DFNA9 genotype ~ phenotype analysis

Cris Lanting

22/04/2020

Introduction

This R Markdown notebook is used to document the various aspects of the genotype-phenotype analysis in many subjects with hereditary hearing loss based on mutation in the DFNA9 gene. We have data collected [. . .]

This notebooks is intended to leave a trail of the analyses done en to make it more reprodicible. It now covers the data cleaning, description of the data (group size, how many subjects per group, how many audiograms per subject), plots of the hearing thresholds across age and other descriptors of the data. The next step is to

Load R-packages

```
library(ggplot2)
library(ggthemr)
library(drc)
```

```
## Loading required package: MASS

##
## 'drc' has been loaded.

## Please cite R and 'drc' if used for a publication,

## for references type 'citation()' and 'citation('drc')'.

##
## Attaching package: 'drc'

## The following objects are masked from 'package:stats':
##
##     gaussian, getInitial
```

```
library(sjPlot)
```

```
## Registered S3 methods overwritten by 'lme4':
##   method                          from
##   cooks.distance.influence.merMod car
##   influence.merMod                car
##   dfbeta.influence.merMod         car
##   dfbetas.influence.merMod        car

## Learn more about sjPlot with 'browseVignettes("sjPlot")'.
```

```
library("readxl")
library("nlme")
library(lme4)
```

```
## Loading required package: Matrix

##
## Attaching package: 'lme4'

## The following object is masked from 'package:nlme':
##
##     lmList
```

```
library(knitr)
library(kableExtra)
library(forcats)
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:kableExtra':
##
##     group_rows

## The following object is masked from 'package:nlme':
##
##     collapse

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
ggthemr('fresh')
```

Load data and clean data-frames

Load data from Excel file and select only relevant columns/rows. The first analyses will be based on pure-tone average (PTA). The selected subset dataframe consists of the columns patient id (pid), group, age (Leeftijd), and the PTA (PTA54ADS).

```
data_raw <- read_excel("../data/raw_data/database_20-04-2020.xlsx")
data_raw$group = factor(data_raw$Domain)
#leave out data with only n=1 dataset per domain/certain unpublished data.
data_subset <-
  subset(data_raw, Smits == 'no' & Domainrec != 1 & group != "Ivd1")
data <-
```

Table 1: Table 1. The number of subjects per group

group	# subjects
LCCL	251
vWFA1	12
vWFA2	20

```r
  subset(data_subset, select = c('pid', 'group', 'Leeftijd', 'PTA54ADS'))

#drop unused levels from a factor in a dataframe, e.g. groups that have no entries anymore.
data <- droplevels(data)
# save processed and cleaned data
save(data,file="../data/processed_data/data_pta_age_group.Rda")

#check for NaNs in PTA and Leeftijd, should be 0.
nrow(data[is.na(data$PTA54ADS) | is.na(data$Leeftijd),])
```

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

Group description

In the group of DFNA9 patients we have some for which there is longitudinal data, i.e. multiple audiograms over time/age (Leeftijd). How many subjects are there for each group?

```r
t1 <- data %>%                          # take the data.frame "data"
  filter(!is.na(pid)) %>%     # Using "data", filter out all rows with NAs in aa
  group_by(group) %>%         # Then, with the filtered data, group it by "group"
  summarise("# subjects" = n_distinct(pid))    # Now summarise with unique elements per group
kable(t1, caption = "Table 1. The number of subjects per group",) %>%
  kable_styling(bootstrap_options = "striped", full_width = F)
```

Next, create a table and histogram of number of measurements for each subject id (pid) across the groups

```r
num_meas_per_id <-
  aggregate(PTA54ADS ~ pid , data, function(x)
    length(unique(x)))
t2 <- table(num_meas_per_id$PTA54ADS)
```

In total there are 283 subjects with 716 measurements; 159 patients with only 1 measurement and 123 patients with 2 or more measurements, see e.g. table 1 or the histogram.

```r
kable(t2,
      caption = "Table 2. The number of subjects that each have n audiograms",
      col.names = c("# audiograms", "# subjects")) %>%
  kable_styling(bootstrap_options = "striped", full_width = F)
```

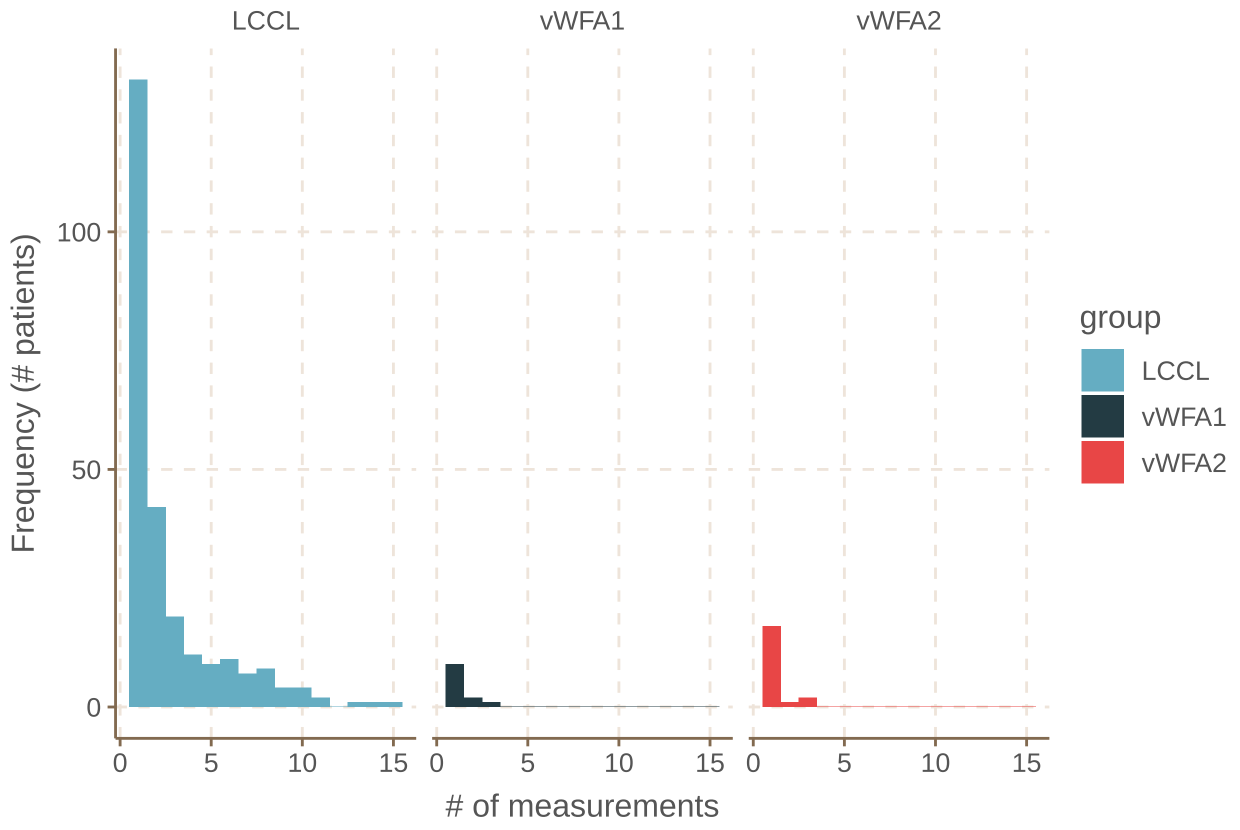
Now make a histogram of the number of audiograms across patients in each of the groups

```r
#number (n) of counts (i.e. audiograms) per subject (pid)
summarytable <- data %>% count(group, pid)

ggplot(data = summarytable, aes(x = n, fill = group)) +
  geom_histogram(binwidth = 1) +
  facet_wrap(~ group) +
  xlab("# of measurements") +
```

Table 2: Table 2. The number of subjects that each have n audiograms

# audiograms	# subjects
1	159
2	47
3	21
4	10
5	13
6	11
7	4
8	7
9	3
10	3
11	3
12	1
15	1

```
  ylab("Frequency (# patients)")
```

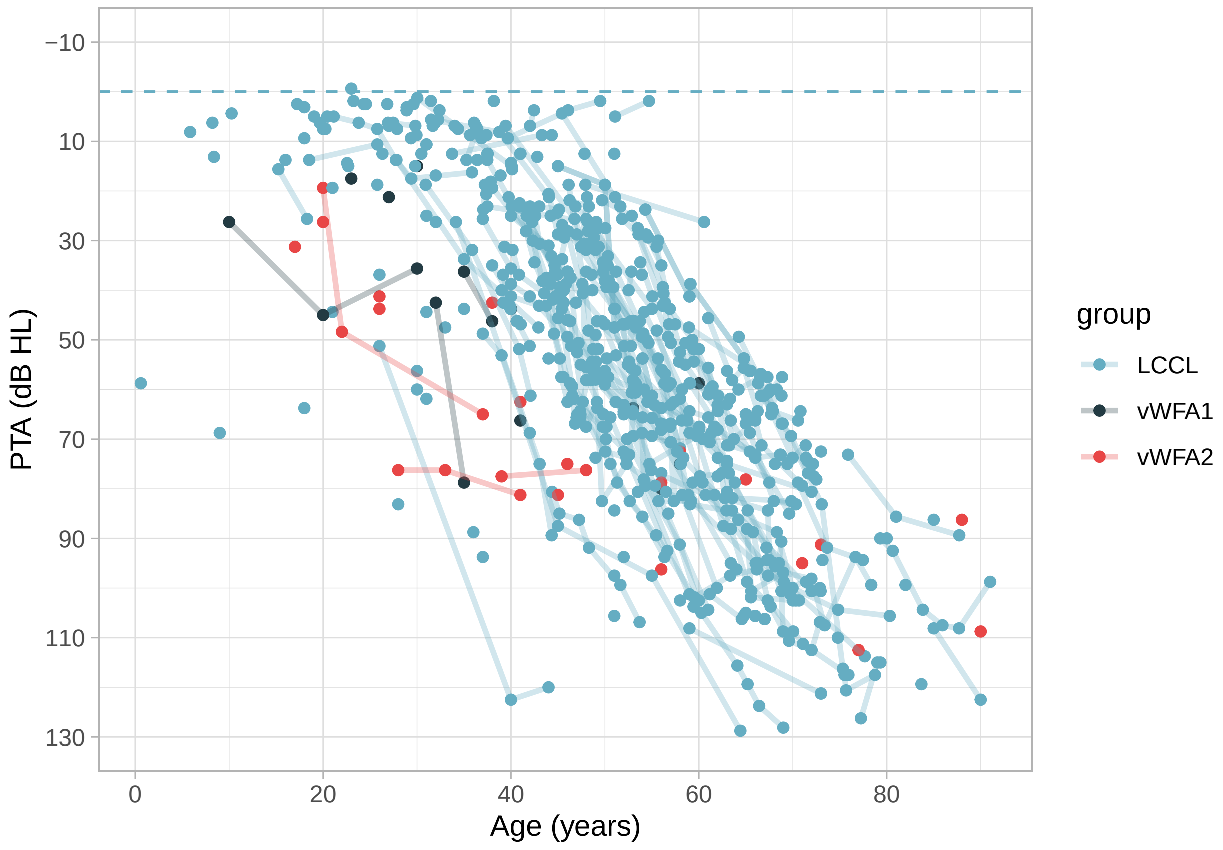


```
dev.print(pdf, '../results/histogram_number_meas_pid.pdf')
```

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

Relation of PTA with age for the different groups; connecting lines show longitudinal data of patients' PTA over time

```
ggplot(data, aes(
  x = Leeftijd,
  y = PTA54ADS,
  group = pid,
  color = group
)) +
  geom_point(aes(colour = factor(group))) +
  geom_line(data = data, size = 1, alpha = .3) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
```



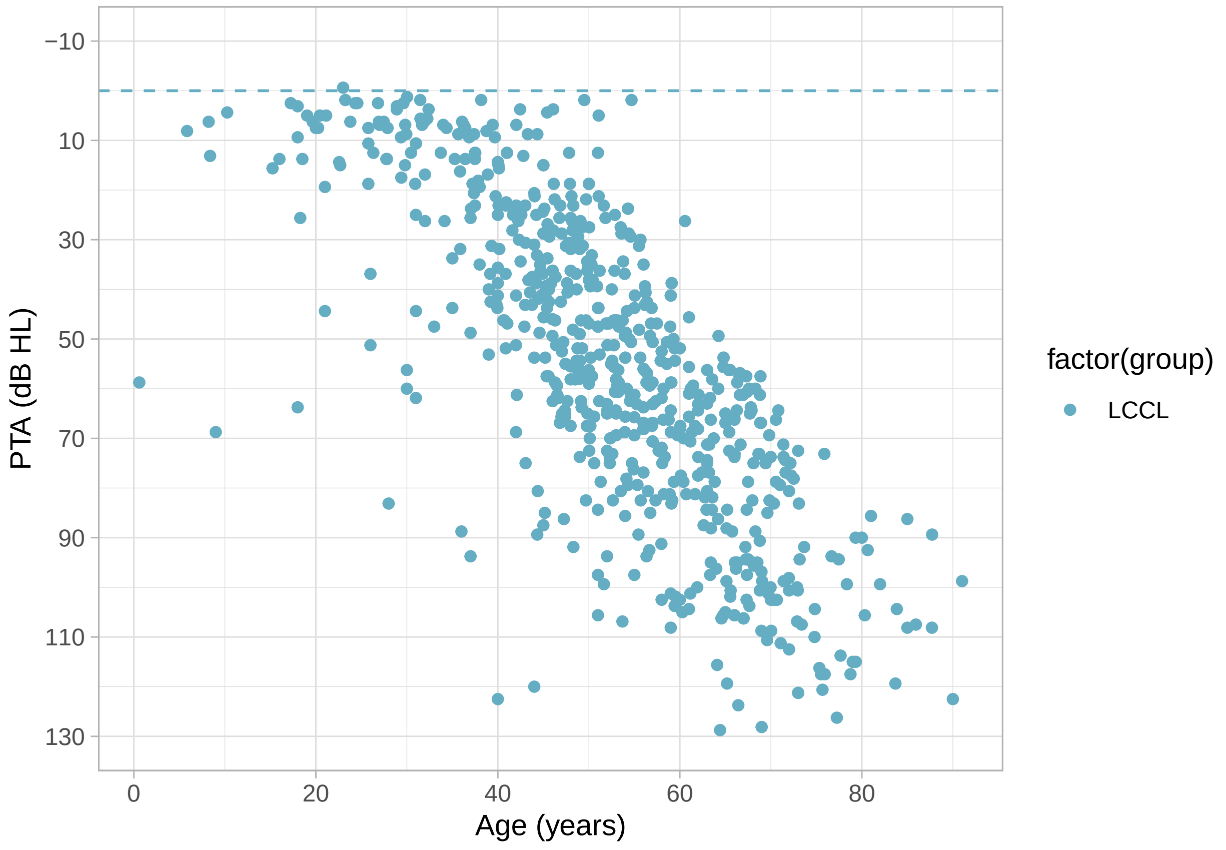
```
dev.print(pdf, '../results/pta_age_pid_groups.pdf')
```

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

Logistic fit of PTA with age

Perform fits to the data; first focus on LCCL domain.

```
lccl = subset(data, group == "LCCL")
ggplot(lccl,
       aes(x = Leeftijd, y = PTA54ADS),
       group = pid,
       color = group) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  geom_point(aes(colour = factor(group))) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130,-10)) +
  theme_light()
```



```
dev.print(pdf, '../results/pta_age_pid_lccl.pdf')
```

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

Try to fit the data with a linear function, a power-law function and a logistic function:

```
lin_fit <-
  nls(PTA54ADS ~ a * Leeftijd + b,
      data = lccl,
      start = list(a = 1.5, b = 0))
summary(lin_fit)
```

```
##
## Formula: PTA54ADS ~ a * Leeftijd + b
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## a   1.60851    0.05331  30.170   <2e-16 ***
## b -28.43929    2.89132  -9.836   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.01 on 673 degrees of freedom
##
## Number of iterations to convergence: 1
## Achieved convergence tolerance: 4.526e-10
```

```
nls_fit <-
  nls(PTA54ADS ~ a * Leeftijd ^ b,
      data = lccl,
      start = list(a = 0.05, b = 1.5))
summary(nls_fit)
```

```
##
## Formula: PTA54ADS ~ a * Leeftijd^b
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a  0.07298    0.01859   3.926 9.51e-05 ***
## b  1.66646    0.06175  26.988  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.54 on 673 degrees of freedom
##
## Number of iterations to convergence: 5
## Achieved convergence tolerance: 2.93e-06
```

```
startvec <- c(Asym = 120, xmid = 50, scal = 15)
nls_logis <- nls(PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal),
                 data = lccl,
                 start = startvec)
summary(nls_logis)
```

```
##
## Formula: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal)
##
## Parameters:
##      Estimate Std. Error t value Pr(>|t|)
## Asym  126.351      9.477   13.33   <2e-16 ***
## xmid   56.974      2.619   21.76   <2e-16 ***
## scal   15.410      1.371   11.24   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.29 on 672 degrees of freedom
##
## Number of iterations to convergence: 4
## Achieved convergence tolerance: 3.603e-06
```

Compare power-law fit and the logistic functione and display the results

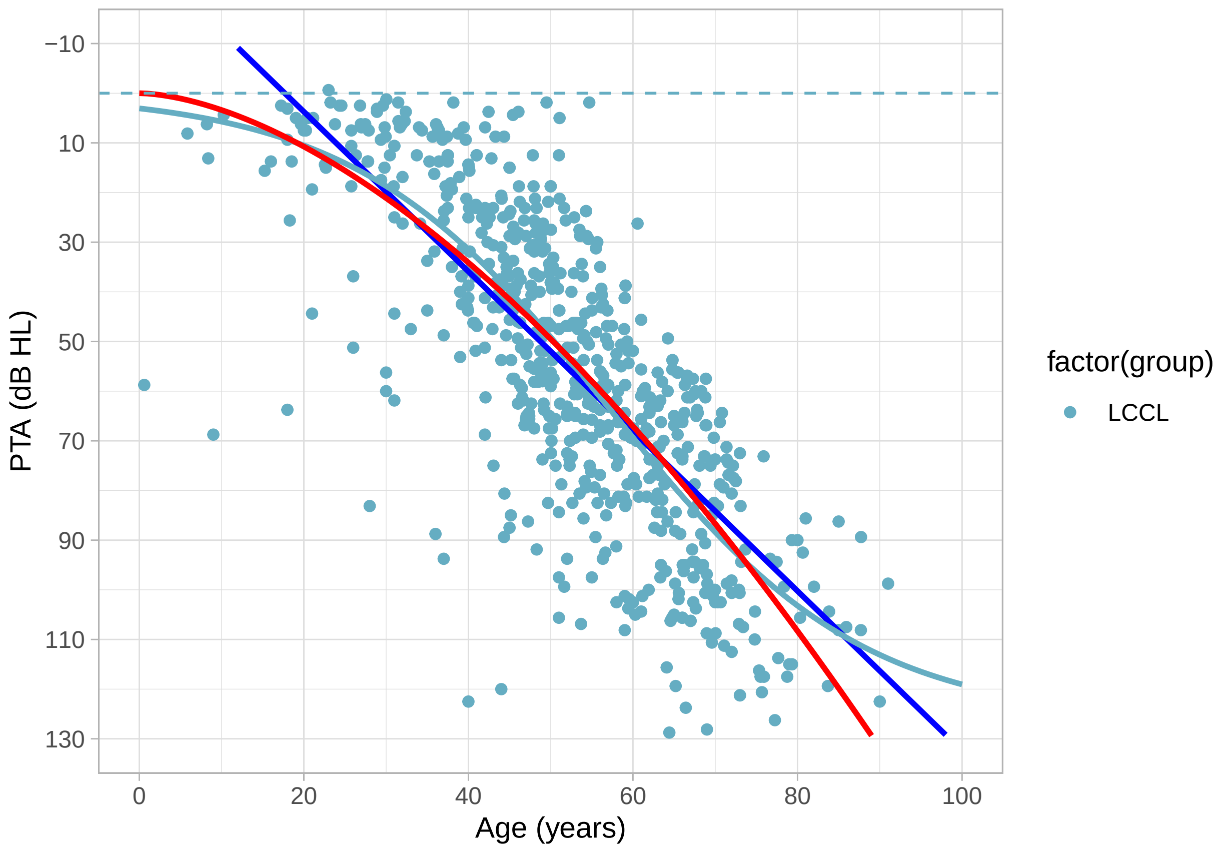
```
anova(nls_fit, nls_logis)
```

```
## Analysis of Variance Table
```

```
##
## Model 1: PTA54ADS ~ a * Leeftijd^b
## Model 2: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal)
##   Res.Df Res.Sum Sq Df Sum Sq F value    Pr(>F)
## 1    673     257033
## 2    672     250020  1 7013.2   18.85 1.632e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
newdat = expand.grid(Leeftijd = seq(0, 100, by = 1))
newdat$lin <- predict(lin_fit, newdata = newdat)
newdat$pta_logistic <- predict(nls_logis, newdata = newdat)
newdat$pta_power <- predict(nls_fit, newdata = newdat)
#newdat
ggplot(lccl, aes(x = Leeftijd, y = PTA54ADS)) +
  geom_point(aes(colour = factor(group))) +
  geom_line(data = newdat,
            aes(y = lin),
            size = 1,
            col = 'blue') +
  geom_line(data = newdat, aes(y = pta_logistic), size = 1) +
  geom_line(data = newdat,
            aes(y = pta_power),
            size = 1,
            col = 'red') +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
```

```
## Warning: Removed 14 row(s) containing missing values (geom_path).

## Warning: Removed 11 row(s) containing missing values (geom_path).
```



```
dev.print(pdf, '../results/pta_age_lccl_fits.pdf')
```

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

As we can see, the logistic function (SSlogis) describes the data better than the power-law function (F = 18,9; p = 1.6 e-5) This function has also been used in desribing the (frequency-specific) thresholds in Pauw et al., 2011 and will used in the subsequent sections.

Group comparison

The main questions is whether the function that describes the PTA (dB HL) as a function of age (years) differs between the groups @ref(fig:plot__pta__age__groups).

Start with a group-fit; discarding grouping information

```
fit0 <-
  nls(PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal), data = data)
summary(fit0)
```

```
##
## Formula: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal)
##
## Parameters:
##      Estimate Std. Error t value Pr(>|t|)
## Asym  135.557     12.188   11.12   <2e-16 ***
## xmid   59.258      3.517   16.85   <2e-16 ***
## scal   17.807      1.639   10.86   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.78 on 713 degrees of freedom
##
## Number of iterations to convergence: 4
## Achieved convergence tolerance: 2.948e-06
```

```
coef(fit0)
```

```
##      Asym      xmid      scal
## 135.55740  59.25776  17.80669
```

Now, add a grouping-variable with the mid-point (xmid)

```
# https://stats.stackexchange.com/questions/27273/how-do-i-fit-a-nonlinear-mixed-effects-model-for-repe
# https://stats.stackexchange.com/questions/316801/how-to-compare-logistic-regression-curves
fit1 <- nls(
  PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal),
  data = data,
  start = list(
    Asym = rep(120, 1),
    xmid = rep(50, 3),
    scal = rep(15, 1)
  )
)
summary(fit1)
```

```
##
## Formula: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal)
##
## Parameters:
##       Estimate Std. Error t value Pr(>|t|)
## Asym   116.114      6.242   18.60   <2e-16 ***
## xmid1   54.188      1.813   29.89   <2e-16 ***
## xmid2   43.644      3.462   12.61   <2e-16 ***
## xmid3   34.631      3.185   10.87   <2e-16 ***
## scal    14.394      1.127   12.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.13 on 711 degrees of freedom
##
## Number of iterations to convergence: 7
## Achieved convergence tolerance: 4.654e-06
```

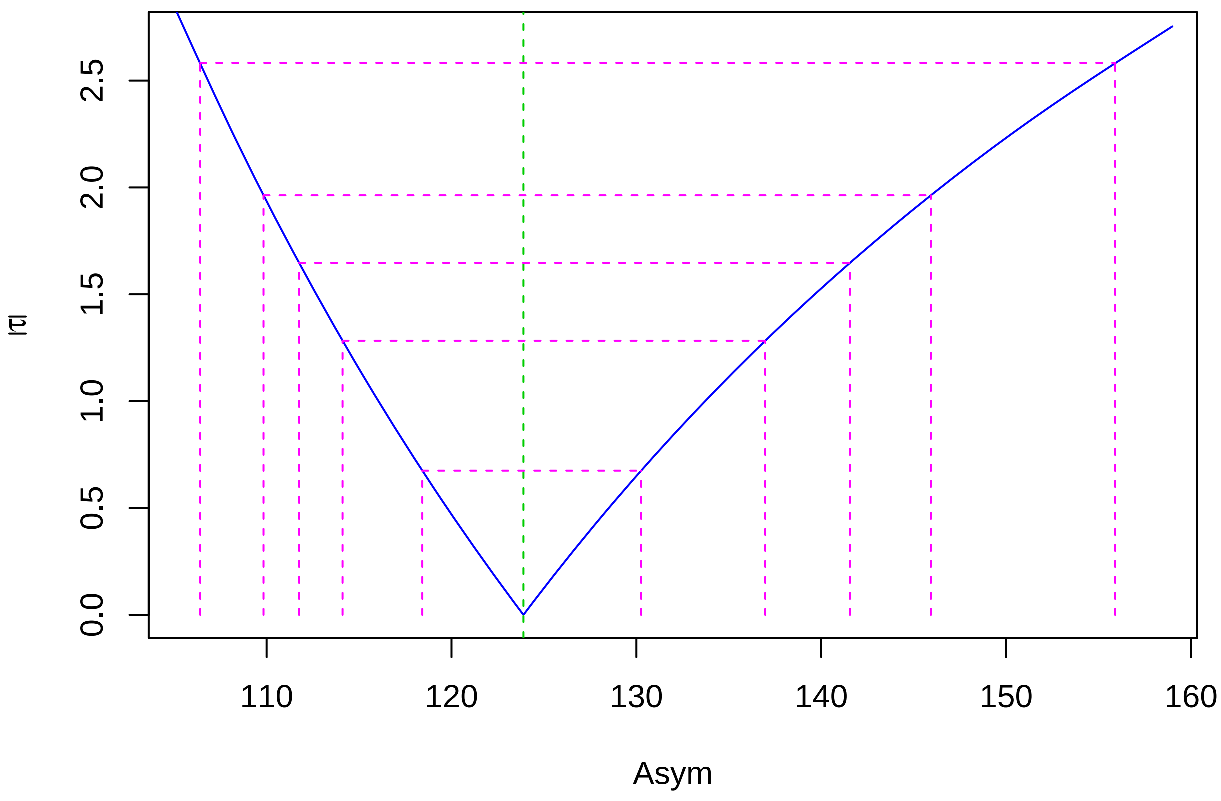
And add the scaling [scal] as a grouping variable; does it futher explain differences between groups?

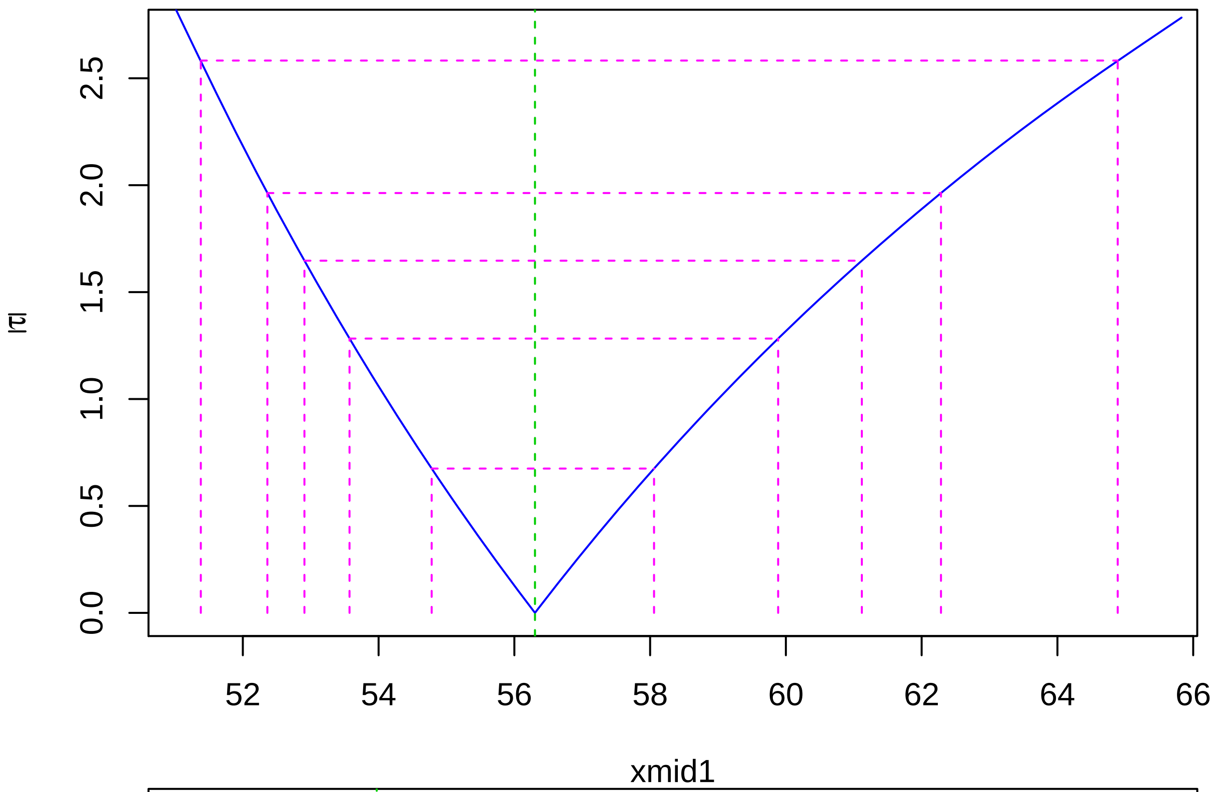
```
fit2 <-
  nls(
    PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal[group]),
    data = data,
    start = list(
      Asym = rep(120, 1),
      xmid = rep(50, 3),
      scal = rep(15, 3)
    )
  )
```

```r
summary(fit2)
```

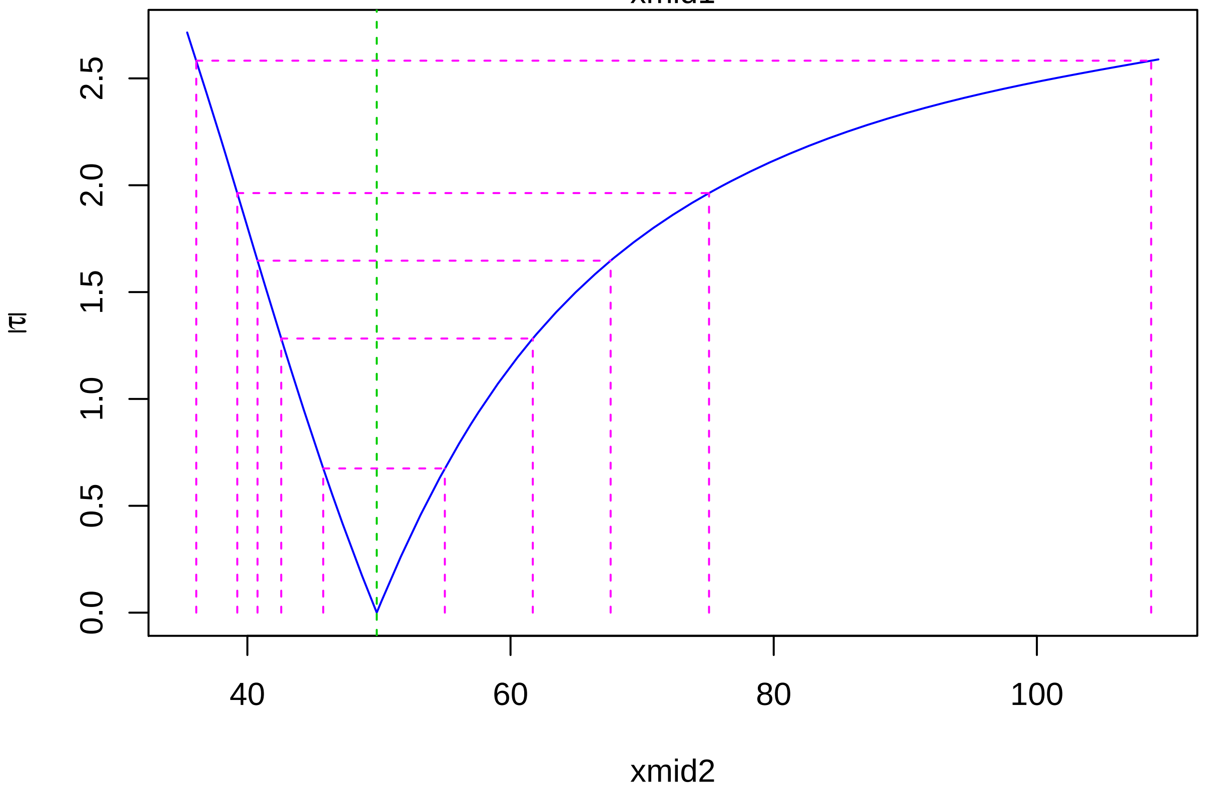
```
##
## Formula: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal[group])
##
## Parameters:
##       Estimate Std. Error t value Pr(>|t|)
## Asym   123.891      8.563  14.468  < 2e-16 ***
## xmid1   56.304      2.381  23.652  < 2e-16 ***
## xmid2   49.833      6.541   7.618 8.24e-14 ***
## xmid3   37.568      5.595   6.715 3.86e-11 ***
## scal1   15.084      1.291  11.682  < 2e-16 ***
## scal2   24.884      8.419   2.956  0.00322 **
## scal3   26.860      6.824   3.936 9.09e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.05 on 709 degrees of freedom
##
## Number of iterations to convergence: 6
## Achieved convergence tolerance: 2.195e-06
```

```r
plot(profile(fit2))
```

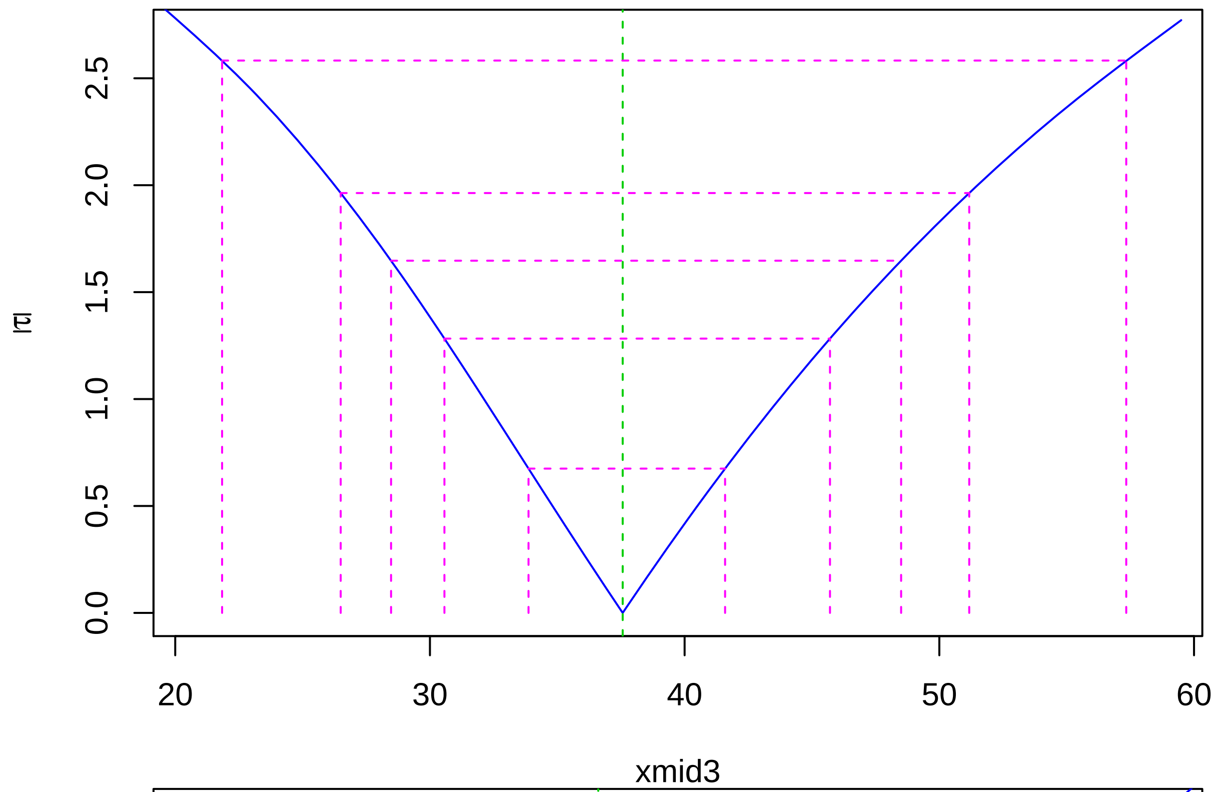




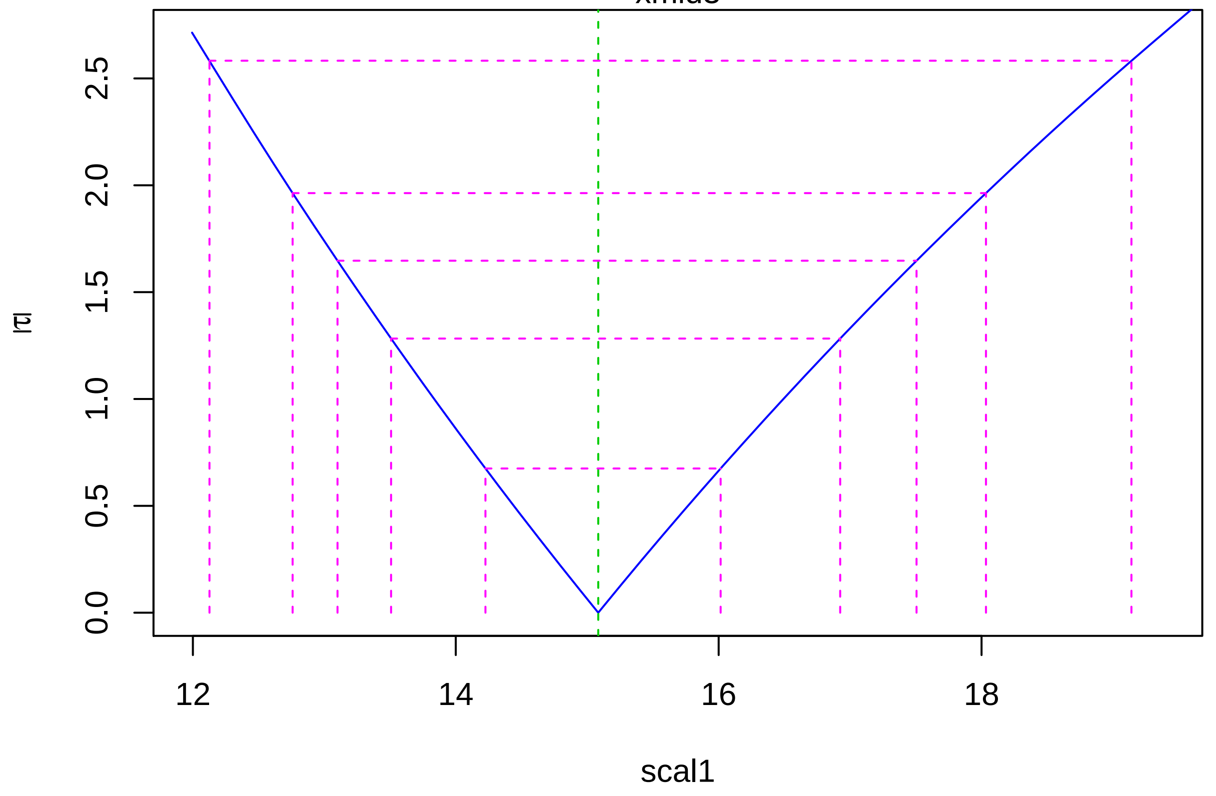
2.5
2.0
1.5
1.0
0.5
0.0
$|\tau|$
52
54
56
58
60
62
64
66
xmid1



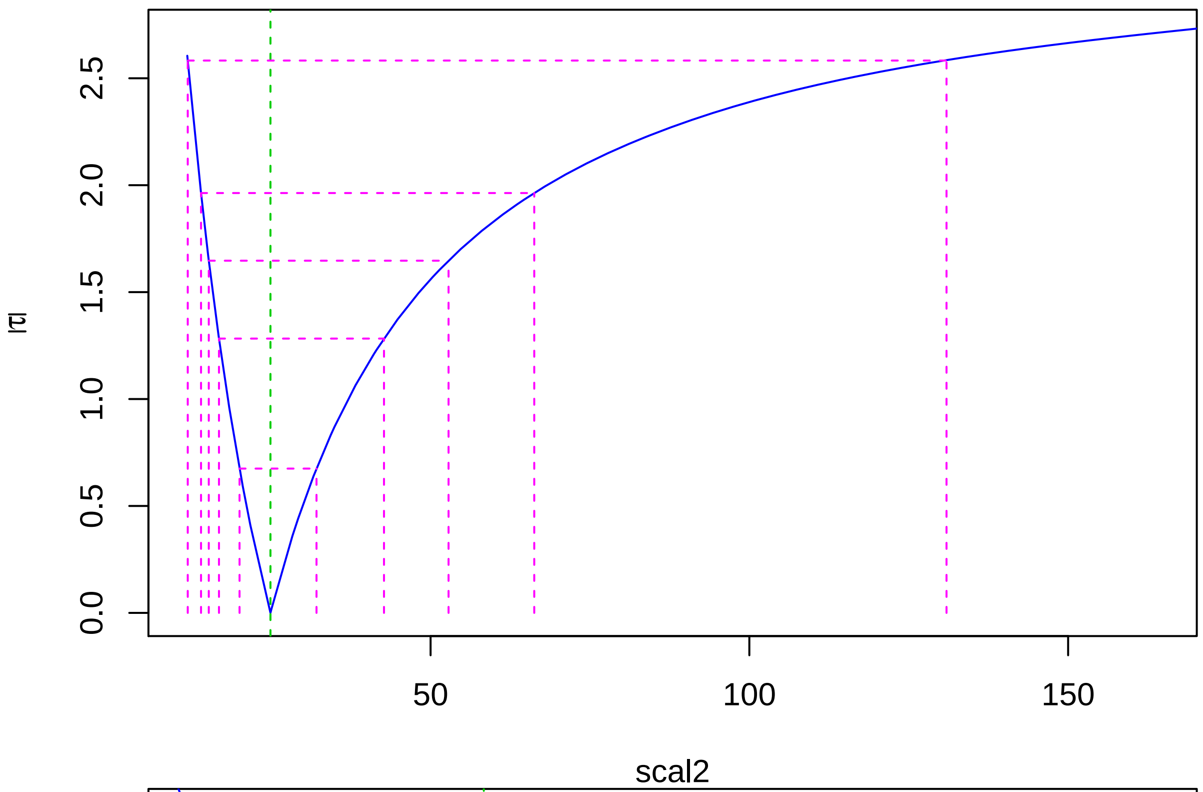
2.5
2.0
1.5
1.0
0.5
0.0
$|\tau|$
40
60
80
100
xmid2

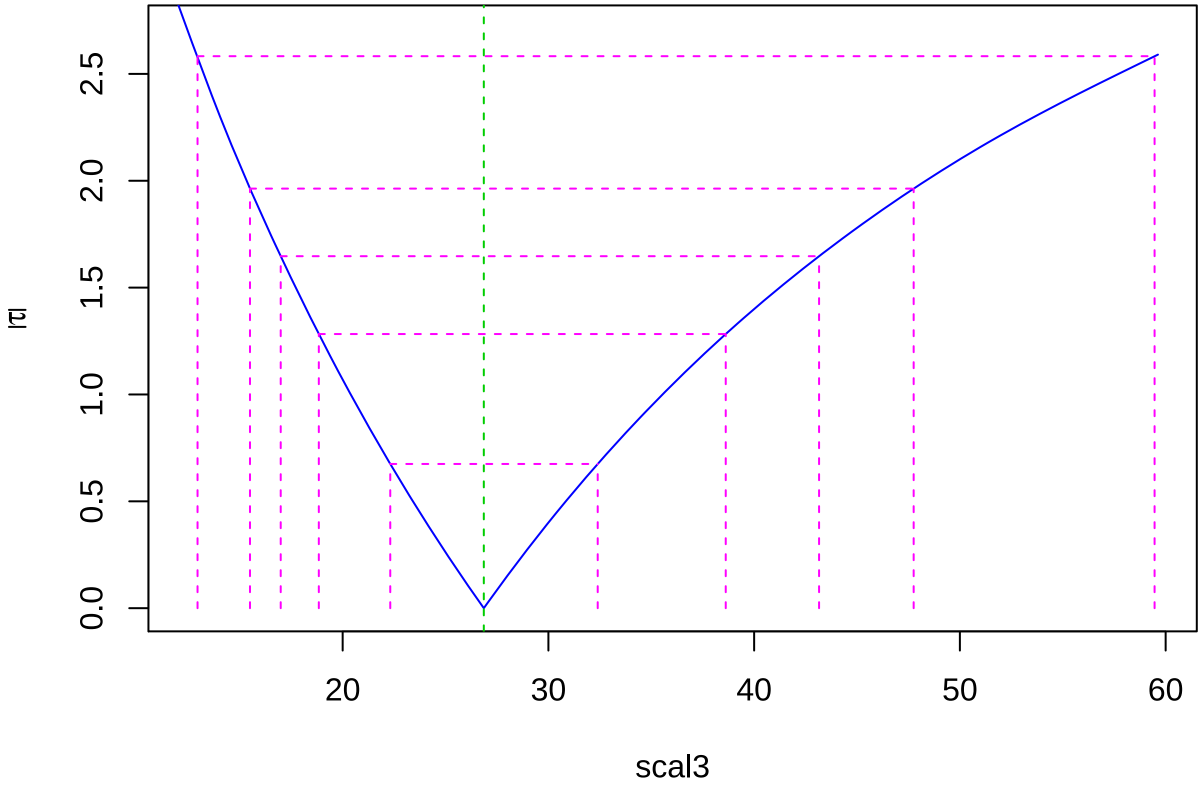


|τ|
0.0
0.5
1.0
1.5
2.0
2.5
20
30
40
50
60
xmid3



|τ|
0.0
0.5
1.0
1.5
2.0
2.5
12
14
16
18
scal1





And now add the asymptotic value of the fit (Leeftijd -> infinity) (Asym):

```
fit3 <-
  nls(
    PTA54ADS ~ SSlogis(Leeftijd, Asym[group], xmid[group], scal[group]),
```

```r
    data = data,
    start = list(
      Asym = rep(120, 3),
      xmid = rep(50, 3),
      scal = rep(15, 3)
    )
  )
summary(fit3)
```

```
##
## Formula: PTA54ADS ~ SSlogis(Leeftijd, Asym[group], xmid[group], scal[group])
##
## Parameters:
##       Estimate Std. Error t value Pr(>|t|)
## Asym1  126.350      9.361  13.497  < 2e-16 ***
## Asym2   89.933     60.449   1.488   0.1373
## Asym3   97.280     10.751   9.049  < 2e-16 ***
## xmid1   56.974      2.587  22.025  < 2e-16 ***
## xmid2   34.989     26.673   1.312   0.1900
## xmid3   27.148      4.666   5.819 8.99e-09 ***
## scal1   15.410      1.354  11.383  < 2e-16 ***
## scal2   17.424     17.929   0.972   0.3315
## scal3   13.988      5.807   2.409   0.0163 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.05 on 707 degrees of freedom
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 3.258e-06
```

Now test the various models. Which of the parameters explain the data best?

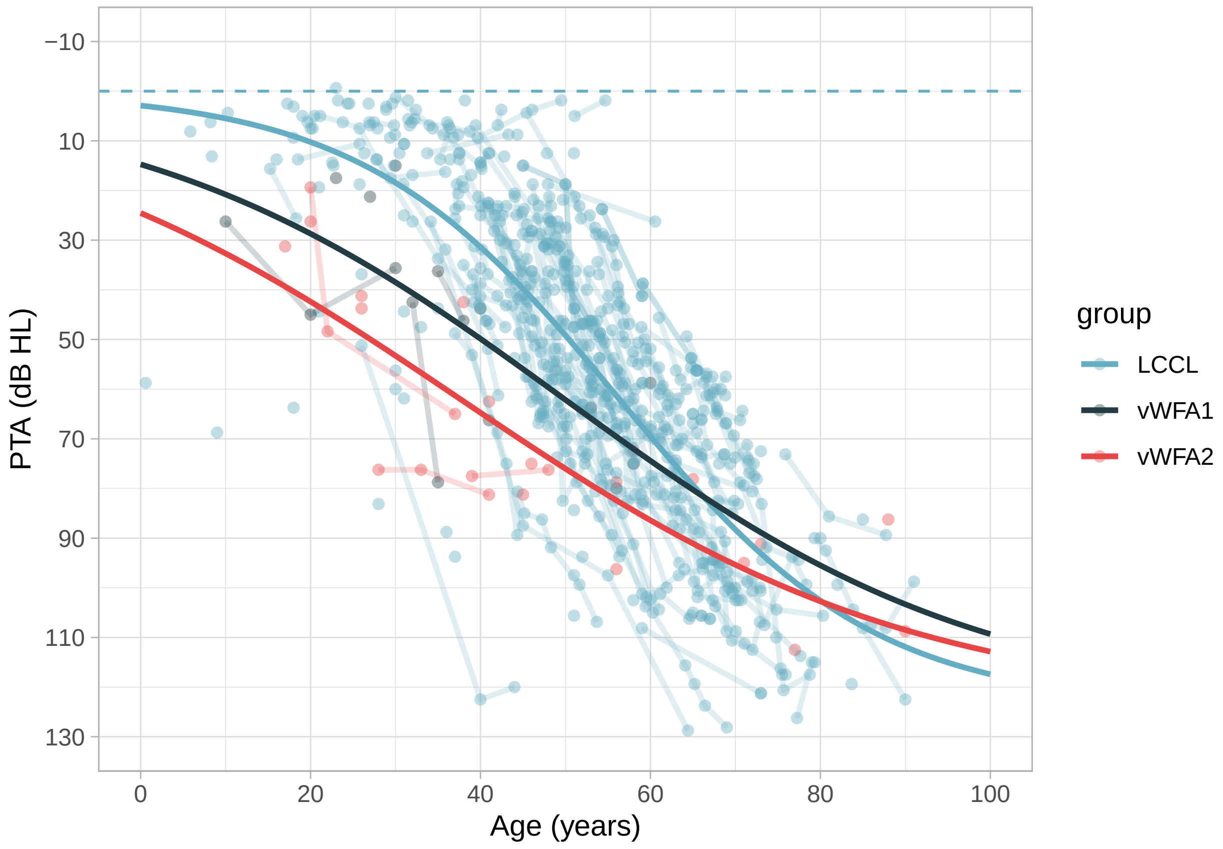
```r
anova(fit0, fit1, fit2, fit3)
```

```
## Analysis of Variance Table
##
## Model 1: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal)
## Model 2: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal)
## Model 3: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal[group])
## Model 4: PTA54ADS ~ SSlogis(Leeftijd, Asym[group], xmid[group], scal[group])
##   Res.Df Res.Sum Sq Df  Sum Sq F value    Pr(>F)
## 1    713     279067
## 2    711     260324  2 18743.2 25.5958 1.844e-11 ***
## 3    709     257309  2  3014.8  4.1535   0.01609 *
## 4    707     256670  2   639.6  0.8809   0.41487
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It turns out the both the variables [xmid] and [scale], i.e. the midpoint and slope at the midpoint significantly differ between the three groups, but that adding the asymptotic value does not describe the data signigicantly better (F=0.89, p=0.41). Fit the data and plot the results:

```r
newdat = expand.grid(Leeftijd = seq(0, 100, by = 1),
                     group = c("LCCL", "vWFA1", "vWFA2"))
newdat$fit <- predict(fit2, newdata = newdat)
```

```r
ggplot(data, aes(
  x = Leeftijd,
  y = PTA54ADS,
  group = pid,
  color = group
)) +
  geom_point(aes(colour = factor(group)), alpha = .4) +
  geom_line(data = data, size = 1, alpha = .2) +
  geom_line(data = newdat,
            aes(
              y = fit,
              group = group,
              colour = factor(group)
            ),
            size = 1) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
```



```r
dev.print(pdf, '../results/pta_age_pid_groups_fits.pdf')
```

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

Now perform fit on individual data by subsetting the data to keep individuals with more that x=2 longitudinal datapoints. Is it the case that using a non-linear mixed-model approach may help us?

```
#https://stackoverflow.com/questions/14439770/filter-rows-in-dataframe-by-number-of-rows-per-level-of-a
pidlengths <- ave(as.numeric(data$pid),
                  data$pid, FUN = length)
#df2 <- lccl[pidlengths > 5, ]
df2 <- data[pidlengths > 2,]
t3 <- with(df2, table(group))
```

So, with only two data-points, only 3 and 6 subjects for the vWFA1 and vWFA2 domain respectively, remain. Now, fit those with a logistic function (SSlogis) using nlslist.

```
models <-
  nlsList(PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal) |
            pid, data = df2)
```

```
## Warning: 37 errors caught in (attr(object, "initial"))(mCall = mCall, data = data, LHS = LHS).  The
##
##                                      singular matrix 'a' in solve
##                                                                 1
##                                                singular gradient
##                                                                 2
## step factor 0.000488281 reduced below 'minFactor' of 0.000976562
##                                                                12
##            too few distinct input values to fit a logistic model
##                                                                22
```

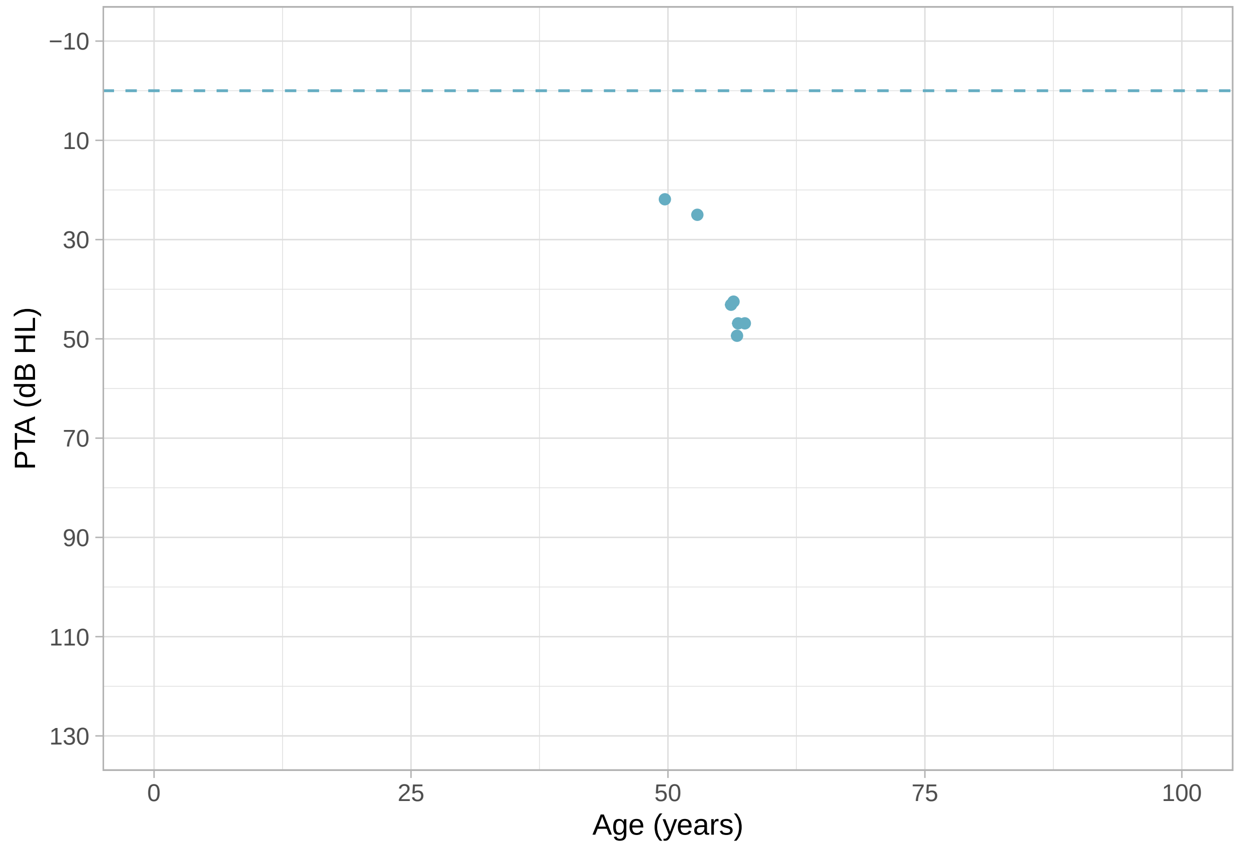
As we can see, some model-predictions failed; they end up with NaNs in the model fit list (nlslist); see e.g. pid 147

```
data_id <- subset(df2, pid == "147")
data_id
```

```
## # A tibble: 7 x 4
##     pid group Leeftijd PTA54ADS
##   <dbl> <fct>    <dbl>    <dbl>
## 1   147 LCCL      49.7     21.9
## 2   147 LCCL      52.9     25
## 3   147 LCCL      56.4     42.5
## 4   147 LCCL      56.1     43.1
## 5   147 LCCL      56.8     46.9
## 6   147 LCCL      57.5     46.9
## 7   147 LCCL      56.7     49.4
```

```
ggplot(data = data_id,  aes(x = Leeftijd, y = PTA54ADS)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  xlim(0,100)+
  theme_light()
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which will
## replace the existing scale.
```



Predict for all pid's the fit to the model and remove the pid's that give NaNs. Check how many subjects per group we end up with.

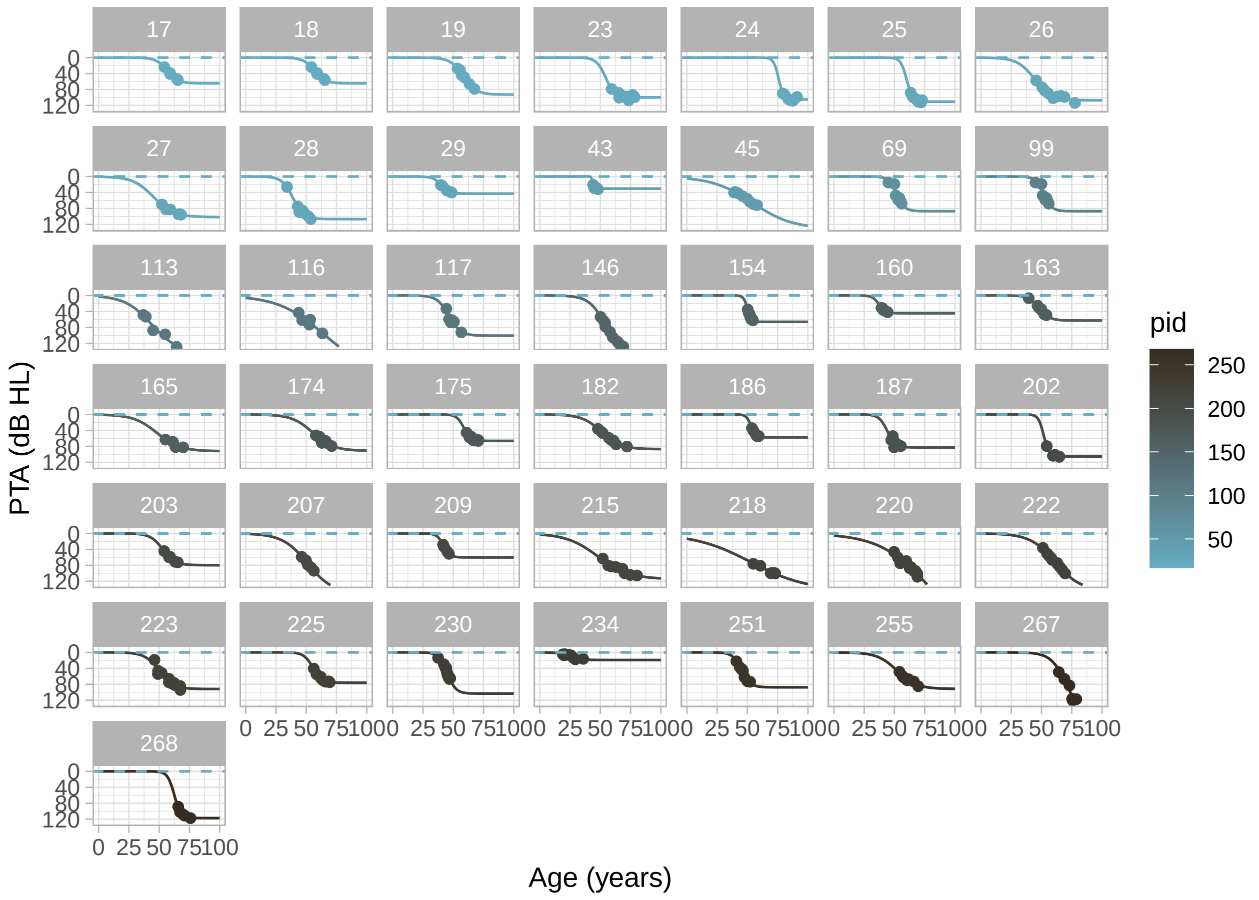
```
df2$Pred <- predict(models)
df2_na <- na.omit(df2)
df2_na_stats <- with(df2_na, table(group, pid))
df2_na_stats
```

```
##        pid
## group   17  18  19  23  24  25  26  27  28  29  43  45  69  99  113  116  117  146  154  160  163
##   LCCL   6   6   6   8   6   6   9   6   9   5   4   8   7   7    5    5    7   10   10    5    8
##   vWFA1  0   0   0   0   0   0   0   0   0   0   0   0   0   0    0    0    0    0    0    0    0
##   vWFA2  0   0   0   0   0   0   0   0   0   0   0   0   0   0    0    0    0    0    0    0    0
##        pid
## group   165  174  175  182  186  187  202  203  207  209  215  218  220  222  223  225  230  234
##   LCCL    4    7    8    8    5    6    5    5    5    4    8    6   11   10   11    9   14   10
##   vWFA1   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##   vWFA2   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##        pid
## group   251  255  267  268
##   LCCL    8    7    8    5
##   vWFA1   0    0    0    0
##   vWFA2   0    0    0    0
```

We only keep the pid's from the LCCL group. The pid's in the other groups are not properly fitted. Also note that the minimum of data-points for a reasonable fit is 4.

```r
le <- unique(df2_na$pid)
newdat = expand.grid(Leeftijd = seq(0, 100, by = 1), pid = le)
newdat$prednlm <- predict(models, newdata = newdat)
```

```r
#https://stackoverflow.com/questions/37122994/plotting-a-list-of-non-linear-regressions-with-ggplot
#https://aosmith.rbind.io/2018/11/16/plot-fitted-lines/
ggplot(data = df2_na,  aes(x = Leeftijd, y = PTA54ADS, colour = pid)) +
  geom_point() +
  geom_line(data = newdat, aes(y = prednlm)) +
  facet_wrap(~ pid) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 25)) +
  scale_y_reverse(breaks = seq(0, 120, 40), limits = c(130,-10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
```



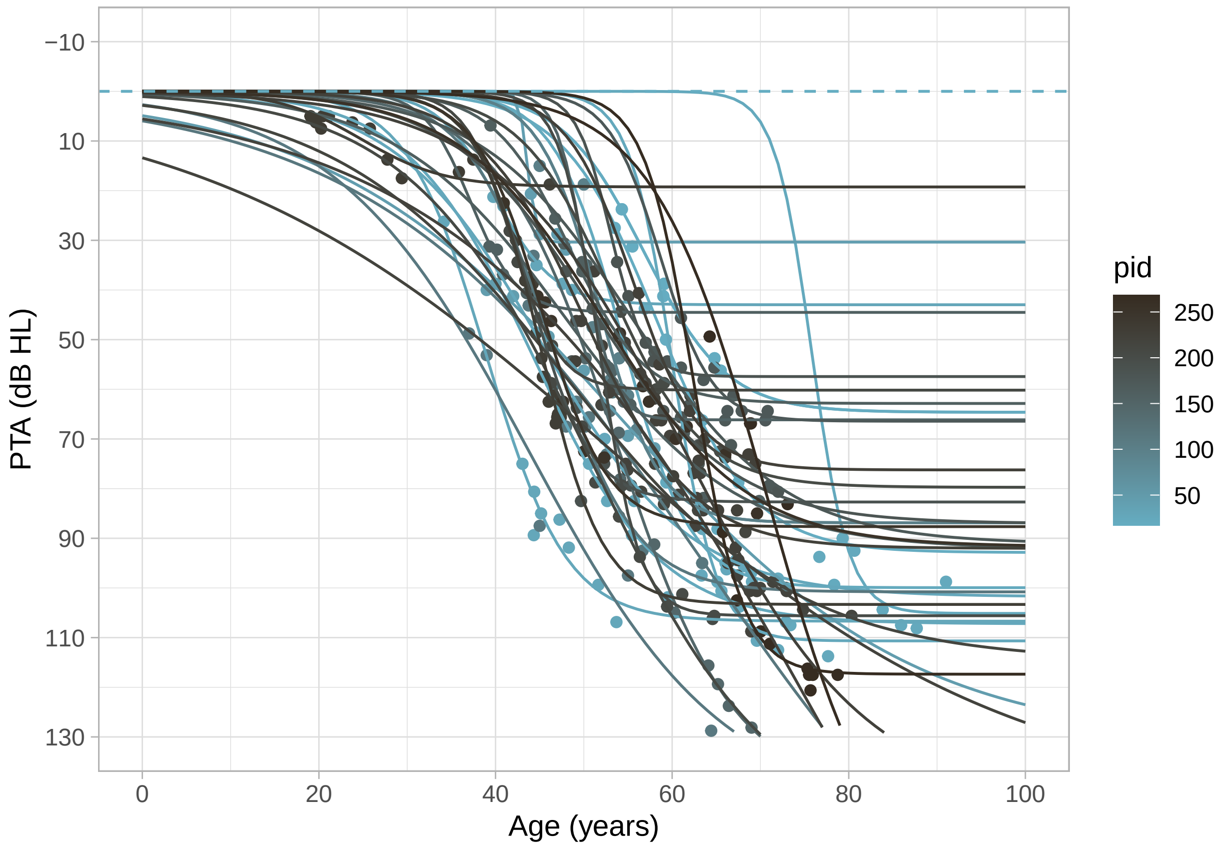
```r
dev.print(pdf, '../results/pta_age_pid_lccl_ind_fits.pdf')
```

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

So, it seeems we can fit the data for individual subjects by some extent. It often ‘fails’ by over- or underestimating the tail (coef.lmlist Asym column). We can also plot it all in one figure.

```r
ggplot(data = df2_na,  aes(x = Leeftijd, y = PTA54ADS, colour = pid)) +
  geom_point() +
  geom_line(data = newdat, aes(y = prednlm, group = pid)) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
```

```
## Warning: Removed 176 row(s) containing missing values (geom_path).
```



```r
dev.print(pdf, '../results/pta_age_pid_lccl_ind_fits_overlay.pdf')
```

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

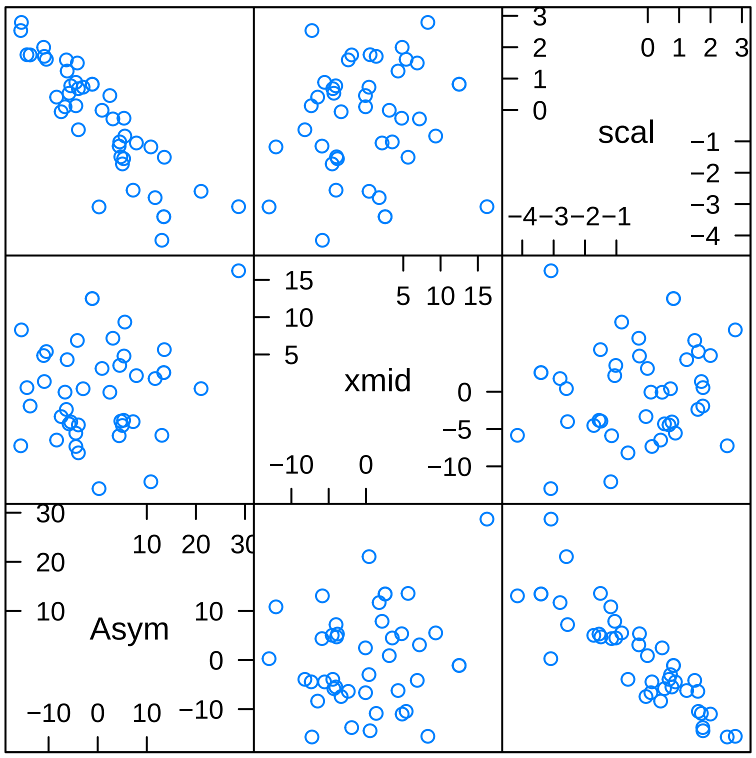
Feed the remaining data into the non-linear mixed-models with the parameters Asym, xmid, and scal as random factors.

```r
nm1 <-
  nlmer(
    PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal) ~ Asym + xmid + scal |
      pid,
    df2_na,
    method="ML",
    start = c(Asym = 100, xmid = 60, scal = 15),
    corr = FALSE
  )
summary(nm1)
```

```
## Warning in vcov.merMod(object, use.hessian = use.hessian): variance-covariance matrix computed from 
## not positive definite or contains NA values: falling back to var-cov estimated from RX

## Warning in vcov.merMod(object, correlation = correlation, sigm = sig): variance-covariance matrix co
## not positive definite or contains NA values: falling back to var-cov estimated from RX

## Nonlinear mixed model fit by maximum likelihood  ['nlmerMod']
## Formula: PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal) ~ Asym + xmid +
##     scal | pid
##    Data: df2_na
##
##      AIC      BIC   logLik deviance df.resid
##   2152.6   2189.9  -1066.3   2132.6      297
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.14923 -0.50051 -0.03471  0.44682  2.94717
##
## Random effects:
##  Groups   Name Variance Std.Dev. Corr
##  pid      Asym 166.742  12.913
##           xmid  44.562   6.675    0.23
##           scal   4.541   2.131   -0.53  0.11
##  Residual       24.231   4.922
## Number of obs: 307, groups:  pid, 43
##
## Fixed effects:
##      Estimate Std. Error t value
## Asym 113.0219     3.6536   30.93
## xmid  52.5762     1.1214   46.88
## scal   8.4703     0.5342   15.86
##
## Correlation of Fixed Effects:
##      Asym  xmid
## xmid 0.409
## scal 0.343 0.265
```

```
plot(ranef(nm1,augFrame=T))
```

```
## Warning in ranef.merMod(nm1, augFrame = T): additional arguments to ranef.merMod
## ignored: augFrame

## $pid
```



Scatter Plot Matrix

```r
params <- coef(nm1)
head(params)
```

```
## $pid
##            Asym     xmid      scal
## 17  111.91187 65.06896  9.290262
## 18  111.91187 65.06896  9.290262
## 19  118.54557 61.92052  7.638449
## 23  105.58459 49.24825  8.413491
## 24  108.87821 59.45191  9.966515
## 25  120.90609 54.73558  7.417888
## 26  109.08970 44.38473  7.839455
## 27  104.65608 46.09939  8.880052
## 28  113.29662 39.58856  5.379525
## 29  110.04858 52.98010  9.196106
## 43  113.90947 55.69913  8.459887
## 45   97.34802 45.33117 11.001626
## 69  126.47538 55.14536  5.067873
## 99  126.47538 55.14536  5.067873
## 113 123.85066 40.51042  7.292063
## 116 108.56522 47.02774  9.350318
## 117 118.05750 48.05015  6.749991
## 146 134.07677 52.99453  5.877904
## 154 124.71754 54.34416  5.675409
## 160 109.08630 48.12987  9.147920
## 163 117.51108 56.10667  7.451837
## 165  99.26396 50.66147 10.224177
## 174 102.01522 57.42787 10.467515
## 175  97.48613 60.86962 11.260595
## 182  98.61739 53.12833 10.230050
## 186 116.11258 59.74771  8.184632
## 187 108.57918 45.24103  8.603256
## 202 118.30537 48.75103  6.916644
## 203 106.80804 56.87895  9.709881
## 207 117.38917 46.67863  7.318511
## 209 117.73015 48.63513  6.977685
## 215 107.53529 48.52569  9.239854
## 218 107.19360 48.27288  9.002671
## 220 115.49471 52.50751  8.928806
## 222 118.37714 57.36307  8.206486
## 223 106.36951 52.52182  8.571337
## 225 102.56742 57.96347 10.083785
## 230 126.08938 46.73509  4.317173
## 234 106.64376 50.20636 10.065763
## 251 120.23932 48.56520  5.913720
## 255 102.15156 53.94762 10.178729
## 267 141.71632 68.79567  5.386864
## 268 126.58949 58.22265  6.962851
```

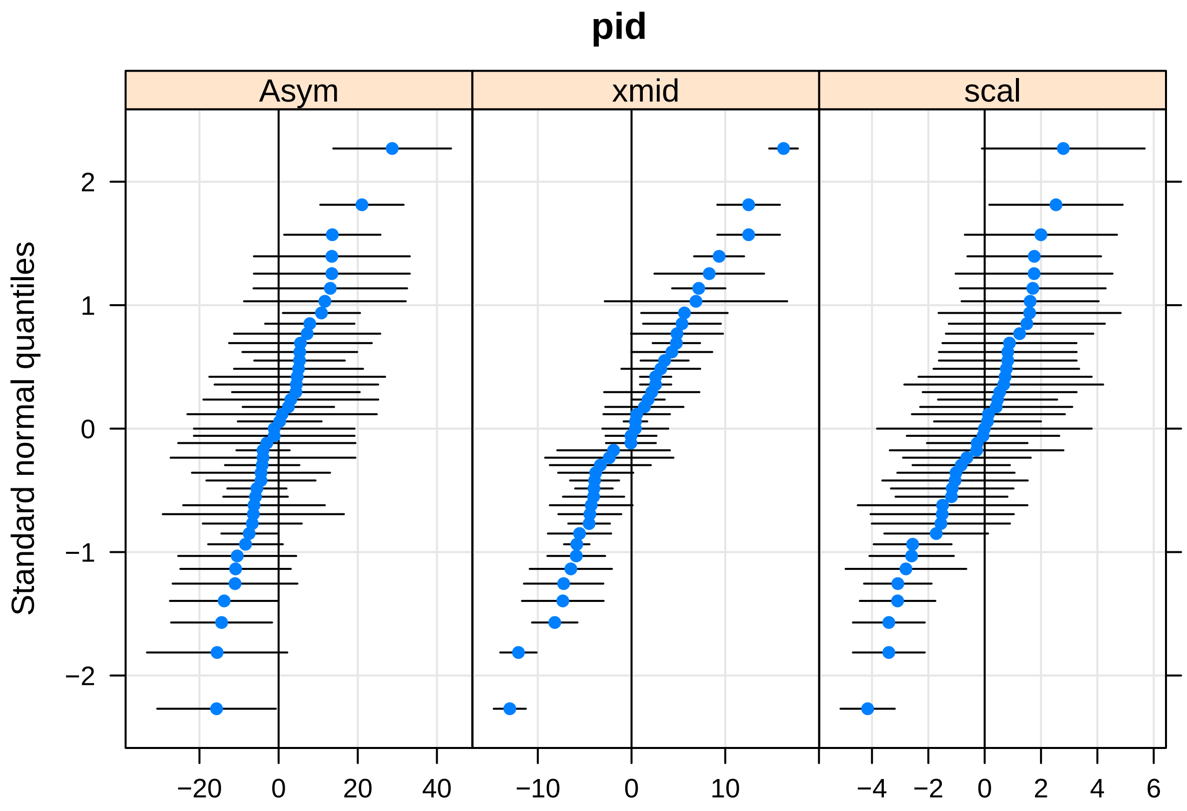
```
#sjplot(nm1)
```

```
require(lattice)
```

```
## Loading required package: lattice
```

```
qqmath(ranef(nm1, condVar = TRUE))
```

```
## $pid
```



Frequencey-specific analyses

Now subset the data to contain individual frequencies

```
data_all <-
  subset(
    data_subset,
    select = c(
      'pid',
      'group',
      'Leeftijd',
      '250.AD',
      '500.AD',
      '1000.AD',
      '2000.AD',
      '4000.AD',
      '8000.AD',
      '250.AS',
      '500.AS',
      '1000.AS',
      '2000.AS',
      '4000.AS',
      '8000.AS'
    )
  )
head(data_all)
```

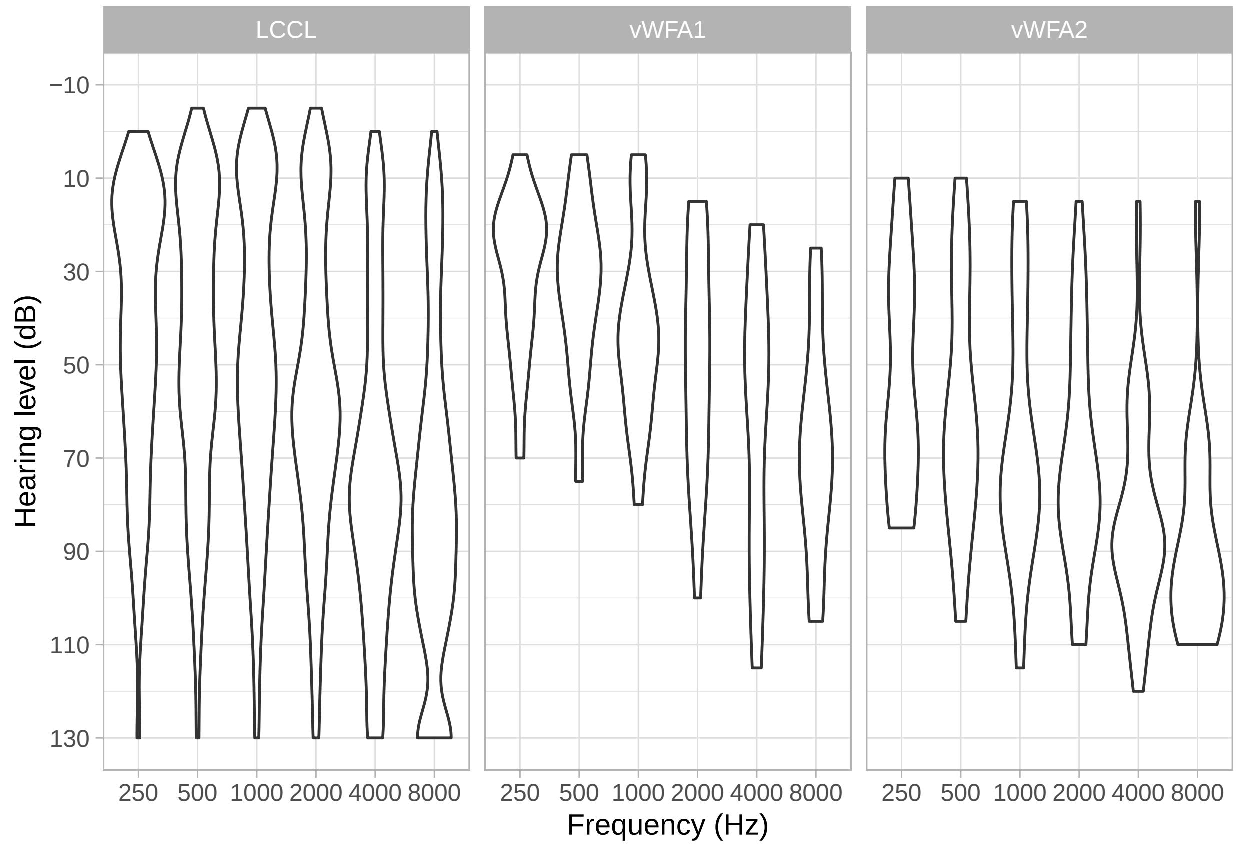
```
## # A tibble: 6 x 15
##     pid group Leeftijd `250.AD` `500.AD` `1000.AD` `2000.AD` `4000.AD` `8000.AD`
##   <dbl> <fct>    <dbl>    <dbl>    <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1     1 LCCL        60       NA       45        50        60        80        85
## 2     2 LCCL        82       70       75       100       130       130        NA
## 3     3 LCCL        68       80       80        80        90       130       130
## 4     4 LCCL        48       10        5        10        15        65        60
## 5     4 LCCL        55       20       20        15        40        80        85
## 6     5 LCCL        48        5        5         0         0        45        60
## # ... with 6 more variables: `250.AS` <dbl>, `500.AS` <dbl>, `1000.AS` <dbl>,
## #   `2000.AS` <dbl>, `4000.AS` <dbl>, `8000.AS` <dbl>
```

Convert 'wide' dataset into 'long' format using tidyr and remove NaNs

```
tidier <- data_all %>%
  gather(f, dB,-pid,-group,-Leeftijd)
data_all_l <- tidier %>%
  separate(f, into = c("frequency", "ear"), sep = "\\.")
#head(data_all_l)
data_all_l$frequency = factor(data_all_l$frequency)
data_all_l$ear = factor(data_all_l$ear)
data_all_l <- na.omit(data_all_l)
head(data_all_l)
```

```
## # A tibble: 6 x 6
##     pid group Leeftijd frequency ear      dB
##   <dbl> <fct>    <dbl> <fct>     <fct> <dbl>
## 1     2 LCCL        82 250       AD       70
## 2     3 LCCL        68 250       AD       80
## 3      4 LCCL         48  250        AD        10
## 4      4 LCCL         55  250        AD        20
## 5      5 LCCL         48  250        AD         5
## 6      5 LCCL         56  250        AD        45
```

```
p <- data_all_l %>%
  mutate(frequency = fct_relevel(frequency, "250", "500", "1000", "2000", "4000", "8000")) %>%
  ggplot(aes(x = frequency, y = dB)) +
  #geom_bar(stat="identity") +
  #geom_histogram() +
  geom_violin() +
  facet_wrap( ~ group, ncol = 3) +
  #geom_point()
  xlab("Frequency (Hz)") +
  ylab("Hearing level (dB)") +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  theme_light()
  #theme_classic()
p
```

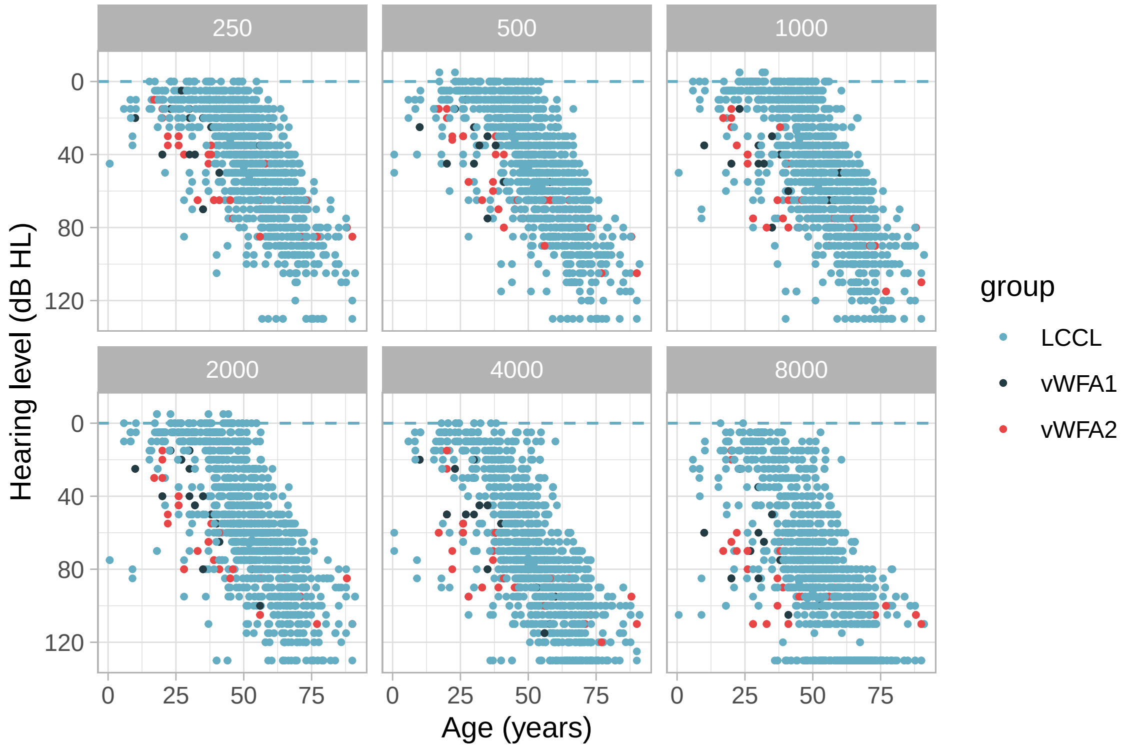


```
dev.print(pdf, '../results/violin_plot_HL_groups.pdf')
```

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

```
data_all_l$f = factor(data_all_l$frequency,
                      levels = c('250', '500', '1000', '2000', '4000', '8000'))
ggplot(data_all_l, aes(
```

```r
  x = Leeftijd,
  y = dB,
  group = f,
  color = group
)) +
  geom_point(size = 0.7) +
  facet_wrap( ~ f) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 25)) +
  scale_y_reverse(breaks = seq(0, 120, 40), limits = c(130, -10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("Hearing level (dB HL)") +
  theme_light()
```



```r
dev.print(pdf, '../results/HL_age_frequency_groups.pdf')
```

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

```r
models <-
  nlsList(
    dB ~ SSlogis(Leeftijd, Asym, xmid, scal) |
      f,
    data = data_all_l,
    start = c(Asym = 100, xmid = 60, scal = 15)
  )
summary(models)
```

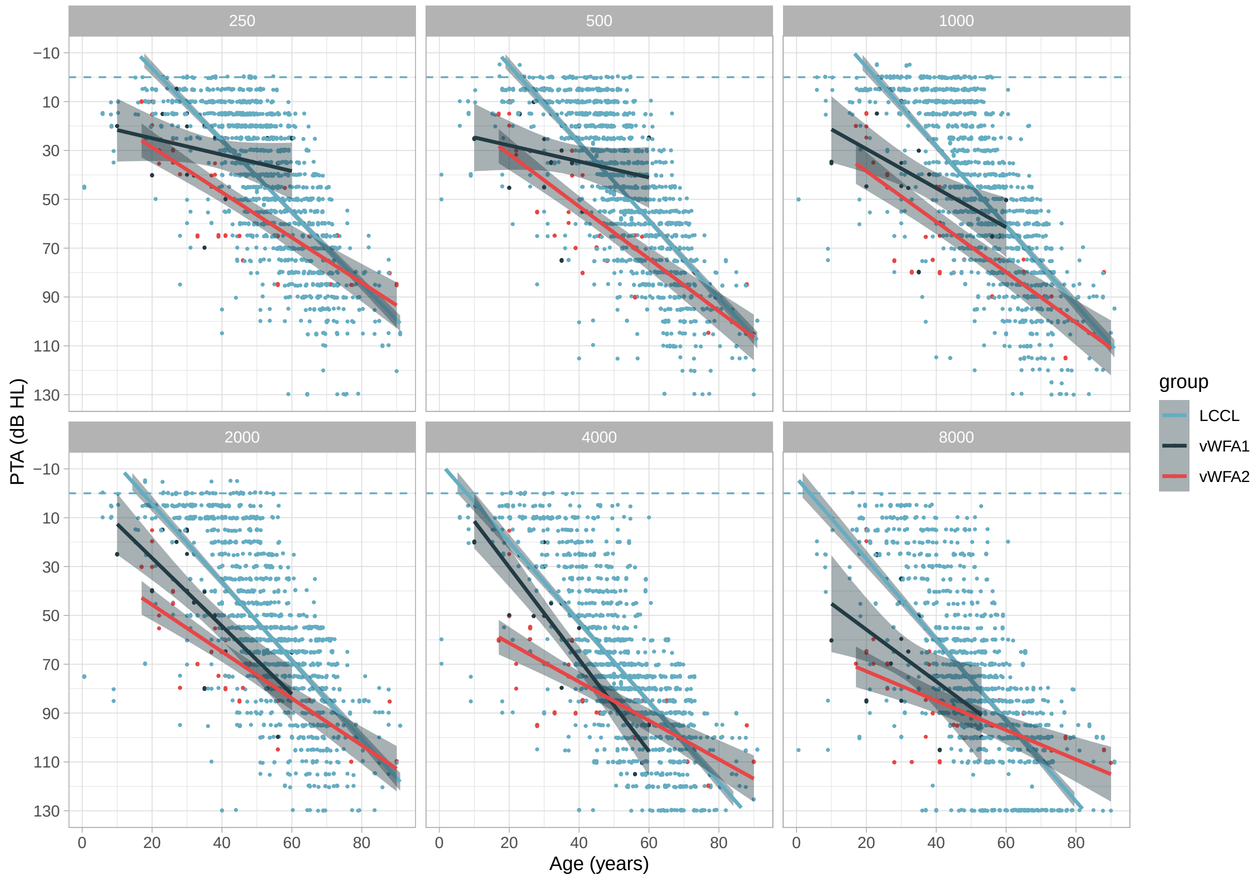
```
## Call:
##   Model: dB ~ SSlogis(Leeftijd, Asym, xmid, scal) | f
##    Data: data_all_l
##
## Coefficients:
##    Asym
##      Estimate Std. Error   t value     Pr(>|t|)
## 250  143.7417  15.611184  9.207613 1.162671e-24
## 500  136.8704  11.267641 12.147213 1.477519e-37
## 1000 133.4395   9.988906 13.358770 7.508358e-38
## 2000 127.9221   8.412012 15.207072 1.571244e-46
## 4000 128.0742   5.506317 23.259510 4.517456e-97
## 8000 136.8990   6.104713 22.425128 2.187622e-77
##    xmid
##      Estimate Std. Error  t value      Pr(>|t|)
## 250  68.56354   3.950056 17.35761  2.625853e-76
## 500  64.58276   2.837787 22.75814 2.394055e-112
## 1000 62.46279   2.596693 24.05475 5.901932e-106
## 2000 56.71821   2.511112 22.58689  5.467414e-93
## 4000 46.43124   1.686311 27.53420 3.096978e-128
## 8000 44.43887   1.880505 23.63135  1.241122e-84
##    scal
##      Estimate Std. Error  t value     Pr(>|t|)
## 250  17.45580   1.382288 12.62819 1.537972e-43
## 500  15.90778   1.135068 14.01482 1.494674e-48
## 1000 15.75834   1.114869 14.13470 6.735219e-42
## 2000 17.06383   1.281637 13.31409 1.214153e-36
## 4000 17.16685   1.229298 13.96475 1.180268e-39
## 8000 18.78508   1.406201 13.35874 6.454529e-31
##
## Residual standard error: 22.81709 on 8329 degrees of freedom
```

```r
ggplot(data = data_all_l, aes(
  x = Leeftijd,
  y = dB,
  group = group,
  color = group
)) +
  #geom_point(size=0.4) +
  geom_jitter(size = 0.4) +
  geom_smooth(method = "lm") +
  facet_wrap( ~ f) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  theme_light()
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 181 rows containing missing values (geom_point).
```

```
## Warning: Removed 66 rows containing missing values (geom_smooth).
```



Now use the linear fits to construct an ARTA

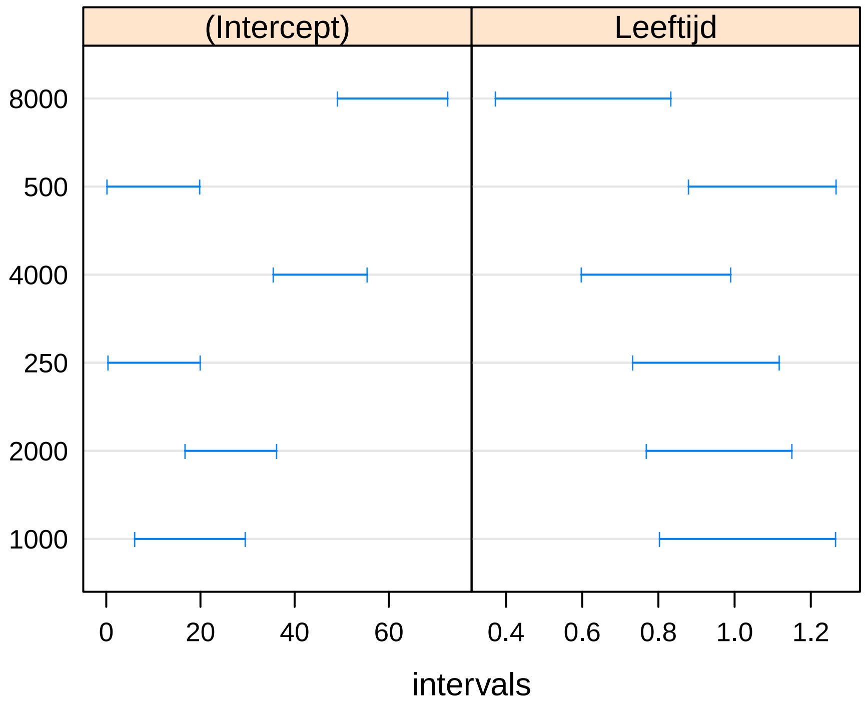
```
par(pty = "s")
#see: https://stackoverflow.com/questions/1169539/linear-regression-and-group-by-in-r
library(lme4)
arta_data <- subset(data_all_l, group == "vWFA2")
fits.plm <- lmList(dB ~ Leeftijd | frequency, data = arta_data)
```

```
## Warning: Unknown or uninitialised column: `(weights)`.

## Warning: Unknown or uninitialised column: `(offset)`.

## Warning: Unknown or uninitialised column: `(weights)`.

## Warning: Unknown or uninitialised column: `(offset)`.

## Warning: Unknown or uninitialised column: `(weights)`.

## Warning: Unknown or uninitialised column: `(offset)`.

## Warning: Unknown or uninitialised column: `(weights)`.

## Warning: Unknown or uninitialised column: `(offset)`.

## Warning: Unknown or uninitialised column: `(weights)`.

## Warning: Unknown or uninitialised column: `(offset)`.

## Warning: Unknown or uninitialised column: `(weights)`.

## Warning: Unknown or uninitialised column: `(offset)`.
```

```r
coef(fits.plm)
```

```
##      (Intercept)  Leeftijd
## 1000    17.79084 1.0340903
## 2000    26.45386 0.9592944
## 250     10.17366 0.9248101
## 4000    45.43748 0.7937173
## 500     10.02381 1.0727420
## 8000    60.80969 0.6025266
```

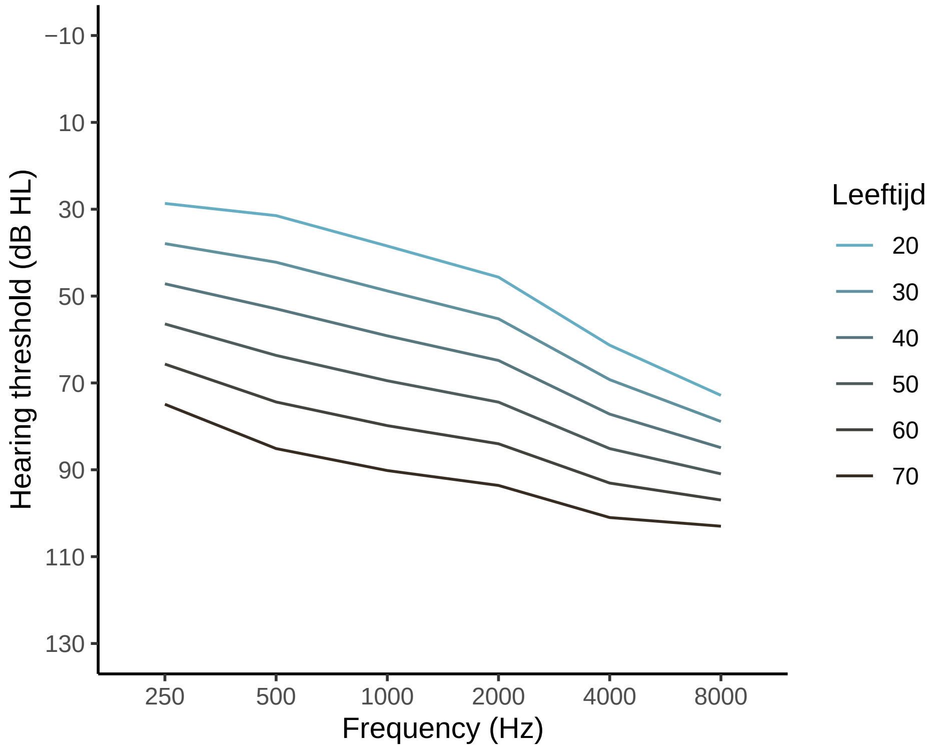
```r
ci <- confint(fits.plm)
plot(ci)
```



```r
newdat = expand.grid(
  Leeftijd = seq(20, 70, by = 10),
  frequency = c("250", "500", "1000", "2000", "4000", "8000")
)
newdat$fit <- predict(fits.plm, newdata = newdat)

head(newdat)
```

```
##   Leeftijd frequency      fit
## 1       20       250 28.66986
## 2       30       250 37.91796
## 3       40       250 47.16607
## 4       50       250 56.41417
## 5       60       250 65.66227
## 6       70       250 74.91037
```

```r
ggplot(data = newdat, aes(x = frequency, y = fit, group = Leeftijd)) +
  geom_line(aes(
    x = frequency,
    y = fit,
    group = Leeftijd,
    color = Leeftijd
  )) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  # scale_x_discrete(breaks=c("250","500","1000","2000","4000","8000"), labels=c("0.25","0.5","1","2","
  xlab("Frequency (Hz)") +
  ylab("Hearing threshold (dB HL)") +
  guides(color = guide_legend("Leeftijd")) +
  theme_classic()
```



```
library(dplyr)
fitted_models = data_all_l %>% group_by(frequency) %>% do(model = lm(dB ~ Leeftijd, data = .))
fitted_models
```

```
## Source: local data frame [6 x 2]
## Groups: <by row>
##
## # A tibble: 6 x 2
##   frequency model
## * <fct>     <list>
## 1 1000      <lm>
## 2 2000      <lm>
## 3 250       <lm>
## 4 4000      <lm>
## 5 500       <lm>
## 6 8000      <lm>
```

simple linear model: PTA is a function of the affected domain; here are two levels in this mixed model; 1: timepoints for each patient; 2: genetic domain, with domain the fixed effect and patient the random effect allowing the intercept to vary across patient (~1|pid).

```r
random_intercept <- lme(
  dB ~ frequency ,
  random = ~ 1 | pid,
  #p. 896
  method = "ML",
  na.action = na.exclude,
  control = list(opt = "optim"),
  correlation = corAR1(),
  #see p.897; timepoints are not equally spaced;use corCAR1
  data = data_all_l
)
summary(random_intercept)
```

```
## Linear mixed-effects model fit by maximum likelihood
##  Data: data_all_l
##      AIC      BIC logLik
##    70816 70879.26 -35399
##
## Random effects:
##  Formula: ~1 | pid
##         (Intercept) Residual
## StdDev:    30.66465 17.63885
##
## Correlation Structure: AR(1)
##  Formula: ~1 | pid
##  Parameter estimate(s):
##       Phi
## 0.4384654
## Fixed effects: dB ~ frequency
##                  Value Std.Error   DF  t-value p-value
## (Intercept)   43.50693 1.9538723 8068 22.26703  0.0000
## frequency2000  6.76735 0.6766851 8068 10.00074  0.0000
## frequency250  -2.45973 0.7709764 8068 -3.19041  0.0014
## frequency4000 22.58725 0.7647918 8068 29.53386  0.0000
## frequency500  -1.96178 0.6766216 8068 -2.89937  0.0037
## frequency8000 30.51271 0.7951280 8068 38.37459  0.0000
##  Correlation:
##               (Intr) fr2000 frq250 fr4000 frq500
## frequency2000 -0.173
## frequency250  -0.197  0.410
## frequency4000 -0.196  0.565  0.479
## frequency500  -0.174  0.362  0.561  0.414
## frequency8000 -0.201  0.454  0.540  0.612  0.436
##
## Standardized Within-Group Residuals:
##        Min         Q1        Med         Q3        Max
## -4.2543260 -0.5913403  0.0373855  0.6194437  3.9226279
##
## Number of Observations: 8347
## Number of Groups: 274
```

```r
anova(random_intercept)
```

```
##             numDF denDF  F-value p-value
## (Intercept)     1  8068 783.1713  <.0001
## frequency       5  8068 538.4169  <.0001
```

now add Leeftijd as fixed effect; PTA ~ Domain + Leeftijd‘ (see. e.g. p.897)

```r
timeRI <- update(random_intercept, . ~ . + Leeftijd)
summary(timeRI)
```

```
## Linear mixed-effects model fit by maximum likelihood
##  Data: data_all_l
##        AIC      BIC    logLik
##   67842.65 67912.94 -33911.32
##
## Random effects:
##  Formula: ~1 | pid
##         (Intercept) Residual
## StdDev:   0.1104064 24.88384
##
## Correlation Structure: AR(1)
##  Formula: ~1 | pid
##  Parameter estimate(s):
##       Phi
## 0.8322339
## Fixed effects: dB ~ frequency + Leeftijd
##                   Value Std.Error   DF   t-value p-value
## (Intercept)   -50.31584 1.6018876 8067 -31.41034  0.0000
## frequency2000   5.13859 0.5814116 8067   8.83812  0.0000
## frequency250   -1.30225 0.7394264 8067  -1.76116  0.0782
## frequency4000  20.24899 0.7377572 8067  27.44668  0.0000
## frequency500   -0.94485 0.5808559 8067  -1.62665  0.1038
## frequency8000  29.31740 0.8033935 8067  36.49196  0.0000
## Leeftijd        1.93967 0.0258142 8067  75.13958  0.0000
##  Correlation:
##               (Intr) fr2000 frq250 fr4000 frq500 fr8000
## frequency2000 -0.196
## frequency250  -0.228  0.283
## frequency4000 -0.262  0.635  0.419
## frequency500  -0.176  0.195  0.631  0.284
## frequency8000 -0.304  0.464  0.551  0.713  0.369
## Leeftijd      -0.819  0.018 -0.023  0.032 -0.015  0.049
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.07541243 -0.70824164 -0.07201688  0.58548467  5.01653320
##
## Number of Observations: 8347
## Number of Groups: 274
```

```r
sessionInfo()
```

```
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS  10.15.4
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lattice_0.20-40  dplyr_0.8.4      tidyr_1.0.2      forcats_0.5.0
##  [5] kableExtra_1.1.0 knitr_1.28       lme4_1.1-21      Matrix_1.2-18
##  [9] nlme_3.1-145     readxl_1.3.1     sjPlot_2.8.2     drc_3.0-1
## [13] MASS_7.3-51.5    ggthemr_1.1.0    ggplot2_3.3.0
##
## loaded via a namespace (and not attached):
##  [1] insight_0.8.1     webshot_0.5.2     httr_1.4.1        tools_3.6.0
##  [5] backports_1.1.6   utf8_1.1.4        R6_2.4.1          sjlabelled_1.1.3
##  [9] mgcv_1.8-31       colorspace_1.4-1  withr_2.2.0       tidyselect_1.0.0
## [13] emmeans_1.4.5     curl_4.3          compiler_3.6.0    performance_0.4.4
## [17] cli_2.0.2         rvest_0.3.5       xml2_1.3.1        sandwich_2.5-1
## [21] labeling_0.3      bayestestR_0.5.2  scales_1.1.0      mvtnorm_1.1-0
## [25] readr_1.3.1       stringr_1.4.0     digest_0.6.25     foreign_0.8-76
## [29] minqa_1.2.4       rmarkdown_2.1     rio_0.5.16        pkgconfig_2.0.3
## [33] htmltools_0.4.0   plotrix_3.7-7     rlang_0.4.5       rstudioapi_0.11
## [37] generics_0.0.2    farver_2.0.3      zoo_1.8-7         gtools_3.8.1
## [41] zip_2.0.4         car_3.0-6         magrittr_1.5      parameters_0.5.0
## [45] Rcpp_1.0.4.6      munsell_0.5.0     fansi_0.4.1       abind_1.4-5
## [49] lifecycle_0.2.0   stringi_1.4.6     multcomp_1.4-12   yaml_2.2.1
## [53] carData_3.0-3     grid_3.6.0        sjmisc_2.8.3      crayon_1.3.4
## [57] ggeffects_0.14.1  haven_2.2.0       splines_3.6.0     sjstats_0.17.9
## [61] hms_0.5.3         pillar_1.4.3      boot_1.3-24       estimability_1.3
## [65] effectsize_0.2.0  codetools_0.2-16  glue_1.4.0        evaluate_0.14
## [69] data.table_1.12.8 modelr_0.1.6      vctrs_0.2.4       nloptr_1.2.1
## [73] cellranger_1.1.0  gtable_0.3.0      purrr_0.3.3       assertthat_0.2.1
## [77] xfun_0.12         openxlsx_4.1.4    xtable_1.8-4      broom_0.5.5
## [81] survival_3.1-8    viridisLite_0.3.0 tibble_3.0.1      TH.data_1.0-10
## [85] ellipsis_0.3.0
```

Code Appendix

```r
library(ggplot2)
library(ggthemr)
library(drc)
library(sjPlot)
library("readxl")
library("nlme")
library(lme4)
library(knitr)
library(kableExtra)
library(forcats)
library(tidyr)
library(dplyr)
ggthemr('fresh')
```

```r
data_raw <- read_excel("../data/raw_data/database_20-04-2020.xlsx")
data_raw$group = factor(data_raw$Domain)
#leave out data with only n=1 dataset per domain/certain unpublished data.
data_subset <-
  subset(data_raw, Smits == 'no' & Domainrec != 1 & group != "Ivd1")
data <-
  subset(data_subset, select = c('pid', 'group', 'Leeftijd', 'PTA54ADS'))

#drop unused levels from a factor in a dataframe, e.g. groups that have no entries anymore.
data <- droplevels(data)
# save processed and cleaned data
save(data,file="../data/processed_data/data_pta_age_group.Rda")

#check for NaNs in PTA and Leeftijd, should be 0.
nrow(data[is.na(data$PTA54ADS) | is.na(data$Leeftijd),])

t1 <- data %>%                      # take the data.frame "data"
  filter(!is.na(pid)) %>%      # Using "data", filter out all rows with NAs in aa
  group_by(group) %>%          # Then, with the filtered data, group it by "group"
  summarise("# subjects" = n_distinct(pid))    # Now summarise with unique elements per group
kable(t1, caption = "Table 1. The number of subjects per group",) %>%
  kable_styling(bootstrap_options = "striped", full_width = F)
num_meas_per_id <-
  aggregate(PTA54ADS ~ pid , data, function(x)
    length(unique(x)))
t2 <- table(num_meas_per_id$PTA54ADS)
kable(t2,
      caption = "Table 2. The number of subjects that each have n audiograms",
      col.names = c("# audiograms", "# subjects")) %>%
  kable_styling(bootstrap_options = "striped", full_width = F)
#number (n) of counts (i.e. audiograms) per subject (pid)
summarytable <- data %>% count(group, pid)

ggplot(data = summarytable, aes(x = n, fill = group)) +
  geom_histogram(binwidth = 1) +
  facet_wrap(~ group) +
  xlab("# of measurements") +
  ylab("Frequency (# patients)")
dev.print(pdf, '../results/histogram_number_meas_pid.pdf')
ggplot(data, aes(
  x = Leeftijd,
  y = PTA54ADS,
  group = pid,
  color = group
)) +
  geom_point(aes(colour = factor(group))) +
  geom_line(data = data, size = 1, alpha = .3) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
```

```r
dev.print(pdf, '../results/pta_age_pid_groups.pdf')
lccl = subset(data, group == "LCCL")
ggplot(lccl,
       aes(x = Leeftijd, y = PTA54ADS),
       group = pid,
       color = group) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  geom_point(aes(colour = factor(group))) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130,-10)) +
  theme_light()
dev.print(pdf, '../results/pta_age_pid_lccl.pdf')
lin_fit <-
  nls(PTA54ADS ~ a * Leeftijd + b,
      data = lccl,
      start = list(a = 1.5, b = 0))
summary(lin_fit)
nls_fit <-
  nls(PTA54ADS ~ a * Leeftijd ^ b,
      data = lccl,
      start = list(a = 0.05, b = 1.5))
summary(nls_fit)
startvec <- c(Asym = 120, xmid = 50, scal = 15)
nls_logis <- nls(PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal),
                 data = lccl,
                 start = startvec)
summary(nls_logis)
anova(nls_fit, nls_logis)
newdat = expand.grid(Leeftijd = seq(0, 100, by = 1))
newdat$lin <- predict(lin_fit, newdata = newdat)
newdat$pta_logistic <- predict(nls_logis, newdata = newdat)
newdat$pta_power <- predict(nls_fit, newdata = newdat)
#newdat
ggplot(lccl, aes(x = Leeftijd, y = PTA54ADS)) +
  geom_point(aes(colour = factor(group))) +
  geom_line(data = newdat,
            aes(y = lin),
            size = 1,
            col = 'blue') +
  geom_line(data = newdat, aes(y = pta_logistic), size = 1) +
  geom_line(data = newdat,
            aes(y = pta_power),
            size = 1,
            col = 'red') +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
```

```r
dev.print(pdf, '../results/pta_age_lccl_fits.pdf')

fit0 <-
  nls(PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal), data = data)
summary(fit0)
coef(fit0)
# https://stats.stackexchange.com/questions/27273/how-do-i-fit-a-nonlinear-mixed-effects-model-for-repe
# https://stats.stackexchange.com/questions/316801/how-to-compare-logistic-regression-curves
fit1 <- nls(
  PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal),
  data = data,
  start = list(
    Asym = rep(120, 1),
    xmid = rep(50, 3),
    scal = rep(15, 1)
  )
)
summary(fit1)
fit2 <-
  nls(
    PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid[group], scal[group]),
    data = data,
    start = list(
      Asym = rep(120, 1),
      xmid = rep(50, 3),
      scal = rep(15, 3)
    )
  )
summary(fit2)
plot(profile(fit2))
fit3 <-
  nls(
    PTA54ADS ~ SSlogis(Leeftijd, Asym[group], xmid[group], scal[group]),
    data = data,
    start = list(
      Asym = rep(120, 3),
      xmid = rep(50, 3),
      scal = rep(15, 3)
    )
  )
summary(fit3)
anova(fit0, fit1, fit2, fit3)
newdat = expand.grid(Leeftijd = seq(0, 100, by = 1),
                     group = c("LCCL", "vWFA1", "vWFA2"))
newdat$fit <- predict(fit2, newdata = newdat)
ggplot(data, aes(
  x = Leeftijd,
  y = PTA54ADS,
  group = pid,
  color = group
)) +
  geom_point(aes(colour = factor(group)), alpha = .4) +
  geom_line(data = data, size = 1, alpha = .2) +
  geom_line(data = newdat,
            aes(
              y = fit,
              group = group,
              colour = factor(group)
            ),
            size = 1) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
dev.print(pdf, '../results/pta_age_pid_groups_fits.pdf')
#https://stackoverflow.com/questions/14439770/filter-rows-in-dataframe-by-number-of-rows-per-level-of-a
pidlengths <- ave(as.numeric(data$pid),
                  data$pid, FUN = length)
#df2 <- lccl[pidlengths > 5, ]
df2 <- data[pidlengths > 2,]
t3 <- with(df2, table(group))

models <-
  nlsList(PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal) |
            pid, data = df2)
data_id <- subset(df2, pid == "147")
data_id
ggplot(data = data_id,  aes(x = Leeftijd, y = PTA54ADS)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  xlim(0,100)+
  theme_light()
df2$Pred <- predict(models)
df2_na <- na.omit(df2)
df2_na_stats <- with(df2_na, table(group, pid))
df2_na_stats
le <- unique(df2_na$pid)
newdat = expand.grid(Leeftijd = seq(0, 100, by = 1), pid = le)
newdat$prednlm <- predict(models, newdata = newdat)
#https://stackoverflow.com/questions/37122994/plotting-a-list-of-non-linear-regressions-with-ggplot
#https://aosmith.rbind.io/2018/11/16/plot-fitted-lines/
ggplot(data = df2_na,  aes(x = Leeftijd, y = PTA54ADS, colour = pid)) +
  geom_point() +
  geom_line(data = newdat, aes(y = prednlm)) +
  facet_wrap(~ pid) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 25)) +
  scale_y_reverse(breaks = seq(0, 120, 40), limits = c(130,-10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()

dev.print(pdf, '../results/pta_age_pid_lccl_ind_fits.pdf')
ggplot(data = df2_na,  aes(x = Leeftijd, y = PTA54ADS, colour = pid)) +
  geom_point() +
  geom_line(data = newdat, aes(y = prednlm, group = pid)) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130,-10)) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  theme_light()
dev.print(pdf, '../results/pta_age_pid_lccl_ind_fits_overlay.pdf')
nm1 <-
  nlmer(
    PTA54ADS ~ SSlogis(Leeftijd, Asym, xmid, scal) ~ Asym + xmid + scal |
      pid,
    df2_na,
    method="ML",
    start = c(Asym = 100, xmid = 60, scal = 15),
    corr = FALSE
  )
summary(nm1)
plot(ranef(nm1,augFrame=T))
params <- coef(nm1)
head(params)
#sjplot(nm1)
require(lattice)
qqmath(ranef(nm1, condVar = TRUE))
data_all <-
  subset(
    data_subset,
    select = c(
      'pid',
      'group',
      'Leeftijd',
      '250.AD',
      '500.AD',
      '1000.AD',
      '2000.AD',
      '4000.AD',
      '8000.AD',
      '250.AS',
      '500.AS',
      '1000.AS',
      '2000.AS',
      '4000.AS',
      '8000.AS'
    )
  )
head(data_all)
```

```r
tidier <- data_all %>%
  gather(f, dB,-pid,-group,-Leeftijd)
data_all_l <- tidier %>%
  separate(f, into = c("frequency", "ear"), sep = "\\.")
#head(data_all_l)
data_all_l$frequency = factor(data_all_l$frequency)
data_all_l$ear = factor(data_all_l$ear)
data_all_l <- na.omit(data_all_l)
head(data_all_l)
p <- data_all_l %>%
  mutate(frequency = fct_relevel(frequency, "250", "500", "1000", "2000", "4000", "8000")) %>%
  ggplot(aes(x = frequency, y = dB)) +
  #geom_bar(stat="identity") +
  #geom_histogram() +
  geom_violin() +
  facet_wrap( ~ group, ncol = 3) +
  #geom_point()
  xlab("Frequency (Hz)") +
  ylab("Hearing level (dB)") +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  theme_light()
  #theme_classic()
p
dev.print(pdf, '../results/violin_plot_HL_groups.pdf')
data_all_l$f = factor(data_all_l$frequency,
                      levels = c('250', '500', '1000', '2000', '4000', '8000'))
ggplot(data_all_l, aes(
  x = Leeftijd,
  y = dB,
  group = f,
  color = group
)) +
  geom_point(size = 0.7) +
  facet_wrap( ~ f) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 25)) +
  scale_y_reverse(breaks = seq(0, 120, 40), limits = c(130, -10)) +
  #scale_y_reverse(limits=c(130,-10)) +
  xlab("Age (years)") +
  ylab("Hearing level (dB HL)") +
  theme_light()
dev.print(pdf, '../results/HL_age_frequency_groups.pdf')

models <-
  nlsList(
    dB ~ SSlogis(Leeftijd, Asym, xmid, scal) |
      f,
    data = data_all_l,
    start = c(Asym = 100, xmid = 60, scal = 15)
  )
summary(models)

ggplot(data = data_all_l, aes(
  x = Leeftijd,
  y = dB,
  group = group,
  color = group
)) +
  #geom_point(size=0.4) +
  geom_jitter(size = 0.4) +
  geom_smooth(method = "lm") +
  facet_wrap( ~ f) +
  xlab("Age (years)") +
  ylab("PTA (dB HL)") +
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  theme_light()

par(pty = "s")
#see: https://stackoverflow.com/questions/1169539/linear-regression-and-group-by-in-r
library(lme4)
arta_data <- subset(data_all_l, group == "vWFA2")
fits.plm <- lmList(dB ~ Leeftijd | frequency, data = arta_data)
coef(fits.plm)
ci <- confint(fits.plm)
plot(ci)
newdat = expand.grid(
  Leeftijd = seq(20, 70, by = 10),
  frequency = c("250", "500", "1000", "2000", "4000", "8000")
)
newdat$fit <- predict(fits.plm, newdata = newdat)

head(newdat)

ggplot(data = newdat, aes(x = frequency, y = fit, group = Leeftijd)) +
  geom_line(aes(
    x = frequency,
    y = fit,
    group = Leeftijd,
    color = Leeftijd
  )) +
  scale_y_reverse(breaks = seq(-10, 130, 20), limits = c(130, -10)) +
  # scale_x_discrete(breaks=c("250","500","1000","2000","4000","8000"), labels=c("0.25","0.5","1","2","
  xlab("Frequency (Hz)") +
  ylab("Hearing threshold (dB HL)") +
  guides(color = guide_legend("Leeftijd")) +
  theme_classic()
library(dplyr)
fitted_models = data_all_l %>% group_by(frequency) %>% do(model = lm(dB ~ Leeftijd, data = .))
fitted_models
random_intercept <- lme(
  dB ~ frequency ,
  random = ~ 1 | pid,
  #p. 896
  method = "ML",
  na.action = na.exclude,
  control = list(opt = "optim"),
  correlation = corAR1(),
  #see p.897; timepoints are not equally spaced;use corCAR1
  data = data_all_l
)
summary(random_intercept)
anova(random_intercept)
timeRI <- update(random_intercept, . ~ . + Leeftijd)
summary(timeRI)
sessionInfo()
```