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, $\boldsymbol{\Sigma}_i(\omega)$ is known. As usual, we use hat to denote parameter estimators (e.g., $\hat{\beta}$ 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,

$$\hat{\beta}_{MLE} = argmax_{\beta}\ell(\beta).$$

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

$$\hat{\beta}_{MLE} = argmin_{\beta}\sum_{i=1}^{n}(\boldsymbol{Y}_i - X_i\beta)^T\Sigma_i^{-1}(\boldsymbol{Y}_i - X_i\beta).$$

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

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

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

Properties of the $\hat{\beta}$

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

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

3. **Irrespective of the error distribution**, the GLS estimator $\hat{\beta}$ 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 $\hat{\beta}$ 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 $\hat{\omega}$ does not have a close form simple expression; numerical algorithms are used to obtain $\hat{\omega}$. When such an estimate is obtained then $\hat{\Sigma}_i = \Sigma_i(\hat{\omega})$ is substituted in the expression of $\hat{\beta}$.

When the sample size n is large, the resulting estimator $\hat{\beta}$ 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 $\hat{\omega}$ 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 $\{I - X(X^T X)^{-1} X^T\} = AA^T$ and $A^T A = I_{N-k}$; 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, $\hat{\omega}_{REML}$, 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)
```

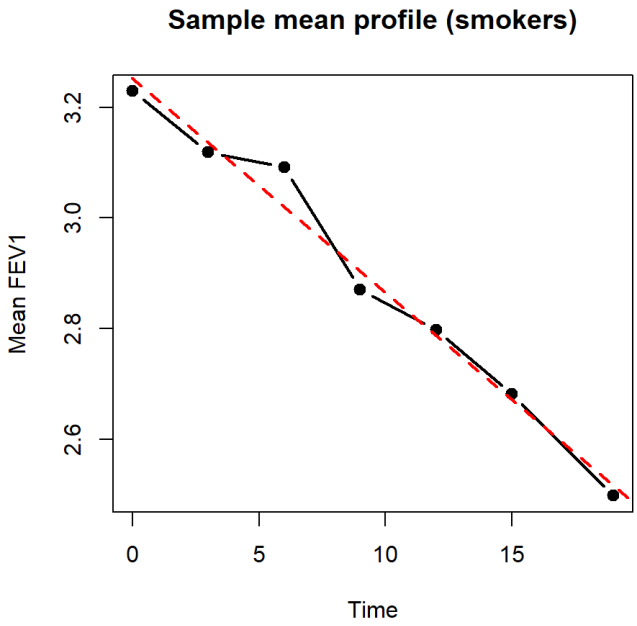
```
##
##   0   1
## 189 582
```

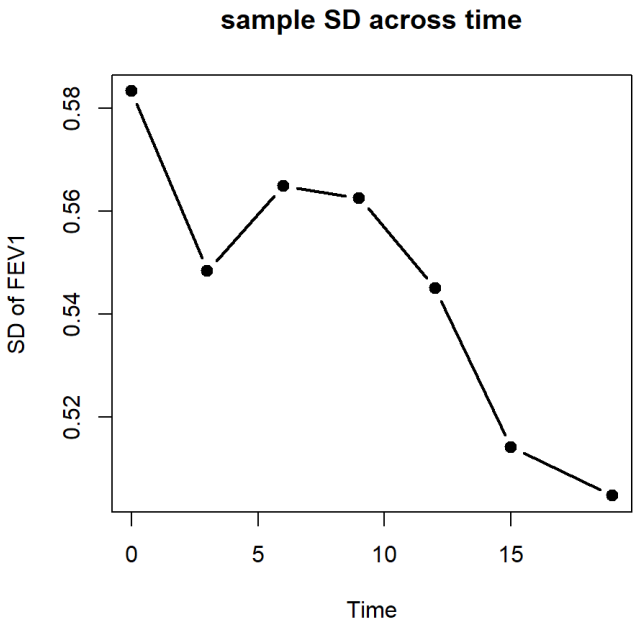
```
## Observations in each smoking group (0=former, 1=current)
## grouped time time points
table(smoking$smoker, smoking$time)
```

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

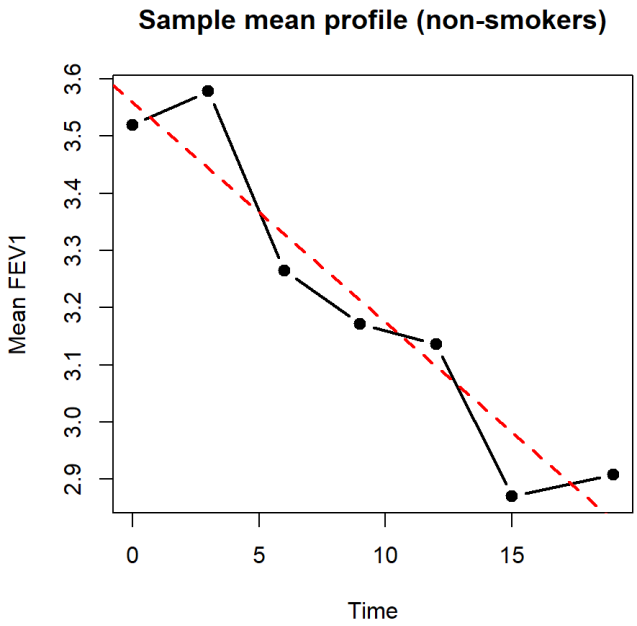
Let us use various visualization tools to assess the mean behavior over time, gain insight into the dependence over time.

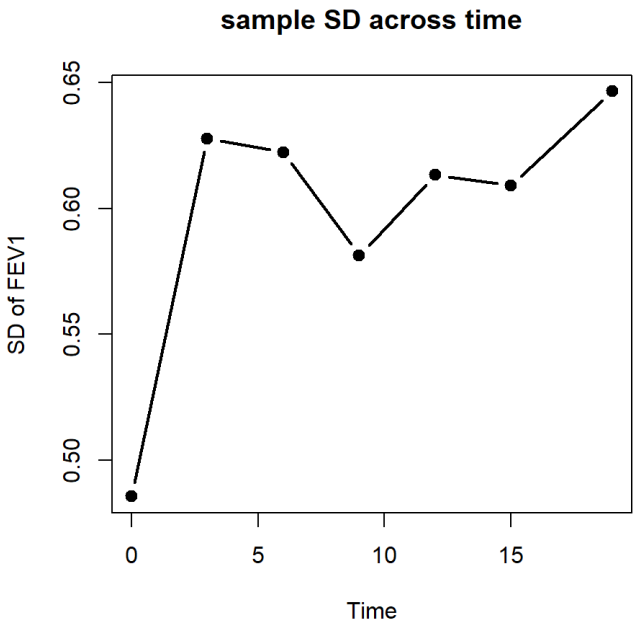
```
###
# Find the sample mean and sd profiles for
# each group (smoker/nonsmoker)
###

# 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 (smoke
rs)", 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", t
ype="b", lwd=2, pch=19)
```






```
# 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-s
mokers)", 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", t
ype="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 $\hat{\beta}$. We can obtain the **variance-covariance** matrix of the estimated regression coefficient $\hat{\beta}$, that is, $cov(\hat{\beta})$, 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, $\hat{\sigma} = 0.57$, 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 $\hat{\sigma}$ 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
```

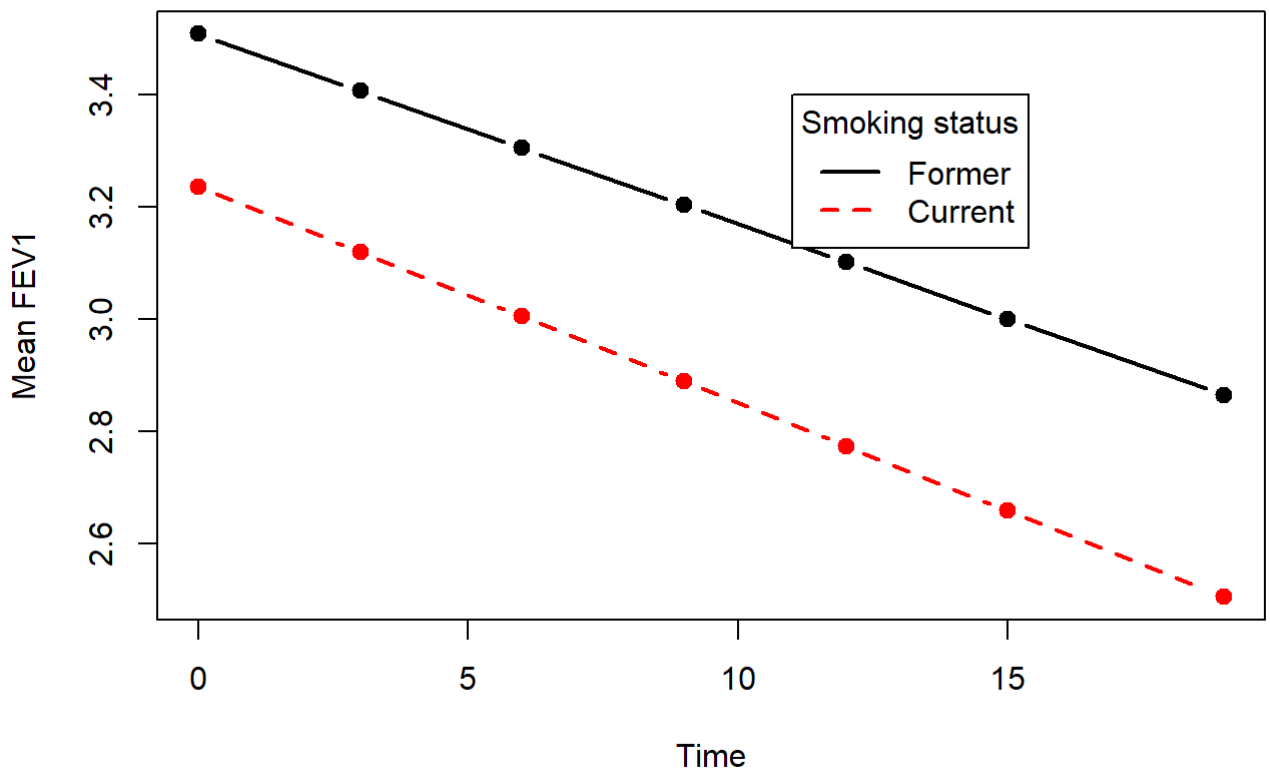
```
## Marginal variance covariance matrix
##         [,1]    [,2]    [,3]    [,4]    [,5]    [,6]
## [1,] 0.32622 0.28039 0.28039 0.28039 0.28039 0.28039
## [2,] 0.28039 0.32622 0.28039 0.28039 0.28039 0.28039
## [3,] 0.28039 0.28039 0.32622 0.28039 0.28039 0.28039
## [4,] 0.28039 0.28039 0.28039 0.32622 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.

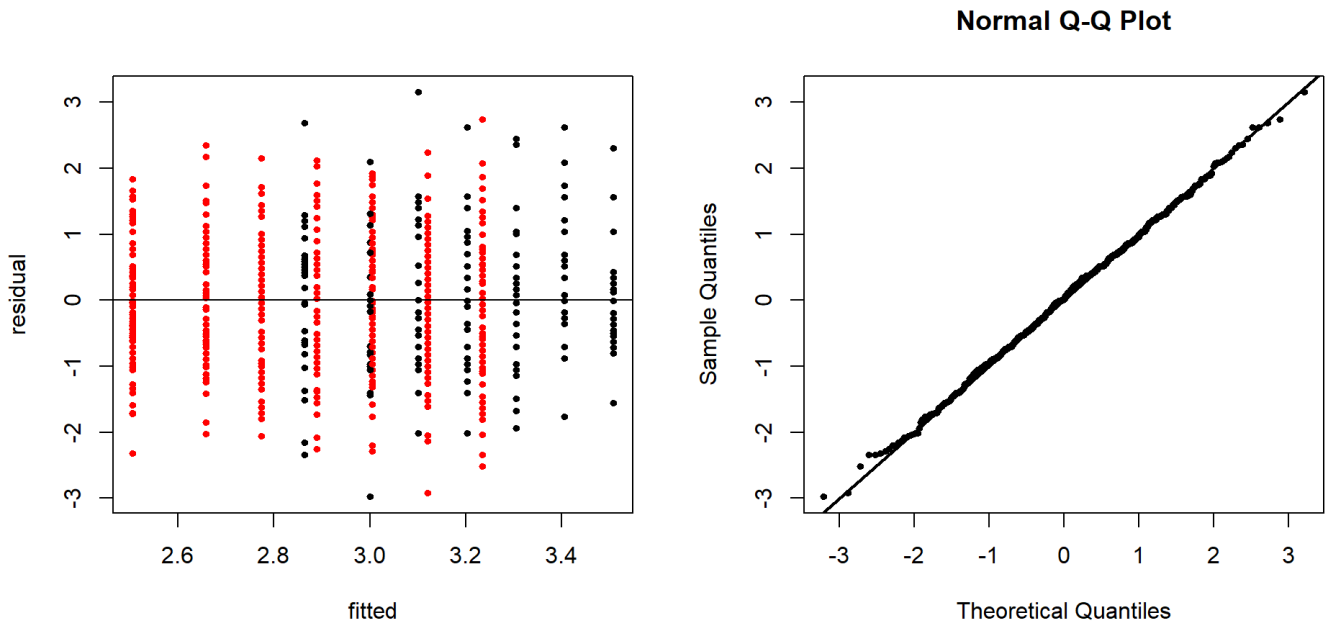
```
# Estimated coefficients
cf <- gls.fit.exch$coefficients

# 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 $\sigma_2^2 = \sigma_1^2 \times$ (inflation factor) = 0.3642067 $\times$ 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:

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

where $\hat{\omega}$ 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:

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

Essentially we used the ℓ-th element of the diagonal of the estimated covariance of $\hat{\beta}$, $\hat{cov}(\hat{\beta})$, and the multivariate normal distribution of the estimator $\hat{\beta}$. 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:

$$Z = \frac{\hat{\beta}_l - 0}{\sqrt{\hat{var}(\hat{\beta}_l)}}.$$

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 $\beta = (\beta_1, \beta_2, \beta_3, \beta_4)$, 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\$ which is $N(L\beta, Lcov(\hat{\beta})L^T)$, based on the distribution of $\hat{\beta}$ 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:

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

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

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

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 $\hat{\ell}_{full}$ and the maximized log-likelihood for the reduced model is denoted by $\hat{\ell}_{red}$. The LRT is obtained as:

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

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 of smoker:time effct)
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.*

To be updated

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