Interpretation problems related to the use of regression models to decide on economy of scale in software development

Magne Jørgensen

Barbara Kitchenham

Abstract

Many research studies report an economy of scale in software development, i.e., an increase in productivity with increasing project size. Several software practitioners seem, on the other hand, to believe in a diseconomy of scale, i.e., a decrease in productivity with increasing project size. In this paper we argue that violations of essential regression model assumptions in the research studies to a large extent may explain this disagreement. Particularly illustrating is the finding that the use of the production function (Size = a·Effort^b), instead of the factor input model (Effort=a·Size^b), would most likely have led to the opposite result, i.e., a tendency towards reporting diseconomy of scale in the research studies. We conclude that there are good reasons to warn against the use of regression analysis parameters to investigate economies of scale and to look for other analysis methods when studying economy of scale in software development contexts.

Keywords: Software economics, economy of scale, regression analysis

1. INTRODUCTION

Economy of scale is a term commonly used in production industries to denote a reduction in cost per unit produced as the quantities of production inputs increases. One frequently used production function for the analysis of economy of scale is the Cobb-Douglas production function (Cobb and Douglas 1928). The original Cobb-Douglas production function is on the format $Y = a \cdot X_1^{\ b} \cdot X_2^{\ c}$, where X_1 is the labor input, X_2 the capital input and Y the quantity of produced output. Using the Cobb-Douglas production function, we have an economy of scale if $b + c > 1$, diseconomy of scale if $b + c < 1$, and constant return on scale if $b + c = 1$. Interestingly, in software engineering we use the reverse relation to decide on economy of scale, i.e., we study a function on the Cobb-Douglas format of the type $X = a \cdot Y^b$. This model is typically formulated as *Effort* = $a \cdot Size^b$ and log-transformed to the linear version ln(*Effort*) = ln(a) + b·ln(*Size*) to ease the calculation of the b-value using least square linear regression analysis. The use of a factor input (effort) function means that we have an economy of scale if $b < 1$, a diseconomy of scale if $b > 1$ and a constant return to scale if $b = 1$.

The interest in scale economies in software development seems to be strong. A search (May 3, 2012) in Google Scholar with the terms ("economy of scale" OR "diseconomy of scale") AND "software

development" gives, for example, more than 1,000 hits. The topic is not only interesting to better understand the nature of software development, but is also of potential relevance to software practitioners. If there is an economy of scale, this is an argument for a manager to try to reduce the software development effort by joining smaller projects into larger ones. If there is a diseconomy of scale, the manager may on the other hand try to reduce the effort by splitting larger projects into smaller projects or deliveries, e.g., through incremental development models. It may also be relevant as input to judgments and decisions related to planning and to optimal usage of resources.

It has been much debated among researchers whether and when there is a tendency towards economy of scale, diseconomy of scale or constant return of scale in software development, see (Kitchenham 2002) for some elements of this discussion. The basis of this debate is frequently the variation of the reported *b*-values calculated using regression analysis of log transformed effort and size data. In the survey of twelve software development studies reported in (Dolado 2001) there are *b*-values varying from 0.66 to 1.49. Eight of the twelve studies found $b < 1$, which may reflect a tendency towards reporting economy of scale in software development contexts. A strong economy of scale is, as far as we have experienced, in conflict with what many software professionals would consider likely. Consider, for example, a *b*-value of 0.8, which is not unusual to report for software development data sets. This *b*-value implies that as the software size gets ten times larger, e.g., from 1000 to 10000 lines of code, we will need only 6.3 times more effort ($10^{0.8} = 6.3$). When the software size gets 100 times larger, e.g., from 1000 to 100000 lines of code, we will need only 40 times more effort ($100^{0.8} = 40$). With the possible exception of some types of maintenance environments (Banker and Slaughter 1997), most software professionals seem to believe in a diseconomy rather than an economy of scale. We conducted an informal review of software project planning advice published on the internet and found several indications of a belief in diseconomy of scale and none in economy of scale. Steve McConnel, for example, claims that one should plan to spend an increasing proportion of effort on non-programming activities with increasing project size (McConnel 2004). He also argues that the programming (coding) productivity is close to constant with increasing software size in lines of code. If the proportion of non-programming activities increases and programming productivity is constant, we will observe a diseconomy of scale. Similarly, Capers Jones claims that one should expect that management and support effort increases as the size of the project increases (Jones 1991). A diseconomy of scale is also in accordance with results reporting increasing administrative overhead with increasing organization size, e.g. (Jamtveit, Jettestuen et al. 2009).

The dominance of researchers reporting economy of scale in software development based on analysis of software data sets, in spite of the belief in diseconomy of scale among several software professionals, is a motivation for the analyses presented in this article. Software practitioners and researchers probably use different strategies to reach their conclusions about scale economies. Software

practitioners may for example rely on their experience with increase in percentage effort used on administration and testing as the project increase in size. Software researchers, on the other hand, seem to rely much more on regression analyses of the relation between development effort and software size. As we will try to show in this article, regression analysis may not be suited for deciding on scale economies in software development and the researchers' use of regression analysis-based parameters to assess economy of scale may have led them to report economy of scale in situations where the underlying relationship is linear or even diseconomy of scale.

It is important to note that this paper is *not* about building estimation models, nor is it about identifying the "true" scale economy in software development. The main aims of the papers are to:

1. Explain why software engineering data sets on effort and size, analyzed using linear regression, seem to be dominated by economies of scale.
2. Explain why there are problems interpreting the estimate of the exponential term in an effort-size regression model, such as those used for effort estimation, as an indicator of scale economy or diseconomy.

The remaining part of the paper is organized as follows: Section 2 examines ten software development data sets and finds several instances of interpretation problems, e.g., data sets where we simultaneously find economy and diseconomy of scale dependent on whether we use the production function (regression of size on effort) or the factor input (regression of effort on size) model. We use the interpretation problems as a first step to suggest that there are severe problems with the use of the *b*-value of an effort-size regression model as indicator of economy or diseconomy of scale. Section 3 discusses three reasons for the interpretation problems: The role of random error in the independent variable, incompletely specified models and non-random sampling. Section 4 briefly discusses alternative strategies to assess scale economies in software development. Section 5 concludes.

2. SIMULTANEOUSLY ECONOMY AND DISECONOMY OF SCALE

The common model of the relation between software development effort (*Effort*) and size (*Size*) is on the following factor input function format:

(1) $Effort = a_1 Size^{b_1}$

The production function of the same relationship is the reverse model:

(2) $Size = a_2 Effort^{b_2}$

In the deterministic case, we may reformulate (1) to:

(1a) $Size = (\frac{Effort}{a_1})^{\frac{1}{b_1}}$, which shows that $b_2 = \frac{1}{b_1}$ and $a_2 = (\frac{1}{a_1})^{\frac{1}{b_1}}$

If $b_1 > 1$, we have a diseconomy of scale and we must also have $b_2 = \frac{1}{b_1} < 1$. If $b_1 < 1$ we have an economy of scale and we must also have $b_2 = = \frac{1}{b_1} > 1$. However, if either *Size* or *Effort* or both variables are subject to random error, the parameters of these models can be estimated using linear regression after we log-transform them:

(3) $\ln(Effort) = a_1 + b_1 \ln(Size)$

(4) $\ln(Size) = a_2 + b_2 \ln(Effort)$

The logarithmic transformation is not simply there to allow the value of the *b*-parameters to be estimated using linear regression, although it is important to note that there is no simple closed form algorithm to directly calculate the parameters in (1) or (2). The logarithmic transformation is a normalizing transformation that also reduces the impact of atypical data points. This is needed because effort and size data are not usually normally distributed. A further advantage of the transformation of a normal distribution is that standard linear regression can be used to test whether or not the estimated *b*-parameter is significantly different from 1 and to calculate the confidence interval of the estimate.

Notice also that the analysis of increasing (economy of scale) or decreasing (diseconomy of scale) productivity with increasing size applying the model *Productivity* = $\frac{Size}{Effort}$ = $a \cdot Size^b$ is the same as the analysis based on the above models (3) and (4). This is the case since $\ln(\frac{Size}{Effort})$ = ln(*Size*) – ln(*Effort*) = $a + b \cdot \ln(Size)$, which can be transformed to the format of for example model (3), i.e., to the model ln(*Effort*) = (-*a*) + (*1-b*) · ln(*Size*).

In the stochastic case, when the estimated value of the exponential parameter b_1 is significantly greater than 1, researchers have suggested that this is as an indicator of a diseconomy of scale. Correspondingly, when the estimate of b_1 is significantly less than 1, researchers have suggested that this is an indicator of economy of scale, see for example (Banker, Chang et al. 1994; Hu 1997; Kitchenham 2002). If this interpretation is robust towards change from one meaningful model of the size-effort relationship to another, we would expect the reverse model to behave in the same fashion as the deterministic model, i.e., if we find $b_1 > 1$ then using the same dataset we should find $b_2 < 1$, alternatively if find $b_1 < 1$ then using the same dataset we should find $b_2 > 1$. However, if we find that the estimates of the parameters do not behave in this fashion, i.e., in the same data set the estimates of b_1 and b_2 are both less than 1, or both greater than 1, such that the estimate of one parameter indicates an economy of scale but the other indicates a diseconomy of scale, we have an interpretation problem. We have then simultaneously found evidence for

economy of scale and for diseconomy of scale on the same data set. Unless we have good reasons to trust one of the two models (one of the two regression lines) and not the other, all that we can conclude is that the data set cannot provide any reliable information about economies of scale. Comparing the regression estimates of the exponential parameter based on the original, as in (3), and its reverse, as in (4), is frequently used to determine whether interpreting the parameter *b* as an indicator of underlying relationships is robust or not, see for example (Birnbaum and Hynan 1986; Clogg, Petkova et al. 1995) and it is recommended for this purpose in for example (Campbell and Kenny 1999). Notice that the presence of different scale economy interpretations of the two models does not imply that these models cannot be used for *estimation* purposes, i.e., explanatory and predictive power of models are not the same.

In the following we assume a positive relationship between *Effort* and *Size*, i.e., that the required development effort increases with increased size of the software. As noted above, if there were a deterministic relationship between *Effort* and *Size* we would expect that b_1 and b_2 represented the same degree of economy or diseconomy of scale when based on the same data set. Assume, for example, that b_1 is 0.8, which indicates that adding one unit of size would require 0.8 more units of effort. We would then expect b_2 to be 1.25, indicating that adding 0.8 more units of effort would lead to the production of one more size unit. Otherwise, the two models would give different interpretations of economy of scale on the same data set. The expected relation, if representing the same economy of scale relationship, between b_1 and b_2, is $b_1 = \frac{1}{b_2}$. In addition to reformulating (1), as we did in (1a), this relationship can be derived from the logarithmic formulation of the models, by mean-centering each variable, so that the intercept term equals zero. Then, when applying (3) and (4) without the intercept terms we have that:

(4) $b_1 = \frac{\ln(Effort)}{\ln(Size)} = \frac{\ln(Effort)}{b_2 \ln(Effort)} = \frac{1}{b_2}$

In the following we use g to define the inverse value of b_2 (which only in the deterministic case equals b_1), i.e.,

(5) $g = \frac{1}{b_2}$

In the non-deterministic case, the standard deviations of ln(*Effort*) and ln(*Size*) are defined as the square root of the variance of the variables. $Effort_i$ and $Size_i$ are, respectively, the effort and size values for the observation *i*. *N* is the number of observations.

(6) $s_{\ln(Effort)} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\ln(Effort_i) - \overline{\ln(Effort)})^2}$, $where\ \overline{\ln(Effort)} = \frac{1}{N}\sum_{i=1}^{N}\ln(Effort_i)$

(7) $s_{\ln(Size)} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\ln(Size_i) - \overline{\ln(Size)})^2}$, $where\ \overline{\ln(Size)} = \frac{1}{N}\sum_{i=1}^{N}\ln(Size_i)$

The (Pearson's) correlation between ln(*Effort*) and ln(*Size*) is defined as the covariance between the two variables divided by the product of their standard deviations:

(8) $r = \frac{cov(\ln(Effort),\ln(Size))}{s_{\ln(Effort)}s_{\ln(Size)}}$

The slope of a regression model with a dependent and an independent variable is defined as the covariance between the variables divided by the variance of the independent variable. This gives, by applying (8):

(9) $b_1 = \frac{cov(\ln(Effort),\ln(Size))}{s_{\ln(Size)}s_{\ln(Size)}} = r\frac{s_{\ln(Effort)}s_{\ln(Size)}}{s_{\ln(Size)}s_{\ln(Size)}} = r\frac{s_{\ln(Effort)}}{s_{\ln(Size)}}$

(10) $b_2 = \frac{cov(\ln(Effort),\ln(Size))}{s_{\ln(Effort)}s_{\ln(Effort)}} = r\frac{s_{\ln(Effort)}s_{\ln(Size)}}{s_{\ln(Effort)}s_{\ln(Effort)}} = r\frac{s_{\ln(Size)}}{s_{\ln(Effort)}}$

If b_1 and b_2 represent the same economy of scale relationship we should have that $b_1 = g$, i.e., that $\frac{b_1}{g} = 1$. What we have, using (5), (9) and (10) is, however that:

(11) $\frac{b_1}{g} = b_1 b_2 = r\frac{s_{\ln(Effort)}}{s_{\ln(Size)}}r\frac{s_{\ln(Size)}}{s_{\ln(Effort)}} = r^2$

A more comprehensive discussion of this relationship can be found in (Hausman 2001).

The expression in (11) shows that it is only when there is a perfect correlation (which corresponds to a deterministic relationship) between ln(*Effort*) and ln(*Size*), i.e., when $r = 1$, we can expect the original and reversed regression to result in *b*-values with the same interpretation regarding economy of scale. Anything contributing to a reduced correlation between effort and size will contribute to increased difference between the interpretations of the b-values in the original and the reversed model. When $r < 1$, and b_1 and g are positive, (11) gives $b_1 = r^2 g < g$. This implies that the original regression model (1) produces *b*-values which may be interpreted as indicating more economy of scale (or less diseconomy of scale) than the reversed model (2) when $r < 1$. Notice that a low correlation, potentially leading to large difference between b_1 and g, does not imply that *both* models are incorrect, only that at least one of them cannot be used to argue about economies of scale.

To illustrate how easily an interpretation problem is created, assume a situation where the standard deviation of ln(*Size*) equals that of ln(*Effort*), i.e., $s_{ln(Size)} = s_{ln(Effort)}$. This is, as we will show later in this section, not far from what is typically the case in many software development data sets. Assume further that the correlation between ln(*Effort*) and ln(*Size*) is $r = 0.8$, which is also not unusual in software development contexts. Then, a regression analysis would find $b_1 = 0.8$ and $b_2 = 0.8$ (by use of the expressions (9) and (10)). $b_1 = 0.8$ would be interpreted as indicating strong economy of scale, since an increase of one size unit would increase the required effort with less than one effort unit. Correspondingly, $b_2 = 0.8$ (which gives $g = \frac{1}{0.8} = 1.25$) would indicate a strong diseconomy of scale, since an increase of one effort unit would correspond to

the development of less than one size unit. Clearly, we cannot have both economy and diseconomy of scale for the same data set. If we do not have good reasons to say that all or most of the inaccuracy is in one of the models, we must conclude that the data set cannot be used to investigate scale economies.

The interpretation problem is not only easily created on simulated data sets, but found in real world software development data sets, as well. As an illustration, Figure 1 shows the regression models for the same data set applying the models in (3), which we term Model 3, and in (4), which we term Model 4. The data set used is the one collected for a large telecom company and found in (Jørgensen 1997). A linear regression on the data we get that Model 3 can be expressed as ln(*Effort*) = 4.42 + 0.52 ln(*Size*), where *Effort* is the log-transformed number of work-hours and *Size* is the log-transformed number of unadjusted function points for each project. Similarly, we get that Model 4 can be expressed as ln(*Size*) = 0.68 + 0.68 ln(*Effort*). Reformulating Model 4 gives ln(*Effort*) = $\frac{\ln(Size)-0.68}{0.68} = 1.47\ln(Size) - 1$. In other words, Model 3 (b_1 = 0.68) suggests a strong economy of scale, while Model 4 suggests a strong diseconomy of scale (g = 1.47). Unless we have good reasons to claim that all or most the errors are in one of the models, we will not know which of these scale economies results we should trust.

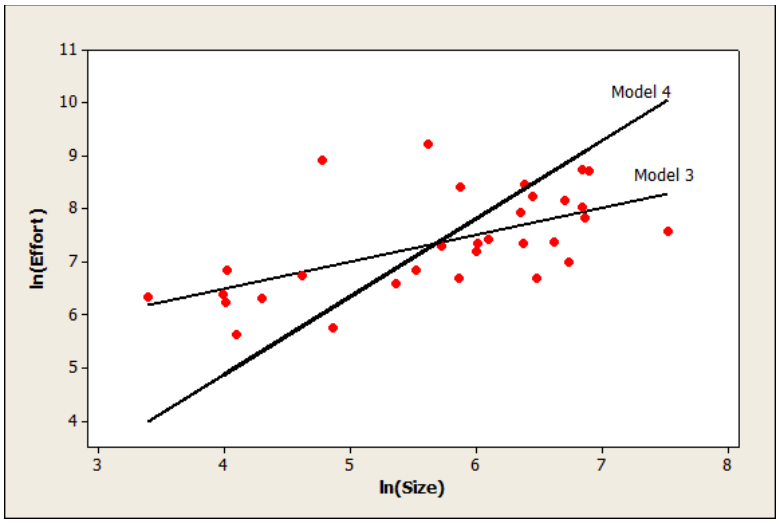


Figure 1 Data set simultaneously displaying economy and diseconomy of scale

Table 1 displays the b-values of the original (3) and the reversed (4) log-transformed regression model of effort and size for ten different data sets. Notice that one data set uses more than one size measure and one data set is divided into more homogeneous subsets of the data sets. There are various size measures used in the data set. For the illustrative purpose of Table 1 it is not essential to know exactly how these size measures are defined, just that they are all meaningful software development size measures. The effects that we study are, as can be seen in (11), depends on imperfect correlation between effort and size, which is the case for all the included size measures. The individual size measures are described in the papers

referenced in Table 1. We have sorted the ten data sets based on increasing *r*-values of the full data sets. As shown in (11) we would expect the difference in interpretation, i.e., the difference between b_1 and *g*, to increase with decreased correlation between effort and size. The data sets selected are those that we had easily available and not based on a careful selection of data sets and we have not carefully examined the quality of the data set. This is in this case not a major limitation for our analyses since the main purpose is *not* to review all possible data sets or to claim that there is an economy or diseconomy of scale based on the data sets, in which case a high quality of the data sets would be essential. Instead, our aim is to use this in our examination of the meaningfulness of the use of the *b*-value as an indicator of scale economies in software development. We have identified the *b*-values significantly different from linearity, i.e., where the two-sided 95% confidence interval did not include 1, with a star (*), and explained what type of scale economy the b-value represents. All *b*-values not significantly different from 1 are characterized as "linear". In many cases lack the label "linear" may be due to low power of the analysis, e.g., low number of observations, rather than a *b*-value close to 1.

For illustrative purposes we have included a simulated data set with the same properties as the one in the previous example, i.e., with identical standard deviation of Size and Effort and a correlation of $r = 0.8$ between ln(*Size*) and ln(*Effort*). This data set was generated so that the underlying (true) relationship represented a constant return on scale, i.e., $b_{TRUE} = 1$. More about the generation and interpretation of the simulated data set is included in Section 3.1, which also includes a visualization of the different scale economy interpretations for that data set.

<Table 1>

Table 1 shows that for the ten datasets, when ignoring subgroups, there are only the data sets in (Kemerer 1987) and (Jeffery and Stathis 1996) where the return of scale of the original and the reversed models are consistent. In both cases, the likely reason for this consistency is that there were too few observations to conclude whether there was a significant return of scale relationship different from a linear one, i.e., the findings of these studies are more related to not being able to exclude a linear relationship than to showing that the *b*-value is close to 1. A low number of observations may also explain the reporting of a linear return of scale for some of the more homogeneous subsets of the data set in (Desharnais 1988).

Notice in particular that Table 1 shows that there is a change from significant economies of scale to significant diseconomies of scale in the data set of (Kitchenham, Pfleeger et al. 2002). In the remaining cases we find that if there is a significant economy of scale, then the inverse model is linear, and if there is a significant diseconomy of scale, then the inverse model is linear. It is also clear that the original model only exhibits linear or economies of scale, while the inverse model only exhibits linear or diseconomies of scale for the studied data sets. Software engineering studies on economy of scale use the original model and, consequently, tend to report economy of scale or linear relationship.

In the next section we discuss potential reasons for the lack of reliability of the *b*-values of the original and the reversed model as indicators of underlying economy or diseconomy of scale in software development. The discussion assumes a continuous (ratio) scale of effort and size. The scale economy interpretation problems are, however, similar when using ordinal scale variables, e.g., dividing size and/or effort into the ordered categories “small”, “medium” and “large”. An ordinal scaled variable may often be viewed as a coarsely measured version of the same, unmeasured continuous variable (Anderson and Philips 1981). In other words, the interpretation problems pointed out in this paper is not purely a problem for regression analyses. It will, for example, also be present if we instead of using regression models choose to divide projects into size categories and use an increase in mean productivity of each size category to indicate an economy of scale.

3. REASONS FOR THE INTERPRETATION PROBLEMS

As reported in Section 2, everything that reduces the correlation between effort and size contributes to an increased difference between b_1 and g ($= \frac{1}{b_2}$), i.e., contributes to difference in the interpretation of the size-effort relationship in the models described in (1) and (2). A difference between the b_1-value and the g-value cannot, as before, exclude that the *b*-value of one of the models (1) and (2) is a meaningful indicator of scale economy. One of the models could be correct, if *all* the interpretation problems were in the other model. It is, as we will illustrate in Section 3.1, possible to have a low correlation between effort and size and still be able to interpret a *b*-value as a reliable indicator of the underlying relationship. Low correlation

between effort and size and low reliability of the interpretation of the *b*-value as economy of scale indicator are related but not identical issues.

This section focuses on three reasons believed to be essential in the contexts of understanding the limitations of interpreting the *b*-values as indicators of economy or diseconomy of scale: random error in the independent variable (Section 3.1), incompletely specified models (Section 3.2), and, non-random samples (3.3). We do not intend to discuss all possible reasons limiting our ability to interpret the *b*-value in terms of scale economy or to give a comprehensive evaluation of methods proposed to solve this type of interpretation problems. Our main goal is to show that there are reasons to believe that the *b*-values of the two models (1) and (2) should not be interpreted as indicators of scale economies in software development contexts. Some of the interpretation problems are, we will argue, inherently hard to solve when relying on observational data of software development effort and size which implies that we should look for other means to examine scale economies.

3.1 Random error in the independent variable

One essential, but frequently ignored, assumption of regression models is that the independent variables should be fixed, i.e., that the variables should have no random error. The lack of accurate measurement and random variation of effort and size in software development contexts means that this assumption is likely to be violated in data sets containing observational data. The consequence of random error in size and effort measurement is that when we use size as the independent variable, and assume as before that there is an increase of effort with increased software size, the b_1-value will be biased downwards and when we use effort as the independent variable, the b_2-value will be biased downward (and consequently $g = \frac{1}{b_2}$ upwards). Since an increase in random error of size and/or effort decreases the correlation between effort and size it will, as described in Section 2, also increase the gap between b_1 and g.

Assume that we measure size and effort with some error, i.e.:

(12) $Size = Size_{TRUE} e'_{Size}$

(13) $Effort = Effort_{TRUE} e'_{Effort}$

The use of a multiplicative error term on the original values leads to an additive error term for the log-transformed values, i.e.:

(14) $\ln(Size) = \ln(Size_{TRUE}) + e_{\ln(Size)}$, where $e_{\ln(Size)} = \ln(e'_{Size})$

(15) $\ln(Effort) = \ln(Effort_{TRUE}) + e_{\ln(Effort)}$, where $e_{\ln(Effort)} = \ln(e'_{Effort})$

The *b*-value corrected for random error in the ln(*Effort*) = a_1 + $b_1$ln(*Size*) model can then, given certain assumptions such as independent error terms of effort and size, be expressed as:

(16) $b_1^{'} = b_1 \left(1 + \frac{s^2_{e_{\ln(Size)}}}{s^2_{\ln(Size)}}\right)$, where $s_{e_{\ln(Size)}}$ is the standard deviation of $e_{\ln(Size)}$.

Equation (16) implies that the ratio of the variance of the random error of ln(*Size*), i.e., $s^2_{e_{\ln(Size)}}$, to the total variance of ln(*Size*), i.e., $s^2_{\ln(Size)}$, affects how much we can trust the observed b-value as indicator of scale economies. The higher the ratio of random error variance to total variance, the more the b_1-value will be biased downwards and the less likely it is to reflect the underlying relationship between effort and size. The equation in (16) also implies that the interval of project sizes, e.g., measured as the standard deviation of ln(*Size*), in the data set will have a strong impact on the effect of random error on the b_1-value. The b_1-value of data sets where the projects vary little in size is likely to be lower than in data sets where the projects vary more in size, given similar random error of the measurements. This is supported by a correlation of -0.51 ($p < 0.06$) between the ranks of $s_{\ln(Size)}$ and b_1 of the data sets in Table 1 (including the three Desharnais sub groups as well as the overall data set analysis), i.e., we observe a strong correlation (although not quite significant at the $p < 0.05$ level) between lower variation in project sizes and studies reporting economy of scale. In the presence of random error, studies of software development contexts with little variation of project sizes will consequently be more likely to report economy of scale than studies with more variation in project sizes, even when the underlying scale economy is the same.

Correspondingly, the corrected *b*-value of the reversed regression model ln(*Size*) = a_2 + b_2·ln(*Effort*) can be expressed as:

(17) $b_2^{'} = b_2 \left(1 + \frac{s^2_{e_{\ln(Effort)}}}{s^2_{\ln(Effort)}}\right)$, where $s^2_{e_{\ln(Effort)}}$ is the standard deviation of $e_{\ln(Effort)}$.

The meaning of the concepts of true value and random error in the context of software size measurement can be illustrated by considering a set of independent repetitions of the "same" size measurement, e.g., a repeated measurement of the lines of code or function points of the same piece of software. We will then expect some variations in the values, e.g., due to inconsistency in counting practices, inconsistency of how the measure is interpreted by different people and in different contexts, unclear counting rules and human errors in the measurement process. Different people responsible for the counting of software size in an organization may, for example, interpret the concept of lines of code differently and

differ in how they count re-used code. There may also be elements leading to variation of size measurements that has nothing to do with actual size of the software, such as differences in coding styles. These elements will contribute to variation in the measured software size of the same piece of software, i.e., will lead to increased random error in the size measurement. The true value of size may be considered as the actual size given no error in the measurement, or, if there is no clear interpretation of true or actual size, a central value of the distribution of measurements of size. This understanding of the true size and random error is similar to the concepts of true score and measurement error in classical test theory, see for example (Cronbach 1972). Correspondingly, a random error in the effort measurement will lead to a deviation between the true effort and the observed effort. The random error in effort measurement can, for example, be a result of inconsistencies in how the client's work effort in a project is counted and incorrect or inconsistent logging of effort.

If either size or effort has very low random error, we may succeed in decreasing the interpretation bias by choosing that variable as independent variable. If, however, both size and effort have substantial amounts of random error, we need to know the amount of random error of at least one of them to find the unbiased (corrected) the *b*-value. Unfortunately, the situation in software development seems to be that we neither know much about the degree of random error of any of the variables, nor are able to measure size or effort without substantial random error. This problem is inherent in how we measure software size and effort and the nature of observational studies.

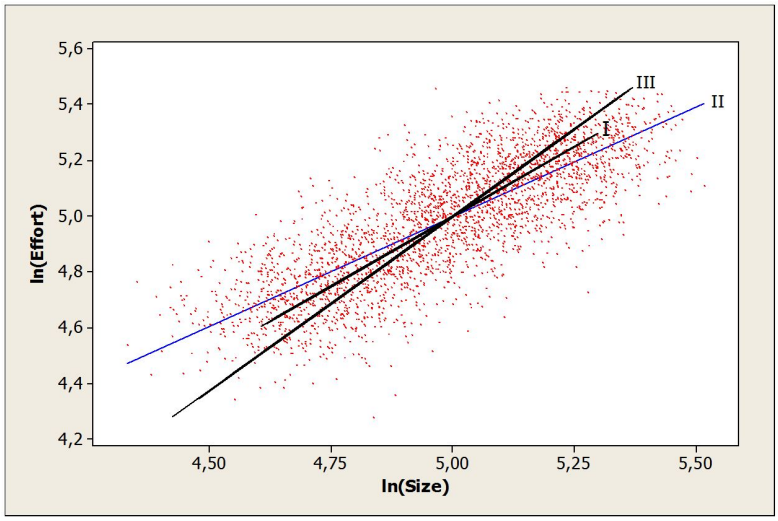
The following shows the effect of random error in the independent variable on the *b*-values and describes how we generated the data set in Table 1. The data set was created as follows:

- The true size values ($Size_{TRUE}$) (n = 3000) were randomly drawn from a uniform distribution with values between 100 and 200 units. The rather low variance in software sizes will, as can be seen in Table 1, lead to lower standard deviation than in most real-world software size data set, but will not change the main message of the simulation. As pointed out earlier, however, a larger variation of project sizes will reduce the effect of random error.
- The true effort values ($Effort_{TRUE}$) were set to equal the size values. This ensured that the underlying relationship between effort and size was a perfect linear relationship, i.e., a constant return on scale and b_{TRUE} = 1.
- The random error values of *Size* (e'_{Size}) and *Effort* (e'_{Effort}) were randomly drawn from a normal distribution with mean 1 and standard deviation 0.1. These error terms are, see (12) and (13), multiplicative.
- The observed *Size* values were calculated as $Size_{TRUE}e'_{Size}$ and the observed *Effort* values as $Effort_{TRUE}e'_{Effort}$.

- The regression analyses of the original and reversed model were conducted on the log-transformed values of *Effort* and *Size*. The distributions of the error terms of the transformed models are then not perfectly normally distributed, but sufficiently normal for our simulation purposes.

Figure 2 displays the three regression lines for the simulated data set. This includes: (I) The true relationship, i.e., the relationship between ln($Effort_{TRUE}$) and ln($Size_{TRUE}$), (II) The regression line of the model $Effort = a_1 Size^{b_1}$ (the original model), (III) The regression line of the model $Size = a_2 Effort^{b_2}$ (the reversed model). As reported in Table 1, the correlation (r) between ln(*Effort*) and ln(*Size*) was 0.80, which gives $b_1 = b_2 = 0.80$ due to identical standard deviations of *Size* and *Effort*. The slopes in Figure 2 are correspondingly 0.80 for regression line II, and $\frac{1}{0.8} = 1.25$ for regression line III. The linear relationship between ln($Effort_{TRUE}$) and ln($Size_{TRUE}$) gives $b_{TRUE} = 1$ for regression line I.

Figure 2: Simulated data set with random error in the independent variable



If we change the data set to *Size* values with no random error, we will find that the original model gives the correct constant scale of return relationship ($b_1 = 1$), in spite of an imperfect correlation, which in this case will be higher than 0.89 since we removed the random error of *Size*. Correspondingly, if we change the data set so there is no random error in the *Effort* values, the reversed model will give the correct relationship ($b_2 = 1$). The problem is, as argued earlier, that we neither know the degree of random error in size or effort, nor can assume no random error in either of them.

The purpose of the simulated data set is mainly to illustrate the effect of random error and the realism of it may be debated. Two arguments in favor of its realism are, however, that the data set represents a typical correlation between effort and size, as can be seen in Table 1, and that the degree of random error does not seem unreasonable. The random errors of both effort and size were drawn from a normal distribution with mean 1 and standard deviation 0.1. This means that 68% of the measurements are

measured with less than 10%, and 95% with less than 20% deviation from the true value. Given the lack of accurate measurement of software size and effort in typical software development contexts, it is not unreasonable to assume that such values are observed in many real-world data sets. If this amount of random error in effort and size measurement is typical for the data sets in Table 1, we see that the random error alone may explain most of the interpretation problem and excludes the use of the *b*-value as a reliable indicator of scale economies. Empirical support for this high level of random error, at least for the size measurement, is documented by the findings in, for example (Kemerer 1993), where the median difference in function point measurement from pairs of raters was 12%. We reanalyzed the COSMIC function point measurements of different raters reported in (Ungan, Demirörs et al. 2009) and found a geometric standard deviation of 28%, which is a substantially higher standard deviation than what we assumed in our simulated data set. An even higher degree of inconsistency when using lines of code as size measure instead of function points, is claimed in several papers, e.g. in (Low and Jeffery 1990). We have been unable to find studies related to the random error of software development effort measurement. Based on our experience with measurement in the software industry we would expect substantial measurement inaccuracies related to inconsistent logging of work effort, i.e., the *b*-value of the reverse regression model is likely to be biased by random error as well.

The problems with statistical analyses of observational data in situations with random error in the independent variables are not unique for economy of scale studies and several solutions have been proposed. To correct for random error in the independent variable we may, for example, use information about the random error of the independent variable, the ratio of the random error to the total variance, or the ratio of random error of the dependent to the independent. Orthogonal regression analysis (Carrol and Ruppert 1996), for example, assumes that the level of random error is the same in the dependent and the independent variable, Deming regression (Mittas, Kosti et al. 2010) requires knowledge about the ratio of variance of the random error of the dependent and the independent variable, and the method proposed in (Blomquist 1986) that the ratio of the variance of random errors of the dependent to the variance of independent variable is known. We may also use "instrumental variables" (Fuller 1987), the geometric mean functional relationship suggested in (Barker, Soh et al. 1988), and, analyses of the change in variance (Oldham 1962). To what degree any of these methods solve the problems related to random error in the independent variable in the type of analyses discussed in this article, without introducing new problems, is hard to tell. A general observation is that the methods seem to require information difficult to collect or assumptions difficult to evaluate. The corrections could therefore easily lead to even larger analyses biases (Hausman 2001). As an illustration of to what extent the error in the independent variable is random is hard to evaluate, which may have as consequence that "corrections" based on this assumptions does not improve the analyses, see (Reichardt 2000).

3.2 INCOMPLETELY SPECIFIED MODELS

Incompletely specified models may bias the b-values. It is, for example, stated in (King, Keohane et al. 1994) that if "*relevant variables are omitted, our ability to estimate causal inferences correctly is limited*".

To demonstrate the interpretation problems due to omitted variables, assume that the correctly specified model is:

(18) $\ln(Effort) = a_1' + b_1' \ln(Size) + c_1' X$, where *X* represents a variable that contributes to explaining the variance of ln(*Effort*), i.e., $c_1' \neq 0$.

The model we use in our analysis of economy of scale is, however ln(*Effort*) = a_1 + b_1·ln(*Size*). Not including *X* in the model biases the *b*-value so that:

(19) $b_1 = b_1' + b_{\ln(Size),X} c_1'$, where $b_{\ln(Size),X}$ is the slope found when regressing the omitted variable *X* on ln(*Size*).

The expression (19) implies that the omitted variable leads to biased interpretation of b_1 in the original model if the omitted variable contributes to some of the variance of effort, i.e., $c_1' \neq 0$, and, when the slope found when regressing *X* on ln(*Size*) is non-zero, i.e., $b_{ln(Size),X} \neq 0$. The model typically used to assess economies of scale in software engineering will consequently find a too high b_1–value if the omitted variable either has a positive correlation with both effort and size or a negative correlation with both effort and size. Otherwise, the analysis will give a too low b_1-value. The expression of the effect of omitted variables in the general case with more omitted variables is given in, for example, (Hanushek and Jackson 1977; Clarke 2005).

In software development contexts there is not only one variable that is missing in the original model (3), to explain the variance in ln(*Effort*). Projects of the same size may vary in effort usage dependent on for example complexity, team size, developer skill, length of the project, client maturity and tool usage. Several of these variables are likely to correlate with software size and, hence, potentially bias the reported b_1-value. There are also other challenges related to the specification of models in software development contexts, e.g., the data may come from populations with different underlying relationships and we may make incorrect assumptions about data point independence and functional relationship between variables.

To fully avoid the biasing effect of omitted variables we must include all relevant variables in our model. Preferably, these variables should have low inter-correlation to avoid multicollinarity. Unfortunately, in software development we are hardly able to measure essential variables with high precision (or at all), the number of potentially relevant variables is high, and some of the relevant variables are inter-correlated. Our

choice is consequently not between a complete and an incomplete model, but between different variants of incomplete models with inter-correlated variables measured with random error. This has as a consequence that it is hardly possible to assess the effect of extending the models with one or a few more variables on the interpretation bias. A factor further complicating this is that we may not even be able to model the included variable correctly, e.g., some of the relationships in the log-transformed model may be non-linear.

To illustrate the interpretation problems on a real world data set, we use the data set found in (Desharnais 1988). It is possible to argue that a model explaining the effort usage should include the calendar time (*Length*) of the projects. With the same software size, projects that last longer may, for example, on average require more effort due to higher likelihood of changing requirements and less efficient use of resources. Length and Size are likely to be correlated and we will then, in accordance with (19), have that models including and omitting the variable *Length* find different b_1-values. The extended model is specified as follows:

$$(20)\ \ln(Effort) = a_1' + b_1' \ln(Size) + c_1' \ln(Length)$$

A regression analysis gives that b_1' = 0.71 and c_1' = 0.37 for the data set. Both parameters are significantly different from zero, with p < 0.01. The data in Table 1 shows that the b_1-value of the original model, i.e., the model in (20) without the ln(*Length*) variable, is 0.94. An increase in the b_1-value due to omission of the ln(*Length*) variable is as expected since we would expect that Length is positively correlated with both Effort and Size. The slope when regressing ln(*Length*) on ln(*Size*) is 0.62. The relevance of (19) is confirmed by that $b_1 = b_1' + b_{ln(Size),ln(Length)}\, c_1'$ = 0.71 + 0.62·0.37 = 0.94. The correlation between ln(*Length*) and ln(*Size*) is 0.63, which is not high enough to expect strong effects from multicollinearity. The $b_{1'}$-value of the extended model (20) consequently suggests an even stronger economy of scale than the original model. It is, as argued earlier, not clear which of the *b*-values we can trust more as the number of other variables we could add is high and the random error of the added variable may potentially contribute to increased interpretation problems. The purpose of the illustration is consequence mainly to demonstrate the lack of robustness of one-variable models in situations where there are other relevant variables correlating with the included independent variable, not to indicate that the bias from omitted variables is in one particular direction. Clarke discusses the effect of omitted variables and concludes (Clarke 2005, p. 350): "*The addition* [of additional relevant variables] *may increase or decrease the bias, and we cannot know for sure which is the case in a particular situation.*"

The problem of incompletely specified models is, as with the problem of random error, not easily solved in software development contexts. We do not have a few theoretically founded, non-correlated, independent variables that explain most of the variance in the dependent variable, i.e., of effort or size. Instead we typically have a high number of potentially important variables, relatively low explanatory power

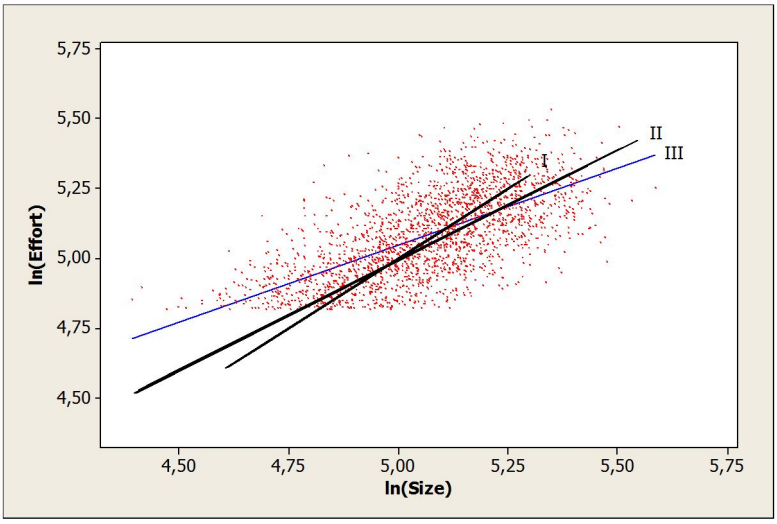
and potential multicollinarity problems. This is certainly not the best situation for reliable interpretation of one of the parameters (b_1) as indicating economy or diseconomy of scale even if we tried to avoid the omitted variable bias through adding several other variables.

3.3 NON-RANDOM SAMPLES

Non-random samples, e.g. exclusion of projects spending less than 100 work-hours, may bias the *b*-values of our regression models even when we restrict the interpretation to the sampled interval. Berk examined the neglect of random sampling and concludes that (Berk 1983): "*This neglect represents a major oversight with potentially dramatic consequences since internal as well as external validity is threatened*".

Figure 3 illustrates the interpretation problem in a software development economy of scale context through the display of three regression lines of the simulated data set where the true *b*-value is 1. Regression line I is the true relationship. Regression line II is the one found when regressing ln(*Size*) on ln(*Effort*) using the complete data set. Regression line III is the regression line found when excluding the smallest 25% of the projects, as measured by the use of effort. The slope of regression line III is 0.55, which is substantially lower of the already too low *b*-value of the regression line II, i.e., the *b*-value 0.79.

Figure 3: Non-random sample, with criterion based on the dependent variable



As can be seen in Figure 3, excluding projects with effort lower than a particular threshold increases the bias towards economy of scale when using the original model. This happens even if we restrict our interpretation to project sizes that are included in the sample. It is therefore not sufficient to state that the reported economies of scale only represent the interval of project sizes included in the data set, when the process of inclusion or exclusion of projects is non-random.

An exclusion of projects purely based on software size will not increase the bias of the b_1-value, i.e., it will produce the regression line II in our simulated example. It is, in this simulated case, the random error in the dependent variable that causes the problems. In other words, while random error in the independent

variable always biases the *b*-value, random error (low explanatory power) in the dependent variable biases the *b*-value when the sample is non-random. If we instead of excluding the project with low effort, exclude the projects with the highest effort this would also lead decrease the *b*-value of the original model. Excluding both the smallest and the largest projects further increases the downward effect on the *b*-value. If we exclude both the smallest and the largest 25% of the project in our simulated data set, we get a b_1-value as low as 0.32.

It is hard to say exactly how large this problem is in software development data sets, but all data collections has to set a minimum size criteria somewhere and the criteria used is likely to be influenced by work effort. In (Kitchenham, Pfleeger et al. 2002), for example, the selection was as follows: "*Changes and enhancements requiring less than 200 h of staff time are tracked separately. We do not include them in the data set analyzed here.*" It is also sometimes the case that some of the projects with very high effort usage are excluded due to a fear of including outliers. In both cases, the non-randomness of the sample will bias the *b*-values of the original model downward. Since the random error in the size variable is likely to already deflate the b_1-value, this will exaggerate the bias towards reporting economy of scale, even in situations where the underlying relationship is a tendency towards linear or diseconomy of scale. Situations like this may potentially contribute to the tendency towards reporting economy of scale in regression-based analysis in spite of the seemingly strong belief in diseconomy of scale among software practitioners.

The issues related to non-random sampling is not only relevant when discussing the interpretation of the b_1-value as indicator of economies of scale, but also highly relevant, and perhaps neglected, when developing effort estimation models. While random error in the independent variable and omitted variables may not be crucial for the accuracy of estimation models, an estimation model derived from non-random selection of data is likely to give inaccurate effort estimates. As illustrated in Figure 3, the model will be inaccurate even when the use of the estimation model is restricted to project sizes from which the model was derived.

4. DISCUSSION

The use of regression models to understand underlying relationships is a source of inconsistent research results in many domains. In sociology, for example, regression model-based studies find that female employees with the same qualification as men have a lower salary than them. Reversing the regression model gives, however, the result that female employees with the same salary as men have lower qualifications than them (Solon 1983). An observation of interpretation differences such as the above naturally makes researchers doubt the reported results and examine possible biasing elements. The elements that are looked into to explain the differences in model interpretations, are frequently the same as

those we have discussed in this paper, i.e., random error in the independent variable, omitted variables, and non-random sampling. Based on the identified biases, several researchers move on to suggest methods to improve of the analysis, such as methods correcting for omitted variables and measurement error, see e.g. (Marais and Wecker 1998). A convincing solution to the interpretation problems, not introducing new problems, seem however to be hard to agree on.

We believe that it will be hard to solve the interpretation problems for the purpose of identifying the underlying economy of scale in software development through regression models or similar statistical analyses. We measure variables with unknown amount of random error, we typically have a high, unknown amount of omitted, relevant variables, important relationships cannot be assumed to be stable, the functional relationship may be hard to specify, and there may be unknown biasing effects from non-random sampling. All the above problems can in theory be solved, but will require sophisticated statistical instruments that can introduce new problems or information that is not realistic to gain access to in software development contexts.

The most promising strategies to gain more knowledge about the underlying scale economies in software development may instead be to:

i) Introduce randomized controlled experiments with fixed software sizes and random allocation of development of software of different sizes. This type of experiments may enable a strong reduction of random error in the size measurement and better control of the problems related to omitted variables. The value of such experiments is however limited by the software sizes possible to study through controlled experiments and the results may not say much about scale economies of larger projects. The problem of non-random samples may also still be a problem.

ii) Use more in-depth of analyses of software projects. Examples of in-depth analysis strategies include comparison of types of activities, communication overhead, administrative overhead, quality assurance processes, etc. in small and large software projects. This type of in-depth studies is, in our experience, currently missing. We were, for example, unable to find good empirical studies on whether there is an increase, constancy or decrease in proportion of project management effort with increasing project size.

5. CONCLUSIONS

There are severe problem with the interpretation of the *b*-value in $Effort = a \cdot Size^b$ as a meaningful indicator of economy or diseconomy of scale in software development and we strongly discourage this type of interpretations in future software engineering research. The observation of instances where we find economy and diseconomy of scale in the same data set dependent on whether we model scale economies as

a factor input model or as production function model, is perhaps the strongest argument in support of the presence of the interpretations problems. The seemingly dominance of studies reporting economy of scale in software development would, for the same reason, be replaced by a dominance of studies reporting diseconomy of scale when examining economy of scale through a product function rather than a factor input model. Elements likely to cause the problems of interpreting the *b*-value as indicator of economy or diseconomy of scale include: i) Random error in the independent variable, ii) Omission of relevant variables, and iii) Non-random samples.

There may be no easy way to solve the interpretation problems in the context of regression models or similar statistical analyses. Future research on this topic may therefore have to base their study designs on controlled experiments or in-depth analyses of software projects rather than try to use even more sophisticated regression models to gain insight into scale economies in software development.

References:

Anderson, J. A. and Philips, P. R. 1981. "Regression, discrimination and measurement models for ordered categorical variables." *Applied Statistics* **30**(1): 22-31.

Banker, R. D., Chang, H. and Kemerer, C. F. 1994. "Evidence on economies of scale in software development." *Information and Software Technology* **36**(5): 275-282.

Banker, R. D. and Slaughter, S. A. 1997. "A field study of scale economies in software maintenance." *Management Science* **43**(12): 1709-1725.

Barker, F., Soh, Y. C. and Evans, R. J. 1988. "Properties of the geometric mean functional relationship." *Biometrics* **44**: 279-281.

Berk, R. A. 1983. "An introduction to sample selection bias in sociological data." *American Sociological Revew* **48**: 386-398.

Birnbaum, M. H. and Hynan, L. G. 1986. "Jugments of salary bias and test bias from statistical evidence." *Organizational Behavior and Human Decision Processes* **37**: 266-278.

Blomquist, N. 1986. "On the bias caused by regression towards the mean in studying the relation between change and initial value." *Journal of clinical peridontology* **13**: 34-37.

Boehm, B. W. 1981. *Software engineering economics*. New Jersey, Prentice-Hall.

Campbell, D. T. and Kenny, D. A. 1999. *A primer on regression artifacts*. New York, The Guilford Press.

Carrol, R. J. and Ruppert, D. 1996. "The Use and Misuse of Orthogonal Regression in Linear Errors-in-Variables Models." *The American Statistician* **50**(1): 1-6.

Clarke, K. A. 2005. "The phantom menace: Omitted variable bias in econometric research." *Conflict Management and Peace Science* **22**: 341-352.

Clogg, C. C., Petkova, E. and Haritou, A. 1995. "Statistical methods for comparing regression coefficients between models." *The Americal Journal of Sociology* **100**(5): 1261-1293.

Cobb, C. W. and Douglas, P. H. 1928. "A theory of production." *American Economic Review* **18**(1): 139-165.

Cronbach, L. J., Gleser, G. C, Nanda, H., Rajaratnam, N. 1972. *The dependability of behavioral measurements: Theory of generalizability for scores and profiles*. New York, John Wiley & Sons.

Desharnais, J. M. 1988. Analyse statistique de la productivité des projects de dévelopment en informatique a partir de la technique des points de fonction, Univ. du Québeck á Montreal. **Master Thesis**.

Dolado, J. J. 2001. "On the problem of the software cost function." *Information and Software Technology* **43**(1): 61-72.

Fuller, W. A. 1987. *Measurement error methods*.

Hanushek, E. A. and Jackson, J. E. 1977. *Statistical methods for social scientists*. New York, Academic Press.

Hausman, J. 2001. "Mismeasured variables in econometric analysis: Problems from the right and problems from the left." *Journal of Economic Perspectives* **15**(4): 57-67.

Hill, J., Thomas, L. C. and Allen, D. E. 2000. "Experts' estimates of task durations in software development projects." *International Journal of Project Management* **18**(1): 13-21.

Hu, Q. 1997. "Evaluating alternative software production functions." *IEEE Transactions on Software Engineering* **23**(6): 379-387.

Jamtveit, B., Jettestuen, E. and Mathiesen, J. 2009. "Scaling properities of European research units." *PNAS* **106**(32): 13160-13163.

Jeffery, R. and Stathis, J. 1996. "Function point sizing: Structure, validity and applicability." *Empirical Software Engineering* **1**(1): 11-30.

Jones, C. 1991. *Applied software measurement*. New York, McGraw-Hill.

Jørgensen, M. 1995. "Experience with the accuracy of software maintenance task effort prediction models." *IEEE Transactions on Software Engineering* **21**(8): 674-681.

Jørgensen, M. 1997. *An empirical evaluation of the MkII FPA estimation model*. Norwegian Informatics Conference, Voss, Norway, Tapir, Oslo: 7-18.

Kemerer, C. F. 1987. "An empirical validation of software cost estimation models." *Communications of the ACM* **30**(5): 416-429.

Kemerer, C. F. 1993. "Reliability of function points measurement: A field experiment." *Communications of the ACM* **36**(2): 85-97.

King, G. R., Keohane, O. and Verba, S. 1994. *Designing social inquiry: Scientific inference in qualitative research*, Princeton University Press.

Kitchenham, B., Pfleeger, S. L., McColl, B. and Eagan, S. 2002. "An empirical study of maintenance and development estimation accuracy." *Journal of Systems and Software* **64**(1): 57-77.

Kitchenham, B. A. 2002. "The question of scale economies in software—why cannot researchers agree?" *Information and Software Technology* **44**(1): 13-24.

Low, G. C. and Jeffery, D. R. 1990. "Function points in the estimation and evaluation of the software process." *IEEE Transactions on Software Engineering* **16**(1): 64-71.

Marais, M. L. and Wecker, W. E. 1998. "Correcting for omitted-variables and measurement-error bias in regression with an application to the effect of lead in IQ." *Journal of the American Statistical Association* **93**(442): 494-505.

McConnel, S. 2004. *Code Complete: A Practical Handbook of Software Construction*, Springer.

Mittas, N., Kosti, M. V., Argyropoulou, V. and Angelis, L. 2010. *Modeling the relationship between software effort and size using Deming regression*. International Conference on Predictive Models in Software Engineering (PROMISE 2010), Timisoara, Romania.

Miyazaki, Y., Terakado, M., Ozaki, K. and Nozaki, H. 1994. "Robust regression for developing software estimation models." *Journal of Systems and Software* **27**(1): 3-16.

Oldham, P. D. 1962. "A note on the analysis of repeated measurements of the same subjects." *Journal of chronic diseases* **15**: 969-977.

Reichardt, C. S. 2000. "Regression facts and artifacts." *Evaluation and Program Planning* **23**: 411-414.

Solon, G. 1983. "Errors in variables and reverse regression in the measurement of wage discrimination." *Economics letters* **13**: 393-396.

Ungan, E., Demirörs, O., Özcan, T. and Özkan, B. 2009. An empirical study of the reliability of COSMIC measurement results. *Software process and product measurement*. A. Abran. Berlin, Springer. **5891:** 321-336.