

Biostat/Stat 571 Exercise #4

Answer Key

February, 2010

Question 1

Consider the following regression model designs that are represented by the linear model:

$$E[Y_{ij}|\mathbf{X}_i] = \beta_0 + \beta_1 \cdot t_{ij} + \beta_2 \cdot x_{ij}, \quad i = 1, \dots, 200, j = 1, \dots, 10,$$

where there are a total of 200 clusters each of size $n_i = 10$. We consider two possible mean models:

- **Mean Model (A) :** A linear model with time, $t_{ij} = j$, and group indicator variable, x_{ij} , which is constant within a cluster. Assume that 100 clusters have $x_{ij} = 0$ and 100 clusters have $x_{ij} = 1$.
- **Mean Model (B) :** A linear model with time, $t_{ij} = j$, and group indicator variable, x_{ij} , which is subject-specific (non-constant) within a cluster. Assume that 100 clusters have $\mathbf{x}_i = (0,0,0,0,0,1,1,1,1,1)$, and 100 clusters have $\mathbf{x}_i = (1,1,1,1,1,0,0,0,0,0)$.

For each of the above mean models, we can write down the covariance matrix for the OLS estimates (assuming an independence correlation structure, $\text{COV}[\mathbf{Y}_i] = \mathcal{I}$) as well as the true covariance matrix (under the correct assumptions of the true correlation structure, $\text{COV}[\mathbf{Y}_i] = \Sigma$). If we let \mathbf{X} denote the design matrix for all 200 clusters, and assuming that σ is known, then

$$\begin{aligned} V^{\mathcal{I}}[\beta] &= \sigma^2 [\mathbf{X}^T \mathbf{X}]^{-1}, \\ V^{\Sigma}[\beta] &= [\mathbf{X}^T \mathbf{X}]^{-1} [\mathbf{X}^T \Sigma \mathbf{X}] [\mathbf{X}^T \mathbf{X}]^{-1}. \end{aligned}$$

Part a

Table 1 gives the standard errors reported by OLS and the correct standard errors when the data have a random intercepts covariance structure for the case when $(\sigma = 20, \tau = 5)$ and $(\sigma = 5, \tau = 20)$ for both Model A and Model B.

Table 1: Standard errors reported by OLS and the correct standard errors when the data have a random intercepts covariance structure.

Mean Model	(σ, τ)		β_0	β_1	β_2
A	$(20, 5)$	OLS	1.0973	0.1605	0.9220
		correct	1.1762	0.1557	1.1402
B	$(20, 5)$	OLS	1.0973	0.1605	0.9220
		correct	1.1218	0.1557	0.8944
A	$(5, 20)$	OLS	1.0973	0.1605	0.9220
		correct	2.0176	0.0389	2.8373
B	$(5, 20)$	OLS	1.0973	0.1605	0.9220
		correct	1.4390	0.0389	0.2236

Part b

Table 2 gives the standard errors reported by OLS and the correct standard errors when the data have a random intercepts and slopes covariance structure for the case when $\sigma = 1$ and $\mathbf{D} = \text{diag}(100, 2)$ and $\mathbf{D} = \text{diag}(50, 4)$ both Model A and Model B.

Table 2: Standard errors reported by OLS and the correct standard errors when the data have a random intercepts covariance structure.

Mean Model	(σ, \mathbf{D})		β_0	β_1	β_2
A	$(1, \text{diag}(100, 2))$	OLS	0.7102	0.1039	0.5967
		correct	1.1425	0.1003	1.7922
B	$(1, \text{diag}(100, 2))$	OLS	0.7102	0.1039	0.5967
		correct	0.7519	0.1003	0.5020
A	$(1, \text{diag}(50, 4))$	OLS	0.7621	0.1115	0.6403
		correct	1.0525	0.1416	1.8499
B	$(1, \text{diag}(50, 4))$	OLS	0.7621	0.1115	0.6403
		correct	0.6147	0.1416	0.7085

Part c

Table 3 gives the standard errors reported by OLS and the correct standard errors when the data have autocorrelated errors for the case when $\sigma = 10$, $\tau = 20$ and $\rho = 0.9$ and $\rho = 0.5$ for both Model A and Model B.

Table 3: Standard errors reported by OLS and the correct standard errors when the data have autocorrelation errors.

Mean Model	(σ, τ, ρ)		β_0	β_1	β_2
A	$(10, 20, 0.9)$	OLS	1.1902	0.1741	1.0000
		correct	2.0571	0.2009	2.4538
B	$(10, 20, 0.9)$	OLS	1.1902	0.1741	1.0000
		correct	1.7391	0.2009	1.0914
A	$(10, 20, 0.5)$	OLS	1.1902	0.1741	1.0000
		correct	1.6472	0.2280	1.5101
B	$(10, 20, 0.5)$	OLS	1.1902	0.1741	1.0000
		correct	1.6009	0.2280	1.2960

Part d

Throughout each of the tables in parts (a) - (c), we find that the impact of correlation on the standard error estimates depends on both the form of the correlation and the mean model. In particular, the only difference between the two mean models is the x_{ij} covariate, where in mean model A it is a cluster level covariate (i.e. constant within a cluster) while in mean model B it is an individual level covariate (i.e. varies within a cluster).

For the random intercepts correlation structure we find that the OLS standard error estimates for $\hat{\beta}_1$ are too big. This is especially so for the case where $(\sigma, \tau) = (5, 20)$, which indicates high between subject variability (and relatively low within subject variability). This impact doesn't seem to depend on the type of the other variable (x_{ij}) in the model. For the other two correlation structures, however, we find that the OLS standard error estimates for $\hat{\beta}_1$ are too small, with no difference between the two mean models.

For $\hat{\beta}_2$, we can see that in general the OLS standard error estimates are too small with the impact being smaller for the model where x_{ij} is an individual level covariate. We do see, however, that for random intercepts, the OLS standard errors for the $\hat{\beta}_2$ are too big in model B. This is typical of what we would see in practice, where, in general, naive standard errors for an individual level covariate will be too large while naive standard errors for a cluster level covariate will be too small.

Question 2

Data has been generated from a model that assumes random intercepts and random slopes:

$$Y_{ij} = \beta_0 + \beta_1 t_{ij} + b_{0,i} + b_{1,i} t_{ij} + \epsilon_{ij}, \\ i = 1, \dots, 200 \quad j = 1, \dots, 10, \\ (b_{0i} \ b_{1i})^T \sim \mathcal{N}(0, \ \mathbf{D}), \\ \epsilon_{ij} \sim \mathcal{N}(0, \ \sigma^2),$$

where $t_{ij} = j$, $(\beta_0, \beta_1) = (100, -5)$, $\mathbf{D} = \begin{bmatrix} 100 & 0 \\ 0 & 2 \end{bmatrix}$, and $\sigma = 1$.

We consider the impact of missing data on ordinary least squares estimation, weighted least squares estimation and relative efficiency. Specifically, the three estimators considered are

- $\hat{\beta}(I)$: Ordinary least squares estimator (OLS), where we assume independence.
- $\hat{\beta}(W^1)$: Weighted least squares estimator (WLS¹), where we assume a random intercepts model with $\sigma = 1$ and $\tau = 10$.
- $\hat{\beta}(W^2)$: Weighted least squares estimator (WLS²), where we assume the true covariance structure Σ (i.e. $W^2 = \Sigma^{-1}$).

Part a

	OLS	WLS ¹	WLS ²
Intercept	99.84	99.89	99.77
Slope	-4.89	-4.88	-4.86

- In the MCAR situation, as long as the mean is correctly specified, least squares estimation yields unbiased results.
- All three estimators produce values that are very close to the true intercept and slope.

Part b

Since the above estimates are all consistent for the true parameter values, we can examine the asymptotic relative efficiencies of $\hat{\beta}(I)$ and $\hat{\beta}(W^1)$ versus $\hat{\beta}(W^2)$.

- Here we see the loss of information that results from using either I or W^1 as weighting schemes.

	ARE.ols	ARE.WLS ¹
Intercept	0.870	0.962
Slope	0.784	0.945

- Assuming independence hurts estimation quite a lot (especially for the slope), while assuming random intercepts only results in a very modest loss.

Part c

	OLS	WLS ¹	WLS ²
Intercept	95.01	98.19	99.81
Slope	-2.97	-4.27	-4.87

- In the MAR situation, there is no guarantee that we will get unbiased estimates using weighted least squares.
- We can see that there is considerable bias in the OLS estimates (especially the slope), and that the WLS¹ estimates are still biased (but in this case they don't seem too unreasonable).
- The WLS² estimator is equivalent to the maximum likelihood estimator in this situation, and thus yields unbiased results even in the MAR situation.

Part d

	OLS	WLS ¹	WLS ²
Intercept	99.67	99.44	99.63
Slope	-4.68	-4.68	-4.77

- In the IM situation, there is no guarantee that any of the three methods (even ML) will result in unbiased estimation.
- Even though we found reasonable results here, there is no theoretical justification for their validity.

Question 3

Please see the posted book chapter on Longitudinal Data Analysis.

R Code

```
#####
## Question 1
#####

#####
##### Linear Model: #####
##### E[Y_{ij} | X_i] = \beta_0 + \beta_1 t_{ij} + \beta_2 x_{ij} #####
#####

#
options( object.size = 10e07 )
#
##
##### Mean Model A
##### - 200 clusters, with n_i = 10
##### - t_{ij} = j , for j = 1, ..., 10
##### - 100 clusters have x_{ij} = 0 for all j within unit i
##### - 100 clusters have x_{ij} = 1 for all j within unit i
##
#
X.A <- cbind( rep(1, 200*10), rep(1:10, 200), c(rep(0, 100*10), rep(1, 100*10)) )
#
##
##### Mean Model B
##### - 200 clusters, with n_i = 10
##### - t_{ij} = j , for j = 1, ..., 10
##### - 100 clusters have x_{i} = (0,0,0,0,0,1,1,1,1,1)
##### - 100 clusters have x_{i} = (1,1,1,1,1,0,0,0,0,0)
##
#
X.B <- cbind( rep(1, 200*10), rep(1:10, 200), c(rep(c(0,0,0,0,0,1,1,1,1,1), 100),
rep(c(1,1,1,1,1,0,0,0,0,0), 100)) )
#
##
##### Function to compute OLS and correct standard errors, given a true variance-covariance matrix
Sigma, design matrix X and true value for sigma
##
#
GetSE <- function( X, Sigma, digits = 4 ) {
  sigmaSqd <- mean( diag(Sigma) )
  OLS.se <- sqrt( diag( sigmaSqd * solve(t(X) %*% X) ) )
  correct.se <- sqrt( diag( solve(t(X) %*% X) %*% (t(X) %*% Sigma %*% X) %*% solve(t(X) %*% X) ) )
  value <- round( rbind( OLS.se, correct.se, OLS.se / correct.se ), digits = digits )
  dimnames( value ) <- list( c("OLS", "correct", "ratio"), c("beta_0", "beta_1", "beta_2") )
  return( value )
}
```

```

#
##
##### Part (a) : Random Intercepts
##
#
GetSigmaRI <- function( sigma, tau ) {
  sub <- matrix( tau^2, nrow = 10, ncol = 10 ) + diag( sigma^2, 10 )
  value <- matrix( 0, nrow = 2000, ncol = 2000 )
  for( i in 0:199 ){
    value[((i*10) + 1:10), ((i*10) + 1:10)] <- sub
  }
  return( value )
}
#
GetSE( X.A, GetSigmaRI( 20, 5 ), 4 ); GetSE( X.A, GetSigmaRI( 5, 20 ), 4 )
GetSE( X.B, GetSigmaRI( 20, 5 ), 4 ); GetSE( X.B, GetSigmaRI( 5, 20 ), 4 )
#
##
##### Part (b) : Random Intercepts/Slopes
##
#
GetSigmaRIS <- function( sigma, D ) {
  sub <- diag( sigma^2, 10 )
  for( i in 1:10 ){
    for( j in 1:10 ){
      sub[i,j] <- sub[i,j] + ( t(c(1, i)) %*% D %*% c(1, j) )
    }
  }
  value <- matrix( 0, nrow = 2000, ncol = 2000 )
  for( i in 0:199 ){
    value[((i*10) + 1:10), ((i*10) + 1:10)] <- sub
  }
  return( value )
}
#
GetSE( X.A, GetSigmaRIS( 1, diag(c(100,2)) ), 4 ); GetSE( X.A, GetSigmaRIS( 1, diag(c(50,4)) ), 4 )
GetSE( X.B, GetSigmaRIS( 1, diag(c(100,2)) ), 4 ); GetSE( X.B, GetSigmaRIS( 1, diag(c(50,4)) ), 4 )
#
##
##### Part (c) : Serial Correlation
##
#
GetSigmaSC <- function( sigma, tau, rho ) {
  sub <- matrix( tau^2, nrow = 10, ncol = 10 ) + diag( sigma^2, 10 )
  for( i in 1:10 ){
    for( j in 1:10 ){
      sub[i,j] <- sub[i,j] * rho^(abs(i - j))
    }
  }
}

```

```

}

value <- matrix( 0, nrow = 2000, ncol = 2000 )
for( i in 0:199 ){
  value[((i*10) + 1:10), ((i*10) + 1:10)] <- sub
}
return( value )
}

GetSE( X.A, GetSigmaSC( 10, 20, 0.9 ), 4 ); GetSE( X.A, GetSigmaSC( 10, 20, 0.5 ), 4 )
GetSE( X.B, GetSigmaSC( 10, 20, 0.9 ), 4 ); GetSE( X.B, GetSigmaSC( 10, 20, 0.5 ), 4 )

#####
## Question 2
#####
#
library(mvtnorm)
#
##
##### Fixed scalars/vectors/matrices
##
#
n <- 10
m <- 200
beta <- c( 100, -5 )
D <- matrix( c(100, 0, 0, 2), nrow = 2 )
sigma <- 1
tau <- 10
t <- 1:10
MeanModel <- beta[1] + beta[2] * t
#
##
##### Generate random intercepts/slopes data and re-structure.
##
#
bi <- rmvnorm( m, mean = c(0, 0), sigma = D )          ## 200 sets of random effects
b0i <- rep( bi[,1], rep( 10, m ) )                   ## expand out the intercept random effects
b1i <- rep( bi[,2], rep( 10, m ) )                   ## expand out slope random effects
epsilon <- rnorm( (n*m), mean = 0, sd = sigma )       ## residual error
Y <- rep(MeanModel, m) + b0i + ( b1i * rep(t, m) ) + epsilon
#
matY <- matrix( Y, ncol = 10, byrow = T )
plot( t, matY[1,], xlab = "time, t_i", ylab = "Outcome, Y_{ij}", ylim = range(matY), type = "l" )
for( i in 2:m ) lines( t, matY[i,] )
#
stacked <- as.data.frame( cbind( rep( 1:m, rep(10,m) ), Y, rep( 1:10, m ) ) )
names( stacked ) <- c( "id", "y", "time" )
#
##
##### Construct the three variance matrixes (for 10 observations)

```

```

##  

#  

var.I <- diag( 10 )  

#  

var.W1 <- matrix( tau^2, nrow = 10, ncol = 10 ) + diag( rep(sigma^2, 10) )  

#  

Tmat <- cbind( rep(1,10), t )  

var.W2 <- Tmat %*% D %*% t(Tmat) + diag( rep(sigma, 10) )  

#  

##  

#####
##### Function to perform all three regressions given a set 'missing' which #####
##### contains the indexes for the missing data #####
#####
##  

#  

RunMissing <- function( missing )  

{  

  stacked.new <- stacked[ -missing, ]  

  ols.numer <- wls1.numer <- wls2.numer <- c( 0, 0 )  

  ols.denom <- wls1.denom <- wls2.denom <- matrix( 0, 2, 2 )  

  ols.cheese <- wls1.cheese <- matrix( 0, 2, 2 )  

  for( i in 1:200 )  

  {  

    observed <- stacked.new$time[stacked.new$id == i]  

    if( length(observed) > 0 )  

    {  

      Y <- stacked.new$y[stacked.new$id == i]  

      X <- cbind( rep( 1, length(observed) ), observed )  

      #  

      ## Weight matrices (i.e. inverse variance matrices), and true variance Sigma  

      #  

      Sigma <- var.W2[observed, observed]  

      W1 <- solve( var.W1[observed, observed] )  

      W2 <- solve( Sigma )  

      #  

      ols.numer <- ols.numer + (t(X) %*% Y)  

      ols.denom <- ols.denom + (t(X) %*% X)  

      ols.cheese <- ols.cheese + (t(X) %*% Sigma %*% X)  

      #  

      wls1.numer <- wls1.numer + (t(X) %*% W1 %*% Y)  

      wls1.denom <- wls1.denom + (t(X) %*% W1 %*% X)  

      wls1.cheese <- wls1.cheese + (t(X) %*% W1 %*% Sigma %*% W1 %*% X)  

      #  

      wls2.numer <- wls2.numer + (t(X) %*% W2 %*% Y)  

      wls2.denom <- wls2.denom + (t(X) %*% W2 %*% X)  

    }  

    if( i == (trunc(i/10)*10) ) cat( paste("Iteration:", i, "\n") )
}

```

```

}
#
## Parameter estimates
#
ols.beta.est <- solve( ols.denom ) %*% ols.numer
wls1.beta.est <- solve( wls1.denom ) %*% wls1.numer
wls2.beta.est <- solve( wls2.denom ) %*% wls2.numer
#
estimates <- cbind( ols.beta.est, wls1.beta.est, wls2.beta.est )
dimnames( estimates ) <- list( c("Intercept", "Slope"), c("OLS", "WLS1", "WLS2") )
#
## Variance-Covariance matrices for coefficient estimates
#
ols.var <- solve( ols.denom ) %*% ols.cheese %*% solve( ols.denom )
wls1.var <- solve( wls1.denom ) %*% wls1.cheese %*% solve( wls1.denom )
wls2.var <- solve( wls2.denom )
#
return( estimates, ols.var, wls1.var, wls2.var )
}

#
#
#######
##### Part (a) #####
#####
##
#
missing.a <- sample( 2000, 400 )
#
Part.a <- RunMissing( missing.a )
Part.a$estimates
#
#
#####
##### Part (b) #####
#####
##
#
ARE.ols <- diag(Part.a$wls2.var) / diag(Part.a$ols.var)
ARE.wls1 <- diag(Part.a$wls2.var) / diag(Part.a$wls1.var)
cbind( ARE.ols, ARE.wls1 )
#
#
#####
##### Part (c) #####
#####

```

```

##  

#  

missing.c <- NULL  

for( i in 0:199 )  

{  

  temp.y <- stacked$y[(i*10)+1:10]  

  first.missing <- F  

  for( j in 1:10 )  

  {  

    if( first.missing == T ) missing.c <- c( missing.c, (i*10) + j )  

    else if( temp.y[j] < 65 ) first.missing <- T  

  }  

}  

#  

Part.c <- RunMissing( missing.c )  

Part.c$estimates  

#  

#  

##  

#####
##### Part (d) #####
#####
##  

#  

##  

##### Create the CRM logits/probabilities for each subject; 200 x 10 matrix  

##  

#  

logits <- matrix( rep(-2.5 + (1:10/10), 200) - (0.1 * b1i), nrow = 200, byrow = T )  

p.dropout <- exp(logits) / (1 + exp(logits))  

#  

##  

##### Generate the dropout times  

##  

#  

dropout.time <- rep( 11, 200 )  

for( i in 1:200 ){  

  dropped <- F  

  for( j in 1:10 ){  

    if( dropped == F ){  

      if( rbinom(1, 1, p.dropout[i,j]) == 1 ){  

        dropped <- T  

        dropout.time[i] <- j  

      }  

    }  

  }  

}
#
```

```
##  
##### Generate the vector of 'missing' values  
##  
#  
missing.e <- NULL  
for( i in 1:200 ){  
    for( j in 1:10 ){  
        if( j >= dropout.time[i] ) missing.e <- c( missing.e, ((i-1)*10) + j )  
    }  
}  
#  
Part.e <- RunMissing( missing.e )  
Part.e$estimates  
#
```