

Growth Curving

Trevor K.M. Day

2020-06-24

Contents

Setup	1
Introduction	1
Basic Gompertz	2
Gompertz II: Electric Pertzaloo	4
Reducing observations	5
Modeling across subjects	6
Distribution of k_u	32
Lexical vs syntactic	34
Lexical vs. syntactic across subjects	34
SE plots	63
All curves	64
Package info	66
References	66

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 K. M. C. Tjørve and Tjørve (2017), 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 - NO) / NO) * exp(-r * t)):
##      K    NO   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 row(s) containing missing values (geom_path).

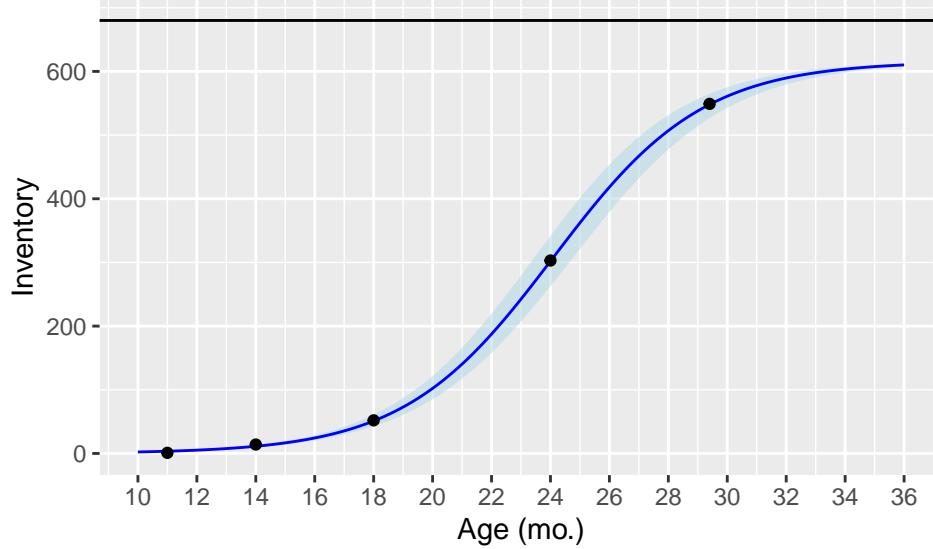


Figure 1: Gompertz curve from growthcurver package

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 otherwise, 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")

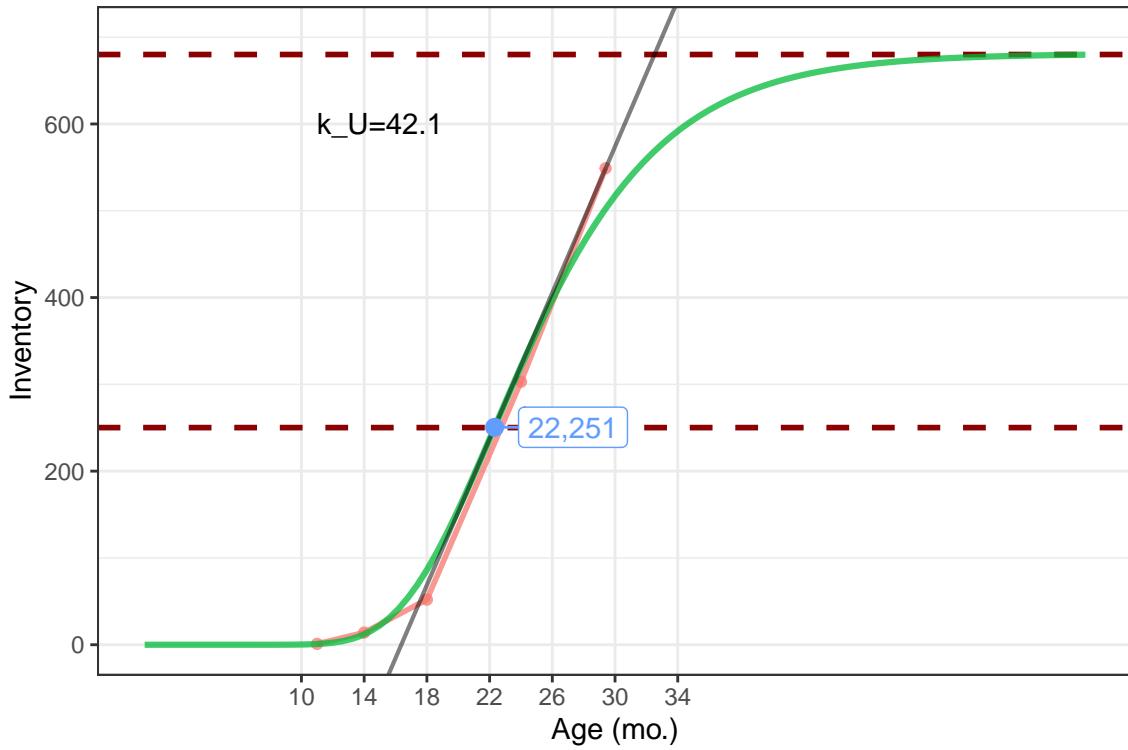
return(result)

}

```

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.

```
## Warning: `data` is not used by stat_function()
```



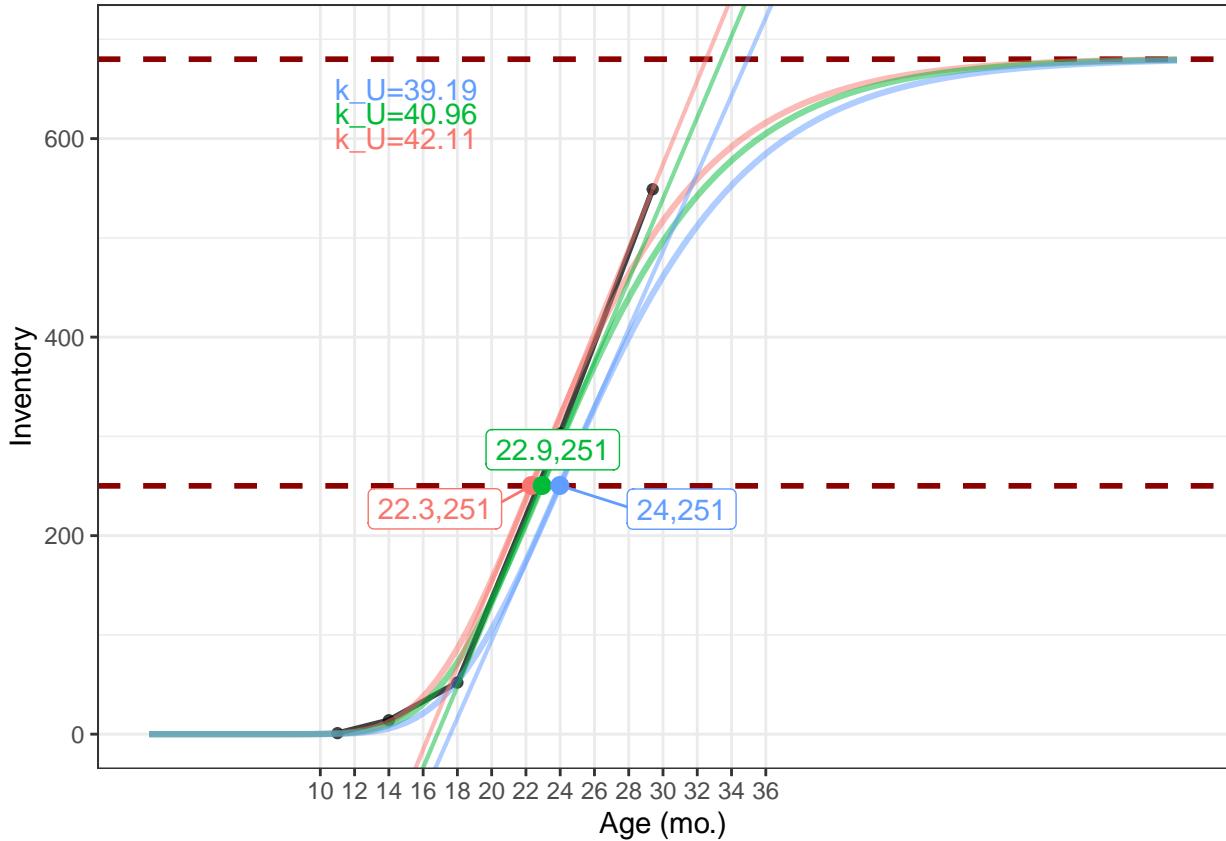
Reducing observations

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

```
## Warning: `data` is not used by stat_function()

## Warning: `data` is not used by stat_function()

## Warning: `data` is not used by stat_function()
```



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

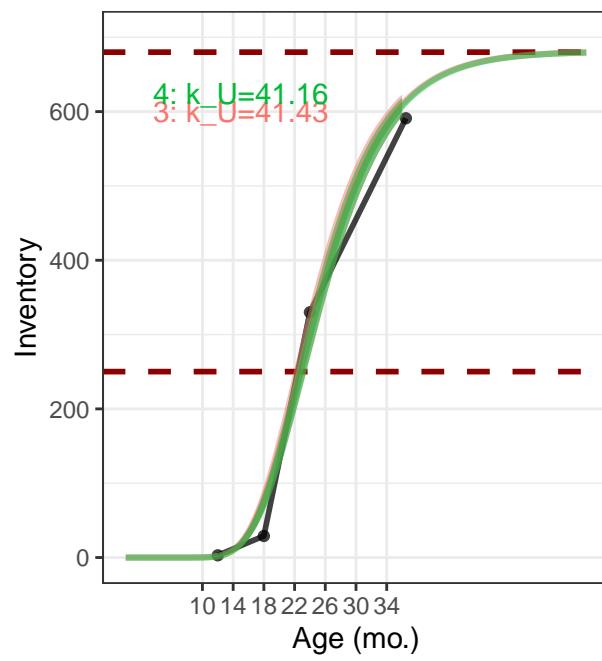
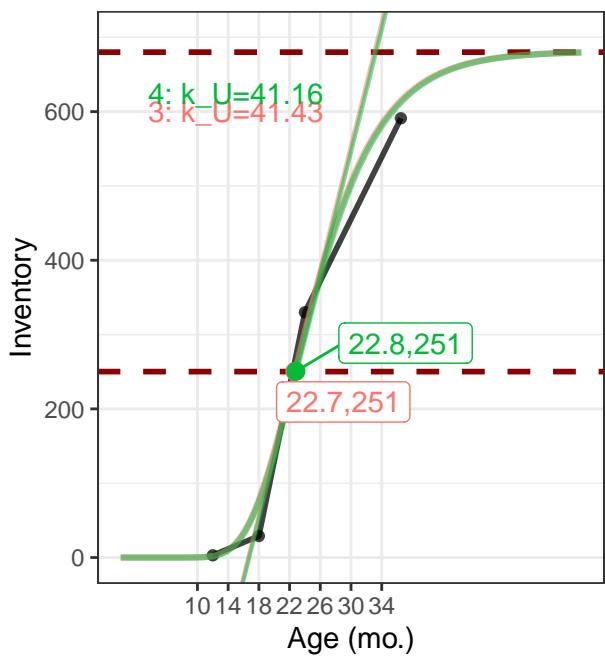
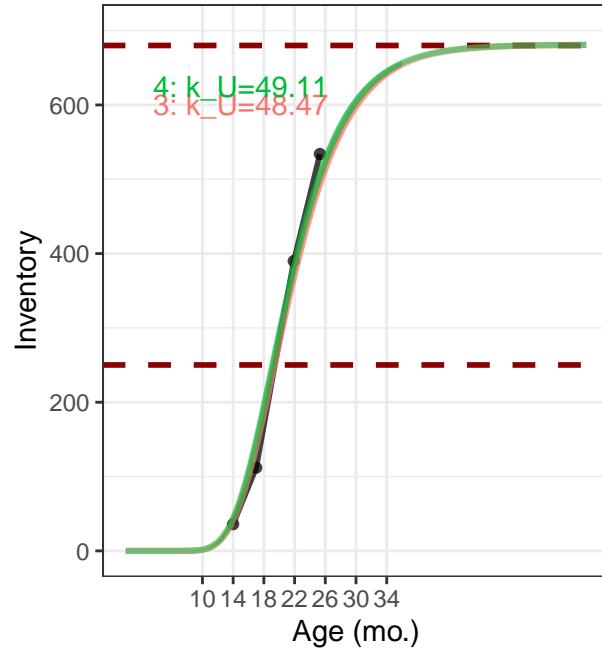
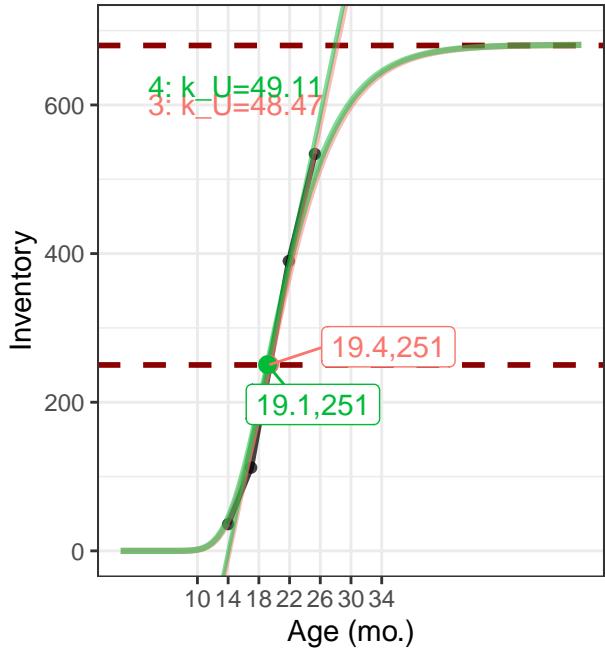
Modeling across subjects

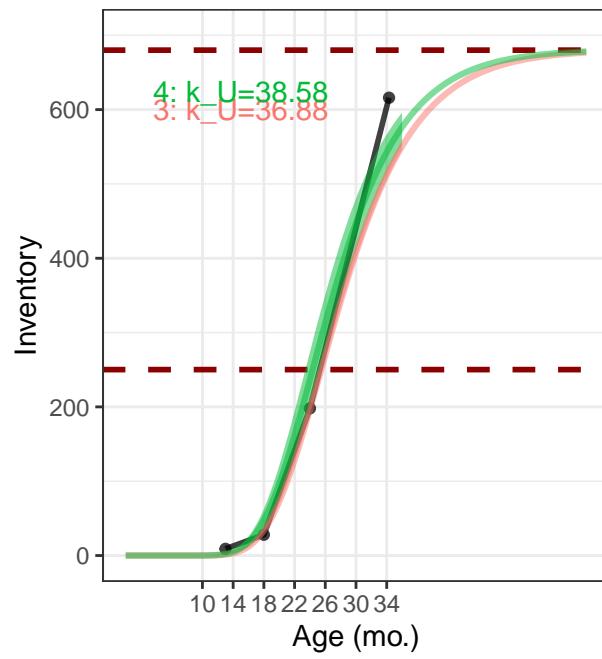
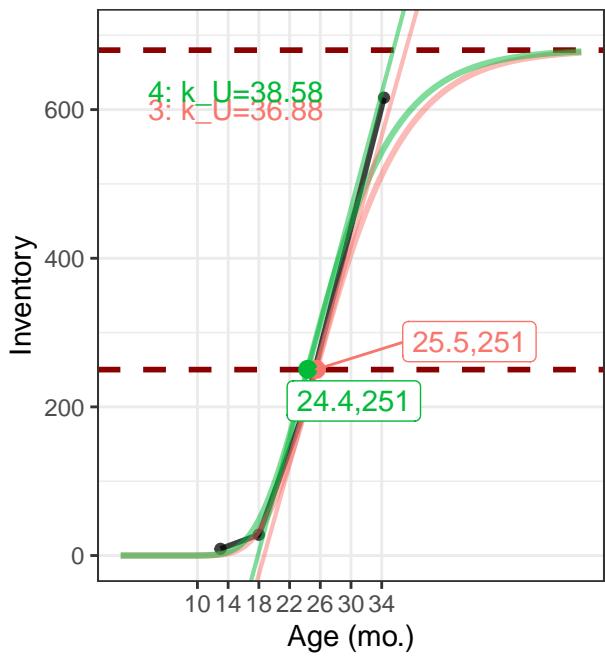
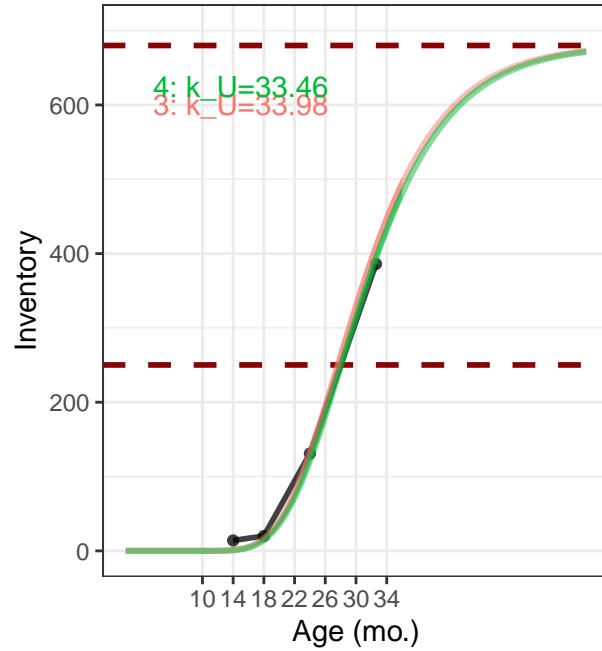
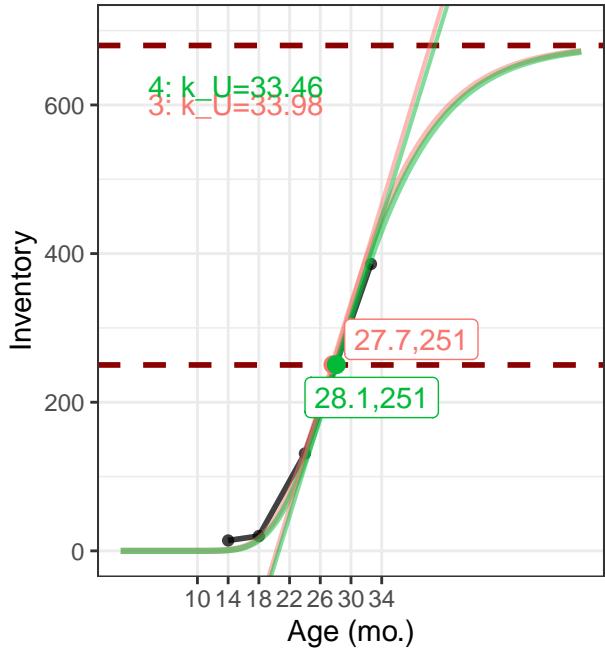
```
# Load data as a list
test_all <- read_csv("../data/test-subjs.csv") %>%
  arrange(data_id, age) %>%
  split(., f = .$data_id)

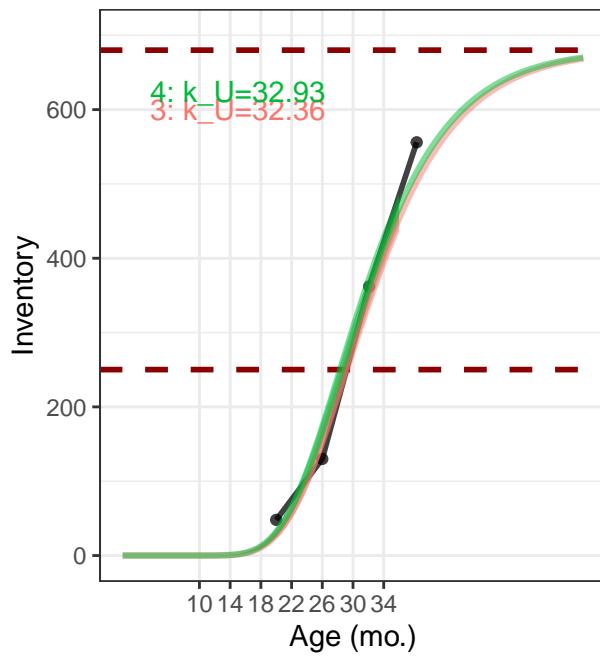
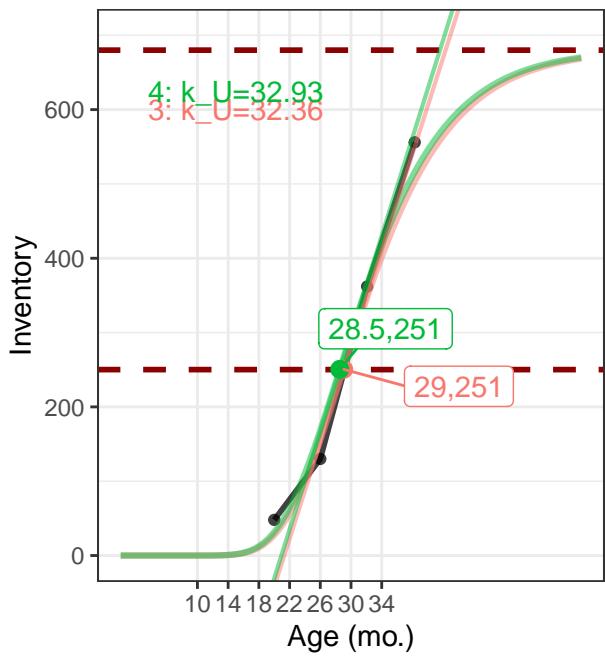
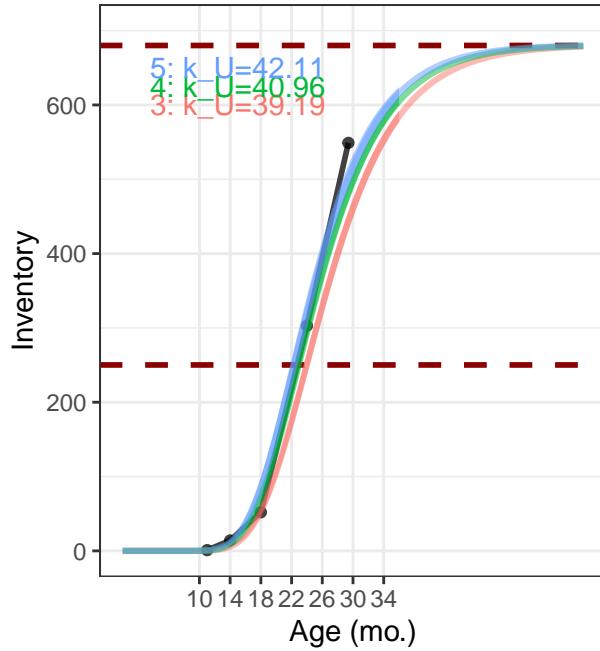
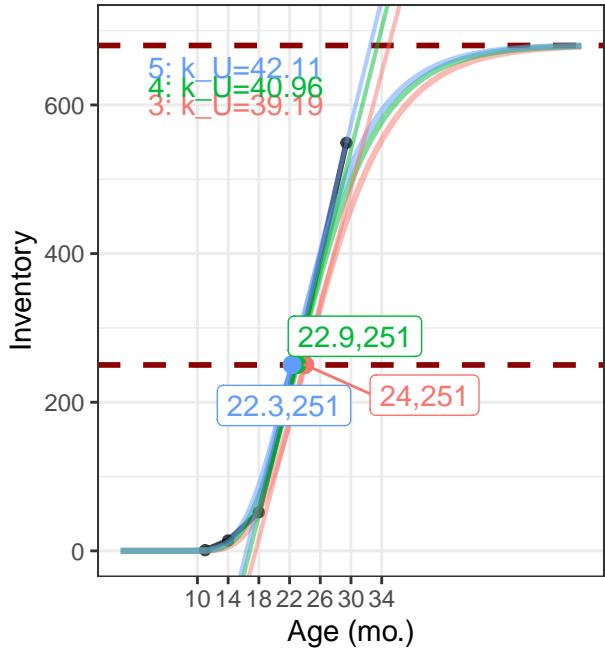
## Parsed with column specification:
## cols(
##   data_id = col_double(),
##   age = col_double(),
##   inventory = col_double()
## )
```

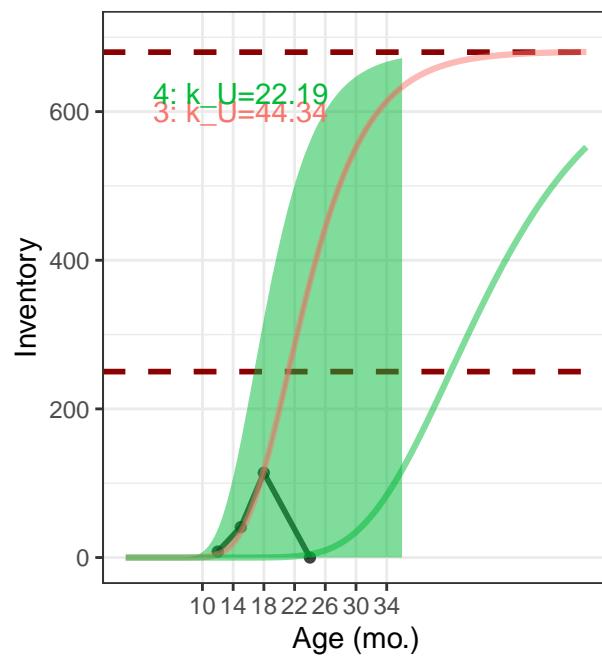
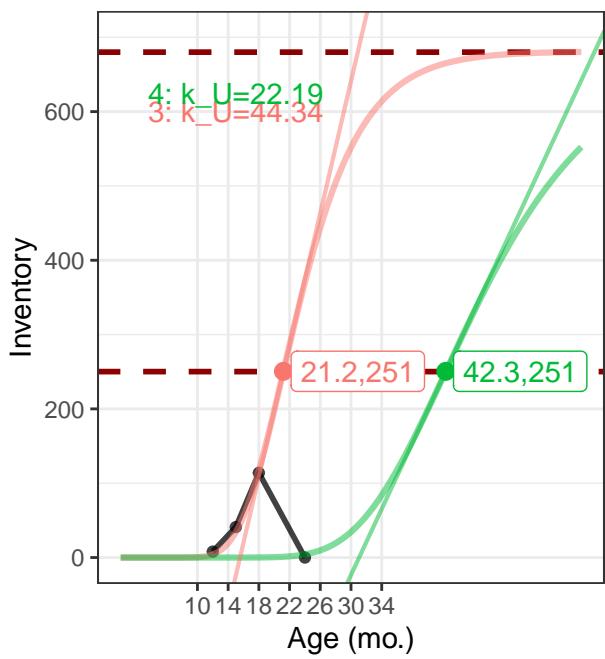
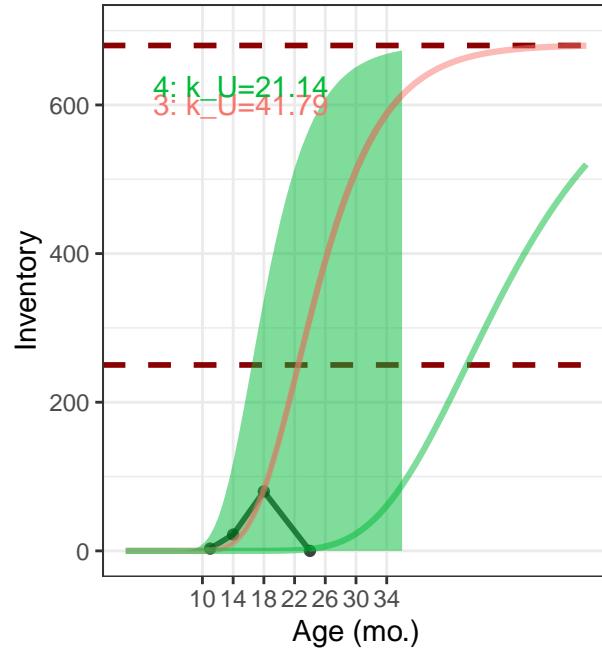
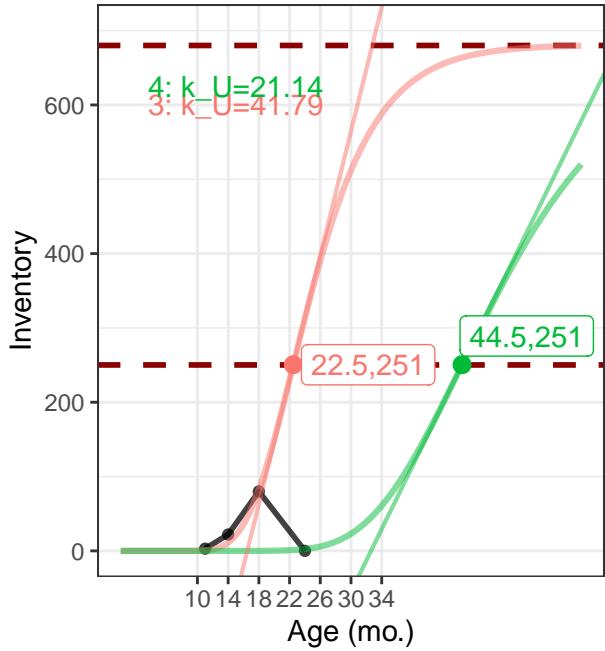
The `test` file contains data from 59 subjects who have at least four observations. Here, we're going to repeat the same steps as above and plot all 51 to examine how consistent curves are with other data.

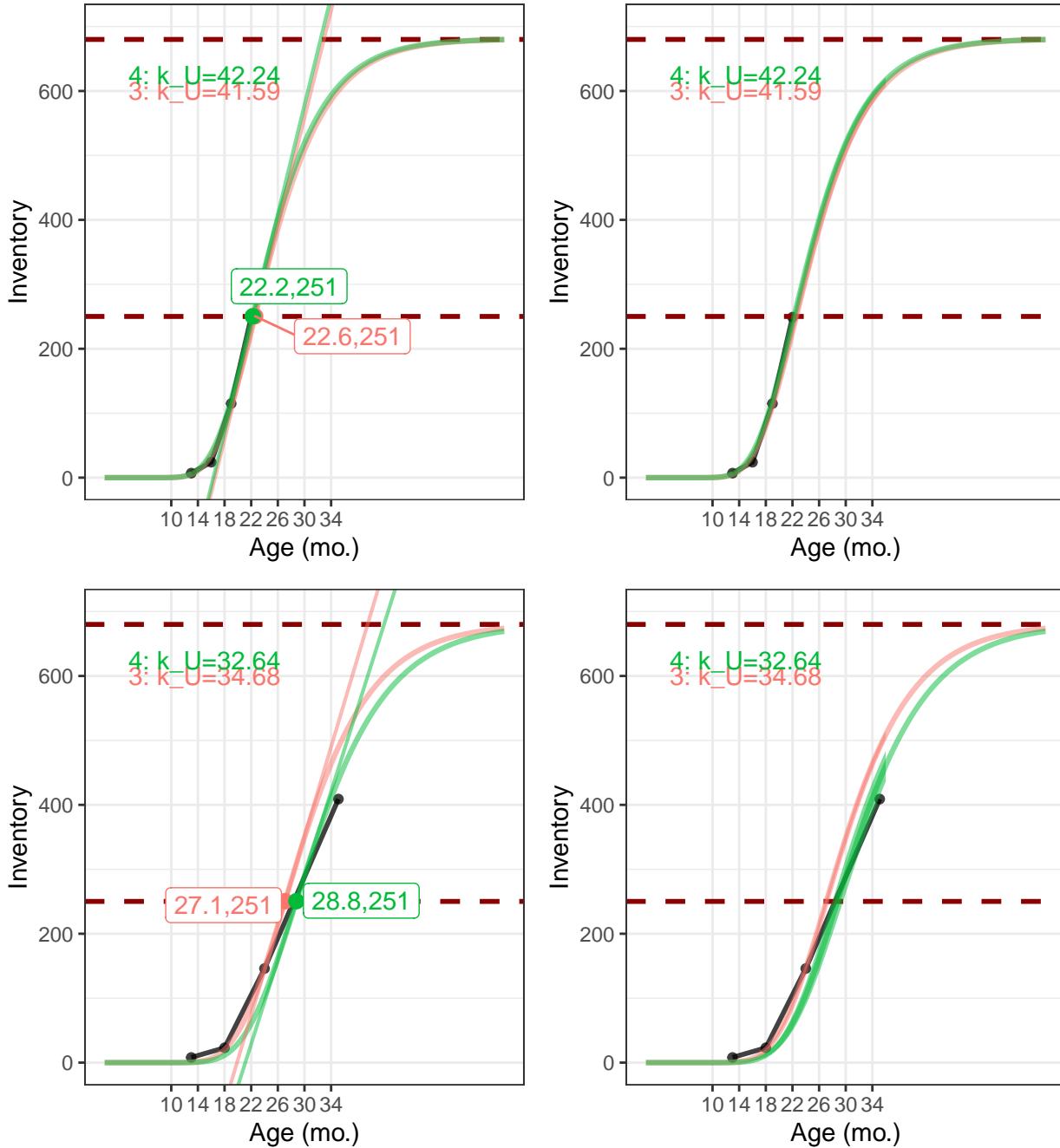
I'm going to invisibly fit the curves here, with tangent lines and $\{T_i, W_i\}$ coordinates on the left and SE envelopes on the right.

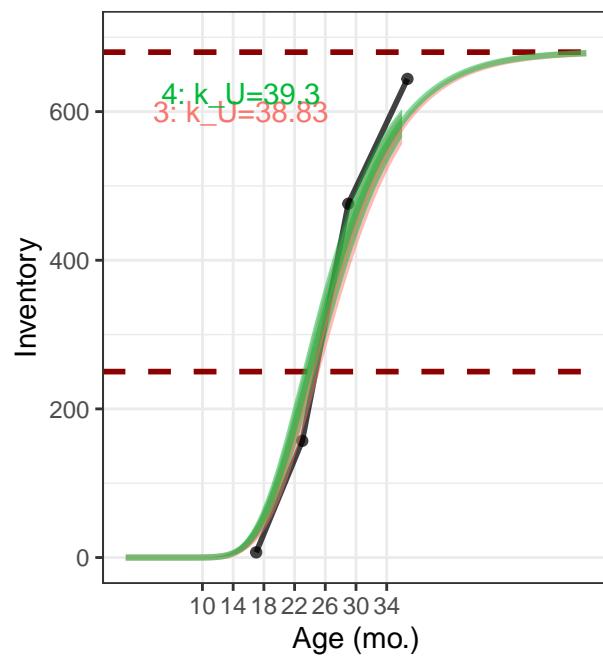
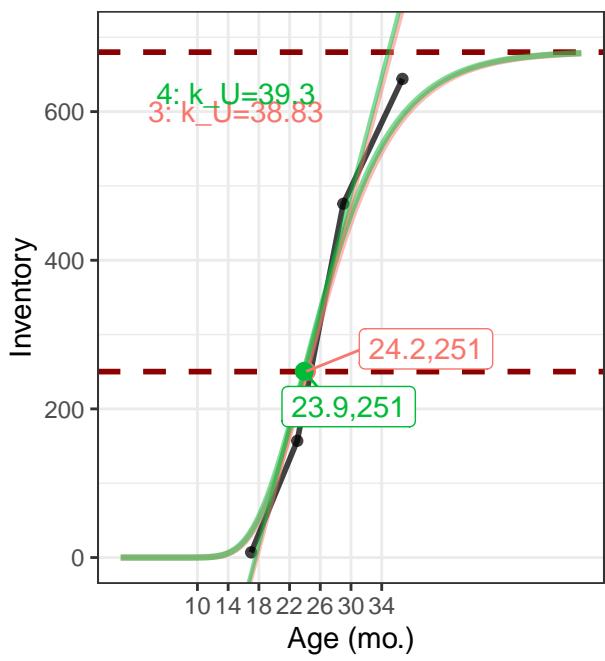
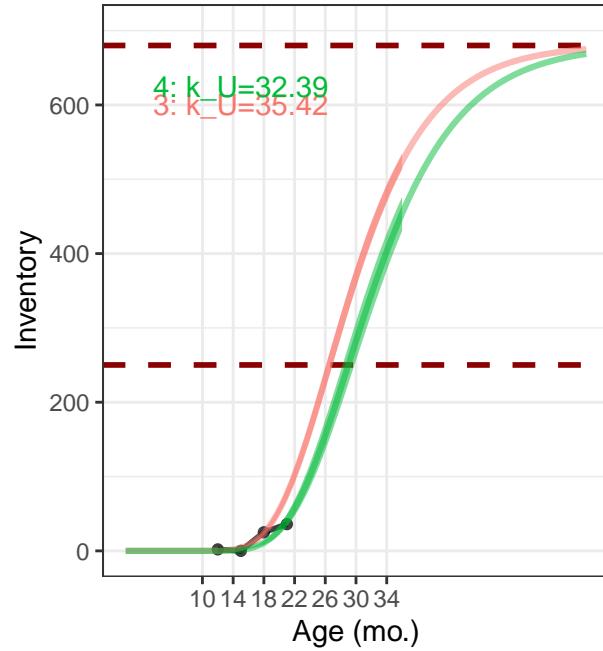
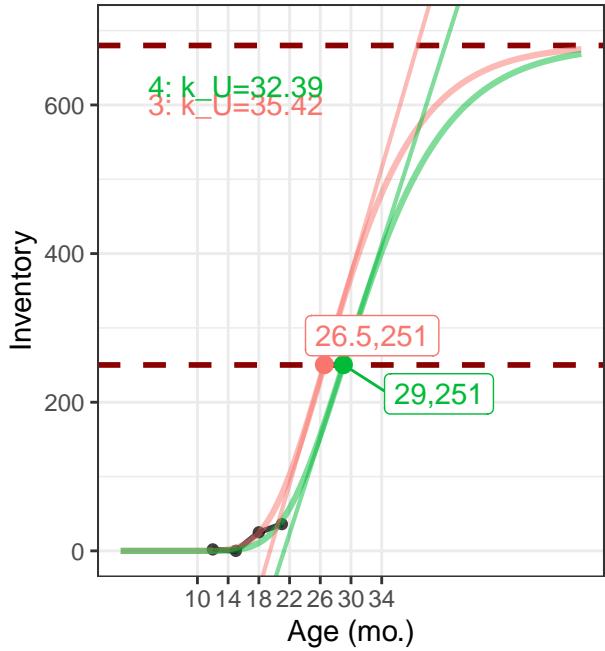


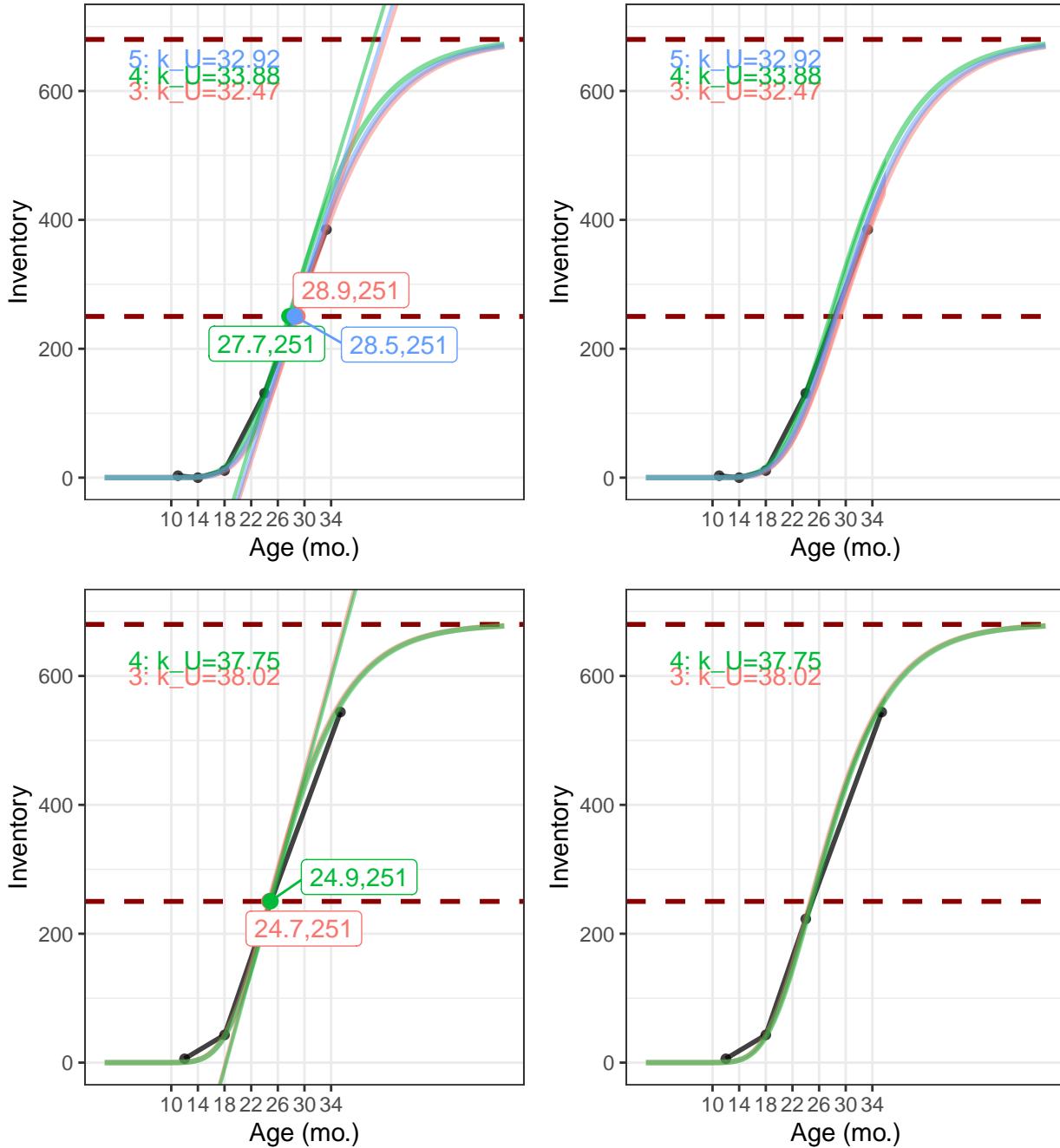


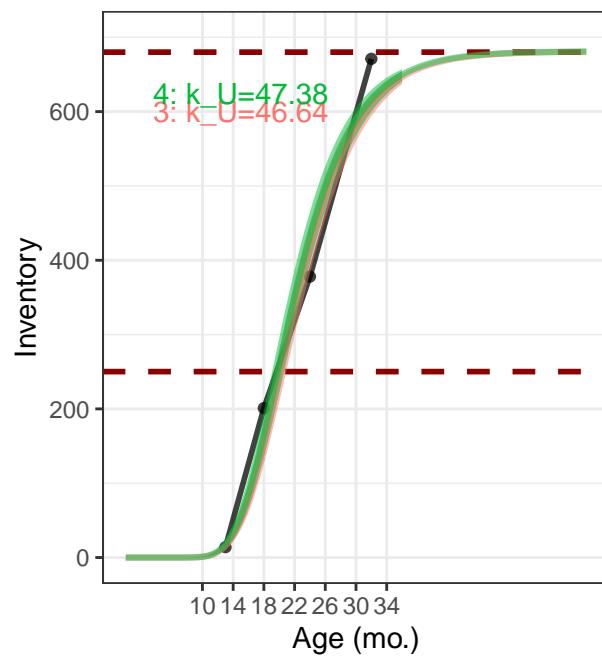
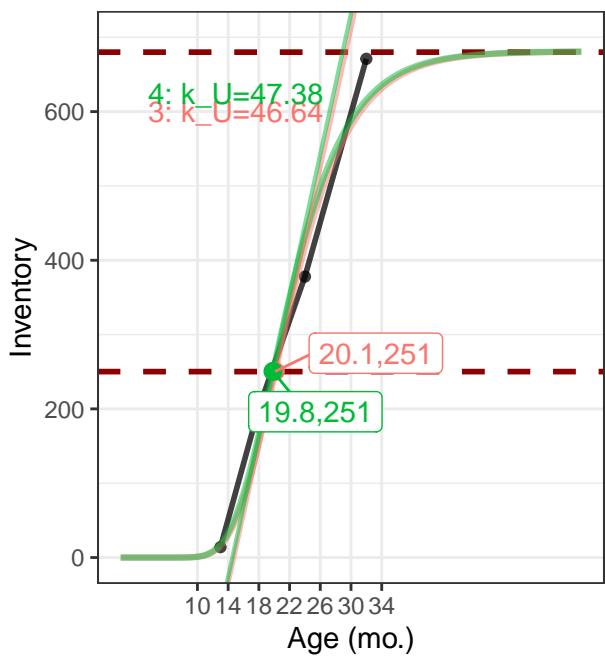
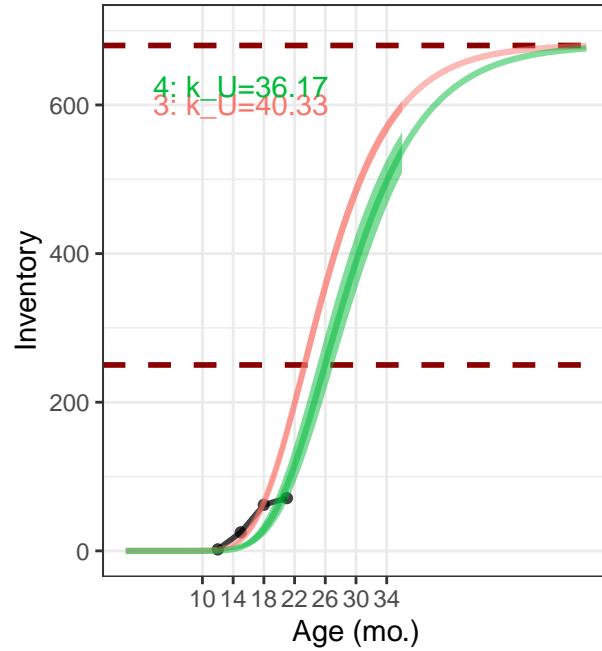
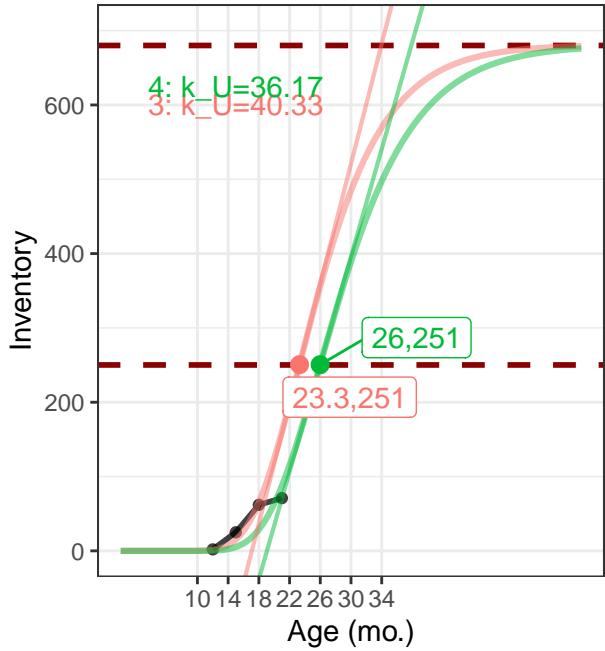


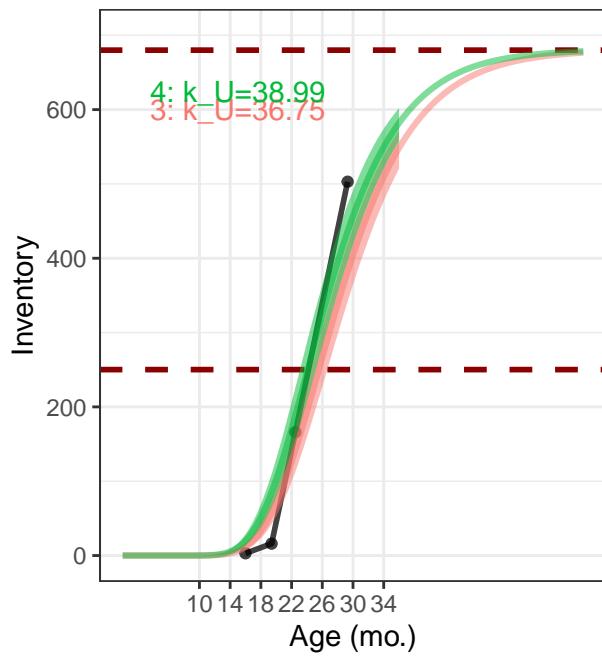
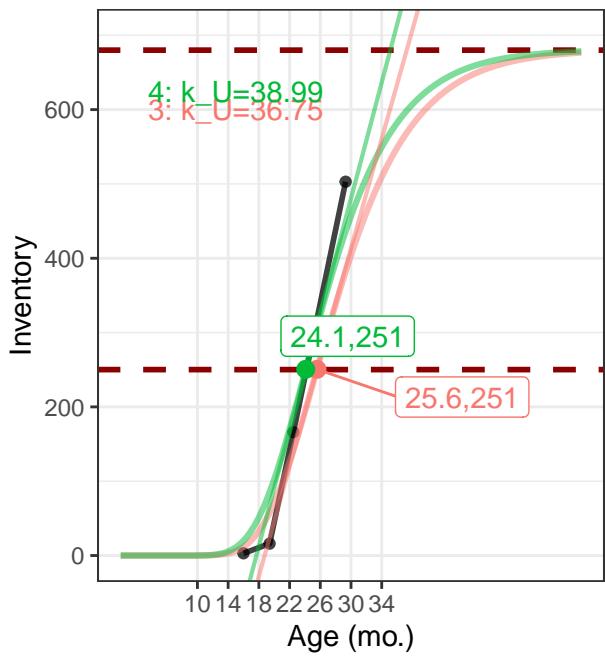
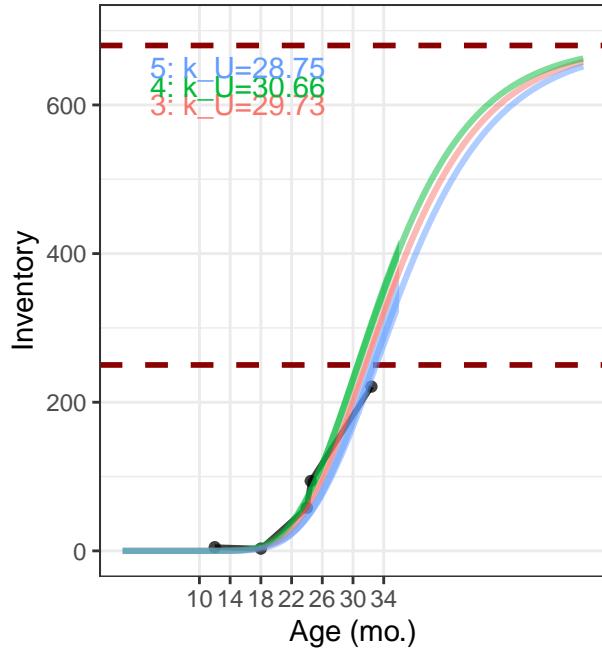
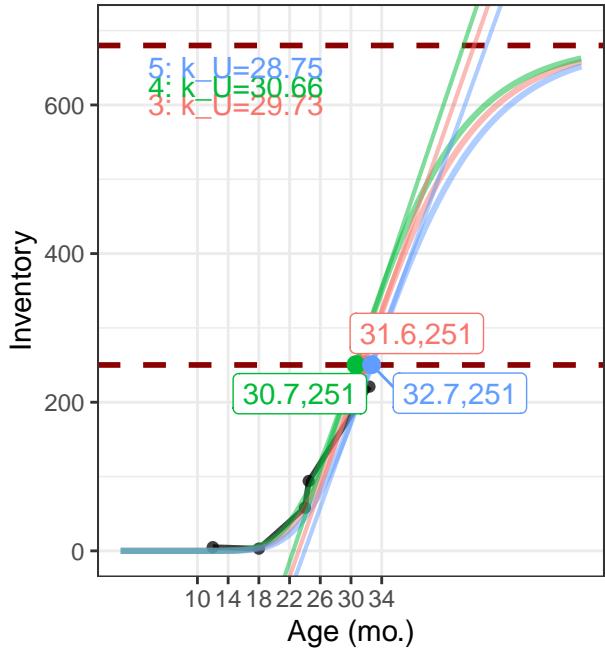


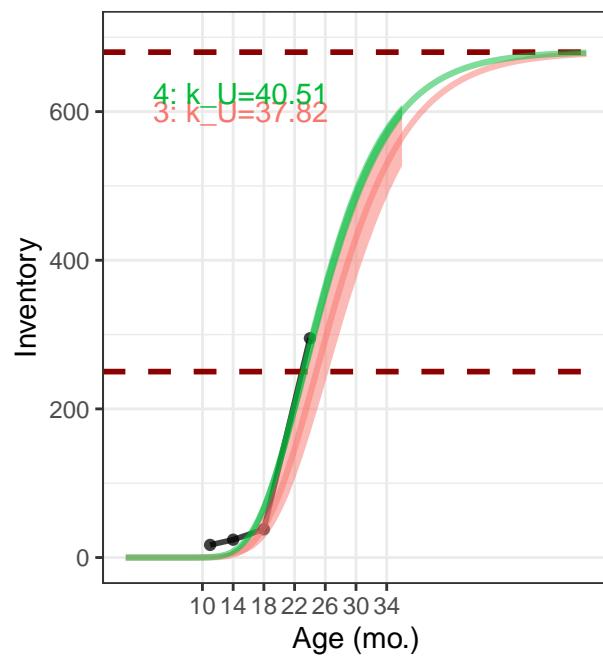
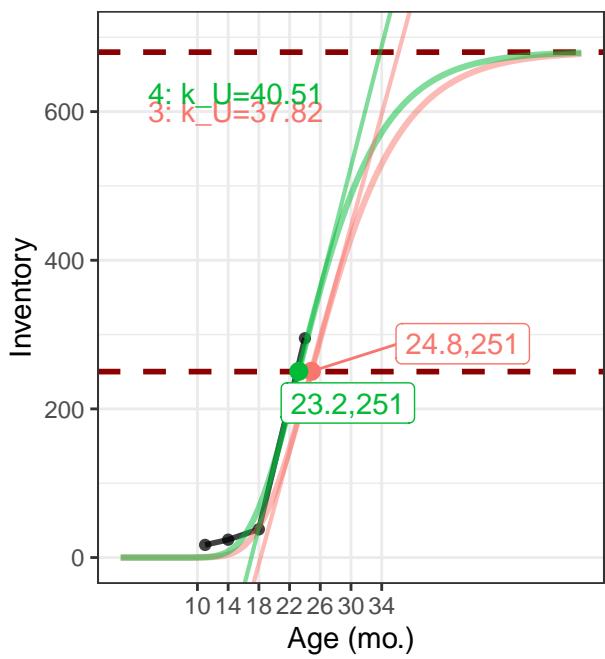
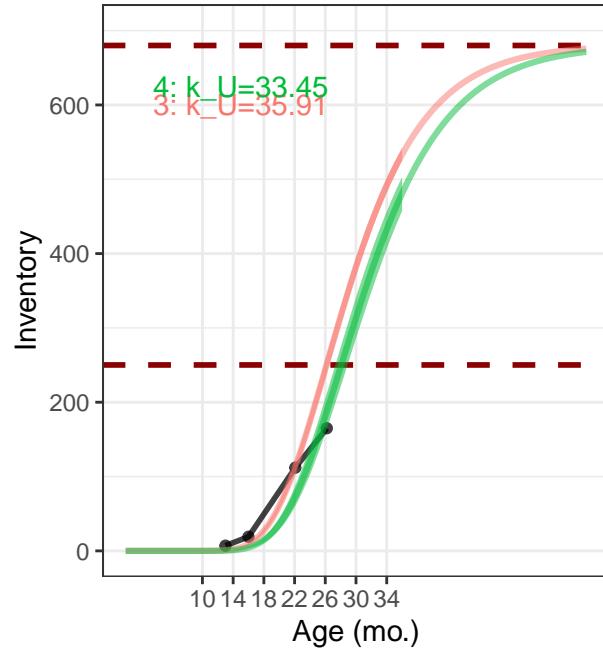
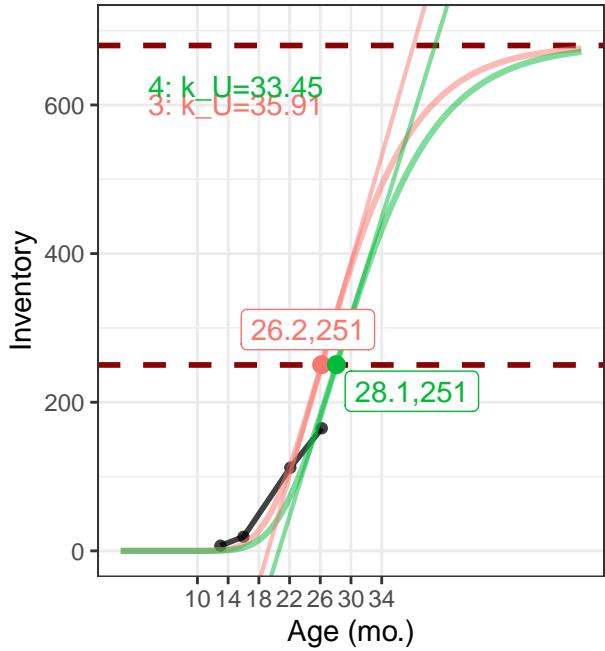


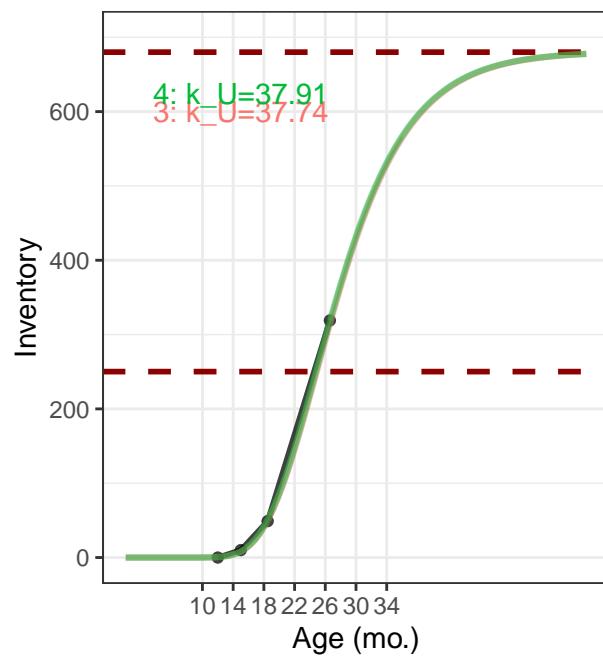
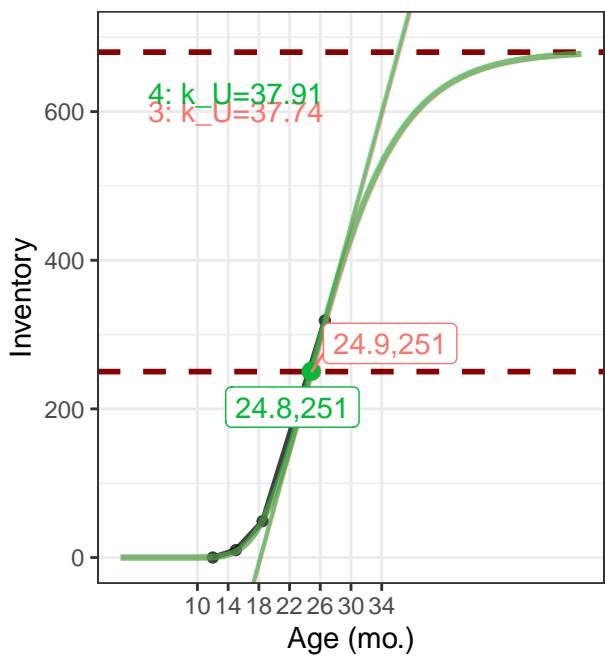
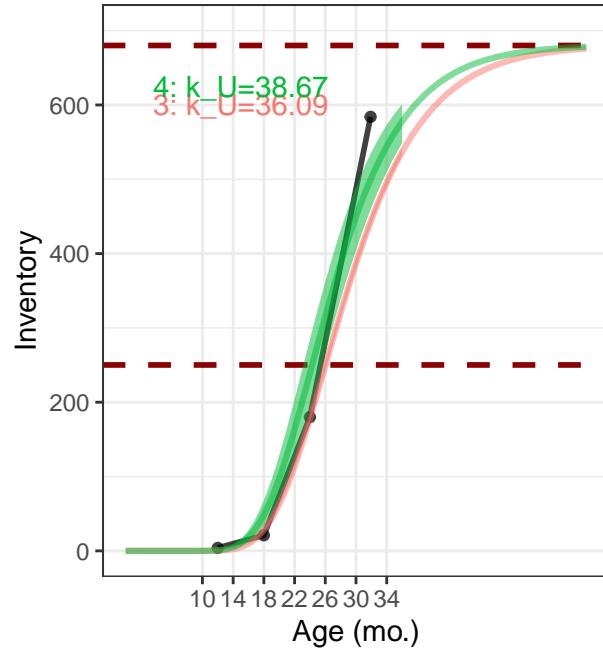
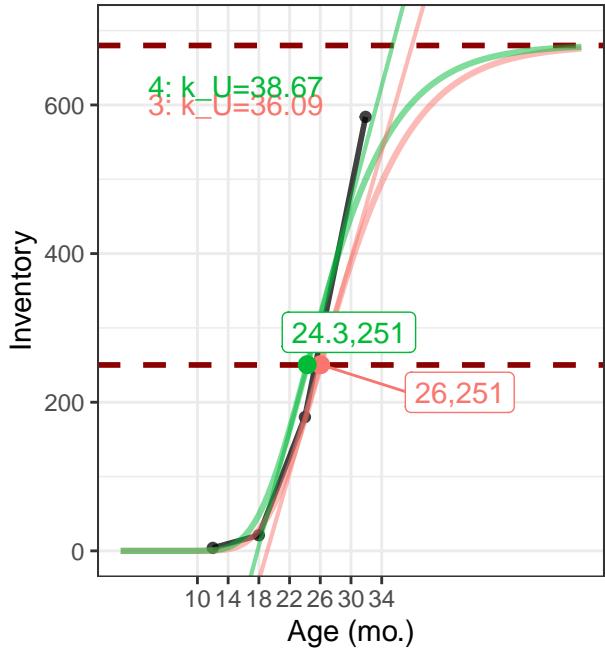


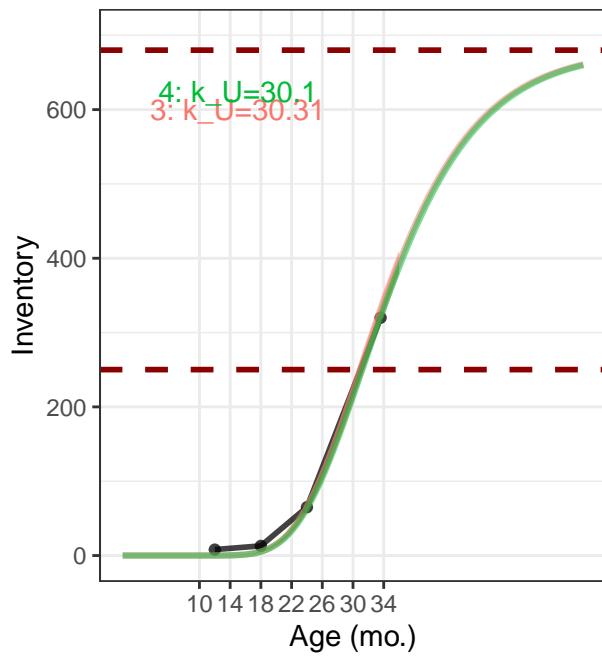
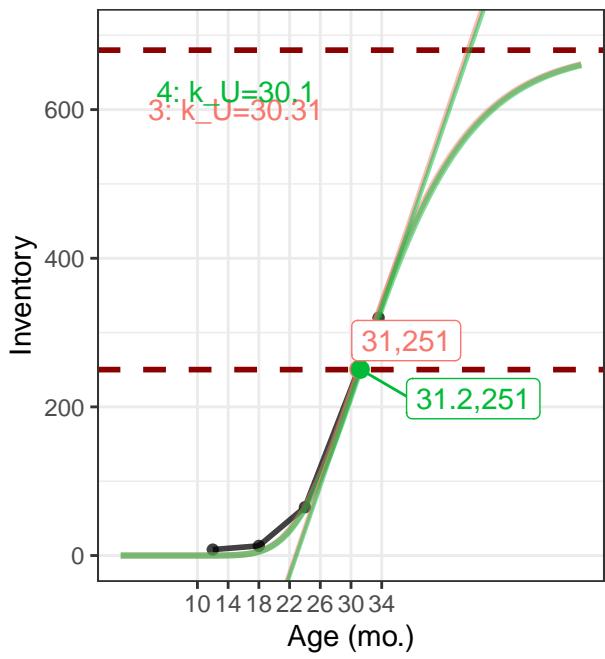
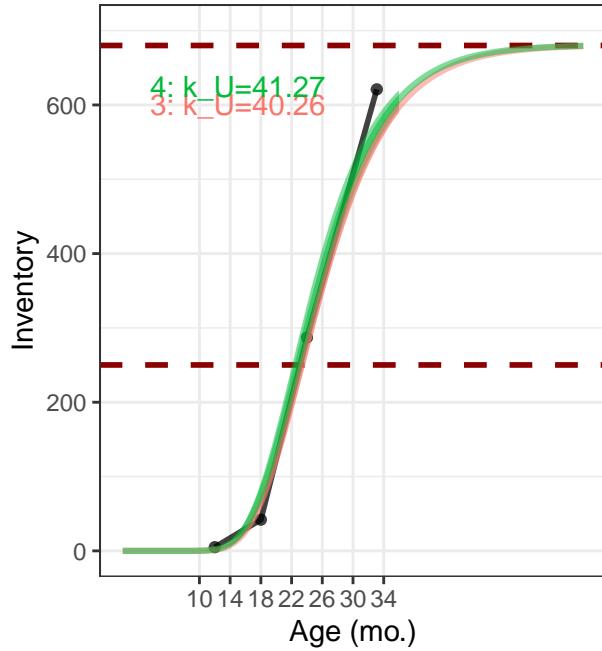
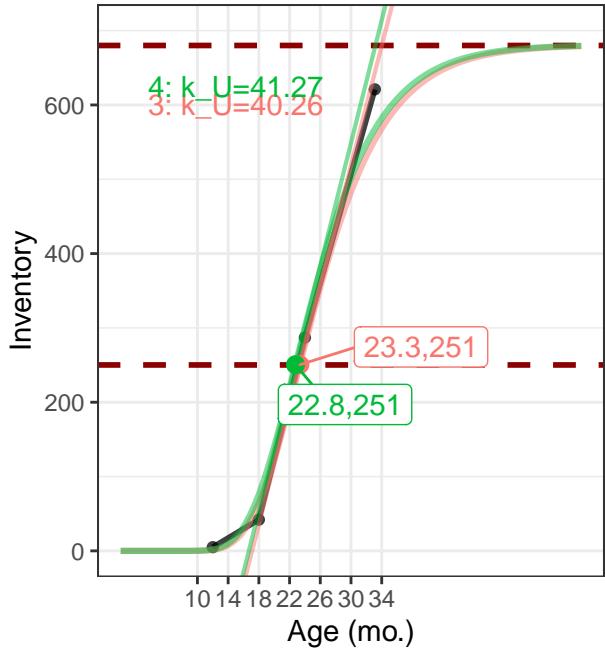


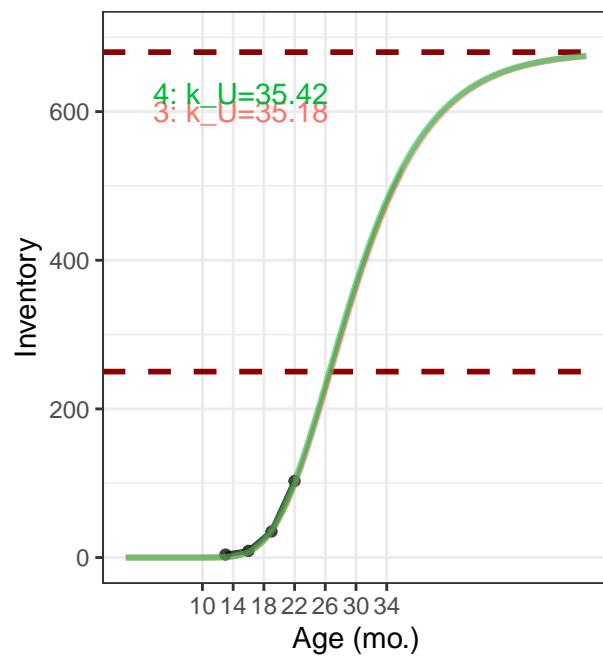
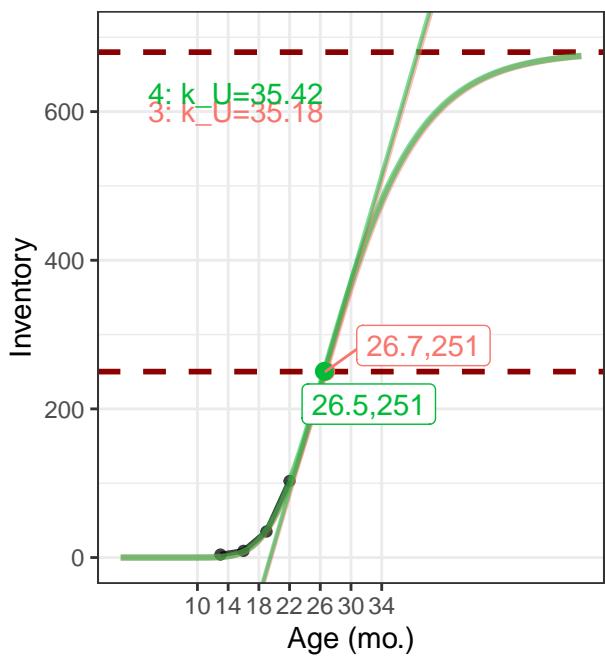
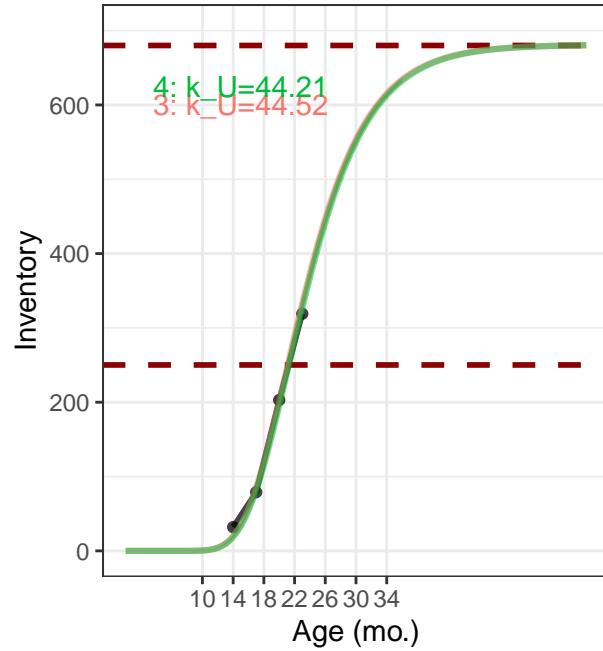
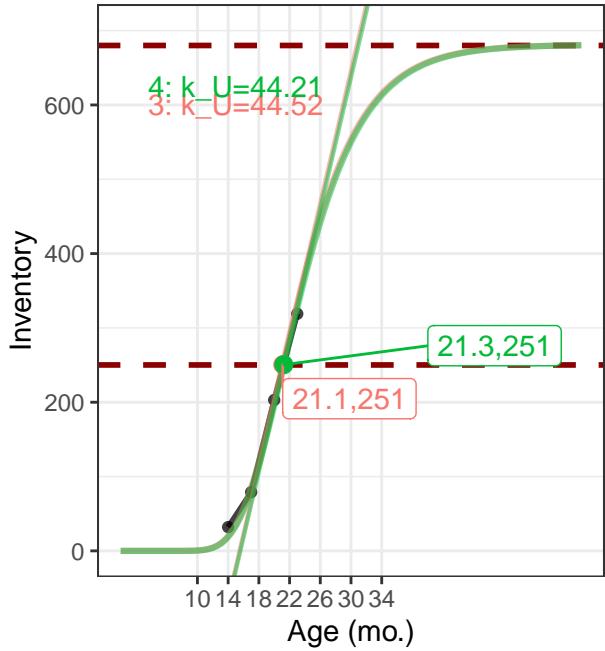


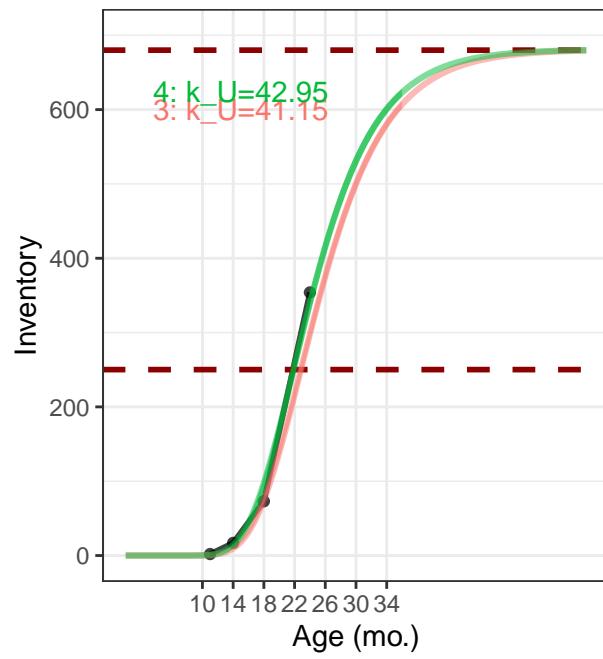
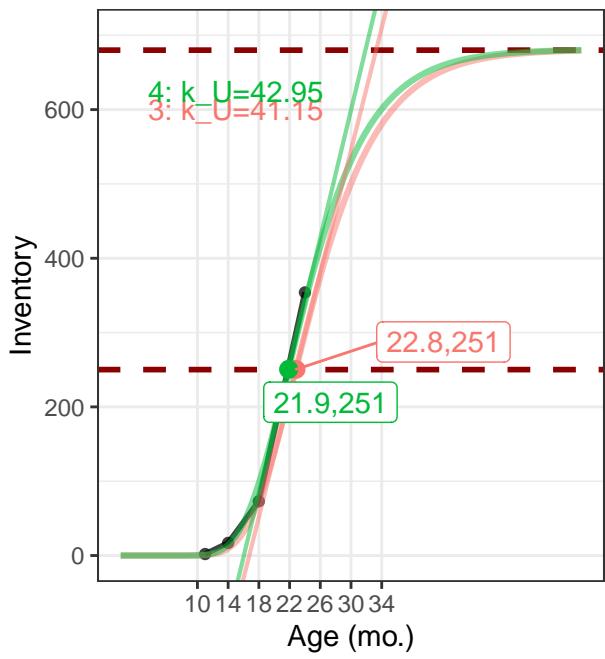
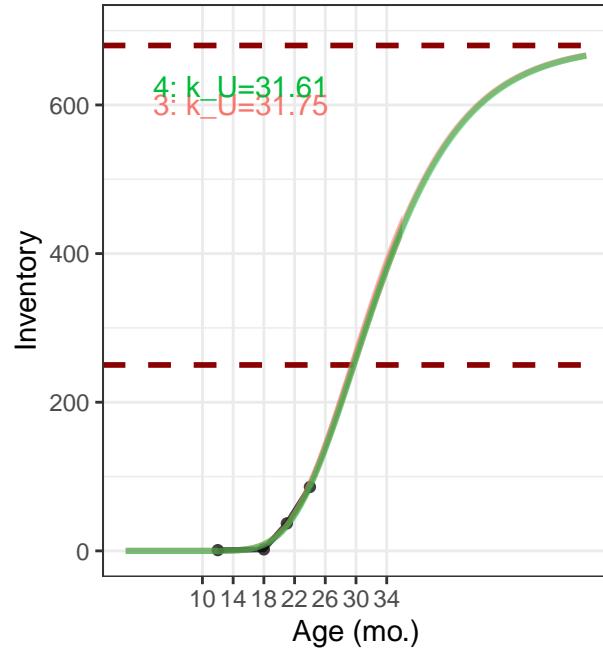
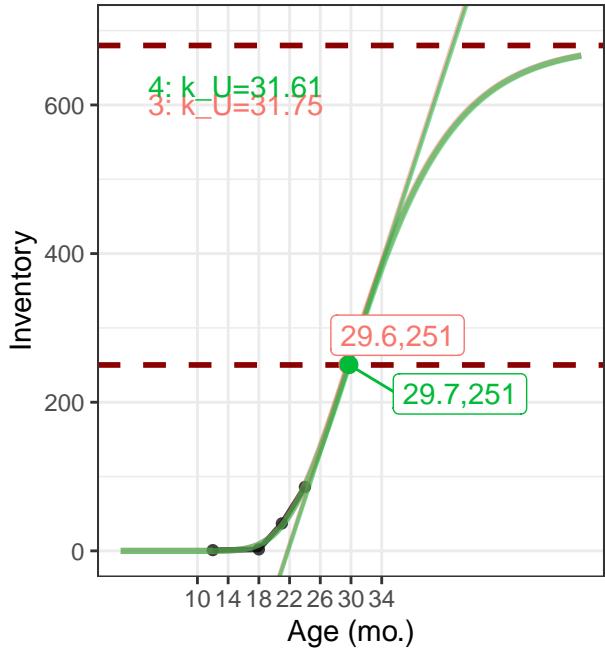


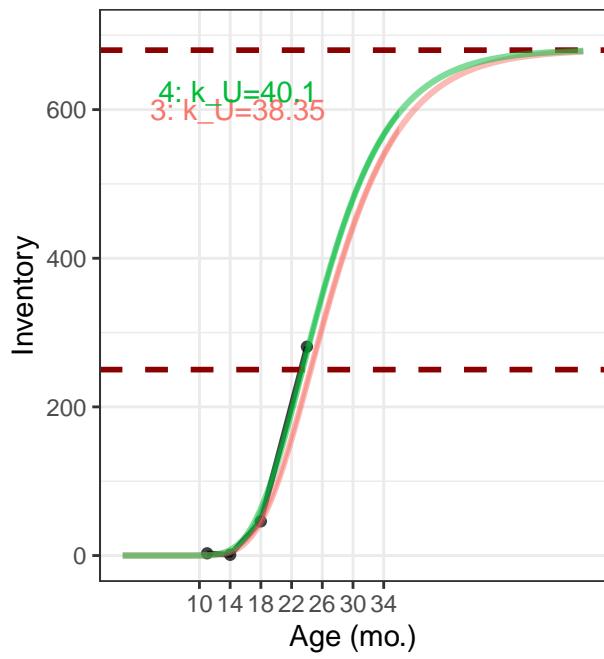
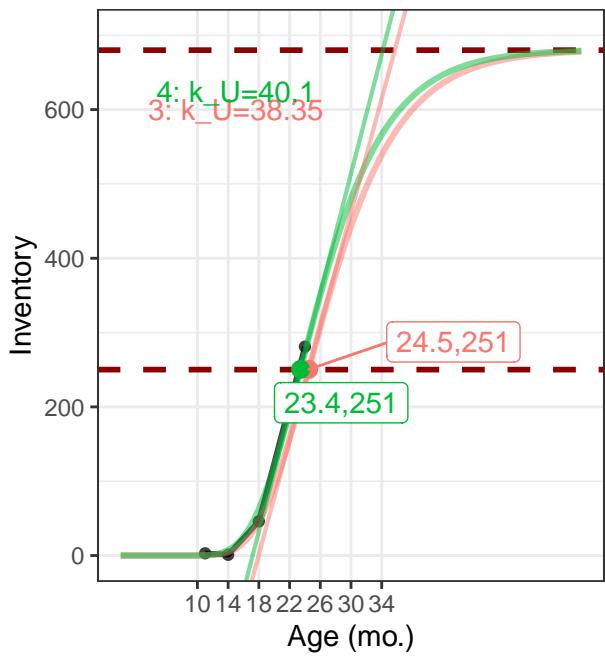
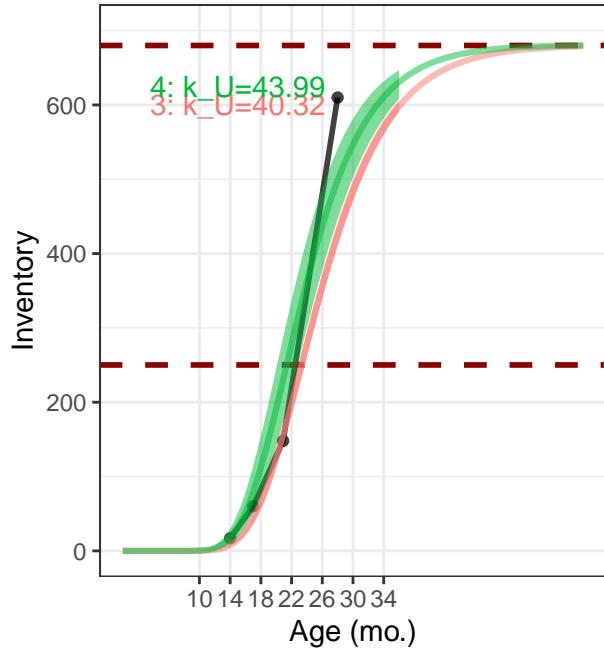
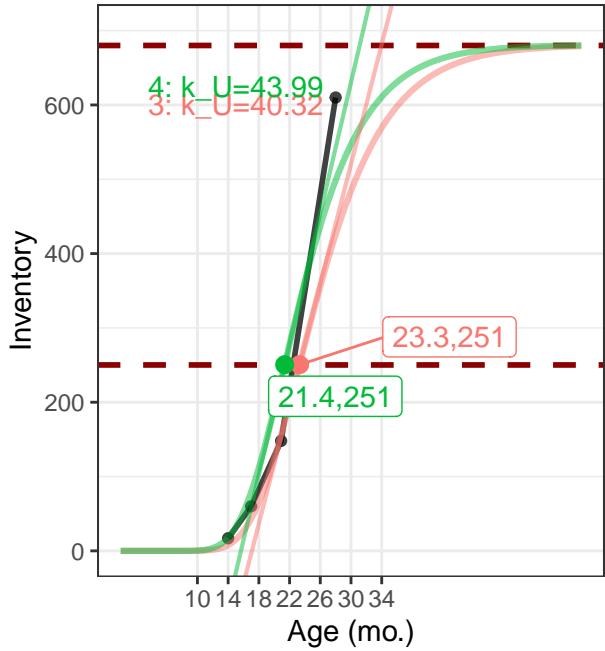


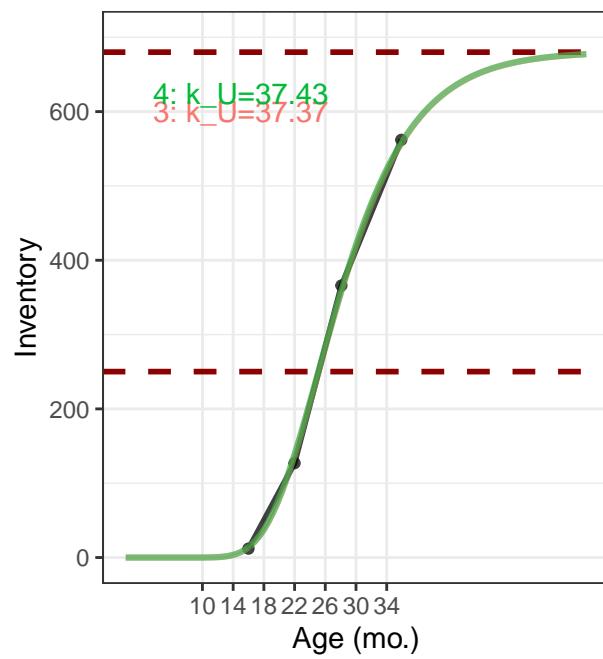
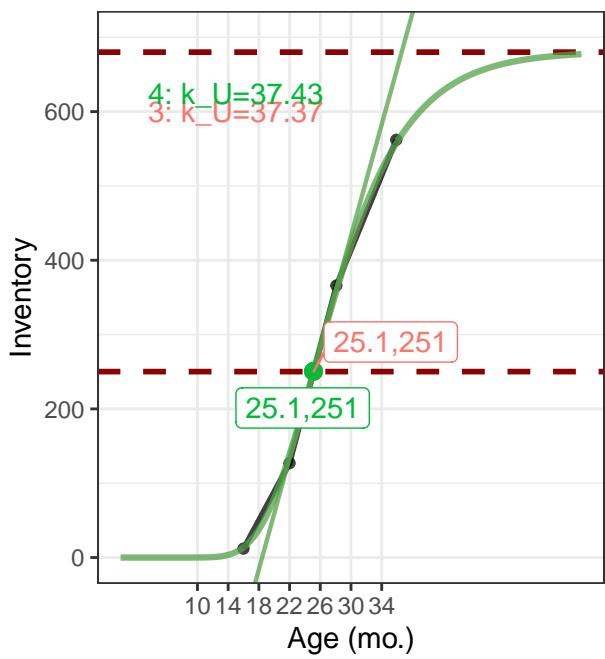
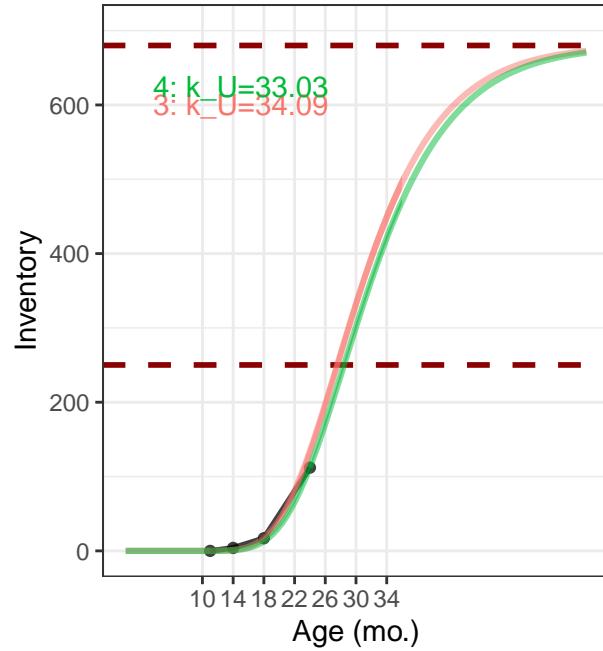
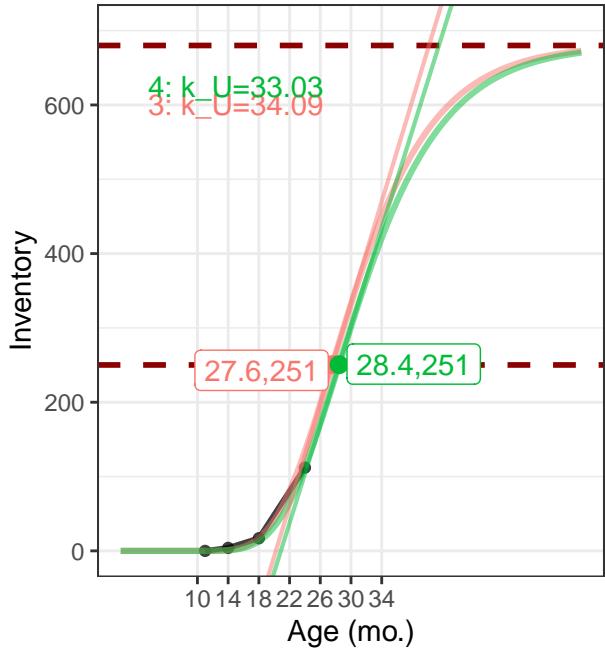


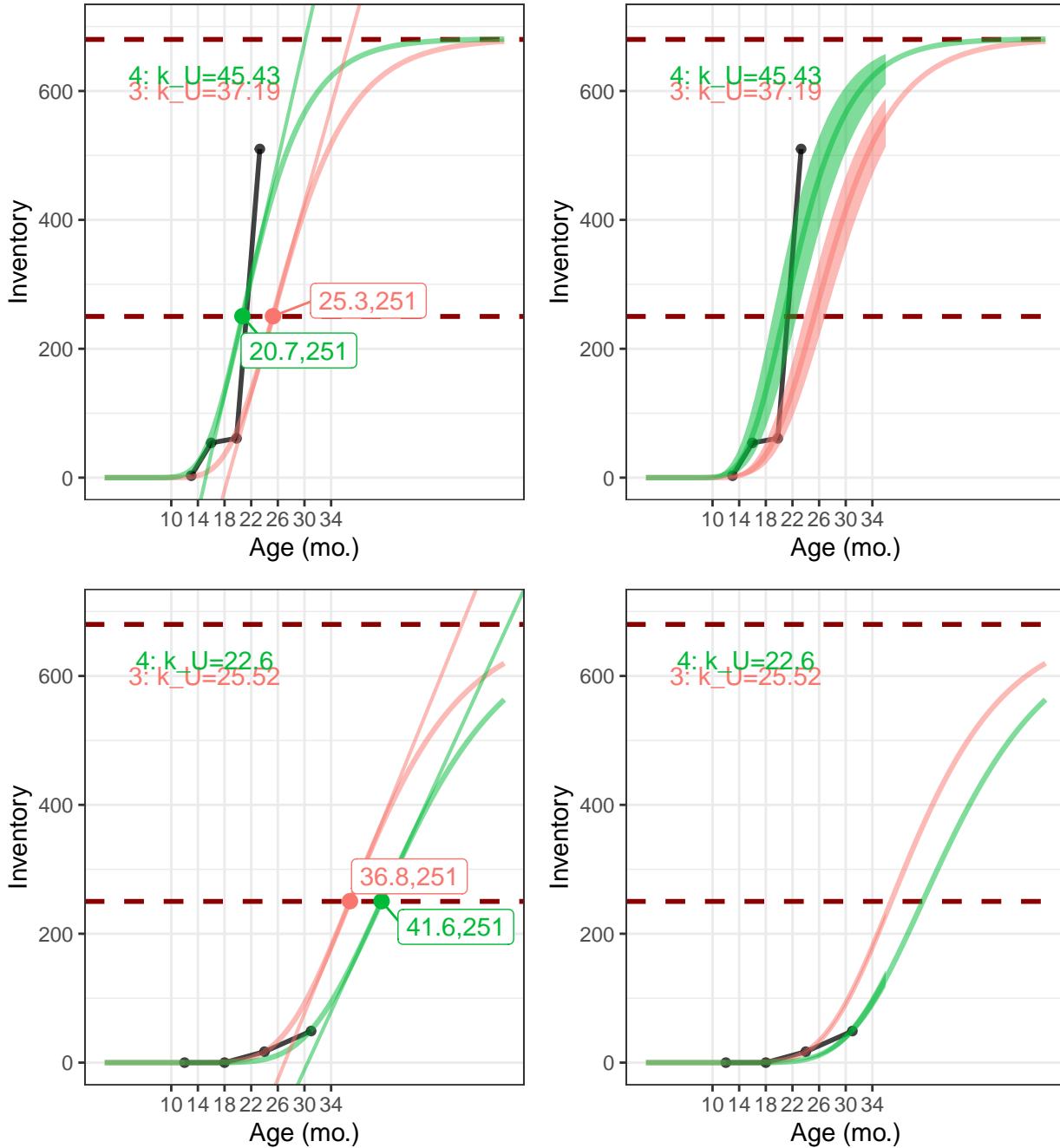


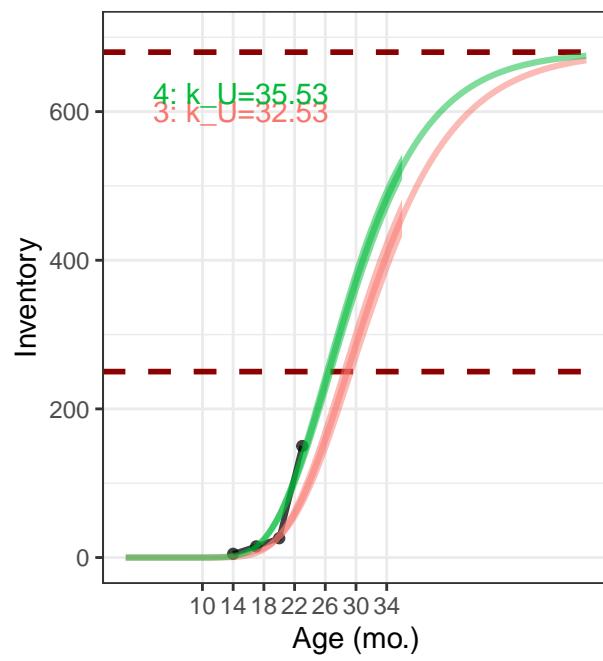
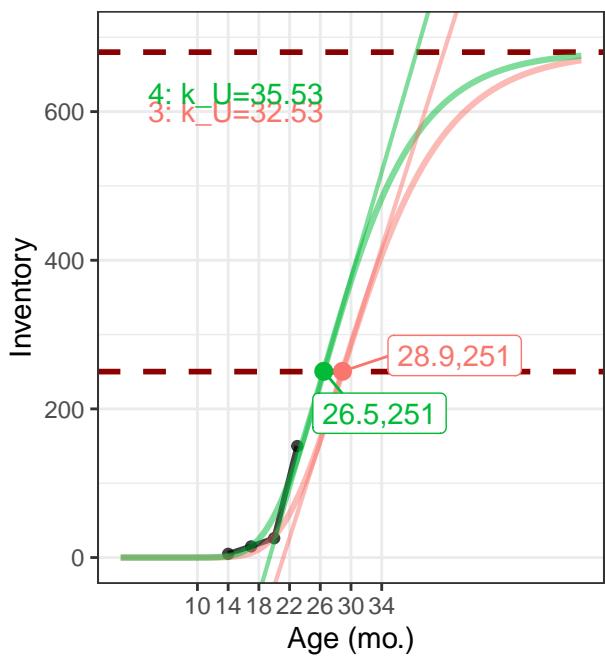
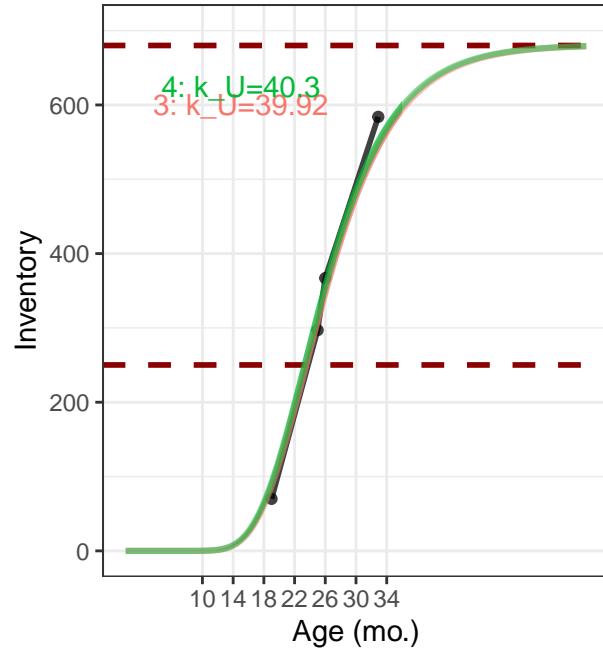
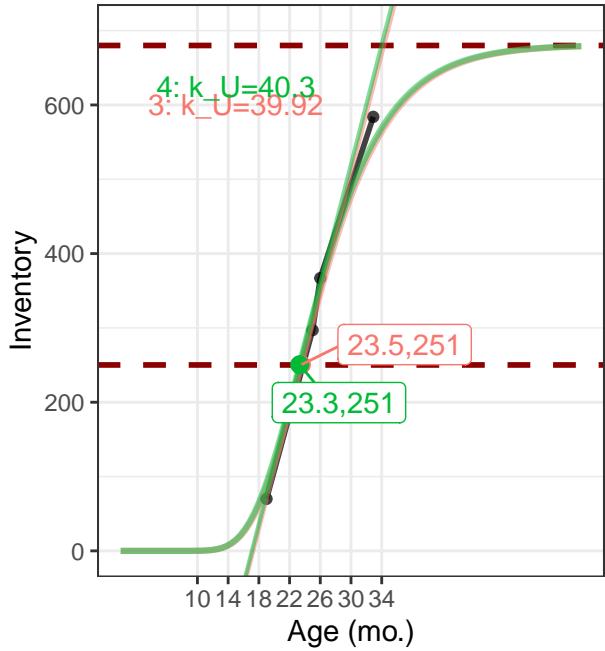


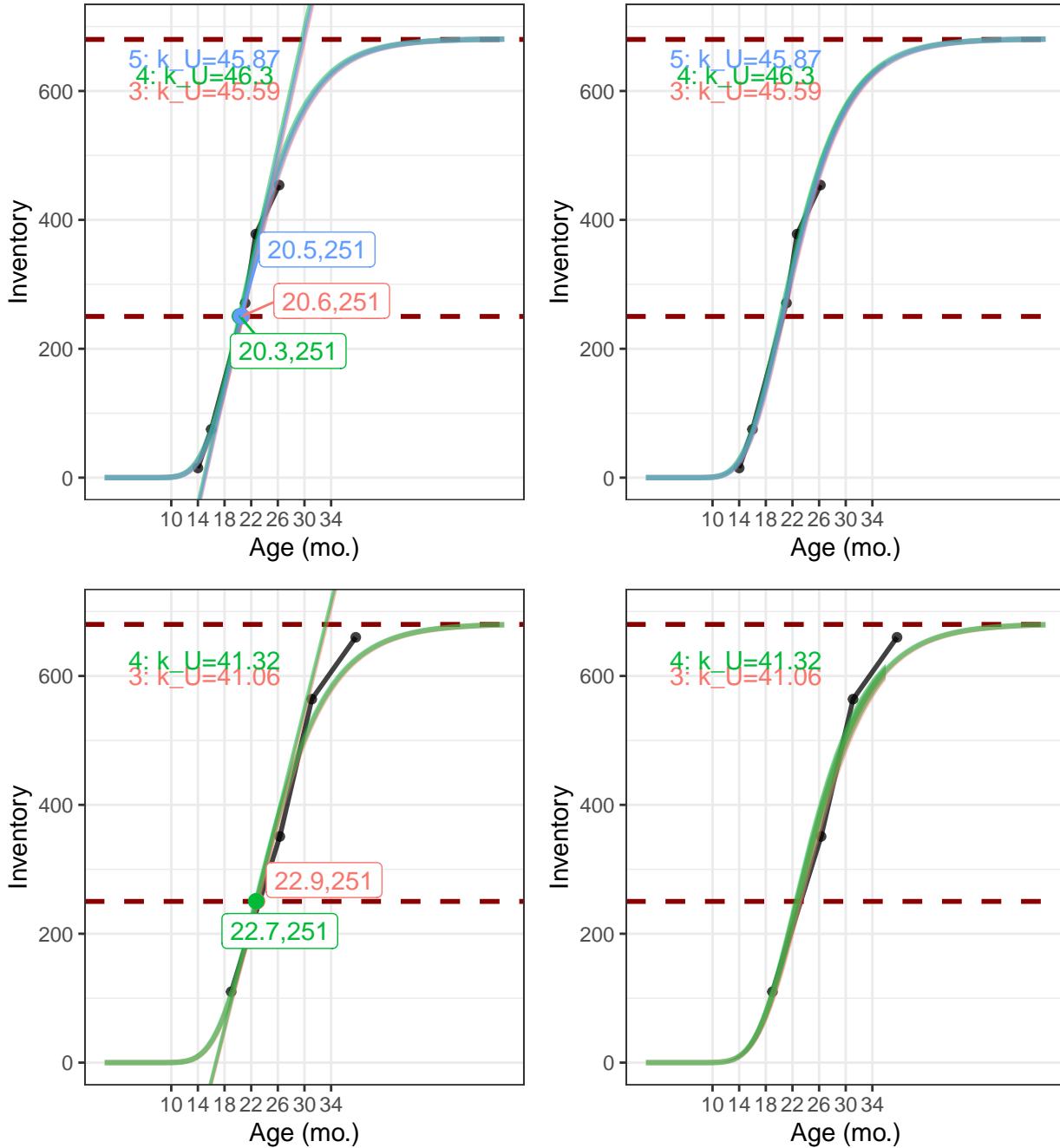


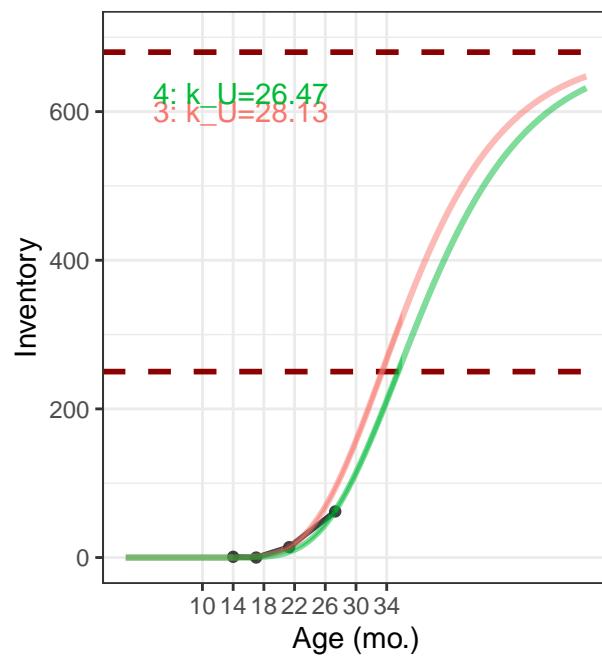
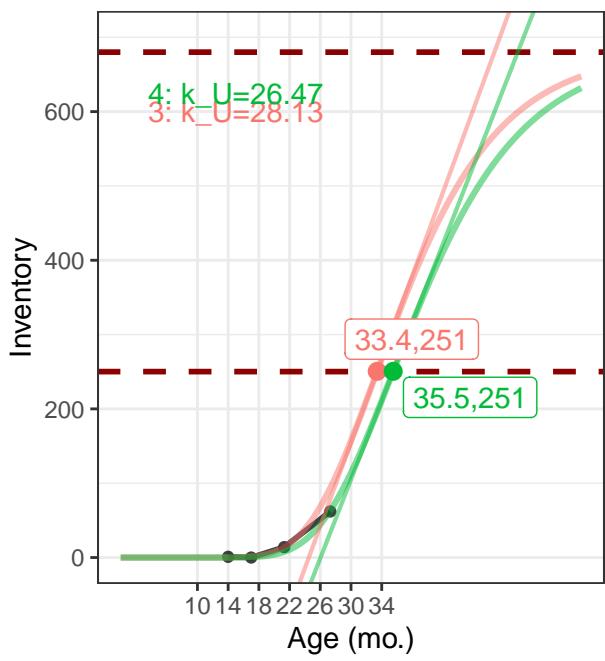
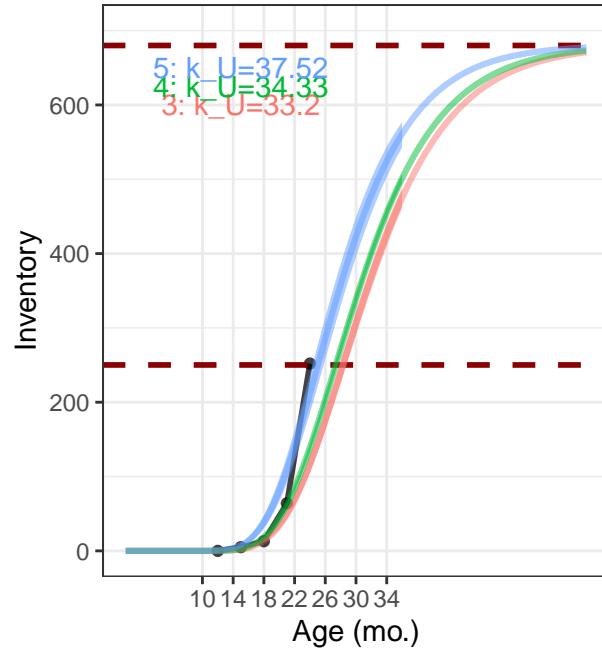
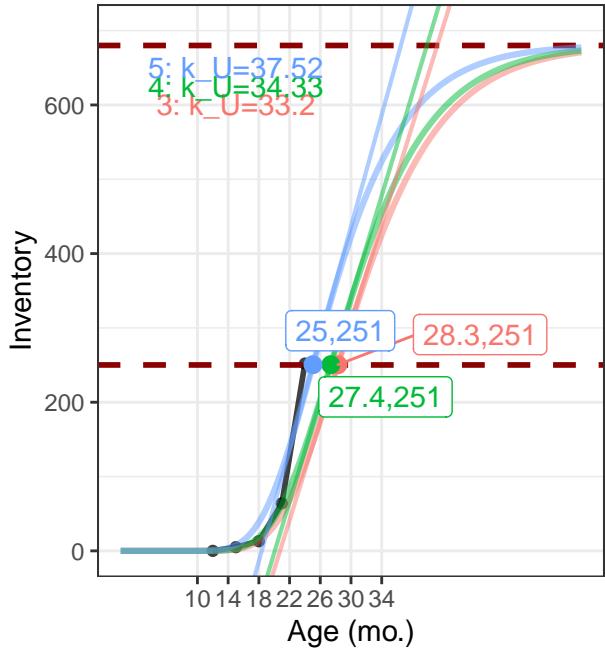


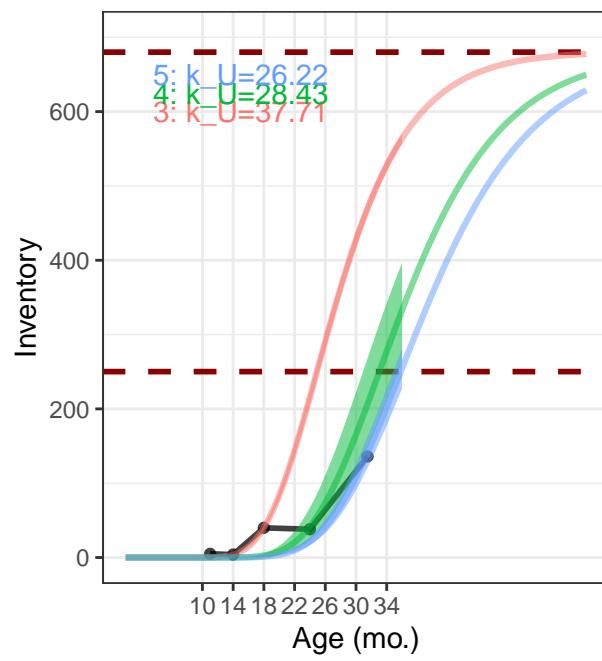
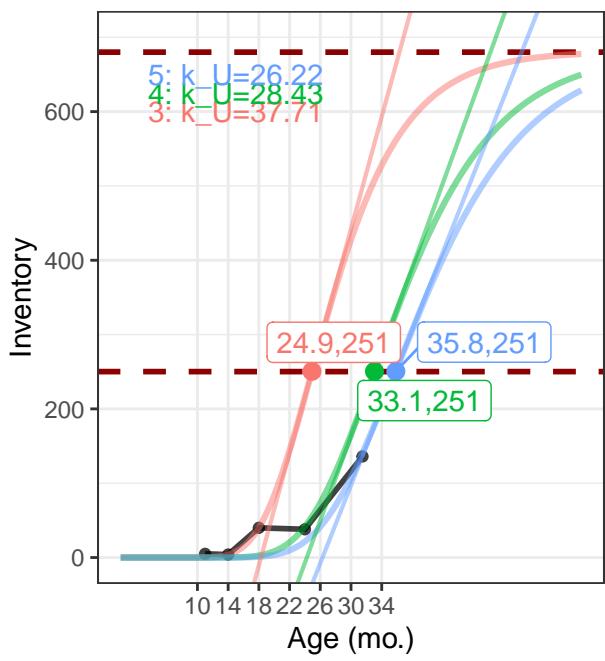
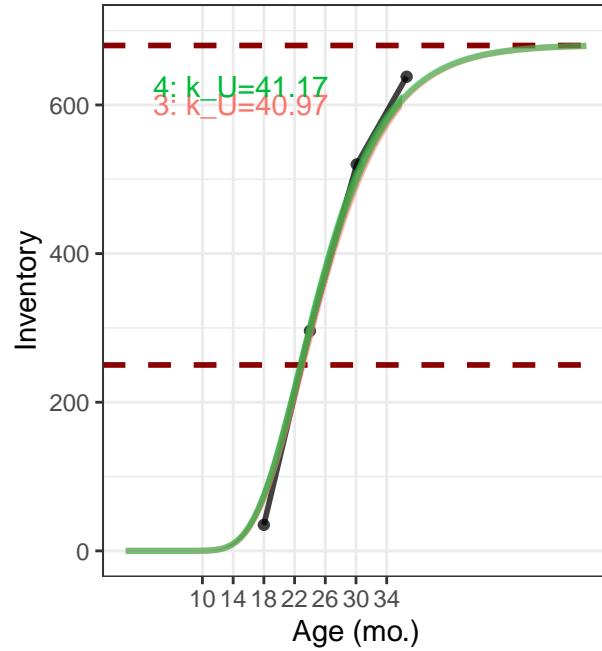
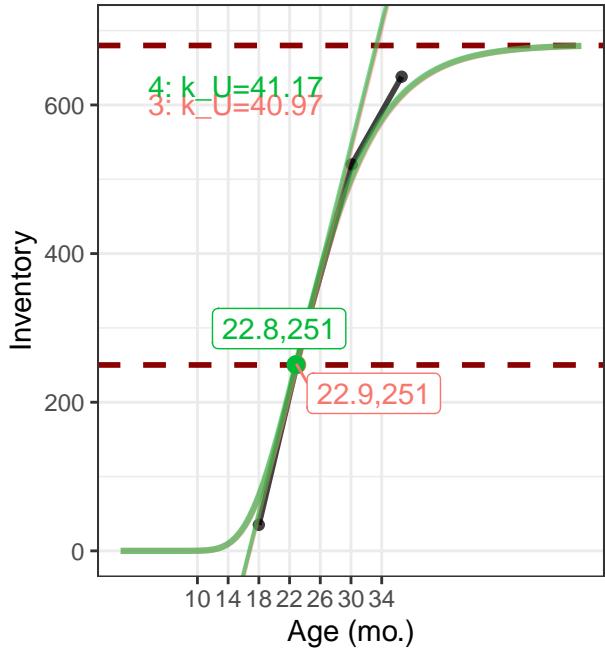


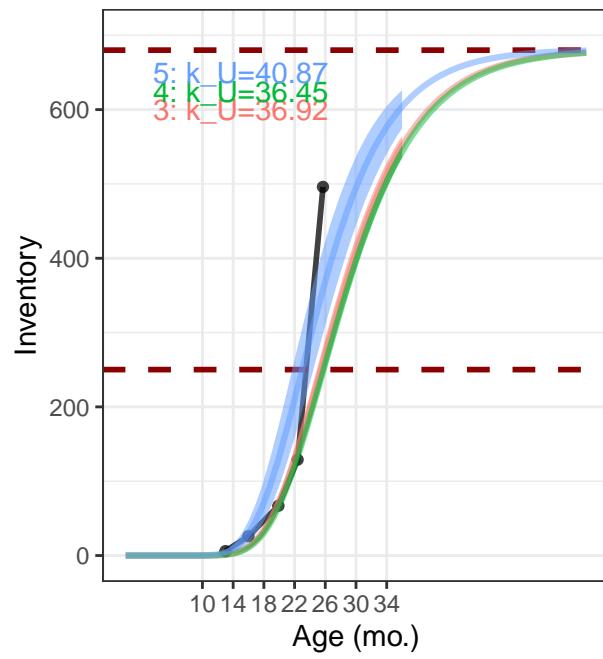
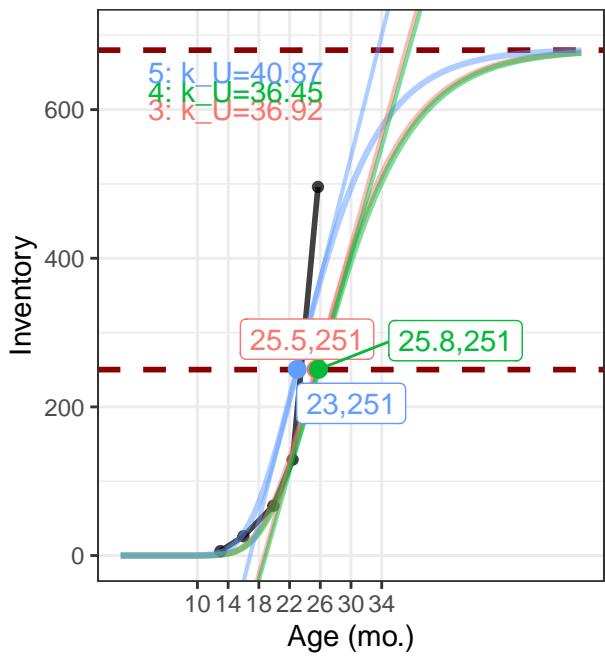
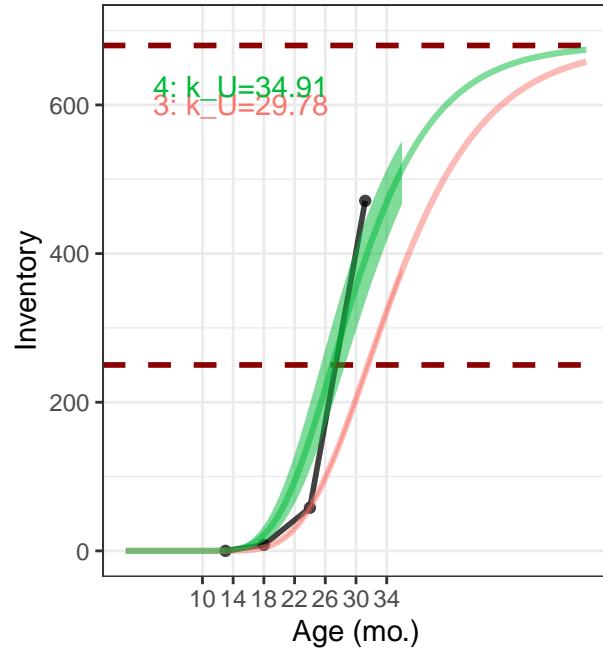
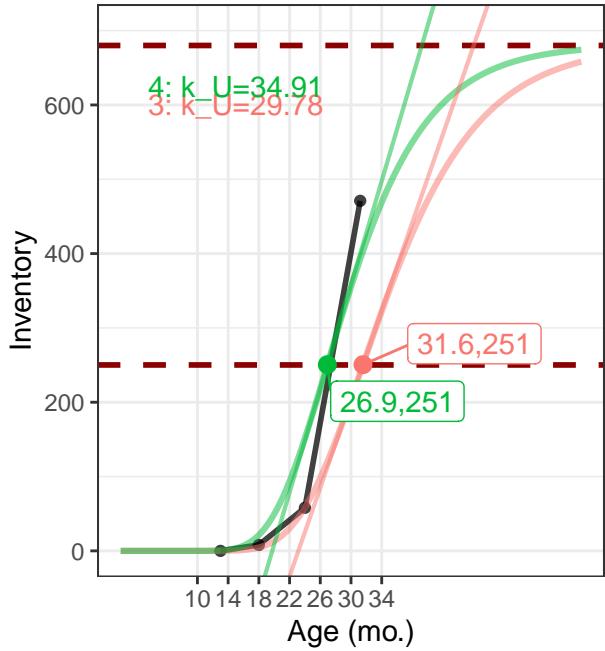


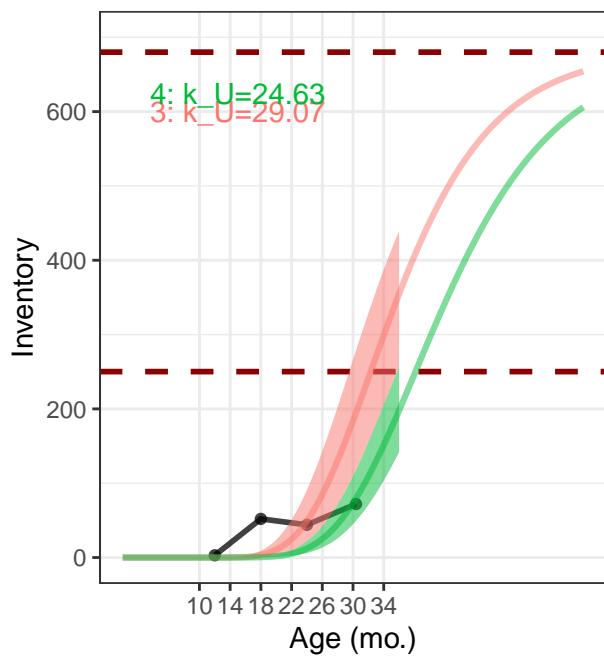
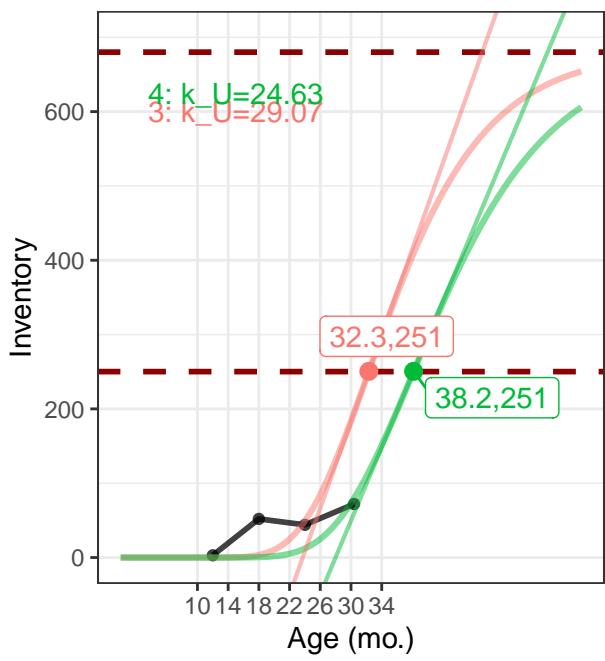
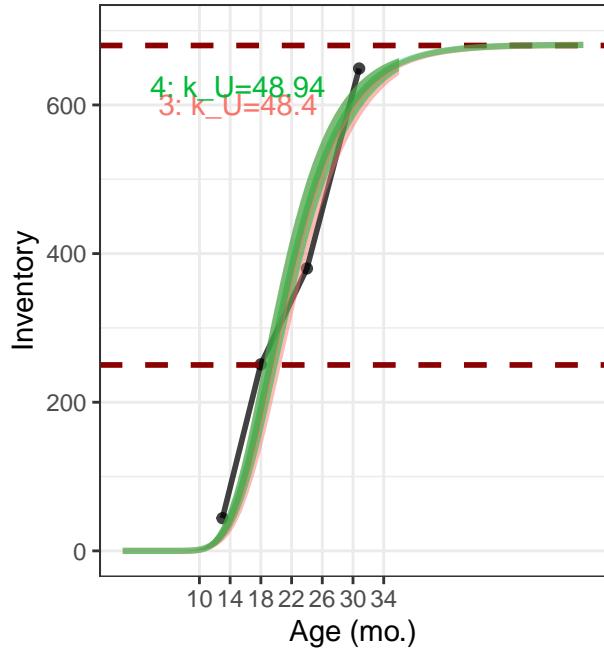
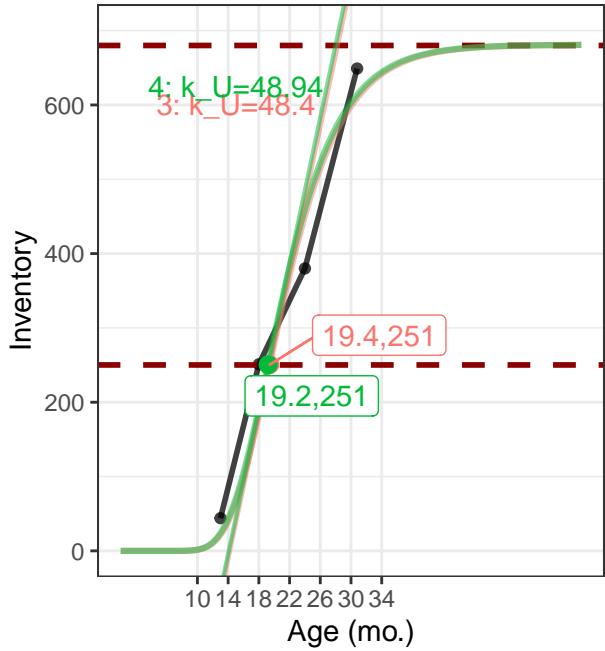


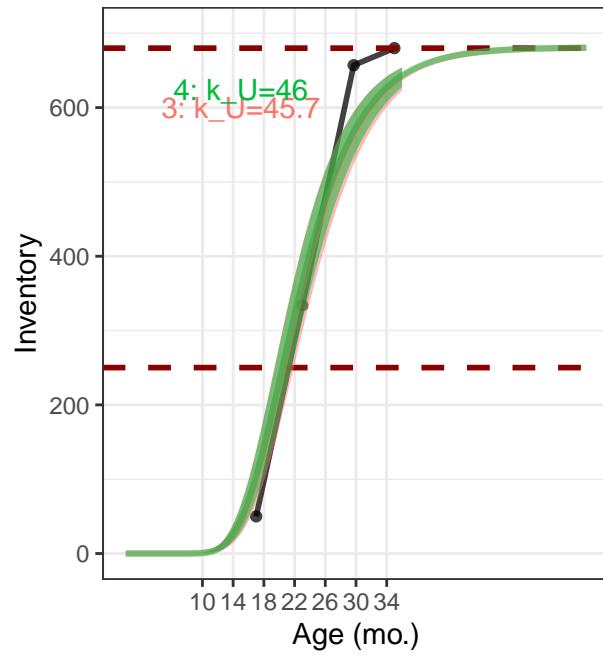
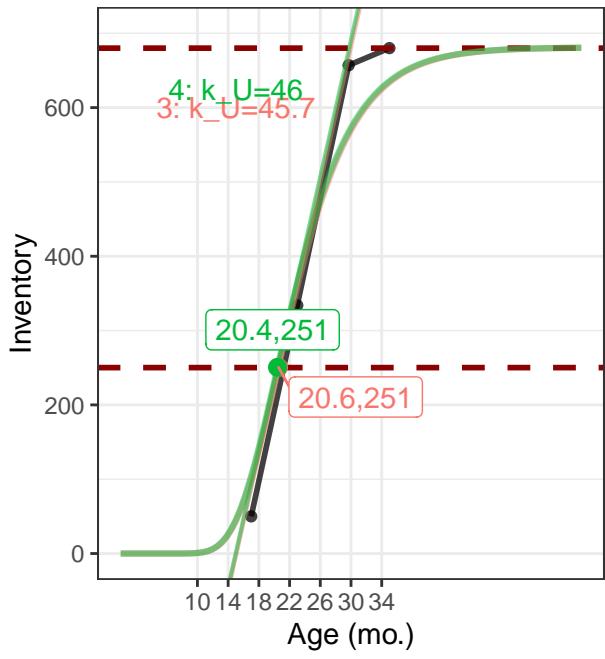
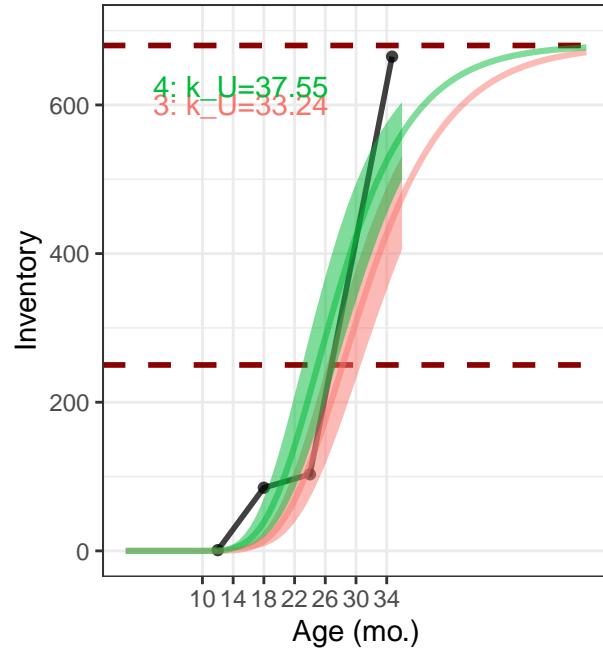
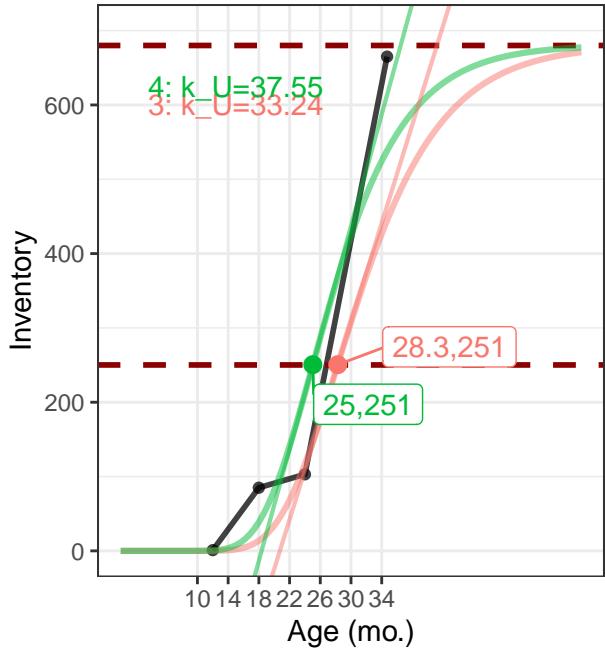


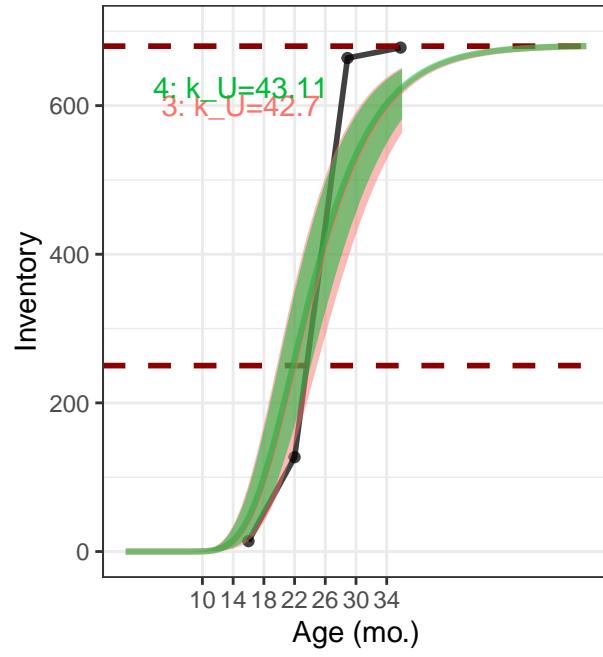
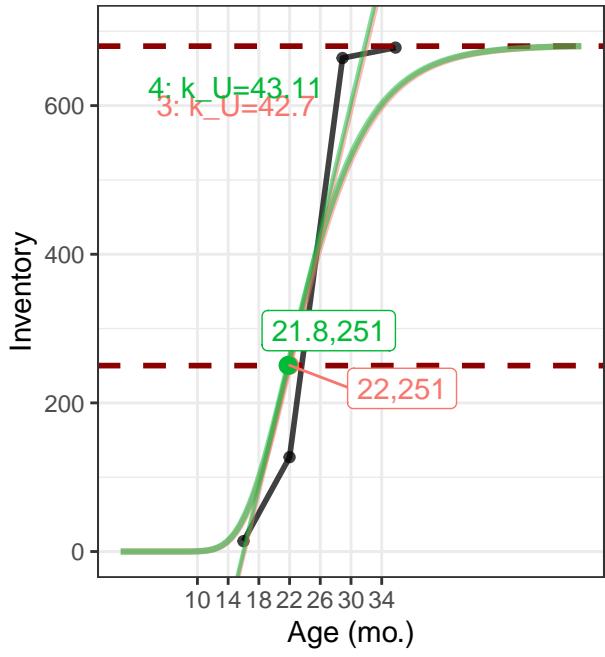
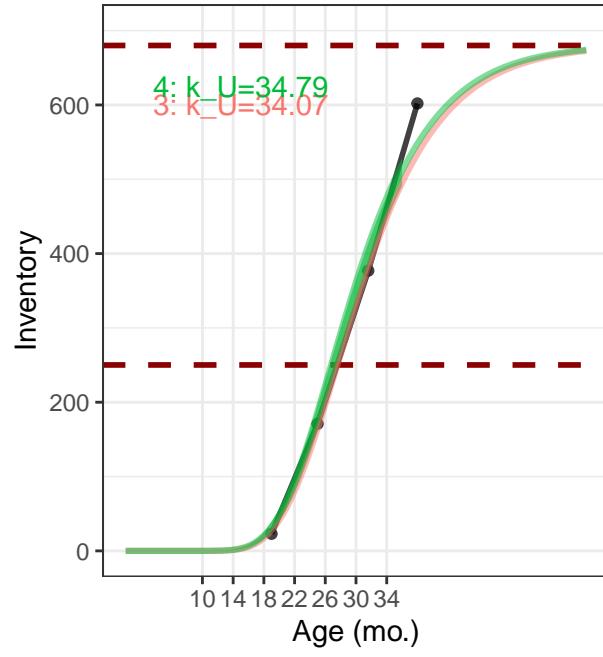
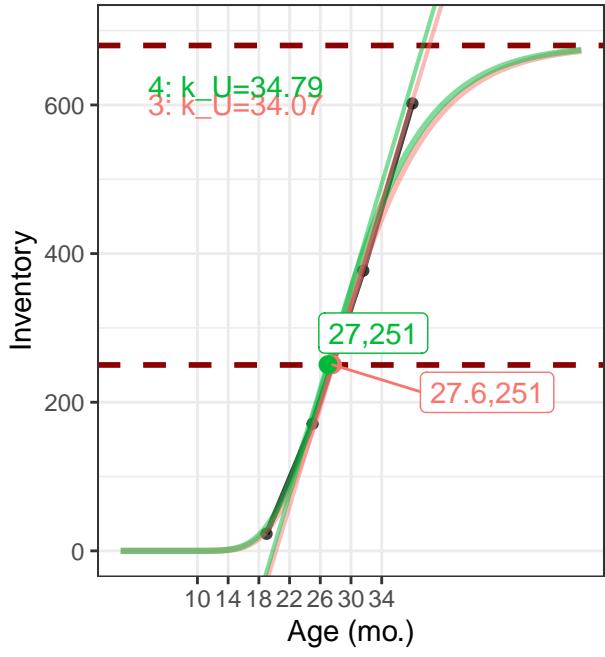


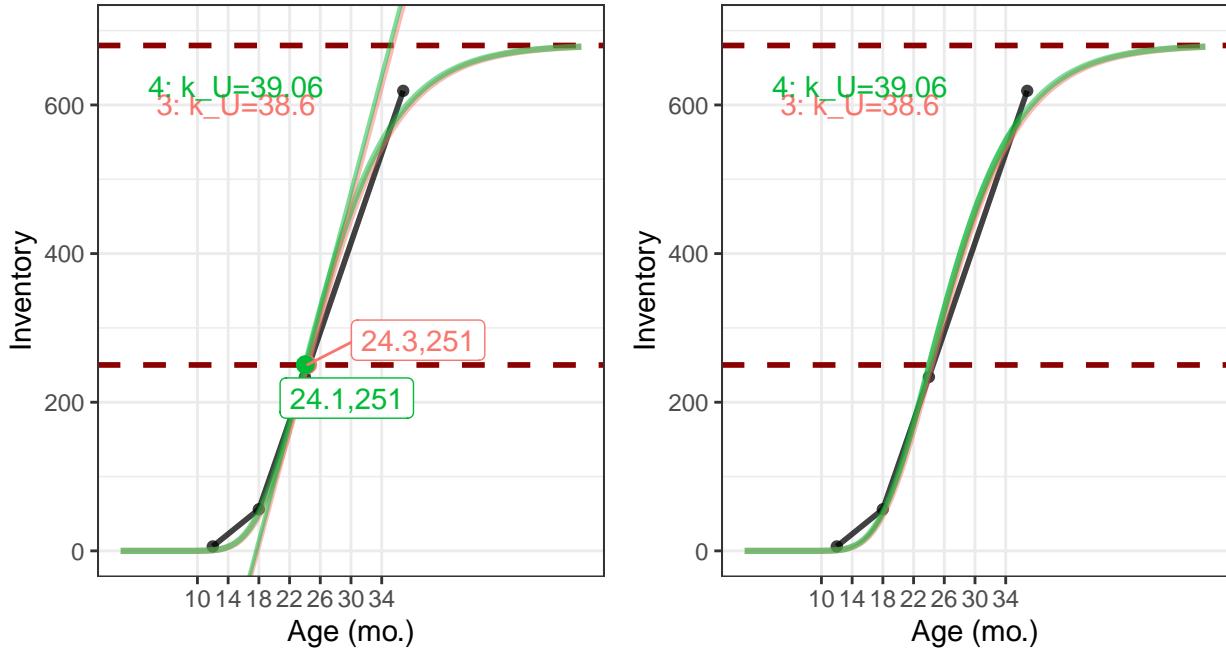












Distribution of k_u

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.

```

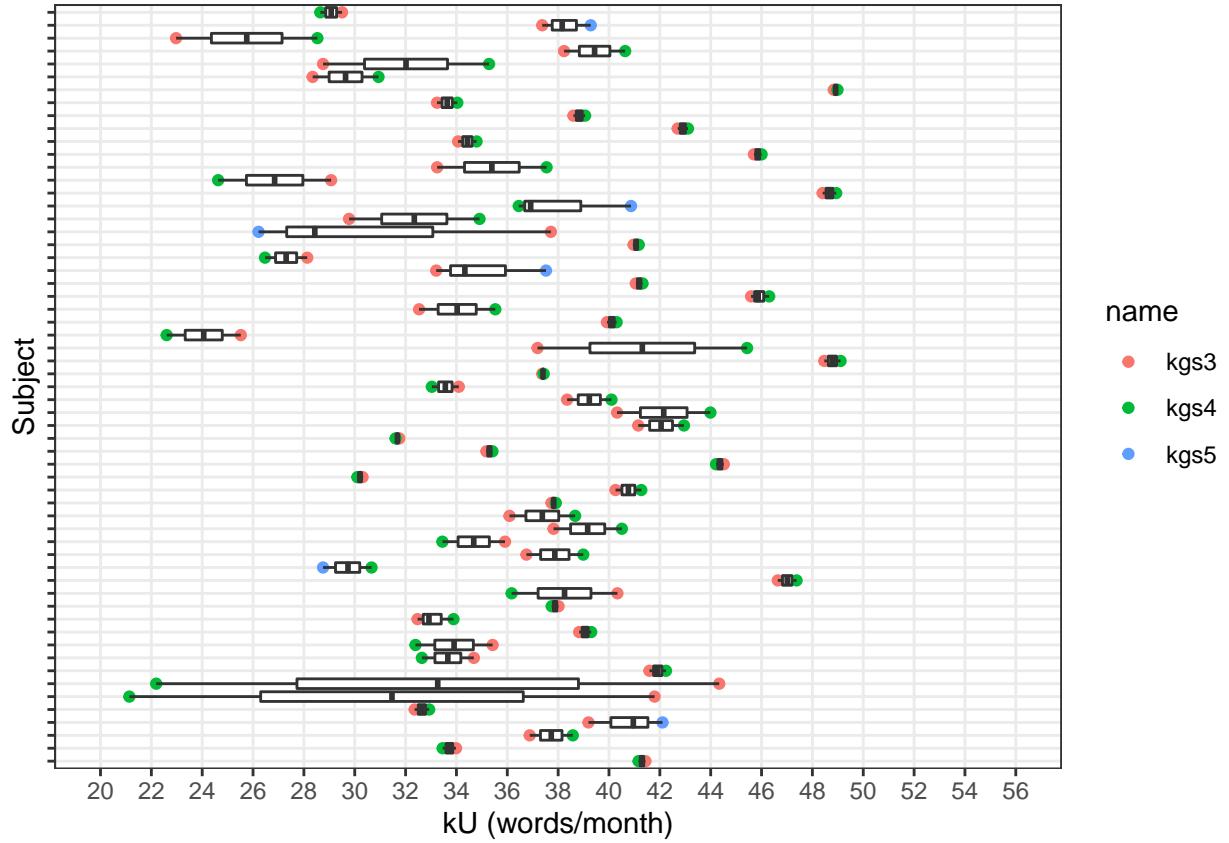
kgs5 <- sapply(all.fits5, function(x) if(is.list(x)) {summary(x)$coefficients[1, 1]} else {NA})
kgs4 <- sapply(all.fits4, function(x) summary(x)$coefficients[1, 1])
kgs3 <- sapply(all.fits3, function(x) summary(x)$coefficients[1, 1])

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.

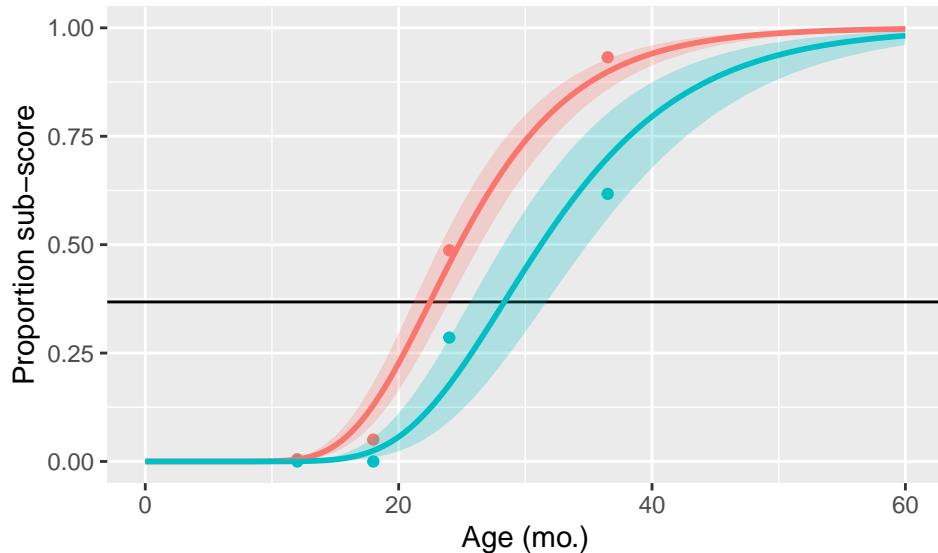
These individuals don't seem to have a systematically low oldest visit or low largest inventory size.

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 $W_i \approx 0.37$.

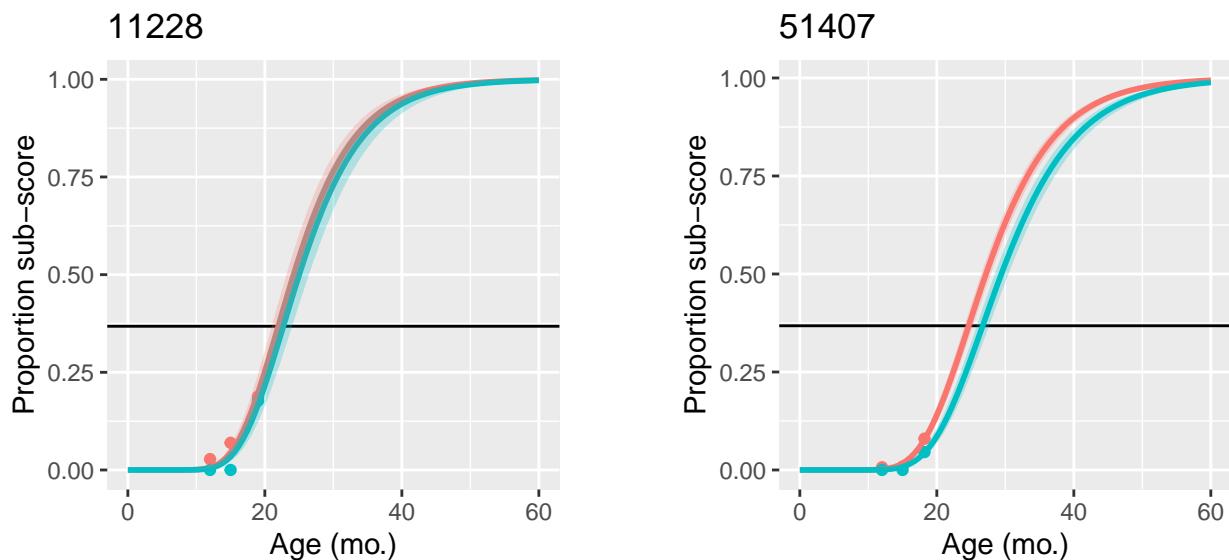
This plot uses the 95% confidence interval rather than the SE.



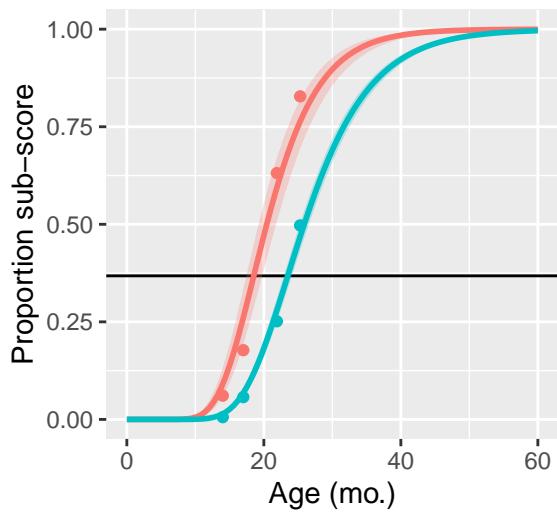
The take-away here is that syntactic development occurs later, and even though the SE is larger, the ranges do *not* overlap. It is the difference between T_i, lex and T_i, syn that is interesting.

Lexical vs. syntactic across subjects

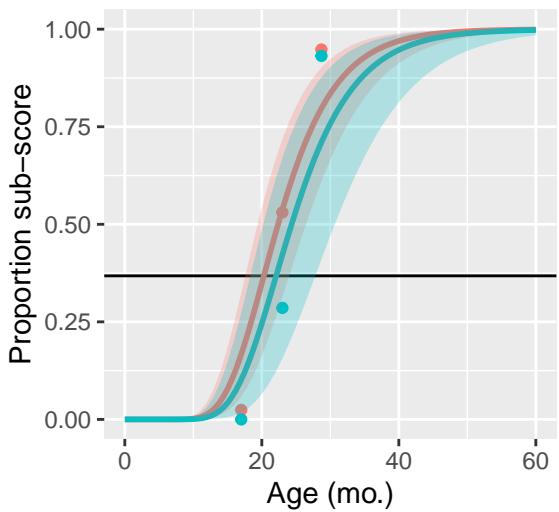
Individuals in rows/columns (not one subject per row as above).



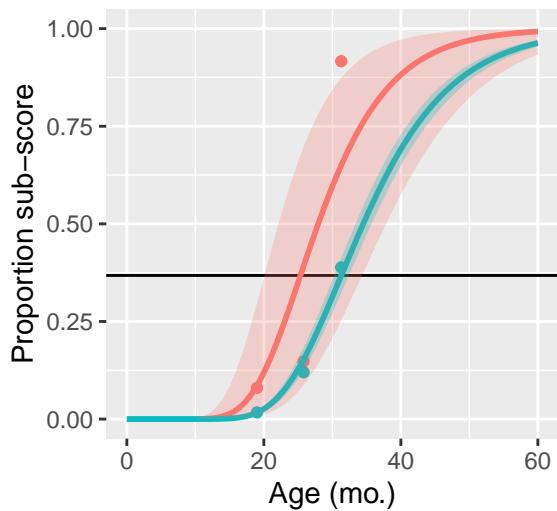
52977



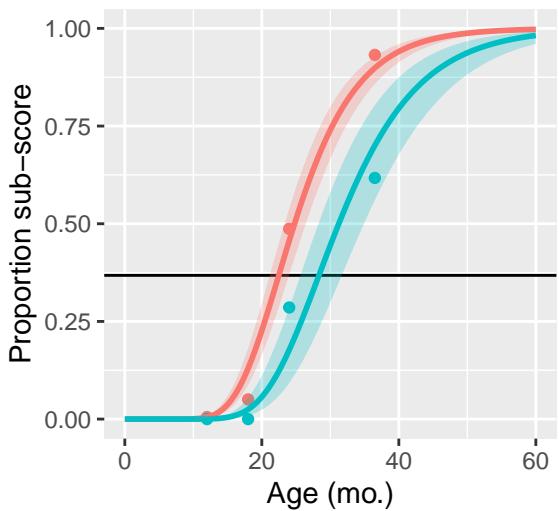
100619



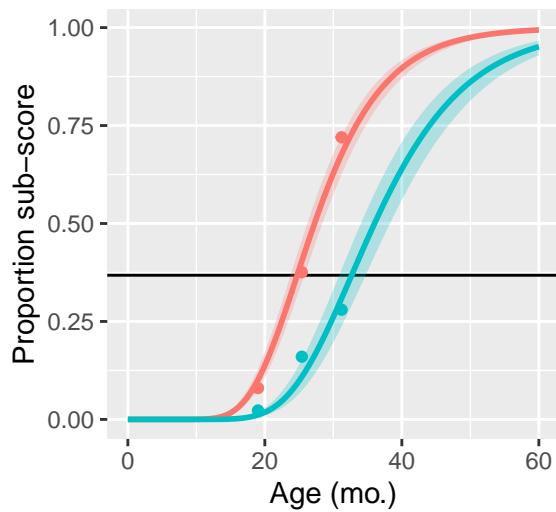
107842



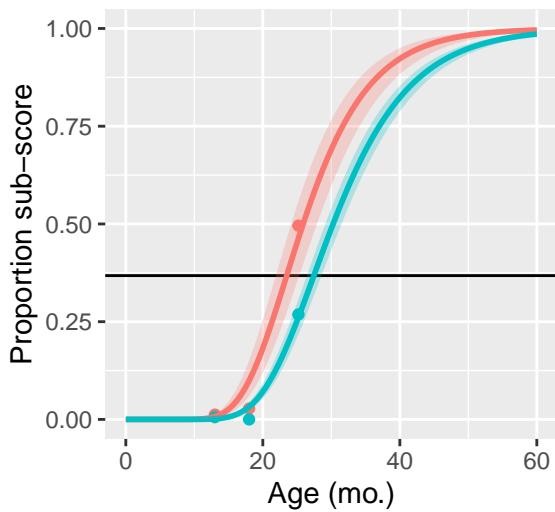
116056



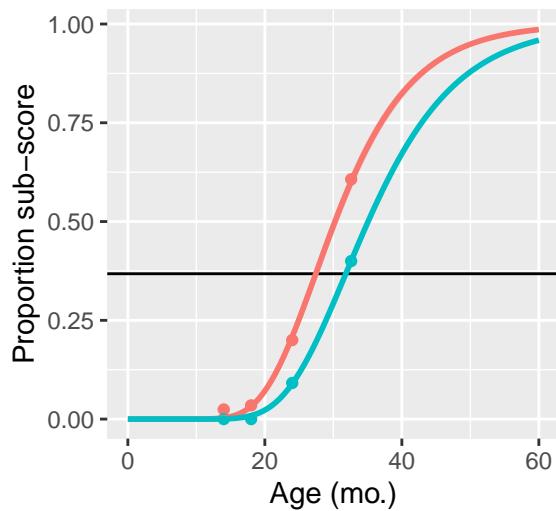
124529



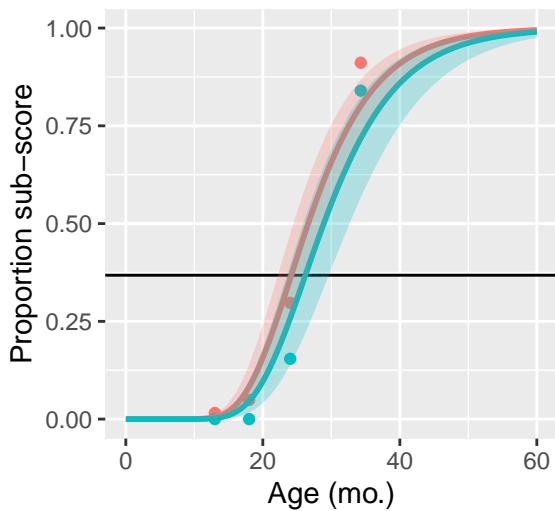
125632



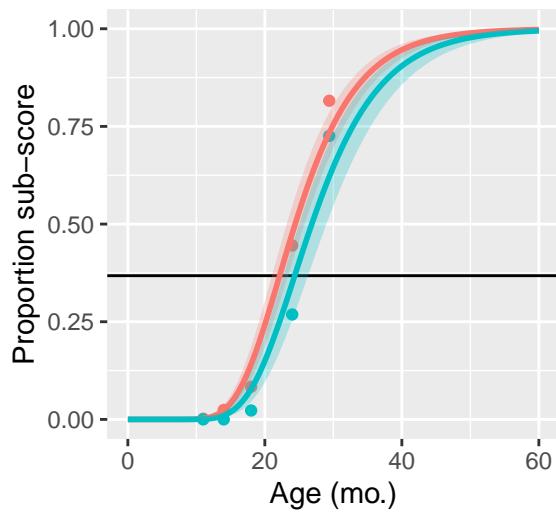
131015



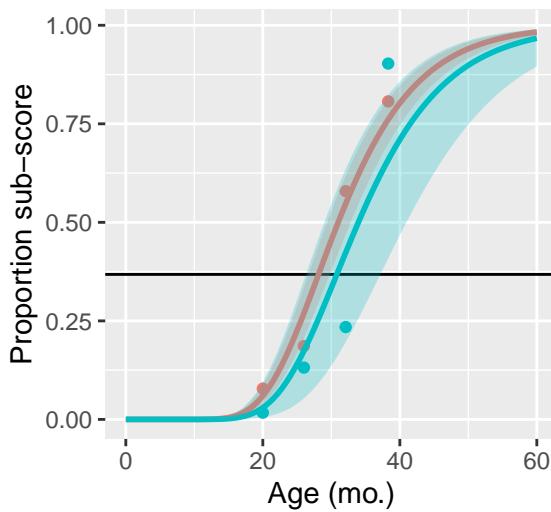
138813



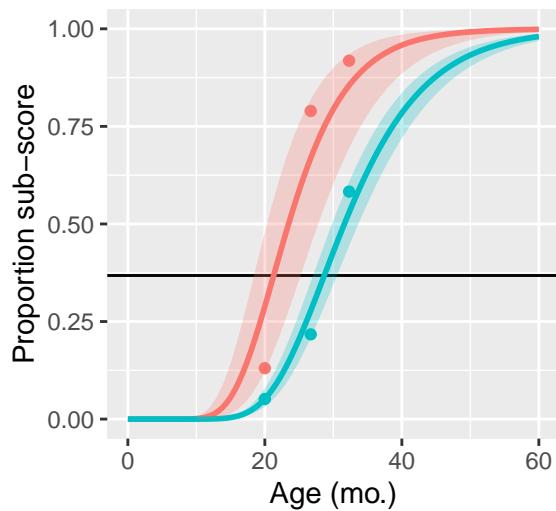
176427



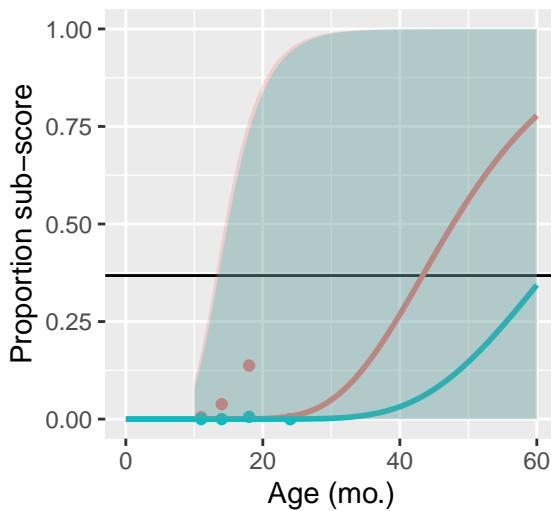
176851



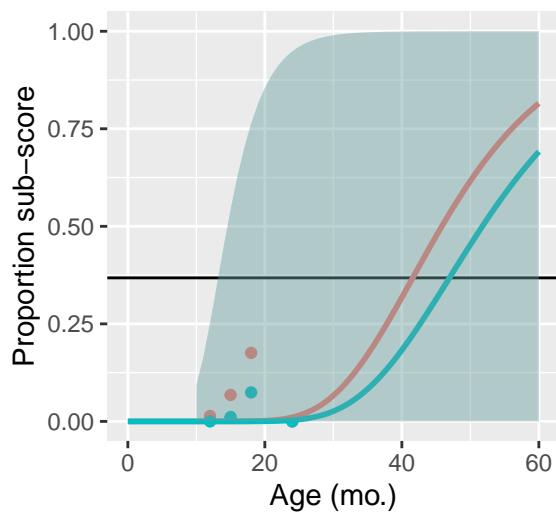
185373



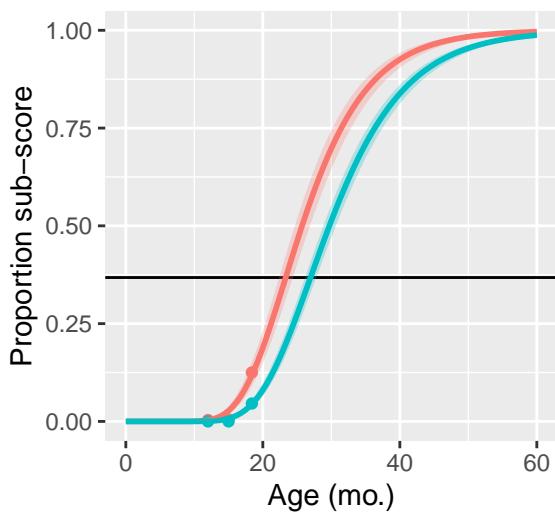
198202



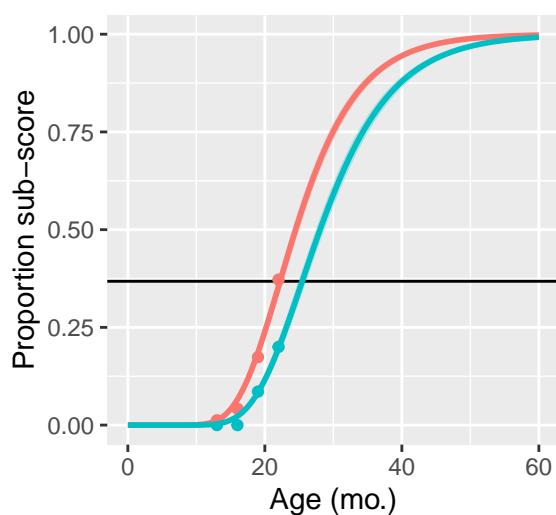
199865



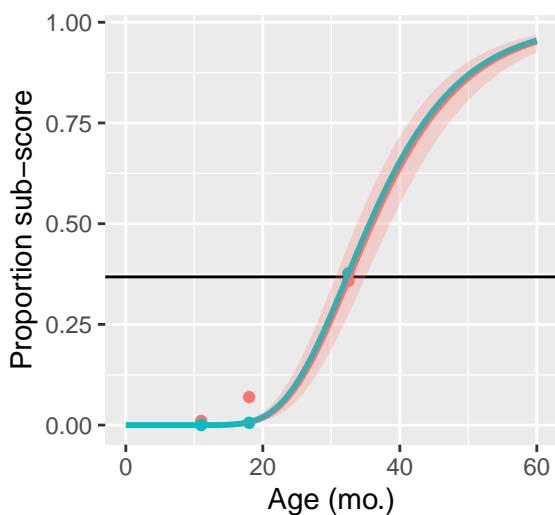
200474



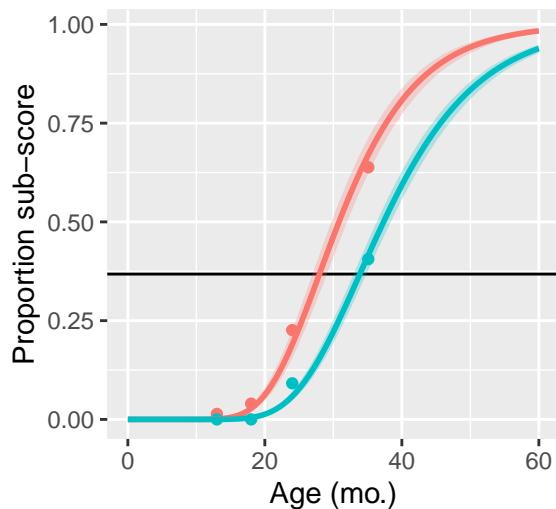
204659



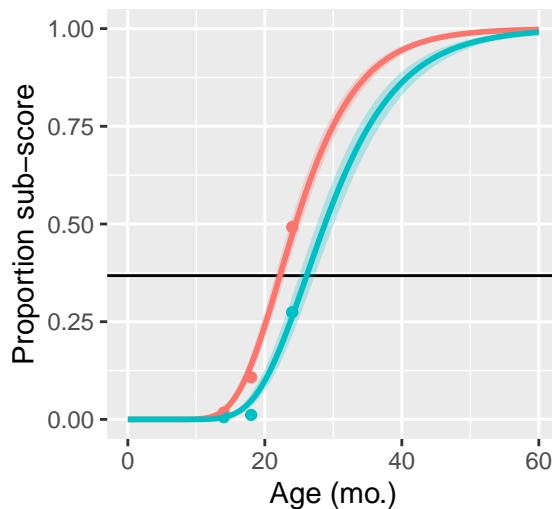
222881



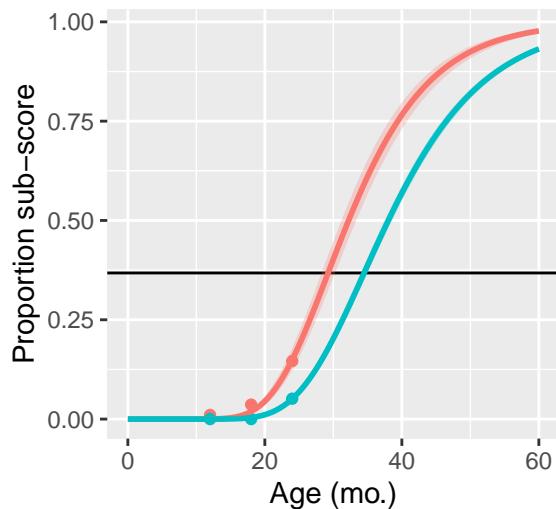
223008



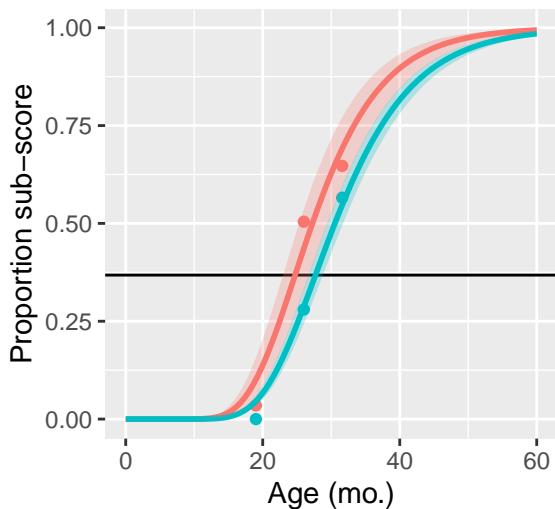
229768



231205

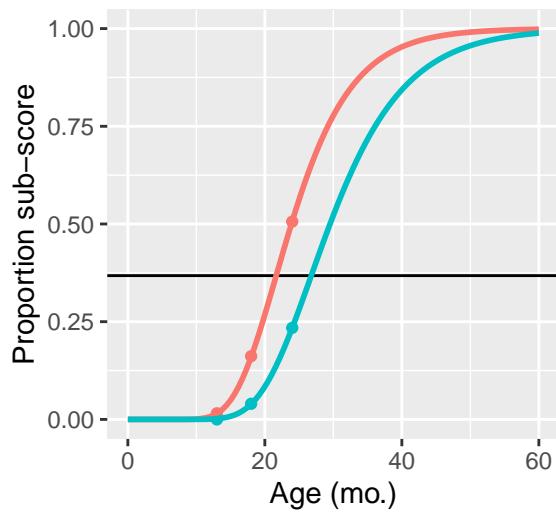


234130

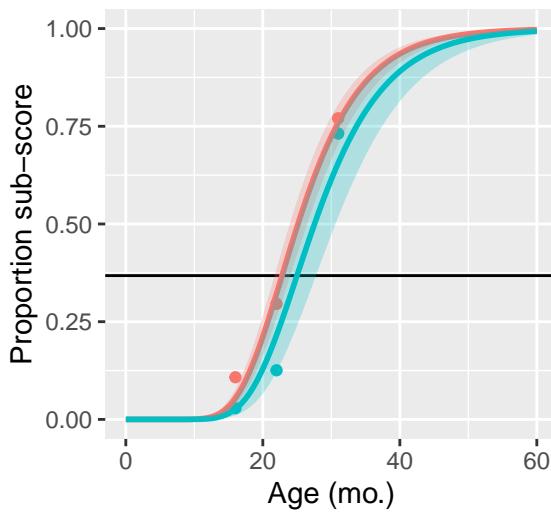


```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")), :  
##   number of iterations exceeded maximum of 500  
## [1] NA
```

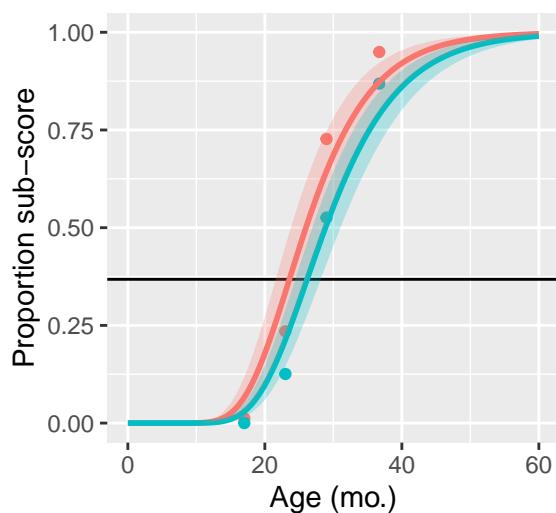
243511



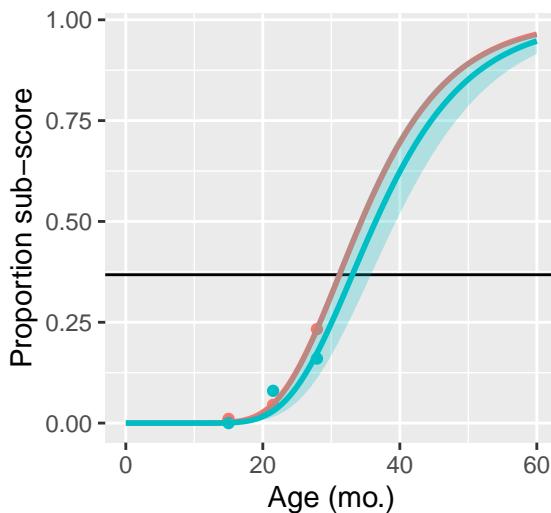
251549



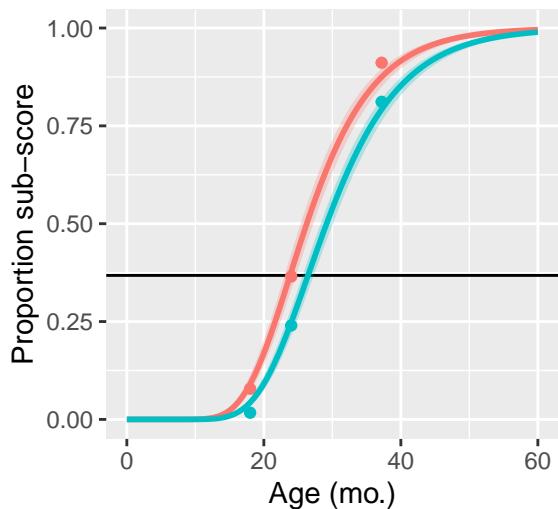
261266



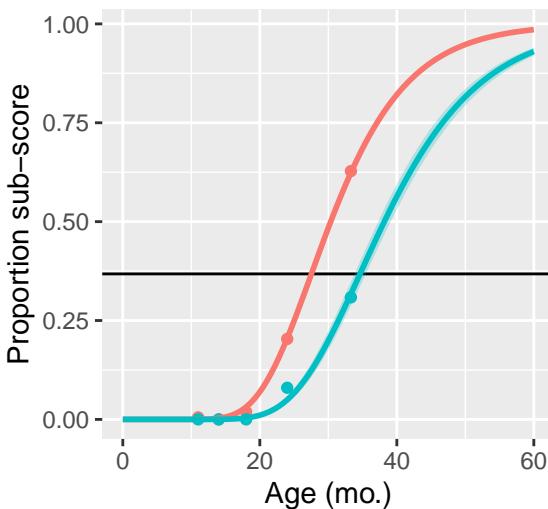
266394



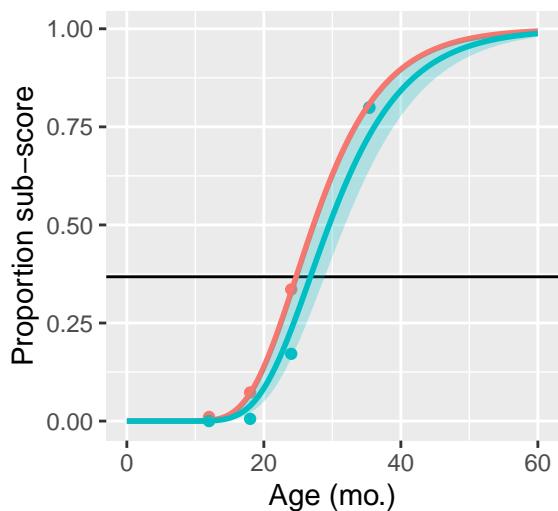
266437



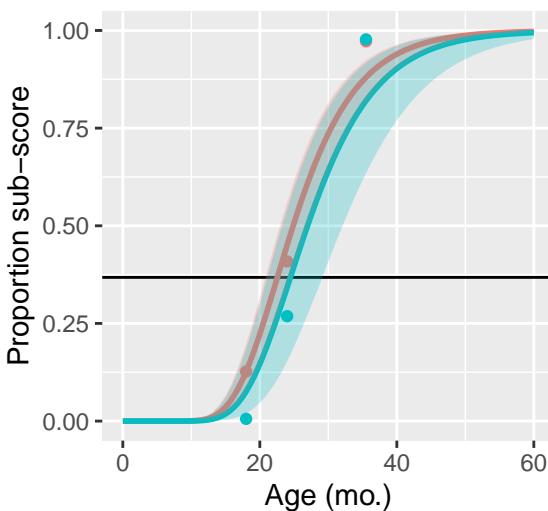
272097



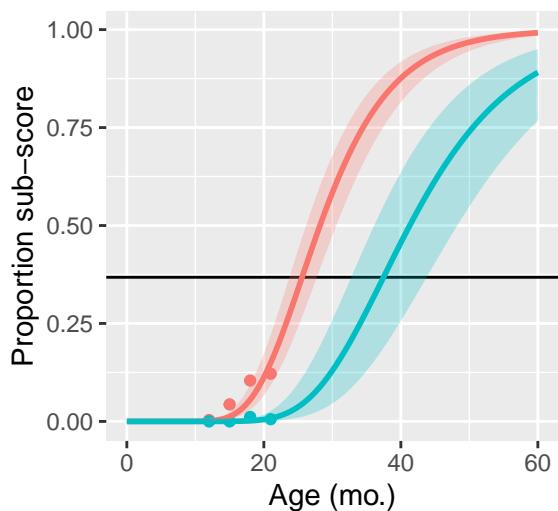
284149



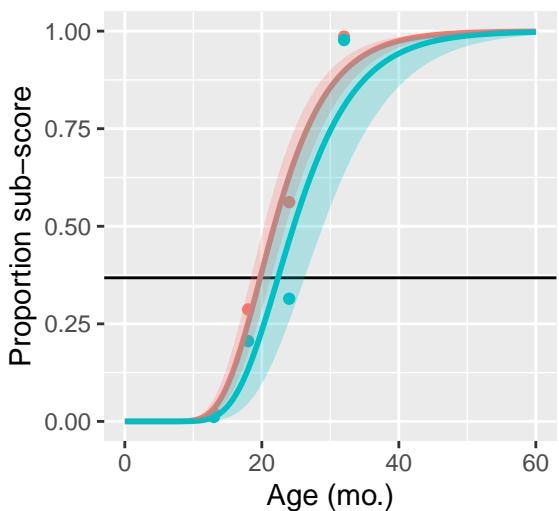
295081



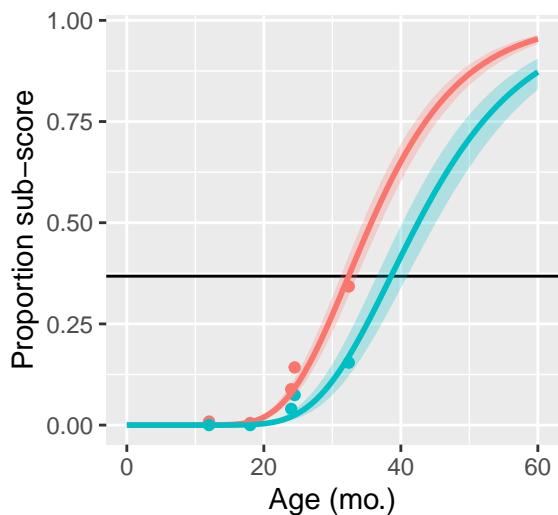
309615



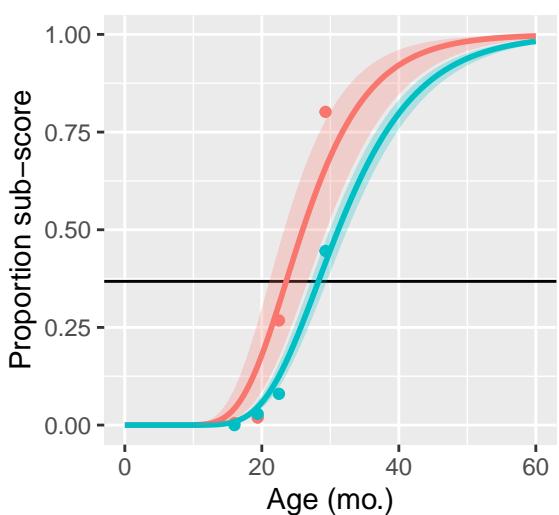
320544



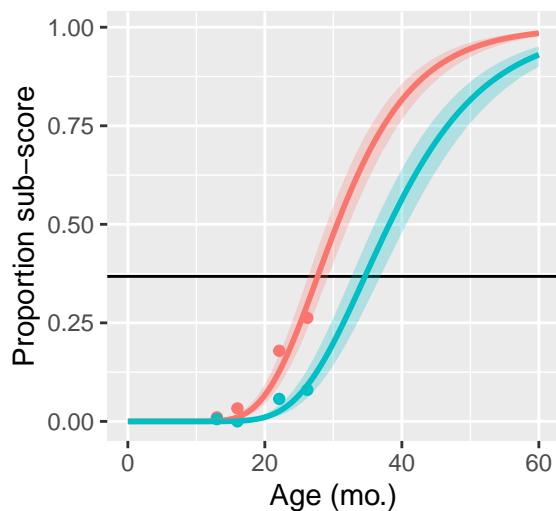
334324



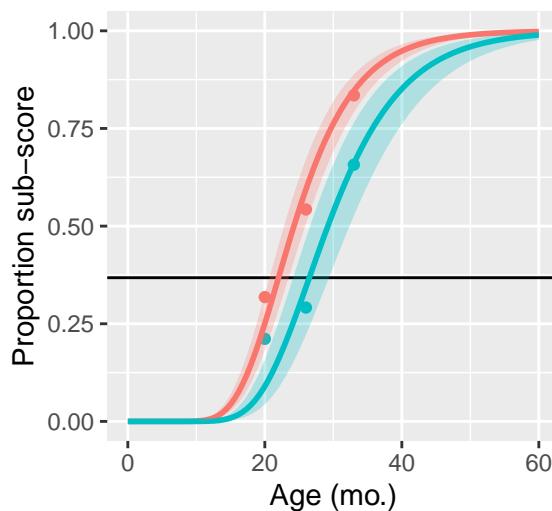
340476



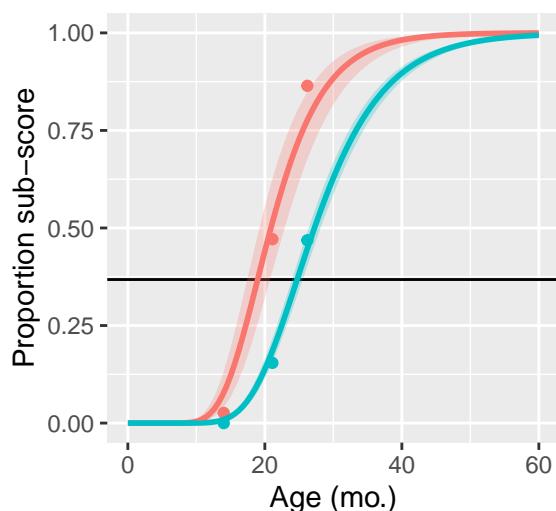
354404



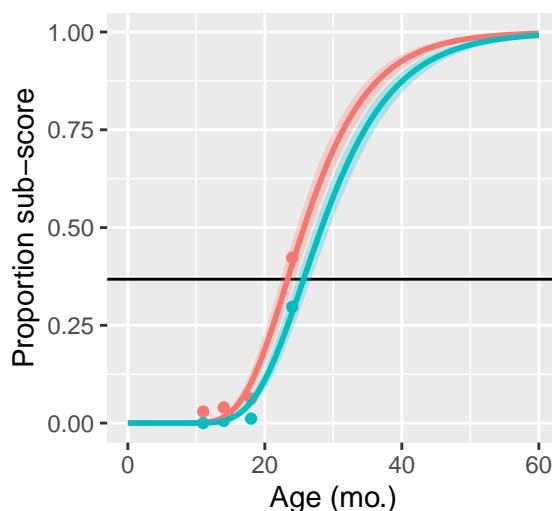
380510



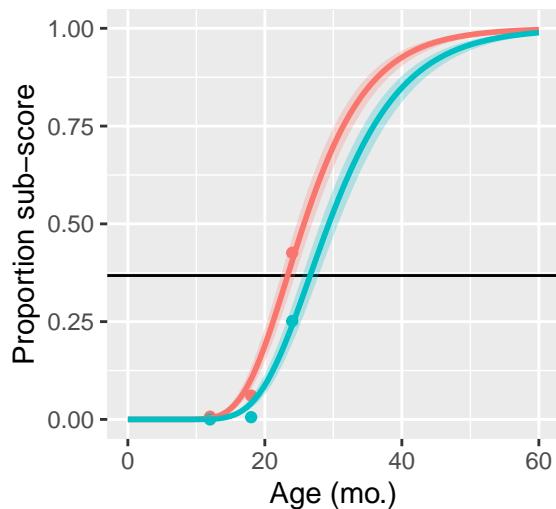
385434



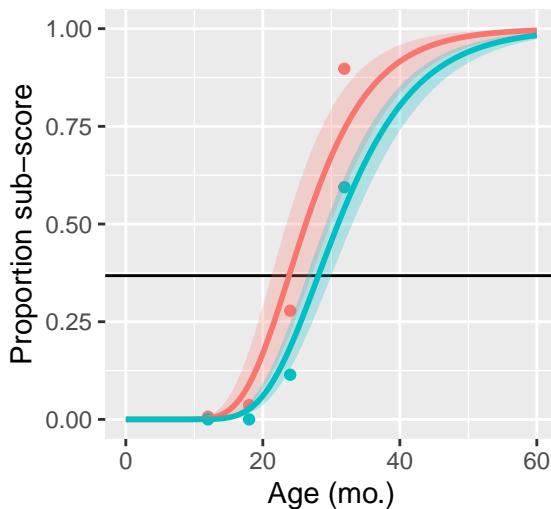
387247



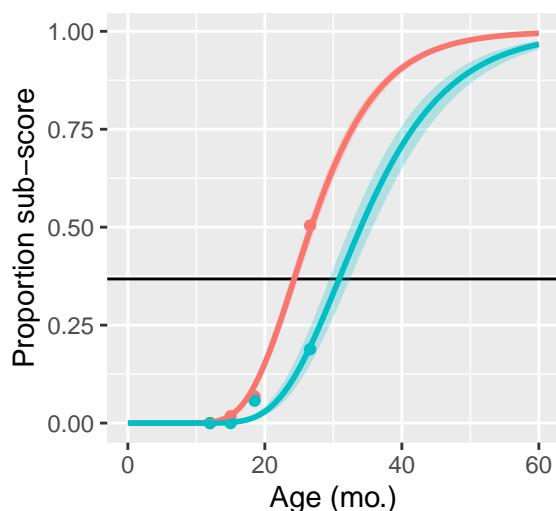
401592



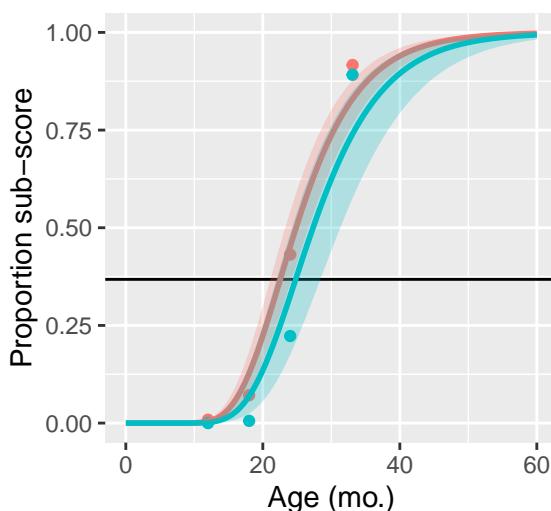
406768



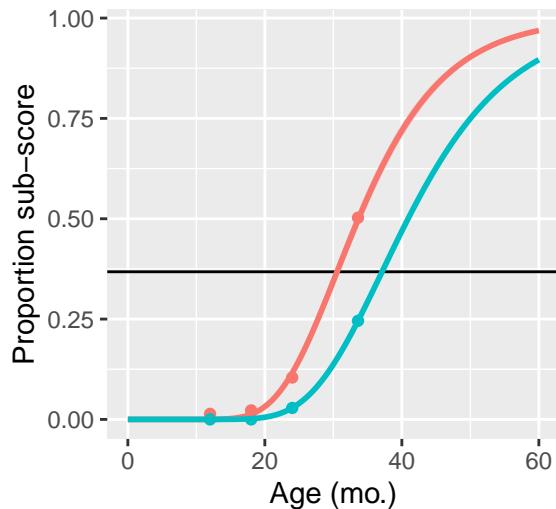
418793



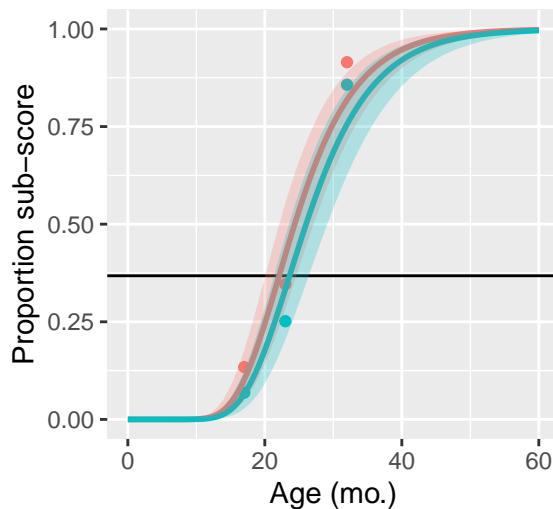
423548



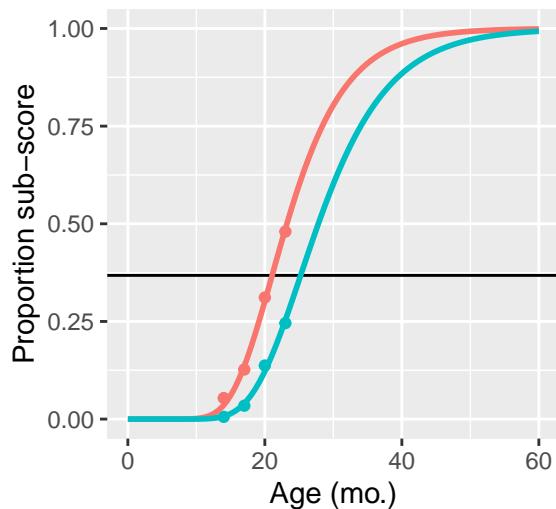
425428



427732

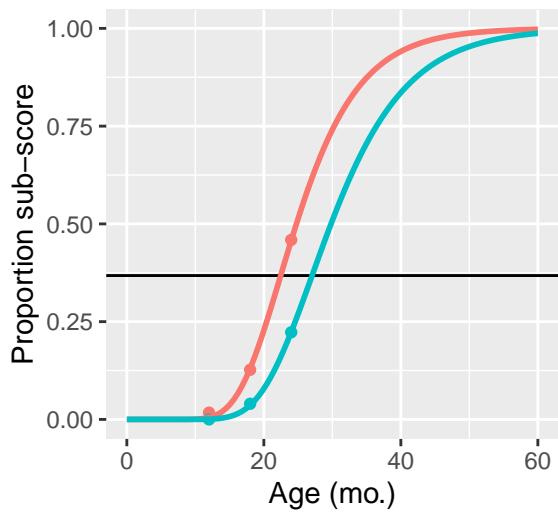


439999

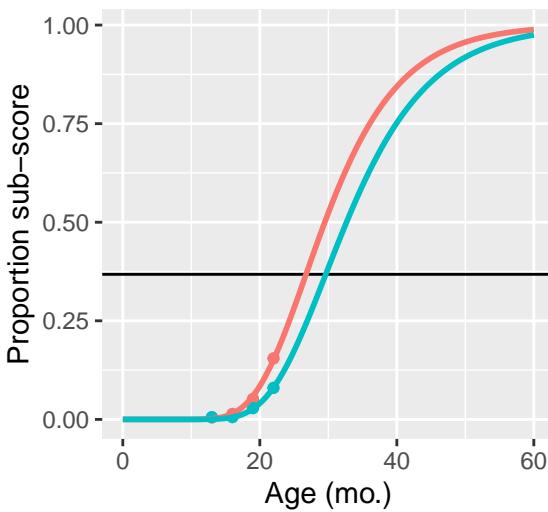


```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")), :  
##   number of iterations exceeded maximum of 500  
## [1] NA
```

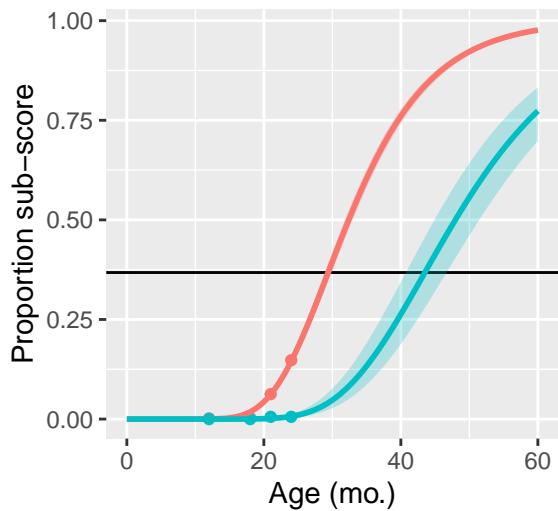
455675



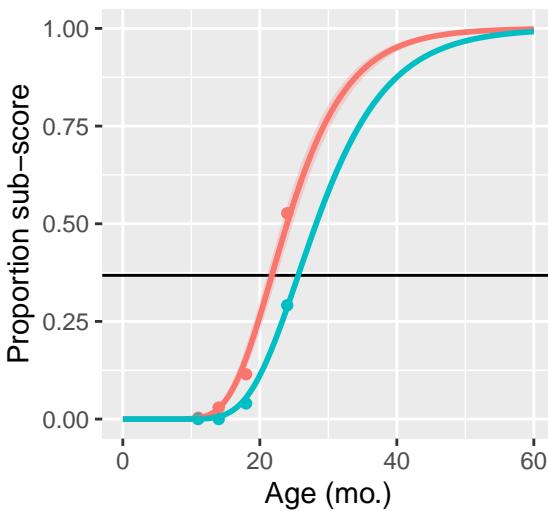
458050



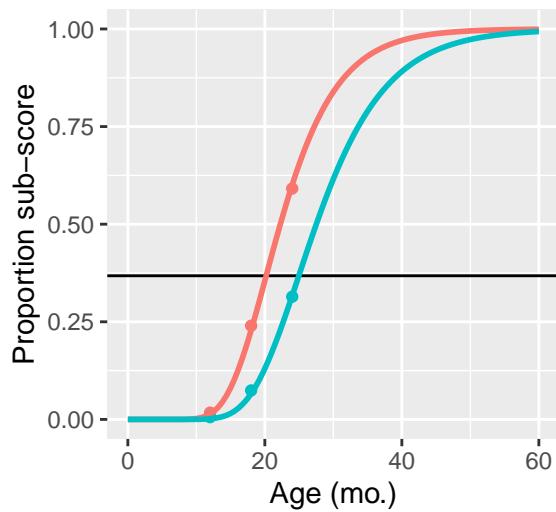
469090



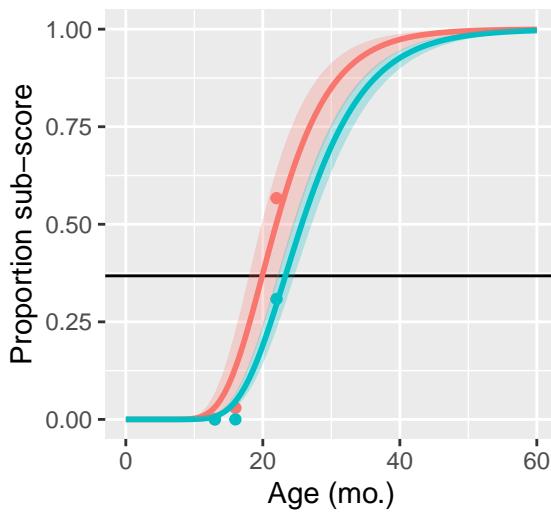
469900



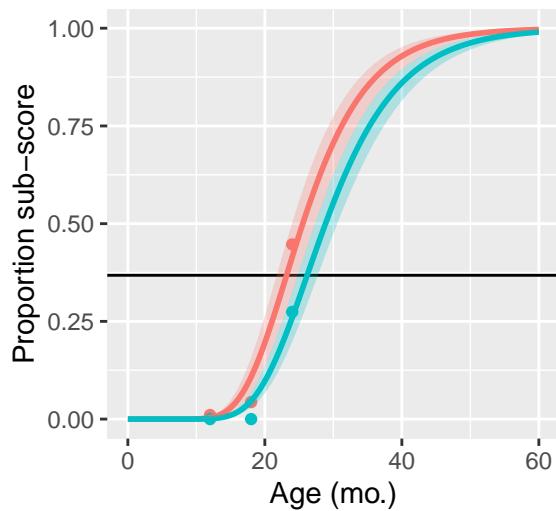
470431



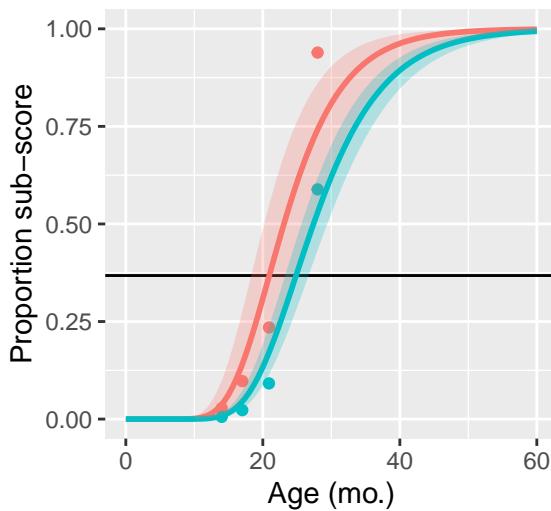
477045



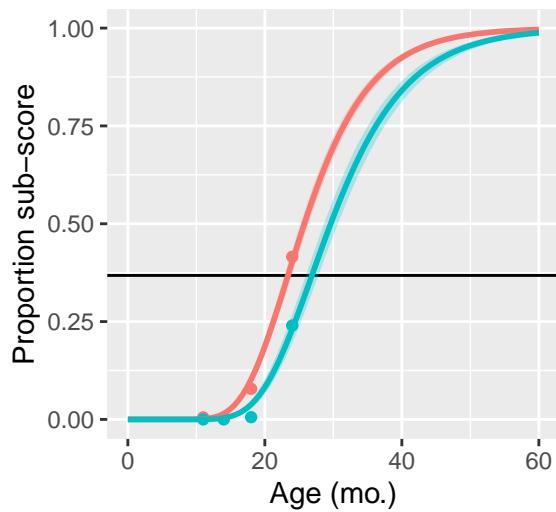
505499



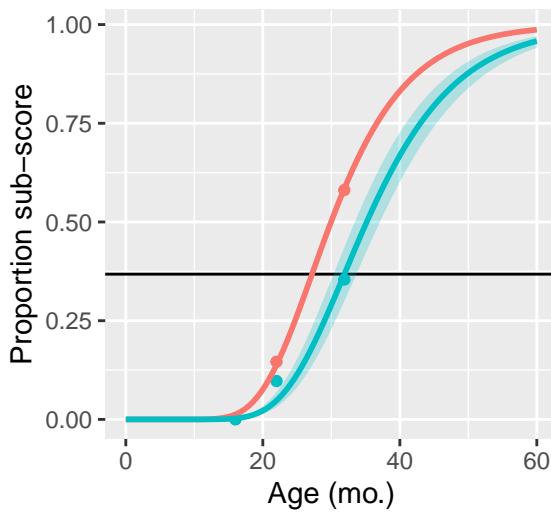
505525



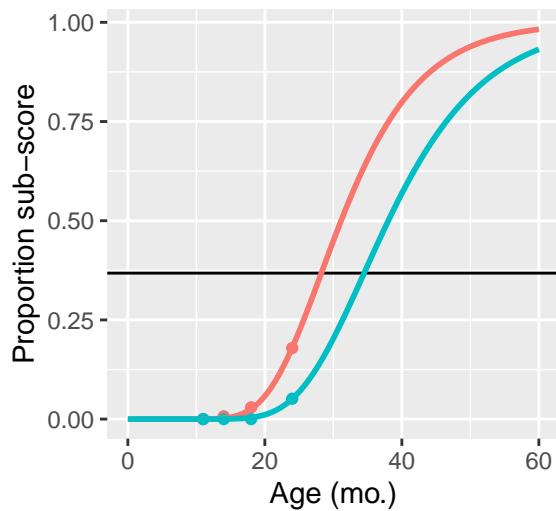
505926



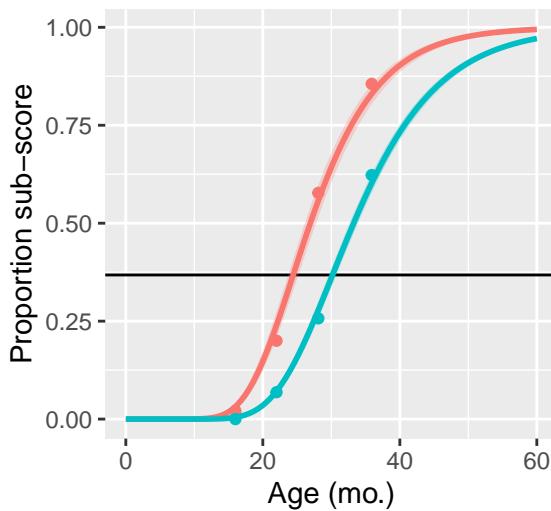
513010



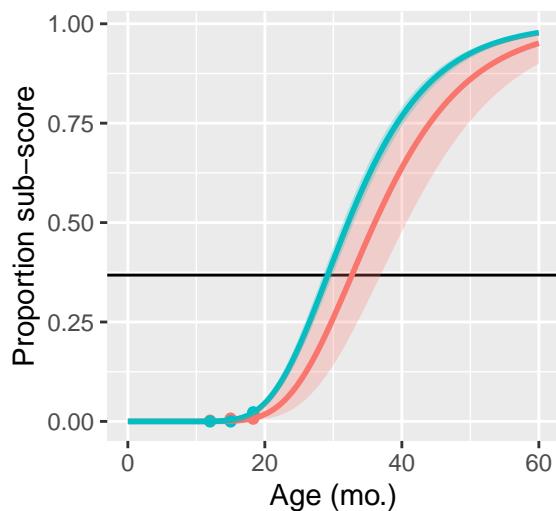
518969



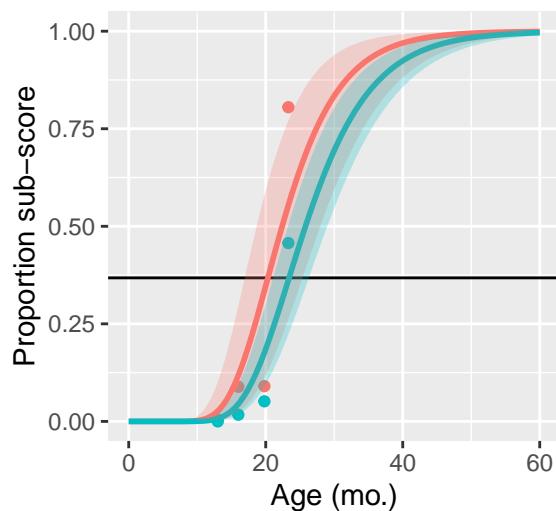
519250



525955

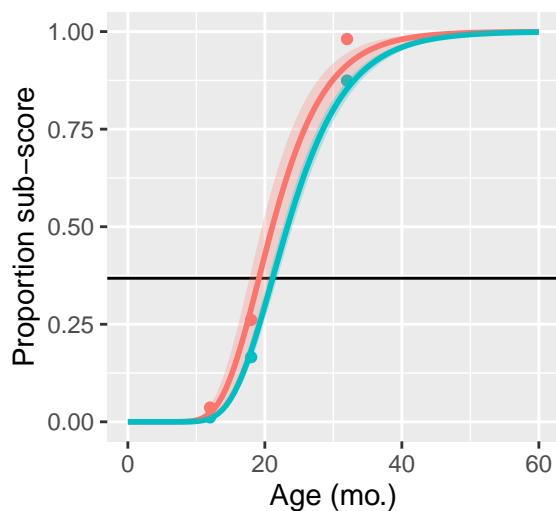


530066

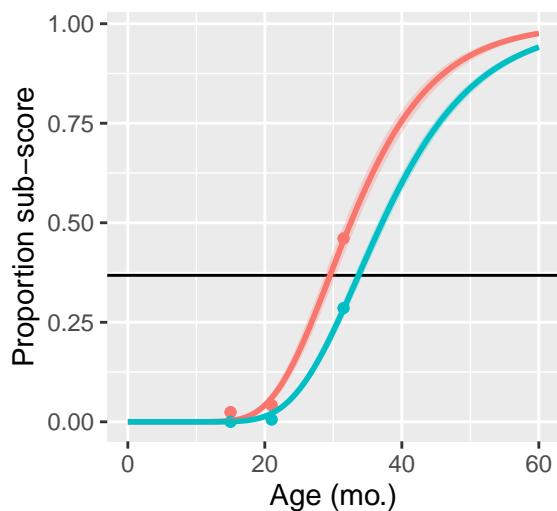


```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")), :  
##   number of iterations exceeded maximum of 500  
## [1] NA
```

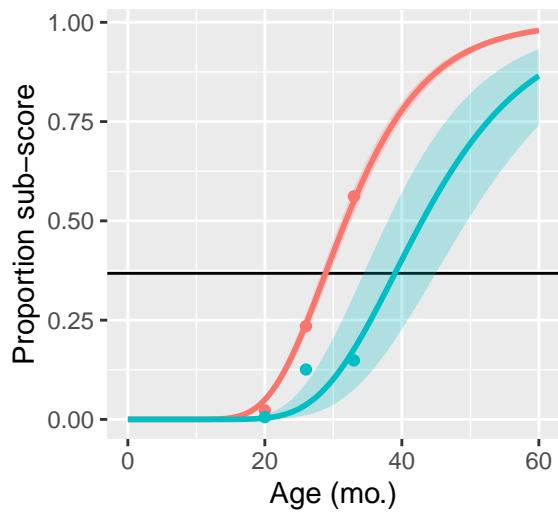
557806



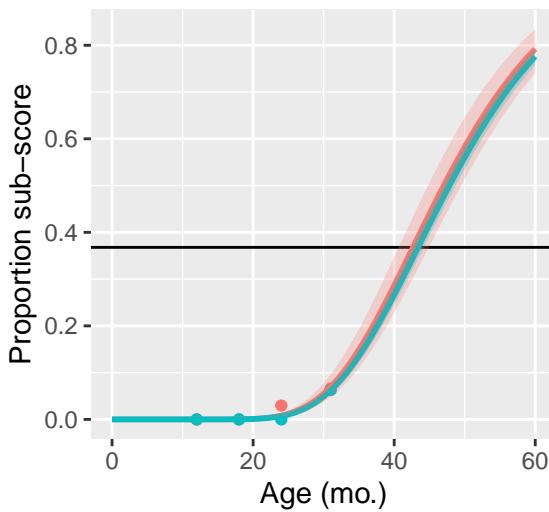
569511



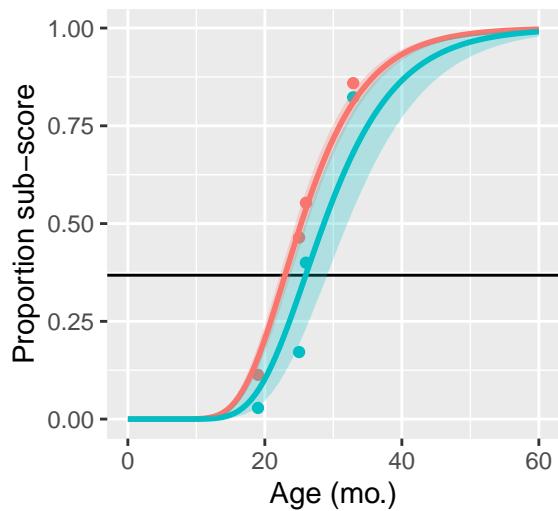
583494



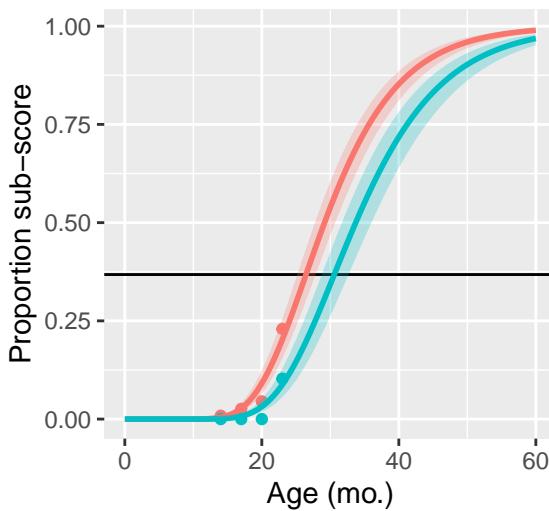
597306



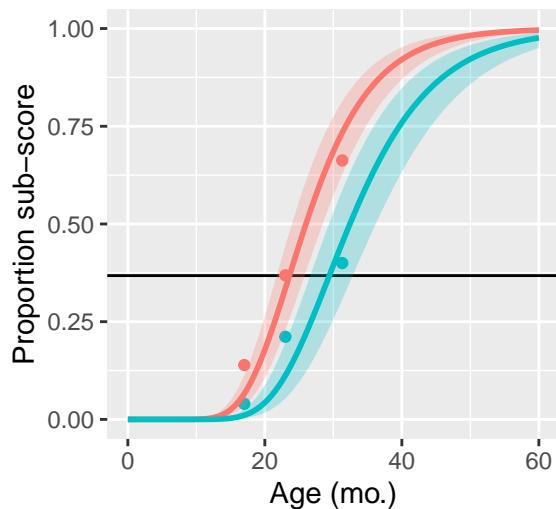
601017



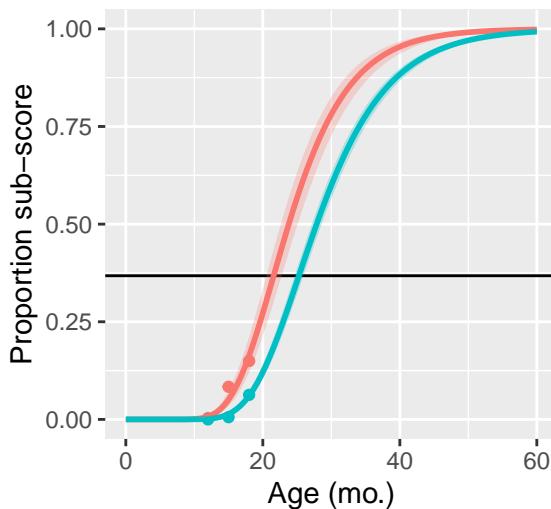
629130



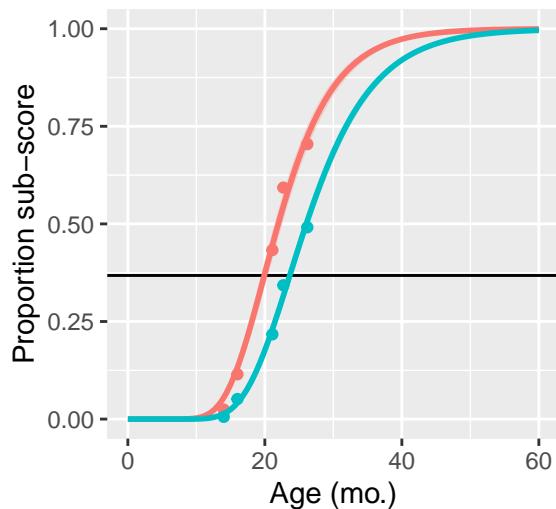
640680



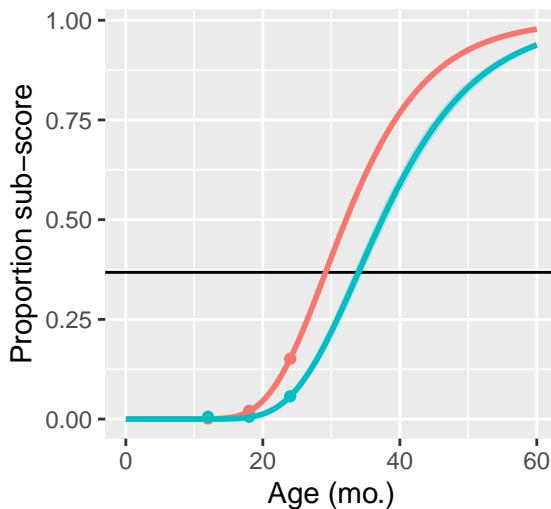
648669



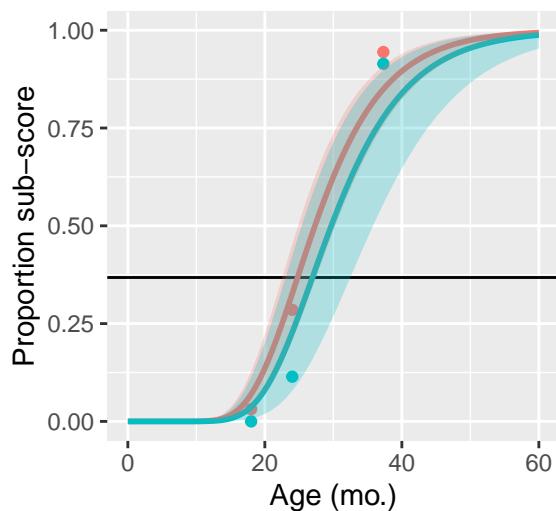
657965



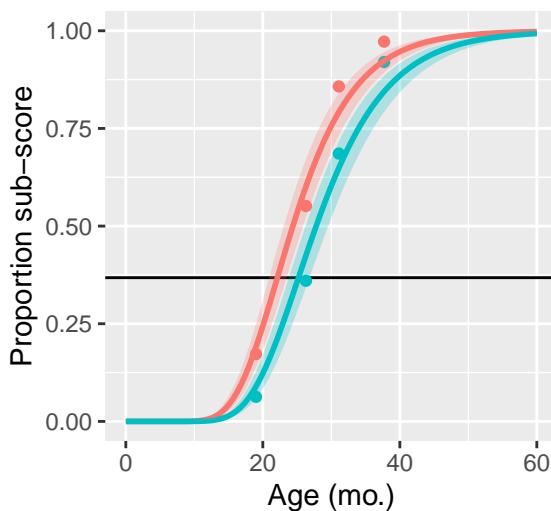
662338



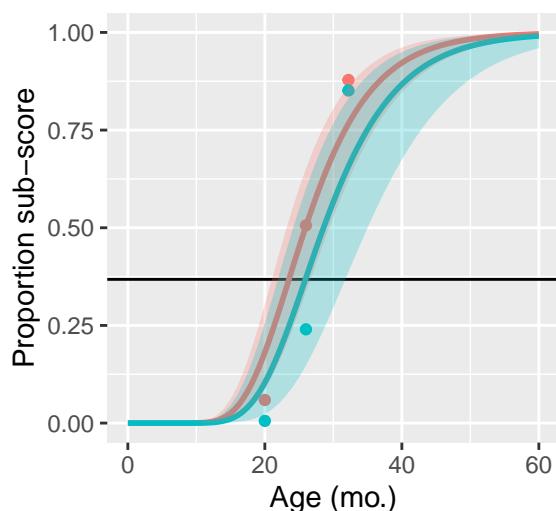
675362



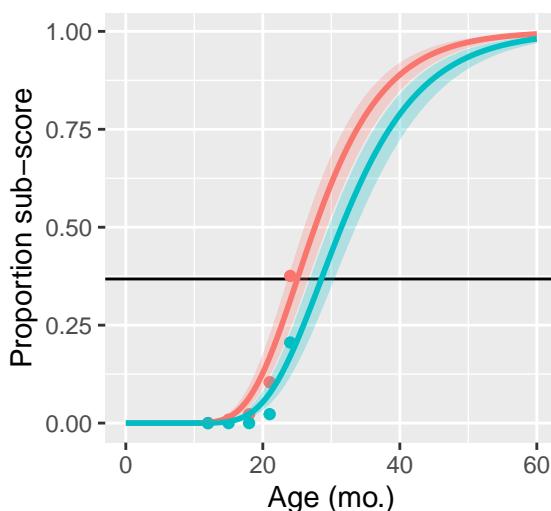
676274



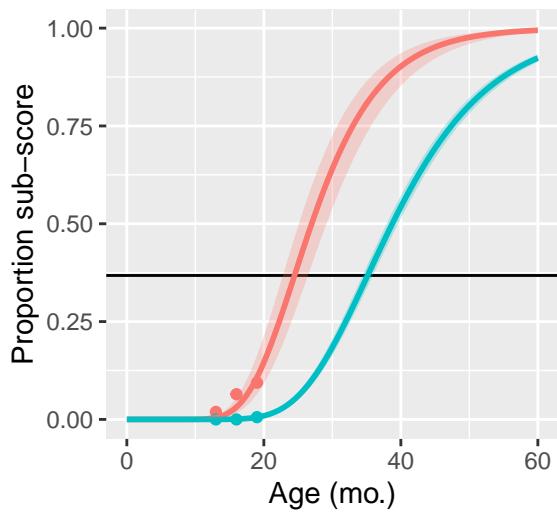
706050



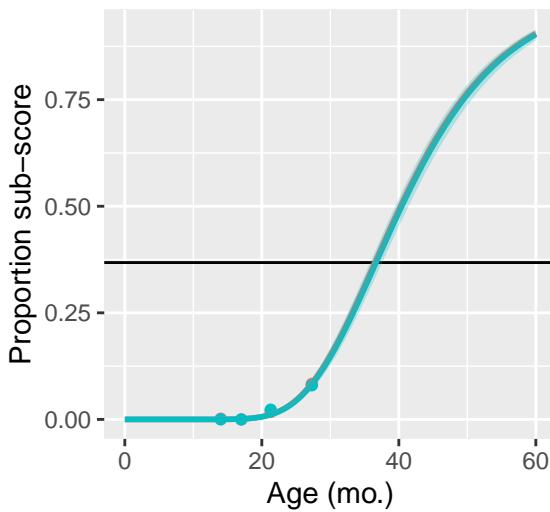
706655



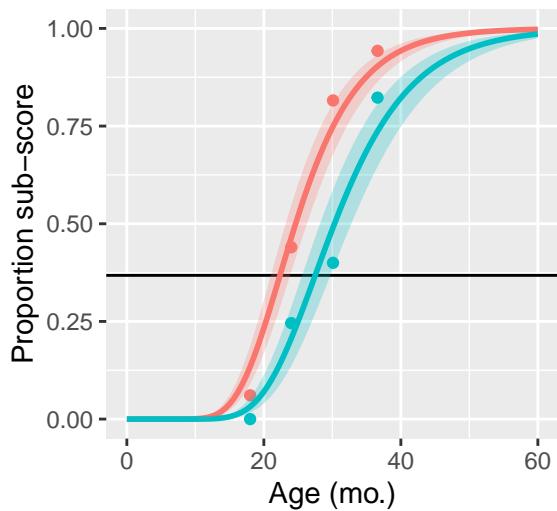
710741



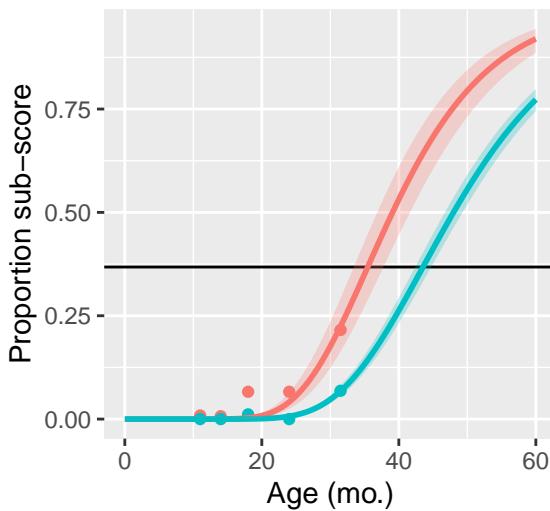
713347



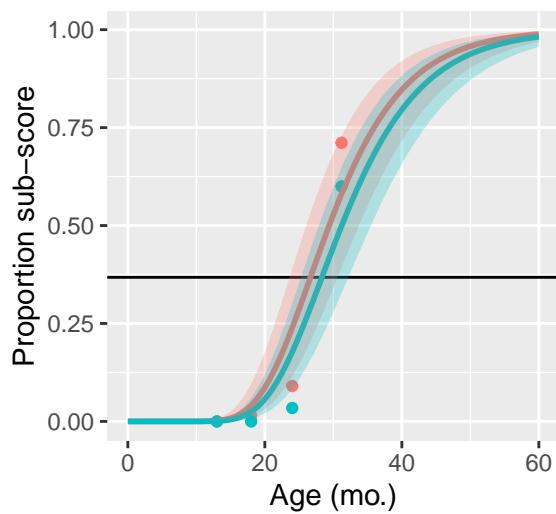
722612



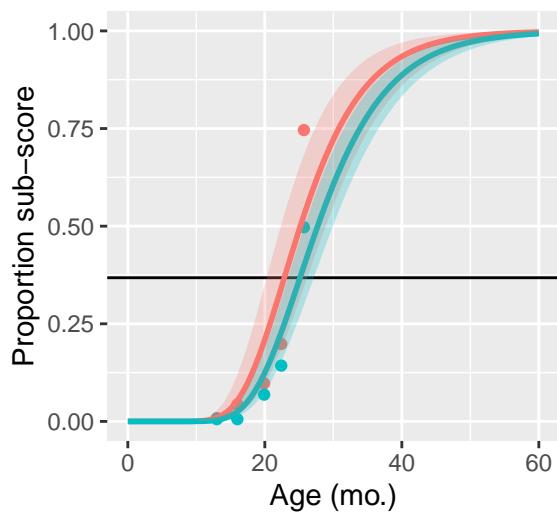
738379



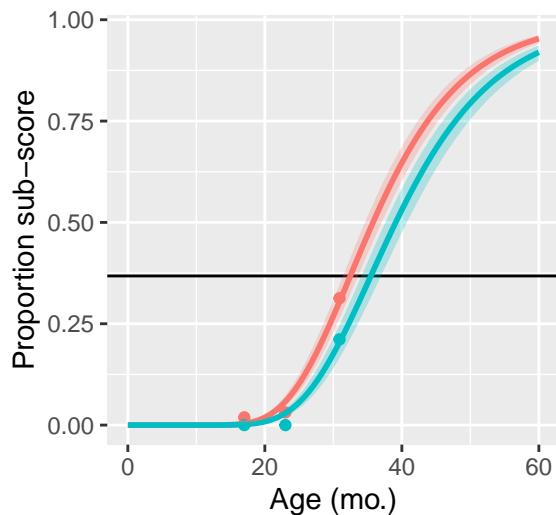
755330



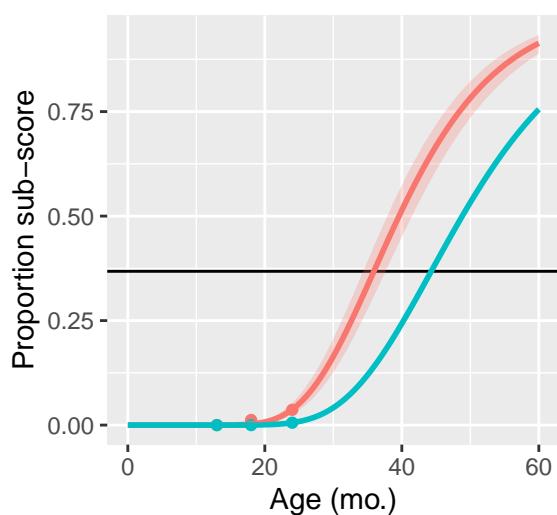
758603



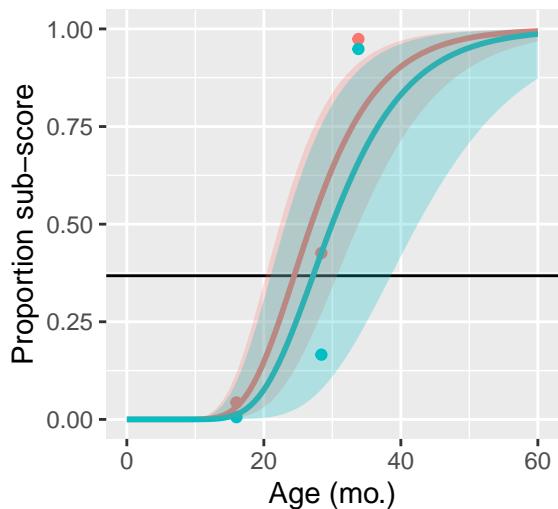
759162



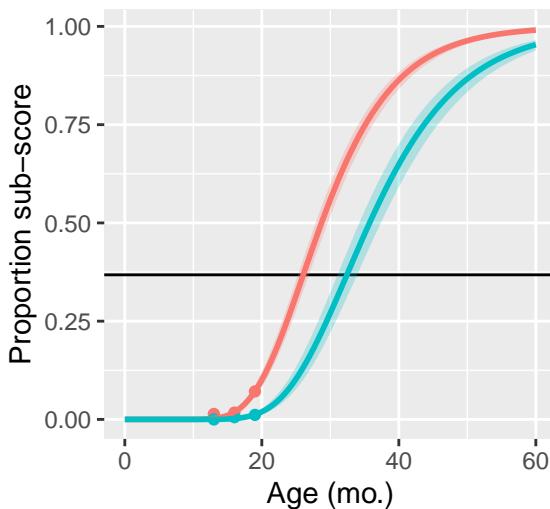
760812



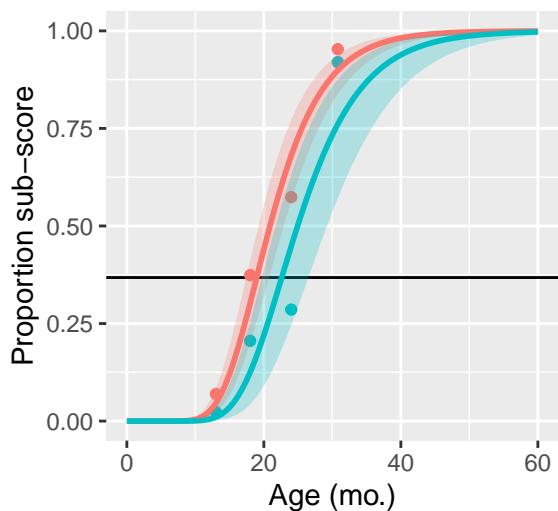
764612



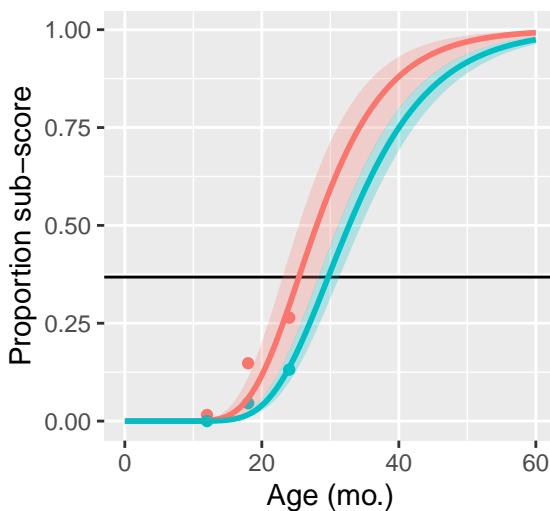
779253



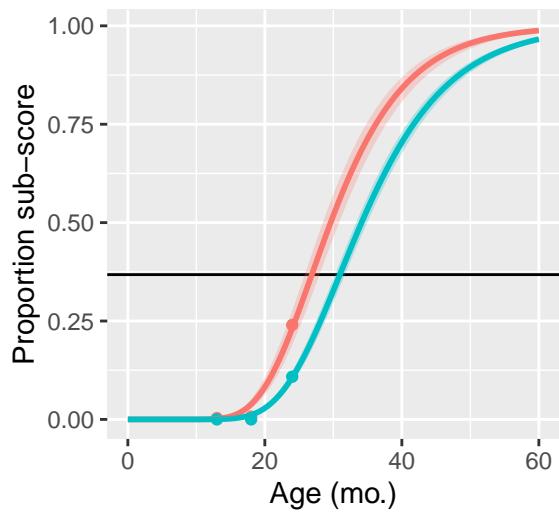
786266



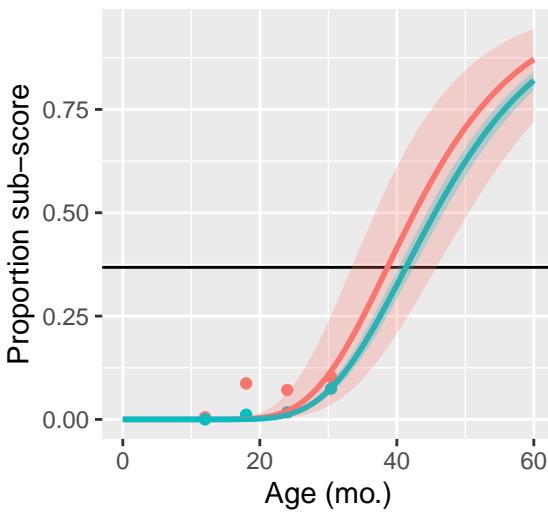
794032



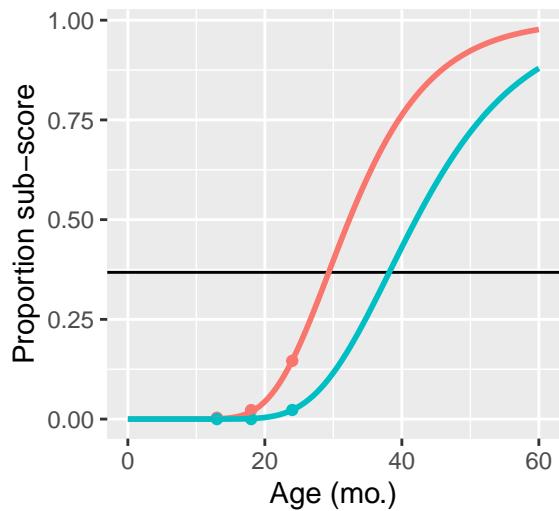
794626



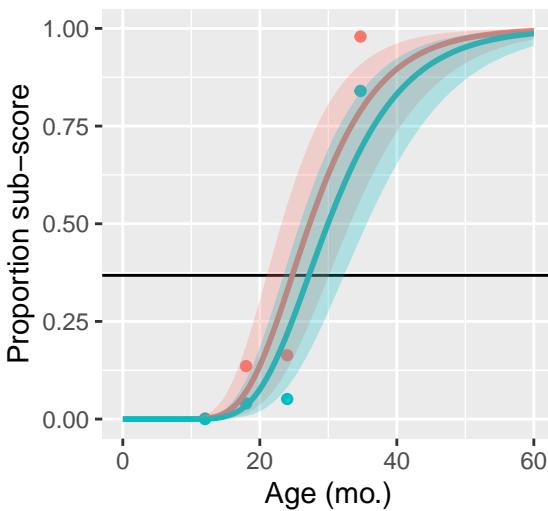
801091



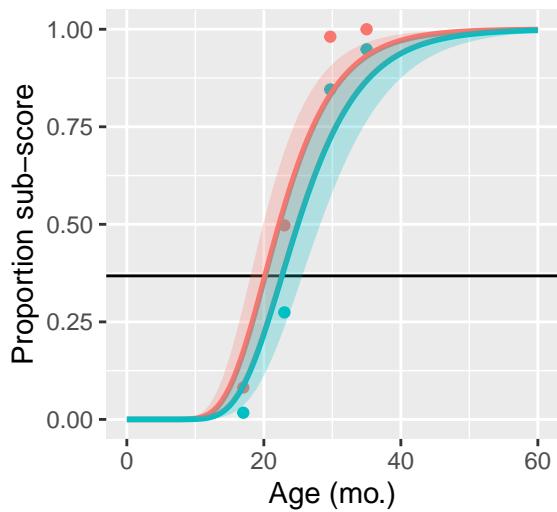
805063



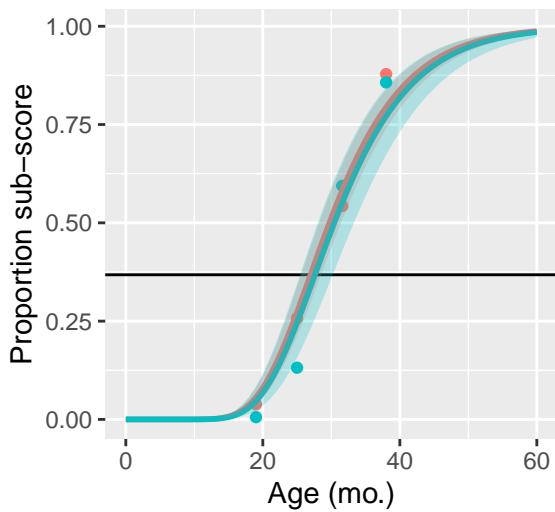
812574



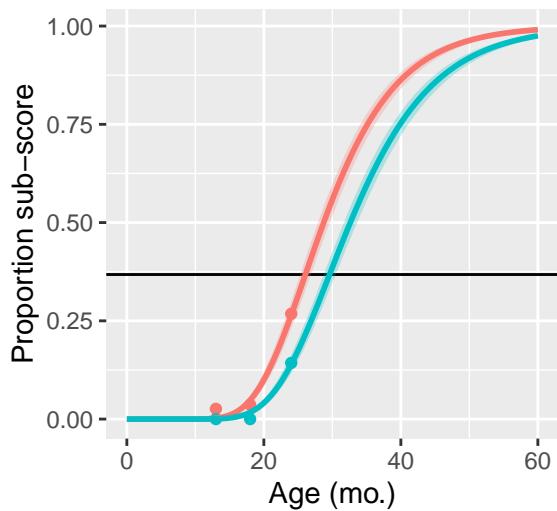
826360



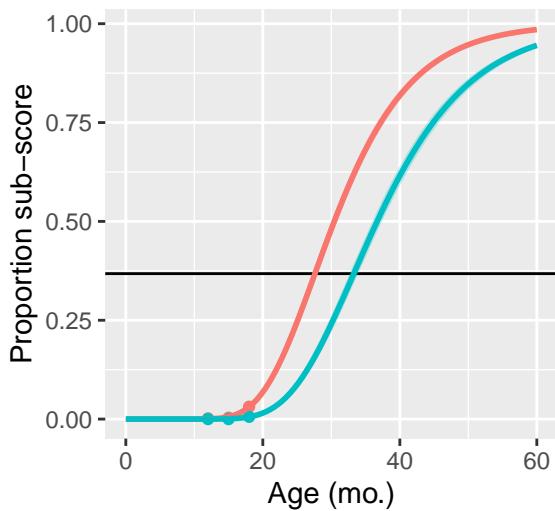
830201



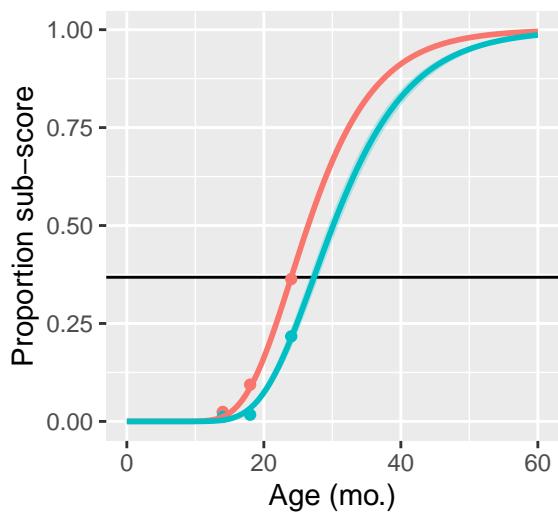
838827



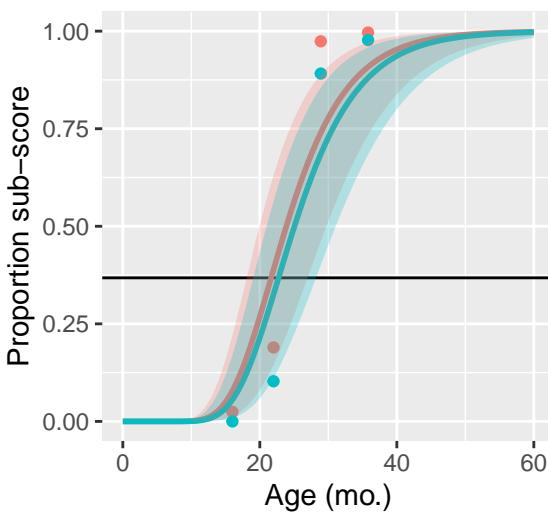
839327



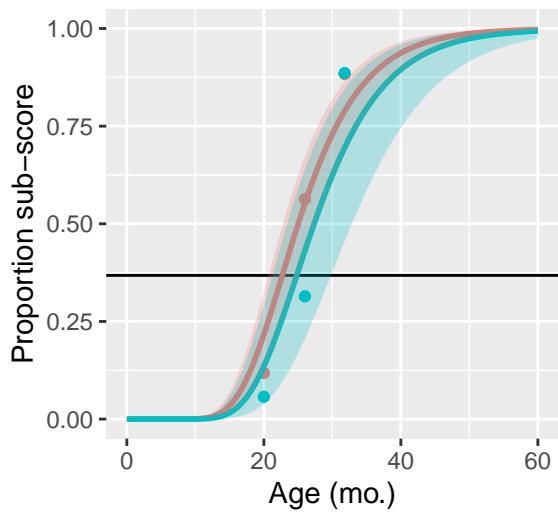
850013



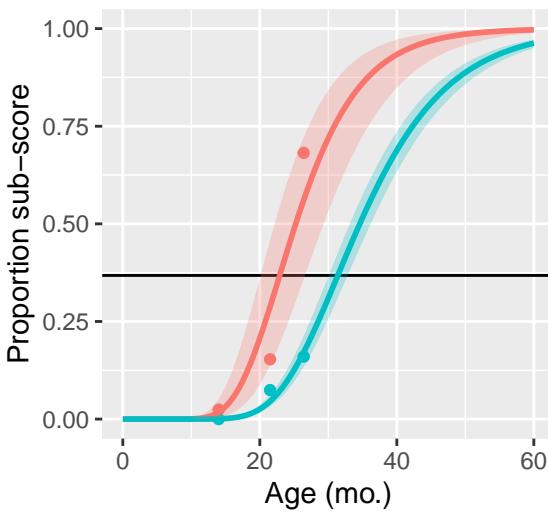
863856



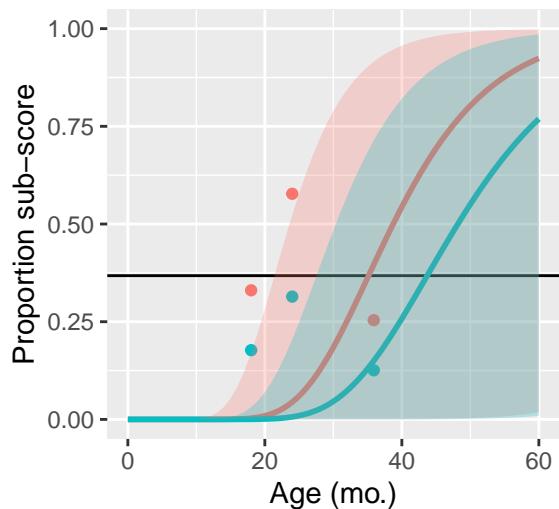
865762



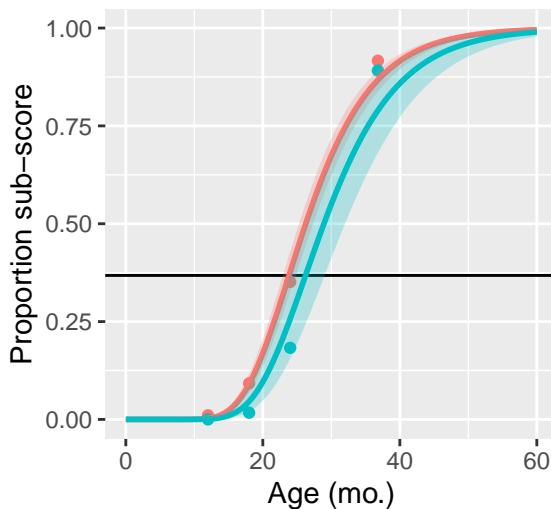
868724



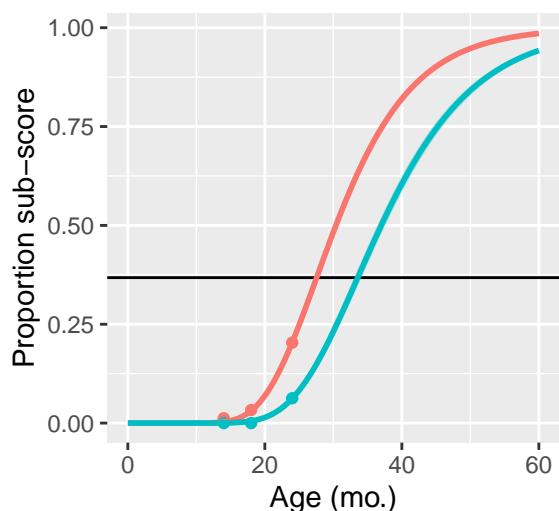
879509



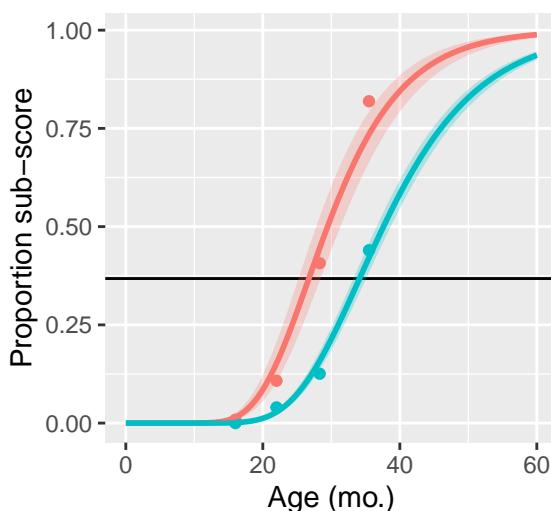
883901



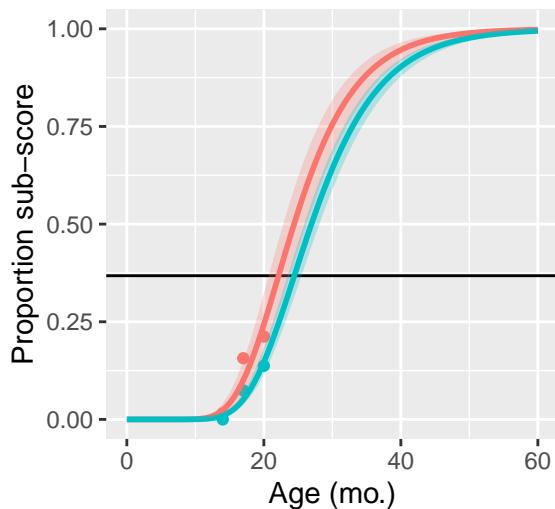
890518



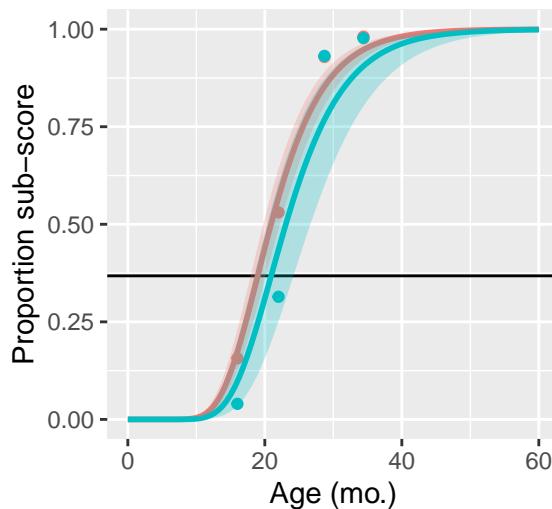
907184



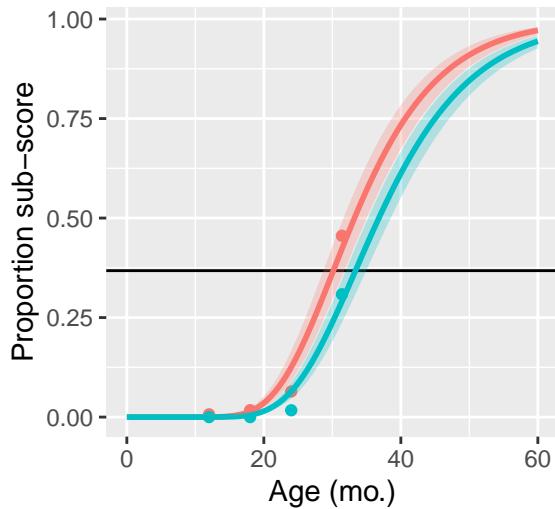
907729



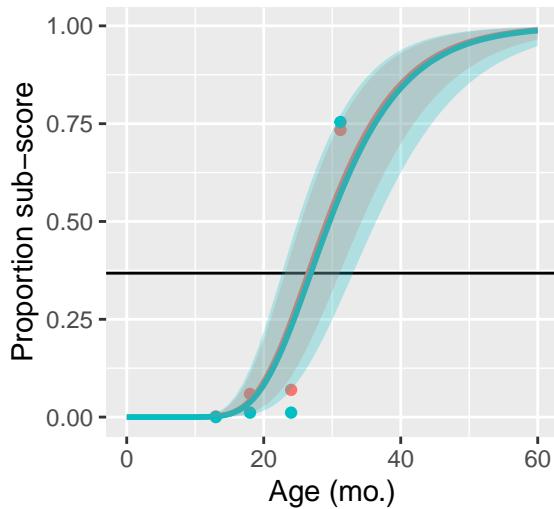
908191



927564

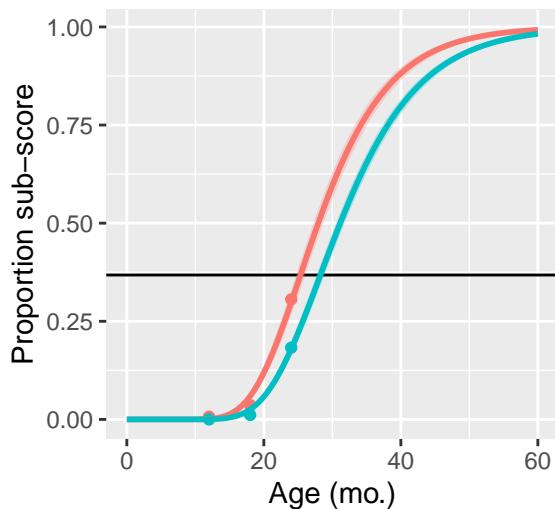


932286

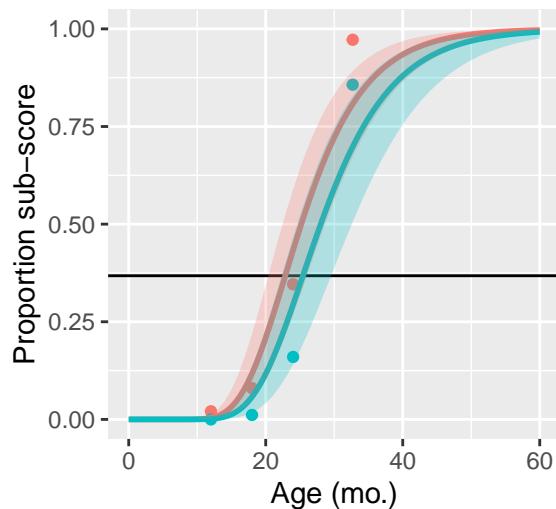


```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")), :  
##   number of iterations exceeded maximum of 500  
## [1] NA
```

939785

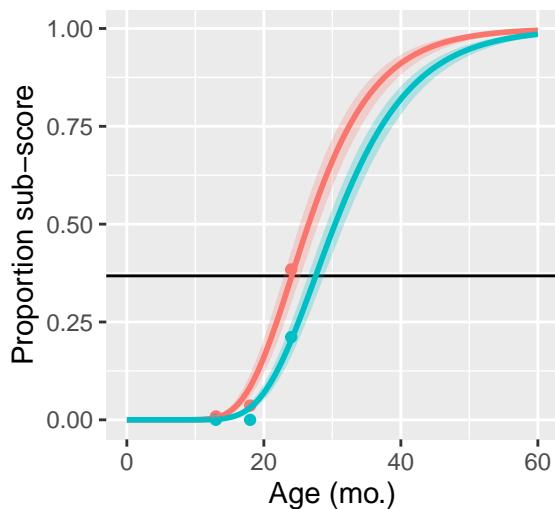


940808

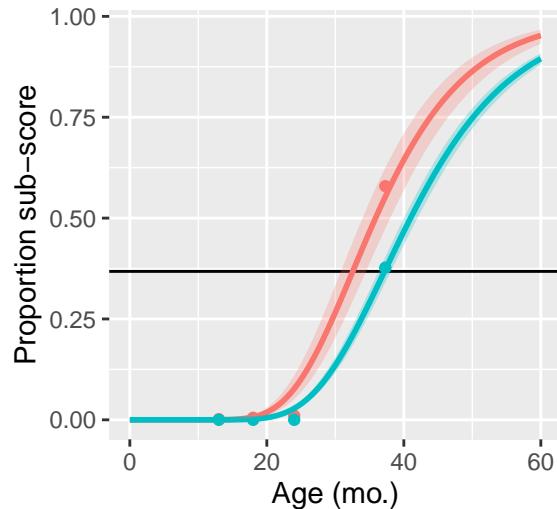


```
## Error in nls(as.formula(paste(response, "~ W_0 * (A / W_0) ^ (1 - exp(-k_g * age))")), :  
##   number of iterations exceeded maximum of 500  
## [1] NA
```

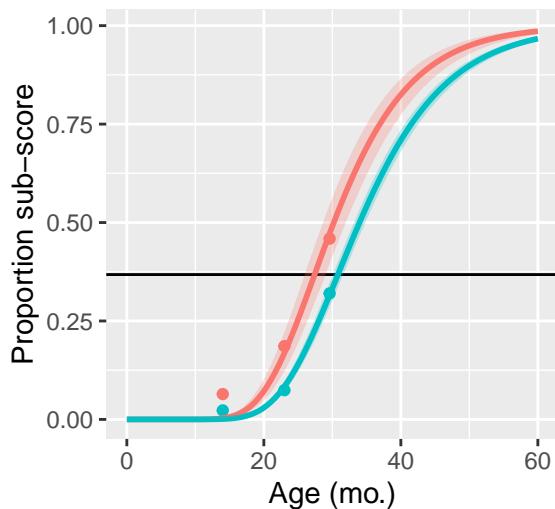
960370



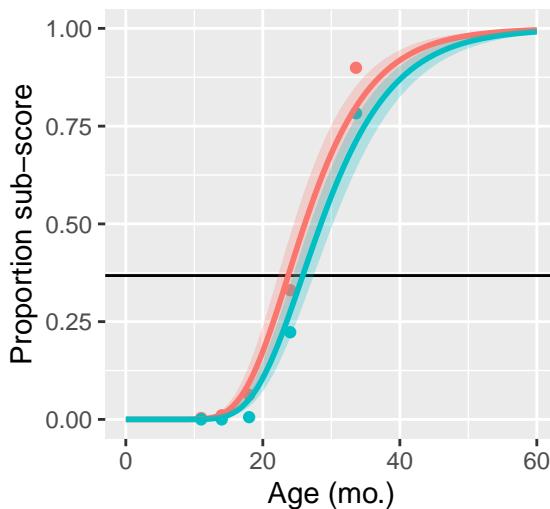
960432



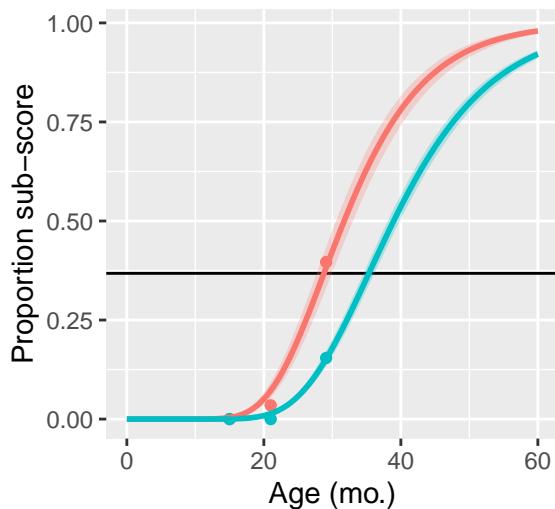
960758



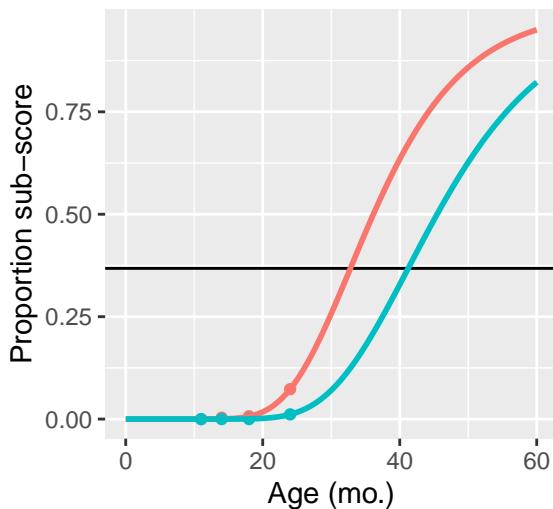
978698



985303

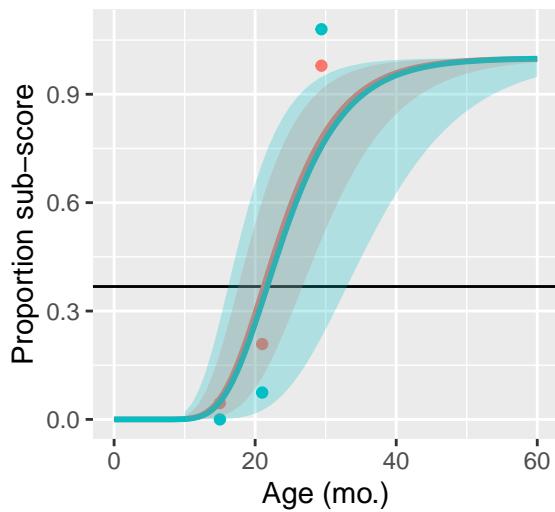


987008



```
## 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
```

999947



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
## 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)

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

## [1] 4.219502
sd(LS.vals$synMlex, na.rm = TRUE)

## [1] 2.895517
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 = -15.559, df = 113, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -4.756779 -3.682226
## sample estimates:
## mean of the differences
## -4.219502

```

All curves

```

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

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

png("plots/lexsyn-lines.png", width = 5, height = 5, units = "in", res = 300)

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

invisible( dev.off() )

```

Package info

```
version

## 
## platform      x86_64-w64-mingw32
## arch         x86_64
## os          mingw32
## system      x86_64, mingw32
## status
## major        4
## minor       0.0
## year        2020
## month       04
## day         24
## svn rev     78286
## language    R
## version.string R version 4.0.0 (2020-04-24)
## nickname    Arbor Day
installed.packages()[names(sessionInfo()$otherPkgs), "Version"]

##      educate growthcurver   gridExtra   ggrepel   scales   plyr
## "0.1.0.2"    "0.3.0"      "2.3"      "0.8.2"   "1.1.1"   "1.8.6"
##     forcats    stringr     dplyr     purrr     readr     tidyR
## "0.5.0"      "1.4.0"      "0.8.5"    "0.3.4"   "1.3.1"   "1.0.3"
##      tibble    ggplot2     tidyverse
## "3.0.1"      "3.3.1"      "1.3.0"
```

References

Tjørve, Kathleen M. C., and Even Tjørve. 2017. “The Use of Gompertz Models in Growth Analyses, and New Gompertz-Model Approach: An Addition to the Unified-Richards Family.” *PLoS ONE* 12 (6). doi:10.1371/journal.pone.0178691.