ST 437/537: Applied Multivariate and Longitudinal Data Analysis

Longitudinal Data Analysis: Estimation in the General Linear Model

Arnab Maity
NCSU Department of Statistics
SAS Hall 5240   919-515-1937   amaity[at]ncsu.edu

References:

- Modeling Longitudinal Data by Robert E. Weiss. New York: Springer.
- Linear Mixed Models for Longitudinal Data by Geert Verbeke and Geert Molenberghs. New York: Springer.
- Applied Longitudinal Analysis by Fitzmaurice by G.M., Laird, N.M., and Ware, J.H. New York: Wiley (on reserve at NCSU library)

Introduction

So far, we have discussed the three main steps in modeling longitudinal data:

- modeling the mean,
- modeling the covariance,
- and selecting the distribution of the data Y.

We have alredy established that we can write a **general linear model** for the i-subject:

$$\boldsymbol{Y}_i = \boldsymbol{X}_i\beta + \boldsymbol{e}_i,$$

where $\boldsymbol{e}_i$ is m_i-dimensional vector of random deviations, β is the fixed effects parameter and corresponds to the design matrix X_i; β (often called the **mean regression parameter**) is the main object of inference. The term $\boldsymbol{e}_i$ is the deviation from the systematic component, which has a multivariate random distribution with mean 0 and covariance matrix $\Sigma_i = \Sigma_i(\omega)$. Here ω is a vector of unknown parameter in the covariance model; these parameter in ω are often called **variance components**.

In this chapter we assume that the responses are normally distributed; that is

$$\boldsymbol{Y}_i \sim N(\boldsymbol{X}_i\beta, \boldsymbol{\Sigma}_i(\omega)).$$

Remark: This approach separates the modeling of the mean (systematic component) and the correlation of the random component; the covariance for the random component does not distinguish between the two main sources of variability (between units and among units). Modeling the correlation in longitudinal data is important to be able to obtain correct inferences on regression coefficients β. The correlation model does not change the interpretation of the β parameters.

Estimation of the regression parameters (β)

Consider a framework for the estimation of the unknown parameters: the mean regression parameters (β) and the variance parameters (ω). When full distributional assumptions have been made about the vector of responses, a standard approach is to employ **Maximum Likelihood Estimation (MLE)**.

Main idea of MLE: The main idea in the MLE is to estimate the parameters by the values that make the observed data most likely to have occurred, under the specified model.

For simplicity assume first that the covariance parameters ω are *known*. In other words, $\mathbf{\Sigma}_i(\omega)$ is known. As usual, we use hat to denote parameter estimators (e.g., $\widehat{\beta}$ will denote an estimator β).

> *(Math details for the curious: not needed for exam) To obtain the MLE of β we need to maximize the following log-likelihood function:*
>
> $$\ell(\beta) = \frac{1}{2}(\sum_{i=1}^{n} m_i) \log(2\pi) - \frac{1}{2}\sum_{i=1}^{n} \log|\Sigma_i| - \frac{1}{2}\left\{ \sum_{i=1}^{n} (\boldsymbol{Y}_i - X_i\beta)^T \Sigma_i^{-1} (\boldsymbol{Y}_i - X_i\beta) \right\};$$
>
> *Thus,*
>
> $$\widehat{\beta}_{MLE} = \text{argmax}_{\beta} \ell(\beta).$$
>
> *Since β does not appear in the first two terms, it follows that maximization of the log-likelihood function $\ell(\beta)$ is equivalent to minimization of:*
>
> $$\sum_{i=1}^{n} (\boldsymbol{Y}_i - X_i\beta)^T \Sigma_i^{-1} (\boldsymbol{Y}_i - X_i\beta);$$
>
> *After some algebra, we can obtain the estimator of β as*
>
> $$\widehat{\beta}_{MLE} = \text{argmin}_{\beta} \sum_{i=1}^{n} (\boldsymbol{Y}_i - X_i\beta)^T \Sigma_i^{-1} (\boldsymbol{Y}_i - X_i\beta).$$

We can show that the **maximum likelihood estimator** is:

$$\widehat{\beta}_{MLE} = \left\{ \sum_{i=1}^{n} (\boldsymbol{X}_i^T \mathbf{\Sigma}_i^{-1} \boldsymbol{X}_i) \right\}^{-1} \sum_{i=1}^{n} (\boldsymbol{X}_i^T \mathbf{\Sigma}_i^{-1} \boldsymbol{Y}_i);$$

The solution presented above is also refered to as the **generalized least squares (GLS)** estimator of β, also denoted as $\widehat{\beta}_{GLS}$. From now on, we will simply refer the estimator as $\widehat{\beta}$.

Properties of the $\widehat{\beta}$

1. $\widehat{\beta}$ is an **unbiased estimator** of β: $E(\widehat{\beta}) = \beta$; This is true **even if we misspecify the covariance structure**.
2. The sampling distribution of $\widehat{\beta}$ is multivariate normal, that is,

$$\widehat{\beta} \sim N\left(\beta, \left\{\sum_{i=1}^{n}(\boldsymbol{X}_i^T \boldsymbol{\Sigma}_i^{-1} \boldsymbol{X}_i)\right\}^{-1}\right).$$

3. **Irrespective of the error distribution**, the GLS estimator $\widehat{\beta}$ is the *best linear unbiased estimator (BLUE)* for β. Specifically, it has “smaller” variance than any other linear estimator of β.
4. **If the error distribution is multovariate normal**, then $\widehat{\beta}$ is the *uniformly minimum variance unbiased estimator (UMVUE)* for β. Specifically, it has “smaller” variance than any other estimator (linear or nonlinear) of β.

Estimation of the variance components (ω)

In practice the covariance parameter ω is not known. Typically Maximum Likelihood Estimation (MLE) or Restricted Maximum Likelihood (REML) estimation is used to obtain an estimate for ω. The ML/REML estimator $\widehat{\omega}$ does not have a close form simple expression; numerical algorithms are used to obtain $\widehat{\omega}$. When such an estimate is obtained then $\widehat{\Sigma}_i = \Sigma_i(\widehat{\omega})$ is substituted in the expression of $\widehat{\beta}$.

When the sample size n is large, the resulting estimator $\widehat{\beta}$ will *approximately* have all the same properties as if ω, and thus Σ_i were known.

Insight: The MLE of the variance component ω is typically biased. Bias arises because the ML estimate $\widehat{\omega}$ does not take into account that β is also estimated. The theory of restricted maximum likelihood (REML) was precisely developed to address this limitation. The REML likelihood is the function for the marginal distribution of the residuals. **REML produces estimates of the variance/covariance parameters that are unbiased**. REML estimation is thus the default method used to estimate the variance component parameters for many algorithms.

> *(Math details for the curious: not needed for exam) In a nutshell, REML approach uses a ML function calculated to a transformed data in a way that ensures that the nuisance parameters have no effect. Intuition behind the procedure: Transform data Y to $Y^* = A^T Y$ where matrix A is chosen $N \times (N-k)$ to make the distribution of Y^* free of β. Here $N = \sum_{i=1}^n m_i$. For example consider A such that $\{I - X(X^TX)^{-1}X^T\} = AA^T$ and $A^TA = I_{N-k}$; then Y^* has multivariate normal distribution, with mean zero and covariance equal to $A\Sigma A^T$ which is free of β. The covariance estimators are obtained by maximizing the likelihood of Y^*. Remark that this likelihood function (which is called the REML function) is in fact the product between the original likelihood function and an `adjustment' factor. The adjustment factor is $\prod_{i=1}^n |X_i^T \Sigma_i^{-1}(\omega) X_i|^{-1/2}$. The REML log-likelihood function is:*
>
> $$\ell_{REML}(\beta,\omega) = \frac{\sum_{i=1}^n m_i}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^n \log|\Sigma_i(\omega)| - \frac{1}{2}\left\{\sum_{i=1}^n (Y_i - X_i\beta)^T \Sigma_i(\omega)^{-1}(Y_i - X_i\beta)\right\}$$
> $$-\frac{1}{2}\sum_{i=1}^n \log|X_i^T\Sigma_i^{-1}(\omega)X_i|;$$
>
> *The solution, again, is obtained by numerical optimization. The REML estimator of ω, $\widehat{\omega}_{REML}$, is unbiased of ω. Since the adjustment is a function solely of ω, the ML and REML-based estimators of the mean regression parameter β coincide.*

Example: the Vlagtwedde-Vlaardingen Study

The original dataset is availabl at [**https://content.sph.harvard.edu/fitzmaur/ala2e/ (https://content.sph.harvard.edu/fitzmaur/ala2e/)**].

Study description (as given in the website above): This is an epidemiologic study conducted in two different areas in the Netherlands - the rural area of Vlagtwedde (N-E) and the urban, industrial area of Vlaardingen (S-W). The residents were followed over time to obtain information on the prevalence of and risk factors for chronic obstructive lung diseases.

This dataset is based on the sample of men and women from the rural area of Vlagtwedde. The sample, initially aged 15-44, participated in follow-up surveys approximately every 3 years for up to 21 years. At each survey, information on respiratory symptoms and smoking status was collected by questionnaire and spirometry was performed. Pulmonary function was determined by spirometry and a measure of *forced expiratory volume (FEV1)* was obtained every three years for the first 15 years of the study, and also at year 19.

The dataset is comprised of a sub-sample of 133 residents aged 36 or older at their entry into the study and whose smoking status did not change over the 19 years of follow-up. Each study participant was either a current or former smoker. Current smoking was defined as smoking at least one cigarette per day. In this dataset FEV1 was not recorded for every subject at each of the planned measurement occasions. The number of repeated measurements of FEV1 on each subject varied from 1 to 7.

Questions of interest:

- How the pulmonary function change over time ?
- Is this different for current smokers than for former ones?

```
# read data
smoking <- read.table("data/smoking.txt")
names(smoking) <-  c("id", "smoker", "time", "FEV1")

## view the first few rows of the dataset
head(smoking)
```

```
##   id smoker time FEV1
## 1  1      0    0 3.40
## 2  1      0    3 3.40
## 3  1      0    6 3.45
## 4  1      0    9 3.20
## 5  1      0   15 2.95
## 6  1      0   19 2.40
```

```
## Observations per subject
table(tabulate(smoking$id))
```

```
## 
##  1  5  6  7 
##  1 54 46 32
```

```
## Observations in each smoking group (0=former, 1=current)
table(smoking$smoker)
```

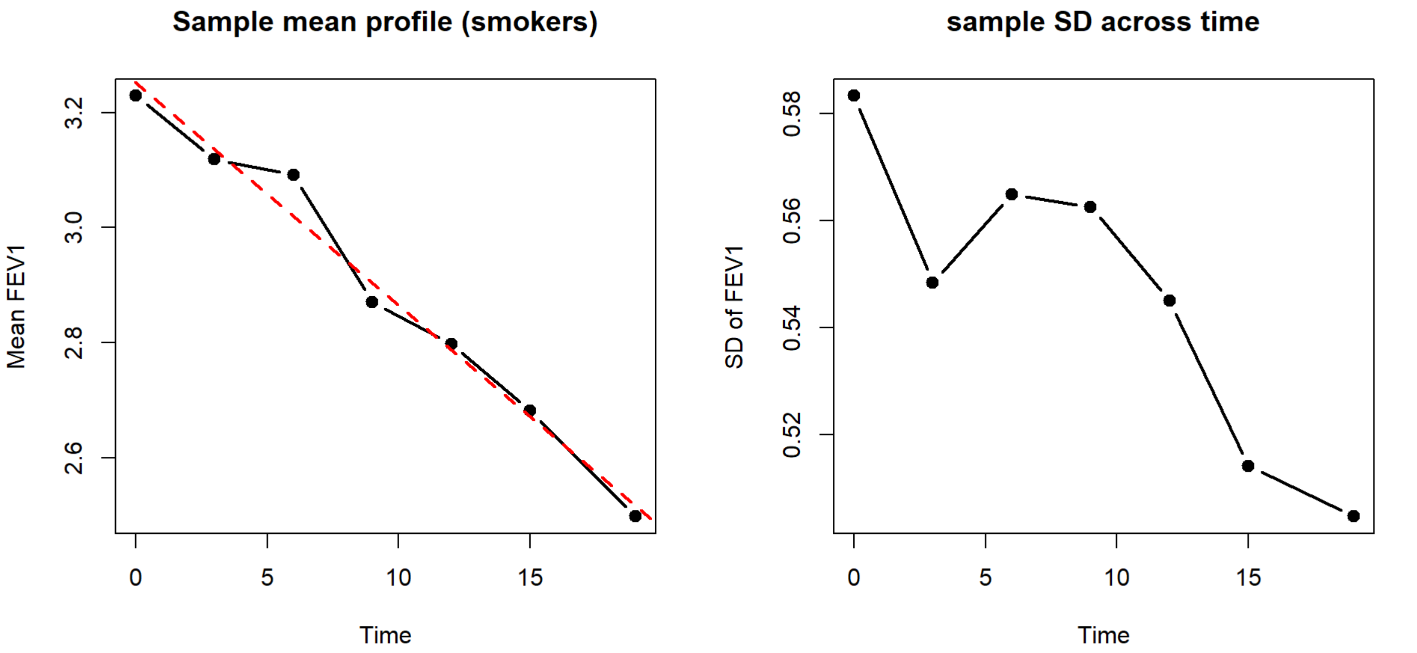
```
## 
##   0   1 
## 189 582
```

```
## Observations in each smoking group (0=former, 1=current)
## grouped time time points
table(smoking$smoker, smoking$time)
```

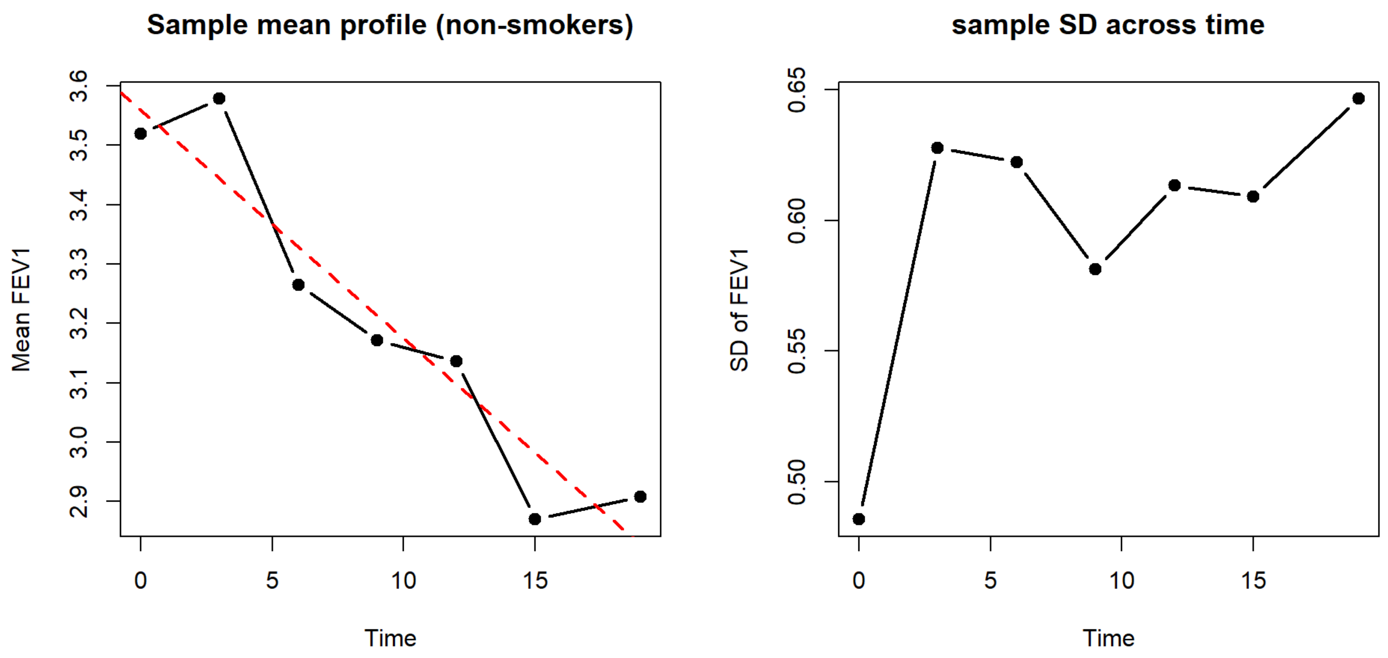
```
##    
##      0  3  6  9 12 15 19
##   0 23 27 28 30 29 24 28
##   1 85 95 89 85 81 73 74
```

Let us use various visualization tools to assess the mean behavior over time, gain insight into the dependence over time.

```
###
# Find the sample mean and sd profiles for
# each group (smoker/nonsmoker)
###

# Observed time points
tm <- c(seq(0, 15, by = 3), 19)

# smokers
sm <- subset(smoking, smoker == 1)
# non-smokers
non <- subset(smoking, smoker == 0)

# profiles for smokers
dat <- sm
muhat <- rep(NA, length(tm))
sdhat <- rep(NA, length(tm))
for(ii in 1:length(tm)){
  tmp <- subset(dat, time == tm[ii])
  muhat[ii] <- mean(tmp$FEV1)
  sdhat[ii] <- sd(tmp$FEV1)
}
par(mfrow = c(1,2))
plot(tm, muhat, xlab = "Time", ylab = "Mean FEV1", main = "Sample mean profile (smokers)", type="b", lwd=2,
 pch=19)
abline(lm(muhat ~ tm), lwd=2, col="red", lty=2)
plot(tm, sdhat, xlab = "Time", ylab = "SD of FEV1", main = "sample SD across time", type="b", lwd=2, pch=19)
```



```r
# profiles for non-smokers
dat <- non
muhat <- rep(NA, length(tm))
sdhat <- rep(NA, length(tm))
for(ii in 1:length(tm)){
  tmp <- subset(dat, time == tm[ii])
  muhat[ii] <- mean(tmp$FEV1)
  sdhat[ii] <- sd(tmp$FEV1)
}
par(mfrow = c(1,2))
plot(tm, muhat, xlab = "Time", ylab = "Mean FEV1", main = "Sample mean profile (non-smokers)", type="b", lwd
=2, pch=19)
abline(lm(muhat ~ tm), lwd=2, col="red", lty=2)
plot(tm, sdhat, xlab = "Time", ylab = "SD of FEV1", main = "sample SD across time", type="b", lwd=2, pch=19)
```



It is evident that a linear function of `time` is sufficient for both the groups. Similar analysis can be performed to determine that we could use a compound symmetric correlation structure for dependence. We also assume the covariance matrix is the same for both groups.

Next, we write down a parametric model for both mean and covariance. Following the discussion in the previous chapter, we write the mean model as

$$Y_{ij} = \beta_1 + \beta_2 t_{ij} + \beta_3 S_{ij} + \beta_4 t_{ij} S_i + e_{ij},$$

where

- t_{ij} are observed time point for the j-measurement of the i-th person, and
- S_i is a dummy variable for whether the i-th person is a smoker or not (1 = smoker, 0 = non-smoker).

For the covariance model (compound symmetric), we assume that $var(e_{ij}) = \sigma^2$, and $cov(e_{ij}, e_{ij'}) = \sigma^2\rho, j \neq j'$.

To fit this model in R, we can use the `gls()` function in the `nlme` package.

> *gls(model, data, correlation, weights, subset, method, na.action, control, verbose)*

- **model:** a formula that specifies the mean model. This is typically of the form `y ~ x1 + x2` where `y` is the response variable, and `x1` , `x2` etc are predictors
- **data:** dataset (a data frame) that we have
- **correlation:** the corrrelation structure we are going to use.
- **weights:** any specification of variances we might have
- **subset:** if we intend to use only a subset of the dataset (e.g. `smoker==0`)
- **method:** "REML" or "ML" to estimate β and ω.

Often it helps to convert the grouping variable (e.g., `smoker` and `id`) to factors. This way, R can automatically create dummy variables when needed.

```
# change 'id' and 'smoker' to factor variables
smoking <- within(smoking, {
  id <- factor(id)
  smoker <- factor(smoker, levels = 0:1, labels = c("former", "current"))
})
head(smoking)
```

```
##   id smoker time FEV1
## 1  1 former    0 3.40
## 2  1 former    3 3.40
## 3  1 former    6 3.45
## 4  1 former    9 3.20
## 5  1 former   15 2.95
## 6  1 former   19 2.40
```

We now set the formula in R format. We can do this step directly in `gls` ; however, I recommend to define the formula separately especially if it involves a lot ot covariate terms and interactions.

```
form <- FEV1 ~ time * smoker
```

Thus we are fitting a linear function in `time` and allowing the intercept and slope to be different in the `smoker=0` and `smoker=1` groups. This will model the baseline (`smoker=0` group) intercept and slope (β_0 and β_1) directly; then model the *change* in intercept and slope in the other group (β_2 and β_3). Thus we have

$$\text{Former smokers}(S_i = 0) : E(Y_{ij}) = \beta_1 + \beta_2 t_{ij},$$

$$\text{Current smokers}(S_i = 1) : E(Y_{ij}) = (\beta_1 + \beta_3) + (\beta_2 + \beta_4) t_{ij}.$$

Let us examine the design matrix for a particular individual (that is, $\boldsymbol{X}_i$ for some i). We can do so by using the `model.matrix()` function. This is a good way to check whether the formula we intend to use is indeed the one we want to fit.

```
model.matrix(form, data = smoking[smoking$id==1,])
```

```
##   (Intercept) time smokercurrent time:smokercurrent
## 1           1    0             0                  0
## 2           1    3             0                  0
## 3           1    6             0                  0
## 4           1    9             0                  0
## 5           1   15             0                  0
## 6           1   19             0                  0
## attr(,"assign")
## [1] 0 1 2 3
## attr(,"contrasts")
## attr(,"contrasts")$smoker
## [1] "contr.treatment"
```

Fitting the **compound symmetry** covariance model

```
library(nlme)

# Compound symmetric correlation
#     - is specified by corCompSymm( , form= ~ 1 | id )
#       where the 'id' variable following the bar notation
#       indicates that observations are repeated within id.
gls.fit.exch <- gls(model = form,
                    data = smoking,
                    correlation = corCompSymm(form= ~ 1 | id ))
```

Explanation of the model specification:

- The `model` argument specifies the formula specified in `form`, that is, `FEV1 ~ time + smoker + smoker*time`. This fit the mean model model:

$$E(Y_{ij}) = \beta_1 + \beta_2 t_{ij} + \beta_3 S_i + \beta_4 t_{ij} S_i,$$

  where Y_{ij} is the FEV1 measure of the i-th subject at time t_{ij}, S_i is the dummy variable for smoking status. Note that even though we have not included an intercept directly in the formula, R will automatically include the intercept term.

- The `data` argument specified the dataset we are using. So R knows where to find the variables refered in the formula (e.g., `FEV1`, `time`, `smoker` etc.)

- the argument `correlation = corCompSymm( , form= ~ 1 | id )` specifies the correlation structure. Specifically, the `corCompSymm()` function specifies the compound symmetric corrrelation structure. The argument in this function `form= ~ 1 | id` defines that compound symmetric correlation structure should be defined for *each id* (that is, observations with the same `id` value belong to the same subject).

- Note that we have left the `weights` argument unspecified. This enforces R to use same variance for all time points.

```
summary(gls.fit.exch)
```

```
## Generalized least squares fit by REML
##   Model: form
##   Data: smoking
##        AIC      BIC    logLik
##   323.6031 351.4581 -155.8016
##
## Correlation Structure: Compound symmetry
##  Formula: ~1 | id
##  Parameter estimate(s):
##       Rho
## 0.8595179
##
## Coefficients:
##                        Value  Std.Error   t-value p-value
## (Intercept)         3.507677 0.09804324  35.77684  0.0000
## time               -0.033852 0.00262840 -12.87912  0.0000
## smokercurrent      -0.272676 0.11239910  -2.42596  0.0155
## time:smokercurrent -0.004570 0.00301829  -1.51419  0.1304
##
##  Correlation:
##                    (Intr) time   smkrcr
## time               -0.250
## smokercurrent      -0.872  0.218
## time:smokercurrent  0.218 -0.871 -0.246
##
## Standardized residuals:
##         Min          Q1         Med          Q3         Max
## -2.97625810 -0.65102769  0.01890054  0.68738226  3.14895565
##
## Residual standard error: 0.5711548
## Degrees of freedom: 771 total; 767 residual
```

Reading the output above:

1. The first block “Generalized least squares fit by REML” tells us that the REML procedure was used to estimate the parameters. This indicates that the estimated variance components are unbiased as well. This block also gives us the model specification and data used, and also some goodness-of-fit measurements (AIC, BIC). We will not use these measures yet; they are used to select models given various choices of mean and covariance models. The `logLik` value gives the log-likelihood value that can be used to formally test goodness-of-fit between two models.
2. The second block gives us the estimate of the variance parameters. It varifies that `Correlation Structure: Compound symmetry` and provids an estimate of ρ (the correlation) as $\hat{\rho} = 0.8595179$.
3. The third block gives the estimated regression coefficients (the β parameters). **Assuming that we have specified the correct covariance**, we can rely on the p-values to test for significance for each β coefficients.
   We can just obtain the results in this block by calling `anova()` .

```
anova(gls.fit.exch, type = "marginal")
```

```
## Denom. DF: 767
##             numDF   F-value p-value
## (Intercept)     1 1279.9824  <.0001
## time            1  165.8718  <.0001
## smoker          1    5.8853  0.0155
## time:smoker     1    2.2928  0.1304
```

4. The fourth block gives the correlation among the *estimated coefficients* in $\widehat{\beta}$. We can obtain the **variance-covariance** matrix of the estimated regression coefficient $\widehat{\beta}$, that is, $cov(\widehat{\beta})$, as below. Entries of this matrix will be used to test for specific contrasts.

```
gls.fit.exch$varBeta
```

```
##                      (Intercept)          time smokercurrent
## (Intercept)         9.612476e-03 -6.454309e-05 -9.612476e-03
## time               -6.454309e-05  6.908510e-06  6.454309e-05
## smokercurrent      -9.612476e-03  6.454309e-05  1.263356e-02
## time:smokercurrent  6.454309e-05 -6.908510e-06 -8.344302e-05
##                    time:smokercurrent
## (Intercept)              6.454309e-05
## time                    -6.908510e-06
## smokercurrent           -8.344302e-05
## time:smokercurrent       9.110083e-06
```

```
# Correlation
cov2cor(gls.fit.exch$varBeta)
```

```
##                    (Intercept)       time smokercurrent time:smokercurrent
## (Intercept)          1.0000000 -0.2504609    -0.8722778          0.2181077
## time                -0.2504609  1.0000000     0.2184715         -0.8708253
## smokercurrent       -0.8722778  0.2184715     1.0000000         -0.2459609
## time:smokercurrent   0.2181077 -0.8708253    -0.2459609          1.0000000
```

5. The fifth block `Standardized residuals` provides a summary of residuals (scaled by their SD). If they are indeed normal, most values should be -3 to 3.

6. The last two lines give the estimated SD of errors `Residual standard error: 0.5711548`, that is, $\widehat{\sigma} = 0.57$, and the degrees of freedoms of the model componets `## Degrees of freedom: 771 total; 767 residual`. The total degrees of freedom is essentially the number of row in the data matrix; the residual degrees of fredom is (total DF - number of parameters in the mean model) = 771 - 4 = 767. These values will be used later to formally test goodness-of-fit.
   We can obtain $\widehat{\sigma}$ also by calling the `sigma()` function:

```
sigma(gls.fit.exch)
```

```
## [1] 0.5711548
```

The covariance/correlation matrix of a particular subject can be extracted as below.

```
# Covariance matrix
Sigma <- getVarCov(gls.fit.exch, individual = 1)
Sigma
```

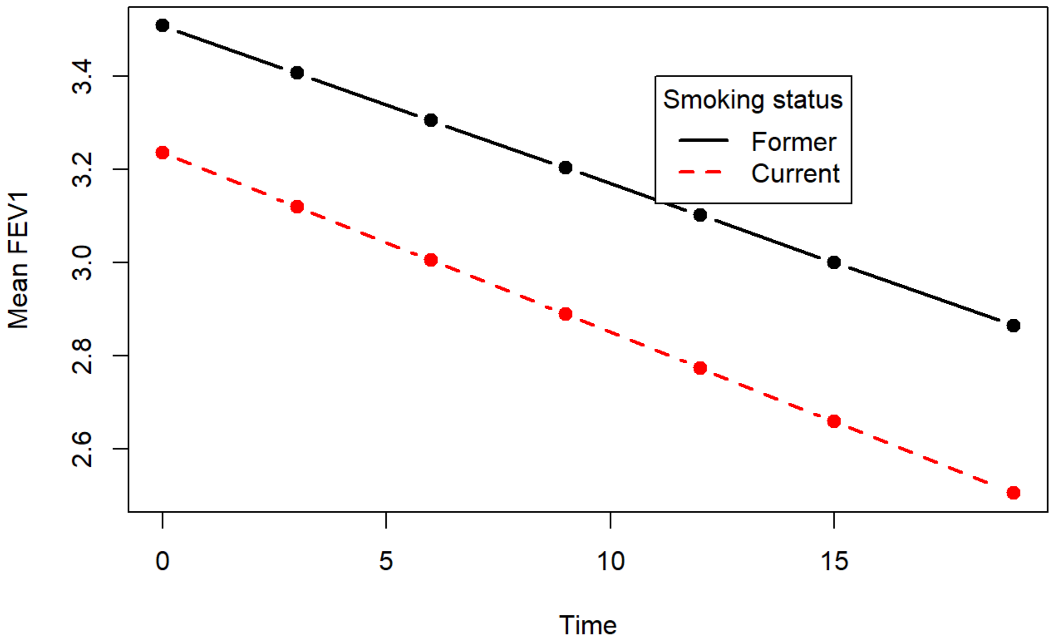
```
## Marginal variance covariance matrix
##         [,1]    [,2]    [,3]    [,4]    [,5]    [,6]
## [1,] 0.32622 0.28039 0.28039 0.28039 0.28039 0.28039
## [2,] 0.28039 0.32622 0.28039 0.28039 0.28039 0.28039
## [3,] 0.28039 0.28039 0.32622 0.28039 0.28039 0.28039
## [4,] 0.28039 0.28039 0.28039 0.32622 0.28039 0.28039
## [5,] 0.28039 0.28039 0.28039 0.28039 0.32622 0.28039
## [6,] 0.28039 0.28039 0.28039 0.28039 0.28039 0.32622
##   Standard Deviations: 0.57115 0.57115 0.57115 0.57115 0.57115 0.57115
```

```
# Correlation matrix
cov2cor(Sigma)
```

```
## Marginal variance covariance matrix
##         [,1]    [,2]    [,3]    [,4]    [,5]    [,6]
## [1,] 1.00000 0.85952 0.85952 0.85952 0.85952 0.85952
## [2,] 0.85952 1.00000 0.85952 0.85952 0.85952 0.85952
## [3,] 0.85952 0.85952 1.00000 0.85952 0.85952 0.85952
## [4,] 0.85952 0.85952 0.85952 1.00000 0.85952 0.85952
## [5,] 0.85952 0.85952 0.85952 0.85952 1.00000 0.85952
## [6,] 0.85952 0.85952 0.85952 0.85952 0.85952 1.00000
##   Standard Deviations: 1 1 1 1 1 1
```

Let us now estimate the mean trajectories of the two groups (former and current smokers). Recall, our baseline was “Former” ($S = 0$) group.

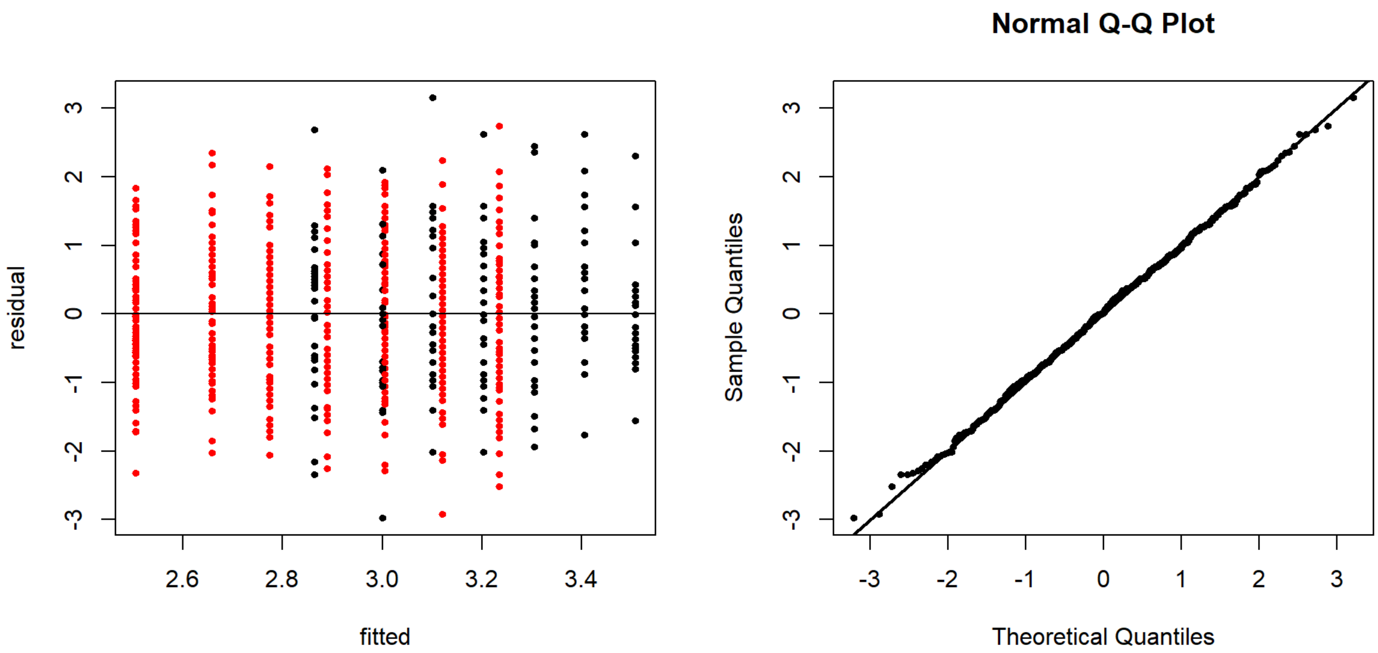
```
# Estimated coefficients
cf <- gls.fit.exch$coefficients

# Observation times
tm <- c(seq(0, 15, by = 3), 19)

# Mean trajectory for former (S = 0)
# beta_1 + beta_2*t
mu.former <- cf[1] + cf[2]*tm

# Mean trajectory for current smokers (S = 1)
# (beta_1 + beta_3) + (beta_2 + \beta_4)*t
mu.current <- (cf[1] + cf[3]) + (cf[2] + cf[4])*tm

# Plot
matplot(tm, cbind(mu.former, mu.current),
        type="b", pch=19, lwd=2, lty = 1:2,
        xlab = "Time", ylab = "Mean FEV1")
legend(x = 11, y = 3.4, legend = c("Former", "Current"), lty=1:2,
       col = c("black", "red"), lwd = 2, title = "Smoking status")
```



Some diagonstic plots:

We can create a few diagnostic plots to assess whether the model fits the data well and whether the normality assumption is reasonable for the errors.

```
par(mfrow=c(1,2))

# Get the residuals (standardized)
residual <- residuals(gls.fit.exch, type='pearson')

# The fitted values
fitted <- fitted(gls.fit.exch)

# Plot residuals vs fitted
plot(fitted, residual, pch=19, cex=0.6, col=smoking$smoker)
abline(h=0)

qqnorm(residual, pch=19, cex=0.6)
abline(0,1, lwd=2)
```



The command `residuals()` is used to extract residuals from the model fit. If we omit the `type='pearson'` argument, we will only obtain the raw residuals; these are difficult to visualize especially if the variance varies over time. The “pearson residuals” are the standardized residuals (raw residuals divided by the corresponding standard errors). See `?residuals.gls` for more details.

The left plot shows the usual residual plot. We see that there is no visible trend/pattern indicating a good model fit. The right plot is a normal Q-Q plot; the straight line pattern indicates that the normality assumption is reasonable.

Compound symmetry covariance model with **unequal variance**

To specify unequal variancees over time, we need to use the `weights` argument and the `varIdent()` function.

```
gls.cs.uv <- gls(model = form,
                    data = smoking,
                    correlation = corCompSymm(form= ~ 1 | id ),
                    weights = varIdent(form = ~ 1|time)
                    )
```

The line `weights = varIdent(form = ~ 1|time)` specifies unequal variances, that is, we set a different variance parameter at each time point. Specifically, `form = ~ 1|time` enforces that the variance depends on `time`.

```
summary(gls.cs.uv)
```

```
## Generalized least squares fit by REML
##   Model: form
##   Data: smoking
##        AIC      BIC    logLik
##   331.1515 386.8613 -153.5757
##
## Correlation Structure: Compound symmetry
##  Formula: ~1 | id
##  Parameter estimate(s):
##      Rho
## 0.860502
## Variance function:
##  Structure: Different standard deviations per stratum
##  Formula: ~1 | time
##  Parameter estimates:
##         0         3         6         9        15        19        12
## 1.0000000 0.9660445 0.9428716 0.9524425 0.9261668 0.9170897 0.9168496
##
## Coefficients:
##                        Value  Std.Error   t-value p-value
## (Intercept)         3.504139 0.10119171  34.62871  0.0000
## time               -0.033629 0.00265466 -12.66775  0.0000
## smokercurrent      -0.256179 0.11606395  -2.20722  0.0276
## time:smokercurrent -0.004861 0.00304898  -1.59444  0.1113
##
##  Correlation:
##                    (Intr) time   smkrcr
## time               -0.384
## smokercurrent      -0.872  0.335
## time:smokercurrent  0.335 -0.871 -0.381
##
## Standardized residuals:
##           Min            Q1           Med            Q3           Max
## -3.0409686866 -0.6627688481  0.0005168067  0.6730519635  3.2520442372
##
## Residual standard error: 0.6034954
## Degrees of freedom: 771 total; 767 residual
```

We will focus on the `Correlation Structure` section of the output shown above. We can obtain just this part by using the command

```
summary(gls.cs.uv$modelStruct)
```

```
## Correlation Structure: Compound symmetry
##  Formula: ~1 | id
##  Parameter estimate(s):
##      Rho
## 0.860502
## Variance function:
##  Structure: Different standard deviations per stratum
##  Formula: ~1 | time
##  Parameter estimates:
##         0         3         6         9        15        19        12
## 1.0000000 0.9660445 0.9428716 0.9524425 0.9261668 0.9170897 0.9168496
```

Under the section `Variance function`, we see information about the variances. Specifically, the `Parameter estimates` part gives the so-called *inflation factors for the variance:* 1, $\sigma_2^2/\sigma_1^2, \ldots, \sigma_m^2/\sigma_1^2$.

The estimate of σ_1^2 (variance at the first time point) is

```
sigma(gls.cs.uv)^2
```

```
## [1] 0.3642067
```

Thus variance at the 2nd time point is $\sigma_2^2 = \sigma_1^2$×(inflation factor) = 0.3642067 × 0.9660445 = 0.3518398.

The covariance matrix for one subject is

```
# Covariance matrix
Sigma <- getVarCov(gls.cs.uv, individual = 1)
Sigma
```

```
## Marginal variance covariance matrix
##         [,1]    [,2]    [,3]    [,4]    [,5]    [,6]
## [1,] 0.36421 0.30276 0.29550 0.29850 0.29026 0.28742
## [2,] 0.30276 0.33989 0.28546 0.28836 0.28041 0.27766
## [3,] 0.29550 0.28546 0.32378 0.28144 0.27368 0.27100
## [4,] 0.29850 0.28836 0.28144 0.33039 0.27646 0.27375
## [5,] 0.29026 0.28041 0.27368 0.27646 0.31241 0.26620
## [6,] 0.28742 0.27766 0.27100 0.27375 0.26620 0.30632
##   Standard Deviations: 0.6035 0.583 0.56902 0.57479 0.55894 0.55346
```

It seems that our original decision of fitting an equal variance model is justified since the SD values accros time points are very similar (equivalently, the inflation factors in the variance estimates above are close to 1).

Unstructured covariance model with **unequal variances**

The most general covariance model is the unstructured model (no specific patter among the correlation/covariances) and each time point has a different variance parameter.

```
# Create a factor variable for time
smoking$timefact <- factor(smoking$time, labels = 0:6)

gls.un <- gls(model = form,
                    data = smoking,
                    correlation = corSymm(form= ~ as.numeric(timefact) | id ),
                    weights = varIdent(form = ~ 1|time)
                    )
```

The line `correlation = corSymm( , form= ~ time | id )` specifies that the subjects are specified using `id` (all observations with the same `id` value belongs to one subject) and that separate correlation parameters should be use for each time point (`~ time | id`).

The line `weights = varIdent(form = ~ 1|time)` specifies unequal variances, that is, we set a different variance parameter at each time point. Specifically, `form = ~ 1|time` enforces that the variance depends on `time`.

The estimated covariance information is shown below.

```
summary(gls.un$modelStruct)
```

```
## Correlation Structure: General
##  Formula: ~as.numeric(timefact) | id
##  Parameter estimate(s):
##  Correlation:
##   1     2     3     4     5     6
## 2 0.863
## 3 0.846 0.888
## 4 0.838 0.833 0.833
## 5 0.855 0.862 0.890 0.886
## 6 0.839 0.874 0.868 0.876 0.932
## 7 0.831 0.824 0.834 0.840 0.858 0.893
## Variance function:
##  Structure: Different standard deviations per stratum
##  Formula: ~1 | time
##  Parameter estimates:
##         0         3         6         9        15        19        12
## 1.0000000 0.9779813 0.9509405 0.9550631 0.9679811 0.9198126 0.9387328
```

It is evident that all the pair-wise correlations are about 0.85. Also, all the variance inflation constants are close to one. This observation further validates that our original dicision to fit a equal variance compound symmetric model is indeed reasonable.

The covariance/correlation matrices for one subject are shown below.

```
# Covariance matrix
Sigma <- getVarCov(gls.un, individual = 1)
Sigma
```

```
## Marginal variance covariance matrix
##         [,1]    [,2]    [,3]    [,4]    [,5]    [,6]
## [1,] 0.35514 0.29967 0.28564 0.28418 0.28842 0.27141
## [2,] 0.29967 0.33968 0.29323 0.27629 0.29385 0.26318
## [3,] 0.28564 0.29323 0.32115 0.26878 0.28368 0.25897
## [4,] 0.28418 0.27629 0.26878 0.32394 0.28769 0.26197
## [5,] 0.28842 0.29385 0.28368 0.28769 0.33277 0.28230
## [6,] 0.27141 0.26318 0.25897 0.26197 0.28230 0.30047
##   Standard Deviations: 0.59594 0.58282 0.5667 0.56916 0.57686 0.54815
```

```
# Correlation matrix
cov2cor(Sigma)
```

```
## Marginal variance covariance matrix
##         [,1]    [,2]    [,3]    [,4]    [,5]    [,6]
## [1,] 1.00000 0.86281 0.84580 0.83785 0.83899 0.83085
## [2,] 0.86281 1.00000 0.88781 0.83291 0.87404 0.82380
## [3,] 0.84580 0.88781 1.00000 0.83332 0.86776 0.83366
## [4,] 0.83785 0.83291 0.83332 1.00000 0.87623 0.83968
## [5,] 0.83899 0.87404 0.86776 0.87623 1.00000 0.89277
## [6,] 0.83085 0.82380 0.83366 0.83968 0.89277 1.00000
##   Standard Deviations: 1 1 1 1 1 1
```

Inference of the regression parameters

In this section we discuss how to make inferences about β. Specifically we consider the construction of **confidence intervals** and **tests of hypotheses**. To this end we use the ML estimator of β and its estimated covariance matrix:

$$\widehat{cov}(\widehat{\beta}) = \left\{ \sum_{i=1}^{N} (X_i^T \widehat{\Sigma}_i^{-1} X_i) \right\}^{-1}, \qquad \widehat{\Sigma}_i = \Sigma_i(\widehat{\omega}),$$

where $\widehat{\omega}$ is obtained either by ML or by REML.

Confidence intervals

Using the result shown above, we can construct approximate confidence intervals for a single component of β, say β_ℓ:

$$95\% \text{ CI for } \beta_\ell : \qquad \widehat{\beta}_\ell \pm 1.96\sqrt{\widehat{var}(\widehat{\beta}_\ell)}.$$

Essentially we used the ℓ-th element of the diagonal of the estimated covariance of $\widehat{\beta}$, $\widehat{cov}(\widehat{\beta})$, and the multivariate normal distribution of the estimator $\widehat{\beta}$. This confidence interval can still be used if the data are not normally distributed, but the number of units n is large.

In R, we can use the `intervals()` function in `nlme` package.

```
intervals(gls.fit.exch, which = "coef")
```

```
## Approximate 95% confidence intervals
## 
##  Coefficients:
##                           lower         est.        upper
## (Intercept)          3.31521241  3.507677331  3.700142253
## time                -0.03901126 -0.033851544 -0.028691824
## smokercurrent       -0.49332241 -0.272676037 -0.052029665
## time:smokercurrent -0.01049537 -0.004570281  0.001354812
## attr(,"label")
## [1] "Coefficients:"
```

Hypothesis tests

Assume it is of interest to test $H_0 : \beta_\ell = 0$ versus the alternative $H_1 : \beta_\ell \neq 0$. One can use the **Wald test** statistic:

$$Z = \frac{\widehat{\beta}_\ell - 0}{\sqrt{\widehat{var}(\widehat{\beta}_\ell)}}.$$

We can simply use the gls output as follows.

```
summary(gls.fit.exch)$tTable
```

```
##                            Value   Std.Error    t-value       p-value
## (Intercept)          3.507677331 0.098043236  35.776842 1.168904e-165
## time                -0.033851544 0.002628404 -12.879123  1.675519e-34
## smokercurrent       -0.272676037 0.112399101  -2.425963  1.549748e-02
## time:smokercurrent -0.004570281 0.003018291  -1.514195  1.303884e-01
```

More generally, it may be of interest to construct tests that certain linear combinations of the components of β are 0. For example, suppose we have $\beta = (\beta_1, \beta_2, \beta_3, \beta_4)^T$, and want to test a hypothesis of the form $H_0 : \beta_1 - \beta_2 = 0$ and so on.

Let L be a $1 \times p$ dimensional matrix of "weights", and assume that we want to test the null hypothesis $H_0 : L\beta = 0$ versus the alternative $H_1 : L\beta \neq 0$.

Statistical inference about $L\beta$ relies on the distribution of $L\widehat{\beta}$ which is $N[L\beta, L\ cov(\widehat{\beta})L^T)$, based on the distribution of $\widehat{\beta}$ for the case when the data is multivariate normal. Here we discuss hypothesis testing; but the ideas can be applied to the construction of confidence intervals.

Wald test statistic for $L\beta$, where L is $1 \times p$-dimensional matrix:

$$Z = \frac{L\widehat{\beta} - 0}{\sqrt{L\widehat{cov}(\widehat{\beta})L^T}}; \qquad Z \sim N(0, 1).$$

Equivalently $W = Z^2$ has chi-square distribution with 1 degrees of freedom, χ^2_1:

$$W = (L\widehat{\beta} - 0)\{L\widehat{cov}(\widehat{\beta})L^T\}^{-1}(L\widehat{\beta} - 0)^T; \qquad W \sim \chi^2_1.$$

The advantage of the former test is that it readily generalizes to cases when L has more than one row, for instance when L is $r \times p$ dimensional matrix. In that case, the null distribution of W would be χ^2_r, and p-values will be calculated based on this distribution.

For the example we have discussed so far, the model is simple (linear in time), and as such there is limited opportunity to create contrasts involving multiple parameters. For just a demonstration, let us test the hypothesis that the slope of the mean trend does not change between current and former smokers, that is,

$$H_0 : \beta_4 = 0.$$

We will use the `multcomp` library. We first define the L matrix (each row is a linear combination).

```
library(multcomp)

# Estimated regression coefs
cf <- gls.fit.exch$coefficients

# number of parameters (regression coefs) in beta
p <- length(cf)

# L with proper dim names (for convenience)
L <- matrix(0, nrow = 1, ncol = p,
            dimnames = list("slope(current-former)", names(cf))
            )
L
```

```
##                         (Intercept) time smokercurrent time:smokercurrent
## slope(current-former)           0    0             0                  0
```

```
# Set the appropriate entries in L
L[1,"time:smokercurrent"] <- 1
L
```

```
##                         (Intercept) time smokercurrent time:smokercurrent
## slope(current-former)           0    0             0                  1
```

Having defined the L matrix, we now use the `glht` function to estimate the linear combination, and the call summary to test the hypothesis.

```
test <- glht(model = gls.fit.exch, linfct = L)
test
```

```
##
##   General Linear Hypotheses
##
## Linear Hypotheses:
##                              Estimate
## slope(current-former) == 0 -0.00457
```

```
summary(test)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: gls(model = form, data = smoking, correlation = corCompSymm(form = ~1 |
##     id))
##
## Linear Hypotheses:
##                              Estimate Std. Error z value Pr(>|z|)
## slope(current-former) == 0 -0.004570   0.003018  -1.514     0.13
## (Adjusted p values reported -- single-step method)
```

We can also obtain the confidence interval for our linear combinations by using the `confint` function.

```
confint(test)
```

```
##
##   Simultaneous Confidence Intervals
##
## Fit: gls(model = form, data = smoking, correlation = corCompSymm(form = ~1 |
##     id))
##
## Quantile = 1.96
## 95% family-wise confidence level
##
##
## Linear Hypotheses:
##                            Estimate  lwr       upr
## slope(current-former) == 0 -0.004570 -0.010486  0.001345
```

We can have multiple rows in L, and perform the analysis shown above similarly.

```
# L with proper dim names (for convenience)
L <- matrix(0, nrow = 2, ncol = p)
rownames(L) <- c("Icept(current-former)", "slope(current-former)")
colnames(L) <- names(cf)

# Set the appropriate entries in L
L[2,"time:smokercurrent"] <- 1
L[1,"smokercurrent"] <- 1
L
```

```
##                       (Intercept) time smokercurrent time:smokercurrent
## Icept(current-former)           0    0             1                  0
## slope(current-former)           0    0             0                  1
```

```
# Estimate the linear combinations
test <- glht(model = gls.fit.exch, linfct = L)
test
```

```
##
##   General Linear Hypotheses
##
## Linear Hypotheses:
##                            Estimate
## Icept(current-former) == 0 -0.27268
## slope(current-former) == 0 -0.00457
```

```
# Test
summary(test)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: gls(model = form, data = smoking, correlation = corCompSymm(form = ~1 |
##     id))
##
## Linear Hypotheses:
##                             Estimate Std. Error z value Pr(>|z|)
## Icept(current-former) == 0 -0.272676   0.112399  -2.426    0.030 *
## slope(current-former) == 0 -0.004570   0.003018  -1.514    0.239
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
# Intervals
confint(test)
```

```
##
##   Simultaneous Confidence Intervals
##
## Fit: gls(model = form, data = smoking, correlation = corCompSymm(form = ~1 |
##     id))
##
## Quantile = 2.2312
## 95% family-wise confidence level
##
##
## Linear Hypotheses:
##                            Estimate  lwr       upr
## Icept(current-former) == 0 -0.272676 -0.523459 -0.021893
## slope(current-former) == 0 -0.004570 -0.011305  0.002164
```

In this case (with multiple rows in L), we can test both the hypotheses together using an F-test, as follows:

```
anova(gls.fit.exch, L = L)
```

```
## Denom. DF: 767
##  F-test for linear combination(s)
##                       smokercurrent time:smokercurrent
## Icept(current-former)             1                  0
## slope(current-former)             0                  1
##   numDF F-value p-value
## 1     2 5.31403  0.0051
```

Likelihood ratio test

An alternative to Wald test statistics is the *likelihood ratio test (LRT)* statistic. The LRT for testing $H_0 : L\beta = 0$ versus the alternative $H_1 : L\beta \neq 0$ is obtained by comparing the maximized likelihood for 2 models

- one model that incorporates the constraint specified in the null hypothesis $L\beta = 0$ (this is called the **reduced model**); and

- one without constraint (this model is called **full model**).

Note that the two models are **nested** in the sense that the reduced model is a special case of the full model.

Thus when the constraint $L\beta = 0$ holds, the full model reduces to the reduced model. The maximized log-likelihood for the full model is denoted by $\widehat{\ell}_{full}$ and the maximized log-likelihood for the reduced model is denoted by $\widehat{\ell}_{red}$. The LRT is obtained as:

$$LRT = 2(\widehat{\ell}_{full} - \widehat{\ell}_{red});$$

when the null hypothesis is true, then the distribution of the LRT is χ^2 with df equal to the difference between the number of parameters in the full and the number of parameters in the reduced models.

> ***When testing two nested models in terms of mean regression parameters, do not use REML*** *(because the adjustment is affected by the structure of the systematic mean part). Wald tests are commonly employed when testing mean regression parameters.*

In view of the comment above, we need to refit out model using ML. We fit both models (fulland reduced) using ML below.

```
# Full model
form <- FEV1 ~ time * smoker
gls.full <- gls(model = form,
                  data = smoking,
                  correlation = corCompSymm(form= ~ 1 | id ),
                  method = "ML"
                )

# Reduced model (No smoker and smoker:time effcts)
form.red <- FEV1 ~ time
gls.red <- gls(model = form.red,
                  data = smoking,
                  correlation = corCompSymm(form= ~ 1 | id ),
                  method = "ML"
                )

# We now need to call anova()
anova(gls.full, gls.red)
```

```
##          Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## gls.full     1  6 295.4603 323.3465 -141.7302
## gls.red      2  4 301.9697 320.5605 -146.9848 1 vs 2 10.50938  0.0052
```

Notice that the test statistic can be computed manually using the `logLik()` function:

```
LRT <- 2*( logLik(gls.full) - logLik(gls.red) )
LRT
```

```
## 'log Lik.' 10.50938 (df=6)
```

Subsequently, the p-value is computed using the χ^2 CDF:

```
pv <- pchisq(LRT, df = 2, lower.tail = F)
pv
```

```
## 'log Lik.' 0.005222976 (df=6)
```

Here the degrees of freedom (`df`) is the difference in the number of parameters between the full model (4 + 2 = 6 parameters) and reduced model (2 + 2 = 4 parameters).

> ***We use REML when testing between two nested covariance models.*** *When testing between competing covariance models, Wald tests are NOT valid.*

Selection of various covariance models

The choices of models for the mean and the covariance are interdependent. Since the confidence intervals and tests of hypotheses for the mean regression parameters depend critically upon the correct specification of the correct model assumed for the covariance it is important to begin with specifying the covariance model.

Nevertheless the model for the covariance depends on the assumed model for the mean: the model for the covariance models the dependence between the residuals $\{Y_{ij} - \mu_{ij}(\beta)\}$ and $\{Y_{ij'} - \mu_{ij'}(\beta)\}$ for $j \neq j'$. Therefore the model for the covariance should be based on a “maximal” model for the mean. Intuitively any systematic part that is left out (due to misspecification of the mean model) will lead to certain amount of spurious covariance among the residuals and will induce spurious dependence of the covariance on the covariates.

In longitudinal models with balanced design for the time points and a very small number of covariates (e.g. group and time-points at which the repeated outcome is measured) it is possible to fit a saturated model - see fitting procedure used in the split-plot model framework. Such a saturated model would allow for arbitrary pattern for the mean response trajectory at every level of the covariates, and thus minimizes the impact of the misspecification of the mean model. Nevertheless determining the maximal model in general is difficult and should be made on subject matter grounds. However once a maximal mean model for the mean response had been fixed, the residual variance and covariance can be used to select an appropriate model for the covariance.

Likelihood ratio test (LRT)

One possible way to choose between two competing covariance models is by **comparing the maximized REML likelihood** for the corresponding covariance models and using a hypothesis testing framework. Specifically, consider the case where we compare two covariance models that are nested within one another (two covariance models are nested when the: “reduced” model is a special case of the “full” model). For example, the compound symmetric covariance model is a special case of the unstructured covariance model. The null hypothesis is

$$H_0 : \Sigma \text{ has compound symmetric} \qquad \text{vs} \qquad H_1 : \Sigma \text{ is unstructured.}$$

Notice that the **less complicated model (compound symmetric) is in H_0**.

1. The LRT is obtained by comparing the maximized REML likelihood for the reduced covariance model (compound symmetric) with the maximized REML likelihood for the full covariance model (unstructured).
2. Formally the test statistic is:

$$LRT = 2(\widehat{\ell}_{full} - \widehat{\ell}_{red}).$$

3. Because of the unbiased properties of the variance estimators obtained using the REML likelihood, the **REML likelihoods are typically used for this test**.
4. Under the null hypothesis, the **sampling distribution of LRT is chi-squared** with degrees of freedom equal to the difference between the number of covariance parameters in the full and the reduced models.
5. We reject H_0 is the observed value of LRT is larger that $\chi^2_{\alpha,\mathrm{df}}$; fail to reject H_0 otherwise.

Let us now look at the example we have been discussing so far. Recall, we that we have alredy fitted

1. compound symmetric structure with equal variance (results were saved in `gls.fit.exch`)
2. compound symmetric structure with unequal variance (results were saved in `gls.cs.uv`)

Let us perform LRT using REML likelihood.

Step 1: First check whether the model fits are done using REML or not. Otherwise, we need to update them using “ML” method.

```
gls.fit.exch$method
gls.cs.uv$method
```

```
## [1] "REML"
## [1] "REML"
```

We see that both models were fit using REML, and thus no adjustments are needed.

Step 2: Compute the LRT test statistic. Recall, here the “full” model is the fit with unstructured covariance (the more complicated model); the “reduced” model is the compound symmetric structure (the model with less parameters)

```
LRT <- 2*(logLik(gls.cs.uv) - logLik(gls.fit.exch))
LRT
```

```
## 'log Lik.' 4.451672 (df=12)
```

Step 3: Determine the degrees of freedom and compute p-value. In out example, the model with unstructured covariance has 12 parameters (4 regression coeficients + 8 covariance parameters). The null model with compound symmetric covariance has 6 parameters (4 regression coeficients + 2 covariance parameters). Thus the degrees of freedom is 12 - 6 = 6.

```
df <- 6
pvalue <- pchisq(LRT, df = df, lower.tail = F)
pvalue
```

```
## 'log Lik.' 0.6157945 (df=12)
```

The function `pchisq()` computes the CDF of a χ^2 distribution. The p-value seems to indicate that, at $\alpha = 0.05$, we will fail to reject H_0, that is, we conclude that compound symmetric structure with equal variance is enough.

We can perform the LRT test in R using the ANOVA command as well:

```
anova(gls.fit.exch, gls.cs.uv)
```

```
##              Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## gls.fit.exch     1  6 323.6031 351.4581 -155.8016                        
## gls.cs.uv        2 12 331.1515 386.8613 -153.5757 1 vs 2 4.451672  0.6158
```

The results are identical to our manual test results.

Remark: In general LRT is preferable for testing between competing nested models. One important limitation is when the null hypothesis includes testing of parameters that are on the boundary of their parameter space. For example when the null hypothesis comes down to $H_0 : \sigma^2 = 0$ (i.e. a variance parameter is equal to zero), where recall $\sigma^2 \geq 0$. Such situation is known in the literature by the name "**testing a null hypothesis that is on the boundary of the parameter space**". In this case, the usual asymptotics used to develop the null distribution of the LRT are no longer valid; in particular the **null distribution of the LRT is no longer chi-square**. We will discuss more about this later when we study linear mixed models.

Akaike's Information Criterion (AIC)

Often it is of interest to compare models that are not nested. One common method is using the Akaike's Information Criterion (AIC), which is also based on the maximized log-likelihood, but it includes a penalty for complexity of the covariance model assumed

$$AIC = -2\widehat{\ell}_{model} + 2c$$

where $\widehat{\ell}_{model}$ is the maximized or fitted (REML) log-likelihood using the assumed model and c is the number of parameters included in this model. **Among all the interested covariance models, the one with the smallest AIC is preferred.**

The basic idea behind the AIC is to strike a balance between the fit to the data and the number of parameters involved in the covariance model (if the competing models assume the same model for the mean trend).

Schwarz’s Bayesian Information Criterion (BIC)

Another information criterion for choosing among competing covariance models is Schwarz’s Bayesian Information Criterion} (BIC), which also uses the maximized log-likelihood and penalizes the complexity of the model (though in a different way). BIC is defined as

$$BIC = -2\widehat{\ell}_{model} + (\log N)c$$

where c is the number of parameters included in the model of interest, and N is the total number of observations in the data $N = \sum_{i=1}^{n} m_i$. **Among all the interested models, the one with the smallest BIC is preferred.**

The main idea of the BIC comes for Bayesian approach to model selection, which is based on the highest posterior probability (or largest Bayes factor); BIC tries to approximate this Bayesian criterion. Because BIC penalizes drastically the number of components in the model, it tends to select the most parsimonious (simplistic) model; because of this BIC is not among the most popular approaches to select covariance models.

Remark: AIC penalizes the number of model parameters less strongly than BIC. In small samples, the corrected AIC (cAIC = AIC with a greater penalty for extra parameters) has been found more successful than AIC/BIC.

We have the following results for our example:

```
AIC(gls.fit.exch, gls.cs.uv, gls.un)
```

```
##               df      AIC
## gls.fit.exch   6 323.6031
## gls.cs.uv     12 331.1515
## gls.un        32 330.0506
```

```
BIC(gls.fit.exch, gls.cs.uv, gls.un)
```

```
##               df      BIC
## gls.fit.exch   6 351.4581
## gls.cs.uv     12 386.8613
## gls.un        32 478.6102
```

We can see that the compound symmetry model with equal variance given the smallest AIC and BIC.

Remark: Inferences about β using the model-based covariance rely heavily on the correct specification of the covariance model. Any misspecification of the covariance model has negligible effects on the estimation of the mean regression parameters β, but it may have serious implications on the inference about these parameters (construction of confidence intervals and hypothesis test). Fortunately one can still make valid inferences even if there are concerns about the specification of the covariance model. In particular valid inferences can be made using the so-called **sandwich estimator of the $cov(\widehat{\beta})$**; the resulting standard error are robust to misspecification of the covariance model. The “sandwich” estimator of the $cov(\widehat{\beta})$ are more common for marginal models for discrete longitudinal observations, and we will study them in detail when we discuss this topic.

Sequential (Type I) and Marginal (Type III) ANOVA

Recall that we used the `anova` function to test for the effects of the regression parameters. The results might be different depending on which type of ANOVA we perform.

Sequantial (Type I) ANOVA

Consider the model we fit before:

```
gls.fit.exch <- gls(model = FEV1 ~ time + smoker + time:smoker,
                    data = smoking,
                    correlation = corCompSymm(form= ~ 1 | id ))
anova(gls.fit.exch)
```

```
## Denom. DF: 767
##             numDF  F-value p-value
## (Intercept)     1 4075.828  <.0001
## time            1  832.685  <.0001
## smoker          1    8.335  0.0040
## time:smoker     1    2.293  0.1304
```

The testing proceeds as follows:

1. The first line tests whether `Intercept` has any effect of not when there are **no other** variable in the model, that is, it compares a model with only an intercept with a model with just an error term.
2. The second line tests whether `time` has an effect while intercept is presnt in the model (but no other variables). Thus we compare a model with intercept and time versus a model with only intercept.
3. Similarly, third line tests for effect of `smoker` while intercept and time are present (but no interaction yet).
4. The fourth line tests the effect of interaction `time:smoker` while intercept, time and smoker are present in the model.

This sequential nature of testing implies that the test results may change depending on the order in which the variables enter the model. For example, consider the following model with the same four effects, but `time` comes after `smoker`.

```
gls.fit.exch.2 <- gls(model = FEV1 ~ smoker + time + time:smoker,
                    data = smoking,
                    correlation = corCompSymm(form= ~ 1 | id ))
anova(gls.fit.exch.2)
```

```
## Denom. DF: 767 
##             numDF  F-value p-value
## (Intercept)     1 4075.828  <.0001
## smoker          1    6.904  0.0088
## time            1  834.116  <.0001
## smoker:time     1    2.293  0.1304
```

Notice that, since the order of `smoker` and `time` changed, the p-value of `smoker` changed. Also, since the interaction term `smoker:time` enters the model as the last term, its p-value remains the same.

Marginal (Type III) ANOVA

Marginal ANOVA tests the effect of a variable while accounting for all other remain variables in the model. Thus marginal testing of smoker will include intercept, time and intercation term in the model. Thus the order of the variables in the fitting process does not change testing results.

```
anova(gls.fit.exch, type = "marginal")
```

```
## Denom. DF: 767 
##             numDF   F-value p-value
## (Intercept)     1 1279.9824  <.0001
## time            1  165.8718  <.0001
## smoker          1    5.8853  0.0155
## time:smoker     1    2.2928  0.1304
```

```
anova(gls.fit.exch.2, type = "marginal")
```

```
## Denom. DF: 767 
##             numDF   F-value p-value
## (Intercept)     1 1279.9824  <.0001
## smoker          1    5.8853  0.0155
## time            1  165.8718  <.0001
## smoker:time     1    2.2928  0.1304
```

Notice the order of the variables are different, but the results are identical.

Small sample inference

The inferential procedures (CI and p-value) we discussed so far are developed using asymptotic (large sample) theory. These results may not be accurate when sample size (n) is small. Specifically, the large sample results can be used when $n - p$ is large. Otherwise, we might need to use resampling techniques.

One major impact of having a small sample size is the calculation of degrees of freedom of the tests. A approximation used by `gls` can be unreliable when n is small. To this end, we can use **Kenward-Roger approximation**.

Kenward and Roger (1997) developed an approximation that modify the usual F (or T) test statistic such that

- $cov(\widehat{\beta})$ is estimated using a small sample approximation
- The statistic F is scaled appropoately to account for small sample size

- The *corrected* error degrees of freedom is calculated accordingly.

The last two steps are done so that the expected value of the test statistic match approximately to that of an F distribution.

The package `lavaSearch2` implements an adaptation of Kenward-Roger approximation for ML based testing (for regression coefficients). We need to refit our model using ML, and use a new function `summary2()`.

```
library(lavaSearch2)
gls.fit.ML <- gls(model = FEV1 ~ time + smoker + time:smoker,
                    data = smoking,
                    correlation = corCompSymm(form= ~ 1 | id ), method = "ML")
summary2(gls.fit.ML)
```

```
## Generalized least squares fit by maximum likelihood
##   Model: FEV1 ~ time + smoker + time:smoker
##   Data: smoking
##        AIC      BIC    logLik
##   295.4603 323.3465 -141.7302
##
## Correlation Structure: Compound symmetry
##  Formula: ~1 | id
##  Parameter estimate(s):
##       Rho
## 0.8580072
##
## Coefficients:
##                         Value  Std.Error   t-value p-value       df
## (Intercept)          3.507698 0.09806414  35.76943  0.0000 150.2038
## time                -0.033853 0.00264242 -12.81153  0.0000 651.2183
## smokercurrent       -0.272687 0.11242292  -2.42554  0.0165 149.6335
## time:smokercurrent -0.004568 0.00303420  -1.50553  0.1327 651.0930
##
##  Correlation:
##                     (Intr) time   smkrcr
## time                -0.252
## smokercurrent       -0.872  0.220
## time:smokercurrent   0.219 -0.871 -0.247
##
## Standardized residuals:
##         Min          Q1         Med          Q3         Max
## -2.99688542 -0.65556536  0.01900769  0.69212175  3.17079532
##
## Residual standard error: 0.5672213
## Degrees of freedom: 771 total; 767 residual
```

Notice the extra `df` column in the t-test table. Since we have a fairly large sample size, the corrected tests do not differ too much fom our original results.

We can also perform an F-test using the function `compare2()`.

```
compare2(gls.fit.ML, par = c("time:smokercurrent = 0"))
```

```
##
##  - Wald test -
##
##  Null Hypothesis:
##  [time:smokercurrent] = 0
##
## data:
## F-statistic = 2.2666, df1 = 1, df2 = 651.09, p-value = 0.1327
## sample estimates:
##                              Estimate   Std.Err      df        2.5%
## [time:smokercurrent] = 0 -0.004568065 0.0030342 651.093 -0.01052606
##                                97.5%
## [time:smokercurrent] = 0 0.001389933
```

Notice that the argument `par = c("time:smokercurrent = 0")` directly specifies which parameter we want to test. We can test multiple parameters (and contrasts) together, as shown below.

```
compare2(gls.fit.ML, par = c("smokercurrent=0", "time:smokercurrent = 0"))
```

```
##
##  - Wald test -
##
##  Null Hypothesis:
##  [smokercurrent] = 0
##  [time:smokercurrent] = 0
##
## data:
## F-statistic = 5.2973, df1 = 2, df2 = 247.12, p-value = 0.005589
## sample estimates:
##                              Estimate   Std.Err       df        2.5%
## [smokercurrent] = 0      -0.272686554 0.1124229 149.6335 -0.49482802
## [time:smokercurrent] = 0 -0.004568065 0.0030342 651.0930 -0.01052606
##                                 97.5%
## [smokercurrent] = 0      -0.050545087
## [time:smokercurrent] = 0  0.001389933
```

For more such examples, see `?compare2`.

Final Remarks: main features and limitations

When confronted with a real data application an important step is the selection of the appropriate covariance model. Such covariance structure incorporates both sources of variation (among-units and between-units).

Useful ideas in the selection of the covariance model are:

1. Informal graphical/numerical summaries and other techniques may be used on a preliminary fit using OLS estimates of the regression parameters
2. AIC and BIC criteria may be used, but a dose of subjectivity is also involved
3. If no model is truly appropriate, that is alright too. The models used in the next chapter offer an alternative approach.

Important features of the regression approach:

1. The regression approach gives the analyst much flexibility in representing the form of the mean of the response. The mean can be modeled smoothly over time; the rate of change is the slope of this function. Also modeling of the mean in this fashion allows estimation of the mean at any time, not just the observed times.
2. The approach does not require a balanced time points design: the vectors of observations may have different lengths m_i. One important aspect we should be aware: if the unbalanced is due to missingness when data were intended to be collected at the same points. If the missingness is completely unrelated to the issues under study (e.g. sample of a certain subject at a certain time is mistakenly destroyed/ misplaced), then the analysis is ok. However if the missingness is related to issues under study (e.g. two treatments are compared and in one treatment a subject does not show up because they are too ill) then the missingness might contain information about the treatment; this analysis would not be valid.
3. The approach allows the analyst to consider an appropriate model for the covariance out of many choices.
4. Multiple groups/populations can be accounted for by appropriately manipulating the design matrix. Recall the explicit parameterization and the difference parameterizations.

Some limitations of this methodology:

1. The modeling of the covariance matrix aggregates the two sources of variation and does not allow the analysts to understand the two sources separately
2. The main focus is modeling of the mean trajectories over time; the reconstruction of the individual trajectories is not considered. Characterizing the subject trajectories may be of interest (the current framework does not allow such study.)

Main page: **ST 437/537: Applied Multivariate and Longitudinal Data Analysis (https://maityst537.wordpress.ncsu.edu/)**