Storck:

Pattern mixture models for multivariate normal data: a simulation study

Sonderforschungsbereich 386, Paper 191 (2000)

Online unter: http://epub.ub.uni-muenchen.de/

Pattern mixture models for multivariate normal data: a simulation study

S. Storck [1]

April 10, 2000

Abstract

Little and Wang (1996) introduced the pattern mixture model for wave nonresponse as a special case of multivariate normal longitudinal data with fixed covariate matrix. This paper was the theoretical foundation and the induce to investigate the pattern mixture model compared with complete case analysis by means of simulations. The main point of interest was the mean square error of the estimated model parameters and the efficiency of the estimations. To estimate the variance of the model parameters we examine the Jackknife method. Parameter estimates by the pattern mixture model are very satisfying under ignorable mechanism but they have to be scanned carefully under nonignorable mechanism. The Jackknife method seems to be, with restrictions, a good estimator for the variance of the model parameters.

1 Introduction

In longitudinal data we can't prevent to have wave nonresponse in our measure points caused by the absence of subjects during the study. One approach to treat this lack of information is to replace nonresponse with Maximum Likelihood (ML) methods (Little and Rubin, 1987). Selection models (SM) and pattern mixture models (PMM) are two approaches belonging to this kind of strategy. The paper about a new class of pattern mixture models for multivariate incomplete data (Little, 1993) introduces an arbitrary unspecified missing function of a linear combination of two variables. Little (1994) has a look on PMM in the bivariate normal case and extent this idea in Little and Wang (1996) to multivariate normal data with fixed covariate matrix.

The aim of this paper is to show how influence factors, which can not be influenced by the analyst, will influence the estimate of the parameters in the PMM. The subject of our investigation were the influence of correlation among the measure points, number of subjects and share of missing values. In one investigation we also examine the effect when misspecification of the missing

1 Institute of Statistics, Ludwig-Maximilians University of Munich, Germany, email:storck@stat.uni-muenchen.de

mechanism occurs. Our valuation criteria were the mean square error (MSE) and the exactness of the estimates (efficiency). Another investigation field was the assessment of the Jackknife method by comparing the estimated with the theoretical confidence interval. All surveys were made for the ignorable and nonignorable nonresponse and compared with the available case (AV) and complete case (CC) method.

In section 2 we want to have a short look on the theory of PMM. In Section 3 we will give an introduction to the C++ Program. The valuation criteria and the structure of the simulation is described in Section 4 and 5. In Section 6 we present the results.

2 Theory

We want to analyze longitudinal data with one dropout point. So we have a data Matrix Y including n subjects measured at T time points with a fixed covariate matrix X including the treatment group of the subject. To separate the completers from subjects dropped out we introduce a missing indicator variable m with $m = 0$ for completers and $m = 1$ else. Additional we want to divide the data matrix in two parts: Y_1 including T_1 complete observed time points and Y_2 including T_2 remaining time points.

Using ML methods we have to factorize the joint distribution of Y and m given X. So we have two possibilities: the selection model

$$f(Y, m \mid X) = f(Y \mid X) f(m \mid Y, X)$$

and the pattern mixture model

$$f(Y, m \mid X) = f(m \mid X) f(Y \mid m, X) \tag{1}$$

A typical feature of PMM is that the distribution of Y has to be specified for every strata indicated by m and that the distribution of the missing mechanism is independent of Y. In contrast SM demands to estimate the marginal distribution of Y given X and the distribution of m depends on Y.

We want to have a closer look to the PMM factorization (1). It could be easily seen, that we are able to estimate the parameters of $f(m \mid Y, X)$ by logistic regression from m on X. Not so $f(Y \mid m, X)$, it has to be factorized again, due to the lack of moments relating to the data of not completers and missing time points. To get unbiased estimates of these model parameters we have to face the missing mechanism. We separate two main groups: the ignorable mechanism depending on the full observed time points:

$$P(m = 1 \mid Y_1, Y_2, X) = P(m = 1 \mid Y_1, X) \tag{2}$$

and the nonignorable mechanism depending on the unobserved outcome and the covariates

$$P(m = 1 \mid Y_1, Y_2, X) = P(m = 1 \mid Y_2, X) \tag{3}$$

(2) and (3) will be changed to identifying restrictions. In the ignorable case (IM) the factorization and the restriction will lead to a just-identified solution of the ML equations. In the nonignorable case we have to distinguish the estimates by the dimension of T_1 and T_2. Is $T_1 = T_2$ then we got a just-identified ML model. We want to call this situation NIMED (nonignorable mechanism with equal dimension). A iterative method is necessary when we have an overidentified model $T_1 > T_2$ called NIMUD (nonignorable missing mechanism with unequal dimension). For this situation we used the EM algorithm. The third situation could be reduced to the the previous ones by deleting one or more time points until we have the situation $T_1 \leq T_{2*}$.

3 C++ program

A useful tool for multivariate regression is the sweep-operator from Dempster (1969). Here we follow the suggestion of Little and Wang (1996) to implement the PMM. The algorithm is implemented using C++ (Stroustrup, 1997). Basic data structures were supplied by the library of Fieger, Heumann, Kastner and Watzka (1997). Random numbers are generated using the DRAND48 generator, which is supplied by SunOS 5.5 as a C-library function (SunOS, 1995, man Pages(3C)).

In the following we want to present all new implemented class. Existing basic calculations were supplied by library of Heumann, Fieger and Kastner (1998). The new implemented classes for the simulation are presented in detail in the appendix. An overview of the course of the program is given in Figure 1-2

check Analyses the simulations by calculating the MSE, theoretical confidence intervall, estimated confidance intervall and the kernel distribution estimate.

ergebnis This class transforms the C++ output to a TeX document. Text and numbers have to be handed over by a variable or matrix.

completecase Calculates the CC and AV estimator.

sweepelements Calculates the PMM estimates by the method of Little and Wang (1996)

aufbereitung Edits the data for the calculations.

initial Produces the random sample data.

sweepoperator Calculates the sweep-operator on row and column.

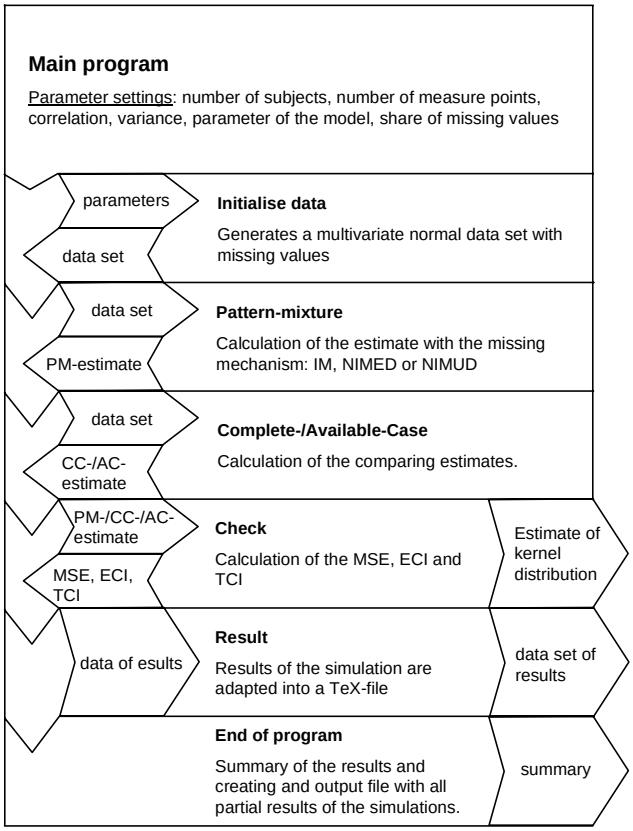

Figure 1: The main program

4 The valuation criterias for the estimators

We decided to choose the mean square error to compare the three estimators. This choice is theoretical based on the matrix-valued MSE criterion as discussed in full detail e. g. in Rao and Toutenburg (1999). The MSE will cover as well bias as variance of the estimates. So we will not assess an estimator higher than another when he has little bias but too strong variance and the other way round. Let $\hat{b}$ be the estimated and b the real parameter vector of the model. To calculate the MSE we used the equation:

$$\mathrm{E}\left\{\left(\hat{b}-b\right)\left(\hat{b}-b\right)'\right\}$$

The exactness of the estimate was assessed by the covering frequency of the theoretical confidence interval (TCI) over the estimated parameters.

$$\left[b - t^{(1-\alpha/2)}\sqrt{\mathrm{V}(b)}, b + t^{(1-\alpha/2)}\sqrt{\mathrm{V}(b)}\right], \tag{4}$$

where $\mathrm{V}(b)$ is the real variance of the parameter b and $\alpha = 0,05$. We want to call this the efficiency of the model.

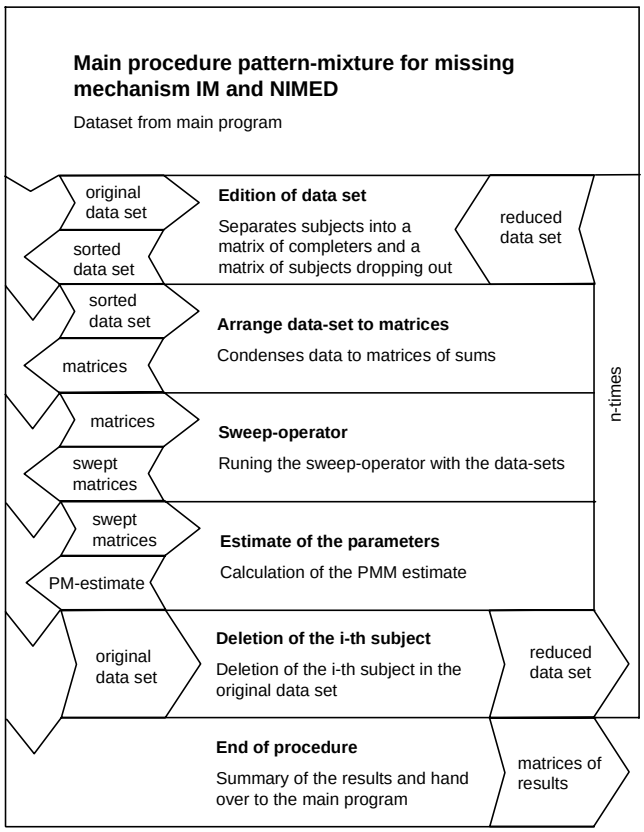

Figure 2: The main procdure for IM and NIMED

Another investigation field was to assess the Jackknife method (Efron and Tibshirani, 1993), which estimates the variance of $\hat{b}$. Therefore we compared the TCI with the estimated confidence interval (ECI)

$$\left[\hat{b} - t^{(1-\alpha/2)}\sqrt{\mathrm{V}(\hat{b})}, \hat{b} + t^{(1-\alpha/2)}\sqrt{\mathrm{V}(\hat{b})}\right], \tag{5}$$

where $\mathrm{V}(\hat{b})$ is the variance of $\hat{b}$ estimated by the Jackknife method and $\alpha = 0,05$.

5 Design of the simulation study

Aim of the survey was to investigate small sample properties and efficiency of PMM compared with those of complete case (CC) and available case (AC) estimator. Therefore, the conduct of the estimators was investigated by changing the frame conditions of the data namely correlation structure between time points, number of subjects, missing mechanism and share of missing values. The second investigation unit became necessary, due to appearing problems in the nonignorable case. Further we investigate the small sample properties under misspecification with regard to the missing mechanism.

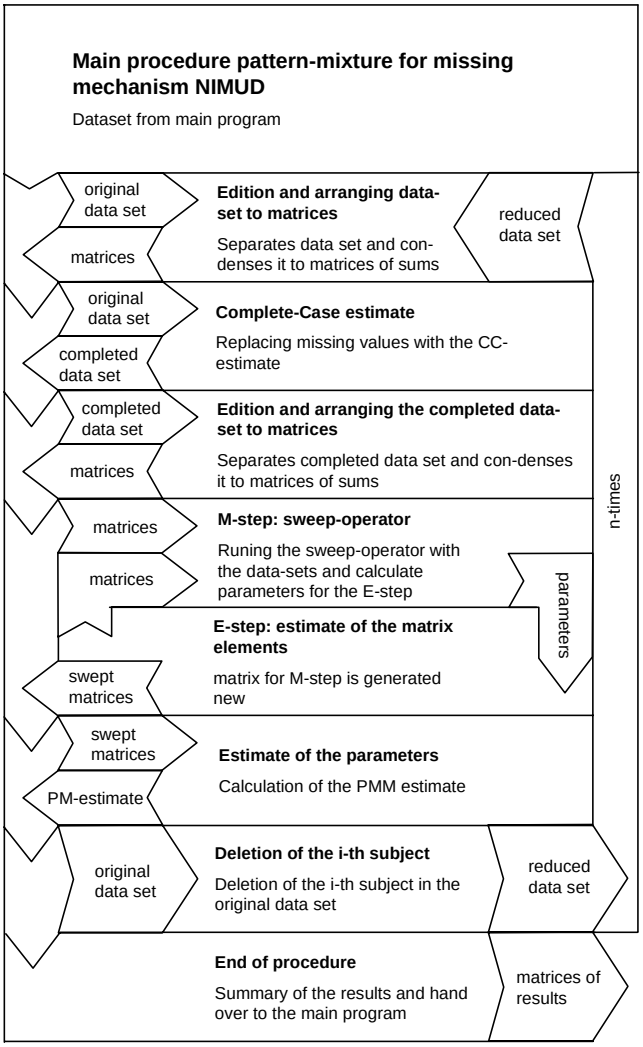

Figure 3: The main procedure for NIMUD

Following the example in Little and Wang (1996) we create datasets with three treatment groups when we have 250 subjects and two treatment groups when we have 50 subjects. The regression parameters are related on the time points and the treatment group. The distribution of the error term was multivariate normal. The regression parameters were modeled like in Figure 4 shown.

After data generation missing values were produced using the function $P(m = 1 \mid Y_1, Y_2, X) = aY_1 + bY_2 + cX$. With the suitable specification of a, b and c the missing mechanism mentioned above can be simulated. With every simulation adjustment we investigate the three missing mechanisms IM, NIMED and NIMUD. Every single simulation contains 1000 runs under the same adjustment. We want to give a short view on the different adjustments for the three

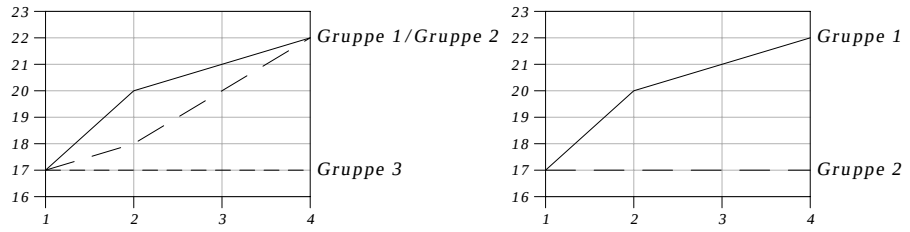

Figure 4: model 1 with three and model 2 with two treatment groups. Measure parameters depending on measure time points.

investigation units.

Unit one has to compare CC, AC and PMM in regard to the MSE criterion. For that we combine the adjustments: number of subjects ($n \in \{50, 250\}$), share of missing value ($mis \in \{30\%, 60\%\}$), correlation ($\rho \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$). Hence we have 60 Simulations in our first unit (Table 1).

number of subjects	share of missing values	missing mechanism	correlation
250/50	0.3 /0.6	IM/NIMED/NIMUD	0.1/0.2/0.5/0.7/0.9

Table 1: Adjustments for investigation unit one

One occurring problem during the simulation was that the NIMED type delivered very bad results when correlation was low. So we decided to calculate NIMED with the EM algorithm of NIMUD. This is subject of our second investigation with ten simulations (Table 5). Possibilities of adjustments were $n = 250$ with $mis = 30\%$ or $n = 50$ with $mis = 60\%$ combined with $\rho \in \{0.1, 0.2, 0.5, 0.7, 0.9\}$

share of missing values number of subject with	missing mechanism	correlation
250 mit 0.3/50 mit 0.6	NIMED	0.1/0.2/0.5/0.7/0.9

Table 2: Adjustments for investigation unit two

In the third unit, containing 15 simulations, we were interested in what will happen if the prior assumption is wrong (Table 5). For this case we chose fixed assumptions with $n = 250$, $mis = 30\%$ and $\rho = 0.5$ and combine those with the extent of misspecification. The misspecification will range from 0 (which means right assumption) to 1 (total wrong assumption). Between this invervall (namely 0.2, 0.4, 0.6, 0.8), missingness will depend on Y_1 and Y_2 in different weights.

missing mechanism	extent of misspecification
IM/NIMED/NIMUD	0.2/0.4/0.6/0.8/1

Table 3: Adjustments for investigation unit three

6 Results

For AV and PMM there was no reason to complain for the estimate of the parameters for the completely observed time points, right through all missing mechanisms. For CC we could see, that only under low correlation and nonignorable mechanism the estimate have a MSE close to zero. For the ignorable mechanism we notify that even at a very low correlation the CC has a high MSE, which will increase with a stronger correlation (Figure 5). The efficiency for the

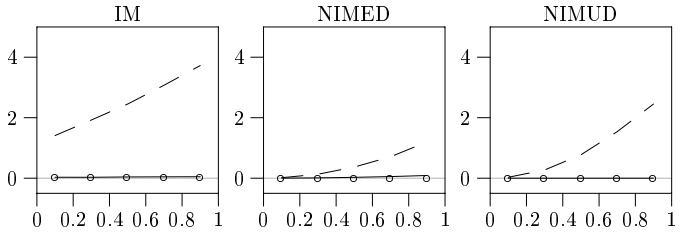

Figure 5: MSE of the full observed time points depending on the correlation. Investigation unit one with 250 subjects and 30% of missing values. line: PMM; dashed line: CC; circles: AC.

AC estimator is very satisfying. It is always near to 95%. Not so the PMM the efficiency was in the IM and NIMED case less efficient than the AC estimator. In the NIMUD case PMM and AC were equal. Of course the efficiency of CC got increasingly bad.

For the incomplete time points, as one can see in Figure 6, AV and CC deliver only under low correlation and IM assumption estimates with low MSE and high efficiency. If missingness depends on Y_2 (NIMED and NIMUD) AC and CC have a stable high MSE and stable low efficiency independent on the correlation. The share of missing values make it even worse, but it seems that the number of subjects is not relevant for the height of the MSE and the efficiency in this case. PMM estimated the parameters of the incomplete time points to our complete satisfaction under IM. Even a low number of subjects with a high share of missing values didn't influence the MSE and the efficiency, it was always zero or near to 95%. In contrast the PMM did disappointingly on the estimates in the NIMED case. We notified, that the MSE will become close to zero only when the correlation grows to a particular strength. This was with 250 subjects about 0.3 and with 50 subjects about 0.7, independent on the share of missing values. Under this correlation barrier the PMM is even worser than the AC and CC model. But it could be mentioned that the efficiency was not too bad in this case. A more positive idea of PMM we got under the NIMUD

assumption, where the parameters were estimated via the EM algorithm. Under a low correlation we could see, that the PMM is of the same quality as the AC or CC estimates. But with increasing correlation PMM will get much better than the AC or the CC model. Even though we reached a MSE close to zero only under the very high correlation of 0.9. In an analogous way the efficiency got only under a high correlation of 0.9 close to 95%.

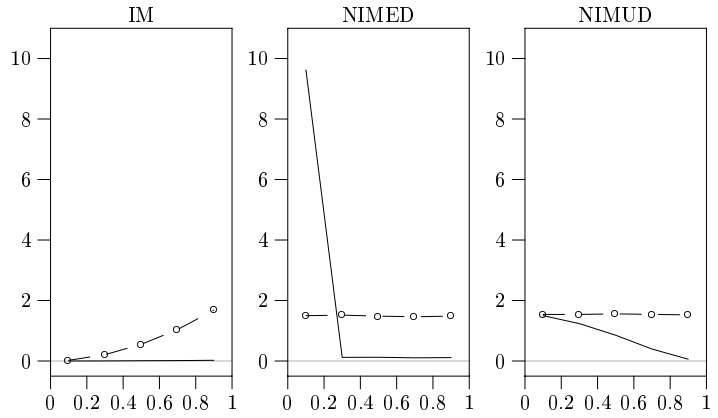

Figure 6: MSE of the incomplete time points depending on the correlation. Investigation unit one with 250 subjects and 30% of missing values. Line: PMM; dashed line: CC; circles: AC.

Because of the very bad result in the NIMED case under low correlation structure in the data, we made the next investigation unit and tried to get better results, when we calculate the NIMED case with the EM algorithm of the NIMUD case. Here we just want to have a look on the interesting incomplete time points. The statement we can make is that the EM algorithm is only an advantage only in case that we have a low correlation. In our model with 250 subjects and a share of missing values about 0.3 the correlation barrier is about 0.3, then the exact estimate method will be better than the asymptotic one (Figure 7), but we can say that it is always a loss of efficiency using the EM algorithm.

Another important question was what will happen, when me make the wrong assumption about the missing mechanism. The estimate of the complete time points are not affected by this assumption, so we have only a look on the incomplete time points. Starting with IM we could see, that the MSE of the PMM was increasing while we intensified the wrong assumption, nevertheless the MSE was always smaller than those of AC and CC (Figure 8). The height of efficiency react in the same way. While increasing the misspecification the efficiency got worser. At the NIMED case we notified a very strong reaction with respect to the wrong assumption. Already with a medium misspecification of the missing mechanism we had a worser MSE and a lower efficiency than AC and CC. For the NIMUD type of missingness we can remark, that a wrong assumption have only a positive effect on the MSE and the efficiency. As these simulations run

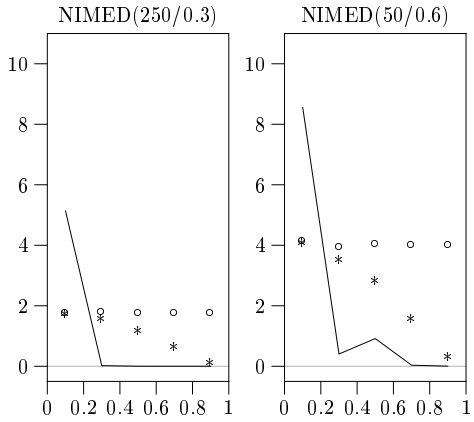

Figure 7: MSE of the incomplete time points depending on the correlation using the EM algorithm for NIMED situation. Investigation unit two compares simulations having 250 subjects and 30% of missing values with 50 subjects and 60% of missing values. Line: PMM; stars: PMM with EM algorithm; circles: AC.

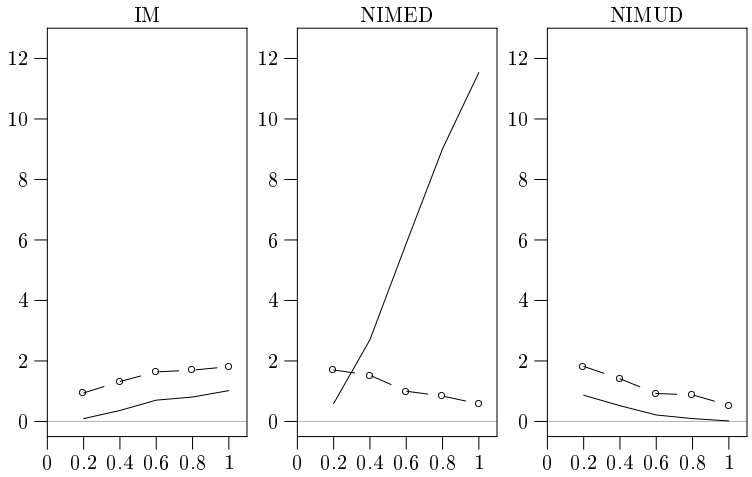

Figure 8: MSE of the incomplete time points depending on the extent of misspecification of the missing mechanism. Investigation unit three with 250 subjects and 30% of missing values. Line: PMM; dashed line: CC; circles: AC.

under the fixed adjustments of $n = 250$, $\rho = 0.5$ and $mis = 0.3$ we could proceed from the assumption that an increasing correlation or share of missing values have the same effects on the estimate quality like described above.

The Jackknife method was a good tool to estimate the variance of the parameters except of PMM in the NIMED case for the incomplete time points. Otherwise right through all missing mechanisms and investigation units the ECI was close to the TCI.

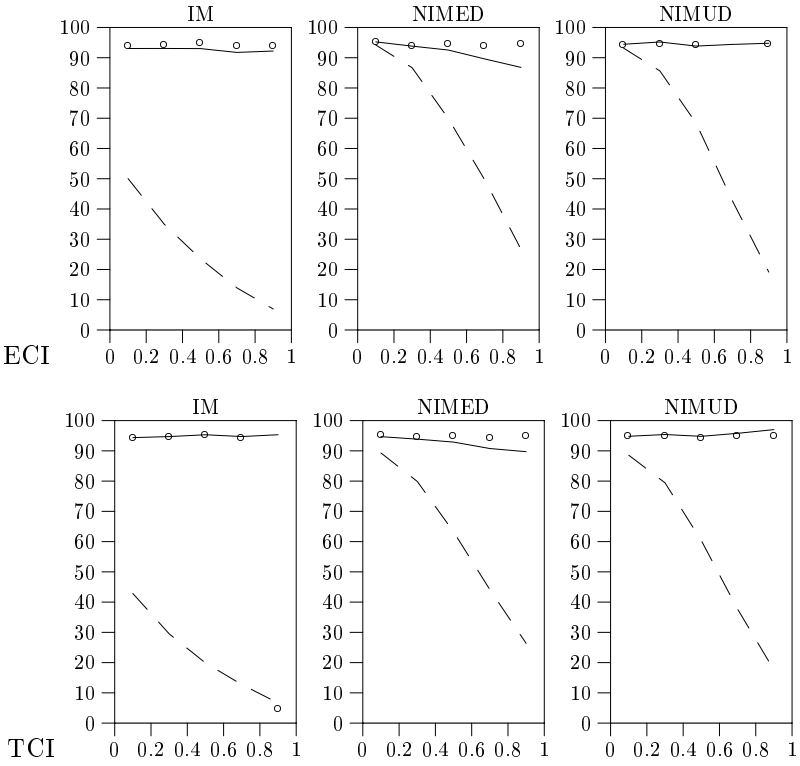

Figure 9: Efficiency of the full observed time points depending on the correlation. Investigation unit one with 250 subjects and 30% of missing values. line: PMM; dashed line: CC; circles: AC.

A The classes

A.1 check

Analyses the simulations.

public

check (const matrix& mat_mu, const realArray& ar_varianz, const matrix& beta, const matrixArray2D& gesamt_block, const matrix& mat_varbeta, const intArray& para, const int& graph): of the class.

const double& get_MSE (void): Gives MSE back.

const matrix& get_dif (void): Gives the difference between expected mean and real mean.

const double& get_konfidenzellipsoid (void): Gives confidence ellipsoid back.

const matrix& get_konfidenzintervall (void): Gives estimated confidence interval back.

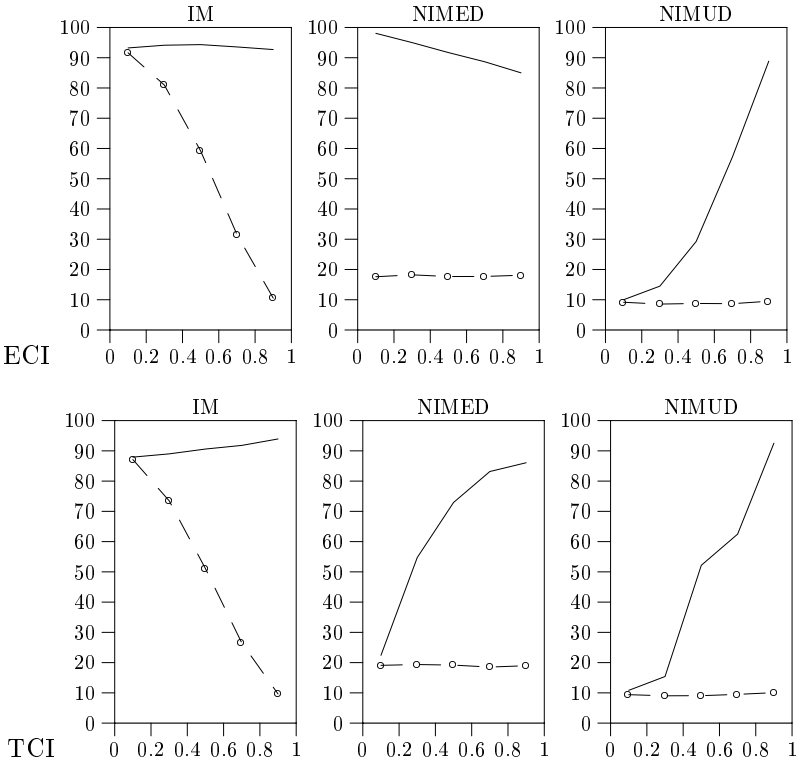

Figure 10: Efficiency of the incomplete time points depending on the correlation. Investigation unit one with 250 subjects and 30% of missing values. Line: PMM; dashed line: CC; circles: AC.

const matrix& get_theokonfidenzintervall (void): Gives theoretical confidence interval back.

void kerndichte (const matrix& cc_mp_matrix, const int& graph: Writes data for the kernel distribution estimator.

A.2 ergebnis

Writes TeX document.

public

ergebnis (){kopf()}: Initializes the class and fixes the head of the document.

~ergebnis (){fuss()}: Closes the document.

void text (const string& text): Ads some text.

void text (const double& zahl): Ads number into text.

void graphik (const string& figurnummer): Ads graphic.

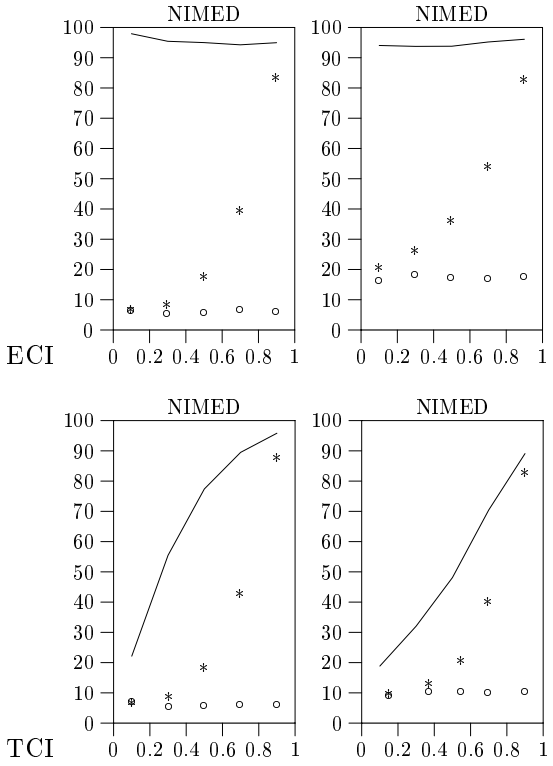

Figure 11: Efficiency of the incomplete time points depending on the correlation using the EM algorithm for NIMED situation. Investigation unit two compares simulations having 250 subjects and 30% of missing values with 50 subjects and 60% of missing values. Line: PMM; stars: PMM with EM algorithm; circles: AC.

void spalten (const Array2D<string>& inhalt,const int& zeilen): Writes text in column.

void absatz (void): Ads paragraph into text.

void section (const string& ueberschrift): Ads heading into text.

void subsection (const string& ueberschrift): Ads sub heading into text.

void paragraph (const string& ueberschrift): Ads little heading into text.

void formel (const string& text): Ads formula into text.

void tabelle1 (const int& anz_zeilen, const int& anz_spalten, const string& ueberschrift, const Array<string>& horizontal, const Array <string>& vertical, const realArray2D& zellen): Ads tabular containing numbers with string in first vertical column.

void tabelle1 (const int& anz_zeilen, const int& anz_spalten, const string& ueberschrift, const Array<string>& horizontal, const Array

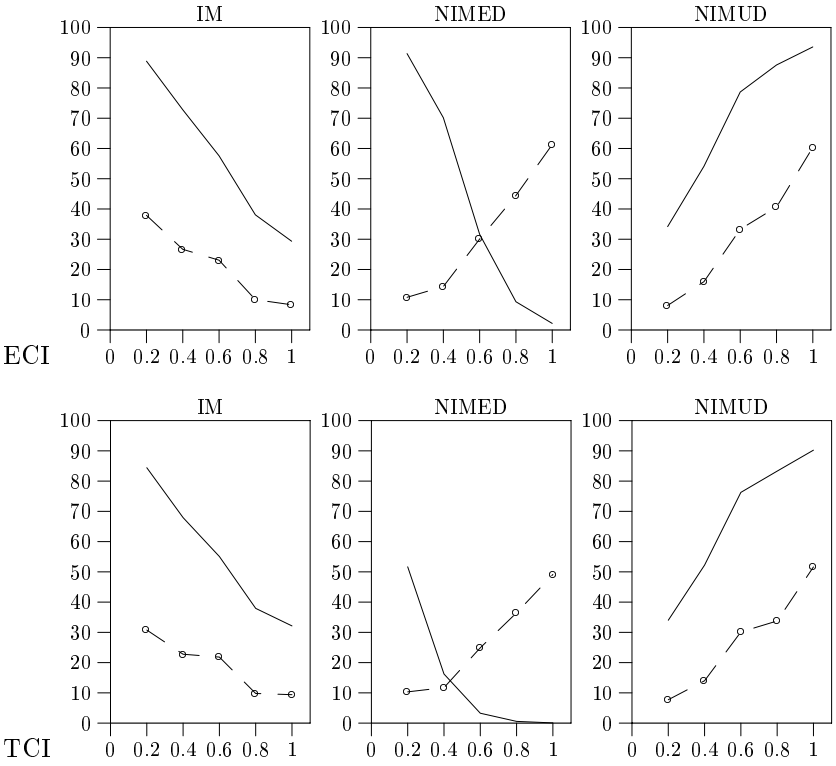

Figure 12: Efficiency of the incomplete time points depending on the extent of misspecification of the missing mechanism. Investigation unit three with 250 subjects and 30% of missing values. Line: PMM; dashed line: CC; circles: AC.

<string>& vertical, const stringArray2D& zellen): Ads tabular containing text.

void tabelle1 (const int& anz_zeilen, const int& anz_spalten, const string& ueberschrift, const Array<string>& horizontal, const matrix& vertical, const realArray2D& zellen): Ads tabular containing numbers with numbers in first vertical column.

A.3 completecase

Calculates the CC and AV estimator.

public

completecase (const matrixArray2D& gesamt_block, const intArray& para, const int& jack): Initializes the class.

completecase (void):

~completecase (void): Destructor.

const matrix& get_beta_cc (void): Gives estimator back.

const matrixArray& get_beta_ar (void): Gives CC estimator back for "sweepelements".

const matrix& get_variance_cc (void): Gives estimated CC variance back.

const matrix& get_varbeta_cc (void): Gives variance of the CC estimator back.

const matrix& get_beta_av (void): Gives AV estimator back.

const matrix& get_variance_av (void): Gives estimated AV variance back.

const matrix& get_varbeta_av (void): Gives variance of AV estimator back.

A.4 sweepelements

Calculates the estimator by the method of Little and Wang (1996)

public

sweepelements (const matrixArray2D& gesamt_block, const intArray& para): Initializes the class.

int chef1 (void): MAR case

int chef2 (void): MNAR case p1=p2

int chef3 (void): MNAR case p1>p2

const matrix& get_mittelwert (void): Gives estimator back.

const matrix& get_variance (void): Gives estimated variance back.

const matrix& get_roh (void): Gives correlation back.

const matrix& get_varbeta (void): Gives back variance of the estimator.

const double& get_px1 (void): Gives share of missing values back.

const matrix& get_ak_mis (void): Gives share of missing values within the group back.

~sweepelements (): Destructor.

A.5 aufbereitung

Edits the data for the calculations.

public

aufbereitung (): Initializes the class.

~aufbereitung (): Destructor.

const matrix& get_y (void): Gives back the y-matrix.

const matrix& get_x (void): Gives back the x-matrix.

const matrix& get_m_y (void): Gives back the missing indicator matrix.

const matrixArray& get_y_block (void): Gives back the y-matrix as block.

const matrixArray& get_x_block (void): Gives back the x-matrix as block.

const matrixArray& get_m_y_block (void): Gives back the missing indicator matrix as block.

const int& get_p1 (void): Gives back the number of missing time points.

void uebersicht (const matrixArray& gesamt, const intArray& para): Print an overview of the data, when the matrices are not divided by the time points.

void uebersicht (const matrixArray2D& gesamt, const intArray& para): Prints an overview of the data, when the matrices are divided by the time points.

matrixArray umstellen (const matrixArray& gesamt, const intArray& para): Sorts an optional data to be ready for estimate.

matrixArray2D blocken (const matrixArray& gesamt, const intArray& para): Separates the matrix by the time points.

matrixArray2D separieren (const matrixArray2D& gesamt, const intArray& para): Separates a matrix, which was already divided by the time points, by the missing indicator.

A.6 initial

Produces the random sample data.

public

initial (const realArray& para, const realArray& varianz, const matrix& mat_mu, const matrix& beta, const matrixArray& ar_gamma): Initializes the class.

~initial (void): Destructor.

const matrixArray2D get_gesamt_block (void): Gives back the random sample data.

const intArray get_parameter (void): Gives back the model parameters.

A.7 sweepoperator

Calculates the sweep-operator on row and column.

public

sweepoperator (void): Constructor.

~sweepoperator (void): Destructor.

const matrixArray2D sweep (const int& zeilen, const matrixArray2D& mat_array2d, const int& block): Takes block matrix and sweeps it on the requested column and row.

const matrix sweep (const matrix& mat, const int& block): Takes block matrix and sweeps it on the requested block.

const stringArray2D sweep (const int& zeilen, const stringArray2D& mat_ar_string, const int& block): Takes block matrix containing strings and sweeps it on the requested row and column.

References

Dempster, A. P. (1969). *Elements of Continuous Multivariate Analysis*, Addison-Wesley, Reading, Massachusetts.

Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*, Chapman and Hall, New York.

Fieger, A., Heumann, C., Kastner, C. and Watzka, K. (1997). Generische Bibliothek zur Linearen Algebra und zur Simulation in C++, *SFB386 – Discussion Paper 63*, Ludwig-Maximilians-Universität München.

Heumann, C., Fieger, A. and Kastner, C. (1998). C++ Utilities zur Implementierung statistischer Verfahren unter Berücksichtigung fehlender Werte, *SFB386 – Discussion paper 109*, Ludwig-Maximilians-Universität München.

Little, R. J. A. (1993). Pattern-mixture models for multivariate incomplete data, *Journal of the American Statistical Association* **88**: 125–133.

Little, R. J. A. (1994). A class of pattern-mixture models for normal incomplete data, *Biometrika* **81**: 471–483.

Little, R. J. A. and Rubin, D. B. (1987). *Statistical Analysis with Missing Data*, Wiley, New York.

Little, R. J. A. and Wang, Y.-X. (1996). Pattern-mixture models for multivariate incomplete data with covariates, *Biometrics* **52**: 98–111.

Rao, C. R. and Toutenburg, H. (1999). *Linear Models: Least Squares and Alternatives*, 2 edn, Springer, New York.

Stroustrup, B. (1997). *Die C++ Programmiersprache*, Addison-Wesley, Bonn.

SunOS (1995). *SunOS Reference Manual*, Sun Microsystems, Mountain View.