TimeStamp: 17–03–2019 @ 08:28

Technical Note: Primer on Longitudinal Data Analysis via Linear Mixed-Effects Models

Stu Field, SomaLogic, Inc.

1 The Linear Model

For a simple linear-*fixed* effect model, the model equation fits the equation:

$$\begin{aligned} y_i &\sim \beta_0 + \beta_1 x_i + \epsilon_i \\ \epsilon &\sim N(0, \sigma_n^2), \end{aligned} \tag{1}$$

for the i^{th} sample/observation. Thus, we can estimate the model coefficients via,

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i \tag{2}$$

where,

$$\begin{aligned} \hat{\beta}_1 &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} \\ \hat{\beta}_0 &= \bar{y} - \hat{\beta}_1 \bar{x} \\ \epsilon_i &= r_i \\ &= y_i - \hat{y}_i \\ RSS &= \sum_{i=1}^n \epsilon_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \\ &= (y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_1)^2 + (y_2 - \hat{\beta}_0 - \hat{\beta}_1 x_2)^2 + \cdots + (y_n - \hat{\beta}_0 - \hat{\beta}_1 x_n)^2 \\ \text{and} & \\ R^2 &= (TSS - RSS)/TSS \\ TSS &= \sum (y_i - \bar{y})^2 \end{aligned}$$

The Total Sum of Squares (TSS) measures the total variance in the response (Y), thus R^2 is the proportion of the total variance in Y that can be explained by the model. The remaining variance is packed into ϵ. To calculate the accuracy of the coefficient estimates, we need a measure of variance in Y:

$$\begin{aligned}
\text{SE}(\hat{\beta}_0)^2 &= \sigma^2\left[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}\right] \\
\text{SE}(\hat{\beta}_1)^2 &= \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \\
\text{where,} & \\
\sigma^2 &= Var(\epsilon),
\end{aligned}$$

and σ^2 can be estimated from the data via the Residual Standard Error (RSE):

$$RSE = \sqrt{RSS/(n-2)}$$

2 Multiple-Linear Regression

The linear model can easily be expanded to include multiple predictors (regressors/variables/proteins). The extension of Eqn (1) can be modeled as:

$$\begin{aligned}
y_i &= \beta_0 x_{0i} + \beta_1 x_{1i} + \cdots + \beta_p x_{pi} + \epsilon_i \\
\epsilon_i &\sim \text{NID}(0, \sigma^2)
\end{aligned} \tag{3}$$

for the i^{th} sample and p^{th} covariate. We typically set $x_{0i} = 1$ so that β_0 is a constant intercept. In this form, the only *random-effect* is the error term, ϵ_i. This equation can be rewritten in matrix form as:

$$\begin{aligned}
\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} &= \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix} * \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix} \\
\vec{y} &= \boldsymbol{X}\vec{\beta} + \vec{\epsilon}
\end{aligned} \tag{4}$$

where $\vec{y}$ is a $n \times 1$ vector, $\boldsymbol{X}$ is the $n \times p$ data matrix, $\vec{\beta}$ is a $p \times 1$ vector of fixed effects coefficients for the p covariates, and $\vec{\epsilon}$ is the $n \times 1$ vector of observation specific errors.

3 The Mixed-Effect Model

Mixed-effects models fit both fixed effects *and* random effects in the same model. We assume a grouping or dependence structure (typically temporal or spatial) that is sampled from a larger population (e.g. plots within a farm, schools within districts, or observations within subjects). We also assume *compound symmetry*, which is a fancy way of saying that the related samples vary in the same fashion for each "group“ (subject), i.e. the off-diagonal of the variance-covariance matrix are all equal. For longitudinal data, samples from the same individual reflect serial dependence and the model must account for this non-independence in sample structure. Typically we are interested in fitting subject-specific (random) intercepts (i.e. parallel linear fits can hit the y-axis independently) and define the following:

$$\begin{aligned} y_{ij} &= b_{0i} + \beta_1 x_{ij} + \epsilon_{ij} \\ b_i &\sim N(0, \sigma_b^2) \\ \epsilon &\sim N(0, \sigma_n^2) \end{aligned}$$

for the j^{th} observation of the i^{th} subject, where b is a parameter treated as a random variable (i.e. i subjects sampled from a large, infinite population of potential subjects). These random effects are thus assumed to vary by group (in this case subjects) ...the goal is to capture this variation in the coefficients and prevent it from being packaged into ϵ!

4 Syntax in R

Use the `nlme` or `lme4` package. R uses the Wilkinson notation in specifying model formulas.

4.1 Random Intercept

```
lme(y ∼ time * group, random = ∼1 | pid, data = adat)
```

4.2 Random Slope

```
lme(y ∼ time * group, random = ∼time | pid, data = adat)
```

4.3 Random Intercept & Random Slope

```
lme(y ∼ time * group, random = ∼1 + time | pid, data = adat)
```

4.4 Syntax Definitions

- `y` $= y_{ij}$
- $\sim 1 =$ random effects (b_{0i})
- `pid` = field containing the dependent groups (subjects)
- `time * group` = the fixed effects with interaction term
- **note:** `time * group` = `time + group + time*group`

5 Example 1: Response to Drug

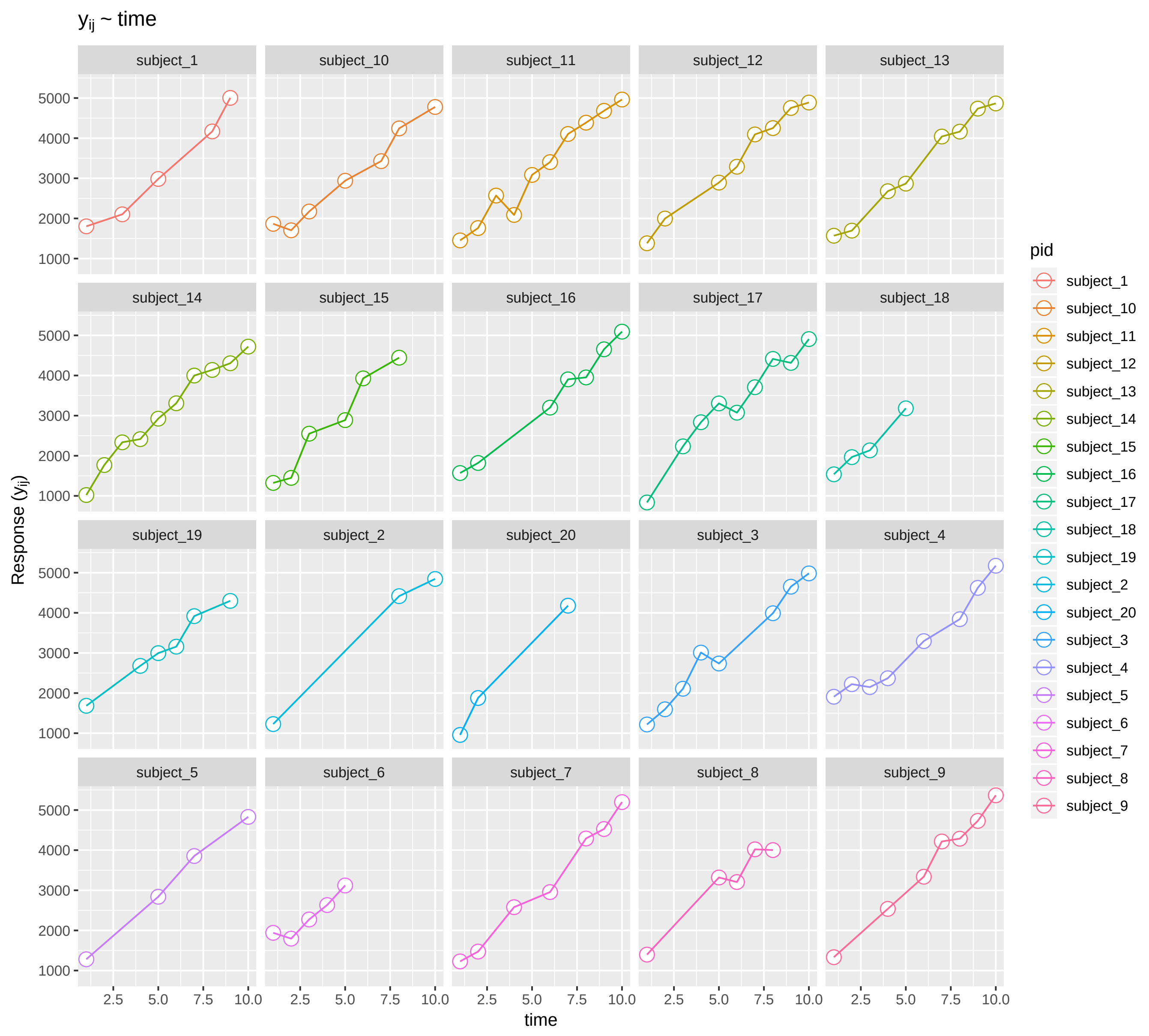
First, simulate longitudinal data for 20 subjects, having $3-10$ serial samples each, intercept ($\beta_0 = 1000$), slope ($\beta_1 = 400$), and a serial autocorrelation parameter of 0.1. The default simulation values can be seen below:

```
> args(simulateLongData)
```

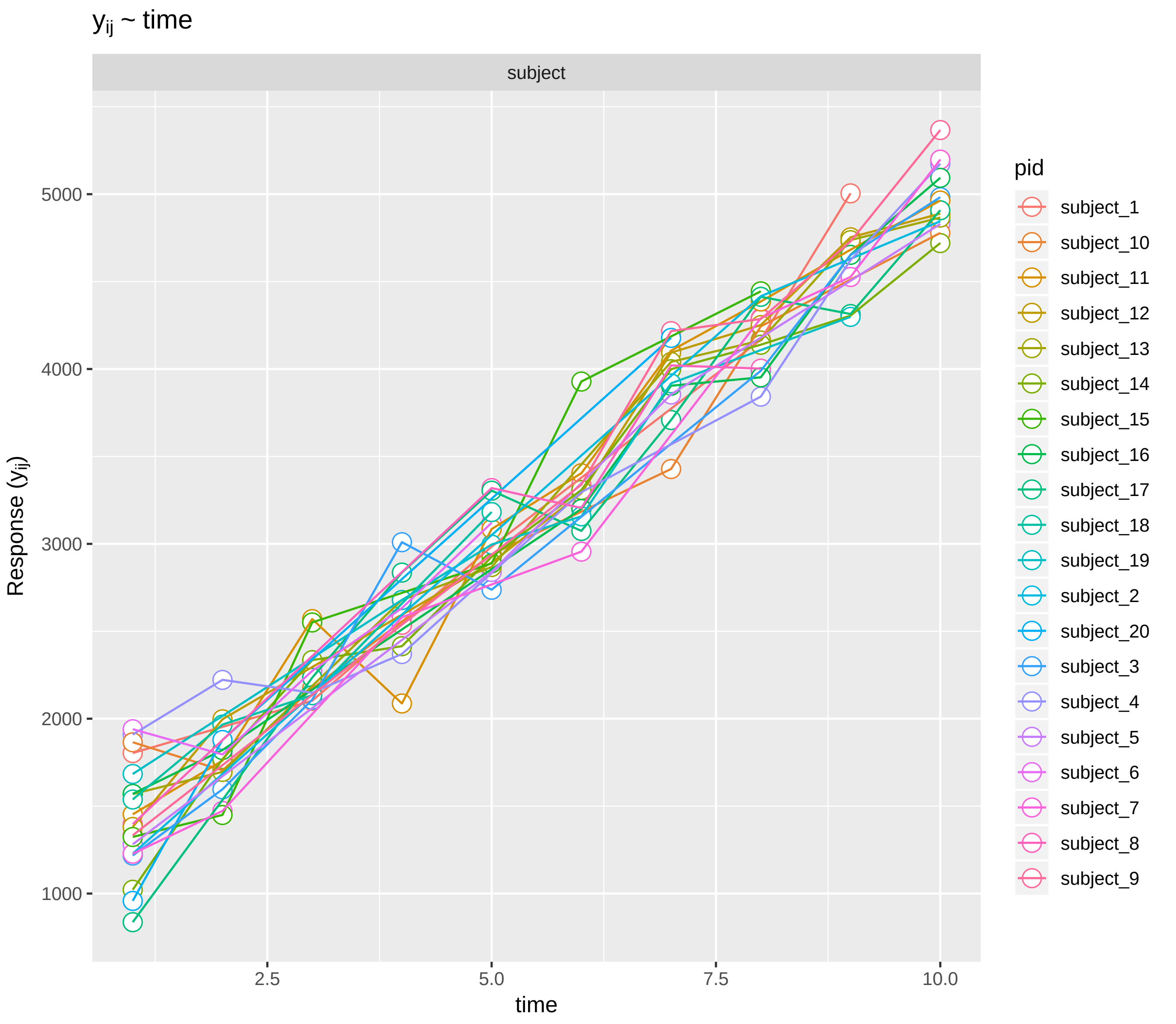
```
function (nsubj = 20, beta0 = 1000, beta1 = 10, max.obs = 10,
    group = NULL, auto.cor = 0.5, sd.pars = list(sigma = beta0/4,
        tau0 = 2, tau1 = 2, tau01 = 0.5), r.seed = sample(1000,
        1))
NULL
```

```
> # simulate longitudinal data
> # createLongData() is a wrapper for simulateLongData()
> p <- list(nsubj = 20, beta0 = 1000, beta1 = 400, max.obs = 10, r.seed = 101,
+           sd.pars = list(sigma = 250), auto.cor = 0.1)
>
> ResponseData <- createLongData(subject=p)
> head(ResponseData)

# A tibble: 6 x 5
  pid        time    eij   yij Group
  <chr>     <dbl>  <dbl> <dbl> <chr>
1 subject_1     1  402.  1804. subject
2 subject_1     3 -101   2103. subject
3 subject_1     5  -21.7 2984. subject
4 subject_1     8  -41.2 4168. subject
5 subject_1     9  394.  5005. subject
6 subject_2     1 -169.  1228. subject

> table(ResponseData$pid)

 subject_1 subject_10 subject_11 subject_12 subject_13 subject_14 subject_15
         5          7         10          8          8         10          6
subject_16 subject_17 subject_18 subject_19  subject_2 subject_20  subject_3
         7          9          4          6          3          3          8
 subject_4  subject_5  subject_6  subject_7  subject_8  subject_9
         8          4          5          7          5          7
```

```
> # plot longitudinal traces by subject ("pid")
> ResponseData %>%
+   ggplot(aes(x = time, y = yij, group = pid, colour = pid)) +
+   geom_point( size = 4, shape = 21, fill = "white") +
+   geom_line() + ylab(y_lab) +
+   guides(colour=guide_legend(ncol = 1)) +
+   facet_wrap( "pid" ) + ggtitle(bquote(y[ij]~"~"~time))
```



```
> # plot longitudinal traces together
> ResponseData %>%
+   ggplot(aes(x = time, y = yij, group = pid, colour = pid)) +
+   geom_point( size=4, shape = 21, fill = "white") +
+   geom_line() + ylab(y_lab) +
+   guides(colour=guide_legend(ncol = 1)) +
+   facet_wrap(vars(Group)) + ggtitle(bquote(y[ij]~"~"~time))
```



5.1 Analysis in R

The model we wish to fit:

$$y_{ij} = b_{0i} + \beta_1 time_{ij} + \epsilon_{ij}, \tag{5}$$

again for the j^{th} observation of the i^{th} subject, where b_{0i} is the subject-specific random *intercept*, and β_1 is the fixed-effect for *time*.

```
> fit1 <- fit_lme_safely(yij ~ time, random=~1 | pid, data = ResponseData)
> summary(fit1)

Linear mixed-effects model fit by REML
 Data: ResponseData
     AIC    BIC  logLik
  1773.7 1785.1 -882.84
```

```
Random effects:
 Formula: ~1 | pid
        (Intercept) Residual
StdDev:    0.020204   228.51

Fixed effects: yij ~ time
              Value Std.Error  DF t-value p-value
(Intercept) 1017.39    40.737 109  24.974       0
time         398.36     6.644 109  59.962       0
 Correlation:
     (Intr)
time -0.871

Standardized Within-Group Residuals:
      Min        Q1       Med        Q3       Max
-2.535320 -0.588746 -0.095637  0.665085  2.295041

Number of Observations: 130
Number of Groups: 20
```

From the fixed-effects column in `Value`, the estimates are reasonably close to the original parameters: $\hat{\beta_0} = 1017.39$ and $\hat{\beta_1} = 398.36$. Both the slope and intercepts differ significantly from zero.

6 Example 2: Group Dependent Response (Interaction)

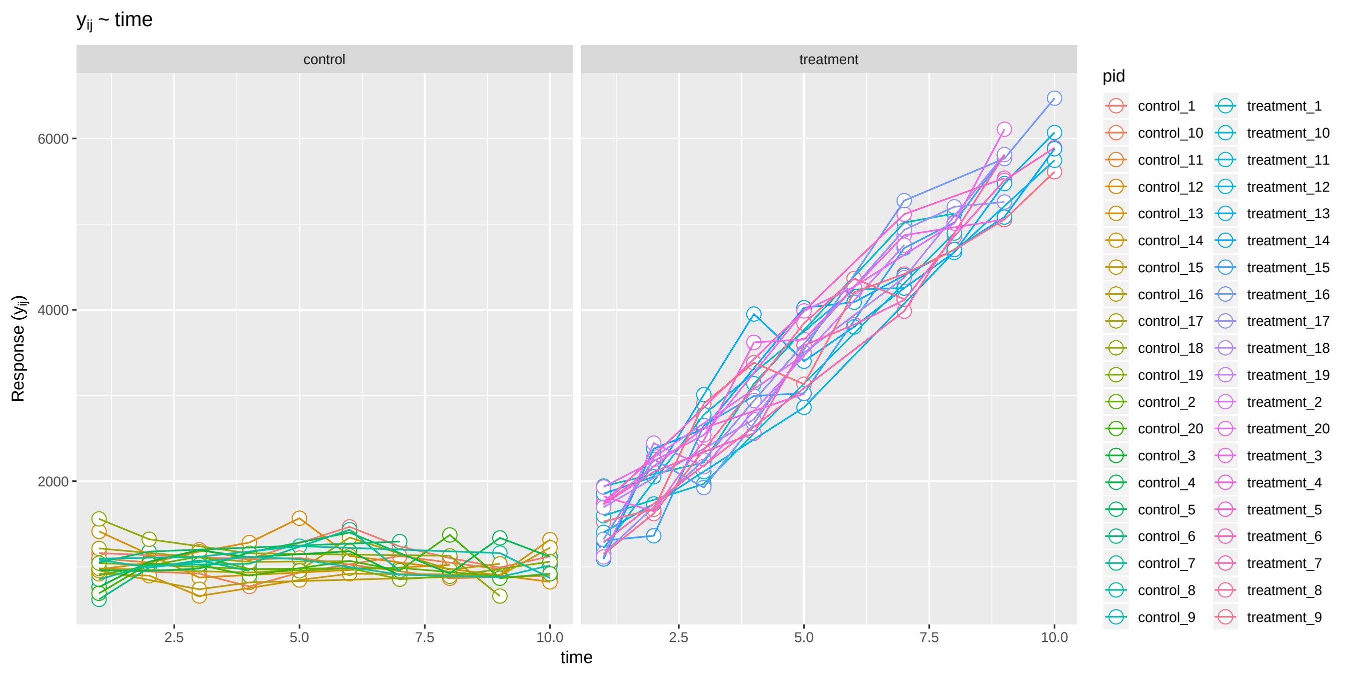
In clinical studies, it is often of interest to understand how the treatment of one group of subjects differs from another group of subjects (e.g. a control group). In a mixed-model setting, this is accomplished by adding an interaction term to the model, which sets up a conditional response variable, given that a subject belongs to a particular group/class.

Next, simulate longitudinal data for 20 subjects, having $3 - 10$ serial samples each, intercepts of $\beta_0 = 1000$, slopes of $\beta_1 = 0$ (crtl) and $\beta_1 = 500$ (treat), and a serial autocorrelation parameter of 0.1. The parameters that differ from the default simulation values can be seen below:

```
> # simulate longitudinal data with group-specific response
> control <- list(nsubj = 20, beta1 = 0, max.obs = 10, r.seed = 10,
+                 sd.pars = list(sigma = 150), auto.cor=0.1)
> treatment <- list(nsubj = 20, beta1 = 500, max.obs = 10, r.seed = 11,
+                   sd.pars = list(sigma = 350), auto.cor = 0.1)
> GroupResponseData <- createLongData(control, treatment)
```

Now plot the longitudinal time traces:

```
> GroupResponseData %>%
+   ggplot(aes(x = time, y = yij, group = pid, colour = pid)) +
+   geom_point(size = 4, shape = 21, fill = "white") +
+   geom_line() + ylab(y_lab) +
+   guides(colour = guide_legend(ncol = 2)) +
+   facet_wrap(vars(Group)) + ggtitle(bquote(y[ij]~"~"~time))
```



6.1 Analysis in R

The model specification to fit:

$$y_{ij} = b_{0i} + \beta_1 time_{ij} + \beta_2 group_i + \beta_3(group_i \times time_{ij}) + \epsilon_{ij}, \tag{6}$$

again for the j^{th} observation of the i^{th} subject, b_{0i} is still the subject-specific random intercept and β_1 is still the fixed-effect for *time*. However, now there is a fixed-effect for *group* and an interaction term (β_3 term) that is multiplied by a dummy variable such that:

$$G = \begin{cases} 0, & control \\ 1, & treatment \end{cases}$$

giving,

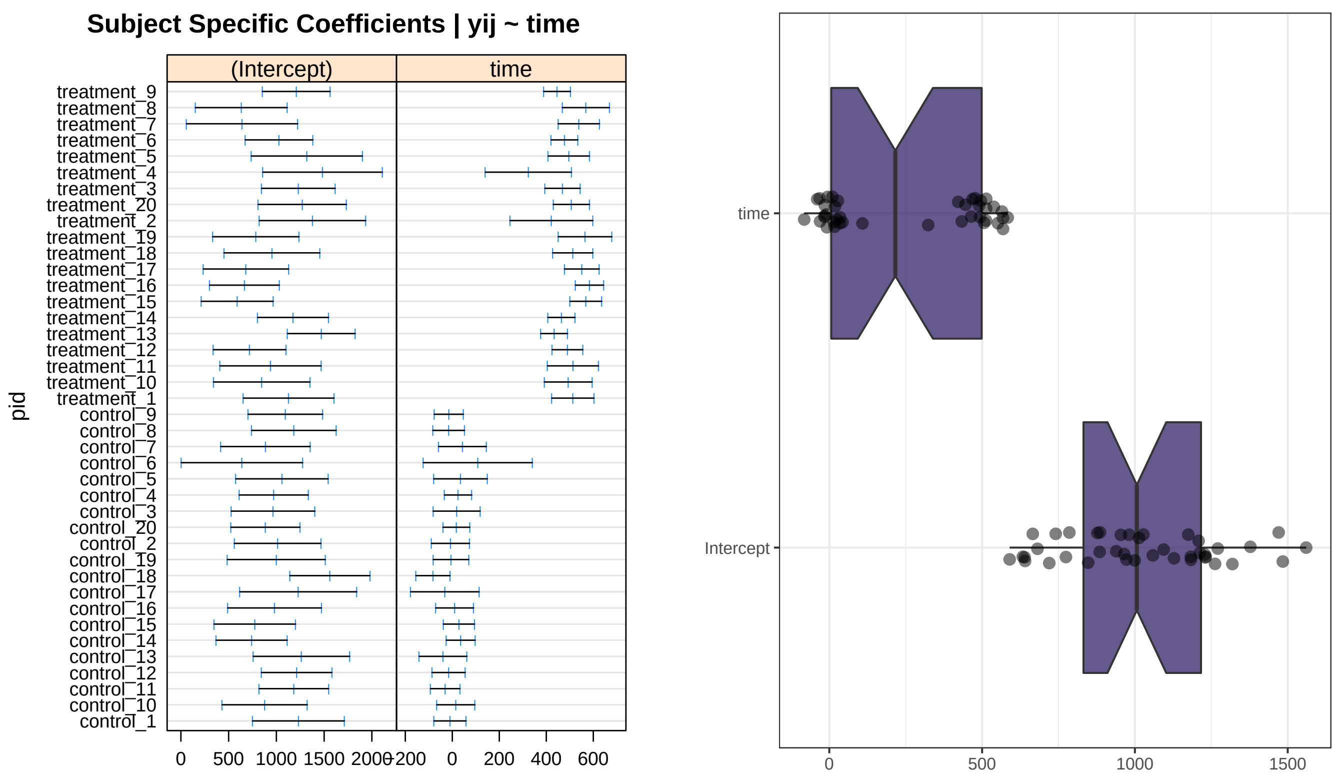
$$y_{ij} = b_{0i} + \beta_1 time_{ij} + \beta_2 group_i + \beta_3(group_i \times time_{ij}) * G + \epsilon_{ij}, \tag{7}$$

```
> fit2 <- fit_lme_safely(yij ~ time * Group, random = ~1 | pid, data = GroupResponseData)
```

6.2 Check Subject-specific linear models

It is often a good idea to check the variation on the coefficients of subject-specific linear fits of the data. The `SomaMixedEffects::lmeDiagnostic()` function creates a list of linear models for each subject (i).

```
> # summarize the estimates and significance for each subject_i
> lmeDiagnostic(fit2)
```



Notice that the subject-subject offsets (β_0) are approximately centered about 1000 (RFU) for both treatment and control (as they should be; we set them to the same intercept), and a vertical line up from 1000 would cross the confidence interval for most subjects. The more variable the estimates (and intervals) of β_0 seen in the left panel, the greater the improvement in model fit can be found in fitting subject-specific offsets in a mixed-effects model framework. Secondly, as can be expected, the slope coefficient (β_1) is vastly different between treated and control. Depending on the statistical question, subject-specific slopes may be desired (but typically not for longitudinal time-series data in a clinical setting).

6.3 Summary

```
> summary(fit2)

Linear mixed-effects model fit by REML
 Data: GroupResponseData
     AIC    BIC  logLik
  3238.2 3258.8 -1613.1

Random effects:
 Formula: ~1 | pid
        (Intercept) Residual
StdDev:      45.663   268.33

Fixed effects: yij ~ time * Group
                      Value Std.Error  DF t-value p-value
(Intercept)          1039.71    49.672 190  20.931  0.0000
```

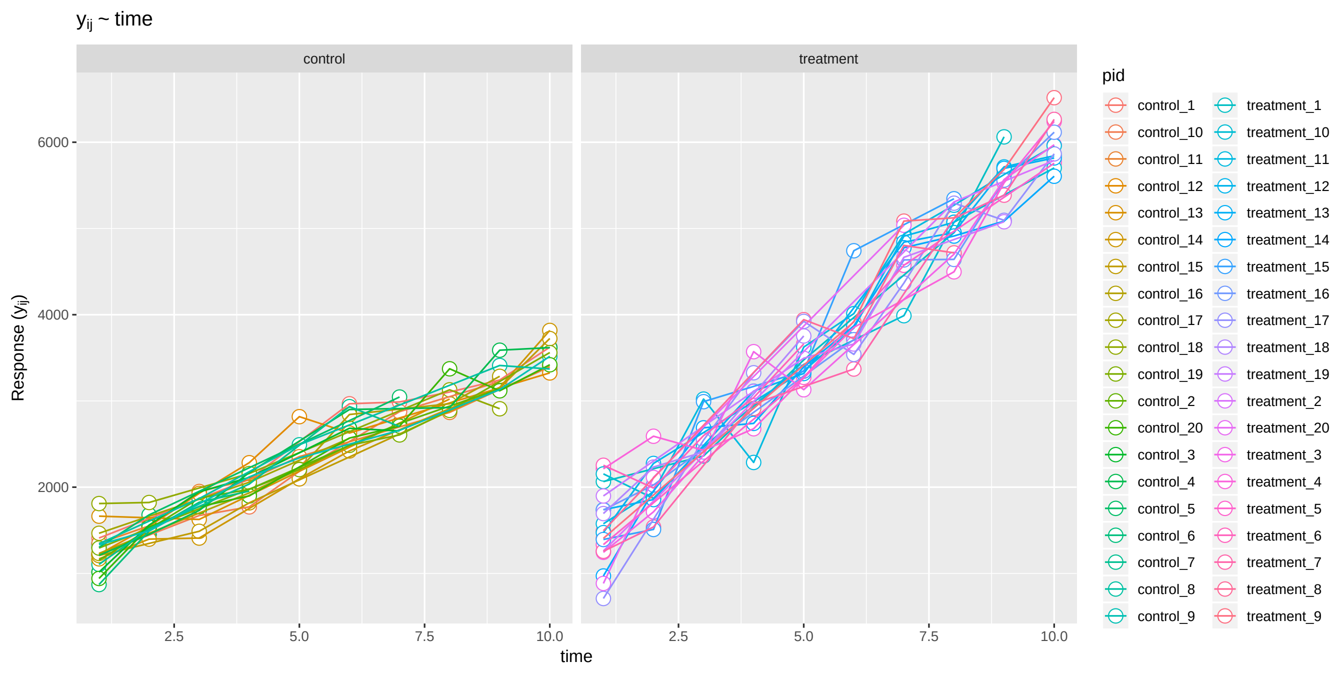
```
time                       0.03     8.409 190   0.003  0.9974
Grouptreatment           -35.92    70.551  38  -0.509  0.6136
time:Grouptreatment      499.75    12.013 190  41.601  0.0000
 Correlation:
                    (Intr) time   Grptrt
time                -0.837
Grouptreatment      -0.704  0.589
time:Grouptreatment  0.586 -0.700 -0.839

Standardized Within-Group Residuals:
      Min        Q1       Med        Q3       Max
-2.329758 -0.579888 -0.054073  0.605159  3.447723

Number of Observations: 232
Number of Groups: 40
```

From the fixed-effects column in `Value`, the estimates are reasonably close to the original parameters: $\hat{\beta}_0 = 1039.71$ and $\hat{\beta}_1 = 0.03$ (essentially zero). This indicates that the effect of time has no effect on y_{ij} (at least for the controls!). The significant **interaction** tells a different story and indicates that the **entire** fixed-effect for *time* occurs in the *treatment* group ($\hat{\beta}_3 = 499.75$); in this case the *control* group does not vary with time.

6.4 Example 3

To illustrate how the model output is constructed, we simulate identically to the above, however give the control group a slope of $\beta_1 = 250$. This results in a significant fixed-effect for time *and* an significant interaction; the additional effect of being in the *treatment* group. Relate the values back to the original Eqn (7) to see how the terms are partitioned.

```
> # simulate longitudinal data with group-specific response
> p1 <- list(nsubj = 20, beta1 = 250, max.obs = 10, r.seed = 10,
+            sd.pars = list(sigma = 150), auto.cor = 0.1)
> p2 <- list(nsubj = 20, beta1 = 500, max.obs = 10, r.seed = 101,
+            sd.pars=list(sigma = 350), auto.cor = 0.1)
> GroupResponseData2 <- createLongData(control = p1, treatment = p2)
>
> GroupResponseData2 %>%
+   ggplot(aes(x = time, y = yij, group = pid, colour = pid)) +
+   geom_point( size = 4, shape = 21, fill = "white") +
+   geom_line() + ylab(y_lab) +
+   guides(colour = guide_legend(ncol = 2)) +
+   facet_wrap(vars(Group)) + ggtitle(bquote(y[ij]~"~"~time))
```



```
> fit3 <- fit_lme_safely(yij ~ time * Group, random = ~1 | pid, data = GroupResponseData2)
> summary(fit3)

Linear mixed-effects model fit by REML
 Data: GroupResponseData2
     AIC    BIC  logLik
  3407.7 3428.6 -1697.8

Random effects:
 Formula: ~1 | pid
        (Intercept) Residual
StdDev:    0.044928   264.27

Fixed effects: yij ~ time * Group
                      Value Std.Error  DF t-value p-value
(Intercept)         1039.15    47.741 203 21.7664  0.0000
time                 250.07     8.235 203 30.3663  0.0000
Grouptreatment       -14.72    67.073  38 -0.2194  0.8275
time:Grouptreatment  247.49    11.263 203 21.9740  0.0000
 Correlation:
                    (Intr) time   Grptrt
time                -0.856
Grouptreatment      -0.712  0.610
time:Grouptreatment  0.626 -0.731 -0.863

Standardized Within-Group Residuals:
     Min        Q1       Med        Q3       Max
-3.07478 -0.55432 -0.10400  0.52539  2.77042
```

```
Number of Observations: 245
Number of Groups: 40
```

Notice that the $\hat{\beta}_0 = 1039.15$ has not changed, as we haven't changed any baseline values, but now half of the effect has moved into the *time* effect ($\hat{\beta}_1 = 250.07$), and away from the interaction coefficient ($\hat{\beta}_3 = 247.49$) which was carrying the entire slope effect for the treatment group. Stated alternatively, the significance of the interaction has dropped, though still a significant $p-$value, shifting from 41.6 to 21.97.

7 Notes

7.1 Mixed-effects Analysis Guidelines

Below are some initial guidelines for the implementation of linear mixed-effects modeling in R.

1. Make sure you are fitting the correct model for your desired question.
2. Be sure you can write the full equation for the model(s) you are fitting.
3. Prior to model fitting, plot the subject-specific coefficients with `nlme::lmList`.
4. The `lme` function uses Expectation Maximization combined with Newton-Raphson iterations to fit the various model coefficients.
5. A lme diagnostic wrapper can found in `SomaMixedEffects::lmeDiagnostic()`.
6. A wrapper for `nlme::lme` to avoid convergence failures to stop your code from running to completion can be found in `SomaMixedEffects::fit_lme_safely()`.
7. The functions below can be found in `SomaMixedEffects::fitMixedEffectsModels()` and `SomaMixedEffects::`

7.2 Sample Code For ADAT Analysis

Univariate fitting of all analytes in an ADAT:

```
> # see SomaMixedEffects/R/fitMixedEffectsModels.R
> fitMixedEffectsModels

function(data, fixed = "TimePoint*SampleGroup",
                                  random = "~ 1 | pid", do.log) {

  if ( !missing(do.log) ) {
    rlang::signal("`do.log` is deprecated. Log-transform upstream if desired.",
                  "error")
  }

  #head(getAptamers(data), 25) %>%
  data %>%
    SomaPlyr::getAptamers() %>%
    purrr::set_names() %>%
      purrr::map(function(apt) {
          message(stringr::str_glue("Fitting ... {apt}"))
          as.formula(stringr::str_glue("{apt} ~ {fixed}")) %>%
            fit_lme_safely(formula = .,
                           random  = as.formula(random),
                           data    = data)
      }) %>%
      structure(class    = c("soma_lme", "list"),
                fixed    = fixed,
                random   = random,
                PIDfield = stringr::str_remove(random, "^~.*\\| *"),
                log      = SomaGlobals::is.logspace(data),
                data.dim = dim(data)) %>%
      invisible()
}
<bytecode: 0x7fb656146d88>
<environment: namespace:SomaMixedEffects>
```

Combine the fitted models above into a statistical table:

```
> # see SomaMixedEffects/R/createMixedEffectsTable.R
> createMixedEffectsTable

function(x) {

  stopifnot(inherits(x, "soma_lme"), # our own object `fitMixedEffectsModels()`
            !is.null(names(x)))      # must be a named list

  if ( !"fixed" %in% names(attributes(x)) ) {
    stop(
      stringr::str_glue(
        "Fixed effects string missing. \\
         Was the model list created using `fitMixedEffectsModels()`?"
        ),
      call. = FALSE)
  }

  stab <- x %>%
    purrr::map_df(~ {
      aov_tab <- stats::anova(.x) %>%
        data.frame() %>%
        dplyr::select(-dplyr::ends_with("DF")) %>%
        tibble::as_tibble(rownames = "Parameter") %>%
        dplyr::filter(row_number() != 1)           # 1st row is `Intercept`
      nms <- list(a = aov_tab$Parameter,
                  b = names(aov_tab)[-1]) %>%      # don't want `Parameter` name
        purrr::cross_df() %>%
        purrr::pmap_chr(paste, sep = "_")
      aov_tab %>%
        dplyr::select(-Parameter) %>%             # remove par col
        purrr::flatten_dbl() %>%                  # vectorize
        purrr::set_names(nms) %>%                 # name vector
        as.list() %>% data.frame() %>%            # recast to 1 row df
        dplyr::select(starts_with("Response_"),
                      starts_with("TimePoint_"),
                      everything()) %>%           # reorder to group F/p-vals
        dplyr::mutate(converged = .x$converged)   # add converged bool
    }, .id = "AptName")

  p_col <- attributes(x)$fixed %>%           # get fixed effects
    stringr::str_replace("\\*", ".") %>%     # replace '*' with '.'
    paste0("_p.value$") %>%                  # paste to end with p-value
    grep(names(stab), value = TRUE) %>%      # search `stab` for a match
    dplyr::sym()                             # convert to symbol for !! below

  stab <- stab %>%
    dplyr::mutate(
      fdr          = stats::p.adjust(!!p_col, method = "fdr"),
      p.bonferroni = stats::p.adjust(!!p_col, method = "bonferroni")
      ) %>%
    dplyr::arrange(!!p_col) %>%
    dplyr::mutate(rank = dplyr::row_number()) %>%
    tibble::column_to_rownames("AptName")

  ret.list <- list()
  ret.list$stat.table   <- stab
  ret.list$test         <- "Linear Mixed-effects Model"
  ret.list$call         <- x[[1]]$call
  ret.list$method       <- x[[1]]$method
  ret.list$fixed        <- attributes(x)$fixed
  ret.list$random       <- attributes(x)$random
  ret.list$log          <- attributes(x)$log
  ret.list$nModels      <- length(x)
  ret.list$data.dim     <- attributes(x)$data.dim
  dims                  <- x[[1]]$dims
  ret.list$observations <- dims$N
  pids                  <- dims$ngrps[dims$Q[1]]
  ret.list$PIDfield     <- attributes(x)$PIDfield
  ret.list$subjects     <- unname(pids)
  ret.list %>%
    addClass(c("stat.table", "mixed.effects.table"))
}
<bytecode: 0x7fb656473370>
<environment: namespace:SomaMixedEffects>
```

7.3 Pitfalls

- Mike Hinterberg: effects are fit sequentially according to the magnitude of the effect.
-