Fitting Multivariate Longitduinal Models in nlme

Peter Norwood, NCSU, Janssen R&D

1 Introduction

This writeup discusses a how to fit linear multivariate longitudinal models in the `nlme` package in R and how to extract relevant output from these models.

We will only discuss fitting joint models, but the ideas should easily generalize to more than two responses. Additionally, while there are numerous mean and covariance structures that we could possibly consider, we will only focus on a few. The hope is that with the understanding these basics will provide the tools to build larger and more complex models.

2 Data Preparation

To fit these models we need the following columns in a dataset:

- `ID`: subject ID
- `y`: column will all responses
- `y_type`: categorical column with labels detailing which response the row contains
- `time`: time variable
- `y_type_time`: categorical column with labels detailing which response-time pair the row contains
- `A,X,...`: other columns with model features

and example is below

ID	y	y_type	time	y_type_time	trt	biomarker
1	10.5	y1	0	y1_1	0	3.5
1	12.3	y1	2	y1_2	0	3.5
1	4.2	y2	0	y2_1	0	3.5
1	3.6	y2	2	y2_2	0	3.5
2	7.3	y1	0	y1_1	1	-1.2
2	7.5	y1	2	y1_2	1	-1.2
2	6.1	y2	0	y2_1	1	-1.2
2	11.2	y2	2	y2_2	1	-1.2

3 Fitting Models

3.1 Multivariate Random Slope, Random Intercept Model, Continuous Time

The first model we fit involves two responses with multiple time points each:

$$y_{i1j} = \beta_{10} + b_{i10} + \beta_{11} * A_i + \beta_{12} * t_{ij} + b_{i11} * t_{ij} + \beta_{13} * A_i * t_{ij} + \epsilon_{i1j}$$

and

$$y_{i2j} = \beta_{20} + b_{i20} + \beta_{21} * A_i + \beta_{22} * t_{ij} + b_{i21} * t_{ij} + \beta_{23} * A_i * t_{ij} + \epsilon_{i2j}$$

.

Here y_{i1j} is the response type 1 for individual i at time point j. y_{i2j} is the response type 2 for individual i at time point j. $A_i \in \{0, 1\}$ is the treatment indicator for individual i and t_{ij} is the time variable (hour, week, month, etc) at time point j for individual i. b_{i10}, b_{i20} are the random intercepts for y_{1ij} and y_{2ij} respectively. Likewise, b_{i11}, b_{i21} are the random slopes. We take:

$$\begin{pmatrix} b_{i10} \\ b_{i11} \\ b_{i20} \\ b_{i21} \end{pmatrix} \sim MVN\left(\mathbf{0}, \mathbf{G}\right)$$

Since we have relatively few parameters, it is typical to leave the $\mathbf{G}$ matrix as an unstructured, symmetric, positive definite matrix:

$$\mathbf{G} = \begin{pmatrix} \sigma^2_{b_{10}} & \sigma_{b_{10},b_{11}} & \sigma_{b_{10},b_{20}} & \sigma_{b_{10},b_{21}} \\ . & \sigma^2_{b_{11}} & \sigma_{b_{11},b_{20}} & \sigma_{b_{11},b_{21}} \\ . & . & \sigma^2_{b_{20}} & \sigma_{b_{20},b_{21}} \\ . & . & . & \sigma^2_{b_{21}} \end{pmatrix}$$

Outside of the mean specification, the main changes one can make with multivariate longitudinal models is how to structure the covariance matrix for the residual error. We will detail three different methods and use a bivariate model with four time points each for illustration.

First we consider the case where we have different error variances based on `y_type` but all errors are uncorrelated. So

$$\begin{pmatrix} \epsilon_{i1j} \\ \epsilon_{i2j} \end{pmatrix} \sim MVN\Bigg(\mathbf{0}, \mathbf{R}\Bigg)$$

where

$$\mathbf{R} = \begin{pmatrix} \sigma^2_{\epsilon_1} & 0 \\ . & \sigma^2_{\epsilon_1} \end{pmatrix}$$

and ultimately

$$\begin{pmatrix} \epsilon_{i11} \\ \epsilon_{i21} \\ \epsilon_{i12} \\ \epsilon_{i22} \\ \epsilon_{i13} \\ \epsilon_{i23} \\ \epsilon_{i14} \\ \epsilon_{i24} \end{pmatrix} \sim MVN\left(\mathbf{0}, \begin{pmatrix} \mathbf{R} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} \\ . & \mathbf{R} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} \\ . & . & \mathbf{R} & \mathbf{0}_{2\times 2} \\ . & . & . & \mathbf{R} \end{pmatrix}\right)$$

To fit this model we use the following command

```
              ## remove common intercept w/ -1
              ## add response specific intercept with y_type
              ## add other fixed effects by each y_type
fit_lme <- lme(y~-1+y_type+y_type:(trt+time+trt:time),
              ## random slope and intercept for each y_type
              random=~-1+(y_type+y_type:time)|ID,
              ## different error variances for different y_type
              weights=varIdent(form=~1|y_type),
              data=dat)
```

If we instead want to only estimate correlation between each residual error term at a time point, so $Cor(\epsilon_{i1j}, \epsilon_{i2j}) = \rho \geq 0$ and keep this correlation the same for all time points, we can update the function slightly. Our $\mathbf{R}$ matrix will now turn into

$$\mathbf{R} = \begin{pmatrix} \sigma^2_{\epsilon_1} & \sigma_{\epsilon_1,\epsilon_2} \\ . & \sigma^2_{\epsilon_1} \end{pmatrix} = \begin{pmatrix} \sigma_{\epsilon_1} & 0 \\ 0 & \sigma_{\epsilon_2} \end{pmatrix} \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \begin{pmatrix} \sigma_{\epsilon_1} & 0 \\ 0 & \sigma_{\epsilon_2} \end{pmatrix}$$

and all the errors will still have the block diagonal covariance matrix as seen before. To fit this model we add in the `correlation` option to the `lme` function.

```
              ## remove common intercept w/ -1
              ## add response specific intercept with y_type
              ## add other fixed effects by each y_type
fit_lme <- lme(y~-1+y_type+y_type:(trt+time+trt:time),
              ## random slope and intercept for each y_type
              random=~-1+(y_type+y_type:time)|ID,
              ## different error variances for different y_type
              weights=varIdent(form=~1|y_type),
              ## add cor between resid error at each time point
              correlation=corSymm(form=~1|ID/time),
              data=dat)
```

If we care to estimate correlation between all pairs of $\epsilon_{pij}, \epsilon_{qik}$, so that

$$\begin{pmatrix} \epsilon_{i11} \\ \epsilon_{i21} \\ \epsilon_{i12} \\ \epsilon_{i22} \\ \epsilon_{i13} \\ \epsilon_{i23} \\ \epsilon_{i14} \\ \epsilon_{i24} \end{pmatrix} \sim MVN\left(\mathbf{0}, \Sigma\right)$$

where Σ is a unstructured, symmetric, positive definite matrix then we can use the following code.

```
               ## remove common intercept w/ -1
               ## add response specific intercept with y_type
               ## add other fixed effects by each y_type
fit_lme <- lme(y~-1+y_type+y_type:(trt+time+trt:time),
               ## random slope and intercept for each y_type
               random=~-1+(y_type+y_type:time)|ID,
               ## different error variances for different y_type
               weights=varIdent(form=~1|y_type),
               ## add cor between ALL resid error
               correlation=corSymm(form=~1|ID),
               data=dat)
```

It is worth noting that this model is quite difficult to fit computationally since we need to estimate two sources of variation and many parameters for both of those sources. Often times, we are ultimately interested in the overall correlation between y_{pij}, y_{qik}, so we are interested in

$$\mathbf{V}_i = \mathbf{Z}_i \mathbf{G} \mathbf{Z}_i^T + \Sigma$$

and this $\mathbf{V}_i$ can still have a flexible form without Σ being totally unstructured. Using different classes like `corCompSymm` allows for different structures for Σ and these classes can be found in the `nlme` documentation.

3.1.1 Extracting Model Output

Extracting model output can sometimes be a bit of a pain, and we present multiple ways to obtain certain information. These results should generalize to each multivariate mixed model. Below is the code walking through the process.

```
> ## fit model
```

```
> fit_lme <- lme(y~-1+y_type+y_type:(trt+time+trt:time),
+                 ## random slope and intercept for each y_type
+                 random=~-1+(y_type+y_type:time)|id,
+                 ## different error variances for different y_type
+                 weights=varIdent(form=~1|y_type),
+                 ## correlation between resid error at each time point
+                 correlation=corSymm(form=~1|id/time),
+                 control=cntrl,
+                 data=dat)
>
> ## extract lme information
>
> ## summary of fit
> sum_lme <- summary(fit_lme)
>
> ## fixed effects
> sum_lme$tTable
                        Value  Std.Error  DF     t-value       p-value
y_typey1           6.77170292 0.18523506 693  36.5573509 7.586141e-164
y_typey2           1.01193627 0.21944875 693   4.6112648  4.766257e-06
y_typey1:trt       0.06777217 0.26196193 693   0.2587100  7.959358e-01
y_typey2:trt       0.15645427 0.31034739 693   0.5041263  6.143329e-01
y_typey1:time      6.90608540 0.04086782 693 168.9859188  0.000000e+00
y_typey2:time      1.22687762 0.02026826 693  60.5319722 7.085281e-279
y_typey1:trt:time -0.09466689 0.05779582 693  -1.6379539  1.018853e-01
y_typey2:trt:time -0.07678394 0.02866365 693  -2.6787919  7.564075e-03
>
> ## random effects covariance matrix
> G_star <- getVarCov(fit_lme,type="random.effects")
> G_star
Random effects variance covariance matrix
               y_typey1  y_typey2 y_typey1:time y_typey2:time
y_typey1       0.860340  0.752340     0.0312630    -0.0095820
y_typey2       0.752340  2.037700     0.0398860    -0.0243450
y_typey1:time  0.031263  0.039886     0.0427820     0.0097842
y_typey2:time -0.009582 -0.024345     0.0097842     0.0029116
  Standard Deviations: 0.92754 1.4275 0.20684 0.053959
>
> ## rearrange to fit with our Z matrix (cols go b10, b11, b20, b11)
> ## we could also just re-write our G matrix in a certain way
> G_lme <- matrix(c(G_star[1,1],G_star[1,3],G_star[1,2],G_star[1,4],
+                    G_star[3,1],G_star[3,3],G_star[3,2],G_star[3,4],
```

```
+                    G_star[2,1],G_star[2,3],G_star[2,2],G_star[2,4],
+                    G_star[4,1],G_star[4,3],G_star[4,2],G_star[4,4]),4,4,byrow=TRUE)
>
>
> ## getting y1 and y2 residual variances
> ## it is woth looking at the model summary first to make sure you
> ## have the y1 and y2 weights in the right place here
> ## since we sill do matrix
> sigmas <- (1/unique(attributes(fit_lme$modelStruct$varStruct)$weights)*fit_lme$sigma)
>
> ## grabbing the correlation matrix between residual error
> ## it is worth looking as as.matrix(fit_lme$modelStruct$corStruct)
> ## to look at the different groups
> ## you can choose form that object.  `1/0` is the y_type/time here
> cor_struc <- as.matrix(fit_lme$modelStruct$corStruct)$'1/0'
> R_lme <- diag(c(sigmas)) %*% cor_struc %*% diag(c(sigmas))
> ## bdiag is in the Matrix package
> Sigma_lme <- bdiag(R_lme,R_lme,R_lme,R_lme)
> Sigma_lme
8 x 8 sparse Matrix of class "dgCMatrix"

[1,] 1.4254400 0.4701199 .         .         .         .         .         .
[2,] 0.4701199 0.6169995 .         .         .         .         .         .
[3,] .         .         1.4254400 0.4701199 .         .         .         .
[4,] .         .         0.4701199 0.6169995 .         .         .         .
[5,] .         .         .         .         1.4254400 0.4701199 .         .
[6,] .         .         .         .         0.4701199 0.6169995 .         .
[7,] .         .         .         .         .         .         1.4254400 0.4701199
[8,] .         .         .         .         .         .         0.4701199 0.6169995
>
> ## Z matrix to multiple the random effects covariance matrix by
> ## the columns here are: int y1, time y1, int y2, time y2
> Z <- rbind(c(1,0,0,0),
+            c(0,0,1,0),
+            c(1,2,0,0),
+            c(0,0,1,2),
+            c(1,4,0,0),
+            c(0,0,1,4),
+            c(1,8,0,0),
+            c(0,0,1,8))
>
> ## Overall covariance matrix
```

```
> V_lme <- Z %*% G_lme %*% t(Z) + Sigma_lme
> V_lme
8 x 8 Matrix of class "dgeMatrix"
          [,1]      [,2]      [,3]      [,4]      [,5]      [,6]     [,7]      [,8]
[1,] 2.2857773 1.2224644 0.9228639 0.7331805 0.9853905 0.7140164 1.110444 0.6756882
[2,] 1.2224644 2.6546874 0.8321173 1.9889984 0.9118900 1.9403088 1.071436 1.8429297
[3,] 0.9228639 0.8321173 2.5819588 1.3222100 1.3901736 0.8720630 1.857483 0.9120086
[4,] 0.7331805 1.9889984 1.3222100 2.5689546 0.9709998 1.9149117 1.208819 1.8408251
[5,] 0.9853905 0.9118900 1.3901736 0.9709998 3.2203967 1.5002294 2.604523 1.1483291
[6,] 0.7140164 1.9403088 0.8720630 1.9149117 1.5002294 2.5065141 1.346203 1.8387204
[7,] 1.1104437 1.0714355 1.8574833 1.2088192 2.6045228 1.3462028 5.524042 2.0910899
[8,] 0.6756882 1.8429297 0.9120086 1.8408251 1.1483291 1.8387204 2.091090 2.4515106
>
> ## Overall correlation matrix
> cov2cor(V_lme)
8 x 8 Matrix of class "dpoMatrix"
          [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]
[1,] 1.0000000 0.4962638 0.3798797 0.3025629 0.3631924 0.2983020 0.3125008 0.2854384
[2,] 0.4962638 1.0000000 0.3178361 0.7616396 0.3118754 0.7521926 0.2797891 0.7224124
[3,] 0.3798797 0.3178361 1.0000000 0.5133902 0.4821028 0.3427979 0.4918377 0.3624995
[4,] 0.3025629 0.7616396 0.5133902 1.0000000 0.3375874 0.7546324 0.3208889 0.7335292
[5,] 0.3631924 0.3118754 0.4821028 0.3375874 1.0000000 0.5280413 0.6175114 0.4086907
[6,] 0.2983020 0.7521926 0.3427979 0.7546324 0.5280413 1.0000000 0.3617820 0.7417605
[7,] 0.3125008 0.2797891 0.4918377 0.3208889 0.6175114 0.3617820 1.0000000 0.5682341
[8,] 0.2854384 0.7224124 0.3624995 0.7335292 0.4086907 0.7417605 0.5682341 1.0000000
```

3.2 Generalized Least Squares, Continuous Time

The next model is a generalized least squares model. Using the same notation and indices as before, the model is:

$$y_{i1j} = \beta_{10} + \beta_{11} * A_i + \beta_{12} * t_{ij} + \beta_{13} A_i * t_{ij} + \epsilon_{i1j}$$

and

$$y_{i2j} = \beta_{20} + \beta_{21} * A_i \beta_{22} * t_{ij} + \beta_{23} * A_i * t_{ij} + \epsilon_{i2j}$$

.

All covariance is estimated through the ϵ terms. We have

$$\mathbf{V}_i = Cov \begin{pmatrix} y_{i11} \\ y_{i21} \\ y_{i12} \\ y_{i22} \\ y_{i13} \\ y_{i23} \\ y_{i14} \\ y_{i24} \end{pmatrix} = Cov \begin{pmatrix} \epsilon_{i11} \\ \epsilon_{i21} \\ \epsilon_{i12} \\ \epsilon_{i22} \\ \epsilon_{i13} \\ \epsilon_{i23} \\ \epsilon_{i14} \\ \epsilon_{i24} \end{pmatrix}$$

where V_i is an unstructured, symmetric, and positive definite matrix.

To fit this model we use

```
## gls model
fit_gls <- gls(y~-1+y_type+y_type:(trt+time+trt:time),
               ## correlation between all error pairs
               correlation=corSymm(form=~1|ID),
               ## different error variances for different
               ## y_type_time
               weights=varIdent(form=~1|y_type_time),
               control=cntrl,
               data=dat)
```

3.2.1 Extracting Model Output

Grabbing the model output form the generalized least squares output is less tricky since we do not have to deal with multiple sources of variation. We do need to be careful about the order of terms in the overall covariance matrix however. The overall covariance matrix will coincide with the sorting of the `y_type_time` variable in dataset.

```
> ## arrange data so its y10,y20,y12,y22...
> dat <- dat %>% arrange(ID,time,y_type)
> head(dat,10)
```

```
   ID y_type time           y trt y_type_time
1   1     y1    0  5.1376735   0         y10
2   1     y2    0  2.1755293   0         y20
3   1     y1    2 19.3955216   0         y12
4   1     y2    2  3.2604114   0         y22
5   1     y1    4 34.2279786   0         y14
6   1     y2    4  6.1075059   0         y24
7   1     y1    8 60.6960950   0         y18
8   1     y2    8 11.4424347   0         y28
9   2     y1    0  6.3607717   0         y10
10  2     y2    0 -0.5094834   0         y20
>
> ## fit the model
> fit_gls <- gls(y~-1+y_type+y_type:(trt+time+trt:time),
+                ## correlation between all error pairs
+                correlation=corSymm(form=~1|ID),
+                ## different error variances for different
+                ## y_type_time
+                weights=varIdent(form=~1|y_type_time),
+                control=cntrl,
+                data=dat)
>
> ## summary of fit
> sum_gls <- summary(fit_gls)
>
> ## fixed effects
> sum_gls$tTable
                        Value  Std.Error      t-value       p-value
y_typey1           6.73267668 0.18003379  37.39673883 3.875703e-177
y_typey2           1.00813967 0.21843567   4.61527026  4.578287e-06
y_typey1:trt       0.02271079 0.25460623   0.08919968  9.289458e-01
y_typey2:trt       0.18311983 0.30891469   0.59278445  5.534946e-01
y_typey1:time      6.90940761 0.04024725 171.67402911  0.000000e+00
y_typey2:time      1.23132397 0.01990108  61.87223114 1.526409e-305
y_typey1:trt:time -0.08407603 0.05691821  -1.47713766  1.400363e-01
y_typey2:trt:time -0.07801792 0.02814437  -2.77206116  5.701006e-03
>
> ## overall covariance matrix
> V_gls <- getVarCov(fit_gls,type="marginal",individual=1)
> V_gls
Marginal variance covariance matrix
        [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]
```

```
[1,] 2.27330 1.00080 0.88373 0.69594 0.86798 0.79231 1.09920 0.59965
[2,] 1.00080 2.68880 0.78436 1.96040 0.90397 1.99590 1.06440 1.92080
[3,] 0.88373 0.78436 2.52980 1.33710 1.60970 1.03700 1.74320 0.80218
[4,] 0.69594 1.96040 1.33710 2.47960 0.97702 1.86860 0.97909 1.66040
[5,] 0.86798 0.90397 1.60970 0.97702 3.38150 1.81970 2.66600 1.33120
[6,] 0.79231 1.99590 1.03700 1.86860 1.81970 2.55750 1.44390 1.95830
[7,] 1.09920 1.06440 1.74320 0.97909 2.66600 1.44390 5.42850 1.98440
[8,] 0.59965 1.92080 0.80218 1.66040 1.33120 1.95830 1.98440 2.45110
  Standard Deviations: 1.5077 1.6397 1.5905 1.5747 1.8389 1.5992 2.3299 1.5656
```

3.3 Generalized Least Squares, Categorical Time

The next model we have is similar to before, but now treating time as a categorical variable.

$$y_{i1j} = \beta_{10}+\beta_{11}*A_i+\beta_{12}*I(t_{ij}=1)+\beta_{13}I(t_{ij}=2)+\beta_{14}*A_i*I(t_{ij}=1)+\beta_{15}*A_i*I(t_{ij}=2)+\epsilon_{i1j}$$

and

$$y_{i2j} = \beta_{20}+\beta_{21}*A_i+\beta_{22}*I(t_{ij}=1)+\beta_{23}*I(t_{ij}=2)+\beta_{24}*A_i*I(t_{ij}=1)+\beta_{25}*A_i*I(t_{ij}=2)+\epsilon_{i2j}$$

.

$I()$ is the indicator function, that is:

$$I(x=1) = \begin{cases} 1, & \text{if } x = 1 \\ 0, & \text{otherwise} \end{cases}$$

Here y_{i1j} is the response type 1 for individual i at time point j. y_{i2j} is the response type 2 for individual i at time point j. $A_i \in \{0,1\}$ is the treatment indicator for individual i and $t_{ij} \in \{1,2\}$ is the time variable (hour, week, month, etc) at time point j for individual i. We will refer to $t_{i1}=1, t_{i2}=2$ and treat this as categorical. Additionally, we let

$$\begin{pmatrix} \epsilon_{i11} \\ \epsilon_{i12} \\ \epsilon_{i21} \\ \epsilon_{i22} \end{pmatrix} \sim MVN\Big(\mathbf{0}, \Sigma\Big)$$

where Σ is an unstructured, symmetric, positive definite matrix:

$$\Sigma = \begin{pmatrix} \sigma_{11}^2 & \sigma_{11,12} & \sigma_{11,21} & \sigma_{11,22} \\ . & \sigma_{12}^2 & \sigma_{12,21} & \sigma_{12,22} \\ . & . & \sigma_{21}^2 & \sigma_{21,22} \\ . & . & . & \sigma_{22}^2 \end{pmatrix}$$

.

So our overall, multivariate response is the following:

$$\mathbf{y}_i = \begin{pmatrix} y_{i11} \\ y_{i12} \\ y_{i21} \\ y_{i22} \end{pmatrix} \sim MVN\left(\begin{pmatrix} \mu_{i11} \\ \mu_{i12} \\ \mu_{i21} \\ \mu_{i22} \end{pmatrix}, \begin{pmatrix} \sigma^2_{11} & \sigma_{11,12} & \sigma_{11,21} & \sigma_{11,22} \\ . & \sigma^2_{12} & \sigma_{12,21} & \sigma_{12,22} \\ . & . & \sigma^2_{21} & \sigma_{21,22} \\ . & . & . & \sigma^2_{22} \end{pmatrix}\right)$$

with the mean vector:

$$\begin{pmatrix} \mu_{i11} \\ \mu_{i12} \\ \mu_{i21} \\ \mu_{i22} \end{pmatrix} = \begin{pmatrix} \beta_{10} + \beta_{11} * A_i + \beta_{12} + \beta_{14} * A_i \\ \beta_{10} + \beta_{11} * A_i + \beta_{13} + \beta_{15} * A_i \\ \beta_{20} + \beta_{21} * A_i + \beta_{22} + \beta_{24} * A_i \\ \beta_{20} + \beta_{21} * A_i + \beta_{23} + \beta_{25} * A_i \end{pmatrix}$$

Since we have categorical time points, the original model is overparameterized. Since we use week one as the reference group, the rest of the parameters are then change from week one. Below is the parameterization used to fit the model:

$$\begin{aligned} y_{i1j} &= \gamma_{10} + \gamma_{11} * A_i + \gamma_{12} * I(t_{ij} = 2) + \gamma_{13} * A_i * I(t_{ij} = 2) + \epsilon_{i1j} \\ &= (\beta_{10} + \beta_{12}) + (\beta_{11} + \beta_{14}) * A_1 + (\beta_{13} - \beta_{12}) * I(t_{ij} = 2) + (\beta_{15} - \beta_{14}) * A_i * I(t_{ij} = 2) + \epsilon_{i1j} \end{aligned}$$

and

$$\begin{aligned} y_{i2j} &= \gamma_{20} + \gamma_{21} * A_i + \gamma_{22} * I(t_{ij} = 2) + \gamma_{23} * A_i * I(t_{ij} = 2) + \epsilon_{i2j} \\ &= (\beta_{20} + \beta_{22}) + (\beta_{21} + \beta_{24}) * A_1 + (\beta_{23} - \beta_{22}) * I(t_{ij} = 2) + (\beta_{25} - \beta_{24}) * A_i * I(t_{ij} = 2) + \epsilon_{i2j} \end{aligned}$$

We fit this model with

```
## fit the model
fit_gls_cat <- gls(y~-1+y_type_time+y_type_time:trt,
                   ## correlation between all error pairs
                   correlation=corSymm(form=~1|ID),
                   ## different error variances for different
                   ## y_type_time
                   weights=varIdent(form=~1|y_type_time),
                   control=cntrl,
                   data=dat)
```

3.3.1 Extracting Model Output

Obtaining the model output for the categorical generalized least squares model is similar to the continuous generalized least squares model. The same sorting of the data is necessary for the overall covariance matrix.

```
> ## just include two time points
> dat2 <- dat %>% filter(time<=2)
>
> ## fit the model
> fit_gls_cat <- gls(y~-1+y_type_time+y_type_time:trt,
+                 ## correlation between all error pairs
+                 correlation=corSymm(form=~1|ID),
+                 ## different error variances for different
+                 ## y_type_time
+                 weights=varIdent(form=~1|y_type_time),
+                 control=cntrl,
+                 data=dat2)
>
> ## summary of fit
> sum_gls_cat <- summary(fit_gls_cat)
>
> ## fixed effects
> sum_gls_cat$tTable
                         Value Std.Error    t-value       p-value
y_type_timey10      6.70280649 0.2137457 31.3587934 6.776852e-109
y_type_timey12     20.75935364 0.2248467 92.3266963 4.606600e-268
y_type_timey20      0.95656476 0.2313376  4.1349292  4.345880e-05
y_type_timey22      3.52564782 0.2226557 15.8345294  5.222579e-44
y_type_timey10:trt  0.11176252 0.3022820  0.3697293  7.117839e-01
y_type_timey12:trt -0.31184242 0.3179813 -0.9806943  3.273484e-01
y_type_timey20:trt  0.04892815 0.3271608  0.1495538  8.811936e-01
y_type_timey22:trt  0.08896029 0.3148827  0.2825188  7.776948e-01
>
> ## overall covariance matrix
> V_gls_cat <- getVarCov(fit_gls_cat,type="marginal",individual=1)
> V_gls_cat
Marginal variance covariance matrix
        [,1]    [,2]    [,3]    [,4]
[1,] 2.28440 1.00550 0.87519 0.68889
[2,] 1.00550 2.67590 0.78988 1.96790
[3,] 0.87519 0.78988 2.52780 1.33640
[4,] 0.68889 1.96790 1.33640 2.47880
  Standard Deviations: 1.5114 1.6358 1.5899 1.5744
```

4 Model Building Suggestions

Based on these examples, we believe we have laid the foundations for fitting a variety of linear multivariate longitudinal models in the `nlme` package. Based on our experience through simulations and real data examples, we have come up with the following model building suggestions.

Model building is an art, not a science, so there is not one process on how to build a multivariate longitudinal model. We first see what we can fit, then evaluate which models best answer our scientific questions. Based on our experience, the following offers a general sequence of steps that can help a researcher build a useful multivariate longitudinal model.

1. Fit as many relevant models as possible. With real data, convergence issues are common, so making slight adjustments to mean and covariance structures is often necessary.

2. Examine model output like $\hat{\mathbf{V}}_i, \hat{\beta}, \hat{\mathbf{R}}, \hat{\mathbf{G}}$. If a model(s) looks to have considerably different output than others, consider why that might be (certain parameter restrictions, false convergence, etc). If the outlier model(s) has an intuitive reason for being different and information criteria suggest a worse fit, then throw it out.

3. Evaluate all models using information criteria, likelihood ratio tests, and practical use (e.g. if examination of random effects' covariance structure is important scientifically, give favor to a mixed model) and choose accordingly.

4. When conducting inference on functions of mean parameters, it can be useful to conduct inference using different reasonable models. If the models agree, then the we can be more confident in the inference. If they disagree, we should be wary.