Statistical Applications in Geographical Health Studies

José Miguel Martínez Martínez
PhD Student

Joan Benach de Rovira
Universitat Pompeu Fabra
PhD Advisor

Yutaka Yasui
University of Alberta
PhD Advisor

Josep Ginebra i Molins
Universitat Politècnica de Catalunya
PhD Tutor

Doctorate in Technical and Computer Applications of Statistics,
Operational Research and Optimization.
Universitat Politècnica de Catalunya
Barcelona, 8/05/2006

CHAPTER 7

Individual and aggregated health outcomes studies

"If exposure to a necessary agent is homogeneous within a population, then case/control and cohort methods will fail to detect it"

Geoffrey Rose

7.1 Introduction.

Studies based on individual data permit the assessment of the relationship between a health outcome (disease or death) and a series of characteristics (risk factors or confounding factors) measured at individual level. In other words for each individual we have information on their health outcome and their risk factors and confounding factors. However, as mentioned in chapter 1, when the individuals are hierarchically organized in groups, for example neighborhoods of a city, the studies using individual data must consider this hierarchical structrure in order to: 1) take account of the relationship of dependence of the individuals within each group to avoid obtaining incorrectly significant relation between exposure factors and disease when these do not exist and 2) incorporate the influence of contextual or group factors into the study of health[124].

In order to deal, for the purposed of health studies, with the organization of individuals into groups, statistical models incorporating random effects may be used[125]. Through random effects models it is possible to simultaneously combine the study of the influence of both individual and group factors on individual health outcomes,

controlling for the dependency of the individuals within each group[28,124]. Random effects models, also known as multilevel models in other contexts[28,124], are also appropriate when our objective is not simply to assess the relationship between exposure factors and health outcome, but also when it is proposed to study and explain variations in health outcome within and between groups. Furthermore, as mentioned in the first part of this thesis, in the study of small areas, random effects models are used to control for the variability in estimated health indicators.

However, analyses of individual disease-exposure data within a population are useful when exposure of interest varies sufficiently within the population. When the within-population variance of exposure is limited, however, power of the individual-data analysis within a population is reduced. As Geoffrey Rose pointed out "*If exposure to a necessary agent is homogeneous within a population, then case/control and cohort methods will fail to detect it*"[126]. Dietary and enviromental factors provide examples that can involved limited ranges within populations available for study but with a significant variability between population groups. In such situations, aggregated health data studies over different populations can be used. Specifically, aggregated-data analyses of disease data across populations proposed by Prentice and Sheppard[31,127,128], with a sample of individual exposure data from populations, can be powerful in estimating the exposure effect if between-population variation of exposure is large. Individual and aggregated-data analyses approaches are useful depending on where the exposure variation exists.

In the following section we describe the individual random effects model (IRM) and the aggregated random effects model (ARM) proposed by Prentice and Sheppard for obtaining relative rates of disease. The ARM will also be compared with the classical ecological random effects model (ERM). Finally, we will explain the process for estimating the IRM and ARM using the estimating equation approach[129,130,131].

7.2 Relative rate analysis of individual- and aggregated-data.

This section reviews the individual- and aggregated-data models, following Prentice and Sheppard's work[127,128].

7.2.1 The individual-data model.

Let p_{ki} denote the probability that the ith individual in the k-th population, with size n_k (k=1,…,K), develops a certain disease within a defined follow-up period. We consider a relative rate model:

$$p_{ki} = p_{k0} e^{z_{ki}^T \beta}$$

where z_{ki} is a vector of covariates, p_{k0} is a 'baseline' disease probability for the k-th population corresponding to $z_{ki} = 0$ and β is a parameter vector to be estimated. The random effects assumption gives:

$$p_{k0} = h_k e^{\gamma_0}$$

where e^{γ_0} denotes the expected baseline rate and h_k denotes the residual baseline rate or 'frailty' of the k-th population. We consider that h_k's are independent random effects with mean 1 and variance σ^2. Under the random effects assumption, the model can be written as

$$p_{ki} = h_k e^{x_{ki}^T \alpha}, \qquad (7.1)$$

where $x_{ki}^T = (1, z_{ki}^T)$ and $\alpha^T = (\gamma_0, \beta^T)$.

The model expressed in (7.1) can be estimated considering that we know the covariate information in all the individuals of the k-th cohort, i.e. in the n_k individuals. Such covariate information usually is not available and we can consider the model in terms of a covariate sample size m_k in the k-th cohort.

7.2.2 The aggregated-data model.

An aggregate-data model as defined by Prentice and Sheppard, can be induced from the random effects model for individual data by averaging $h_k e^{x_{ki}^T \alpha}$ over the n_k

individuals within k-th population, and considering the average disease probability $\overline{p}_k$ of the population among the n_k individuals:

$$\overline{p}_k = h_k\left(\sum_{i=1}^{n_k} e^{x_{ki}^T\alpha} / n_k\right)$$

Following Prentice and Sheppard's notation we can express the aggregated data model as

$$\overline{p}_k = h_k \varepsilon_{n_k}\{e^{x_k^T\alpha}\} \qquad (7.2)$$

where $\varepsilon_{n_k}\{a_k\} = n_k^{-1}(a_{k1} + ... + a_{kn_k})$ denotes the average of the argument over the n_k individuals in the k-th population.

The model expressions presented consider that we know the covariate information in all the individuals of the k-th cohort, i.e. in the n_k individuals. As we pointed out in the section 7.2.1, such covariate information is not usually available and we can express the models in terms of a covariate sample size m_k in the k-th cohort. In the same way, an aggregate-data model can be induced from the random effects model for individual data by averaging $h_k e^{x_{ki}^T\alpha}$ over the m_k individuals in the sample within k-th population, and considering the average disease probability $\overline{p}_k$ of the population among the n_k individuals:

$$\overline{p}_k = h_k \varepsilon_{m_k}\{e^{x_k^T\alpha}\} \qquad (7.3)$$

where $\varepsilon_{m_k}\{a_k\} = m_k^{-1}(a_{k1} + ... + a_{km_k})$ denotes the average of the argument over the m_k individuals in the k-th population. Note that the left-hand side of the equation is the average based on the aggregated data (i.e., n_k individuals), while the right-hand side is the average based on the individual data (i.e., m_k individuals). Under the assumption that the individual samples are random samples of a sufficient size from the population, the aggregated-data model holds.

7.3 Differences between aggregated data and ecological studies.

Aggregated data and ecological studies are different, even though both are group-level studies that use aggregate health outcomes[31,132]. In this section, we show the difference between the ARM in (7.2) and the ecological random effects model (ERM), commonly used in small areas studies.

An ecological random effects model can be induced from the individual data by averaging x_{ki} over the n_k individuals within k-th population, instead of averaging $e^{x_{ki}^T\alpha}$ as in the ARM, and considering the average disease probability $\overline{p}_k$ of the population among the n_k individuals again just as for the ARM:

$$\overline{p}_k = h_k e^{\overline{x}^T\alpha}$$

where $\overline{x}^T = \sum_{i=1}^{n_k} x_{ki}^T / n_k$. As has been shown, the ARM proposed by Prentice and Sheppard arises from the aggregation of that component of the IRM which contains the covariates and represents the relationship of these covariates with the health outcome, whereas ecological studies only consider averages of individual covariates or group variables. For more details on ecological studies the reader is referred to chapter 1 and 4.

7.4 Relative rate inference for individual and aggregated random effects models based on estimating equations.

We consider individual data for a random sample of m_k individuals on K populations with population sizes n_k's (k=1,…,K). We denote the disease outcome variable of the i-th individual in the k-th population as Y_{ki}. The variable Y_{ki} takes a value of one if the outcome of interest (disease/death) occurs on the individual within the defined study follow-up period, and zero otherwise. We denote $Y_k^I = (Y_{k1},...,Y_{km_k})^T$ and $\mu_k^I = (\mu_{k1},...,\mu_{km_k})^T$ with

$$\mu_{ki} = E(Y_{ki}) = E(E(Y_{ki} | h_k)) = e^{x_{ki}^T \alpha}.$$

Then the individual-data estimating equation for model (7.1) is[128,133]

$$\sum_{k=1}^{K} (D_k^I)^T (V_k^I)^{-1} (Y_k^I - \mu_k^I) = 0, \qquad (7.4)$$

where $D_k^I = \partial \mu_k^I / \partial \alpha^T$, $V_k^I = \Delta_k + \sigma^2 \mu_k^I (\mu_k^I)^T$ with $\Delta_k = \text{diag}[\mu_{ki}\{1-(1+\sigma^2)\mu_{ki}\}]$.

The inverse of the variance-covariance matrix $(V_k^I)^{-1}$ can be computed[128,134] by

$$(V_k^I)^{-1} = (\Delta_k)^{-1} - \sigma^2 (\Delta_k)^{-1} \mu_k^I (\mu_k^I)^T (\Delta_k)^{-1} \{1 + \sigma^2 (\mu_k^I)^T (\Delta_k)^{-1} \mu_k^I\}^{-1}.$$

We consider aggregated data on a disease or mortality outcome are available on the K populations corresponding to the total number of disease cases (note that the total number of disease cases is aggregated data that should be easy to obtain from governmental agencies) and the total number of individuals at risk n_k during the study period. For model (7.3), we define

$$\bar{Y}_k^A = (n_k)^{-1} (\sum_{i=1}^{n_k} Y_{ki}) \text{ and } \hat{\mu}_k^A = E(\bar{Y}_k^A) = E(E(\bar{Y}_k^A | h_k)) = \varepsilon_{m_k} \{e^{x_k^T \alpha}\},$$

Then the aggregated-data estimating equation[127,128] is

$$\sum_{k=1}^{K} (\hat{D}_k^A)^T (\hat{V}_k^A)^{-1} (\bar{Y}_k^A - \hat{\mu}_k^A) = 0, \qquad (7.5)$$

where $\hat{D}_k^A = \partial \hat{\mu}_k^A / \partial \alpha^T$ and $\hat{V}_k^A = \sigma^2 \{(\hat{\mu}_k^A)^2 - \hat{\phi}_k (n_k)^{-1}\} + (\hat{\mu}_k^A - \hat{\phi}_k)(n_k)^{-1}$ with $\hat{\phi}_k = \varepsilon_{m_k} \{e^{2x_k^T \alpha}\}$. Note that we don't observe the individual covariates of every individual in the population, except those in the sample. Prentice and Sheppard,

therefore, estimated the average values $\mu_k^A = \varepsilon_{n_k}\{e^{x_k^T\alpha}\}$ and $\phi_k = \varepsilon_{n_k}\{e^{2x_k^T\alpha}\}$ with the average values in the sample, $\hat{\mu}_k^A$ and $\hat{\phi}_k$, respectively.

In each model, statistical inference on α can generally be based on the asymptotic normality of $\hat{\alpha}$ whose variance can be estimated consistently by the robust-sandwich variance estimator[128,34,135]. For the IRM, we can compute the information matrix as

$$I_\alpha^I = \sum_{k=1}^{K} (D_k^I)^T (V_k^I)^{-1} D_k^I$$

and the robust-sandwich estimator by

$$(I_\alpha^I)^{-1} \left[\sum_{k=1}^{K} (D_k^I)^T (V_k^I)^{-1} (Y_k^I - \mu_k^I)(Y_k^I - \mu_k^I)^T (V_k^I)^{-1} D_k^I \right] (I_\alpha^I)^{-1}$$

with all quantities evaluated at $\hat{\alpha}^T$. On the other hand, the ARM has information matrix define as

$$I_\alpha^A = \sum_{k=1}^{K} (\hat{D}_k^A)^T (\hat{V}_k^A)^{-1} \hat{D}_k^A$$

and robust-sandwich estimator by

$$(I_\alpha^A)^{-1} \left[\sum_{k=1}^{K} (\hat{D}_k^A)^T (\hat{V}_k^A)^{-1} (\overline{Y}_k^A - \hat{\mu}_k^A)(\overline{Y}_k^A - \hat{\mu}_k^A)^T (\hat{V}_k^A)^{-1} \hat{D}_k^A \right] (I_\alpha^A)^{-1}$$

with all quantities evaluated at $\hat{\alpha}^T$.

The estimation procedure is completed by inserting a $K^{1/2}$-consistent estimator for σ^2. Such estimators are given by a moment estimators defined for the IRM and ARM[127,128],respectively:

$$(\hat{\sigma}^2)_I = \frac{1}{K}\sum_{k=1}^{K}\left(\left[\varepsilon_{m_k}\{Y_k\}(\varepsilon_{m_k}\{Y_k\}m_k - 2\hat{\mu}_k^A m_k - 1) + 2\varepsilon_{m_k}\{\mu_k Y_k\}\right]\left[(\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k\right]^{-1} + 1\right)$$

$$(\hat{\sigma}^2)_A = \frac{1}{K}\sum_{k=1}^{K}\left[(\bar{Y}_k^A - \hat{\mu}_k^A)^2 - (\hat{\mu}_k^A - \hat{\phi}_k)(n_k)^{-1}\right]\left[(\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k)^{-1}\right]^{-1}.$$

The estimating equations can be solved through the Newton-Raphson procedures (see R[105] program in appendix A.3).

CHAPTER 8

Geographical regression extension: an integrated analysis of individual and aggregated health outcomes

"Don't be a novelist , be a statistician, much more scope for the imagination".

Darell Huff and Mel Calman

8.1 Introduction.

In chapter 7 we considered individual- and aggregated-data analyses of disease-exposure on K populations. As we described, individual- and aggregated-data analyses approaches are useful depending on where the exposure variation exists. However, in epidemiological studies we usually consider two or more covariates (exposures and confounding variables) that can have different types of variations, i.e., we can have covariates with high within-population variability and others with high between-population variability. In these cases, an individual-data analysis approach can perform poorly on the estimation of the covariables with high between-population variability, while an aggregated-data analysis approach can perform poorly on the estimation of the covariables with high within-population variability. In addittion, if we have an exposure covariable and a confounding variable that have different within- and between-population variabilities, the individual-data analysis can perfom poorly in estimating the exposure effect even if the exposure is subject to a high within-population variability.

This is due to the influence of the confounding covariate, which is by definition related to the exposure of interest[50] and can have high between-population variability. Similarly, the aggregated-data analysis approach could perfom poorly in estimating exposure effects even if the exposure is subject to high between-population variability, due to the influence of the confounding variable with high within-population variability.

In this chapter, we consider a new analytical framework that is an integrated data approach based on combining the individual- and aggregated-data analyses, presented in chapter 7. This method uses an estimating equation approach following the originals papers of Prentice and Sheppard[127,128]. The proposed analysis utilizes strengths of both individual- and aggregated-health data analysis approaches in the estimation of the exposure effect of interest, depending on which of the exposure variations (within- vs. between-population) dominates. As we pointed out above, this approach can be useful in epidemiological studies where we include exposure and confounding variables that can have different source of within and between-population variability. For example, in the study of the aetiology of bladder cancer we can jointly include variables where the within-population variability is higher than the between-population variation, such as smoking status, and variables where the between-population variation can be higher than the within-population, such as chlorinated drinking water[136].

The utilization of both types of data has been proposed under the fully Bayesian framework by Jackson et al.[137]. Our proposal follows the same basic concept of Jackson et al., but applies it under the estimating equation approach that Prentice and Sheppard[127,128] proposed originally.

In Section 8.2, we explain the study design and data structure of the proposed analytical framework. Section 8.3, describe the combination of the individual- and aggregated-data random effects models, a "population-based estimating equation" (PBEE) approach. Section 8.4 describes a simulation study that illustrates advantages of the PBEE over individual- and aggregated-data analyses presented in chapter 7. Finally, Section 8.5 contains discussion.

8.2 Study design.

We consider a study design in which 1) aggregated data on a disease or mortality outcome are available on K populations with population sizes n_k's (k=1,…,K), and 2) individual data for a random sample of m_k individuals ($m_k \leq n_k$) from the k-th population are collected. We denote the disease outcome variable of the i-th individual in the k-th population as Y_{ki} and a vector of covariates associated as z_{ki}. The variable Y_{ki} takes a value of one if the outcome of interest (disease/death) occurs on the individual within the defined study follow-up period, and zero otherwise. In each population's aggregate data, we have the total number of disease cases $\sum_{i}^{n_k} Y_{ki}$ and the total number of individuals at risk n_k during the study period, and, possibly, a vector of population-level covariates Z_k. These aggregated data are often available and published periodically from governmental agencies. A diagram of the data structure is given in Figure 8.1.

Figure 8.1 *Diagram of the data structure.*

Population 1		**Population k**		**Population K**
Population-level data $[\sum_{i}^{n_1} Y_{1i}, n_1, Z_1]$	…	*Population-level data* $[\sum_{i}^{n_k} Y_{ki}, n_k, Z_k]$	…	*Population-level data* $[\sum_{i}^{n_K} Y_{Ki}, n_K, Z_K]$
Individual-level data $[(Y_{11}, z_{11}),..,(Y_{1m_1}, z_{1m_1})]$		*Individual-level data* $[(Y_{k1}, z_{k1}),..,(Y_{km_k}, z_{km_k})]$		*Individual-level data* $[(Y_{K1}, z_{K1}),..,(Y_{Km_K}, z_{Km_K})]$

8.3 Relative rate inference based on population-based estimating equations.

To utilize the entire data for parameter estimation under the study design of Figure 8.1, we propose to combine estimating equations for the individual- and aggregated-data analyses into one equation. Using $Y_k^I = (Y_{k1},...,Y_{km_k})^T$ and $\mu_k^I = (\mu_{k1},...,\mu_{km_k})^T$ with $\mu_{ki} = E(Y_{ki}) = E(E(Y_{ki}|h_k)) = e^{x_{ki}^T\alpha}$, the individual-data estimating equation is defined as (7.4.), i.e

$$\sum_{k=1}^{K}(D_k^I)^T(V_k^I)^{-1}(Y_k^I - \mu_k^I) = 0,$$

where $D_k^I = \partial\mu_k^I/\partial\alpha^T$, $V_k^I = \Delta_k + \sigma^2\mu_k^I(\mu_k^I)^T$ with $\Delta_k = \text{diag}[\mu_{ki}\{1-(1+\sigma^2)\mu_{ki}\}]$.

The inverse of the variance-covariance matrix $(V_k^I)^{-1}$ can be computed[128,134] by

$$(V_k^I)^{-1} = (\Delta_k)^{-1} - \sigma^2(\Delta_k)^{-1}\mu_k^I(\mu_k^I)^T(\Delta_k)^{-1}\{1+\sigma^2(\mu_k^I)^T(\Delta_k)^{-1}\mu_k^I\}^{-1}.$$

For the aggregated part we exclude the individual data from the aggregated data in each population, that is, we now define $\bar{Y}_k^A$ as

$$\bar{Y}_k^A = (n_k - m_k)^{-1}(\sum_{i=1}^{n_k} Y_{ki} - \sum_{i=1}^{m_k} Y_{ki}) \text{ and } \hat{\mu}_k^A = E(\bar{Y}_k^A) = E(E(\bar{Y}_k^A|h_k)) = \varepsilon_{m_k}\{e^{x_k^T\alpha}\}.$$

The aggregated-data estimating equation is

$$\sum_{k=1}^{K}(\hat{D}_k^A)^T(\hat{V}_k^A)^{-1}(\bar{Y}_k^A - \hat{\mu}_k^A) = 0,$$

where $\hat{D}_k^A = \partial\hat{\mu}_k^A/\partial\alpha^T$ and $\hat{V}_k^A = \sigma^2\{(\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k - m_k)^{-1}\} + (\hat{\mu}_k^A - \hat{\phi}_k)(n_k - m_k)^{-1}$ with $\hat{\phi}_k = \varepsilon_{m_k}\{e^{2x_k^T\alpha}\}$ [127,128] (see appendix A.4 for demonstration). Note that we don't observe the individual covariates of every individual in the population, except those in

the sample. We, therefore, estimate the average values $\mu_k^A = \varepsilon_{n_k - m_k}\{e^{x_k^T \alpha}\}$ and $\phi_k = \varepsilon_{n_k - m_k}\{e^{2x_k^T \alpha}\}$ with the average values in the sample, $\hat{\mu}_k^A$ and $\hat{\phi}_k$, respectively.

The two estimating equations above can be combined to utilize both the individual and aggregate components of the entire data:

$$\sum_{k=1}^{K} \begin{pmatrix} D_k^I \\ \hat{D}_k^A \end{pmatrix}^T \begin{pmatrix} V_k^I & 0 \\ 0 & \hat{V}_k^A \end{pmatrix}^{-1} \begin{pmatrix} Y_k^I - \mu_k^I \\ \bar{Y}_k^A - \hat{\mu}_k^A \end{pmatrix} = 0 .$$

Note that we are proposing a simple addition of the two estimating equations. The combined estimating equation is a slight deviation from the optimal linear estimating function[138] of the form,

$$(\partial E[Y] / \partial \alpha^T)^T (\mathrm{Var}[Y])^{-1} (Y - E[Y]) .$$

Specifically, $\begin{pmatrix} V_k^I & 0 \\ 0 & \hat{V}_k^A \end{pmatrix}$ is not the variance-covariance matrix of $\begin{pmatrix} Y_k^I - \mu_k^I \\ \bar{Y}_k^A - \hat{\mu}_k^A \end{pmatrix}$: $(Y_k^I - \mu_k^I)$ and $(\bar{Y}_k^A - \hat{\mu}_k^A)$ are correlated and $\hat{\mu}_k^A$ has a sampling variation that is unaccounted for in $\hat{V}_k^A$. As these second-order assumptions are difficult to verify, our proposal is to keep the "weights" of the combined estimating function to correspond to a simple sum of the two estimating equations, and use a robust-sandwich variance estimator[139] of $\hat{\alpha}$ that reflects empirical second-order characteristics of $\begin{pmatrix} Y_k^I - \mu_k^I \\ \bar{Y}_k^A - \hat{\mu}_k^A \end{pmatrix}$. This is in the spirit of Prentice and Sheppard[127,128] and Liang and Zeger[34,135] in their use of a robust-sandwich variance estimator of mean parameters.

Statistical inference on α can generally be based on the asymptotic normality of $\hat{\alpha}$ whose variance can be estimated consistently by the following robust-sandwich variance estimator:

$$(I_\alpha)^{-1}\left[\sum_{k=1}^{K}\begin{pmatrix} D_k^I \\ \hat{D}_k^A \end{pmatrix}^T \begin{pmatrix} V_k^I & 0 \\ 0 & \hat{V}_k^A \end{pmatrix}^{-1} \begin{pmatrix} Y_k^I - \mu_k^I \\ \overline{Y}_k^A - \hat{\mu}_k^A \end{pmatrix}\begin{pmatrix} Y_k^I - \mu_k^I \\ \overline{Y}_k^A - \hat{\mu}_k^A \end{pmatrix}^T \begin{pmatrix} V_k^I & 0 \\ 0 & \hat{V}_k^A \end{pmatrix}^{-1} \begin{pmatrix} D_k^I \\ \hat{D}_k^A \end{pmatrix}\right](I_\alpha)^{-1},$$

where $I_\alpha = \sum_{k=1}^{K}\begin{pmatrix} D_k^I \\ \hat{D}_k^A \end{pmatrix}^T \begin{pmatrix} V_k^I & 0 \\ 0 & \hat{V}_k^A \end{pmatrix}^{-1} \begin{pmatrix} D_k^I \\ \hat{D}_k^A \end{pmatrix}$ with all quantities evaluated at $\hat{\alpha}$. As Prentice and Sheppard[127,128], the estimation procedure is completed by inserting a $K^{1/2}$-consistent estimator for σ^2. Such estimators are given by a moment estimators defined for the individual and aggregated components of the estimating equation, respectively (see appendix A.4 for demonstration):

$$(\hat{\sigma}^2)_I = \frac{1}{K}\sum_{k=1}^{K}\left(\left[\varepsilon_{m_k}\{Y_k\}(\varepsilon_{m_k}\{Y_k\}m_k - 2\hat{\mu}_k^A m_k - 1) + 2\varepsilon_{m_k}\{\mu_k Y_k\}\right]\left[(\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k\right]^{-1} + 1\right)$$

$$(\hat{\sigma}^2)_A = \frac{1}{K}\sum_{k=1}^{K}\left[(\overline{Y}_k^A - \hat{\mu}_k^A)^2 - (\hat{\mu}_k^A - \hat{\phi}_k)(n_k - m_k)^{-1}\right]\left[(\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k - m_k)^{-1}\right]^{-1}.$$

We do not unify the two estimators of σ^2 in line with the idea of combining two estimating equations into one. The estimating equation can be solved through the Newton-Raphson procedures (See R program in appendix A.3).

8.4 Simulation design and efficiency comparison.

A simulation study was conducted to compare the inferential performance of three approaches (IRM, ARM, PBEE) described in chapters 7 and section 8.2. We considered four different sample-size scenarios depending on the number of populations, K, and the sample size in each population, m_k: (K,m_k) = (100,100), (100,50), (50,100), and (50,50). These different scenarios can take place in real small-area geographical health studies. The population size n_k in each population was fixed at 2,000. In each scenario, we considered the following simulation similar to that of Prentice and Sheppard[127,128]. Two covariates, denoted as z_{k1} and z_{k2}, were generated, where z_{k1} represents exposure of interest and z_{k2} can represent a confounding factor. For the i-th person in the k-th population, we generate individual covariate values (z_{k1i}, z_{k2i}) from a bivariate normal

distribution with population mean (Z_{k1}, Z_{k2}) and variance-covariance matrix $\begin{pmatrix} \sigma_w^2 & \theta_I\sigma_w \\ \theta_I\sigma_w & 1 \end{pmatrix}$, where the population mean (Z_{k1}, Z_{k2}) is also a bivariate normal random vector with mean (0, 0) and variance-covariance $\begin{pmatrix} 1 & \theta_P \\ \theta_p & 1 \end{pmatrix}$. To consider a range of exposure variance in within- and between-populations, the within-population variance σ_w^2 of exposure, were set at 0.25, 0.5, 1, 2, 4 or 16 but the between-population variance of exposure was fixed at 1. This covers a wide range of the within- vs. between-variance ratio of exposure from 0.25/1 where the between-population variance dominates the within-population variance, to 16/1 where the within-population variance dominates the between-population variance. We considered (θ_I, θ_P) = (0,0) for no confounding case (NCC) and (0.3, 0.3) for a simple confounding case (SCC) by z_{k2}. In the NCC and SCC cases, we changed the within- versus between-population ratio and considered that the additional covariate, that in the SCC is a confounding factor, had the same within- and between-population variance. We considered that (θ_I, θ_P) = (0.3, 0.3) were reasonable values to analyze the influence of the confounding factor. We also considered an extended confounding case (ECC) with (θ_I, θ_P) = (0.3,0.3), where the within- versus between-population variance ratio for the two covariates, z_{k1} and z_{k2}, interchangeable with respect to their roles as an exposure or as a confounding factor, were above and below one, respectively, i.e., z_{k1}'s within- and between-population variances are 0.85 and 3.4, respectively, and z_{k2}'s within- and between-population variances are 4 and 0.25, respectively. In this way, the two covariates have the same total variance, but z_{k1} has the variance ratio of 0.25 indicating that the between-population variance dominates this covariate's variability and z_{k2} has the variance ratio of 16 indicating that the within-population variance dominates.

With the NCC and SCC, we can study situations where an exposure covariate can have different within- and between-population variabilities, and, in addition, we can have, respectively, a no-confouding and a confounding covariate which have the same within- and between-population variance. In this way, we can evaluate if 1) the IRM perfoms well when the within-population variability is high for the exposure of interest

and poorly when the between-population variability is high, 2) the ARM perfoms well when the between-population variability is high and poorly when the within-population variability is high, and 3) if the PBEE approach perfoms well in both situations.

With the ECC, we can study the situation where we have an exposure covariate with different within- and between-population variabilities with respect to a confounding variable. In this way, we can evaluate if 1) the IRM perfoms poorly in estimating the exposure effect in spite of high within-population variability, due to the influence of the confounding factor which may not be estimated well because of its high between-population variability, 2) the ARM perfoms poorly in estimating the exposure effect in spite of high between-population variability, due to the influence of confounding variable which may not be estimated well because of its high within-population variability and 3) the PBEE approach perfoms well in the two situations described.

Population-specific frailties, h_k, were generated as independent realized values from a gamma distribution with mean 1 and variance 0.05. The disease event indicator, Y_{ki}, was generated as a Bernoulli random variable with probability $h_k \exp(\gamma_0 + \beta_1 z_{ki1} + \beta_2 z_{ki2})$ where $\gamma_0 = -3$, $\beta_1 = 0.2$, and $\beta_2 = 0.2$. The 0.2 value represents a relative risk value of aproximately 1.2 that reflects a positive association with the disease outcome for the exposure and confounding factors. A total of 1,000 simulation runs were carried out in each of the different parameter sets. The simulation was programmed in the free software R[105] (see appendix A.3 for a detailed description of the steps followed).

Tables 8.1 and 8.2 present the bias and coverage of the 95% confidence interval for the 1,000 simulation runs for NCC and SCC. These results were presented in all the four (K, m_k) combinations for the three approaches: individual random effects model (IRM), aggregated random effects model (ARM) and the PBEE approach. For NCC (Table 8.1), when the between-population variance is larger than the within-population variance, the ARM model generally presents lower bias than the IRM model, and when the within-population variance is larger, the IRM generally presents lower bias than the ARM model. The PBEE approach presents either the lowest value, or close to the

Table 8.1 *Bias and 95% confidence interval coverage for the individual random effects model (IRM), aggregated random effects model (ARM) and population-based estimating equation approach (PBEE) for the no confounding case (NCC), in the different within- and between -population variance ratios for the four scenarios (K, m_k) = (100,100), (100,50), (50,100), and (50,50).*

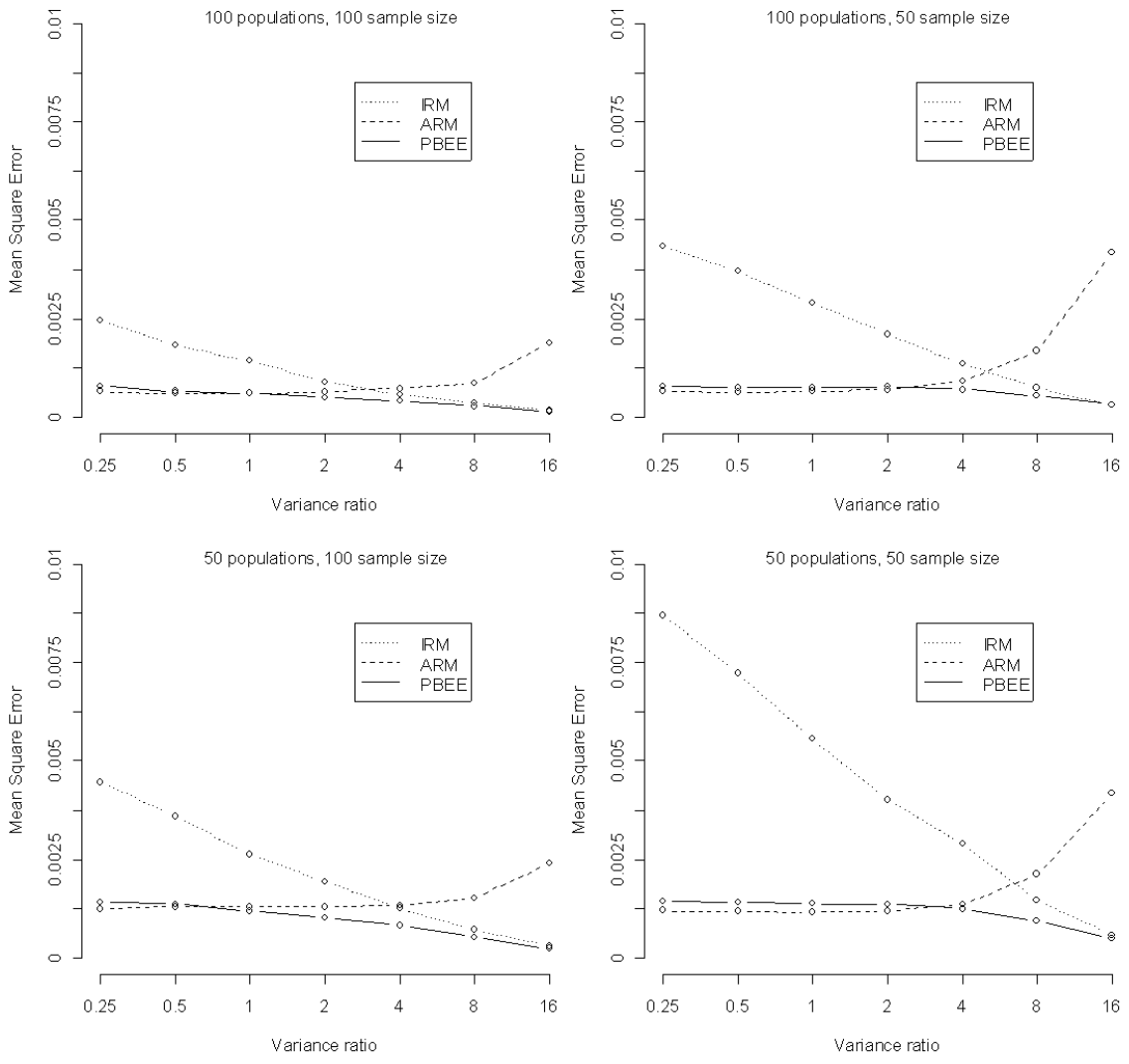
		Survey sample size in k-th population											
		100						**50**					
Number of Populations	**Variance Ratio**	**% Bias**			**95% Interval coverage**			**% Bias**			**95% Interval coverage**		
		IRM	ARM	**PBEE**	**IRM**	ARM	**PBEE**	**IRM**	ARM	**PBEE**	**IRM**	ARM	**PBEE**
100	**0.25**	0.20	0.00	0.07	92	93	93	1.99	-0.50	0.10	92	93	93
	0.5	-0.33	-0.28	-0.40	94	95	94	1.51	-1.58	-1.04	92	93	93
	1	0.74	-0.56	-0.08	92	94	93	1.86	-2.66	-1.51	92	92	92
	2	0.90	-2.22	-0.56	94	93	93	0.84	-4.84	-2.68	92	90	91
	4	0.99	-4.63	-0.59	94	91	93	1.18	-9.12	-3.26	92	87	90
	8	0.39	-8.98	-1.01	92	87	91	0.80	-17.25	-3.55	93	67	90
	16	-2.79	-19.16	-3.47	90	56	87	-2.45	-30.90	-5.17	91	13	83
50	**0.25**	0.85	0.32	0.60	91	93	92	-1.94	-0.09	-0.39	91	94	93
	0.5	-1.49	-1.13	-1.24	93	92	92	2.86	-0.89	-0.11	92	93	93
	1	-0.55	-1.56	-1.14	93	92	93	2.52	-1.92	-0.67	92	93	93
	2	0.62	-2.50	-0.92	93	92	92	2.54	-4.04	-1.56	93	94	92
	4	1.25	-4.71	-0.60	92	91	93	2.83	-8.22	-2.38	91	91	91
	8	1.14	-9.44	-0.45	92	88	92	2.21	-16.83	-2.63	92	81	90
	16	-2.32	-19.20	-3.04	91	74	89	-1.51	-29.21	-4.29	90	46	89

Table 8.2 *Bias and 95% confidence interval coverage for the individual random effects model (IRM), aggregated random effects model (ARM) and population-based estimating equation approach (PBEE) for the simple confounding case (SCC), in the different within- and between -population variance ratios for the four scenarios (K, m_k) = (100,100), (100,50), (50,100), and (50,50).*

		Survey sample size in k-th population											
		100						**50**					
Number of Populations	**Variance Ratio**	**% Bias**			**95% Interval coverage**			**% Bias**			**95% Interval coverage**		
		IRM	ARM	**PBEE**	**IRM**	ARM	**PBEE**	**IRM**	ARM	**PBEE**	**IRM**	ARM	**PBEE**
100	**0.25**	0.59	0.26	0.29	92	93	92	1.05	0.05	0.24	91	93	93
	0.5	-0.30	-0.84	-0.48	93	93	92	0.92	-1.40	-0.96	92	93	93
	1	-0.33	-1.43	-0.83	92	94	92	1.88	-2.51	-1.18	93	93	92
	2	-0.08	-2.87	-1.12	93	93	93	0.72	-5.16	-2.57	93	91	92
	4	0.68	-5.55	-0.77	92	90	94	1.04	-10.34	-3.30	92	85	90
	8	0.29	-11.44	-1.01	94	83	93	0.92	-19.90	-3.42	92	61	89
	16	-5.01	-22.89	-5.34	84	45	81	-4.03	-35.04	-6.42	86	8	78
50	**0.25**	0.71	0.28	0.48	91	93	92	-1.81	0.77	0.13	92	93	91
	0.5	1.34	-0.13	0.11	90	92	91	2.59	-0.85	0.20	90	93	92
	1	1.74	-0.83	0.19	90	92	91	1.60	-2.19	-0.75	91	92	91
	2	1.11	-2.38	-0.14	92	92	92	3.67	-4.48	-0.64	92	92	92
	4	1.51	-5.08	-0.09	92	91	92	3.10	-9.78	-1.76	92	88	91
	8	0.76	-10.91	-0.75	92	87	92	1.66	-19.56	-2.74	91	74	89
	16	-4.55	-22.54	-5.03	86	68	85	-3.37	-34.86	-5.67	87	31	82

lowest value, of bias in the three approaches, regardless of the within- and between-variances. The results for the confidence coverage interval are similar for the three models when the variance ratio is not large: they are all slightly lower than the 95% coverage. However, when the within-variance dominates, the ARM's coverage probability goes low due to the large bias in estimating the parameter: the PBEE approach is affected by the same problem but to a much lesser degree. Similar results and patterns are observed for SCC (Table 8.2).

Figure 8.2 *Mean square error for the individual random effects model (IRM), aggregated random effects model (ARM) and population-based estimating equation approach (PBEE) for the no confounding case (NCC), in the different within- and between -population variance ratios for the four scenarios (K, m_k) = (100,100), (100,50), (50,100), and (50,50).*



Figures 8.2 and 8.3 show mean squared errors of parameter estimation for NCC and SCC, respectively. In all four (K, m_k) combinations for the NCC and SCC, the mean squared error of the IRM decreases, and that of ARM increases as the ratio of within- to between-population exposure variance increases. However, the PBEE approach consistently provides the smallest (or close to the smallest) mean squared errors among the three methods in all the scenarios considered.

Figure 8.3 *Mean square error for the individual random effects model (IRM), aggregated random effects model (ARM) and population-based estimating equation approach (PBEE) for the simple confounding case (SCC), in the different within- and between -population variance ratios for the four scenarios (K, m_k) = (100,100), (100,50), (50,100), and (50,50).*

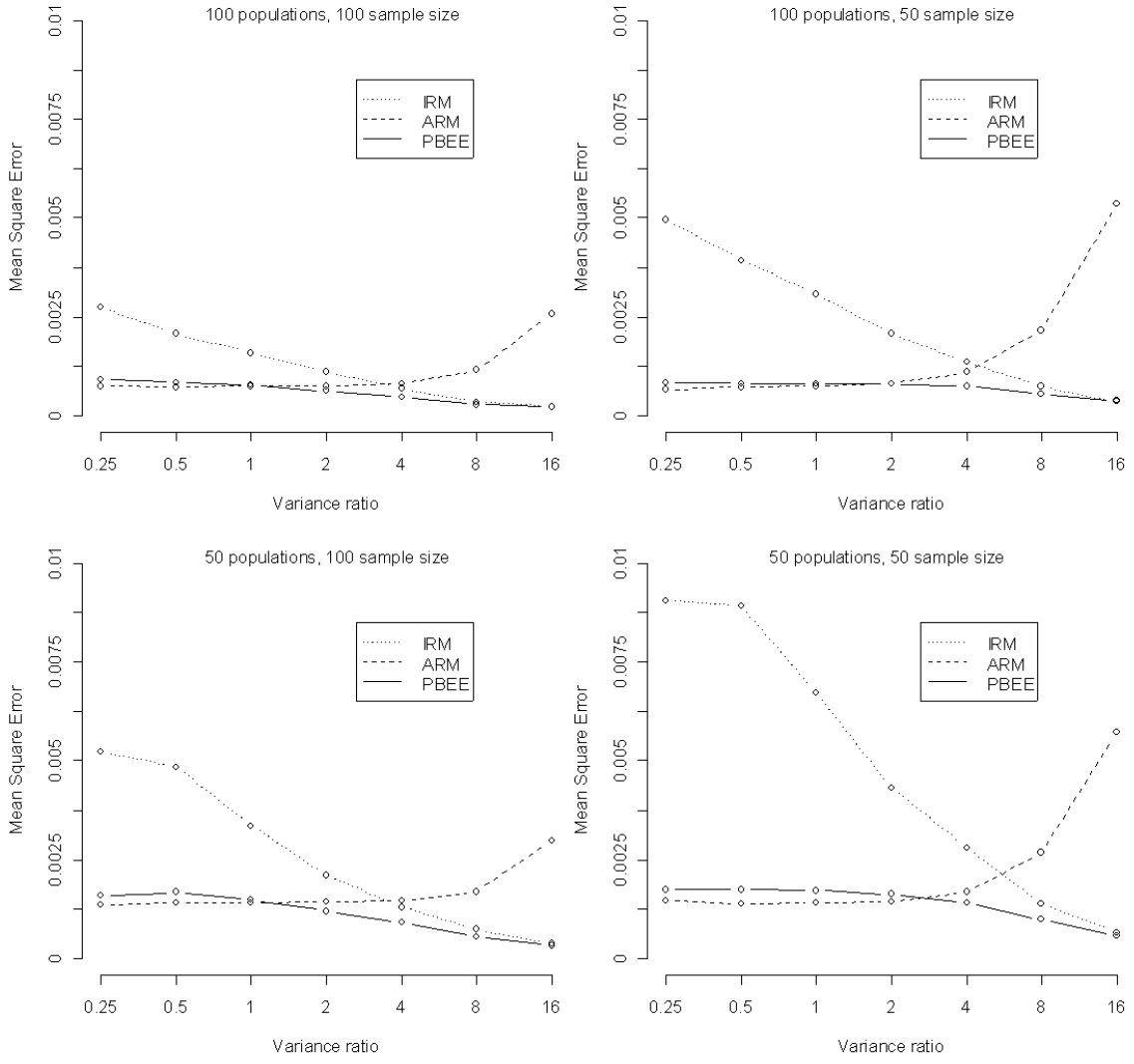


Table 8.3 compares the estimation performance for ECC. The bias is small in both β_1 and β_2 estimation by the three approaches except for ARM's β_2 estimation subject to the large within- vs. between-variation of z_{k2}. In terms of the mean squared error, the ARM results in smaller errors than the IRM for the covariate with 0.25 variance ratio and, while the IRM provides smaller errors than the ARM for the covariate with 16 variance ratio. However, the PBEE approach performs well for the two covariates variance ratios providing the best (or close to the best) results in all the different scenarios we considered.

Table 8.3 *Bias and mean square error (mse) for the individual random effects model (IRM), aggregated random effects model (ARM) and population-based estimating equation approach (PBEE) for the extended confounding case (ECC), in the different within- and between -population variance ratios for the four scenarios (K, m_k) = (100,100), (100,50), (50,100), and (50,50).*

		Survey sample size in k-th population											
		100						**50**					
Number of Populations	**Parameter**	**% Bias**			**Mse x 10^{-3}**			**% Bias**			**Mse x 10^{-3}**		
		IRM	ARM	**PBEE**	**IRM**	ARM	**PBEE**	**IRM**	ARM	**PBEE**	**IRM**	ARM	**PBEE**
100	β_1	0.03	0.86	-0.09	1.00	0.21	0.34	-0.15	1.50	0.16	1.56	0.22	0.36
	β_2	0.27	-18.04	-1.51	0.65	3.45	0.56	0.87	-31.85	-4.86	1.20	5.90	1.05
50	β_1	-0.04	0.83	0.12	2.01	0.43	0.72	-0.38	1.62	0.21	3.46	0.43	0.76
	β_2	0.72	-17.59	-1.43	1.29	5.82	1.10	2.15	-33.09	-4.22	2.75	8.14	2.14

8.5 Discussion.

This chapter, considered an "integrated" data analysis method, valuable for epidemiological purposes, that combines all the available information in the health outcomes and exposure variables at the different levels of data organization. This analysis performed with the proposed PBEE method presents a powerful analytical framework that takes into account both within- and between-population exposure variation and combines the strengths of both individual- and aggregated-data. It is applicable with the same study design and data structure (Figure 8.1) as the aggregated data analysis and without knowledge of which of the exposure variations (within- vs.

between-population) dominates. In addition, although we may have knowledge of which variations dominates on each variable, two or more exposure variables of interest do not necessarily have the same type of variations. As we have shown by ECC simulations, the PBEE approach will be particularly more advantageous over an individual- or aggregated-data analyses in such cases.

While the PBEE approach shares the same basic concept with the fully Bayesian approach of Jackson et al.[137] in that it utilizes both individual and aggregated data in epidemiological analyses, its estimating equation approach following Prentice and Sheppard[127,128] makes mean-parameter inference robust against misspecification of the second-order characteristics of disease outcomes. In addition, Jackson et al.[137] multiplied the likelihood of individual data and that of aggregated data, while we removed the samples in the individual data from the aggregated data. If the sample sizes of individuals (m_k's) is appreciable, this difference may be important. As a future research topic, a more detailed study can be pursued to evaluate the advantages and disadvantatges of the PBEE approach versus the fully Bayesian approach proposed by Jackson et al.

At least, some extensions or modifications are possible for the PBEE approach. We can study multiple imputation techniques to estimate the covariate information in the aggregated part of the PBEE approach[140] and adapt it epidemiological designs other than cohort studies, such as case-control data. In addition, it may be of interest to study the efficiency of the estimates obtained with the true variance-covariance matrix vs. with other different choices. Another important point is to apply the PBEE approach to an example with real data. Currently, we don't have real data yet because the process to obtain individual data is slow due to legal steps that we have to be followed due to confidentiality reasons, as we explained in Chapter 1.

The study design/data structure considered is currently not a common study design in epidemiology, but it offers certain advantages discussed in this chapter and in a previous article by Jackson et al.[137]. The gain in the parameters estimates can be achieved easily using aggregated mortality or disease data available from governmental agencies. For this and other results shown in this chapter, we recommended the

"integrated" design/data structure with the PBEE approach as a new analytical framework that can be considered in future epidemiological studies.

APPENDICES

A.1 Publications derived from the thesis.

A.1.1 Books.

1. Benach J, **Martínez JM**, Yasui Y, Borrell C, Pasarín MI, Español E, Benach N. *Atles de mortalitat en àrees petites a Catalunya (1984-1998) / Atlas de mortalidad en áreas pequeñas en Cataluña (1984-1998) / Atlas of mortality in small areas in Catalonia (1984-1998).* Barcelona: Mediterránea. Fundació Jaume Bofill and Universitat Pompeu Fabra; 2004 [Catalan-Spanish-English].

A.1.2 Book chapters.

1. Benach J, **Martínez JM**, Borrell C, Pasarín MI, Yasui Y, Buxó M. *Geographical health inequalities in small areas of Catalonia.* In: Borrell C, Benach J (editors). Evolution of health inequalities in Catalonia. Barcelona: Mediterránea. Fundació Bofill; 2005 [Catalan].

A.1.3 Scientific articles.

1. **Martínez JM**, Benach J, Yasui Y, Ginebra J, Clèries R, Ocaña R, Torné MM, Almansa J, Borrell C, Español E. *An innovative approach for comprehensiblely display spatial and temporal trends in small area atlases* (submitted).

2. Benach J, **Martínez JM**, Yasui Y, Borrell C, Pasarín MI, Español E, Benach N. *Investigating geographical and temporal patterns of mortality. The Atlas of mortality in small areas in Catalonia (1984-1998)* (submitted).

3. **Martínez JM**, Benach J, Ginebra J, Benavides FG, Yasui Y. *An Integrated Analysis of Individual and Aggregated Health Data: An Estimating Equation Approach* (submitted).

A.1.4 Scientific conferences.

1. Benach J, **Martínez JM**, Yasui Y, Borrell C, Pasarín MI, Español E, Benach N, Ginebra J, Clèries R et al. *Atlas of mortality in small areas in Catalonia (1984-1998)*. X congress of the Spanish Biometry Society. Oviedo May 25th-27th 2005 [Spanish].

2. **Martínez JM**, Yasui Y, Benach J, Ginebra J, Benavides FG. *An Integrated Analysis of Individual and Aggregated Health Data*. III Jornadas Científicas de las Sociedades Españolas de Epidemiología y Biometría. Valencia Jun 22th-23th 2006 (acepted) [Spanish].

A.2 General SAS program part 1 (chapter 6).

```
*************************************************************;
*GENERAL PROGRAM CATALONIA (RELATIVE RISK AND TREND ESTIMATES) *;
*This is a general program, for some causes it is necessary not*;
*include age groups where there are not deaths                 *;
*************************************************************;


Data Cat2;
 INFILE 'C:\jmiguel\CARPETA ATLAS DE CATALUÑA\DATOS\data8498_cat.txt' DLM='09'x;
 input zone age sex year cause observed population;
 age1=0;age2=0;age3=0;age4=0;age5=0;age6=0;age7=0;age8=0;age9=0;
 age10=0;age11=0;age12=0;age13=0;age14=0;age15=0;age16=0;age17=0;age18=0;
 IF age=1 then age1=1;IF age=2 then age2=1;IF age=3 then age3=1;
 IF age=4 then age4=1;IF age=5 then age5=1;IF age=6 then age6=1;
 IF age=7 then age7=1;IF age=8 then age8=1;IF age=9 then age9=1;
 IF age=10 then age10=1;IF age=11 then age11=1;IF age=12 then age12=1;
 IF age=13 then age13=1;IF age=14 then age14=1;IF age=15 then age15=1;
 IF age=16 then age16=1;IF age=17 then age17=1;IF age=18 then age18=1;
 IF year=8486 then yearr=1;IF year=8789 then yearr=2;IF year=9092 then yearr=3;
 IF year=9395 then yearr=4;IF year=9698 then yearr=5;
run;


proc freq data=Cat2;
 weight observed;
 tables sex*causer*yearr/nocol norow nocum nopercent;
run;

proc sort data=Cat2;
 by zone age sex causer;
run;

data Cat2_Agr1;
 set Cat2;
 by zone age sex causer;
 retain observado;
 if first.zone or first.age or first.sex or first.causer then observado=observed;
 else observado=(observado+observed);
 if not(last.zone or last.age or last.sex or last.causer)then delete;
 keep zone age sex causer observado;
run;

data Cat2_Agr2;
 set Cat2;
 by zone age sex causer;
 retain poblacion;
 if first.zone or first.age or first.sex or first.causer then poblacion=population;
 else poblacion=(poblacion+population);
 if not(last.zone or last.age or last.sex or last.causer) then delete;
 keep zone age sex causer poblacion;
run;

proc sort data=Cat2_Agr1;
 by zone causer age sex;
run;

proc sort data=Cat2_Agr2;
 by zone causer age sex;
run;

data final_Cat2;
 MERGE Cat2_Agr1 Cat2_Agr2;
 by zone causer age sex;
run;

data final_Cat3;
 set final_Cat2;
 lpy=log(poblacion);
run;


%macro DATA_EXPM;
    %do i=1 %to 18;
```

```
        Data Catm_&i;
         set final_Cat3;
         if causer=&i and sex=1;
        run;
    %end;
%MEND;
%DATA_EXPM;

*GEE approach to obtain reference rates internally;
%macro DATAGEEM;
    %do i=1 %to 18;
        PROC GENMOD data=Catm_&i;
         class zone age;
         model observado=age /d=poisson offset=lpy;
         repeated subject=zone/type=exch;
         ods output GEEEmpPEst=gee&i;
        run;
    %end;
%MEND;
%DATAGEEM;

*Reference rates;
%macro DATA_EXPM2;
    %do i=1 %to 18;
     Data gee2_&i;
       set gee&i;
       dum=1;
     run;
    %end;
%MEND;
%DATA_EXPM2;

%macro DATA_EXPM3;
    %do i=1 %to 18;
     Data gee3_&i;
      set gee2_&i;
      by dum;
      retain valor;
      if first.dum then valor=estimate;
       rate=exp(valor+estimate);
     run;
     Data gee4_&i;
      set gee3_&i;
      cause=&i;
      if level1 <>'';
       age=level1;
      keep cause age estimate valor rate;
     run;
    %end;
%MEND;
%DATA_EXPM3;

%macro DATA_EXPM4;
  %do i=2 %to 18;
   PROC APPEND BASE=gee4_1 DATA=gee4_&i;
   RUN;
  %end;
%MEND;
%DATA_EXPM4;

data geemale1;
 set gee4_1;
 agen=age*1;
 causer=cause;
 keep causer agen rate;
run;



data Cat3;
 set Cat2;
 if sex=1;
 agen=age;
 keep zone causer yearr agen observed population;
run;

proc sort data=geemale1;
```

```
 by causer agen;
run;
proc sort data=Cat3;
 by causer agen;
run;

data fmale1;
 MERGE Cat3 geemale1;
 by causer agen;
run;

*Expected deaths;
data fmale2;
 set fmale1;
 expected=population*rate;
run;

*Observed deaths;
proc sort data=fmale2;
 by zone yearr causer;
run;

data fmale2_Agr1;
 set fmale2;
 by zone yearr causer;
 retain obs;
 if first.zone or first.yearr or first.causer then obs=observed;
 else obs=(obs+observed);
 if not(last.zone or last.yearr or last.causer)then delete;
 keep zone yearr causer obs;
run;

data fmale2_Agr2;
 set fmale2;
 by zone yearr causer;
 retain expec;
 if first.zone or first.yearr or first.causer then expec=expected;
 else expec=(expec+expected);
 if not(last.zone or last.yearr or last.causer) then delete;
 keep zone yearr causer expec;
run;

*Merge observed and expected deaths;
proc sort data=fmale2_Agr1;
 by zone causer yearr;
run;
proc sort data=fmale2_Agr2;
 by zone causer yearr;
run;

data final_male1;
 MERGE fmale2_Agr1 fmale2_Agr2;
 by zone causer yearr;
run;

%macro DATOS;
    %do i=1 %to 18;
       Data Datosm_&i;
        set final_male1;
        if causer=&i;
       run;
    %end;
%MEND;
%DATOS;

*Empirical Bayes model;
%macro CATALONIAMALE;
    %do i=1 %to 18;
       PROC NLMIXED DATA=Datosm_&i TECH=TRUREG;
        parms log_sig=0 log_tau=0 beta0=0;
        ETA=beta0 + s*(yearr-3)+ bi + si*(yearr-3);
        LAMDA=EXP(ETA);
        MEAN=LAMDA*EXPEC;
        MODEL OBS ~ POISSON(MEAN);
        RANDOM si bi ~
              normal ([0,0],
                      [exp(2*log_sig),0,
```

```
                              exp(2*log_tau)])
              subject=zone OUT=Chmap1_a&i;
        run;

        Data Chmap1af&i;
         set Work.Chmap1_a&i;
         RR=exp(Estimate);
         keep zone effect estimate probt RR;
        run;

        Proc sort data=Chmap1af&i;
         by zone;
         run;

        Proc sort data=datosm_&i;
         by zone;
         run;

        data datosmfobs_&i;
         set datosm_&i;
         by zone;
         retain o;
         if first.zone then o=obs;
         else o=(o+obs);
         if not(last.zone)then delete;
         keep zone o;
        run;

        data datosmfexp_&i;
         set datosm_&i;
         by zone;
         retain e;
         if first.zone then e=expec;
         else e=(e+expec);
         if not(last.zone) then delete;
         keep zone e;
        run;

        Proc sort data=datosmfobs_&i;
         by zone;
        run;
        Proc sort data=datosmfexp_&i;
         by zone;
        run;

       data fin_male&i;
         MERGE datosmfobs_&i datosmfexp_&i Chmap1af&i;
         by zone;
        run;

        Data Chmap1arr&i;
         set fin_male&i;
         if effect='bi';
         smr=o/e;
        run;


           Data Chmap1atrend&i;
            set fin_male&i;
            if effect='si';
           run;

         PROC EXPORT DATA= Work.Chmap1arr&i
          OUTFILE= "C:\jmiguel\CARPETA ATLAS DE CATALUÑA\RESULTADOS\MAP\Resrra_&i..xls"
          DBMS=EXCEL2000 REPLACE;
         RUN;

         PROC EXPORT DATA= Work.Chmap1atrend&i
          OUTFILE="C:\jmiguel\CARPETA ATLAS DE CATALUÑA\RESULTADOS\MAP\Restrenda_&i..xls"
          DBMS=EXCEL2000 REPLACE;
         RUN;
   %end;
%MEND;

%CATALONIAMALE;
```

A.3 R programs part 2 (chapters 7 and 8).

A.3.1 NCC simulation program.

```
library(MASS)


#####################################################.
#####################################################.
##FUNCTION INDIVIDUAL RANDOM EFFECTS MODEL (farem)##.
##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMA^2   ##.
#####################################################.
#####################################################.

farem<-function(betanew,data) {

#We suposse the next structure in the dataset:Id (Identification of individual),.
#Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
#observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
#...,Zp (covariate Zp).

Dataprove=data

#K is the number of groups. We suppose that groups are ordered and they have all.
#the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
#N is the number of observations and p is the number of covariates.
N<-dim(Dataprove)[1]
K<-Dataprove[N,2]
p<-dim(Dataprove)[2]-5

ngr<-matrix(,nrow=1,ncol=K)
for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

gamma0<-betanew[1]
beta=betanew[-1]

#Individual outcome.
Yki<-matrix(,nrow=N,ncol=1)
Yki[,1]<-Dataprove[,3]

#Individual mean.
muki<-matrix(,nrow=N,ncol=1)
muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

#Matrix D for the IRM.
Dki<-matrix(,nrow=N,ncol=p+1)
Dki[,1]<-muki[,1]
for (j in 1:p){
Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
}

#Inverse variance-covariance matrix for the IRM.

        ##First, we compute sigma square.
        muki2<-matrix(,nrow=N,ncol=1)
        muki2[,1]<-muki[,1]^2

        Yaver<-matrix(,nrow=1,ncol=K)
        muk<-matrix(,nrow=1,ncol=K)
        phik<-matrix(,nrow=K,ncol=1)
        sigma2rek<-matrix(,nrow=K,ncol=1)

        ini<-1
        end<-ngr[1]
        for (i in 1:K) {
                Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                muk[1,i]<-sum(muki[ini:end])/ngr[i]
                phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                ini<-end+1
                end<-ngr[i+1]+end}

        sigma2re<-sum(sigma2rek[1:K])/K

        #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
        sumk<-matrix(,nrow=1,ncol=K)
        ini<-1
        end<-ngr[1]
        for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                        ini<-end+1
```

```
                        end<-ngr[i+1]+end}

    #Finally, we define the elements of the inverse of V.
    Vki<-list(matrix(,nrow=K,ncol=1))

    for (j in 1:K){
      Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

      for (i in 1:ngr[1,j]) {
        yy=1-(1+sigma2re)*muki[i]
              Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
        Vki[[j]][i,i]<-(1/(muki[i]*yy))-(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
         }
        Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
           }

  ############################.
  #Gradient, Hessian for IRM#.
  ############################.

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
         for (n in 1:(p+1)){
             Dkilist[[j]][,n]<-Dki[ini:end,n]}
             ini<-end+1
             end<-ngr[j+1]+end}

  #Gradient.
  ElementKGr<-list(matrix(,nrow=K,ncol=1))
  Grlist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {
         ElementKGr[[i]]<-t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]
         Grlist[[1]]<-Grlist[[1]]+ElementKGr[[i]]}

  Gr<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gr[i,1]<-Grlist[[1]][i]}

  #Hessian.
  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
         ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
         Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}

  #We return the Gradient (Gr), Hessian(Hs) and sigma^2(sigma2re).


  list(Gradient=Gr,Hessian=Hs,sigma2re=sigma2re)}


  ######################################################.
  ######################################################.
  ##FUNCTION AGGREGATED RANDOM EFFECTS MODEL (fagrem)##.
  ##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMA^2    ##.
  ######################################################.
  ######################################################.


  fagrem<-function(betanew,data) {

  #We suposse the next structure in the dataset:Id (Identification of individual),.
```

```
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).

  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]

  #Individual mean (muki).
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Variance for the ARM.
  muki2<-matrix(,nrow=N,ncol=1)
  muki2[,1]<-muki[,1]^2

  #Outcome for the ARM as defined in Sheppard and Prentice (Biometrics,1995).
  Y<-matrix(,nrow=1,ncol=K)

  muk<-matrix(,nrow=1,ncol=K)
  phik<-matrix(,nrow=K,ncol=1)
  #Matrix D for the ARM.
  Dk<-matrix(,nrow=K,ncol=p+1)

  #First, we compute sigma square.
  sigma2amk<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                  Y[1,i]<-((Dataprove[ini,4])/(Dataprove[ini,5]))
                  muk[1,i]<-sum(muki[ini:end])/ngr[i]
                  phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                  for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                  sigma2amk[i,]<-max(((Y[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]))),-100)
                  ini<-end+1
                  end<-ngr[i+1]+end}

  sigma2am<-sum(sigma2amk[1:K])/K

  #Finally, we define the variance.
  Vk<-matrix(,nrow=1,ncol=K)
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vk[1,i]<-sigma2am*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5])))+(muk[i]-
phik[i,])*(1/(Dataprove[ini,5]))
                  ini<-1
                  end<-ngr[1]}

  ############################.
  #Gradient, Hessian for ARM#.
  ############################.

  #Gradient.
  Dkt<-t(Dk)
  ElementKaGr<-list(matrix(,nrow=K,ncol=1))
  Gralist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {ElementKaGr[[i]]<-Dkt[,i]*(1/Vk[i])*(Y[i]-muk[i])
  Gralist[[1]]<-Gralist[[1]]+ElementKaGr[[i]]}

  Gra<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gra[i,1]<-Gralist[[1]][i]}

  #Hessian.
  ElementKaHs<-list(matrix(,nrow=K,ncol=1))
  Hsalist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKaHs[[i]]<--1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vk[i])%*%Dk[i,]
                  Hsalist[[1]]<-Hsalist[[1]]+ElementKaHs[[i]]}
```

```
  Hsa<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsa[i,j]<-Hsalist[[1]][i,j]}}

  #We return the Gradient (Gra), Hessian(Hsa) and sigma^2(sigma2am).

  list(Gradienta=Gra,Hessiana=Hsa,sigma2am=sigma2am)}


  ##########################################################.
  ##########################################################.
  ##FUNCTION POPULATION-BASED ESTIMATING EQUATION (fpbm)##.
  ##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMAS ^ 2    ##.
  ##########################################################.
  ##########################################################.

  fpbm<-function(betanew,data) {

  #We suposse the next structure in the dataset:Id (Identification of individual),.
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).

  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]


  ##########################################.
  #Gradient, Hessian for the individual part#.
  ##########################################.

  #Individual outcome.
  Yki<-matrix(,nrow=N,ncol=1)
  Yki[,1]<-Dataprove[,3]

  #Individual mean.
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Inverse variance-covariance matrix individual part.
          #First, we compute sigma square for the individual part.

          muki2<-matrix(,nrow=N,ncol=1)
          muki2[,1]<-muki[,1]^2

          Yaver<-matrix(,nrow=1,ncol=K)
          muk<-matrix(,nrow=1,ncol=K)
          phik<-matrix(,nrow=K,ncol=1)
          sigma2rek<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                  Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                  muk[1,i]<-sum(muki[ini:end])/ngr[i]
                  phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                  sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                  ini<-end+1
                  end<-ngr[i+1]+end}

          sigma2re<-sum(sigma2rek[1:K])/K

          #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
          sumk<-matrix(,nrow=1,ncol=K)
          ini<-1
          end<-ngr[1]
```

```
        for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                        ini<-end+1
                        end<-ngr[i+1]+end}

    #Finally we define the elements of the inverse of V.
    Vki<-list(matrix(,nrow=K,ncol=1))

    for (j in 1:K){
      Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

      for (i in 1:ngr[1,j]) {
        yy=1-(1+sigma2re)*muki[i]
         Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
              Vki[[j]][i,i]<-(1/(muki[i]*yy))-
(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
      }
        Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
           }

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
          for (n in 1:(p+1)){
              Dkilist[[j]][,n]<-Dki[ini:end,n]}
              ini<-end+1
              end<-ngr[j+1]+end}

  #Gradient Individual part.
  ElementKGr<-list(matrix(,nrow=K,ncol=1))
  Grlist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {
          ElementKGr[[i]]<-t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]
          Grlist[[1]]<-Grlist[[1]]+ElementKGr[[i]]}

  Gr<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gr[i,1]<-Grlist[[1]][i]}

  ##Hessian individual part.
  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
          ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
          Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}



  ##########################################.
  #Gradient, Hessian for the aggregated part#.
  ##########################################.


  #Outcome for the aggregated data model with combined analytical and aggregated models.
  Ybar<-matrix(,nrow=1,ncol=K)

  #Matrix D for the aggregated part.
  Dk<-matrix(,nrow=K,ncol=p+1)

        ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ybar[1,i]<-((Dataprove[ini,4]-sum(Yki[ini:end]))/(Dataprove[ini,5]-ngr[i]))
                  for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
```

```
                ini<-end+1
                end<-ngr[i+1]+end}

  #Sigma square aggregated part.
  sigma2pbk<-matrix(,nrow=K,ncol=1)
  ini<-1
  end<-ngr[1]
  for  (i in 1:K) {sigma2pbk[i,]<-max(((Ybar[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]-ngr[i]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]-ngr[i]))),0)
                ini<-end+1
                end<-ngr[i+1]+end}

  sigma2pb<-sum(sigma2pbk[1:K])/K

  #Variance aggregated part.
  Dkt<-t(Dk)
  Vkbar<-matrix(,nrow=1,ncol=K)
  ElementKarGr<-list(matrix(,nrow=K,ncol=1))
  Grarlist<-list(matrix(0,nrow=(p+1),ncol=1))
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vkbar[1,i]<-sigma2pb*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5]-
ngr[i])))+(muk[i]-phik[i,])*(1/(Dataprove[ini,5]-ngr[i]))
                ElementKarGr[[i]]<-Dkt[,i]*(1/Vkbar[i])*(Ybar[i]-muk[i])
                Grarlist[[1]]<-Grarlist[[1]]+ElementKarGr[[i]]
                ini<-end+1
                end<-ngr[i+1]+end}

  #Gradient aggregated part.
  Grar<-matrix(,nrow=(p+1),ncol=1)
  for (i in 1:(p+1)) {Grar[i,1]<-Grarlist[[1]][i]}

  #Hessian aggregated part.

  ElementKarHs<-list(matrix(,nrow=K,ncol=1))
  Hsarlist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKarHs[[i]]<--
1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vkbar[i])%*%Dk[i,]
                Hsarlist[[1]]<-Hsarlist[[1]]+ElementKarHs[[i]]}

  Hsar<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsar[i,j]<-Hsarlist[[1]][i,j]}}

  ##########################################################################.
  #Gradient, Hessian for the individual and aggregated part combination#.
  ##########################################################################.

  #Gradient PBEE.

  Grpb<-(Gr+Grar)

  #Hessian PBEE.

  Hspb<-(Hs+Hsar)

  #We return the Gradient (Grpb), Hessian(Hspb), sigma^2 individual.
  #part (sigma2re) and sigma^2 aggregated part (sigma2pb).

  list(Gradientpb=Grpb,Hessianpb=Hspb,sigma2re=sigma2re,sigma2pb=sigma2pb)}



  ##########################################################################.
  ##########################################################################.
  ##PARAMETER ESTIMATES USING THE NEWTON-RAPHSON ALGORITHM FOR THE IRM##.
  ##(function farem2, ARM (function fagrem2) AND PBEE (function fpbm2)##.
  ##########################################################################.
  ##########################################################################.

  #We suposse the next structure in the dataset:Id (Identification of individual),.
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).


  farem2<-function(data,tol,maxiter=100,betainitial){

  betanew<-betainitial
  betaold<-betanew+1
  itercount=0

  #Solutions for the IRM.

  while(max(abs((betaold-betanew)/betaold))>tol){
       q<-farem(betanew,data)
       betaold<-betanew
```

```
     betanew<-betanew-ginv(q$Hessian)%*%q$Gradient
     itercount=itercount+1
     if(is.na(q$sigma2re)) itercount=100
     if(itercount>maxiter) break
     if(q$sigma2re>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanew=betanew,sigma2re=q$sigma2re, iterN=itercount)
      }

fagrem2<-function(data,tol,maxiter=100,betainitial){

betanewa<-betainitial
betaold<-betanewa+1
itercount=0

#Solutions for the ARM.

while(max(abs((betaold-betanewa)/betaold))>tol){
     d<-fagrem(betanewa,data)
     betaold<-betanewa
     betanewa<-betanewa-ginv(d$Hessiana)%*%d$Gradienta
     itercount=itercount+1
     if(is.na(d$sigma2am)) itercount=100
     if(itercount>maxiter) break
     if(d$sigma2am>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanewa=betanewa,sigma2am=d$sigma2am, iterN=itercount)
      }

fpbm2<-function(data,tol,maxiter=100,betainitial){

betanewpb<-betainitial
betaold<-betanewpb+1
itercount=0

#Solutions for the PBEE.

while(max(abs((betaold-betanewpb)/betaold))>tol){
     e<-fpbm(betanewpb,data)
     betaold<-betanewpb
     betanewpb<-betanewpb-ginv(e$Hessianpb)%*%e$Gradientpb
     itercount=itercount+1
     if(is.na(max(e$sigma2re,e$sigma2pb))) itercount=100
     if(itercount>maxiter) break
     if(max(e$sigma2re,e$sigma2pb)>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanewpb=betanewpb,sigma2re=e$sigma2re,sigma2pb=e$sigma2pb, iterN=itercount)
      }


################################################.
################################################.
##FUNCTION FOR SIMULATE THE DATA (fgenerate2)##.
##IN THE NON CONFOUNDING CASE (NCC)          ##.
################################################.
################################################.


fgenerate2<-function(group,populationsize,samplesize,variance){

#group=number of groups, populationsize= population size.
#samplesize= sample size, variance=within group variance.

K<-group
nk<-populationsize
mk<-samplesize
varwithin<-variance

#Covariate X1ki with ratio (within variance)/(between variance) equal varwithin/1.

Z1kg<-rnorm(K,0,sqrt(1))

X1ki<-t(matrix(rnorm(nk*K,Z1kg,sqrt(varwithin)),nrow=K,ncol=nk))

#Covariate X2ki with ratio (within variance)/(between variance) equal 1/1.

Z2kg<-matrix(rnorm(K,0,sqrt(1)),nrow=K,ncol=1)

X2ki<-t(matrix(rnorm(nk*K,Z2kg,sqrt(1)),nrow=K,ncol=nk))

#Country specific frailties were generated as independent.
#realized  values from a gamma  distribution with  mean 1.
#and variance sigma^2. The mean of a gamma is shape*scale.
#and the variance is shape*(scale^2).
```

```
  meanhk<-1
  varhk<-0.05
  shape<-(meanhk^2)/(varhk)
  scale<-(varhk)/(meanhk)
  hk<-rgamma(K,shape=shape,scale=scale)[]
  hk=t(matrix(rep(hk,nk),nrow=K,ncol=nk))

  #The disease events, yki, were generated by determining.
  #wether  a  uniform  random  variable  wass  less  than.
  #hk*exp(gamma0+beta1*X1ki+beta2*X2ki).

  gamma0<--3
  beta1<-0.2
  beta2<-0.2

  yki<-matrix(,nrow=nk,ncol=K)
  unif<-matrix(runif(nk*K,0,1),nrow=nk,ncol=K)

        yki=ifelse(unif<hk*exp(gamma0+beta1*X1ki+beta2*X2ki),1,0)

  ######################################.
  #selection of random sample of size mk#.
  #and organize data to apply functions #.
  #farem, fagrem and fpbm               #.
  ######################################.

  datalist<-list(matrix(,nrow=K,ncol=1))
  sampledatalist<-list(matrix(,nrow=K,ncol=1))
  ini<-1
  end<-mk
  data<-matrix(,nrow=mk*K,ncol=5)
  for (i in 1:K){

  datalist[[i]]<-
cbind(matrix(c(1:nk),nrow=nk,ncol=1),matrix(yki[,i],nrow=nk,ncol=1),matrix(X1ki[,i],nrow=nk,n
col=1),matrix(X2ki[,i],nrow=nk,ncol=1),matrix(c(i),nrow=nk,ncol=1))
  sampledatalist[[i]]<-datalist[[i]][as.matrix(sample(datalist[[i]][,1],mk)),]
  data[ini:end,]<-sampledatalist[[i]][]
                 ini<-end+1
                 end<-mk*(i+1)
  }

  O<-matrix(apply(yki,2,sum),nrow=K,ncol=1)

  ini<-1
  end<-mk
  datapop<-matrix(,nrow=mk*K,ncol=1)
  for (i in 1:K) {datapop[ini:end,1]<-O[i,]
                  ini<-end+1
                  end<-mk*(i+1)}

  datadatapop<-cbind(data,datapop,c(nk))
  datafin<-
data.frame(id=matrix(datadatapop[,1]),group=matrix(datadatapop[,5]),YIND=matrix(datadatapop[,
2]),O=matrix(datadatapop[,6]),n=matrix(datadatapop[,7]),X1ki=matrix(datadatapop[,3]),X2ki=mat
rix(datadatapop[,4]))}


  ######################.
  ##Simulation results##.
  ######################.

  fsimulationA<-function(seed,Niter,sigma2,K,N){
  set.seed(seed)
  count1=0
  result1=matrix(0,nrow=Niter,ncol=16)

  while(count1<Niter){
  tempdata1=fgenerate2(K,2000,N,sigma2)
  tol<-0.001
  gg<-glm(YIND~X1ki+X2ki,data=tempdata1,family=binomial)
  betaini<-
as.vector(c(matrix(gg$coefficients[1]),matrix(gg$coefficients[2]),matrix(gg$coefficients[3]))
)
  tempdata1a1<-farem2(tempdata1,tol,maxiter=50,betaini)
  tempdata1a2<-fagrem2(tempdata1,tol,maxiter=50,betaini)
  tempdata1a3<-fpbm2(tempdata1,tol,maxiter=50,betaini)
  count1=count1+1
  result1[count1,1:5]=matrix(c(t(tempdata1a1$betanew),tempdata1a1$sigma2re,tempdata1a1$iterN)
,nrow=1,ncol=5)
  result1[count1,6:10]=matrix(c(t(tempdata1a2$betanewa),tempdata1a2$sigma2am,tempdata1a2$iter
N),nrow=1,ncol=5)
  result1[count1,11:16]=matrix(c(t(tempdata1a3$betanewpb),tempdata1a3$sigma2re,tempdata1a3$si
gma2pb,tempdata1a3$iterN),nrow=1,ncol=6)
  print(count1)
  }
```

```
list(result=result1,sigma2=sigma2,K=K,N=N)
}


#100 groups-100 sample size in each group.

finalresultA100100A.25=fsimulationA(123,1000,.25,100,100)
save(list=c("finalresultA100100A.25",".Random.seed"),file="100100A025.RData")
savedseed=.Random.seed

finalresultA100100A.5=fsimulationA(savedseed,1000,.5,100,100)
save(list=c("finalresultA100100A.5",".Random.seed"),file="100100A05.RData")
savedseed=.Random.seed

finalresultA100100A1=fsimulationA(savedseed,1000,1,100,100)
save(list=c("finalresultA100100A1",".Random.seed"),file="100100A1.RData")
savedseed=.Random.seed

finalresultA100100A2=fsimulationA(savedseed,1000,2,100,100)
save(list=c("finalresultA100100A2",".Random.seed"),file="100100A2.RData")
savedseed=.Random.seed

finalresultA100100A4=fsimulationA(savedseed,1000,4,100,100)
save(list=c("finalresultA100100A4",".Random.seed"),file="100100A4.RData")
savedseed=.Random.seed

finalresultA100100A8=fsimulationA(savedseed,1000,8,100,100)
save(list=c("finalresultA100100A8",".Random.seed"),file="100100A8.RData")
savedseed=.Random.seed

finalresultA100100A16=fsimulationA(savedseed,1000,16,100,100)
save(list=c("finalresultA100100A16",".Random.seed"),file="100100A16.RData")


#100 groups-50 sample size in each group.

finalresultA10050A.25=fsimulationA(123,1000,.25,100,50)
save(list=c("finalresultA10050A.25",".Random.seed"),file="10050A025.RData")
savedseed=.Random.seed

finalresultA10050A.5=fsimulationA(savedseed,1000,.5,100,50)
save(list=c("finalresultA10050A.5",".Random.seed"),file="10050A05.RData")
savedseed=.Random.seed

finalresultA10050A1=fsimulationA(savedseed,1000,1,100,50)
save(list=c("finalresultA10050A1",".Random.seed"),file="10050A1.RData")
savedseed=.Random.seed

finalresultA10050A2=fsimulationA(savedseed,1000,2,100,50)
save(list=c("finalresultA10050A2",".Random.seed"),file="10050A2.RData")
savedseed=.Random.seed

finalresultA10050A4=fsimulationA(savedseed,1000,4,100,50)
save(list=c("finalresultA10050A4",".Random.seed"),file="10050A4.RData")
savedseed=.Random.seed

finalresultA10050A8=fsimulationA(savedseed,1000,8,100,50)
save(list=c("finalresultA10050A8",".Random.seed"),file="10050A8.RData")
savedseed=.Random.seed

finalresultA10050A16=fsimulationA(savedseed,1000,16,100,50)
save(list=c("finalresultA10050A16",".Random.seed"),file="10050A16.RData")


#50 groups-100 sample size in each group.

finalresultA50100A.25=fsimulationA(123,1000,.25,50,100)
save(list=c("finalresultA50100A.25",".Random.seed"),file="50100A025.RData")
savedseed=.Random.seed

finalresultA50100A.5=fsimulationA(savedseed,1000,.5,50,100)
save(list=c("finalresultA50100A.5",".Random.seed"),file="50100A05.RData")
savedseed=.Random.seed

finalresultA50100A1=fsimulationA(savedseed,1000,1,50,100)
save(list=c("finalresultA50100A1",".Random.seed"),file="50100A1.RData")
savedseed=.Random.seed

finalresultA50100A2=fsimulationA(savedseed,1000,2,50,100)
save(list=c("finalresultA50100A2",".Random.seed"),file="50100A2.RData")
savedseed=.Random.seed

finalresultA50100A4=fsimulationA(savedseed,1000,4,50,100)
save(list=c("finalresultA50100A4",".Random.seed"),file="50100A4.RData")
savedseed=.Random.seed
```

```
finalresultA50100A8=fsimulationA(savedseed,1000,8,50,100)
save(list=c("finalresultA50100A8",".Random.seed"),file="50100A8.RData")
savedseed=.Random.seed

finalresultA50100A16=fsimulationA(savedseed,1000,16,50,100)
save(list=c("finalresultA50100A16",".Random.seed"),file="50100A16.RData")


#50 groups-50 sample size in each group.

finalresultA5050A.25=fsimulationA(123,1000,.25,50,50)
save(list=c("finalresultA5050A.25",".Random.seed"),file="5050A025.RData")
savedseed=.Random.seed

finalresultA5050A.5=fsimulationA(savedseed,1000,.5,50,50)
save(list=c("finalresultA5050A.5",".Random.seed"),file="5050A05.RData")
savedseed=.Random.seed

finalresultA5050A1=fsimulationA(savedseed,1000,1,50,50)
save(list=c("finalresultA5050A1",".Random.seed"),file="5050A1.RData")
savedseed=.Random.seed

finalresultA5050A2=fsimulationA(savedseed,1000,2,50,50)
save(list=c("finalresultA5050A2",".Random.seed"),file="5050A2.RData")
savedseed=.Random.seed

finalresultA5050A4=fsimulationA(savedseed,1000,4,50,50)
save(list=c("finalresultA5050A4",".Random.seed"),file="5050A4.RData")
savedseed=.Random.seed

finalresultA5050A8=fsimulationA(savedseed,1000,8,50,50)
save(list=c("finalresultA5050A8",".Random.seed"),file="5050A8.RData")
savedseed=.Random.seed

finalresultA5050A16=fsimulationA(savedseed,1000,16,50,50)
save(list=c("finalresultA5050A16",".Random.seed"),file="5050A16.RData")

################################.
#Import results to a text file#.
################################.

#100 groups-100 sample size in each group.

write.table(finalresultA100100A.25$result,"dataA1var1.txt",row.names=F)  #variance 0.25.
write.table(finalresultA100100A.5$result,"dataA1var2.txt",row.names=F)   #variance 0.5.
write.table(finalresultA100100A1$result,"dataA1var3.txt",row.names=F)    #variance 1.
write.table(finalresultA100100A2$result,"dataA1var4.txt",row.names=F)    #variance 2.
write.table(finalresultA100100A4$result,"dataA1var5.txt",row.names=F)    #variance 4.
write.table(finalresultA100100A8$result,"dataA1var6.txt",row.names=F)    #variance 8.
write.table(finalresultA100100A16$result,"dataA1var7.txt",row.names=F)   #variance 16.


#50 groups-100 sample size in each group.

write.table(finalresultA50100A.25$result,"dataA2var1.txt",row.names=F)   #variance 0.25.
write.table(finalresultA50100A.5$result,"dataA2var2.txt",row.names=F)    #variance 0.5.
write.table(finalresultA50100A1$result,"dataA2var3.txt",row.names=F)     #variance 1.
write.table(finalresultA50100A2$result,"dataA2var4.txt",row.names=F)     #variance 2.
write.table(finalresultA50100A4$result,"dataA2var5.txt",row.names=F)     #variance 4.
write.table(finalresultA50100A8$result,"dataA2var6.txt",row.names=F)     #variance 8.
write.table(finalresultA50100A16$result,"dataA2var7.txt",row.names=F)    #variance 16.


#100 groups-50 sample size in each group.

write.table(finalresultA10050A.25$result,"dataA3var1.txt",row.names=F)   #variance 0.25.
write.table(finalresultA10050A.5$result,"dataA3var2.txt",row.names=F)    #variance 0.5.
write.table(finalresultA10050A1$result,"dataA3var3.txt",row.names=F)     #variance 1.
write.table(finalresultA10050A2$result,"dataA3var4.txt",row.names=F)     #variance 2.
write.table(finalresultA10050A4$result,"dataA3var5.txt",row.names=F)     #variance 4.
write.table(finalresultA10050A8$result,"dataA3var6.txt",row.names=F)     #variance 8.
write.table(finalresultA10050A16$result,"dataA3var7.txt",row.names=F)    #variance 16.

#100 groups-100 sample size in each group.

write.table(finalresultA5050A.25$result,"dataA4var1.txt",row.names=F)   #variance 0.25.
write.table(finalresultA5050A.5$result,"dataA4var2.txt",row.names=F)    #variance 0.5.
write.table(finalresultA5050A1$result,"dataA4var3.txt",row.names=F)     #variance 1.
write.table(finalresultA5050A2$result,"dataA4var4.txt",row.names=F)     #variance 2.
write.table(finalresultA5050A4$result,"dataA4var5.txt",row.names=F)     #variance 4.
write.table(finalresultA5050A8$result,"dataA4var6.txt",row.names=F)     #variance 8.
write.table(finalresultA5050A16$result,"dataA4var7.txt",row.names=F)    #variance 16.
```

A.3.2 SCC simulation program.

```
library(MASS)


#####################################################.
#####################################################.
##FUNCTION INDIVIDUAL RANDOM EFFECTS MODEL (farem)##.
##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMA^2   ##.
#####################################################.
#####################################################.

farem<-function(betanew,data) {

#We suposse the next structure in the dataset:Id (Identification of individual),.
#Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
#observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
#...,Zp (covariate Zp).

Dataprove=data

#K is the number of groups. We suppose that groups are ordered and they have all.
#the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
#N is the number of observations and p is the number of covariates.
N<-dim(Dataprove)[1]
K<-Dataprove[N,2]
p<-dim(Dataprove)[2]-5

ngr<-matrix(,nrow=1,ncol=K)
for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

gamma0<-betanew[1]
beta=betanew[-1]

#Individual outcome.
Yki<-matrix(,nrow=N,ncol=1)
Yki[,1]<-Dataprove[,3]

#Individual mean.
muki<-matrix(,nrow=N,ncol=1)
muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

#Matrix D for the IRM.
Dki<-matrix(,nrow=N,ncol=p+1)
Dki[,1]<-muki[,1]
for (j in 1:p){
Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
}

#Inverse variance-covariance matrix for the IRM.

        ##First, we compute sigma square.
        muki2<-matrix(,nrow=N,ncol=1)
        muki2[,1]<-muki[,1]^2

        Yaver<-matrix(,nrow=1,ncol=K)
        muk<-matrix(,nrow=1,ncol=K)
        phik<-matrix(,nrow=K,ncol=1)
        sigma2rek<-matrix(,nrow=K,ncol=1)

        ini<-1
        end<-ngr[1]
        for (i in 1:K) {
                Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                muk[1,i]<-sum(muki[ini:end])/ngr[i]
                phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                ini<-end+1
                end<-ngr[i+1]+end}

        sigma2re<-sum(sigma2rek[1:K])/K

        #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
        sumk<-matrix(,nrow=1,ncol=K)
        ini<-1
        end<-ngr[1]
        for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                        ini<-end+1
                        end<-ngr[i+1]+end}

    #Finally, we define the elements of the inverse of V.
    Vki<-list(matrix(,nrow=K,ncol=1))
```

```
    for (j in 1:K){
      Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

      for (i in 1:ngr[1,j]) {
        yy=1-(1+sigma2re)*muki[i]
              Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
        Vki[[j]][i,i]<-(1/(muki[i]*yy))-(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
         }
        Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
           }

  ############################.
  #Gradient, Hessian for IRM#.
  ############################.

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
          for (n in 1:(p+1)){
              Dkilist[[j]][,n]<-Dki[ini:end,n]}
              ini<-end+1
              end<-ngr[j+1]+end}

  #Gradient.
  ElementKGr<-list(matrix(,nrow=K,ncol=1))
  Grlist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {
          ElementKGr[[i]]<-t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]
          Grlist[[1]]<-Grlist[[1]]+ElementKGr[[i]]}

  Gr<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gr[i,1]<-Grlist[[1]][i]}

  #Hessian.
  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
          ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
          Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}

  #We return the Gradient (Gr), Hessian(Hs) and sigma^2(sigma2re).


  list(Gradient=Gr,Hessian=Hs,sigma2re=sigma2re)}


  ######################################################.
  ######################################################.
  ##FUNCTION AGGREGATED RANDOM EFFECTS MODEL (fagrem)##.
  ##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMA^2    ##.
  ######################################################.
  ######################################################.


  fagrem<-function(betanew,data) {

  #We suposse the next structure in the dataset:Id (Identification of individual),.
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).
```

```
  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]

  #Individual mean (muki).
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Variance for the ARM.
  muki2<-matrix(,nrow=N,ncol=1)
  muki2[,1]<-muki[,1]^2

  #Outcome for the ARM as defined in Sheppard and Prentice (Biometrics,1995).
  Y<-matrix(,nrow=1,ncol=K)

  muk<-matrix(,nrow=1,ncol=K)
  phik<-matrix(,nrow=K,ncol=1)
  #Matrix D for the ARM.
  Dk<-matrix(,nrow=K,ncol=p+1)

  #First, we compute sigma square.
  sigma2amk<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                 Y[1,i]<-((Dataprove[ini,4])/(Dataprove[ini,5]))
                 muk[1,i]<-sum(muki[ini:end])/ngr[i]
                 phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                 for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                 sigma2amk[i,]<-max(((Y[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]))),-100)
                 ini<-end+1
                 end<-ngr[i+1]+end}

  sigma2am<-sum(sigma2amk[1:K])/K

  #Finally, we define the variance.
  Vk<-matrix(,nrow=1,ncol=K)
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vk[1,i]<-sigma2am*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5])))+(muk[i]-
phik[i,])*(1/(Dataprove[ini,5]))
                 ini<-1
                 end<-ngr[1]}

  ############################.
  #Gradient, Hessian for ARM#.
  ############################.

  #Gradient.
  Dkt<-t(Dk)
  ElementKaGr<-list(matrix(,nrow=K,ncol=1))
  Gralist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {ElementKaGr[[i]]<-Dkt[,i]*(1/Vk[i])*(Y[i]-muk[i])
  Gralist[[1]]<-Gralist[[1]]+ElementKaGr[[i]]}

  Gra<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gra[i,1]<-Gralist[[1]][i]}

  #Hessian.
  ElementKaHs<-list(matrix(,nrow=K,ncol=1))
  Hsalist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKaHs[[i]]<--1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vk[i])%*%Dk[i,]
                 Hsalist[[1]]<-Hsalist[[1]]+ElementKaHs[[i]]}

  Hsa<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsa[i,j]<-Hsalist[[1]][i,j]}}
```

```
  #We return the Gradient (Gra), Hessian(Hsa) and sigma^2(sigma2am).

  list(Gradienta=Gra,Hessiana=Hsa,sigma2am=sigma2am)}


  ##########################################################.
  ##########################################################.
  ##FUNCTION POPULATION-BASED ESTIMATING EQUATION (fpbm)##.
  ##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMAS ^ 2    ##.
  ##########################################################.
  ##########################################################.

  fpbm<-function(betanew,data) {

  #We suposse the next structure in the dataset:Id (Identification of individual),.
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).

  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]


  ##########################################.
  #Gradient, Hessian for the individual part#.
  ##########################################.

  #Individual outcome.
  Yki<-matrix(,nrow=N,ncol=1)
  Yki[,1]<-Dataprove[,3]

  #Individual mean.
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Inverse variance-covariance matrix individual part.
          #First, we compute sigma square for the individual part.

          muki2<-matrix(,nrow=N,ncol=1)
          muki2[,1]<-muki[,1]^2

          Yaver<-matrix(,nrow=1,ncol=K)
          muk<-matrix(,nrow=1,ncol=K)
          phik<-matrix(,nrow=K,ncol=1)
          sigma2rek<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                  Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                  muk[1,i]<-sum(muki[ini:end])/ngr[i]
                  phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                  sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                  ini<-end+1
                  end<-ngr[i+1]+end}

          sigma2re<-sum(sigma2rek[1:K])/K

          #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
          sumk<-matrix(,nrow=1,ncol=K)
          ini<-1
          end<-ngr[1]
          for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                          ini<-end+1
                          end<-ngr[i+1]+end}
```

```
    #Finally we define the elements of the inverse of V.
    Vki<-list(matrix(,nrow=K,ncol=1))

    for (j in 1:K){
      Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

      for (i in 1:ngr[1,j]) {
        yy=1-(1+sigma2re)*muki[i]
         Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
             Vki[[j]][i,i]<-(1/(muki[i]*yy))-
(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
      }
        Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
          }

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
          for (n in 1:(p+1)){
              Dkilist[[j]][,n]<-Dki[ini:end,n]}
              ini<-end+1
              end<-ngr[j+1]+end}

  #Gradient Individual part.
  ElementKGr<-list(matrix(,nrow=K,ncol=1))
  Grlist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {
          ElementKGr[[i]]<-t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]
          Grlist[[1]]<-Grlist[[1]]+ElementKGr[[i]]}

  Gr<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gr[i,1]<-Grlist[[1]][i]}

  ##Hessian individual part.
  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
          ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
          Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}



  ##########################################.
  #Gradient, Hessian for the aggregated part#.
  ##########################################.


  #Outcome for the aggregated data model with combined analytical and aggregated models.
  Ybar<-matrix(,nrow=1,ncol=K)

  #Matrix D for the aggregated part.
  Dk<-matrix(,nrow=K,ncol=p+1)

        ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ybar[1,i]<-((Dataprove[ini,4]-sum(Yki[ini:end]))/(Dataprove[ini,5]-ngr[i]))
                  for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                  ini<-end+1
                  end<-ngr[i+1]+end}

  #Sigma square aggregated part.
```

```
  sigma2pbk<-matrix(,nrow=K,ncol=1)
  ini<-1
  end<-ngr[1]
  for  (i in 1:K) {sigma2pbk[i,]<-max(((Ybar[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]-ngr[i]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]-ngr[i]))),0)
                ini<-end+1
                end<-ngr[i+1]+end}

  sigma2pb<-sum(sigma2pbk[1:K])/K

  #Variance aggregated part.
  Dkt<-t(Dk)
  Vkbar<-matrix(,nrow=1,ncol=K)
  ElementKarGr<-list(matrix(,nrow=K,ncol=1))
  Grarlist<-list(matrix(0,nrow=(p+1),ncol=1))
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vkbar[1,i]<-sigma2pb*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5]-
ngr[i])))+(muk[i]-phik[i,])*(1/(Dataprove[ini,5]-ngr[i]))
                ElementKarGr[[i]]<-Dkt[,i]*(1/Vkbar[i])*(Ybar[i]-muk[i])
                Grarlist[[1]]<-Grarlist[[1]]+ElementKarGr[[i]]
                ini<-end+1
                end<-ngr[i+1]+end}

  #Gradient aggregated part.
  Grar<-matrix(,nrow=(p+1),ncol=1)
  for (i in 1:(p+1)) {Grar[i,1]<-Grarlist[[1]][i]}

  #Hessian aggregated part.

  ElementKarHs<-list(matrix(,nrow=K,ncol=1))
  Hsarlist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKarHs[[i]]<--
1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vkbar[i])%*%Dk[i,]
                Hsarlist[[1]]<-Hsarlist[[1]]+ElementKarHs[[i]]}

  Hsar<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsar[i,j]<-Hsarlist[[1]][i,j]}}

  ##########################################################################.
  #Gradient, Hessian for the individual and aggregated part combination#.
  ##########################################################################.

  #Gradient PBEE.

  Grpb<-(Gr+Grar)

  #Hessian PBEE.

  Hspb<-(Hs+Hsar)

  #We return the Gradient (Grpb), Hessian(Hspb), sigma^2 individual.
  #part (sigma2re) and sigma^2 aggregated part (sigma2pb).

  list(Gradientpb=Grpb,Hessianpb=Hspb,sigma2re=sigma2re,sigma2pb=sigma2pb)}



  ##########################################################################.
  ##########################################################################.
  ##PARAMETER ESTIMATES USING THE NEWTON-RAPHSON ALGORITHM FOR THE IRM##.
  ##(function farem2, ARM (function fagrem2) AND PBEE (function fpbm2)##.
  ##########################################################################.
  ##########################################################################.

  #We suposse the next structure in the dataset:Id (Identification of individual),.
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).

  farem2<-function(data,tol,maxiter=100,betainitial){

  betanew<-betainitial
  betaold<-betanew+1
  itercount=0
  Hdim=length(betanew)^2

  #Solutions for the IRM.

  while(max(abs((betaold-betanew)/betaold))>tol){
       q<-farem(betanew,data)
       betaold<-betanew
       if(sum(is.finite(q$Hessian))<Hdim) itercount=999
       if(itercount>maxiter) break
       betanew<-betanew-ginv(q$Hessian)%*%q$Gradient
       itercount=itercount+1
```

```
     if(is.na(q$sigma2re)) itercount=100
     if(itercount>maxiter) break
     if(q$sigma2re>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanew=betanew,sigma2re=q$sigma2re, iterN=itercount)
      }

fagrem2<-function(data,tol,maxiter=100,betainitial){

betanewa<-betainitial
betaold<-betanewa+1
itercount=0
Hdim=length(betanewa)^2

#Solutions for the ARM.

while(max(abs((betaold-betanewa)/betaold))>tol){
     d<-fagrem(betanewa,data)
     betaold<-betanewa
     if(sum(is.finite(d$Hessiana))<Hdim) itercount=999
     if(itercount>maxiter) break
     betanewa<-betanewa-ginv(d$Hessiana)%*%d$Gradienta
     itercount=itercount+1
     if(is.na(d$sigma2am)) itercount=100
     if(itercount>maxiter) break
     if(d$sigma2am>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanewa=betanewa,sigma2am=d$sigma2am, iterN=itercount)
      }

fpbm2<-function(data,tol,maxiter=100,betainitial){

betanewpb<-betainitial
betaold<-betanewpb+1
itercount=0
Hdim=length(betanewpb)^2

#Solutions for the PBEE.

while(max(abs((betaold-betanewpb)/betaold))>tol){
     e<-fpbm(betanewpb,data)
     betaold<-betanewpb
     if(sum(is.finite(e$Hessianpb))<Hdim) itercount=999
     if(itercount>maxiter) break
     betanewpb<-betanewpb-ginv(e$Hessianpb)%*%e$Gradientpb
     itercount=itercount+1
     if(is.na(max(e$sigma2re,e$sigma2pb))) itercount=100
     if(itercount>maxiter) break
     if(max(e$sigma2re,e$sigma2pb)>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanewpb=betanewpb,sigma2re=e$sigma2re,sigma2pb=e$sigma2pb, iterN=itercount)
      }

##############################################.
##############################################.
##FUNCTION FOR SIMULATE THE DATA (fgenerate2)##.
##IN THE SIMPLE CONFOUNDING CASE (SCC)       ##.
##############################################.
##############################################.

fgenerate2<-function(group,populationsize,samplesize,variance){

#group=number of groups, populationsize= population size.
#samplesize= sample size, variance=within group variance.

K<-group
nk<-populationsize
mk<-samplesize
varwithin<-variance

#Covariate X1ki & X2ki.  They are correlated 0.3 at the community.
#and individual levels (see Prentice & Sheppard).

zK=mvrnorm(K,c(0,0),matrix(c(1,.3,.3,1),2,2))

cov=0.3*sqrt(varwithin)*1
covm=matrix(c(varwithin,cov,cov,1),2,2)

X1ki=matrix(0,nk,K)
X2ki=matrix(0,nk,K)

for(i in 1:K){
 znk=mvrnorm(nk,zK[i,],covm)
 X1ki[,i]=znk[,1]
```

```
   X2ki[,i]=znk[,2]
  }

  #Country specific frailties were generated as independent.
  #realized  values from a gamma  distribution with  mean 1.
  #and variance sigma^2. The mean of a gamma is shape*scale.
  #and the variance is shape*(scale^2).

  meanhk<-1
  varhk<-0.05
  shape<-(meanhk^2)/(varhk)
  scale<-(varhk)/(meanhk)
  hk<-rgamma(K,shape=shape,scale=scale)[]
  hk=t(matrix(rep(hk,nk),nrow=K,ncol=nk))

  #The disease events, yki, were generated by determining.
  #wether  a  uniform  random  variable  wass  less  than.
  #hk*exp(gamma0+beta1*X1ki+beta2*X2ki).

  gamma0<--3
  beta1<-0.2
  beta2<-0.2

  yki<-matrix(,nrow=nk,ncol=K)
  unif<-matrix(runif(nk*K,0,1),nrow=nk,ncol=K)

        yki=ifelse(unif<hk*exp(gamma0+beta1*X1ki+beta2*X2ki),1,0)

  ######################################.
  #selection of random sample of size mk#.
  #and organize data to apply functions #.
  #farem, fagrem and fpbm               #.
  ######################################.

  datalist<-list(matrix(,nrow=K,ncol=1))
  sampledatalist<-list(matrix(,nrow=K,ncol=1))
  ini<-1
  end<-mk
  data<-matrix(,nrow=mk*K,ncol=5)
  for (i in 1:K){

  datalist[[i]]<-
cbind(matrix(c(1:nk),nrow=nk,ncol=1),matrix(yki[,i],nrow=nk,ncol=1),matrix(X1ki[,i],nrow=nk,n
col=1),matrix(X2ki[,i],nrow=nk,ncol=1),matrix(c(i),nrow=nk,ncol=1))
  sampledatalist[[i]]<-datalist[[i]][as.matrix(sample(datalist[[i]][,1],mk)),]
  data[ini:end,]<-sampledatalist[[i]][]
                ini<-end+1
                end<-mk*(i+1)
  }

  O<-matrix(apply(yki,2,sum),nrow=K,ncol=1)

  ini<-1
  end<-mk
  datapop<-matrix(,nrow=mk*K,ncol=1)
  for (i in 1:K) {datapop[ini:end,1]<-O[i,]
                  ini<-end+1
                  end<-mk*(i+1)}

  datadatapop<-cbind(data,datapop,c(nk))
  datafin<-
data.frame(id=matrix(datadatapop[,1]),group=matrix(datadatapop[,5]),YIND=matrix(datadatapop[,
2]),O=matrix(datadatapop[,6]),n=matrix(datadatapop[,7]),X1ki=matrix(datadatapop[,3]),X2ki=mat
rix(datadatapop[,4]))}


  ######################.
  ##Simulation results##.
  ######################.


  fsimulationB<-function(seed,Niter,sigma2,K,N){
  set.seed(seed)
  count1=0
  result1=matrix(0,nrow=Niter,ncol=16)

  while(count1<Niter){
  tempdata1=fgenerate2(K,2000,N,sigma2)
  tol<-0.001
  gg<-glm(YIND~X1ki+X2ki,data=tempdata1,family=binomial)
  betaini<-
as.vector(c(matrix(gg$coefficients[1]),matrix(gg$coefficients[2]),matrix(gg$coefficients[3]))
)
  tempdata1a1<-farem2(tempdata1,tol,maxiter=50,betaini)
  tempdata1a2<-fagrem2(tempdata1,tol,maxiter=50,betaini)
  tempdata1a3<-fpbm2(tempdata1,tol,maxiter=50,betaini)
  count1=count1+1
```

```
  result1[count1,1:5]=matrix(c(t(tempdata1a1$betanew),tempdata1a1$sigma2re,tempdata1a1$iterN)
,nrow=1,ncol=5)
  result1[count1,6:10]=matrix(c(t(tempdata1a2$betanewa),tempdata1a2$sigma2am,tempdata1a2$iter
N),nrow=1,ncol=5)
  result1[count1,11:16]=matrix(c(t(tempdata1a3$betanewpb),tempdata1a3$sigma2re,tempdata1a3$si
gma2pb,tempdata1a3$iterN),nrow=1,ncol=6)
  print(count1)
  }

  list(result=result1,sigma2=sigma2,K=K,N=N)
  }


  #100 groups-100 sample size in each group.

  finalresultB100100B.25=fsimulationB(123,1000,.25,100,100)
  save(list=c("finalresultB100100B.25",".Random.seed"),file="100100B025.RData")
  savedseed=.Random.seed

  finalresultB100100B.5=fsimulationB(savedseed,1000,.5,100,100)
  save(list=c("finalresultB100100B.5",".Random.seed"),file="100100B05.RData")
  savedseed=.Random.seed

  finalresultB100100B1=fsimulationB(savedseed,1000,1,100,100)
  save(list=c("finalresultB100100B1",".Random.seed"),file="100100B1.RData")
  savedseed=.Random.seed

  finalresultB100100B2=fsimulationB(savedseed,1000,2,100,100)
  save(list=c("finalresultB100100B2",".Random.seed"),file="100100B2.RData")
  savedseed=.Random.seed

  finalresultB100100B4=fsimulationB(savedseed,1000,4,100,100)
  save(list=c("finalresultB100100B4",".Random.seed"),file="100100B4.RData")
  savedseed=.Random.seed

  finalresultB100100B8=fsimulationB(savedseed,1000,8,100,100)
  save(list=c("finalresultB100100B8",".Random.seed"),file="100100B8.RData")
  savedseed=.Random.seed

  finalresultB100100B16=fsimulationB(savedseed,1000,16,100,100)
  save(list=c("finalresultB100100B16",".Random.seed"),file="100100B16.RData")


  #100 groups-50 sample size in each group.

  finalresultB10050B.25=fsimulationB(123,1000,.25,100,50)
  save(list=c("finalresultB10050B.25",".Random.seed"),file="10050B025.RData")
  savedseed=.Random.seed

  finalresultB10050B.5=fsimulationB(savedseed,1000,.5,100,50)
  save(list=c("finalresultB10050B.5",".Random.seed"),file="10050B05.RData")
  savedseed=.Random.seed

  finalresultB10050B1=fsimulationB(savedseed,1000,1,100,50)
  save(list=c("finalresultB10050B1",".Random.seed"),file="10050B1.RData")
  savedseed=.Random.seed

  finalresultB10050B2=fsimulationB(savedseed,1000,2,100,50)
  save(list=c("finalresultB10050B2",".Random.seed"),file="10050B2.RData")
  savedseed=.Random.seed

  finalresultB10050B4=fsimulationB(savedseed,1000,4,100,50)
  save(list=c("finalresultB10050B4",".Random.seed"),file="10050B4.RData")
  savedseed=.Random.seed

  finalresultB10050B8=fsimulationB(savedseed,1000,8,100,50)
  save(list=c("finalresultB10050B8",".Random.seed"),file="10050B8.RData")
  savedseed=.Random.seed

  finalresultB10050B16=fsimulationB(savedseed,1000,16,100,50)
  save(list=c("finalresultB10050B16",".Random.seed"),file="10050B16.RData")


  #50 groups-100 sample size in each group.

  finalresultB50100B.25=fsimulationB(123,1000,.25,50,100)
  save(list=c("finalresultB50100B.25",".Random.seed"),file="50100B025.RData")
  savedseed=.Random.seed

  finalresultB50100B.5=fsimulationB(savedseed,1000,.5,50,100)
  save(list=c("finalresultB50100B.5",".Random.seed"),file="50100B05.RData")
  savedseed=.Random.seed

  finalresultB50100B1=fsimulationB(savedseed,1000,1,50,100)
  save(list=c("finalresultB50100B1",".Random.seed"),file="50100B1.RData")
  savedseed=.Random.seed
```

```
finalresultB50100B2=fsimulationB(savedseed,1000,2,50,100)
save(list=c("finalresultB50100B2",".Random.seed"),file="50100B2.RData")
savedseed=.Random.seed

finalresultB50100B4=fsimulationB(savedseed,1000,4,50,100)
save(list=c("finalresultB50100B4",".Random.seed"),file="50100B4.RData")
savedseed=.Random.seed

finalresultB50100B8=fsimulationB(savedseed,1000,8,50,100)
save(list=c("finalresultB50100B8",".Random.seed"),file="50100B8.RData")
savedseed=.Random.seed

finalresultB50100B16=fsimulationB(savedseed,1000,16,50,100)
save(list=c("finalresultB50100B16",".Random.seed"),file="50100B16.RData")


#50 groups-50 sample size in each group.

finalresultB5050B.25=fsimulationB(123,1000,.25,50,50)
save(list=c("finalresultB5050B.25",".Random.seed"),file="5050B025.RData")
savedseed=.Random.seed

finalresultB5050B.5=fsimulationB(savedseed,1000,.5,50,50)
save(list=c("finalresultB5050B.5",".Random.seed"),file="5050B05.RData")
savedseed=.Random.seed

finalresultB5050B1=fsimulationB(savedseed,1000,1,50,50)
save(list=c("finalresultB5050B1",".Random.seed"),file="5050B1.RData")
savedseed=.Random.seed

finalresultB5050B2=fsimulationB(savedseed,1000,2,50,50)
save(list=c("finalresultB5050B2",".Random.seed"),file="5050B2.RData")
savedseed=.Random.seed

finalresultB5050B4=fsimulationB(savedseed,1000,4,50,50)
save(list=c("finalresultB5050B4",".Random.seed"),file="5050B4.RData")
savedseed=.Random.seed

finalresultB5050B8=fsimulationB(savedseed,1000,8,50,50)
save(list=c("finalresultB5050B8",".Random.seed"),file="5050B8.RData")
savedseed=.Random.seed

finalresultB5050B16=fsimulationB(savedseed,1000,16,50,50)
save(list=c("finalresultB5050B16",".Random.seed"),file="5050B16.RData")

################################.
#Import results to a text file#.
################################.

#100 groups-100 sample size in each group.

write.table(finalresultB100100B.25$result,"dataB1var1.txt",row.names=F)  #variance 0.25.
write.table(finalresultB100100B.5$result,"dataB1var2.txt",row.names=F)   #variance 0.5.
write.table(finalresultB100100B1$result,"dataB1var3.txt",row.names=F)    #variance 1.
write.table(finalresultB100100B2$result,"dataB1var4.txt",row.names=F)    #variance 2.
write.table(finalresultB100100B4$result,"dataB1var5.txt",row.names=F)    #variance 4.
write.table(finalresultB100100B8$result,"dataB1var6.txt",row.names=F)    #variance 8.
write.table(finalresultB100100B16$result,"dataB1var7.txt",row.names=F)   #variance 16.


#50 groups-100 sample size in each group.

write.table(finalresultB50100B.25$result,"dataB2var1.txt",row.names=F)   #variance 0.25.
write.table(finalresultB50100B.5$result,"dataB2var2.txt",row.names=F)    #variance 0.5.
write.table(finalresultB50100B1$result,"dataB2var3.txt",row.names=F)     #variance 1.
write.table(finalresultB50100B2$result,"dataB2var4.txt",row.names=F)     #variance 2.
write.table(finalresultB50100B4$result,"dataB2var5.txt",row.names=F)     #variance 4.
write.table(finalresultB50100B8$result,"dataB2var6.txt",row.names=F)     #variance 8.
write.table(finalresultB50100B16$result,"dataB2var7.txt",row.names=F)    #variance 16.


#100 groups-50 sample size in each group.

write.table(finalresultB10050B.25$result,"dataB3var1.txt",row.names=F)   #variance 0.25.
write.table(finalresultB10050B.5$result,"dataB3var2.txt",row.names=F)    #variance 0.5.
write.table(finalresultB10050B1$result,"dataB3var3.txt",row.names=F)     #variance 1.
write.table(finalresultB10050B2$result,"dataB3var4.txt",row.names=F)     #variance 2.
write.table(finalresultB10050B4$result,"dataB3var5.txt",row.names=F)     #variance 4.
write.table(finalresultB10050B8$result,"dataB3var6.txt",row.names=F)     #variance 8.
write.table(finalresultB10050B16$result,"dataB3var7.txt",row.names=F)    #variance 16.

#100 groups-100 sample size in each group.

write.table(finalresultB5050B.25$result,"dataB4var1.txt",row.names=F)   #variance 0.25.
write.table(finalresultB5050B.5$result,"dataB4var2.txt",row.names=F)    #variance 0.5.
write.table(finalresultB5050B1$result,"dataB4var3.txt",row.names=F)     #variance 1.
write.table(finalresultB5050B2$result,"dataB4var4.txt",row.names=F)     #variance 2.
```

```
write.table(finalresultB5050B4$result,"dataB4var5.txt",row.names=F)     #variance 4.
write.table(finalresultB5050B8$result,"dataB4var6.txt",row.names=F)     #variance 8.
write.table(finalresultB5050B16$result,"dataB4var7.txt",row.names=F)    #variance 16.
```

A.3.3 ECC simulation program.

```
library(MASS)


#####################################################.
#####################################################.
##FUNCTION INDIVIDUAL RANDOM EFFECTS MODEL (farem)##.
##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMA^2   ##.
#####################################################.
#####################################################.

farem<-function(betanew,data) {

#We suposse the next structure in the dataset:Id (Identification of individual),.
#Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
#observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
#...,Zp (covariate Zp).

Dataprove=data

#K is the number of groups. We suppose that groups are ordered and they have all.
#the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
#N is the number of observations and p is the number of covariates.
N<-dim(Dataprove)[1]
K<-Dataprove[N,2]
p<-dim(Dataprove)[2]-5

ngr<-matrix(,nrow=1,ncol=K)
for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

gamma0<-betanew[1]
beta=betanew[-1]

#Individual outcome.
Yki<-matrix(,nrow=N,ncol=1)
Yki[,1]<-Dataprove[,3]

#Individual mean.
muki<-matrix(,nrow=N,ncol=1)
muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

#Matrix D for the IRM.
Dki<-matrix(,nrow=N,ncol=p+1)
Dki[,1]<-muki[,1]
for (j in 1:p){
Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
}

#Inverse variance-covariance matrix for the IRM.

        ##First, we compute sigma square.
        muki2<-matrix(,nrow=N,ncol=1)
        muki2[,1]<-muki[,1]^2

        Yaver<-matrix(,nrow=1,ncol=K)
        muk<-matrix(,nrow=1,ncol=K)
        phik<-matrix(,nrow=K,ncol=1)
        sigma2rek<-matrix(,nrow=K,ncol=1)

        ini<-1
        end<-ngr[1]
        for (i in 1:K) {
                Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                muk[1,i]<-sum(muki[ini:end])/ngr[i]
                phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                ini<-end+1
                end<-ngr[i+1]+end}

        sigma2re<-sum(sigma2rek[1:K])/K

        #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
        sumk<-matrix(,nrow=1,ncol=K)
        ini<-1
        end<-ngr[1]
        for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                        ini<-end+1
```

```
                        end<-ngr[i+1]+end}

    #Finally, we define the elements of the inverse of V.
    Vki<-list(matrix(,nrow=K,ncol=1))

    for (j in 1:K){
      Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

      for (i in 1:ngr[1,j]) {
        yy=1-(1+sigma2re)*muki[i]
              Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
        Vki[[j]][i,i]<-(1/(muki[i]*yy))-(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
         }
        Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
           }

  ############################.
  #Gradient, Hessian for IRM#.
  ############################.

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
         for (n in 1:(p+1)){
             Dkilist[[j]][,n]<-Dki[ini:end,n]}
             ini<-end+1
             end<-ngr[j+1]+end}

  #Gradient.
  ElementKGr<-list(matrix(,nrow=K,ncol=1))
  Grlist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {
         ElementKGr[[i]]<-t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]
         Grlist[[1]]<-Grlist[[1]]+ElementKGr[[i]]}

  Gr<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gr[i,1]<-Grlist[[1]][i]}

  #Hessian.
  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
         ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
         Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}

  #We return the Gradient (Gr), Hessian(Hs) and sigma^2(sigma2re).


  list(Gradient=Gr,Hessian=Hs,sigma2re=sigma2re)}


  ######################################################.
  ######################################################.
  ##FUNCTION AGGREGATED RANDOM EFFECTS MODEL (fagrem)##.
  ##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMA^2    ##.
  ######################################################.
  ######################################################.


  fagrem<-function(betanew,data) {

  #We suposse the next structure in the dataset:Id (Identification of individual),.
```

```
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).

  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]

  #Individual mean (muki).
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Variance for the ARM.
  muki2<-matrix(,nrow=N,ncol=1)
  muki2[,1]<-muki[,1]^2

  #Outcome for the ARM as defined in Sheppard and Prentice (Biometrics,1995).
  Y<-matrix(,nrow=1,ncol=K)

  muk<-matrix(,nrow=1,ncol=K)
  phik<-matrix(,nrow=K,ncol=1)
  #Matrix D for the ARM.
  Dk<-matrix(,nrow=K,ncol=p+1)

  #First, we compute sigma square.
  sigma2amk<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                  Y[1,i]<-((Dataprove[ini,4])/(Dataprove[ini,5]))
                  muk[1,i]<-sum(muki[ini:end])/ngr[i]
                  phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                  for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                  sigma2amk[i,]<-max(((Y[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]))),-100)
                  ini<-end+1
                  end<-ngr[i+1]+end}

  sigma2am<-sum(sigma2amk[1:K])/K

  #Finally, we define the variance.
  Vk<-matrix(,nrow=1,ncol=K)
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vk[1,i]<-sigma2am*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5])))+(muk[i]-
phik[i,])*(1/(Dataprove[ini,5]))
                  ini<-1
                  end<-ngr[1]}

  ############################.
  #Gradient, Hessian for ARM#.
  ############################.

  #Gradient.
  Dkt<-t(Dk)
  ElementKaGr<-list(matrix(,nrow=K,ncol=1))
  Gralist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {ElementKaGr[[i]]<-Dkt[,i]*(1/Vk[i])*(Y[i]-muk[i])
  Gralist[[1]]<-Gralist[[1]]+ElementKaGr[[i]]}

  Gra<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gra[i,1]<-Gralist[[1]][i]}

  #Hessian.
  ElementKaHs<-list(matrix(,nrow=K,ncol=1))
  Hsalist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKaHs[[i]]<--1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vk[i])%*%Dk[i,]
                  Hsalist[[1]]<-Hsalist[[1]]+ElementKaHs[[i]]}
```

```
  Hsa<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsa[i,j]<-Hsalist[[1]][i,j]}}

  #We return the Gradient (Gra), Hessian(Hsa) and sigma^2(sigma2am).

  list(Gradienta=Gra,Hessiana=Hsa,sigma2am=sigma2am)}


  ##########################################################.
  ##########################################################.
  ##FUNCTION POPULATION-BASED ESTIMATING EQUATION (fpbm)##.
  ##FOR COMPUTE THE GRADIENT, HESSIAN AND SIGMAS ^ 2    ##.
  ##########################################################.
  ##########################################################.

  fpbm<-function(betanew,data) {

  #We suposse the next structure in the dataset:Id (Identification of individual),.
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).

  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]


  ##########################################.
  #Gradient, Hessian for the individual part#.
  ##########################################.

  #Individual outcome.
  Yki<-matrix(,nrow=N,ncol=1)
  Yki[,1]<-Dataprove[,3]

  #Individual mean.
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Inverse variance-covariance matrix individual part.
          #First, we compute sigma square for the individual part.

          muki2<-matrix(,nrow=N,ncol=1)
          muki2[,1]<-muki[,1]^2

          Yaver<-matrix(,nrow=1,ncol=K)
          muk<-matrix(,nrow=1,ncol=K)
          phik<-matrix(,nrow=K,ncol=1)
          sigma2rek<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                  Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                  muk[1,i]<-sum(muki[ini:end])/ngr[i]
                  phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                  sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                  ini<-end+1
                  end<-ngr[i+1]+end}

          sigma2re<-sum(sigma2rek[1:K])/K

          #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
          sumk<-matrix(,nrow=1,ncol=K)
          ini<-1
          end<-ngr[1]
```

```
        for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                        ini<-end+1
                        end<-ngr[i+1]+end}

    #Finally we define the elements of the inverse of V.
    Vki<-list(matrix(,nrow=K,ncol=1))

    for (j in 1:K){
      Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

      for (i in 1:ngr[1,j]) {
        yy=1-(1+sigma2re)*muki[i]
         Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
              Vki[[j]][i,i]<-(1/(muki[i]*yy))-
(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
      }
        Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
           }

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
          for (n in 1:(p+1)){
              Dkilist[[j]][,n]<-Dki[ini:end,n]}
              ini<-end+1
              end<-ngr[j+1]+end}

  #Gradient Individual part.
  ElementKGr<-list(matrix(,nrow=K,ncol=1))
  Grlist<-list(matrix(0,nrow=(p+1),ncol=1))
  for (i in 1:K) {
          ElementKGr[[i]]<-t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]
          Grlist[[1]]<-Grlist[[1]]+ElementKGr[[i]]}

  Gr<-matrix(,nrow=(p+1),ncol=1)
        for (i in 1:(p+1)) {Gr[i,1]<-Grlist[[1]][i]}

  ##Hessian individual part.
  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
          ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
          Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}



  ##########################################.
  #Gradient, Hessian for the aggregated part#.
  ##########################################.


  #Outcome for the aggregated data model with combined analytical and aggregated models.
  Ybar<-matrix(,nrow=1,ncol=K)

  #Matrix D for the aggregated part.
  Dk<-matrix(,nrow=K,ncol=p+1)

        ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ybar[1,i]<-((Dataprove[ini,4]-sum(Yki[ini:end]))/(Dataprove[ini,5]-ngr[i]))
                  for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
```

```
                ini<-end+1
                end<-ngr[i+1]+end}

  #Sigma square aggregated part.
  sigma2pbk<-matrix(,nrow=K,ncol=1)
  ini<-1
  end<-ngr[1]
  for  (i in 1:K) {sigma2pbk[i,]<-max(((Ybar[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]-ngr[i]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]-ngr[i]))),0)
                ini<-end+1
                end<-ngr[i+1]+end}

  sigma2pb<-sum(sigma2pbk[1:K])/K

  #Variance aggregated part.
  Dkt<-t(Dk)
  Vkbar<-matrix(,nrow=1,ncol=K)
  ElementKarGr<-list(matrix(,nrow=K,ncol=1))
  Grarlist<-list(matrix(0,nrow=(p+1),ncol=1))
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vkbar[1,i]<-sigma2pb*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5]-
ngr[i])))+(muk[i]-phik[i,])*(1/(Dataprove[ini,5]-ngr[i]))
                ElementKarGr[[i]]<-Dkt[,i]*(1/Vkbar[i])*(Ybar[i]-muk[i])
                Grarlist[[1]]<-Grarlist[[1]]+ElementKarGr[[i]]
                ini<-end+1
                end<-ngr[i+1]+end}

  #Gradient aggregated part.
  Grar<-matrix(,nrow=(p+1),ncol=1)
  for (i in 1:(p+1)) {Grar[i,1]<-Grarlist[[1]][i]}

  #Hessian aggregated part.

  ElementKarHs<-list(matrix(,nrow=K,ncol=1))
  Hsarlist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKarHs[[i]]<--
1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vkbar[i])%*%Dk[i,]
                Hsarlist[[1]]<-Hsarlist[[1]]+ElementKarHs[[i]]}

  Hsar<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsar[i,j]<-Hsarlist[[1]][i,j]}}

  ##########################################################################.
  #Gradient, Hessian for the individual and aggregated part combination#.
  ##########################################################################.

  #Gradient PBEE.

  Grpb<-(Gr+Grar)

  #Hessian PBEE.

  Hspb<-(Hs+Hsar)

  #We return the Gradient (Grpb), Hessian(Hspb), sigma^2 individual.
  #part (sigma2re) and sigma^2 aggregated part (sigma2pb).

  list(Gradientpb=Grpb,Hessianpb=Hspb,sigma2re=sigma2re,sigma2pb=sigma2pb)}



  ##########################################################################.
  ##########################################################################.
  ##PARAMETER ESTIMATES USING THE NEWTON-RAPHSON ALGORITHM FOR THE IRM##.
  ##(function farem2, ARM (function fagrem2) AND PBEE (function fpbm2)##.
  ##########################################################################.
  ##########################################################################.

  #We suposse the next structure in the dataset:Id (Identification of individual),.
  #Group (Group number), Y (Individual outcome: 1 death, 0 alive), O (Population's.
  #observed deaths), n (Population at risk), Z1 (covariate Z1), Z2 (covariate Z2),.
  #...,Zp (covariate Zp).

  farem2<-function(data,tol,maxiter=100,betainitial){

  betanew<-betainitial
  betaold<-betanew+1
  itercount=0
  Hdim=length(betanew)^2

  #Solutions for the IRM.

  while(max(abs((betaold-betanew)/betaold))>tol){
       q<-farem(betanew,data)
       betaold<-betanew
```

```
     if(sum(is.finite(q$Hessian))<Hdim) itercount=999
     if(itercount>maxiter) break
     betanew<-betanew-ginv(q$Hessian)%*%q$Gradient
     itercount=itercount+1
     if(is.na(q$sigma2re)) itercount=100
     if(itercount>maxiter) break
     if(q$sigma2re>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanew=betanew,sigma2re=q$sigma2re, iterN=itercount)
      }

fagrem2<-function(data,tol,maxiter=100,betainitial){

betanewa<-betainitial
betaold<-betanewa+1
itercount=0
Hdim=length(betanewa)^2

#Solutions for the ARM.

while(max(abs((betaold-betanewa)/betaold))>tol){
     d<-fagrem(betanewa,data)
     betaold<-betanewa
     if(sum(is.finite(d$Hessiana))<Hdim) itercount=999
     if(itercount>maxiter) break
     betanewa<-betanewa-ginv(d$Hessiana)%*%d$Gradienta
     itercount=itercount+1
     if(is.na(d$sigma2am)) itercount=100
     if(itercount>maxiter) break
     if(d$sigma2am>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanewa=betanewa,sigma2am=d$sigma2am, iterN=itercount)
      }

fpbm2<-function(data,tol,maxiter=100,betainitial){

betanewpb<-betainitial
betaold<-betanewpb+1
itercount=0
Hdim=length(betanewpb)^2

#Solutions for the PBEE.

while(max(abs((betaold-betanewpb)/betaold))>tol){
     e<-fpbm(betanewpb,data)
     betaold<-betanewpb
     if(sum(is.finite(e$Hessianpb))<Hdim) itercount=999
     if(itercount>maxiter) break
     betanewpb<-betanewpb-ginv(e$Hessianpb)%*%e$Gradientpb
     itercount=itercount+1
     if(is.na(max(e$sigma2re,e$sigma2pb))) itercount=100
     if(itercount>maxiter) break
     if(max(e$sigma2re,e$sigma2pb)>50) itercount=100
     if(itercount>maxiter) break
     }
list(betanewpb=betanewpb,sigma2re=e$sigma2re,sigma2pb=e$sigma2pb, iterN=itercount)
      }

################################################.
################################################.
##FUNCTION FOR SIMULATE THE DATA (fgenerate2)##.
##IN THE EXTENDED CONFOUNDING CASE (ECC)     ##.
################################################.
################################################.

fgenerate2<-function(group,populationsize,samplesize,var1=.85,var2=4){

#group=number of groups, populationsize= population size.
#samplesize= sample size, var1 within group-variance X1Ki,.
#var2=within-group variance X2ki.

K<-group
nk<-populationsize
mk<-samplesize

#Covariate X1ki & X2ki.  They are correlated 0.3 at the community.
#and individual levels (see Prentice & Sheppard).

cov1=0.3*3.4*0.25
zK=mvrnorm(K,c(0,0),matrix(c(3.4,cov1,cov1,0.25),2,2))

cov=0.3*sqrt(var1*var2)
covm=matrix(c(var1,cov,cov,var2),2,2)

X1ki=matrix(0,nk,K)
```

```
  X2ki=matrix(0,nk,K)

  for(i in 1:K){
   znk=mvrnorm(nk,zK[i,],covm)
   X1ki[,i]=znk[,1]
   X2ki[,i]=znk[,2]
  }

  #Country specific frailties were generated as independent.
  #realized  values from a gamma  distribution with  mean 1.
  #and variance sigma^2. The mean of a gamma is shape*scale.
  #and the variance is shape*(scale^2).

  meanhk<-1
  varhk<-0.05
  shape<-(meanhk^2)/(varhk)
  scale<-(varhk)/(meanhk)
  hk<-rgamma(K,shape=shape,scale=scale)[]
  hk=t(matrix(rep(hk,nk),nrow=K,ncol=nk))

  #The disease events, yki, were generated by determining.
  #wether  a  uniform  random  variable  wass  less  than.
  #hk*exp(gamma0+beta1*X1ki+beta2*X2ki).

  gamma0<--3
  beta1<-0.2
  beta2<-0.2

  yki<-matrix(,nrow=nk,ncol=K)
  unif<-matrix(runif(nk*K,0,1),nrow=nk,ncol=K)

        yki=ifelse(unif<hk*exp(gamma0+beta1*X1ki+beta2*X2ki),1,0)

  ########################################.
  #selection of random sample of size mk#.
  #and organize data to apply functions #.
  #farem, fagrem and fpbm               #.
  ########################################.

  datalist<-list(matrix(,nrow=K,ncol=1))
  sampledatalist<-list(matrix(,nrow=K,ncol=1))
  ini<-1
  end<-mk
  data<-matrix(,nrow=mk*K,ncol=5)
  for (i in 1:K){

  datalist[[i]]<-
cbind(matrix(c(1:nk),nrow=nk,ncol=1),matrix(yki[,i],nrow=nk,ncol=1),matrix(X1ki[,i],nrow=nk,n
col=1),matrix(X2ki[,i],nrow=nk,ncol=1),matrix(c(i),nrow=nk,ncol=1))
  sampledatalist[[i]]<-datalist[[i]][as.matrix(sample(datalist[[i]][,1],mk)),]
  data[ini:end,]<-sampledatalist[[i]][]
                 ini<-end+1
                 end<-mk*(i+1)
  }

  O<-matrix(apply(yki,2,sum),nrow=K,ncol=1)

  ini<-1
  end<-mk
  datapop<-matrix(,nrow=mk*K,ncol=1)
  for (i in 1:K) {datapop[ini:end,1]<-O[i,]
                  ini<-end+1
                  end<-mk*(i+1)}

  datadatapop<-cbind(data,datapop,c(nk))
  datafin<-
data.frame(id=matrix(datadatapop[,1]),group=matrix(datadatapop[,5]),YIND=matrix(datadatapop[,
2]),O=matrix(datadatapop[,6]),n=matrix(datadatapop[,7]),X1ki=matrix(datadatapop[,3]),X2ki=mat
rix(datadatapop[,4]))}


  ######################.
  ##Simulation results##.
  ######################.

  fsimulationC<-function(seed,Niter,K,N){
  set.seed(seed)
  count1=0
  result1=matrix(0,nrow=Niter,ncol=16)

  while(count1<Niter){
  tempdata1=fgenerate2(K,2000,N)
  tol<-0.001
  gg<-glm(YIND~X1ki+X2ki,data=tempdata1,family=binomial)
  betaini<-
as.vector(c(matrix(gg$coefficients[1]),matrix(gg$coefficients[2]),matrix(gg$coefficients[3]))
)
```

```
  tempdata1a1<-farem2(tempdata1,tol,maxiter=50,betaini)
  tempdata1a2<-fagrem2(tempdata1,tol,maxiter=50,betaini)
  tempdata1a3<-fpbm2(tempdata1,tol,maxiter=50,betaini)
  count1=count1+1
  result1[count1,1:5]=matrix(c(t(tempdata1a1$betanew),tempdata1a1$sigma2re,tempdata1a1$iterN)
,nrow=1,ncol=5)
  result1[count1,6:10]=matrix(c(t(tempdata1a2$betanewa),tempdata1a2$sigma2am,tempdata1a2$iter
N),nrow=1,ncol=5)
  result1[count1,11:16]=matrix(c(t(tempdata1a3$betanewpb),tempdata1a3$sigma2re,tempdata1a3$si
gma2pb,tempdata1a3$iterN),nrow=1,ncol=6)
  print(count1)
  }

  list(result=result1,K=K,N=N)
  }

  #100 groups-100 sample size in each group.
  finalresultC100100=fsimulationC(123,1000,100,100)
  save(list=c("finalresultC100100",".Random.seed"),file="100100C.RData")

  #100 groups-50 sample size in each group.
  finalresultC10050=fsimulationC(123,1000,100,50)
  save(list=c("finalresultC10050",".Random.seed"),file="10050C.RData")

  #50 groups-100 sample size in each group.
  finalresultC50100=fsimulationC(123,1000,50,100)
  save(list=c("finalresultC100100",".Random.seed"),file="50100C.RData")

  #50 groups-50 sample size in each group.
  finalresultC5050=fsimulationC(123,1000,50,50)
  save(list=c("finalresultC100100",".Random.seed"),file="5050C.RData")
```

A.3.4 NCC coverage program.

```
  library(MASS)

  ############################################################.
  ############################################################.
  ##FUNCTION INDIVIDUAL RANDOM EFFECTS MODEL (faremcoverage)##.
  ##FOR COMPUTE NAIVE AND SANDWICH ESTIMATOR #################.
  ############################################################.
  ############################################################.

  faremcoverage<-function(betain,data) {

  betanew=as.vector(betain[1,],mode="numeric")
  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta=betanew[-1]

  #Individual outcome.
  Yki<-matrix(,nrow=N,ncol=1)
  Yki[,1]<-Dataprove[,3]

  #Individual mean.
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Matrix D for the IRM.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Inverse variance-covariance matrix for the IRM.

          ##First, we compute sigma square.
          muki2<-matrix(,nrow=N,ncol=1)
          muki2[,1]<-muki[,1]^2

          Yaver<-matrix(,nrow=1,ncol=K)
          muk<-matrix(,nrow=1,ncol=K)
```

```
         phik<-matrix(,nrow=K,ncol=1)
         sigma2rek<-matrix(,nrow=K,ncol=1)

         ini<-1
         end<-ngr[1]
         for (i in 1:K) {
                Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                muk[1,i]<-sum(muki[ini:end])/ngr[i]
                phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                ini<-end+1
                end<-ngr[i+1]+end}

         sigma2re<-sum(sigma2rek[1:K])/K

         #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
         sumk<-matrix(,nrow=1,ncol=K)
         ini<-1
         end<-ngr[1]
         for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                         ini<-end+1
                         end<-ngr[i+1]+end}

     #Finally, we define the elements of the inverse of V.
     Vki<-list(matrix(,nrow=K,ncol=1))

     for (j in 1:K){
       Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

       for (i in 1:ngr[1,j]) {
         yy=1-(1+sigma2re)*muki[i]
               Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
         Vki[[j]][i,i]<-(1/(muki[i]*yy))-(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
          }
         Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
            }

  ########################################.
  #NAIVE AND SANDWICH ESTIMATORS for IRM#.
  ########################################.

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
         for (n in 1:(p+1)){
             Dkilist[[j]][,n]<-Dki[ini:end,n]}
             ini<-end+1
             end<-ngr[j+1]+end}

  ElementKM<-list(matrix(,nrow=K,ncol=1))
  Mlist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))

  for (i in 1:K) {
         ElementKM[[i]]<-
t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]%*%t(Ykiminusmuki[[i]])%*%Vki[[i]]%*%Dkilist[[i
]]
         Mlist[[1]]<-Mlist[[1]]+ElementKM[[i]]}

  M<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {M[i,j]<-Mlist[[1]][i,j]}}

  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
```

```
  for (i in 1:K) {
         ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
         Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}

  naive=ginv(-Hs)
  robust=naive%*%M%*%naive

  list(naive=naive, robust=robust)
  }

  ##############################################################.
  ##############################################################.
  ##FUNCTION AGGREGATED RANDOM EFFECTS MODEL (fagremcoverage)##.
  ##FOR COMPUTE NAIVE AND SANDWICH ESTIMATOR ##################.
  ##############################################################.
  ##############################################################.

  fagremcoverage<-function(betain,data) {

  betanew=as.vector(betain[1,],mode="numeric")
  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]

  #Individual mean (muki).
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Variance for the ARM.
  muki2<-matrix(,nrow=N,ncol=1)
  muki2[,1]<-muki[,1]^2

  #Outcome for the ARM as defined in Sheppard and Prentice (Biometrics,1995).
  Y<-matrix(,nrow=1,ncol=K)

  muk<-matrix(,nrow=1,ncol=K)
  phik<-matrix(,nrow=K,ncol=1)
  #Matrix D for the ARM.
  Dk<-matrix(,nrow=K,ncol=p+1)

  #First, we compute sigma square.
  sigma2amk<-matrix(,nrow=K,ncol=1)

         ini<-1
         end<-ngr[1]
         for (i in 1:K) {
                Y[1,i]<-((Dataprove[ini,4])/(Dataprove[ini,5]))
                muk[1,i]<-sum(muki[ini:end])/ngr[i]
                phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                sigma2amk[i,]<-max(((Y[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]))),-100)
                ini<-end+1
                end<-ngr[i+1]+end}

  sigma2am<-sum(sigma2amk[1:K])/K

  #Finally, we define the variance.
  Vk<-matrix(,nrow=1,ncol=K)
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vk[1,i]<-sigma2am*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5])))+(muk[i]-
phik[i,])*(1/(Dataprove[ini,5]))
                ini<-1
                end<-ngr[1]}
```

```
  ######################################.
  #NAIVE AND SANDWICH ESTIMATORS for ARM#.
  ######################################.

  Dkt<-t(Dk)
  ElementKMa<-list(matrix(,nrow=K,ncol=1))
  Malist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKMa[[i]]<-
matrix(Dkt[,i],(p+1),1)%*%matrix(Dkt[,i],1,(p+1))*((1/Vk[i])*(Y[i]-muk[i]))^2
  Malist[[1]]<-Malist[[1]]+ElementKMa[[i]]}

  Ma<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Ma[i,j]<-Malist[[1]][i,j]}}

  ElementKaHs<-list(matrix(,nrow=K,ncol=1))
  Hsalist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKaHs[[i]]<--1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vk[i])%*%Dk[i,]
                  Hsalist[[1]]<-Hsalist[[1]]+ElementKaHs[[i]]}

  Hsa<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsa[i,j]<-Hsalist[[1]][i,j]}}

  naive=ginv(-Hsa)
  robust=naive%*%Ma%*%naive

  list(naive=naive, robust=robust)
  }


  ###################################################################.
  ###################################################################.
  ##FUNCTION POPULATION-BASED ESTIMATING EQUATION (fpbmcoverage)##.
  ##FOR COMPUTE NAIVE AND SANDWICH ESTIMATOR ####################.
  ###################################################################.
  ###################################################################.

  fpbmcoverage<-function(betain,data) {

  betanew=as.vector(betain[1,],mode="numeric")
  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]

  #Individual outcome.
  Yki<-matrix(,nrow=N,ncol=1)
  Yki[,1]<-Dataprove[,3]

  #Individual mean.
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Inverse variance-covariance matrix individual part.
  #First, we compute sigma square for the individual part.

          muki2<-matrix(,nrow=N,ncol=1)
          muki2[,1]<-muki[,1]^2

          Yaver<-matrix(,nrow=1,ncol=K)
          muk<-matrix(,nrow=1,ncol=K)
          phik<-matrix(,nrow=K,ncol=1)
          sigma2rek<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
```

```
                Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                muk[1,i]<-sum(muki[ini:end])/ngr[i]
                phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-100)
                ini<-end+1
                end<-ngr[i+1]+end}

        sigma2re<-sum(sigma2rek[1:K])/K

        #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
        sumk<-matrix(,nrow=1,ncol=K)
        ini<-1
        end<-ngr[1]
        for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                        ini<-end+1
                        end<-ngr[i+1]+end}

    #Finally we define the elements of the inverse of V.
    Vki<-list(matrix(,nrow=K,ncol=1))

    for (j in 1:K){
      Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

      for (i in 1:ngr[1,j]) {
        yy=1-(1+sigma2re)*muki[i]
         Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
               Vki[[j]][i,i]<-(1/(muki[i]*yy))-
(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
       }
        Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
           }

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                 ini<-end+1
                end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
         for (n in 1:(p+1)){
             Dkilist[[j]][,n]<-Dki[ini:end,n]}
             ini<-end+1
             end<-ngr[j+1]+end}


  #Outcome for the aggregated data model with combined analytical and aggregated models.
  Ybar<-matrix(,nrow=1,ncol=K)

  #Matrix D for the aggregated part.
  Dk<-matrix(,nrow=K,ncol=p+1)

        ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ybar[1,i]<-((Dataprove[ini,4]-sum(Yki[ini:end]))/(Dataprove[ini,5]-ngr[i]))
                for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                ini<-end+1
                end<-ngr[i+1]+end}

  Dkt<-t(Dk)

  #Sigma square aggregated part.
  sigma2pbk<-matrix(,nrow=K,ncol=1)
  ini<-1
  end<-ngr[1]
  for  (i in 1:K) {sigma2pbk[i,]<-max(((Ybar[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]-ngr[i]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]-ngr[i]))),0)
                ini<-end+1
                end<-ngr[i+1]+end}
```

```
  sigma2pb<-sum(sigma2pbk[1:K])/K

  Vkbar<-matrix(,nrow=1,ncol=K)
  Mpb=matrix(0,nrow=(p+1),ncol=(p+1))
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {
  Vkbar[1,i]<-sigma2pb*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5]-ngr[i])))+(muk[i]-
phik[i,])*(1/(Dataprove[ini,5]-ngr[i]))
  junkmat=diag(c(rep(0,ngr[i]),1/Vkbar[1,i]))
  junkmat[1:ngr[i],1:ngr[i]]=Vki[[i]]
  sub=t(rbind(Dkilist[[i]],Dk[i,]))%*%junkmat%*%rbind(Ykiminusmuki[[i]],Ybar[i]-muk[i])
  Mpb=Mpb+sub%*%t(sub)
  ini<-end+1
  end<-ngr[i+1]+end
        }

  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
         ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
         Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}

  ElementKarHs<-list(matrix(,nrow=K,ncol=1))
  Hsarlist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKarHs[[i]]<--
1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vkbar[i])%*%Dk[i,]
                 Hsarlist[[1]]<-Hsarlist[[1]]+ElementKarHs[[i]]}

  Hsar<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsar[i,j]<-Hsarlist[[1]][i,j]}}

  Hspb<-(Hs+Hsar)

  naive=ginv(-Hspb)
  robust=naive%*%Mpb%*%naive

  list(naive=naive, robust=robust)
  }


  ################################################################.
  ################################################################.
  ##CONFIDENCE INTERVAL NAIVE AND SANDWICH FOR THE IRM########.
  ##(function farem2coverage, ARM (function fagrem2coverage)##.
  ##AND PBEE (function fpbm2coverage)#########################.
  ################################################################.
  ################################################################.

  farem2coverage<-function(data,datavar,beta1=.2){

  result=rep(NA,21)
  result[1]=datavar[2]

  if (datavar[5]<51) {

        q<-faremcoverage(datavar[1:3],data)
        naive=q$naive
        robust=q$robust

  if (naive[2,2]<0){
  result[2:3]=rep(-1,2)
  result[16]=-1}

  if (robust[2,2]<0){
  result[4:5]=rep(-1,2)
  result[17]=-1}

  if (naive[2,2]>=0){
  ll=(datavar[2]-1.96*sqrt(naive[2,2]))
  ul=(datavar[2]+1.96*sqrt(naive[2,2]))
  result[2:3]=c(ll,ul)
  result[16]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

  if (robust[2,2]>=0){
  ll=(datavar[2]-1.96*sqrt(robust[2,2]))
  ul=(datavar[2]+1.96*sqrt(robust[2,2]))
  result[4:5]=c(ll,ul)
  result[17]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

  }
```

```
else {
result[c(2:5,16:17)]=rep(-1,6)
}

unlist(result)
      }

fagrem2coverage<-function(data,datavar,beta1=.2){

result=rep(NA,21)
result[6]=datavar[7]

if (datavar[10]<51) {

      d<-fagremcoverage(datavar[6:8],data)

      naive=d$naive
      robust=d$robust

if (naive[2,2]<0){
result[7:8]=rep(-1,2)
result[18]=-1}

if (robust[2,2]<0){
result[9:10]=rep(-1,2)
result[19]=-1}

if (naive[2,2]>=0){
ll=(datavar[7]-1.96*sqrt(naive[2,2]))
ul=(datavar[7]+1.96*sqrt(naive[2,2]))
result[7:8]=c(ll,ul)
result[18]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

if (robust[2,2]>=0){
ll=(datavar[7]-1.96*sqrt(robust[2,2]))
ul=(datavar[7]+1.96*sqrt(robust[2,2]))
result[9:10]=c(ll,ul)
result[19]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

}

else {
result[c(7:10,18:19)]=rep(-1,6)
}

unlist(result)
      }

fpbm2coverage<-function(data,datavar,beta1=.2){

result=rep(NA,21)
result[11]=datavar[12]

if (datavar[16]<51) {

      pb<-fpbmcoverage(datavar[11:13],data)

      naive=pb$naive
      robust=pb$robust

if (naive[2,2]<0){
result[12:13]=rep(-1,2)
result[20]=-1}

if (robust[2,2]<0){
result[14:15]=rep(-1,2)
result[21]=-1}

if (naive[2,2]>=0){
ll=(datavar[12]-1.96*sqrt(naive[2,2]))
ul=(datavar[12]+1.96*sqrt(naive[2,2]))
result[12:13]=c(ll,ul)
result[20]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

if (robust[2,2]>=0){
ll=(datavar[12]-1.96*sqrt(robust[2,2]))
ul=(datavar[12]+1.96*sqrt(robust[2,2]))
result[14:15]=c(ll,ul)
result[21]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

}

else {
result[c(12:15,20:21)]=rep(-1,6)
}
```

```
  unlist(result)
        }


  ##############################################.
  ##############################################.
  ##FUNCTION FOR SIMULATE THE DATA (fgenerate2)##.
  ##IN THE NON CONFOUNDING CASE (NCC)          ##.
  ##############################################.
  ##############################################.

  fgenerate2<-function(group,populationsize,samplesize,variance){

  #group=number of groups, populationsize= population size.
  #samplesize= sample size, variance=within group variance.

  K<-group
  nk<-populationsize
  mk<-samplesize
  varwithin<-variance

  #Covariate X1ki with ratio (within variance)/(between variance) equal varwithin/1.

  Z1kg<-rnorm(K,0,sqrt(1))

  X1ki<-t(matrix(rnorm(nk*K,Z1kg,sqrt(varwithin)),nrow=K,ncol=nk))

  #Covariate X2ki with ratio (within variance)/(between variance) equal 1/1.

  Z2kg<-matrix(rnorm(K,0,sqrt(1)),nrow=K,ncol=1)

  X2ki<-t(matrix(rnorm(nk*K,Z2kg,sqrt(1)),nrow=K,ncol=nk))

  #Country specific frailties were generated as independent.
  #realized  values from a gamma  distribution with  mean 1.
  #and variance sigma^2. The mean of a gamma is shape*scale.
  #and the variance is shape*(scale^2).

  meanhk<-1
  varhk<-0.05
  shape<-(meanhk^2)/(varhk)
  scale<-(varhk)/(meanhk)
  hk<-rgamma(K,shape=shape,scale=scale)[]
  hk=t(matrix(rep(hk,nk),nrow=K,ncol=nk))

  #The disease events, yki, were generated by determining.
  #wether  a  uniform  random  variable  wass  less  than.
  #hk*exp(gamma0+beta1*X1ki+beta2*X2ki).

  gamma0<--3
  beta1<-0.2
  beta2<-0.2

  yki<-matrix(,nrow=nk,ncol=K)
  unif<-matrix(runif(nk*K,0,1),nrow=nk,ncol=K)

        yki=ifelse(unif<hk*exp(gamma0+beta1*X1ki+beta2*X2ki),1,0)

  ########################################.
  #selection of random sample of size mk#.
  #and organize data to apply functions #.
  #farem, fagrem and fpbm               #.
  ########################################.

  datalist<-list(matrix(,nrow=K,ncol=1))
  sampledatalist<-list(matrix(,nrow=K,ncol=1))
  ini<-1
  end<-mk
  data<-matrix(,nrow=mk*K,ncol=5)
  for (i in 1:K){

  datalist[[i]]<-
cbind(matrix(c(1:nk),nrow=nk,ncol=1),matrix(yki[,i],nrow=nk,ncol=1),matrix(X1ki[,i],nrow=nk,n
col=1),matrix(X2ki[,i],nrow=nk,ncol=1),matrix(c(i),nrow=nk,ncol=1))
  sampledatalist[[i]]<-datalist[[i]][as.matrix(sample(datalist[[i]][,1],mk)),]
  data[ini:end,]<-sampledatalist[[i]][]
                 ini<-end+1
                 end<-mk*(i+1)
  }

  O<-matrix(apply(yki,2,sum),nrow=K,ncol=1)

  ini<-1
  end<-mk
  datapop<-matrix(,nrow=mk*K,ncol=1)
  for (i in 1:K) {datapop[ini:end,1]<-O[i,]
                  ini<-end+1
```

```
                  end<-mk*(i+1)}

  datadatapop<-cbind(data,datapop,c(nk))
  datafin<-
data.frame(id=matrix(datadatapop[,1]),group=matrix(datadatapop[,5]),YIND=matrix(datadatapop[,
2]),O=matrix(datadatapop[,6]),n=matrix(datadatapop[,7]),X1ki=matrix(datadatapop[,3]),X2ki=mat
rix(datadatapop[,4]))}


  ####################.
  ##Coverage results##.
  ####################.

  fsimulationAcoverage<-function(seed,Niter,sigma2,K,N,datavar){
  set.seed(seed)
  count1=0
  result1=matrix(NA,Niter,21)

  while(count1<Niter){
  tempdata1=fgenerate2(K,2000,N,sigma2)
  count1=count1+1
  a1<-farem2coverage(tempdata1,datavar[count1,])
  a2<-fagrem2coverage(tempdata1,datavar[count1,])
  a3<-fpbm2coverage(tempdata1,datavar[count1,])
  result1[count1,c(1:5,16:17)]=a1[c(1:5,16:17)]
  result1[count1,c(6:10,18:19)]=a2[c(6:10,18:19)]
  result1[count1,c(11:15,20:21)]=a3[c(11:15,20:21)]
  print(count1)
  }

  list(result=result1,sigma2=sigma2,K=K,N=N)
  }

  ########################################.
  #100 groups-100 sample size in each group#.
  ########################################.

  #The next  text files  are  the  results  files  from  the  simulation runs.
  #to obtain parameter estimates in each variation ratio for the 100-100 case.
  #For example, dataA1var1.txt is from finalresultA100100A.25$result.

  dataA1var1<-read.table("dataA1var1.txt",header=T) #Variance 0.25.
  dataA1var2<-read.table("dataA1var2.txt",header=T) #Variance 0.5.
  dataA1var3<-read.table("dataA1var3.txt",header=T) #Variance 1.
  dataA1var4<-read.table("dataA1var4.txt",header=T) #Variance 2.
  dataA1var5<-read.table("dataA1var5.txt",header=T) #Variance 4.
  dataA1var6<-read.table("dataA1var6.txt",header=T) #Variance 8.
  dataA1var7<-read.table("dataA1var7.txt",header=T) #Variance 16.


  coverage100100A.25=fsimulationAcoverage(123,1000,.25,100,100,dataA1var1)
  save(list=c("coverage100100A.25",".Random.seed"),file="100100coverageA025.RData")
  savedseed=.Random.seed

  coverage100100A.5=fsimulationAcoverage(savedseed,1000,.5,100,100,dataA1var2)
  save(list=c("coverage100100A.5",".Random.seed"),file="100100coverageA05.RData")
  savedseed=.Random.seed

  coverage100100A1=fsimulationAcoverage(savedseed,1000,1,100,100,dataA1var3)
  save(list=c("coverage100100A1",".Random.seed"),file="100100coverageA1.RData")
  savedseed=.Random.seed

  coverage100100A2=fsimulationAcoverage(savedseed,1000,2,100,100,dataA1var4)
  save(list=c("coverage100100A2",".Random.seed"),file="100100coverageA2.RData")
  savedseed=.Random.seed

  coverage100100A4=fsimulationAcoverage(savedseed,1000,4,100,100,dataA1var5)
  save(list=c("coverage100100A4",".Random.seed"),file="100100coverageA4.RData")
  savedseed=.Random.seed

  coverage100100A8=fsimulationAcoverage(savedseed,1000,8,100,100,dataA1var6)
  save(list=c("coverage100100A8",".Random.seed"),file="100100coverageA8.RData")
  savedseed=.Random.seed

  coverage100100A16=fsimulationAcoverage(savedseed,1000,16,100,100,dataA1var7)
  save(list=c("coverage100100A16",".Random.seed"),file="100100coverageA16.RData")

  ########################################.
  #50 groups-100 sample size in each group #.
  ########################################.

  #The next  text files  are  the  results  files  from  the  simulation runs.
  #to obtain parameter estimates in each variation ratio for the 50-100 case.

  dataA2var1<-read.table("dataA2var1.txt",header=T) #Variance 0.25.
  dataA2var2<-read.table("dataA2var2.txt",header=T) #Variance 0.5.
  dataA2var3<-read.table("dataA2var3.txt",header=T) #Variance 1.
```

```
dataA2var4<-read.table("dataA2var4.txt",header=T) #Variance 2.
dataA2var5<-read.table("dataA2var5.txt",header=T) #Variance 4.
dataA2var6<-read.table("dataA2var6.txt",header=T) #Variance 8.
dataA2var7<-read.table("dataA2var7.txt",header=T) #Variance 16.


coverage50100A.25=fsimulationAcoverage(123,1000,.25,50,100,dataA2var1)
save(list=c("coverage50100A.25",".Random.seed"),file="50100coverageA025.RData")
savedseed=.Random.seed

coverage50100A.5=fsimulationAcoverage(savedseed,1000,.5,50,100,dataA2var2)
save(list=c("coverage50100A.5",".Random.seed"),file="50100coverageA05.RData")
savedseed=.Random.seed

coverage50100A1=fsimulationAcoverage(savedseed,1000,1,50,100,dataA2var3)
save(list=c("coverage50100A1",".Random.seed"),file="50100coverageA1.RData")
savedseed=.Random.seed

coverage50100A2=fsimulationAcoverage(savedseed,1000,2,50,100,dataA2var4)
save(list=c("coverage50100A2",".Random.seed"),file="50100coverageA2.RData")
savedseed=.Random.seed

coverage50100A4=fsimulationAcoverage(savedseed,1000,4,50,100,dataA2var5)
save(list=c("coverage50100A4",".Random.seed"),file="50100coverageA4.RData")
savedseed=.Random.seed

coverage50100A8=fsimulationAcoverage(savedseed,1000,8,50,100,dataA2var6)
save(list=c("coverage50100A8",".Random.seed"),file="50100coverageA8.RData")
savedseed=.Random.seed

coverage50100A16=fsimulationAcoverage(savedseed,1000,16,50,100,dataA2var7)
save(list=c("coverage50100A16",".Random.seed"),file="50100coverageA16.RData")

########################################.
#100 groups-50 sample size in each group#.
########################################.

#The next  text files  are  the  results  files  from  the  simulation runs.
#to obtain parameter estimates in each variation ratio for the 50-100 case.

dataA3var1<-read.table("dataA3var1.txt",header=T) #Variance 0.25.
dataA3var2<-read.table("dataA3var2.txt",header=T) #Variance 0.5.
dataA3var3<-read.table("dataA3var3.txt",header=T) #Variance 1.
dataA3var4<-read.table("dataA3var4.txt",header=T) #Variance 2.
dataA3var5<-read.table("dataA3var5.txt",header=T) #Variance 4.
dataA3var6<-read.table("dataA3var6.txt",header=T) #Variance 8.
dataA3var7<-read.table("dataA3var7.txt",header=T) #Variance 16.


coverage10050A.25=fsimulationAcoverage(123,1000,.25,100,50,dataA3var1)
save(list=c("coverage10050A.25",".Random.seed"),file="10050coverageA025.RData")
savedseed=.Random.seed

coverage10050A.5=fsimulationAcoverage(savedseed,1000,.5,100,50,dataA3var2)
save(list=c("coverage10050A.5",".Random.seed"),file="10050coverageA05.RData")
savedseed=.Random.seed

coverage10050A1=fsimulationAcoverage(savedseed,1000,1,100,50,dataA3var3)
save(list=c("coverage10050A1",".Random.seed"),file="10050coverageA1.RData")
savedseed=.Random.seed

coverage10050A2=fsimulationAcoverage(savedseed,1000,2,100,50,dataA3var4)
save(list=c("coverage10050A2",".Random.seed"),file="10050coverageA2.RData")
savedseed=.Random.seed

coverage10050A4=fsimulationAcoverage(savedseed,1000,4,100,50,dataA3var5)
save(list=c("coverage10050A4",".Random.seed"),file="10050coverageA4.RData")
savedseed=.Random.seed

coverage10050A8=fsimulationAcoverage(savedseed,1000,8,100,50,dataA3var6)
save(list=c("coverage10050A8",".Random.seed"),file="10050coverageA8.RData")
savedseed=.Random.seed

coverage10050A16=fsimulationAcoverage(savedseed,1000,16,100,50,dataA3var7)
save(list=c("coverage10050A16",".Random.seed"),file="10050coverageA16.RData")

########################################.
#50 groups-50 sample size in each group#.
########################################.

#The next  text files  are  the  results  files  from  the  simulation runs.
#to obtain parameter estimates in each variation ratio for the 50-100 case.

dataA4var1<-read.table("dataA4var1.txt",header=T) #Variance 0.25.
dataA4var2<-read.table("dataA4var2.txt",header=T) #Variance 0.5.
dataA4var3<-read.table("dataA4var3.txt",header=T) #Variance 1.
dataA4var4<-read.table("dataA4var4.txt",header=T) #Variance 2.
```

```
dataA4var5<-read.table("dataA4var5.txt",header=T) #Variance 4.
dataA4var6<-read.table("dataA4var6.txt",header=T) #Variance 8.
dataA4var7<-read.table("dataA4var7.txt",header=T) #Variance 16.


coverage5050A.25=fsimulationAcoverage(123,1000,.25,50,50,dataA4var1)
save(list=c("coverage5050A.25",".Random.seed"),file="5050coverageA025.RData")
savedseed=.Random.seed

coverage5050A.5=fsimulationAcoverage(savedseed,1000,.5,50,50,dataA4var2)
save(list=c("coverage5050A.5",".Random.seed"),file="5050coverageA05.RData")
savedseed=.Random.seed

coverage5050A1=fsimulationAcoverage(savedseed,1000,1,50,50,dataA4var3)
save(list=c("coverage5050A1",".Random.seed"),file="5050coverageA1.RData")
savedseed=.Random.seed

coverage5050A2=fsimulationAcoverage(savedseed,1000,2,50,50,dataA4var4)
save(list=c("coverage5050A2",".Random.seed"),file="5050coverageA2.RData")
savedseed=.Random.seed

coverage5050A4=fsimulationAcoverage(savedseed,1000,4,50,50,dataA4var5)
save(list=c("coverage5050A4",".Random.seed"),file="5050coverageA4.RData")
savedseed=.Random.seed

coverage5050A8=fsimulationAcoverage(savedseed,1000,8,50,50,dataA4var6)
save(list=c("coverage5050A8",".Random.seed"),file="5050coverageA8.RData")
savedseed=.Random.seed

coverage5050A16=fsimulationAcoverage(savedseed,1000,16,50,50,dataA4var7)
save(list=c("coverage5050A16",".Random.seed"),file="5050coverageA16.RData")


########################################.
###COVERAGE INTERVAL OF THE ESTIMATES ##.
########################################.

covinter<-function(result,Niter){

#result is the file with the estimate parameter b1 and the confidence
#interval with the naive estimator and sandwich in the IRM, ARM & PBEE.

matrixone=matrix(1,nrow=Niter,ncol=1)
resultarem=matrix(result[,16:17],ncol=2)
resultarem2=cbind(resultarem,matrixone)
resultagrem=matrix(result[,18:19],ncol=2)
resultagrem2=cbind(resultagrem,matrixone)
resultpbm=matrix(result[,20:21],ncol=2)
resultpbm2=cbind(resultpbm,matrixone)

resultaremsubnaive=matrix(subset(resultarem2,resultarem2[,1]>=0),ncol=3)
resultagremsubnaive=matrix(subset(resultagrem2,resultagrem2[,1]>=0),ncol=3)
resultpbmsubnaive=matrix(subset(resultpbm2,resultpbm2[,1]>=0),ncol=3)

resultaremsubsandwich=matrix(subset(resultarem2,resultarem2[,2]>=0),ncol=3)
resultagremsubsandwich=matrix(subset(resultagrem2,resultagrem2[,2]>=0),ncol=3)
resultpbmsubsandwich=matrix(subset(resultpbm2,resultpbm2[,2]>=0),ncol=3)

sumaremnaive=sum(resultaremsubnaive[,1])
naremnaive=sum(resultaremsubnaive[,3])
sumagremnaive=sum(resultagremsubnaive[,1])
nagremnaive=sum(resultagremsubnaive[,3])
sumpbmnaive=sum(resultpbmsubnaive[,1])
npbmnaive=sum(resultpbmsubnaive[,3])


sumaremsandwich=sum(resultaremsubsandwich[,2])
naremsandwich=sum(resultaremsubsandwich[,3])
sumagremsandwich=sum(resultagremsubsandwich[,2])
nagremsandwich=sum(resultagremsubsandwich[,3])
sumpbmsandwich=sum(resultpbmsubsandwich[,2])
npbmsandwich=sum(resultpbmsubsandwich[,3])

#coverage interval for IRM.
aremnaive=sumaremnaive/naremnaive
aremsandwich=sumaremsandwich/naremsandwich

#coverage interval for ARM.
agremnaive=sumagremnaive/nagremnaive
agremsandwich=sumagremsandwich/nagremsandwich

#coverage interval for PBEE.
pbmnaive=sumpbmnaive/npbmnaive
pbmsandwich=sumpbmsandwich/npbmsandwich

cat("Coverage interval naive
(AREM,AGREM,PBM)","\n",aremnaive,"\n",agremnaive,"\n",pbmnaive,"\n")
```

```
cat("Coverage interval sandwich
(AREM,AGREM,PBM)","\n",aremsandwich,"\n",agremsandwich,"\n",pbmsandwich,"\n")

       }

#100 groups-100 sample size in each group.

covinter(coverage100100A.25$result,1000)
covinter(coverage100100A.5$result,1000)
covinter(coverage100100A1$result,1000)
covinter(coverage100100A2$result,1000)
covinter(coverage100100A4$result,1000)
covinter(coverage100100A8$result,1000)
covinter(coverage100100A16$result,1000)

#50 groups-100 sample size in each group.

covinter(coverage50100A.25$result,1000)
covinter(coverage50100A.5$result,1000)
covinter(coverage50100A1$result,1000)
covinter(coverage50100A2$result,1000)
covinter(coverage50100A4$result,1000)
covinter(coverage50100A8$result,1000)
covinter(coverage50100A16$result,1000)

#100 groups-50 sample size in each group.

covinter(coverage10050A.25$result,1000)
covinter(coverage10050A.5$result,1000)
covinter(coverage10050A1$result,1000)
covinter(coverage10050A2$result,1000)
covinter(coverage10050A4$result,1000)
covinter(coverage10050A8$result,1000)
covinter(coverage10050A16$result,1000)

#50 groups-50 sample size in each group.

covinter(coverage5050A.25$result,1000)
covinter(coverage5050A.5$result,1000)
covinter(coverage5050A1$result,1000)
covinter(coverage5050A2$result,1000)
covinter(coverage5050A4$result,1000)
covinter(coverage5050A8$result,1000)
covinter(coverage5050A16$result,1000)
```

A.3.5 SCC coverage program.

```
library(MASS)

############################################################.
############################################################.
##FUNCTION INDIVIDUAL RANDOM EFFECTS MODEL (faremcoverage)##.
##FOR COMPUTE NAIVE AND SANDWICH ESTIMATOR #################.
############################################################.
############################################################.

faremcoverage<-function(betain,data) {

betanew=as.vector(betain[1,],mode="numeric")
Dataprove=data

#K is the number of groups. We suppose that groups are ordered and they have all.
#the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
#N is the number of observations and p is the number of covariates.
N<-dim(Dataprove)[1]
K<-Dataprove[N,2]
p<-dim(Dataprove)[2]-5

ngr<-matrix(,nrow=1,ncol=K)
for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

gamma0<-betanew[1]
beta=betanew[-1]

#Individual outcome.
Yki<-matrix(,nrow=N,ncol=1)
Yki[,1]<-Dataprove[,3]

#Individual mean.
muki<-matrix(,nrow=N,ncol=1)
muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

#Matrix D for the IRM.
Dki<-matrix(,nrow=N,ncol=p+1)
Dki[,1]<-muki[,1]
```

```
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Inverse variance-covariance matrix for the IRM.

          ##First, we compute sigma square.
          muki2<-matrix(,nrow=N,ncol=1)
          muki2[,1]<-muki[,1]^2

          Yaver<-matrix(,nrow=1,ncol=K)
          muk<-matrix(,nrow=1,ncol=K)
          phik<-matrix(,nrow=K,ncol=1)
          sigma2rek<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                 Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                 muk[1,i]<-sum(muki[ini:end])/ngr[i]
                 phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                 sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-
100)
                 ini<-end+1
                 end<-ngr[i+1]+end}

          sigma2re<-sum(sigma2rek[1:K])/K

          #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
          sumk<-matrix(,nrow=1,ncol=K)
          ini<-1
          end<-ngr[1]
          for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                         ini<-end+1
                         end<-ngr[i+1]+end}

      #Finally, we define the elements of the inverse of V.
      Vki<-list(matrix(,nrow=K,ncol=1))

      for (j in 1:K){
        Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

        for (i in 1:ngr[1,j]) {
          yy=1-(1+sigma2re)*muki[i]
                Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
          Vki[[j]][i,i]<-(1/(muki[i]*yy))-(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
           }
          Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
             }

  ######################################.
  #NAIVE AND SANDWICH ESTIMATORS for IRM#.
  ######################################.

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
          for (n in 1:(p+1)){
              Dkilist[[j]][,n]<-Dki[ini:end,n]}
              ini<-end+1
              end<-ngr[j+1]+end}

  ElementKM<-list(matrix(,nrow=K,ncol=1))
  Mlist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
```

```
  for (i in 1:K) {
          ElementKM[[i]]<-
t(Dkilist[[i]])%*%Vki[[i]]%*%Ykiminusmuki[[i]]%*%t(Ykiminusmuki[[i]])%*%Vki[[i]]%*%Dkilist[
[i]]
          Mlist[[1]]<-Mlist[[1]]+ElementKM[[i]]}

  M<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {M[i,j]<-Mlist[[1]][i,j]}}

  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
          ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
          Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}

  naive=ginv(-Hs)
  robust=naive%*%M%*%naive

  list(naive=naive, robust=robust)
  }

  ################################################################.
  ################################################################.
  ##FUNCTION AGGREGATED RANDOM EFFECTS MODEL (fagremcoverage)##.
  ##FOR COMPUTE NAIVE AND SANDWICH ESTIMATOR ###################.
  ################################################################.
  ################################################################.

  fagremcoverage<-function(betain,data) {

  betanew=as.vector(betain[1,],mode="numeric")
  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]

  #Individual mean (muki).
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Variance for the ARM.
  muki2<-matrix(,nrow=N,ncol=1)
  muki2[,1]<-muki[,1]^2

  #Outcome for the ARM as defined in Sheppard and Prentice (Biometrics,1995).
  Y<-matrix(,nrow=1,ncol=K)

  muk<-matrix(,nrow=1,ncol=K)
  phik<-matrix(,nrow=K,ncol=1)
  #Matrix D for the ARM.
  Dk<-matrix(,nrow=K,ncol=p+1)

  #First, we compute sigma square.
  sigma2amk<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                  Y[1,i]<-((Dataprove[ini,4])/(Dataprove[ini,5]))
                  muk[1,i]<-sum(muki[ini:end])/ngr[i]
                  phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                  for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                  sigma2amk[i,]<-max(((Y[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]))),-100)
```

```
                ini<-end+1
                end<-ngr[i+1]+end}

  sigma2am<-sum(sigma2amk[1:K])/K

  #Finally, we define the variance.
  Vk<-matrix(,nrow=1,ncol=K)
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Vk[1,i]<-sigma2am*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5])))+(muk[i]-
phik[i,])*(1/(Dataprove[ini,5]))
                ini<-1
                end<-ngr[1]}

  ######################################.
  #NAIVE AND SANDWICH ESTIMATORS for ARM#.
  ######################################.

  Dkt<-t(Dk)
  ElementKMa<-list(matrix(,nrow=K,ncol=1))
  Malist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKMa[[i]]<-
matrix(Dkt[,i],(p+1),1)%*%matrix(Dkt[,i],1,(p+1))*((1/Vk[i])*(Y[i]-muk[i]))^2
  Malist[[1]]<-Malist[[1]]+ElementKMa[[i]]}

  Ma<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Ma[i,j]<-Malist[[1]][i,j]}}

  ElementKaHs<-list(matrix(,nrow=K,ncol=1))
  Hsalist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKaHs[[i]]<--
1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vk[i])%*%Dk[i,]
                Hsalist[[1]]<-Hsalist[[1]]+ElementKaHs[[i]]}

  Hsa<-matrix(,nrow=(p+1),ncol=(p+1))
        for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsa[i,j]<-Hsalist[[1]][i,j]}}

  naive=ginv(-Hsa)
  robust=naive%*%Ma%*%naive

  list(naive=naive, robust=robust)
  }


  ###################################################################.
  ###################################################################.
  ##FUNCTION POPULATION-BASED ESTIMATING EQUATION (fpbmcoverage)##.
  ##FOR COMPUTE NAIVE AND SANDWICH ESTIMATOR ######################.
  ###################################################################.
  ###################################################################.

  fpbmcoverage<-function(betain,data) {

  betanew=as.vector(betain[1,],mode="numeric")
  Dataprove=data

  #K is the number of groups. We suppose that groups are ordered and they have all.
  #the correlatives numbers. For example:1,2,3 and not 1,3 (There are no number 2).
  #N is the number of observations and p is the number of covariates.
  N<-dim(Dataprove)[1]
  K<-Dataprove[N,2]
  p<-dim(Dataprove)[2]-5

  ngr<-matrix(,nrow=1,ncol=K)
  for (i in 1:K) ngr[1,i]<-dim(subset(Dataprove,Dataprove[2]==i))[1]

  gamma0<-betanew[1]
  beta<-betanew[-1]

  #Individual outcome.
  Yki<-matrix(,nrow=N,ncol=1)
  Yki[,1]<-Dataprove[,3]

  #Individual mean.
  muki<-matrix(,nrow=N,ncol=1)
  muki[,1]<-exp(gamma0+as.vector(as.matrix(Dataprove[,6:(5+p)])%*%beta))

  #Individual matrix D.
  Dki<-matrix(,nrow=N,ncol=p+1)
  Dki[,1]<-muki[,1]
  for (j in 1:p){
  Dki[,j+1]<-as.numeric(Dataprove[,5+j])*muki[,1]
  }

  #Inverse variance-covariance matrix individual part.
```

```
  #First, we compute sigma square for the individual part.

          muki2<-matrix(,nrow=N,ncol=1)
          muki2[,1]<-muki[,1]^2

          Yaver<-matrix(,nrow=1,ncol=K)
          muk<-matrix(,nrow=1,ncol=K)
          phik<-matrix(,nrow=K,ncol=1)
          sigma2rek<-matrix(,nrow=K,ncol=1)

          ini<-1
          end<-ngr[1]
          for (i in 1:K) {
                  Yaver[1,i]<-sum(Yki[ini:end])/ngr[i]
                  muk[1,i]<-sum(muki[ini:end])/ngr[i]
                  phik[i,1]<-sum(muki2[ini:end])/ngr[i]
                  sigma2rek[i,]<-max((Yaver[1,i]*(Yaver[1,i]*ngr[i]-2*ngr[i]*muk[1,i]-
1)+2*((t(muki[ini:end,1])%*%Yki[ini:end,1])/ngr[i]))/(ngr[i]*(muk[1,i]^2)-phik[i,1])+1,-
100)
                  ini<-end+1
                  end<-ngr[i+1]+end}

          sigma2re<-sum(sigma2rek[1:K])/K

          #We compute the expression for one part (transpose(muk)*Inverse(Deltak)*muk).
          sumk<-matrix(,nrow=1,ncol=K)
          ini<-1
          end<-ngr[1]
          for (i in 1:K) {sumk[1,i]<-sum((muki[ini:end]^2)/(muki[ini:end]*(1-
(1+sigma2re)*muki[ini:end])))
                          ini<-end+1
                          end<-ngr[i+1]+end}

      #Finally we define the elements of the inverse of V.
      Vki<-list(matrix(,nrow=K,ncol=1))

      for (j in 1:K){
        Vki[[j]]<-matrix(0,nrow=ngr[1,j],ncol=ngr[1,j])

        for (i in 1:ngr[1,j]) {
          yy=1-(1+sigma2re)*muki[i]
           Vki[[j]][i,(i:ngr[1,j])]<-(-sigma2re*(1/yy)*(1/(1-
(1+sigma2re)*muki[(i:ngr[1,j])])))*(1/(1+sigma2re*sumk[1,j]))
                 Vki[[j]][i,i]<-(1/(muki[i]*yy))-
(sigma2re*(1/yy)^2)*(1/(1+sigma2re*sumk[1,j]))
        }
          Vki[[j]]=Vki[[j]]+t(Vki[[j]])-diag(diag(Vki[[j]]))
             }

  #Vectors of individual responses for each group. For ngr[4] is NA but we don't use it.
  Ykilist<-list(matrix(,nrow=K,ncol=1))

  #Vectors of mean for each group.
  mukilist<-list(matrix(,nrow=K,ncol=1))

  ini<-1
  end<-ngr[1]
  for (i in 1:K) {Ykilist[i]<-list(matrix(Dataprove[ini:end,3],nrow=ngr[i],ncol=1))
                  mukilist[i]<-list(matrix(muki[ini:end],nrow=ngr[i],ncol=1))
                   ini<-end+1
                  end<-ngr[i+1]+end}

  #Vector diference response and mean.
  Ykiminusmuki<-list(matrix(,nrow=K,ncol=1))
  for (i in 1:K) {Ykiminusmuki[[i]]<-(Ykilist[[i]]-mukilist[[i]])}

  #Matrix Dk for each group.
  Dkilist<-list(matrix(,nrow=K,ncol=1))
        for (i in 1:K) Dkilist[[i]]<-matrix(,nrow=ngr[1,i],ncol=p+1)
  ini<-1
  end<-ngr[1]
  for (j in 1:K){
          for (n in 1:(p+1)){
              Dkilist[[j]][,n]<-Dki[ini:end,n]}
              ini<-end+1
              end<-ngr[j+1]+end}


  #Outcome for the aggregated data model with combined analytical and aggregated models.
  Ybar<-matrix(,nrow=1,ncol=K)

  #Matrix D for the aggregated part.
  Dk<-matrix(,nrow=K,ncol=p+1)

        ini<-1
  end<-ngr[1]
```

```
  for (i in 1:K) {Ybar[1,i]<-((Dataprove[ini,4]-sum(Yki[ini:end]))/(Dataprove[ini,5]-
ngr[i]))
                for (j in 1:(p+1)) {Dk[i,j]<-sum(Dki[ini:end,j])/ngr[i]}
                ini<-end+1
                end<-ngr[i+1]+end}

  Dkt<-t(Dk)

  #Sigma square aggregated part.
  sigma2pbk<-matrix(,nrow=K,ncol=1)
  ini<-1
  end<-ngr[1]
  for  (i in 1:K) {sigma2pbk[i,]<-max(((Ybar[,i]-muk[,i])^2-(muk[,i]-
phik[i,])/(Dataprove[ini,5]-ngr[i]))/(muk[,i]^2-phik[i,]*(1/(Dataprove[ini,5]-ngr[i]))),0)
                ini<-end+1
                end<-ngr[i+1]+end}

  sigma2pb<-sum(sigma2pbk[1:K])/K

  Vkbar<-matrix(,nrow=1,ncol=K)
  Mpb=matrix(0,nrow=(p+1),ncol=(p+1))
  ini<-1
  end<-ngr[1]
  for (i in 1:K) {
  Vkbar[1,i]<-sigma2pb*((muk[i]^2)-(phik[i,]/(Dataprove[ini,5]-ngr[i])))+(muk[i]-
phik[i,])*(1/(Dataprove[ini,5]-ngr[i]))
  junkmat=diag(c(rep(0,ngr[i]),1/Vkbar[1,i]))
  junkmat[1:ngr[i],1:ngr[i]]=Vki[[i]]
  sub=t(rbind(Dkilist[[i]],Dk[i,]))%*%junkmat%*%rbind(Ykiminusmuki[[i]],Ybar[i]-muk[i])
  Mpb=Mpb+sub%*%t(sub)
  ini<-end+1
  end<-ngr[i+1]+end
        }

  ElementKHs<-list(matrix(,nrow=K,ncol=1))
  Hslist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {
         ElementKHs[[i]]<--1*t(Dkilist[[i]])%*%Vki[[i]]%*%Dkilist[[i]]
         Hslist[[1]]<-Hslist[[1]]+ElementKHs[[i]]}

  Hs<-matrix(,nrow=(p+1),ncol=(p+1))
       for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hs[i,j]<-Hslist[[1]][i,j]}}

  ElementKarHs<-list(matrix(,nrow=K,ncol=1))
  Hsarlist<-list(matrix(0,nrow=(p+1),ncol=(p+1)))
  for (i in 1:K) {ElementKarHs[[i]]<--
1*matrix(Dkt[,i],nrow=(p+1),ncol=1)%*%(1/Vkbar[i])%*%Dk[i,]
                Hsarlist[[1]]<-Hsarlist[[1]]+ElementKarHs[[i]]}

  Hsar<-matrix(,nrow=(p+1),ncol=(p+1))
       for (i in 1:(p+1)){
      for (j in 1:(p+1)) {Hsar[i,j]<-Hsarlist[[1]][i,j]}}

  Hspb<-(Hs+Hsar)

  naive=ginv(-Hspb)
  robust=naive%*%Mpb%*%naive

  list(naive=naive, robust=robust)
  }


  ##############################################################.
  ##############################################################.
  ##CONFIDENCE INTERVAL NAIVE AND SANDWICH FOR THE IRM########.
  ##(function farem2coverage, ARM (function fagrem2coverage)##.
  ##AND PBEE (function fpbm2coverage)#########################.
  ##############################################################.
  ##############################################################.

  farem2coverage<-function(data,datavar,beta1=.2){

  result=rep(NA,21)
  result[1]=datavar[2]

  if (datavar[5]<51) {

        q<-faremcoverage(datavar[1:3],data)
        naive=q$naive
        robust=q$robust

  if (naive[2,2]<0){
  result[2:3]=rep(-1,2)
  result[16]=-1}

  if (robust[2,2]<0){
```

```
result[4:5]=rep(-1,2)
result[17]=-1}

if (naive[2,2]>=0){
ll=(datavar[2]-1.96*sqrt(naive[2,2]))
ul=(datavar[2]+1.96*sqrt(naive[2,2]))
result[2:3]=c(ll,ul)
result[16]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

if (robust[2,2]>=0){
ll=(datavar[2]-1.96*sqrt(robust[2,2]))
ul=(datavar[2]+1.96*sqrt(robust[2,2]))
result[4:5]=c(ll,ul)
result[17]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

}

else {
result[c(2:5,16:17)]=rep(-1,6)
}

unlist(result)
      }

fagrem2coverage<-function(data,datavar,beta1=.2){

result=rep(NA,21)
result[6]=datavar[7]

if (datavar[10]<51) {

      d<-fagremcoverage(datavar[6:8],data)

      naive=d$naive
      robust=d$robust

if (naive[2,2]<0){
result[7:8]=rep(-1,2)
result[18]=-1}

if (robust[2,2]<0){
result[9:10]=rep(-1,2)
result[19]=-1}

if (naive[2,2]>=0){
ll=(datavar[7]-1.96*sqrt(naive[2,2]))
ul=(datavar[7]+1.96*sqrt(naive[2,2]))
result[7:8]=c(ll,ul)
result[18]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

if (robust[2,2]>=0){
ll=(datavar[7]-1.96*sqrt(robust[2,2]))
ul=(datavar[7]+1.96*sqrt(robust[2,2]))
result[9:10]=c(ll,ul)
result[19]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

}

else {
result[c(7:10,18:19)]=rep(-1,6)
}

unlist(result)
      }

fpbm2coverage<-function(data,datavar,beta1=.2){

result=rep(NA,21)
result[11]=datavar[12]

if (datavar[16]<51) {

      pb<-fpbmcoverage(datavar[11:13],data)

      naive=pb$naive
      robust=pb$robust

if (naive[2,2]<0){
result[12:13]=rep(-1,2)
result[20]=-1}

if (robust[2,2]<0){
result[14:15]=rep(-1,2)
result[21]=-1}

if (naive[2,2]>=0){
ll=(datavar[12]-1.96*sqrt(naive[2,2]))
```

```
ul=(datavar[12]+1.96*sqrt(naive[2,2]))
result[12:13]=c(ll,ul)
result[20]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

if (robust[2,2]>=0){
ll=(datavar[12]-1.96*sqrt(robust[2,2]))
ul=(datavar[12]+1.96*sqrt(robust[2,2]))
result[14:15]=c(ll,ul)
result[21]=ifelse(beta1>=ll & beta1<=ul, 1, 0)}

}

else {
result[c(12:15,20:21)]=rep(-1,6)
}

unlist(result)
      }


###############################################.
###############################################.
##FUNCTION FOR SIMULATE THE DATA (fgenerate2)##.
##IN THE SIMPLE CONFOUNDING CASE (SCC)       ##.
###############################################.
###############################################.

fgenerate2<-function(group,populationsize,samplesize,variance){

#group=number of groups, populationsize= population size.
#samplesize= sample size, variance=within group variance.

K<-group
nk<-populationsize
mk<-samplesize
varwithin<-variance

#Covariate X1ki & X2ki.  They are correlated 0.3 at the community.
#and individual levels (see Prentice & Sheppard).

zK=mvrnorm(K,c(0,0),matrix(c(1,.3,.3,1),2,2))

cov=0.3*sqrt(varwithin)*1
covm=matrix(c(varwithin,cov,cov,1),2,2)

X1ki=matrix(0,nk,K)
X2ki=matrix(0,nk,K)

for(i in 1:K){
 znk=mvrnorm(nk,zK[i,],covm)
 X1ki[,i]=znk[,1]
 X2ki[,i]=znk[,2]
}

#Country specific frailties were generated as independent.
#realized  values from a gamma  distribution with  mean 1.
#and variance sigma^2. The mean of a gamma is shape*scale.
#and the variance is shape*(scale^2).

meanhk<-1
varhk<-0.05
shape<-(meanhk^2)/(varhk)
scale<-(varhk)/(meanhk)
hk<-rgamma(K,shape=shape,scale=scale)[]
hk=t(matrix(rep(hk,nk),nrow=K,ncol=nk))

#The disease events, yki, were generated by determining.
#wether  a  uniform  random  variable  wass  less  than.
#hk*exp(gamma0+beta1*X1ki+beta2*X2ki).

gamma0<--3
beta1<-0.2
beta2<-0.2

yki<-matrix(,nrow=nk,ncol=K)
unif<-matrix(runif(nk*K,0,1),nrow=nk,ncol=K)

      yki=ifelse(unif<hk*exp(gamma0+beta1*X1ki+beta2*X2ki),1,0)

#######################################.
#selection of random sample of size mk#.
#and organize data to apply functions #.
#farem, fagrem and fpbm               #.
#######################################.

datalist<-list(matrix(,nrow=K,ncol=1))
sampledatalist<-list(matrix(,nrow=K,ncol=1))
```

```
  ini<-1
  end<-mk
  data<-matrix(,nrow=mk*K,ncol=5)
  for (i in 1:K){

  datalist[[i]]<-
cbind(matrix(c(1:nk),nrow=nk,ncol=1),matrix(yki[,i],nrow=nk,ncol=1),matrix(X1ki[,i],nrow=nk
,ncol=1),matrix(X2ki[,i],nrow=nk,ncol=1),matrix(c(i),nrow=nk,ncol=1))
  sampledatalist[[i]]<-datalist[[i]][as.matrix(sample(datalist[[i]][,1],mk)),]
  data[ini:end,]<-sampledatalist[[i]][]
                ini<-end+1
                end<-mk*(i+1)
  }

  O<-matrix(apply(yki,2,sum),nrow=K,ncol=1)

  ini<-1
  end<-mk
  datapop<-matrix(,nrow=mk*K,ncol=1)
  for (i in 1:K) {datapop[ini:end,1]<-O[i,]
                  ini<-end+1
                  end<-mk*(i+1)}

  datadatapop<-cbind(data,datapop,c(nk))
  datafin<-
data.frame(id=matrix(datadatapop[,1]),group=matrix(datadatapop[,5]),YIND=matrix(datadatapop
[,2]),O=matrix(datadatapop[,6]),n=matrix(datadatapop[,7]),X1ki=matrix(datadatapop[,3]),X2ki
=matrix(datadatapop[,4]))}


  ####################.
  ##Coverage results##.
  ####################.

  fsimulationBcoverage<-function(seed,Niter,sigma2,K,N,datavar){
  set.seed(seed)
  count1=0
  result1=matrix(NA,Niter,21)

  while(count1<Niter){
  tempdata1=fgenerate2(K,2000,N,sigma2)
  count1=count1+1
  a1<-farem2coverage(tempdata1,datavar[count1,])
  a2<-fagrem2coverage(tempdata1,datavar[count1,])
  a3<-fpbm2coverage(tempdata1,datavar[count1,])
  result1[count1,c(1:5,16:17)]=a1[c(1:5,16:17)]
  result1[count1,c(6:10,18:19)]=a2[c(6:10,18:19)]
  result1[count1,c(11:15,20:21)]=a3[c(11:15,20:21)]
  print(count1)
  }

  list(result=result1,sigma2=sigma2,K=K,N=N)
  }

  ########################################.
  #100 groups-100 sample size in each group#.
  ########################################.

  #The next  text files  are  the  results  files  from  the  simulation runs.
  #to obtain parameter estimates in each variation ratio for the 100-100 case.
  #For example, dataA1var1.txt is from finalresultA100100A.25$result.

  dataB1var1<-read.table("dataB1var1.txt",header=T) #Variance 0.25.
  dataB1var2<-read.table("dataB1var2.txt",header=T) #Variance 0.5.
  dataB1var3<-read.table("dataB1var3.txt",header=T) #Variance 1.
  dataB1var4<-read.table("dataB1var4.txt",header=T) #Variance 2.
  dataB1var5<-read.table("dataB1var5.txt",header=T) #Variance 4.
  dataB1var6<-read.table("dataB1var6.txt",header=T) #Variance 8.
  dataB1var7<-read.table("dataB1var7.txt",header=T) #Variance 16.


  coverage100100B.25=fsimulationBcoverage(123,1000,.25,100,100,dataB1var1)
  save(list=c("coverage100100B.25",".Random.seed"),file="100100coverageB025.RData")
  savedseed=.Random.seed

  coverage100100B.5=fsimulationBcoverage(savedseed,1000,.5,100,100,dataB1var2)
  save(list=c("coverage100100B.5",".Random.seed"),file="100100coverageB05.RData")
  savedseed=.Random.seed

  coverage100100B1=fsimulationBcoverage(savedseed,1000,1,100,100,dataB1var3)
  save(list=c("coverage100100B1",".Random.seed"),file="100100coverageB1.RData")
  savedseed=.Random.seed

  coverage100100B2=fsimulationBcoverage(savedseed,1000,2,100,100,dataB1var4)
  save(list=c("coverage100100B2",".Random.seed"),file="100100coverageB2.RData")
  savedseed=.Random.seed
```

```
coverage100100B4=fsimulationBcoverage(savedseed,1000,4,100,100,dataB1var5)
save(list=c("coverage100100B4",".Random.seed"),file="100100coverageB4.RData")
savedseed=.Random.seed

coverage100100B8=fsimulationBcoverage(savedseed,1000,8,100,100,dataB1var6)
save(list=c("coverage100100B8",".Random.seed"),file="100100coverageB8.RData")
savedseed=.Random.seed

coverage100100B16=fsimulationBcoverage(savedseed,1000,16,100,100,dataB1var7)
save(list=c("coverage100100B16",".Random.seed"),file="100100coverageB16.RData")

########################################.
#50 groups-100 sample size in each group#.
########################################.

#The next  text files  are  the  results  files  from  the  simulation runs.
#to obtain parameter estimates in each variation ratio for the 50-100 case.

dataB2var1<-read.table("dataB2var1.txt",header=T) #Variance 0.25.
dataB2var2<-read.table("dataB2var2.txt",header=T) #Variance 0.5.
dataB2var3<-read.table("dataB2var3.txt",header=T) #Variance 1.
dataB2var4<-read.table("dataB2var4.txt",header=T) #Variance 2.
dataB2var5<-read.table("dataB2var5.txt",header=T) #Variance 4.
dataB2var6<-read.table("dataB2var6.txt",header=T) #Variance 8.
dataB2var7<-read.table("dataB2var7.txt",header=T) #Variance 16.


coverage50100B.25=fsimulationBcoverage(123,1000,.25,50,100,dataB2var1)
save(list=c("coverage50100B.25",".Random.seed"),file="50100coverageB025.RData")
savedseed=.Random.seed

coverage50100B.5=fsimulationBcoverage(savedseed,1000,.5,50,100,dataB2var2)
save(list=c("coverage50100B.5",".Random.seed"),file="50100coverageB05.RData")
savedseed=.Random.seed

coverage50100B1=fsimulationBcoverage(savedseed,1000,1,50,100,dataB2var3)
save(list=c("coverage50100B1",".Random.seed"),file="50100coverageB1.RData")
savedseed=.Random.seed

coverage50100B2=fsimulationBcoverage(savedseed,1000,2,50,100,dataB2var4)
save(list=c("coverage50100B2",".Random.seed"),file="50100coverageB2.RData")
savedseed=.Random.seed

coverage50100B4=fsimulationBcoverage(savedseed,1000,4,50,100,dataB2var5)
save(list=c("coverage50100B4",".Random.seed"),file="50100coverageB4.RData")
savedseed=.Random.seed

coverage50100B8=fsimulationBcoverage(savedseed,1000,8,50,100,dataB2var6)
save(list=c("coverage50100B8",".Random.seed"),file="50100coverageB8.RData")
savedseed=.Random.seed

coverage50100B16=fsimulationBcoverage(savedseed,1000,16,50,100,dataB2var7)
save(list=c("coverage50100B16",".Random.seed"),file="50100coverageB16.RData")

########################################.
#100 groups-50 sample size in each group#.
########################################.

#The next  text files  are  the  results  files  from  the  simulation runs.
#to obtain parameter estimates in each variation ratio for the 50-100 case.

dataB3var1<-read.table("dataB3var1.txt",header=T) #Variance 0.25.
dataB3var2<-read.table("dataB3var2.txt",header=T) #Variance 0.5.
dataB3var3<-read.table("dataB3var3.txt",header=T) #Variance 1.
dataB3var4<-read.table("dataB3var4.txt",header=T) #Variance 2.
dataB3var5<-read.table("dataB3var5.txt",header=T) #Variance 4.
dataB3var6<-read.table("dataB3var6.txt",header=T) #Variance 8.
dataB3var7<-read.table("dataB3var7.txt",header=T) #Variance 16.


coverage10050B.25=fsimulationBcoverage(123,1000,.25,100,50,dataB3var1)
save(list=c("coverage10050B.25",".Random.seed"),file="10050coverageB025.RData")
savedseed=.Random.seed

coverage10050B.5=fsimulationBcoverage(savedseed,1000,.5,100,50,dataB3var2)
save(list=c("coverage10050B.5",".Random.seed"),file="10050coverageB05.RData")
savedseed=.Random.seed

coverage10050B1=fsimulationBcoverage(savedseed,1000,1,100,50,dataB3var3)
save(list=c("coverage10050B1",".Random.seed"),file="10050coverageB1.RData")
savedseed=.Random.seed

coverage10050B2=fsimulationBcoverage(savedseed,1000,2,100,50,dataB3var4)
save(list=c("coverage10050B2",".Random.seed"),file="10050coverageB2.RData")
savedseed=.Random.seed

coverage10050B4=fsimulationBcoverage(savedseed,1000,4,100,50,dataB3var5)
```

```
save(list=c("coverage10050B4",".Random.seed"),file="10050coverageB4.RData")
savedseed=.Random.seed

coverage10050B8=fsimulationBcoverage(savedseed,1000,8,100,50,dataB3var6)
save(list=c("coverage10050B8",".Random.seed"),file="10050coverageB8.RData")
savedseed=.Random.seed

coverage10050B16=fsimulationBcoverage(savedseed,1000,16,100,50,dataB3var7)
save(list=c("coverage10050B16",".Random.seed"),file="10050coverageB16.RData")

########################################.
#50 groups-50 sample size in each group#.
########################################.

#The next  text files  are  the  results  files  from  the  simulation runs.
#to obtain parameter estimates in each variation ratio for the 50-100 case.

dataB4var1<-read.table("dataB4var1.txt",header=T) #Variance 0.25.
dataB4var2<-read.table("dataB4var2.txt",header=T) #Variance 0.5.
dataB4var3<-read.table("dataB4var3.txt",header=T) #Variance 1.
dataB4var4<-read.table("dataB4var4.txt",header=T) #Variance 2.
dataB4var5<-read.table("dataB4var5.txt",header=T) #Variance 4.
dataB4var6<-read.table("dataB4var6.txt",header=T) #Variance 8.
dataB4var7<-read.table("dataB4var7.txt",header=T) #Variance 16.


coverage5050B.25=fsimulationBcoverage(123,1000,.25,50,50,dataB4var1)
save(list=c("coverage5050B.25",".Random.seed"),file="5050coverageB025.RData")
savedseed=.Random.seed

coverage5050B.5=fsimulationBcoverage(savedseed,1000,.5,50,50,dataB4var2)
save(list=c("coverage5050B.5",".Random.seed"),file="5050coverageB05.RData")
savedseed=.Random.seed

coverage5050B1=fsimulationBcoverage(savedseed,1000,1,50,50,dataB4var3)
save(list=c("coverage5050B1",".Random.seed"),file="5050coverageB1.RData")
savedseed=.Random.seed

coverage5050B2=fsimulationBcoverage(savedseed,1000,2,50,50,dataB4var4)
save(list=c("coverage5050B2",".Random.seed"),file="5050coverageB2.RData")
savedseed=.Random.seed

coverage5050B4=fsimulationBcoverage(savedseed,1000,4,50,50,dataB4var5)
save(list=c("coverage5050B4",".Random.seed"),file="5050coverageB4.RData")
savedseed=.Random.seed

coverage5050B8=fsimulationBcoverage(savedseed,1000,8,50,50,dataB4var6)
save(list=c("coverage5050B8",".Random.seed"),file="5050coverageB8.RData")
savedseed=.Random.seed

coverage5050B16=fsimulationBcoverage(savedseed,1000,16,50,50,dataB4var7)
save(list=c("coverage5050B16",".Random.seed"),file="5050coverageB16.RData")


########################################.
###COVERAGE INTERVAL OF THE ESTIMATES ##.
########################################.

covinter<-function(result,Niter){

#result is the file with the estimate parameter b1 and the confidence
#interval with the naive estimator and sandwich in the IRM, ARM & PBEE.

matrixone=matrix(1,nrow=Niter,ncol=1)
resultarem=matrix(result[,16:17],ncol=2)
resultarem2=cbind(resultarem,matrixone)
resultagrem=matrix(result[,18:19],ncol=2)
resultagrem2=cbind(resultagrem,matrixone)
resultpbm=matrix(result[,20:21],ncol=2)
resultpbm2=cbind(resultpbm,matrixone)

resultaremsubnaive=matrix(subset(resultarem2,resultarem2[,1]>=0),ncol=3)
resultagremsubnaive=matrix(subset(resultagrem2,resultagrem2[,1]>=0),ncol=3)
resultpbmsubnaive=matrix(subset(resultpbm2,resultpbm2[,1]>=0),ncol=3)

resultaremsubsandwich=matrix(subset(resultarem2,resultarem2[,2]>=0),ncol=3)
resultagremsubsandwich=matrix(subset(resultagrem2,resultagrem2[,2]>=0),ncol=3)
resultpbmsubsandwich=matrix(subset(resultpbm2,resultpbm2[,2]>=0),ncol=3)

sumaremnaive=sum(resultaremsubnaive[,1])
naremnaive=sum(resultaremsubnaive[,3])
sumagremnaive=sum(resultagremsubnaive[,1])
nagremnaive=sum(resultagremsubnaive[,3])
sumpbmnaive=sum(resultpbmsubnaive[,1])
npbmnaive=sum(resultpbmsubnaive[,3])
```

```
sumaremsandwich=sum(resultaremsubsandwich[,2])
naremsandwich=sum(resultaremsubsandwich[,3])
sumagremsandwich=sum(resultagremsubsandwich[,2])
nagremsandwich=sum(resultagremsubsandwich[,3])
sumpbmsandwich=sum(resultpbmsubsandwich[,2])
npbmsandwich=sum(resultpbmsubsandwich[,3])

#coverage interval for IRM.
aremnaive=sumaremnaive/naremnaive
aremsandwich=sumaremsandwich/naremsandwich

#coverage interval for ARM.
agremnaive=sumagremnaive/nagremnaive
agremsandwich=sumagremsandwich/nagremsandwich

#coverage interval for PBEE.
pbmnaive=sumpbmnaive/npbmnaive
pbmsandwich=sumpbmsandwich/npbmsandwich

cat("Coverage interval naive
(AREM,AGREM,PBM)","\n",aremnaive,"\n",agremnaive,"\n",pbmnaive,"\n")
cat("Coverage interval sandwich
(AREM,AGREM,PBM)","\n",aremsandwich,"\n",agremsandwich,"\n",pbmsandwich,"\n")

       }

#100 groups-100 sample size in each group.

covinter(coverage100100B.25$result,1000)
covinter(coverage100100B.5$result,1000)
covinter(coverage100100B1$result,1000)
covinter(coverage100100B2$result,1000)
covinter(coverage100100B4$result,1000)
covinter(coverage100100B8$result,1000)
covinter(coverage100100B16$result,1000)

#50 groups-100 sample size in each group.

covinter(coverage50100B.25$result,1000)
covinter(coverage50100B.5$result,1000)
covinter(coverage50100B1$result,1000)
covinter(coverage50100B2$result,1000)
covinter(coverage50100B4$result,1000)
covinter(coverage50100B8$result,1000)
covinter(coverage50100B16$result,1000)

#100 groups-50 sample size in each group.

covinter(coverage10050B.25$result,1000)
covinter(coverage10050B.5$result,1000)
covinter(coverage10050B1$result,1000)
covinter(coverage10050B2$result,1000)
covinter(coverage10050B4$result,1000)
covinter(coverage10050B8$result,1000)
covinter(coverage10050B16$result,1000)

#50 groups-50 sample size in each group.

covinter(coverage5050B.25$result,1000)
covinter(coverage5050B.5$result,1000)
covinter(coverage5050B1$result,1000)
covinter(coverage5050B2$result,1000)
covinter(coverage5050B4$result,1000)
covinter(coverage5050B8$result,1000)
covinter(coverage5050B16$result,1000)
```

A.3.6 Bias and mean square error program.

```
###############################################.
###BIAS AND MEAN SQUARE ERROR OF ESTIMATES ##.
###############################################.

#The function fbiasmse  computes the bias and mean  square  error for the results.
#of the simulations  ( The  simulation  file  has  the  next structure: first the.
#parameters of the IRM: gamma0, beta1, beta2, variance, number of iterations. The.
#next  columns  are  the parameters  for the ARM: gamma0, beta1, beta2, variance,.
#number of iterations. Finally, the  parameters for the PBEE: gamma0,beta1,beta2,.
#variance individual,variance aggregated, number of iterations.

fbiasmse<-function(result,g0,b1,b2,aremvar,agremvar,pbavar,pbagvar){

   #result is the file with the runs of the  simulation, g0 is the  true  value of.
   #gamma0, b1 is the true value  of  b1, b2 is the true value of  b2,  aremvar is.
   #the true value of the variance  of the  IRM, agremvar is the true value of the.
   #variance of the ARM,pbavar is the true value of the variance of the individual.
```

```
   #part of the PBEE, pbagvar  is the true value of the variance of the aggregated.
   #part of the PBEE.

        resultarem<-matrix(result[,1:5],ncol=5)
        resultagrem<-matrix(result[,6:10],ncol=5)
        resultpbm<-matrix(result[,11:16],ncol=6)

        resultaremsub<-matrix(subset(resultarem,resultarem[,5]<51),ncol=5)
        resultagremsub<-matrix(subset(resultagrem,resultagrem[,5]<51),ncol=5)
        resultpbmsub<-matrix(subset(resultpbm,resultpbm[,6]<51),ncol=6)

        meanarem<-apply(resultaremsub,2,mean)
        meanagrem<-apply(resultagremsub,2,mean)
        meanpbm<-apply(resultpbmsub,2,mean)

        vararem<-apply(resultaremsub,2,var)
        varagrem<-apply(resultagremsub,2,var)
        varpbm<-apply(resultpbmsub,2,var)

        #bias IRM.
        arembiasgamma0=meanarem[1]-g0
        arembiasbeta1=meanarem[2]-b1
        arembiasbeta2=meanarem[3]-b2
        arembiasvariance=meanarem[4]-aremvar
        #bias ARM.
        agrembiasgamma0=meanagrem[1]-g0
        agrembiasbeta1=meanagrem[2]-b1
        agrembiasbeta2=meanagrem[3]-b2
        agrembiasvariance=meanagrem[4]-agremvar
        #bias PBEE.
        pbmbiasgamma0=meanpbm[1]-g0
        pbmbiasbeta1=meanpbm[2]-b1
        pbmbiasbeta2=meanpbm[3]-b2
        pbmbiasvarianceanalytical=meanpbm[4]-pbavar
        pbmbiasvarianceaggregated=meanpbm[5]-pbagvar

        #MSE IRM.
        aremmsegamma0=apply((matrix(resultaremsub[,1])-g0)^2,2,mean)
        aremmsebeta1=apply((matrix(resultaremsub[,2])-b1)^2,2,mean)
        aremmsebeta2=apply((matrix(resultaremsub[,3])-b2)^2,2,mean)
        aremmsevariance=apply((matrix(resultaremsub[,4])-aremvar)^2,2,mean)
        #MSE ARM.
        agremmsegamma0=apply((matrix(resultagremsub[,1])-g0)^2,2,mean)
        agremmsebeta1=apply((matrix(resultagremsub[,2])-b1)^2,2,mean)
        agremmsebeta2=apply((matrix(resultagremsub[,3])-b2)^2,2,mean)
        agremmsevariance=apply((matrix(resultagremsub[,4])-agremvar)^2,2,mean)
        #MSE PBEE.
        pbmmsegamma0=apply((matrix(resultpbmsub[,1])-g0)^2,2,mean)
        pbmmsebeta1=apply((matrix(resultpbmsub[,2])-b1)^2,2,mean)
        pbmmsebeta2=apply((matrix(resultpbmsub[,3])-b2)^2,2,mean)
        pbmmsevarianceanalytical=apply((matrix(resultpbmsub[,4])-pbavar)^2,2,mean)
        pbmmsevarianceaggregated=apply((matrix(resultpbmsub[,5])-pbagvar)^2,2,mean)


        cat("Bias b1
        (IRM,ARM,PBEE)","\n",round((arembiasbeta1/0.2)*100,digit=2),"\n",round((agrembiasb
        eta1/0.2)*100,digit=2),"\n",round((pbmbiasbeta1/0.2)*100,digit=2),"\n")
        cat("Mse b1
        (IRM,ARM,PBEE)","\n",round(aremmsebeta1*1000,digit=2),"\n",round(agremmsebeta1*100
        0,digit=2),"\n",round(pbmmsebeta1*1000,digit=2),"\n")

        cat("Bias b2
        (IRM,ARM,PBEE)","\n",round((arembiasbeta2/0.2)*100,digit=2),"\n",round((agrembiasb
        eta2/0.2)*100,digit=2),"\n",round((pbmbiasbeta2/0.2)*100,digit=2),"\n")
        cat("Mse b2
        (IRM,ARM,PBEE)","\n",round(aremmsebeta2*1000,digit=2),"\n",round(agremmsebeta2*100
        0,digit=2),"\n",round(pbmmsebeta2*1000,digit=2),"\n")

}


#NON-CONFOUNDING CASE.

#100 groups-100 sample size in each group.
fbiasmse(finalresultA100100A.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA100100A.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA100100A1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA100100A2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA100100A4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA100100A8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA100100A16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#100 groups-50 sample size in each group.
fbiasmse(finalresultA10050A.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA10050A.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA10050A1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
```

```
fbiasmse(finalresultA10050A2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA10050A4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA10050A8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA10050A16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#50 groups-100 sample size in each group.
fbiasmse(finalresultA50100A.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA50100A.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA50100A1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA50100A2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA50100A4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA50100A8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA50100A16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#50 groups-50 sample size in each group.
fbiasmse(finalresultA5050A.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA5050A.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA5050A1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA5050A2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA5050A4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA5050A8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultA5050A16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#SIMPLE CONFOUNDING CASE.

#100 groups-100 sample size in each group.
fbiasmse(finalresultB100100B.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB100100B.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB100100B1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB100100B2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB100100B4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB100100B8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB100100B16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#100 groups-50 sample size in each group.
fbiasmse(finalresultB10050B.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultBA10050B.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB10050B1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB10050B2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB10050B4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB10050B8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB10050B16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#50 groups-100 sample size in each group.
fbiasmse(finalresultB50100B.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB50100B.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB50100B1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB50100B2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB50100B4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB50100B8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB50100B16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#50 groups-50 sample size in each group.
fbiasmse(finalresultB5050B.25$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB5050B.5$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB5050B1$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB5050B2$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB5050B4$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB5050B8$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
fbiasmse(finalresultB5050B16$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)

#EXTENDED CONFOUNDING CASE.

#100 groups-100 sample size in each group.
fbiasmse(finalresultC100100$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
#100 groups-50 sample size in each group.
fbiasmse(finalresultC10050$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
#50 groups-100 sample size in each group.
fbiasmse(finalresultC50100$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
#50 groups-50 sample size in each group.
fbiasmse(finalresultC5050$result,-3,0.2,0.2,0.05,0.05,0.05,0.05)
```

A.4 Demonstrations.

A.4.1 Calculation of $\hat{V}_k^A$.

We know

$$\mathrm{Var}(\overline{Y}_k^A) = \mathrm{Var}(\mathrm{E}(\overline{Y}_k^A|h_k)) + \mathrm{E}(\mathrm{Var}(\overline{Y}_k^A|h_k)) \qquad [A4]$$

First, we compute the quantities $\mathrm{E}(\overline{Y}_k^A|h_k)$ and $\mathrm{Var}(\overline{Y}_k^A|h_k)$

$$\mathrm{E}(\overline{Y}_k^A|h_k) = \mathrm{E}(\frac{\sum_{i=1}^{n_k-m_k} Y_{ki}}{n_k-m_k}|h_k) = \frac{\sum_{i=1}^{n_k-m_k} \mathrm{E}(Y_{ki}\,|\,h_k)}{n_k-m_k} = \frac{\sum_{i=1}^{n_k-m_k} h_k e^{x_{ki}^T\alpha}}{n_k-m_k} = \frac{h_k \sum_{i=1}^{n_k-m_k} e^{x_{ki}^T\alpha}}{n_k-m_k}$$

$$\mathrm{Var}(\overline{Y}_k^A|h_k) = \mathrm{Var}(\frac{\sum_{i=1}^{n_k-m_k} Y_{ki}}{n_k-m_k}|h_k) = \frac{\sum_{i=1}^{n_k-m_k} \mathrm{Var}(Y_{ki}\,|\,h_k)}{(n_k-m_k)^2} \overset{(1)}{=}$$

$$= \frac{\sum_{i=1}^{n_k-m_k} \left(\mathrm{E}(Y_{ki}\,|\,h_k) - \mathrm{E}(Y_{ki}\,|\,h_k)^2\right)}{(n_k-m_k)^2} = \frac{\sum_{i=1}^{n_k-m_k} \left(h_k e^{x_{ki}^T\alpha} - h_k^2 e^{2x_{ki}^T\alpha}\right)}{(n_k-m_k)^2}$$

We know that h_k is a random effect with $\mathrm{E}[h_k]=1$ and $\mathrm{Var}[h_k]=\sigma^2$ and

$$\mu_k^A = \mathrm{E}[\overline{Y}_k^A] = \mathrm{E}[\frac{h_k \sum_{i=1}^{n_k-m_k} e^{x_{ki}^T\alpha}}{n_k-m_k}] = \frac{\mathrm{E}[h_k] \sum_{i=1}^{n_k-m_k} e^{x_{ki}^T\alpha}}{n_k-m_k} = \frac{\sum_{i=1}^{n_k-m_k} e^{x_{ki}^T\alpha}}{n_k-m_k}$$

so we can compute [A4] as

$$\mathrm{Var}[\overline{Y}_k^A] = \mathrm{Var}[\frac{h_k \sum_{i=1}^{n_k-m_k} e^{x_{ki}^T\alpha}}{n_k-m_k}] + \mathrm{E}[\frac{\sum_{i=1}^{n_k-m_k} \left(h_k e^{x_{ki}^T\alpha} - h_k^2 e^{2x_{ki}^T\alpha}\right)}{(n_k-m_k)^2}]$$

$$= \mathrm{Var}[h_k\mu_k^A] + \mathrm{E}[\frac{h_k\mu_k^A}{n_k-m_k} - \frac{h_k^2\phi_k}{n_k-m_k}] = (\mu_k^A)^2\mathrm{Var}[h_k] + \frac{\mu_k^A\mathrm{E}[h_k]}{n_k-m_k} - \frac{\phi_k\mathrm{E}[h_k^2]}{n_k-m_k}$$

$$= (\mu_k^A)^2\sigma^2 + \frac{\mu_k^A}{n_k-m_k} - \frac{\phi_k(\sigma^2+1)}{n_k-m_k} = (\mu_k^A)^2\sigma^2 + \{\mu_k^A - \phi_k(\sigma^2+1)\}(n_k-m_k)^{-1}$$

$$= \sigma^2((\mu_k^A)^2 - \phi_k(n_k-m_k)^{-1}) + (\mu_k^A - \phi_k)(n_k-m_k)^{-1}$$

where $\phi_k = \sum_{i=1}^{n_k-m_k} e^{2x_{ki}^T\alpha} \Big/ n_k-m_k$. We consider $\hat{\phi}_k = \varepsilon_{m_k}\{e^{2x_k^T\alpha}\}$ and $\hat{\mu}_k^A = \varepsilon_{m_k}\{e^{x_k^T\alpha}\}$ so

$$\hat{V}_k^A = \sigma^2((\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k-m_k)^{-1}) + (\hat{\mu}_k^A - \hat{\phi}_k)(n_k-m_k)^{-1}$$

(1) Y_{ki} follow a Bernoulli(p_{ki}) so $p_{ki} = \mathrm{E}(Y_{ki}\,|\,h_k) = \mathrm{E}(Y_{ki}^2\,|\,h_k)$

A.4.2 Estimation for σ^2 in the individual part of the PBEE approach.

We know

$$E[(Y_{ki}-\mu_{ki})(Y_{kj}-\mu_{kj})]=\sigma^2\mu_{ki}\mu_{kj} \qquad \text{for } i\neq j \text{ and } k=1,\ldots,K$$

so

$$\sum_{k=1}^{K}\sum_{i\neq j}(Y_{ki}-\mu_{ki})(Y_{kj}-\mu_{kj})=\sum_{k=1}^{K}\sum_{i\neq j}\sigma^2\mu_{ki}\mu_{kj}$$

$$\sum_{k=1}^{K}\left(\sum_{i=1}^{m_k}\sum_{j=1}^{m_k}\left(Y_{ki}Y_{kj}-\mu_{ki}Y_{kj}-Y_{ki}\mu_{kj}+\mu_{ki}\mu_{kj}\right)-\sum_{i=1}^{m_k}\left(Y_{ki}^2-2\mu_{ki}Y_{ki}+\mu_{ki}^2\right)\right)$$

$$=\sum_{k=1}^{K}\left(\sigma^2\left(\sum_{i=1}^{m_k}\sum_{j=1}^{m_k}\mu_{ki}\mu_{kj}-\sum_{i=1}^{m_k}\mu_{ki}^2\right)\right)$$

$$\sum_{k=1}^{K}\left(\overline{Y}_k^2m_k^2-2\overline{Y}_k\hat{\mu}_k^{A}m_k^2+(\hat{\mu}_k^{A})^2m_k^2-\sum_{i=1}^{m_k}Y_{ki}^2+2\sum_{i=1}^{m_k}\mu_{ki}Y_{ki}-\sum_{i=1}^{m_k}\mu_{ki}^2\right)$$

$$=\sum_{k=1}^{K}\sigma^2\left((\hat{\mu}_k^{A})^2m_k^2-\sum_{i=1}^{m_k}\mu_{ki}^2\right)$$

where $\hat{\mu}_k^{A}=\sum_{i=1}^{m_k}\mu_{ki}\Big/m_k=\varepsilon_{m_k}\{e^{x_k^{T}\alpha}\}$,

$$\sum_{k=1}^{K}\left(\overline{Y}_k(\overline{Y}_km_k-2\hat{\mu}_k^{A}m_k-1)+\overline{Y}_k+(\hat{\mu}_k^{A})^2m_k-\frac{\sum_{i=1}^{m_k}Y_{ki}^2}{m_k}+\frac{2\sum_{i=1}^{m_k}\mu_{ki}Y_{ki}}{m_k}-\frac{\sum_{i=1}^{m_k}\mu_{ki}^2}{m_k}\right)$$

$$=\sum_{k=1}^{K}\sigma^2\left((\hat{\mu}_k^{A})^2m_k-\frac{\sum_{i=1}^{m_k}\mu_{ki}^2}{m_k}\right)$$

$$\sum_{k=1}^{K}\left(\overline{Y}_k(\overline{Y}_km_k-2\hat{\mu}_k^{A}m_k-1)+\overline{Y}_k+(\hat{\mu}_k^{A})^2m_k-\frac{\sum_{i=1}^{m_k}Y_{ki}^2}{m_k}+2\varepsilon_{m_k}\{\mu_kY_k\}-\hat{\phi}_k\right)$$

$$=\sum_{k=1}^{K}\sigma^2\left((\hat{\mu}_k^{A})^2m_k-\hat{\phi}_k\right)$$

where $\hat{\phi}_k=\sum_{i=1}^{m_k}\mu_{ki}^2/m_k=\varepsilon_{m_k}\{e^{2x_k^{T}\alpha}\}$

$$\sum_{k=1}^{K}\left(\overline{Y}_k(\overline{Y}_k m_k - 2\hat{\mu}_k^A m_k - 1) + \left(\overline{Y}_k - \frac{\sum_{i=1}^{m_k} Y_{ki}^2}{m_k}\right) + 2\varepsilon_{m_k}\{\mu_k Y_k\} + ((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k)\right)$$
$$= \sum_{k=1}^{K} \sigma^2\left((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k\right)$$

Due to the Bernoulli assumption on Y_{ki} then $\left(\overline{Y}_k - \frac{\sum_{i=1}^{m_k} Y_{ki}^2}{m_k}\right) = 0$, so

$$\sum_{k=1}^{K}\left(\overline{Y}_k(\overline{Y}_k m_k - 2\hat{\mu}_k^A m_k - 1) + 2\varepsilon_{m_k}\{\mu_k Y_k\} + ((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k)\right)$$
$$= \sum_{k=1}^{K} \sigma^2\left((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k\right)$$

$$\sum_{k=1}^{K}\left(\varepsilon_{m_k}\{Y_k\}(\varepsilon_{m_k}\{Y_k\} m_k - 2\hat{\mu}_k^A m_k - 1) + 2\varepsilon_{m_k}\{\mu_k Y_k\} + ((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k)\right)$$
$$= \sum_{k=1}^{K} \sigma^2\left((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k\right)$$

then

$$\sigma^2 = \sum_{k=1}^{K}\left(\left(\varepsilon_{m_k}\{Y_k\}(\varepsilon_{m_k}\{Y_k\} m_k - 2\hat{\mu}_k^A m_k - 1) + 2\varepsilon_{m_k}\{\mu_k Y_k\}\right)\left((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k\right)^{-1} + 1\right)$$

and an unbiassed estimate of σ^2 is

$$(\hat{\sigma}^2)_I = \frac{1}{K}\sum_{k=1}^{K}\left(\left(\varepsilon_{m_k}\{Y_k\}(\varepsilon_{m_k}\{Y_k\} m_k - 2\hat{\mu}_k^A m_k - 1) + 2\varepsilon_{m_k}\{\mu_k Y_k\}\right)\left((\hat{\mu}_k^A)^2 m_k - \hat{\phi}_k\right)^{-1} + 1\right)$$

A.4.3 Estimation of σ^2 in the aggregated part of the PBEE approach.

We know

$$E[(\overline{Y}_k^A - \hat{\mu}_k^A)^2] = \sigma^2((\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k - m_k)^{-1}) + (\hat{\mu}_k^A - \hat{\phi}_k)(n_k - m_k)^{-1} \quad \text{for k=1,…,K}$$

so

$$\sum_{k=1}^{K}\frac{(\overline{Y}_k^A - \hat{\mu}_k^A)^2}{K} = \sum_{k=1}^{K}\left(\frac{\sigma^2\left((\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k - m_k)^{-1}\right) + \left(\hat{\mu}_k^A - \hat{\phi}_k\right)(n_k - m_k)^{-1}}{K}\right)$$

$$\sum_{k=1}^{K}(\overline{Y}_k^A - \hat{\mu}_k^A)^2 = \sum_{k=1}^{K}\left(\sigma^2\left((\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k - m_k)^{-1}\right) + \left(\hat{\mu}_k^A - \hat{\phi}_k\right)(n_k - m_k)^{-1}\right)$$

$$\sum_{k=1}^{K}\left((\overline{Y}_k^A - \hat{\mu}_k^A)^2 - \left(\hat{\mu}_k^A - \hat{\phi}_k\right)(n_k - m_k)^{-1}\right) = \sigma^2\sum_{k=1}^{K}\left((\hat{\mu}_k^A)^2 - \hat{\phi}_k(n_k - m_k)^{-1}\right)$$

$$\sigma^2 = \sum_{k=1}^{K} \left((\overline{Y}_k^A - \hat{\mu}_k^A)^2 - \left(\hat{\mu}_k^A - \hat{\phi}_k \right) (n_k - m_k)^{-1} \right) \left((\hat{\mu}_k^A)^2 - \hat{\phi}_k (n_k - m_k)^{-1} \right)^{-1}$$

and an unbiassed estimate of σ^2 is

$$(\hat{\sigma}^2)_A = \frac{1}{K} \sum_{k=1}^{K} \left((\overline{Y}_k^A - \hat{\mu}_k^A)^2 - \left(\hat{\mu}_k^A - \hat{\phi}_k \right) (n_k - m_k)^{-1} \right) \left((\hat{\mu}_k^A)^2 - \hat{\phi}_k (n_k - m_k)^{-1} \right)^{-1}$$

A.5 Construction of geographics units.

Objective

The goal was to construct well-defined contiguous small-areas or zones, with an appropriate population size and the maximum level of social homogeneity. This Atlas has used the results previously obtained in the Atlas of Mortality in Spain[16].

Criteria

Three important features had to be taken into account in order to construct small-areas: availability of information, population size and social homogeneity of the areas[141,142].

1) Availability of information. In Spain, for confidentiality reasons, annual mortality data at the municipal level are available only for areas of 10,000 people or greater. However, information was available for smaller areas (i.e., at least 3,500 inhabitants) if the mortality data were aggregated for a period of three or more years.

2) Population size. Spanish municipalities are heterogeneous in terms of their socio-economic characteristics and population size. For example, regarding their population size, more than 80% of the municipalities have fewer than 3,500 inhabitants. Thus, in order to yield reliable estimates of mortality rates, areas had to have a minimum population size.

3) Social homogeneity. Adjacent areas are often similar in terms of their social characteristics. It was possible to group municipalities with less than 3,500 inhabitants into bigger homogenous areas based on criteria of contiguity and socio-economic characteristics.

Methods

The smallest municipalities of Spain (municipalities fewer than 3,500 inhabitants) were used as the geographical building blocks to construct the zones. Information of data and demarcation lines of municipalities was provided by the Spanish Geographic National Institute. An available proxy of income level was assigned to each zone[143]. The zones were aggregated automatically or by hand according to specific criteria[115] . Thus, areas were aggregated automatically using an algorithm developed with a Geographic Information System program. The three specific steps followed were: 1) Small municipalities were selected for each Spanish Autonomous Community; 2) Estimated income level of all municipalities were classified into four categories: A = "Low income". Income level less than 700,001 pts, B = "Relatively low income". Income level from 700,001 to 880,000 pts, C = "Relatively high income". Income level from 880,001 to 1,100,000 pts, D = "High income". Income level more or equal than 1,100.001 pts; and 3) contiguous small areas with similar income level categories were merged automatically by using the GIS system to reach a minimum population size of 3,500 people.

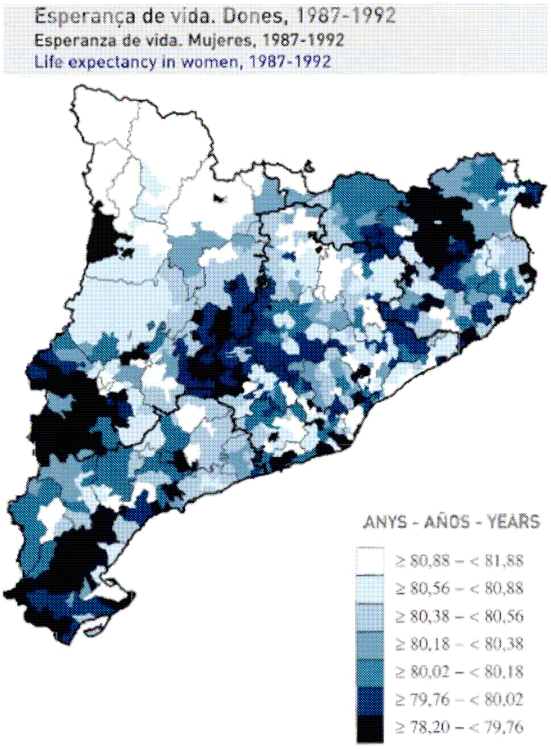
Remaining areas were aggregated by hand using specific rules modified from the criteria proposed by Haining[142]. The three main criteria followed where:

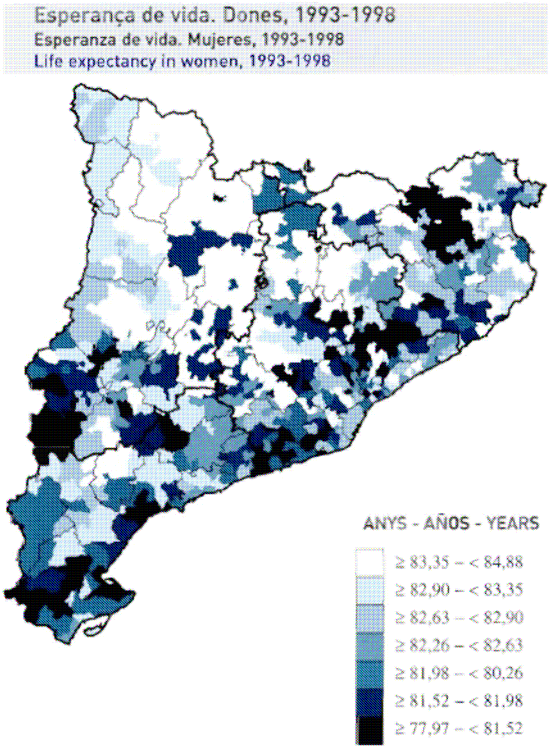
1) The "Island" criterion: A small municipality of one income category, entirely surrounded by a different municipality with a different income category, was absorbed into the surrounding area if that surrounding municipality had less than 10,000 people. If the population size of the surrounding municipality was greater than 10,000, the smaller area was joined using "Income level and proximity criteria".

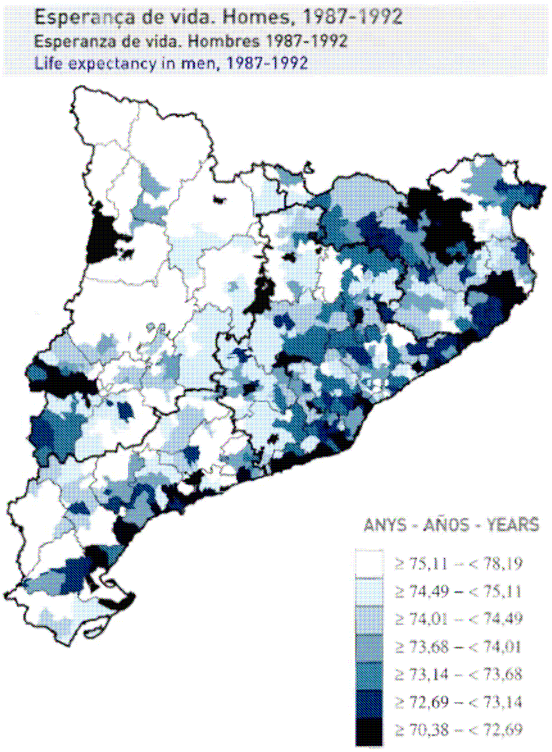
2) The "Neighborhood" criterion: Small municipalities can be joined with those larger neighbouring municipalities (except when they are larger than 10,000 people) having similar income levels.

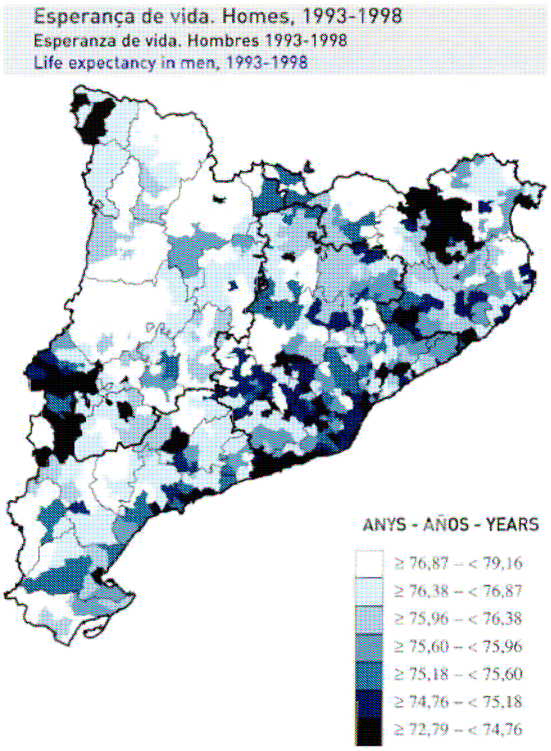
3) The “Income level and proximity” criteria: Municipalities entirely surrounded by other municipalities with more than 10,000 people are joined with non-adjacent areas using the most similar income level and proximity criteria.

A.6 Life expectancy maps.









BIBLIOGRAPHY

1 Koertge N. Curs de filosofia de la ciència (The nature of Scientific Inquiry). Translation by Beltrán J, Roig A, De la Fuente P, López S, Martínez Riu A. Edicions de la Magrana. Barcelona, 1999. [In Catalan].

2 Beaglehole R, Bonita R, Kjellström T. Basic Epidemiology. World Health Organization. Geneva, 1993.

3 English D. Geographical epidemiology and ecological estudies. In: Elliott P, Cuzick J, English D, Stern R (editors). Geographical and environmental epidemiology: methods for small area studies. Oxford: Oxford University Press, 1992:14-21.

4 Elliott P, Wakefield JC, Best NG, Briggs DJ. Spatial epidemiology: methods and applications. In: Elliott P, Wakefield JC, Best NG, Briggs DJ (editors). Spatial epidemiology. Methods and applications. Oxford: Oxford University Press, 2000:3-14.

5 Snow J. On the mode of communication of cholera, 2 ed. New York: The Commonwealth Fund, 1855.

6 Benach J, Yasui Y, Borrell C, Rosa E, Pasarín MI, Benach N, Español E, Martínez JM, Daponte A. Atlas of mortality in small areas in Spain (1987-1995). UPF/MSD, 2001. [Spanish-English].

7 Lawson AB, Browne WJ, Vidal Roderio CL. Disease mapping with WinBUGS and MlwinN. John Wiley and Sons Ltd. England, 2003.

8 Lawson AB, Williams FLR. An introductory guide to disease mapping. John Wiley and Sons Ltd. England, 2001.

9 Diggle PJ. Overview of statistical methods for disease mapping and its relationship to cluster detection. In: Elliott P, Wakefield JC, Best NG, Briggs DJ (editors). Spatial epidemiology. Methods and applications. Oxford: Oxford University Press, 2000:87-103.

10 Martínez-Beneito MA, López Quílez A, Amador Iscla A, Melchor Alós I, Botella Rocamora P, Abellán Andrés C, Abellán Andrés JJ, Verdejo Máñez F, Zurriaga Llorens O, Vanaclocha Luna H, Escolano Puig M. Atlas de mortalidad de la comunidad Valenciana, 1991-2000. Generalitat Valenciana. Conselleria de Sanitat, 2005. [In Spanish].

11 Banerjee S, Carlin BP, Gelfand AE. Hierarchical modeling and analysis for spatial data. London, Chapman & Hall, 2004.

12 Møller J (Editor). Spatial Statistics and Computational Methods. Springer Verlag, Lecture Notes in Statistics, 2003.

13 Benach J, Martínez JM, Borrell C, Pasarín MI, Yasui Y. Desigualtats geogràfiques en arees petites. In: Borrell C, Benach J (editors). Les desigualtats en la salut en Catalunya. Barcelona: Editorial Mediterránea. Fundació Bofill, 2003:57-90. [In Catalan].

14 Morgenstern H. Ecologic studies in epidemiology: Concepts, Principles, and Methods. *Annual Review of Public Health* 1995; 16:61-81.

15 Borrell C, Mompart A, Brugal MT, Rohlfs I, Pérez G. La Salut. In: Fundación Jaume Bofill. Informe per a la Catalunya del 2000, societat, economia, política, cultura. Barcelona: Editorial Mediterránea. Fundació Jaume Bofill, 1999. [In Catalan].

16 Benach J, Yasui Y, Borrell C, Rosa E, Pasarin MI, Benach N, Español E, Martinez JM, Daponte A. Examining geographic patterns of mortality: the atlas of mortality in small areas in Spain (1987-1995). *European Journal of Public Health* 2003 Jun;13(2):115-23.

17 Benach J, Yasui Y. Geographical patterns of excess mortality in Spain explained by two indices of deprivation. *Journal of Epidemiology and Community Health* 1999; 53:423-31.

18 Cuzick J, Elliott P. Small area studies: purpose and methods Geographical epidemiology and ecological studies. In: Elliott P, Cuzick J, English D, Stern R (editors). Geographical and environmental epidemiology: methods for small area studies. Oxford: Oxford University Press, 1992:14-21.

19 Wakefield JC, Best NC, Waller L. Clustering, cluster detection and spatial variation in risk. In: Elliott P, Wakefield JC, Best NG, Briggs DJ (editors). Spatial epidemiology. Methods and applications. Oxford: Oxford University Press, 2000:128-152.

20 Pickle LW, Hermann. Cognitive aspects of statistical mapping. Technical Report 18, NCHS. Washington DC, 1995.

21 Pickle LW, Mungiole M, Jones GK, White AA. Atlas of United States mortality. Hyattsville: National Center for Health Statistics, 1996.

22 Devesa SS, Grauman DJ, Blot WJ, Pennello GA, Hoover RN, Fraumeni JF. Atlas of Cancer Mortality in the United States. 1950-1994. National Cancer Institute, September 1999.

23 Lopez-Abente G, Pollán M, Escolar A, Errezola M, Abraira V. Atlas de mortalidad por cáncer y otras causas en España. Madrid: Fundación Científica de la Asociación Española Contra el Cáncer, 1996. [Spanish-English].

24 Atlas de mortalidad por enfermedades cardiovasculares en la Comunidad Valenciana. http://www.sp.san.gva.es/epidemiologia/ (accesed 24/02/2006). [In Spanish].

25 Domínguez F, Borrell C, Benach J, Pasarin MI. Medidas de privación material en el estudio de las desigualdades sociales en salud en áreas geográficas pequeñas. *Gaceta Sanitaria* Suppl 4:23-33. [In Spanish].

26 Benach J, Martínez JM, Borrell C, Pasarín MI, Muntaner C, Ocaña-Riola R, Yasui Y, Benach N, Daponte A, Buxó M, Vergara M. Evolución temporal de la mortalidad en áreas pequeñas de España (1990-2001). Fundación BBVA (in press). [In Spanish].

27 Morgenstern H. Uses of Ecologic Análisis in Epidemiologic Research. *American Journal of Public Health* 1982; 72 Nº12:1336-1344.

28 Diez-Roux AV. Multilevel Analysis in Public Health Research. *Annual Review of Public Health* 2000; 21:171-92.

29 Diez-Roux AV. A glossary for multilevel analysis. *Journal of Epidemiology and Community Health* 2002; 56:588-594.

30 Kim HJ, Fay MP, Feuer EJ, Midthune DN. Permutation tests for joinpoint regression with applications to cancer rates. *Statistics in Medicine* 2000; 19:335-351 (correction: 2001; 20:655).

31 Sheppard L, Prentice RL, Rossing MA. Design considerations for estimation of exposure effects on disease risk using aggregate data studies. *Statistics in Medicine* 1996; 15(17-18):1849-1858.

32 Basagaña X. Nivell socio-economic i prevalença d'asma en adults joves: un anàlisi multinivell en l'european community respiratory health survey. Proyecto final de carrera. Universitat Politècnica de Cataluña, 2003 [In Catalan].

33 Benach J, Gimeno D, Benavides FG, Martínez JM, Torné MM. Types of employment and health in the European Union: a comparison between two European surveys on working conditions. *European Journal of Public Health* 2004; 14:314-321.

34 Liang KY and Zeger S. Longitudinal Data Analysis Using Generalized Lineal Models. *Biometrika* 1986; 13-12.

35 Liang KY and Zeger S. Regression analysis for correlated data. *Annual Review of Public Health* 1993; 14:43-68.

36 Horton NJ, Lipsitz SR. Review of software to fit generalized estimating equation regression models. *The American Statistician* 1999; 53:160-169.

37 Hardin JW, Hilbe JM. Generalized estimating equations. Chapman and Hall/CRC, 2003.

38 Tuffte ER. The visual display of quantitative information. Graphics Press, Cheshire. Connectitut, 1983.

39 Pickle LW. Preface. In: Benach J, Yasui Y, Borrell C, Rosa E, Pasarín MI, Benach N, Español E, Martínez JM, Daponte A. Atlas of mortality in small areas in Spain (1987-1995). UPF/MSD, 2001. [Spanish-English].

40 Cuzick J, Elliott P. Small-area studies: purpose and methods. In: Elliott P, Cuzick J, English D, Stern R (editors). Geographical and environmental epidemiology: methods for small area studies. Oxford: Oxford University Press, 1992:14-21.

41 Benavides FG, Bolúmar F, Peris R. Quality of death certificates in Spain. *American Journal of Public Health* 1989; 79(10):1352-4.

42 Regidor E. Fuentes de información de mortalidad y morbilidad. *Medicina Clínica* 1992; 99(5):183-187. [In Spanish].

43 Monmonier M. How to lie with Maps. The University of Chicago Press. Second Edition, 1996.

44 MacEachren, AM. How maps Work: Representation, Visualization and Design. New York: Guilford Press, 1995.

45 Brewer CA, McEachren AM, Pickle LW. Evalution of Map color schemes for the NCHS Mortality Atlas. In: Proceedings of the International Symposium on Computer Mapping in Epidemiology and Enviromental Health. Tampa (Florida), 1995. world computer Graphics Foundation and the University of South Florida, 1997:14-20.

46 Elant-Johnson RC. Definition of rates:some remarks on their use and misuse. *American Journal of Epidemiology* 1975; 102:267-271.

47 Bolúmar F. Medición de los fenómenos de salud y enfermedad en epidemiología. In: Piedrola Gil. Medicina preventiva y salúd pública. Masson, 2001:71-79. [In Spanish].

48 Benavides FG. La medición en epidemiología. In: Martínez Navarro F, Anto JM, Castellanos PL, Gili M, Marsé P, Navarro V. (Editors) Salud Pública. Madrid: Mc Graw Hill/Interamericana, 1998:139-164. [In Spanish].

49 Sierra A, Doreste JL, Almaraz A. Demografía dinámica (I): natalidad, fecundidad y mortalidad. In: Piedrola Gil. Medicina preventiva y salúd pública. Masson, 2001:27-43. [In Spanish].

50 Szklo M, Nieto FJ. Epidemiology. Beyond the basics. Gaithersburg: Aspen Publication, 2000.

51 Inskip H, Beral V, Fraser P. Methods for age-adjusment of rates. *Statistics in Medicine* 1983; 2:455-466.

52 Breslow NE, Day NE. Statistical Methods in cancer research. Volume II-The design and analysis of cohort studies. International Agency for reseach on cancer, 1987.

53 Inskip H. Standardization Methods. In: Gail MH, Benichou J (editors). Encyclopedia of epidemiologic methods. John Wiley and Sons, 2000:871-884.

54 Breslow NE, Day NE. Indirect standardization and multiplicative models for rates, with reference to the age adjusment of cancer incidence and relative frequency data. *Journal of Chronic Diseases* 1975; Vol 28:289-303.

55 Pascutto C, Wakefield JC, Best NG, Richardson S, Bernardinelli L, Staines A, Elliot P. Statistical issues in the analysis of disease mapping data. *Statistics in Medicine* 2000; 19:2493-2519.

56 Ferrándiz J, López-Quílez A, Gómez-Rubio V, Sanmartín P, Martínez-Beneito MA, Melchor I, Vanaclocha H, Zurriaga O, Ballester F, Gil JM, Pérez-Hoyos S, Abellán JJ. Statistical relationship between hardness of drinking water and cerebrovascular mortality in Valencia: a comparison of spatiotemporal models. *Environmetrics* 2003; 14:491-510.

57 Goldman DA, Brender JD. Are standardized mortality ratios valid for public health data analysis? *Statistics in Medicine* 2000;19:1081-1088.

58 Wakefield JC, Best NG, Waller L. Bayesian approach to disease mapping. In: Elliott P, Wakefield JC, Best NG, Briggs DJ (editors). Spatial epidemiology. Methods and applications. Oxford: Oxford University Press, 2000:3-14.

59 Clayton D, Kaldor J. Empirical Bayes estimates of age-standardized relative risks for use in disease mapping. *Biometrics* 1987; 43:671-681.

60 Mantel N, Stark CR. Computation of indirect-adjusted rates in the presence of confounding. *Biometrics* 1968; 24:997-1005.

61 Clayton D, Bernardinelli L. Bayesian methods for mapping disease risk. In: Elliott P, Cuzick J, English D, Stern R (editors). Geographical and environmental epidemiology: methods for small area studies. Oxford: Oxford University Press, 1992:205-220.

62 Molliè A. Bayesian Mapping of disease. In: WR Gilks, Richardson S, Spiegelhalter DJ, eds. Markov Chain MonteCarlo in Practice. London: Chapman and Hall, 1995:359-379.

63 Lawson AB, Biggeri AB, Boehning D, Lesaffre E, Viel J-F, Clark A, Schlattman P, Divino F. Disease mapping models: an empirical evaluation. *Statistics in Medicine* 2000; 19:2217-2241.

64 Mollié A. Bayesian mapping of Hodgkin's disease in France. In: Elliott P, Wakefield JC, Best NG, Briggs DJ (editors). Spatial epidemiology. Methods and applications. Oxford: Oxford University Press, 2000:267-285.

65 Bernardinelli L, Montomoli C. Empirical Bayes versus fully Bayesian analysis of geographical variation in disease risk. *Statistics in Medicine* 1992;11:983-1007.

66 Robert CP. The Bayesian Choice: from Decision-Theoretic Motivations to Computational Implementation. Second edition. Springer-Verlag, New York. 2001.

67 Carlin BP, Louis TA. Bayes and Empirical Bayes methods for data analysis. London: Chapman and Hall, 1996.

68 Procedure NLMIXED. SAS@ version 8. SAS/STAT User's Guide. SAS Institute Inc.

69 Molliè A, Richardson S. Empirical Bayes estimates of cancer mortality rates using spatial models. *Statistics in Medicine* 1991; 10:95-112.

70 Breslow NE, Clayton DG. Aproximate inference in generalized linear mixed models. *Journal of the American Statistical Association* 1993; 88, 421:9-25.

71 Littell RC, Milliken GA, Stroup WW, Wolfinger RD. SAS@ System for Mixed Models, Cary, NC: SAS Institute Inc., 1996. 633 pp.

72 Gilks WR, Richardson S, Spiegelhalter DJ (editors). Markov Chain MoteCarlo in practice. London: Chapman and Hall, 1996.

73 Smith AFM, Gelfand AE. Bayesian statistics without tears: a sampling-resampling perspective. *The American Statistician* 1992; 46:84-8.

74 Bithell JF, A classification of disease mapping methods. *Statistics in Medicine* 2000; 19:2203-2215.

75 Gelfand AE, Sahu SK. Identifiability, Improper Priors, and Gibbs Sampling for Generalized Linear Models. *Journal of the American Statistical Association* 1999; Vol 94. No 445, Theory and Methods: 247-253.

76 Eberly LE, Carlin BP. Identifiability and convergence issues for Markov chain Monte Carlo fitting of spatial models. *Statistics in Medicine* 2000; 19:2279-2294.

77 Thomas A, Best N, Arnold R, Spiegelhalter D. Geobugs user manual. Demonstration Version 1.0. April 2000.

78 Thomas A, Best N, Arnold R, Spiegelhalter D. Geobugs user manual. Version 1.1. Beta. June 2002.

79 Besag JE. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, Series B. 1974; 36:192-236.

80 Besag J, Kooperberg CL. On conditional and intrinsic autoregressions. *Biometrika* 1995; 82:733-746.

81 Böhning D. Computer-assisted analysis of mixtures and applications meta-analysis, disease mapping, and others. Boca Raton, Fla. London: Chapman and Hall, 1999.

82 Schlattman P, Böhning D. Mixture models and disease mapping. *Statistics in Medicine* 1993;12: 1943-1950.

83 Böhning D, Schlatman P. Computer-Assisted Analysis of Mixtures (C.A.MAN): Statistical Algorithms. *Biometrics* 1992; 48:283-303.

84 Schlatman P, Dietz E, Böhning D. Covariate adjusted mixture models and disease mapping with the program dismapwin. *Statistics in Medicine* 1996; 15:919-929.

85 Laird N. Nonparametric maximum likelihood estimation of a mixing distribution. *Journal of the American Statistical Association* 73; 805-811.

86 Clèries R, Ribes J, Moreno V, Martínez JM, Bosch FX. Meta analysis: Dealing with heterogeneity and discrete distributions. XXI Reunión Científica de la Sociedad Española de Epidemiología (Toledo 1-4 Octubre 2003).

87 Yasui Y, Lele S. A regression method for spatial disease rates: an estimating function approach. *Journal of the American statistical association* 1997; Vol 92, No 437. Applications and Case Studies: 21-32.

88 Besag J, York j, Mollié A. Bayesian image restoration, with two applications in spatial statistics. *Annals of the Institute of Statistical Mathematics* 1991; 43:1-59 (With discussion).

89 Bernardinelli L, Clayton D, Montomoli C. Bayesian estimates of disease maps: How important are priors? *Statistics in Medicine* 1995; 14:2411-2431.

90 Kesall JE, Wakefield JC. Discussion of "Bayesian models for spatially correlated disease and exposure data", by Best et al. In Bayesian statistics 6. Bernardo JM, Berger JO, Dawid AP, Smith AFM (eds), Oxford: Oxford University Press: p 151.

91 Gelman A. Prior distributions for variance parameters in Hierarchical models. Bayesian Analysis 2005; 1, Nº2:1-19.

92 Spiegelhalter DJ, Thomas A, Best NG, Carlin BP, Lunn D. WinBugs User Manual. Version 1.4. Cambridge, England: MRC Biostatistics Unit. 2002a.

93 Best N, Richardson S, Thomson A. A comparison of Bayesian spatial models for disease mapping. *Statistical Methods in Medical Research* 2005; 14:35-59

94 Knorr-Held, L., Raßer, G. and Becker, N. Disease mapping of stage-specific cancer incidence data. *Biometrics* 2002; 58:492-501.

95 Bernardinelli L, Clayton D, Pascutto C, Montomoli C, Ghislandi M, Songini M. Bayesian analysis of space-time variation in disease risk. *Statistics in Medicine* 1995; 14:2433-2443.

96 Yasui Y, Liu H, Benach J, Winget M. An empirical evaluation of various priors in the empirical Bayes estimation of small area disease risks. *Statistics in Medicine* 2000; 19:2409-2420.

97 Militino AF, Ugarte MD, Dean CB. The use of mixture models for identifying high risks in disease mapping. *Statistics in Medicine* 2001; 20(13):2035-2049.

98 Spiegelhalter DJ, Best NG, Carlin BP, Van der Linde A . Bayesian deviance, the effective number of parameters and the comparison of arbitrarily complex models. *Journal of the Royal Statistical Society* B 2002a; 64:583-640.

99 Stern HS, Cressie NA. Inference for extremes in disease mapping. In: Lawson AB, Böhning D, Lasaffree E, Biggeri A, Viel JF, Bertolline R (editors). Disease mapping and risk assessment for Public Health. Chichester. John Wiley and Sons, Ltd, 1999.

100 Leyland H, Goldstein H. Multilevel modelling of health statistics. Chichester: John Wiley and Sons, Ltd, 1999.

101 Langford I, Leyland A, Rasbash J, Goldstein H. Multilevel Modelling of the geographical distribution of rare diseases. *Journal of the Royal Statistical Society* 1999; 48:253-268.

102 Browne WH, Goldstein H, Rasbash J. Multiple membership multiple classification (MMMC) models. *Statistical Modelling* 2001;1:103-124.

103 Spiegelhalter DJ, Thomas A, Best NG, Gilks WR. BUGS: Bayesian Inference Using Gibbs Sampling, Versión 0.5. Cambridge, England: MRC Biostatistics Unit. 1995.

104 Spiegelhalter DJ, Thomas A, Best NG, Gilks WR. BUGS: Bayesian Inference Using Gibbs Sampling, Versión 0.5. Cambridge, England: MRC Biostatistics Unit. 1995.

105 R Development Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org, 2005.

106 Best NG, Cowles MK, Vines SK. CODA Manual versión 0.30. Cambridge, England: MRC Biostatistics Unit, 1995.

107 Smith B. BOA, BUGS Output Análisis Program. The University of Iowa College of Public Health, 1999.

108 Schlattman P, Böhning D. Computer Packages C.A.MAN (Computer assisted mixture analysis) and dismap. *Statistics in Medicine* 1993;12:1965.

109 Pickle LW, Mungiole M, Jones GK, White AA. Exploring spatial patterns of mortality: the new atlas of United States mortality. *Statistics in Medicine* 1999; Dec 15;18(23):3211-20.

110 Knorr-Held L, Besag J. Modelling risk from a disease in time and space. *Statistics in Medicine* 1998; 17:2045-60.

111 MacNab YC, Dean CB. Spatio-temporal modelling of rates fo the construction of disease maps. *Statistics in Medicine* 2002; 21:347-358.

112 Böhning D. Empirical Bayes estimators and non-parametric mixture models for space and time-space disease mapping and surveillance. *Environmetrics* 2003; 14:431-451.

113 Pickle LW. Exploring spatio-temporal patterns of mortality using mixed effects models. *Statistics in Medicine* 2000; 19:2251-2263.

114 Ocaña-Riola R, Saez M, Sachez-Cantalejo C, Barcelo MA, Fernandez A, Saurina C; Grupo AMCAC. Research protocol for the mortality atlas of the provincial capitals of Andalusia and Catalonia (AMCAC Project). *Revista Española de Salud Pública* 2005 Nov-Dec; 79(6):613-20. [In Spanish].

115 Benach J, García MD, Donado-Campos J. Gis for Mapping Mortality Inequalities in Spain and its Socioeconomic Determinants. Constructing Regions using Small Areas. In: Proceedings of the International Symposium on computer Mapping in Epidemiology and Environmental Health 1995. Tampa, Florida (USA), 1997:314-22.

116 Dos Santos Silva I. Cancer Epidemiology: principles and methods. International Agency for Research on Cancer: Lyon 1999.

117 Pinheiro JC, Bates DM. Approximations to the Log-likelihood Function in the Nonlinear Mixed-effects Model. *Journal of computational and Graphical Statistics* 1995; 4:12-35.

118 Booth JG, Hobert JP. Standard Errors of Prediction in Generalized Linear Mixed Models. *Journal of the American Statistical Association* 1998; 93:262-272.

119 Congdom P. A life table approach to small area health need profiling. *Statistical Modelling* 2002; 2:63-68.

120 Jagger C, Hauet E, Brouard N. Health expectancy calculation by the Sullivan method: A practical Guide. European Concerted Action on the Harmonization of Health Expectancy Calculations in Europe (EURO-REVES), 1997.

121 Walter SD. Disease mapping: a historical perspective. In: Elliott P, Wakefield JC, Best NG, Briggs DJ (editors). Spatial epidemiology. Methods and Applications. Oxford: Oxford University Press, 2000:223-239.

122 Anàlisi de la mortalitat de Catalunya, 2002. Barcelona: Servei d'informació i Estudis. Direcció General de Recursos Sanitaris. Departament de Salut. Barcelona 2004. http://www.gencat.net/salut/depsan/units/sanitat/pdf/evomor02.pdf (accessed 24/02/06).

123 Semenciw RW, Le ND, Marrett LD, Robson DL, Turner D, Walter SD. Methodological issues in the development of the Canadian Cancer Incidence Atlas. *Statistics in Medicine* 2000; 19:2437-49.

124 Subramanian SV, Jones K, Duncan C. Multilevel Methods for Public Health Research. In: Neighborhoods and health. Kawachi I, Berkman LF (Editors). Oxford University Press, 2003.

125 Breslow NE, Clayton DG. Aproximate inference in generalized linear mixed models. *Journal of the American Statistical Association* 1993; 88,421:9-25.

126 Rose G. Sick Individuals and sick populations. *International Journal of Epidemiology* 1985; 14:32-38.

127 Prentice RL, Sheppard L. Aggregate data studies of disease risk factors. *Biometrika* 1995; 82:113-125.

128 Sheppard L, Prentice RL. On the reliability and Precision of within- and between-population estimates of relative rate parameters. *Biometrics* 1995; 51:853-863.

129 Godambe VP, Kale BK. Estimating function: an overview. In: Estimating functions. Godambe VP (Editor). Oxford University Press, USA 1991.

130 Desmond AF. Quasi-likelihood, stochastic processes, and optimal estimating equations In: Estimating functions. Godambe VP (Editor). Oxford University Press, USA 1991.

131 McCullagh P. Quasi-likelihood functions: A review of some properties, examples and outstanding problems. Technical Report Nº267. Department of Statistics. University of Chicago, September 1989.

132 Sheppard L. Insights on bias and information in group-level studies. *Biostatistics* 2003 Apr; 4(2):265-278.

133 Liang K-Y, Zeger S. Longitudinal data analysis using generalized linear models. *Biometrika* 1986; 73:13-22.

134 Henderson HV, Searle SR. On deriving the inverse of a sum of matrices. *Society for industrial and applied mathematics* 1981; 23(1):53-60.

135 Zeger S, Liang K-Y. Longitudinal data analysis for discrete and continuous outcomes. *Biometrics* 1986; 42:121-130.

136 Villanueva CM, Fernández F, Malats N, Grimalt JM and Kogevinas M. Meta-analysis of studies on individual consumption of chlorinated drinking water and bladder cancer. *Journal of Epidemiology and Community Health* 2003; 57:166-173.

137 Jackson C, Best N, Richardson S. Improving ecological inference using individual-level data. *Statistics in Medicine* 2005 (in press).

138 Godambe VP, Thompson ME. An extension of quasi-likelihood estimation. *Journal of Statistical Planning and Inference* 1989; 22:137-152.

139 Royall RM. Model robust confidence intervals using maximum likelihood estimators. *International Statistical Review* 1986; 54:221-226.

140 Reilly M, Pepe M. A mean score method for missing and auxiliary covariate data in regression models. *Biometrika* 1995; 82, 2:299-314.

141 Reading RF, Openshaw S, Jarvis SN. Measuring child health inequalities using aggregations of Enumeration Districts. *Journal of Public Health Medicine* 1990;12:160-167.

142 Haining R, Wise S, Blake M. Constructing regions for small area analysis: material deprivation and colorectal cancer. *Journal of Public Health Medicine* 1990:12:160-167.

143 Banesto (Banco Español de Crédito). Anuario del Mercado Español. Madrid: Banesto, 1993.