Supplementary file 1 (revised)

A Prayle on behalf of the group authors.

23-12-2020b

Association between treatments and short term-biochemical and clinical outcomes in PIMS-TS

This document includes the code and the output from that code for the PIMS-TS data analysis. Herein follows a narrative description of the analysis undertaken (which elaborates on the methods and results presented in the paper), followed by the code and output from a R session.

Stats section - methods

The aim of this analysis was to explore whether there is evidence from the early patients treated with and without 3 main groups of treatments had impact on the trajectories of CFP, lymphocyte count and CRP. We note that this observational data collection was undertaken before a treatment orthodoxy developed which led to most patients receiving these treatments. Now that the majority of patients nationally are commenced on one or more of these treatments, it would be difficult in a larger observational study to have any patients on no immunomodulatory therapies (and therefore no control group).

We considered other approaches to the analysis, but in the end opted for a series of 3 simple linear mixed effects models (LMEMs), as we have only 78 participants in the study. We note that we may be under-powered to detect any treatment effect, but we are limited by the available data. We considered if we could undertake a propensity score matching study, but we have 3 treatments and therefore 8 treatment combinations. We also considered a repeated measures ANOVA, but we were more concerned with the trajectories of the response variables than the mean levels for each participant, and we had missing data. We therefore opted for LMEMs.

To compare the effects of treatments (steroids, IVIG or immunomodulation; the explanatory variables) on the trajectories of the patients' CRP, lymphocyte and platelet counts over the first 5 days of the admission (the response variables) we used linear mixed effects models (LMEM), which allowed modeling of the multiple measurements per participant. We defined day 1 from the moment of admission to 24 hours after admission.

As the response variables tended not to follow a straight line, we undertook orthogonal polynomial transformation of the day number of admission, to allow the responses to follow more complex patterns. Following the nomenclature of the lme4 package, we modeled the treatments as fixed effects which were additive (i.e. the treatments as individual terms without interaction terms), and the slope and intercept of each patient as random effects. To attempt to adjust for baseline differences between untreated and treated populations, we fit a second set of models which also included age, sex and need for inotropes as fixed effects parameters.

We compared the model with and without all three treatments with both a calculated likelihood ratio test (considering $p < 0.05$ to be statistically significant), and a semi parametric bootstrap of the log-likelihood for the models. To further explore the effect of the treatments upon the outcome measures, we compared the fitted values from the model for children who received no treatments, vs the model estimate for a patient receiving each one of the three treatments graphically by plotting the fitted values and 95% confidence

intervals (see below) from the model. For the adjusted models, we compared fitted values for the 'average' child, who was male, had the median age of 11 years, and was on inotropes.

There are multiple methods for calculating a 95% confidence interval to fitted values linear mixed effects models, and we compared three approaches. In the simplest approach, we computed the standard errors of the fitted values using the combinations of explanatory variables for which the predictions were computed and the variance–covariance matrix of the estimated parameters fitted model, (i.e. we used the approach usually taken for generalised linear models and ignored the random effects terms). In the second approach and third approaches, we used the predictionInterval (which calculates a prediction interval rather than a confidence interval) function from the package merTools, and bootMer from lem4 respectively (see supplementary information). As the first approach gave the narrowest interval, we proceeded with this with caution, but noted that it is the least conservative of the intervals. The narrowest interval was chosen as this would be most sensitive to differences in treatment effects. To elaborate on our reasoning, we wanted to understand if there was any evidence of differences between groups in the trajectories of the response variable, and as the intention of this study was to generate hypotheses to test in future studies, we were prepared to accept the risk of type I error.

We used R (version 4.0.2) with the packages lme4 (version 1.1-23) and merTools (0.5.2).

Stats section - results

We fitted linear mixed effects models LMEMs for all three response variables (CRP, platelets and lymphocytes), with the treatments as fixed effects, and a random slope and intercept for each patient. As our data had fairly low numbers, we did not model for interactions between the treatments. We fit a second model which also included age, sex and need for baseline inotropes. Incorporating treatments did not improve model fit (by either likelihood ratio test, or a bootstrap likelihood ratio test) in either the model with or without baseline covariate adjustment, providing no evidence that the treatments were associated with the outcome measures of patient trends in CRP, platelets or lymphocyte count in the first 5 days of therapy.

The plots of the fitted model predictions for children with and without any of the three treatments show considerable overlap between 95% confidence intervals, for the mean fitted values for participants on no treatment compared with treatment with either steroids, IVIG or immunomodulators (figure 1 in the paper). We therefore found no evidence of differences in the outcome variables studied.

```
##
##  0  1
## 21 57
```

Baseline characteristics table

This section calculates the demographics for the table 1.

```
# we use the data3 dataframe as this has all the variables in it

data3$treatment_group <- "one_or_two"
data3$treatment_group[which(data3$STEROIDS + data3$IVIG + data3$IMMUNOMODULATION == 0)] <- "none"
data3$treatment_group[which(data3$STEROIDS + data3$IVIG + data3$IMMUNOMODULATION == 3)] <- "three"
kable(table(data3$treatment_group))
```

Var1	Freq
none	12
one_or_two	52
three	14

```r
print("Summary of baseline age")
```

```
## [1] "Summary of baseline age"
```

```r
by(as.numeric(data3$'Patient\'s Age (years)'),  INDICES = data3$treatment_group, FUN = summary)
```

```
## data3$treatment_group: none
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    8.00   10.28   10.63   15.00   16.00
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.25    7.00   10.75   10.10   13.00   17.00
## ------------------------------------------------------------
## data3$treatment_group: three
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    9.50   12.00   11.14   14.75   17.20
```

```r
kruskal.test(data3$'Patient\'s Age (years)' ~ data3$treatment_group)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  data3$"Patient's Age (years)" by data3$treatment_group
## Kruskal-Wallis chi-squared = 2.8296, df = 2, p-value = 0.243
```

```r
data3$'Patient\'s Sex'[which(data3$'Patient\'s Sex' == "male")] <- "Male"
kable(table(data3$'Patient\'s Sex'))
```

Var1	Freq
Female	26
Male	52

```r
by(data3$'Patient\'s Sex', INDICES = data3$treatment_group, FUN = table)
```

```
## data3$treatment_group: none
##
## Female   Male
##      7      5
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
##
## Female   Male
##     17     35
## ------------------------------------------------------------
## data3$treatment_group: three
##
## Female   Male
##      2     12
```

```r
print("Proportion female")
```

```
## [1] "Proportion female"
```

```r
by(data3$'Patient\'s Sex', INDICES = data3$treatment_group, FUN = function(x){length(which(x == "Female
```

```
## data3$treatment_group: none
## [1] 0.5833333
```

```
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
## [1] 0.3269231
## ------------------------------------------------------------
## data3$treatment_group: three
## [1] 0.1428571
```

```
prop.test(x = by(data3$'Patient\'s Sex', INDICES = data3$treatment_group, FUN = function(x){length(whic
          n = by(data3$'Patient\'s Sex', INDICES = data3$treatment_group, FUN = function(x){length(x)})
)
```

```
## Warning in prop.test(x = by(data3$"Patient's Sex", INDICES =
## data3$treatment_group, : Chi-squared approximation may be incorrect
```

```
##
##  3-sample test for equality of proportions without continuity
##  correction
##
## data:  by(data3$"Patient's Sex", INDICES = data3$treatment_group, FUN = function(x) {     length(whi
## X-squared = 5.6703, df = 2, p-value = 0.05871
## alternative hypothesis: two.sided
## sample estimates:
##    prop 1    prop 2    prop 3
## 0.5833333 0.3269231 0.1428571
```

```
print("Invasive ventilation")
```

```
## [1] "Invasive ventilation"
```

```
by(ifelse(data3$"Respiratory: Invasive ventilation" == "Respiratory: Invasive ventilation", T, F)
   ,  INDICES = data3$treatment_group, FUN = table)
```

```
## data3$treatment_group: none
##
## FALSE  TRUE
##     7     5
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
##
## FALSE  TRUE
##    25    27
## ------------------------------------------------------------
## data3$treatment_group: three
##
## FALSE  TRUE
##    10     4
```

```
print("proportion invasively ventilated")
```

```
## [1] "proportion invasively ventilated"
```

```
by(data3$'Respiratory: Invasive ventilation', INDICES = data3$treatment_group, FUN = function(x){length
```

```
## data3$treatment_group: none
## [1] 0.4166667
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
## [1] 0.5192308
```

```
## ------------------------------------------------------------
## data3$treatment_group: three
## [1] 0.2857143
```

```
prop.test(x = by(data3$'Respiratory: Invasive ventilation', INDICES = data3$treatment_group, FUN = funct
          n = by(data3$'Respiratory: Invasive ventilation', INDICES = data3$treatment_group, FUN = funct
)
```

```
##
##  3-sample test for equality of proportions without continuity
##  correction
##
## data:  by(data3$"Respiratory: Invasive ventilation", INDICES = data3$treatment_group, FUN = function
## X-squared = 2.5351, df = 2, p-value = 0.2815
## alternative hypothesis: two.sided
## sample estimates:
##    prop 1    prop 2    prop 3
## 0.4166667 0.5192308 0.2857143
```

```
print("Inotropes")
```

```
## [1] "Inotropes"
```

```
by(ifelse(data3$"Cardiac: inotropes" == "Cardiac: inotropes", T, F)
   ,  INDICES = data3$treatment_group, FUN = table)
```

```
## data3$treatment_group: none
##
## FALSE  TRUE
##     5     7
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
##
## FALSE  TRUE
##     6    46
## ------------------------------------------------------------
## data3$treatment_group: three
##
## FALSE  TRUE
##     2    12
```

```
by(data3$"Cardiac: inotropes", INDICES = data3$treatment_group, FUN = function(x){length(which(x == "Ca
```

```
## data3$treatment_group: none
## [1] 0.5833333
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
## [1] 0.8846154
## ------------------------------------------------------------
## data3$treatment_group: three
## [1] 0.8571429
```

```
prop.test(x = by(data3$"Cardiac: inotropes", INDICES = data3$treatment_group, FUN = function(x){length(
          n = by(data3$"Cardiac: inotropes", INDICES = data3$treatment_group, FUN = function(x){length(
)
```

```
## Warning in prop.test(x = by(data3$"Cardiac: inotropes", INDICES =
## data3$treatment_group, : Chi-squared approximation may be incorrect
```

```
##
##  3-sample test for equality of proportions without continuity
##  correction
##
## data:  by(data3$"Cardiac: inotropes", INDICES = data3$treatment_group, FUN = function(x) {     lengt
## X-squared = 6.4418, df = 2, p-value = 0.03992
## alternative hypothesis: two.sided
## sample estimates:
##    prop 1    prop 2    prop 3
## 0.5833333 0.8846154 0.8571429
```

```
print("Summary of baseline CRP")
```

```
## [1] "Summary of baseline CRP"
```

```
by(data3$'Maximum CRP.1',  INDICES = data3$treatment_group, FUN = summary)
```

```
## data3$treatment_group: none
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    16.0   122.5   213.0   200.8   268.0   403.0       1
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    54.0   167.8   250.5   249.8   312.8   556.0       6
## ------------------------------------------------------------
## data3$treatment_group: three
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      99     187     292     261     316     416       1
```

```
kruskal.test(data3$'Maximum CRP.1' ~ data3$treatment_group)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  data3$"Maximum CRP.1" by data3$treatment_group
## Kruskal-Wallis chi-squared = 1.9183, df = 2, p-value = 0.3832
```

```
print("Summary of baseline Platelet count")
```

```
## [1] "Summary of baseline Platelet count"
```

```
by(data3$'Maximum platelet count.1',  INDICES = data3$treatment_group, FUN = summary)
```

```
## data3$treatment_group: none
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   62.00   90.75  177.00  162.80  204.50  265.00       2
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    61.0   111.0   161.0   188.8   232.0   658.0       5
## ------------------------------------------------------------
## data3$treatment_group: three
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    63.0   129.0   183.0   175.2   225.0   278.0       1
```

```
kruskal.test(data3$'Maximum CRP.1' ~ data3$treatment_group)
```

```
##
```

```
##  Kruskal-Wallis rank sum test
##
## data:  data3$"Maximum CRP.1" by data3$treatment_group
## Kruskal-Wallis chi-squared = 1.9183, df = 2, p-value = 0.3832
```

```
print("Summary of baseline Lymphocyte count")
```

```
## [1] "Summary of baseline Lymphocyte count"
```

```
by(data3$'Maximum Lymphocyte count.1',  INDICES = data3$treatment_group, FUN = summary)
```

```
## data3$treatment_group: none
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.0100  0.6975  0.7850  0.8770  0.9375  2.0000       2
## ------------------------------------------------------------
## data3$treatment_group: one_or_two
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.120   0.720   1.100   1.552   1.820   7.960       5
## ------------------------------------------------------------
## data3$treatment_group: three
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.2000  0.4100  0.8000  0.7908  0.9000  1.9000       1
```

```
kruskal.test(data3$'Maximum Lymphocyte count.1' ~ data3$treatment_group)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  data3$"Maximum Lymphocyte count.1" by data3$treatment_group
## Kruskal-Wallis chi-squared = 6.9087, df = 2, p-value = 0.03161
```

Overlap between treatments

	No IVIG	IVIG	Sum
No steroids	13	8	21
Steroids	6	51	57
Sum	19	59	78

Propensity score analysis.

The final analysis in the paper uses a hierarchical mixed effects model. We also undertook a propensity matched analysis of the effects of treatment. The goal of our work is to identify any evidence that these children had different outcomes depending upon treatment. Our data are of course observational. A commonly used approach to try to reduce the impact of confounders on the result is to match treated patients with controls. We investigate this using matching with propensity scores. This section reports this approach. We used the MatchIt package for R. Given limited data, we decided to match on the pre=treatment confounding variables of age and sex.

Our outcome variable is the inflammatory marker (CRP, lymphocyte count or platelet count) on day 5. As there are missing values for the outcome variable, we need to do last observation carried forward (which is not ideal, as the CRP is descending from day 3 to 5). We used the genetic matching algorithm (one of the defaults built into MatchIt) as we found that this tended to have better balance between those treated vs those not treated. To investigate for evidence of treatment effect, we compared no treatment to treatment with either one or two of the therapies, or all three treatments. We repeated the analyses for each of the 3 inflammatory markers. We note that this results in multiple statistical testing, but we reasoned that we

wanted to identify any evidence of treatment effect as hypothesis generating for future studies, rather than to make recommendations for treatment.

```
## time to make a function:

pdata <- data3[ , c("ID",
                    "Maximum CRP.3",
                    "Maximum CRP.4",
                    "Maximum CRP.5",
                    "Maximum Lymphocyte count.3",
                    "Maximum Lymphocyte count.4",
                    "Maximum Lymphocyte count.5",
                    "Maximum platelet count.3",
                    "Maximum platelet count.4",
                    "Maximum platelet count.5",
                    "treatment_group",
                    "Patient's Age (years)",
                    "Patient's Sex",
                    "Patient's ethnicity")]

colnames(pdata)[which(colnames(pdata) == "Patient's Age (years)")] <- "age_yrs"
pdata$age_yrs <- as.numeric(pdata$age_yrs)
colnames(pdata)[which(colnames(pdata) == "Patient's Sex")] <- "sex"

get_propensity_score_analysis <- function(outcome_parameter,
                                          mean_or_threshold = "mean",
                                          threshold = NA,
                                          less_or_greater = NA,
                                          treatment_type){
  ## function which takes the pdata, extracts out the rows for the
  ## treatment type, makes the outcome variable,
  ## then computes a propsenity score analysis and reports a row
  ## with sufficient detail for the paper.
  ## treatment type, outcome parameter, n_treat, n_control,
  ## mean_age_treat, mean_age_control
  ## percent_sex_treat, percent_sex_control
  ## result_treat, result_control,

  # here is the function code:
  identified_outcome_parameter <- F
  identified_treatment_type <- F
  # used for internal debugging

  fdata <- pdata
  fdata$outcome <- NA
  if (outcome_parameter == "CRP_mean"){
    identified_outcome_parameter <- T
    fdata$outcome <- pdata$`Maximum CRP.5`
    fdata$outcome[which(is.na(fdata$outcome))] <- pdata$`Maximum CRP.4`[which(is.na(fdata$outcome))]
    fdata$outcome[which(is.na(fdata$outcome))] <- pdata$`Maximum CRP.3`[which(is.na(fdata$outcome))]
  }
  if (outcome_parameter == "lymphocyte_mean"){
    identified_outcome_parameter <- T
    fdata$outcome <- pdata$`Maximum Lymphocyte count.5`
```

```
  fdata$outcome[which(is.na(fdata$outcome))] <- pdata$`Maximum Lymphocyte count.4`[which(is.na(fdata$
  fdata$outcome[which(is.na(fdata$outcome))] <- pdata$`Maximum Lymphocyte count.3`[which(is.na(fdata$
}
if (outcome_parameter == "platelet_mean"){
  identified_outcome_parameter <- T
  fdata$outcome <- pdata$`Maximum platelet count.5`
  fdata$outcome[which(is.na(fdata$outcome))] <- pdata$`Maximum platelet count.4`[which(is.na(fdata$ou
  fdata$outcome[which(is.na(fdata$outcome))] <- pdata$`Maximum platelet count.3`[which(is.na(fdata$ou
}



if(treatment_type == "one_or_two"){
  fdata <- fdata[which(fdata$treatment_group == "none" | fdata$treatment_group == "one_or_two"), ]
  identified_treatment_type <- T
}
if(treatment_type == "three"){
  fdata <- fdata[which(fdata$treatment_group == "none" | fdata$treatment_group == "three"), ]
  identified_treatment_type <- T
}
table(fdata$treatment_group)
fdata$treatment_group_numeric <- ifelse(fdata$treatment_group == "none", 0, 1)

# now the treatment type and the outcome measures are defined, it is time to do the analysis
# remove the missing data rows
fdata <- fdata[-which(is.na(fdata$outcome)), ]
# remove the missing data columns
fdata <- fdata[ , - c(2:10) ]
m.out <- matchit(treatment_group_numeric ~ age_yrs + sex,
                 method = 'genetic',
                 data = fdata)
summary(m.out)
# the genetic matching algorithm worked best
m.data <- match.data(m.out)

# now we do a regression without covariates on the outcome using the weights calculated by MatchIt
# note that is we use exact matching, we get the same answer if we include the covariates or not
# but if we use other matching algorithms such as athe genetic one used here
# the coefficients are slightly different
# we use lm rather than zelig for ease of access to the results for later tabulation
prop_model <- lm(outcome ~ treatment_group_numeric,
                 weights = weights,
                 data = m.data)


summary(prop_model)



fresult <- c(outcome_parameter,
             mean_or_threshold,
             treatment_type,
```

```
                summary(m.out)$nn[2, 2], # n matched treated
                summary(m.out)$nn[2, 1], # n matched control
                summary(m.out)$sum.matched[2, 1], # age treated
                summary(m.out)$sum.matched[2, 2], # age control
                summary(m.out)$sum.matched[3, 1], # female treated
                summary(m.out)$sum.matched[3, 2], # female control
                summary(prop_model)$coefficients[1, 1], # control mean
                summary(prop_model)$coefficients[2, 1], # treatment effect
                summary(prop_model)$coefficients[2, 4] # treatment effect p value
  )


  return(fresult)
}


result_table <- get_propensity_score_analysis(outcome_parameter = "CRP_mean",
                                              mean_or_threshold = "mean",
                                              treatment_type = "one_or_two" )
result_table <- rbind(result_table,
                      get_propensity_score_analysis(outcome_parameter = "CRP_mean",
                                                    mean_or_threshold = "mean",
                                                    treatment_type = "three" ))
result_table <- rbind(result_table,
                      get_propensity_score_analysis(outcome_parameter = "lymphocyte_mean",
                                                    mean_or_threshold = "mean",
                                                    treatment_type = "one_or_two" ))
result_table <- rbind(result_table,
                      get_propensity_score_analysis(outcome_parameter = "lymphocyte_mean",
                                                    mean_or_threshold = "mean",
                                                    treatment_type = "three" ))
result_table <- rbind(result_table,
                      get_propensity_score_analysis(outcome_parameter = "platelet_mean",
                                                    mean_or_threshold = "mean",
                                                    treatment_type = "one_or_two" ))
result_table <- rbind(result_table,
                      get_propensity_score_analysis(outcome_parameter = "platelet_mean",
                                                    mean_or_threshold = "mean",
                                                    treatment_type = "three" ))
colnames(result_table) <- c("outcome_parameter",
                            "outcome_statistic",
                            "treatment_group",
                            "matched_n_treat",
                            "matched_n_control",
                            "matched_mean_age_treated",
                            "matched_mean_age_control",
                            "matched_proportion_female_treated",
                            "matched_proportion_female_control",
                            "control_mean",
                            "treatment_effect",
                            "treatment_effect_p_value"
)
result_table <- as.data.frame(result_table)
```

```r
rownames(result_table) <- c()
for(i in 4:12){
  result_table[, i] <- as.numeric(result_table[, i])
  result_table[, i] <- round(result_table[, i], 2)
}
```

outcome_parameter	outcome_statistic	treatment_group	matched_n_treat	matched_n_control
CRP_mean	mean	one_or_two	49	9
CRP_mean	mean	three	14	6
lymphocyte_mean	mean	one_or_two	47	9
lymphocyte_mean	mean	three	14	6
platelet_mean	mean	one_or_two	47	9
platelet_mean	mean	three	14	6

outcome_parameter	outcome_statistic	treatment_group	matched_mean_age_treated	matched_mean_age_control
CRP_mean	mean	one_or_two	10.03	10.52
CRP_mean	mean	three	11.14	11.30
lymphocyte_mean	mean	one_or_two	9.79	10.47
lymphocyte_mean	mean	three	11.14	11.30
platelet_mean	mean	one_or_two	9.79	10.47
platelet_mean	mean	three	11.14	11.30

outcome_parameter	outcome_statistic	treatment_group	matched_proportion_female_treated	matched_proportion_female_control
CRP_mean	mean	one_or_two	0.33	0.29
CRP_mean	mean	three	0.14	0.14
lymphocyte_mean	mean	one_or_two	0.34	0.32
lymphocyte_mean	mean	three	0.14	0.14
platelet_mean	mean	one_or_two	0.34	0.32
platelet_mean	mean	three	0.14	0.14

outcome_parameter	outcome_statistic	treatment_group	control_mean	treatment_effect	treatment_effect_p_value
CRP_mean	mean	one_or_two	89.65	9.97	0.77
CRP_mean	mean	three	121.00	-37.07	0.38
lymphocyte_mean	mean	one_or_two	2.78	2.41	0.26
lymphocyte_mean	mean	three	2.99	0.07	0.96
platelet_mean	mean	one_or_two	236.56	11.15	0.89
platelet_mean	mean	three	139.16	107.42	0.17

In this table, the 6 analyses done are presented. The **control_mean** represents the mean outcome for the untreated group. The **treatment_effect** represents the mean difference for the treated group. The p value column indicates that there is no evidence of a treatment effect.

At this stage, we could have stopped the analysis, but we were concerned that the risk of missing a potential treatment effect. We note that the propensity score approach removes data, and the analysis looked at the mean level of the blood tests on day 5, rather than a potentially more useful model which incorporates the trajectory of the blood test. Therefore, we proceeded with the LMEM analysis.

Mixed effects model

Missing data.

The response variables (CRP, lymphocytes and platelet counts) were not measured on every day during the first 5 days. Therefore, we remove those with only one measurement (as this prevents calculation of a trajectory).

Initially we analyse the CRP data (then repeat this for the other outcomes). This table shows how many observations of CRP we have for each of the 78 participants, out of a maximum 5.

```
## [1] "CRP completness"
```

Number of CRP measurements per patient	Frequency
0	3
1	2
2	1
3	13
4	38
5	21
Sum	78

	Number of CRP measurements	total n
Maximum CRP.1	70	78
Maximum CRP.2	68	78
Maximum CRP.3	66	78
Maximum CRP.4	60	78
Maximum CRP.5	36	78

```
## [1] "Lymphocyte completeness"
```

Number of lymph measurements per patient	Frequency
0	5
1	2
2	3
3	7
4	34
5	27
Sum	78

	Number of lymph measurements	total n
Maximum Lymphocyte count.1	70	78
Maximum Lymphocyte count.2	64	78
Maximum Lymphocyte count.3	66	78
Maximum Lymphocyte count.4	65	78
Maximum Lymphocyte count.5	35	78

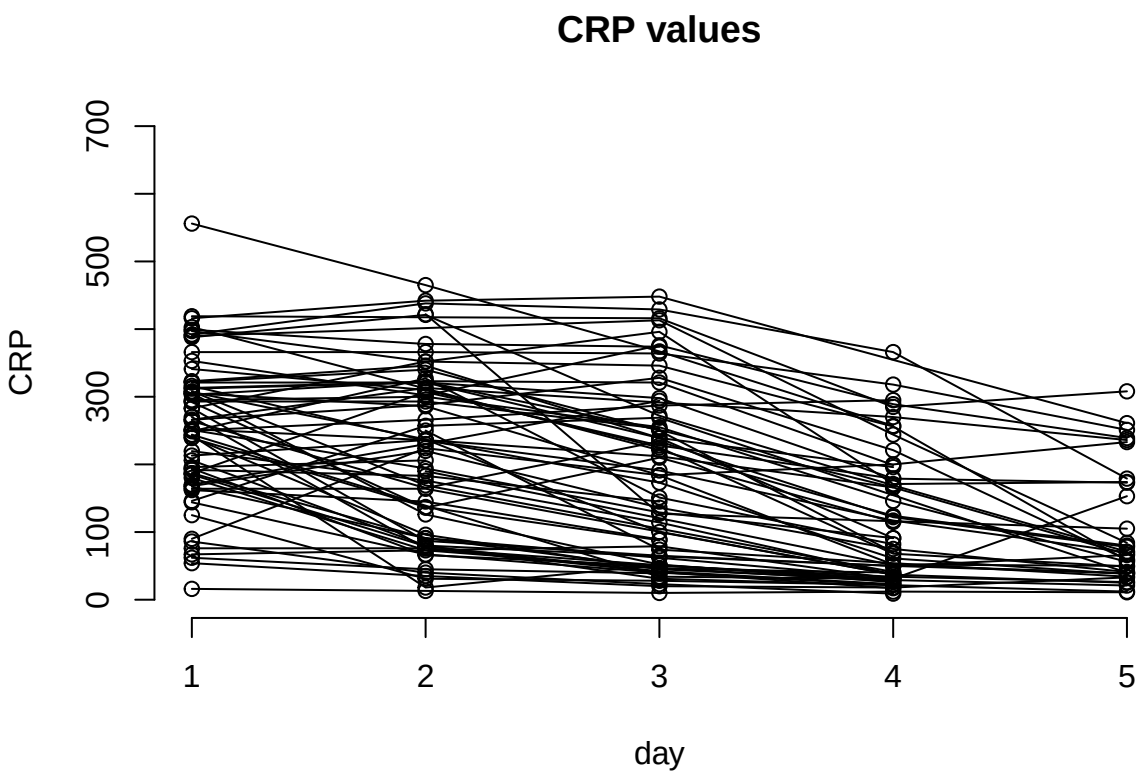
```
## [1] "Platelet count completeness"
```

Number of plat measurements per patient	Frequency
0	5
1	2
2	3
3	6
4	34
5	28
Sum	78

	Number of plat measurements	total n
Maximum platelet count.1	70	78
Maximum platelet count.2	64	78
Maximum platelet count.3	67	78
Maximum platelet count.4	65	78
Maximum platelet count.5	36	78

Graphical exploration of the CRP data

After some adjustment of the data (converting from wide to long format, we undertake a graphical exploration. In this graph, the lines join patients, and one can see the trajectory of CRP decline that occurs in the first 5 days of the disease.



Linear mixed effects models.

We now undertake a series of LMEMs. We start with a simple model:

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula: max_CRP ~ day + (1 + day | ID)
##    Data: long_data_CRP
##
##      AIC      BIC   logLik deviance df.resid
##   3274.0   3295.8  -1631.0   3262.0      277
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.02324 -0.51677 -0.07946  0.50368  2.41890
```

```
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 13692.0  117.01
##           day           270.8   16.46   -0.63
##  Residual              2779.9   52.72
## Number of obs: 283, groups:  ID, 69
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  304.108     15.949   19.07
## day          -44.957      3.202  -14.04
##
## Correlation of Fixed Effects:
##     (Intr)
## day -0.679
```

We note that the average trajectory may not follow a straight line, and therefore we use the code.poly function to fit orthogonal polynomials. code.poly() which was written by Matt Winn, revised by Dan Miran. This code is reproduced here unchanged, with references.

```
code.poly <- function(df=NULL, predictor=NULL, poly.order=NULL, orthogonal=TRUE, draw.poly=FALSE){
  require(reshape2)
  require(ggplot2)
  #' Build polynomial predictor variables
  #'
  #' Takes a data frame, name of predictor variable, and polynomial order. Creates polynomial-transform
  #'
  #' @param df data frame, should not contain any variables called "predictor"
  #' @param predictor string name of predictor variable
  #' @param poly.order integer order of polynomial to be created
  #' @param orthogonal logical value indicating whether polynomial should be orthogonal (default) or na
  #' @param draw.poly logical value indicating whether to create a graph showing tranformed polynomial
  #' @return Returns a data frame containing the original data and at least two new columns: "predictor
  #' @examples
  #' WordLearnEx.gca <- code.poly(df=WordLearnEx, predictor="Block", poly.order=2)
  #' Az.gca <- code.poly(df=Az, predictor="Time", poly.order=2, orthogonal=FALSE)
  #' @section Contributors:
  #' Originally written by Matt Winn \url{http://www.mattwinn.com/tools/R_add_polynomials_to_df.html} a
  # Codes raw or orthogonal polynomial transformations of a predictor variable
  # be sure to not have an actual variable named "predictor" in your data.frame
  # ultimately adds 2 or more columns, including:
  # (predictor).Index, and a column for each order of the polynomials
  #
  # Written by Matt Winn: http://www.mattwinn.com/tools/R_add_polynomials_to_df.html
  # Revised by Dan Mirman (11/3/2014):
  #   - call to poly now uses predictor.index and 1:max instead of unique() to deal with out-of-order t
  #   - combined indexing and alignment with original data, mainly to avoid problems when poly.order==1
  # Revised by Matt Winn (2/12/2015)
  #   - uses `[` instead of `$` to use variable name to extract column
  #   - computes polynomial on unique(sort(predictor.vector))
  #       rather than 1:max(predictor.vector)
  #       to accomodate non-integer predictor (e.g. time) levels
  #   - Accomodates missing/unevenly-spaced time bins
  #       by indexing each sorted unique time bin and using the index to extract
```

```
#       the polynomial value

#=========================================================#
# convert choice for orthogonal into choice for raw
raw <- (orthogonal-1)^2

# make sure that the declared predictor is actually present in the data.frame
if (!predictor %in% names(df)){
  warning(paste0(predictor, " is not a variable in your data frame. Check spelling and try again"))
}

# Extract the vector to be used as the predictor
predictor.vector <- df[,which(colnames(df)==predictor)]

# create index of predictor (e.g. numbered time bins)
# the index of the time bin will be used later as an index to call the time sample
predictor.indices <- as.numeric(as.factor(predictor.vector))

df$temp.predictor.index <- predictor.indices

#create x-order order polys (orthogonal if not raw)
predictor.polynomial <- poly(x = unique(sort(predictor.vector)),
                             degree = poly.order, raw=raw)

# use predictor index as index to align
# polynomial-transformed predictor values with original dataset
# (as many as called for by the polynomial order)
df[, paste("poly", 1:poly.order, sep="")] <-
  predictor.polynomial[predictor.indices, 1:poly.order]

# draw a plot of the polynomial transformations, if desired
if (draw.poly == TRUE){
  # extract the polynomials from the df
  df.poly <- unique(df[c(predictor, paste("poly", 1:poly.order, sep=""))])

  # melt from wide to long format
  df.poly.melt <- melt(df.poly, id.vars=predictor)

  # Make level names intuitive
  # don't bother with anything above 6th order.
  levels(df.poly.melt$variable)[levels(df.poly.melt$variable)=="poly1"] <- "Linear"
  levels(df.poly.melt$variable)[levels(df.poly.melt$variable)=="poly2"] <- "Quadratic"
  levels(df.poly.melt$variable)[levels(df.poly.melt$variable)=="poly3"] <- "Cubic"
  levels(df.poly.melt$variable)[levels(df.poly.melt$variable)=="poly4"] <- "Quartic"
  levels(df.poly.melt$variable)[levels(df.poly.melt$variable)=="poly5"] <- "Quintic"
  levels(df.poly.melt$variable)[levels(df.poly.melt$variable)=="poly6"] <- "Sextic"

  # change some column names for the output
  colnames(df.poly.melt)[colnames(df.poly.melt) == "variable"] <- "Order"

  poly.plot <- ggplot(df.poly.melt, aes(y=value, color=Order))+
    aes_string(x=predictor)+
    geom_line()+
```

```
      xlab(paste0(predictor, " (transformed polynomials)"))+
      ylab("Transformed value")+
      scale_color_brewer(palette="Set1")+
      theme_bw()

    print(poly.plot)
  }

  # restore correct column names
  colnames(df)[colnames(df) == "temp.predictor.index"] <- paste0(predictor,".Index")
  return(df)
}
```

Then we investigate if orthogonal polynomial regression gives better fit.

```
long_data_CRP <- code.poly(df=long_data_CRP, predictor="day", poly.order=3, orthogonal=T)
p1  <- lmer(max_CRP ~ (poly1 + poly2 + poly3) + (1 + day | ID),
            data = long_data_CRP,
            REML = F)
summary(p1)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula: max_CRP ~ (poly1 + poly2 + poly3) + (1 + day | ID)
##    Data: long_data_CRP
##
##      AIC      BIC   logLik deviance df.resid
##   3272.0   3301.1  -1628.0   3256.0      275
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.18383 -0.57188 -0.02669  0.50500  2.35873
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 13845.9  117.67
##           day           289.8   17.02   -0.63
##  Residual              2657.2   51.55
## Number of obs: 283, groups:  ID, 69
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  168.248     11.851  14.197
## poly1       -145.078     10.473 -13.852
## poly2        -13.059      7.699  -1.696
## poly3         10.441      7.142   1.462
##
## Correlation of Fixed Effects:
##       (Intr) poly1  poly2
## poly1 -0.077
## poly2  0.063  0.236
## poly3  0.026  0.120  0.184
```

```
anova(m0, p1)
```

```
## Data: long_data_CRP
```

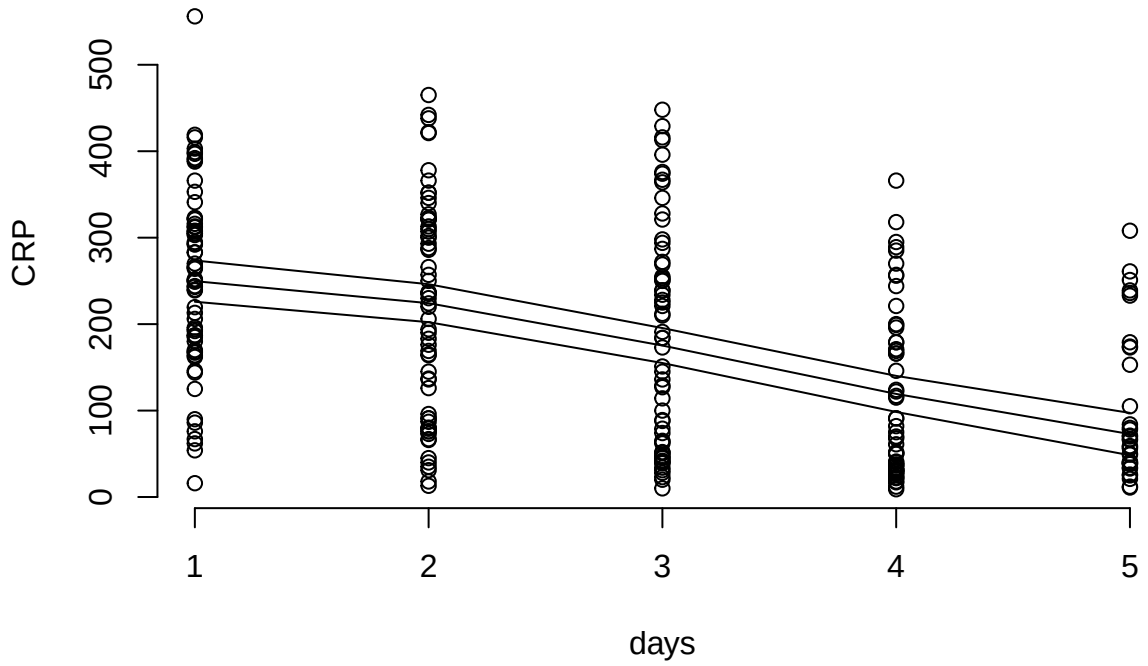
```
## Models:
## m0: max_CRP ~ day + (1 + day | ID)
## p1: max_CRP ~ (poly1 + poly2 + poly3) + (1 + day | ID)
##    npar  AIC    BIC logLik deviance  Chisq Df Pr(>Chisq)
## m0    6 3274 3295.8  -1631     3262
## p1    8 3272 3301.1  -1628     3256 5.9808  2    0.05027 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It does look like the polynomial regression is a better fit. And so we use this approach. We construct 95% confidence intervals for the average from the model.

```
# ~prediction and 95% confidence intervals
poly_data <- expand.grid(
  day = c(1:5)
)
# need to add a step to compute the orthogonal polynomials
poly_data <- code.poly(df=poly_data, predictor="day", poly.order=3, orthogonal=T)
# this is the predicted mean for just the mean of the polynomial regression
poly_data$predict <- predict(p1, newdata = poly_data, re.form = NA)
mm <- model.matrix(~ poly1 + poly2 + poly3,
                   poly_data)
vcov.m <- vcov(p1)
vars <- mm %*% vcov.m %*% t(mm)
sds <- sqrt(diag(vars))
z.val <- qnorm(1 - (1 - 0.9)/2)
poly_data$LoCI <- poly_data$predict - z.val * sds
poly_data$HiCI <- poly_data$predict + z.val * sds

kable(poly_data)
```

day	day.Index	poly1	poly2	poly3	predict	LoCI	HiCI
1	1	-0.6324555	0.5345225	-0.3162278	249.72112	225.85949	273.5827
2	2	-0.3162278	-0.2672612	0.6324555	224.21913	202.27512	246.1631
3	3	0.0000000	-0.5345225	0.0000000	175.22792	155.00026	195.4556
4	4	0.3162278	-0.2672612	-0.6324555	119.25651	98.40136	140.1117
5	5	0.6324555	0.5345225	0.3162278	72.81391	48.32102	97.3068

```
# plot of this
plot(x = long_data_CRP$day,
     y = long_data_CRP$max_CRP,
     bty = "n",
     ylab = "CRP",
     xlab = 'days')
lines(x = poly_data$day,
      y = poly_data$predict)
lines(x = poly_data$day,
      y = poly_data$LoCI)
lines(x = poly_data$day,
      y = poly_data$HiCI)
```



This then shows the approach which gives a fitted curve for this model, and 95% confidence intervals. The next step is to fit a model which includes the treatments that we are interested in. The individual subjects are added as random effects of both the slope and intercept, and the treatments as fixed effects. We use Nelder_Mead, as this algorithm tended to converge.

```
p2  <- lmer(max_CRP ~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) + (1 + day | ID),
            data = long_data_CRP,
            REML = F,
            control = lmerControl(optimizer ="Nelder_Mead"))
summary(p2)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula:
## max_CRP ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
##     (1 + day | ID)
##    Data: long_data_CRP
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   3288.5   3361.4  -1624.3   3248.5      263
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.16254 -0.54106 -0.03453  0.49685  2.43770
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
```

```
##  ID       (Intercept) 13740.0  117.22
##           day           284.6   16.87   -0.63
##  Residual              2565.3   50.65
## Number of obs: 283, groups:  ID, 69
##
## Fixed effects:
##                          Estimate Std. Error t value
## (Intercept)              183.4311    26.7217   6.865
## poly1                   -122.9456    23.0737  -5.328
## poly2                    -12.3292    16.5145  -0.747
## poly3                     24.6696    16.3361   1.510
## IVIG                      -0.2183    33.8784  -0.006
## STEROIDS                 -21.8395    33.8412  -0.645
## IMMUNOMODULATION          14.1203    30.6690   0.460
## poly1:IVIG                35.6090    29.6137   1.202
## poly1:STEROIDS           -59.3195    31.3773  -1.891
## poly1:IMMUNOMODULATION   -11.4173    26.1697  -0.436
## poly2:IVIG                25.7067    21.6718   1.186
## poly2:STEROIDS           -22.4594    22.9212  -0.980
## poly2:IMMUNOMODULATION    -8.6046    19.1356  -0.450
## poly3:IVIG               -12.8923    20.2454  -0.637
## poly3:STEROIDS            -8.0505    20.8660  -0.386
## poly3:IMMUNOMODULATION    12.9860    17.5488   0.740

##
## Correlation matrix not shown by default, as p = 16 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```
anova(p1, p2)
```

```
## Data: long_data_CRP
## Models:
## p1: max_CRP ~ (poly1 + poly2 + poly3) + (1 + day | ID)
## p2: max_CRP ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
## p2:     (1 + day | ID)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## p1    8 3272.0 3301.1 -1628.0   3256.0
## p2   20 3288.5 3361.4 -1624.3   3248.5 7.4564 12      0.826
```

Adding treatments doesn't improve the model fit, at least by the criterion of log-likelihood. We can also compare models using the bootMer function. If the 95% CI crosses 0, then there is no evidence of improved model fit with a bootstrapped LRT.

```
# using bootMer to compute 1000 bootstrapped log-likelihood
b1 <- bootMer(p1, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)
b2 <- bootMer(p2, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)

# the 1000 bootstrapped LRT - now  comparing p1 with p2
lrt.b <- -2 * b1$t + 2 * b2$t
# plot
quant <- quantile(lrt.b, probs = c(0.025, 0.975))
print("comparing p1 with p2")
```

```
## [1] "comparing p1 with p2"
```

```
print(quant)
```

```
##     2.5%    97.5%
## -52.0662  84.2311
```

Again, this model doesn't appear much better by the bootstrapped LRT test.

More complex models including baseline covariates

More complex models were suggested by the peer reviewers. We discussed on a conference call with the core team on 22/12/2020, and a priori the baseline age, sex, inotrope prescription and CRP were considered to be potentially useful covariates to include. We explore this.

```
# a reminder of the previous models explored:
# p1 <- max_CRP ~ (poly1 + poly2 + poly3) + (1 + day | ID)
# p2 <- max_CRP ~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) + (1 + day | ID)

# p3 will be p1 but with covariates, which we will model as fixed effect

colnames(long_data_CRP)[85] <- "baseline_CRP"

long_data_CRP$age_yrs <- as.numeric(long_data_CRP$age_yrs)



p3 <- lmer(max_CRP ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes + (1 + day | ID),
            data = long_data_CRP,
            REML = F,
            control = lmerControl(optimizer ="Nelder_Mead"))
# note that including baseline CRP in the model introduces singularities, therefore this is omitted

p4 <- lmer(max_CRP ~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) + age_yrs + sex + 
            data = long_data_CRP,
            REML = F,
            control = lmerControl(optimizer ="Nelder_Mead"))



summary(p3)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula: max_CRP ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes +
##     (1 + day | ID)
##    Data: long_data_CRP
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   3270.8   3310.9  -1624.4   3248.8      272
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.20704 -0.54553 -0.03724  0.51081  2.47797
##
## Random effects:
```

```
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 10655    103.23
##           day           307     17.52   -0.51
##  Residual              2633     51.31
## Number of obs: 283, groups:  ID, 69
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   55.985     43.077   1.300
## poly1       -146.215     10.636 -13.747
## poly2        -14.100      7.726  -1.825
## poly3          9.643      7.130   1.352
## age_yrs        7.813      2.749   2.843
## sex           11.231     23.759   0.473
## inotropes     28.318     30.731   0.921
##
## Correlation of Fixed Effects:
##           (Intr) poly1  poly2  poly3  ag_yrs sex
## poly1      0.021
## poly2      0.019  0.243
## poly3      0.002  0.123  0.191
## age_yrs   -0.642  0.007 -0.002  0.000
## sex       -0.405  0.001 -0.005  0.003  0.007
## inotropes -0.600 -0.011  0.005  0.007 -0.031  0.055
```

```
summary(p4)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula:
## max_CRP ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
##     age_yrs + sex + inotropes + (1 + day | ID)
##    Data: long_data_CRP
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   3287.0   3370.8  -1620.5   3241.0      260
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.14331 -0.51681 -0.02737  0.51864  2.56112
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 10727.8  103.58
##           day           301.5   17.36   -0.52
##  Residual              2541.4   50.41
## Number of obs: 283, groups:  ID, 69
##
## Fixed effects:
##                          Estimate Std. Error t value
## (Intercept)               70.5965    45.2563   1.560
## poly1                   -122.2772    23.4604  -5.212
## poly2                    -12.3552    16.5388  -0.747
## poly3                     23.8946    16.2825   1.467
## IVIG                       0.9864    32.1844   0.031
```

```
## STEROIDS                -32.6114    34.0735  -0.957
## IMMUNOMODULATION          3.4625    29.2232   0.118
## age_yrs                   7.4263     2.7727   2.678
## sex                      19.9469    25.1846   0.792
## inotropes                37.8866    32.6899   1.159
## poly1:IVIG               36.4381    30.0867   1.211
## poly1:STEROIDS          -62.4965    31.9180  -1.958
## poly1:IMMUNOMODULATION  -10.3259    26.5284  -0.389
## poly2:IVIG               25.9440    21.7287   1.194
## poly2:STEROIDS          -23.9642    23.0226  -1.041
## poly2:IMMUNOMODULATION   -8.2954    19.1767  -0.433
## poly3:IVIG              -13.6130    20.1917  -0.674
## poly3:STEROIDS           -7.2755    20.8265  -0.349
## poly3:IMMUNOMODULATION   13.1817    17.5226   0.752
```

```
##
## Correlation matrix not shown by default, as p = 19 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```
anova(p3, p4)
```

```
## Data: long_data_CRP
## Models:
## p3: max_CRP ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes +
## p3:     (1 + day | ID)
## p4: max_CRP ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
## p4:     age_yrs + sex + inotropes + (1 + day | ID)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## p3   11 3270.8 3310.9 -1624.4   3248.8
## p4   23 3287.0 3370.8 -1620.5   3241.0 7.7912 12     0.8012
```

```
# using bootMer to compute 1000 bootstrapped log-likelihood
b1 <- bootMer(p3, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)
b2 <- bootMer(p4, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)

# the 1000 bootstrapped LRT - now  comparing p1 with p2
lrt.b <- -2 * b1$t + 2 * b2$t
# plot
quant <- quantile(lrt.b, probs = c(0.025, 0.975))
print("comparing p1 with p2")
```

```
## [1] "comparing p1 with p2"
```

```
print(quant)
```

```
##      2.5%     97.5%
## -50.74938  88.37704
```

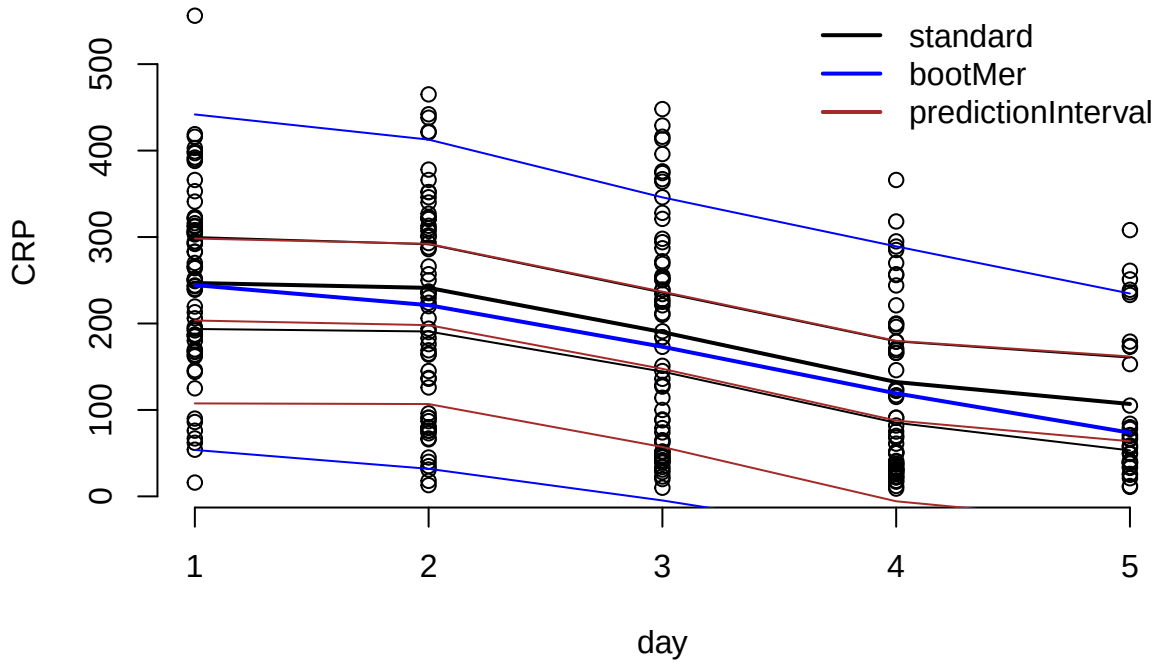
Prediction invervals - comparing different models.

It would be nice to plot models, and compare trajectories of the CRP response with 95% confidence intervals depending upon treatment. However, it is possible that the approach with calculating the CI is not the best one. There are a couple of other options, and these include lme4::bootMer and merTools::predictionIntervals (this being a prediction interval not a confidence interval).

```r
## approach used for the above graph
poly_data_nil <- expand.grid(
  day = c(1:5),
  IVIG = 0,
  STEROIDS = 0,
  IMMUNOMODULATION = 0
)
# need to add a step to compute the orthogonal polynomials
poly_data_nil <- code.poly(df=poly_data_nil, predictor="day", poly.order=3, orthogonal=T)
# this is the predicted mean for just the mean of the polynomial regression
poly_data_nil$predict <- predict(p2, newdata = poly_data_nil, re.form = NA)
mm <- model.matrix(~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) ,
                   poly_data_nil )
vcov.m <- vcov(p2)
vars <- mm %*% vcov.m %*% t(mm)
sds <- sqrt(diag(vars))
z.val <- qnorm(1 - (1 - 0.9)/2)
poly_data_nil$LoCI <- poly_data_nil$predict - z.val * sds
poly_data_nil$HiCI <- poly_data_nil$predict + z.val * sds

## new approach

f_expand_grid <- function(fday){
  fnewdat  <- expand.grid(
    day = fday,
    poly1 = poly_data$poly1[fday],
    poly2 = poly_data$poly2[fday],
    poly3 = poly_data$poly3[fday],
    IVIG = 0,
    STEROIDS = 0,
    IMMUNOMODULATION = 0,
    ID = unique(long_data_CRP[which((long_data_CRP$STEROIDS == 0) & (long_data_CRP$IVIG == 0) & (long_da
  )
  return(fnewdat)
}


mySumm <- function(.) {
  predict(., newdata=newdat, re.form=NULL)
}
####Collapse bootstrap into median, 95% PI
sumBoot <- function(merBoot) {
  return(
    data.frame(fit = apply(merBoot$t, 2, function(x) as.numeric(quantile(x, probs=.5, na.rm=TRUE))),
               lwr = apply(merBoot$t, 2, function(x) as.numeric(quantile(x, probs=.025, na.rm=TRUE))),
               upr = apply(merBoot$t, 2, function(x) as.numeric(quantile(x, probs=.975, na.rm=TRUE)))
    )
  )
}
newdat <- f_expand_grid(1)
boot1 <- lme4::bootMer(p2, mySumm, nsim=250, use.u=FALSE, type="parametric")
PI.boot1 <- sumBoot(boot1)
CI.boot <- c(apply(PI.boot1, MAR = 2, FUN = mean), day = 1)
```

```
for (i in 2:5){
  newdat <- f_expand_grid(i)
  boot1 <- lme4::bootMer(p1, mySumm, nsim=250, use.u=FALSE, type="parametric")
  PI.boot1 <- sumBoot(boot1)
  CI.boot <- rbind(CI.boot,
              c(apply(PI.boot1, MAR = 2, FUN = mean), day = i))
}

# use prediction interval
newdat <- f_expand_grid(1)
PI.predict_int <- predictInterval(merMod = p2, newdata = newdat,
                        level = 0.95, n.sims = 1000,
                        stat = "median", type="linear.prediction",
                        include.resid.var = F,
                        fix.intercept.variance = T
                        )
CI.predict_int <- c(apply(PI.predict_int, MAR = 2, FUN = mean), day = 1)
for (i in 2:5){
  newdat <- f_expand_grid(i)
  PI.predict_int <- predictInterval(merMod = p2, newdata = newdat,
                        level = 0.95, n.sims = 1000,
                        stat = "median", type="linear.prediction",
                        include.resid.var = F,
                        fix.intercept.variance = T
                        )
  CI.predict_int <- rbind(CI.predict_int,
              c(apply(PI.predict_int, MAR = 2, FUN = mean), day = i))
}


plot(x <- long_data_CRP$day,
     y <- long_data_CRP$max_CRP,
     bty = "n",
     xlab = "day",
     ylab = "CRP")

lines(y = poly_data_nil$predict,
      x = poly_data_nil$day,
      col = "black",
      lwd = 2)
lines(y = poly_data_nil$LoCI,
      x = poly_data_nil$day,
      col = "black")
lines(y = poly_data_nil$HiCI,
      x = poly_data_nil$day,
      col = "black")
CI.boot <- as.data.frame(CI.boot)
lines(y = CI.boot$fit,
      x = CI.boot$day,
      col = "blue",
      lwd = 2)
lines(y = CI.boot$lwr,
      x = CI.boot$day,
```

```
      col = "blue")
lines(y = CI.boot$upr,
      x = CI.boot$day,
      col = "blue")
CI.predict_int <- as.data.frame(CI.predict_int)
lines(y = CI.predict_int$fit,
      x = CI.predict_int$day,
      col = "brown")
lines(y = CI.predict_int$lwr,
      x = CI.predict_int$day,
      col = "brown")
lines(y = CI.predict_int$upr,
      x = CI.predict_int$day,
      col = "brown")
legend("topright",
       legend = c("standard", "bootMer", "predictionInterval"),
       lwd = 2,
       col = c("black", "blue", "brown"),
       bty = "n")
```



As can be seen, the standard approach used initially gives the narrowest confidence intervals. It may be overstating the precision with which we are estimating the average CRP. However, we want to be sensitive to any evidence of treatment effect - this study is hypothesis generating in a sense.

CRP data, comparison of the effect of steroids on the average trajectory of the CRP

We are using the mixed effects model, with CRP as the outcome. All the three treatments (STEROIDS, IMMUNOMODULATION AND IVIG) are fixed effects, and the slope and random intercept of each participant are random effects. In this scenario, for clarity, the variable IMMUNOMODULATION means a biologic agent (noting that steroids and IVIG are also immunomodulators of sorts). This nomenclature is a hangover from the way the data were acquired - in the paper they are referred to as biologics.

For these models, we draw the fitted curve and 95% confidence intervals, comparing the fitted curve for no treatment with the one from treatment (either STEROIDS, IMMUNOMODULATION or IVIG).

First we check steroids:-

```
make_graph <- function(fIVIG = 0, fSTEROIDS = 0, fIMMUNOMODULATION = 0,
                       fmodel,
                       ftitle = "Insert title",
                       fylim = c(0, 400),
                       fylab = "",
                       predict_freq = c(1:5)){


  foreground_treat   <- rgb( 25, 121, 169, max = 255, alpha = 255)
  background_treat   <- rgb( 25, 121, 169, max = 255, alpha = 125)
  foreground_cont    <- rgb(128,  57,  30, max = 255, alpha = 255)
  background_cont    <- rgb(128,  57,  30, max = 255, alpha = 125)

  # make the control data
  fpoly_data_nil <- expand.grid(
    day = predict_freq,
    IVIG = 0,
    STEROIDS = 0,
    IMMUNOMODULATION = 0)
  # need to add a step to compute the orthogonal polynomials
  fpoly_data_nil <- code.poly(df=fpoly_data_nil, predictor="day", poly.order=3, orthogonal=T)

  # this is the predicted mean for just the mean of the polynomial regression
  fpoly_data_nil$predict <- predict(fmodel, newdata = fpoly_data_nil, re.form = NA)
  mm <- model.matrix(~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) ,
                     fpoly_data_nil )
  vcov.m <- vcov(fmodel)
  vars <- mm %*% vcov.m %*% t(mm)
  sds <- sqrt(diag(vars))
  z.val <- qnorm(1 - (1 - 0.9)/2)
  fpoly_data_nil$LoCI <- fpoly_data_nil$predict - z.val * sds
  fpoly_data_nil$HiCI <- fpoly_data_nil$predict + z.val * sds

  ## here is the treatment:
  # adjusting the fpoly_data_treat is the way to get the
  # contrasts for the graph
  fpoly_data_treat <- expand.grid(
    day = predict_freq,
    IVIG = fIVIG,
    STEROIDS = fSTEROIDS,
    IMMUNOMODULATION = fIMMUNOMODULATION)
```

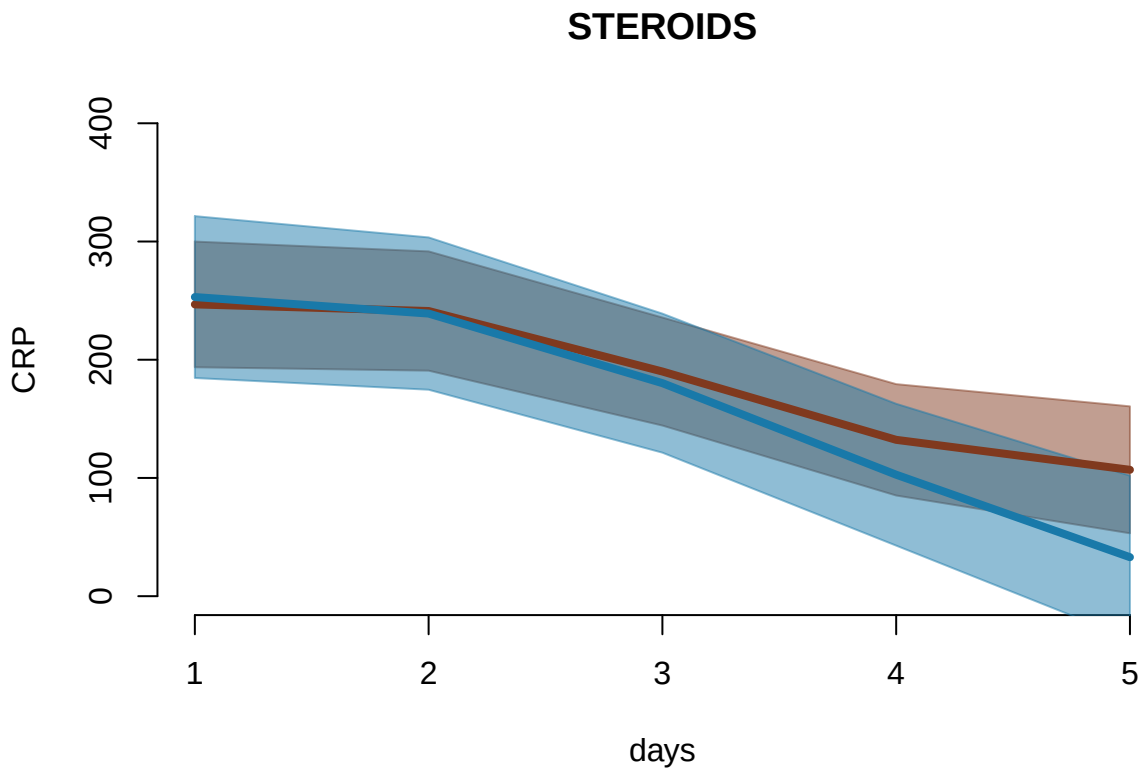
```
fpoly_data_treat <- code.poly(df=fpoly_data_treat, predictor="day", poly.order=3, orthogonal=T)

# this is the predicted mean for just the mean of the polynomial regression
fpoly_data_treat$predict <- predict(fmodel, newdata = fpoly_data_treat, re.form = NA)
mm <- model.matrix(~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) ,
                   fpoly_data_treat )
vcov.m <- vcov(fmodel)
vars <- mm %*% vcov.m %*% t(mm)
sds <- sqrt(diag(vars))
z.val <- qnorm(1 - (1 - 0.9)/2)
fpoly_data_treat$LoCI <- fpoly_data_treat$predict - z.val * sds
fpoly_data_treat$HiCI <- fpoly_data_treat$predict + z.val * sds
print("about to plot...")

# plot it
plot(x = 1,
     y = 1,
     bty  = "n",
     ylab = ifelse(make_pngs, "", fylab),
     xlab = ifelse(make_pngs, "", 'days'),
     col  = 'white',
     ylim = fylim,
     xlim = c(1, 5),
     main = ifelse(make_pngs, "", ftitle))

## next step is to fill in the colors
## let's use polygon

upper_line_control <- cbind ( fpoly_data_nil$day, fpoly_data_nil$HiCI )
colnames(upper_line_control) <- c("day", "value")
lower_line_control <- cbind ( fpoly_data_nil$day, fpoly_data_nil$LoCI )
lower_line_control <- lower_line_control[ nrow ( lower_line_control ) : 1, ]
colnames(lower_line_control) <- c("day", "value")
polygon_control <- rbind(upper_line_control, lower_line_control)
polygon_control <- as.data.frame(polygon_control)
polygon(x = polygon_control$day, y = polygon_control$value,
        col = background_cont, density = NA)

upper_line_treat <- cbind ( fpoly_data_treat$day, fpoly_data_treat$HiCI )
colnames(upper_line_treat) <- c("day", "value")
lower_line_treat <- cbind ( fpoly_data_treat$day, fpoly_data_treat$LoCI )
lower_line_treat <- lower_line_treat[ nrow ( lower_line_treat ) : 1, ]
colnames(lower_line_treat) <- c("day", "value")
polygon_treat <- rbind(upper_line_treat, lower_line_treat)
polygon_treat <- as.data.frame(polygon_treat)
polygon(x = polygon_treat$day, y = polygon_treat$value,
        col = background_treat, density = NA)

# then draw the mean lines
lines(x = fpoly_data_nil$day,
      y = fpoly_data_nil$predict,
      col = foreground_cont,
      lwd = 4)
```

```
  lines(x = fpoly_data_treat$day,
        y = fpoly_data_treat$predict,
        col = foreground_treat,
        lwd = 4)
}
make_graph_2 <- function(fIVIG = 0, fSTEROIDS = 0, fIMMUNOMODULATION = 0,
                         fmodel,
                         ftitle = "Insert title",
                         fylim = c(0, 400),
                         fylab = "",
                         predict_freq = c(1:5)){


  foreground_treat   <- rgb( 25, 121, 169, max = 255, alpha = 255)
  background_treat   <- rgb( 25, 121, 169, max = 255, alpha = 125)
  foreground_cont    <- rgb(128,  57,  30, max = 255, alpha = 255)
  background_cont    <- rgb(128,  57,  30, max = 255, alpha = 125)

  # make the control data
  fpoly_data_nil <- expand.grid(
    day = predict_freq,
    age_yrs = 11, # this is the median age
    inotropes = 1,
    sex = 1,
    IVIG = 0,
    STEROIDS = 0,
    IMMUNOMODULATION = 0)
  # need to add a step to compute the orthogonal polynomials
  fpoly_data_nil <- code.poly(df=fpoly_data_nil, predictor="day", poly.order=3, orthogonal=T)

  # this is the predicted mean for just the mean of the polynomial regression
  fpoly_data_nil$predict <- predict(fmodel, newdata = fpoly_data_nil, re.form = NA)
  mm <- model.matrix(~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) + age_yrs + sex +
                     fpoly_data_nil )
  vcov.m <- vcov(fmodel)
  vars <- mm %*% vcov.m %*% t(mm)
  sds <- sqrt(diag(vars))
  z.val <- qnorm(1 - (1 - 0.9)/2)
  fpoly_data_nil$LoCI <- fpoly_data_nil$predict - z.val * sds
  fpoly_data_nil$HiCI <- fpoly_data_nil$predict + z.val * sds

  ## here is the treatment:
  # adjusting the fpoly_data_treat is the way to get the
  # contrasts for the graph
  fpoly_data_treat <- expand.grid(
    day = predict_freq,
    age_yrs = 11, # this is the median age
    inotropes = 1,
    sex = 1,
    IVIG = fIVIG,
    STEROIDS = fSTEROIDS,
    IMMUNOMODULATION = fIMMUNOMODULATION)
  fpoly_data_treat <- code.poly(df=fpoly_data_treat, predictor="day", poly.order=3, orthogonal=T)
```

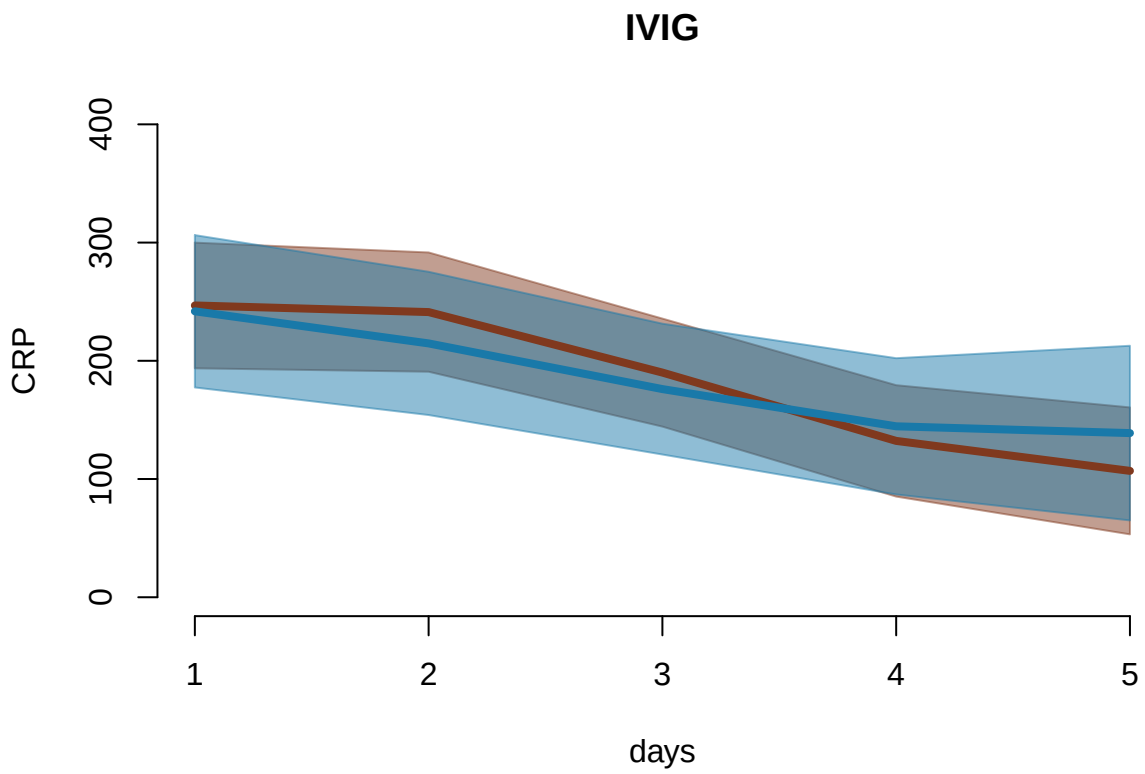
```
# this is the predicted mean for just the mean of the polynomial regression
fpoly_data_treat$predict <- predict(fmodel, newdata = fpoly_data_treat, re.form = NA)
mm <- model.matrix(~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) + age_yrs + sex +
                   fpoly_data_treat )
vcov.m <- vcov(fmodel)
vars <- mm %*% vcov.m %*% t(mm)
sds <- sqrt(diag(vars))
z.val <- qnorm(1 - (1 - 0.9)/2)
fpoly_data_treat$LoCI <- fpoly_data_treat$predict - z.val * sds
fpoly_data_treat$HiCI <- fpoly_data_treat$predict + z.val * sds
print("about to plot...")

# plot it
plot(x = 1,
     y = 1,
     bty  = "n",
     ylab = ifelse(make_pngs, "", fylab),
     xlab = ifelse(make_pngs, "", 'days'),
     col  = 'white',
     ylim = fylim,
     xlim = c(1, 5),
     main = ifelse(make_pngs, "", ftitle))

## next step is to fill in the colors
## let's use polygon

upper_line_control <- cbind ( fpoly_data_nil$day, fpoly_data_nil$HiCI )
colnames(upper_line_control) <- c("day", "value")
lower_line_control <- cbind ( fpoly_data_nil$day, fpoly_data_nil$LoCI )
lower_line_control <- lower_line_control[ nrow ( lower_line_control ) : 1, ]
colnames(lower_line_control) <- c("day", "value")
polygon_control <- rbind(upper_line_control, lower_line_control)
polygon_control <- as.data.frame(polygon_control)
polygon(x = polygon_control$day, y = polygon_control$value,
        col = background_cont, density = NA)

upper_line_treat <- cbind ( fpoly_data_treat$day, fpoly_data_treat$HiCI )
colnames(upper_line_treat) <- c("day", "value")
lower_line_treat <- cbind ( fpoly_data_treat$day, fpoly_data_treat$LoCI )
lower_line_treat <- lower_line_treat[ nrow ( lower_line_treat ) : 1, ]
colnames(lower_line_treat) <- c("day", "value")
polygon_treat <- rbind(upper_line_treat, lower_line_treat)
polygon_treat <- as.data.frame(polygon_treat)
polygon(x = polygon_treat$day, y = polygon_treat$value,
        col = background_treat, density = NA)

# then draw the mean lines
lines(x = fpoly_data_nil$day,
      y = fpoly_data_nil$predict,
      col = foreground_cont,
      lwd = 4)

lines(x = fpoly_data_treat$day,
```

```
        y = fpoly_data_treat$predict,
        col = foreground_treat,
        lwd = 4)
}
```

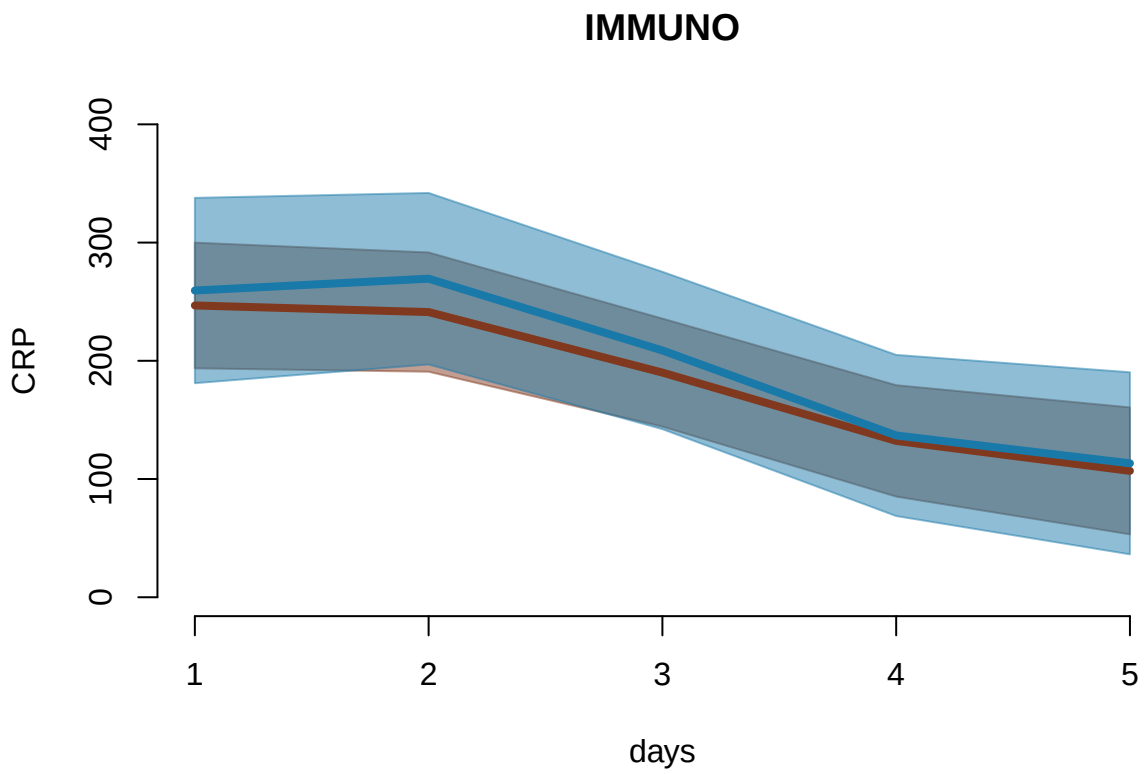
```
## [1] "about to plot..."
```



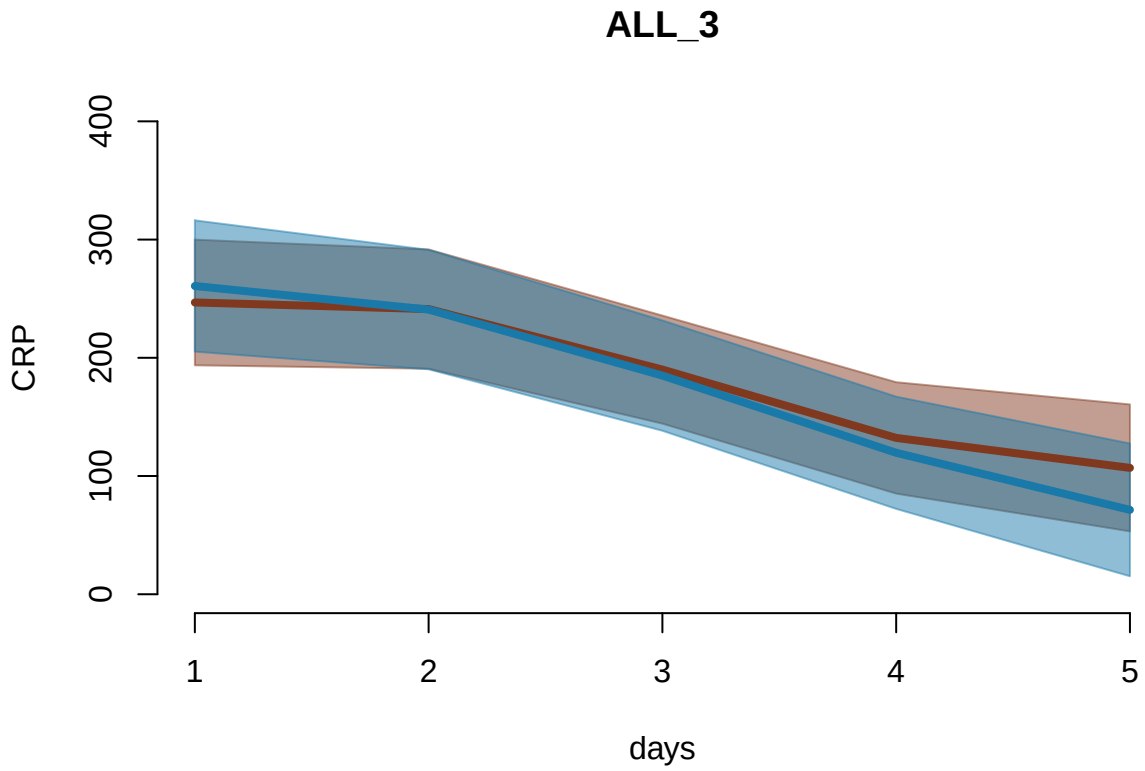

```
## [1] "about to plot..."
```



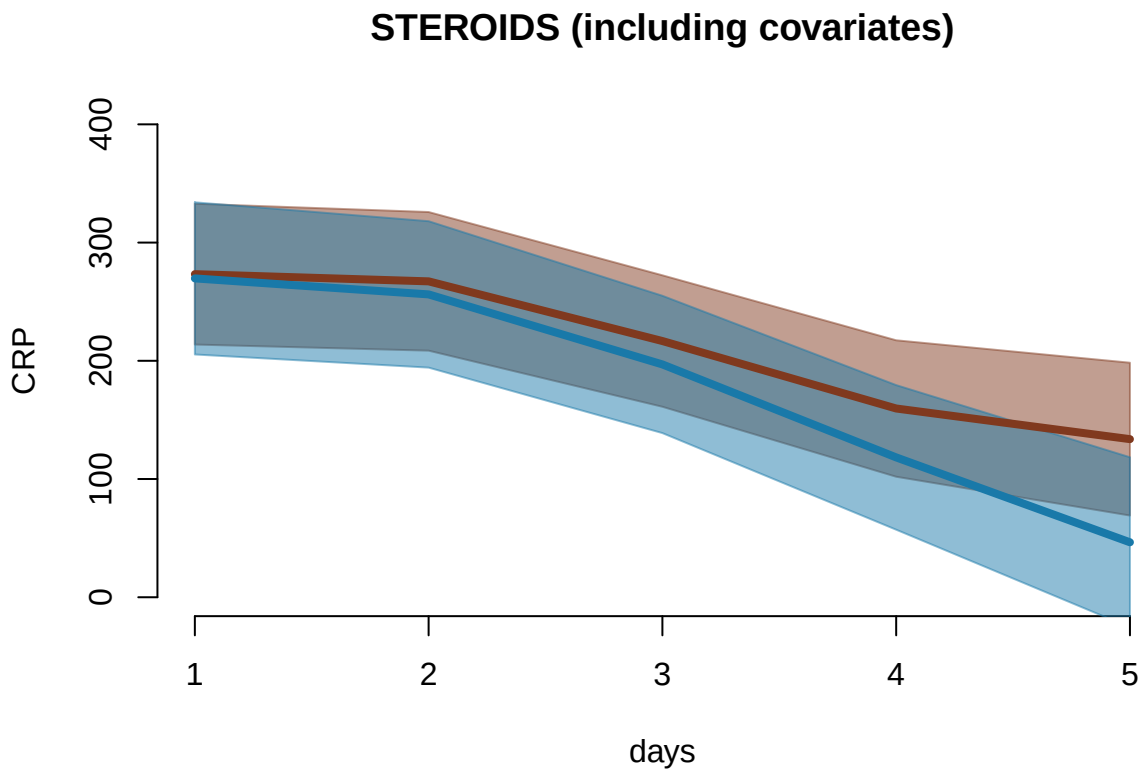

```
## [1] "about to plot..."
```



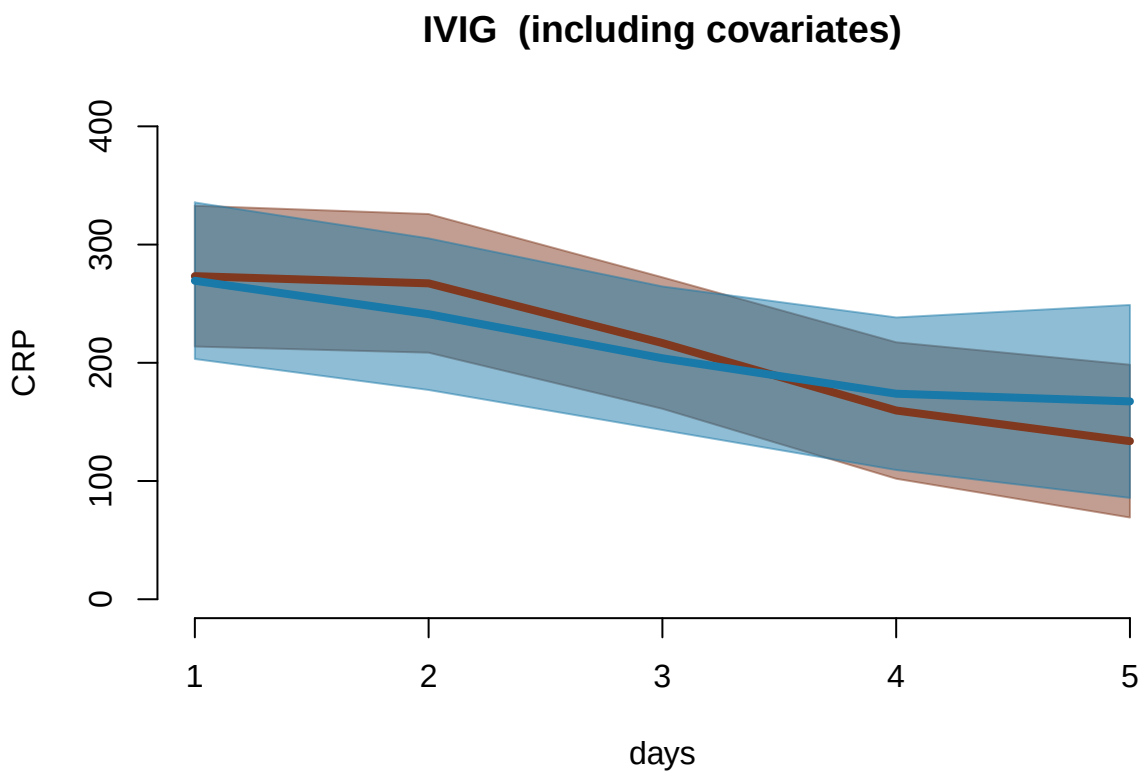

```
## [1] "about to plot..."
```



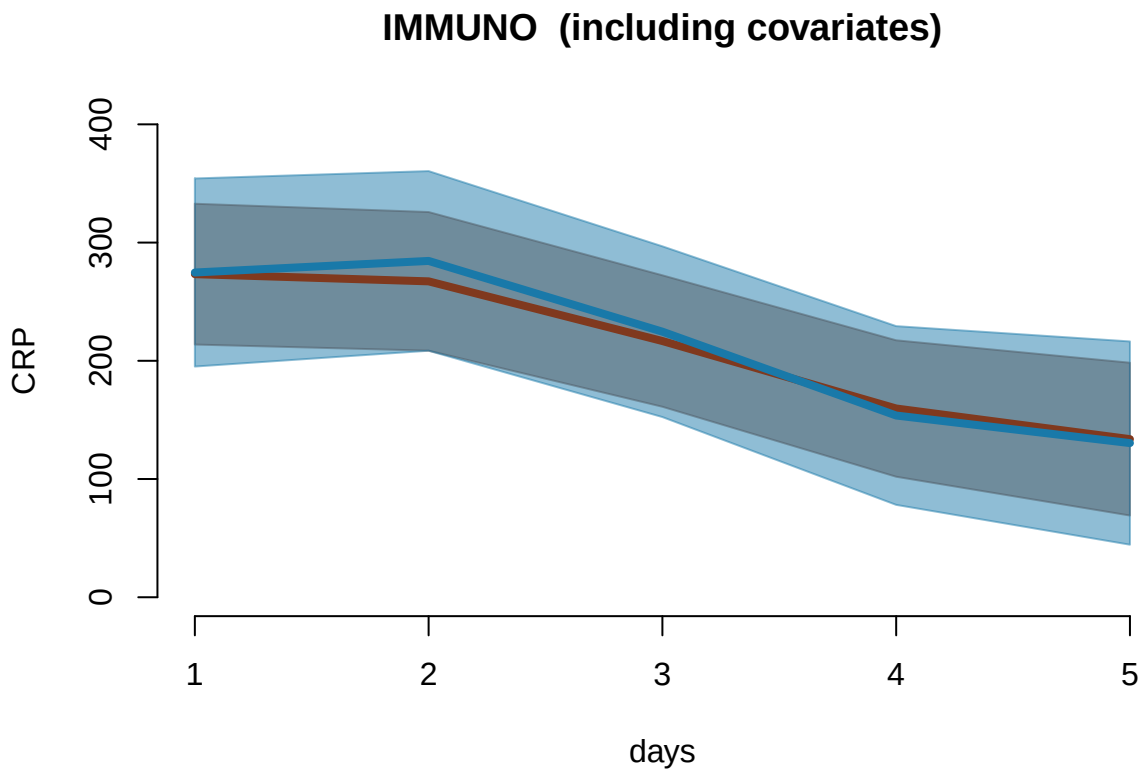

```
## [1] "about to plot..."
```



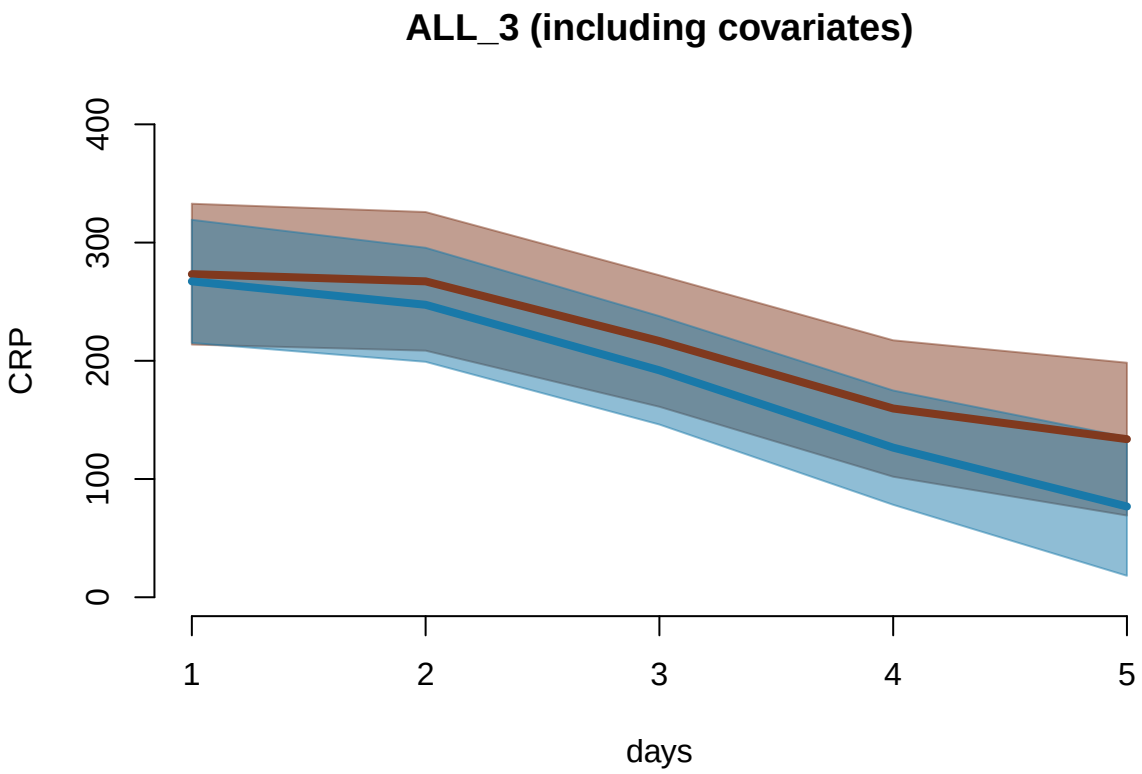

```
## [1] "about to plot..."
```




```
## [1] "about to plot..."
```




```
## [1] "about to plot..."
```

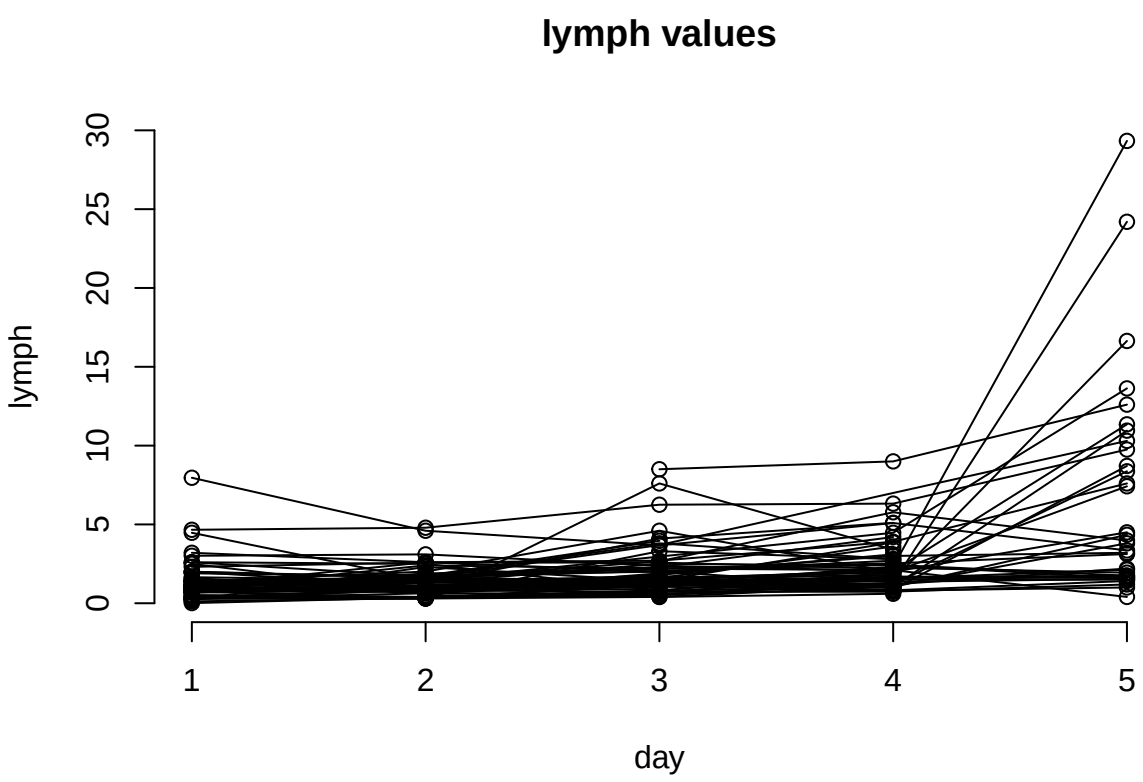


Lymphocyte counts

We essentially repeat this analysis for lymphocyte count and platelet count as the response variables. We note that it is essentially identical to the above approach, and therefore we have not repeated the full code.

The second inflammatory marker we need to consider is the lymphocyte count.
We again look for missing values:

Var1	Freq
0	5
1	2
2	3
3	7
4	34
5	27
Sum	78




```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula:
## max_lymph ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
##     (1 + day | ID)
##    Data: long_data_lymph
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   1276.3   1348.7   -618.2   1236.3      256
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.8099 -0.4106 -0.0674  0.3336  5.9613
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 2.7640   1.6625
##           day         0.8771   0.9365   -1.00
##  Residual             3.8570   1.9639
## Number of obs: 276, groups:  ID, 65
##
## Fixed effects:
##                       Estimate Std. Error t value
## (Intercept)            1.94626    0.44833   4.341
```

```
## poly1                    2.18072    1.12854   1.932
## poly2                    1.02990    0.68070   1.513
## poly3                    0.60204    0.64024   0.940
## IVIG                     0.51624    0.55820   0.925
## STEROIDS                 0.43784    0.58288   0.751
## IMMUNOMODULATION         0.04503    0.51050   0.088
## poly1:IVIG               1.11175    1.40587   0.791
## poly1:STEROIDS           0.66698    1.48920   0.448
## poly1:IMMUNOMODULATION -0.17909    1.28066  -0.140
## poly2:IVIG               0.69586    0.86758   0.802
## poly2:STEROIDS           0.91619    0.93511   0.980
## poly2:IMMUNOMODULATION -1.50600    0.78041  -1.930
## poly3:IVIG               0.82509    0.79449   1.039
## poly3:STEROIDS          -0.12339    0.83344  -0.148
## poly3:IMMUNOMODULATION -0.70513    0.71127  -0.991

##
## Correlation matrix not shown by default, as p = 16 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## optimizer (Nelder_Mead) convergence code: 0 (OK)
## boundary (singular) fit: see ?isSingular

## Data: long_data_lymph
## Models:
## p1_lymph: max_lymph ~ (poly1 + poly2 + poly3) + (1 + day | ID)
## p2_lymph: max_lymph ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
## p2_lymph:     (1 + day | ID)
##          npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## p1_lymph    8 1263.8 1292.8 -623.90   1247.8
## p2_lymph   20 1276.3 1348.7 -618.16   1236.3 11.482 12     0.4881

## [1] TRUE
```

We get singularties in the model, thought this should not affect the answers (see Pasch, Bret, Benjamin M. Bolker, and Steven M. Phelps. 2013. "Interspecific Dominance via Vocal Interactions Mediates Altitudinal Zonation in Neotropical Singing Mice." The American Naturalist 182 (5): E161–E173. https://doi.org/10.1086/673263.)

We compare the models with the bootstrap:

```
# using bootMer to compute 100 bootstrapped log-likelihood
b1 <- bootMer(p1, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)
b2 <- bootMer(p2_lymph, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)


# the 100 bootstrapped LRT - first comparing m0 with p1
lrt.b <- -2 * b1$t + 2 * b2$t
# plot
quant <- quantile(lrt.b, probs = c(0.025, 0.975))
print(quant)
```

```
##     2.5%    97.5%
## 1969.261 2100.070
```

We also make models with baseline covariates:-

```r
# a reminder of the previous models explored:
# p1 <- max_CRP ~ (poly1 + poly2 + poly3) + (1 + day | ID)
# p2 <- max_CRP ~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) + (1 + day | ID)

# p3 will be p1 but with covariates, which we will model as fixed effect

colnames(long_data_lymph)[85] <- "baseline_CRP"

long_data_lymph$age_yrs <- as.numeric(long_data_lymph$age_yrs)



p3_lymph <- lmer(max_lymph ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes + (1 + day | ID),
            data = long_data_lymph,
            REML = F,
            control = lmerControl(optimizer ="Nelder_Mead"))
```

```
## boundary (singular) fit: see ?isSingular
```

```r
# note that including baseline CRP in the model introduces singularities, therefore this is omitted

p4_lymph <- lmer(max_lymph ~ (poly1 + poly2 + poly3) * ( IVIG + STEROIDS + IMMUNOMODULATION) + age_yrs +
            data = long_data_lymph,
            REML = F,
            control = lmerControl(optimizer ="Nelder_Mead"))
```

```
## boundary (singular) fit: see ?isSingular
```

```r
summary(p3_lymph)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula: max_lymph ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes +
##     (1 + day | ID)
##    Data: long_data_lymph
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   1262.2   1302.0   -620.1   1240.2      265
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.9858 -0.4131 -0.0830  0.3647  6.1635
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 3.1131   1.7644
##           day         0.9524   0.9759   -1.00
##  Residual             3.8779   1.9692
## Number of obs: 276, groups:  ID, 65
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  3.82026    0.55479   6.886
## poly1        3.44430    0.50386   6.836
## poly2        1.94758    0.30277   6.432
```

```
## poly3        0.98575    0.27395   3.598
## age_yrs     -0.08746    0.03605  -2.426
## sex          0.12685    0.29673   0.428
## inotropes   -0.41625    0.39913  -1.043
##
## Correlation of Fixed Effects:
##           (Intr) poly1  poly2  poly3  ag_yrs sex
## poly1      0.252
## poly2      0.070  0.228
## poly3      0.056  0.121  0.196
## age_yrs   -0.606  0.007 -0.021 -0.005
## sex       -0.371  0.005 -0.007  0.002  0.027
## inotropes -0.553 -0.012  0.015 -0.025 -0.096 -0.013
## optimizer (Nelder_Mead) convergence code: 0 (OK)
## boundary (singular) fit: see ?isSingular
```

```
summary(p4_lymph)
```

```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula:
## max_lymph ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
##     age_yrs + sex + inotropes + (1 + day | ID)
##    Data: long_data_lymph
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   1273.5   1356.8   -613.8   1227.5      253
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.8653 -0.4273 -0.0757  0.3227  6.0242
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 2.9592   1.7202
##           day         0.8914   0.9441   -1.00
##  Residual             3.7256   1.9302
## Number of obs: 276, groups:  ID, 65
##
## Fixed effects:
##                         Estimate Std. Error t value
## (Intercept)              3.26225    0.63433   5.143
## poly1                    2.19281    1.12621   1.947
## poly2                    1.03059    0.66968   1.539
## poly3                    0.65733    0.63028   1.043
## IVIG                     0.52854    0.54840   0.964
## STEROIDS                 0.54621    0.58346   0.936
## IMMUNOMODULATION         0.19821    0.50195   0.395
## age_yrs                 -0.08803    0.03603  -2.443
## sex                     -0.04477    0.30959  -0.145
## inotropes               -0.60040    0.41049  -1.463
## poly1:IVIG               1.09496    1.40272   0.781
## poly1:STEROIDS           0.69376    1.48422   0.467
## poly1:IMMUNOMODULATION  -0.29805    1.27973  -0.233
## poly2:IVIG               0.66428    0.85335   0.778
```

```
## poly2:STEROIDS          0.94536    0.91999   1.028
## poly2:IMMUNOMODULATION -1.45576    0.76792  -1.896
## poly3:IVIG              0.80737    0.78174   1.033
## poly3:STEROIDS         -0.16174    0.81991  -0.197
## poly3:IMMUNOMODULATION -0.69634    0.69934  -0.996
```

```
##
## Correlation matrix not shown by default, as p = 19 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```
## optimizer (Nelder_Mead) convergence code: 0 (OK)
## boundary (singular) fit: see ?isSingular
```

```
anova(p3_lymph, p4_lymph)
```

```
## Data: long_data_lymph
## Models:
## p3_lymph: max_lymph ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes +
## p3_lymph:     (1 + day | ID)
## p4_lymph: max_lymph ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
## p4_lymph:     age_yrs + sex + inotropes + (1 + day | ID)
##          npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## p3_lymph   11 1262.2 1302.0 -620.10   1240.2
## p4_lymph   23 1273.5 1356.8 -613.76   1227.5 12.678 12     0.3929
```

```
# using bootMer to compute 1000 bootstrapped log-likelihood
b1 <- bootMer(p3, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)
b2 <- bootMer(p4, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)

# the 1000 bootstrapped LRT - now  comparing p1 with p2
lrt.b <- -2 * b1$t + 2 * b2$t
# plot
quant <- quantile(lrt.b, probs = c(0.025, 0.975))
print("comparing p1 with p2")
```

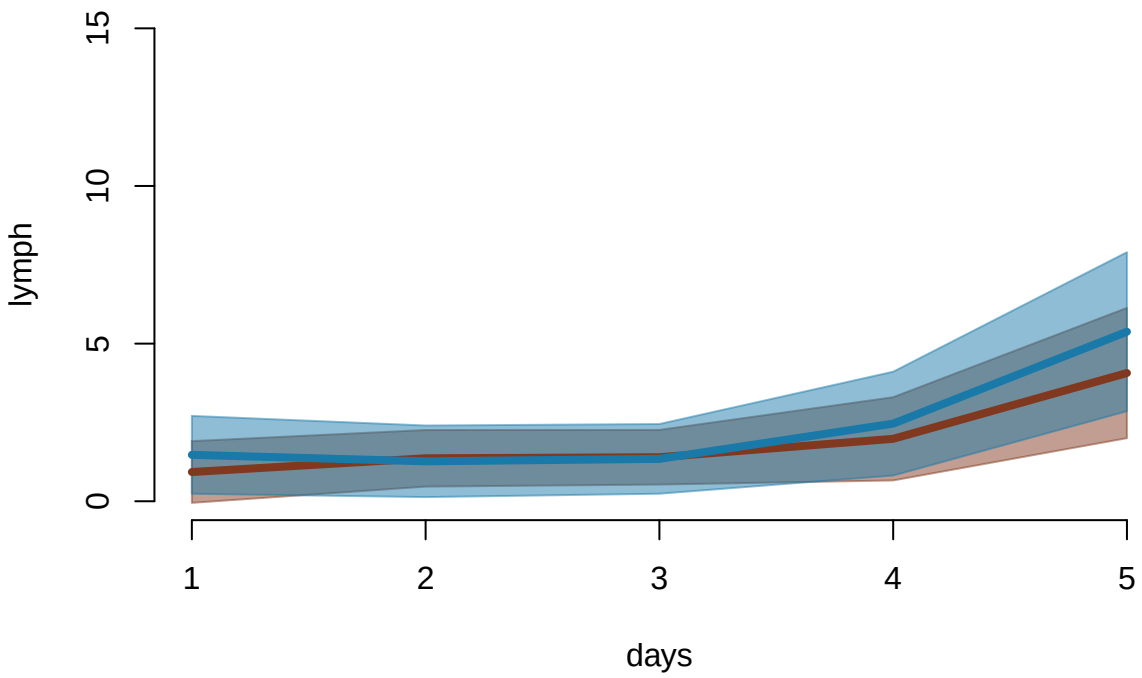
```
## [1] "comparing p1 with p2"
```

```
print(quant)
```

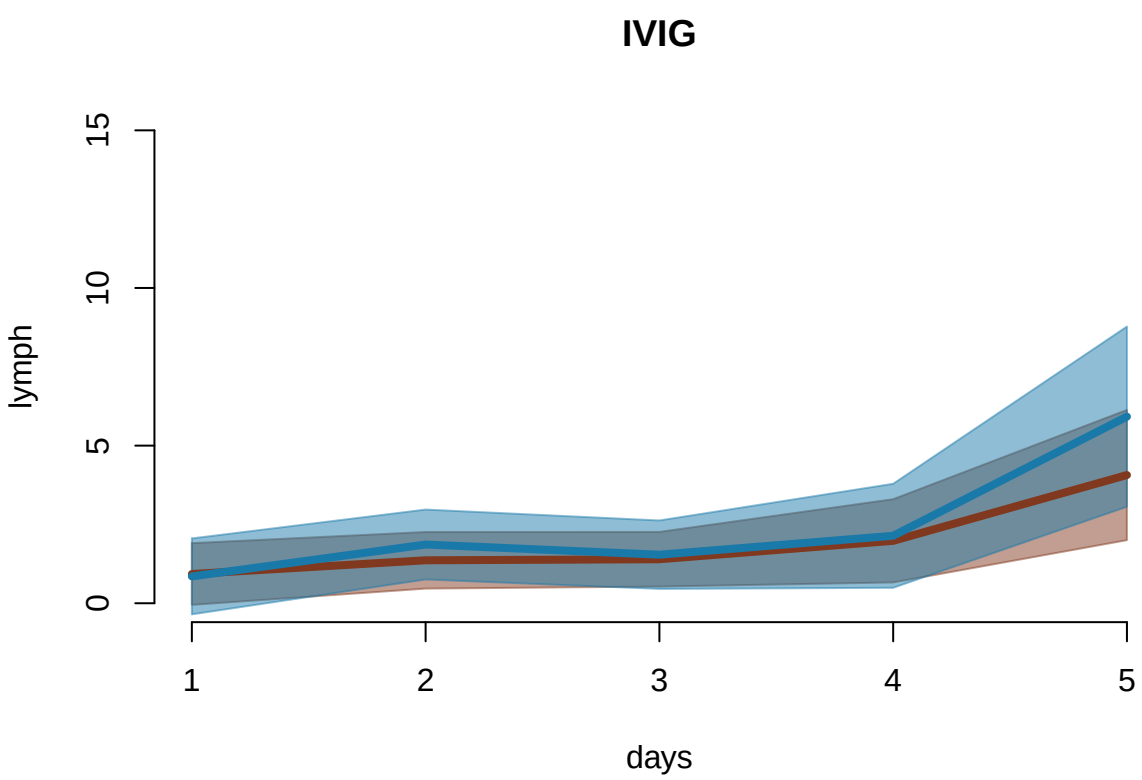
```
##      2.5%     97.5%
## -51.48303  88.42671
```

So we proceed to the modeling of fitted curves.

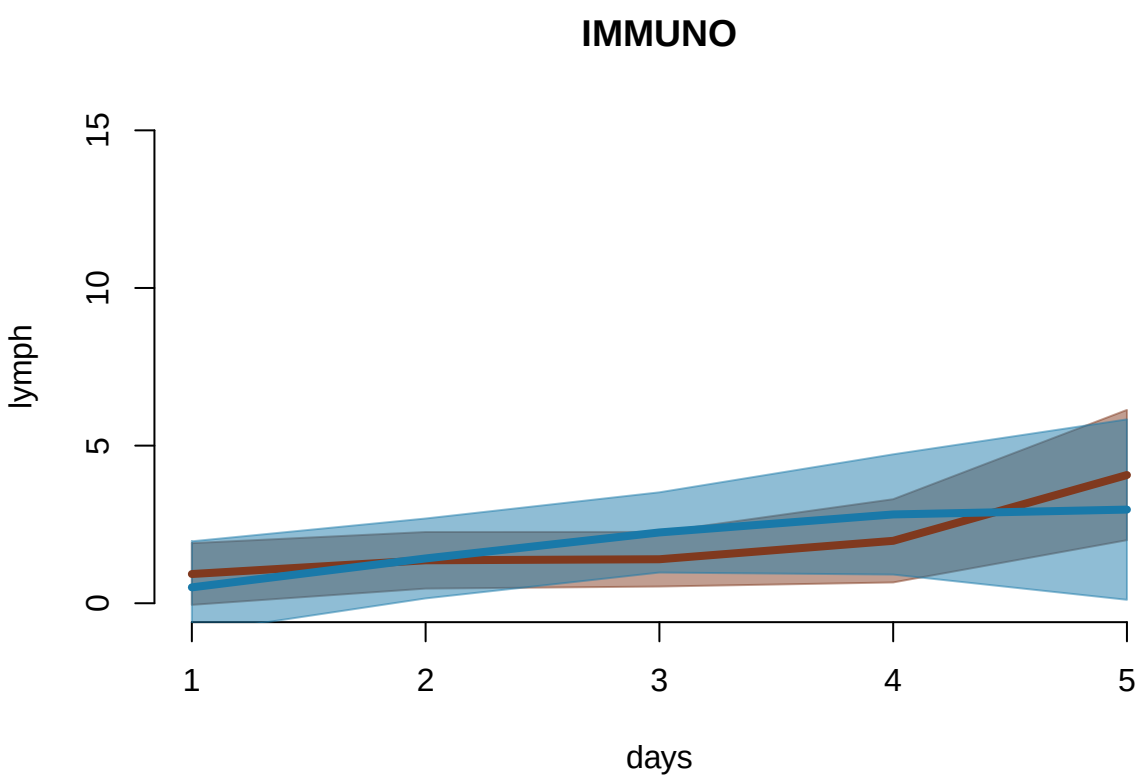
```
## [1] "about to plot..."
```



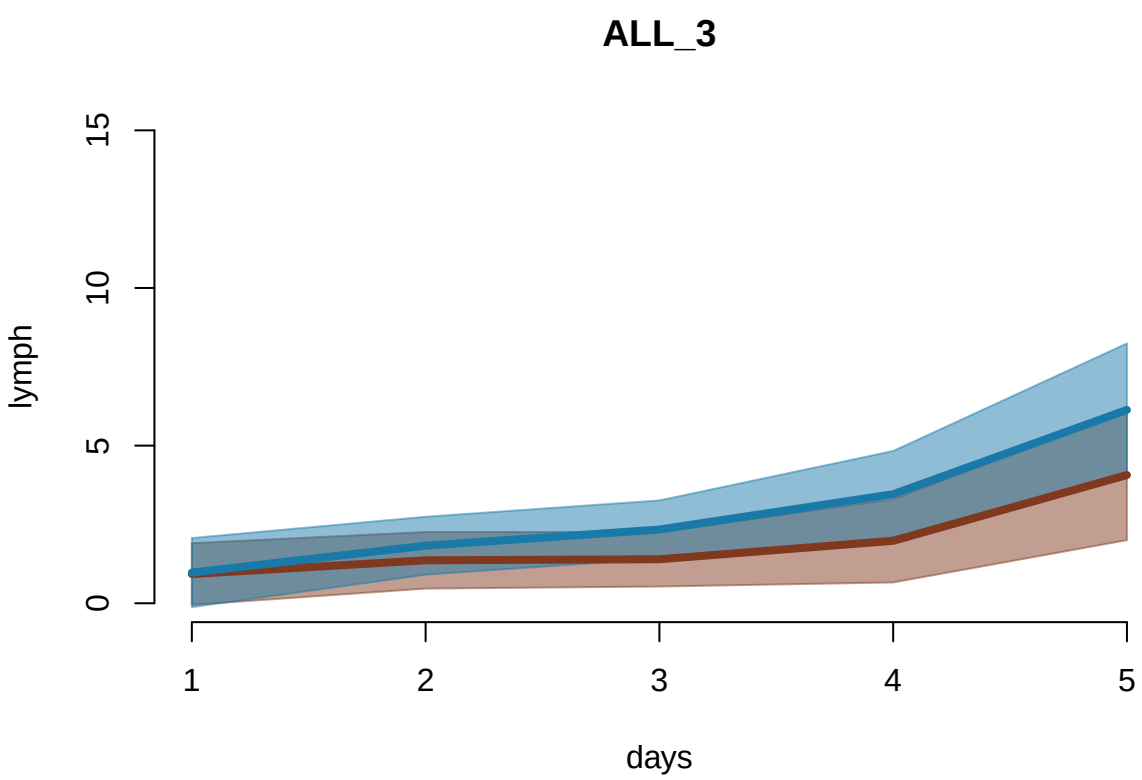

```
## [1] "about to plot..."
```



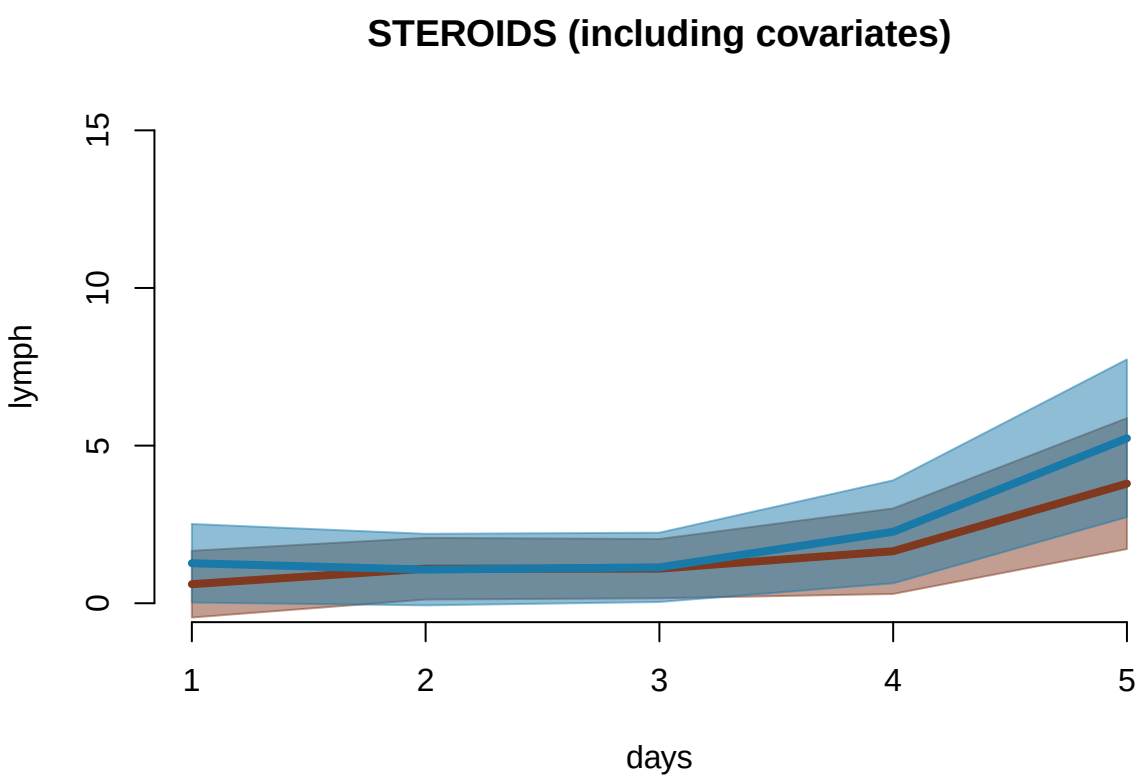

```
## [1] "about to plot..."
```



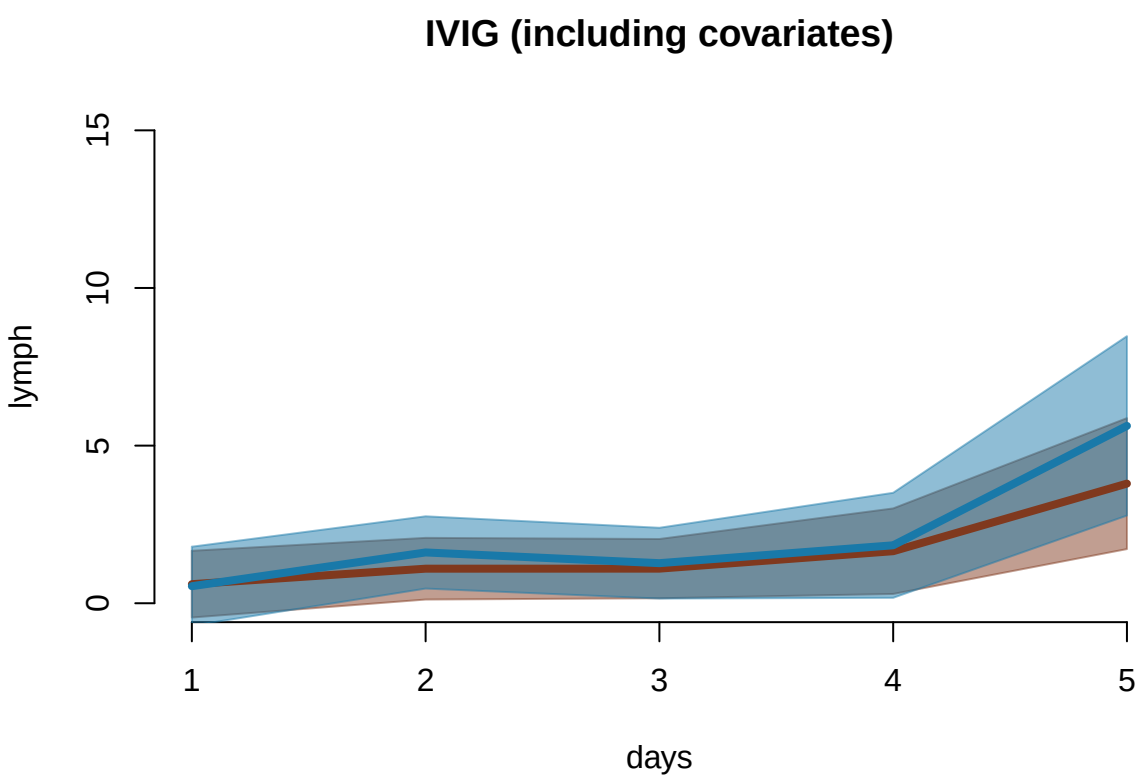

```
## [1] "about to plot..."
```



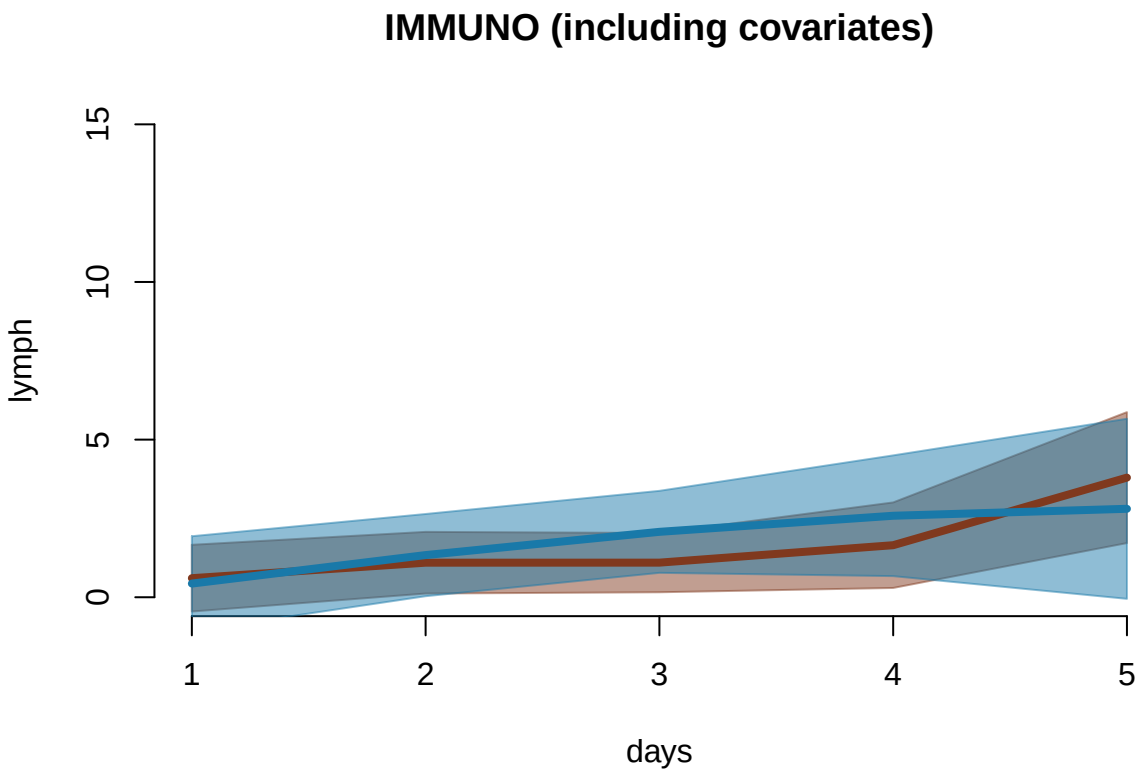

```
## [1] "about to plot..."
```



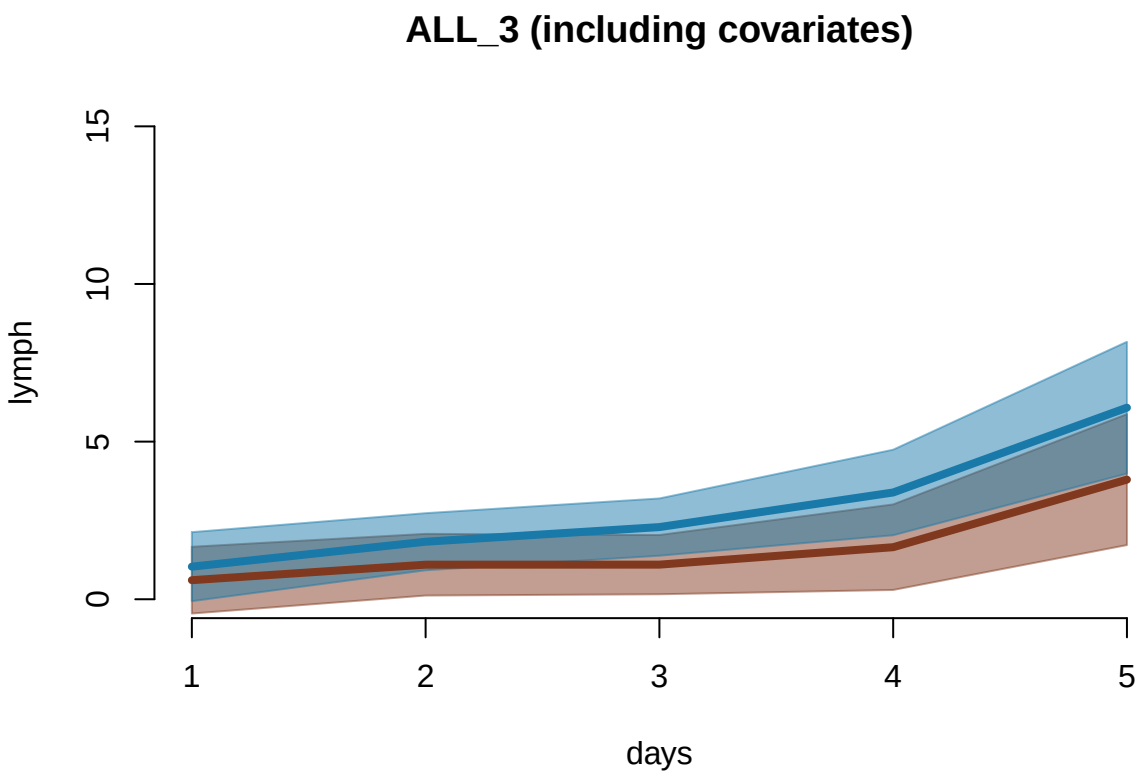

```
## [1] "about to plot..."
```




```
## [1] "about to plot..."
```




```
## [1] "about to plot..."
```



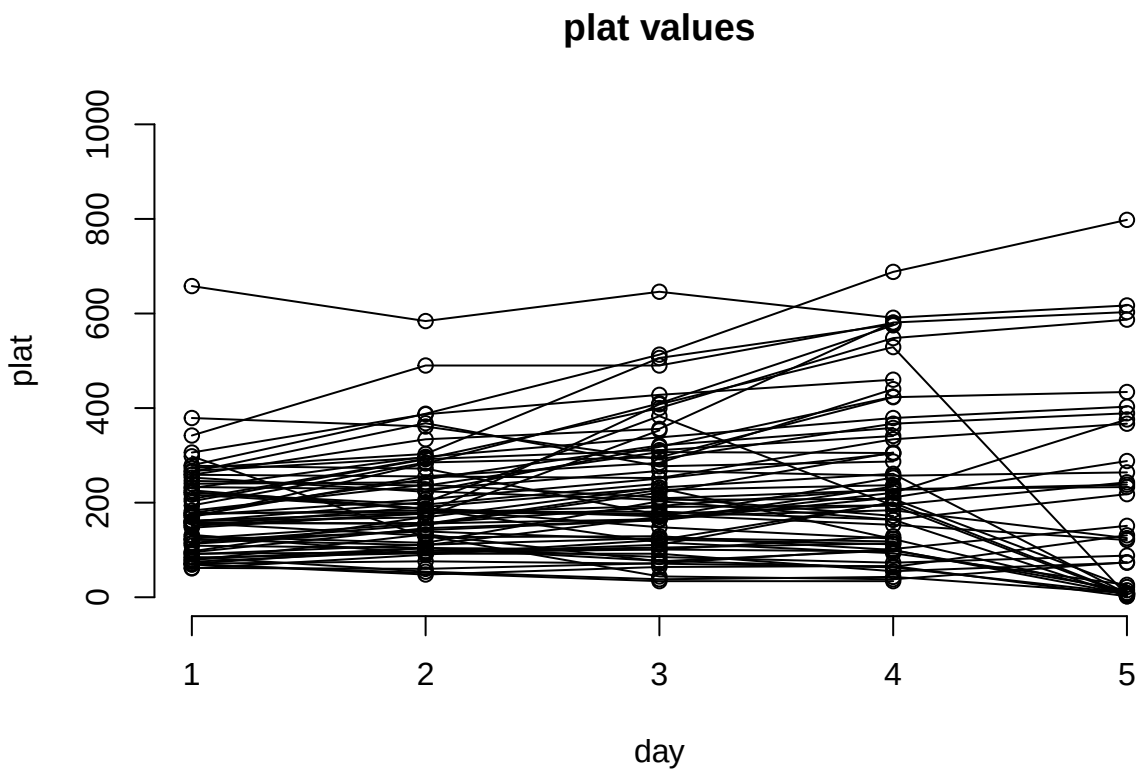
Again, no clear evidence that any of the treatments have any effect on these parameters.

Platelet data

We repeat the above analyses, but examining platelets.

Var1	Freq
0	5
1	2
2	3
3	6
4	34
5	28
Sum	78

We next explore these data graphically:




```
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula:
## max_plat ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
##     (1 + day | ID)
##    Data: long_data_plat
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   3278.0   3350.5  -1619.0   3238.0      258
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.6718 -0.4226 -0.0434  0.3045  4.7040
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 7672     87.59
##           day         1339     36.60    -0.27
##  Residual             2493     49.93
## Number of obs: 278, groups:  ID, 65
##
## Fixed effects:
##                       Estimate Std. Error t value
## (Intercept)            185.887     35.346   5.259
## poly1                   25.695     38.550   0.667
## poly2                  -17.172     17.653  -0.973
```

```
## poly3                   -19.204     16.624  -1.155
## IVIG                     61.744     44.208   1.397
## STEROIDS                -12.527     44.853  -0.279
## IMMUNOMODULATION        -42.008     41.322  -1.017
## poly1:IVIG               54.797     48.029   1.141
## poly1:STEROIDS          -27.774     50.123  -0.554
## poly1:IMMUNOMODULATION  -22.051     44.289  -0.498
## poly2:IVIG              -59.895     22.014  -2.721
## poly2:STEROIDS           33.767     24.139   1.399
## poly2:IMMUNOMODULATION   42.578     19.920   2.137
## poly3:IVIG               -7.577     20.548  -0.369
## poly3:STEROIDS           -6.821     21.615  -0.316
## poly3:IMMUNOMODULATION   32.649     18.204   1.793

##
## Correlation matrix not shown by default, as p = 16 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## Data: long_data_plat
## Models:
## p1: max_plat ~ (poly1 + poly2 + poly3) + (1 + day | ID)
## p2_plat: max_plat ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
## p2_plat:     (1 + day | ID)
##         npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## p1         8 3273.7 3302.7 -1628.8   3257.7
## p2_plat   20 3278.0 3350.5 -1619.0   3238.0 19.695 12    0.07308 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## [1] FALSE
```

Again, we compare models with the bootstrap of the log-likelihood:

```
# using bootMer to compute 100 bootstrapped log-likelihood

b1 <- bootMer(p1, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)
b2 <- bootMer(p2_plat, FUN = function(x) as.numeric(logLik(x)), nsim = 1000)


# the 100 bootstrapped LRT - now  comparing p1 with p2
lrt.b <- -2 * b1$t + 2 * b2$t
# plot
quant <- quantile(lrt.b, probs = c(0.025, 0.975))
print(quant)
```

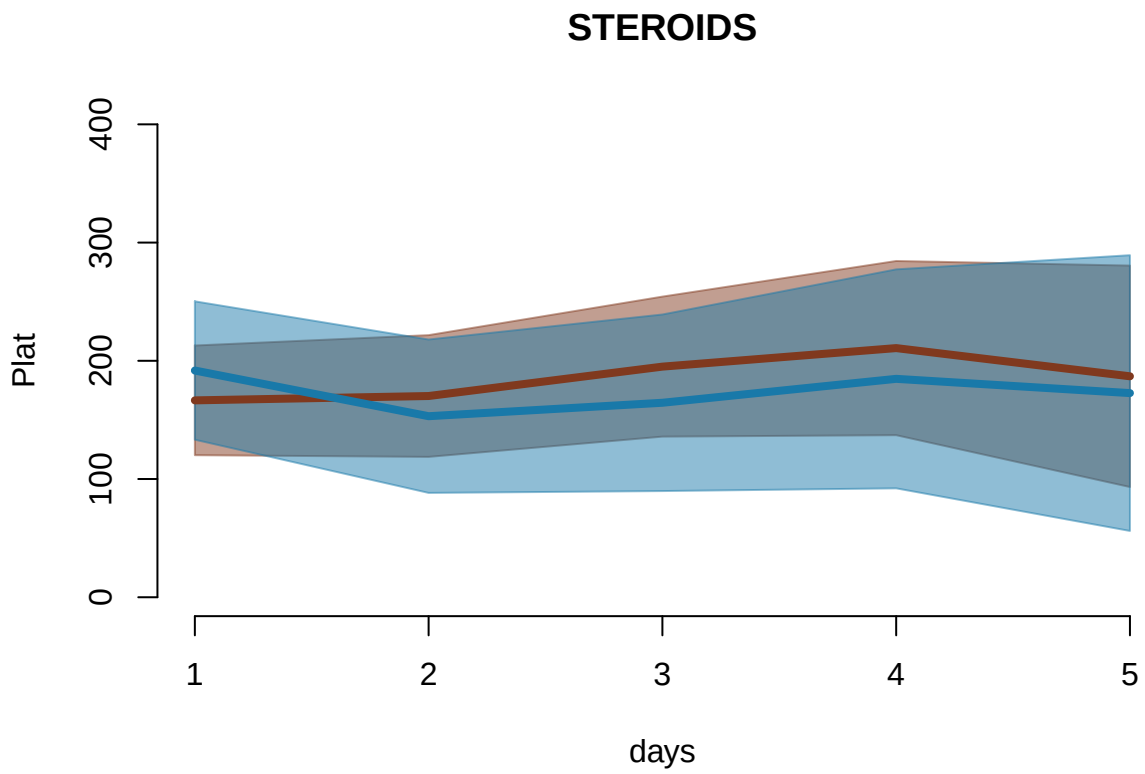
```
##      2.5%     97.5%
## -34.84511 100.34802

## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula: max_plat ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes +
##     (1 + day | ID)
##    Data: long_data_plat
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   3277.1   3317.0  -1627.5   3255.1      267
```

```
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.7548 -0.3688 -0.0758  0.3514  4.7496
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 7326     85.59
##           day         1468     38.32    -0.27
##  Residual             2700     51.96
## Number of obs: 278, groups:  ID, 65
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  266.712     45.772   5.827
## poly1         46.246     17.520   2.640
## poly2        -26.623      8.051  -3.307
## poly3        -22.169      7.304  -3.035
## age_yrs       -1.186      2.993  -0.396
## sex            6.772     25.137   0.269
## inotropes    -49.479     32.712  -1.513
##
## Correlation of Fixed Effects:
##           (Intr) poly1  poly2  poly3  ag_yrs sex
## poly1      0.216
## poly2      0.017  0.178
## poly3      0.024  0.082  0.193
## age_yrs   -0.611  0.015  0.000  0.002
## sex       -0.383  0.003 -0.009 -0.006  0.003
## inotropes -0.556  0.005  0.013 -0.021 -0.081  0.015
## Linear mixed model fit by maximum likelihood  ['lmerMod']
## Formula:
## max_plat ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
##     age_yrs + sex + inotropes + (1 + day | ID)
##    Data: long_data_plat
## Control: lmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##   3280.4   3363.8  -1617.2   3234.4      255
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.6762 -0.4365 -0.0401  0.3527  4.7004
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 6967     83.47
##           day         1338     36.57    -0.25
##  Residual             2495     49.95
## Number of obs: 278, groups:  ID, 65
##
## Fixed effects:
##                         Estimate Std. Error t value
```

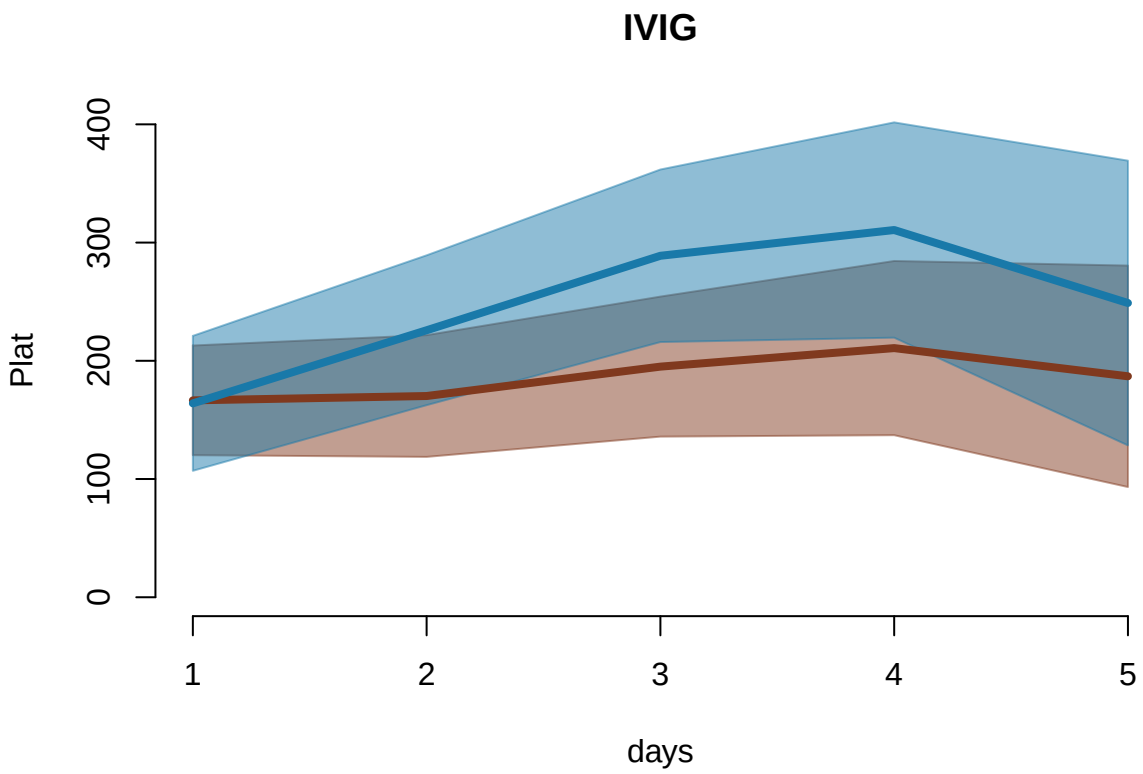
```
## (Intercept)             237.213     51.508   4.605
## poly1                    24.745     38.528   0.642
## poly2                   -17.807     17.651  -1.009
## poly3                   -18.701     16.628  -1.125
## IVIG                     70.208     44.461   1.579
## STEROIDS                 -1.473     46.095  -0.032
## IMMUNOMODULATION        -37.726     41.364  -0.912
## age_yrs                  -1.055      2.974  -0.355
## sex                      -5.135     26.204  -0.196
## inotropes               -62.615     33.758  -1.855
## poly1:IVIG               55.622     48.008   1.159
## poly1:STEROIDS          -27.771     50.089  -0.554
## poly1:IMMUNOMODULATION  -21.924     44.270  -0.495
## poly2:IVIG              -59.206     22.013  -2.690
## poly2:STEROIDS           33.496     24.128   1.388
## poly2:IMMUNOMODULATION   43.423     19.941   2.178
## poly3:IVIG               -7.398     20.550  -0.360
## poly3:STEROIDS           -7.418     21.608  -0.343
## poly3:IMMUNOMODULATION   32.793     18.216   1.800

##
## Correlation matrix not shown by default, as p = 19 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## Data: long_data_plat
## Models:
## p3_plat: max_plat ~ (poly1 + poly2 + poly3) + age_yrs + sex + inotropes +
## p3_plat:     (1 + day | ID)
## p4_plat: max_plat ~ (poly1 + poly2 + poly3) * (IVIG + STEROIDS + IMMUNOMODULATION) +
## p4_plat:     age_yrs + sex + inotropes + (1 + day | ID)
##         npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## p3_plat   11 3277.1 3317.0 -1627.5   3255.1
## p4_plat   23 3280.4 3363.8 -1617.2   3234.4 20.697 12    0.05499 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## [1] "comparing p1 with p2"

##      2.5%     97.5%
## -33.06324 101.38996
```

So we proceed to the modelling of fitted curves.

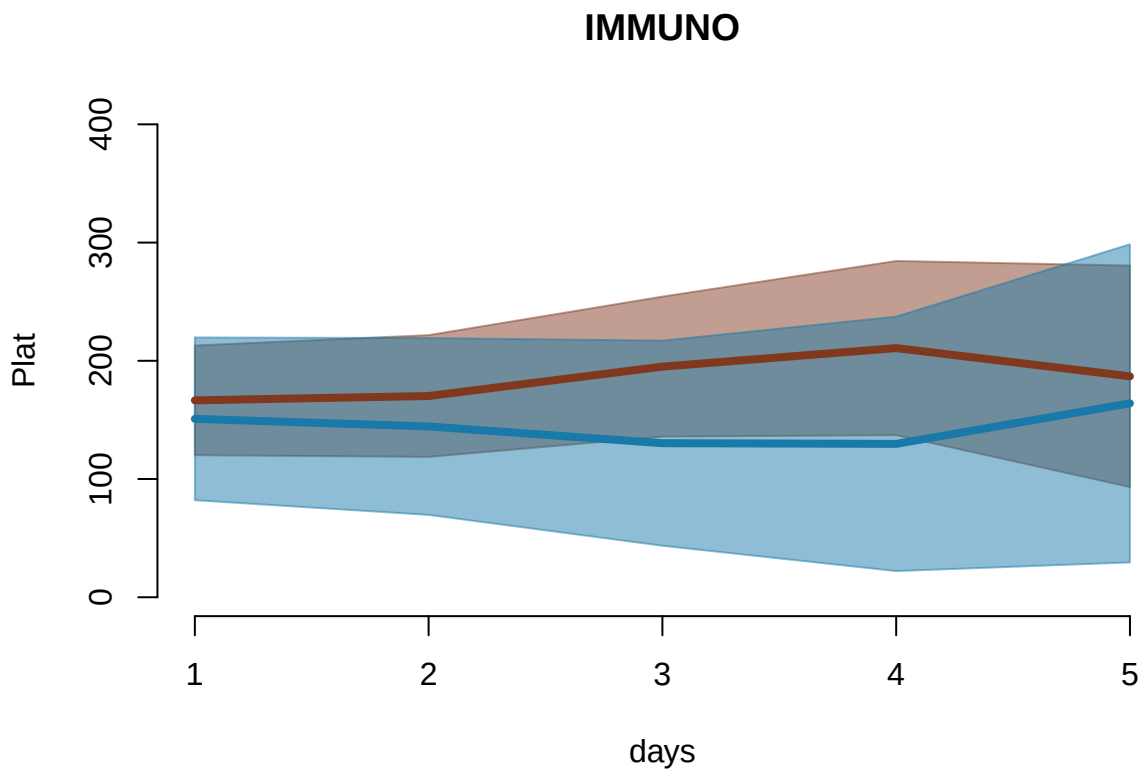
```
## [1] "about to plot..."
```



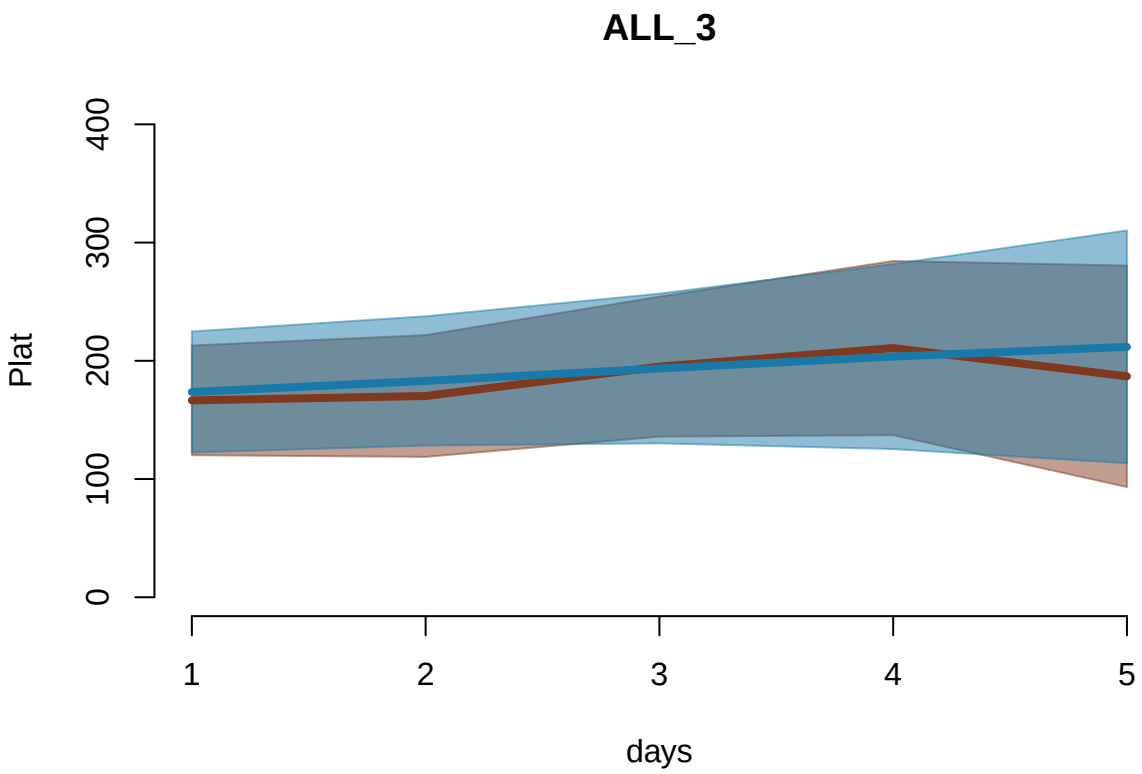

```
## [1] "about to plot..."
```



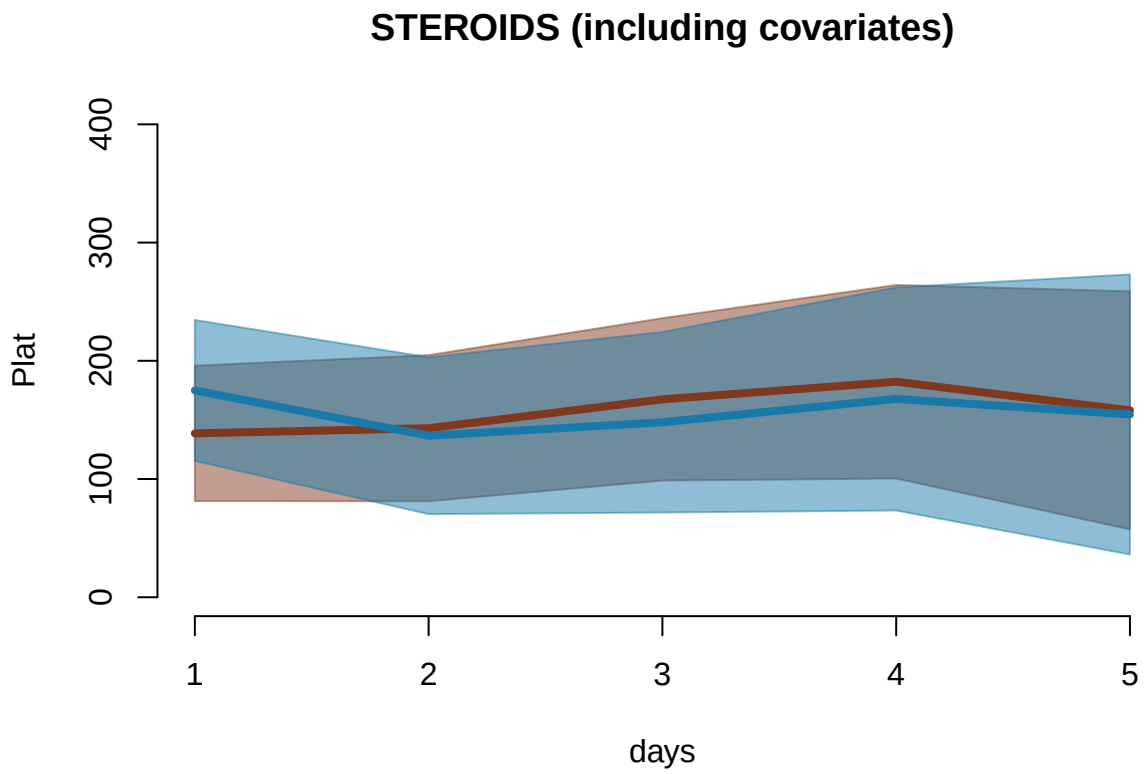

```
## [1] "about to plot..."
```



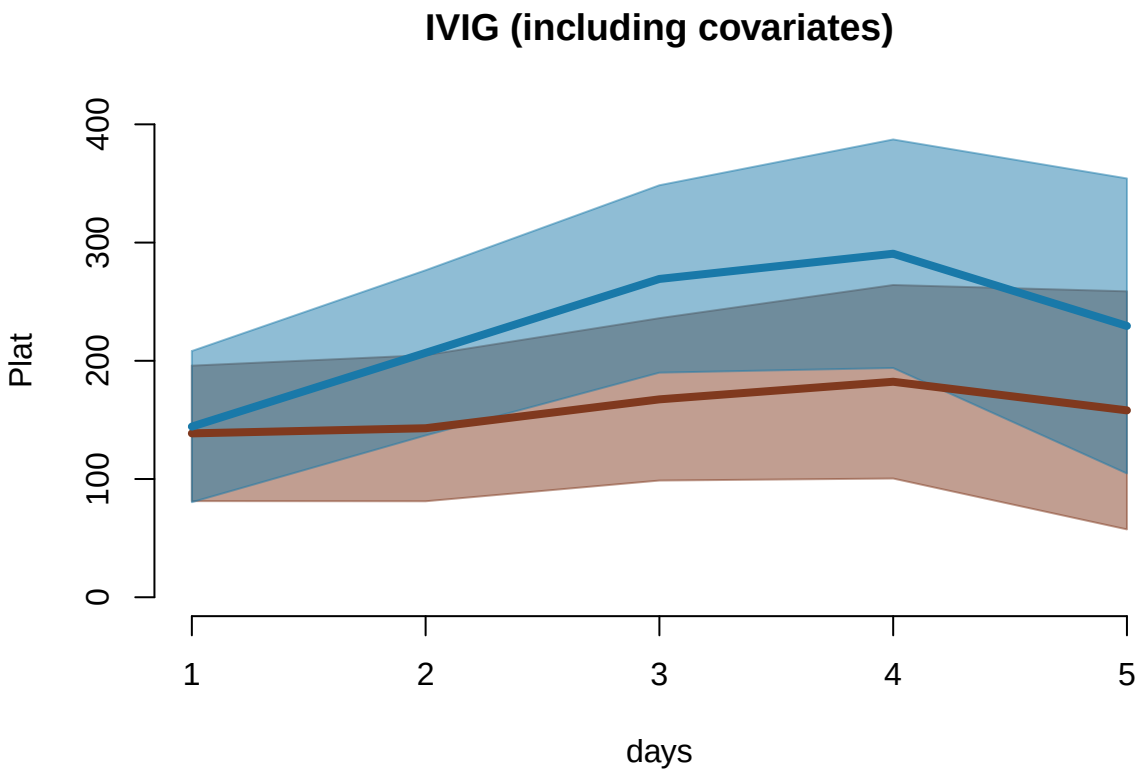

```
## [1] "about to plot..."
```



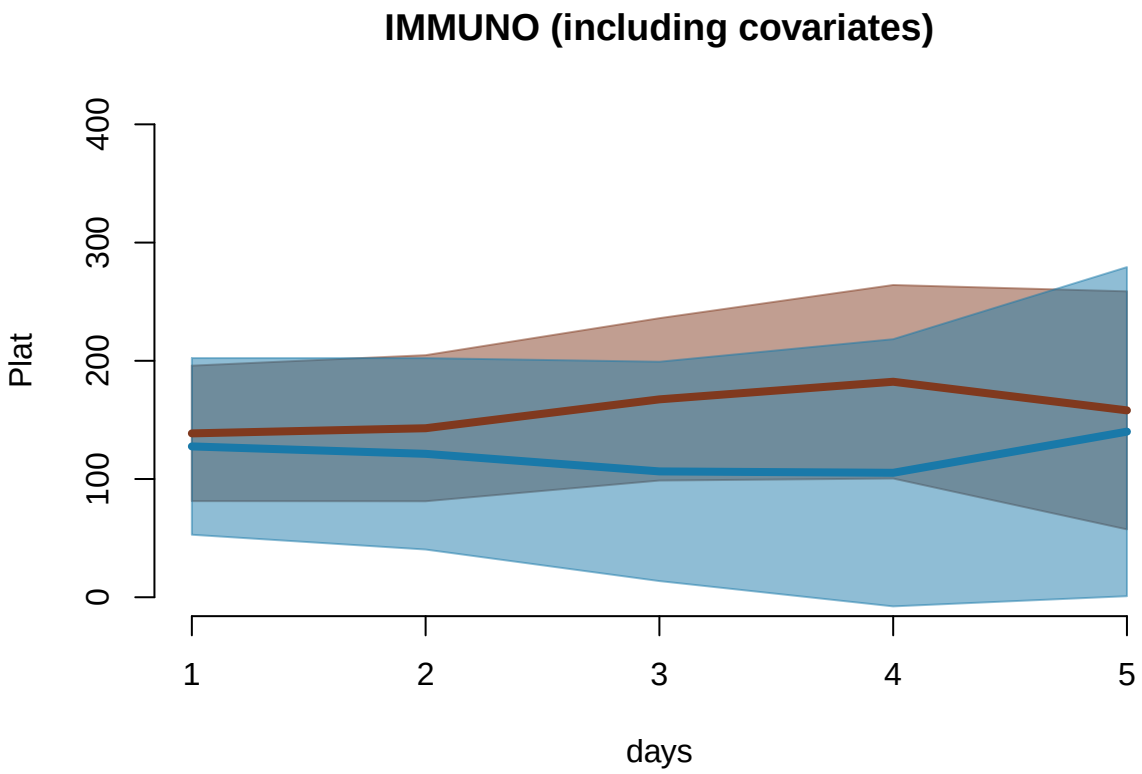

```
## [1] "about to plot..."
```



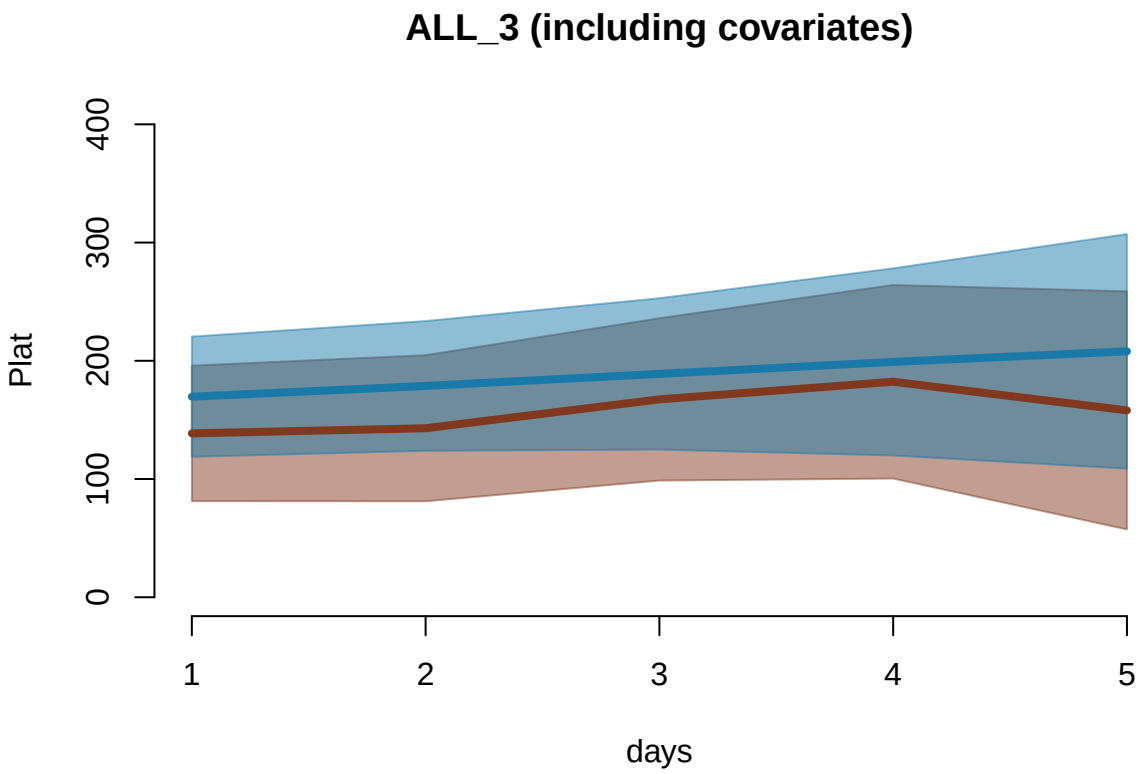

```
## [1] "about to plot..."
```




```
## [1] "about to plot..."
```




```
## [1] "about to plot..."
```



Again, no clear evidence that any of the treatments have any effect on these parameters.

```
## [1] "about to plot..."

## pdf
##   2

## [1] "about to plot..."

## pdf
##   2

## [1] "about to plot..."

## pdf
##   2

## [1] "about to plot..."

## pdf
##   2

## [1] "about to plot..."

## pdf
##   2

## [1] "about to plot..."

## pdf
##   2

## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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

```
## [1] "about to plot..."
```

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