Reliability and Validity in Comparative Studies of Software Prediction Models

Ingunn Myrtveit, Erik Stensrud, *Member*, *IEEE*, and Martin Shepperd

Abstract—Empirical studies on software prediction models do not converge with respect to the question "which prediction model is best?" The reason for this lack of convergence is poorly understood. In this simulation study, we have examined a frequently used research procedure comprising three main ingredients: a single data sample, an accuracy indicator, and cross validation. Typically, these empirical studies compare a machine learning model with a regression model. In our study, we use simulation and compare a machine learning and a regression model. The results suggest that it is the research procedure itself that is unreliable. This lack of reliability may strongly contribute to the lack of convergence. Our findings thus cast some doubt on the conclusions of any study of competing software prediction models that used this research procedure as a basis of model comparison. Thus, we need to develop more reliable research procedures before we can have confidence in the conclusions of comparative studies of software prediction models.

Index Terms—Software metrics, cost estimation, cross-validation, empirical methods, arbitrary function approximators, machine learning, estimation by analogy, regression analysis, simulation, reliability, validity, accuracy indicators.

1 Introduction

Predicting software development effort with high precision is still a largely unsolved problem. Consequently, there is an ongoing, high level of activity in this research field. A large number of different prediction models[1] have been proposed over the last 20+ years. These range from mathematical functions (e.g., regression analysis and COCOMO) to machine learning models (ML) (e.g., estimation by analogy—EBA, classification, and regression trees—CART, and artificial neural networks—ANN). In contrast to a regression model which is defined by a mathematical formula, the ML models typically are not defined by a mathematical formula but may take on many different shapes. Such "functions" are also termed arbitrary function approximators (AFA). In this study, we will use the term AFA rather than ML to make the distinction between mathematical functions and functions that are not constrained to predetermined form.

Despite this substantial level of research activity, we are still not in a strong position to advise practitioners as to what prediction models they should select because the studies do not converge with respect to the question "which model is best?" Indeed, very contradictory results have been reported in studies comparing an AFA with a function. Furthermore, the performance of AFAs varies wildly across studies.

Some studies conclude that EBA models outperform regression models [33]. Other studies find the exactly opposite result, namely, that regression models are superior to EBA models [29]. Other studies again find CART models superior to regression models [5] whereas other studies report the opposite result [6]. Others again find ANN models superior to regression models [35] whereas Jørgensen reports the opposite result [19].

So far, the lack of convergence of studies on software prediction models is poorly understood, and it has been a puzzle to the research community on software prediction systems for many years. Clearly, we need to consolidate the knowledge on software prediction models and research procedures; we need to understand why we have obtained so wildly opposing conclusions on this matter.

We may speculate on various explanations of the reason for contradictory conclusions. One obvious explanation is that the studies vary in their quality of execution (data quality, research procedure quality, etc.). Many techniques require considerable expertise to use; so, some studies may use the technique more effectively than others. This is also a problem in the machine learning and data mining communities [26]. To put it bluntly, a badly performed study can arrive at any conclusion.

Another explanation may be that the lack of convergence is due to spurious effects in otherwise well conducted studies hampered by small sample size. Many software studies do indeed suffer from small sample size.

A third explanation may be that the commonly used *measuring procedures* suffer from some fundamental flaws so that they are either invalid, unreliable, or both. Indeed, Foss et al. [15] investigated and questioned the validity of several measures ("accuracy indicators") that are in frequent use in

1. We use *prediction* model and *estimation* model as synonyms.

• I. Myrtveit is with the Norwegian School of Management BI, Elias Smiths vei 15, Box 580, N-1301 Sandvika, Norway.
E-mail: ingunn.myrtveit@bi.no.
• E. Stensrud is with Myrtveit og Stensrud ANS, Austliveien 30, 0752 Oslo, Norway. E-mail: erik.stensrud@ieee.org.
• M. Shepperd is with the School of Design, Engineering and Computing, Bournemouth University, Poole House, P104d Bournemouth, BH12 5BB, United Kingdom. E-mail: mshepper@bmth.ac.uk.

Manuscript received 19 July 2004; revised 28 Dec. 2004; accepted 1 Mar. 2005; published online 26 May 2005.
Recommended for acceptance by P. Jalote.
For information on obtaining reprints of this article, please send e-mail to: tse@computer.org, and reference IEEECS Log Number TSE-0147-0704.

empirical software engineering. They showed that several of these accuracy indicators do not consistently select the best from two competing, linear prediction models. In other words, the accuracy indicators did not measure what they were supposed to measure and were thus misleading for this purpose. Furthermore, both Myrtveit and Stensrud [29] and Kitchenham et al. [22] showed that different accuracy indicators lead to rank reversal problems, i.e., that one particular accuracy indicator selects model *A* whereas another selects model *B* as best. Clearly, it is highly undesirable that the selection of competing models depends on the choice of the measure as long as we lack knowledge on which measure is the most valid and reliable.

Researchers may approach this problem, the lack of convergence, in two steps. The first and crucial step is to do a careful evaluation of the individual studies' research procedures. If the research procedures are unreliable, they will not yield consistent results and this might thus explain the lack of convergence. In fact, several recent findings directly or indirectly question the validity or reliability of our measuring procedures [15], [22], [29], [25].

Given that the research procedures are reliable, we must look elsewhere for explanations. For example, small sample size may also account for the lack of convergence. If this is the case, it may be appropriate to perform a meta-analysis.

"Meta-analysis involves aggregating results across studies in order to obtain a more powerful and stable estimate of effect magnitudes. ... Meta-analysis is extremely useful in aggregating well-done studies hampered by small sample size. ... Meta-analysis is, however, no substitute for careful evaluation of individual studies' procedures and results, and it was never intended as a "meat-grinder" to average out results of studies that vary in their quality of execution. Careful evaluation of individual studies may suggest why the effect of interest occurred in certain studies and not in others" ([31, p. 101]).

Recent research by Foss et al. has shown that the frequently used accuracy indicators are invalid in studies comparing linear models [15]. This study extends their study in several directions by investigating:

- Comparisons between linear and nonlinear (i.e., AFA) prediction models.
- The reliability (rather than the validity) of the commonly used measuring procedure.
- The whole measuring procedure, i.e., cross-validation applied to a single sample and evaluated with commonly used goodness of fit accuracy indicators, rather than a part of it (i.e., the accuracy indicator, only).

The paper is organized as follows: Section 2 defines some central terminology. Section 3 provides a taxonomy for cost estimation models that is useful for this study. Section 4 provides a short summary of empirical studies on cost estimation. The purpose of this section is just to show that there is a substantial body of recent research in this area. Section 5 presents the common validation procedure used in many of the studies mentioned in the previous section and some of the arguments used to advocate the procedure. We also introduce statistical terminology. Section 6 describes our research method, a simulation study. Section 7 presents the results whereas Section 8 discusses threats to validity. Section 9 concludes by discussing the implications for researchers in empirical software engineering and suggests some topics for further research.

2 Terminology

The term *Accuracy indicator* is often used in a dual sense in the software engineering literature denoting both the basic measure and the summary statistic of the measure. As an example, MRE (magnitude of relative error) is a basic measure whereas MMRE (mean MRE) is a summary statistic of this measure. This particular type of summary statistic is a goodness-of-fit statistic. In this study, we use the term *accuracy indicator* to denote the summary statistic, only. We stick to this term because of its widespread use in software engineering.

Measure—In this study, it means the basic measure of the accuracy indicator, e.g., MRE, AR (absolute residual), and BRE (balanced relative error). We may also use the term *basic measure* to stress what we mean.

Measuring procedure—This term is best defined by using an example of a measuring procedure: The commonly used measuring procedure in software engineering in comparative studies on prediction models comprises cross-validation applied to a single sample and evaluated with one or more accuracy indicators.

3 Types of Cost Estimation Models

The term *arbitrary function approximator*, AFA, is probably not wide spread in the software engineering community. In this section, we provide a taxonomy in order to better explain the concept of an AFA. (A taxonomy explains a concept, e.g., AFA, by highlighting the differences and similarities between this concept and related concepts.) In this study, we are particularly interested in how mathematical functions relate to AFAs since these are the two types of estimation models we use in the study. There exist a number of taxonomies, but none of them serve our purpose quite well, and they all suffer from some flaws [7]. Also, classification schemes are subjective and there is no agreement about the best one [7].

There are several approaches to cost estimation. One can group them as in Fig. 1. Broadly, we may distinguish between sparse-data methods and many-data methods. Sparse-data methods are estimation methods requiring few or no historical data. They include Analytic Hierarchy Process, AHP [34], expert judgment [40], and automated case-based reasoning—CBR[2] [28].

Many-data methods may be subdivided into functions and arbitrary function approximators (AFA). Functions are of the general form $y = Ax^B$, that is, there is a mathematical relationship between the variables expressed in a formula. Linear regression models belong to this class. By contrast, arbitrary function approximators do not make any assumptions regarding the relationship between the predictor and

2. Obviously, there is a spectrum between sparse and many-data methods. CBR may belong to both. If CBR is used to identify the closest case, it is a many-data method. EBA is an example of this use of CBR. On the other hand, if you use CBR to reason from a case that is already selected, it is a single-data method. We therefore view EBA as a kind of CBR method.

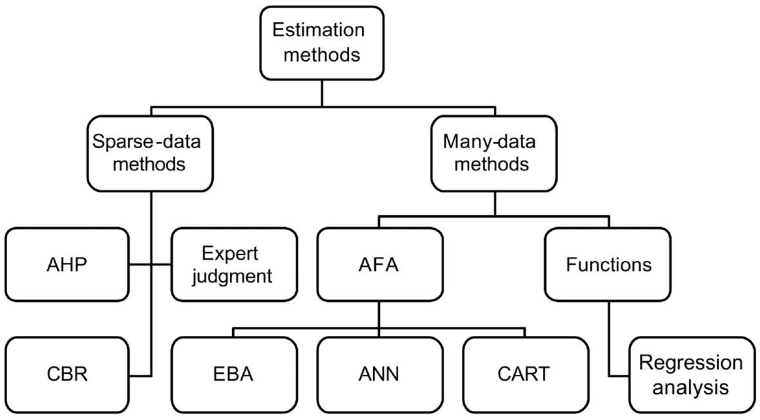

Fig. 1. A taxonomy of SW cost estimation methods.

response variables (i.e., between x and y). The argument for proposing them is that "*it is very difficult to make valid assumptions about the form of the functional relationship between variables*.... [Therefore] ... [the] *analysis procedure should avoid assumptions about the relationship between the variables....using more complex functional forms would be difficult since we usually have a poor understanding of the phenomena we are studying*." [1]. EBA, CART, and ANN models belong to the AFA class.

In this paper, we only investigate validation methods applied to many-data methods.

4 Previous Work on Software Prediction Methods

There exists a relatively large number of empirical studies on software cost estimation models. In particular there are a large number of studies on regression analysis models since this model often serves as the baseline against which the performance of the other models is compared. See the *Encyclopedia of Software Engineering* [7] for an overview. Most of the studies have applied the ordinary least squares method. Some studies have also reported on various robust regression methods [14], [16], [18], [27], [30], [43].

There is also a substantial body of research on AFAs. The latter include CART models [3], [5], [20], [35], OSR—Optimized Set Reduction, a subtype of CART [1], [2], [19], EBA models [17], [29], [33], [38], [41], [44], and, finally, ANN models [32], [35] [44].

In these studies, the default accuracy indicator reported is MMRE. More recent studies tend, however, to report more than one indicator (e.g., MMRE, median MRE, MBRE, MAR). The accuracy indicators are computed following standard evaluation processes such as cross-validation applied to a single sample [7]. Thus, the research procedure in this study is well aligned with the research procedures of previous studies.

5 Measuring Procedures

5.1 Reliability and Validity

Measurement is crucial to empirical software engineering. At the most general level, measuring procedures need to possess two basic properties: reliability and validity. We therefore have to know whether the measuring procedures are in fact reliable and valid. "First, one can examine the reliability of an indicator. Fundamentally, reliability concerns the extent to which an experiment, test, or any measuring procedure yields the same results on repeated trials [...] the more consistent the results given by repeated measurements, the higher the reliability of the measuring procedure [...] But an indicator must be more than reliable if it is to provide an accurate representation of some abstract concept. It must also be valid. In a very general sense, any measuring device is valid if it does what is intended to do. An indicator of some abstract concept is valid to the extent that it measures what it purports to measure." [8].

There are two basic kinds of error that affect empirical measurements, random error and nonrandom error. Random error is the term used to designate all those chance factors that confound the measurement of any phenomenon. For example, if the shots fired from a well-anchored rifle are scattered widely about the target, then the rifle is unreliable. But, if the shots are concentrated around the target, the rifle is reliable. Nonrandom error has a systematic biasing effect on measuring instruments. If all shots aimed at the center of the target hit approximately the same location but not the center, then some form of nonrandom error has affected the targeting of the rifle. That is, you are consistently and systematically measuring the wrong value for all respondents. This measure is reliable, but not valid (that is, it is consistent but wrong). Thus, reliability concerns the degree to which results are consistent across repeated measurements. Validity is the degree that a particular indicator measures what it is supposed to measure rather than reflecting some nonrandom measurement error.

The measuring procedure in comparative studies of software prediction models frequently is generally as follows: The prediction models are validated on a single sample (data set) by computing an accuracy indicator using cross-validation. The model obtaining the highest accuracy (i.e., a low value of the accuracy indicator) is deemed "best" (In more recent studies, significance tests are added). In the following sections, we provide further details on cross-validation and the accuracy indicators that are frequently used in empirical software engineering.

5.2 Cross-Validation

Cross-validation is a way of obtaining nearly unbiased estimators of prediction error. The method consists of

1. deleting the observations from the data set one at a time,
2. calibrating the model to the $n-1$ remaining observations,
3. measure how well the calibrated model predicts the deleted observation, and
4. averaging these predictions over all n observations [12].

In software engineering, MRE and MMRE are used as the de facto standard in Steps 3 and 4, respectively [7]. More recently, some researchers have replaced MRE with other measures (e.g., BRE or absolute residual, AR) in Step 3. Also, in Step 4, the median is now routinely used in addition to the mean.

Cross-validation comes in two variants, n-fold and v-fold (where n is sample size and k is some number smaller than n). The default cross-validation method is n-fold cross-validation, also termed leave-one-out cross-validation. In the software engineering and machine learning communities, a variant of the cross-validation method, v-fold cross-validation, is also used in Step 1 [2], [4]. V-fold cross-validation divides the data set into v subsets, each with approximately t observations with $t > 1$. That is, $v^*t \approx n$. Thus, instead of deleting one observation at a time, t observations are deleted each time. In the machine learning communities, these subsets are often termed training sets and test sets, respectively.

In the literature, there has been a discussion on which cross-validation method is the better. (Also, bootstrapping has been suggested.) There has been a substantial amount of theoretical and empirical study on cross-validation. Kuha [24] presents many of these in a bibliography—which indicates that this is still an unsettled matter. We have chosen to investigate n-fold cross-validation in this study for two reasons. First, it is a widely used variant. Second, since no conclusive evidence exists regarding n-fold versus v-fold, we have reverted to what to us seems like common sense. To us, n-fold cross-validation makes more sense than v-fold for the following reason. Viewed from a practitioner's standpoint, we want a method that comes as close as possible to a realistic real world situation.

What, then, is a real-world situation closest to, n-fold cross-validation or v-fold cross-validation? To us, a realistic scenario is as follows: We have a data set with n relevant (nonobsolete) historical projects, and we are to estimate a single new project. Now, we think it would be wise to use all the n observations to calibrate a prediction model and not just, say, half of the observations. This real-world situation seems almost perfectly approximated by n-fold cross-validation where the model is calibrated with $n-1$ observations, i.e., only one observation less than we would have in the real world case. As opposed to this, the v-fold cross-validation removes t observations at a time, thereby using a smaller subset to calibrate the model than the data set that would be available in reality.

However, v-fold is less computationally intensive than n-fold and has for this reason been attractive in the ML community.

5.3 Accuracy Indicators

In this section, we present the accuracy indicators that we evaluate in this study. All of them are variants of goodness-of-fit statistics.

The most widely used evaluation criterion to assess the performance of software prediction models is the mean magnitude of relative error (MMRE). Conte et al. [10] consider $MMRE \leq 0.25$ as acceptable for effort prediction models. MRE is defined as

$$MRE = \frac{|y - \hat{y}|}{y},$$

where $y = \text{actual}$ and $\hat{y} = \text{prediction}$. There exist a number of supposed reasons to use MMRE. It is considered a versatile assessment criterion lending itself to a number of situations. The claimed advantages include the following:

1. Comparisons can be made across data sets [6], [41].
2. It is independent of units. Independence of units means that it does not matter whether effort is reported in workhours or workmonths. An MMRE will be, say, 10 percent whatever unit is used.
3. Comparisons can be made across all kinds of prediction model types [10]. This means, for example, that it is considered as a valid and reliable measure to compare AFAs with linear models.
4. It is scale independent. Scale independence means that the expected value of MRE does not vary with size. In other words, an implicit assumption in using MRE as a measure of predictive accuracy is that the error is proportional to the size (effort) of the project [39]. For example, a one person-month error for a 10 person-month project and a 10 person-month error for a 100 person-month will result in equal MREs (10 percent for both projects).

In this study, we investigate claims 1 and 3 and show that they do not hold. Foss et al. [15] demonstrated that MMRE is an invalid measure for selecting between two competing, linear prediction models, thus refuting claim 3. In this study, we show that the whole validation procedure is unreliable, thus refuting claims 1 and 3. As for claim 4, it seems to hold [36], and claim 2 obviously holds.

Another measure akin to MRE, the magnitude of error relative to the *estimate* (MER), has been proposed by Kitchenham et al. [22]. Intuitively, it seems preferable to MRE since it measures the error relative to the estimate. MER is defined as

$$MER = \frac{|y - \hat{y}|}{\hat{y}}.$$

We use the notation MMER to denote the mean MER. Kitchenham et al. also have proposed the mean (or median) of the absolute residual (MAR, MdAR) instead of MMRE. AR is defined as:

$$AR = |y - \hat{y}|.$$

Another, and simple, measure of residual error is the standard deviation (SD). It is computed as follows:

$$SD = \sqrt{\frac{\sum (y_i - \hat{y}_i)^2}{n-1}}.$$

We also propose and evaluate two other measures. They are the relative standard deviation (RSD) and the logarithmic standard deviation (LSD). RSD is defined as follows:

$$RSD = \sqrt{\frac{\sum \left(\frac{y_i - \hat{y}_i}{x_i}\right)^2}{n-1}}.$$

The variable x is Function Points in our case. The rationale behind RSD is to measure the dispersion relative to the x value (e.g., *FP*—Function Points) rather than relative to the y value (effort) to avoid one of the problems with $MMRE$ which is that small actuals (small ys) will have a disproportionate influence on the mean MRE since a number divided by a small number tends to be a large number. Contrary to MRE, which is almost uncorrelated with size [36], SD is positively correlated with size because software project data sets are often heteroscedastic. Also, unlike SD, RSD is almost uncorrelated with size.

We observe that RSD is limited to models with a single predictor variable. In many software studies this is, however, not a serious limitation since it is common to create prediction models based on FP and effort. More important, we can provide a rationale for choosing this metric as well as an interpretation of its meaning. As for the rationale, let us assume that we have the following model:

$$y = \alpha + \beta x + x\varepsilon, \tag{1}$$

where ε is normally distributed: $E(\varepsilon) = 0$ and $\mathrm{var}(\varepsilon) = \sigma^2$. This model will generate data where the variance increases with x. Dividing (1) by x gives:

$$\frac{y}{x} = \alpha \cdot \frac{1}{x} + \beta + \varepsilon. \tag{2}$$

The error term in this regression model (2), ε, is normal: $E(\varepsilon) = 0$ and $\mathrm{var}(\varepsilon) = \sigma^2$. OLS will, therefore, be an *efficient* estimating method. Let $\hat{\alpha}$ and $\hat{\beta}$ be estimates of α and β. Our prognosis (sample prediction model) for effort is then:

$$\hat{y} = \hat{\alpha} + \hat{\beta}x.$$

But,

$$\frac{\hat{y}}{x} = \hat{\alpha}\frac{1}{x} + \hat{\beta}.$$

$\frac{y}{x} - \frac{\hat{y}}{x}$ is, therefore, and estimate, e, of the error term ε. Since we also have that

$$e = \frac{y}{x} - \frac{\hat{y}}{x} = \frac{y - \hat{y}}{x}.$$

RSD is, therefore, an estimate of the standard deviation of the error term ε in the regression equation. Thus, RSD is a relevant measure of how good the prediction model is.

It remains to give RSD an interpretation making sense since x and y are measured in different units (hours versus *FP*s). We can interpret $\frac{y}{x}$ as the effort per *FP*, that is to say, the productivity. If α is close to zero or if the project is large (in terms of x), we observe that $\frac{y}{x}$ will approximate β.

We note that RSD is based on an additive model and many software effort estimation models use an exponential model. Thus, RSD may be less effective as a goodness of fit statistic for an exponential model if the exponent is significantly different from one. However, these two models are very close to each other, see Section 6.2. The exponent is 1.05 and, thus, very close to 1, see Table 2. Therefore, RSD is an appropriate measure in this single-predictor case.)

LSD is defined as follows:

$$LSD = \sqrt{\frac{\sum \left(e_i - \left(-\frac{s^2}{2}\right)\right)^2}{n-1}}.$$

The term s^2 is an estimator of the variance of the residual e_i where e_i is given by

$$e_i = \ln y_i - \ln \hat{y}_i.$$

The rationale behind LSD is as follows: Data sets with a large heteroscedasticity like the Finnish data set (cf. Section 6.1) will be very influenced by the large projects. Thus, the usual SD is more sensitive to large projects than to small projects, and it may therefore not be a stable, reliable measure for such data sets. On the other hand, LSD lends itself well to data sets that comply with a log-linear model because the residual error is independent of size (i.e., homoscedastic) on the log scale. In fact, we use a log-linear transformation for our simulation (see Section 6.2), so LSD should theoretically be more reliable than SD in this case. (The reason for the $-s^2/2$ term will become clearer in Section 6.2. See also [15] for more details.) To summarize, LSD is useful for comparing multiplicative models but it may be inappropriate for comparing additive models.

Finally, we evaluate the mean of the balanced relative error (BRE) and the inverted balanced relative error ($IBRE$) proposed by Miyazaki et al. [27].

$$BRE = \frac{(\hat{y} - y)}{y}, \ \hat{y} - y \geq 0,$$

$$BRE = \frac{(\hat{y} - y)}{\hat{y}}, \ \hat{y} - y < 0,$$

$$IBRE = \frac{(\hat{y} - y)}{y}, \ \hat{y} - y < 0,$$

$$IBRE = \frac{(\hat{y} - y)}{\hat{y}}, \ \hat{y} - y \geq 0.$$

The mean of the absolute values of BRE and $IBRE$ are termed $MBRE$ and $MIBRE$, respectively.

6 Research Method

Since the 1950s, various computer simulation techniques have become increasingly important research tools across a wide range of sciences. Software packages based on these

TABLE 1
Descriptive Statistics for Finnish Data Set

Variable	N	Mean	Median	StDev	Min	Max
FP	40	761	638	511	65	1814
Effort	38	7573	5430	6872	460	23000

techniques are also widely used in more applied fields such as engineering, finance, or environmental management, often in connection with computerized databases and electronic gathering devices.

There are several advantages of simulation compared with using real data. One advantage is that we can create a large number of samples rather than having to use only one, single sample. Thus, we obtain the *distributions* for statistical parameters that are estimated (the estimators). Using a single sample of real data, we can obtain the distribution only when it can be calculated analytically. In cases where we cannot calculate the distribution analytically, we would obtain one single value for the estimator, not its distribution. In such cases, simulation adds value. Clearly, we obtain more information and can draw more reliable conclusions based on a distribution than based on a single value.

In this study, we are interested in the reliability of the measuring procedure. In comparative studies, the measuring procedure is reliable if it consistently selects the *same* among two (or more) competing models provided other external factors are kept constant. An important external factor is the data set. Data sets may exhibit different characteristics in terms of linearity, heteroscedasticity, sample size, type, and number of variables, etc. In this study, we keep this factor constant by drawing samples from the same population. The rationale is that we want to identify which is the best model for a given population, not for a particular sample drawn from this population. It is the population model that provides unbiased predictions. The procedure is in addition *valid* if it consistently selects the "best" model, i.e., selects the true from the false model.

The simulation method in this study is aligned with the common evaluation procedure and used n-fold cross-validation. It consists of the following main steps:

```
Generate $m = 1,000$ random data samples (see details in Section 6.4)
For $i = 1$ to $m$
    For $j = 1$ to $n$ where $n$ is sample size (this loop is the n-fold cross-validation)
        Fit the AFA and OLS prediction models (see details in Section 6.5)
        Compute $\hat{y}$ and the basic measures (see details in Section 6.5)
    Next j
    Compute the accuracy indicators (MMRE, MBRE, etc.)
    Compare the accuracy indicators of the AFA and OLS models
Next $i$
Count the number of wins for each model and per type of accuracy indicator
```

TABLE 2
The Log-Linear Model (8)

Predictor	Coef	SE Coef	T	P
Constant	1.70	0.99	1.71	0.096
ln(x)	1.05	0.16	6.79	0.000
SD	0.79			
R^2	0.56			

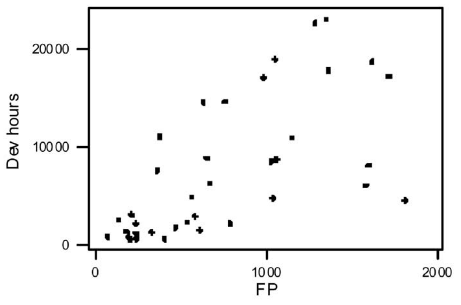

Fig. 2. Plot of effort versus FP for Finnish data set.

6.1 Data "Template"

It is important that a simulation method for generating data generates samples from a population that represents an actual population, i.e., the simulation model has to be as close as possible to an actual software data set. In this study, we use the Finnish[3] data set as a model for our simulation model. The Finnish data set exhibits properties that we consider representative of other data sets of software projects with respect to (non)linearity, heteroscedasticity, and sample size. The projects span from 65 to 1,814 function points (FP), that is, from small to medium software size. The data set consists of 40 projects. (Typical software engineering data sets in previous studies have 10 to 100 observations [7], [36], [42] although a small number are considerably larger such as the ISBSG benchmarking data set.) Two projects have missing data. The data comes from different companies, and a single person performed the data collection. This ensures high interrater reliability. The projects span from 460 to 23,000 workhours. Descriptive statistics are provided in Table 1. More details on the data may be found in Kitchenham and Kansala [21].

In Fig. 2, we have plotted effort against FP. We observe that the data are heteroscedastic (increasing variance).

6.2 Exploratory Investigation of the Population Model

It is a common, and reasonable, starting point to assume a linear model. This is further justified by recent research, using a genetic programming approach as a flexible method of model fitting, which nonetheless found no significant deviations from a linear model [11]:

3. There exist more recent versions of the Finnish data set with more observations. However, the degree of recency is irrelevant in this study.

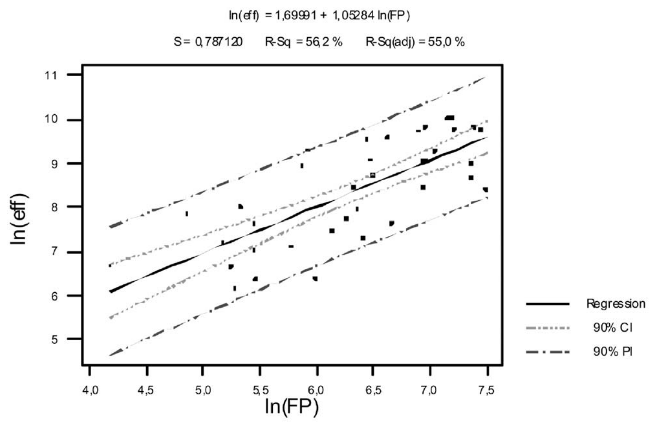

Fig. 3. Finnish data set: ln(FP) versus ln(Effort).

$$\mathrm{y} = \alpha + \beta \cdot \mathrm{x} + \mathrm{u}, \tag{3}$$

where y is effort and x is function points. If we apply (3) to the Finnish data set, we obtain the following OLS linear regression model:

$$\mathrm{y} = 877 + 8.77 \cdot \mathrm{x}. \tag{4}$$

On closer inspection of the Finnish data set in Fig. 2, we observe that it seems reasonably linear but exhibits heteroscedasticity (increasing variance). OLS regression analysis assumes that the data are homoscedastic (equal variance). Model (4) is therefore not sufficiently correct. We need to transform the data in an attempt to make them better comply with the assumptions of the OLS method, in particular the homoscedasticity assumption.

There exist several alternatives for transforming the data. One alternative is to perform a log-linear regression where we assume that the relationship between effort and FP is of the form

$$\mathrm{y} = \mathrm{e}^{\alpha}\mathrm{x}^{\beta} \cdot \mathrm{I}, \tag{5}$$

where I is lognormal with the mean equal to 1. Thus, $I = e^u$ with u normally distributed. It has been proven that if $Var(u) = \sigma^2$ and $E(u) = -\frac{\sigma^2}{2}$, then $E(I) = E(e^u) = 1$ ([13, p. 134]).

Rewriting (5) in the form:

$$\mathrm{y} = \mathrm{e}^{\alpha} \cdot \mathrm{x}^{\beta} \cdot \mathrm{e}^{\mathrm{u}} \tag{6}$$

and taking the logarithm of (6), we obtain:

$$\ln(\mathrm{y}) = \alpha + \beta \ln(\mathrm{x}) + \mathrm{u}. \tag{7}$$

A fitted line plot of the transformed data using (7) is presented in Fig. 3. Inspecting the plot, the data exhibit a reasonable homoscedasticity and linearity.

Applying (7) to the Finnish data set, we get the following OLS regression model:

$$\ln(\mathrm{y}) = 1.70 + 1.05 \cdot \ln(\mathrm{x}). \tag{8}$$

Back transforming (8), we get:

$$\mathrm{y} = \mathrm{e}^{1.70} \cdot \mathrm{x}^{1.05} = 5.47 \cdot \mathrm{x}^{1.05}. \tag{9}$$

From Table 2, we observe that the standard deviation is 0.79. Comparing (3) with (5), we can state that, in (3), we believe there to be a linear relationship between effort and FP, whereas, in (5), we believe it to be exponential. We observe, however, that model (9) fitted to the Finnish data set is not strongly exponential since the exponent is close to 1 (1.05 and with a standard error of 0.16, cf. Table 2). From this, we cannot draw any conclusions regarding returns to scale, i.e., whether to assume increasing, decreasing, like COCOMO [9], or constant returns to scale. See, for example, [23], [37] for a more detailed account of returns to scale. However, none of the two models reject the assumption of constant returns to scale. Therefore, a linear model seems to be a good first order approximation of reality.

Given that the data seem reasonably linear, we could have used (4) except for the fact that the data are heteroscedastic. Therefore, it is interesting to investigate a third model that is linear rather than log-linear but corrects the heteroscedasticity of (4).

$$\mathrm{y} = \alpha + \beta \cdot \mathrm{x} + \mathrm{x} \cdot \mathrm{u}. \tag{10}$$

In model (10), we assume a linear relationship between FP and effort as we do in (3), but unlike (3), we transform the data in an attempt to obtain a more constant variance. We investigated this model. However, we did not find that it improved on the log-linear model.

In the simulation study that follows, we have therefore opted for model (9), the log-linear model. The log-linear model also has an additional benefit compared with the two other models in that it forces the line through the origin in addition to correcting for heteroscedasticity. That is, when zero FP is delivered, zero effort is expended.

We find it useful to have performed this explorative exercise because, unfortunately, there is no strong theory to guide us in software engineering. We have therefore chosen (9) based on empirical evidence from using the Finnish data set. This empirical evidence weakly suggests a multiplicative model rather than an additive model. This finding is supported by several studies on other software engineering data sets. See, for example, [15], [36].

6.3 Simulation Model Specification

Let us assume that we have estimated the regression parameters based on a *large* sample (or alternatively, suppose that we happen to know the population) and that we therefore know the *true* (or population) regression model. Assume that the true model is exponential of the form (5) and with coefficients and standard deviation (σ) identical to model (9) in Table 2:

$$\ln(\mathrm{y}) = 1.70 + 1.05 \cdot \ln(\mathrm{x}) + \mathrm{u}, \sigma = 0.79. \tag{11}$$

Model (11), i.e., the regression model including the error term and the standard deviation figure, is our simulation model describing the population. The parameters describing the population should describe aspects of the population that are relevant to the simulation study. For this study, it is relevant to include a model for the variance of the population. The error term, u, accounts for its stochastic nature. Some projects use more effort, and some less effort, than the expected effort based on FP counts. u is normal with mean equal to $-\sigma^2/2$ and variance equal to σ^2 ([13, p. 134]). This simulation model can be used to generate data samples with characteristics similar to the Finnish data set.

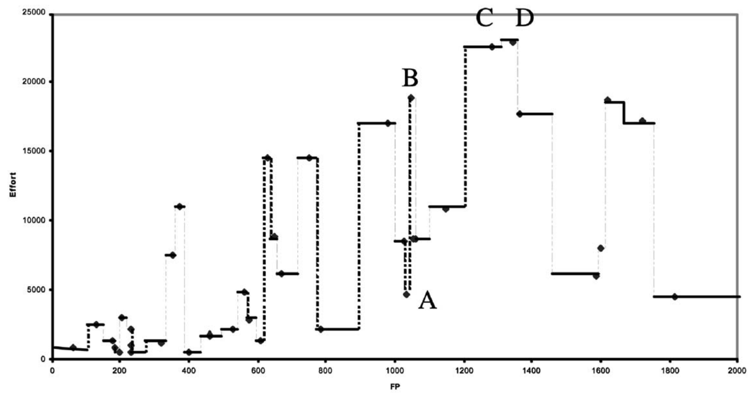

Fig. 4. Estimation by analogy model (ANGEL), closest analogy, for Finnish data set.

6.4 Generation of Data Samples

If the population is given by model (11), we may simulate sampling from this population. Let the span of the x variable, function points, be the same as for the Finnish data set: [65, 1814], and let x be uniformly distributed in this range. Furthermore, let the sample size be the same as for the Finnish data set, i.e., $n = 40$. Each sample was generated with the following algorithm:

```
For $i = 1$ to $n$ /* observation $i$ */
    Draw a random integer $x(i)$ from the uniform $x$ distribution;
    Draw a random $u(i)$ from the normal distribution $N\left(\frac{-\sigma^2}{2}, \sigma\right)$;
    Compute $y(i) = e^{\alpha} \cdot x^{\beta} \cdot e^{u(i)}$;
    Store observation $i$ as $x(i), y(i)$;
Next $i$
```

To draw random numbers from a uniform distribution, we used the RANDBETWEEN() function, and to draw numbers from a normal distribution, we used the function RANDOM(), both functions provided in Microsoft® Excel 2000, the latter in the Excel Add-In: Analysis ToolPak. In total, we created 1,000 samples.

6.5 Specification of the Prediction Models

We specified two kinds of models, one linear regression model and one estimation-by-analogy (EBA) model as implemented in the tool ANGEL [33]. The EBA model is representative of arbitrary function approximators (AFAs). In Fig. 4, we have depicted the functional form using the closest analogy (nearest neighbor). The linear regression model was fitted to the data using the ordinary least squares (OLS) method. We used a multiplicative model of the form:

$$y = Ax^B.$$

To fit the OLS model to the data, we transformed the sample data to log-log and applied a log-log, additive, linear OLS model of the form:

$$\ln(y) = \ln(A) + B \cdot \ln(x).$$

As for the EBA model, it was fitted to the data by finding the number of closest analogies (or nearest neighbors) minimizing some criterion. As a fitting criterion, we minimized all the accuracy indicators in turn, for example, "minimum MMRE," "minimum MAR," etc. The predicted effort was calculated taking the mean effort of the corresponding closest analogies.

Finally, comparing the accuracy indicators of the OLS and the AFA models, when comparing, say, the pair wise MAR values, we used the AFA model fitted with the "minimum MAR" method. Likewise for the other measures, MMRE, MBRE, etc. The closest analogies were found by measuring the Euclidean distances in the FP dimension, the only independent variable used in this study.

```
/***** Fitting the AFA ****/

For $k = 1$ to $n$  // $k$ is the number of closest analogies
    For j = 1 to $n$ where $n$ is sample size (this loop is the n-fold cross-validation)
        Find the $k$ closest analogies (or nearest neighbors);
        Compute $\hat{y} = \frac{1}{k}\sum_{i=1}^{k} y_i$;
        Compute the basic measures (MRE, BRE, etc.);
    Next $j$
Next $k$
Compute all the accuracy indicators (MMRE, MBRE, etc.);
Among the $n$ resulting AFA models, select the "best" AFA
        model, i.e., select the $k$ that minimises one particular
        criterion, e.g., "Minimum MMRE";

/**** Fitting the linear regression model *****/
For $j = 1$ to $n$ where $n$ is sample size (this loop is the n-fold cross-validation)
    Compute $\hat{y} = e^a \cdot x^b$, where $a$ and $b$ are obtained in the
      preceding step;
    Compute the basic measures (MRE, BRE, etc.);
Next $j$
Compute all the accuracy indicators;
```

Effectively, this procedure favors the AFA. The reason is that when we compare, say, MMRE(AFA) with MMRE(OLS), we pick the lowest MMRE(AFA) because we use the number of closest analogies that minimizes MMRE for the AFA.

To investigate if this impacts on results, i.e., on the conclusions of the study, we also did a variant where we kept the fitting criterion constant, say, MBRE as the fitting

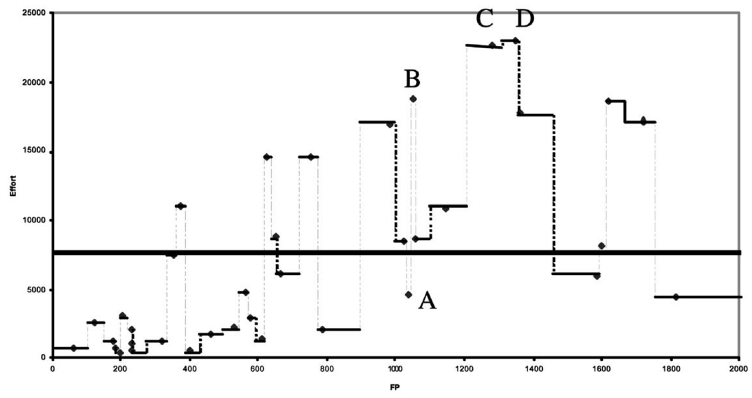

Fig. 5. ANGEL prediction model taking the average over the 38 closest projects, Finnish data set (thick solid line).

criterion for the AFA, picked the AFA model minimizing MBRE and used this model in comparing MMRE(AFA) with MMRE(OLS), and so on. In this case, we should only favor the AFA when comparing MBRE(AFA) with MBRE(OLS) but not when comparing, say, MMRE(AFA) and MMRE(OLS).

6.6 General Presentation of the EBA Estimating Approach

EBA methods identify analogues (or similar cases) in the database. Commonly used similarity measures are Euclidean distance and correlation coefficients. Euclidean distance is employed in the EBA tool ANGEL [33]. ANGEL predicts effort based on identifying analogous or *similar* projects in p-dimensional feature space (where each project is characterised by p features or independent variables) for which effort is known. The predicted effort is basically identical to the effort of the most similar project in the "nearest neighbor" case. When identifying the "k nearest neighbors," the predicted effort is the average effort of these k nearest neighbors. The ANGEL model is illustrated in Fig. 4 for $k = 1$, the single nearest neighbor case.

Using the Finnish data set to see how ANGEL works, the most similar project is the project which is closest in terms of FP (in the univariate case). For example, to estimate the effort for a 1,300 FP project, we would measure the distance to every project in the database and identify project C (1,282 FP) in Fig. 4 as closest (a distance of 18 FP). C is closer than for example D (1,347 FP). The effort for C is 22,670 workhours. Therefore, the estimated effort for the 1,300 FP project would be 22,670 workhours in the "nearest neighbor" case. For $k = 2$, the predicted effort would be the average of C and D.

We observe that the similarity measurements may be used to *rank* all the projects in the database with respect to closeness with project X where X is 1,300 FP in our particular example.

ANGEL may also compute estimates that are averages of the k closest projects where k may be any value chosen by the user. At the extreme, we may average over all the 38 projects in the Finnish data set (i.e., use the sample mean). In this case, the ANGEL function would be the solid, horizontal, thick line in Fig. 5. (The average effort of all projects is 7,573 workhours.) For any other k, the model would be a stepwise function somewhere in between the solid thick line and the collection of the thin horizontal, discontinuous line segments in Fig. 5.

6.7 Validating the Prediction Models

For each data sample, we fitted and validated the two models using n-fold cross-validation. Both models were therefore fitted to subsamples of size $n - 1$ (n is sample size). For each subsample, we computed the predicted effort, $\hat{y}$, of the left-out observation, and all the different error terms (e.g., MRE, AR, BRE, etc.). Finally, we computed all the different accuracy indicators for this sample and per model (e.g., MMRE, MAR, MBRE, etc.).

7 Results

The results for the comparisons are presented in Table 3 and Table 4. We have reported the number of times each model obtains the highest accuracy.

The results in Table 3 suggest that the AFA model obtains the highest accuracy with a 99.4 percent probability using MMRE both as the fitting method for the AFA model as well as the selection criterion. This comes as no surprise. MMRE tends to select the bad to the good provided the bad model has smaller slope and/or intercept than the good (true) model [15]. In other words, a model fitted with the "minimum MMRE" method will underfit whereas a regression model fitted with the "minimum least squares" method will fit to the central tendency of the data and, thus,

TABLE 3
Results of Comparison between AFA and OLS Models, Highest Accuracy

Variable	N	AFA(%)	OLS (%)
MAR	1000	15.4	84.6
MMRE	1000	99.4	0.6
MMER	1000	2.8	97.2
MBRE	1000	8.1	91.9
MIBRE	1000	30.6	69.4
RSD	1000	29.2	60.8
LSD	1000	68.4	31.6

TABLE 4
Results of Comparison between AFA and OLS Models, Highest Accuracy, Minimizing Only MBRE for the AFA

Variable	N	AFA(%)	OLS (%)
MAR	1000	10.9	89.1
MMRE	1000	86.5	13.5
MMER	1000	2.0	98.0
MBRE	1000	8.1	91.9
MIBRE	1000	20.5	79.5
RSD	1000	24.5	75.5
LSD	1000	60.3	39.7

not minimize MMRE. As we have used "minimum MMRE" as criterion to fit the AFA to each sample whereas we have used the OLS method to fit the regression line to each sample, we should expect the AFA to have highest accuracy in terms of MMRE.

Using the "minimum MMER" fitting method for the AFA and MMER as selection criterion between the AFA and OLS models, the OLS model has highest accuracy 97.2 percent of the time. In other words, we obtain complete opposite results by using another fitting and selection criterion. This is not unexpected. Minimizing MMER is quite close to minimizing OLS. Both will fit to some central tendency of the data as opposed to MMRE that fits to some value below the central tendency.

Using the "minimum MAR" fitting method for the AFA and MAR as selection criterion between the AFA and OLS models, the OLS model has highest accuracy 84.6 percent of the time. Similarly, using the "minimum MBRE" fitting method for the AFA and MBRE as selection criterion between the AFA and OLS models, the OLS model has highest accuracy 91.9 percent of the time.

An interesting observation is that several of the accuracy indicators (MAR, MBRE, and MIBRE) tend to select the OLS model, but not with a very high probability. Keeping in mind that previous studies are essentially single sample studies, we observe there is still an 8-31 percent probability that the AFA obtains the highest accuracy using these three accuracy indicators.

Another interesting observation is that although we have favored the AFA model in every respect by using the same fitting and selection criterion in the comparison, it generally does not obtain the highest accuracy except when using MMRE in both cases. This may explain the results of previous studies whose results often tend to go in favor of some new proposed ML technique when compared against a linear regression model since MMRE has been the most common accuracy indicator. From Table 4, we also observe that using a "minimum X" (X=MBRE in the reported table) fitting criterion for the AFA together with a comparison accuracy indicator Y (where Y is any of the accuracy indicators), the results exhibit a similar pattern, but less consistent than in Table 3. This implies that the outcome of single sample studies would be even less consistent (reliable) with such a procedure.

8 Discussion of Threats to Validity

We believe that there are no serious threats to validity in this study. However, there are a few hypothetical ones that we discuss in this section.

Population model. One potential threat is the population model chosen. We repeat that we believe the characteristics of the population model are representative of typical software data sets. Hypothetically, it could be possible that we would obtain more consistent results, i.e., higher reliability, comparing an AFA and an OLS regression model using a different population model with different characteristics in terms of slope and intercept coefficients, (non)linearity, heteroscedasticity, kurtosis, multivariate in stead of univariate, project age, etc. As an example, suppose we vary the slope coefficient from $\beta = 1.05$ (our model) to $\beta = 1.12$ (as in the COCOMO model). As another example, suppose we used an error term with higher kurtosis. In this case, it is well-known that a least absolute deviation method (LAD) would be more efficient than the OLS method. In either case, we are not able to see that this would alter the results at all.

Sample size. Similarly, we do not see how a smaller or larger sample size should all of a sudden yield consistent (reliable) results, that is, choose the AFA, for example, with a probability above, say, 95 percent for any accuracy indicator.

Simulation procedure. In the paper, we have reported the results drawing $m = 1,000$ samples. However, we also did $m = 100$ samples. The results were very similar. We do not think that an increase to $m = 10,000$ samples would alter the results towards more consistency across and within the accuracy indicators.

Significance tests. We did not test for significance for each pair wise comparison of the accuracy indicators for the AFA and the OLS models. The reason is twofold. First, and most important, studies before 1999 tended to draw conclusions without testing for significance [29]. Therefore, our procedure is aligned with a large part of previous studies, probably the major part. Second, testing for significance would imply that a certain proportion of the results would not be significant. However, we do not expect that this would alter the overall results, for example that both MMRE and MMER would all of a sudden consistently pick the same model as best in the significant cases. Also, there would be an additional issue of which alfa-level to choose since different studies used different levels.

Other potential objections. Some may object that it is perfectly reasonable to choose the AFA as best on one sample and the OLS model as best on another sample, even when both are drawn from the same population. This would, however, leave us in an impossible position with regard to giving advice to a project manager on which model to use to predict his or her next project. The next project is, we should remember, part of the population and not of a particular sample. Therefore, we are actually in quest of the population model, and our single sample studies may give us the wrong choice a large percentage of the time as evidenced by the results in this study.

9 Conclusions

In this study, we have demonstrated that one of the commonly used measuring procedures in empirical software engineering is unreliable when used in comparative studies of software prediction models. First, the procedure is highly unreliable *across* accuracy indicators. That is, one accuracy indicator may tend to select the AFA whereas another accuracy indicator may tend to select the OLS model. Studies in the past have to some extent used different accuracy indicators. Our study thus suggests that the conclusions on "which model is best" to a large extent will depend on the accuracy indicator chosen. This is a serious problem because, at present, we have no theoretical foundation to prefer, say, MMRE to MMER or MAR to MBRE.

Second, also for most of the accuracy indicators, the results are not sufficiently reliable across the samples for the same accuracy indicator. Typically, there is a 20/80 percent split. This implies that the conclusions on "which model is best" to some extent depend on the particular sample at hand, even for samples drawn from the same population. When other studies have used samples from populations with different characteristics, this contributes further to the lack of convergence.

Third, the conclusions also depend on how the AFA is fitted to the sample. If we use the fitting criterion for the AFA also as the subsequent selection criterion, we increase the probability that the AFA obtains the highest accuracy. On the other hand, if we use one fitting criterion for the AFA and another accuracy indicator as the selection criterion (e.g., "minimize MBRE" and MMRE as selection criterion), we arrive at different conclusions regarding "which model is best." For the regression model, we do not introduce this fitting complication because it is fitted all the time with the same method, namely, OLS.

In summary, the lack of convergence in empirical studies may be attributed to a large degree to low reliability in the measuring procedure. Which of the factors in the measuring procedure that contributes most to this problematic state of affairs is uncertain. Below, we offer our thoughts on which factors we consider important.

Single sample studies. The results suggest that it may be very difficult to get compelling evidence from single sample studies.

Choice of accuracy indicator. The choice of accuracy indicator has a large impact on the results.

Fitting criterion for the AFA. The choice of fitting criterion also impacts on the results. This is a problem for AFAs, only, as the regression model is consistently fitted with the same method.

9.1 Generalization of Results

In this study, we have only evaluated the measuring procedure on one type of AFA, estimation by analogy. However, we believe that one would obtain similar results comparing a regression model with other ML type models (classification and regression trees, artificial neural networks). This, however, remains to be investigated, and we think it is prudent not to generalize our findings to other AFAs based on this study alone.

9.2 Implications for Researchers and Topics for Further Research

The implications for researchers in empirical software engineering are that the commonly used measuring procedures hinder meaningful comparative studies on prediction models. This may be an important reason why we are still not in a strong position to advise practitioners as to what prediction models they should select. The results suggest several topics for future research:

- Identify better measuring procedures.
- Investigate if, and how, one may draw conclusions from individual studies that use single data sets.
- Investigate the degree of confidence we may have in each accuracy indicator, to see if one of them stand out as more valid and reliable than the others.
- Replicate this study with other AFAs in order to generalize the findings.
- Examine the influence on the cross-validation method chosen, for example, n-fold versus v-fold, and investigate if there exist optimal values for v. There is a large body of ongoing research on variants of cross-validation (and bootstrapping for that sake).
- Investigate if other more reliable and valid measuring procedures have been used in some of the previous studies.

Acknowledgments

The authors would like to thank the anonymous reviewers for their very helpful comments. They would also like to thank Barbara Kitchenham for her review and constructive comments on an early draft of this paper and Tron Foss for his review of the simulation part. Finally, we wish to thank STTF for providing the "Finnish" data set that served as a template for the simulation model.

References

[1] L.C. Briand, V.R. Basili, and W.M. Thomas, "A Pattern Recognition Approach for Software Engineering Data Analysis," *IEEE Trans. Software Eng.*, vol. 18, no. 11, pp. 931-942, Nov. 1992.

[2] L.C. Briand, V.R. Basili, and C.J. Hetmanski, "Developing Interpretable Models with Optimized Set Reduction for Identifying High-Risk Software Components," *IEEE Trans. Software Eng.*, vol. 19, no. 11, pp. 1028-1044, Nov. 1993.

[3] L.C. Briand, K. El-Emam, and I. Wieczorek, "A Case Study in Productivity Benchmarking: Methods and Lessons Learned," *Proc. Ninth European Software Control and Metrics Conf. (ESCOM)*, pp. 4-14, 1998.

[4] L.C. Briand, K. El-Emam, and I. Wieczorek, "Explaining the Cost of European Space and Military Projects," *Proc. 21st Int'l Conf. Software Eng. (ICSE 21)*, pp. 303-312, 1999.

[5] L.C. Briand, K. El-Emam, K. Maxwell, D. Surmann, and I. Wieczorek, "An Assessment and Comparison of Common Cost Software Project Estimation Methods," *Proc. 21st Int'l Conf. Software Eng. (ICSE 21)*, pp. 313-322, 1999.

[6] L.C. Briand, T. Langley, and I. Wieczorek, "A Replicated Assessment and Comparison of Common Software Cost Modeling Techniques," *Proc. Int'l Conf. Software Eng. (ICSE 22)*, pp. 377-386, 2000.

[7] L.C. Briand and I. Wieczorek, "Resource Modeling in Software Engineering," *Encyclopedia of Software Eng.*, 2001.

[8] E.G. Carmines and R.A. Zeller, *Reliability and Validity Assessment.* Sage Univ. papers, 1979.

[9] The COCOMO II Suite, http://sunset.usc.edu/research/cocomosuite/index.html, 2004.

[10] S.D. Conte, H.E. Dunsmore, and V.Y. Shen, *Software Engineering Metrics and Models.* Menlo Park, Calif.: Benjamin/Cummings, 1986.

[11] J.J. Dolado, "On the Problem of the Software Cost Function," *Information Software Technology,* vol. 43, no. 1, pp. 61-72, 2001.

[12] B. Efron and G. Gong, "A Leisurely Look at the Bootstrap, the Jackknife, and Cross-Validation," *The Am. Statistician,* vol. 37, no. 1, pp. 36-48, Feb. 1983.

[13] *Encyclopedia of Statistical Sciences,* S. Kotz et al. eds. Wiley, 1982-1998.

[14] T. Foss, I. Myrtveit, and E. Stensrud, "A Comparison of LAD and OLS Regression for Effort Prediction of Software Projects," *Proc. 12th European Software Control and Metrics Conf. (ESCOM 2001),* pp. 9-15, 2001.

[15] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A Simulation Study of the Model Evaluation Criterion MMRE," *IEEE Trans. Software Eng.,* vol. 29, no. 11, pp. 985-995, Nov. 2003.

[16] A.R. Gray and S.G. MacDonell, "Software Metrics Data Analysis —Exploring the Relative Performance of Some Commonly Used Modeling Techniques," *Empirical Software Eng.,* vol. 4, pp. 297-316, 1999.

[17] R. Jeffery and F. Walkerden, "Analogy, Regression and Other Methods for Estimating Effort and Software Quality Attributes," *Proc. European Conf. Optimising Software Development and Maintenance (ESCOM '99),* pp. 37-46, 1999.

[18] R. Jeffery, M. Ruhe, and I. Wieczorek, "Using Public Domain Metrics to Estimate Software Development Effort," *Proc. METRICS 2001 Conf.,* pp. 16-27, 2001.

[19] M. Jørgensen, "Experience with the Accuracy of Software Maintenance Task Effort Prediction Models," *IEEE Trans. Software Eng.,* vol. 21, no. 8, pp. 674-681, Aug. 1995.

[20] B.A. Kitchenham, "A Procedure for Analyzing Unbalanced Datasets," *IEEE Trans. Software Eng.,* vol. 24, no. 4, pp. 278-301, Apr. 1998.

[21] B.A. Kitchenham and K. Kansala, "Inter-Item Correlations among Function Points," *Proc. First METRICS Conf.,* pp. 11-14, 1993.

[22] B.A. Kitchenham, S.G. MacDonell, L. Pickard, and M.J. Shepperd, "What Accuracy Statistics Really Measure," *IEE Proc. Software Eng.,* vol. 148, pp. 81-85, 2001.

[23] B.A. Kitchenham, "The Question of Scale Economies in Software—Why Cannot Researchers Agree?" *Information and Software Technology,* vol. 44, no. 1, pp. 13-24, 2002.

[24] J. Kuha, "Model Assessment and Model Choice: An Annotated Bibliography," http://www.stat.psu.edu/jkuha/msbib/biblio.html, 2004.

[25] C. Mair, G. Kadoda, M. Lefley, K. Phalp, C. Schofield, M. Shepperd, and S. Webster, "An Investigation of Machine Learning Based Prediction Systems," *J. Systems Software,* vol. 53, pp. 23-29, 2000.

[26] D. Michie, D.J. Spiegelhalter, and C.C. Taylor, *Machine Learning, Neural and Statistical Classification.* J Campbell, ed. Sussex, U.K.: Ellis Horwood, 1994.

[27] Y. Miyazaki, M. Terakado, K. Ozaki, and H. Nozaki, "Robust Regression for Developing Software Estimation Models," *J. Systems and Software,* vol. 27, pp. 3-16, 1994.

[28] T. Mukhopadhyay, S.S. Vicinanza, and M.J. Prietula, "Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation," *MIS Quarterly,* pp. 155-171, June 1992.

[29] I. Myrtveit and E. Stensrud, "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models," *IEEE Trans. Software Eng.,* vol. 25, no. 4, pp. 510-525, Apr. 1999.

[30] P. Nesi and T. Querci, "Effort Estimation and Prediction for Object-Oriented Systems," *J. Systems and Software,* vol. 42, pp. 89-102, 1998.

[31] J.C. Nunnally and I.H. Bernste, *Psychometric Theory,* third ed. McGraw-Hill, 1994.

[32] B. Samson, D. Ellison, and P. Dugard, "Software Cost Estimation Using and Albus Perceptron (CMAC)," *Information and Software Technology,* vol. 39, pp. 55-60, 1997.

[33] M.J. Shepperd and C. Schofield, "Estimating Software Project Effort Using Analogies," *IEEE Trans. Software Eng.,* vol. 23, no. 12, pp. 736-743, Dec. 1997.

[34] M.J. Shepperd and M. Cartwright, "Predicting with Sparse Data," *IEEE Trans. Software Eng.,* vol. 27, no. 11, pp. 987-998, Nov. 2001.

[35] R. Srinivasan and D. Fisher, "Machine Learning Approaches to Estimating Software Development Effort," *IEEE Trans. Software Eng.,* vol. 21, no. 2, pp. 126-137, Feb. 1995.

[36] E. Stensrud, T. Foss, B. Kitchenham, and I. Myrtveit, "A Further Empirical Investigation of the Relationship between MRE and Project Size," *Empirical Software Eng.,* vol. 8, no. 2, pp. 139-161, 2003.

[37] E. Stensrud and I. Myrtveit, "Identifying High Performance ERP Projects," *IEEE Trans. Software Eng.,* vol. 29, no. 5, pp. 398-416, May 2003.

[38] E. Stensrud and I. Myrtveit, "Human Performance Estimating with Analogy and Regression Models: An Empirical Validation," *Proc. METRICS'98 Conf.,* pp. 205-213, 1998.

[39] K. Strike, K. El-Emam, and N. Madhavji, "Software Cost Estimation with Incomplete Data," *IEEE Trans. Software Eng.,* vol. 27, no. 10, pp. 890-908, Oct. 2001.

[40] S.S. Vicinanza, T. Mukhopadhyay, and M.J. Prietula, "Software Effort Estimation: An Exploratory Study of Expert Performance," *IS Research,* vol. 2, no. 4, pp. 243-262, 1991.

[41] F. Walkerden and R. Jeffery, "An Empirical Study of Analogy-Based Software Effort Estimation," *Empirical Software Eng.,* vol. 4, no. 2, pp. 135-158, 1999.

[42] C. Mair and M.J. Shepperd, "Making Software Cost Data Available for Meta-Analysis," *Proc. Conf. Empirical Assessment in Software Eng. (EASE 2004),* May 2004.

[43] L. Pickard, B. Kitchenham, and S. Linkman, "An Investigation of Analysis Techniques for Software Datasets," *Proc. METRICS 99 Conf.,* pp. 130-142, 1999.

[44] M.J. Shepperd and G. Kadoda, "Comparing Software Prediction Techniques Using Simulation," *IEEE Trans. Software Eng.,* vol. 27, no. 11, pp. 1014-1022, Nov. 2001.

Ingunn Myrtveit received the MS degree in management from the Norwegian School of Management in 1985 and the PhD degree in economics from the Norwegian School of Economics and Business Administration in 1995. She is an associate professor in management accounting and software economics at the Norwegian School of Management. She has also been a senior manager at Accenture's World Headquarters R&D Center in Chicago.

Erik Stensrud received the MS degree in physics from the Norwegian Institute of Technology in 1982, the MS degree in petroleum economics from the Institut Francais du Petrole in 1984, and the PhD degree in software engineering from the University of Oslo in 2000. He is an associate professor at Buskerud University College, Norway, and he also consults on software engineering. Before that, he managed software projects in Accenture, Ernst and Young, and other companies. He is a member of the IEEE and the IEEE Computer Society.

Martin Shepperd received the PhD degree in computer science from the Open University, United Kingdom, in 1991. He is professor of software engineering at Bournemouth University, United Kingdom. He has published more than 90 refereed papers and three books in the area of empirical software engineering, machine learning and statistics. He is editor of the journal *Information & Software Technology* and was an associate editor of *IEEE Transactions on Software Engineering* (2000-2004).