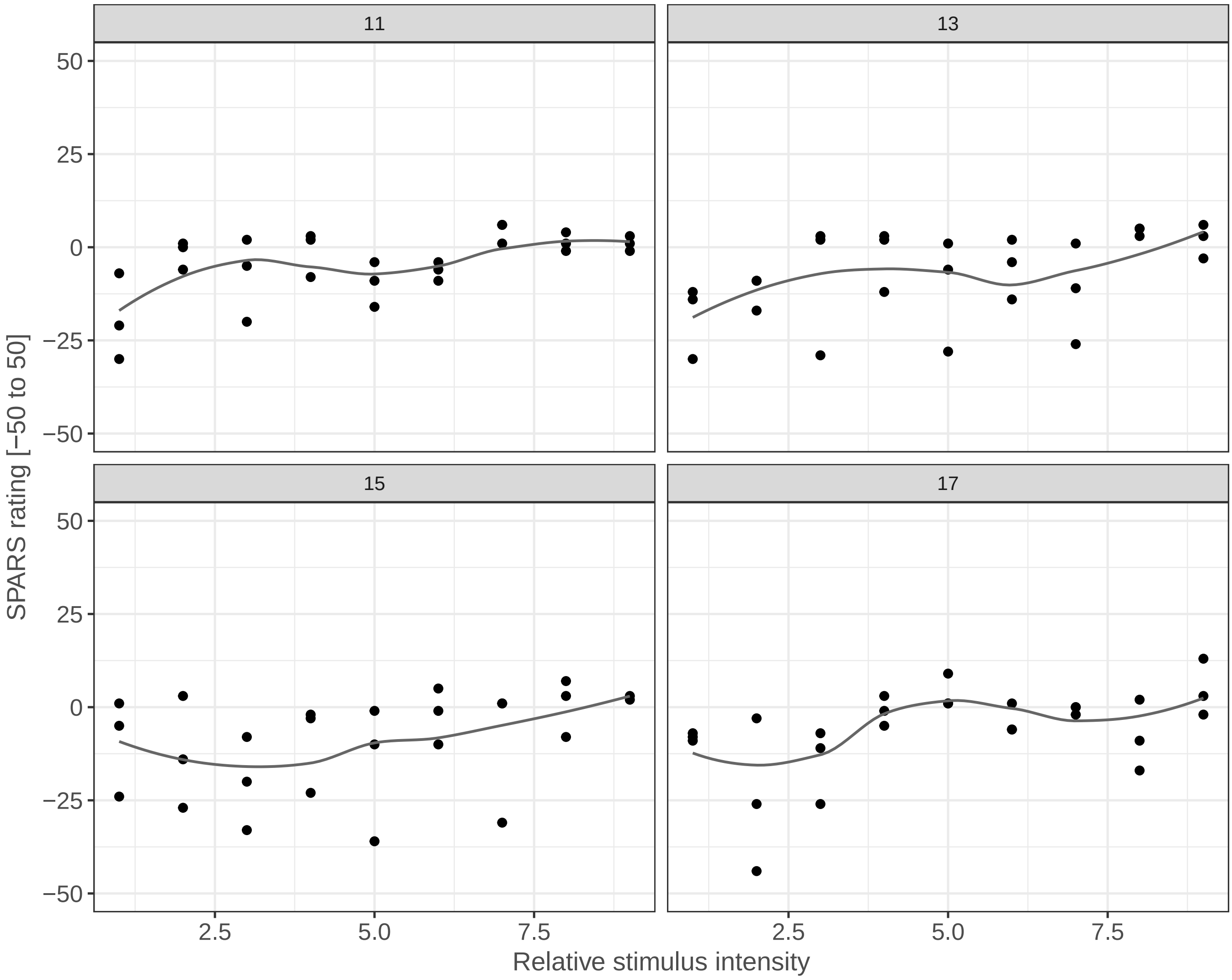




ID02 : Participant−level stimulus−response plots conditioned on experimental block (relative i
Black circles: individual data points | Grey line: loess curve
Relative intensity was calculated using the rank of the ordered (ascending) stimulus intensities each participant was expo
2
4
6
8
50
25
0
−25
−50
SPARS rating [−50 to 50]
2.5
5.0
7.5
Relative stimulus intensity

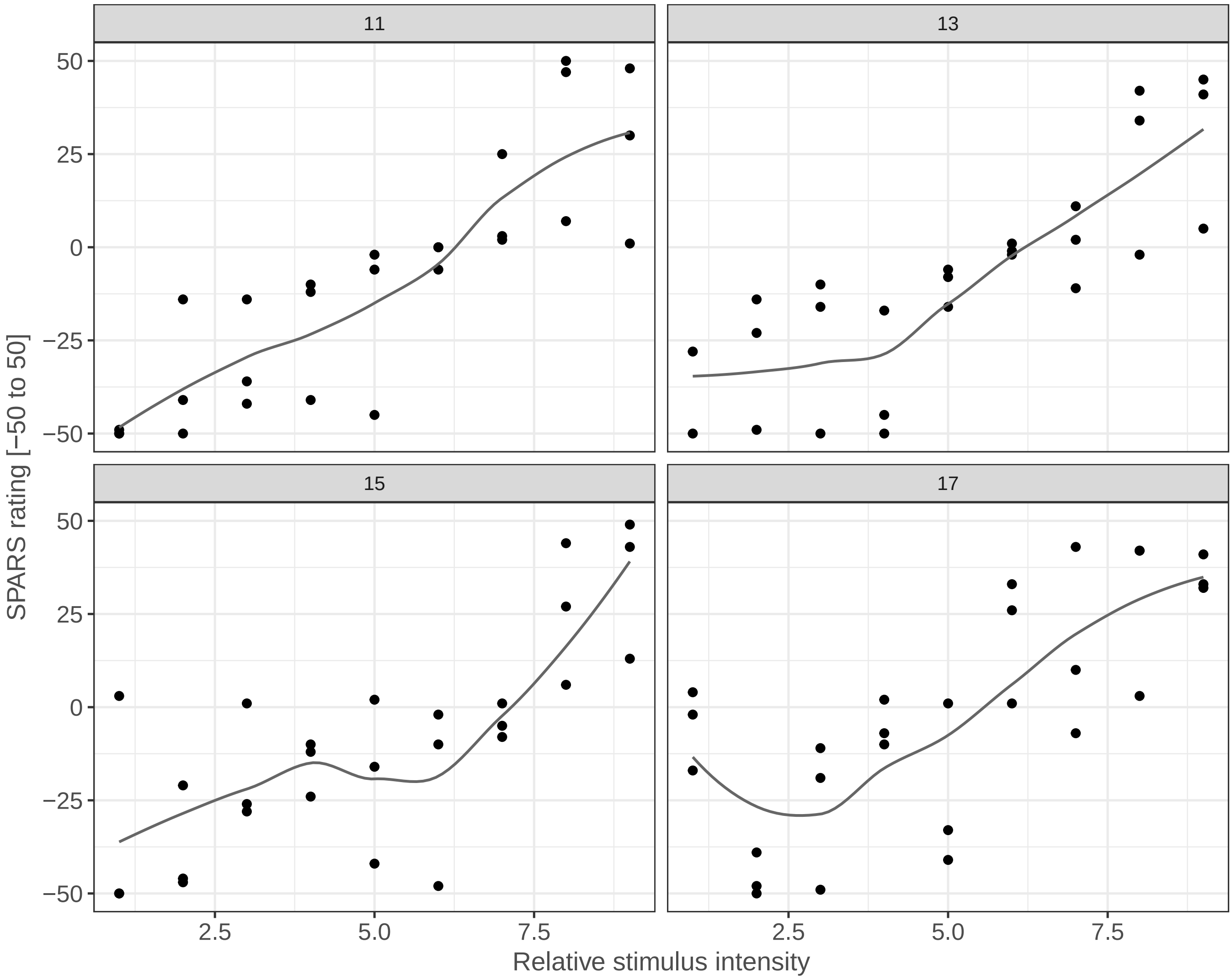
ID03 : Participant−level stimulus−response plots conditioned on experimental block (relative i

Black circles: individual data points | Grey line: loess curve
Relative intensity was calculated using the rank of the ordered (ascending) stimulus intensities each participant was expo



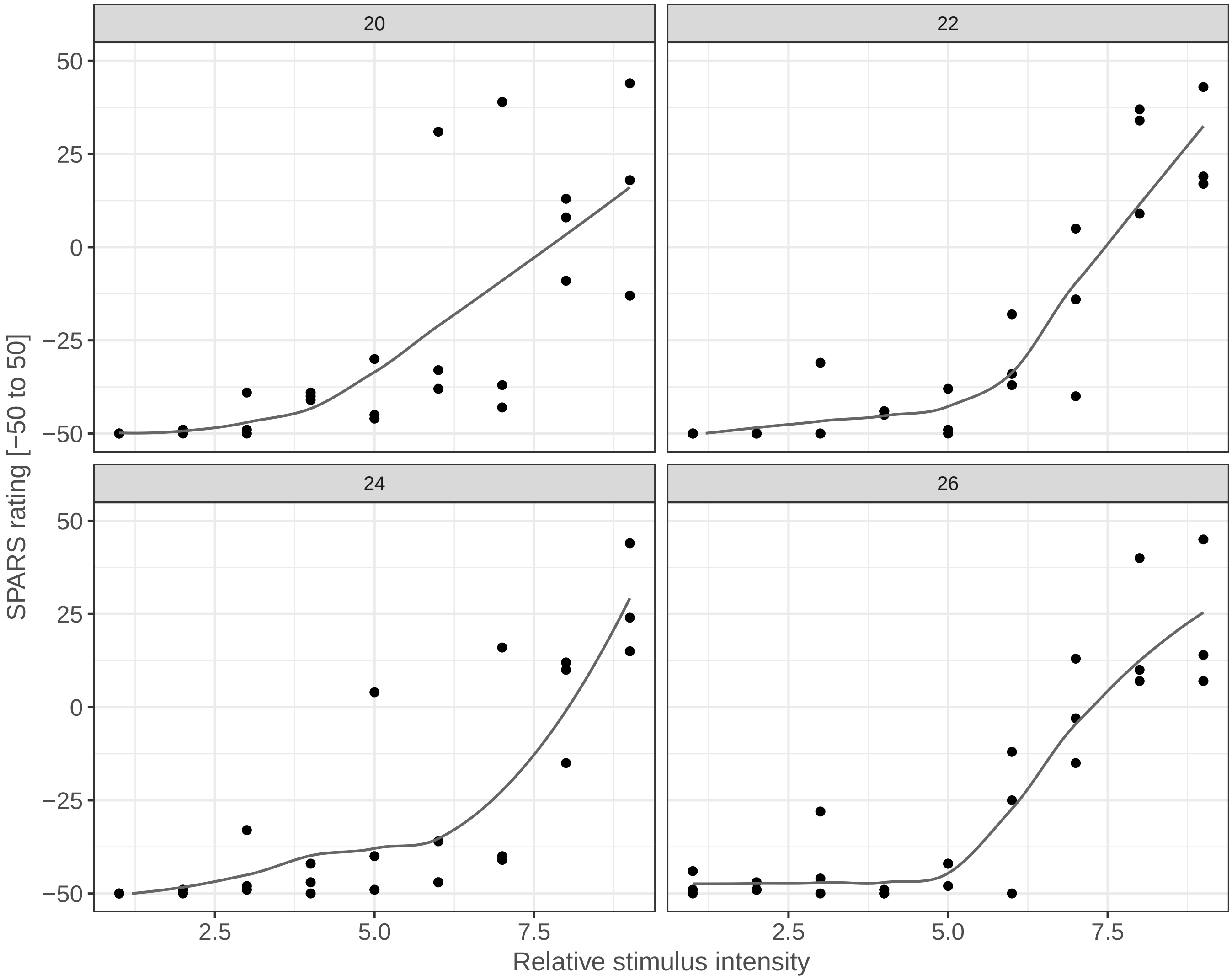
ID04 : Participant−level stimulus−response plots conditioned on experimental block (relative i

Black circles: individual data points | Grey line: loess curve
Relative intensity was calculated using the rank of the ordered (ascending) stimulus intensities each participant was expo



ID05 : Participant−level stimulus−response plots conditioned on experimental block (relative i

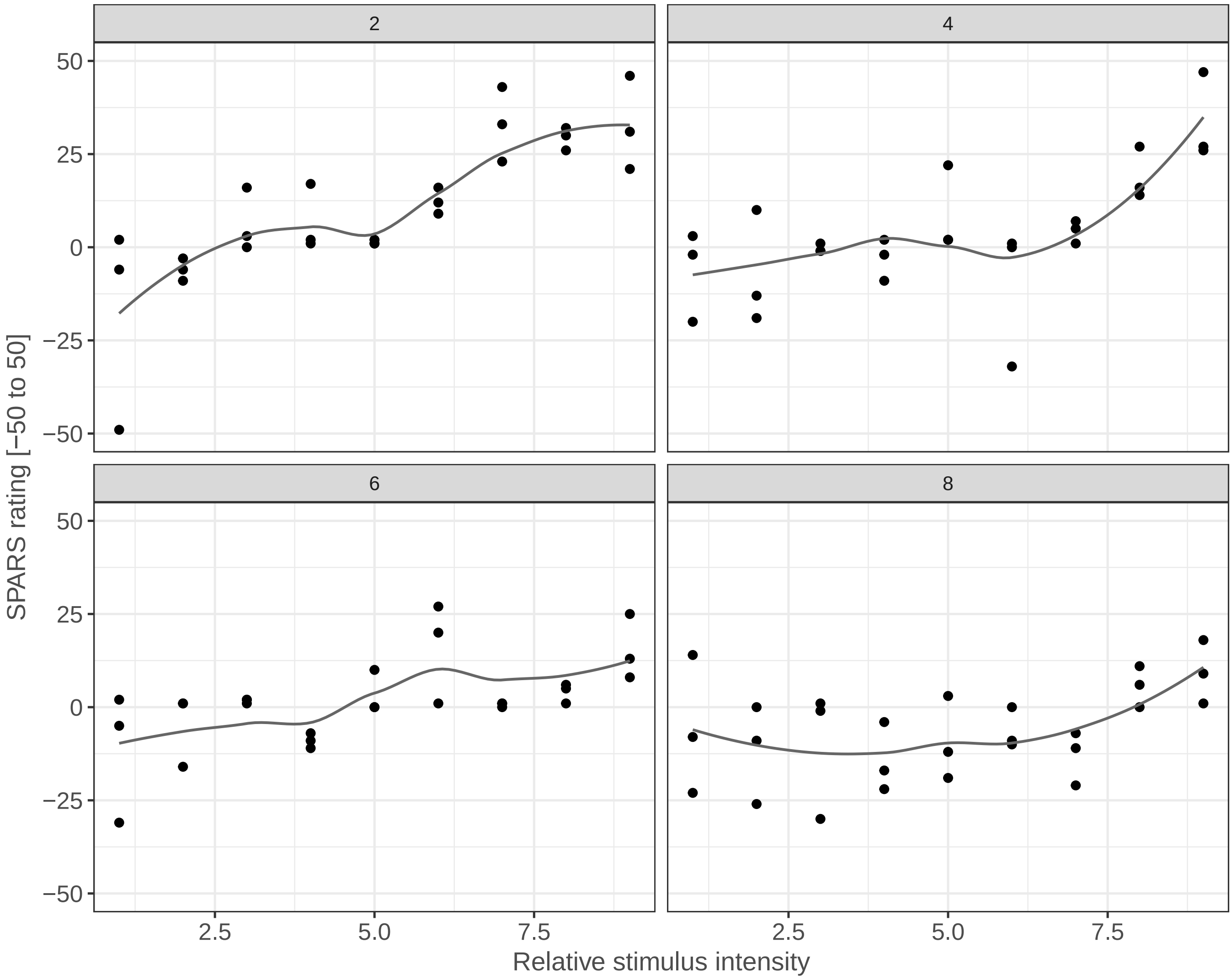
Black circles: individual data points | Grey line: loess curve
Relative intensity was calculated using the rank of the ordered (ascending) stimulus intensities each participant was expo

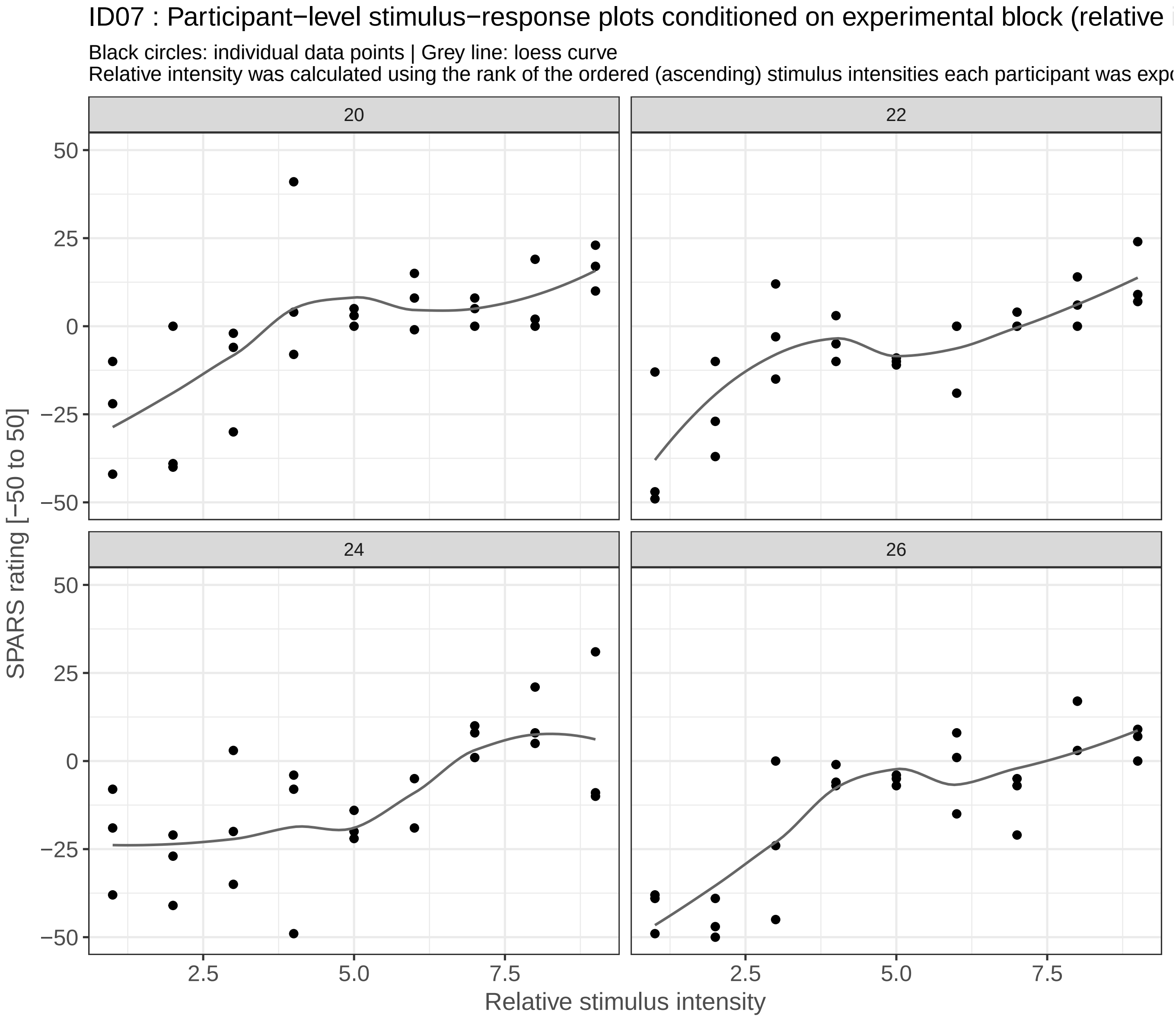


ID06 : Participant−level stimulus−response plots conditioned on experimental block (relative i

Black circles: individual data points | Grey line: loess curve

Relative intensity was calculated using the rank of the ordered (ascending) stimulus intensities each participant was expo





Linear mixed model regression

To allow for a curvilinear relationship between stimulus intensity and rating, we modelled the data using polynomial regression, with 1st (linear), 2nd (quadratic), and 3rd (cubic) order orthogonal polynomials. For each polynomial expression, we modelled the random effects as random intercept only, and as random intercept and slope.

The random intercept only and random intercept and slope models were compared using the loglikelihood test, and the better model taken forward. Diagnostics were run on the final model only, and we examined level 1 residuals (conditional / fixed effects), and level 2 residuals (random effects) and influence points [1].

1st-order (linear) polynomial

```
# Intercept only
lmm1 <- lmer(tri_mean ~ intensity_rank + (1 | PID),
             data = data_tmR,
```

[1] Loy A, Hofmann H. HLMdiag: A suite of diagnostics for hierarchical linear models in R. J. Stat. Softw. 2014;56:1–28. Available

```r
             REML = TRUE)

# Intercept and slope
lmm1b <- lmer(tri_mean ~ intensity_rank + (intensity_rank | PID),
              data = data_tmR,
              REML = TRUE)

# Better model?
anova(lmm1, lmm1b)
```

```
## Data: data_tmR
## Models:
## lmm1: tri_mean ~ intensity_rank + (1 | PID)
## lmm1b: tri_mean ~ intensity_rank + (intensity_rank | PID)
##       Df    AIC    BIC  logLik deviance Chisq Chi Df Pr(>Chisq)
## lmm1   4 495.89 504.46 -243.94   487.89
## lmm1b  6 475.23 488.08 -231.61   463.23 24.66      2  4.418e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Anova of better model
Anova(lmm1b,
      type = 2,
      test.statistic = 'F')
```

```
## Analysis of Deviance Table (Type II Wald F tests with Kenward-Roger df)
##
## Response: tri_mean
##                     F Df Df.res   Pr(>F)
## intensity_rank 28.612  1      6 0.001746 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Print better model
summary(lmm1b)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: tri_mean ~ intensity_rank + (intensity_rank | PID)
##    Data: data_tmR
##
## REML criterion at convergence: 457.4
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.33646 -0.51117 -0.07924  0.56052  1.99579
##
## Random effects:
##  Groups   Name           Variance Std.Dev. Corr
##  PID      (Intercept)    509.896  22.581
##           intensity_rank   7.863   2.804   -0.96
##  Residual                 58.333   7.638
## Number of obs: 63, groups:  PID, 7
##
## Fixed effects:
```

```
##                 Estimate Std. Error t value
## (Intercept)      -38.873      8.789  -4.423
## intensity_rank     6.010      1.123   5.349
##
## Correlation of Fixed Effects:
##             (Intr)
## intnsty_rnk -0.948
```

```
# Does not work on LaTex
# sjt.lmer(lmm1b,
#           show.header = TRUE,
#           string.dv = "Response",
#           string.pred = "Coefficients",
#           depvar.labels = '',
#           pred.labels = 'intensity_rank',
#           string.est = 'Estimate',
#           string.ci = '95% CI',
#           string.p = 'p-value',
#           show.icc = FALSE,
#           show.r2 = FALSE)
```

2nd-order (quadratic) polynomial

```
# Intercept only
lmm2 <- lmer(tri_mean ~ poly(intensity_rank, 2) + (1 | PID),
             data = data_tmR,
             REML = TRUE)

# Intercept and slope
lmm2b <- lmer(tri_mean ~ poly(intensity_rank, 2) + (intensity_rank | PID),
              data = data_tmR,
              REML = TRUE)

# Better model?
anova(lmm2, lmm2b)
```

```
## Data: data_tmR
## Models:
## lmm2: tri_mean ~ poly(intensity_rank, 2) + (1 | PID)
## lmm2b: tri_mean ~ poly(intensity_rank, 2) + (intensity_rank | PID)
##       Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## lmm2   5 495.50 506.22 -242.75   485.50
## lmm2b  7 472.71 487.71 -229.35   458.71 26.793      2  1.521e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Anova for better model
Anova(lmm2b,
      type = 2,
      test.statistic = 'F')
```

```
## Analysis of Deviance Table (Type II Wald F tests with Kenward-Roger df)
##
```

```
## Response: tri_mean
##                             F Df Df.res    Pr(>F)
## poly(intensity_rank, 2) 15.85  2 13.527 0.0002848 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Print better model
summary(lmm2b)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: tri_mean ~ poly(intensity_rank, 2) + (intensity_rank | PID)
##    Data: data_tmR
##
## REML criterion at convergence: 441
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.10038 -0.64598  0.00681  0.50085  2.38158
##
## Random effects:
##  Groups   Name           Variance Std.Dev. Corr
##  PID      (Intercept)    512.02   22.628
##           intensity_rank   7.93    2.816   -0.96
##  Residual                 54.30    7.369
## Number of obs: 63, groups:  PID, 7
##
## Fixed effects:
##                          Estimate Std. Error t value
## (Intercept)                -8.825      3.902  -2.262
## poly(intensity_rank, 2)1  123.159     23.024   5.349
## poly(intensity_rank, 2)2   15.865      7.369   2.153
##
## Correlation of Fixed Effects:
##               (Intr) p(_,2)1
## ply(nt_,2)1 -0.695
## ply(nt_,2)2  0.000  0.000
```

```
# Doesn't work on LaTex
# sjt.lmer(lmm2b,
#           show.header = TRUE,
#           string.dv = "Response",
#           string.pred = "Coefficients",
#           depvar.labels = '',
#           pred.labels = 'intensity_rank',
#           string.est = 'Estimate',
#           string.ci = '95% CI',
#           string.p = 'p-value',
#           show.icc = FALSE,
#           show.r2 = FALSE)
```

3rd-order (cubic) polynomial

```
# Intercept only
lmm3 <- lmer(tri_mean ~ poly(intensity_rank, 3) + (1 | PID),
             data = data_tmR,
             REML = TRUE)

# Intercept and slope
lmm3b <- lmer(tri_mean ~ poly(intensity_rank, 3) + (intensity_rank | PID),
              data = data_tmR,
              REML = TRUE)

# Better model?
anova(lmm3, lmm3b)
```

```
## Data: data_tmR
## Models:
## lmm3: tri_mean ~ poly(intensity_rank, 3) + (1 | PID)
## lmm3b: tri_mean ~ poly(intensity_rank, 3) + (intensity_rank | PID)
##       Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## lmm3   6 497.33 510.19 -242.66   485.33
## lmm3b  8 474.37 491.52 -229.19   458.37 26.956      2  1.402e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Anova for better model
Anova(lmm3b,
      type = 2,
      test.statistic = 'F')
```

```
## Analysis of Deviance Table (Type II Wald F tests with Kenward-Roger df)
##
## Response: tri_mean
##                              F Df Df.res    Pr(>F)
## poly(intensity_rank, 3) 10.617  3 21.094 0.0001842 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Print better model
summary(lmm3b)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: tri_mean ~ poly(intensity_rank, 3) + (intensity_rank | PID)
##    Data: data_tmR
##
## REML criterion at convergence: 434.9
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.08782 -0.59452  0.01548  0.49782  2.36396
##
## Random effects:
##  Groups   Name           Variance Std.Dev. Corr
##  PID      (Intercept)    511.612  22.619
```

```
##           intensity_rank   7.917   2.814   -0.96
##  Residual                 55.081   7.422
## Number of obs: 63, groups:  PID, 7
##
## Fixed effects:
##                          Estimate Std. Error t value
## (Intercept)                -8.825      3.902  -2.262
## poly(intensity_rank, 3)1  123.159     23.024   5.349
## poly(intensity_rank, 3)2   15.865      7.422   2.138
## poly(intensity_rank, 3)3   -4.216      7.422  -0.568
##
## Correlation of Fixed Effects:
##              (Intr) p(_,3)1 p(_,3)2
## ply(nt_,3)1 -0.695
## ply(nt_,3)2  0.000   0.000
## ply(nt_,3)3  0.000   0.000   0.000
```

```
# Doesn't wotk with LaTex
# sjt.lmer(lmm3b,
#          show.header = TRUE,
#          string.dv = "Response",
#          string.pred = "Coefficients",
#          depvar.labels = '',
#          pred.labels = 'intensity_rank',
#          string.est = 'Estimate',
#          string.ci = '95% CI',
#          string.p = 'p-value',
#          show.icc = FALSE,
#          show.r2 = FALSE)
```

Compare models

```
knitr::kable(broom::tidy(anova(lmm1b, lmm2b, lmm3b)),
             caption = 'Linear model vs quadratic model and cubic model')
```

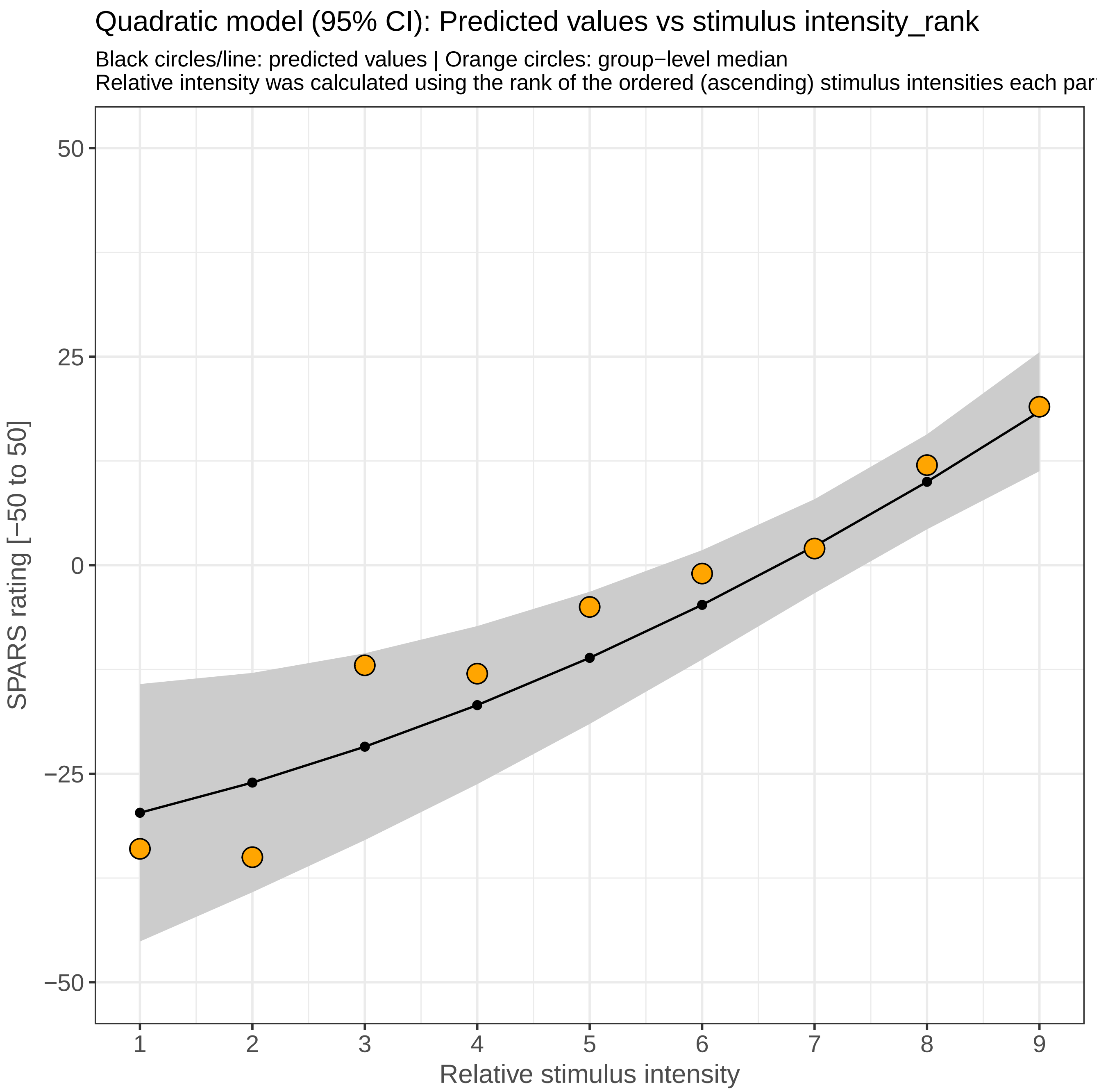
Table 2: Linear model vs quadratic model and cubic model

term	df	AIC	BIC	logLik	deviance	statistic	Chi.Df	p.value
lmm1b	6	475.2254	488.0842	-231.6127	463.2254	NA	NA	NA
lmm2b	7	472.7085	487.7104	-229.3542	458.7085	4.5169285	1	0.0335610
lmm3b	8	474.3731	491.5182	-229.1866	458.3731	0.3353413	1	0.5625307

Plot the model

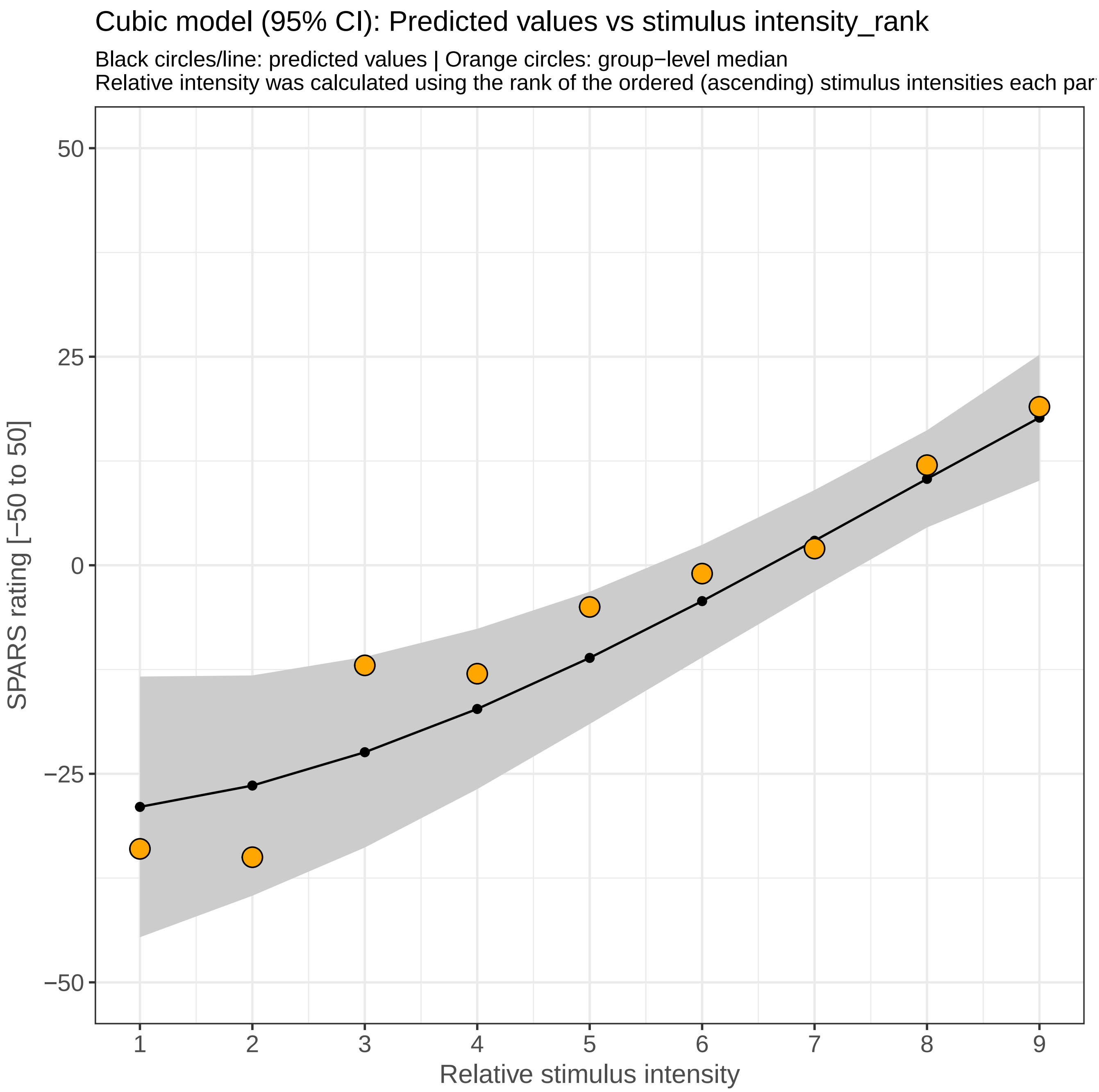
```
# Plot quadratic model
predicted <- ggeffects::ggpredict(model = lmm2b,
                                  terms = 'intensity_rank',
                                  ci.lvl = 0.95)
ggplot() +
```

```r
  geom_ribbon(data = predicted,
              aes(x = x,
                  ymin = conf.low,
                  ymax = conf.high),
              fill = '#cccccc') +
  geom_line(data = predicted,
            aes(x = x,
                y = predicted)) +
  geom_point(data = predicted,
            aes(x = x,
                y = predicted)) +
  geom_point(data = data_groupR,
             aes(x = intensity_rank,
                 y = median),
             shape = 21,
             size = 4,
             fill = '#FFA500') +
labs(title = 'Quadratic model (95% CI): Predicted values vs stimulus intensity_rank',
     subtitle = 'Black circles/line: predicted values | Orange circles: group-level medi
     x = 'Relative stimulus intensity',
     y = 'SPARS rating [-50 to 50]') +
scale_y_continuous(limits = c(-50, 50)) +
scale_x_continuous(breaks = seq(from = 1, to = 9, by = 1))
```




```r
# Plot cubic model
predicted_cubic <- ggeffects::ggpredict(model = lmm3b,
                                        terms = 'intensity_rank',
                                        ci.lvl = 0.95)

ggplot() +
    geom_ribbon(data = predicted_cubic,
                aes(x = x,
                    ymin = conf.low,
                    ymax = conf.high),
                fill = '#cccccc') +
    geom_line(data = predicted_cubic,
              aes(x = x,
                  y = predicted)) +
    geom_point(data = predicted_cubic,
               aes(x = x,
                   y = predicted)) +
```

```
    geom_point(data = data_groupR,
               aes(x = intensity_rank,
                   y = median),
               shape = 21,
               size = 4,
               fill = '#FFA500') +
  labs(title = 'Cubic model (95% CI): Predicted values vs stimulus intensity_rank',
       subtitle = 'Black circles/line: predicted values | Orange circles: group-level medi
       x = 'Relative stimulus intensity',
       y = 'SPARS rating [-50 to 50]') +
  scale_y_continuous(limits = c(-50, 50)) +
  scale_x_continuous(breaks = seq(from = 1, to = 9, by = 1))
```



The quadratic and cubic models were better fits than the linear model, and did not differ significantly from each other. Therefore we took the simpler of the two models (quadratic) for further inspection, performing diagnostics on the model to confirm that the model was properly specified.

Diagnostics on the quadratic model

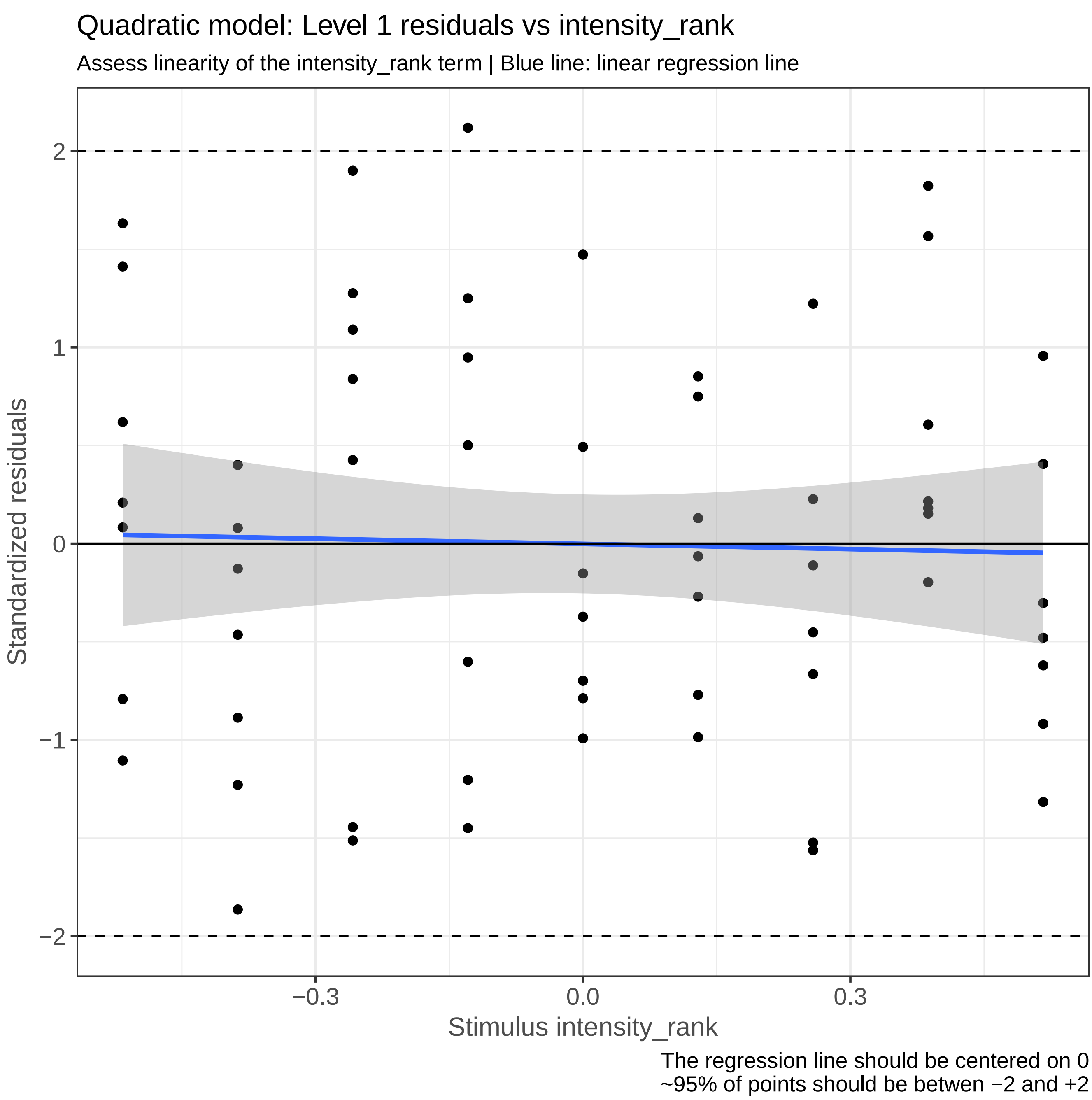
Generate residuals

```
# Level 1 residuals
## Standardized
lmm_resid1 <- HLMresid(lmm2b,
                       level = 1,
                       type = 'LS',
                       standardize = TRUE)

# Semi-standardized residuals (used for assessing homoscedasticity)
lmm_ssresid1 <- HLMresid(lmm2b,
                         level = 1,
                         type = 'LS',
                         standardize = 'semi')

# Level 2 residuals
## Standardized
lmm_resid2 <- HLMresid(lmm2b,
                       level = 'PID',
                       type = 'EB')
```

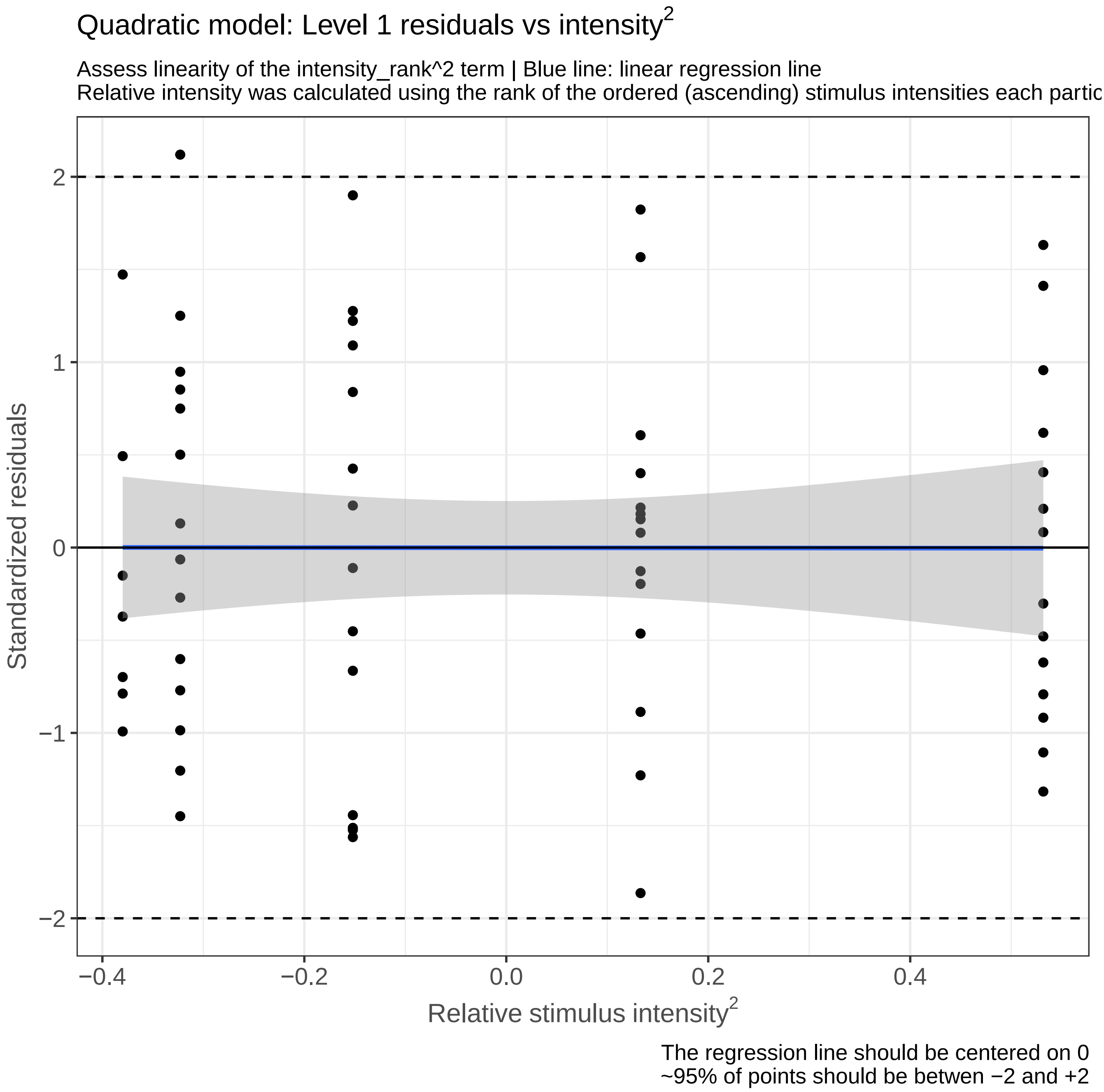
Level 1 residuals: linearity

The relationship between predictor(s) and outcome for a linear model should be linear. This relationship can be observed by plotting the level 1 standardized residuals against the predictors. The scatter of residuals should show no pattern, and be centred around 0.

```
# Standardized residuals vs intensity_rank
ggplot(data = lmm_resid1) +
    aes(x = `poly(intensity_rank, 2)`[, 1],
        y = std.resid) +
    geom_point() +
    geom_smooth(method = 'lm') +
    geom_hline(yintercept = 0) +
    geom_hline(yintercept = -2,
               linetype = 2) +
    geom_hline(yintercept = 2,
               linetype = 2) +
    labs(title = 'Quadratic model: Level 1 residuals vs intensity_rank',
         subtitle = 'Assess linearity of the intensity_rank term | Blue line: linear regre
         caption = 'The regression line should be centered on 0\n~95% of points should be
         y = 'Standardized residuals',
         x = 'Stimulus intensity_rank')
```




```r
# Standardized residuals vs intensity^2
ggplot(data = lmm_resid1) +
    aes(x = `poly(intensity_rank, 2)`[, 2],
        y = std.resid) +
    geom_point() +
    geom_smooth(method = 'lm') +
    geom_hline(yintercept = 0) +
    geom_hline(yintercept = -2,
               linetype = 2) +
    geom_hline(yintercept = 2,
               linetype = 2) +
    labs(title = expression(paste('Quadratic model: Level 1 residuals vs ', intensity^2)),
         subtitle = 'Assess linearity of the intensity_rank^2 term | Blue line: linear reg
         caption = 'The regression line should be centered on 0\n~95% of points should be
         y = 'Standardized residuals',
         x = expression(Relative~stimulus~intensity^2))
```



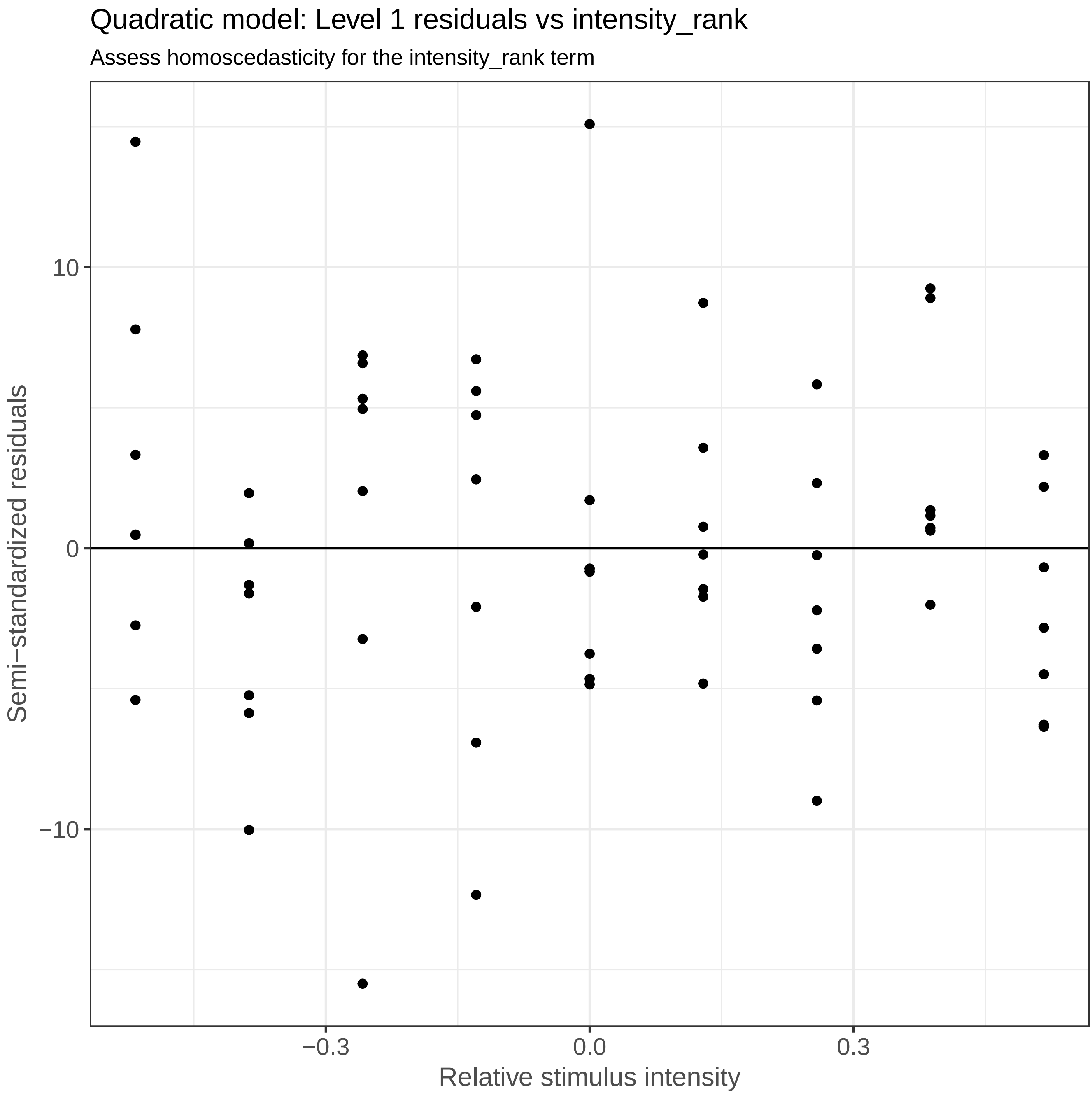
Based on the plot of the linear and quadratic terms' residuals, we accept that the condition of linearity for the quadratic model.

Level 1 residuals: homoscedasticity

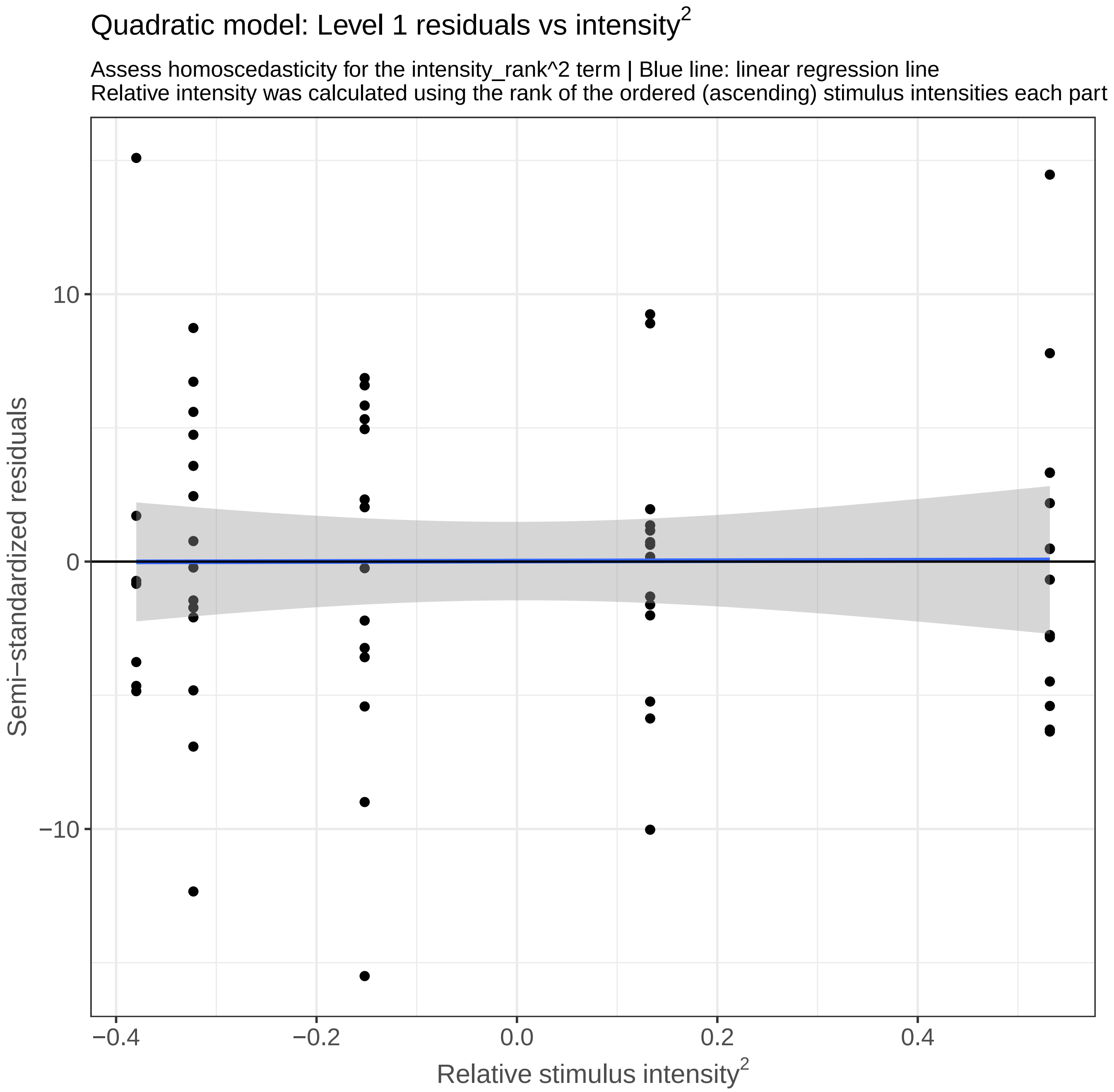
The variance of residuals should be constant across the range of the predictor(s). This relationship can be observed by plotting the level 1 semi-standardized residuals against the predictors. Like the assessment of linearity, the residuals should be centred on 0, and show no pattern in the scatter of points.

```
# Standardized residuals vs intensity_rank
ggplot(data = lmm_ssresid1) +
    aes(x = `poly(intensity_rank, 2)`[ ,1],
        y = semi.std.resid) +
    geom_point() +
    geom_hline(yintercept = 0) +
    labs(title = 'Quadratic model: Level 1 residuals vs intensity_rank',
         subtitle = 'Assess homoscedasticity for the intensity_rank term',
```

```
       y = 'Semi-standardized residuals',
       x = 'Relative stimulus intensity')
```




```
# Standardized residuals vs intensity^2
ggplot(data = lmm_ssresid1) +
    aes(x = `poly(intensity_rank, 2)`[, 2],
        y = semi.std.resid) +
    geom_point() +
    geom_smooth(method = 'lm') +
    geom_hline(yintercept = 0) +
    labs(title = expression(paste('Quadratic model: Level 1 residuals vs ', intensity^2)),
         subtitle = 'Assess homoscedasticity for the intensity_rank^2 term | Blue line: li
         y = 'Semi-standardized residuals',
         x = expression(Relative~stimulus~intensity^2))
```

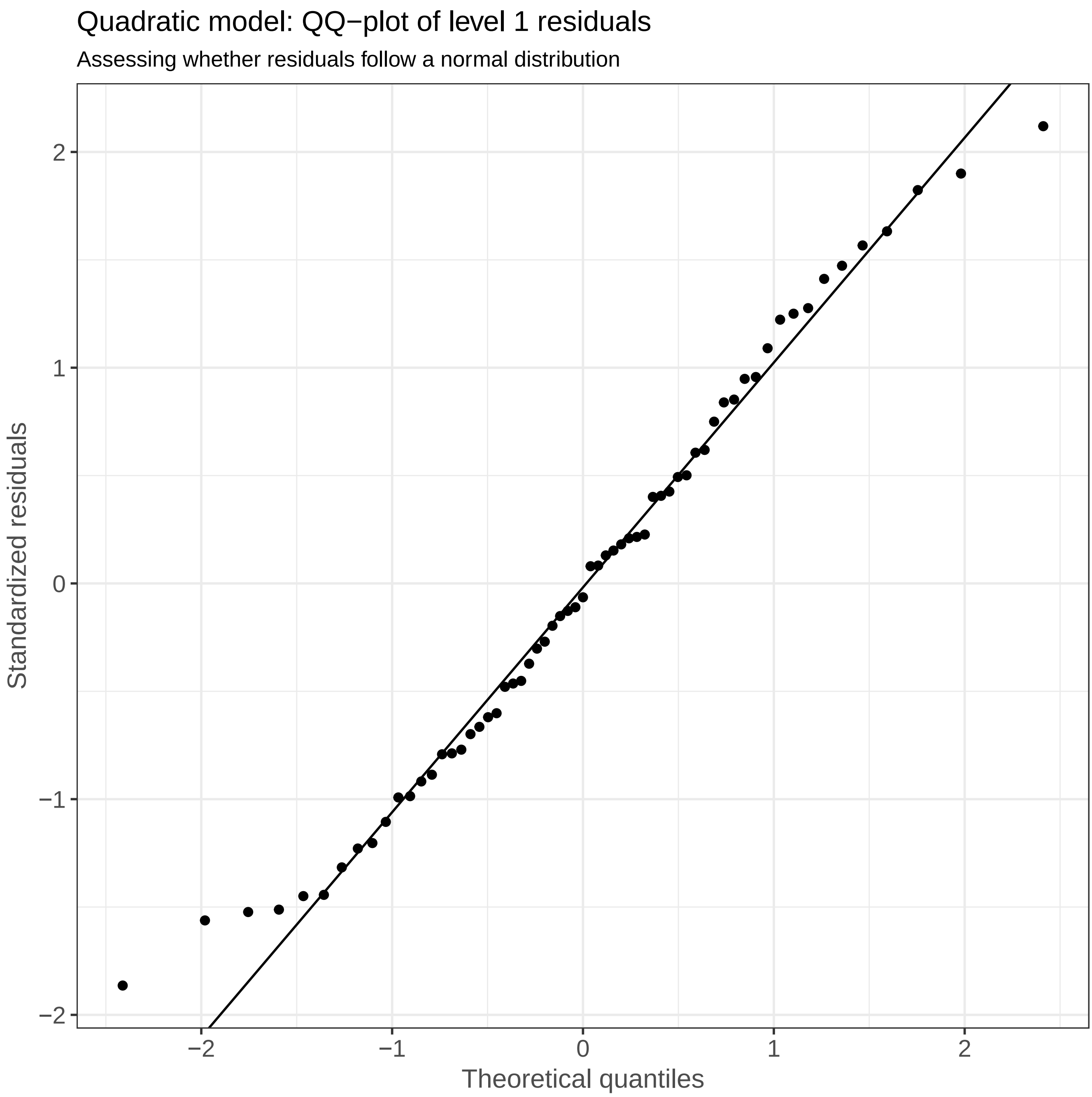


There is no obvious pattern to the scatter of residuals across any of the fixed effect terms. So we accept that the residuals are homoscedastic in the quadratic model.

Level 1 residuals: residual distribution

Residuals should be normally distributed. There are various methods of examining the distribution, and we have chosen the QQ-plot method, which plots the quantiles of the standardized residuals against a theoretical (Gaussian) quantile distribution. Points should line on the line of identity of the two sets of quantiles follow the same distribution.

```
# Standardized residuals vs intensity_rank
ggplot_qqnorm(x = lmm_resid1$std.resid,
              line = "rlm") +
    labs(title = 'Quadratic model: QQ-plot of level 1 residuals',
         subtitle = 'Assessing whether residuals follow a normal distribution',
         x = 'Theoretical quantiles',
         y = 'Standardized residuals')
```



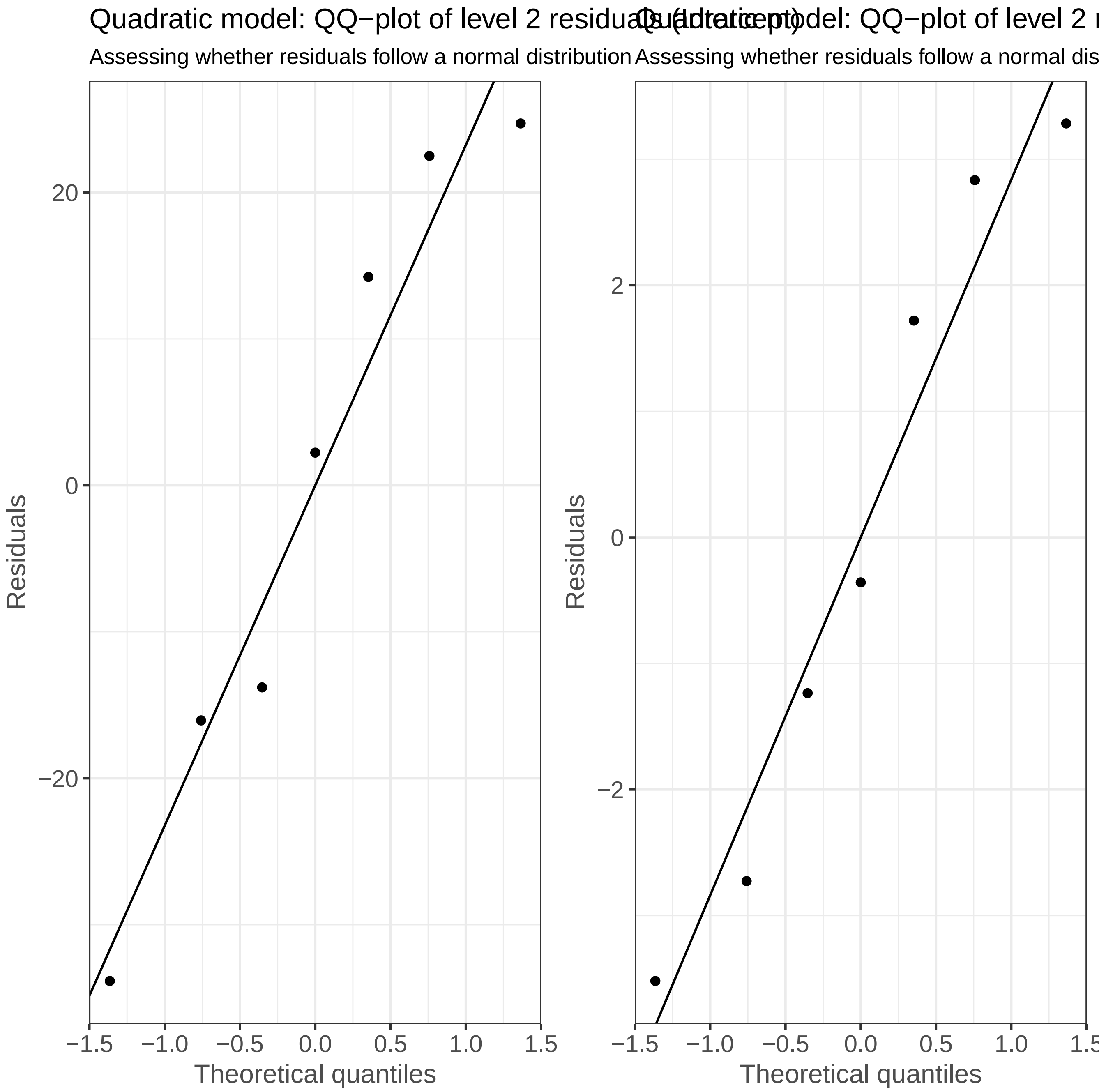
There is minor deviation at the extremes (possibly a thin left tail), but on the whole, we are satisfied that the quadratic model fits the assumption of normally distributed residuals.

Level 2 residuals: residual distribution

Level 2 residuals can be used to identify predictors that should be included in the model, but since we are only assessing the effect of stimulus strength on SPARS rating, we have only assessed whether the level 2 residuals (intercept and slope) meet the assumption of being normally distributed (assessed using QQ-plots).

```
# Generate QQplots
qq1 <- ggplot_qqnorm(x = lmm_resid2$`(Intercept)`,
              line = "rlm") +
    labs(title = 'Quadratic model: QQ-plot of level 2 residuals (Intercept)',
         subtitle = 'Assessing whether residuals follow a normal distribution',
         x = 'Theoretical quantiles',
         y = 'Residuals')

qq2 <- ggplot_qqnorm(x = lmm_resid2$intensity_rank,
```

```
                line = "rlm") +
    labs(title = 'Quadratic model: QQ-plot of level 2 residuals (slope: intensity_rank)',
         subtitle = 'Assessing whether residuals follow a normal distribution',
         x = 'Theoretical quantiles',
         y = 'Residuals')

# Plot
qq1 + qq2
```



Although the data are sparse, we are satisfied that the level 2 residuals for the intercept and the slope of the quadratic model fit the assumption of being normally distributed.

influence points

We assessed three aspects of influence (data that significantly model coefficients):

- The variance component (random effects) was assessed using the relative variance change metric, which calculates the impact of deleting observational units of the variance of the residuals, random intercept, random slope, and covariance of the random slope and random intercept.

- Leverage was used to assess fitted values. The assessment involves assessing the rate of change in the predicted response with respect to the observed response.
- Cook's Distance was used to assess the influence of fixed effects. The metric measures the distance between the fixed effects estimates obtained from the full model to that obtained from the reduced data (observations removed).

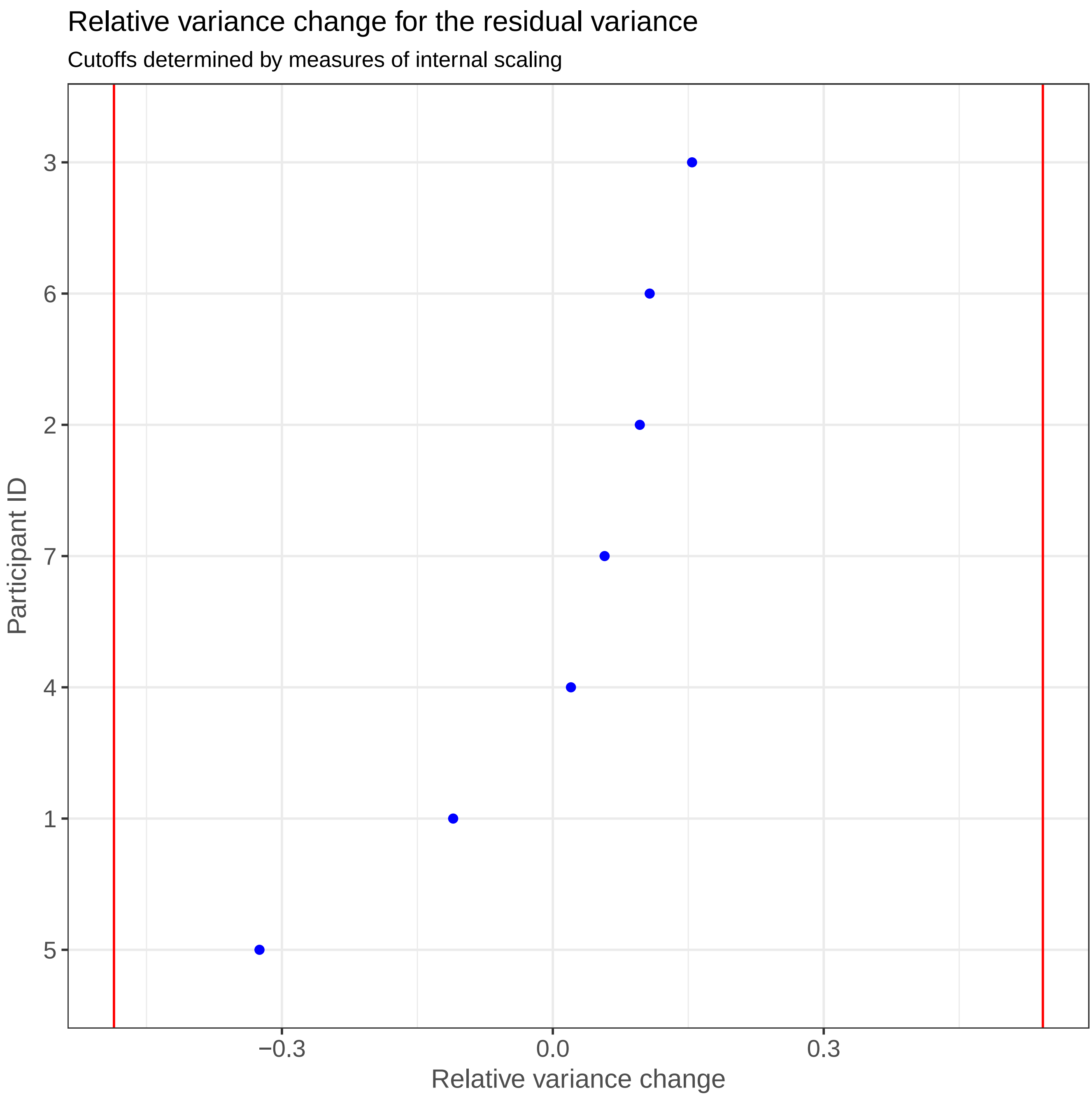
In all cases, we treated the individual (indicated using PID) as the unit of observation, and we used internal scaling to set the diagnostic cut-offs for each metric. The cut-offs were determined as: $3^{rd}\ Quartile + (3 \cdot IQR)$.

```
# Prepare relative variance change (RCV)
influence_rvc <- rvc(lmm1b,
                     group = 'PID')

# Prepare Cook's distance
influence_cooks <- cooks.distance(lmm1b,
                                  group = 'PID')

# Prepare leverage
## (Assessed at the level of PID, and not the individual observation)
influence_leverage <- leverage(lmm1b,
                               level = 'PID')
```

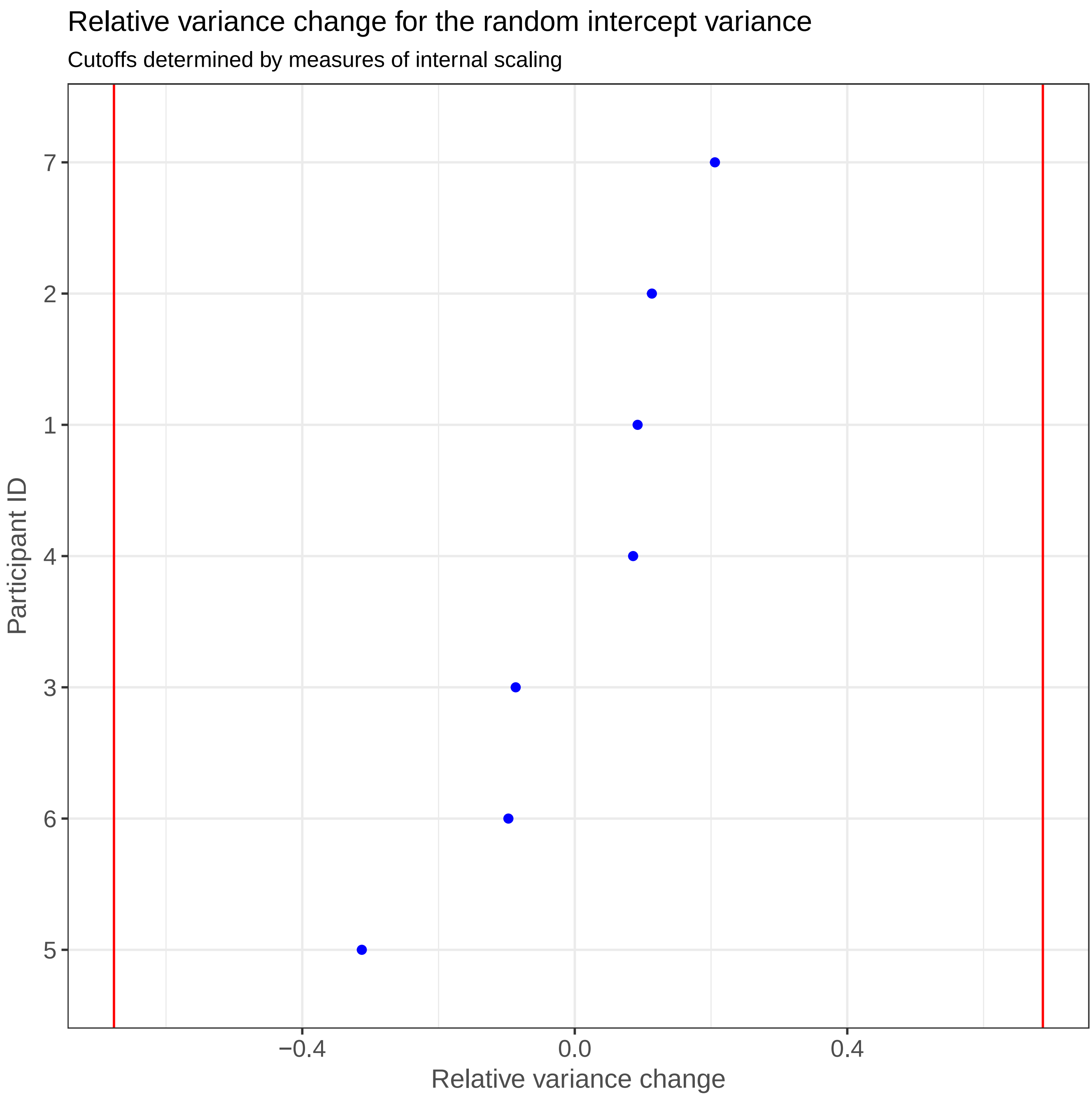
Random effects

Estimation of the variance component was undertaken by calculating relative variance change (RCV). RVC is close to zero when deletion of observational units from the model does not have a large influence on the variance component.

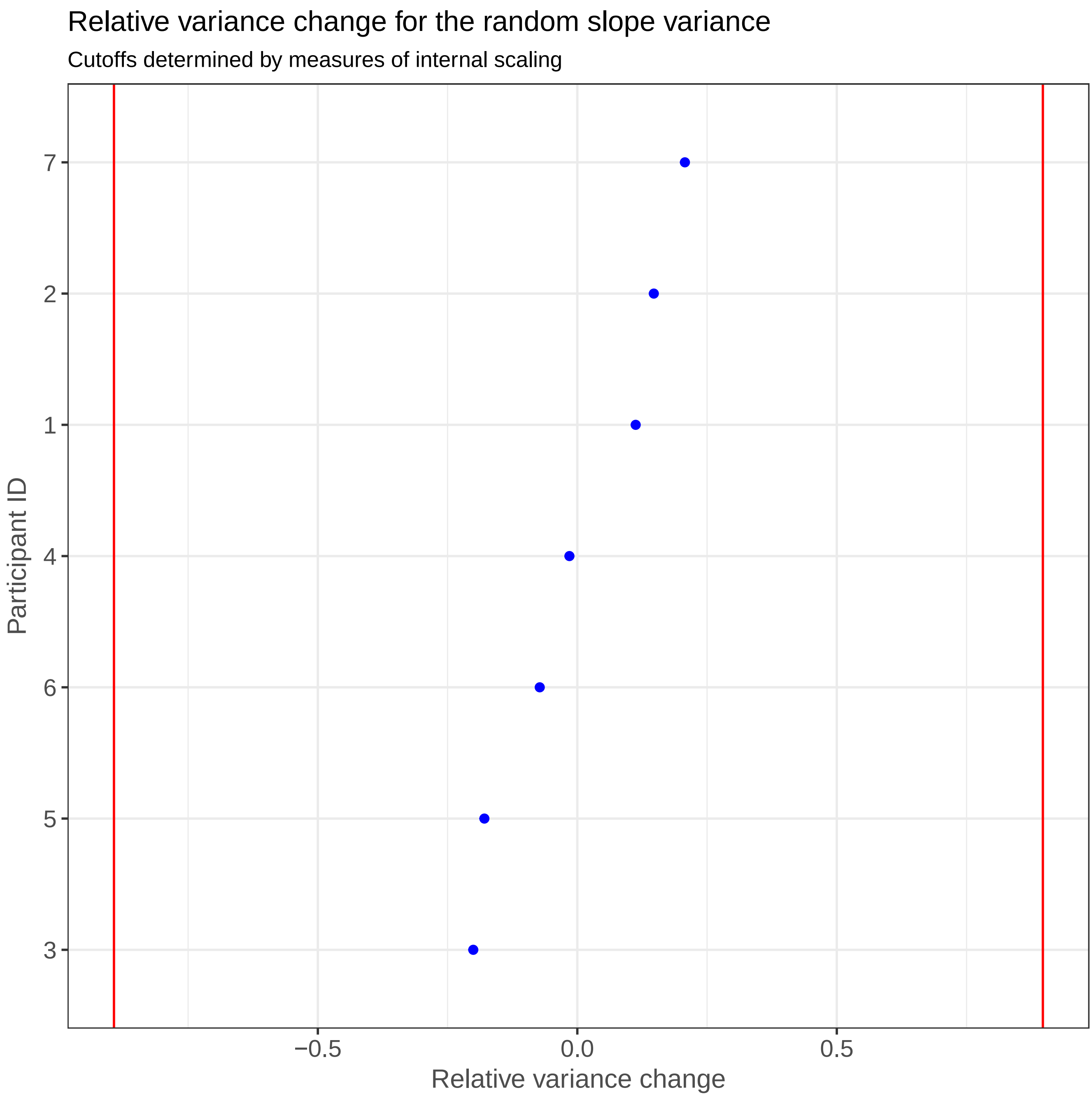
```
# Plot
dotplot_diag(x = influence_rvc[ , 1],
             cutoff = 'internal',
             name = 'rvc') +
    labs(title = 'Relative variance change for the residual variance',
         subtitle = 'Cutoffs determined by measures of internal scaling',
         y = 'Relative variance change',
         x = 'Participant ID')
```



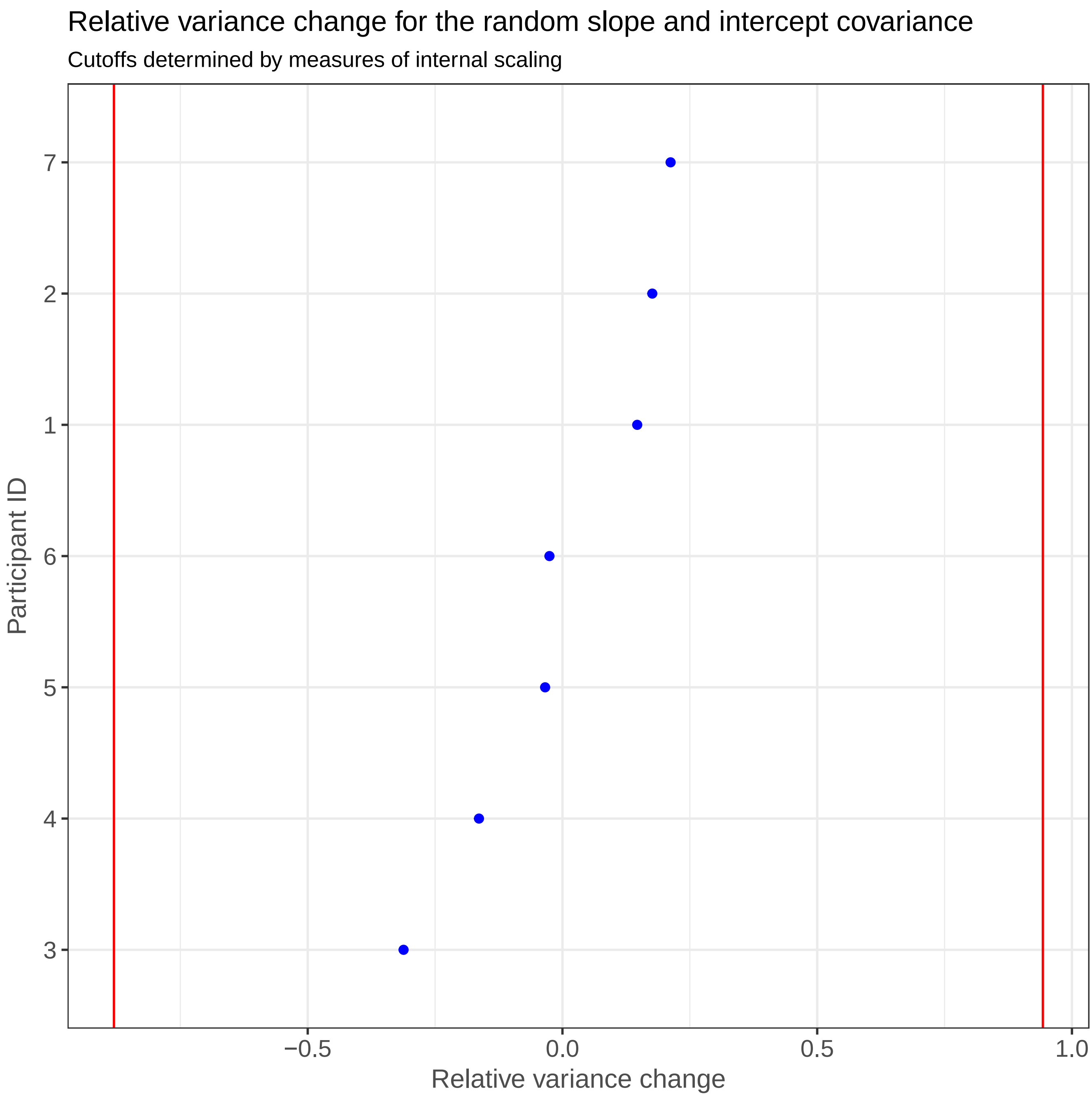

```
dotplot_diag(x = influence_rvc[ , 2],
             cutoff = 'internal',
             name = 'rvc') +
    labs(title = 'Relative variance change for the random intercept variance',
         subtitle = 'Cutoffs determined by measures of internal scaling',
         y = 'Relative variance change',
         x = 'Participant ID')
```




```
dotplot_diag(x = influence_rvc[ , 3],
             cutoff = 'internal',
             name = 'rvc') +
    labs(title = 'Relative variance change for the random slope variance',
         subtitle = 'Cutoffs determined by measures of internal scaling',
         y = 'Relative variance change',
         x = 'Participant ID')
```




```
dotplot_diag(x = influence_rvc[ , 4],
             cutoff = 'internal',
             name = 'rvc') +
    labs(title = 'Relative variance change for the random slope and intercept covariance',
         subtitle = 'Cutoffs determined by measures of internal scaling',
         y = 'Relative variance change',
         x = 'Participant ID')
```

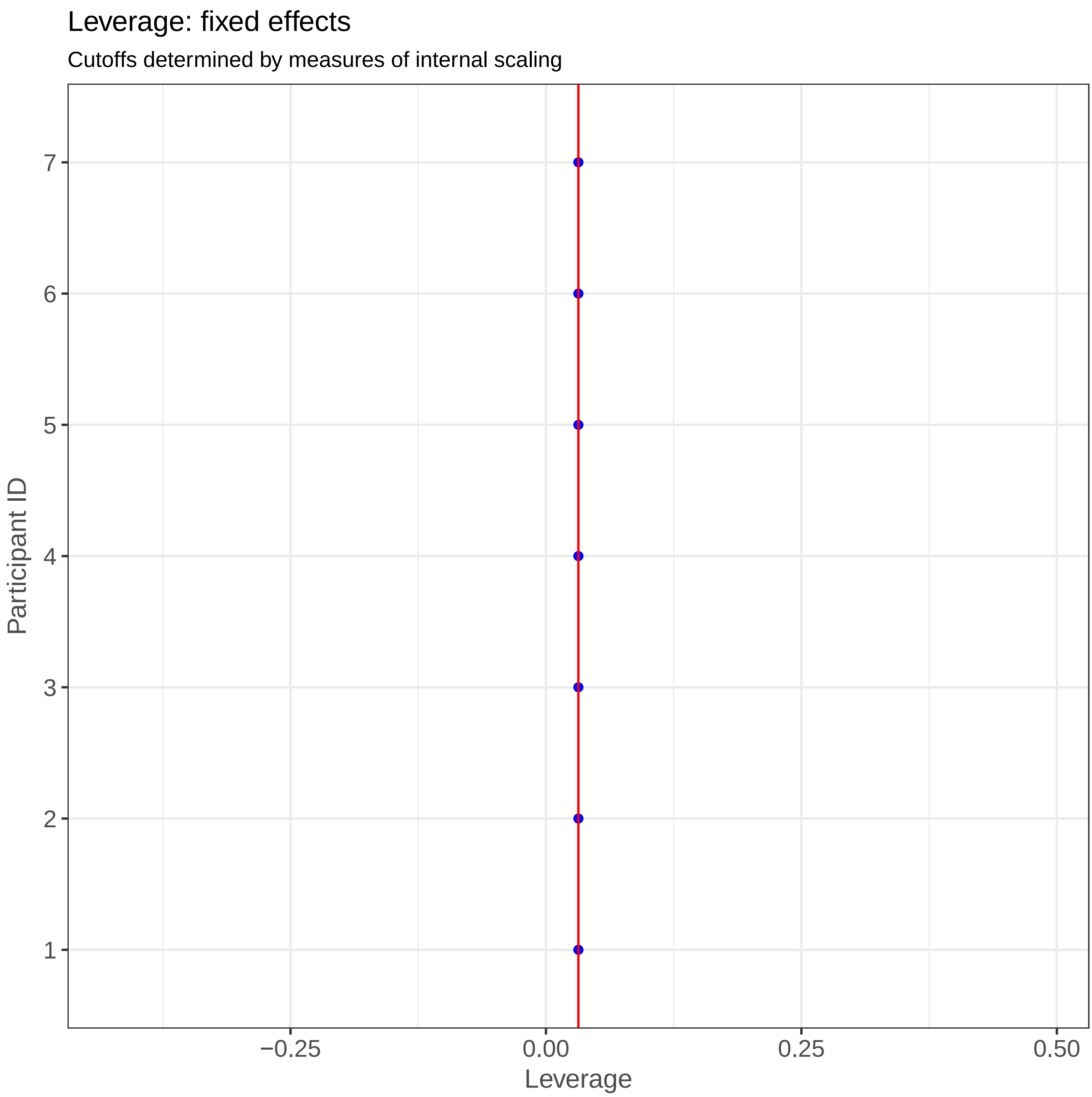


One value (PID11) is below the cut-off for the relative variance change for random slope and intercept covariance. The extent of the deviation is minor, and was ignored.

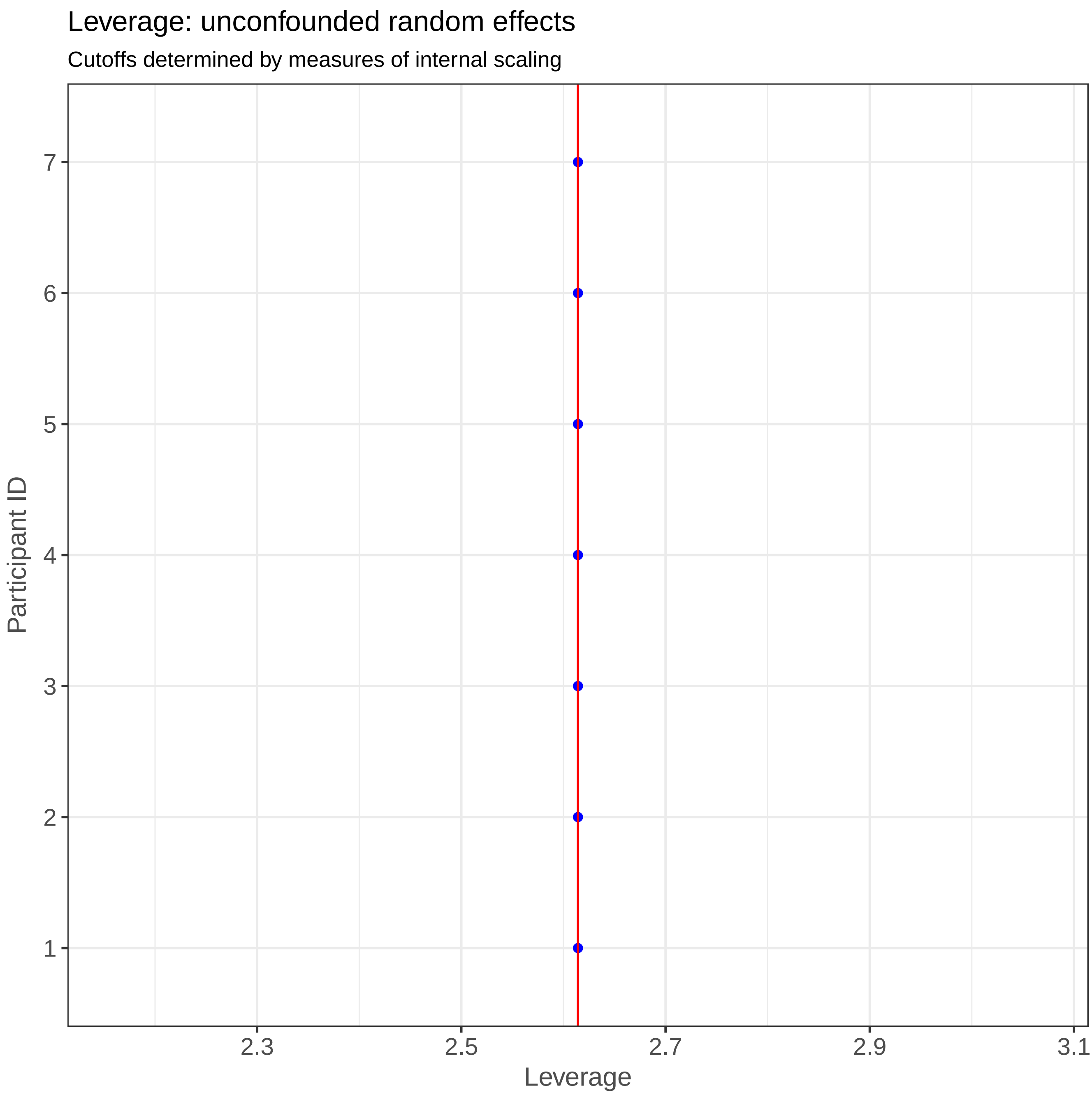
Fitted values

Assessing whether observations are unusual with regard to the fitted values and explanatory variables using leverage. We assessed leverage at two levels: i) fixed effects, and ii) unconfounded (by fixed effects) random effects.

```
dotplot_diag(x = influence_leverage[, 2],
             cutoff = "internal",
             name = "leverage") +
    labs(title = 'Leverage: fixed effects',
         subtitle = 'Cutoffs determined by measures of internal scaling',
         y = 'Leverage',
         x = 'Participant ID')
```



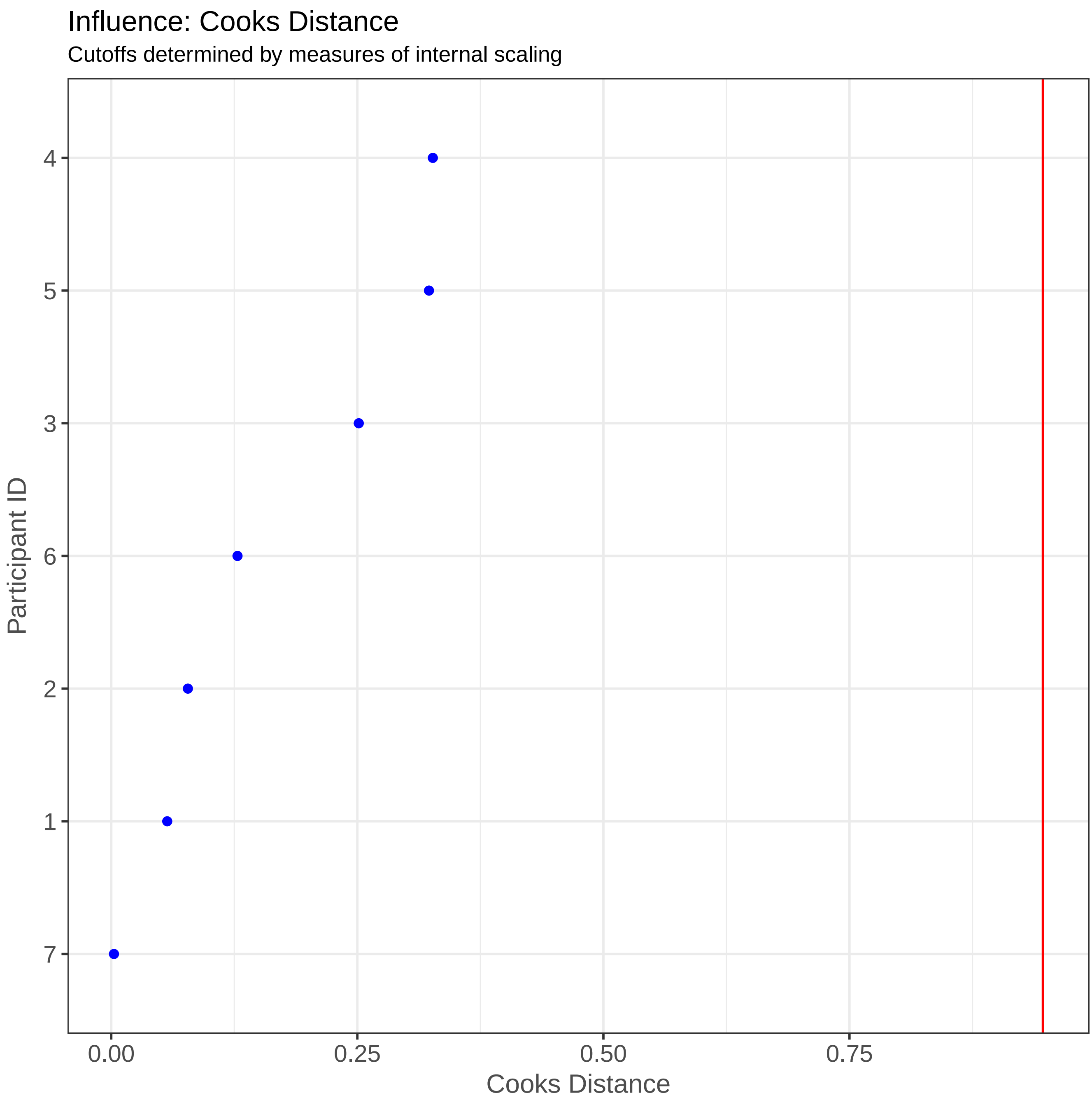

```r
dotplot_diag(x = influence_leverage[, 4],
             cutoff = "internal",
             name = "leverage") +
    labs(title = 'Leverage: unconfounded random effects',
         subtitle = 'Cutoffs determined by measures of internal scaling',
         y = 'Leverage',
         x = 'Participant ID')
```



Fixed effects

Influence points were assessed by calculating Cook's Distance metrics.

```
# Plot data
dotplot_diag(x = influence_cooks,
             cutoff = "internal",
             name = "cooks.distance") +
    labs(title = 'Influence: Cooks Distance',
         subtitle = 'Cutoffs determined by measures of internal scaling',
         y = 'Cooks Distance',
         x = 'Participant ID')
```



There are no influential fixed effects.

Summary

The linear is well-specified.

Quantile mixed model regression

```
# Quantile model with 2.5, 25, 50, 75, and 97.5% quantiles
qmm <- lqmm(fixed = tri_mean ~ poly(intensity_rank, 2),
            random = ~ intensity_rank,
            group = PID,
            data = data_tmR,
```

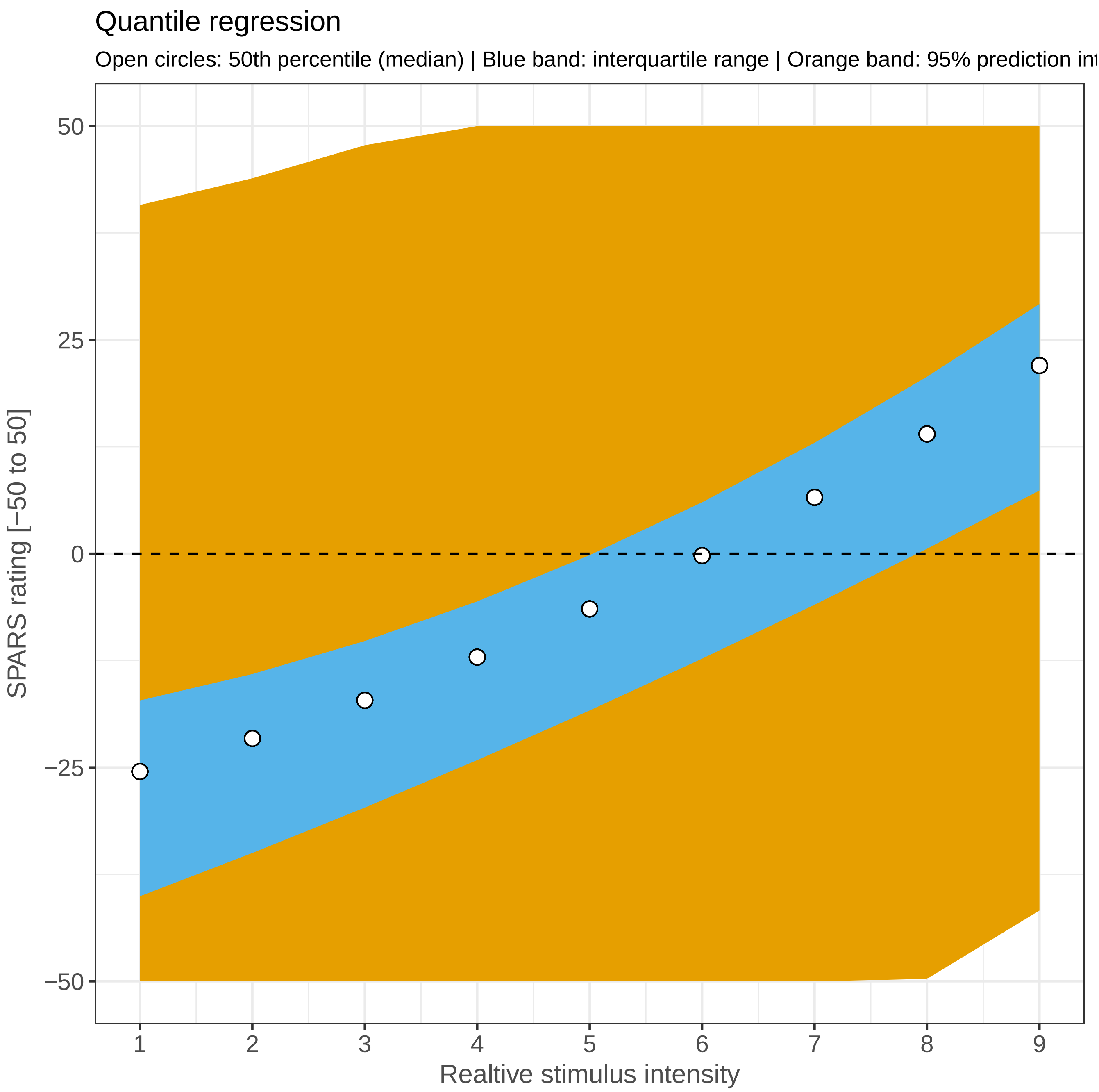
```r
            tau = c(0.025, 0.25, 0.5, 0.75, 0.975))

# Summary
summary(qmm)
```

```
## Call: lqmm(fixed = tri_mean ~ poly(intensity_rank, 2), random = ~intensity_rank,
##     group = PID, tau = c(0.025, 0.25, 0.5, 0.75, 0.975), data = data_tmR)
##
## tau = 0.025
##
## Fixed effects:
##                              Value Std. Error lower bound upper bound
## (Intercept)                -69.078     22.952    -115.203     -22.954
## poly(intensity_rank, 2)1   128.368     17.026      94.153     162.582
## poly(intensity_rank, 2)2    11.399     13.790     -16.314      39.111
##                            Pr(>|t|)
## (Intercept)                0.004124 **
## poly(intensity_rank, 2)1 9.724e-10 ***
## poly(intensity_rank, 2)2   0.412484
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## tau = 0.25
##
## Fixed effects:
##                              Value Std. Error lower bound upper bound
## (Intercept)               -17.4995     9.6694    -36.9308      1.9318
## poly(intensity_rank, 2)1 121.5351    17.4512     86.4657    156.6046
## poly(intensity_rank, 2)2   5.7801    14.3579    -23.0732     34.6333
##                            Pr(>|t|)
## (Intercept)                 0.07646 .
## poly(intensity_rank, 2)1 7.535e-09 ***
## poly(intensity_rank, 2)2    0.68902
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## tau = 0.5
##
## Fixed effects:
##                              Value Std. Error lower bound upper bound
## (Intercept)                -4.4901     6.8584    -18.2725      9.2923
## poly(intensity_rank, 2)1 121.6123    18.4323     84.5712    158.6534
## poly(intensity_rank, 2)2  13.7000    13.7295    -13.8904     41.2904
##                            Pr(>|t|)
## (Intercept)                  0.5157
## poly(intensity_rank, 2)1 2.786e-08 ***
## poly(intensity_rank, 2)2     0.3233
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## tau = 0.75
##
```

```
## Fixed effects:
##                              Value Std. Error lower bound upper bound
## (Intercept)                 2.4089     7.5663    -12.7962      17.614
## poly(intensity_rank, 2)1 118.7761    23.0832     72.3886     165.164
## poly(intensity_rank, 2)2  17.9548    16.0914    -14.3821      50.292
##                            Pr(>|t|)
## (Intercept)                  0.7516
## poly(intensity_rank, 2)1 4.687e-06 ***
## poly(intensity_rank, 2)2    0.2699
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## tau = 0.975
##
## Fixed effects:
##                              Value Std. Error lower bound upper bound
## (Intercept)                60.2116    15.5247     29.0135      91.410
## poly(intensity_rank, 2)1 117.4004    18.1633     80.8998     153.901
## poly(intensity_rank, 2)2  17.2735    12.9651     -8.7808      43.328
##                            Pr(>|t|)
## (Intercept)               0.0003135 ***
## poly(intensity_rank, 2)1 4.497e-08 ***
## poly(intensity_rank, 2)2 0.1889215
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## AIC:
## [1] 585.5 (df = 6) 509.7 (df = 6) 522.3 (df = 6) 524.7 (df = 6)
## [5] 562.1 (df = 6)
```

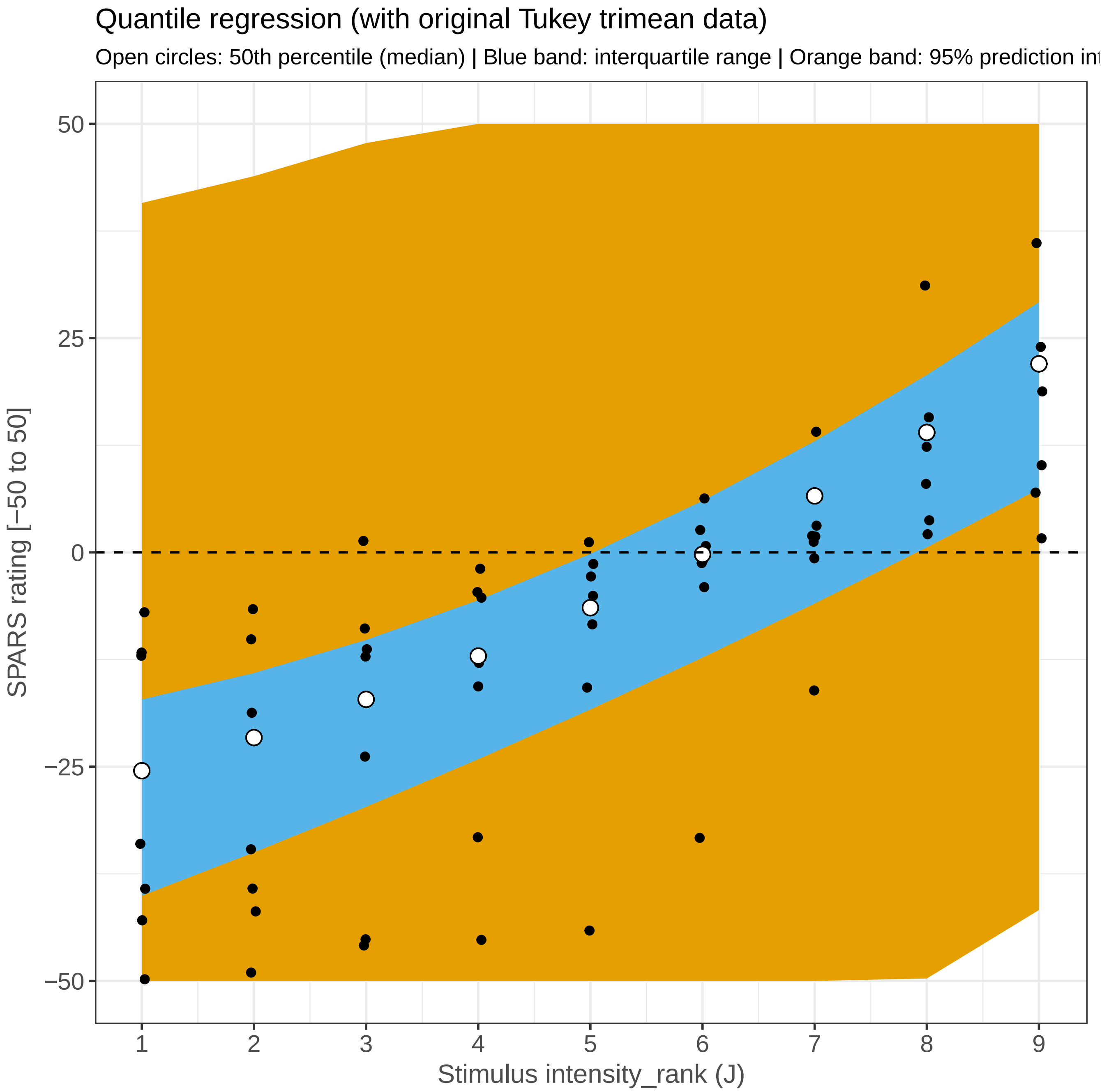
```r
# Get predicted values
## Level 0 (conditional, note difference to the lmer diagnostics)
quant_predict <- as.data.frame(predict(qmm, level = 0))
names(quant_predict) <- paste0('Q', c(2.5, 25, 50, 75, 97.5))

# Join with 'central_lmm'
data_lqmm <- data_tmR %>%
  bind_cols(quant_predict)

# Trim prediction to upper and lower limits of the scale
data_lqmm %<>%
  mutate_if(is.numeric,
            funs(ifelse(. > 50,
                        yes = 50,
                        no = ifelse(. < -50,
                                    yes = -50,
                                    no = .))))

# Plot
ggplot(data = data_lqmm) +
  aes(x = intensity_rank,
      y = Q50) +
  geom_ribbon(aes(ymin = `Q2.5`,
```

```
                ymax = `Q97.5`),
            fill = '#E69F00') +
  geom_ribbon(aes(ymin = `Q25`,
                ymax = `Q75`),
            fill = '#56B4E9') +
  geom_point(size = 3,
           shape = 21,
           fill = '#FFFFFF',
           colour = '#000000') +
  geom_hline(yintercept = 0,
           linetype = 2) +
  labs(title = paste('Quantile regression'),
       subtitle = 'Open circles: 50th percentile (median) | Blue band: interquartile range
       x = 'Realtive stimulus intensity',
       y = 'SPARS rating [-50 to 50]') +
  scale_y_continuous(limits = c(-50, 50)) +
  scale_x_continuous(breaks = unique(data_lqmm$intensity_rank))
```




```r
## With original data
ggplot(data = data_lqmm) +
  aes(x = intensity_rank,
      y = Q50) +
  geom_ribbon(aes(ymin = `Q2.5`,
                  ymax = `Q97.5`),
              fill = '#E69F00') +
  geom_ribbon(aes(ymin = `Q25`,
                  ymax = `Q75`),
              fill = '#56B4E9') +
  geom_point(data = data_tmR,
             aes(y = tri_mean),
             position = position_jitter(width = 0.03)) +
  geom_point(size = 3,
             shape = 21,
             fill = '#FFFFFF',
             colour = '#000000') +
```

```r
  geom_hline(yintercept = 0,
             linetype = 2) +
  labs(title = paste('Quantile regression (with original Tukey trimean data)'),
       subtitle = 'Open circles: 50th percentile (median) | Blue band: interquartile range
       x = 'Stimulus intensity_rank (J)',
       y = 'SPARS rating [-50 to 50]') +
  scale_y_continuous(limits = c(-50, 50)) +
  scale_x_continuous(breaks = unique(data_lqmm$intensity_rank))
```



The response is consistent across the range of stimulus intensities, but the prediction interval is extremely broad.

Session information

```r
sessionInfo()
```

```
## R version 3.5.0 (2018-04-23)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS High Sierra 10.13.5
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] bindrcpp_0.2.2     car_3.0-0          carData_3.0-1
##  [4] sjPlot_2.4.1       HLMdiag_0.3.1      lqmm_1.5.4
##  [7] lme4_1.1-17        Matrix_1.2-14      patchwork_0.0.1
## [10] forcats_0.3.0      stringr_1.3.1      dplyr_0.7.5
## [13] purrr_0.2.5        readr_1.1.1        tidyr_0.8.1
## [16] tibble_1.4.2       ggplot2_2.2.1.9000 tidyverse_1.2.1
## [19] magrittr_1.5
##
## loaded via a namespace (and not attached):
##   [1] TH.data_1.0-8      minqa_1.2.4        colorspace_1.3-2
##   [4] rio_0.5.10         modeltools_0.2-21  ggridges_0.5.0
##   [7] sjlabelled_1.0.11  rprojroot_1.3-2    estimability_1.3
##  [10] snakecase_0.9.1    rstudioapi_0.7     glmmTMB_0.2.1.0
##  [13] DT_0.4             mvtnorm_1.0-8      lubridate_1.7.4
##  [16] coin_1.2-2         xml2_1.2.0         codetools_0.2-15
##  [19] splines_3.5.0      mnormt_1.5-5       knitr_1.20
##  [22] sjmisc_2.7.2       effects_4.0-1      bayesplot_1.5.0
##  [25] jsonlite_1.5       nloptr_1.0.4       ggeffects_0.3.4
##  [28] pbkrtest_0.4-7     broom_0.4.4        shiny_1.1.0
##  [31] compiler_3.5.0     httr_1.3.1         sjstats_0.15.0
##  [34] emmeans_1.2.1      backports_1.1.2    assertthat_0.2.0
##  [37] lazyeval_0.2.1     survey_3.33-2      cli_1.0.0
##  [40] later_0.7.3        htmltools_0.3.6    tools_3.5.0
##  [43] SparseGrid_0.8.2   coda_0.19-1        gtable_0.2.0
##  [46] glue_1.2.0         reshape2_1.4.3     merTools_0.4.1
##  [49] Rcpp_0.12.17       cellranger_1.1.0   nlme_3.1-137
##  [52] psych_1.8.4        lmtest_0.9-36      openxlsx_4.1.0
##  [55] rvest_0.3.2        mime_0.5           stringdist_0.9.5.1
##  [58] MASS_7.3-50        zoo_1.8-1          scales_0.5.0.9000
##  [61] promises_1.0.1     hms_0.4.2          parallel_3.5.0
##  [64] sandwich_2.4-0     pwr_1.2-2          TMB_1.7.13
##  [67] curl_3.2           yaml_2.1.19        stringi_1.2.2
##  [70] highr_0.6          blme_1.0-4         zip_1.0.0
```

```
##  [73] rlang_0.2.1         pkgconfig_2.0.1     arm_1.10-1
##  [76] evaluate_0.10.1     lattice_0.20-35     prediction_0.3.6
##  [79] bindr_0.1.1         labeling_0.3        htmlwidgets_1.2
##  [82] tidyselect_0.2.4    plyr_1.8.4          R6_2.2.2
##  [85] multcomp_1.4-8      RLRsim_3.1-3        pillar_1.2.3
##  [88] haven_1.1.1         foreign_0.8-70      withr_2.1.2
##  [91] mgcv_1.8-23         survival_2.42-3     abind_1.4-5
##  [94] nnet_7.3-12         modelr_0.1.2        crayon_1.3.4
##  [97] rmarkdown_1.9       grid_3.5.0          readxl_1.1.0
## [100] data.table_1.11.4   digest_0.6.15       xtable_1.8-2
## [103] httpuv_1.4.3        stats4_3.5.0        munsell_0.4.3
```