Solution Growth of guinea pigs exercise (R software)

Contents

NOTE: This document proposes an R syntax giving the necessary outputs to answer to the questions of the exercise. The focus here is then on the implementation in R software and not on the interpretation or the validity of the results: we refer to the SAS solution for a discussion of the two latter points.

Because of the multiplicity of the packages in R, there are often several ways to perform a given operation (e.g. fitting a mixed model or converting a dataset from the long to the wide format). Some can be more efficient (in term of computation time or memory usage), other can be closer to the natural language or enabling a concise syntax. We don't claim to propose here the "best" R syntax but we tried to provide an readable syntax that could be re-used in other problems. In some cases an alternative syntax, usually more complex but more efficient/generalisable, is proposed in appendix.

First, load the necessary packages

```
> library(nlme)
> library(lattice)
>
> # optional
> library(ggplot2)
```

Question 1: Data import

```
> df.data_vitamin <- read.table("http://publicifsv.sund.ku.dk/~jufo/courses/repeated15/vitamin.txt",
+                         header = TRUE, na.strings = ".")
```

Rename the levels of the variable group:

```
> df.data_vitamin$grp <- factor(df.data_vitamin$grp, levels = 1:2, labels = c("C","T"))
```

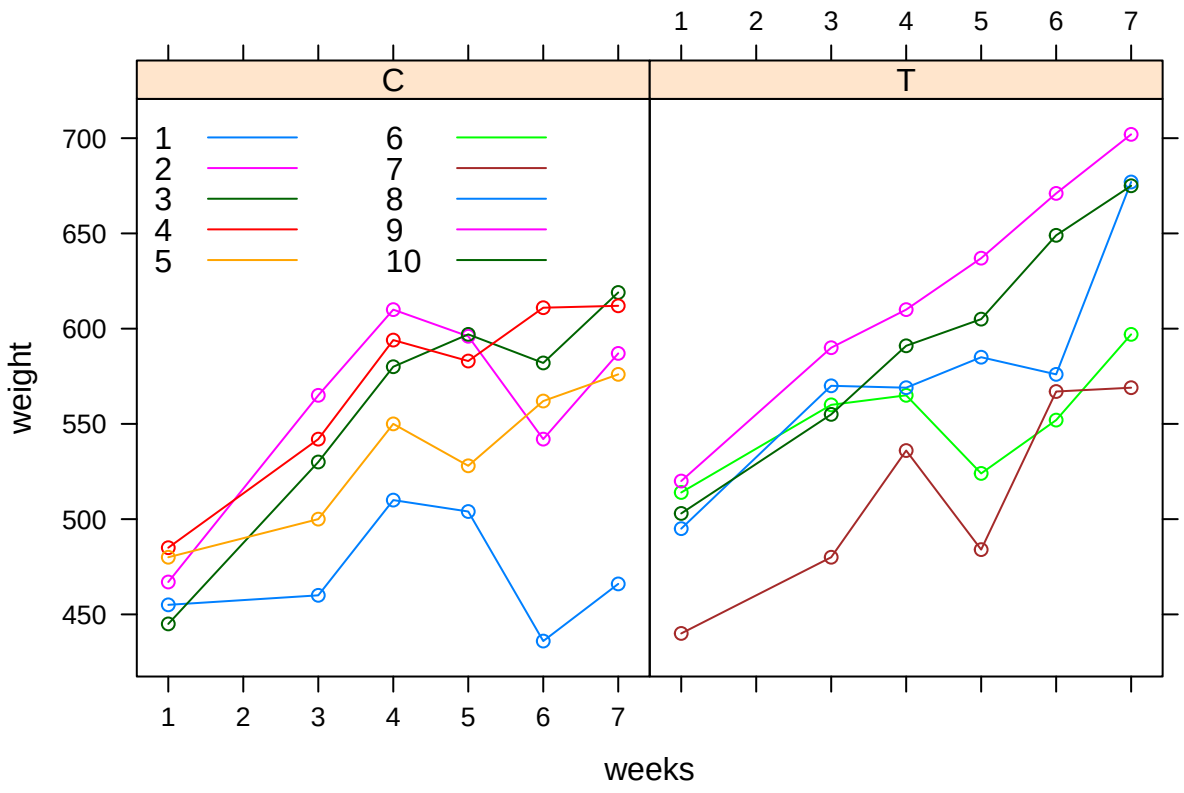
Assign correct type to other variables:

```
> df.data_vitamin$animal <- factor(df.data_vitamin$animal) # animal should not be numeric but factor
> df.data_vitamin$week.factor <- factor(df.data_vitamin$week) # we will use week as a factor variable
```

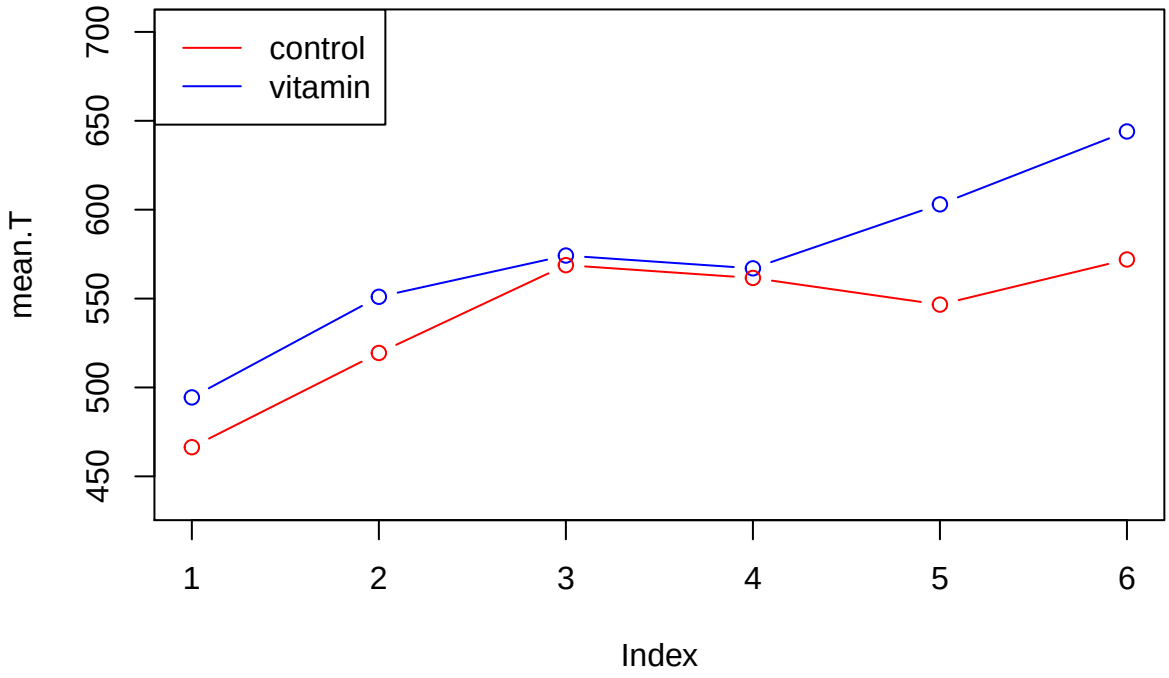
Question 2: Individual weight curves

Two widely used R packages for displaying longitudinal data: lattice and ggplot. Here we will use lattice package, which should be more intuitive at first sight.

```
> #### lattice
>
> # optinal argument (personnalized display)
> auto.key <- list(x = 0, y = 0.9, cex = 1,
+                  lines = TRUE, points = FALSE, columns = 2)
>
> ## longitudinal display with a pannel for each group
> xyplot(weight ~ week | grp, groups = animal,
+                 data = df.data_vitamin,
+                 auto.key = auto.key,
+                 type = "b" , xlab = "weeks", ylab = "weight")
```




```
> ## longitudinal display for all animals
> # xyplot(weight ~ week, groups = animal,
> # data = df.data_vitamin,
> # auto.key = auto.key, # optional
> # type = "b" ,xlab = "weeks",ylab = "weight")
>
> ## longitudinal display with a pannel for each individual
> # xyplot(weight ~ week | animal,
> # data = df.data_vitamin,
> # type = "b" , xlab = "weeks", ylab = "weight")
>
> ## display the mean value per group
> # indexes containing the position of each group within the dataset
> indexT <- which(df.data_vitamin$grp == "T")
> indexC <- which(df.data_vitamin$grp == "C")
>
> ## compute the mean value within each group for each week using tapply
> mean.T <- tapply(df.data_vitamin[indexT,"weight"], df.data_vitamin[indexT,"week"], mean)
> mean.C <- tapply(df.data_vitamin[indexC,"weight"], df.data_vitamin[indexC,"week"], mean)
>
> plot(mean.T,  ylim = range(df.data_vitamin$weight),
+      type = "b", col = "blue")
> points(mean.C,  type = "b", col = "red")
> legend("topleft", legend = c("control","vitamin"), col = c("red","blue"), lty = 1)
```




```
> # See appendix A for an alternative using ggplot2
```

Question 3: CS correlation structure

First, set the reference level for each categorical explanatory variable:

```
> df.data_vitamin$grp <- relevel(df.data_vitamin$grp, ref = "T")
> df.data_vitamin$week.factor <- relevel(df.data_vitamin$week.factor, ref = "7")
```

Test the interaction:

```
> gls.CSinteraction <- gls(weight ~ week.factor + grp + grp:week.factor - 1,
+                          data = df.data_vitamin,
+                          correlation = corCompSymm(form = ~1 | animal))
>
> anova(gls.CSinteraction, type = "marginal")
```

```
Denom. DF: 48
                numDF   F-value p-value
week.factor         6 176.94625  <.0001
grp                 1   5.71965  0.0207
week.factor:grp     5   2.60390  0.0366
```

```
> # See appendix B for explaination about the choice of the type argument
```

Very close to SAS output except for the p.value (no correction in R for the degree of freedom), e.g. :

```
                             R       SAS
-2 log-likelihood -491.0611344 -491.0600
sigma2_11         2265.8721193 2265.8700
sigma2_12         1570.9659350 1570.9600
betaT              -72.0000000  -72.0000
sd.betaT            30.1056282   30.1056
p_value.betaT        0.0207428    0.0313
```

Test the identity between the groups:

```
> gls.CS1 <- gls(weight ~ week.factor + grp:week.factor - 1,
+                data = df.data_vitamin,
+                correlation = corCompSymm(form = ~1 | animal))
>
> anova(gls.CS1, type = "marginal")
```

```
Denom. DF: 48
                numDF  F-value p-value
week.factor         6 176.9462  <.0001
week.factor:grp     6   2.4411  0.0386
```

```
> intervals(gls.CS1, which = "coef")$coef["week.factor7:grpC",,drop = FALSE]
```

```
                      lower est.     upper
week.factor7:grpC -132.5314  -72 -11.46858
```

Question 4: Analysis stratified by period

```
> df.data_vitamin$period <- factor(df.data_vitamin$week > 4, levels = c(FALSE, TRUE),
+                           labels = c("early", "late"))
>
> gls.CS1_early <- gls(weight ~ week.factor + grp:week.factor - 1,
+                      data = df.data_vitamin,
+                      correlation = corCompSymm(form = ~1 | animal),
+                      subset = period == "early" )
>
> anova(gls.CS1_early, type = "marginal")
```

```
Denom. DF: 24
                numDF  F-value p-value
week.factor         3 532.2168  <.0001
week.factor:grp     3   1.3690  0.2761
```

```
> gls.CS1_late <- gls(weight ~ week.factor + grp:week.factor - 1,
+                     data = df.data_vitamin,
+                     correlation = corCompSymm(form = ~1 | animal),
+                     subset = period == "late" )
>
> anova(gls.CS1_late, type = "marginal")
```

```
Denom. DF: 24
                numDF   F-value p-value
week.factor         3 211.92731  <.0001
week.factor:grp     3   4.46683  0.0125
```

```
> gls.CSinteraction_late <- gls(weight ~ week.factor*grp,
+                               data = df.data_vitamin,
+                               correlation = corCompSymm(form = ~1 | animal),
+                               subset = period == "late" )
>
> anova(gls.CSinteraction_late, type = "marginal")
```

```
Denom. DF: 24
                numDF  F-value p-value
(Intercept)         1 617.9452  <.0001
week.factor         2  14.3736  0.0001
grp                 1   3.8620  0.0611
week.factor:grp     2   5.8746  0.0084
```

Question 5: Test treatment effect on period 2

We first define the new variables:

```
> df.data_vitamin$week5 <- (df.data_vitamin$week == 5) * (df.data_vitamin$grp == "T")
> df.data_vitamin$week6 <- (df.data_vitamin$week == 6) * (df.data_vitamin$grp == "T")
> df.data_vitamin$week7 <- (df.data_vitamin$week == 7) * (df.data_vitamin$grp == "T")
> df.data_vitamin$number_week <- sapply( (df.data_vitamin$week) * (df.data_vitamin$grp == "T"),
+                                 function(x){max(x - 4, 0)}
+                                 )
>
> gls.CS2 <- gls(weight ~  week.factor + week5 + week6 + week7 - 1,
+                data = df.data_vitamin,
+                correlation = corCompSymm(form = ~1 | animal))
>
> anova(gls.CS2, type = "marginal")
```

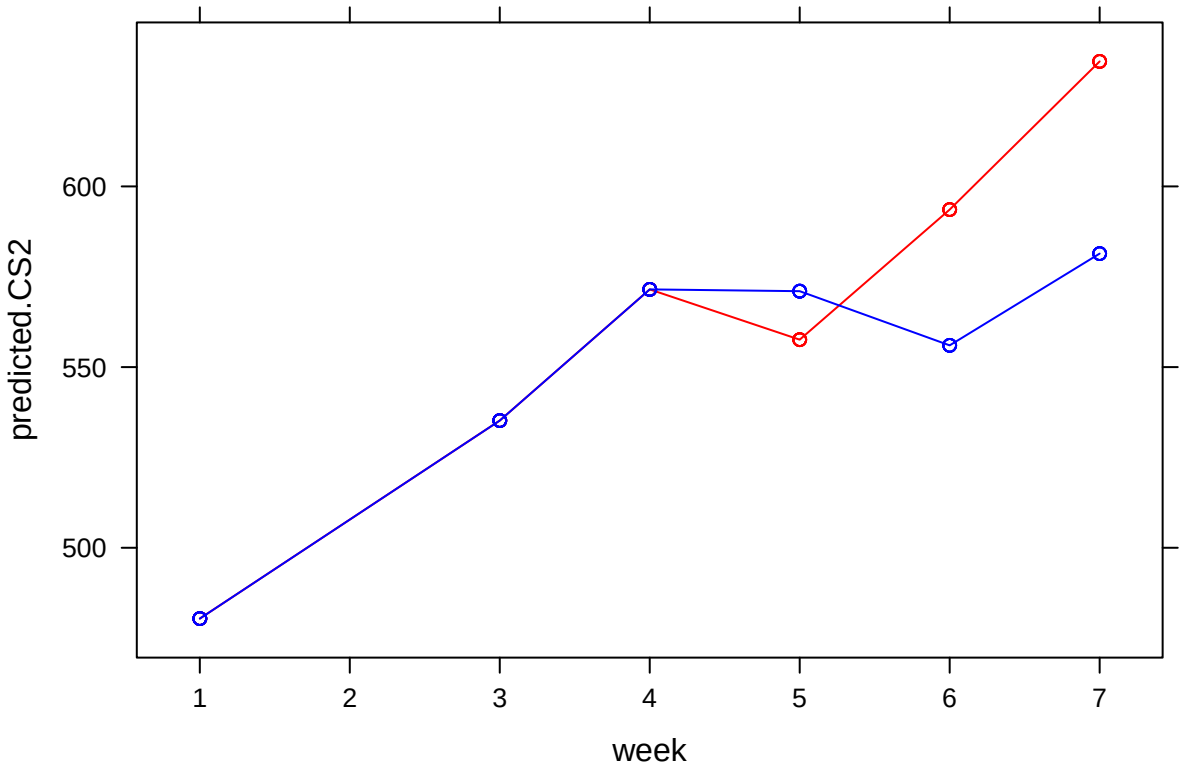
```
Denom. DF: 51
            numDF   F-value p-value
week.factor     6 302.40596  <.0001
week5           1   0.50937  0.4787
week6           1   3.99408  0.0510
week7           1   7.99838  0.0067
```

```
> intervals(gls.CS2)$coef[c("week5","week6","week7"),]
```

```
            lower      est.    upper
week5 -51.1704322 -13.42029 24.32985
week6  -0.1704322  37.57971 75.32985
week7  15.4295678  53.17971 90.92985
```

We then can use the predict function to extract the predicted value for each observation and draw the mean response:

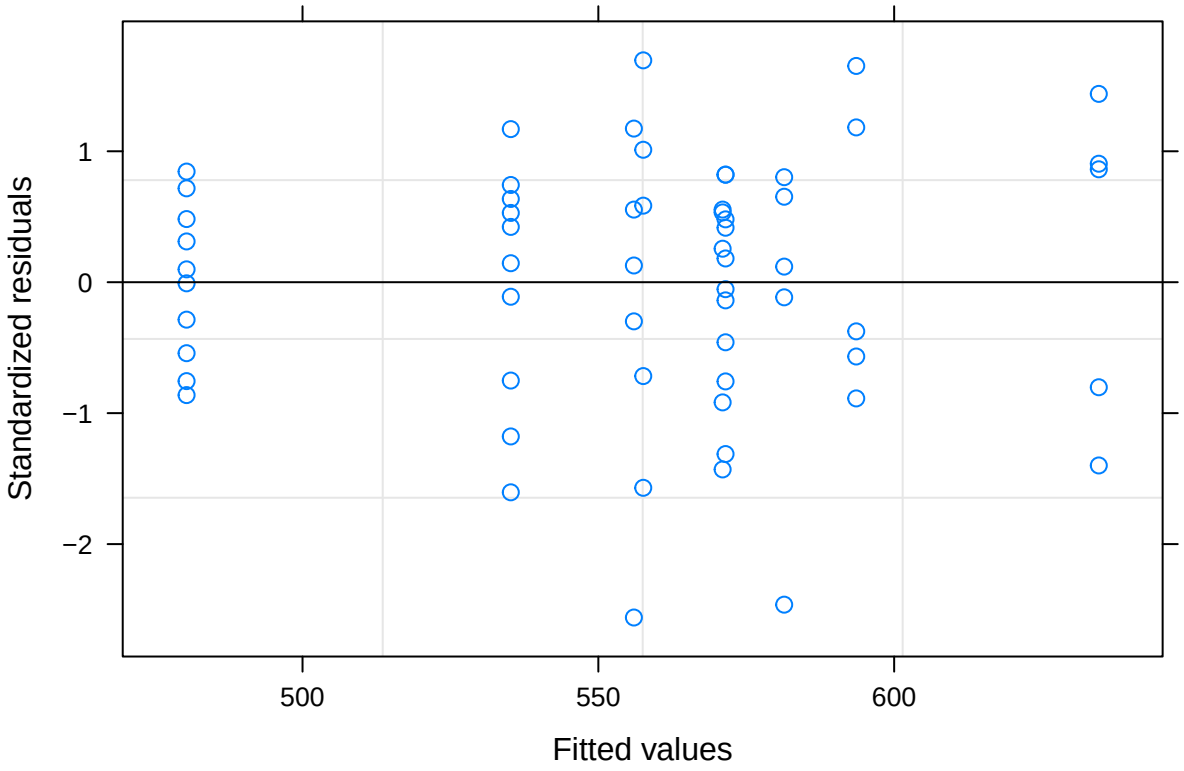
```
> df.Pred_vitamin <- data.frame(df.data_vitamin,
+                               predicted.CS2 = predict(gls.CS2, type = "response"))
>
> xyplot(predicted.CS2 ~ week, group = grp,
+        data = df.Pred_vitamin[order(df.Pred_vitamin$week),],
+        type = "b", col = c("red", "blue"))
```



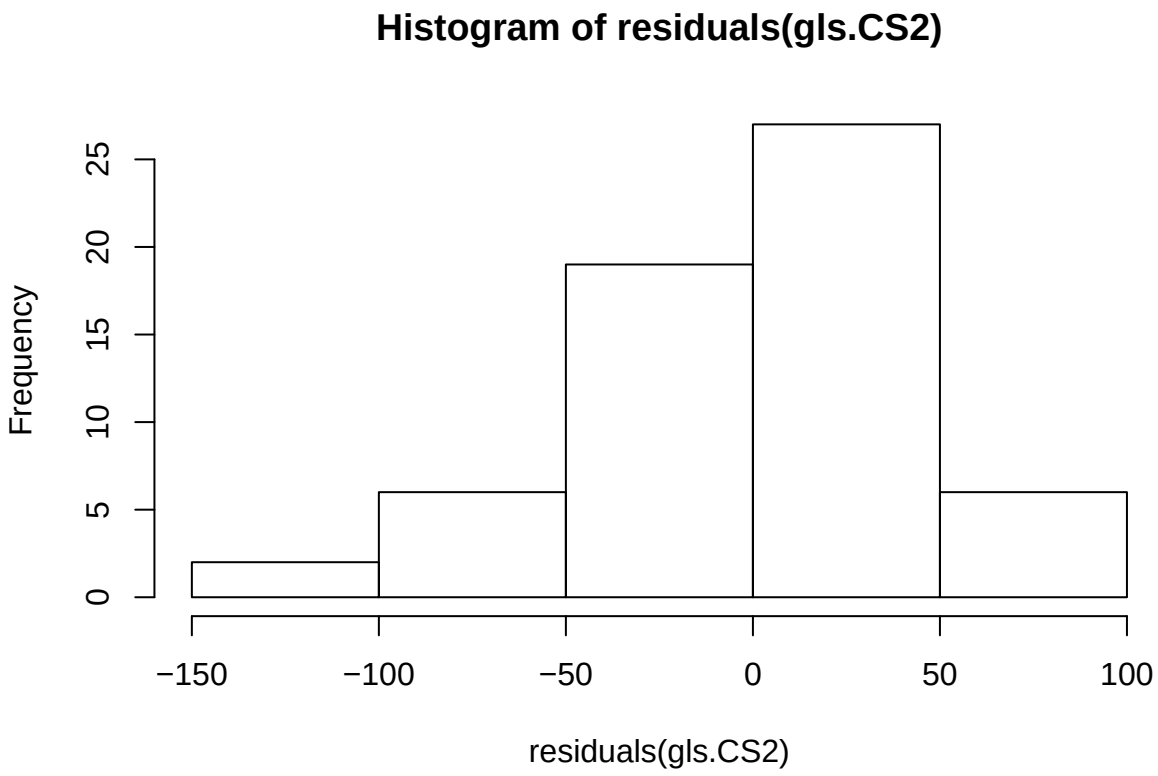
Despite it leads here to the same result, instead of using the fitted values for the observations we should estimate the fitted mean response for the various times (and various levels of treatment). This is done in appendix C but as it requires a more complex syntax so we will stick to this "quick way" in the following.

The nlme package provides also some diagnostic tools:

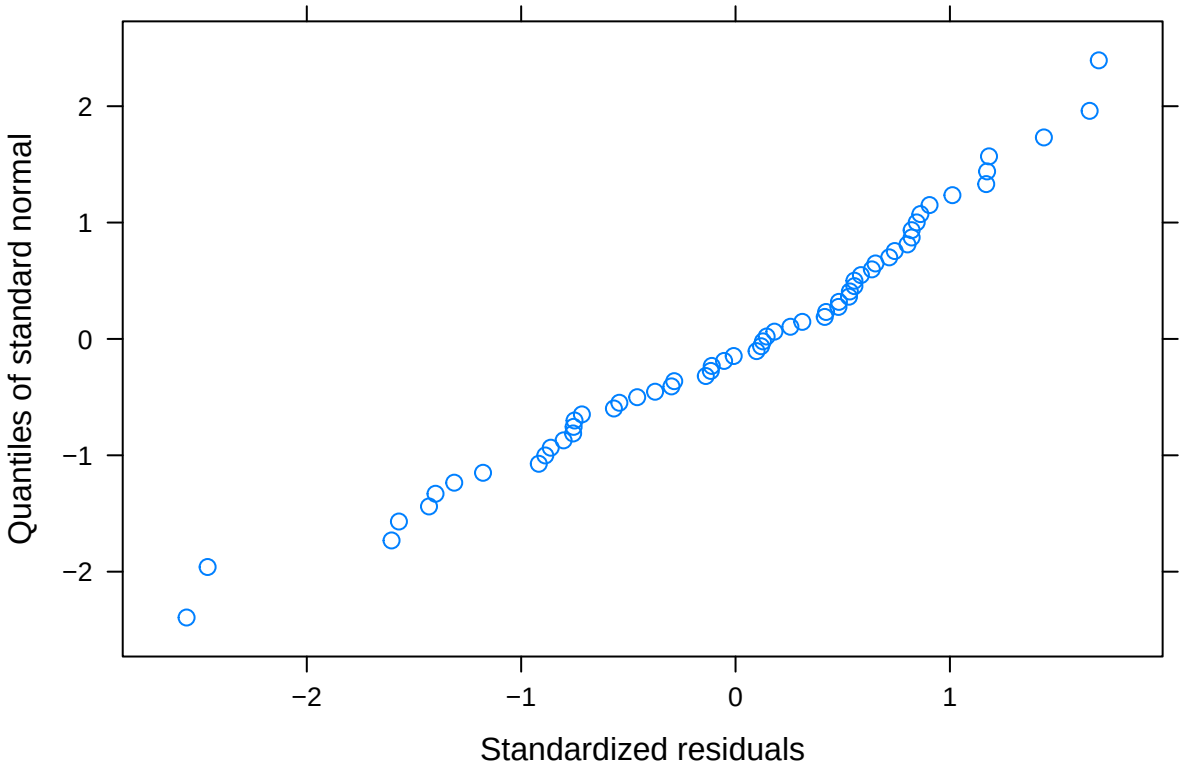
```
> ## diagnostics
> # scatter plot of the residuals
> plot(gls.CS2)
```




```
> # histogram of the residuals
> hist(residuals(gls.CS2))
```




```
> # qqplot of the residuals
> qqnorm(gls.CS2)
```




```
> # autocorrelation between residuals
> ACF(gls.CS2)
```

```
  lag       ACF
1   0 1.0000000
2   1 0.8160501
3   2 0.7526659
4   3 0.6048586
5   4 0.6212563
6   5 0.3755540
```

We then proceed in the same way for model (b):

```
> gls.CS3 <- gls(weight ~  week.factor + number_week - 1,
+                data = df.data_vitamin,
+                correlation = corCompSymm(form = ~1 | animal))
>
> anova(gls.CS3, type = "marginal")
```

```
Denom. DF: 53
            numDF   F-value p-value
week.factor     6 312.02548  <.0001
number_week     1   9.45616  0.0033
```

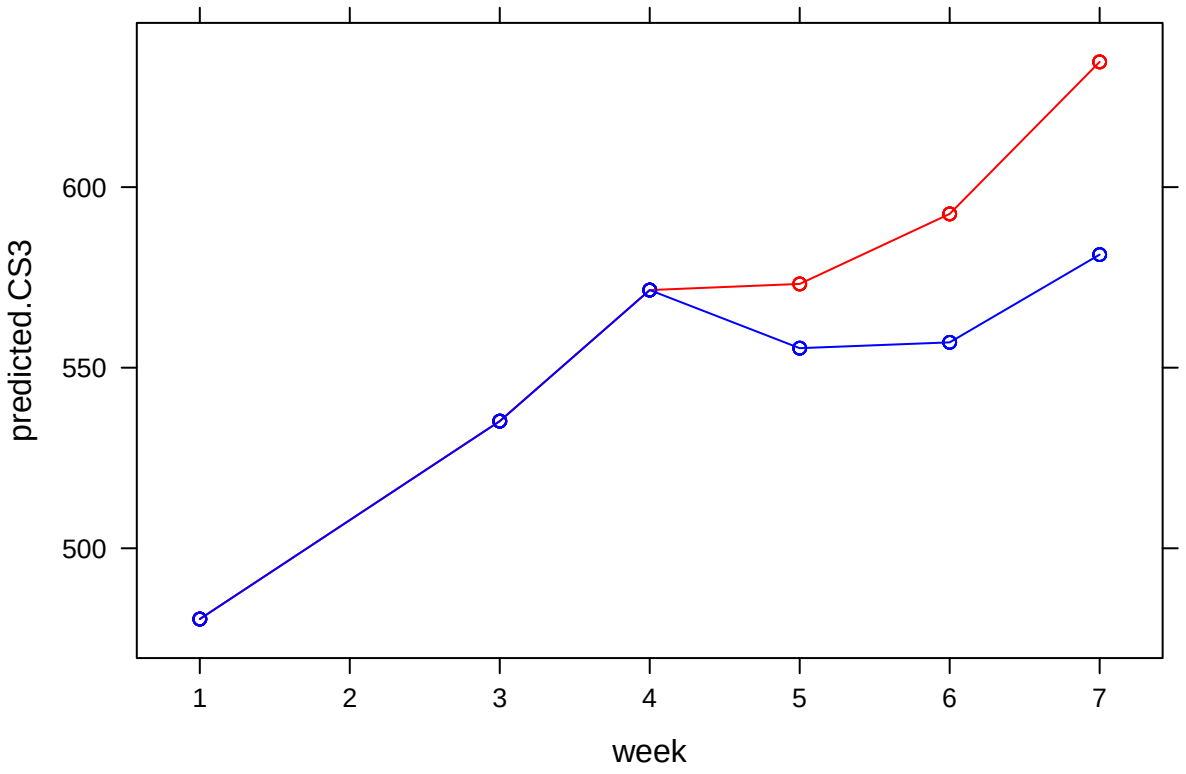
```
> intervals(gls.CS3)$coef["number_week",]
```

```
   lower     est.    upper
 6.18503 17.78620 29.38737
```

```
> intervals(gls.CS3)$coef["number_week",]*3
```

```
   lower     est.    upper
18.55509 53.35860 88.16211
```

```
> # WARNING: this is only valid because we display the IC for a single parameter
> # when several variable are implied we cannot add IC because we would
> # neclect the covariance between parameters.
>
> df.Pred_vitamin$predicted.CS3 <- predict(gls.CS3, type = "response")
>
> xyplot(predicted.CS3 ~ week, group = grp, data = df.Pred_vitamin[order(df.Pred_vitamin$week),],
+        type = "b", col = c("red", "blue"))
```



Question 6: Linearity of the treatment effect

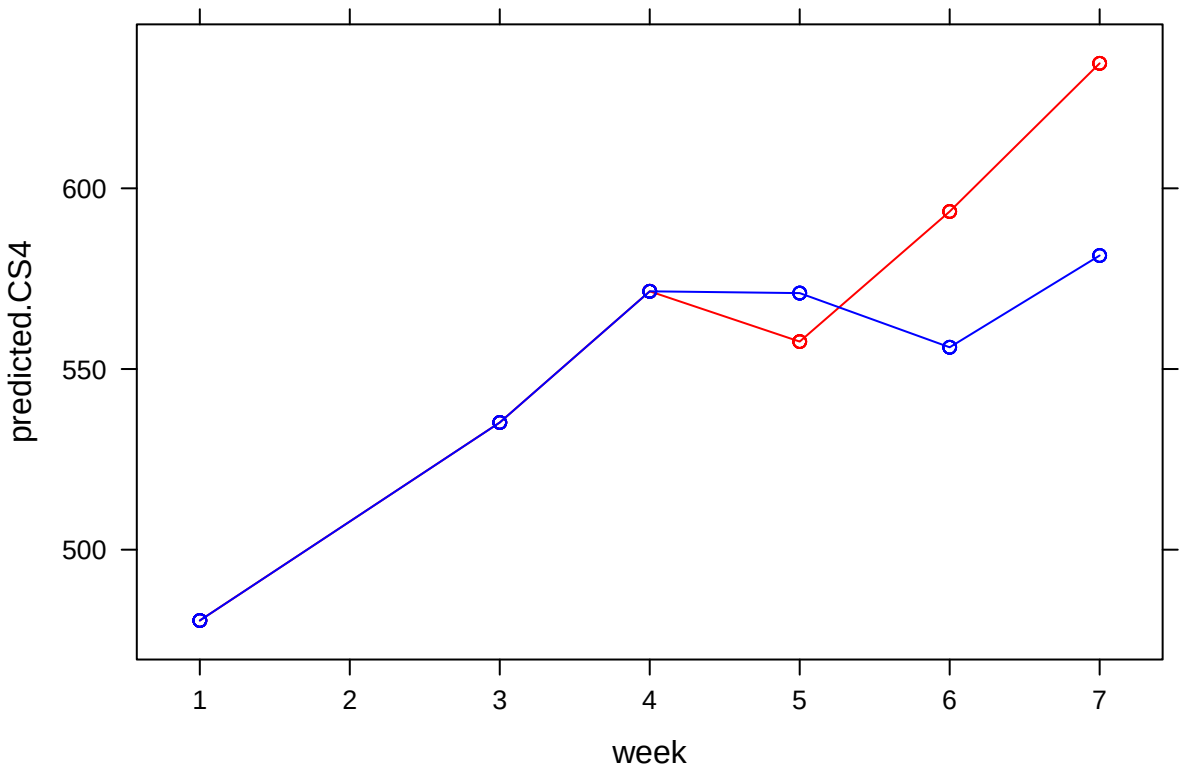
```
> gls.CS4 <- gls(weight ~  week.factor + number_week + week6 + week7 - 1,
+                data = df.data_vitamin,
+                correlation = corCompSymm(form = ~1 | animal))
>
> range(gls.CS4$fitted - gls.CS2$fitted) # identifcal fit
```

```
[1] -5.435368e-10  5.447873e-10
```

```
> # test week6 = 0 and week7 = 0
> anova( update(gls.CS3, method = "ML"),
+        update(gls.CS4, method = "ML"))
```

```
                              Model df      AIC      BIC    logLik   Test
update(gls.CS3, method = "ML")     1  9 600.3858 619.2349 -291.1929
update(gls.CS4, method = "ML")     2 11 600.8624 623.9002 -289.4312 1 vs 2
                               L.Ratio p-value
update(gls.CS3, method = "ML")
update(gls.CS4, method = "ML") 3.523315  0.1718
```

```
> #### display
> df.Pred_vitamin$predicted.CS4 <- predict(gls.CS4, type = "response")
>
> xyplot(predicted.CS4 ~ week, group = grp, data = df.Pred_vitamin[order(df.Pred_vitamin$week),],
+        type = "b", col = c("red", "blue"))
```



Question 7: Using an unstructured covariance matrix

```
> gls.UN4 <- gls(weight ~  week.factor + week6 + week7 + number_week - 1,
+                data = df.data_vitamin,
+                correlation = corSymm(form = ~1 | animal),
+                    weights = varIdent(form = ~1 | week.factor))
>
> getVarCov(gls.UN4)
```

```
Marginal variance covariance matrix
       [,1]    [,2]    [,3]    [,4]    [,5]   [,6]
[1,] 794.72  881.03  444.67  767.09  417.59  786.7
[2,] 881.03 1791.50 1205.00 1847.90 1213.90 1945.2
[3,] 444.67 1205.00 1052.90 1469.80 1444.30 1583.7
[4,] 767.09 1847.90 1469.80 2676.80 2114.30 2692.9
[5,] 417.59 1213.90 1444.30 2114.30 3428.30 2848.5
[6,] 786.70 1945.20 1583.70 2692.90 2848.50 3346.6
  Standard Deviations: 28.191 42.326 32.449 51.738 58.551 57.85
```

Noticeable difference with SAS output regarding the variance covariance matrix of the β:

```
                             R       SAS
-2 log-likelihood -464.1636611 -464.2000
sigma2_11          794.7159995  794.7100
sigma2_12          881.0303888  881.0200
betaT              -12.3275781  -12.3279
sd.betaT            14.9458637   20.3233
p_value.betaT        0.4133177    0.5609
```

```
> anova(gls.UN4, type = "marginal")
```

```
Denom. DF: 51
            numDF  F-value p-value
week.factor     6 3684.059  <.0001
week6           1   12.453  0.0009
week7           1    6.365  0.0148
number_week     1    0.680  0.4133
```

```
> n.coef <- length(coef(gls.UN4))
> Contrast <- matrix(0, nrow = 2, ncol = n.coef)
> colnames(Contrast) <- names(coef(gls.UN4))
>
> Contrast[1,"week6"] <- 1
> Contrast[2,"week7"] <- 1
> anova(gls.UN4, L = Contrast)
```

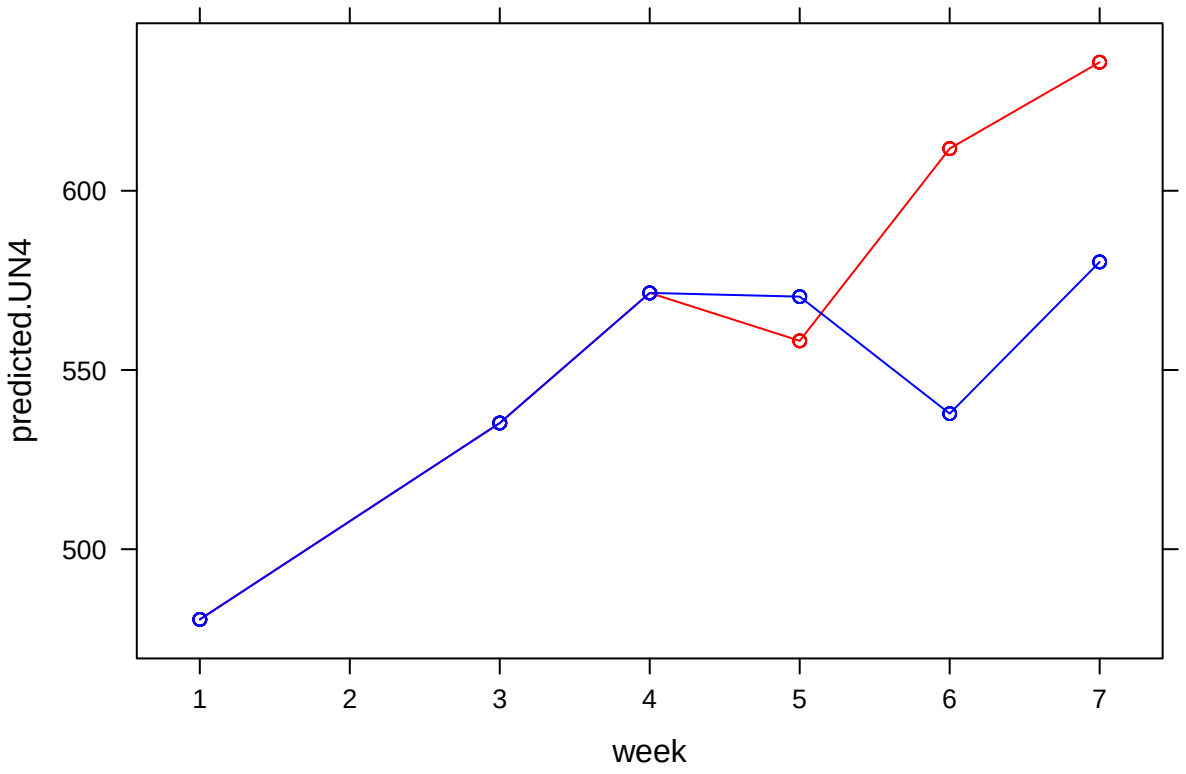
```
Denom. DF: 51
 F-test for linear combination(s)
  week6 week7
1     1     0
```

```
2     0     1
  numDF  F-value p-value
1     2 12.18008  <.0001
```

```
> anova(gls.UN4, gls.CS4)
```

```
        Model df      AIC      BIC    logLik   Test  L.Ratio p-value
gls.UN4     1 30 524.1637 582.1184 -232.0818
gls.CS4     2 11 539.6013 560.8514 -258.8007 1 vs 2 53.43764  <.0001
```

```
> #### display
> df.Pred_vitamin$predicted.UN4 <-  predict(gls.UN4, type = "reponse")
>
> xyplot(predicted.UN4 ~ week, group = grp, data = df.Pred_vitamin[order(df.Pred_vitamin$week),],
+        type = "b", col = c("red", "blue"))
```

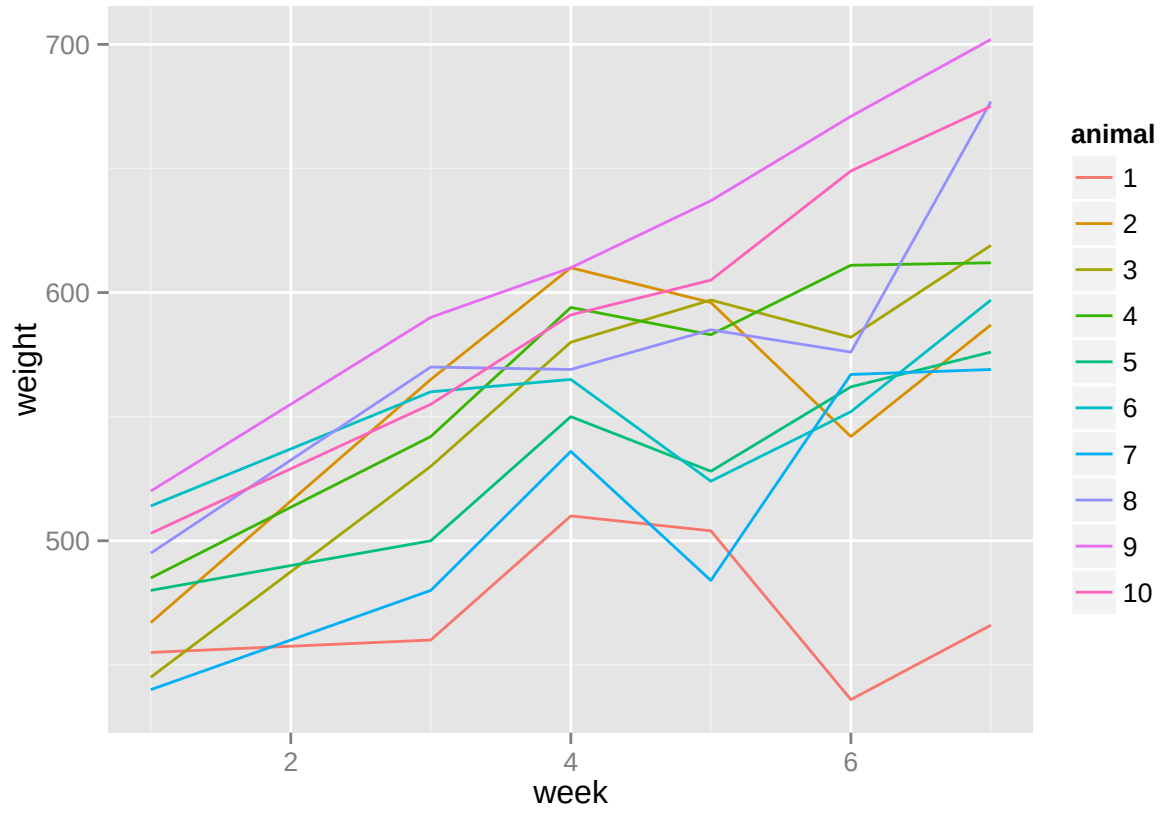


Appendix A: ggplot2 package

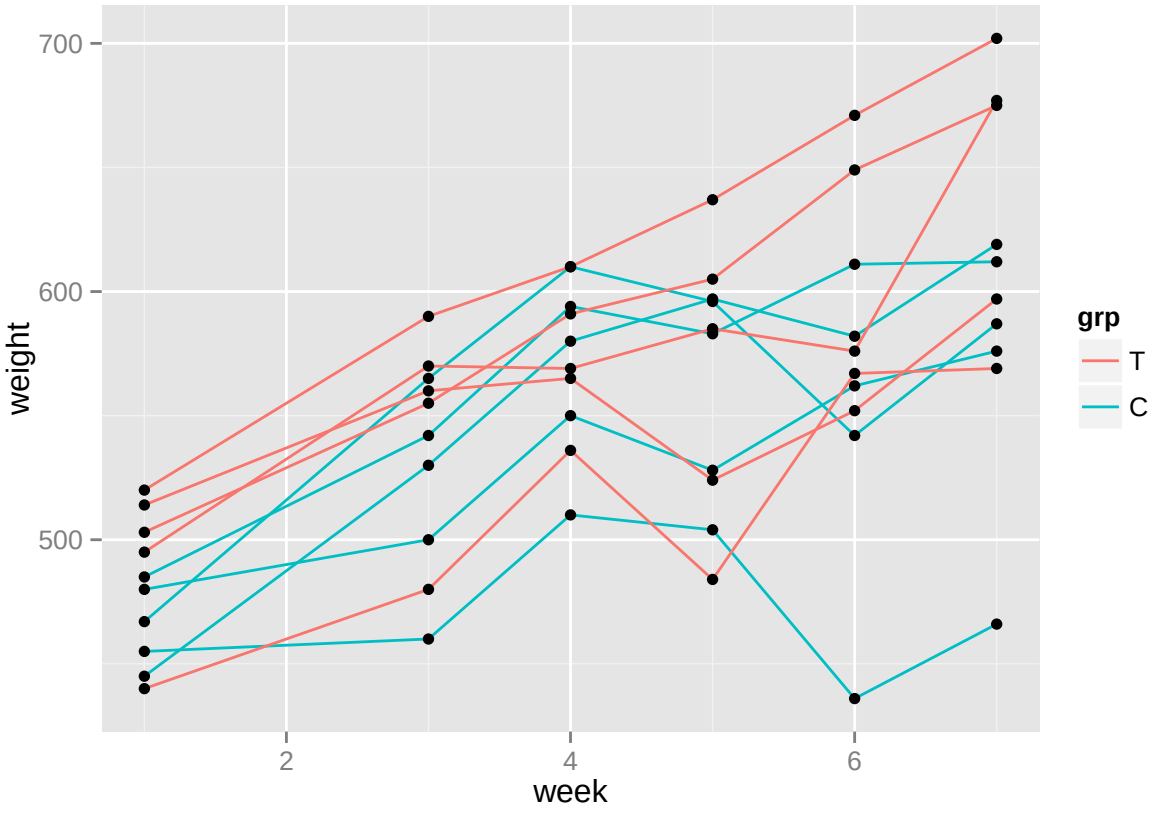
ggplot2 enables a nice and flexible display of the data using a specific grammar. The syntax may not be straightforward to understand/use at first sight and won't be explayed here. If you are interested, you can easily find tutorials on internet or look at the book: ggplot2: Elegant Graphics for Data Analysis, 2010, Hadley Wickham.

Here the answer to question 2 using the ggplot:

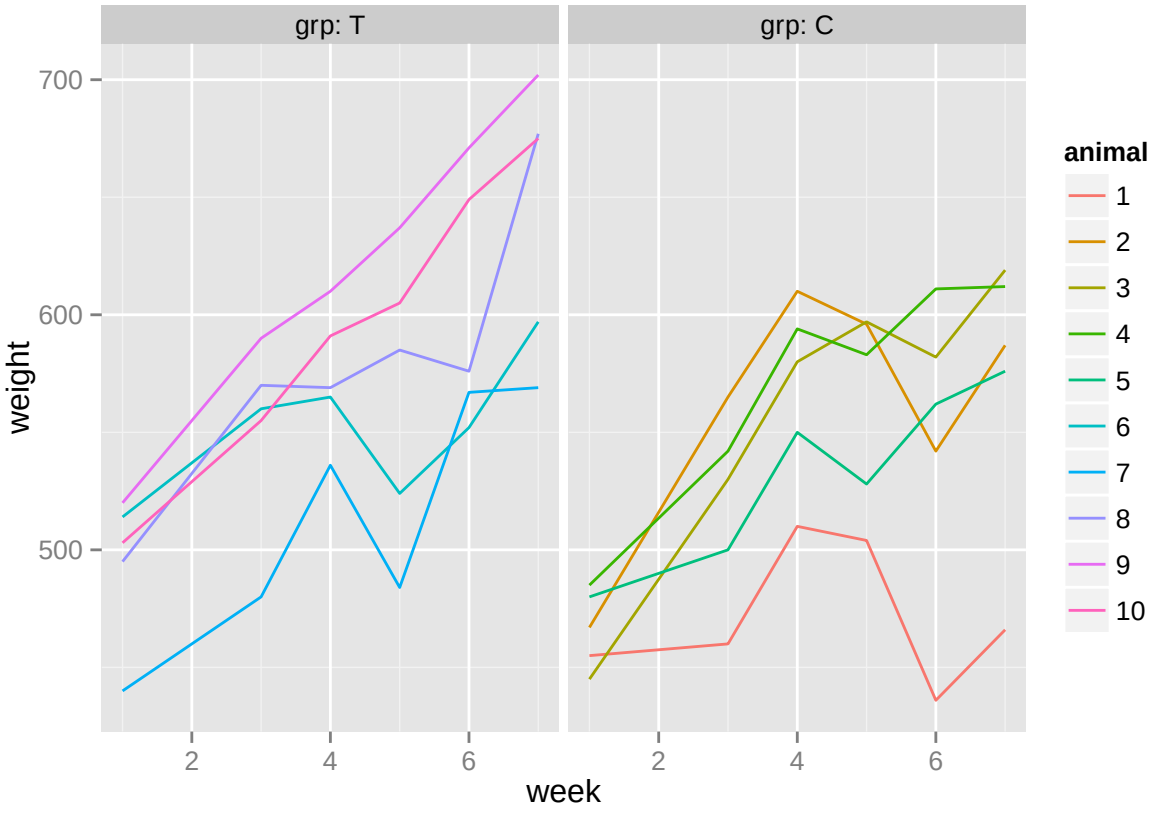
```
> gg.base <- ggplot(df.data_vitamin, aes(x = week, y = weight, group = animal))
>
> ## longitudinal display for all animals
> gg.idline <- gg.base + geom_line(aes(color = animal))
> gg.idline
```



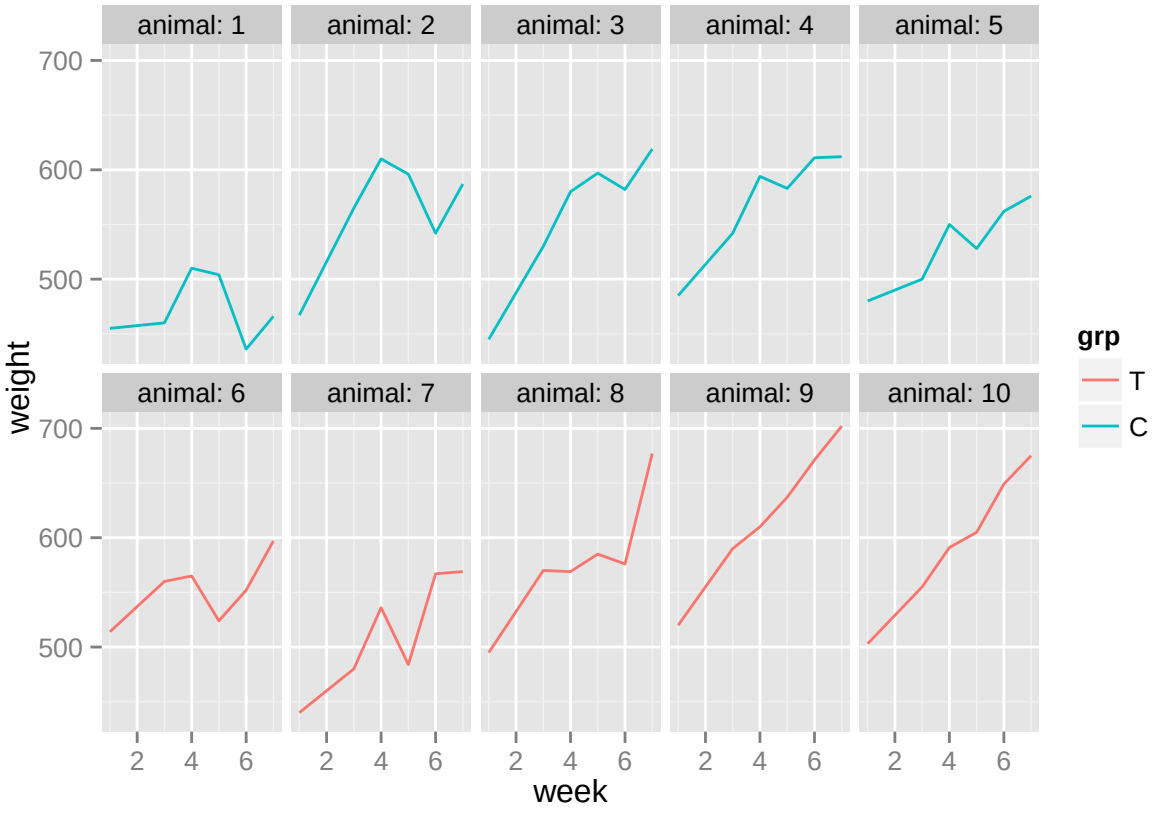

```
> gg.Gline <- gg.base + geom_line(aes(color = grp))
> gg.Gline + geom_point()
```



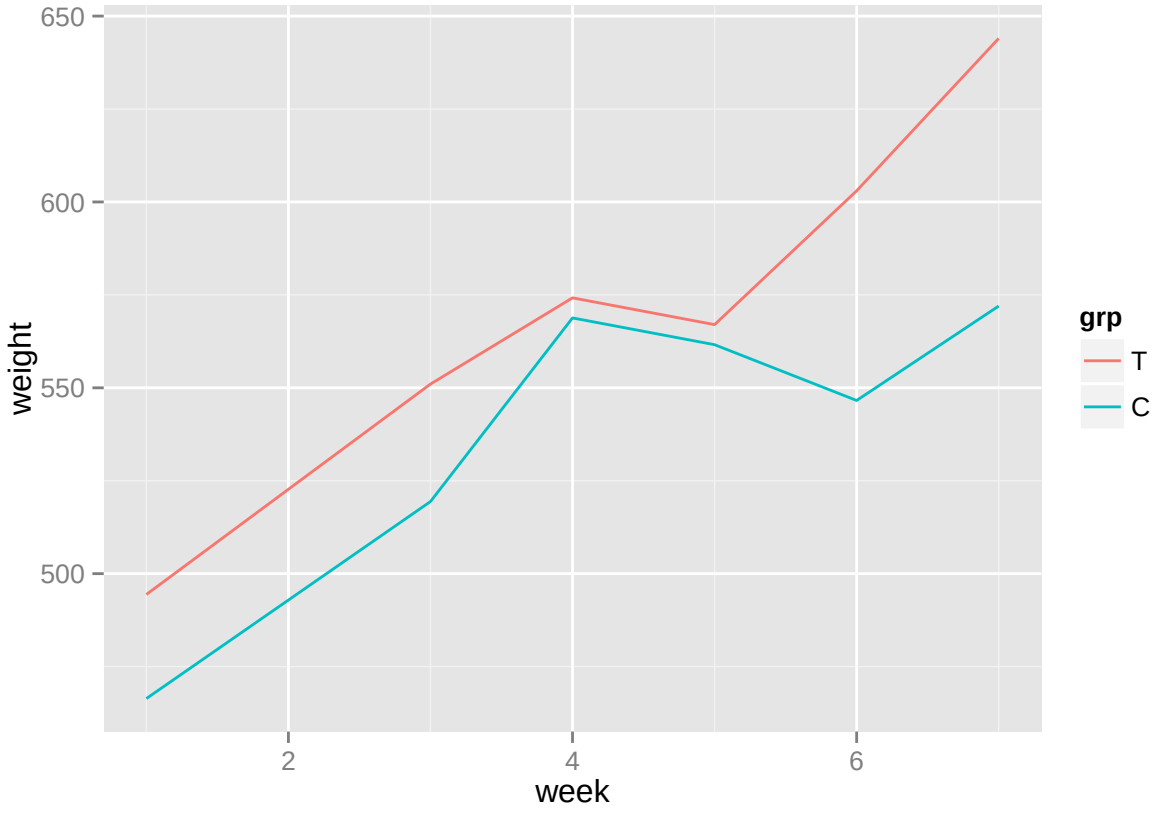

```
> ## longitudinal display with a pannel for each individual
> gg.idline + facet_grid(. ~ grp, labeller = label_both)
```



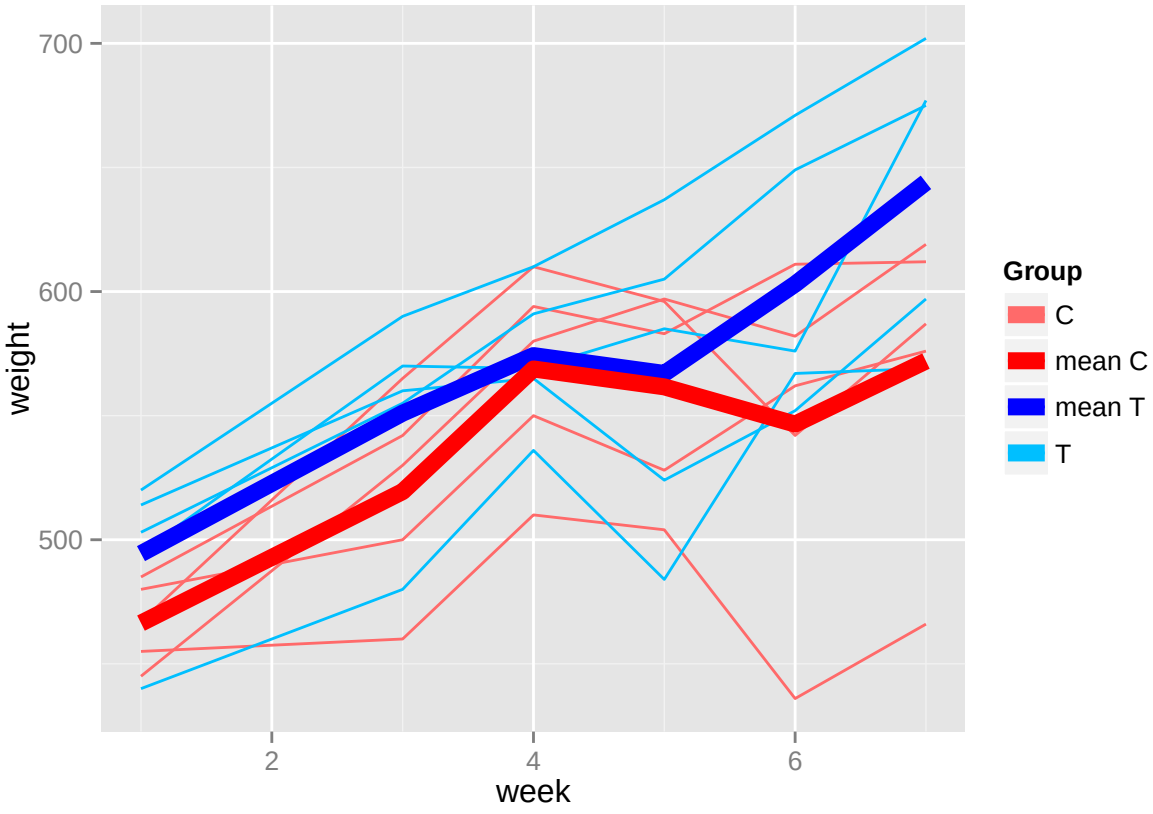

```
> ## longitudinal display with a panel for each individual
> df.data_vitamin$animal.label <- factor(paste0("animal: ", df.data_vitamin$animal),
+                                        levels = paste0("animal: ", 1:10))
>
> ggplot(df.data_vitamin, aes(x = week, y = weight, group = animal.label)) +
+ geom_line(aes(color = grp)) + facet_wrap(~ animal.label, ncol = 5)
```




```
> ## display the mean value per group
> gg.base + stat_summary(aes(group = grp, color = grp),
+                        geom = "line", fun.y = mean)
```

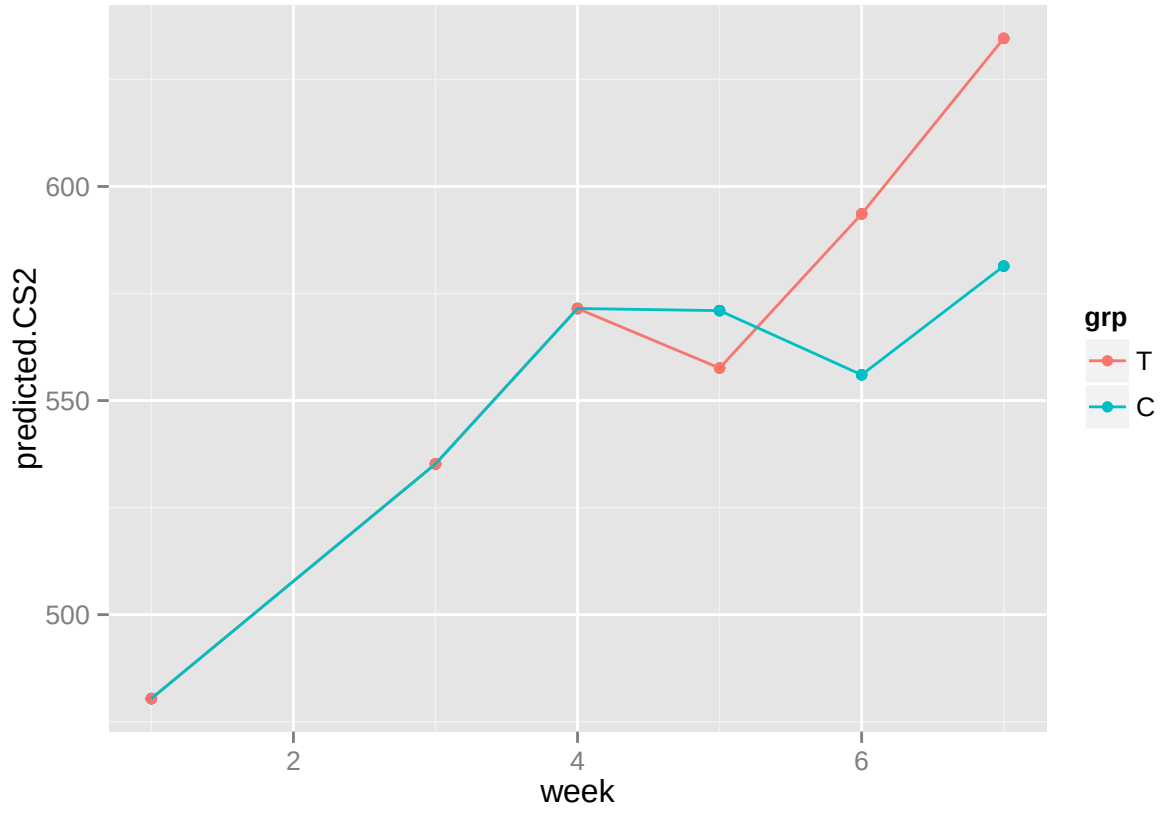



```
> # with individual trajectories
> gg.Gline + stat_summary(aes(group = grp, color = paste("mean", grp)),
+                         geom = "line", fun.y = mean, size = 3) +
+   scale_colour_manual(name = "Group",
+                       values = c("C" = "indianred1", "mean C" = "red",
+                                  "T" = "deepskyblue", "mean T" = "blue"))
```



Here the display the mean response of model 5(a) using the ggplot:

```
> ggplot(df.Pred_vitamin, aes(x = week, y = predicted.CS2, color = grp)) + geom_point() + geom_line()
```



Appendix B: Difference between type = "sequential" and type = "marginal" in the ANOVA function

Recall

Denote SS(A) the explained sum of squares by the variable A.

Sequential anova = Type 1 Anova:

- `SS(A) for factor A.`
- `SS(B | A) = SS(A, B) - SS(A) for factor B.`
- `SS(AB | B, A) = SS(A, B, AB) - SS(A, B) for interaction AB.`

It will give different results for unbalanced data depending on which main effect is considered first. It is testing the first factor without controlling for the other factor(s).

Marginal anova = Type II/III Anova (depending if we consider an interaction or not):

- `SS(A | B) for factor A if no interaction else SS(A | B, AB)`
- `SS(B | A) for factor B if no interaction else SS(B | A, AB)`

This type tests for each main effect after the other effects.

(from http://goanna.cs.rmit.edu.au/~fscholer/anova.php)

Anova function for nlme

```
> #### By default sequential ANOVA
>
> anova(gls.CS2)
```

```
Denom. DF: 51
            numDF  F-value p-value
week.factor     6 339.8439  <.0001
week5           1   2.7853  0.1013
week6           1   1.9579  0.1678
week7           1   7.9984  0.0067
```

```
> # equivalent anova(gls.CS2, type = "sequential")
>
> #### marginal ANOVA
> anova(gls.CS2, type = "marginal")
```

```
Denom. DF: 51
            numDF   F-value p-value
week.factor     6 302.40596  <.0001
week5           1   0.50937  0.4787
week6           1   3.99408  0.0510
week7           1   7.99838  0.0067
```

```
> ## equivalent to separate F tests
> n.coef <- length(coef(gls.CS2))
> Contrast <- matrix(0, nrow = 1, ncol = n.coef)
> colnames(Contrast) <- names(coef(gls.CS2))
> Contrast1 <- Contrast2 <- Contrast3 <- Contrast
>
> Contrast1[,"week5"] <- 1
> anova(gls.CS2, L = Contrast1)
```

```
Denom. DF: 51
 F-test for linear combination(s)
[1] 1
  numDF   F-value p-value
1     1 0.5093708  0.4787
```

```
> Contrast2[,"week6"] <- 1
> anova(gls.CS2, L = Contrast2)
```

```
Denom. DF: 51
 F-test for linear combination(s)
[1] 1
  numDF  F-value p-value
1     1 3.994082   0.051
```

```
> Contrast3[,"week7"] <- 1
> anova(gls.CS2, L = Contrast3)
```

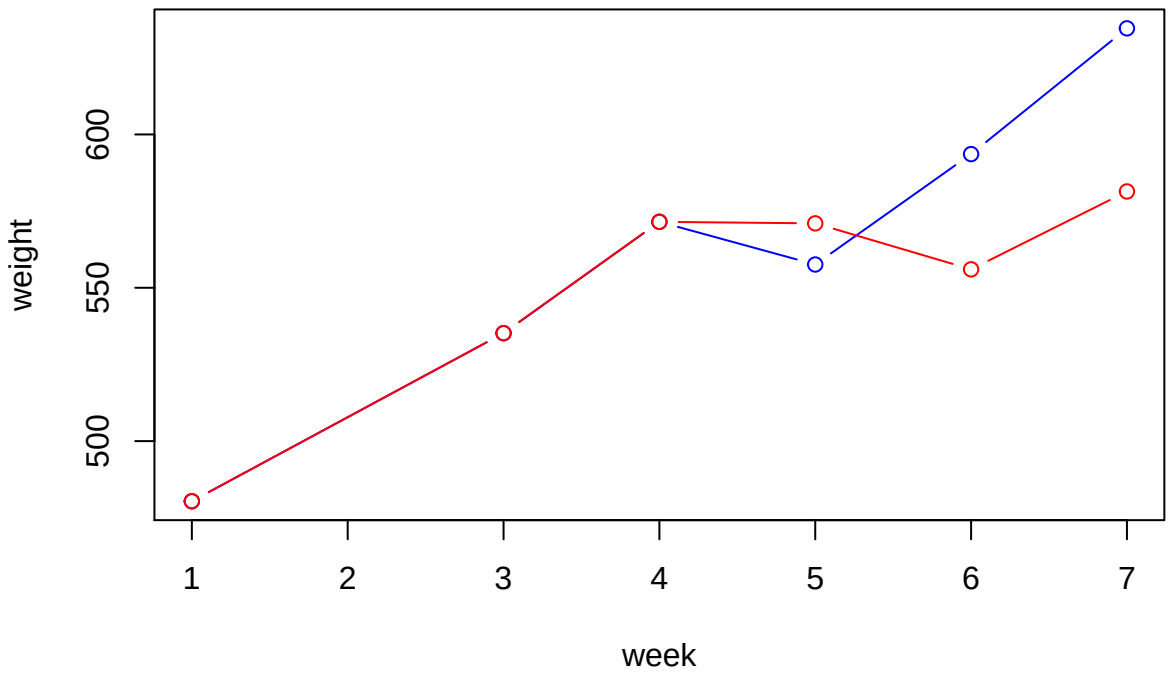
```
Denom. DF: 51
 F-test for linear combination(s)
[1] 1
  numDF  F-value p-value
1     1 7.998381  0.0067
```

```
> ## and very close to Wald tests
> summary(gls.CS2)$tTable[c("week5","week6","week7"),"p-value"]
```

```
      week5       week6       week7
0.478664916 0.051004637 0.006675368
```

Appendix C: Perform predictions using gls

```
> pred.week <- factor(c(1,3:7))
> pred.grpC <- factor(rep("C",6), levels = c("C","T"))
> pred.grpT <- factor(rep("T",6), levels = c("C","T"))
> pred.week5C <- pred.week6C <- pred.week7C <- rep(0,6)
> pred.week5T <- as.numeric(pred.week == 5)
> pred.week6T <- as.numeric(pred.week == 6)
> pred.week7T <- as.numeric(pred.week == 7)
>
> df.predC <- data.frame(week.factor = pred.week,
+                        grp = pred.grpC,
+                        week5 = pred.week5C,
+                        week6 = pred.week6C,
+                        week7 = pred.week7C)
>
> df.predT <- data.frame(week.factor = pred.week,
+                        grp = pred.grpT,
+                        week5 = pred.week5T,
+                        week6 = pred.week6T,
+                        week7 = pred.week7T)
>
> plot(x = c(1,3:7), y = predict(gls.CS2, newdata = df.predT),
+       type = "b", col = "blue", ylab = "weight", xlab = "week")
> points(x = c(1,3:7), y = predict(gls.CS2, newdata = df.predC),
+         type = "b", col = "red")
```


weight
500
550
600
1
2
3
4
5
6
7
week