National Research Council Canada
Conseil national de recherches Canada

Institute for Information Technology
Institut de Technologie de l'information

ERB-1071

NRC·CNRC

Software Cost Estimation with Incomplete Data

Kevin Strike, Khaled El Emam,
and Nazim Madhavji
January 2000

Canada

NRC 43618

National Research Council Canada
Institute for Information Technology

Conseil national de recherches Canada
Institut de Technologie de l'information

Software Cost Estimation with Incomplete Data

Kevin Strike, Khaled El Emam,
and Nazim Madhavji
January 2000

Software Cost Estimation with Incomplete Data

Kevin Strike
School of Computer Science
McGill University
3480 University Street
McConnell Engineering Building
Montreal, Quebec
Canada H3A 2A7
strk@cs.mcgill.ca

Khaled El Emam
National Research Council, Canada
Institute for Information Technology
Building M-50, Montreal Road
Ottawa, Ontario
Canada K1A OR6
khaled.el-emam@iit.nrc.ca

Nazim Madhavji
School of Computer Science
McGill University
3480 University Street
McConnell Engineering Building
Montreal, Quebec
Canada H3A 2A7
madhavji@cs.mcgill.ca

Abstract

The construction of software cost estimation models remains an active topic of research. The basic premise of cost modelling is that a historical database of software project cost data can be used to develop a quantitative model to predict the cost of future projects. One of the difficulties faced by workers in this area is that many of these historical databases contain substantial amounts of missing data. Thus far, the common practice has been to ignore observations with missing data. In principle, such a practice can lead to gross biases, and may be detrimental to the accuracy of cost estimation models. In this paper we describe an extensive simulation where we evaluate different techniques for dealing with missing data in the context of software cost modelling. Three techniques are evaluated: listwise deletion, mean imputation and eight different types of hot-deck imputation. Our results indicate that all the missing data techniques perform well, with small biases and high precision. This suggests that the simplest technique, listwise deletion, is a reasonable choice. However, this will not necessarily provide the best performance. Consistent best performance (minimal bias and highest precision) can be obtained by using hot-deck imputation with Euclidean distance and a z-score standardisation.

1 Introduction

There exists a vast literature on the construction of software cost estimation models, for example [60][17][1][2][12][25][42][28][49][82][88][79][87]. The basic premise is that one can develop accurate quantitative models that predict development effort using historical project data. The predictors typically constitute a measure of size, whether measured in terms of LOC or a functional size measure, and a number of *productivity factors* that are collected through a questionnaire, such as questions on required reliability, documentation match to life cycle needs and analyst capability [85].

Knowing the estimated cost of a particular software project early in the development cycle is a valuable asset. Management can use cost estimates to approve or reject a project proposal or to manage the development process more effectively. For example, additional developers may need to be hired for the complete project or for areas that will require a large amount of effort. Furthermore, accurate cost estimates would allow organizations to make more realistic bids on external contracts. Cost estimation models have not been limited to prediction of total project cost. For instance, some recent work constructed a model to estimate the effort to perform a software process assessment [44], and to estimate the effort required to become ISO 9001 certified [71][72], both of which are relevant to contemporary software organizations.

A common practical problem with constructing cost estimation models is that the historical software engineering data sets frequently contain substantial numbers of missing values [14][15][35]. Such missingness would impact the productivity factors in historical data sets most severely since they are the variables collected through a questionnaire. While one should strive to minimise missing values, in practice their existence is usually unavoidable. Missing values are not unique to software cost estimation, but is a problem that concerns empirical scientists in other disciplines [48][38][55].

The most common factors that lead to missing data include individuals not responding to all questions in a questionnaire, either because they run out of time, they do not understand the questions or they do not have sufficient knowledge to answer the questions and opt not to respond, or individuals may not wish to divulge certain information that is perceived to be harmful or embarrassing to them. Furthermore, missing values increase as more variables are included in a data set [74]. It is common for cost estimation data sets to have a multitude of productivity factors.

There are many techniques that have been developed to deal with missing data in the construction of quantitative models (henceforth missing data techniques or MDTs) [56]. The simplest technique is to ignore observations that have missing values (this is called *listwise deletion*). In fact, this is the default approach in most statistical packages [56]. This, however, can result in discarding large proportions of a data set and hence in a loss of information that was costly to collect. For example, Kim and Curry [48] note that with only 2% of the values missing at random in each of 10 variables, one would lose 18.3% of the observations on average using listwise deletion. Furthermore, with 5 variables having 10% of their values missing at random, 41% of the observation would be lost with listwise deletion, on average. Furthermore, listwise deletion may bias correlations downwards. If a predictor variable has many missing high values, for example, then this would restrict its variance, resulting in a deflation of its correlation with a completely observed criterion variable. The same applies if the missing values were on the criterion variable. Measures of central tendency may be biased upwards or downwards depending on where in the distribution the missing data appear. For example, the mean may be biased downward if the data are missing from the upper end of the distribution.

Another set of techniques impute the missing values. A common approach is to use mean imputation. This involves filling in the missing values on a variable with the mean of observations that are not missing. However, mean imputation attenuates variance estimates. For example, if there are 30 observations of which 5 have missing values, then we could substitute five means. This would increase the number of observations without affecting the deviations from the overall mean, hence reducing the variance [57] [59]. There are alternative forms of imputation that are based on estimates of the missing values using other variables from the subset of the data that have no missing values.

In the context of cost estimation, researchers rarely mention how they dealt with missing values.[1] When they do, their solution tends to be to ignore observations with missing values, i.e. listwise deletion. For example, in the Walston and Felix study [88], different analyses rely on a different number of observations from the historical database, indicating that for some of the variables there were missing values. In one recent cost estimation study of European space and military projects, the authors removed observations that had missing values, resulting in some instances to the loss of approximately 38% of the observations [17]. A comparison study of different cost estimation modelling techniques noted that for approximately 46% of the observations there were missing values [19]. The authors then excluded observations with missing values for the different types of comparisons performed. Another study used a regression model to predict the effort required to perform a software process assessment based on the emerging ISO/IEC 15504 international standard [44]. In this study 34% of the total number of observations were excluded from the analysis due to missing values.

To date, there is no evidence that such a simple practice is the best one, or if it is detrimental to the accuracy of cost estimation models. It is plausible that certain types of imputation techniques would save the large proportions of discarded data and result in models with much improved prediction accuracy. It would be of practical utility then to have substantiated guidelines on how to deal with missing values in a manner that would minimise harm to model accuracy. Our study takes a step in that direction.

In this paper we present a detailed simulation study of different techniques for dealing with missing values when building cost estimation models: listwise deletion, (unconditional) mean imputation, and eight different types of hot-deck imputation. We also simulate three different types of missingness mechanisms:

[1] This is also a typical problem in other disciplines. For example, in one study [75], articles from the Journal of Applied Psychology and Personnel Psychology were randomly chosen and examined to see what methods were used to handle missing data. It was found that many studies do not state whether or not observations contained missing data. It follows that in these cases the technique for dealing with missing data is not mentioned either. One reason for this could be that journals only accept those studies that are deemed strong and may hesitate to publish studies that report high levels of missing data. Evidence suggested that 1/2 to 2/3 of the studies ignored observations with missing values.

missing completely at random, where missingness depends on the size of the project, and where missingness depends on the value of the variable with missing values; two types of missing data patterns: univariate (random) and monotone; and missingness on one productivity factor up to all productivity factors in a model. Our evaluative criteria focus on the accuracy of prediction, and consist of the common measures: Absolute Relative Error and Pred25 [25]. The summary measures are the bias and variation of the accuracy measures (precision). We focus on ordinary least squares regression as the modelling technique since this is one of the most common modelling techniques used in practice [35], e.g. [69][88][23][22][17][60]. Furthermore, there has been recent compelling evidence that ordinary least squares regression is as good as or better than many competing modelling techniques in terms of prediction accuracy [41][18][19].

Briefly, our results indicate that all MDTs have a good performance in terms of bias and precision under the different contexts simulated. This suggests that the simplest technique, listwise deletion, is a reasonable choice. However, listwise deletion will not provide the best performance amongst the different MDTs. Consistently better performance is obtained by using hot-deck imputation with Euclidean distance and a z-score standardisation, even for large percentages of missing data.

In the following section we present an overview of the missing data problem and the techniques that have been developed for dealing with it. Section 3 describes our research method in detail. The results of our simulation are described in Section 4 with a discussion of their implications and limitations. We conclude the paper in Section 5 with a summary and directions for future research.

2 Background

In this section we define some terminology and provide an overview of MDTs and their general applicability.

2.1 Terminology

An important distinction when discussing MDTs is the mechanism that caused the data to be missing [56]. Consider a data set with two variables, X_1 and X_2. Let us assume that missing data occurs on variable X_1 only. To make the scenario concrete, let variable X_1 be analyst capability and variable X_2 be project size. If the probability of response to X_1 does not depend on either X_1 or X_2, then it is said that the missing data mechanism is Missing Completely At Random (MCAR). Thus, if the missingness of the analyst capability variable is independent of project size and analyst capability, the mechanism is MCAR. If the probability of response depends on X_2 but not on X_1, then we say that the missing data mechanism is Missing At Random (MAR). This would be exemplified by the situation whereby the missingness on the analyst capability variable is higher for small projects than for large projects. The third mechanism is if the probability of response depends on the value of X_1 itself. This would occur if respondents tend not to answer the question when the analyst capability is low. This is termed non-ignorable response.

In general, the suitability of MDTs will be influenced by the missing data mechanism and the percentage of observations with missing values. We outline some common MDTs below.

2.2 Common MDTs

There exist several strategies for dealing with missing data. It is generally accepted that if the data set contains a relatively small amount of missing data and if this data is missing randomly, then all MDTs will be equally suitable [29][51][48][3][73]. It should be noted that caution must be exercised when classifying data sets as having small amounts of missing data. For example, if the small amount of missing data is only found in a few variables and is distributed randomly among all observations, the total percentage of observations containing missing data may still be relatively large. For example, if only 5% of the observations on each of 5 variables have missing values, then approximately 23% of the observations will have a missing value. The choice of MDT becomes more important as the amount of missing data in the data set increases [73]. Another important factor in choosing a suitable MDT is the mechanism that leads to the missing values, whether MCAR, MAR or non-ignorable [56]. For example, if we have many missing values for large projects, an inappropriate MDT may distort the relationship with effort, potentially leading to less accurate predictions.

There are two general classes of MDTs that can be applied: deletion methods and imputation methods. These are described below.

2.2.1 Deletion Methods

Deletion methods ignore missing values. These procedures may result in the loss of a significant amount of data, but are widely used because of their simplicity [75][48].

2.2.1.1 Listwise Deletion

Analysis with this method makes use of only those observations that do not contain any missing values. This may result in many observations being deleted but may be desirable as a result of its simplicity [48]. This method is generally acceptable when there are small amounts of missing data and when the data is missing randomly.

2.2.1.2 Pairwise Deletion

In an attempt to reduce the considerable loss of information that may result from using listwise deletion, this method considers each variable separately. For each variable, all recorded values in each observation are considered and missing values are ignored. For example, if the objective is to find the mean of the X_1 variable, the mean is computed using all recorded values. In this case, observations with recorded values on X_1 will be considered, regardless of whether they are missing other variables. This technique will likely result in the sample size changing for each considered variable. Note that pairwise deletion becomes listwise deletion when all the variables are needed for a particular analysis, (e.g. multiple regression). This method will perform well, without bias, if the data is missing at random [56].

It seems intuitive that since pairwise deletion makes use of all observed data, it should outperform listwise deletion in cases where the missing data is MCAR and correlations are small [56]. This was found to be true in the Kim and Curry study [48] . In contrast, other studies have found that when correlations are large, listwise outperforms pairwise deletion [3]. The disadvantage of pairwise deletion is that it may generate an inconsistent covariance matrix in the case where multiple variables contain missing values. In contrast listwise deletion will always generate consistent covariance matrices [48].

In cases where the data set contains large amounts of missing data, or the mechanism leading to the missing values is non-random, Haitovsky proposed that imputation techniques might perform better than deletion techniques [36].

2.2.2 Imputation Methods

The basic idea of imputation methods is to replace missing values with estimates that are obtained based on reported values [78][30]. In cases where much effort has been expended in collecting data, the researcher will likely want to make the best possible use of all available data and prefer not to use a deletion technique [24]. Imputation methods are especially useful in situations where a complete data set is required for the analysis [57]. For example, in the case of multiple regression all observations must be complete. In these cases, substitution of missing values results in all observations of the data set being used to construct the regression model. It is important to note that no imputation method should *add* information to the data set. In the case of multivariate data, it makes sense that we might be able to obtain information about the missing variable from those observed variables. This forms the basis for imputation methods. The primary reason for using imputation procedures is to reduce the non-response bias that would result if all the observations that have missing values are deleted.

2.2.2.1 Mean Imputation

This method imputes each missing value with the mean of observed values. The advantage of using this method is that it is simple to implement and no observations are excluded, as would be the case with listwise deletion. The disadvantage is that the measured variance for that variable will be underestimated [76][56]. For example, if a question about personal income is less likely to be answered by those with low incomes, then imputing a large amount of incomes equal to the mean income of reported values decreases the variance.

2.2.2.2 Hot-deck Imputation

Hot-deck imputation involves filling in missing data by taking values from other observations in the same data set. The choice of which value to take depends on the observation containing the missing value. The latter property is what distinguishes hot-deck imputation from mean imputation.

In addition to reducing non-response bias and generating a complete data set, hot-deck imputation preserves the distribution of the sample population. Unlike mean imputation, which distorts the distribution by repeating the mean value for all the missing observations, hot-deck imputation attempts to preserve the sample distribution by substituting different observed values for each missing observation.

Hot-deck imputation selects an observation (donor) that best matches the observation containing the missing value (client). The donor then provides the value to be imputed into the client observation. For example, a study may be able to gather a certain variable for all observations, such as geographic location. In this case a categorical hot-deck is created in which all observations are separated into categories according to one or more classification variables, in this case geographic location. Observations containing missing values are imputed with values obtained from complete observations within each category. It is assumed that the distribution of the observed values is the same as that of the missing values. This places great importance on the selection of the classification variables.

In some studies there may not be any categorical data and the variables in which to assess ‘similarity’ may be numerical. In this case, a donor is selected that is the most similar to the client. Similarity is measured by using a distance function that calculates the distance between the client and prospective donors. The hot-deck is the set of all complete observations. For each client, a donor (or set of donors) is chosen from the hot-deck that contains the smallest distance to the client. This distance can be based on one or more variables. The selection of which variables to use in the distance function is ideally those variables that are highly correlated to the variable being imputed. In the case where a set of donors has been obtained, the value to impute may be taken from the best donor, random donor, or an average over all donors. The purpose of selecting a set of donors is to reduce the likelihood of an extreme value being imputed one or more times [30][78].

Colledge et al. concluded that hot-deck imputation appears to be a good technique for dealing with missing data, but suggested that further analysis be done before widespread use [24].

2.2.2.3 Cold Deck Imputation

This method is similar to hot-deck imputation except that the selection of a donor comes from the results of a previous survey [56].

2.2.2.4 Regression Imputation

Regression imputation involves replacing each missing value with a predicted value based on a regression model. First, a regression model is built using the complete observations. For each incomplete observation, each missing value is replaced by the predicted value found by replacing the observed values for that observation in the regression model [56].

2.2.2.5 Multiple Imputation Methods

Modelling techniques that impute one value for each missing value underestimate standard error. This is the case because imputing one value assumes no uncertainty. Multiple imputation remedies this situation by imputing more than one value, taken from a predicted distribution of values [58]. The set of values to impute may be taken from the same or different models displaying uncertainty towards the value to impute or the model being used respectively. For each missing value an imputed value is selected from the set of values to impute, each creating a complete data set. Each data set is analysed individually and final conclusions are obtained by merging those of the individual data sets. This technique introduces variability due to imputation, contrary to the single imputation techniques.

2.3 Summary

To our knowledge, there have been no previous studies of MDTs within software engineering. Therefore it is not possible to determine which MDTs are suitable and under what conditions for software engineering studies.

In our study, we focus on simple methods that would allow researchers to easily implement the result. Three types of MDTs are evaluated: listwise deletion, mean imputation, and hot-deck imputation. We chose listwise deletion because it is common practice in software cost estimation studies, and therefore we wished to determine its performance. Furthermore, it has been noted that in general empirical enterprises, listwise deletion and mean imputation are the most popular MDTs [73]. Hot-deck imputation is of interest since it has been adopted in some highly visible surveys, such as the British Census [7][30], the U.S. Bureau of the Census Current Population Survey, the Canadian Census of Construction [30], and the National Medical Care Utilization and Expenditure Survey [55]. Furthermore, some authors contend that the hot-deck is the most common MDT for complex surveys [29].

3 Research Method

3.1 Objective of Study

The objective of this study is to compare different MDTs for dealing with the problem of missing values in historical data sets when building software cost estimation models. Since for cost estimation models the most important performance measure is their prediction accuracy, we evaluate how this accuracy is affected by using the different missing data techniques. By identifying the most appropriate technique, future researchers would have substantiated guidance as to how to deal with missing values.

3.2 Data Source

The data set used in this study is called the *Experience Database.* The Experience Database began with the co-operation of 16 companies in Finland. Each company must purchase the Experience tool and contribute the annual maintenance fee. This entitles them to the tool that incorporates the database, new versions of software and updated data sets. Each company can add their own data to the tool and are encouraged through incentives to donate their data to the shared database. For each project donated, the company is given a reduction in the annual maintenance fee. Since all companies collect the data using the same tool and the value of each variable is well defined, integrity and comparability of the data is maintained. In addition, companies that provide data are subsequently contacted in order to verify their submission.

The primary advantage of this data base for our study is that it does not contain missing values. The fact that this relatively large data set does not contain missing values is due to the careful manner in which the data was collected and extensive follow up. This allows us to simulate various missing data patterns and mechnisms, as will be explained below.

The data set is composed of 206 software projects from 26 different companies. The projects are mainly business applications in banking, wholesale/retail, insurance, public administration and manufacturing. This wide range of projects allows for generalisable conclusions that can be applied to other projects in the business application domain. Six of the companies provided data from more than 10 projects. The system size is measured in unweighted and unadjusted Function Points [2]. For each project, we had the total effort in person-months and values on fifteen productivity factors. The variables considered in our analysis are presented in Table 1.

Variable	**Description**	**Values / Range / Unit**
Effort	Total project effort	Person hours (ph)
FP	System size measured in function points	Unadjusted Unweighted Function Points
PF01-PF15	15 Productivity Factors: Customer Participation, Development Environment, Staff Availability, Level and use of standards, Level and use of Methods, Level and use of Tools, Logical Complexity of the Software, Requirements Volatility, Quality Requirements, Efficiency Requirements, Installation Requirements, Analysis Skills of Staff, Application Experience of Staff, Tool Skills of Staff, Project and Team Skills of Staff	1 – 5 (very small – very large)

Table 1: Variables Used in our simulation from the Experience Database.

Table 2 summarises the descriptive statistics for system size (FP) and project effort (person-hours: ph). The breakdown of projects per organisation type is 38% banking, 27% insurance, 19% manufacturing, 9% wholesale, and 7% public administration. Figure 1 and Figure 2 illustrate the proportions of projects for different application types and target platforms.

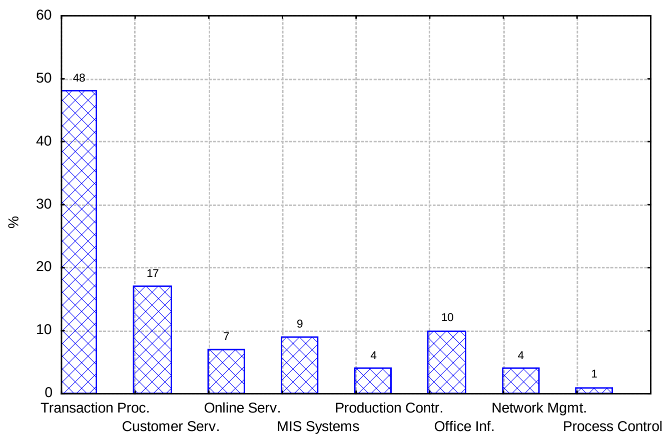

Figure 1: Distribution of projects by application type.

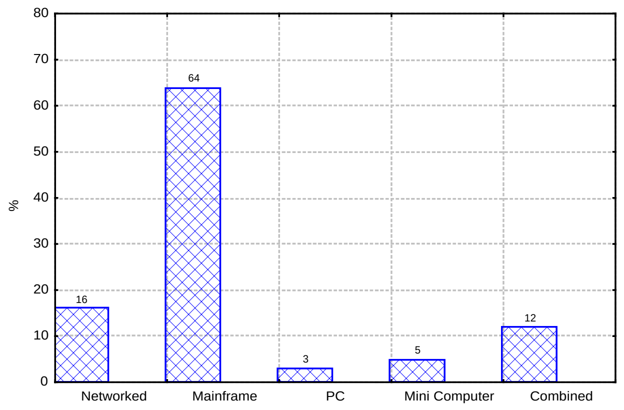

Figure 2: Distribution of projects by target platform.

	Whole data set	
	Size (FP)	Effort (ph)
Min	10	250
Mean	118.7	6644.9
Max	558	63694
st. dev	101.6	8684.3
Obs	206	206

Table 2: Descriptive statistics for system size and effort.

3.3 Validation Approach

If a cost estimation model is developed using a particular data set and the accuracy of this model is evaluated using the same data set, the accuracy obtained will be optimistic. The calculated error will be low and will not represent the performance of the model on another separate data set. We therefore randomly split the total data set into equal sized parts (with 103 observations each), to obtain a training data set and a test data set. We used the training data set for building the estimation model, and the test data set for evaluating the performance of the model.

3.4 Regression Model

We used multivariate least squares regression analysis to fit the data to a specified model that predicts effort [37]. The selected model specification is exponential, because linear models revealed marked heteroscedasticity, violating one assumption for applying regression analysis.

One can make other substantive arguments for selecting this functional form. In software engineering there has been a debate over whether economies of scale do indeed exist, and if so, what is the appropriate functional form for modelling such economies. The concept of economies of scale states that average productivity increases as the system size increases. This has been attributed, for example, to software development tools whereby the initial tool institutionalisation investment may preclude their use on small projects [12]. Furthermore, there may be fixed overhead costs, such as project management, that do not increase directly with system size, hence affording the larger projects economies of scale. On the other hand, it has been noted that some overhead activities, such as documentation, grow at a faster rate than project size [43], contributing to diseconomies of scale. Furthermore, within a single organisation, it is plausible that as systems grow larger, then larger teams will be employed. Larger teams introduce inefficiencies due to an increase in communication paths [21], the potential for personality conflicts [12], and more complex system interfaces [25]. A series of studies on the existence of (dis)economies of scales provided inconsistent results [9][10][50]. In another effort to determine whether such (dis)economies of scale exist, Hu [39] compared a simple linear model with a quadratic, log-linear, and translog models, and used objective statistical procedures to make that determination. He also investigated what the appropriate functional form should be. He concluded that the quadratic form is the most appropriate. Subsequently his study was criticised on methodological grounds [70][17]. Another study that compared functional forms that addressed some of these shortcomings concluded that the log-linear form, which we use, is the most plausible one [17].

The general form of the regression model is as follows:

$$\log(Effort) = \beta_0 + \beta_1 \log(FP) + \sum_{i>0} \beta_{i+1} \log(T_i) \qquad \textbf{Eqn. 1}$$

where the T_i values are the productivity factors.

It is known that many of the productivity factors are strongly correlated with each other [17][50][84]. Although some cost estimation models have been developed that contain many productivity factors, [88][5][12][61] found that for a given environment, only a small number of significant productivity factors are needed in order to develop an accurate effort estimation model. This conclusion is supported by [50][66][8].

Therefore, we first reduced the number of variables down from 15 using only the training data set. Two approaches were considered. The first was a mixed stepwise process to select variables having a significant influence on effort. The second is the leaps and bounds algorithm [32], which is an efficient algorithm for performing regressions for all possible combinations of the 15 productivity factors (size was always included). The model with the largest adjusted R^2 value was selected. The leaps and bounds algorithm is advantageous in the sense that it performs an exhaustive search, and therefore we used that as the basis for variable selection. We retained the variables identified by the all variable subsets search that were statistically significant at a one-tailed alpha level of 0.05 (we used a one-tailed test because we have a priori expectations about the direction of the relationship).

The final model is summarised in Table 3. We refer to this model as the *baseline model* since it has been developed with the complete training data set (i.e., no missing values). We used the condition number of Belsley et al. [11] as an indicator of collinearity. It was lower than the threshold of 30, and hence we can be confident that there are no multicollinearity problems in this model.

Variable (log)	**Regression Coefficient**	**Standard Error**	**t value**	**1-tailed p-value**
Intercept	2.96	0.512	5.78	<0.0001
System size (function points)	0.85	0.09	9.34	<0.0001
Logical complexity of software	0.68	0.27	2.53	0.013
Requirements volatility	0.904	0.26	3.5	0.0007
Application knowledge of staff	-0.392	0.201	-1.95	0.054

Table 3: Baseline model parameters. The adjusted R^2 is 0.61, and the F test of all parameters equal to zero had a p value of <0.0001.

3.5 Scale Type Assumptions

According to some authors, one of the assumptions of the OLS regression model is that all the variables should be measured at least on an interval scale [13]. This assumption is based on the mapping originally developed by Stevens [83] between scale types and "permissible" statistical procedures. In our context, this raises two questions. First, to what extent can the violation of this assumption have an impact on our results ? Second, what are the levels of our measurement scales ?

Given the proscriptive nature of Stevens' mapping, the permissible statistics for scales that do not reach an interval level are distribution-free (or nonparametric) methods (as opposed to parametric methods, of which OLS regression is one) [80]. Such a broad proscription is viewed by Nunnally as being "narrow" and would exclude much useful research [68]. Furthermore, studies that investigated the effect of data transformations on the conclusions drawn from parametric methods (e.g., F ratios and t tests) found little evidence supporting the proscriptive viewpoint [53][52][6]. Suffice it to say that the issue of the validity of the above proscription is, at best, debatable. As noted by many authors, including Stevens himself, the basic point is that of pragmatism: useful research can still be conducted even if, strictly speaking, the proscriptions are violated [83][13][33][86]. Therefore, consideration should be given to the appropriateness of making the interval scale assumptions in each individual case rather than adopting a blanked proscription. A detailed discussion of this point and the supporting literature is given in [16].

Our productivity factors utilised a single item each. In practice, single item measures are treated as if they are interval in many instances. For example, in the construction and empirical evaluation of the User Information Satisfaction instrument, inter-item correlations and principal components analysis are commonly performed [40]. It is also useful to note a study by Spector [81] which indicated that whether scales used have equal or unequal intervals does not actually make a practical difference. In particular, the mean of responses from using scales of the two types do not exhibit significant differences, and that the test-retest reliabilities (i.e., consistency of questionnaire responses when administered twice over a period of time) of both types of scales are both high and very similar. He contends, however, that scales with unequal intervals are more difficult to use, but that respondents conceptually adjust for this.

3.6 Evaluative Measures

In order to evaluate the impact of MDTs, we define two different evaluative measures:[2] magnitude of relative error (MRE) and prediction at level l (PRED(l)) [25]. These are calculated from the model developed using the training data set and evaluated on the test data set.

The MRE is defined as:

$$\mathrm{MRE}_i = \frac{\left|\mathrm{Actual\ Effort}_i - \mathrm{Predicted\ Effort}_i\right|}{\mathrm{Actual\ Effort}_i} \qquad \textbf{Eqn. 2}$$

The MRE value is calculated for each observation i whose effort is predicted. The aggregation of MRE over all predicted observations, N, was achieved by taking the median of the MRE (MdMRE) over N observations.[3]

A complementary criterion that is also used is the prediction at level l, $PRED(l) = \frac{k}{N}$, where k is the number of observations where MRE is less than or equal to l. For our study we set l to 25%. The Pred25 provides the percentage of observations whose effort estimates were within 25% error.

The ordinary least squares modelling technique minimises the sum of squared error criterion, whereas we are evaluating the resulting model in terms of the absolute relative error. Miyazaki et al. [64] have noted this discrepancy in the software cost estimation literature, and it was alluded to by Shepperd and Schofield [79]. Ideally, one would wish to optimise on the same criterion as one is evaluating on. Not doing so would be expected to lead to sub-optimal cost estimation models. However, thus far, it is not established to what extent such a sub-optimisation exists in the software cost estimation area.

Relative error (or absolute relative error) has a clear intuitive appeal as an evaluative criterion (i.e., it is easy to interpret and convey its meaning to project managers who are the ultimate users of cost estimation models). For example, it is regularly used to compare the accuracy of cost estimation models and modelling techniques [47][41][18][19][79][82]. One possible solution to the above incongruence is to use alternative modelling techniques that optimise on the relative error. To our knowledge, this has rarely been performed thus far in the software engineering literature, with notable exceptions being [63][79]. In fact, ordinary least squares regression is one of the most common modelling techniques used in practice [35], e.g. [18][19][41][66][69][88][23][22][17][60]. Furthermore, there has been recent compelling evidence that ordinary least squares regression is as accurate as or more accurate than many competing modelling techniques [41][18][19]. Also, using least squares regression with a (absolute) relative error evaluative criterion is also common practice, [41][18][19][69][60][25][66][79]. Therefore, should other modelling techniques emerge, enjoy wide acceptance, *and* are consistently demonstrated to be superior, then it would be prudent to perform an evaluative study of MDTs on these as well.

[2] A third potential evaluative measure is concerned with consistency of estimation. This is defined as the correlation between the estimated and the actual effort, and has been used in a number of previous studies [66][2][47]. The logic of using this measure is that the existence of consistency, even if it is consistency in under or overestimation, project managers would be able to easily adjust for that using say a constant multiplier and the estimation model would still be of value. However, it was shown in a recent report [20] that adjusting consistent underestimates with a constant multiplier will increase accuracy but dramatically increase variability, and will reduce the acccuracy of overestimates. In our simulation we, therefore, focus only on accuracy of prediction.

[3] An implicit assumption in using MRE as a measure of predictive accuracy is that the error is proportional to the size of the project. For example, a 15 person-month overestimate for a 15 person-month project is more serious than for a 100 person-month project. On the other hand, Miyazaki et al. have criticised the MRE measure because it penalises overestimates more than underestimates [63], and propose a new measure to alleviate this. However, Shepperd and Schofield [79] note that the proposed Myiazaki measure of accuracy is effectively two distinct measures that should not be combined. We therefore utilise the commonly used MRE as no alternatives have enjoyed acceptance within the software engineering community.

3.7 Simulation Approach

There are three general approaches that one can use to study the effect of dealing with missing values, two being simulations. The first possibility is to use an actual data set that had missing data, that were subsequently obtained through follow-up activities. This would allow the researcher to compare the performance of models using the data set with missing values after an MDT is applied, and the complete data set after follow-up. This is the approach used in the study by Cox and Folsom [26]. However, it is rare in practice to have such a data set.

The second is a Monte Carlo simulation. Under this approach, one constructs artificial data sets whose variables have known distributions and known inter-correlations. Then one creates missing values in these artificial data sets following various missingness schemes that one wants to study. Subsequently, different MDTs are applied and their performance evaluated in comparison to the known characteristics of the artificial data.

An example of such a Monte Carlo simulation is [76]. The analysis considered only cases where observations were partially incomplete, not those in which there was a complete non-response. Techniques considered were listwise deletion, pairwise deletion, mean imputation, regression imputation, and hot-deck imputation. The complete data set used for the analysis was generated from a population correlation matrix. Various statistics were generated based on the complete data set to be compared with those generated after inducing missing values and application of the various MDTs. Missing values were induced in 10, 20 and 30% of the observations randomly. The effectiveness was based on calculation of root mean squared error and absolute error.

A potential disadvantage of the Monte Carlo approach is that one cannot be certain that the population characteristics that are being simulated are congruent with real data sets. Therefore, one would not know the extent to which the conclusions can be generalised to actual practical situations. This approach can be mitigated by basing the population parameters on values obtained from previous studies, such as in the Monte Carlo simulation performed by Kim and Curry [48].

An alternative approach is to use an actual data set relevant to the problem. This data set would have to be complete (i.e., no missing data). One would then create missing values that follow a known pattern and mechanism. Subsequently, different MDTs are applied and their performance evaluated in comparison to the results that would be obtained had there been no missing data.

An example of this kind of approach is the study of [51]. This study analysed the performance of five MDTs for dealing with data missing nonrandomly namely listwise deletion, pairwise deletion, mean imputation, simple regression imputation, and multiple imputation. Performance of each technique was based on parameter estimates of a two predictor regression model. Bootstrap samples were taken from actual field data and missing values were assigned to one variable, based on the value of that variable.

The approach we have followed in our study is in the final category. We used the Experience Database as our complete data set. The overall simulation approach is summarised in Figure 3.

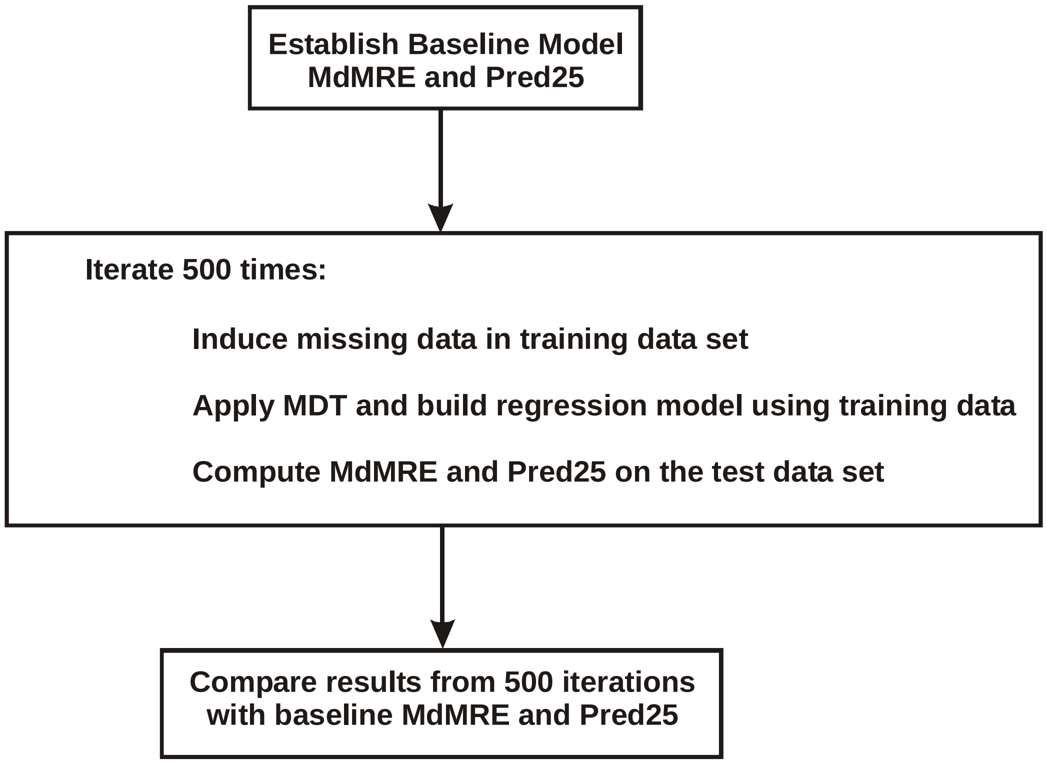
Establish Baseline Model MdMRE and Pred25
Iterate 500 times:
Induce missing data in training data set
Apply MDT and build regression model using training data
Compute MdMRE and Pred25 on the test data set
Compare results from 500 iterations with baseline MdMRE and Pred25

Figure 3: Summary of simulation approach.

The baseline model was already described in Section 3.4. Using the baseline model, we predicted the effort on the test data set and found the MdMRE to be 47% and the Pred25 to be 24%. Note that the calculation of these measures is based on the original units, not the log transformed ones. These serve as our baseline MdMRE and baseline Pred25 and will be referred to as $MdMRE_{baseline}$ and $Pred25_{baseline}$ respectively.

We describe the remaining steps of our simulation approach below.

3.8 Inducing Missing Data

We consider four parameters in simulating the missing data:

- The percentage of observations with missing data.
- The number of variables that have missing data (we only consider missing values on the productivity factors)
- The missing data mechanism
- The pattern of missing data

Five different percentages of missing values were simulated (5,10,15,25,40). It is generally accepted that data sets with more than 40% missing data are not useful for detailed analysis.

Since our final model has three productivity factors, we consider missing values on one up to all three variables for all permutations.

Three missing data mechanisms were evaluated: missing completely at random (MCAR), missing at random (MAR), and non-ignorable missingness.

The implementation of the MCAR mechanism to induce missing data was straightforward. Missing values were induced for the desired variable completely at random.

For MAR, we simulated the situation where missing values depend on the size of the project. The implementation of the MAR mechanism to induce missing data involves first ordering the observations according to project size. Missing values are induced with biases for both small and large project sizes. Once the data set is ordered, it is split up into quintiles. Each quintile will receive different percentages of missing values. The total percent of missing values is divided by 10 to get a value for k. The separate quintiles will be induced with 4k, 3k, 2k, k, and 0k missing values. For example, if the total amount of missing data to be created is 10%, then each quintile will be induced with 4, 3, 2, 1, and 0% missing values. It is important to note that within each quintile, missing values are induced randomly.

For non-ignorable missingness, we simulated missing values that depend on the particular variable in question. Implementation was identical to MAR except that the observations were ordered by the variable to be induced with missing data. Quintiles were formed and missing values were created as described for MAR. Missing values are induced with biases for both low and high values for each variable.

Two patterns of missing data were simulated. The first is univariate missing data, whereby the values are missing on each variable separately according to one of the mechanisms described above. Each independent variable was induced with missing values according to the pattern shown in Figure 4 (left panel).

The second is monotone missing data, whereby the variables can be ordered in terms of their extent of missingness. This means that all observations with missing data for X_2 also have missing data for X_1, but the reverse is not always true [58].

The implementation of monotone missing data is illustrated with two variables. It easily generalises to the case of more than two variables. The monotone pattern of missingness is shown in Figure 4 (right panel). The first variable is induced the same way as for the univariate case. Next, half of the observations that contain missing values at the first variable will have their second variable induced with a missing value. For the MCAR mechanism of missing data, half of the total number of observations containing missing data will have missing data on two variables. For the MAR mechanism, half of the observations *for each quintile* will contain missing data for both variables. It follows that although the total amount of missing values being induced is greater in the monotone case as compared to the univariate case, the total number of observations that contain *at least* one missing value will be the same.

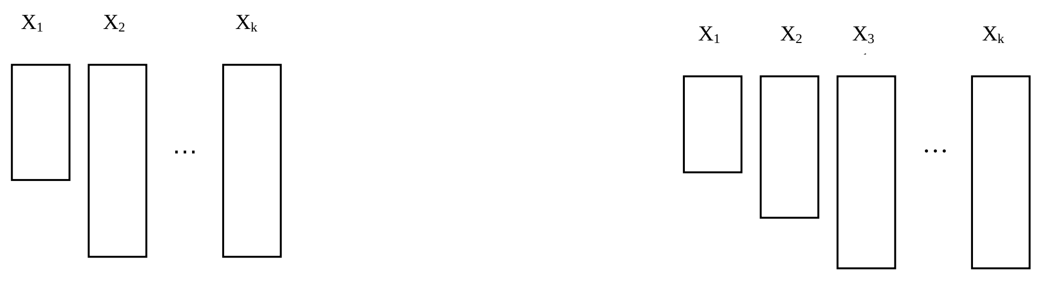

Figure 4 : Univariate and monotone patterns of missing data**.**

In total, we considered all combinations of percentage of missing data, number of variables and their permutations, missingness mechanism, missingness pattern, and MDT. Each one of these is a study point. For each study point and MDT combination we performed the simulation 500 times.[4] For each of the 500 iterations, we built a new ordinary least squares regression model. Subsequently, we performed a prediction on the test set and computed the evaluative measures.

[4] We randomly selected some of the study points and performed the simulations with 1000 iterations. Our conclusions were not affected, and in fact even the summary values obtained were very similar.

3.9 Summary Measures

For each run of the simulation we computed the MdMRE and Pred25. From these numbers we have to produce concise summaries of the results. We consider two types of summaries: *bias* and *precision*. Bias tells us how different the results are from those that would be obtained had there been no missing data (i.e., the baseline). Precision informs us about the dispersion or variability in this accuracy difference.

From the 500 simulations we computed the $(MdMRE_i - MdMRE_{baseline})$ difference for each simulation i. This was then plotted on a box-and-whisker plot to illustrate bias (median) and precision (inter-quartile range). The same was done for the Pred25 as $(Pred25_i - Pred25_{baseline})$ for each simulation i.

3.10 MDTs Evaluated

The implementation of the MDTs that we studied is described below.

3.10.1 Listwise Deletion

For listwise deletion, observations containing missing data are ignored. The regression model is built using only observations that contain no missing values.

3.10.2 Mean Imputation

After the database has been induced with missing values, each missing value is replaced by the mean calculated for all observed cases for that variable. For the univariate case, the mean is calculated for the variable that contains missing values and this value is imputed for all observations that contain a missing value. For the multivariate case, multiple means are calculated, one for each variable that contains missing values. The same value will be imputed for all missing observations.

Once all the values have been imputed the regression model is generated. Unlike listwise deletion, no observations are lost and the regression model is based on the complete data set containing imputed values.

3.10.3 Hot-Deck Imputation

Once the missing observations have been created in the database, each missing value is imputed with a donor value picked from the one of the observations without missing data. We now present some notation to help explain the different types of hot-deck imputation that we implemented.

We divide the data set into those observations with missing values, the missing set, and those observations without missing values, the complete set. Let x_i be the vector of all variables measured on the i^{th} observation in the missing set, and x_{ij} be the value of the j^{th} variable measured on the observation. Further, let c_k be the vector of all variables measured on the k^{th} observation in the complete set, and let c_{kj} be the value of the j^{th} variable measured on that observation.

We constructed a numeric hot-deck function. In setting up a hot-deck function, different hot-deck parameters can potentially have a non-negligible impact on its performance:

- In calculating a distance between an observation in the missing set and the complete set, variables containing values that are much larger than those in other variables will dominate the distance (e.g., size has a much larger range than the productivity factors). Standardisation will prevent those variables from having a larger influence (in effect, treating all variables as being equally important). Which standardisation technique should be used ?
- There are multiple possible distance measures that can be employed. Which one should be used?

Below we describe these parameters and the particular values that we evaluated, including justifications for the selections made.

First, we consider three different standardisation approaches: z-score, mean absolute, and Z_5, in addition to the case of no standardisation.

A simulation study in the area of cluster analysis found one type of standardisation scheme to be superior in recovering the underlying cluster structure under different conditions including error free data, and data with noise and with outliers [62]. However, the parameters of that simulation are not necessarily reflective of software cost data, and therefore this standardisation scheme, referred to as Z_5, is evaluated here:

$$Z_5 = \frac{c_{kj} - \min(c_k)}{\max(c_k) - \min(c_k)} \qquad \textbf{Eqn. 3}$$

As will be noted, this has a non-robust denominator in that it will be easily affected by even a single outlier. A more traditional standardization scheme that makes the unit of the variables the sample standard deviation is the z-score:

$$Z - score = \frac{c_{kj} - \frac{1}{n}\sum_k c_{kj}}{\sqrt{\frac{1}{n-1}\sum_k \left(c_{kj} - \frac{1}{n}\sum_k c_{kj} \right)^2}} \qquad \textbf{Eqn. 4}$$

A more robust approach for standardisation is to use the *mean absolute deviation* [89] instead of the standard deviation in the denominator. This is robust because the deviations are not squared, therefore atypical points do not exaggerate it [46]. Robustness is desirable because software measurements typically have a few extreme values that may exert strong influence on the analysis results (e.g., see [67]).

$$Mean\ Absolute = \frac{c_{kj} - \frac{1}{n}\sum_k c_{kj}}{\frac{1}{n}\sum_k \left| c_{kj} - \frac{1}{n}\sum_k c_{kj} \right|} \qquad \textbf{Eqn. 5}$$

For each observation that contains a missing value a distance is determined between it and all observations in the complete set. A multitude of different distance functions can reasonably be used in a hot-deck function. We limit ourselves to the common distance functions in other disciplines. We also limit ourselves to the context where the predictor variables are continuous, since we treat them this way in the regression model.

Kaufman and Rousseeaw [46] define the Minkowski distance as follows:

$$d_{ik} = \left(\sum_j \left| c_{kj} - x_{ij} \right|^q \right)^{\frac{1}{q}} \qquad \textbf{Eqn. 6}$$

The most commonly used distance functions for continuous variables are the Euclidean and Manhattan distances.

The Euclidean distance between a component i in the missing data set and a component k in the complete set is given by (i.e., $q = 2$):

$$Euclidean_{ik} = \sqrt{\sum_j \left(c_{kj} - x_{ij}\right)^2}$$ **Eqn. 7**

The Manhattan distance is defined by (i.e., $q = 1$):

$$Manhattan_{ik} = \sum_j \left|c_{kj} - x_{ij}\right|$$ **Eqn. 8**

A priori, there is no compelling reason to prefer one distance function over another, and therefore, it is prudent to evaluate them empirically.

Using both functions, a value is determined for the distance between the observation containing a missing value and all observations in the hot-deck. The distance function takes as input the size variable and all productivity factors except the one that is missing. The missing value is imputed with the variable from the observation in the hot-deck that has the smallest distance from the observation containing the missing value.

Once all missing values have been imputed with values obtained from the hot-deck, the data set is complete and the regression model is generated.

3.11 Summary

In this section we have described the overall simulation approach. The defined study points include all techniques for inducing missing values combined with the MDTs studied. Such a comprehensive simulation should provide us with a reasonable picture of the strengths and weaknesses of each MDT under different missing data scenarios. Furthermore, the fact that the simulation is based on an actual data set should give us confidence in the applicability of the results within the same application domain.

4 Results

In this section we present the results for each MDT in turn.

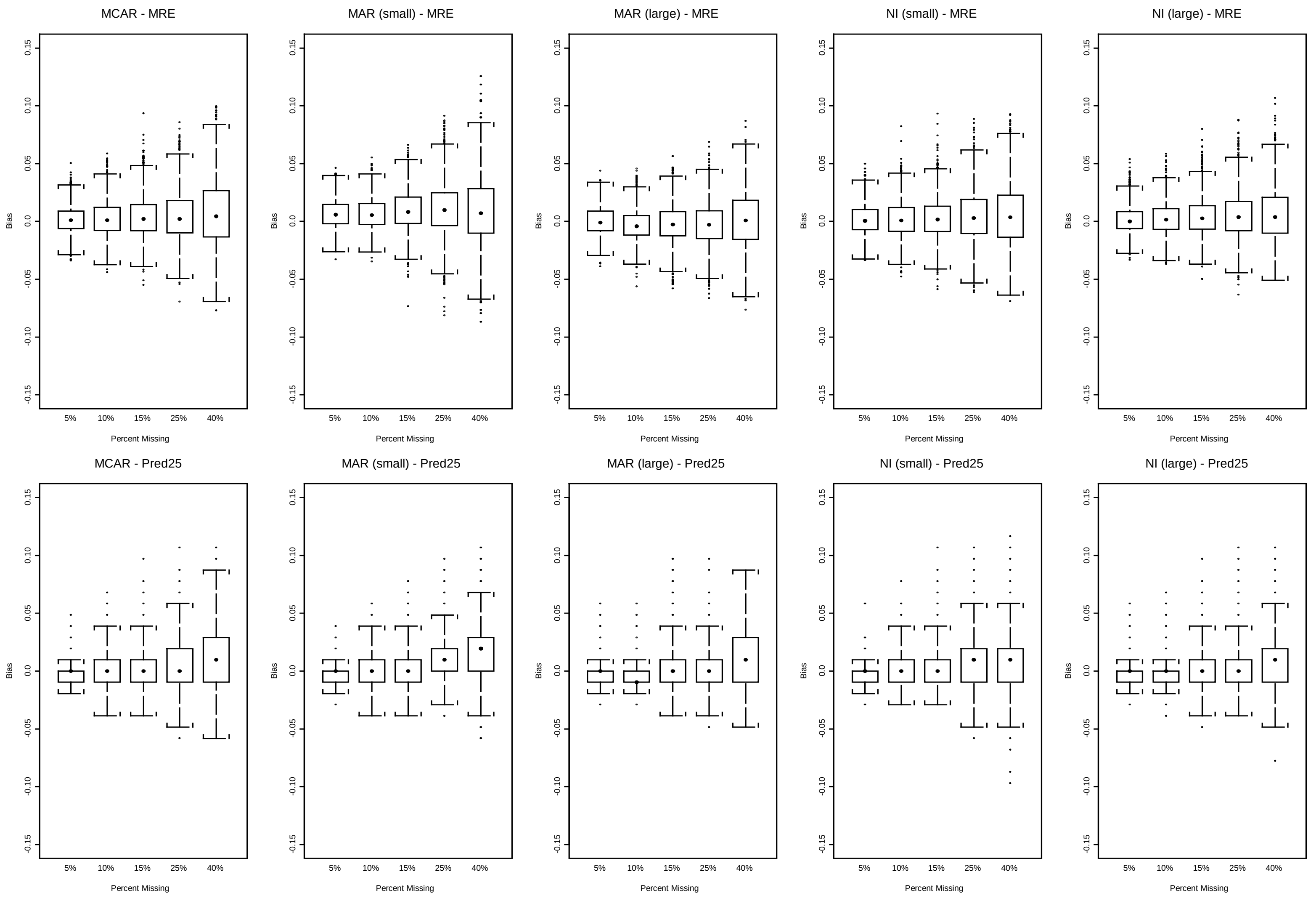

Figure 5: Listwise deletion for univariate missing data pooled across all productivity factors.

4.1 Listwise Deletion

Figure 5 shows the box-and-whisker plots for listwise deletion under the three missingness mechanisms. Here, the plots are only for the univariate pattern. The results for the monotone pattern are not presented because with listwise deletion they are, by definition, exactly the same as in the univariate case.

A number of conclusions can be drawn from these plots:

- The bias is systematically small, being always less than 0.03 difference from the complete data set results. It can be argued that such a small bias is almost negligible in practice given that typical parametric cost estimation models have relative errors that can be quite large.
- The precision of the accuracy measures decreases as the extent of missing data increases (i.e., there is more variability).
- For non-ignorable missingness, the bias increases slightly as the extent of missing data increases.
- For the MAR and non-ignorable mechanisms, there is no striking difference between biases for large vs. small.
- For low percentages of missing data, up to 15%, listwise deletion performs remarkably well, with a negligible bias in its performance. At higher extents of missing data, the bias in terms of MdMRE and Pred25 is still rather small in absolute terms. Since these results hold even for missingness on all productivity factors, they indicate that listwise deletion is a reasonable approach.

In general, we can conclude that listwise deletion has a remarkably small bias, even at high percentages of missing data. The disadvantage of listwise is that the variability in the accuracy tends to be large at higher percentages of missing data. As will be seen later, this variability is smaller for other MDTs.

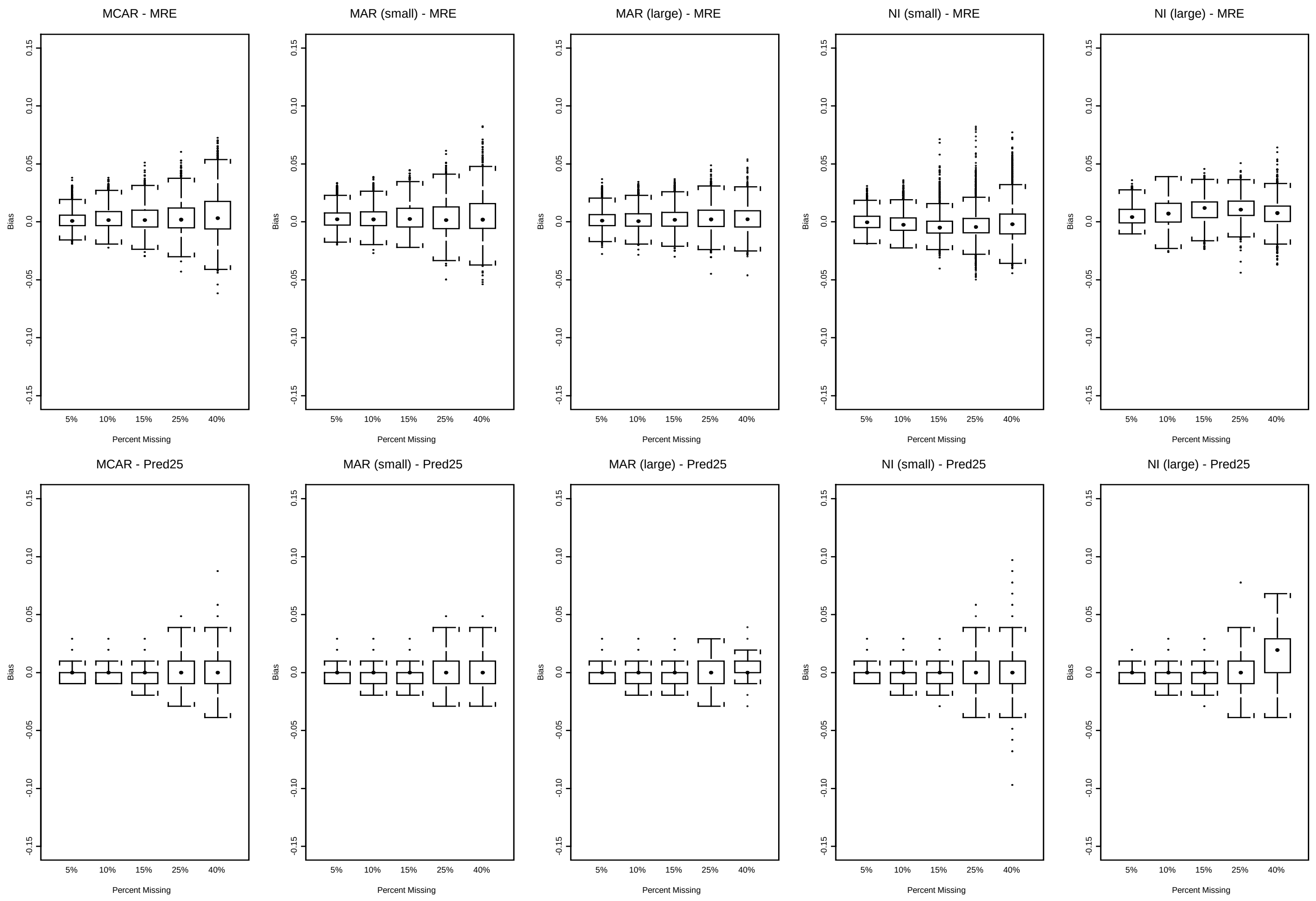

Figure 6: Mean Imputation for univariate missing data pooled across all productivity factors.

4.2 Mean Imputation

4.2.1 Univariate Missingness

Figure 6 shows the results for mean imputation for univariate missing data. A number of conclusions can be drawn from these plots:

- In general, the precision is higher (less variability) than the comparable situation with listwise deletion.
- For MCAR and MAR, bias tends to be almost zero. There is more fluctuation in bias for non-ignorable missingness, especially at higher extents of missing values.
- Mean imputation seems to be a good MDT given that its use for the univariate case results in little bias compared to the complete case results.

4.2.2 Monotone Pattern

Figure 7 shows the results for the monotone pattern of missingness on two variables (pooled across all permutations). We can observe that:

- Variability tends to be larger then for univariate missingness. Otherwise the results are similar to the univariate case for MCAR and MAR mechanisms.
- The largest bias is obtained for 25% and 40% missing data with the non-ignorable missingness mechanism.
- In comparison to listwise deletion, the mean imputation MDT tends to exhibit less variability in its accuracy.

Figure 8 shows the results when we consider the monotone pattern for all three productivity factors. The results are very similar to Figure 7 with missing values on only two productivity factors, and hence so are the conclusions.

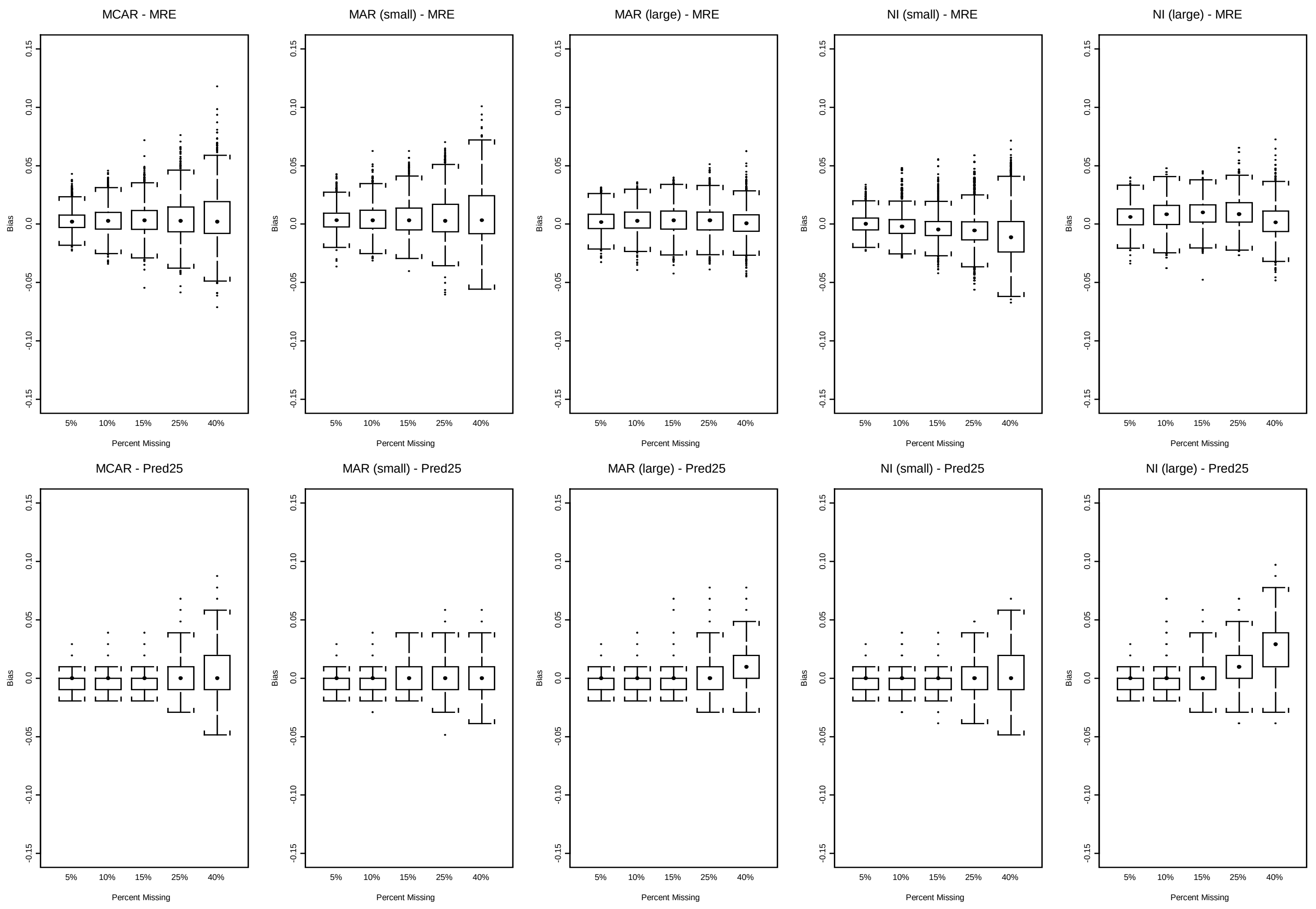

Figure 7: Mean Imputation MDT for monotone missing data pooled across permutations of two productivity factors.

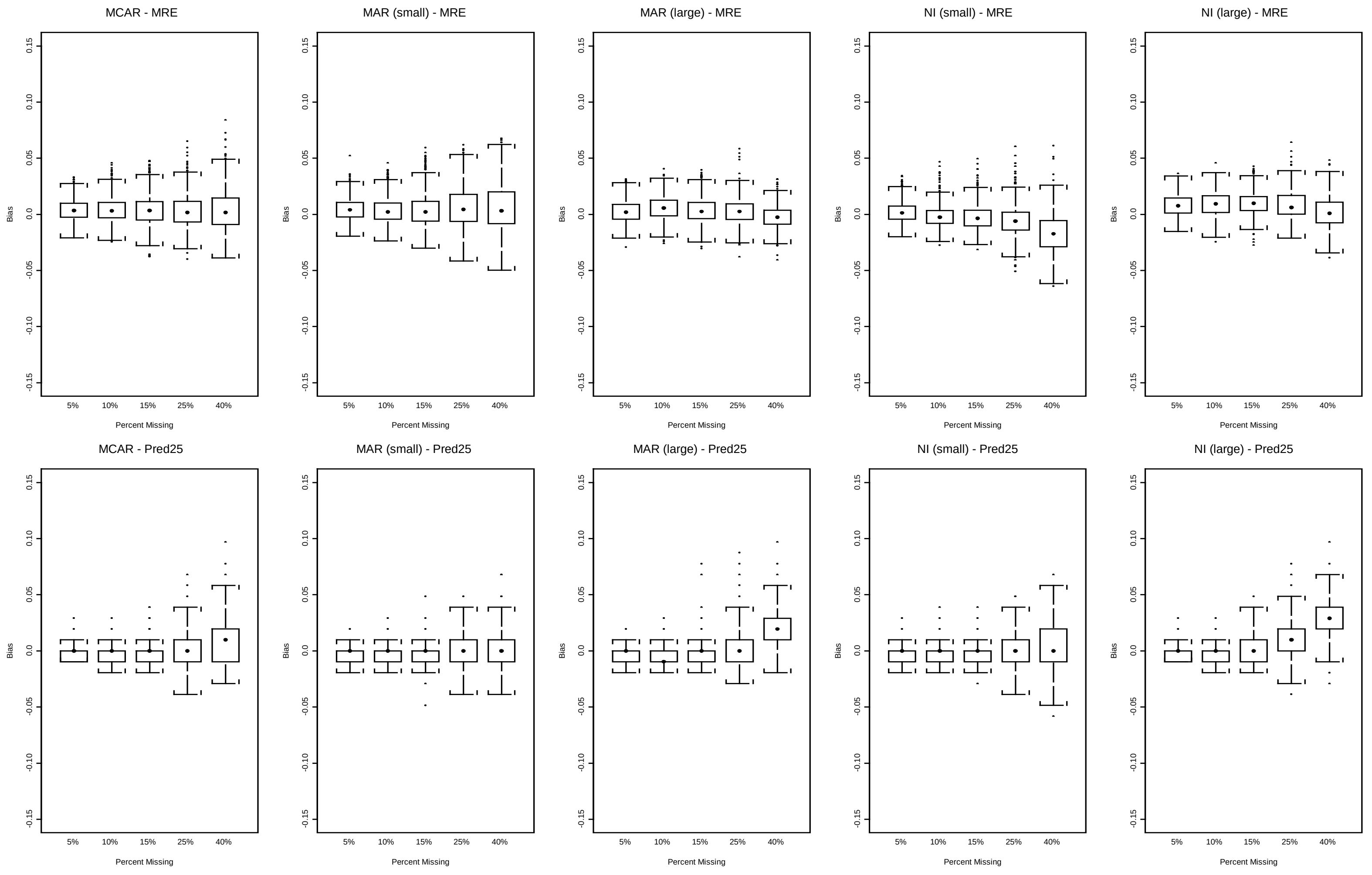

Figure 8: Mean Imputation MDT for monotone missing data pooled across permutations of three productivity factors.

4.3 Hot-Deck

For hot-deck imputation we did not find any non-negligible difference amongst the alternative implementations of hot-deck (i.e., different distance measures and standardization schemes). Therefore we present the results for the hot-deck variant with Euclidean distance and z-score standardization. The conclusions we draw, however, are applicable to the other variants as well.

4.3.1 Univariate Missingness

Figure 9 shows the results for the univariate case using hot-deck imputation. The following observations can be made:

- Compared to mean imputation, hot-deck imputation has a consistently smaller bias, but the variability in the accuracy measures tends to be slightly larger. The variability is still smaller than for listwise deletion.
- The bias is almost zero except that it increases slightly for non-ignorable missingness with high percentages of missing values.

4.3.2 Monotone Pattern

As for mean imputation, the results are the same for the cases with missing values on two variables and on three variables. Therefore we only present the results for the two variable case. Figure 10 shows the box-and-whisker plots for the cases with missing values on two variables, pooled across all permutations. We can observe that:

- The variation in the accuracy measures is slightly larger than for the univariate case.
- Compared to mean imputation, the bias tends to be smaller but the variation in the accuracy measure is slightly larger. In particular, with non-ignorable missingness and high percentages of missing values, hot-deck tends to have a lower bias.

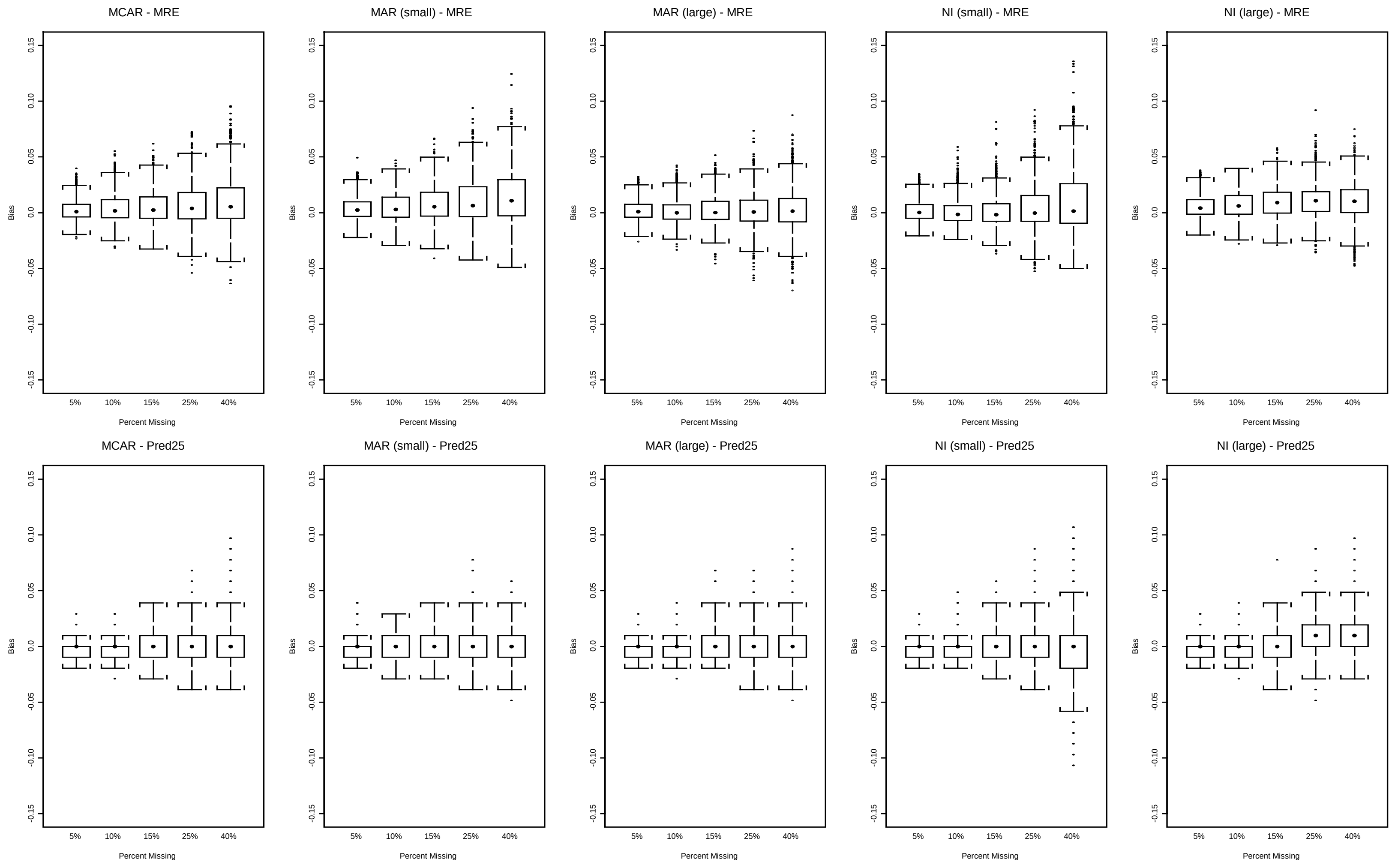

Figure 9: Hot-Deck MDT for univariate missing data pooled across productivity factors.

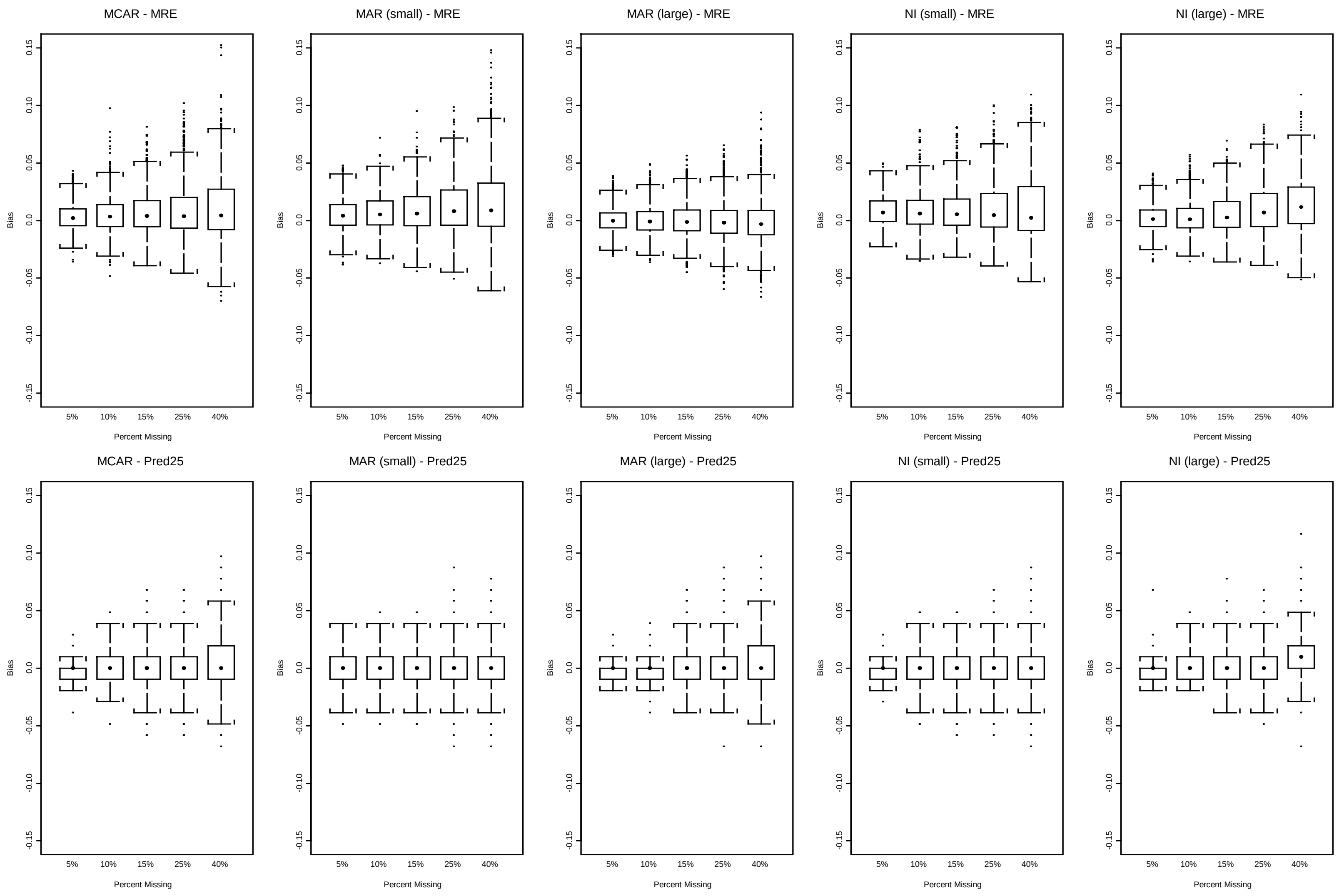

Figure 10: Hot-Deck MDT for monotone missing data pooled across permutations of two productivity factors.

4.4 Summary

Below we provide an overall summary of our findings:

- In general, all MDTs tend to perform well in absolute terms, with their bias being consistently of a low percentage. This has some implications on previous software engineering research. Given that the most common practice thus far has been to use listwise deletion, our results indicate that the detrimental consequences of this would be rather minor. Listwise deletion has the appeal of being a simple approach. However, researchers could do better, as we discuss below.
- For all mechanisms and patterns, we found that the performance of the MDTs (in terms of bias and precision) deteriorates as the percentage of missing data increases. This is more pronounced for non-ignorable missingness. Although this deterioration was only slight in the case of hot-deck imputation.
- The precision of listwise deletion tends to be worse than the other MDTs, especially as the percentage of missing data increases.
- For hot-deck imputation, the differences amongst the various types of hot-deck parameters were not marked. Therefore, we suggest a traditional hot-deck is appropriate. A traditional hot-deck uses Euclidean distance and z-score standardisation. Euclidean distance is the most common distance measure, and thus its appeal. Standardisation is a reasonable thing to do given that otherwise size would systematically dominate any distance.
- In general, the MDTs tend to perform slightly worst with monotone patterns of missingness compared with the univariate pattern. [5]
- For monotone patterns of missingness, increasing the number of productivity factors with missing values to more than two does not have a marked impact on the performance of MDTs.[6]
- The MDTs tend to perform slightly worst under non-ignorable missingness compared with MCAR and MAR.
- Any hot-deck imputation technique will work well in absolute terms, with accuracy differences from a complete data set rarely, if ever, going above 0.03. Mean imputation performs slightly worst than hot-deck, but the degradation is minor. Listwise deletion still performs very well, but its performance is a slight degradation over mean imputation and hot-deck.

4.5 Discussion

The best way to handle missing data is to maximise response in the original sample. In some cases it may be desirable to resample if large amounts of missing data are in the original sample. But in the most likely scenario, in which there is missing data, we can make some practical recommendations. Furthermore, if ensuring complete data sets would be too costly, then our recommendations would result in models that would be almost as accurate had complete data sets been attained.

We found that in general, all MDTs perform rather well in terms of bias and precision. The differences between the MDTs under the various simulated conditions are small. This implies that common practices thus far of using listwise deletion would not have penalised the accuracy of cost estimation models too greatly.

One potential reason for this finding is that the impact of the productivity factors is rather small compared with the effect of size. This is similar to the conclusion drawn in a previous study, where Matson et al. showed that there was only a small improvement from adding more variables to the baseline model relating effort to project size [60]. This can be explained in a number of ways. It is plausible that the impact of the productivity factors is marginal after size has been accounted for, at least from a prediction perspective. It is also plausible that the productivity factors are not measured reliably and therefore, even

[5] Except for listwise deletion, where the results are exactly the same as for the univariate missingness case.

[6] Except for listwise deletion, where the results are exactly the same as for the univariate missingness case.

though the concepts they measure influence effort, this is not apparent from the data. If this is the case, then predictions will not be grossly affected by missing values.

Amongst the MDTs examined, we found that imputation techniques perform better than listwise deletion. To obtain the best performance from cost estimation models, it would therefore be prudent to apply an imputation technique. From our simulation results, we can recommend using hot-deck imputation for cost estimation modelling. This tends to have low bias, even in the case where there is non-ignorable missingness and a high percentage of missing values. Furthermore, the variability in the accuracy measures tend to be the smallest for hot-deck imputation.

It can be argued that in practice the assumption of an MCAR mechanism is unrealistic in software engineering [65]. Therefore, the missingness mechanism is expected to be either MAR or non-ignorable in practice. The robust performance of hot-deck imputation, especially with both of these mechanisms, makes it more attractive.

4.6 Comparison to Previous Work

The literature suggests that under MCAR and a low percentage of missing data, listwise deletion tends to perform well. For instance, Gilley and Leone [34] state “If the item nonresponse is nonsystematic and represents a small percentage of a reasonably large sample, then excluding the nonrespondents from the sample would have a negligible effect on any statistical results”, and Frane [31] notes “If the number of subjects with missing data is small, if data are missing at random, and if interest lies in statements regarding a population rather than individuals, the elimination of subjects with missing data is likely to lead to a satisfactory analysis.” Furthermore, one Monte Carlo simulation provides corroborative results in that at low percentages of missing values (10% or less), they found that listwise deletion does not give markedly distorted estimates of regression coefficients and R^2 when data is missing at random [73].

Another study [51] analysed the performance of five MDTs for dealing with data missing nonrandomly, namely listwise deletion, pairwise deletion, mean imputation, simple regression imputation, and multiple imputation. Performance of each technique was based on parameter estimates of a two predictor regression model. The results showed that in general the three imputation techniques examined, mean imputation, simple regression imputation and multiple imputation, did not perform well with nonrandomly missing data. In contrast, the listwise and pairwise deletion techniques performed well when the level of missing data was less than 30%. This is consistent with our results, in that listwise deletion produces reasonable results under a variety of different missingness schemes and relatively high percentages of missing data.

For low percentages of missing values under MCAR, Roth [75], based on a literature review, recommended using hot-deck imputation. A simulation study by Lee and Chiu [54] led to the conclusion that listwise deletion is a preferred MDT to mean imputation when computing the polychoric correlation. We are, however, assuming a regression model.

Roth and Switzer [76] performed a Monte Carlo simulation comparing different MDTs. Techniques considered were listwise deletion, pairwise deletion, mean imputation, regression imputation, and hot-deck imputation under an MCAR mechanism. The results showed that pairwise deletion had the least amount of dispersion and average error around true scores for bivariate correlations. In the case of multiple regression, the performance of pairwise deletion was similar to that of listwise deletion. Furthermore, the authors recommended against the use of mean imputation. For regression parameters, the authors noted that there are slight differences between the various MDTs studied. This is consistent with our conclusions, even for MAR and non-ignorable missingness mechanisms. In our study, however, we did find that mean imputation performed better than listwise deletion for the MCAR setting.

Another simulation found that for data missing at random, mean imputation tended to perform slightly better than listwise deletion [73]. This is consistent with our results. For high percentages of missing values under MCAR and MAR, Roth recommended hot-deck imputation [75]. This is the same conclusion that we drew based on our simulations for the software cost estimation problem. Similarly, he recommends hot-deck for low percentages of data that is not missing at random. At high percentages of data that are not missing at random, Roth suggested maximum likelihood estimation [75]. However, we did not evaluate this as part of our study. We did find that hot-deck imputation worked best under these conditions for the software cost estimation problem.

One study by Kaiser found that the performance of hot-deck methods decreases with an increase in the proportion of records containing missing values, the increase in the number of missing values in each record, or the combination of both [45]. This is somewhat consistent with our results, except that we did not find that increases in the number of variables that have missing values beyond two had a substantial impact on the accuracy of hot-deck imputation. Further evaluations of the hot-deck procedure were performed [29][4][26][27].

Cox and Folsom [26] performed a simulation using an actual data set, where they were evaluating estimates of univariate means and proportions under different MDTs. They found that for discrete questionnaire items, hot-deck tended to reduce bias compared to listwise deletion, with better performance for items that had the greatest missingness. However, they also noted that the variance of the estimates was inflated by hot-deck. This was also noted by Ernst [27]. For continuous items, they concluded that the performance of hot-deck was inferior: it did not reduce the bias in the estimates and when it did, the variance was large. This is consistent with our results, in that the productivity factors can be considered as discrete variables by their definition, and we found that hot-deck tended to outperform listwise deletion. Another simulation was performed by Ford [29] where he compared six different MDTs, including four variants of hot-deck imputation. His criteria were estimates of the mean and its variance. Ford found that there was no difference in the performance amongst the different MDTs, although they were better than listwise deletion. This finding is also consistent with our simulation results. Both of the above studies, however, focus on parameters that are different from those that we evaluated (i.e., the accuracy of predictions from a regression model).

Perhaps the most important message from the above review is that it is critical to evaluate quantitative techniques using software engineering data sets. There is a mixed correspondence between our results and those obtained from previous studies. This is not surprising since the previous work used different evaluation criteria, and the distributions of the variables they simulated were different. Furthermore, techniques such as hot-deck, which tend to work best when there are strong correlations between the covariates and the variable with missing values, will perform differently depending on the correlation structure amongst the variables. This correlation structure may not be transportable from other disciplines.

4.7 Limitations

In this paper we reported on a simulation study to evaluate some MDTs for dealing with missing data during the construction of software cost estimation models. While we have attempted to design the simulation to be as comprehensive as possible, covering many plausible different missingness scenarios, we cannot claim that this study is the last word on missingness for cost estimation modelling. Below we identify the limitations of the study, which in turn suggest avenues for further research. However, it should be noted that most of these limitations are a consequence of the scope that we have defined for our study. We already simulated a large number of study points. Further simulations can be performed to address situations that we have not considered.

Our simulation study was performed using one data set in the business application domain. Although the data set was large, giving us confidence in the conclusions we draw, other simulations would have to be performed on different data sets to confirm our findings. We have attempted to be precise in describing the details of our simulation as an aid for future replications.

We only simulated missingness on the productivity factors. Our rationale has been that if data on the size variable and the effort variable are missing, when building a cost estimation model, then one has little to go on. This may be a reflection of a serious data collection problem. Whereas missingness on questionnaire responses, while not desirable, is more likely to happen.

Our evaluation criteria concerned only prediction accuracy. We did not consider the parameter estimates in the regression models or the effect of MDTs on statistical tests of significance. The objective of our study was to focus on the utility of such models for making effort predictions for new projects. It is plausible that a simulation that used other evaluative criteria may come up with different recommendations.

The simulations we performed selected three productivity factors out of an initial fifteen. We did not utilise the dropped twelve productivity factors during the imputation procedures we used. Since the excluded factors are likely strongly correlated with some of the included factors (and this is one reason for droping them in our analysis due to the increased risk of collinearity in the regression model otherwise), they may have served as useful covariates in a hot-deck and have resulted in even better performance for this imputation MDT. However, in practice, researchers would be more likely to exclude variables with many missing values from their analysis, and therefore excluded variables would not necessarily serve as good covariates.

We only considered up to 40% of the observations having missing values, which encompasses most practical situations. It is possible that in some studies there would be more than 40% missing values. However, it is contended that this would be an indicator of a severe data collection problem. For instance, Raymond and Roberts [73] state "With 40 percent of the data missing, one would have to question seriously the appropriateness of conducting any anaylsis." Furthermore, the wisdom of doing any analysis with 30-40% missing data has been questioned by Roth [75] and Ford [29].

Finally, our simulation focused on ordinary least squares regression as the modelling technique. This is the most popular technique for building cost estimation models, and has been found in recent studies to perform at least as good as alternatives. It is plausible that other modelling techniques would require different MDTs.

5 Conclusions

The objective of this paper was to perform a comprehensive simulation to evaluate techniques for dealing with missing data in cost estimation models. We simulated a total of 825 study points, varying: the number of variables with missing data (from 1 to three), the percentage of missing data (from 5% to 40%), the missing data mechanism (MCAR, MAR, and non-ignorable missingness), the missing data pattern (univariate and monotone), and comparing 10 different techniques for dealing with missing data. The performance of this study was greatly facilitated by the existence of a large and complete software project data set. This is not very common, as many cost data sets do have missing values.

Our results provide practical and substantiated guidelines for researchers and practitioners constructing cost estimation models when their data sets have missing values. Although we show that deleting observations with missing values has a small penalty, better performance would be obtained from applying imputation techniques.

We encourage the replication of this simulation study on alternative data sets to confirm, or otherwise, our conclusions. If indeed further replications confirm our findings, then this has important practical significance to those building cost estimation models. Furthermore, future work ought to examine alternative imputation techniques, as these may provide even better performance than the ones we studied.

6 Acknowledgements

We wish to thank Risto Nevalainen and Pekka Forselius of Software Technology Transfer Finland (STTF) for defining, collecting, and providing us with the data set used in this study. We also thank the anonymous reviewers for their comments, which resulted in substantial improvements to this work.

7 References

[1] A. Albrecht: "Measuring Application Development Productivity". In *SHARE/GUIDE: Proceedings of the IBM Applications Development Symposium*, pages 83-92, 1979.

[2] A. Albrecht and J. Gaffney: “Software Function, Source Lines of Code, and Development Effort Prediction”. In *IEEE Transactions on Software Engineering*, 9(6):639-648, 1983.

[3] S. Azen and M. Van Guilder: “Conclusions Regarding Algorithms for Handling Incomplete Data”. In *Proceedings of the Statistical Computing Section*, American Statistical Association, 53-56, 1981.

[4] J. Bailar III, B. Bailar: “Comparison of Two Procedures For Imputing Missing Survey Values”. In *Proceedings of the Section on Survey Research Methods*, American Statistical Association. 462-467, 1978.

[5] J. Bailey and V. Basili: “A Meta-Model For Software Development Resource Expenditures”. In *Proceedings 5th International Conference on Software Engineering*, Lund, Sweden, July 1983, 107-116, 1981.

[6] B. Baker, C. Hardyck, and L. Petrinovich: “Weak Measurements vs. Strong Statistics: An Empirical Critique of S. S. Stevens’ Proscriptions on Statistics”. In *Educational and Psychological Measurement,* 26:291-309, 1966.

[7] K. Baker, P. Harris, and J. O’Brien: “Data Fusion: An Appraisal and Experimental Evaluation”. In *Journal of the Market Research Society*, 31:153-212, 1989.

[8] R. Banker, S. Datar, and C. Kemerer: “A Model to Evaluate Variables Impacting the Productivity of Software Maintenance Projects”. In *Management Science*, 37(1), 1-18, 1991.

[9] R. Banker and C. Kemerer: “Scale Economies in New Software Development”. In *IEEE Transactions on Software Engineering*, 15(10):1199-1205, 1989.

[10] R. Banker and C. Kemerer: “The Evidence on Economies of Scale in Software Development”. In *Information and Software Technology*, 36(5):275-282, 1994.

[11] D. Belsley, E. Kuh, and R. Welsch: *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. John Wiley and Sons, 1980.

[12] B. Boehm: *Software Engineering Economics*. Prentice-Hall, 1981.

[13] G. Bohrnstedt and T. Carter: “Robustness in Regression Analysis”. In: Costner, H. (ed). Chapter 5, *Sociological Methodology*. Jossey-Bass, 1971.

[14] L. Briand, V. Basili, and W. Thomas: “A Pattern Recognition Approach for Software Engineering Data Analysis”. In *IEEE Transactions on Software Engineering*, 18(11):931-942, 1992.

[15] L. Briand: “Quantitative Empirical Modeling for Managing Software Development: Constraints, Needs and Solutions”. In H.D. Rombach, V. Basili, and R. Selby (eds.): *Experimental Software Engineering Issues: Critical Assessment and Future Directions*. Springer-Verlag, 1993.

[16] L. Briand, K. El Emam, and S. Morasca: “On the Application of Measurement Theory in Software Engineering”. In *Empirical Software Engineering: An International Journal*, 1(1), 61-88, 1996.

[17] L. Briand, K. El Emam, and I. Wieczorek: “Explaining the Cost of European Space and Military Projects”. In *Proceedings of the International Conference on Software Engineering*, pages 303-312, 1999.

[18] L. Briand, K. El Emam, D. Surmann, I. Wieczorek, and K. Maxwell: “An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques”. In *Proceedings of the International Conference on Software Engineering*, pages 313-322, 1999.

[19] L. Briand, T. Langley, and I. Wieczorek: “A Replicated Assessment and Comparison of Common Software Cost Modeling Techniques”. To appear in *Proceedings of the International Conference on Software Engineering*, 2000.

[20] L. Briand, K. El Emam, B. Freimut, and O. Laitenberger: “A Comprehensive Evaluation of Capture-Recapture Models for Estimating Software Defect Content”. To appear in *IEEE Transactions on Software Engineering*, 2000.

[21] F. Brooks: *The Mythical Man Month*. Addison-Wesley, 1975.

[22] B. Clark: *The Effects of Software Process Maturity on Software Development Effort*. PhD Thesis, University of Southern California, 1997.

[23] B. Clark, S. Devnani-Chulani, and B. Boehm: “Calibrating the COCOMO II Post-Architecture Model”. In *Proceedings of the 20th International Conference on Software Engineering*, pages 477-480, 1998.

[24] M. Colledge, J. Johnson, R. Pare, I. Sande: “Large Scale Imputation of Survey Data”. In *Proceedings of the Section on Survey Research Methods*, American Statistical Association, pages 431-436, 1978.

[25] S. Conte, H. Dunsmore, and V. Shen: *Software Engineering Metrics and Models*. Benjamin/Cummings Publishing Company, 1986.

[26] B. Cox and R. Folsom: "An Empirical Investigation of Alternate Item Nonresponse Adjustments". In *Proceedings of the Section on Survey Research Methods*, American Statistical Association. 219-223, 1978.

[27] L. Ernst: "Weighting to Adjust For Partial Nonresponse". In *Proceedings of the Section on Survey Research Methods*, American Statistical Association. 468-472, 1978.

[28] G. Finnie and G. Wittig: "A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models". In *Journal of Systems and Software*, 39:281-289, 1997.

[29] B. Ford: "Missing Data Procedures: A Comparative Study". In *Proceedings of the Social Statistics Section*, American Statistical Association. 324-329, 1976.

[30] B. Ford: "An Overview of Hot-Deck Procedures". In W. Madow, I. Olkin, and D. Rubin (eds.): *Incomplete Data in Sample Surveys, Volume 2: Theory and Bibliographies*. Academic Press, 1983.

[31] J. Frane: "Some Simple Procedures for Handling Missing Data in Multivariate Analysis". In *Psychometrika*, 41(3):409-415, 1976.

[32] G. Furnival and R. Wilson: "Regressions by Leaps and Bounds". In *Technometrics*, 16(4):499-511, 1974.

[33] P. Gardner: "Scales and Statistics". In *Review of Educational Research*, 45(1), 43-57, 1975.

[34] O. Gilley and R. Leone: "A Two-Stage Imputation Procedure for Item Nonresponse in Surveys". In *Journal of Business Research*, 22:281-291, 1991.

[35] A. Gray and D. MacDonnell: "A Comparison of Techniques for Developing Predictive Models of Software Metrics". In *Information and Software Tehcnology*, 39: 425-437, 1997.

[36] Y. Haitovsky: "Missing data in regression analysis". In *Journal of the Royal Statistical Society*, B30,67-81, 1968.

[37] W. Hayes: *Statistics.* Fifth Edition, Hartcourt Brace College Publishers, 1994.

[38] D. Heitjan: "Annotation: What Can Be Done about Missing Data? Approaches to Imputation". In *American Journal of Public Health*, American Public Health Association, pages 548-550, 1997.

[39] Q. Hu: "Evaluating alternative Software Production Functions". In *IEEE Transactions on Software Engineering*, 23(6):379-387, 1997.

[40] B. Ives, M. Olson, and J. Baroudi: "The Measurement of User Information Satisfaction". In *Communications of the ACM.* 26(10), 785-793, 1983.

[41] R. Jeffery, M. Ruhe, and I. Wieczorek: "A Comparative Study of Cost Modeling Techniques Using Public Domain Multi-Organizational and Company-Specific Data". To appear in *Proceedings of the European Software Control and Metrics Conference (ESCOM)*, 2000. Also, to appear in *Information and Software Technology*, 2000.

[42] R. Jensen: "A Comparison of the Jensen and COCOMO Schedule and Cost Estimation Models". In *Proceedings of the International Society of Parametric Analysis*, pages 96-106, 1984.

[43] C. Jones: *Programming Productivity*. McGraw-Hill, 1986.

[44] H. Jung and R. Hunter: "Modeling the Assessor Effort in Software Process Assessment". Submitted for publication, 1999.

[45] J. Kaiser: "The Effectiveness of Hot-Deck Procedures in Small Samples". Paper presented at the Annual Meeting of the American Statistical Association, 1983.

[46] L. Kaufman and P. Rousseeuw: *Finding Groups in Data.* John Wiley & Sons, 1990.

[47] C. Kemerer: "An Empirical Validation of Software Cost Estimation Models". In *Communications of the ACM*, 30:416-429, 1987.

[48] J. Kim and J. Curry: "The Treatment of Missing Data in Multivariate Analysis". In *Social Methods & Research*. 6:215-240, 1977.

[49] B. Kitchenham and N. Taylor: "Software Project Development Cost Estimation". In *Journal of Systems and Software*, 5:267-278, 1985.

[50] B. Kitchenham: "Empirical Studies of Assumptions that Underlie Software Cost-Estimation Models". In *Information and Software Technology*, 34(4):211-218, 1992.

[51] J. Kromrey and C. Hines: "Nonrandomly Missing Data In Multiple Regression: An Empirical Comparison Of Common Missing-Data Treatments". In *Educational and Psychological Measurement*, 54(3), 573-593, 1994.

[52] S. Labovitz: "Some Observations on Measurement and Statistics". In *Social Forces*, 46(2), 151-160, 1967.

[53] S. Labovitz: “The Assignment of Numbers to Rank Order Categories”. In *American Sociological Review*, 35:515-524, 1970.
[54] S-Y Lee and Y-M Chiu: “Analysis of Multivariate Polychoric Correlation Models with Incomplete Data”. In *British Journal of Mathematical and Statistical Psychology*, 43:145-154, 1990.
[55] J. Lepowski, J. Landis, and S. Stehouwer: “Strategies for the Analysis of Imputed Data From A Sample Survey: The National Medical Care Utilization and Expenditure Survey”. In *Medical Care*, 25:705-716, 1987.
[56] R. Little and D. Rubin: *Statistical Analysis with Missing Data*. John Wiley & Sons, 1987.
[57] R. Little: “Missing-Data Adjustments in Large Surveys”. In *Journal of Business & Economic Statistics*, 6(3), 287-296, 1988.
[58] R. Little: “Regression With Missing X’s: A Review”. In *Journal of the American Statistical Association*, 87(420):1227-1237, 1992.
[59] N. Malhorta: “Analyzing Marketing Research Data with Incomplete Information on the Dependent Variable”. In *Journal of Marketing Resea*rch, 24:74-84, 1987.
[60] J. Matson, B. Barrett, and J. Mellichamp: “Software Development Cost Estimation Using Function Points”. In *IEEE Transactions on Software Engineering*, 20(4), 275-287, 1994.
[61] K. Maxwell, L. Van Wassenhove, and S. Dutta: “Performance Evaluation of General and Company Specific Models in Software Development Effort Estimation”. In *Management Science*, 45(6), 1999.
[62] G. Milligan and M. Cooper: “A Study of Standardization of Variables in Cluster Analysis”. In *Journal of Classification*, 5:181-204, 1988.
[63] Y. Miyazaki, A. Takanou, H. Nozaki, N. Nakagawa, and K. Okada: “Method to Estimate Parameter Values in Software Prediction Models”. In *Information and Software Technology*, 33:239-243, 1991.
[64] Y. Miyazaki, M. Terakado, K. Ozaki, and H. Nozaki: “Robust Regression for Developing Software Estimation Models”. In *Journal of Systems and Software*, 27:3-16, 1994.
[65] A. Mockus: “Missing Data in Software Engineering”. In K. El Emam and J. Singer (eds.): *Empirical Methods in Software Engineering*. The MIT Press (to appear), 2000.
[66] T. Mukhopadhyay and S. Kekre: “Software Effort Models For Early Estimation of Process Control Applications”. In *IEEE Transactions on Software Engineering*, 18(10), 915-924, 1992.
[67] A. Myrvold: “Data Analysis for Software Metrics”. In *Journal of Systems and Software*, 12:271-275, 1990.
[68] J. Nunnally and I. Bernstein: *Psychometric Theory.* McGraw Hill, 1994.
[69] S. Oligny, P. Bouque, and A. Abran: “An Empirical Assessment of Project Duration Models in Software Engineering”. In *Proceedings of the Eighth European Software Control and Metrics Conference*, 1997.
[70] L. Pickard, B. Kitchenham, and P. Jones: “Comments on: Evaluating Alternative Software Production Functions”. In *IEEE Transactions on Software Engineering*, 25(2):282-283, 1999.
[71] S. Rahhal: *An Effort Estimation Model for Implementing ISO 9001 in Software Organizations*. Master's Thesis, School of Computer Science, McGill University, October 1995.
[72] S. Rahhal and N. Madhavji: “An Effort Estimation Model for Implementing ISO 9001”. In *Proceedings of the Second IEEE International Software Engineering Standards Symposium*, pages 278-286, 1995.
[73] M. Raymond and D. Roberts: “A Comparison of Methods for Treating Incomplete Data in Selection Research”. In *Education and Psychological Measurement*, 47:13-26, 1987.
[74] M. Raymond: “Missing Data in Evaluation Research”. In *Evaluation & the Health Profession*, 9(4):395-420, 1986.
[75] P. Roth: “Missing Data: A Conceptual Review for Applied Psychologists”. In *Personnel Psychology*, 47:537-560, 1994.
[76] P. Roth and F. Switzer: “A Monte Carlo Analysis of Missing Data Techniques in a HRM Setting”. In *Journal of Management*, 21(5), 1003-1023, 1995.
[77] D. Rubin: *Multiple Imputation for Nonresponse in Surveys*. John Wiley & Sons, 1987.
[78] I. Sande: “Hot-Deck Imputation Procedures”. In W. Madow and I. Olkin (eds.): *Incomplete Data in Sample Surveys, Volume 3: Proceedings of the Symposium*. Academic Press, 1983.
[79] M. Shepperd and C. Schofield: “Estimating Software Project Effort Using Analogies”. In *IEEE Transactions on Software Engineering*, 23(12):736-743, 1997.
[80] S. Siegel and J. Castellan: *Nonparametric Statistics for the Behavioral Sciences*. McGraw Hill, 1988.

[81] P. Spector: “Ratings of Equal and Unequal Response Choice Intervals”. In *The Journal of Social Psychology*, 112:115-119, 1980.

[82] K. Srinivasan and D. Fisher: “Machine Learning Approaches to Estimating Software Development Effort”. In *IEEE Transactions on Software Engineering*, 21(2):126-137, 1995.

[83] S. Stevens: “Mathematics, Measurement, and Psychophysics”. In S. Stevens (ed.), *Handbook of Experimental Psychology*, John Wiley, 1951.

[84] G. Subramanian and S. Breslawski: “Dimensionality Reduction in Software Development Effort Estimation”. In *Journal of Systems and Software*, 21:187-196, 1993.

[85] University of Southern California: *COCOMO 2.0 Model User’s Manual*. Version 1.1, 1994.

[86] P. Velleman and L. Wilkinson: “Nominal, Ordinal, Interval, and Ratio Typologies Are Misleading”. In *The American Statistician*, 47(1), 65-72, 1993.

[87] F. Walkerden and R. Jeffery: “Software Cost Estimation: A Review of Models, Process, and Practice”. In *Advances in Computers*, 44:59-125, 1997.

[88] C. Walston and C. Felix: “A Method of Programming Measurement and Estimation”. In *IBM Systems Journal*, 1:54-73, 1977.

[89] H. Weisberg: *Central Tendency and Variability.* Sage Publications, 1992.

[90] C. Wrigley and A. Dexter: “Software Development Estimation Models: A Review and Critique”. In *ASAC 1987 Conference*, University of Toronto, 1987.