

Using Chronological Splitting to Compare Cross- and Single-company Effort Models: Further Investigation

Chris Lokan

School of IT&EE UNSW@ADFA
Canberra ACT 2600, Australia,
+61 2 6268 8060

`c.lokan@adfa.edu.au`

Emilia Mendes

Computer Science Department
University of Auckland
Private Bag 92019, Auckland, New Zealand
+64 9 373 7599 86137

`emilia@cs.auckland.ac.nz`

Abstract

Numerous studies have used historical datasets to build and validate models for estimating software development effort. Very few used a chronological split (where projects' end dates are used so that training sets only contain projects that were completed before the start date of each project in the validation set), and only one compared chronological split to random split. Therefore the aim of this study is to investigate further and compare the use of chronological and random splitting. We do so in the context of comparing cross-company and single-company models for effort estimation. We used 450 single-company projects and 741 cross-company projects from the ISBSG Release 10 repository, and estimates were obtained using manual stepwise regression. We found that with these data the use of chronological splitting, and different splitting dates, did not affect prediction accuracy. We were not able to obtain a converging set of findings when comparing cross- to single-company predictions given that different accuracy measures presented contradictory results.

Keywords: Chronological Splitting, Effort estimation, Cost estimation, Cross-company data, Single-company data, estimation accuracy.

1 Introduction

Models for estimating software development effort are normally built and evaluated using a set of historical projects. The normal approach involves separating the data into a training set (from which the model is built) and a testing set (on which the model's accuracy is assessed).

When using leave-one-out cross-validation, each of the N projects in the data set is estimated in turn using a model built from the other $N-1$ points. Alternatively, the data can be allocated randomly or in a stratified way into a single training set (usually of about two thirds of the data) and a single testing set. Bootstrap and k -fold validation fall somewhere in between, involving multiple repetitions of model building and validation on different training and testing sets.

Almost invariably, the assignment of projects to training and testing sets is done without regard to the date in which the projects were completed. This makes it likely that the data used to build a model to estimate the effort for a given project includes projects that had not even started at the time the given project commenced.

In a real setting, an estimator building a model to estimate a new project could only consider the set of projects already completed, i.e., future projects could not be considered.

Thus there is an apparent mismatch between typical research practice and industrial practice. The question is, does it matter? If it makes a difference to the accuracy of the estimates, it would seem that in order to reflect reality when building estimation models from historical data, the allocation of projects to training and testing sets should be based on chronology.

An important point to add is that less information is available when estimating effort in a real industrial environment than in traditional research approaches that ignore the chronological sequence of projects. This occurs for two reasons: First, in a date-based approach, insight for building models is only drawn from part of the time span; a better understanding of trends and relevant factors can be expected when data is considered from the entire span. Second, early projects in the chronological sequence only have small sets of training data available. If we assume that estimates based on less information are likely to be less accurate, we conjecture that the apparent accuracy of estimates obtained with traditional research approaches is too optimistic.

With this background in mind, in this paper we revisit an earlier study (Mendes and Lokan 2008) in which we compared the accuracy of single-company and cross-company estimation models on projects from a single

company, using data drawn from the ISBSG repository. The previous study involved going through the entire process of building a new regression model for each individual project. Although this is probably most realistic as a simulation of estimation in practice, with a large data set it takes far too long for a researcher unless one uses automated means. Hence we wish to investigate here whether a simpler approach, simply splitting data into training and testing sets based on chronology, makes any difference.

Research that considers dates could take several approaches, including comparing chronological and random splits; estimating projects one by one in sequence and studying the effect of a growing portfolio (as in Mendes and Lokan 2008); and considering only a window of recent projects. In this paper we study the first of these options, using two different splitting dates and two corresponding random splits.

The research questions addressed are as follows:

1. Using a date-based selection of cross-company (CC) and single-company (SC) projects:
 - a) How successful is a CC model at estimating effort for projects from a single company, when the model is built from a data set that does not include that company's projects?
 - b) How successful is a CC model, compared to a SC model?
2. Using a random selection of CC and SC projects:
 - a) How successful is a CC model at estimating effort for projects from a single company, when the model is built from a data set that does not include that company's projects?
 - b) How successful is a CC model, compared to a SC model?
3. Does the use of a chronological split, instead of a random split, affect the accuracy of the models? The null hypothesis is that there is no difference.
4. Does the use of different chronological splits produce different results? The null hypothesis is that there is no difference.

The contributions of the paper are to extend previous work on CC vs. SC models, using the latest version of the ISBSG database and using a splitting approach that reflects more closely a real situation; and to investigate the impact of chronological splitting based on a splitting date.

The remainder of the paper is organized as follows: Section 2 briefly summarizes related work. Section 3 describes the research method employed in this study. Results are presented in Section 4. Research questions are answered in Section 5. Conclusions and directions for future work are given in Section 6.

2 Related work

2.1 Use of chronological splitting

Chronological splitting has been applied as a data-splitting approach in several other domains, but is rare so far in software engineering. The few studies in software

engineering that considered chronological splitting are presented next.

Lefley and Shepperd (2003), and Sentas et al. (2005), used chronological splitting as the basis for splitting their data into training and testing sets, when comparing effort & productivity models; however they did not investigate chronological splitting as a research question in its own right.

Two studies have addressed a slightly different problem, using information from previous phases within a project to estimate effort in later tasks in the same project. MacDonell and Shepperd (2003) studied effort distribution in major waterfall phases in 16 projects. They found the patterns of effort distribution between phases varied too much for this approach to work by itself, though it was helpful in conjunction with expert estimation. Abrahamsson et al. (2007) were more successful in two projects from an extreme programming environment, finding that data from early iterations produced increasingly accurate effort estimates for later iterations.

Auer and Biffl (2004), and Auer et al. (2006) considered chronology directly in their research into estimation by analogy. They tracked changes in accuracy as the portfolio of completed projects grew. However, they did not consider chronological splitting vs other methods as a separate research question.

The chronology of projects was also important in (Premraj et al. 2005), which investigated changes in software development productivity over time. Their focus was on characterizing productivity in different years, rather than using past data to estimate future projects.

Recently Lokan and Mendes (2008) investigated the use of project-by-project chronological split to compare the prediction accuracy between cross- and single-company effort models, and compared this type of split to another two different types of cross-validation (leave-one-out and leave-two-out). No differences were observed; however, only one type of chronological split was investigated, leaving a clear need for further investigation in this area.

2.2 Cross-company vs. single-company models

Kitchenham et al. (2007) published a systematic review of studies comparing the effort prediction accuracy between CC and SC models. Another five papers have since addressed the same issue (Lokan and Mendes 2006, Mendes and Lokan 2008, Mendes et al. 2008, Premraj and Zimmerman 2007, Lokan and Mendes 2008).

The conclusions of the studies vary, and it is not clear what characteristics of the data sets and analysis methods affect the outcome. Kitchenham et al. (2007) note that there are no consistent patterns concerning quality controls on data collection, the quality of the overall study, the size metrics used, the procedure used to build the SC model, or the strength of the CC relationship.

If there is a trend emerging, it is related to the homogeneity of the data. Kitchenham et al. (2007) note that all studies in which SC models were significantly better than CC models used small SC data sets; they conjecture that large data sets from a single company may be an indication of the size of the company and the homogeneity of the data set. The range of effort values in

the SC data set, compared to the range in the CC data set, also seems important: the greater the difference, the less likely it is that a CC model will provide accurate estimations for single company project.

3 Research Method

3.1 Data set description

The analysis presented herein was based on software projects from Release 10 of the ISBSG¹ database. This release contains data on 80 variables for 4106 projects, including 566 from a single company². Not all projects provided the chronological data we needed (i.e. known duration and completion date, from which we could calculate start date), and those that did, varied in data quality and definitions. To form a data set suitable for our analysis, we removed projects according to carefully chosen criteria (detailed in the Appendix) in order to maximize comparability between projects. After our preliminary screening, we had a set of 1191 software projects, where 450 came from a single company, and 741 from other companies. All have high data quality, and comparable definitions for size and effort.

The number of variables was also reduced to a small set (see Table 1) we believed could potentially have an impact on effort.

Summary statistics for the ratio-scale variables are presented in Table 2. They do not indicate significant differences between the SC and CC data, although the maximum values are higher in the CC data. SC projects were completed between 1996 and 2003, and CC projects were completed between 1991 and 2006.

3.2 Modelling techniques

We used a single technique (multivariate regression) to build the effort models, since it was not our aim to also compare the estimation accuracy between different techniques. The choice to use multivariate regression was motivated by it being the single technique employed in all previous studies comparing CC and SC models, where it either provided the best accuracy or was amongst the best Kitchenham et al. (2007).

All models used in this investigation were built using Kitchenham's manual stepwise regression (MSWR) method (Kitchenham 2004), using SPSS v.15. This method selects variables with a significant relationship to the dependent variable, and automatically deals with collinearity problems.

All remaining analyses in this paper were also carried out using SPSS, and statistical significance $\alpha = 0.05$.

Variable	Scale	Description
Effort	Ratio	Project effort in person hours
Ufp	Ratio	Application size in unadjusted function points
Maxteam	Ratio	Maximum team size
LangType	Nominal	Language type (e.g. 3GL, 4GL)
DevType	Nominal	New development, enhancement or re-development
Sector	Nominal	Business sector (e.g. Banking, manufacturing, insurance)
Platform	Nominal	The type of hardware the system was developed for (mainframe, midrange, PC, multi-platform)

Table 1: Variables used in this study

SC data – 450 projects					
Variables	Mean	Median	Std. Dev.	Min.	Max.
Ufp	493	277	632	6	6294
Effort	4270	2415	5372	62	57749
Maxteam	8.1	5	6.5	1	22
CC data – 741 projects					
Variables	Mean	Median	Std. Dev.	Min.	Max.
Ufp	473	234	998	4	16148
Effort	4947	2019	9752	26	134211
Maxteam	8.5	5.2	9.2	1	77

Table 2: Project summary statistics for the ratio-scaled variables

Before building the CC and SC models, it is important to ensure numerical variables are normally distributed, and independent variables have a reasonable relationship with effort (our dependent variable). Therefore, for every model built, we used the Shapiro-Wilk test on both training and testing sets to check if the numerical variables were normally distributed. The One-Sample Kolmogorov-Smirnov Test (K-S test) was also used for confirmation. In every case, Ufp and Effort were not normally distributed. Therefore, these variables were transformed to a natural logarithmic scale. Once transformed, their distributions were re-checked; the tests confirmed that $\ln(\text{Effort})$ and $\ln(\text{Ufp})$ were both normally distributed. The transformed variables' names were preceded by 'L'; so Effort became LEffort, and so on.

To verify the stability of each model we used the following steps (Kitchenham and Mendes 2004):

- Use a residual plot showing residuals vs. fitted values to investigate if the residuals are random and normally distributed.
- Calculate Cook's distance values (Cook 1977) for all projects to identify influential data points. Any projects with distances higher than $3 \times (4/N)$, where N represents the total number of projects, are immediately removed from the data analysis. Those with distances higher than $4/N$ but smaller than $(3 \times (4/N))$ are removed temporarily in order to test the model stability, by observing the effect of their removal on the model. If the model coefficients remain stable and the goodness of fit

¹ www.isbsg.org

² ISBSG's rules about confidentiality mean that we do not know the identity of this single company.

improves, the highly influential projects are retained in the data analysis (Maxwell 2002).

3.3 Accuracy measures

To date the three measures most commonly used in software engineering to compare different effort estimation techniques have been (Kitchenham et al. 2007):

- Mean Magnitude of Relative Error (Mean MRE).
- Median Magnitude of Relative Error (Median MRE).
- Prediction at level l (Pred(l)), which measures the percentage of estimates that are within l % of the actual values.

MRE is used as basis for calculating the Mean MRE (MMRE) and Median MRE (MdMRE):

$$\text{MRE} = \frac{|e - \hat{e}|}{e} \quad (1)$$

where e represents actual effort and \hat{e} estimated effort.

Conte et al. (1986) suggest that l should be set at 25% and that a good prediction system should offer this accuracy level 75% of the time. They also suggest that only MMREs up to 0.25 represent good prediction accuracy.

Another comparative measure is the Magnitude of Relative Error relative to the Estimate (EMRE) (Kitchenham et al. 2001). EMRE uses the estimate, not the actual value, as the divisor. Mean EMRE (MEMRE) and Median EMRE (MdEMRE) are summary measures of EMRE.

Kitchenham et al. (2001) showed that MMRE and Pred(l) are respectively measures of the spread and kurtosis of z , where ($z = \hat{e} / e$), and suggest that accuracy should be measured in terms of z . They note however that z has undesirable asymmetry (biased towards minimizing overestimates), and also suggest that simple residuals be used as an accuracy measure. They recommend that boxplots of z and boxplots of the residuals ($\hat{e} - e$) are useful alternatives to simple summary measures.

In this paper we use residuals and z to compare the effort models. We also report summary measures of MRE, EMRE and Pred(l), because they are so often used as evaluation criteria.

3.4 Splitting the data

To split the data sets chronologically, we sorted the projects by start date and chose a suitable start date such that reasonable numbers of CC and SC projects were available for training, and a reasonable number of SC projects were available to be validation sets.

The first split date chosen was 1 January 2001. This means that both CC and SC training sets only contained projects that had been completed prior to the 1st of January 2001; CC projects completed after that date were ignored, so the hold-out sample only contained SC projects initiated on or after the 1st of January 2001. With this splitting date, the CC and SC training sets, and the SC validation set contained 520, 200 and 161 projects respectively.

To form a random set for comparison, we sorted projects by ID (since ISBSG assigns project ID numbers

randomly this is effectively a random selection). The first 520 CC projects became the CC training set; the first 200 SC projects became the SC training set, and the remaining 139 SC projects whose effort was not altered by normalization became the holdout sample. (For this SC training set, projects whose effort was affected by up to 8% were accepted, as otherwise the training set would have been too small to use.)

As a second split date we chose 1 January 2002, by which time more projects in the repository had finished and fewer had not yet started. After applying this split, the CC training set contained 539 projects, the SC training set 312 projects, and the SC holdout sample contained 75 projects.

To form the second random set we followed the same approach as above, but this time sorted projects in descending order of ID. We selected 539 CC projects as a second CC training set, and 283 projects as a second SC training set. The holdout sample contained 68 projects.

4 Results

4.1 Building the estimation models

We begin by illustrating the process used to develop one estimation model: that with the training data consisting of 200 SC projects completed before 1st January 2001. All models were built and checked for suitability in the same way, but space limitation prevents full presentation of them all.

Variables with more than 40% missing values were not considered for inclusion in the model. Maximum team size had no values and so was ignored. The other variables all had sufficient data to be considered.

Language type had two levels: 3GL and 4GL. 3GL was most common. One dummy variable was created to represent language type: Lang4GL (1 if 4GL, 0 if not). No dummy variable was created for Lang3GL as it had the highest frequency.

The development type had two levels: new developments (most common) and enhancement. One dummy variable was created: DevEnh (1 for enhancement, 0 otherwise).

Similarly, three dummy variables were created for the platform type: PlatMF for mainframe projects, PlatMR for midrange projects, PlatPC for PC projects. No dummy variable was created for multi-platform projects, the most common type.

12 different business sectors are represented in the 200 single-company projects. Five sectors which each had 2 projects or fewer were merged into a single "Other" category. Seven dummy variables were created, representing Banking, Communication, Financial, Manufacturing, Construction, Service industry, and Other. No dummy variable was created for the most common value (Insurance).

The first variable selected by MSWR was LUfp, followed by LLang4GL and LSectorManufact. No other variables were significantly correlated with the residuals.

When the regression model was built, the residual plot for the 200 projects showed several that seemed to have large residuals. This trend was confirmed using Cook's distance. 18 projects had their Cook's distance above the cut-off point (4/200). When these projects were

temporarily removed and the regression line calculated again, adjusted R² declined from 0.384 to 0.365. As all coefficients remained similar, however, the projects were not removed from the data set.

Independent Variables	Coefficient	Std. Error	t	p> t
(constant)	4.794	0.303	15.814	0.000
LUfp	0.591	0.055	10.821	0.000
LLang4GL	-1.105	0.277	-3.995	0.000
LSectorManufact	-0.616	0.270	-2.281	0.024

Table 3: Best Fitting SC Model; projects split at 1 January 2001

The best model built from the SC training set is described in Table 3. Its adjusted R² is 0.384. When the equation of Table 3 is transformed to the raw data, this gives the Equation:

$$\text{Effort} = 120.783 \text{ ufp}^{0.59} e^{-1.105\text{Lang4GL}} e^{-0.616\text{SectorManufact}} \quad (2)$$

The plots shown in Figures 1 and 2 suggest that residuals are normally distributed.

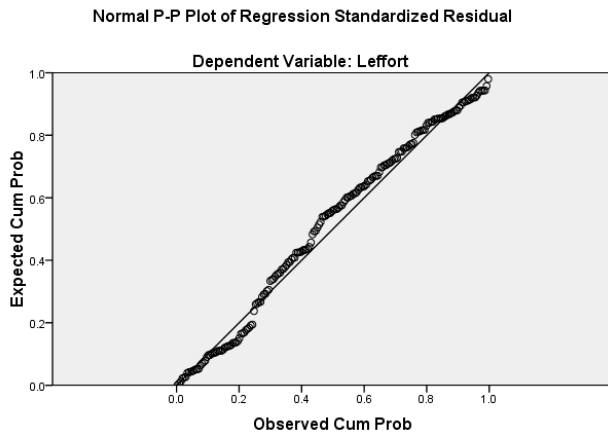


Figure 1: P-P plot for best SC model; projects split at 1 January 2001

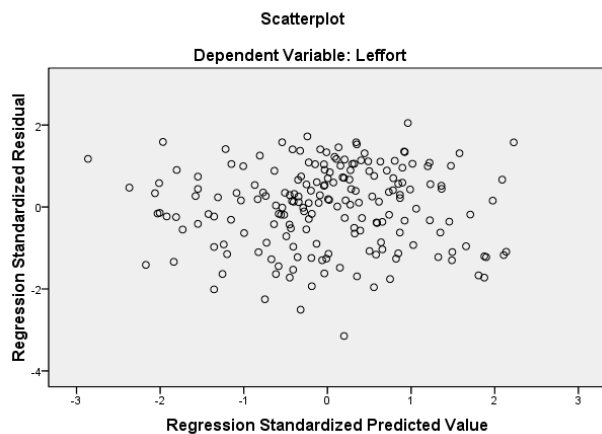


Figure 2: Residual plot for best SC model; projects split at 1 January 2001

Similar procedures were used to derive the other models:

CC, chronological split at 1st January 2001 (520 training projects; adjusted R² for the model is 0.719):

$$\text{Effort} = 26.762 \text{ ufp}^{0.504} \text{Maxteam}^{0.779} e^{-0.616\text{DevNew}} \quad (3)$$

SC, random split 1 (200 training projects; adjusted R² for the model is 0.395):

$$\text{Effort} = 110.057 \text{ ufp}^{0.589} e^{-0.932\text{Lang4GL}} e^{-1.334\text{SectorOther}} e^{-0.617\text{SectorManufact}} \quad (4)$$

CC, random split 1 (200 training projects; adjusted R² for the model is 0.714):

$$\text{Effort} = 26.259 \text{ ufp}^{0.509} \text{Maxteam}^{0.772} e^{0.474\text{DevNew}} \quad (5)$$

SC, chronological split at 1st January 2002 (312 training projects; adjusted R² for the model is 0.417):

$$\text{Effort} = 87.97 \text{ ufp}^{0.634} e^{-0.758\text{Lang4GL}} e^{-0.711\text{SectorManufact}} e^{-0.957\text{SectorOther}} \quad (6)$$

CC, chronological split at 1st January 2002 (539 training projects; adjusted R² for the model is 0.704):

$$\text{Effort} = 30.569 \text{ ufp}^{0.475} \text{Maxteam}^{0.794} e^{0.497\text{DevNew}} \quad (7)$$

SC, random split 2 (215 training projects; adjusted R² for the model is 0.532):

$$\text{Effort} = 47.181 \text{ ufp}^{0.768} e^{-1.269\text{SectorManufact}} e^{0.726\text{PlatMR}} e^{-1.439\text{SectorComms}} e^{-1.127\text{SectorOther}} e^{-0.795\text{SectorService}} \quad (8)$$

CC, random split 2 (539 training projects; adjusted R² for the model is 0.741):

$$\text{Effort} = 79.60 \text{ Maxteam}^{0.936} e^{-1.849\text{SectorManufact}} \text{ ufp}^{0.367} e^{-0.594\text{DevEnh}} e^{-0.703\text{PlatPC}} \quad (9)$$

4.2 Evaluating the estimation models

Each CC and corresponding SC estimation models were evaluated by applying them to each of the projects in the associated SC holdout set.

How Split	Eqn	Mean MRE	Md MRE	Pred(25)	Mean EMRE	Md EMRE
1 Jan 01	(2)	1.97	0.61	20.50	0.78	0.58
1 Jan 02	(6)	1.50	0.70	21.33	1.01	0.55
Rand 1	(4)	1.59	0.57	24.46	1.35	0.61
Rand 2	(8)	2.67	0.82	0.00	0.91	0.59

Table 4: Accuracy of SC models on SC holdout data

How Split	Eqn	Mean MRE	Md MRE	Pred(25)	Mean EMRE	Md EMRE
1 Jan 01	(3)	0.75	0.77	14.91	4.78	2.76
1 Jan 02	(7)	0.67	0.77	17.33	4.82	2.44
Rand 1	(5)	0.79	0.81	12.95	6.74	4.00
Rand 2	(9)	0.85	0.79	11.76	6.34	3.31

Table 5: Accuracy of CC models on SC holdout data

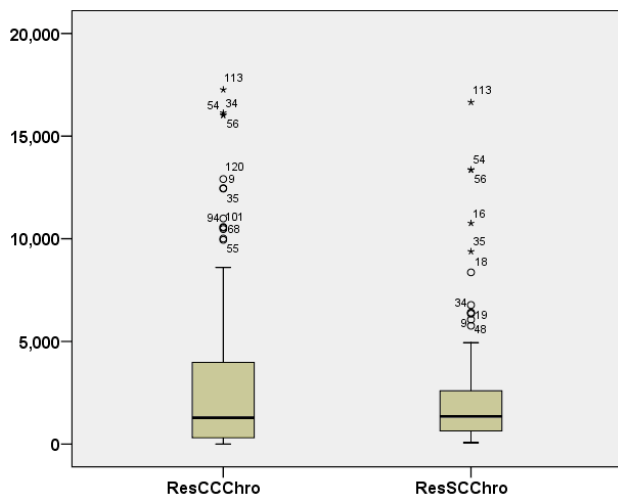


Figure 3: Absolute residuals for CC (left) and SC (right) models on SC data; split at 1 January 2001

Where all the necessary variables were available in the holdout set, the model was applied directly. If that was not possible (for example, the CC model featured Maxteam, which is missing for many SC projects in the holdout set), a solution was needed. To limit the holdout set to only those projects with known Maxteam would mean too small a sample. To build a model from the CC dataset that ignores Maxteam does not seem right either, as ignoring useful information is probably not what an estimator would do in practice. We chose to apply the full model to each holdout project; if a variable was not available, we replaced it in the Equation by the value 1.

Table 4 shows the performance of each SC estimation model when it is applied to each of the projects in its corresponding SC holdout set. Table 5 shows the performance of each CC estimation model when it is applied to each of the projects in its corresponding SC holdout set.

Tables 4 and 5 show that no model, whether CC or SC, seem to perform well. The best value of MMRE is 0.67, where 0.25 is the suggested threshold indicating good prediction accuracy (Conte et al. 1986). Pred(25) also falls well short of the 75% that is considered good. This confirms previous results (Mendes and Lokan 2008).

Tables 4 and 5 suggest that SC models are generally worse in MMRE, slightly better in MdMRE and Pred(25), and much better in MEMRE and MdEMRE.

To get a more detailed understanding it helps to look at actual residuals and z values, rather than just averages of relative errors. Figures 3 and 4 present boxplots of absolute residuals and z values respectively, for the first chronological split at 1 January 2001. Boxplots for the other chronological split and the two random splits are not shown; they show the same patterns.

Given that neither the absolute residuals nor z values were normally distributed, tests for significant differences were done using the nonparametric Wilcoxon Signed Ranks Test, and paired samples ($\alpha = 0.05$).

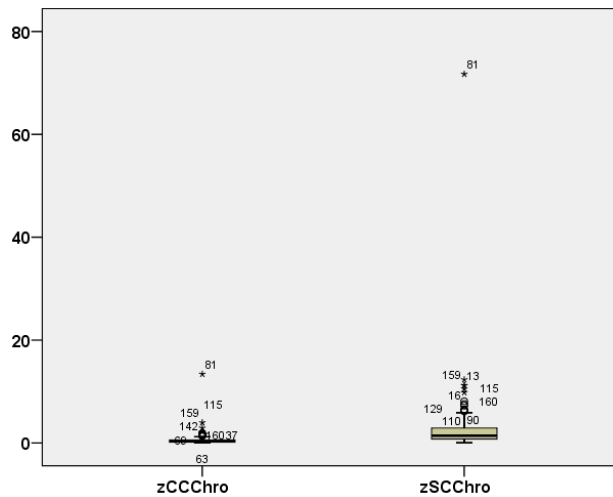


Figure 4: z values for CC (left) and SC (right) models on SC data; projects split at 1 January 2001

When comparing z values, the same result is found in both chronological splits and both random splits: SC predictions are significantly less accurate than CC predictions. When comparing absolute residuals, however, SC estimates were found to be significantly better in both chronological splits and the first random split; in the second random split, testing for significant differences found no significant differences between SC and CC predictions.

5 Answering Our Research Questions

Research questions 1a) and 2a) (see Section 1) relate to how well CC models predict single company projects. This is addressed by the results from Table 5: no model performed well.

Research questions 1b) and 2b) relate to comparing CC to SC models. Note that SC models also performed poorly. Here we carried out a statistical significance test comparing predictions between both models. When the significance tests were based on z values, in all cases the predictions obtained with CC models were significantly superior to predictions obtained using SC models. Conversely, results based on absolute residuals showed SC estimates to be significantly better than CC predictions for three sets (both chronological splits and the first random split), and not significantly different in the fourth set (second random split).

Z and absolute residuals are the accuracy measures suggested in (Kitchenham et al. 2001), but our results differ with these different measures. What might cause the difference?

MMRE and z favor minimizing overestimates, while EMRE favors the opposite. The patterns we observe in MMRE, EMRE and z with our models are what might be expected if the CC models are more likely to underestimate effort. Hence our results might be related to dataset characteristics or the research method that tends towards underestimates using the CC models. One possibility is that by ignoring missing terms in an estimation model we may be more likely to underestimate than overestimate effort: this would depend on whether the coefficients of the omitted variables for each

individual project are positive or negative. Whether it applies in our data is a matter for further investigation.

Conversely, absolute residuals are not biased towards either under- or over-estimate.

Given that our findings with z and absolute residuals conflict, and properties of z and our method may introduce bias, we believe that the results obtained using absolute residuals should take priority. This suggests that on these data, SC predictions were superior to CC predictions.

Research question 3) relates to whether accuracy differs between chronological or random splits. We used the Kruskal-Wallis Test to compare the z values and absolute residuals obtained from applying the CC and SC models to the holdout samples. In relation to CC models, there were no significant differences between either absolute residuals, or z values for three models (both chronological splits, and second random split), and between z values for the first random model. Residuals for the first random model were significantly worse than the other residual sets; however this was not considered enough evidence to suggest that the use of chronological splits affected the accuracy of CC models. In relation to the SC models, there were no significant differences between either absolute residuals, or z values for three models (both chronological splits, and second random split), and between residuals for the first random model. Z values for the first random model were significantly superior to the other z value sets; however this was also not considered enough evidence to suggest that the use of chronological splits affected the accuracy of SC models. Therefore, based on the data used in this study, we cannot reject the null hypothesis and therefore there is no difference in accuracy between random and chronological splits.

We noted in the introduction that with a chronological split insight can only be drawn from part of the time span. Hence one would expect the results with a chronological split to be worse than with a random split, so our results contradict what we expected; however they could have been influenced by the wide range of projects' ages used in our analysis. It is also possible that even with the first chronological split the training data set is large enough already to provide the same range of insights as a random split. As part of future work we intend to use training sets that are smaller but more homogenous as far as projects' ages are concerned.

Finally, research question 4) relates to whether different chronological split dates make a difference to accuracy. We compared absolute residuals and z values for the chronologically split CC and SC models, using the Mann-Whitney U on two independent samples. Except for the absolute residuals obtained using a SC model and the residuals for the first chronological split, there were no significant differences between z values and absolute residuals. Again we cannot reject the null hypothesis, and therefore, based on the data sets employed, the use of different date-based chronological splits does not produce significantly different results. Having more data to work with at the second chronological split has not led to a difference in accuracy. These results could have been caused by the use of splitting dates that were only one year apart. However, this was the only available option in

order to keep the training and testing data sets with reasonable sizes. Again, it is also possible that the training data set at the first chronological split already contains as much information as can be learned from the full data set, so adding more data at a second split date does not help.

Our results regarding the use of chronological split supported those found by Lokan and Mendes (2008), when using a different type of chronological split, and different data sets. What these results suggest in practice is that organizations that make use of cross-company data to help with their effort estimation and productivity benchmarking can use random allocation of projects to training and validation sets without hampering the results.

6 Conclusions

When comparing CC and SC predictions, we obtained conflicting results depending on the accuracy measure used – z values or absolute residuals. These findings suggest that different measures of accuracy can provide drastically different results, which may also suggest that had previous studies comparing CC to SC predictions used z values, instead of absolute residuals or MREs, the evidence gathered to date might have been quite different. Therefore, given that absolute residuals are symmetrical, we urge other researchers to use absolute residuals when comparing models, even if they also present results based on z values, MMRE, EMRE or Pred(l).

Based on the data used in this study, our results also suggest that the use of date-based chronological splits instead of random splits, and the use of different splitting dates, does not seem to affect the accuracy of either SC or CC estimates. This may be because the data set is sufficiently large and varied that the lack of information is not a problem.

Going back to our original aim of seeing how the simpler approach of chronological splitting compares to the time-consuming approach of building a separate model for each individual project, it seems that there is no loss to a researcher by using the simpler approach – at least with these large and varied data sets.

Future work includes repeating this experimental approach using more homogeneous and more complete data sets.

7 References

- Abrahamsson, P., Moser, R., Pedrycz, W., Sillitti, A., and Succi, G (2007): Effort Prediction in Iterative Software Development Processes – Incremental Versus Global Prediction Models, *Proceedings of ESEM 2007*, Madrid, IEEE Computer Society, pp 344-353, September.
- Auer, M. and Biffel, S.(2004): Increasing the Accuracy and Reliability of Analogy-based Cost Estimation with Extensive Project Feature Dimension Weighting, *Proceedings of ISESE'04*, pp 147-155, August.
- Auer, M., Trendowicz, A., Graser, B., Haunschmid, E., and Biffel, S. (2006): Optimal Project Feature Weights in Analogy-based Cost Estimation: Improvement and Limitations, *IEEE Trans. Software Engineering*. 32(2), pp 83-92, February.

- Conte, S.D., Dunsmore, H.E., and Shen, V.Y. (1986): *Software Engineering Metrics and Models*, Benjamin-Cummins.
- Cook, R.D. (1977): Detection of influential observations in linear regression, *Technometrics*, 19, pp 15-18.
- Kitchenham, B.A. (2004): A procedure for analysing unbalanced data sets, *IEEE Trans. Software Engineering*. 24(4), pp 278-301, April.
- Kitchenham, B.A., and Mendes, E. (2004): A Comparison of Cross-company and Within-company Effort Estimation Models for Web Applications, *Proceedings EASE 2004*, pp 47-55.
- Kitchenham, B.A., Mendes, E., and Travassos, G. (2007): Cross versus Within-Company Cost Estimation Studies: A Systematic Review, *IEEE Trans. Software Engineering*. 33(5), pp 316-329, May.
- Kitchenham, B.A., Pickard, L.M., MacDonell, S.G., and Shepperd, M.J. (2001): What accuracy statistics really measure, *IEE Proceedings - Software*, 148(3), pp 81-85, June.
- Lefley, M., and Shepperd, M.J. (2003): Using Genetic Programming to Improve Software Effort Estimation Based on General Data Sets, *Proceedings of GECCO 2003*, LNCS 2724, Springer-Verlag, pp 2477-2487, July.
- Lokan, C., and Mendes, E. (2006): Cross-company and Single-company Effort Models Using the ISBSG Database: a Further Replicated Study, *Proceedings of ISESE'06*, pp 75-84, September.
- Lokan, C., and Mendes, E. (2008): Investigating the use of Chronological Splitting to compare Software Cross-company and Single-company Effort Predictions, *Proceedings of EASE'08*, pp 151-160.
- MacDonell, S.G., and Shepperd, M.J. (2003): Using Prior-Phase Effort Records for Re-estimation During Software Projects, *Proceedings of 9th IEEE International Symposium on Software Metrics (Metrics'03)*, Sydney, September.
- Maxwell, K. (2002): *Applied Statistics for Software Managers*. Software Quality Institute Series, Prentice Hall.
- Mendes, E., and Lokan, C. (2008): Replicating Studies on Cross- vs. Single-company Effort Models using the ISBSG Database, *Empirical Software Engineering*, 13(1): 3-37. February.
- Mendes, E., Di Martino, S., Ferrucci, F., and Gravino, C. (2008): Cross-company vs Single-company Web Effort Models Using the Tukutuku Database: an Extended Study, *Journal of Systems and Software*, 81, pp 673-690, 2008.
- Premraj, R., Shepperd, M.J., Kitchenham, B.A., and Forselius, P. (2005): An Empirical Analysis of Software Productivity Over Time, *Proceedings of 11th IEEE International Symposium on Software Metrics (Metrics'05)*, Como, Italy, September.
- Premraj, R., and Zimmerman, T. (2007): Building Software Cost Estimation Models using Homogenous Data, *Proceedings of ESEM 2007*, Madrid, IEEE Computer Society, pp 393-400, September.
- Sentas, P., Angelis, L., Stamelos, I., and Bleris, G. (2005): Software productivity and effort prediction with ordinal regression, *Information and Software Technology*, 47, pp 17-29.

Appendix. Data selection criteria

We selected projects for inclusion in our analysis according to these criteria, designed to maximize data quality and comparability between projects:

- Remove projects for which the implementation date and/or overall project elapsed time is unknown.
- Remove projects if they were not assigned a high data quality rating (A or B) by ISBSG.
- Remove projects if their size is measured in lines of code, or in a version of function points other than IFPUG, or in an outdated version of function points (size measured with an older version is not directly comparable with size measured with IFPUG version 4.0 or later). Also remove projects which measured size with an unspecified version of function points, and whose completion pre-dated IFPUG version 4.0.
- Remove projects for which the size in unadjusted function points is unknown.
- Remove projects with for which the development team effort (resource level 1) is unknown. Only the development team's effort was used in our analysis.
- Remove CC and SC projects whose normalized effort differs from recorded effort. The only exception to this rule was one of the randomly allocated SC training sets, where we also kept projects in which normalized effort differed from reported effort by no more than 8%; the uncertainty introduced in the effort value for these projects is small (probably well under 8% for most projects) and this avoided the use of a very small training set.