Discussion Paper
3/2002

A Simulation Study of the Model Evaluation Criterion MMRE

Tron Foss (*), Erik Stensrud (*), Barbara Kitchenham (**), Ingunn Myrtveit (*)
(*) The Norwegian School of Management
(**) Keele University
{tron.foss, erik.stensrud, ingunn.myrtveit}@bi.no
barbara@cs.keele.ac.uk

Abstract

The Mean Magnitude of Relative Error, MMRE, is probably the most widely used evaluation criterion for assessing the performance of competing software prediction models. It seems obvious that the purpose of MMRE is to assist us to select the best model. In this paper, we have performed a simulation study demonstrating that MMRE does not *select the best model. The consequences are dramatic for a vast body of knowledge in software engineering. The implications of this finding are that the results and conclusions on prediction models over the past 15-25 years are unreliable and may have misled the entire software engineering discipline. We therefore strongly recommend not using MMRE to evaluate and compare prediction models. Instead, we recommend using a combination of theoretical justification of the models we propose together with other metrics proposed in this paper.*

Index terms

Mean magnitude of relative error, simulation, regression analysis, prediction models, software engineering.

1. Introduction

Software cost estimates and defect rate estimates are important deliverables of software projects. As a consequence, researchers have proposed and evaluated a plethora of prediction systems. There are a number of empirical studies including studies on *generic* model-based methods [12][14][18][28][30][38] as well as on *specific* model-based methods. The latter methods include CART (Classification and Regression Trees) [4][5][7][35][50][48], OSR (Optimized Set Reduction) [2][3][29], Stepwise ANOVA (Analysis of Variance) [35], OLS (Ordinary Least Squares) regression (more than 30 studies, see [10] for an account), Robust regression [23] [24] [26] [40] [44] [45], Composite Estimation models (like COBRA) [4], Analogy-based models [19] [27] [41] [42] [49] [51] [55] [48] and, finally, artificial neural network-based models [47] [50]. (We have adopted the classification scheme proposed in the *Encyclopedia of Software Engineering* [10] except possibly for neural networks.)

The most widely used evaluation criterion to assess the performance of software prediction models is the Mean Magnitude of Relative Error (MMRE) [10]. This is usually computed following standard evaluation processes such as cross-validation [6]. Conte et al. [13] consider $MMRE \leq 0.25$ as acceptable for effort prediction models. There exist a number of alleged reasons to use MMRE. The claimed advantages include the following:

1. Comparisons can be made across data sets [8][48].
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% whatever unit is used.
3. Comparisons can be made across prediction model types (on the same data set) [13] e.g. compare a CART model with an artificial neural net model.
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 actual size (effort) of the project [54]. For example, a 1 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% for both projects).

The only claim that is true is claim 2. Claims 1 and 4 are untrue. We have strong evidence that MRE is a decreasing function of effort [53]. That means that MRE is not independent of scale, thus refuting claim 4. Furthermore, it follows that a measure that decreases with project size cannot be used to compare MMRE across data sets, thus refuting claim 1 (unless the projects in the two data sets are of similar size). This can be understood by considering two data sets A and B. Suppose that data set A consists of small projects whereas B consists of large projects. Given everything else equal, MMRE(B) will very likely be smaller than MMRE(A). Therefore, a prediction model evaluated on data set B will be identified as better than a competing model evaluated on data set A.

In this paper, we challenge and present evidence to refute claim 3. We show that claim 3 is untrue because when MMRE is used for comparison of two prediction models, we are at great risk of identifying a false model as better than a true model. We perform a simulation study that demonstrates that MMRE does not tend to prefer the true model to any other model. On the contrary, MMRE will tend to prefer a model that *underestimates* to the true model. That is, MMRE will be lower (i.e. "better") for a fluke model than for a correct model even when the correct model happens to be the *true* model. Miazaki et al. [40] pointed this out a while ago, but neither they nor anybody else have seriously questioned the far reaching implications of this fact. The very serious implication of this fact is that MMRE is an *unreliable* measure that is *not* suitable for evaluating which prediction model is best.

The consequences are dramatic for a vast body of knowledge in software engineering in that the results and conclusions on prediction models over the past 15-25 years are unreliable and thus, have not added as much value as we have believed to this important branch of software engineering. Actually, our situation is much worse. The use of MMRE may have misled us and actually hindered progress. MMRE should therefore not be used to compare and evaluate prediction models (unless one can present good reasons for using it other than the common justification that "It is widely used in SE").

Actually, we are not aware of MMRE being used to evaluate prediction models (like regression analysis, CART, etc.) in disciplines other than computer science and software engineering. We are, however, aware of its use in time series analysis. See for example Makridakis et al. [37].

2. Illustrating the problem

To illustrate the problem of MMRE, let us consider two prediction models A and B, respectively. If MMRE of model B is significantly lower than MMRE of model A, one would conclude that model B is better than model A (B is "more accurate" than A in current software engineering terminology). MRE is the basic metric in MMRE and is defined as follows [13] (where y = actual, $\hat{y}$ = prediction).

$$MRE = \frac{\left| y - \hat{y} \right|}{y}$$

To be able to draw the correct conclusion with regard to whether model A or model B is best, it is crucial that the model evaluation metric *selects the model that is closest to the true model most of the time*. (The true, or population, model is the model we would obtain if our data set comprised all the relevant past and future data points, e.g. the population of all past and future software projects that are similar to the project to be predicted). This seems like a reasonable, common sense requirement of an evaluation criterion. Otherwise, the evaluation criterion may lead you to wrongly choose model B when model A ought to have been chosen.

Consider the two models A and B in Figure 1. Model A is fitted to the data by OLS log-linear regression. (The data is the Finnish data set, see [32] for details on the data). Next, assume that model B is fitted by some other method (Say, the novel method "SuperX" which we have recently developed and which we believe holds great promise.) We observe that model B has the same slope as A but a different intercept than A. (Model B Constant=0.7, Model A Constant=1.7, see Table 1and Table 2, respectively). Thus, the intercept of model B is *underestimated* (compared with A).

By visual examination, A seems to represent the central tendency of the data reasonably well, at least far better than model B. Since A has been fitted using OLS, the estimates from A equal the expected value (or mean). Also, model A is a good regression model in terms of the commonly used criteria SE (standard error) and R^2 (See Table 3). Both coefficients of model A are significant ($p<0.10$).

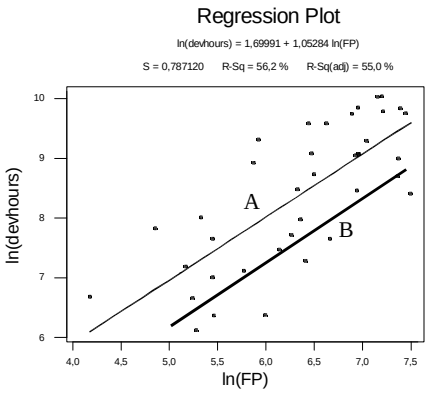

Figure 1. Two regression models A and B

Nevertheless, in terms of MMRE, model B is preferred to model A (Table 3). (MRE may be calculated using the formula derived in Appendix 1.) As a consequence, we would be misled by MMRE to identify model B as better (or more "accurate") than model A. Obviously, this is a serious flaw of the evaluation criterion MMRE when it is used to select between competing prediction systems. As an evaluation criterion, it therefore clearly does not comply with common sense (nor with statistical science) with regard to identifying which model is the better one.

Table 1. Model A (fitted by OLS regression)

Predictor	Coef	SE Coef	T	P
Constant	1.7	0.99	1.7	0.096
ln(FP)	1.05	0.16	6.8	0.000

Table 2. Model B

Predictor	Coef
Constant	0.7
ln(FP)	1.05

Table 3. MMRE, SE and R^2 of models A and B

Model	N	MMRE	SE	R^2
A	38	0.79	0.79	0.56
B	38	0.59	N/A	N/A

In this paper, we have performed a simulation study demonstrating that MMRE unfortunately is an unreliable measure that indeed leads us to draw wrong conclusions with respect to the question "which prediction model is to be preferred".

3. Related work

There are two categories of work that may be considered related to this study: studies on evaluation metrics as well as simulation studies.

Evaluation of the evaluation metrics themselves seems to have received little attention since Conte, Dunsmore and Chen [13] publicised MMRE and other measures. The only related work we are aware of is Miazaki et al. [40], Kitchenham et al. [31], Foss et al. [20], and Stensrud et al. [53]. Miazaki et al. acknowledge that MMRE is lower for models that underestimate, but they have not demonstrated the implications of this fact. Kitchenham et al. take a different approach. They attempt to understand what MMRE (and other measures) really measure. Stensrud et al. found that there is a negative relationship between MRE and effort implying that MMRE is not reliable when comparing values across different data sets. None of them have investigated the reliability of MMRE when used as a criterion to select between *competing* prediction systems.

Regarding simulation studies in software engineering, several papers have recently used simulation in software engineering: Rosenberg [46], El Emam [22], Briand and Pfahl [9], Angelis and Stamelos [1], Strike et al. [54], Shepperd and Kadoda [48], and Pickard et al. [45]. Therefore, simulation is becoming an accepted research method in software engineering.

4. Alternative evaluation metrics

In this section, we present other potential evaluation metrics that we evaluate in this study.

Another measure akin to MRE, the magnitude of error relative to the *estimate* (MER), has been proposed by Kitchenham et al. [31]. Intuitively, it seems preferable to MRE since it measures the error relative to the estimate. MER is a measure of the dispersion of the variable $y/\hat{y}$ if the mean of $y/\hat{y}$ is approximately 1. They further pointed out that researchers should look at the full distribution of the variable and not just the dispersion. MER is defined as

$$MER = \frac{\left| y - \hat{y} \right|}{\hat{y}}$$

We use the notation MMER to denote the mean MER.

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

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

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) rather than relative to the y value (effort) to avoid one of the problems with MMRE. One of MMRE's problems is that small actuals (small y's) will have a (too) large influence on the mean MRE since a number divided by a small number tends to be a large number. In addition, we have shown that MRE is not independent of size. Rather, it decreases with size [53]. Contrary to MRE, SD increases with size because software project data sets are often heteroscedastic. As opposed to MRE and SD, RSD is almost independent of size.

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 DMR data set (Table 4) 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. Since we apply a log-linear transformation to the DMR data set, LSD potentially ought to be a more reliable measure than SD. (The reason for the $-s^2/2$ term will become clearer in section 5.2. See also Appendix 2.)

Finally, we evaluate the mean of the balanced relative error (BRE) and the inverted balanced relative error (IBRE) proposed by Miazaki et al. [40].

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

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

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

$$IBRE = \frac{\left(\hat{y} - y\right)}{\hat{y}}, \quad \hat{y} - y \geq 0$$

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

5. Simulation Method

Since the 1950's, various computer simulation techniques have become increasingly important research tools across a wide range of sciences. Software packages based on these techniques are also widely used in more applied fields such as engineering, finance, or environmental management, often in connection with computerised 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. It should be obvious that we obtain more information and can draw more reliable conclusions based on a distribution than based on a single value.

Another advantage of simulation is that we know the true answer. For example, we may study missing data techniques by removing data from a complete (simulated) data set in a controlled manner and study the effects of the missing data techniques by comparing results with the true, complete data set. Strike et al. [54] have used this approach.

Still another advantage with simulation is that it enables the studying of phenomena that are too complex to describe and solve analytically, e.g. the behaviour of oil reservoirs.

In this study, we use simulation (combined with regression models) to investigate MMRE and alternative evaluation metrics. Using simulation, we demonstrate that, in many cases, MMRE likely will fail to select the best prediction model among competing models. The simulation study therefore is a more conclusive demonstration of the conjecture stated in sections 1 and 2 and illustrated in Figure 1.

5.1 Data "template"

It is important that a simulation method for generating data generates data that are as close as possible to actual, and representative, software data sets. In this study, we have used the Desharnais (DMR) data set as a model for our simulation model. The Desharnais data set exhibits properties that are representative of other data sets of software projects with respect to (non-)linearity and heteroscedasticity. (Several software data sets are presented in e.g. [53]). In addition, it is a reasonably large sample. (It contains 81 projects, see Table 4). For more details on DMR, see [15]. The DMR data come from a single company but have used three different language types. (For some kind of analysis it may therefore require partitioning based on the language type. For this analysis it is however unnecessary to partition the data into more homogeneous subsets since it is only used as model for generating a simulation model). The projects span from 1116 to 23940 workhours.

Overall, we believe that a simulation model approximating the DMR data set should be close to reality and representative of software project data sets. (On the other hand, data sets of defect rates may exhibit different characteristics from those of project effort. The simulation model on this study is therefore not necessarily a good approximation of the former data sets.)

Table 4. Descriptive statistics for DMR data set

Variable	N	Mean	Median	StDev	Min	Max
FP Adj	81	289	255	186	62	1116
Effort	81	5046	3647	4419	546	23940

5.2 Model specification

It is common, and a reasonable starting point, to assume a linear model of the following form. (Recent research, using genetic algorithms as a flexible method of model fitting, did not find any significant deviations from a linear model [16].)

$$eff = a + b \cdot FP + u \quad (1)$$

If we apply model (1) to the DMR data set [15], we obtain the following OLS linear regression model

$$eff = -40 + 17.6 \cdot FP \quad (2)$$

Table 5. The linear model (2)

Predictor	Coef	SE Coef	T	P
Constant	-40	620	-0.06	0.949
FP	17.6	1.8	9.73	0.000

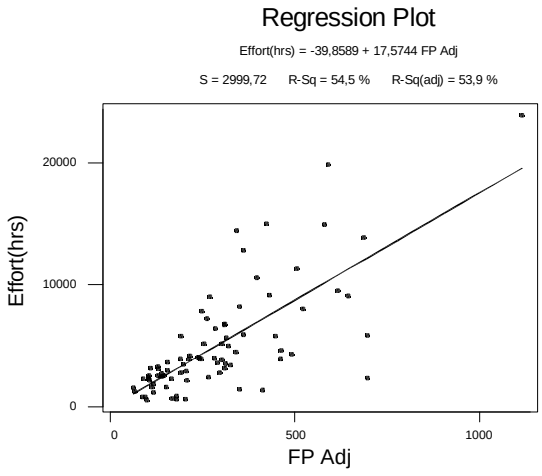

Figure 2. DMR data set: FP versus Effort

On closer inspection of the DMR data set in Figure 2, we observe that it seems reasonably linear but exhibits a pronounced heteroscedasticity (increasing variance). OLS regression

analysis assumes that the data are homoscedastic (equal variance). Model (2) is therefore not sufficiently correct. We therefore 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

$$eff = e^{a}\,(FP)^{b} \cdot I \quad (3)$$

where I is lognormal with mean equal to 1. Thus, $I = e^{u}$ with u normally distributed. It has been proved that if $Var(u) = s^{2}$ and $E(u) = -\frac{s^{2}}{2}$, then $E(I) = E(e^{u}) = 1$ ([17], vol. 5, p. 134).

Stating (3) in the form

$$eff = e^{a} \cdot (FP)^{b} \cdot e^{u} \quad (3b)$$

and taking the logarithm of equation (3b), we obtain

$$\ln(eff) = a + b\ln(FP) + u \quad (4)$$

A plot of the transformed data when applying model (4) is presented in Figure 3. Inspecting the plot, the data exhibit a reasonable homoscedasticity and linearity.

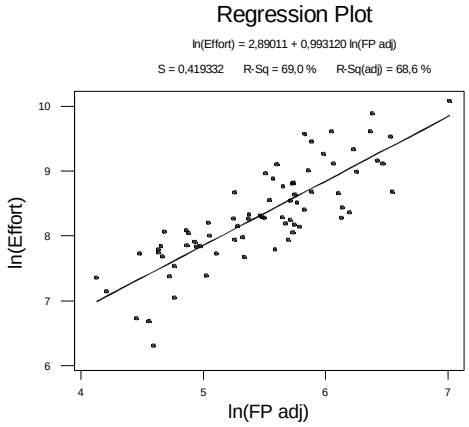

Figure 3. DMR data set: ln(FP) versus ln(Effort)

Applying model (4) to the DMR data set, we get the following OLS regression model (5) or (6) or in table form in Table 6.

$$\ln(eff) = 3.03 + 0.943 \cdot \ln(FP)\text{, SD=0.6000} \quad (5)$$

Backtransforming (5), we get

$$eff = e^{3.03} \cdot (FP)^{0.943} = 20.6972 \cdot (FP)^{0.943} \quad (6)$$

Table 6. The log-linear model (5) (6)

Predictor	Coef	SE Coef	T	P
Constant	3.03	0.59	5.11	0.000
ln(FP)	0.943	0.11	8.77	0.000

Comparing (1) with (3), we can state that in (1) we believe in a linear relationship between effort and FP whereas we in (3) we believe in an exponential relationship between these two variables. We observe, however, that model (6) fitted to the DMR data set is not particularly exponential since the exponent is close to 1 (0.943 and with a standard error of 0.11, see Table 6). From this, we cannot draw any conclusions regarding returns to scale (i.e. whether to assume increasing, decreasing, like COCOMO [12], or constant returns or scale. See [43] for an 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 model (1) 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 model (1).

$$eff = a + b \cdot (FP) + (FP)u \quad (7)$$

In model (7), we assume a linear relationship between FP and effort as we do in equation (1), but unlike (1), we transform the data in an attempt to obtain a more constant variance. Equation (7) may be restated as

$$\frac{eff}{(FP)} = a \cdot \frac{1}{(FP)} + b + u \quad (8)$$

Thus, in (8), $\frac{eff}{(FP)}$ is the dependent variable and $\frac{1}{(FP)}$ is the independent variable. The OLS regression model applying (8) on the DMR data is

$$\frac{eff}{(FP)} = 16.9 + 127 \cdot \frac{1}{(FP)} \quad (9)$$

The standard deviation of the residual in (9) is 8.4. Model (9) may alternatively be stated as

$$eff = 127 + 16.9 \cdot (FP) \quad (10)$$

Table 7. Model (9) (10)

Predictor	Coef	SE Coef	T	P
Constant	127	298	0.43	0.671
FP	16.9	1.8	9.6	0.000

A plot of model (9) is provided in Figure 4. In model (9), the residual, u, seems less heteroscedastic than the residual of model (1). Unfortunately, it is not perfectly homoscedastic either. This is observed by comparing the plots of Figure 2 and Figure 4. Compared with the log-linear model (3), model (9) seems no better. In addition, comparing Table 6 and Table 7, we observe that the intercept of the log-linear model (3) has a higher t-value than the intercept of model (9), and that the slope coefficients have similar t-values. Overall, the log-linear model seems the best choice.

In the simulation study that follows, we have therefore opted for model (3), the log-linear model. The log-linear model also has an additional benefit compared with the two other models in that it corrects heteroscedasticity as well as forces the line through the origin. (That is, when zero FP is delivered, zero effort is expended.)

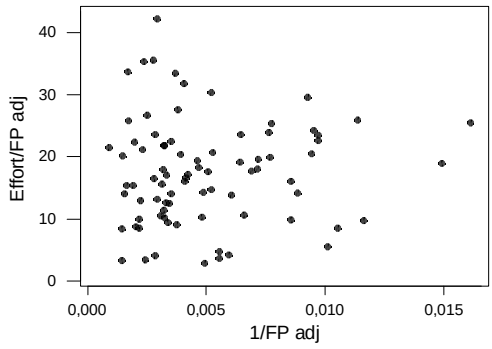

Figure 4. DMR data set: 1/FP versus Effort/FP

We find it useful to have performed and reported this explorative exercise because there, unfortunately, is no strong theory to guide us in software engineering. We have therefore chosen (3) based on empirical evidence of the DMR data set. This empirical evidence weakly suggests a multiplicative model rather than an additive model.

5.3 Simulation model and procedure

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 (3) and with coefficients and standard deviation (ó) identical to model (5):

$$\ln(eff) = 3.03 + 0.943 \cdot \ln(FP) + u \,,$$

$$\text{ó} = 0.6000 \quad (11)$$

Model (11), i.e. the regression model plus 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 the stochastics (e.g. variables not captured in our model). Some projects use more effort, and some less effort, than the expected effort based on FP counts. u is normal with mean equal to $-\text{ó}^2/2$ and variance equal to ó^2 ([17] vol. 5, p. 134). This simulation model generates data sets with characteristics similar to the DMR data set (and presumably similar to other software project data sets).

If the population is given by model (11), we may simulate sampling from this population. Let us assume that we have conducted 30 projects of different size, say, $FP_i = 50 \cdot i$, i=1,2, ….30. Then, the smallest project will have FP=50 and the largest in the sample will have FP=1500. (This is close to the span of the DMR data set, see Table 4. Only, we draw 30 rather than 81 observations. This because many software data sets have around 30 observations.) For each project i, we draw a random number u_i from a normal distribution with mean equal to –0.18 ($-\text{ó}^2/2$) and standard deviation (ó) equal to 0.6000. (This is standard functionality in statistical packages e.g. Minitab [39]: Menu tree: Calc.random.normal). Thus, we loop from i=1,2, ….30 and generate FP_i each time using the formula $FP_i = 50 \cdot i$. Next, we compute $Effort_i$ using model (11) with FP_i as input as well as input of an u_i value. This procedure gives us a data set of 30 observations with characteristics similar to the DMR data set. In this way, we may create as many samples of 30 observations as we wish. For the simulation study, we created 1000 samples of 30 observations each. We observe that the simulation procedure is a reverse procedure of the ordinary regression analysis procedure. (In regression analysis we start with the observations and estimate the

regression model whereas in this simulation we start with a know population regression model and create a number of observations).

6. Simulating competing models

Let us assume that we have created a variety of prediction models based on a variety of procedures and based on samples drawn from the population described by the simulation model (11). (These procedures could have been classification and regression trees, estimation by analogy, neural nets or some novel method X.) For simplicity, assume that we have created four different log-linear models based on different samples and different methods X, Y and Z as follows

$$eff = e^{2.50} \cdot (FP)^{0.943} \qquad (12)$$

$$eff = e^{3.03} \cdot (FP)^{0.920} \qquad (13)$$

$$eff = e^{3.50} \cdot (FP)^{0.943} \qquad (14)$$

$$eff = e^{3.03} \cdot (FP)^{0.970} \qquad (15)$$

We observe that (12) has a smaller intercept than the true model (6) but equal β. That is, model (12) underestimates. We should therefore expect it to be considered as superior to the true model terms of MMRE since MMRE favours models that underestimate. Model (13) has intercept equal to the true model whereas the slope is smaller than the slope of the true model. Thus, model (13) also underestimates, and we should therefore expect also this model to be identified as superior to the true model in terms of MMRE. Model (14) has a greater intercept than the true model and equal slope. We should therefore expect this model to be identified as inferior to the true model in terms of MMRE. Model (15) has a greater slope than the true model and equal intercept. We should therefore expect also this model to be identified as inferior to the true model in terms of MMRE.

7. Research procedure

The research procedure simply consisted of creating 1000 samples based on the simulation model (11). Thereafter, we computed MMRE, MdMRE, MMER, SD, RSD, LSD, MBRE and MIBRE values for the five models (true, 12, 13, 14 and 15) on all the 1000 samples. We compared models pairwise with the true model. For each pairwise comparison, we counted the number of times each model obtained the best MMRE, MdMRE, MMER, SD, RSD LSD,

MBRE and MIBRE values, respectively. For example, suppose we compare the true model and model (12) on 1000 samples by computing MMRE for each model on each sample. Suppose the results are that (12) obtains the lowest (best) MMRE on 900 of the samples and the true model obtains the lowest MMRE on the remaining 100 samples. Then, we report 900 for model (12) and 100 for the true model. This should be interpreted as follows: When MMRE is used to select between competing prediction systems, the estimated probability that we select (12) is 0.9 whereas the estimated probability of selecting the true model is 0.1.

MRE may be computed using the formula (A3) in Appendix, or it may be computed directly using the multiplicative form of the function. Similar formulas may be used to compute the other measures (not reported).

ResultsTable 8 and Table 9, we have reported the number of times each model obtains the best score (i.e. lowest MMRE, etc.). We reiterate that a sensible evaluation criterion ought to have a high probability of identifying the true model as best.

Focussing on MMRE in the four tables, we observe that MMRE identifies models that underestimate as superior to the true model and to models that overestimate. MdMRE exhibits a pattern similar to MMRE, however less clear-cut. MMER identifies the true model as best in three out of four cases. It is therefore not sufficiently consistent in identifying the true model as best. SD identifies the true model as best in all cases and is therefore consistent. RSD also identifies the true model as better than the four other models and is therefore consistent, too. Compared with SD, RSD also identifies the true model as best with a higher probability than SD. LSD, too, consistently identifies the true model as best with a reasonably high probability (>0.7). MBRE and MIBRE perform similarly to MMER. They both identify the true model as best in three out of four cases.

To conclude, MMRE, MdMRE, MMER, MBRE and MIBRE are substantially more unreliable as evaluation criteria than SD, RSD and LSD. All of the latter three criteria are consistent, and RSD and LSD seem slightly better than SD. It seems obvious that a good selection criterion ought to prefer the true model, the truth, most of the time. (Ideally, it ought to prefer the truth all of the time, but this is, of course, infeasible for evaluation criteria based on statistical methods.)

Table 8. Results of true model vs. model (14)

	True	Model (14)
Best MMRE	1000	0
Best MdMRE	996	4
Best MMER	590	410
Best SD	943	57
Best RSD	981	19
Best LSD	964	34
Best MBRE	1000	0
Best MIBRE	998	2

Table 9. Results of true model vs. model (15)

	True	Model (15)
Best MMRE	1000	0
Best MdMRE	870	130
Best MMER	260	740
Best SD	734	266
Best RSD	765	235
Best LSD	746	254
Best MBRE	988	12
Best MIBRE	924	76

8. Discussion

In this section, we discuss the simulation study and the proposed evaluation measures. We also discuss alternative ways to evaluate prediction models.

8.1 Simulation study

Regarding the simulation study, we generated 1000 samples. This is common in simulation studies. Each sample contained 30 observations. This is a sample size that reasonably well reflects typical software project sample sizes. See e.g. [53]. We did not perform significance tests of the difference in MMRE (and of the other criteria) values. We feel that this was not necessary to prove our point, namely that MMRE, MdMRE and MMER are unreliable and inferior to a simple statistic like SD. Our point is just that, using these metrics, there is a high probability of preferring a bad model to a good model.

We also compared two models at a time. In most SE studies, a new, proposed model is compared against a baseline model like OLS regression. Therefore, our pairwise comparisons reflect the majority of SE studies. There are, however, cases where more than two models are compared simultaneously. This would require a different set-up of the simulation study.

8.2 Evaluation metrics

As for the evaluation measures proposed in this paper, we observe the following.

RSD. 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:

$$y = a + bx + xe \quad \text{(RSD1)}$$

where e is normally distributed: $E(e) = 0$ and $\text{var}(e) = s^2$. This model will generate data where the variance increases with x. Dividing (RSD1) by x gives:

$$\frac{y}{x} = a \cdot \frac{1}{x} + b + e \quad \text{(RSD2)}$$

The error term in this regression model (RSD2), e, is normal: $E(e) = 0$ and $\text{var}(e) = s^2$. OLS will, therefore, be an *efficient* estimating method. Let $\hat{a}$ and $\hat{b}$ be estimates of a and b. Our prognosis (sample prediction model) for effort is then:

$$\hat{y} = \hat{a} + \hat{b}x$$

But then:

$$\frac{\hat{y}}{x} = \hat{a}\frac{1}{x} + \hat{b}$$

$\frac{y}{x} - \frac{\hat{y}}{x}$ is, therefore, and estimate, e, of the error term e. 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 e in the regression equation. Thus, RSD is a relevant measure of how good the prediction model is.

Of course, it can be objected that the model we simulate is exponential whereas RSD is based on an additive model. However, these two models are very close to each other (The exponent is 0.943 which is very close to 1). Therefore, RSD is an appropriate measure in this single-predictor case. The simulation study also demonstrates that RSD is well behaved, but we do not conjecture that it is the ideal metric under all circumstances.

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

Finally, we would like to add that we have used RSD to compare models on the same data set. We believe it generally is inappropriate to compare prediction models across different data sets, in particular if the projects in the data sets are of different size, say, one data set consists of small projects and another data set consists of large projects.

LSD is useful for comparing exponential models because the residual error will be independent of size (i.e. constant expected value) on the log scale. As long as we log transform the data sets, LSD is appropriate both to select between competing models as well as to assess absolute dispersion (on the log scale). LSD may, however, be inappropriate if a linear scale is assumed.

MMER does not perform particularly well in this study. In theory, however, MMER might perform well for data sets exhibiting a more pronounced heteroscedasticity than the DMR data. (We have so far, however, not identified any software project data sets with more heteroscedasticity than the DMR data set. See [53].) Like SD, and unlike MMRE, it favours models with a good fit to the data, and unlike MMRE, it is not sensitive to small actuals. Therefore, given the hypothetical choice between MMRE and MMER, we argue that MMER is to be preferred.

Among all the measures evaluated in this study**, MMRE** is probably the *worst choice*. It is unreliable both on normal scales as well as on log scales. (Because MRE varies with size on both scales.) It clearly favours models that underestimate, and it is extremely sensitive to small actuals.

As an aside, we also observe that MMRE does not take the number of observations, N, into account. Common-sense dictates that we generally should have more confidence in a

model based on 100 observations than a model based on two observations. The latter model would likely obtain a better MMRE but would, nevertheless, inspire less confidence, in general.

9. Conclusion

MMRE has for many years been, and still is, the de facto criterion to select between competing prediction models in software engineering. It is therefore important that we can rely on MMRE in selecting the best model among the choices. What then is *best*?

Econometrics [25] provides us with one possible answer to this question: It proposes that the best model is the model constructed with the most *efficient* fitting method, a fitting method being, say, OLS or LAD.

(*Efficient* is a reserved word in statistics meaning *best unbiased*. That is, an efficient method provides estimators (estimates of the coefficients) with smaller variance than any competing method (therefore *best*) and a point estimate equal to the mean (therefore unbiased). In short, an efficient method results in a model closer to the truth than any competitor models.)

According to econometrics, it is the characteristics of the data set that decides which method is efficient. For example, given that the OLS assumptions are fulfilled, the OLS method will be efficient. On the other hand, if the kurtosis is high, LAD will be more efficient than OLS. Thus, we have to have theories (hypotheses) as well as data to defend our use of a particular type of fitting method (e.g. OLS, LAD) and model characteristics (e.g. linear, exponential).

In the following, we divide our conclusions in two categories, "conclusive" conclusions and speculative conclusions, respectively. The conclusive conclusions that we can draw directly from the empirical results are the following.

1. MMRE will always select a model that provides an estimate *below the mean* (obtains a low MMRE) to a model that predicts the mean (obtains a high MMRE). That is to say, MMRE will consistently choose a biased model to an unbiased model. Given that we prefer information on a precisely defined statistic as the *mean* to some non-defined, optimistic prediction somewhere between the mean and zero effort, we may strongly conclude that MMRE is completely inappropriate as a model selection criterion. Worse, it is dangerously misleading.

2. LSD is appropriate to evaluate exponential models. Given that other SE data sets exhibit characteristics similar to the DMR data set, exponential models fit well with reality.
3. RSD is also appropriate to evaluate models that fit with data similar to the DMR data, that is to say, data that are fairly linear as well as heteroscedastic. However, RSD is limited to data with a single predictor variable.
4. Although we have not shown it, we have theoretical reasons to suspect that SD is appropriate to evaluate linear models with no heteroscedasticity.
5. SD, RSD and LSD are the only criteria that consistently select the true model with a probability $p>0.5$.
6. The other metrics, MMER, MBRE and MIBRE, are unreliable, too. Although, from a theoretical perspective, we would prefer MMER to MMRE.

From conclusion 1, we may infer that the conclusions drawn in all software engineering studies based on MMRE with respect to which model is best, are of limited value, indeed. This includes tens of papers over the last 20 years or so, and it includes very recent papers. From conclusions 1-6, we may also derive a more speculative conclusion.

7. Since we observe that all the metrics (including SD, LSD and RSD) suffer from either a flaw or a limitation, we may speculate that there may probably not exist a universal, "silver bullet" metric that allows us to compare apples and oranges with the same ruler, that is to say, a metric that, in a simple manner, allows us to compare a variety of models like linear models, exponential models, higher order functions (like arbitrary function approximators), CART, OSR, estimation by analogy, CBR, neural nets, etc. Specifically, and contrary to what has been advocated in SE studies, MMRE is definitely not this sought after, universal metric.

Speculation 7 suggests that we may have to look in other directions than solely seeking single-mindedly for the ultimate model evaluation criterion to solve our problems in SE, the replacement for MMRE, so to speak. In the next section, we outline some possible directions for future research on evaluating prediction models.

10. Future directions for evaluating prediction models

It is said that it is easy to criticize but difficult to invent. As we have criticised MMRE and advised the reader not to use it, we anticipate the reader's next question is: "MMRE is bad, and MMER, RSD and LSD have limitations, too, so what am I then to do? How am I to

evaluate prediction models?" In this section we attempt to provide a partial answer acknowledging that it is not completely satisfactory. Providing a more satisfactory answer is a topic for future research now that we have given conclusive evidence of the flaws of current evaluation practices in software engineering.

To provide a partial answer, we first need to address a basic question: what is a prediction? Or rather, what do we want it to be? To us, it makes sense that the answer is: the *expected value* (the mean). Given that we accept this answer, we should proceed to evaluate prediction models *theoretically* before proceeding to empirical investigations.

First of all, we may screen prediction models into two broad categories based on their theoretical properties: those that predict the mean and those that do not, respectively. (The latter typically lack a theory and therefore provide a number without any information on how to interpret this number: is it the mean, the median, the most optimistic value, the most pessimistic, or just some none-defined quantity?) The models in the first category are further scrutinized whereas second category models may be put on the wait list until it is demonstrated that they predict the mean (or some other well-defined measure).

Next, to select among models in category one, we probably need recourse to statistical concepts like *efficiency* (See also e.g. [25], Appendix A.7 for a definition of the term efficiency). Given theoretical assumptions and/or empirical evidence of the data, we may select the model obtained with an efficient method for data with this kind of characteristics. For example, regression methods come in a large variety including OLS and LAD. LAD is more efficient than OLS on data exhibiting high kurtosis. If we, therefore, have theoretical arguments or empirical evidence to advance the hypothesis that SE project data exhibit high kurtosis, we have justified using LAD instead of OLS. Note that we would then choose LAD instead of OLS based on *a priori* assumptions and not choose it just because it happened to perform better in terms of some evaluation metric on a single data set that may be more or less representative of the population.

In a similar manner, we believe that arguments need to be advanced in favour of methods like CART, neural nets, CBR, etc. Specifically, we believe it is insufficient to justify these methods solely by empirical evidence, especially when the evaluation criterion is the flawed MMRE measure.

We also believe we have to justify the functional form of the models by *theoretical* arguments in addition to empirical studies. After all, when we use FP as predictor variable

for effort, we assume there is a relationship between FP and effort of the form: more FP implies more effort.

Therefore, a linear relationship may be a good first order approximation for many SE datasets spanning over a limited range in project effort. When we depart from a linear relationship to an exponential relationship, we must justify it by hypotheses in favour of variable (increasing or decreasing) returns to scale.

If we use higher order polynomials such as so called arbitrary function approximators, we need to provide some theoretical foundation for these, too. Specifically, we believe that it is completely insufficient to argue for using an arbitrary function approximator model providing a "serpent" like function merely by reporting a seemingly good (low) MMRE figure on a single data set.

In our view, this is what makes the difference between being a naïve empiricist and a sophisticated empiricist. The latter advances theories or hypotheses before proceeding to experiment. He does not merely observe and report in a naïve way. This attitude is in concordance with modern philosophy of science [11].

Actually, philosophy of science may guide us some more on this matter. They also state that a good hypothesis must be *falsifiable*, the more the better. Applying the criterion of falsifiability, we clearly see that a hypothesis like "There is a linear relationship between FP and effort" is a lot more falsifiable than the hypothesis "There is an arbitrary relationship between FP and effort". In fact, the latter cannot be falsified (or disproved), and it is, therefore, not a bold conjecture as the falsificationists urge us to advance.

To summarise, it seems that empirical software engineering has been conducted in primarily a naïve empiricist fashion. It seems that we ought to develop our discipline into a more sophisticated form of empiricism to improve our profession.

Our partial answer above also presupposes that we deal with a large enough amount of data to use statistical methods. (This is also implicitly assumed when using MMRE as evaluation criterion as the MMRE of one data point is meaningless.) We have stressed the need for theory and clear hypotheses in empirical software engineering. In addition, we should, of course, continue the search for better empirical evaluation criteria. The criteria we have proposed in this paper (RSD, LSD and SD) is a contribution towards this aim.

Finally, we would like to utter some opinions on how to improve the discipline of software cost estimation although, strictly speaking, it is probably beyond the scope of this paper.

We believe we generally need a stronger emphasis on expert opinion in research on cost estimation. Some research (e.g. [34] [42]) plus anecdotal evidence as well as personal experience support the belief that human judgement adds substantial value in project cost estimation.

We also believe that further improvement must come primarily by improving the data quality and quantity. We should aim our research at improving the data quality of our data sets by investigating more and better predictor variables (cost drivers) in the spirit of the COCOMO work [12] and of Stensrud [52] and by using methods for specifying software data sets in a standardised manner as proposed by Kitchenham et al. [33]. Researchers should therefore spend more time on the tedious task of gathering more and better data rather than by devising new, exotic functional forms and data fitting methods for predicting cost and defects.

Acknowledgements

This work was funded by Accenture's Research Fund in Norway.

Appendix 1. Calculation of MRE in Log-linear Regression Models

This appendix shows how the formula for calculating MRE is derived when one applies a log-linear regression model to predict effort. Let y be the actual and $\hat{y}$ be the prediction. Further, let the log-linear population model be

$$\ln y = \ln a + b \ln X + \ln u$$

Then, the sample model is

$$\ln \hat{y} = a + b \ln X$$

The residual is given by

$$residual = \ln y - \ln \hat{y}$$

which is equal to

$$residual = \ln(\frac{y}{\hat{y}})$$

This may be transformed to

$$e^{-residual} = \frac{\hat{y}}{y}$$

Thus,

$$1 - e^{-residual} = \frac{y - \hat{y}}{y} \quad \text{(A1)}$$

By definition, MRE is

$$MRE = \left| \frac{y - \hat{y}}{y} \right| \quad \text{(A2)}$$

From (A1) and (A2) we may restate MRE as

$$MRE = \left| 1 - e^{-residual} \right| \quad \text{(A3)}$$

Appendix 2. The issue of -ó²/2

In sections 5.2 and 5.3, we state that the error term of the exponential model (3), u, is normal with mean equal to $-\frac{s^2}{2}$ and equal variance s^2. Therefore, when we apply OLS to the log-transformed regression equation

$$\ln y = a + bx + u,$$

we obtain the estimator of a, $\hat{a}^*$, where

$$E\left(\hat{a}^*\right) = \left(a - \frac{s^2}{2}\right).$$

Thus, $\hat{a}^*$ is not an unbiased estimator of a. We would have to add $\frac{s^2}{2}$ to obtain an unbiased estimator, $\hat{a}$, for a (s^2 is an estimator for s^2) with $\hat{a}$ thus given as

$$\hat{a} = \hat{a}^* + \frac{s^2}{2}$$

To obtain unbiased predictions, we ought to use the unbiased estimator $\hat{a}$. However, we have not used the models to make any predictions in this paper. On the contrary, we have used the true multiplicative model (3) solely to generate data samples.

References

[1] Angelis L and I Stamelos (2000), "A Simulation Tool for Efficient Analogy Based Cost Estimation", *Empirical Software Engineering*, 5, pp. 35-68.

[2] LC Briand, VR Basili, and WM Thomas (1992), "A Pattern Recognition Approach for Software Engineering Data Analysis", *IEEE Trans. Software Eng*., vol. 18, no. 11, pp. 931-942.

[3] LC Briand, VR Basili, and CJ Hetmanski (1993), "Developing Interpretable Models with Optimized Set Reduction for Identifying High-Risk Software Components", *IEEE Trans. Software Eng.*, vol. 19, no. 11, pp. 1028-1044.

[4] LC Briand, K El-Emam, and F Bomarius (1998), "COBRA: A Hybrid Method for Software Cost Estimation, Benchmarking and Risk Assessment", *Proc. 20th International Conference on Software Engineering* (ICSE 20), IEEE Computer Society Press, Los Alamitos CA, pp. 390-399.

[5] LC Briand, K El-Emam, and I Wieczorek (1998), "A Case Study in Productivity Benchmarking: Methods and Lessons Learned", *Proc. 9th European Software Control and Metrics Conference (ESCOM)*, Shaker Publishing BV, The Netherlands pp. 4-14.

[6] LC Briand, K El-Emam, and I Wieczorek (1999), "Explaining the Cost of European Space and Military Projects", *Proc. 21st International Conference on Software Engineering* (ICSE 21), ACM Press, pp. 303-312.

[7] LC Briand, K El-Emam, K Maxwell, D Surmann, and I Wieczorek (1999), "An Assessment and Comparison of Common Cost Software Project Estimation Methods", *Proc. 21st International Conference on Software Engineering* (ICSE 21), ACM Press, pp. 313-322.

[8] LC Briand, T Langley, and I Wieczorek (2000), "A replicated Assessment and Comparison of Common Software Cost Modeling Techniques", *Proc. International Conference on Software Engineering*, (ICSE 22), ACM Press, pp. 377-386.

[9] LC Briand and D Pfahl (2000), "Using Simulation for Assessing the Real Impact of Test Coverage on Defect Coverage", *IEEE Trans. On Reliability*, vol. 49, no. 1, pp. 60-70.

[10] LC Briand and I Wieczorek, "Resource Modeling in Software Engineering", in *Encyclopedia of Software Engineering*, second edition, (Ed. J. Marciniak) Wiley, in press.

[11] AF Chalmers (1982), *What is this thing called science?*, Second edition, Open University Press, Buckingham.

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

[13] SD Conte, HE Dunsmore, and VY Shen (1986), *Software Engineering Metrics and Models*, Benjamin/ Cummings, Menlo Park CA.

[14] AME Cuelenare, MJI van Genuchten, and FJ Heemstra (1987), "Calibrating a Software Cost Estimating Model: Why and How", *Inf. and Software Tech.* vol. 29, no. 10, pp. 558-569.

[15] JM Desharnais (1989), "Analyse statistique de la productivite des projects de developpement en informatique a partir de la technique des points de fonction", Master's thesis, Universite du Quebec a Montreal.

[16] JJ Dolado (2001), "On the problem of the software cost function", *Inf. Software Technology*, 43(1), pp. 61-72.

[17] *Encyclopedia of statistical sciences*, eds-in-chief: S Kotz, NL Johnson; ass. ed: CB Read, New York, Wiley, 1982-1998.

[18] DV Ferens (1996), "Software Cost Estimation in the DoD Environment", *American Programmer*, July, pp. 28-34.

[19] GR Finnie, GE Wittig, and JM Desharnais (1997), "A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models", *Journal of Systems and Software*, vol. 39, no. 3, pp. 281-289.

[20] T Foss, I Myrtveit, and E Stensrud (2001), "MRE and heteroscedasticity", *Proc. 12th European Software Control and Metrics Conference* (*ESCOM 2001)*, Shaker Publishing BV, The Netherlands, pp. 157-164.

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

[22] K El-Emam (1998), "The Predictive Validity Criterion for Evaluating Binary Classifiers", *Proc. METRICS'98*, IEEE Computer Society Press, Los Alamitos CA, pp. 235-244.

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

[24] AR Gray, and SG MacDonell (1999), "Software Metrics Data Analysis - Exploring the Relative Performance of Some Commonly Used Modeling Techniques", *Empirical Software Engineering*, 4, pp.297-316.

[25] DN Gujarati (1995), *Basic Econometrics* – Third Edition, McGraw-Hill, New York.

[26] R Jeffery, M Ruhe, and I Wieczorek (2001), "Using Public Domain Metrics to Estimate Software Development Effort", *Proc. METRICS 2001*, IEEE Computer Society, Los Alamitos CA, pp. 16-27.

[27] R Jeffery and F Walkerden (1999), "Analogy, Regression and Other Methods for Estimating Effort and Software Quality Attributes", *Proc. ESCOM'99*, Shaker Publishing BV, The Netherlands, pp. 37-46.

[28] RL Jenson and JW Bartley (1991), "Parametric Estimation of Programming Effort: An Object-Oriented Model", *J. Sys. and Software*, 15, pp. 107-114.

[29] M Jørgensen M (1995), "Experience With the Accuracy of Software Maintenance Task Effort Prediction Models", *IEEE Trans. Software Eng.* 21, 8, pp. 674-681.

[30] CF Kemerer (1987), "An Empirical Validation of Cost Estimation Models", *Comm. ACM*, 30, 5, pp. 416-429.

[31] BA Kitchenham, SG MacDonell, LM Pickard, and MJ Shepperd (2001), "What Accuracy Statistics Really Measure", *IEE Proceedings Software*, 148(3), pp. 81-85.

[32] BA Kitchenham, and K Kansala (1993), "Inter-item correlations among function points", *Proc. First METRICS*, IEEE Computer Society Press, Los Alamitos CA, pp. 11-14.

[33] BA Kitchenham, RT Hughes, and SG Linkman (2001), "Modeling Software Measurement Data", *IEEE Trans. Software Eng*., Vol. 27, No. 9, pp. 788-804.

[34] B Kitchenham, SL Pfleeger, B McColl, and S Eagan (2001), "An Empirical Study of Maintenance and Development Estimation Accuracy", *J. Systems & Software*, (fortcoming).

[35] BA Kitchenham (1998), "A Procedure for Analyzing Unbalanced Datasets", *IEEE Trans. Software Eng.*, vol. 24, no. 4, pp. 278-301.

[36] A Koutsoyiannis (1977), *Theory of Econometrics*, second edition, MacMillan, London.

[37] S Makridakis, SC Wheelwright and RJ Hyndman (1998), *Forecasting Methods and Applications*, Third edition, John Wiley & Sons Inc.

[38] R Martin (1988), "Evaluation of Current Software Costing Tools", *ACM SIGSOFT Software Eng. Notes*, 13, 3, pp. 49-51.

[39] Minitab Statistical Software Release 13, www.minitab.com.

[40] Y Miyazaki, M Terakado, K Ozaki, and H Nozaki (1994), "Robust Regression for Developing Software Estimation Models", *Journal of Systems and Software*, Vol. 27, pp. 3-16.

[41] T Mukhopadhyay, SS Vicinanza, and MJ Prietula (1992), "Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation", *MIS Quarterly*, June, pp. 155-171.

[42] I Myrtveit, and E Stensrud (1999), "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models", *IEEE Trans. Software Eng*. 25, 4, pp. 510-525.

[43] I Myrtveit, and E Stensrud (1999), "Benchmarking COTS Projects Using Data Envelopment Analysis", *Proc. 6th International Software Metrics Symposium* (*METRICS 99*), IEEE Computer Society Press, Los Alamitos, CA, pp. 269-278.

[44] P Nesi and T Querci (1998), "Effort Estimation and Prediction for Object-Oriented Systems", *Journal of Systems and Software*, vol. 42, pp. 89-102.

[45] L Pickard, B Kitchenham, and S Linkman (1999), "An Investigation of Analysis Techniques for Software Datasets", *Proc. METRICS 99*, IEEE Computer Society, Los Alamitos CA, pp. 130-142.

[46] J Rosenberg (1997), "Some misconceptions about Lines of Code", *Proc. METRICS'97*, IEEE Computer Society Press, Los Alamitos CA, pp. 137-142.

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

[48] M Shepperd and G Kadoda (2001), "Using Simulation to Evaluate Predition Techniques", *Proc. METRICS 2001*, IEEE Computer Society, Los Alamitos CA, pp. 349-359.

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

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

[51] E Stensrud and I Myrtveit (1998), "Human Performance Estimating with Analogy and Regression Models: An Empirical Validation", *Proc. METRICS'98*, IEEE Computer Society Press, Los Alamitos CA, pp. 205-213.

[52] E Stensrud (1998), "Estimating with enhanced object points vs. function points", In *Proc. COCOMO'13*, B Boehm and R Madachy (Eds.), USC Center for Software Engineering, Los Angeles CA.

[53] E Stensrud, T Foss, B Kitchenham, and I Myrtveit, "An Empirical Validation of the Relationship Between the Magnitude of Relative Error and Project Size", To appear in *Proc. METRICS 2002*, IEEE Computer Society Press, Los Alamitos CA.

[54] K Strike, K El-Emam, and N Madhavji, "Software Cost Estimation with Incomplete Data", ERB-1071 NRC, http://wwwsel.iit.nrc.ca/~elemam/documents/1071.pdf, also to appear in *IEEE Trans. Software Eng.*

[55] F Walkerden and R Jeffery (1999), "An Empirical Study of Analogy-based Software Effort Estimation", *Empirical Software Eng.*, 4, 2, pp. 135-158.