Growth Curving

Trevor Day

2/22/2020

Contents

Setup **1**

Introduction **1**

Basic Gompertz **2**

Gompertz II Electric Pertzaloo **4**
Reducing observations 6
Modeling across subjects 7
Distribution of k_u 33

Lexical vs syntactic **35**
L v S across subjects 36
SE plots 61
Bootstrapping significance of factor structure 64
All curves 64

Setup

Load packages `tidyverse`, `plyr`, `ggplot2`, `scales`, `ggrepel`, `gridExtra`, `growthcurver`, `educate` (from Andy).

Load the first person with five points

```
test <- read_csv("data/test-subjs.csv") %>%
          arrange(data_id, age) %>%
          filter(data_id == 176427)
```

```
## Parsed with column specification:
## cols(
##   data_id = col_double(),
##   age = col_double(),
##   inventory = col_double()
## )
```

Introduction

Throughout, I will be using notation from XXX, where:

- W_0: The starting population. For our purposes, can be estimated at 1 or less.

- A: The upper asymptote, here I use $680 + 1$, the size of MCDI plus one so the participant is actually modeled as learning all words.
- t: Time. Here, age in months.
- k_g: *Relative* (or *intrinsic*) growth rate.
- k_U: Absolute maximum growth rate (at inflection). Given as $k_U = \frac{A \times k_g}{e}$. Has units $\frac{\text{words}}{\text{month}}$.
- W_i, T_i: Value and time at inflection. W_i is fixed at $\frac{A}{e} \approx 251$, and T_i can be solved for.

```
# Upper asymptote
A <- 681

# Lower asymptote: the smallest value R can represent
W_0 <- .Machine$double.eps
```

Basic Gompertz

The Gompertz equation given by `growthcurver`, converted to the new notation is:

$$y = \frac{A}{1 + \frac{A - W_0}{W_0}(\exp(-k_g \times x))}$$

This can be created like so:

```
gomp.fit <- SummarizeGrowth(test$age, test$inventory)
gomp.fit
```

```
## Fit data to K / (1 + ((K - N0) / N0) * exp(-r * t)):
##       K    N0  r
##   val:   615.792 0.046   0.394
##   Residual standard error: 2.757251 on 2 degrees of freedom
##
## Other useful metrics:
##   DT 1 / DT  auc_l   auc_e
##   1.76   5.7e-01 3447.73 3501.5
```

We write a quick function to calculate the fit over a given range. We use `0.01` as a step as a day is `0.03` months. See the first part of `predict.gomp`.

```
predict.gomp <- function(fit, range = seq(0, 36, by = 0.01), max = 681,
                         ci = FALSE) {

  if (class(fit) == "gcfit"){

    # Result from growth curver
    A <- fit$vals$k
    W_0 <- fit$vals$n0
    k_g <- fit$vals$r
    k_g.se <- fit$vals$r_se

    if (ci)
      k_g.se <- 1.96 * k_g.se

    result <- A/(1 + ((A - W_0)/W_0) * exp(-1 * k_g * range))
    result.lo <- A/(1 + ((A - W_0)/W_0) * exp(-1 * (k_g - k_g.se) * range))
    result.hi <- A/(1 + ((A - W_0)/W_0) * exp(-1 * (k_g + k_g.se) * range))
```

```
  } else if (class(fit) == "nls") {

    # Result from my constrained model
    A <- max
    W_0 <- .Machine$double.eps
    k_g <- coef(fit) %>% unname()
    k_g.se <- summary(fit)$coefficients[1, 2]

    if (ci)
      k_g.se <- 1.96 * k_g.se

    result <- W_0 * (A / W_0) ^ (1 - exp(-1 * k_g * range))
    result.lo <- W_0 * (A / W_0) ^ (1 - exp(-1 * (k_g - k_g.se) * range))
    result.hi <- W_0 * (A / W_0) ^ (1 - exp(-1 * (k_g + k_g.se) * range))


  }


  out <- cbind(range, result, result.lo, result.hi) %>%
          as.data.frame()

  return(out)

}
```

```
## Warning: Removed 1000 rows containing missing values (geom_path).
```

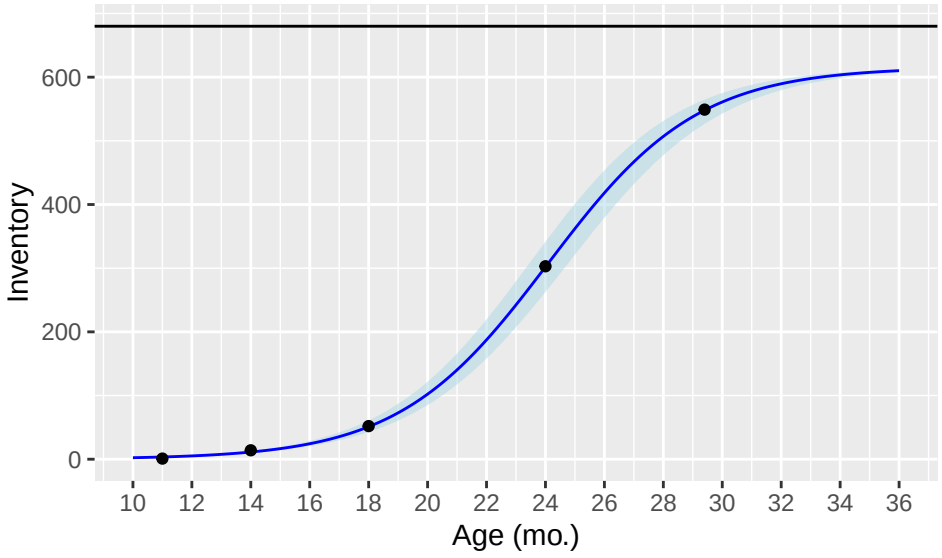


Figure 1: Gompertz curve from growthcurver package

The standard error of k_g is used for two additional lines and plotted here as an error range.

However, this has the limitation that A is estimated by the function, which means it may pick an asymptote well below 680.

I also want to point out this is a remarkably good fit; other subjects are not so good.

Gompertz II Electric Pertzaloo

Through trial and error, I identified a form of the Gompertz equation that works well with `nls` and allows A and W_0 as parameters. As a result, only k_g has to converge for this model.

For A, we use $680 + 1$, as otherwie, a kid who never learns the 680th word on the MCDI would be modeled. Although, as a reminder, we are most interested in the shape of the first two-thirds of the model, since the greater asymptote is an artifact of the instrument.

This equation is:

$$y = W_0 \left(\frac{A}{W_0} \right)^{1-\exp(-k_G \times t)}$$

The absolute growth rate, k_u, which technically has the units words/month, can be solved for as follows:

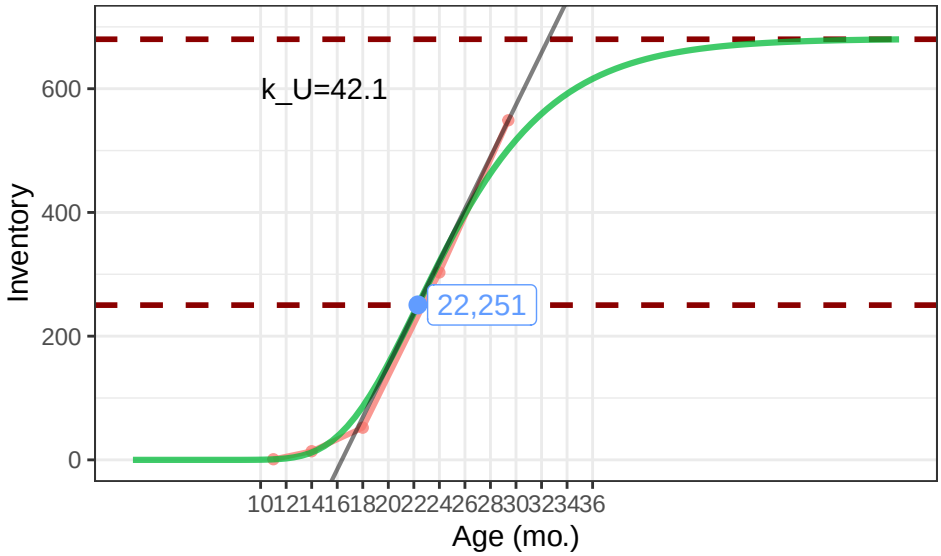
$$k_u = \frac{A \times k_g}{e} = \frac{681}{e} k_g \approx 251 \times k_g$$

Function and solver below:

```
gomp2.fit <- function(data, response = "inventory", max = 681, max.iter = 50) {

  A <- max
  W_0 <- .Machine$double.eps

  # This is the formula that works best to solve in R
  # 19 refers to its number in the Tjorve and Tjorve paper
  fit19 <- NULL
  try( fit19 <- nls(as.formula(paste(response,
                                "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),
                    data = data,
                    start = list(k_g = .1),
                    control = list(maxiter = max.iter)) )

  if (is.null(fit19))
    fit19 <- NA

  return(fit19)

}

# Estimate relative growth rate
gomp2 <- gomp2.fit(test)
gomp2.kg <- summary(gomp2)$coefficients[1, 1]

# Solve for abs growth rate
gomp2.kU <- 681 * gomp2.kg / exp(1)

# Solve for a given X or Y, given kg
solve.gomp2 <- function(x = NA, y = NA, k_g, A = 681) {

  W_0 <- .Machine$double.eps

  if (is.na(y) & !is.na(x))
```

```
    result <-  W_0 * (A / W_0) ^ (1 - exp(-k_g * x))
  else if (is.na(x) & !is.na(y))
    result <- -log(1 - log(y / W_0) / log(A / W_0)) / k_g
  else
    message("Exactly one of x/y must be specified")

  return(result)

}
```

Now we plot the Gompertz fit, including the point of inflection, and the maximum absolute growth rate, the rate at $\{T_i, W_i\}$. The value of W_i, $\frac{A}{e} = 251$ is a feature of the model, so we solve for T_i given a k_g. k_u is the slope of the tangent line at that point.

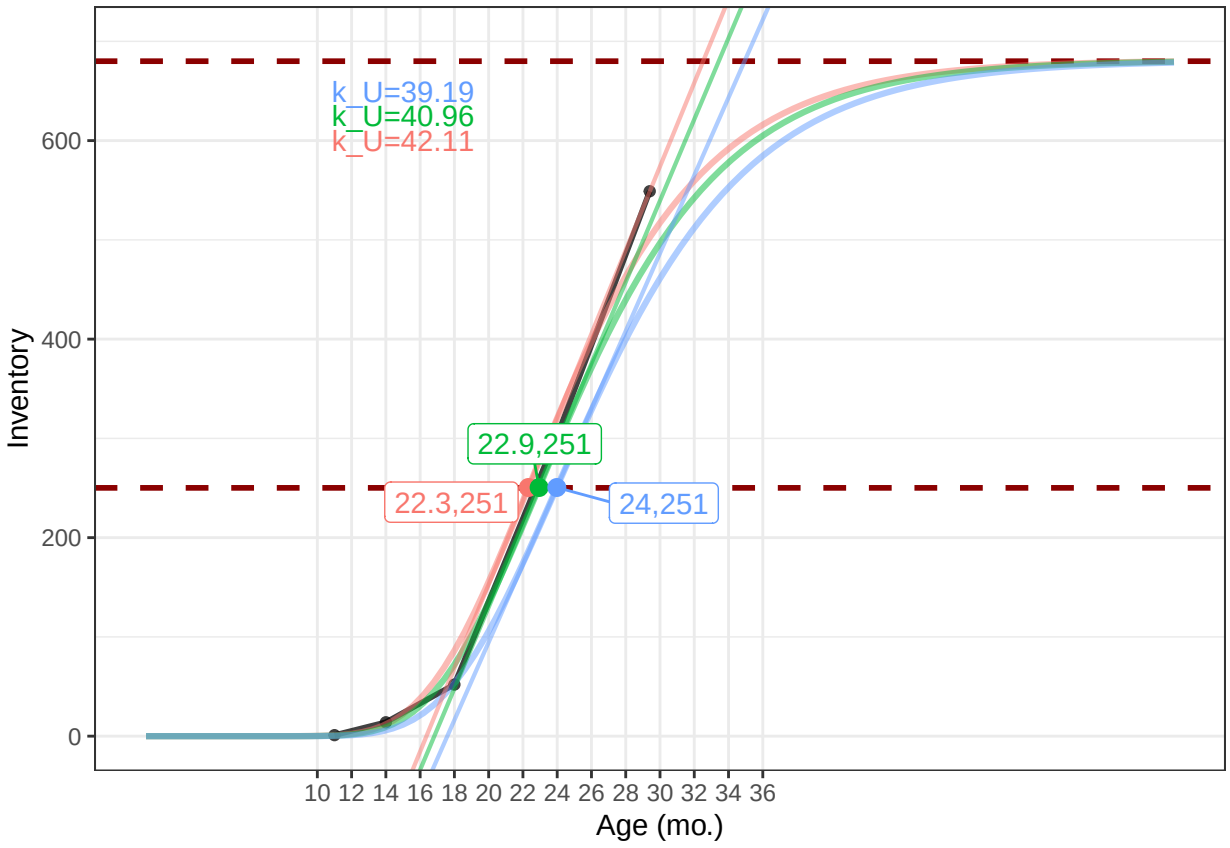



```
p1 <- ggplot(NULL) +
        geom_point(data = test,
                   aes(x = age, y = inventory),
                   size = 2,
                   color = "black") +
        geom_line(data = test,
                   aes(x = age, y = inventory),
                   color = "black",
                   size = 1) +
        scale_x_continuous(limits = c(0, 60),
                           breaks = seq(10, 36, by = 2),
                           minor_breaks = NULL) +
        scale_y_continuous(limits = c(NA, 700)) +
        labs(x = "Age (mo.)", y = "Inventory") +
        geom_hline(yintercept = 680, size = 1, linetype = "dashed",
                   color = "darkred")

p2 <- p1 +
        stat_function(data = test,
                      fun = function(age) { W_0 * (A / W_0) ^ (1 - exp(-gomp2.kg * age)) },
                      color = colors[2],
```

```
                      size = 1.1,
                      alpha = 0.75)

p3 <- p2 +
        geom_hline(yintercept = 680 / exp(1), size = 1, linetype = "dashed",
                   color = "darkred")

p4 <- p3 +
        geom_abline(intercept = b,
                    slope = gomp2.kU,
                    color = "red",
                    size = 1) +
        geom_label_repel(aes(x = T_i, y = W_i),
                         size = 4,
                         label = paste0(round(T_i), ",", round(W_i)),
                         nudge_x = 5,
                         color = "red") +
        annotate("text",
                 x = 15, y = 600,
                 label = paste0("k_U=", round(gomp2.kU, 1)))


names <- paste0("plots/gompertz", 1:4, ".png")
plots <- list(p1, p2, p3, p4)

for (i in 1:4) {

  png(names[i], width = 6, height = 5, res = 300, units = "in")
  print(plots[[i]])
  dev.off()

}
```

Reducing observations

Here we are modeling five points. However, not everyone has five points, so how few can we use?



It looks like four and five give reasonably close values, with three not doing too poorly for this individual.

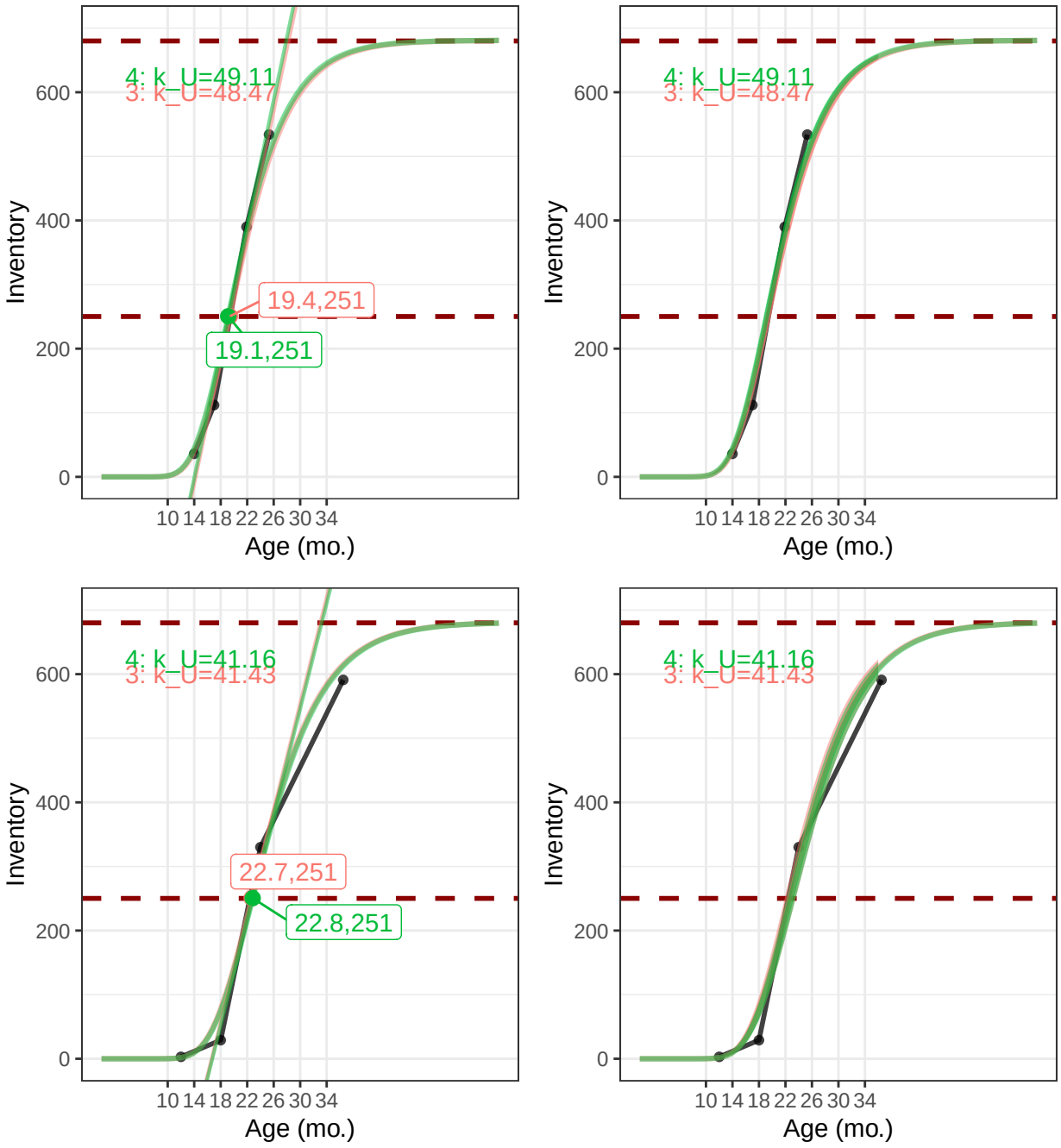
Modeling across subjects

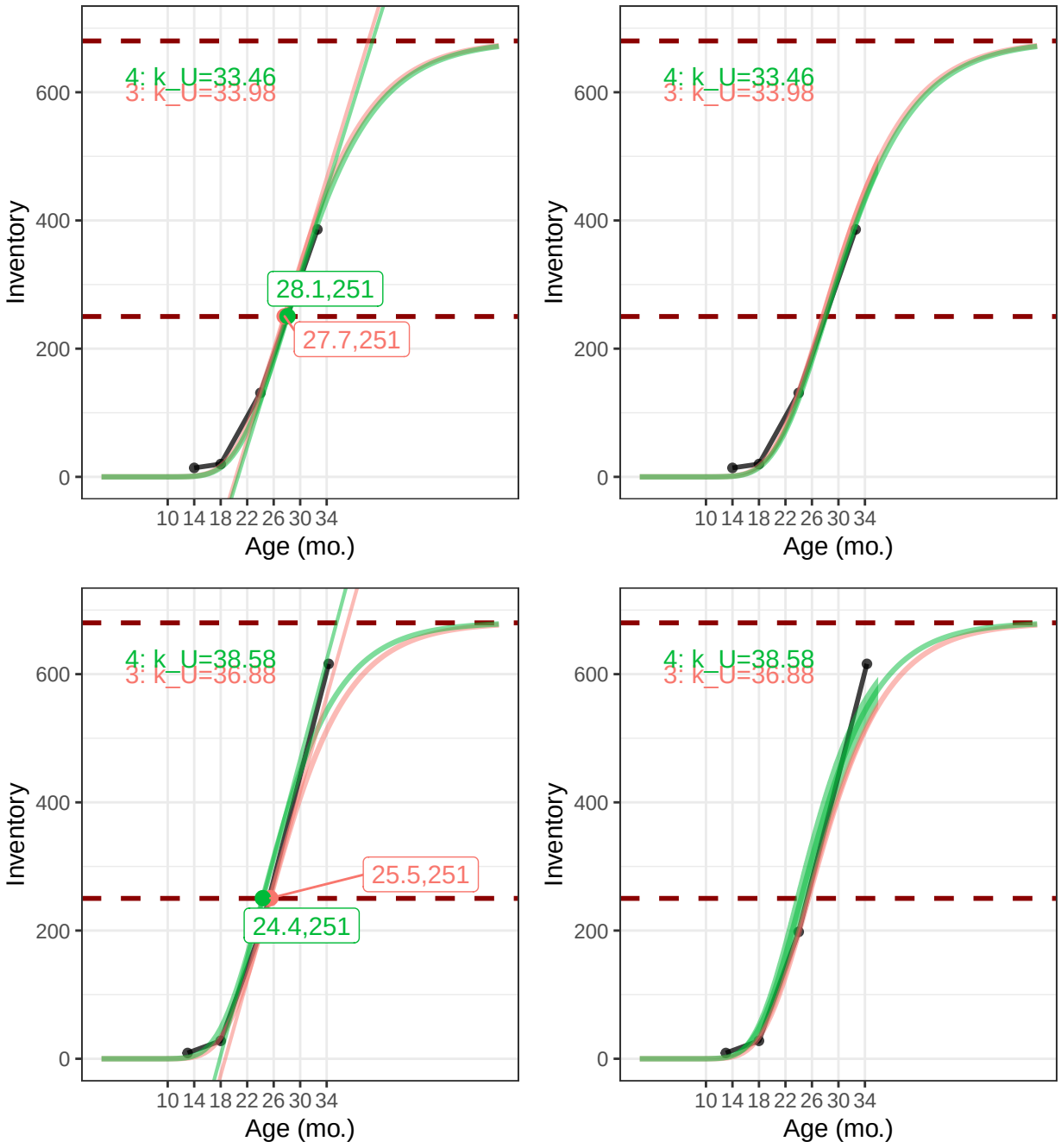
The `test` file contains data from 51 subjects who have at least four observations. Here, we're going to repeat the same steps as above and plot all 51 to examine how consistent curves are with other data.

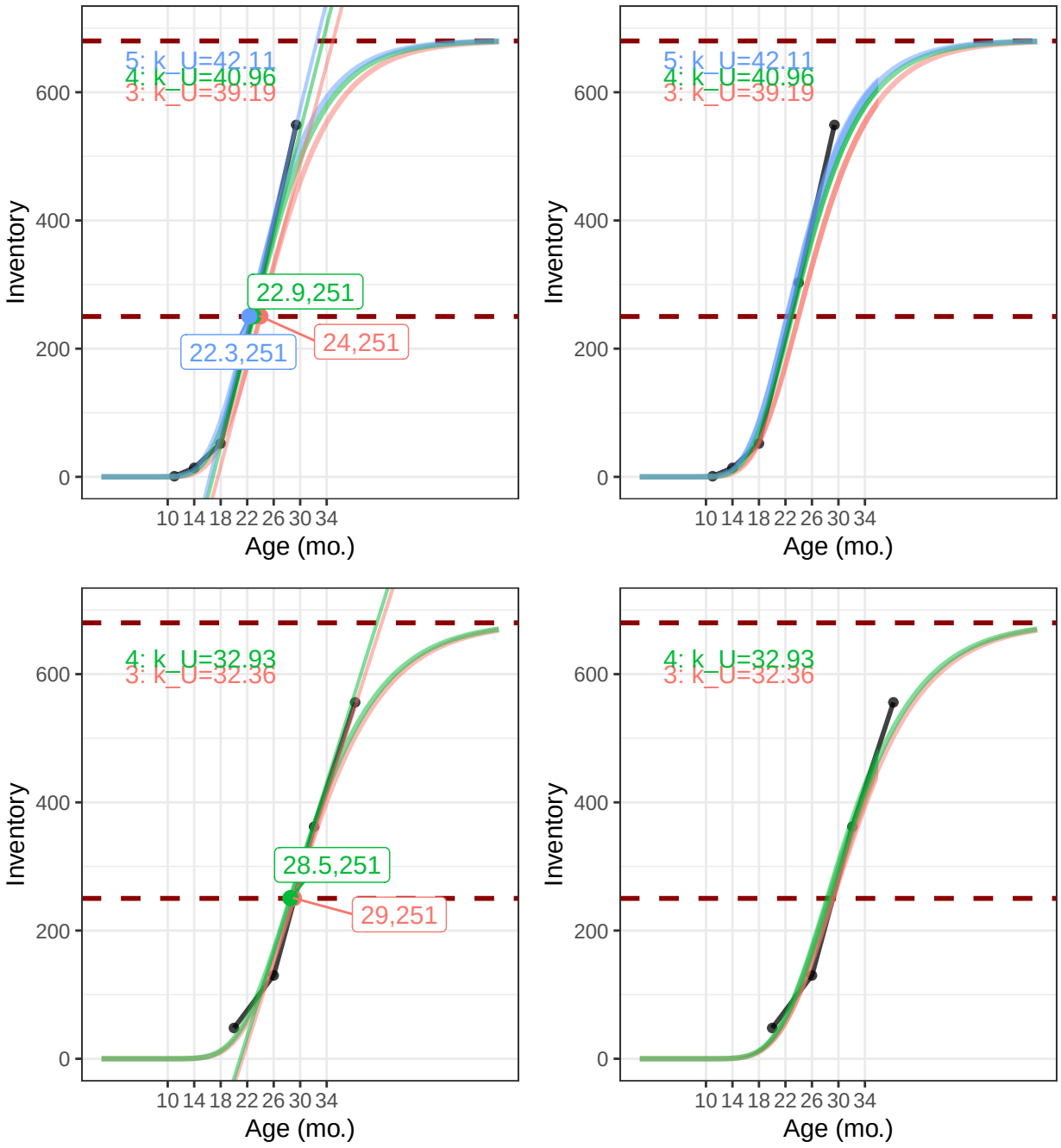
```
# Load data as a list
test_all <- read_csv("data/test-subjs.csv") %>%
              arrange(data_id, age) %>%
              split(., f = .$data_id)
```

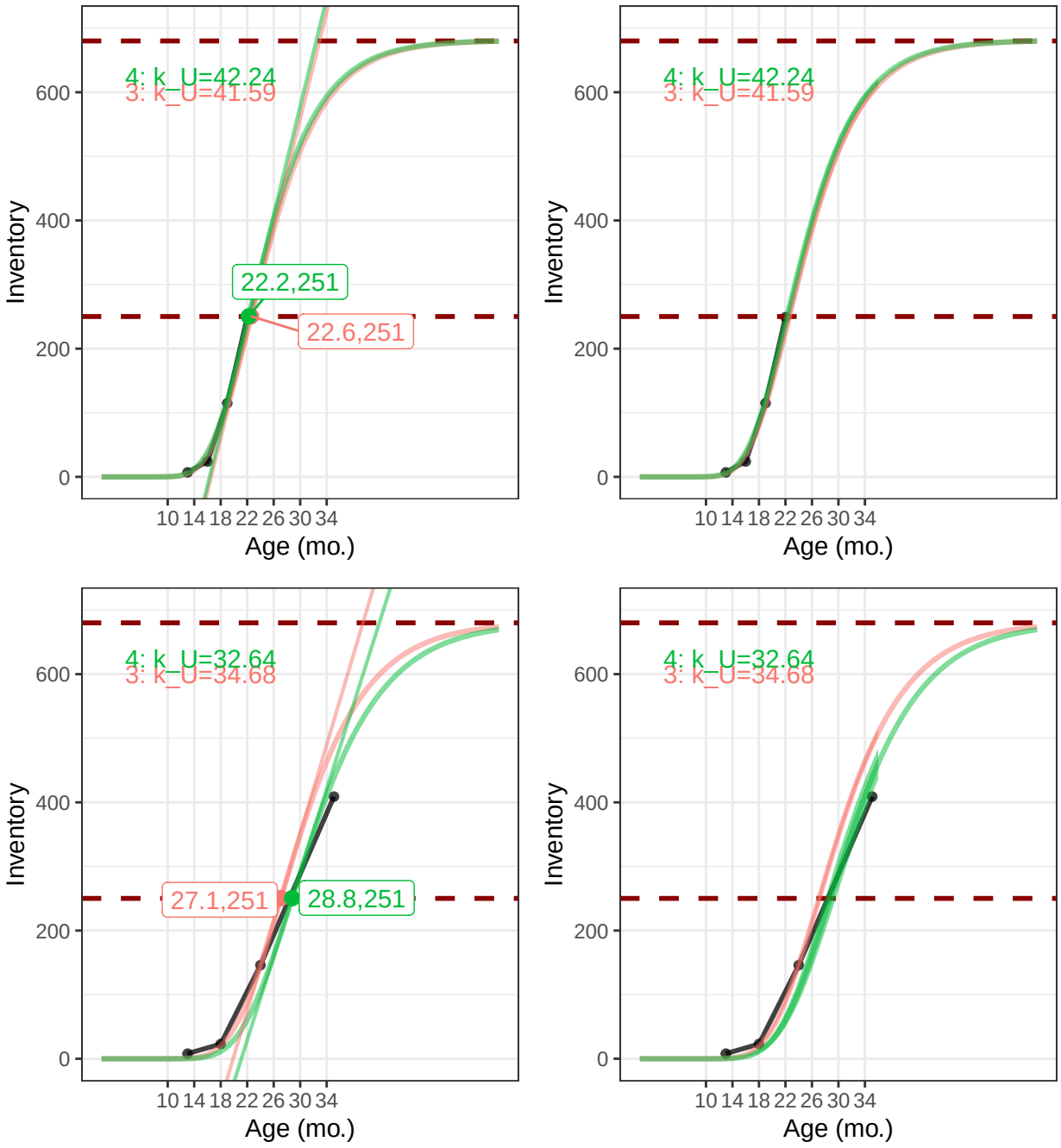
```
## Parsed with column specification:
## cols(
##   data_id = col_double(),
##   age = col_double(),
##   inventory = col_double()
## )
```

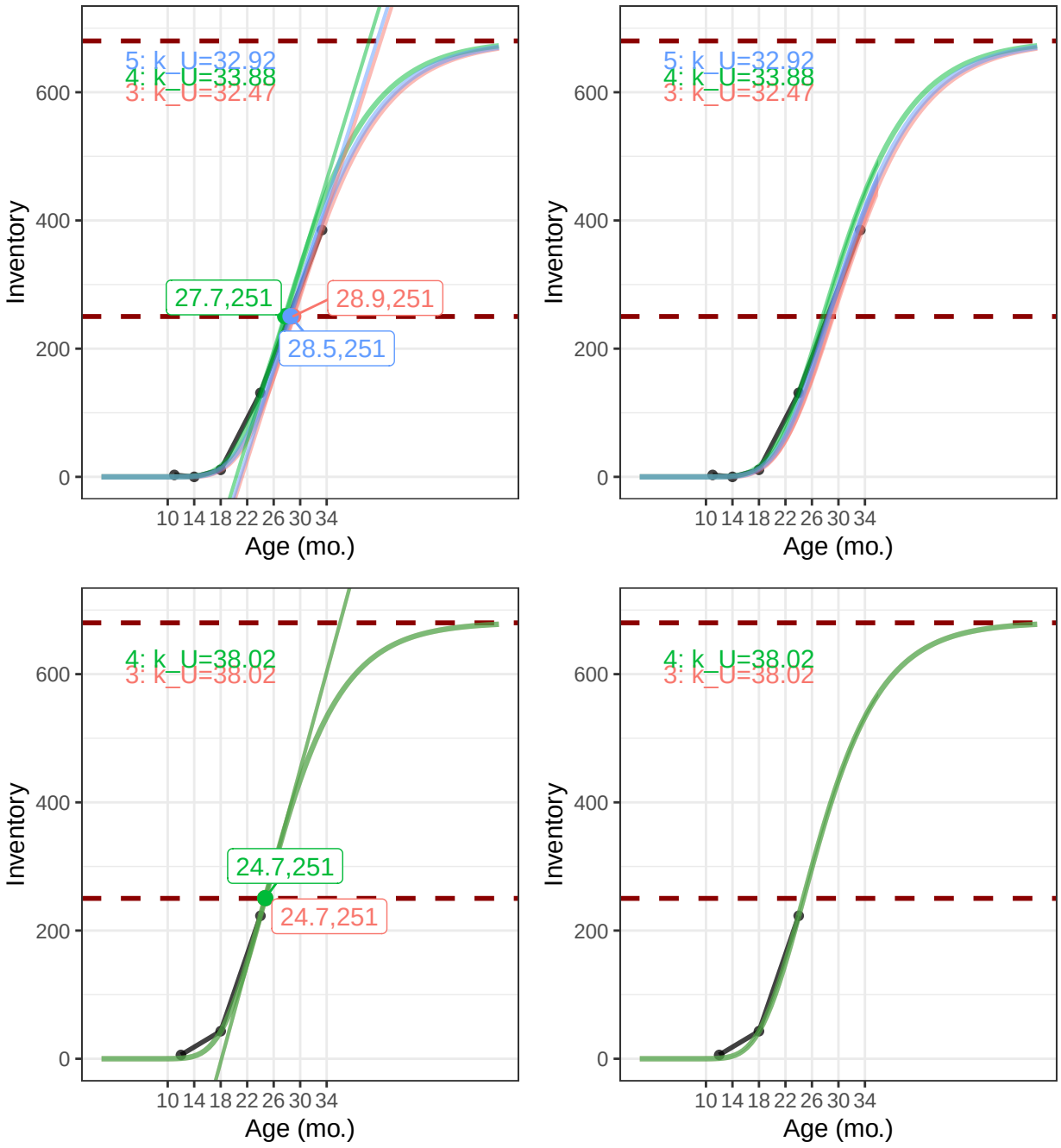
I'm going to invisibly fit the curves here, with tangent lines and $\{T_i, W_i\}$ coordinates on the left and SE envelopes on the right.

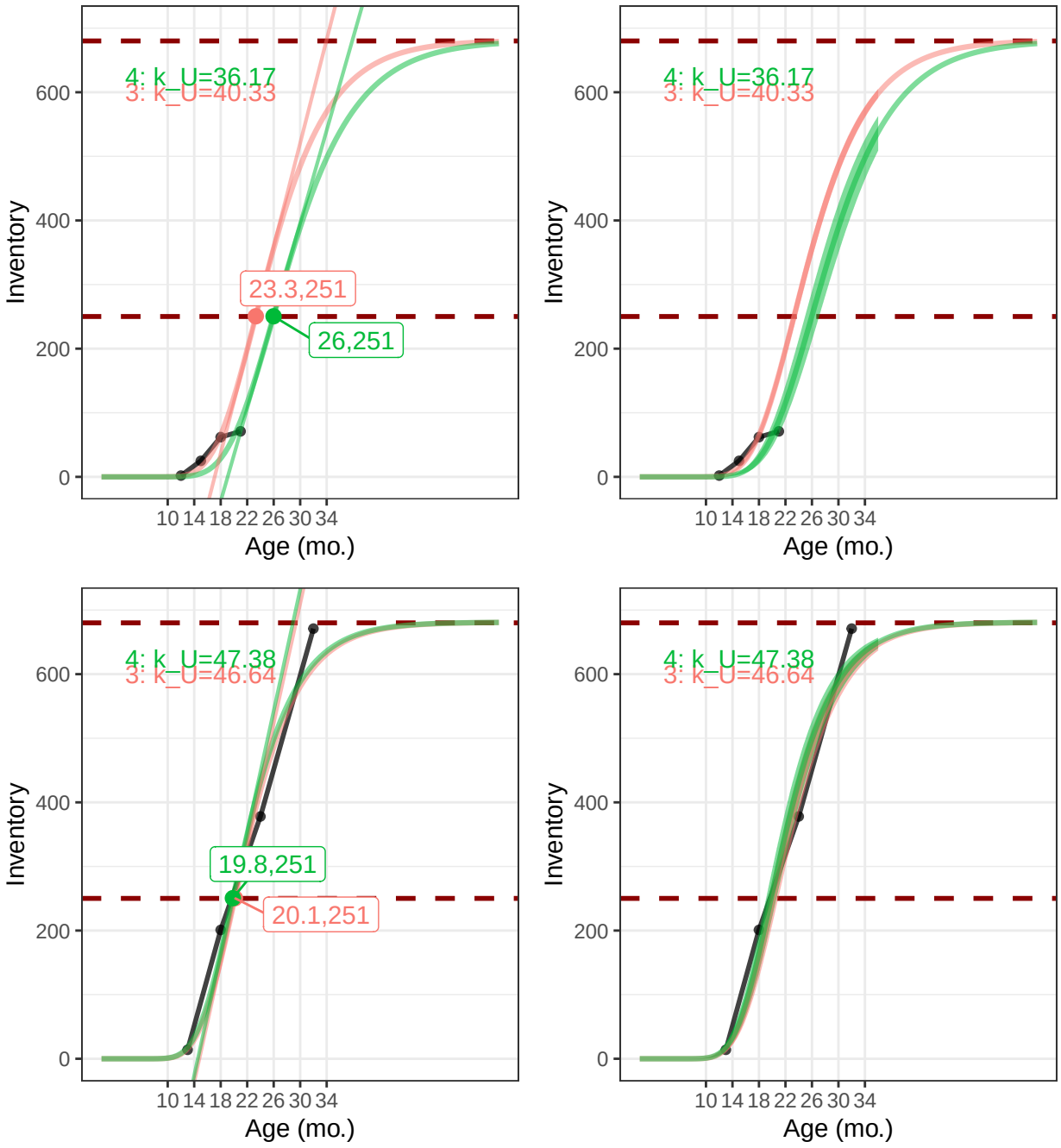

4: k_U=49.11
3: k_U=48.47
19.4,251
19.1,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=49.11
3: k_U=48.47
Inventory
Age (mo.)
4: k_U=41.16
3: k_U=41.43
22.7,251
22.8,251
Inventory
Age (mo.)
4: k_U=41.16
3: k_U=41.43
Inventory
Age (mo.)

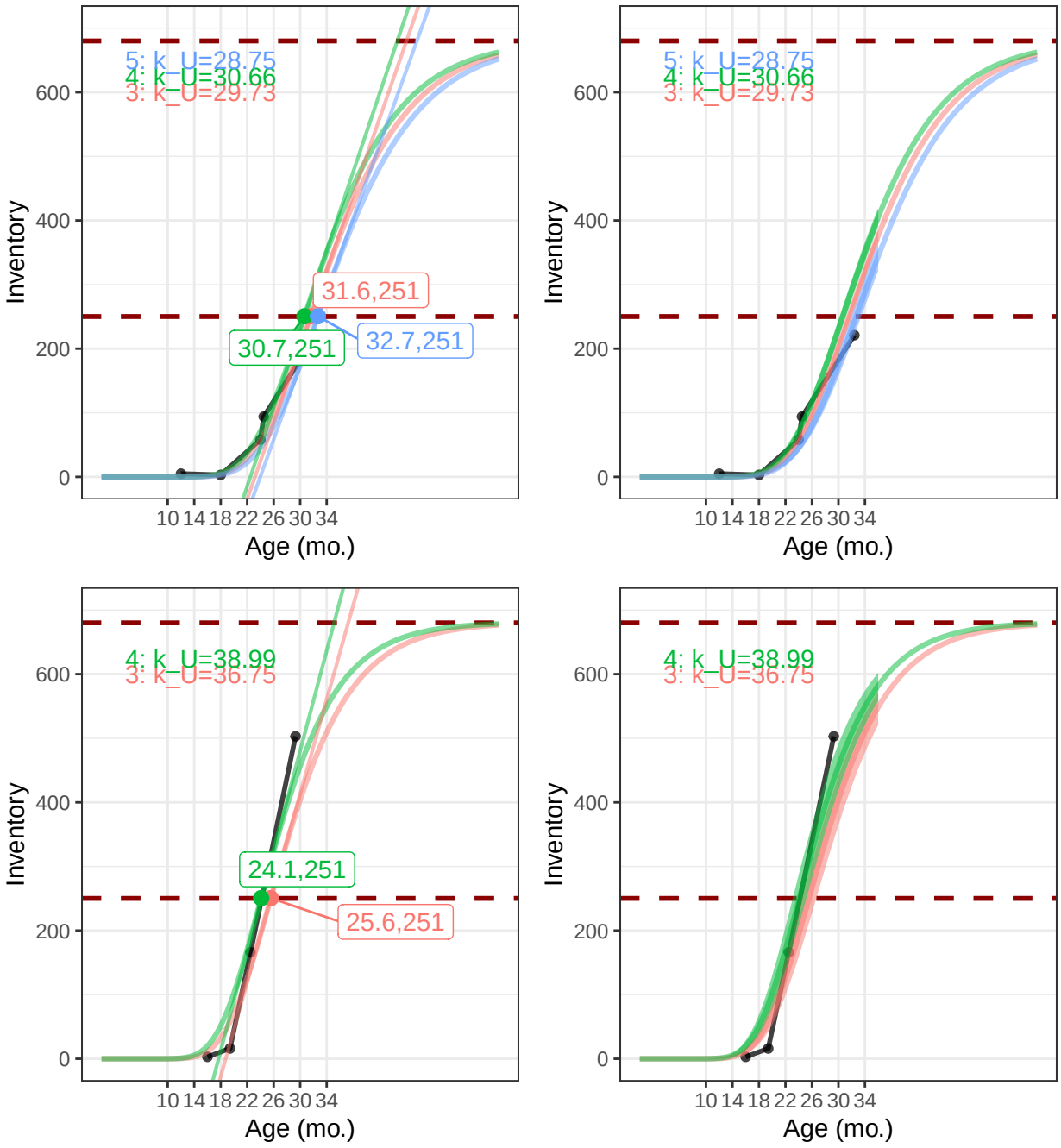

4: k_U=33.46
3: k_U=33.98
28.1,251
27.7,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=38.58
3: k_U=36.88
25.5,251
24.4,251

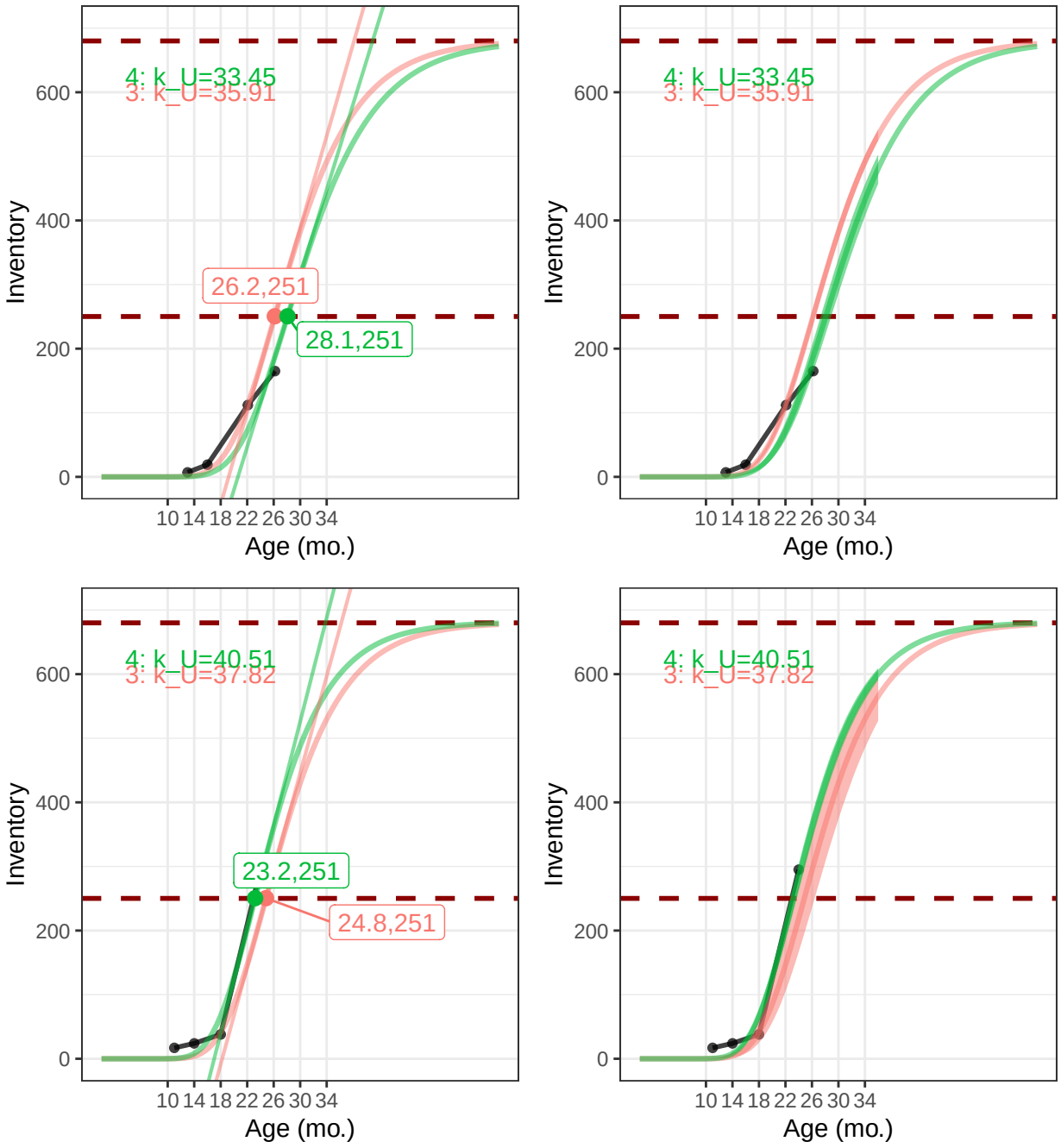

5: k_U=42.11
4: k_U=40.96
3: k_U=39.19
22.9,251
22.3,251
24,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=32.93
3: k_U=32.36
28.5,251
29,251

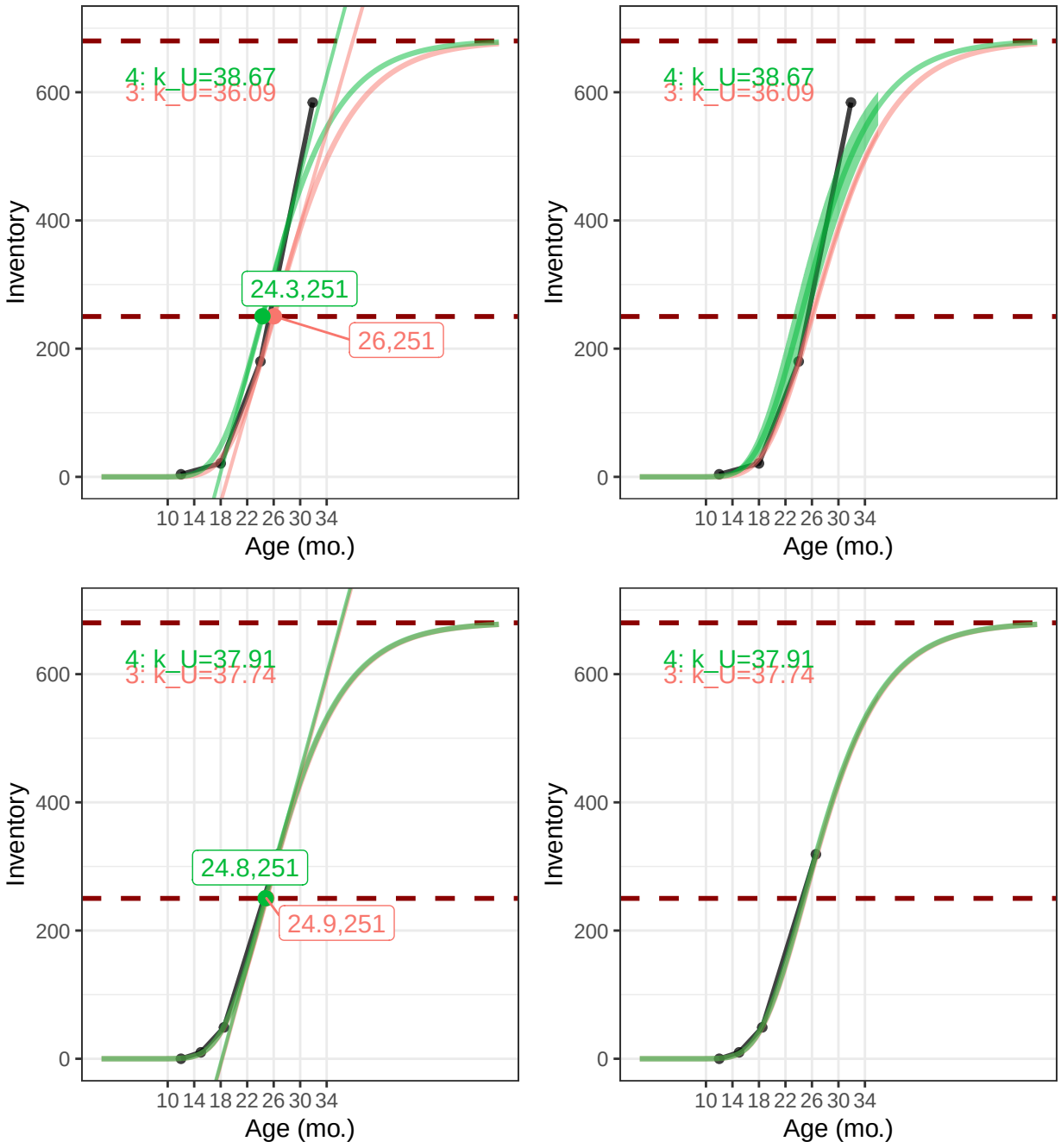

4: k_U=42.24
3: k_U=41.59
22.2,251
22.6,251
Inventory
Age (mo.)
4: k_U=32.64
3: k_U=34.68
27.1,251
28.8,251

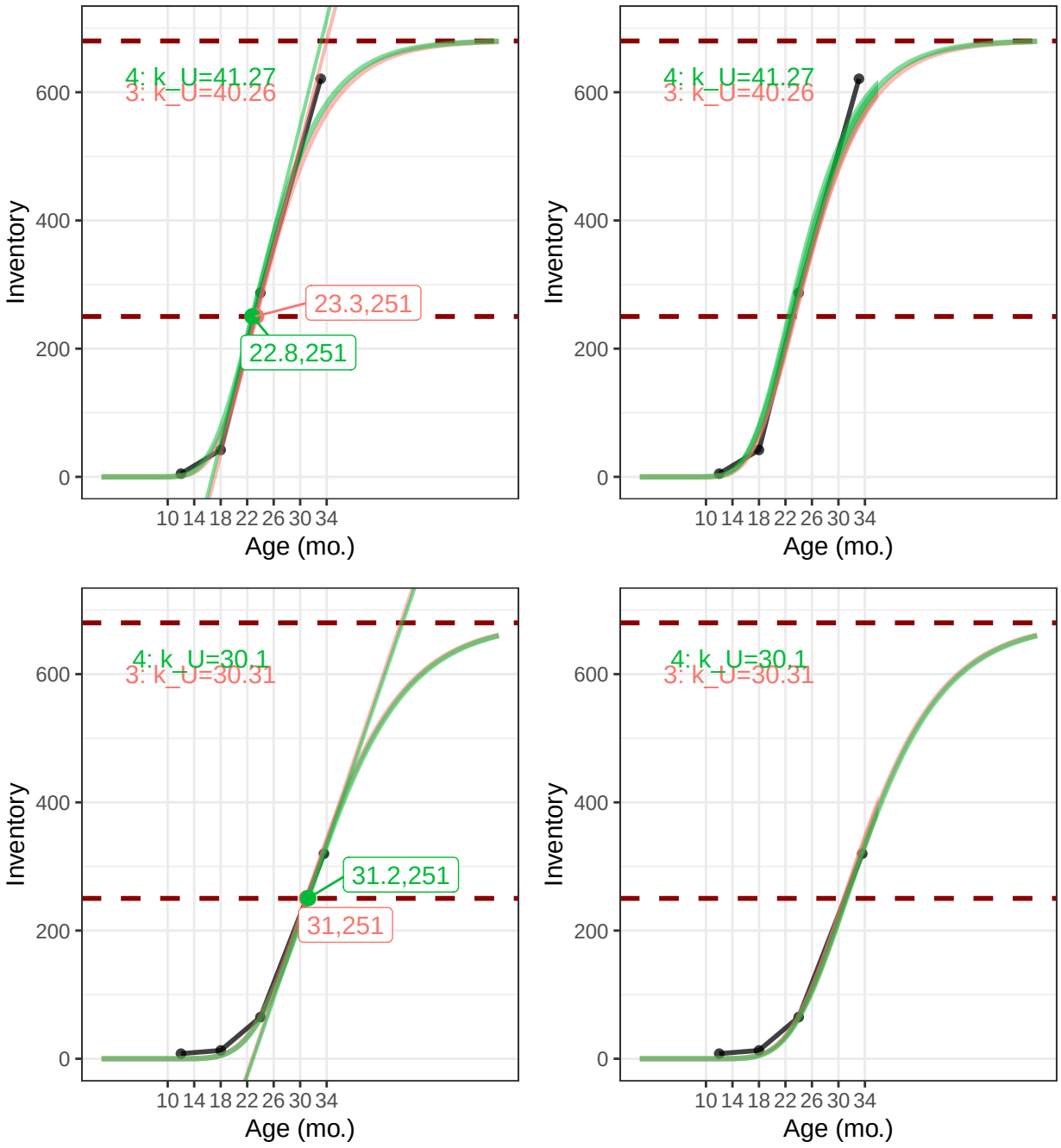

5: k_U=32.92
4: k_U=33.88
3: k_U=32.47
27.7,251
28.9,251
28.5,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
5: k_U=32.92
4: k_U=33.88
3: k_U=32.47
Inventory
Age (mo.)
4: k_U=38.02
3: k_U=38.02
24.7,251
24.7,251
Inventory
Age (mo.)
4: k_U=38.02
3: k_U=38.02
Inventory
Age (mo.)

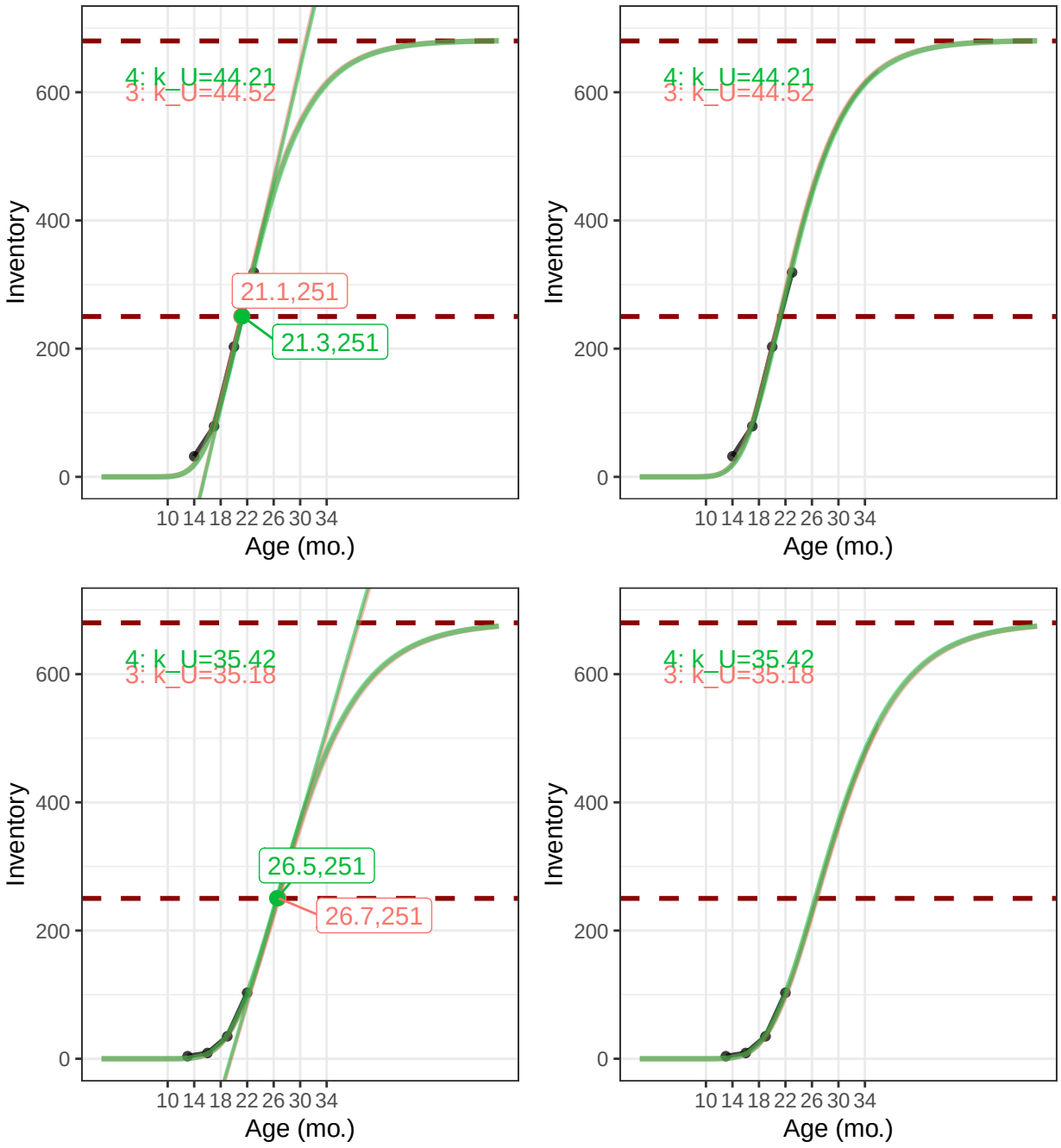

4: k_U=36.17
3: k_U=40.33
23.3,251
26,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=36.17
3: k_U=40.33
Inventory
Age (mo.)
4: k_U=47.38
3: k_U=46.64
19.8,251
20.1,251
Inventory
Age (mo.)
4: k_U=47.38
3: k_U=46.64
Inventory
Age (mo.)

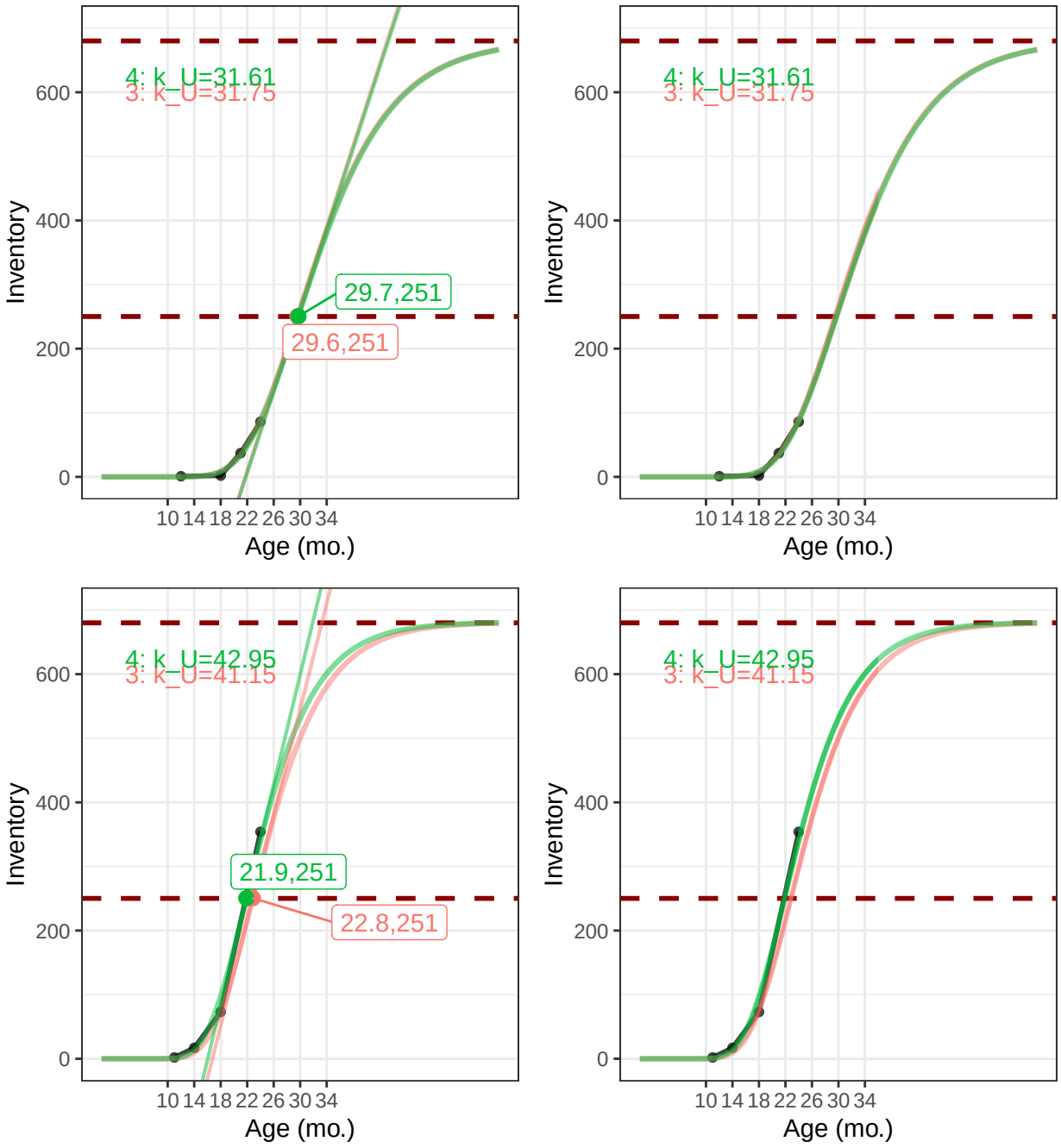

5: k_U=28.75
4: k_U=30.66
3: k_U=29.73
31.6,251
30.7,251
32.7,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=38.99
3: k_U=36.75
24.1,251
25.6,251

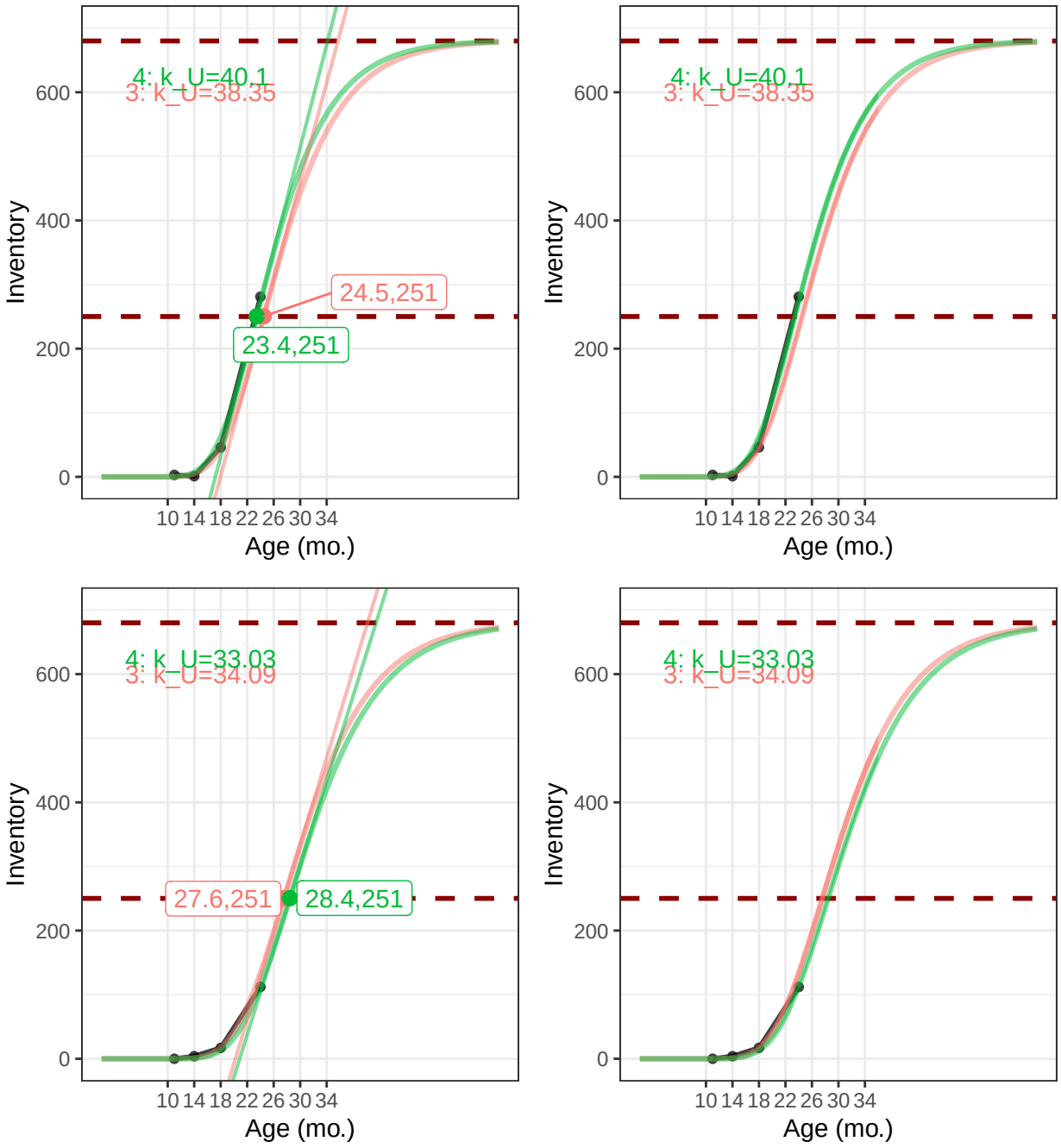

4: k_U=33.45
3: k_U=35.91
26.2,251
28.1,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=33.45
3: k_U=35.91
Inventory
Age (mo.)
4: k_U=40.51
3: k_U=37.82
23.2,251
24.8,251
Inventory
Age (mo.)
4: k_U=40.51
3: k_U=37.82
Inventory
Age (mo.)

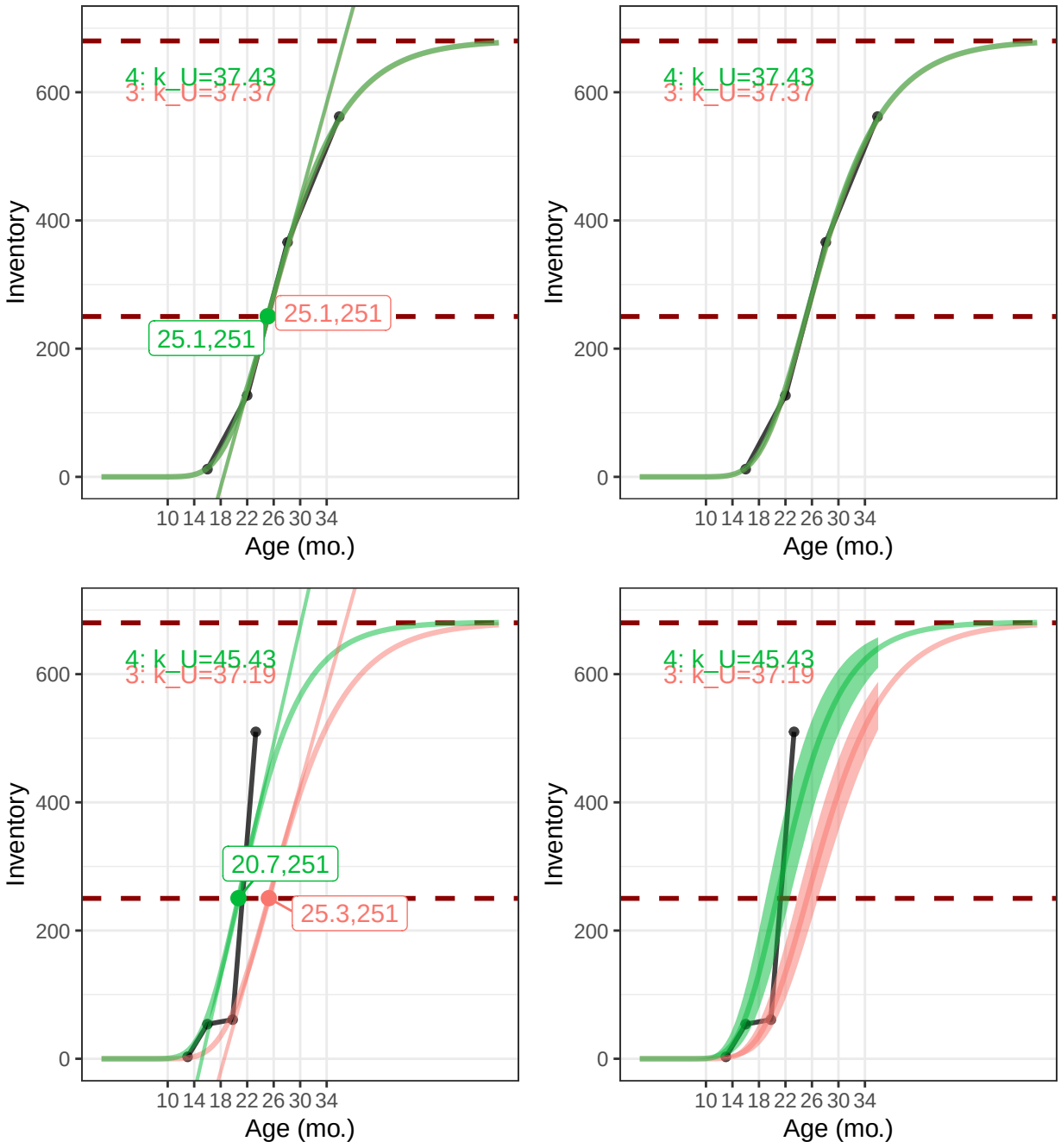

4: k_U=38.67
3: k_U=36.09
24.3,251
26,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=38.67
3: k_U=36.09
Inventory
Age (mo.)
4: k_U=37.91
3: k_U=37.74
24.8,251
24.9,251
Inventory
Age (mo.)
4: k_U=37.91
3: k_U=37.74
Inventory
Age (mo.)

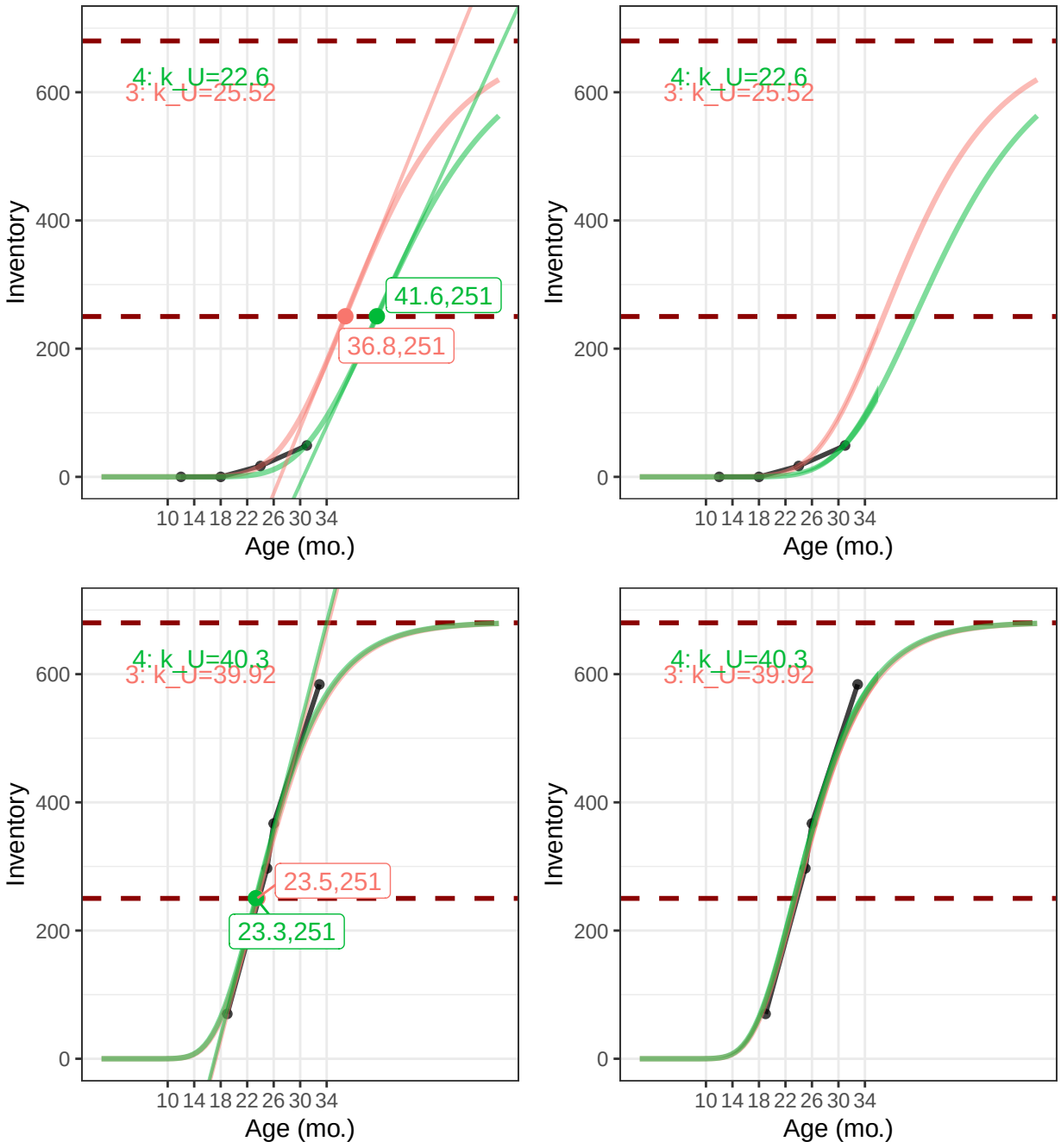

4: k_U=41.27
3: k_U=40.26
23.3,251
22.8,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=41.27
3: k_U=40.26
Inventory
Age (mo.)
4: k_U=30.1
3: k_U=30.31
31.2,251
31,251
Inventory
Age (mo.)
4: k_U=30.1
3: k_U=30.31
Inventory
Age (mo.)

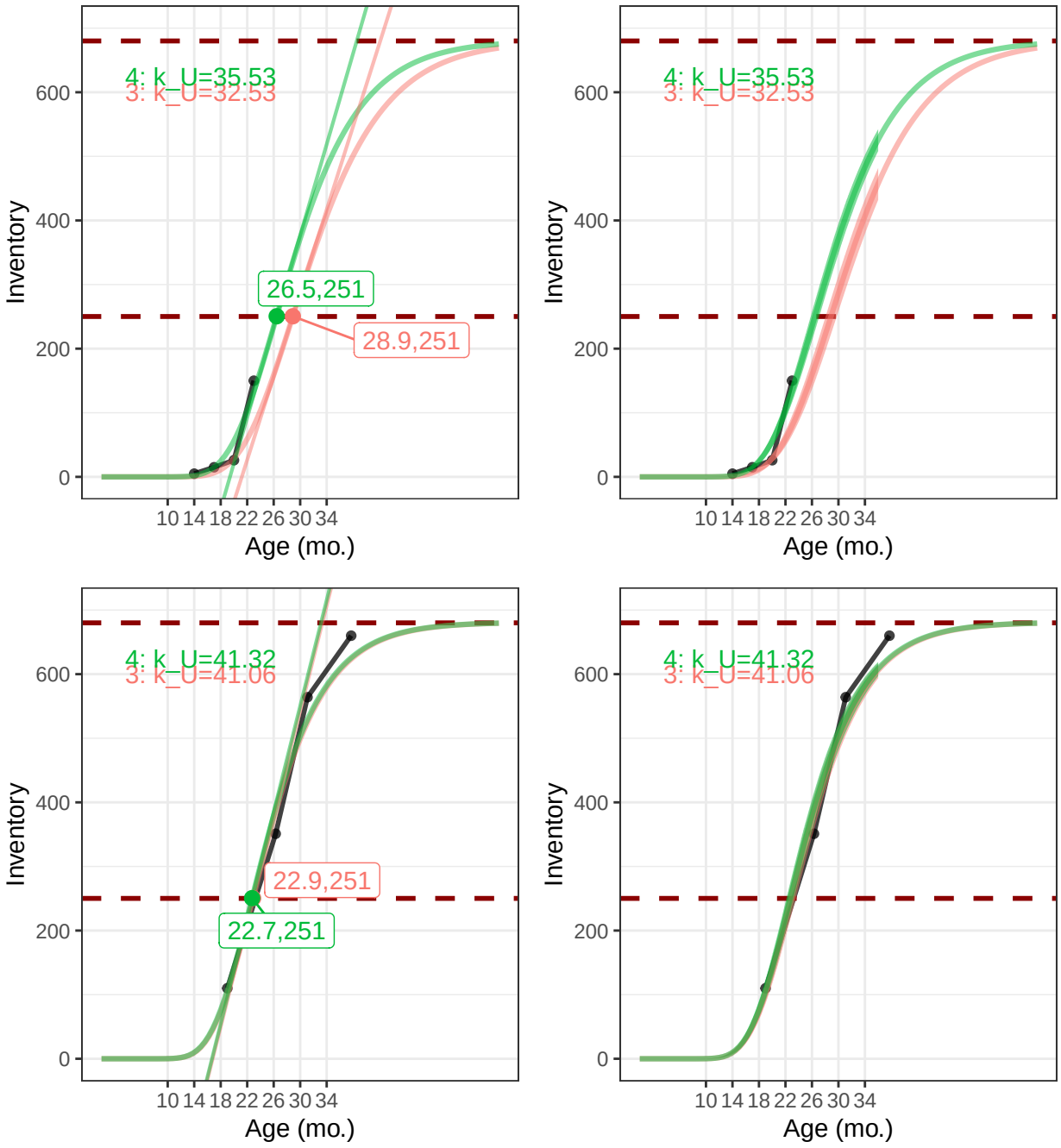

4: k_U=44.21
3: k_U=44.52
21.1,251
21.3,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=44.21
3: k_U=44.52
Inventory
Age (mo.)
4: k_U=35.42
3: k_U=35.18
26.5,251
26.7,251
Inventory
Age (mo.)
4: k_U=35.42
3: k_U=35.18
Inventory
Age (mo.)

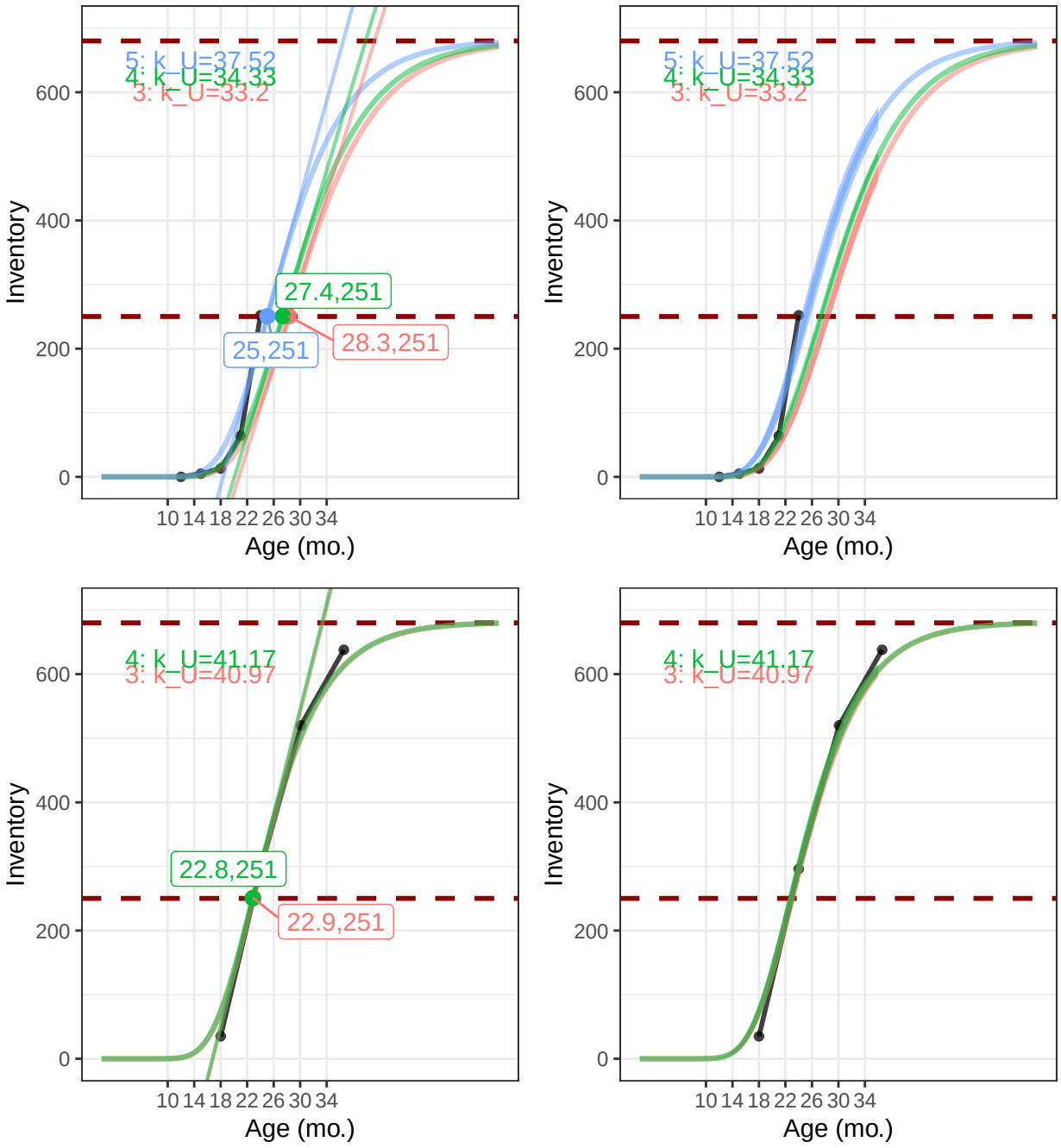

4: k_U=31.61
3: k_U=31.75
29.7,251
29.6,251
Inventory
Age (mo.)
4: k_U=42.95
3: k_U=41.15
21.9,251
22.8,251

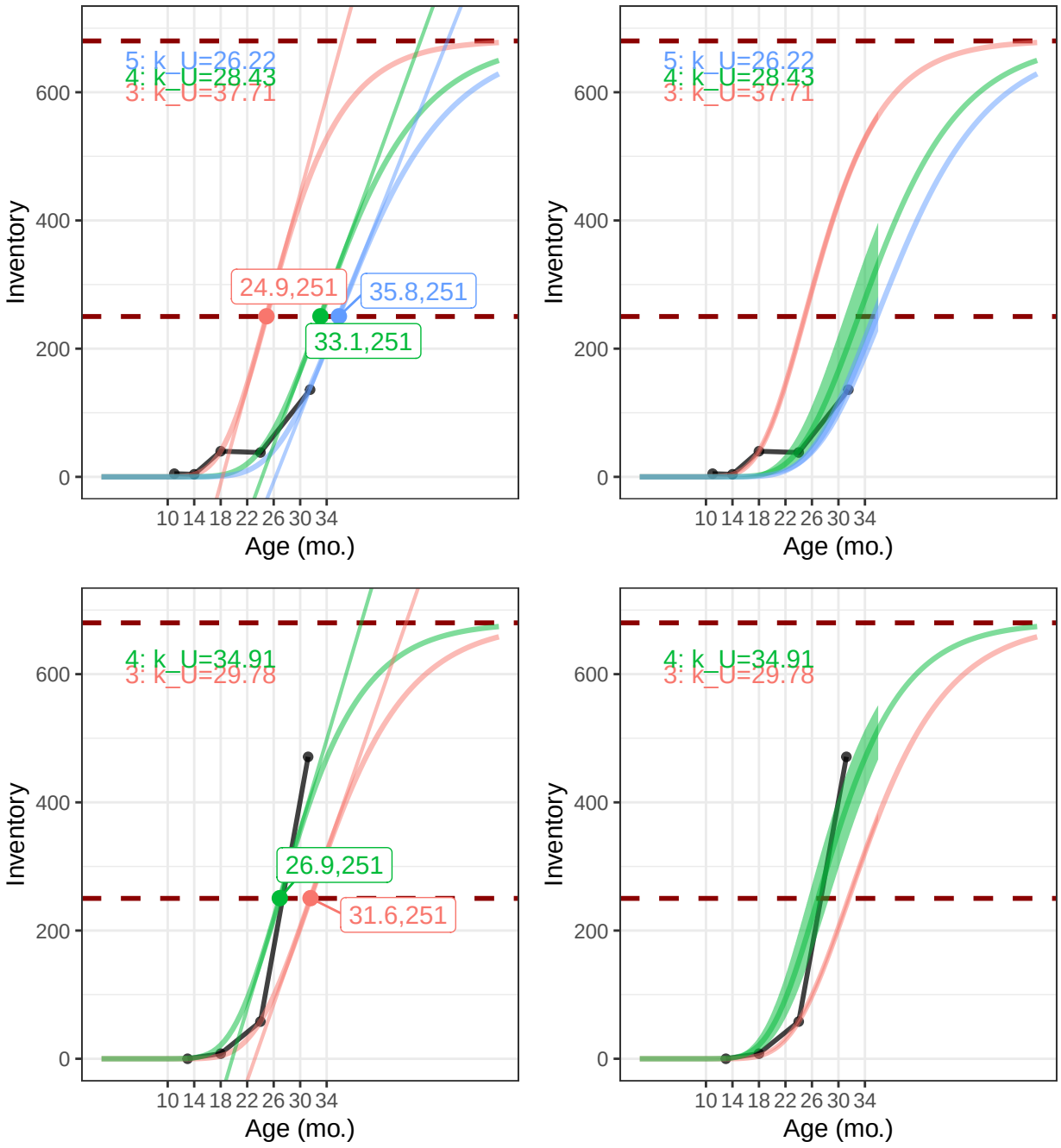

4: k_U=40.1
3: k_U=38.35
24.5,251
23.4,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=40.1
3: k_U=38.35
Inventory
Age (mo.)
4: k_U=33.03
3: k_U=34.09
27.6,251
28.4,251
Inventory
Age (mo.)
4: k_U=33.03
3: k_U=34.09
Inventory
Age (mo.)

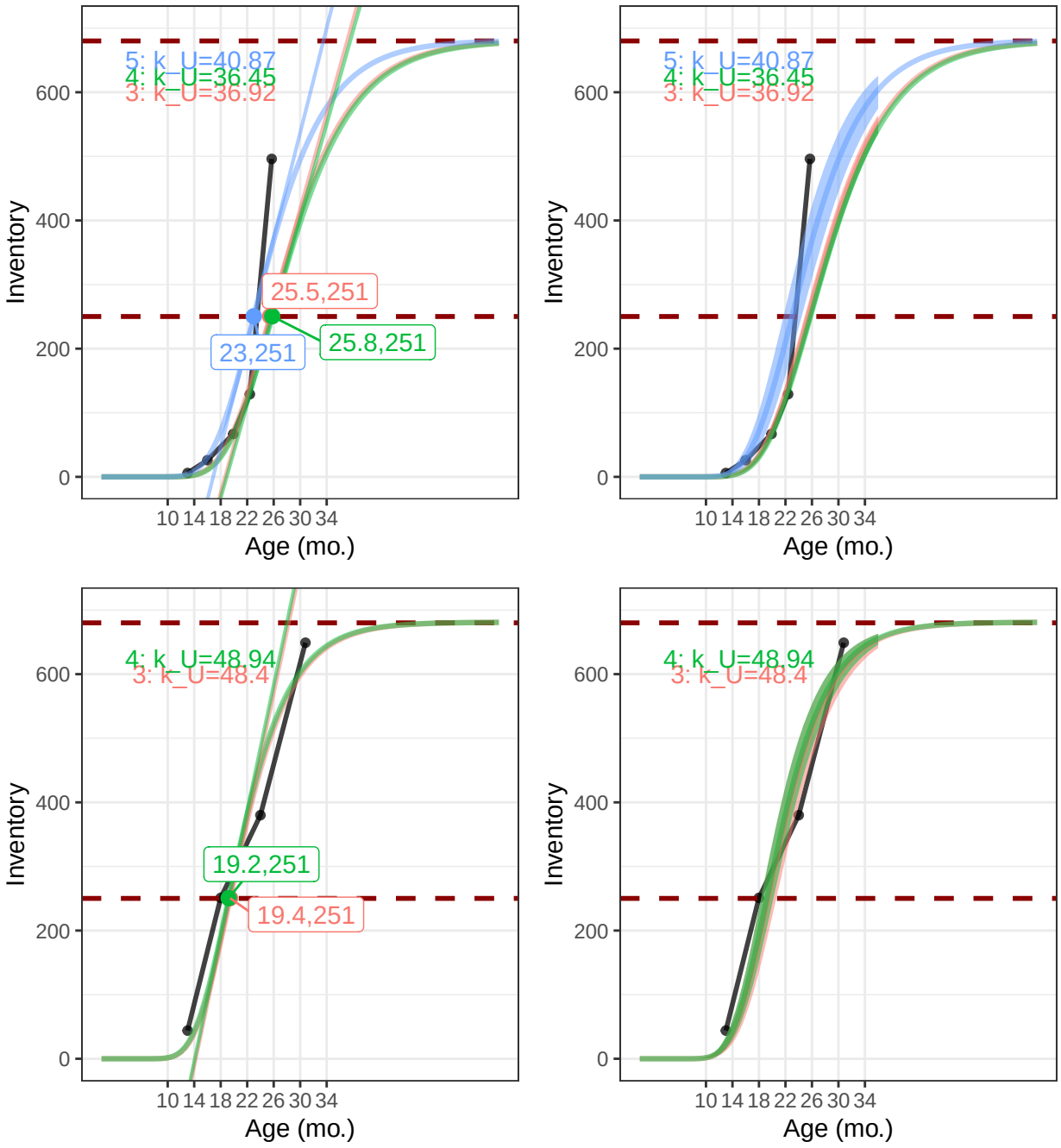

4: k_U=37.43
3: k_U=37.37
25.1,251
25.1,251
4: k_U=37.43
3: k_U=37.37
4: k_U=45.43
3: k_U=37.19
20.7,251
25.3,251
4: k_U=45.43
3: k_U=37.19
Inventory
0
200
400
600
10 14 18 22 26 30 34
Age (mo.)

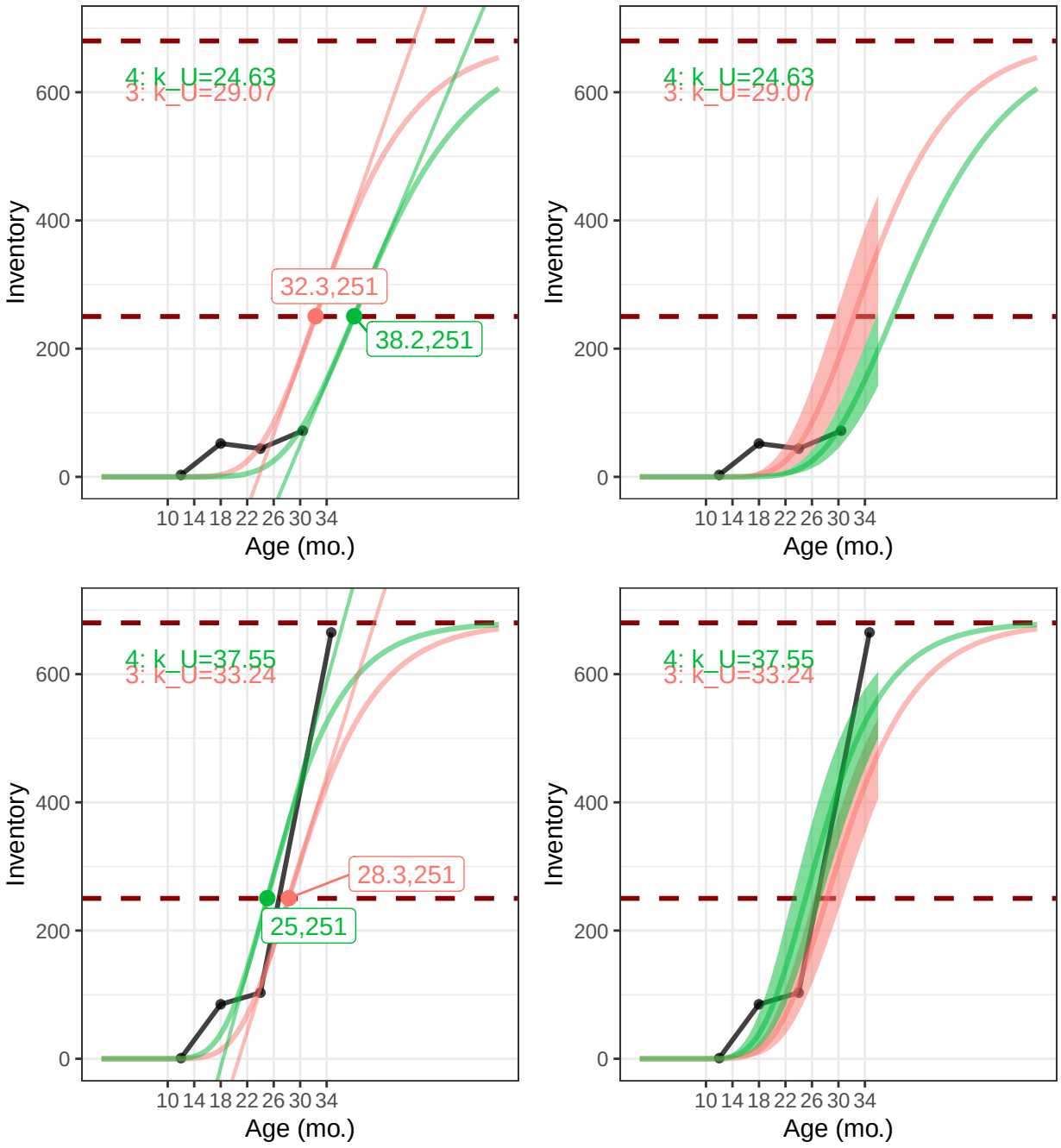

4: k_U=22.6
3: k_U=25.52
41.6,251
36.8,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=40.3
3: k_U=39.92
23.5,251
23.3,251

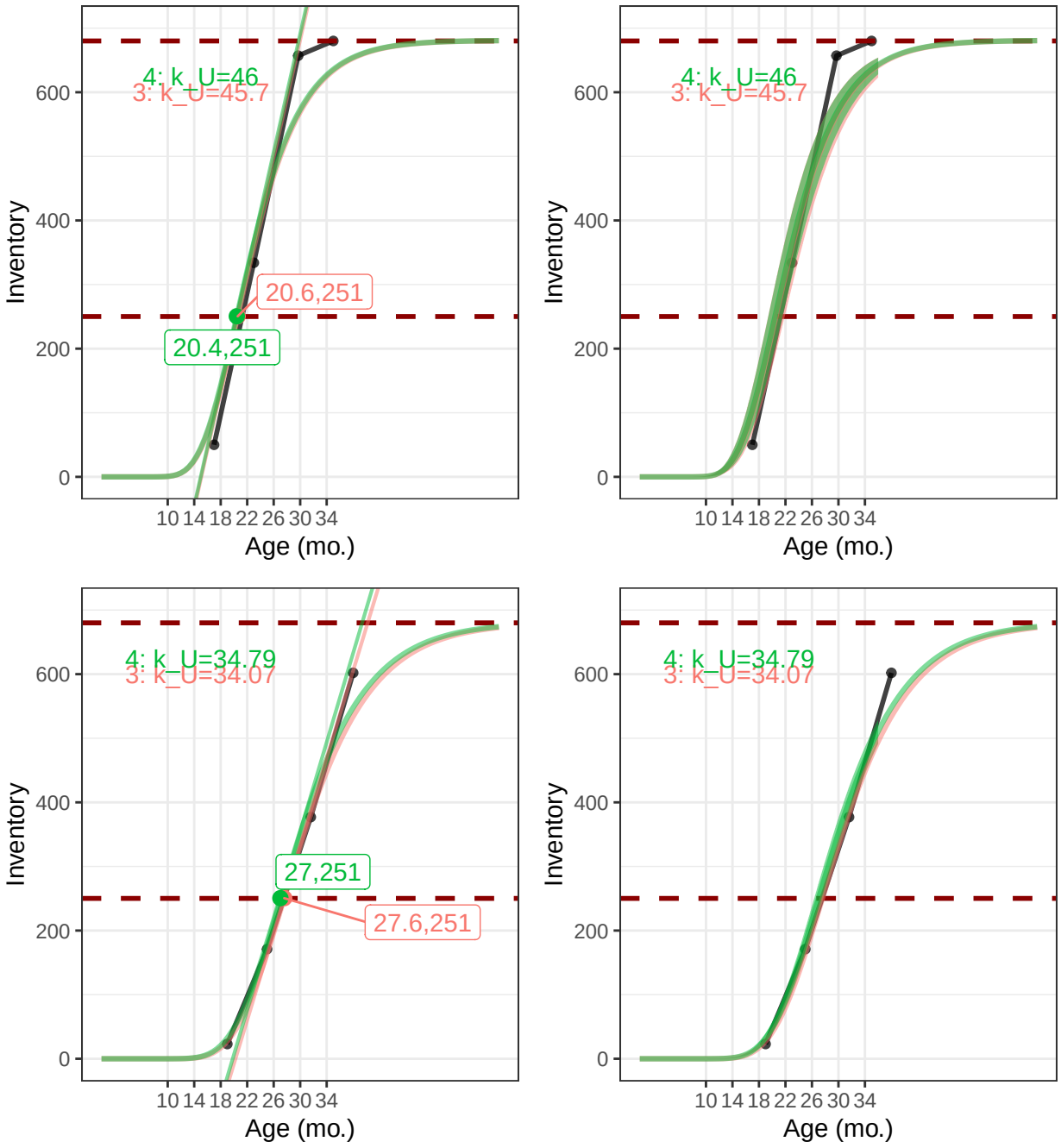

4: k_U=35.53
3: k_U=32.53
26.5,251
28.9,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=41.32
3: k_U=41.06
22.9,251
22.7,251

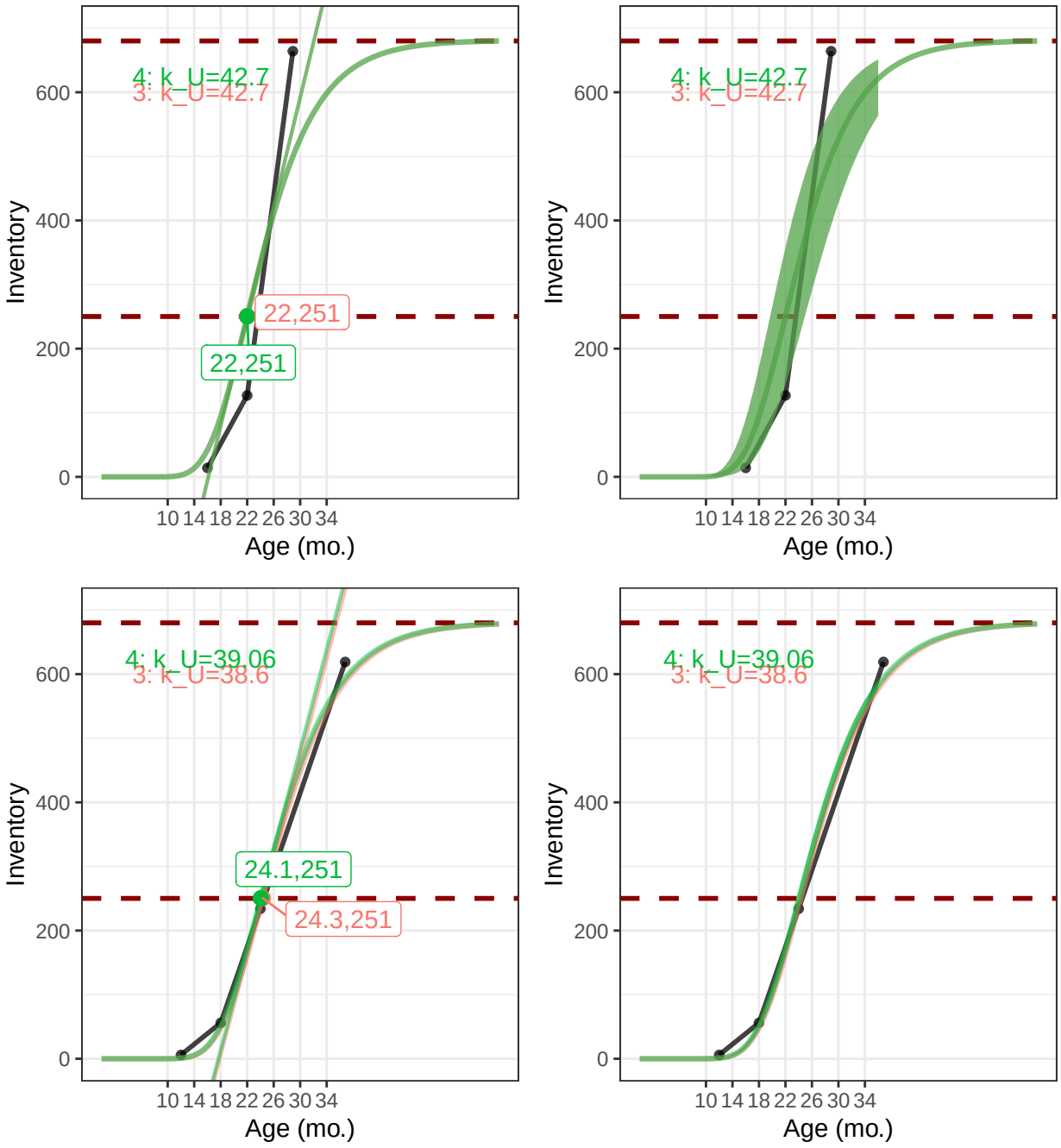

5: k_U=37.52
4: k_U=34.33
3: k_U=33.2
27.4,251
25,251
28.3,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
5: k_U=37.52
4: k_U=34.33
3: k_U=33.2
Inventory
Age (mo.)
4: k_U=41.17
3: k_U=40.97
22.8,251
22.9,251
Inventory
Age (mo.)
4: k_U=41.17
3: k_U=40.97
Inventory
Age (mo.)

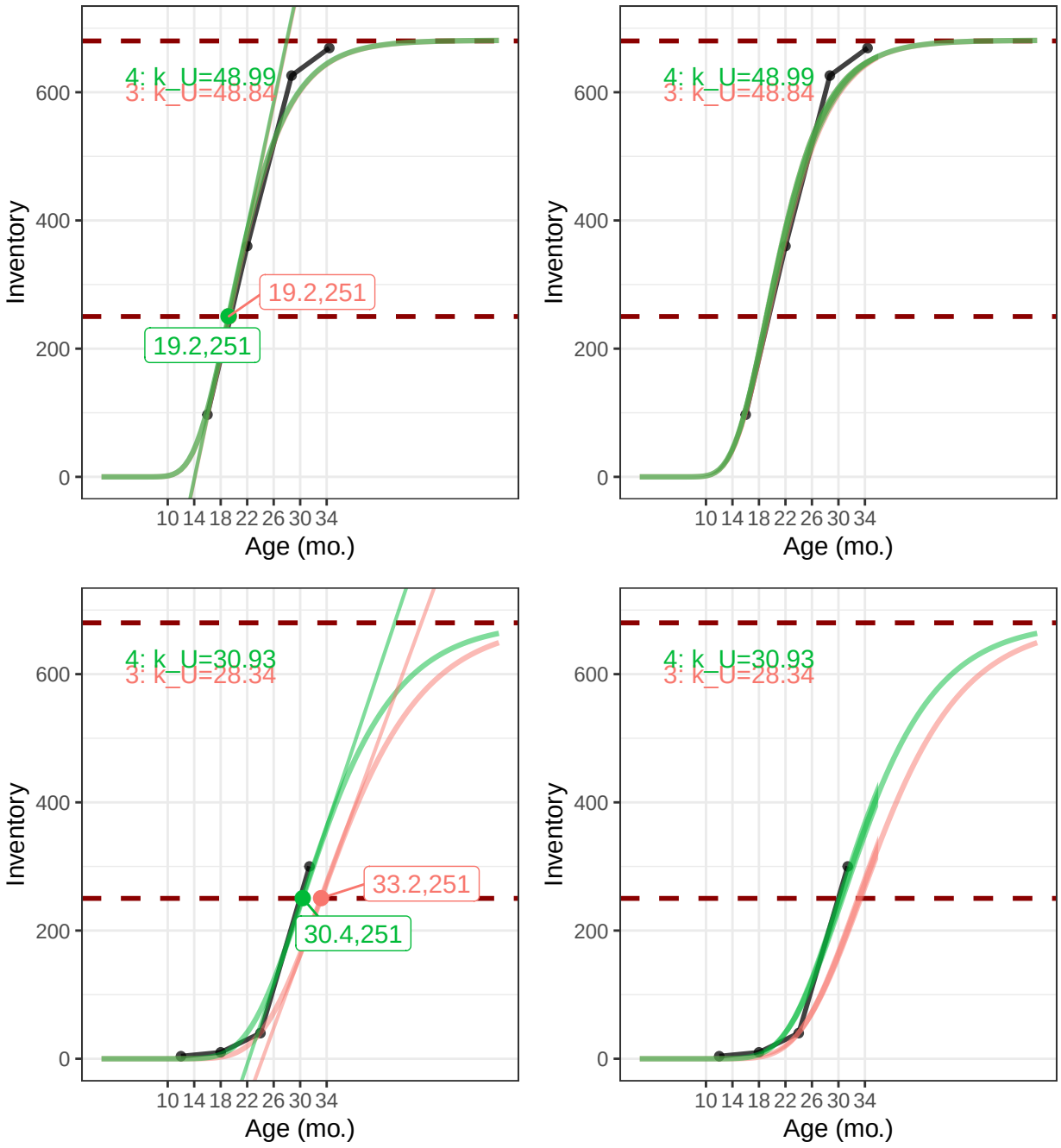

5: k_U=26.22
4: k_U=28.43
3: k_U=37.71
24.9,251
35.8,251
33.1,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=34.91
3: k_U=29.78
26.9,251
31.6,251

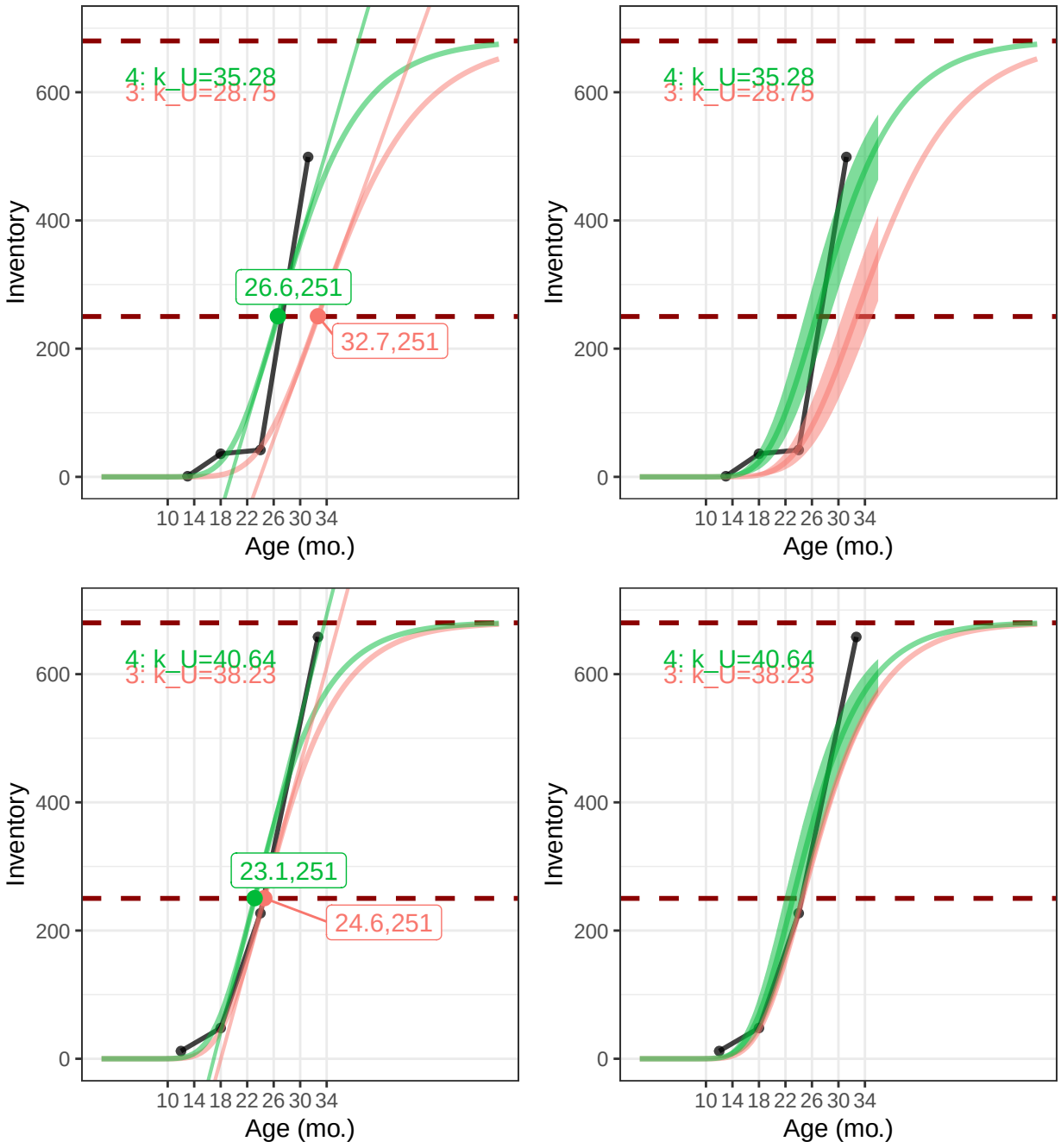

5: k_U=40.87
4: k_U=36.45
3: k_U=36.92
25.5,251
23,251
25.8,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
5: k_U=40.87
4: k_U=36.45
3: k_U=36.92
Inventory
Age (mo.)
4: k_U=48.94
3: k_U=48.4
19.2,251
19.4,251
Inventory
Age (mo.)
4: k_U=48.94
3: k_U=48.4
Inventory
Age (mo.)

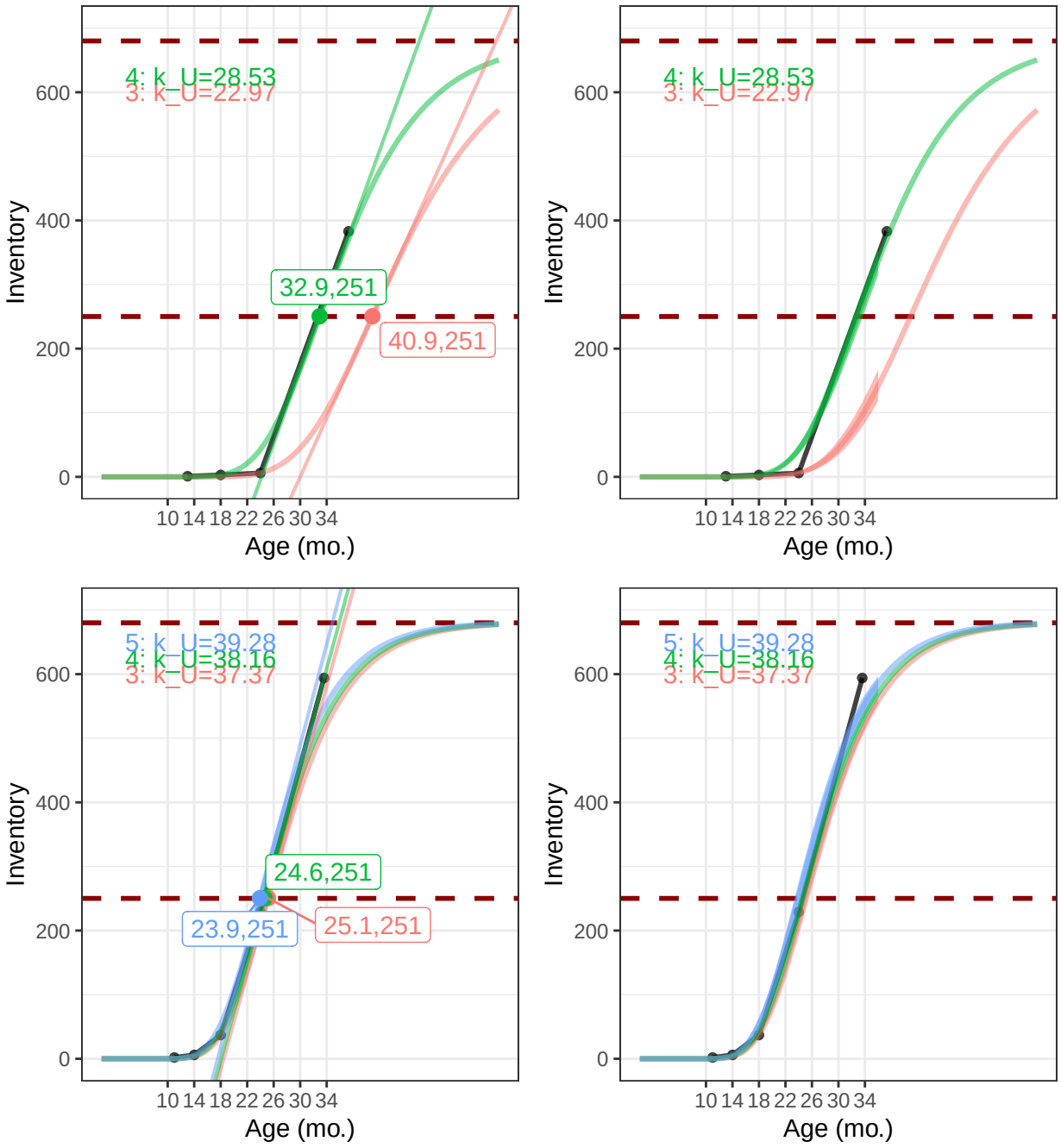

4: k_U=24.63
3: k_U=29.07
32.3,251
38.2,251
Inventory
Age (mo.)
4: k_U=37.55
3: k_U=33.24
28.3,251
25,251

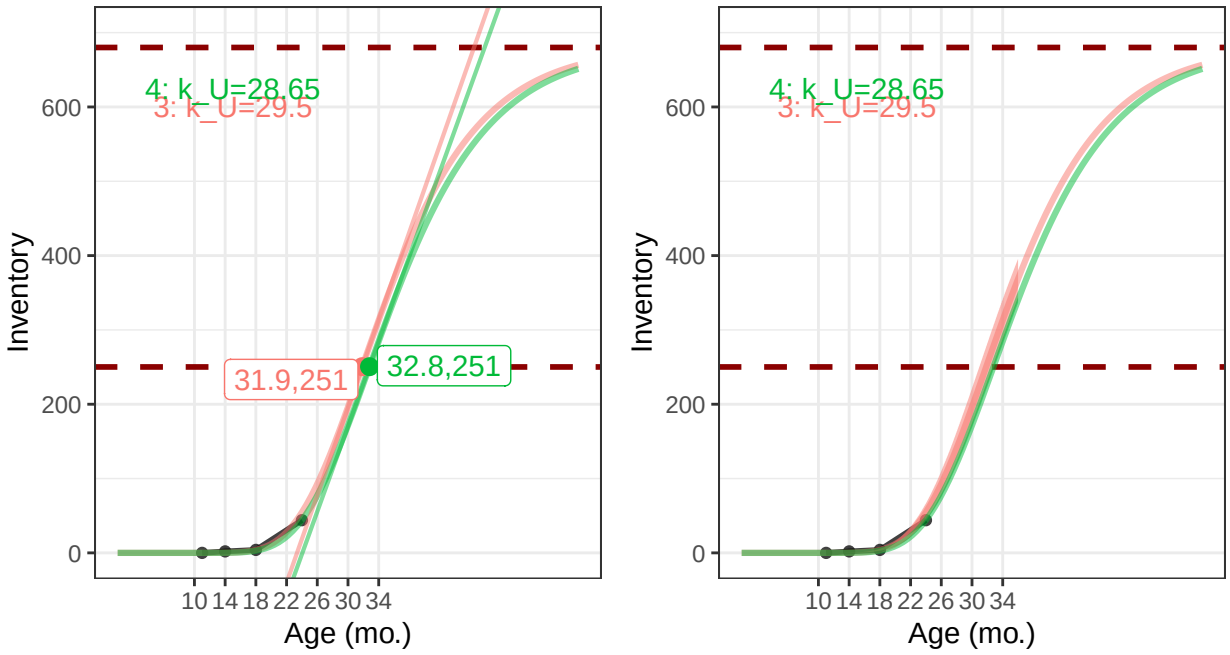

4: k_U=46
3: k_U=45.7
20.6,251
20.4,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=34.79
3: k_U=34.07
27,251
27.6,251


4: k_U=42.7
3: k_U=42.7
22,251
22,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=42.7
3: k_U=42.7
Inventory
Age (mo.)
4: k_U=39.06
3: k_U=38.6
24.1,251
24.3,251
Inventory
Age (mo.)
4: k_U=39.06
3: k_U=38.6
Inventory
Age (mo.)


4: k_U=48.99
3: k_U=48.84
19.2,251
19.2,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=30.93
3: k_U=28.34
33.2,251
30.4,251


4: k_U=35.28
3: k_U=28.75
26.6,251
32.7,251
4: k_U=40.64
3: k_U=38.23
23.1,251
24.6,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34

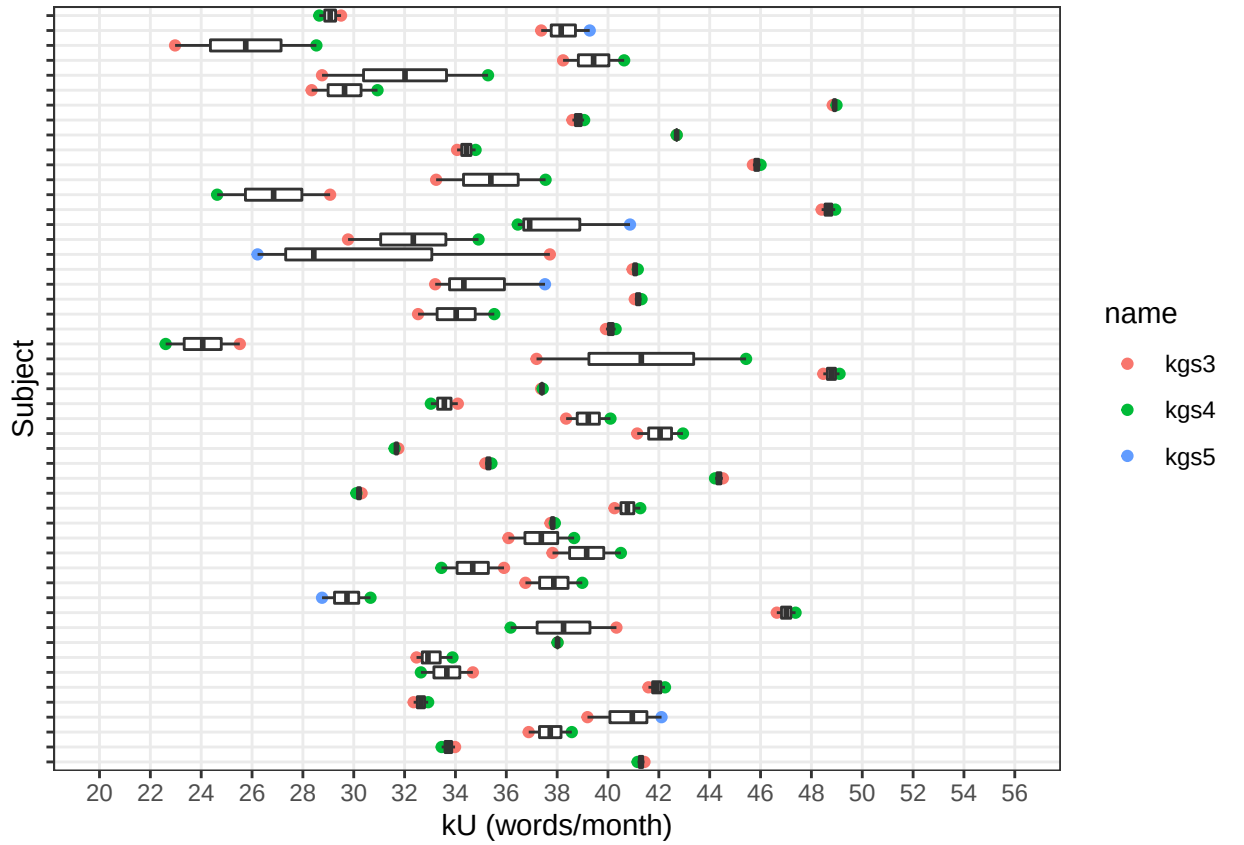

4: k_U=28.53
3: k_U=22.97
32.9,251
40.9,251
Inventory
Age (mo.)
0
200
400
600
10 14 18 22 26 30 34
4: k_U=28.53
3: k_U=22.97
Inventory
Age (mo.)
5: k_U=39.28
4: k_U=38.16
3: k_U=37.37
24.6,251
23.9,251
25.1,251
Inventory
Age (mo.)
5: k_U=39.28
4: k_U=38.16
3: k_U=37.37
Inventory
Age (mo.)



Distribution of k_u

How widely distributed is k_u? In other words, are we getting meaningful variation in k_u, or does everyone develop at the same rate? Boxplots are shown to visualize variation in estimates per-person, per number of points.

```
kgs5 <- sapply(all.fits5, function(x) if(is.list(x)) {summary(x)$coefficients[1, 1]} else {NA})
kgs4 <- sapply(all.fits4, function(x) summary(x)$coefficients[1, 1])
kgs3 <- sapply(all.fits3, function(x) summary(x)$coefficients[1, 1])

all.kgs <- cbind(kgs3, kgs4, kgs5)
all.kUs <- (A * all.kgs / exp(1)) %>%
            as.data.frame() %>%
            rownames_to_column() %>%
            pivot_longer(-rowname) %>%
            na.omit()

# max_vals <- bind_rows(test_all) %>%
#               group_by(data_id) %>%
#               summarize(age = max(age), W = max(inventory)) %>%
#               mutate(data_id = as.character(data_id))

ggplot(all.kUs, aes(x = rowname, y = value)) +
  geom_point(aes(color = name)) +
  geom_boxplot(outlier.shape = NULL) +
  coord_flip() +
  theme_bw() +
  scale_y_continuous(limits = c(20, 56), breaks = seq(20, 56, by = 2),
                     minor_breaks = NULL) +
  scale_x_discrete(labels = NULL) +
  labs(y = "kU (words/month)", x = "Subject")
```



This shows that for the majority of individuals, using 3, 4, or 5 points provide reasonable estimates, however, there are some individuals whose spread is large.

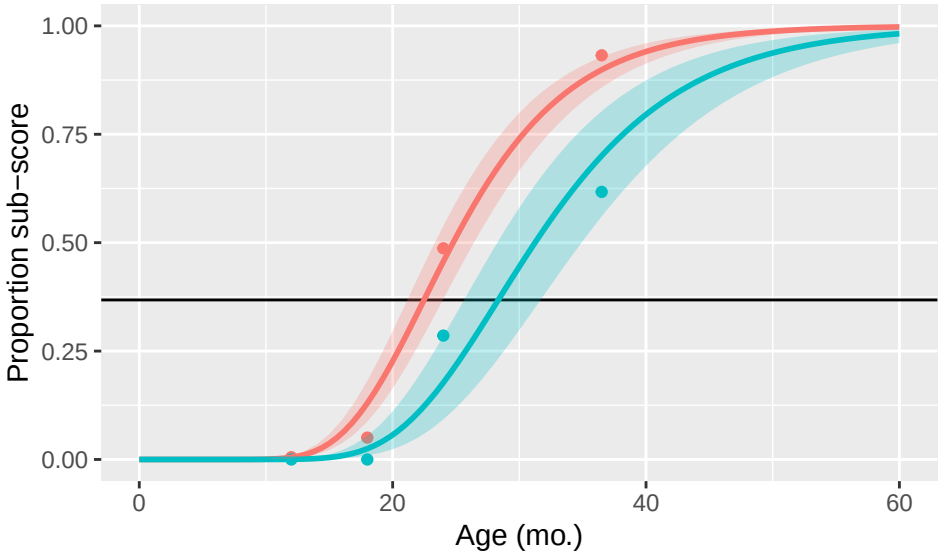
These individuals don't seem to have a systematically low oldest visit or low largest inventory size.

However, it appears, as a rule of thumb, at least the final value should exceed ≈ 100, or for simplicity's sake, $0.5 * W_i = 125$.

Lexical vs syntactic

Are the lexical vs. syntactic scores discriminable? (Subject selected at random from those with five visits.) Here, the curve is fit between 0 and 1, as a proportion of items endorsed. This makes $W_i \approx 0.37$.

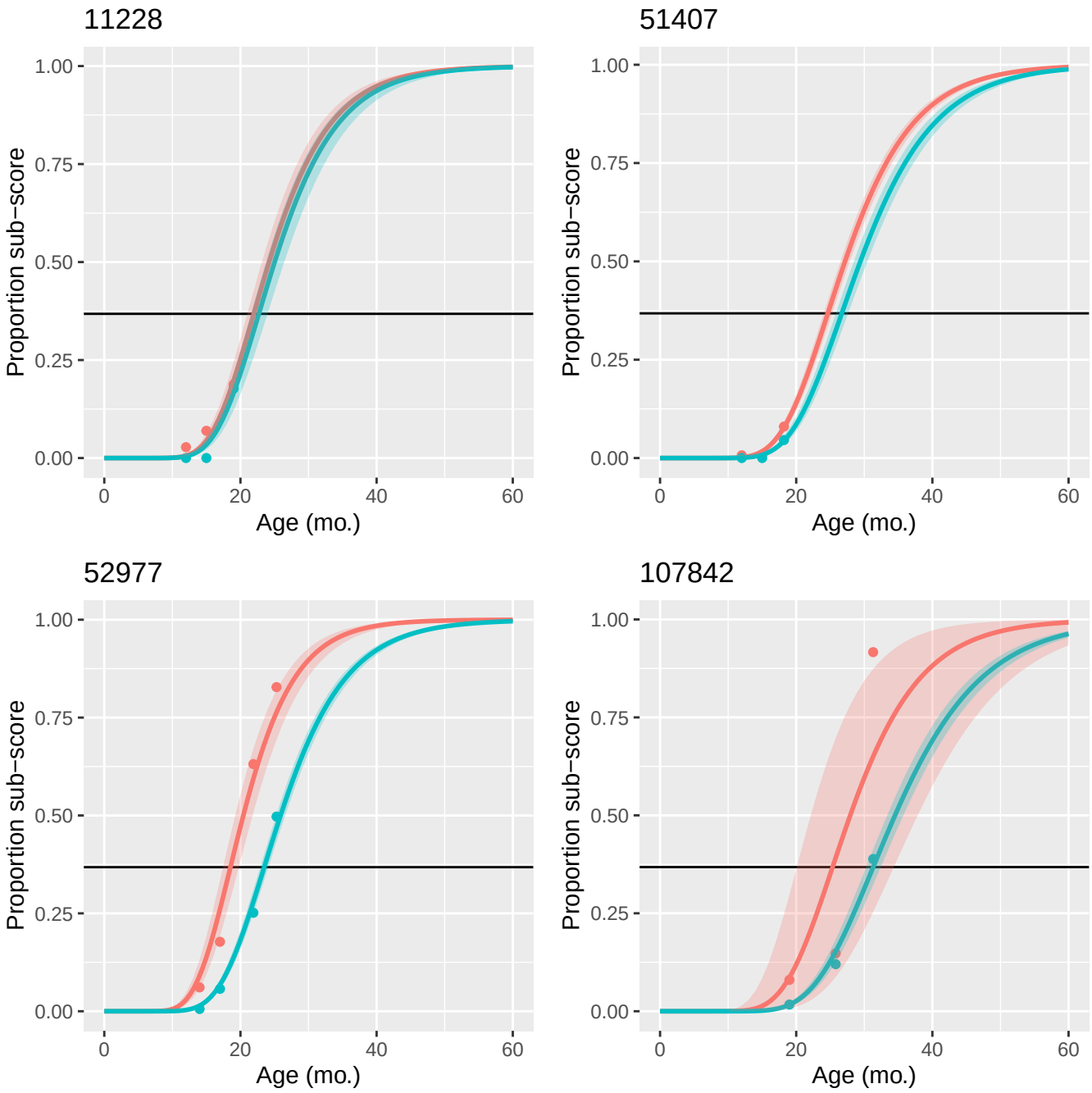
This plot uses the 95% confidence interval rather than the SE.

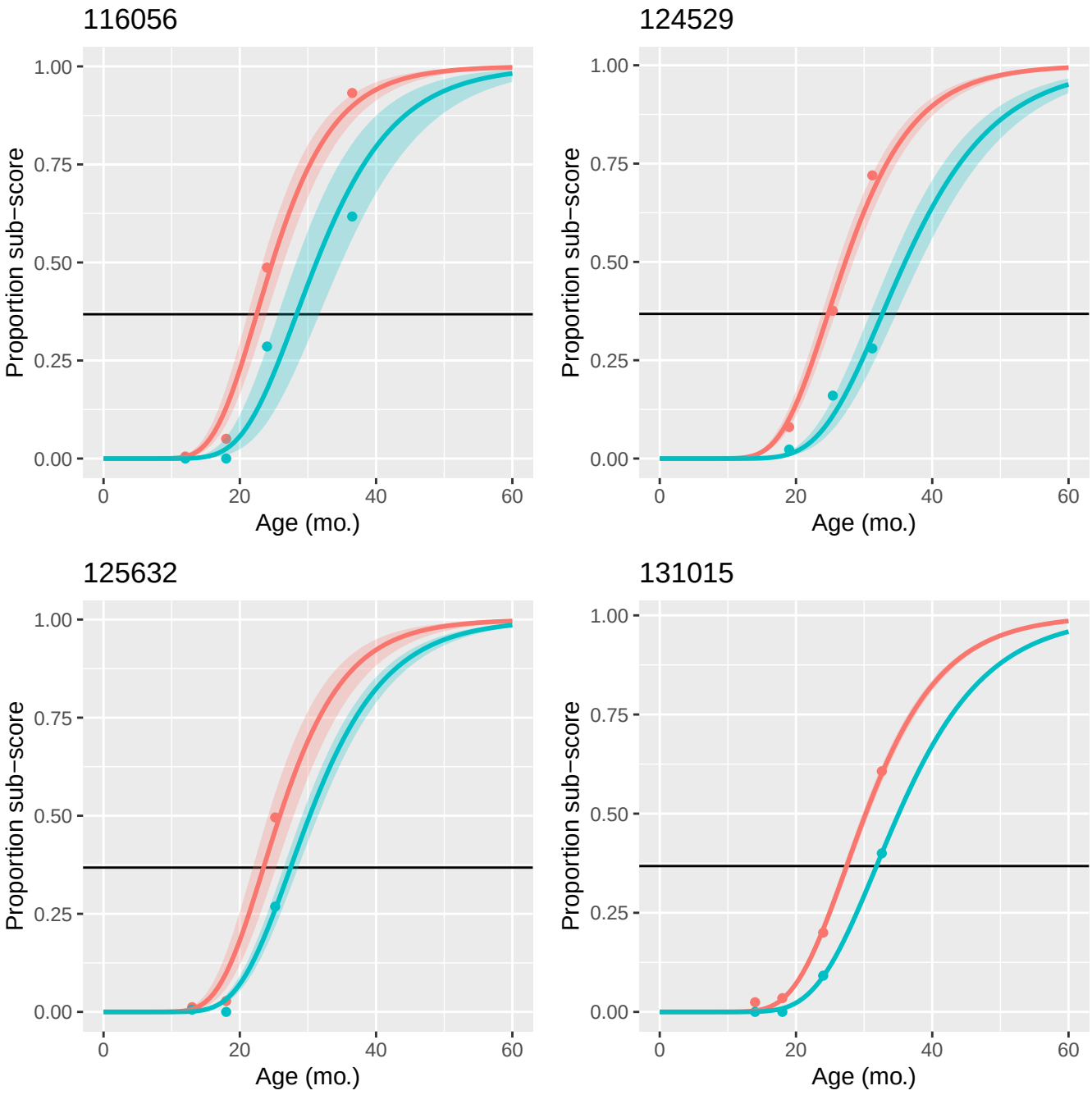


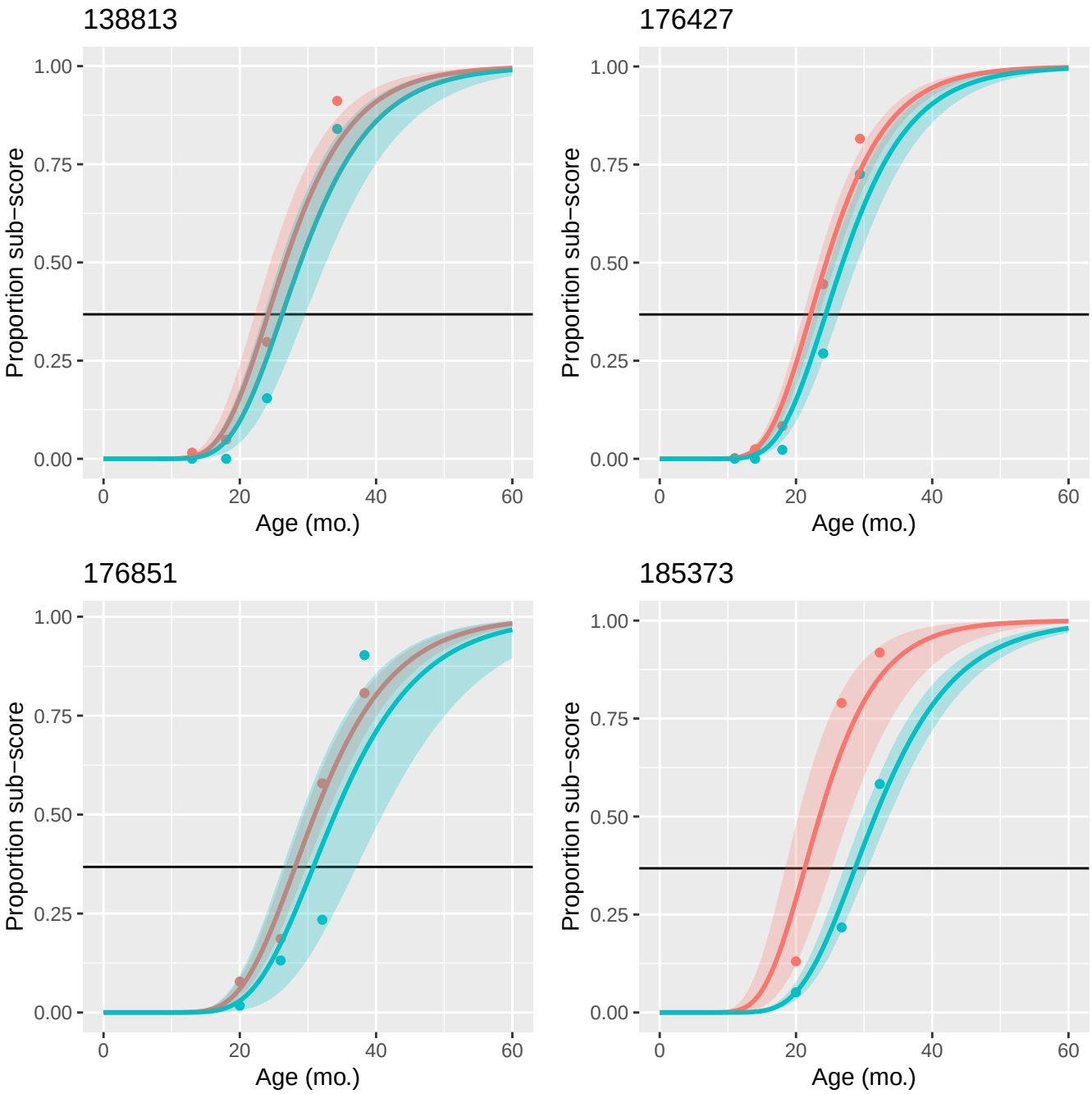

```
## pdf
##   2
```

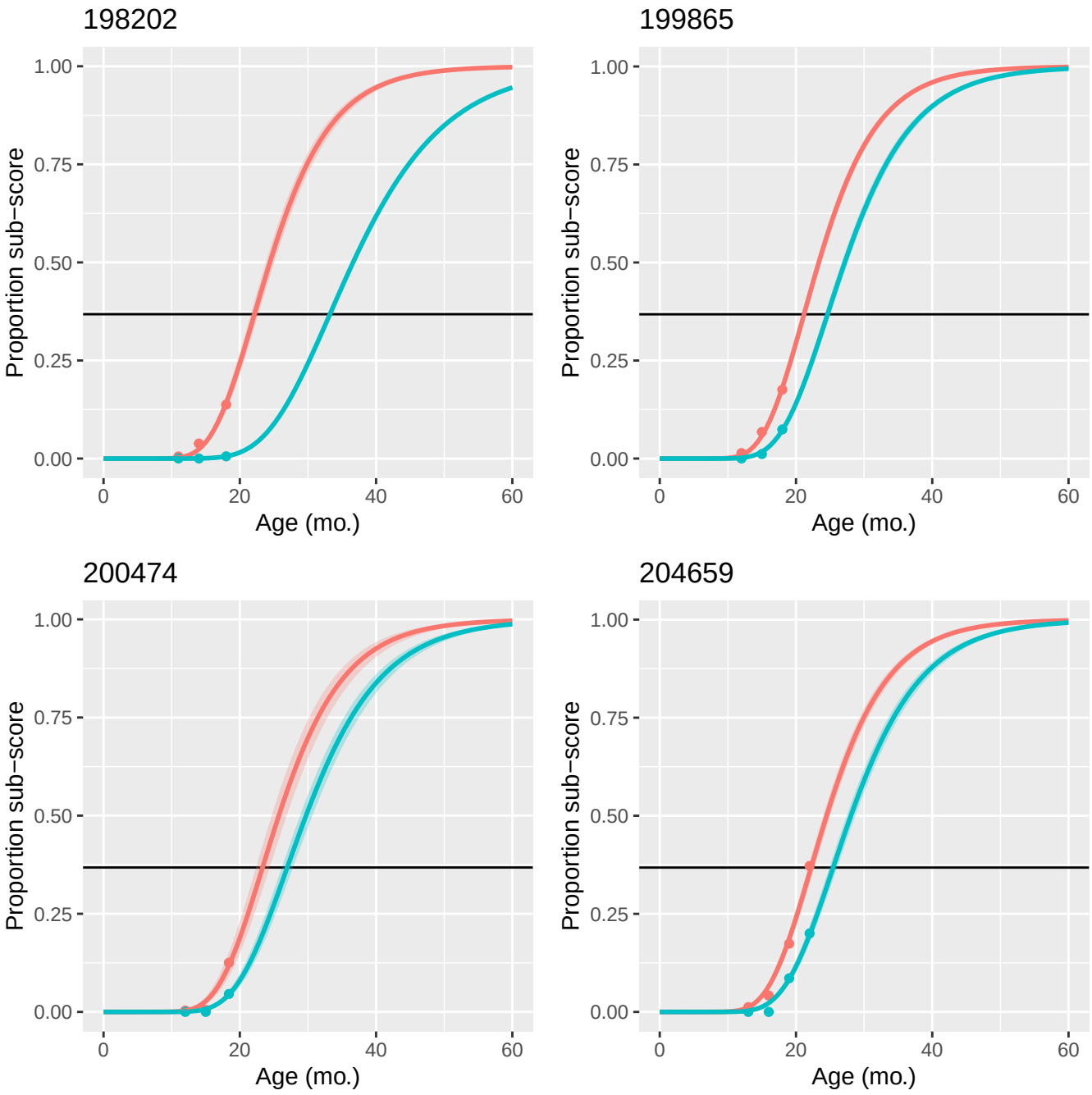
The take-away here is that syntactic development occurs later, and even though the SE is larger, the ranges do *not* overlap. It is the difference between T_i, lex and T_i, syn that is interesting.

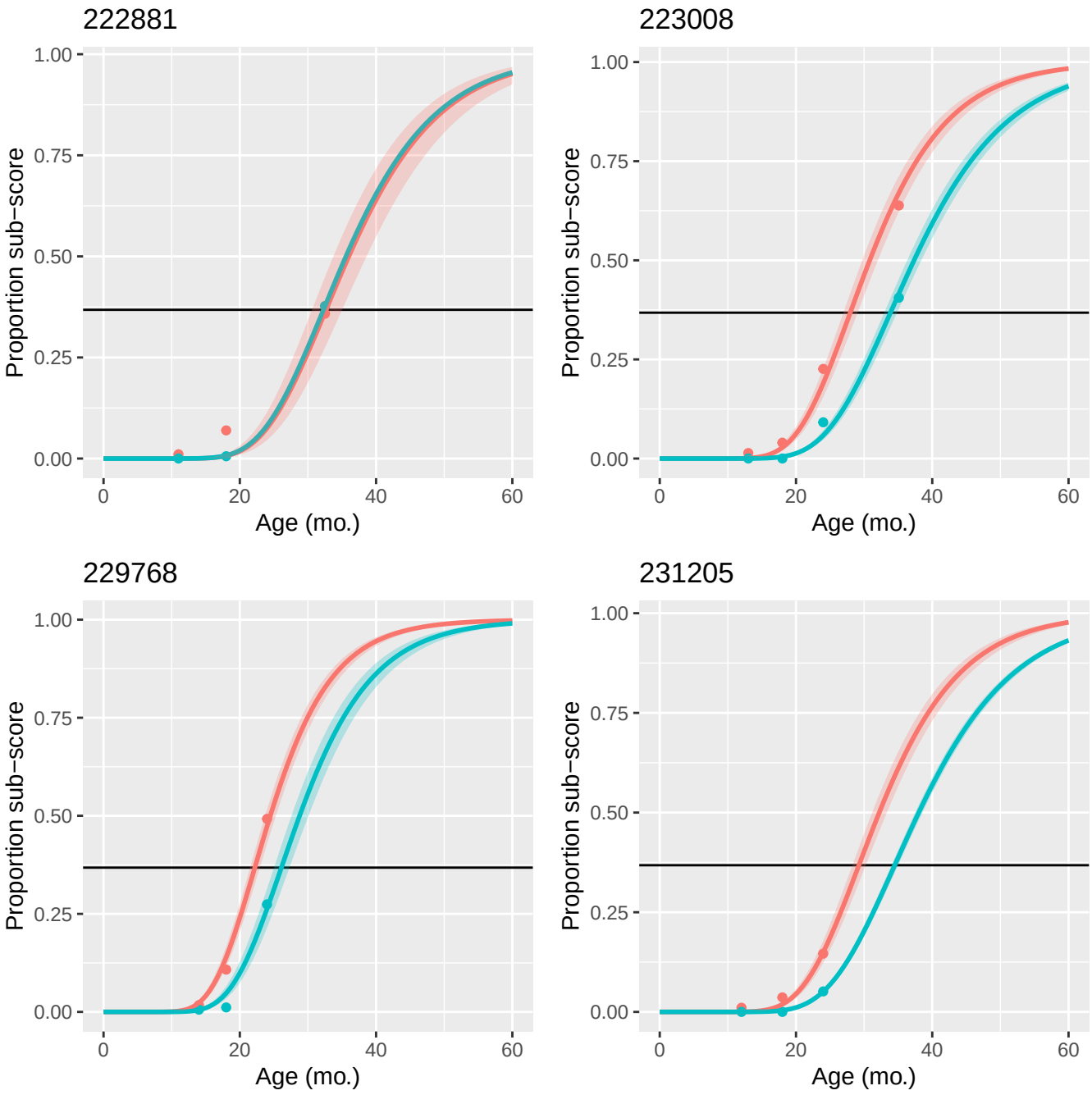
L v S across subjects

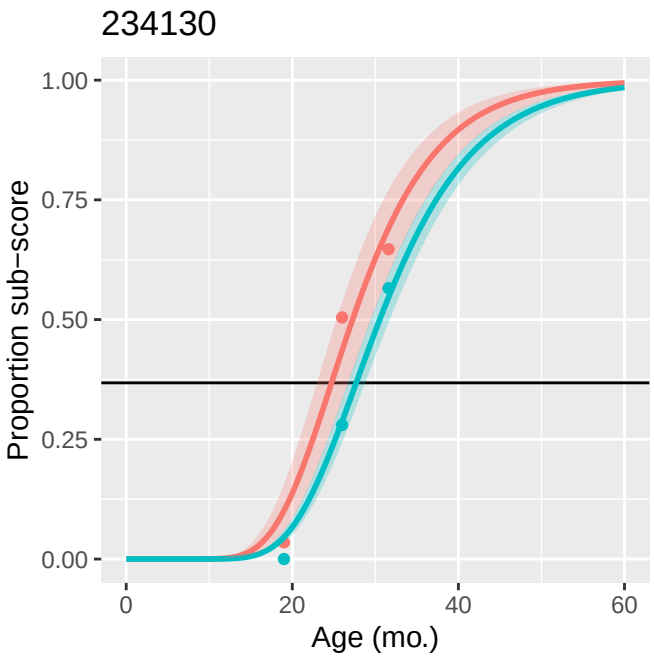



116056
124529
125632
131015
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

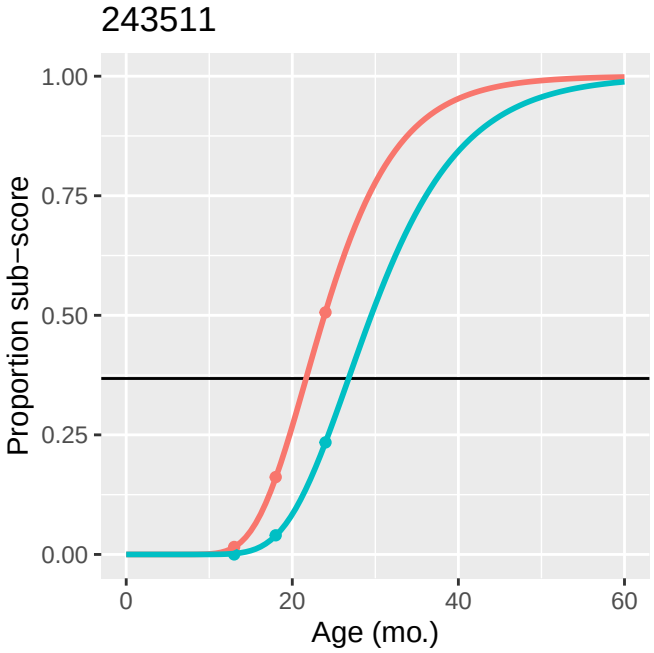

138813
176427
176851
185373
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

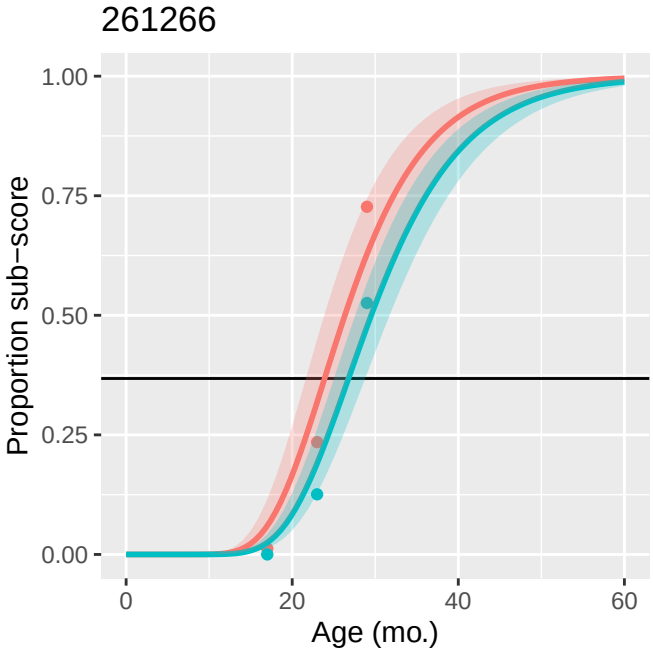

198202
199865
200474
204659
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

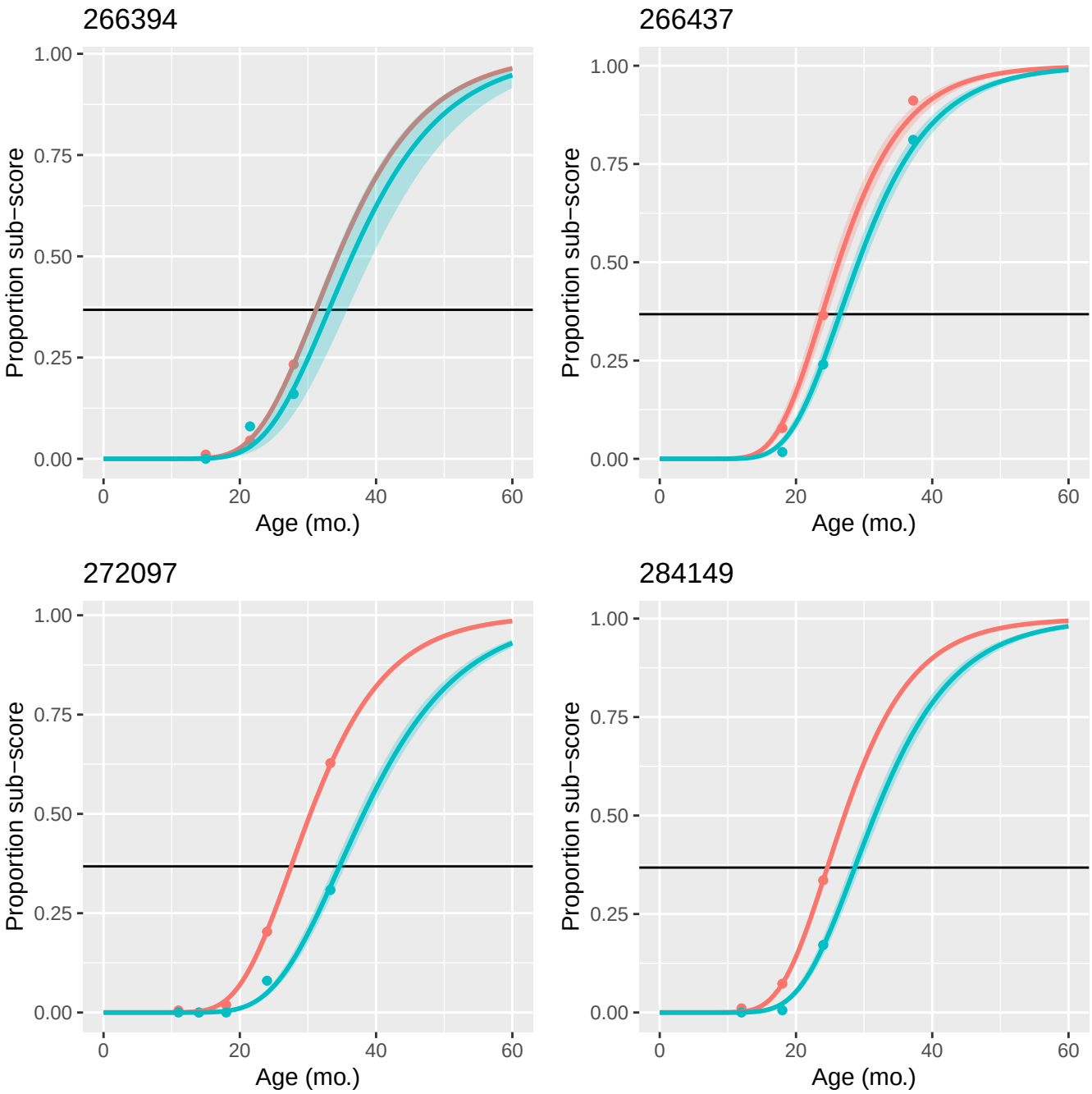

222881
223008
229768
231205
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

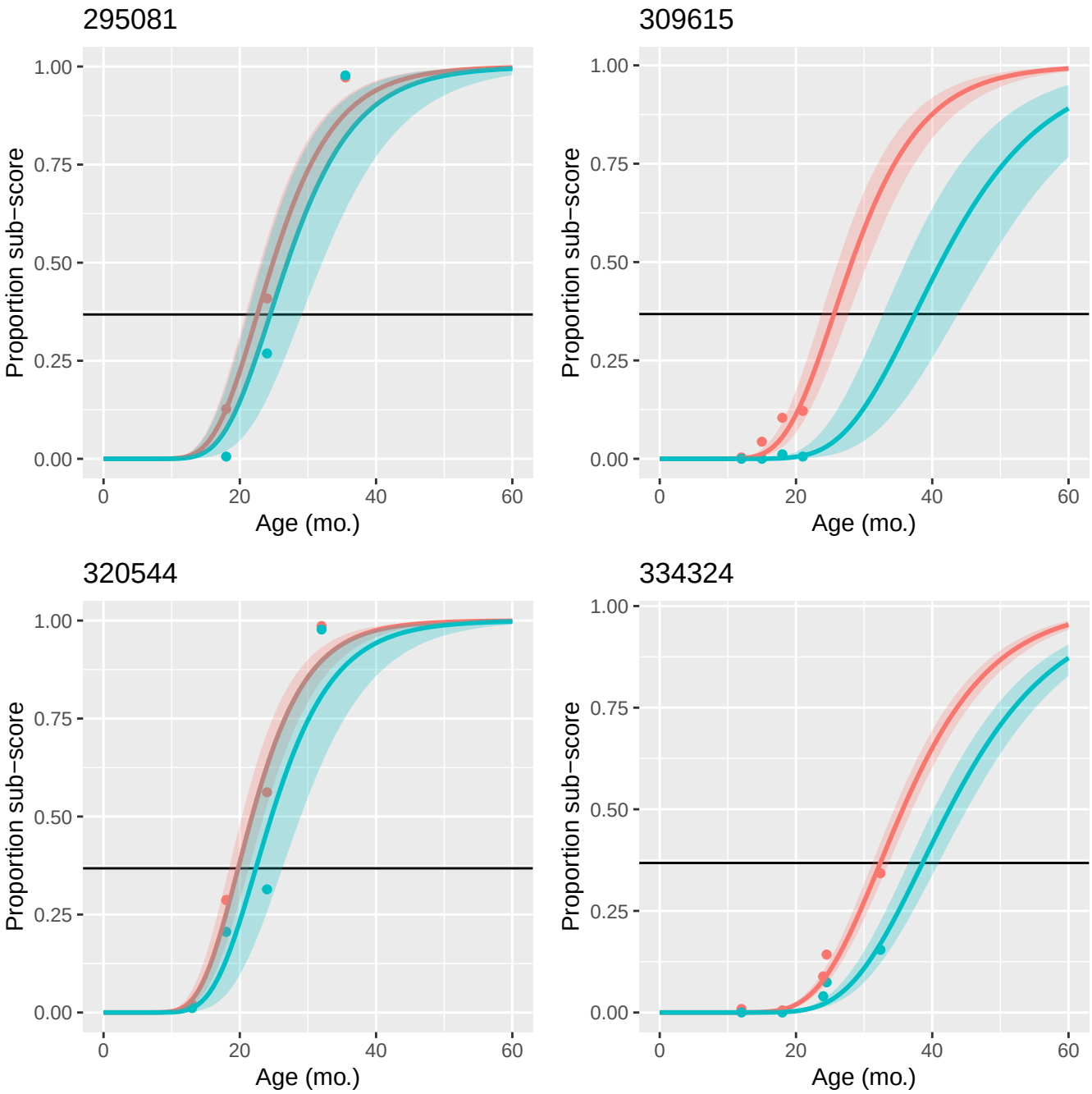


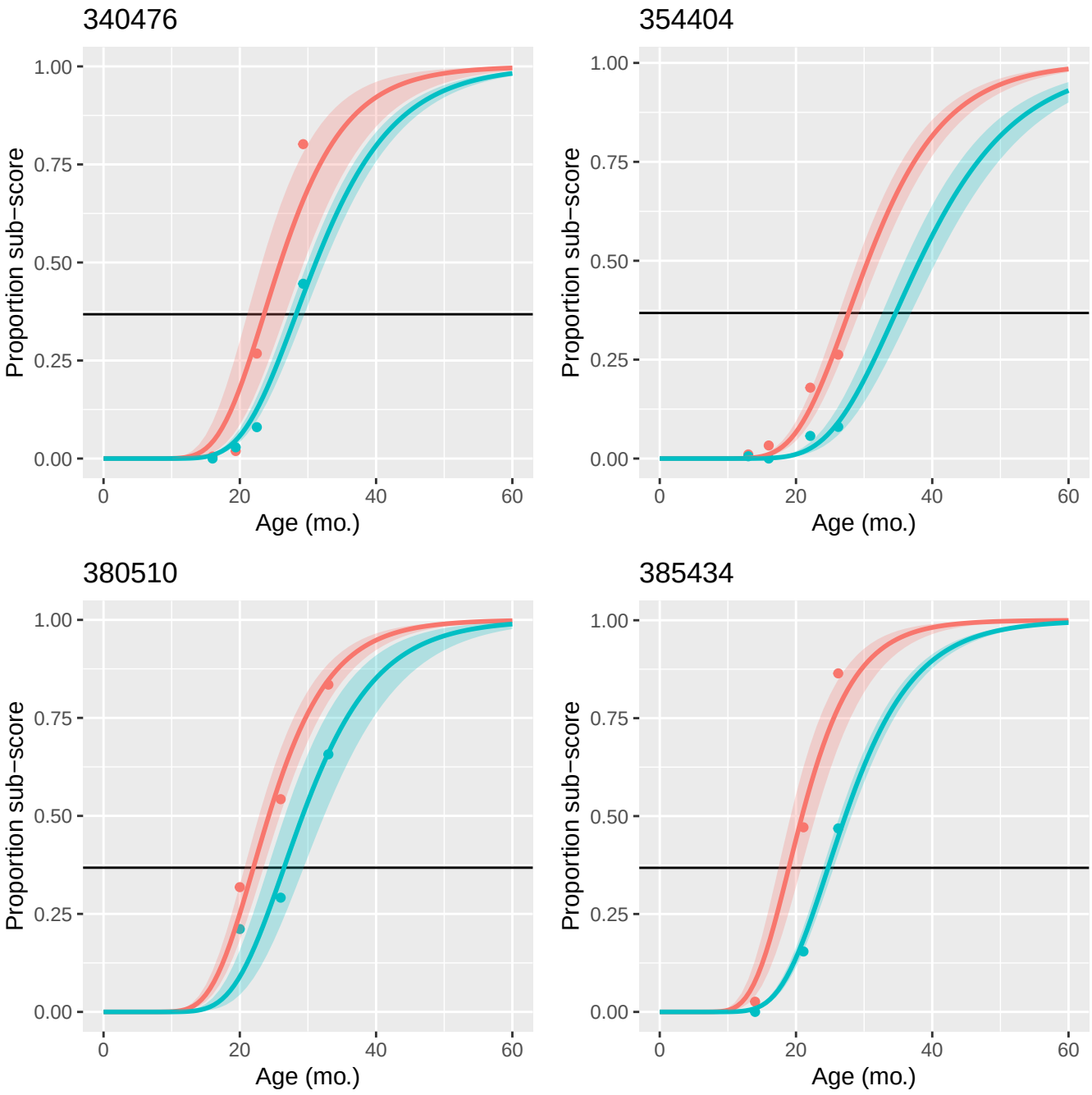

```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## [1] NA
```

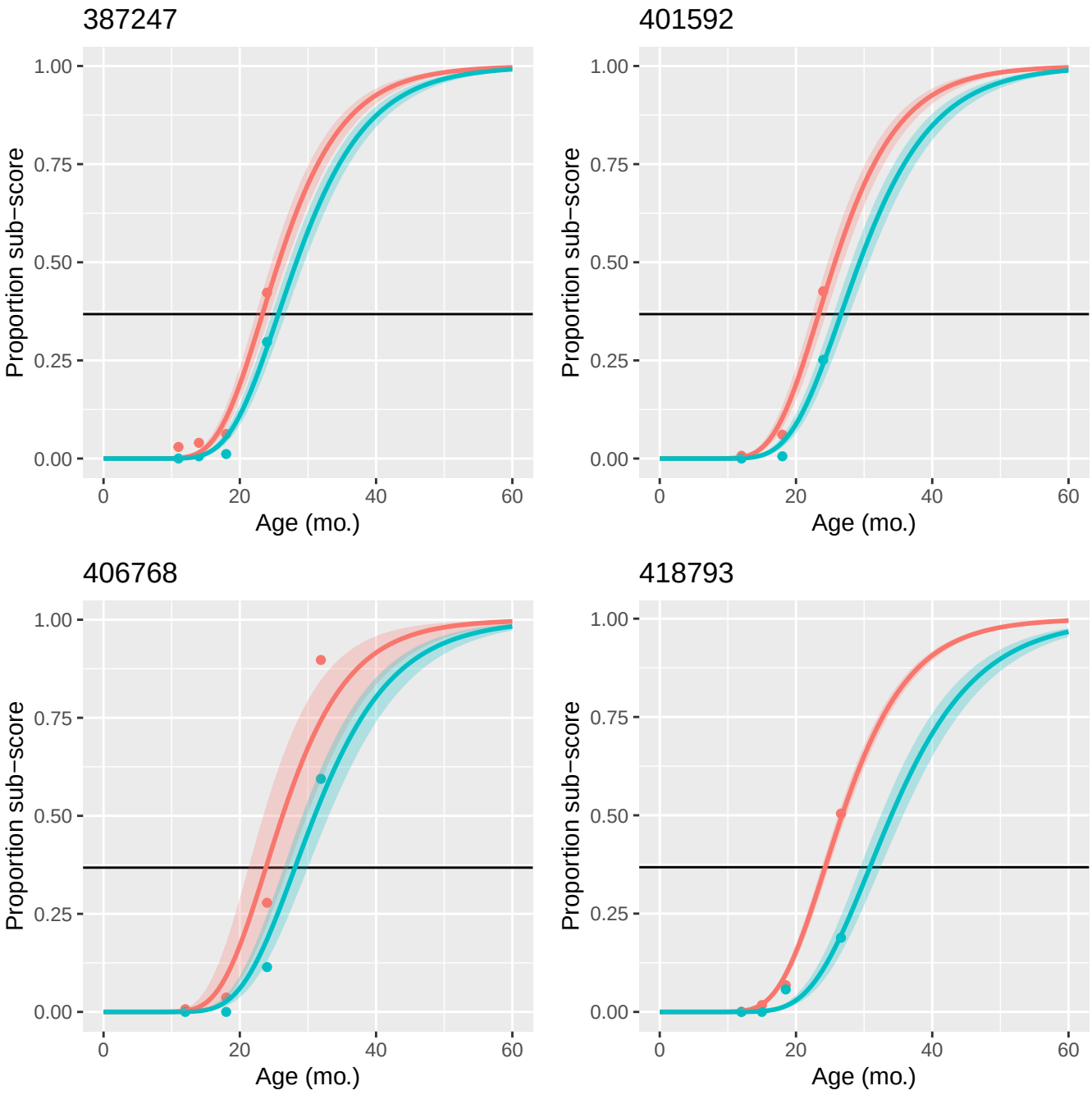


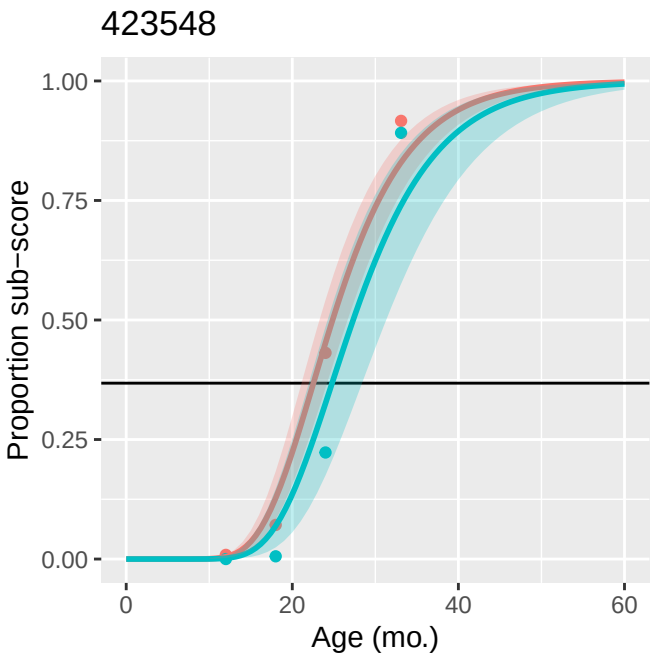


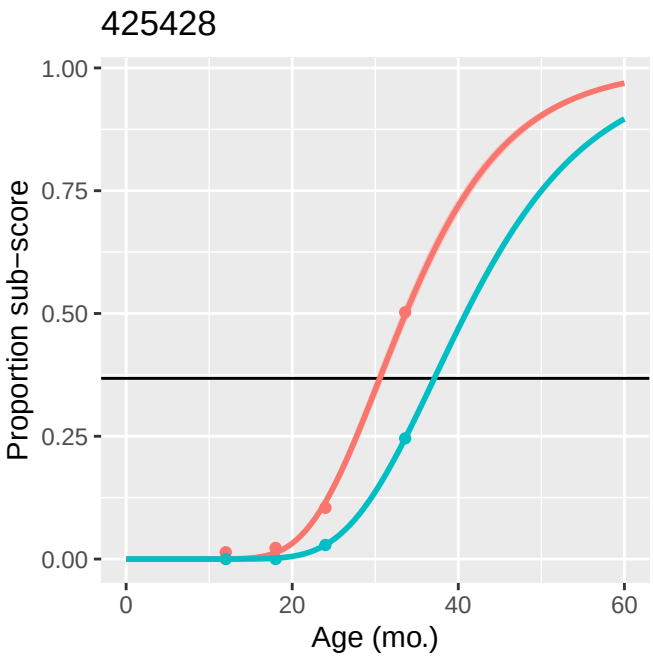

266394
266437
272097
284149
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

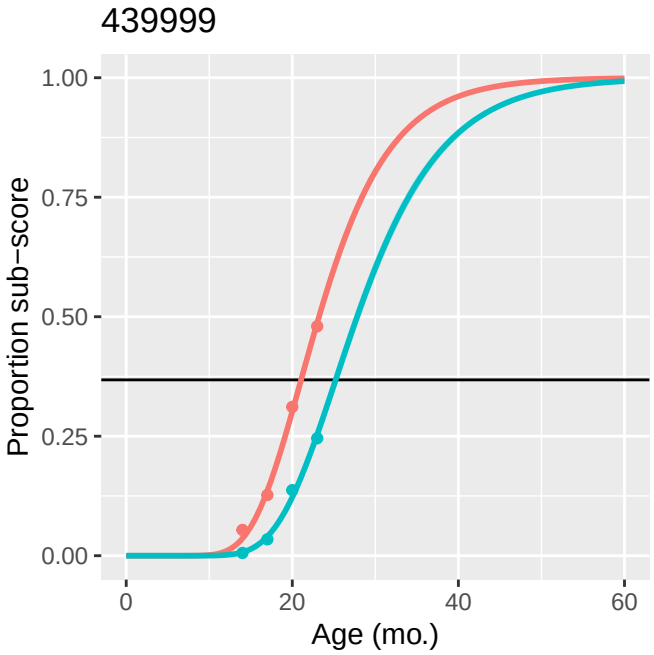

295081
309615
320544
334324
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)


340476
354404
380510
385434
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

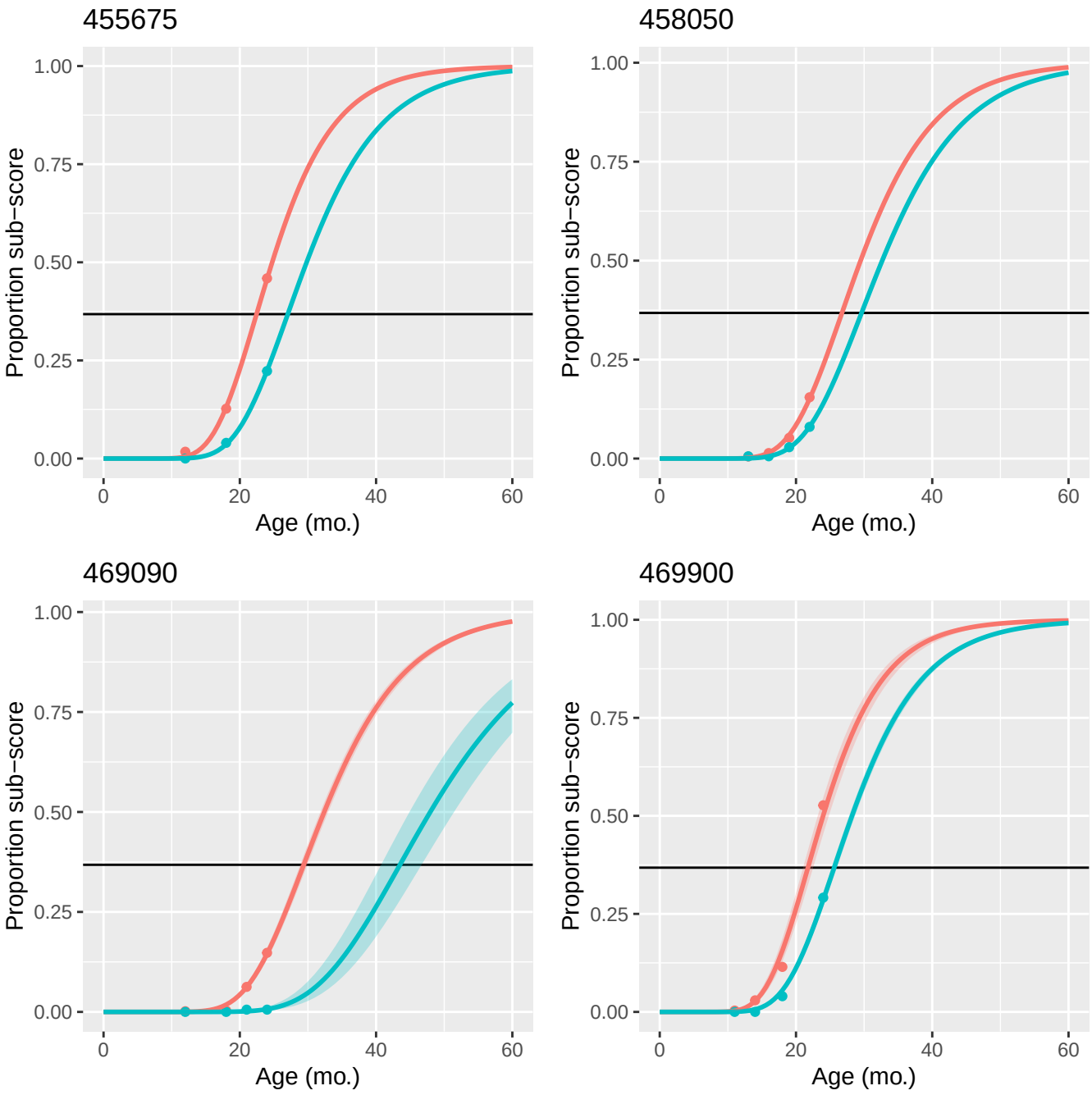

387247
401592
406768
418793
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

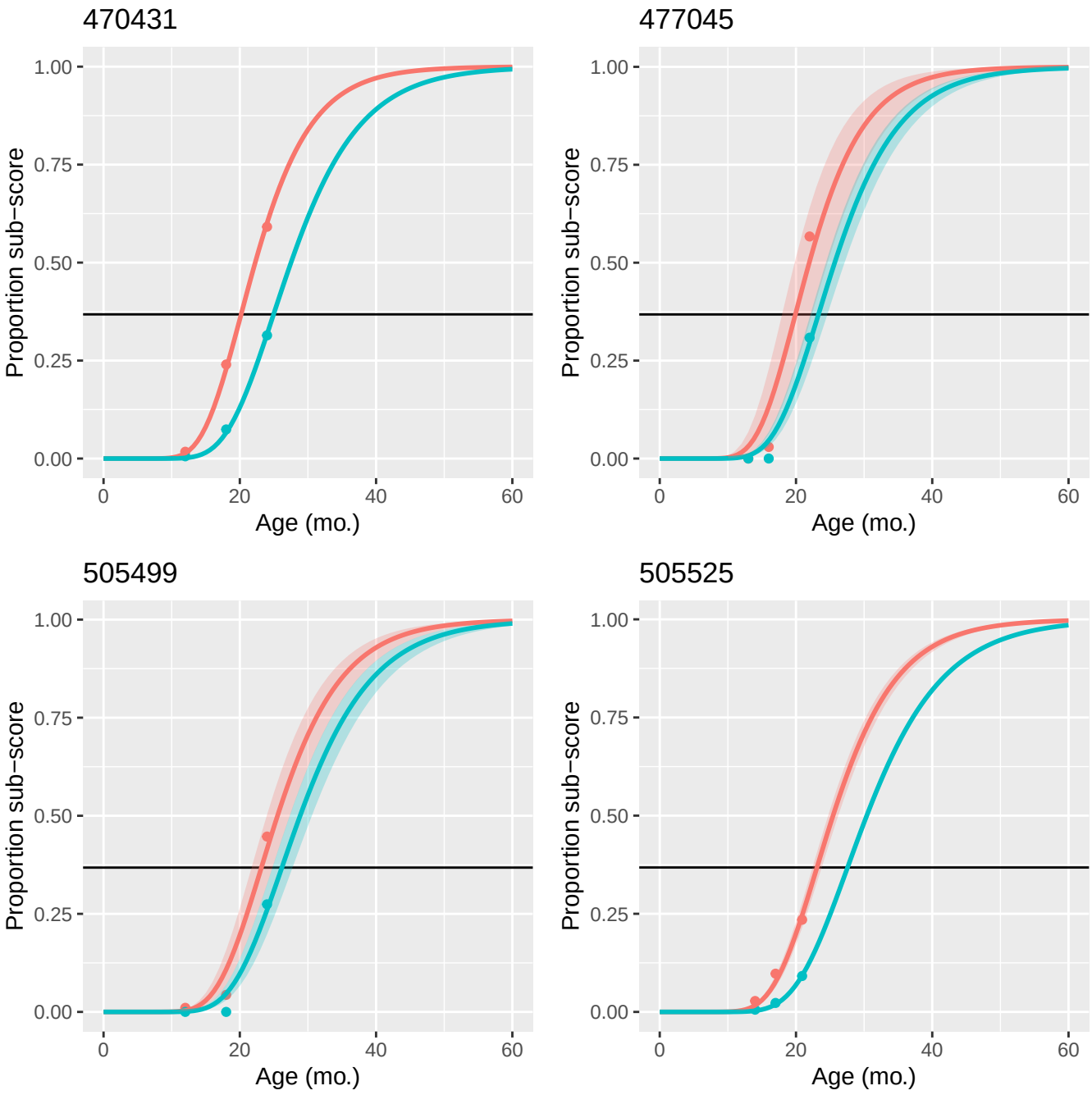


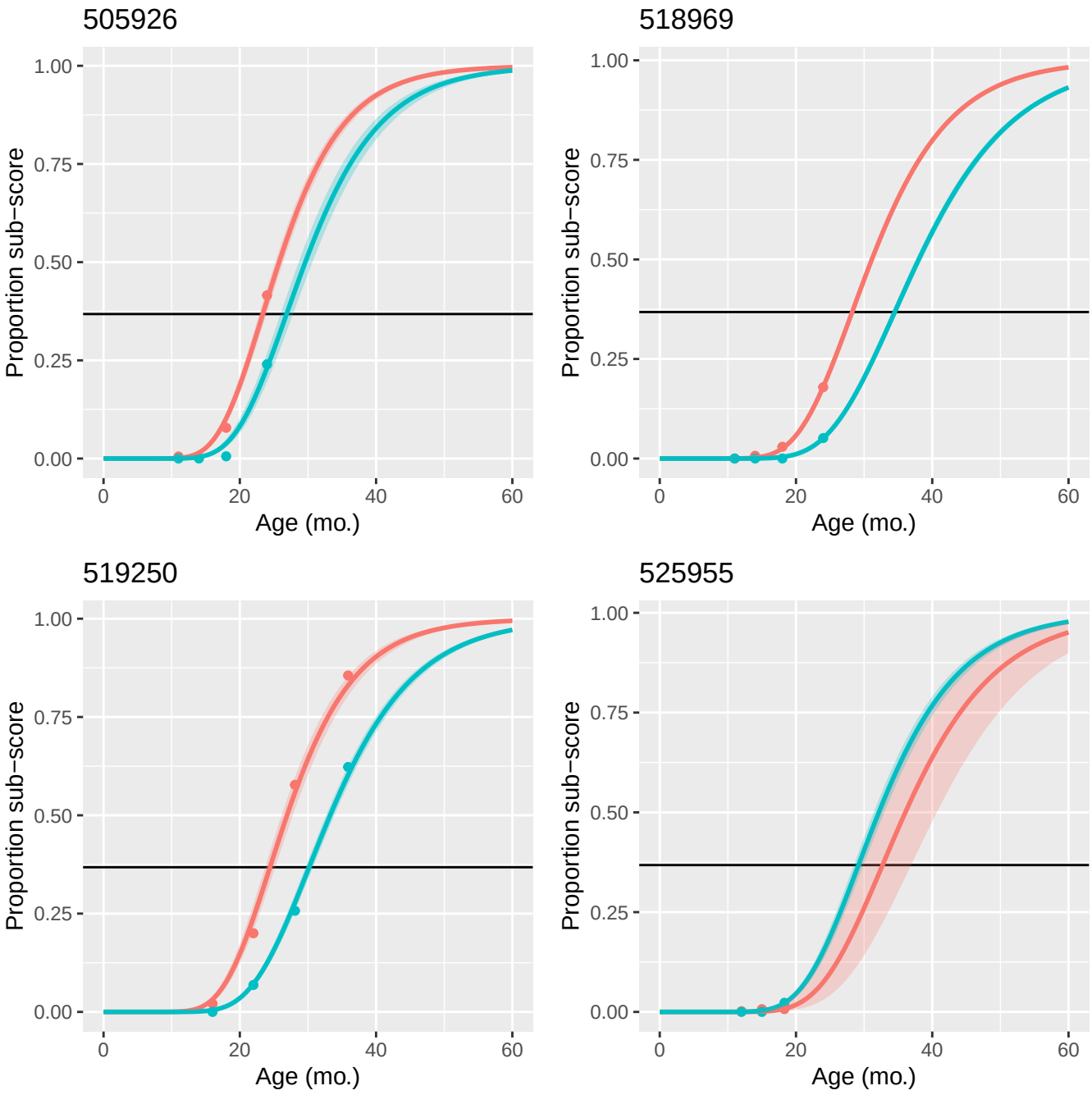


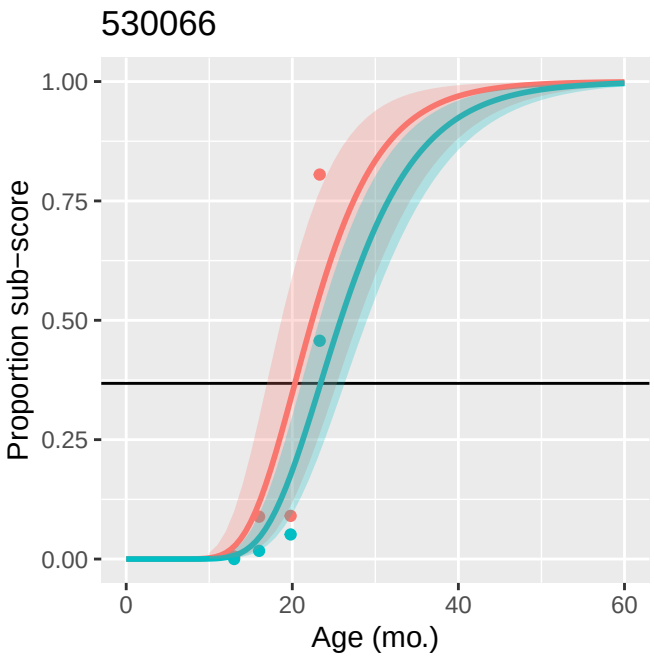



```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## [1] NA
```

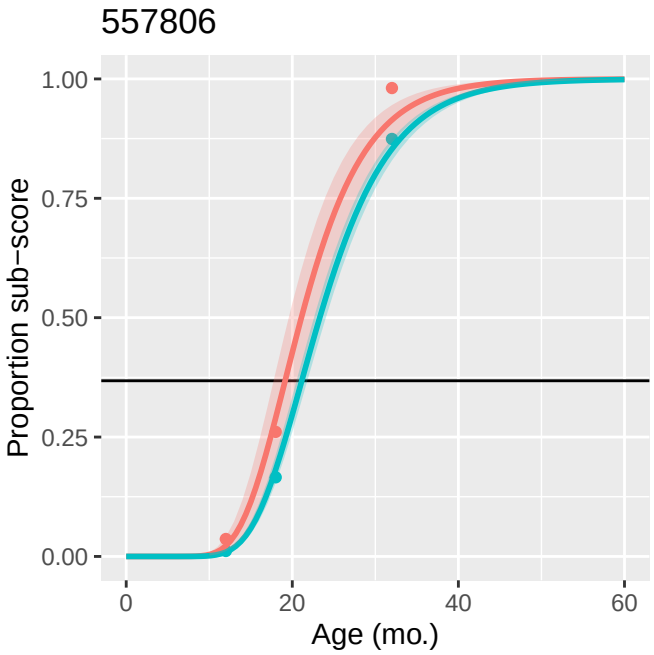

455675
458050
469090
469900
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

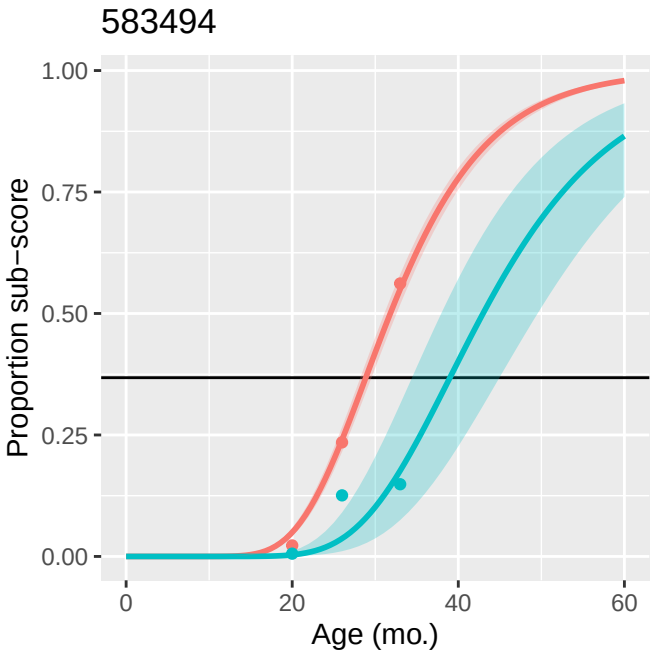

470431
477045
505499
505525
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

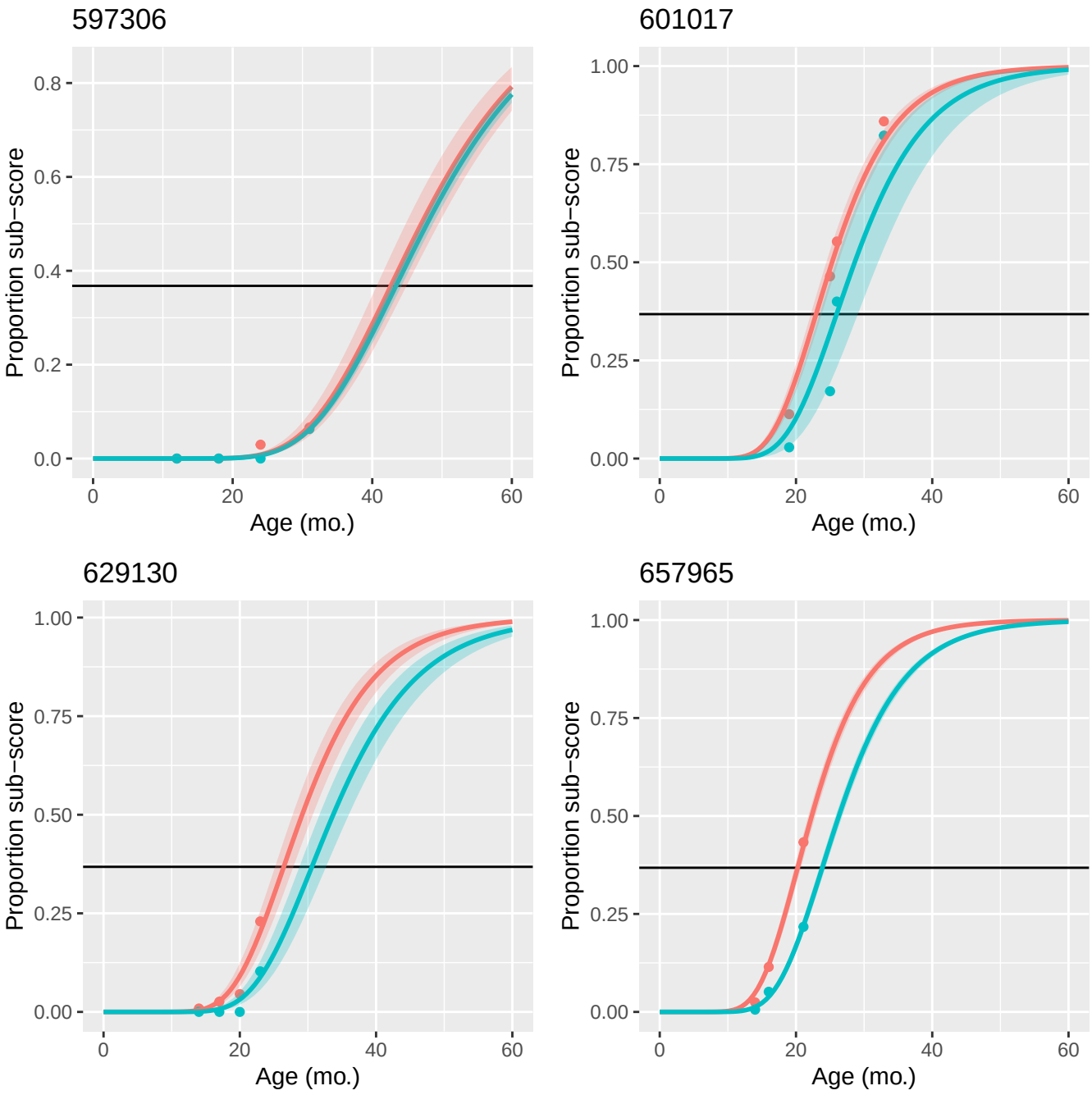

505926
518969
519250
525955
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

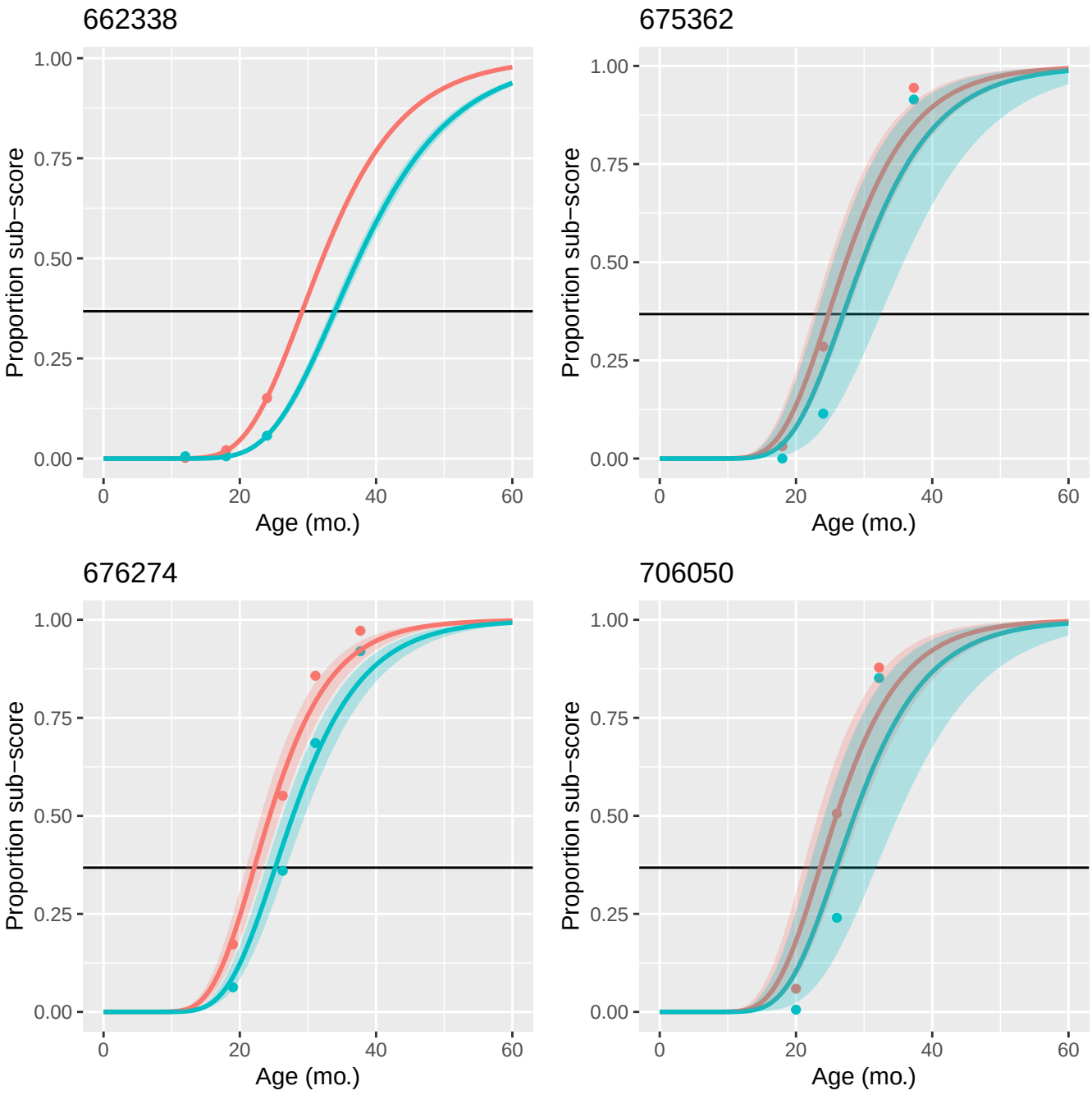


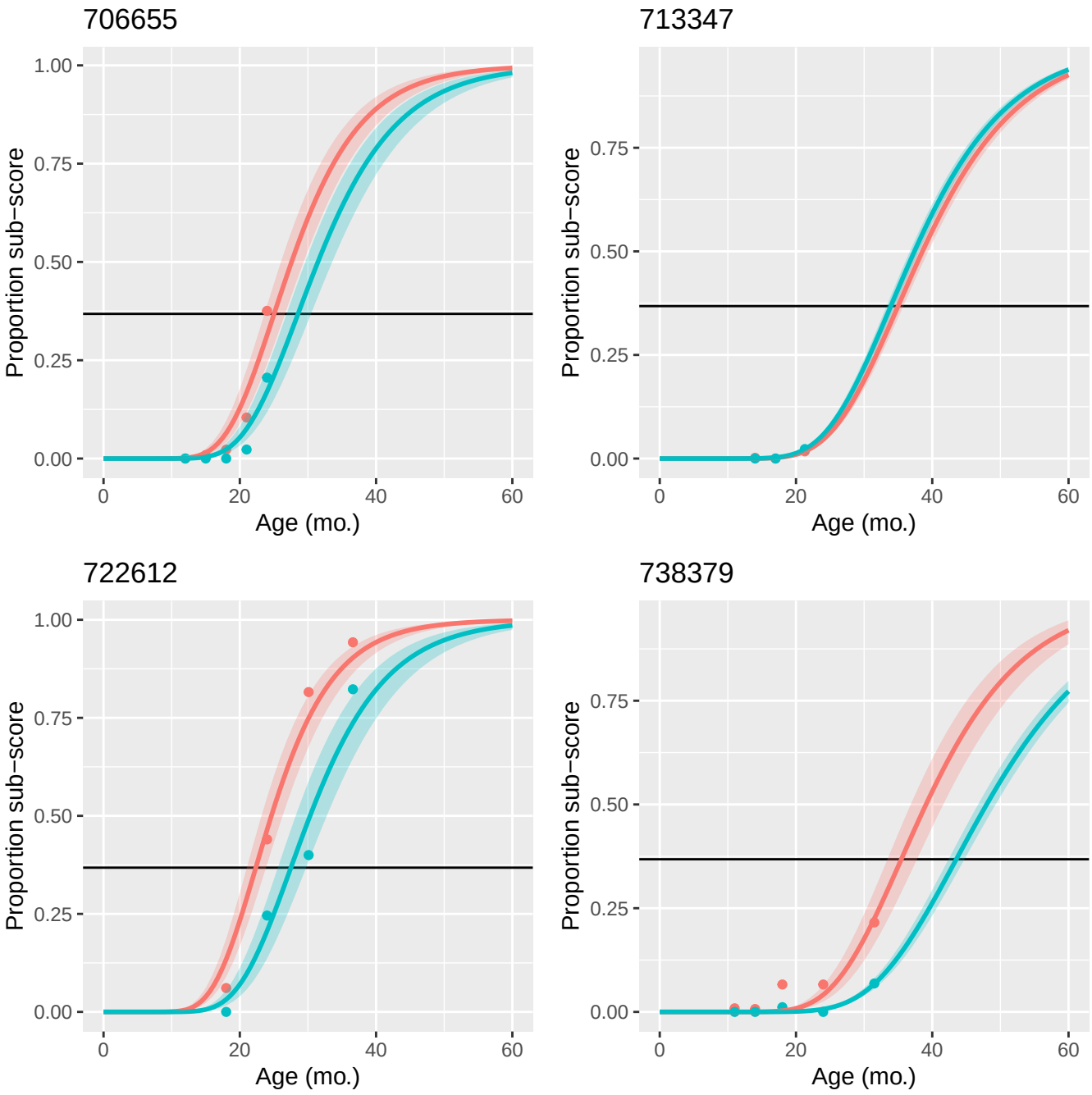

```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## [1] NA
```

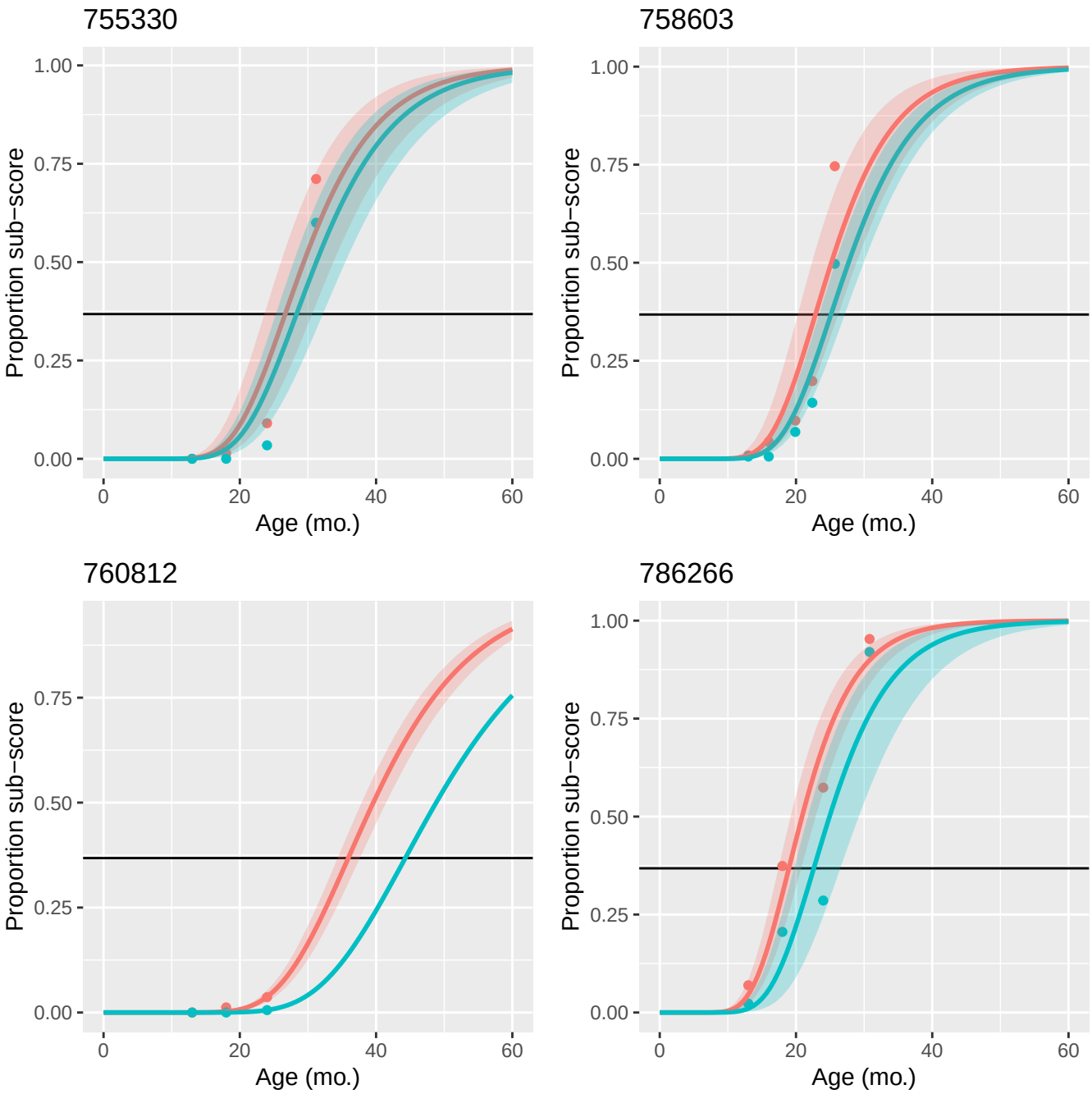


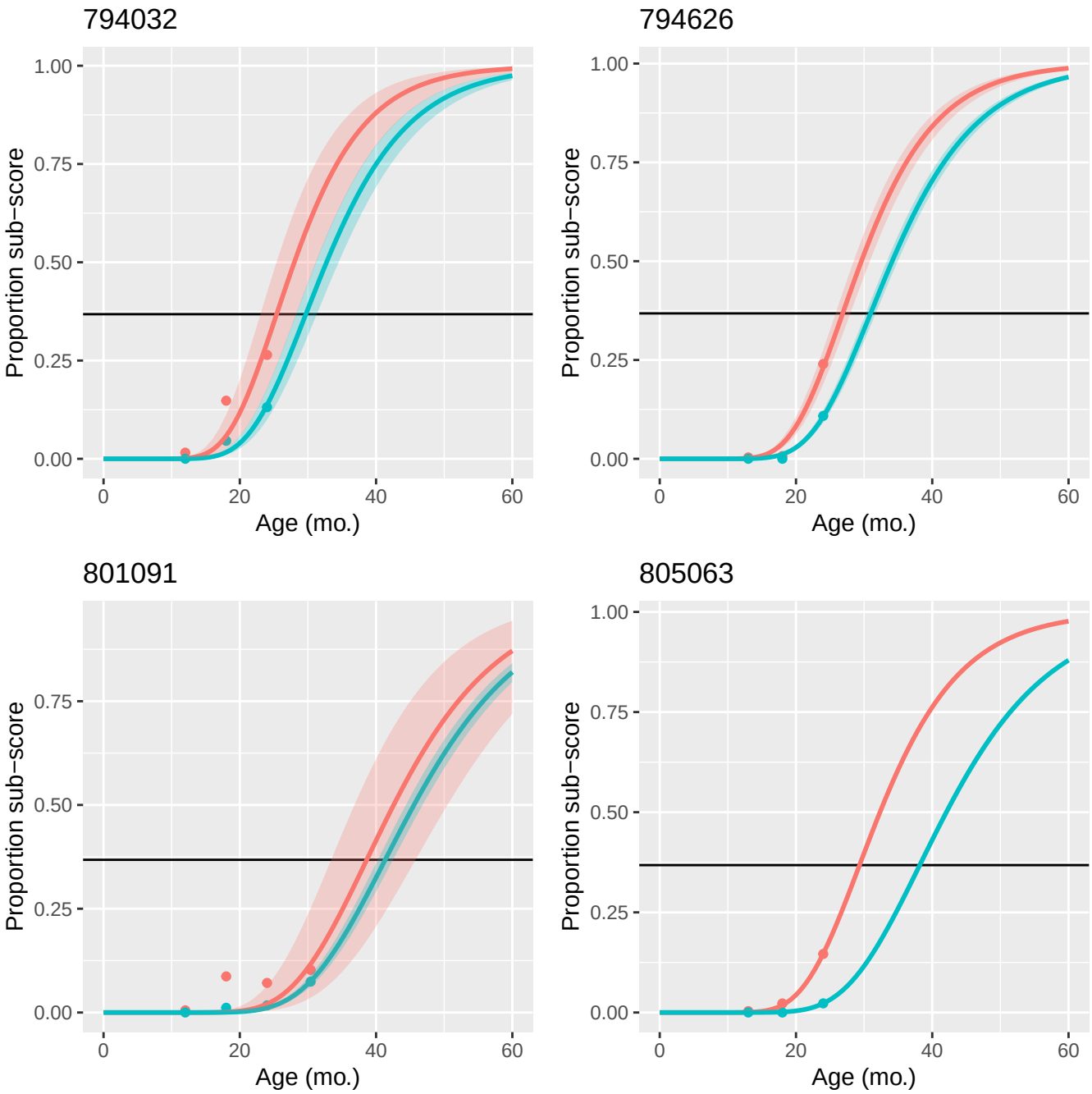


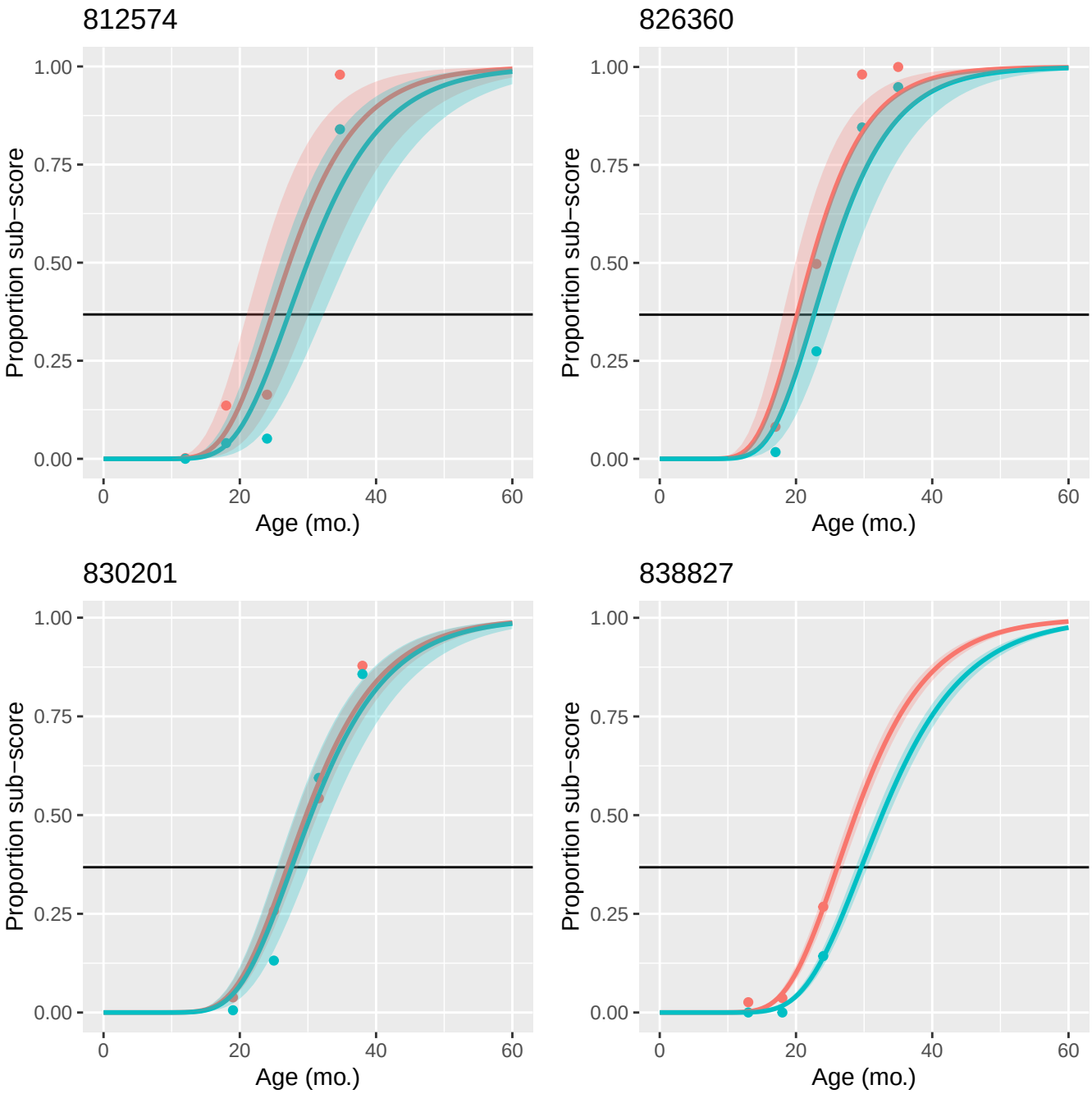

597306
601017
629130
657965
Proportion sub−score
Age (mo.)
0.0
0.2
0.4
0.6
0.8
0.00
0.25
0.50
0.75
1.00
0
20
40
60

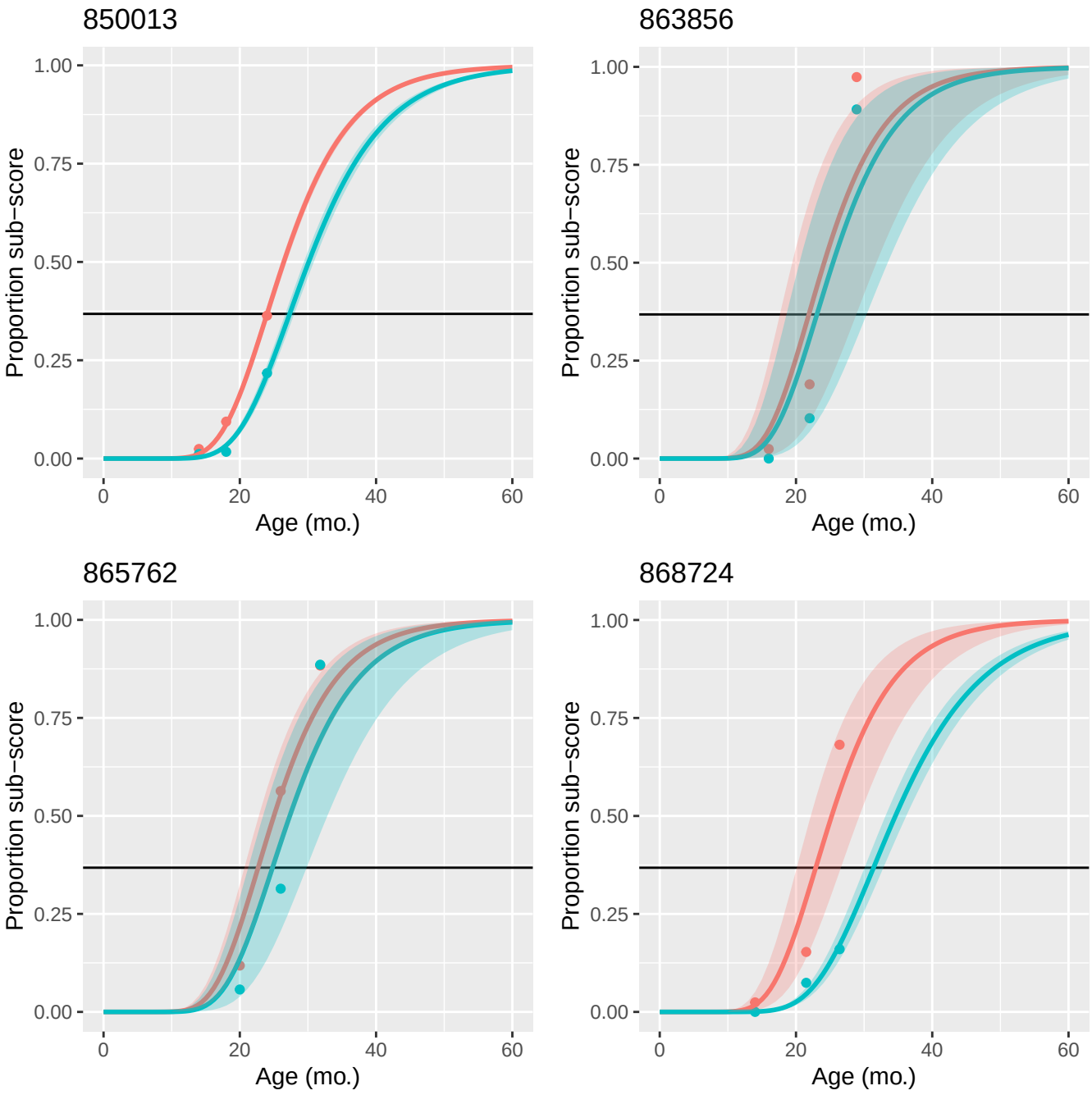

662338
675362
676274
706050
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

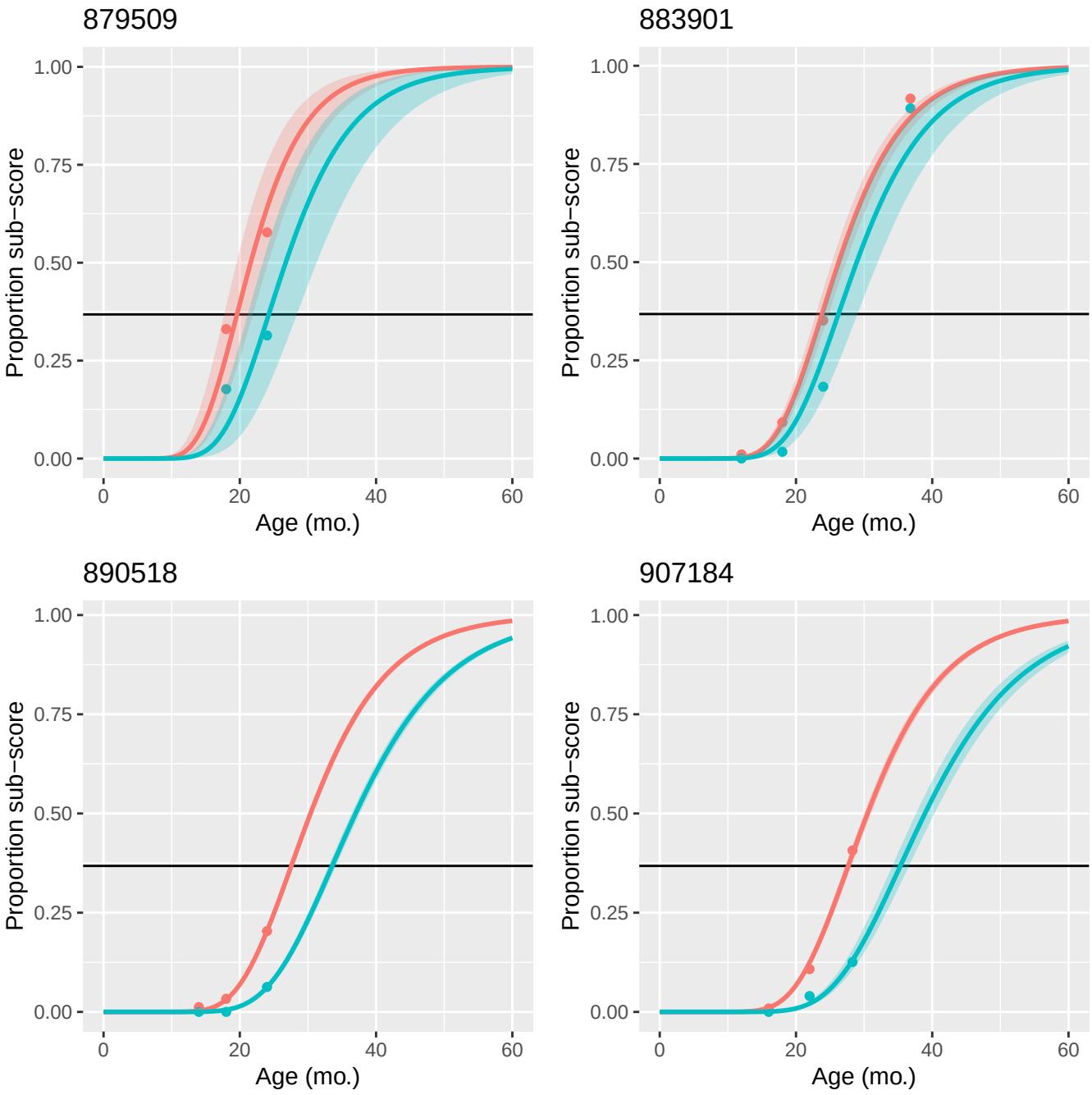

706655
713347
722612
738379
Proportion sub−score
Age (mo.)
0.00
0.25
0.50
0.75
1.00
0
20
40
60

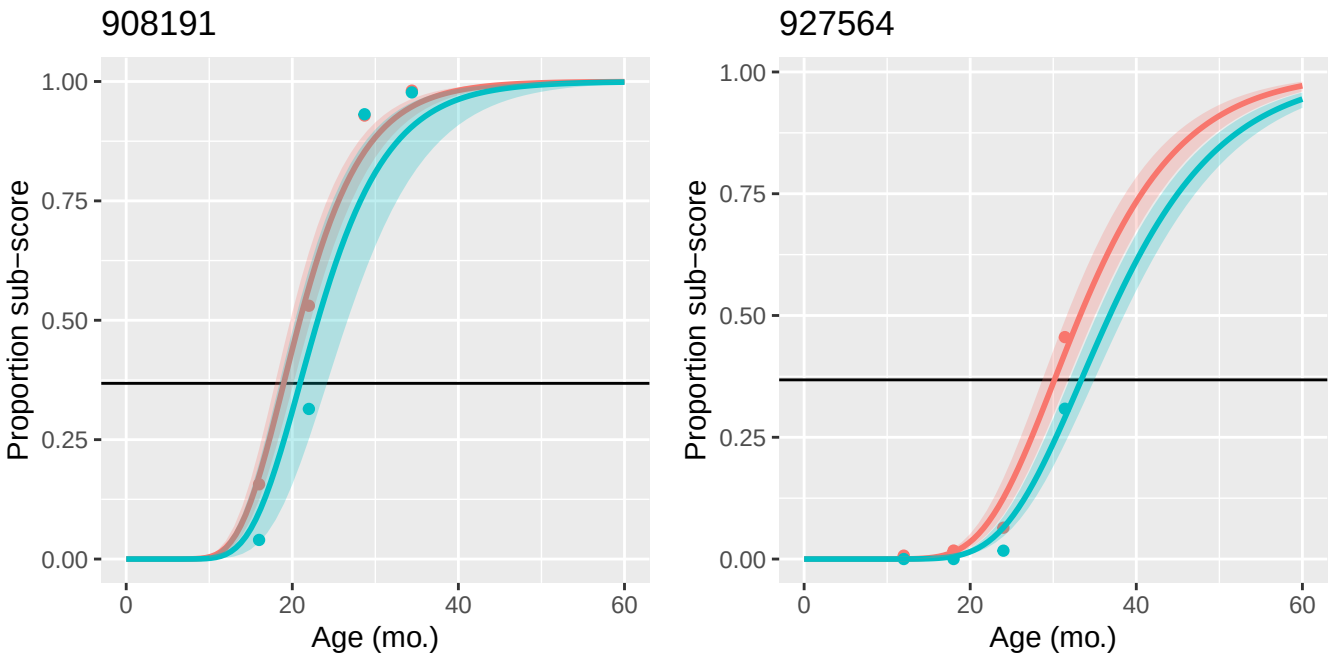

755330
758603
760812
786266
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

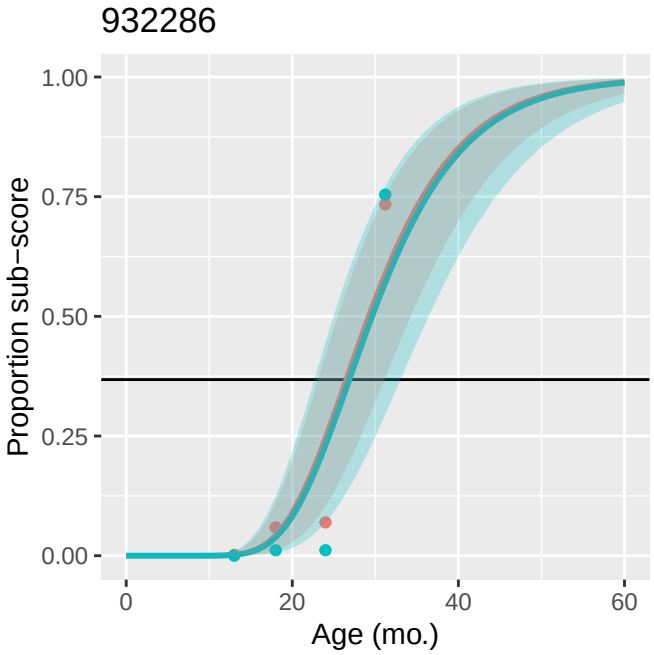

794032
794626
801091
805063
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)


812574
826360
830201
838827
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

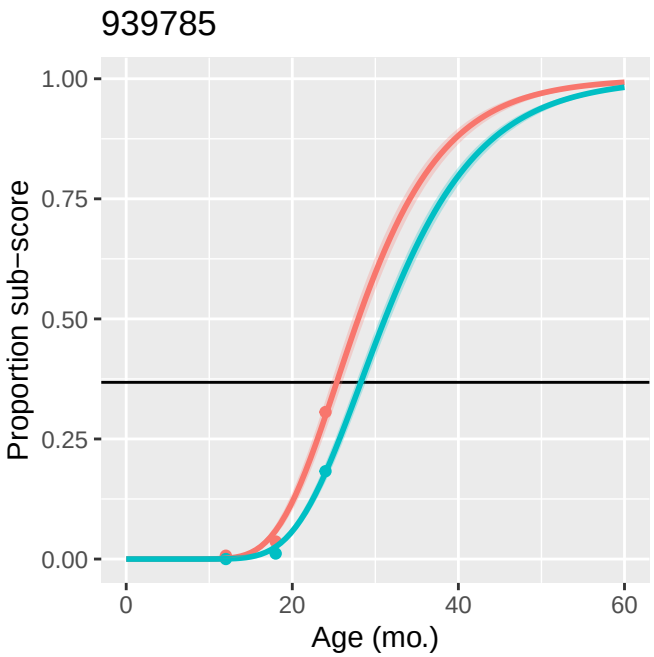

850013
863856
865762
868724
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)

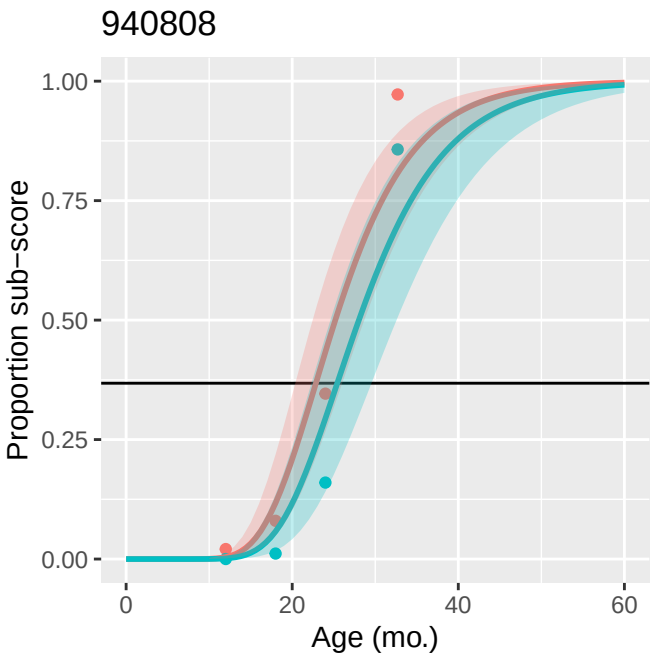

879509
883901
890518
907184
Proportion sub−score
1.00
0.75
0.50
0.25
0.00
0
20
40
60
Age (mo.)



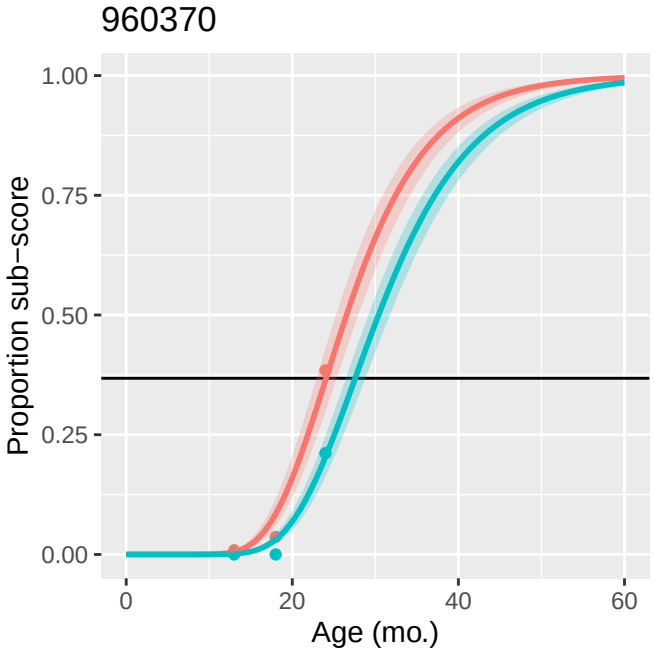


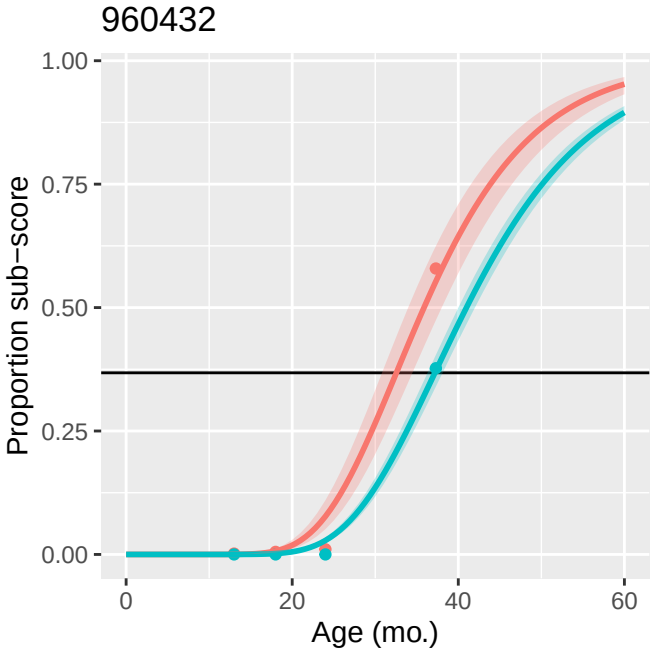

```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## [1] NA
```

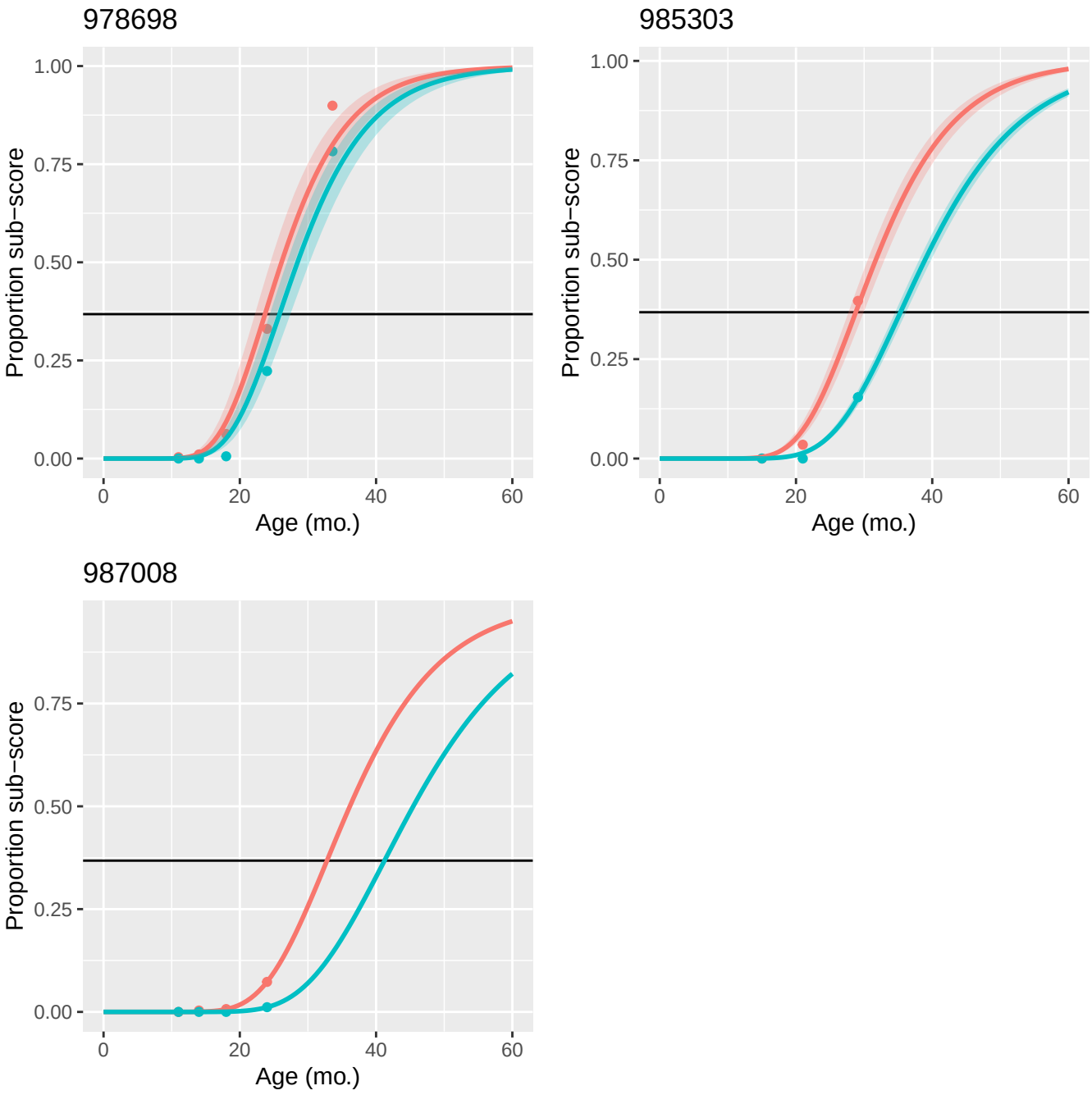





```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## [1] NA
```








```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## [1] NA
```

SE plots

```
LS.vals <- data.frame(data_id = ID3)
lex.fits <- lapply(LS.vals$data_id,
                   function(x) gomp2.fit(filter(lexsyn,
                                                data_id == x,
                                                name == "lex.p"),
```

```
                                          response = "value", max = 1,
                                          max.iter = 500) )
syn.fits <- lapply(LS.vals$data_id,
                   function(x) gomp2.fit(filter(lexsyn,
                                                data_id == x,
                                                name == "syn.p"),
                                          response = "value", max = 1,
                                          max.iter = 500) )
```

```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")),  :
##   number of iterations exceeded maximum of 500
```

```
# Get values
LS.vals <- LS.vals %>%
            mutate(lex = sapply(lex.fits, coef),
                   lex.se = sapply(lex.fits,
                                   function(x) coef(summary(x))[1, 2]),
                   syn = sapply(syn.fits,
                                function(x) ifelse(class(x) != "nls",
                                                   NA,
                                                   coef(x))),
                   syn.se = sapply(syn.fits,
                                   function(x) ifelse(class(x) != "nls",
                                                      NA,
                                                      coef(summary(x))[1, 2])))

# Value at Wi
W_i.1 <- 1 / exp(1)
LS.vals <- LS.vals %>%
            mutate(lex.m = solve.gomp2(y = W_i.1, k_g = lex, A = 1),
                   lex.lo = solve.gomp2(y = W_i.1, k_g = lex - (2 * lex.se),
                                        A = 1),
                   lex.hi = solve.gomp2(y = W_i.1, k_g = lex + (2 * lex.se),
                                        A = 1),
                   syn.m = solve.gomp2(y = W_i.1, k_g = syn, A = 1),
                   syn.lo = solve.gomp2(y = W_i.1, k_g = syn - (2 * syn.se),
                                        A = 1),
                   syn.hi = solve.gomp2(y = W_i.1, k_g = syn + (2 * syn.se),
                                        A = 1),
                   synMlex = syn.m - lex.m)

# LS.vals.melt <-  LS.vals %>%
#                    select(-syn.less.lex) %>%
#                    pivot_longer(-data_id)
```

```
png("plots/rangediff-presentation.png", width = 5, height = 5, units = "in",
    res = 300)

ggplot(LS.vals) +
  geom_pointrange(aes(x = data_id,
                      y = 0, ymin = lex.hi - lex.m, ymax = lex.lo - lex.m),
                  color = hue_pal()(2)[1],
                  size = 0.75, fatten = 0.25) +
  geom_pointrange(aes(x = data_id,
                      y = syn.m - lex.m, ymin = syn.hi - lex.m,
                      ymax = syn.lo - lex.m),
                  color = hue_pal()(2)[2],
                  size = 0.75, fatten = 0.25) +
  coord_flip() +
  scale_x_discrete(labels = NULL) +
  scale_y_continuous(limits = c(-6, NA)) +
  labs(y = "Time between {lex = Wi} and {syn = Wi} (mo)", x = "Participant")
```

```
## Warning: Removed 1 rows containing missing values (geom_pointrange).
```

```
## Warning: Removed 6 rows containing missing values (geom_pointrange).
```

```
dev.off()
```

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

```
png("plots/differences-presentation.png", width = 5, height = 5, units = "in",
    res = 300)

ggplot(LS.vals, aes(synMlex)) +
  geom_histogram(binwidth = .25) +
  labs(x = "Ti,syn - Ti,lex (mo.)")
```

```
## Warning: Removed 6 rows containing non-finite values (stat_bin).
```

```
mean(LS.vals$synMlex, na.rm = TRUE)
```

```
## [1] 4.158983
```

```
dev.off()
```

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

```
t.test(LS.vals$lex.m, LS.vals$syn.m, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  LS.vals$lex.m and LS.vals$syn.m
## t = -15.413, df = 98, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.694475 -3.623491
## sample estimates:
## mean of the differences
##               -4.158983
```

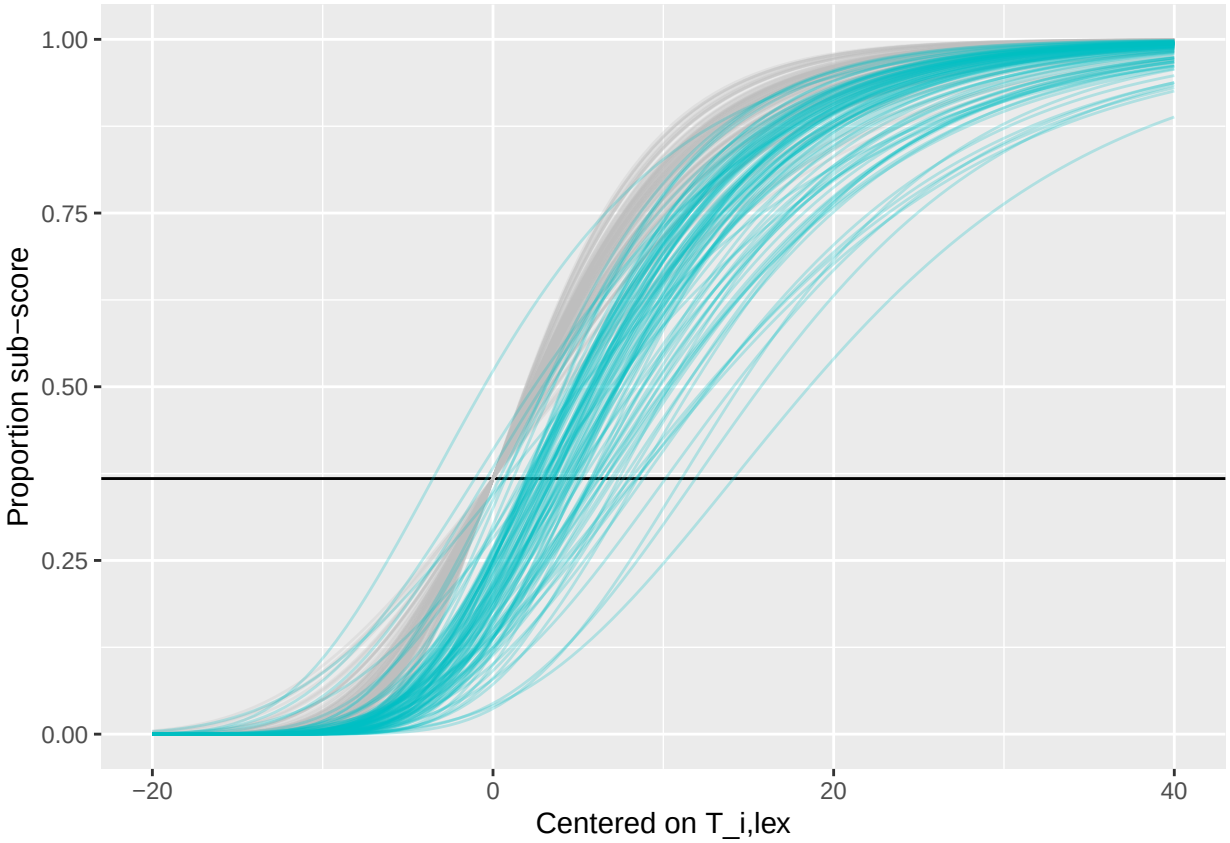
Bootstrapping significance of factor structure

Although the CFA technically failed (fit parameters marginal), it does seem there is a real difference in T_i between syntax and lexical acquisition. We can compare this difference by randomly selecting identically sized parts, calculating the average difference and comparing that distribution to the average difference between T_i, syn and T_i, lex (4.16).

All curves

```
LS.vals <- LS.vals %>%
            mutate(syn.m)

# coeflines <-
# alply(as.matrix(coefs), 1, function(coef) {
#   stat_function(fun=function(x){coef[1]*x^coef[2]}, colour="grey")
# })

gformula2 <- function(kg, W_0 = .Machine$double.eps, A = 1, adj = 0) {

  f <- function(.x) {W_0 * (A/W_0)^(1 - exp(-kg * (.x + adj)))}
  return(f)

}

lexlines <- alply(as.matrix(LS.vals[, c("lex", "lex.m")]), 1,
                  function(x) stat_function(fun = gformula2(x[1], adj = x[2]),
                                            color = "grey",
                                            alpha = .25) )

synlines <- alply(as.matrix(LS.vals[, c("syn", "lex.m")]), 1,
                  function(x) stat_function(fun = gformula2(x[1], adj = x[2]),
                                            color = colors[2],
                                            alpha = 0.25) )

ggplot(LS.vals) +
  scale_x_continuous(limits = c(-20, 40)) +
  scale_y_continuous(limits = c(0, 1)) +
  geom_hline(yintercept = 1 / exp(1)) +
  labs(x = "Centered on T_i,lex", y = "Proportion sub-score") +
  lexlines + synlines
```

```
## Warning: Removed 101 rows containing missing values (geom_path).

## Warning: Removed 101 rows containing missing values (geom_path).

## Warning: Removed 101 rows containing missing values (geom_path).

## Warning: Removed 101 rows containing missing values (geom_path).

## Warning: Removed 101 rows containing missing values (geom_path).

## Warning: Removed 101 rows containing missing values (geom_path).
```


1.00
0.75
0.50
0.25
0.00
Proportion sub−score
−20
0
20
40
Centered on T_i,lex