An Empirical Evaluation of Effort Prediction Models Based on Functional Size Measures

Luigi Lavazza Sandro Morasca
Dipartimento di Scienze Teoriche e Applicate
Università degli Studi dell'Insubria
Varese, Italy
e-mail: {luigi.lavazza, sandro.morasca}@uninsubria.it

Gabriela Robiolo
Facultad de Ingeniería
Universidad Austral
Buenos.Aires, Argentina
e-mail: grobiolo@austral.edu.ar

***Abstract*— Software development effort estimation is among the most interesting issues for project managers, since reliable estimates are at the base of good planning and project control. Several different techniques have been proposed for effort estimation, and practitioners need evidence, based on which they can choose accurate estimation methods.**

The work reported here aims at evaluating the accuracy of software development effort estimates that can be obtained via popular techniques, such as those using regression models and those based on analogy.

The functional size and the development effort of twenty software development projects were measured, and the resulting dataset was used to derive effort estimation models and evaluate their accuracy.

Our data analysis shows that estimation based on the closest analogues provides better results for most models, but very bad estimates in a few cases. To mitigate this behavior, the correction of regression toward the mean proved effective.

According to the results of our analysis, it is advisable that regression to the mean correction is used when the estimates are based on closest analogues. Once corrected, the accuracy of analogy-based estimation is not substantially different from the accuracy of regression based models.

Keywords- Functional size measurement; function points; effort estimation; Regression Toward the Mean; Least Median of Squares.

I. INTRODUCTION

Several different types of models have been proposed for estimating the effort required to develop a software system whose functional size is known.

In this paper, we use a dataset of 20 projects to evaluate the accuracy of different estimation models. For each project in the dataset, we assume that the other 19 projects' sizes and effort data are known, and that the considered project development effort is estimated based on this data.

The model types considered are:

$$\text{Estimated\_Effort} = a + b \times \text{Size} \quad (1)$$

$$\text{Estimated\_Effort} = a\ \text{Size}^b \quad (2)$$

$$\text{Estimated\_Effort} = \text{Size} / \text{Productivity} \quad (3)$$

where Productivity is defined as the ratio Size/Effort; Effort is measured in person hours and Size is measured in Function Points [1][2].

Models of type (1) are obtained via Ordinary Least Square (OLS) and Least Median of Squares (LMS) linear regressions, while models of type (2) are obtained via OLS regression after log-log transformation. Models of type (3) are obtained using two different values of productivity:

a) Productivity_{CA}: the productivity of the projects that are Closest Analogues (CA) to the project to be estimated.
b) $\text{Productivity}_{RTM}$: the productivity obtained from Productivity_{CA} by correcting Regression Toward the Mean (RTM).

The goal of the paper is to evaluate the accuracy of software development effort estimates obtained via popular techniques, such as regression and analogy. These estimation techniques are among the most used by practitioners. Maybe LMS is not so widely used as the other ones, however, since its introduction [3], LMS has been used in several Empirical Software Engineering studies (a list appears in [4]). The great advantage of LMS for practitioners is that it takes the burden of dealing with outlier identification and exclusion away from the user. A disadvantage for practitioners is that LMS excludes half of the datapoints from the model, so that relatively large datasets are needed to apply it.

More sophisticated techniques were not considered because they have not yet achieved great popularity among practitioners. In fact, our paper is mainly directed to practitioners that have collected –often with some difficulty and effort– a set of historical data, and wonder what is the best way to use this data. Accordingly, we show how some popular estimation methods can be applied to historical datasets of average size, and what the accuracy of the resulting estimates is. This may help practitioners choosing among the many available types of effort estimation models.

The paper is organized as follows: Section 2 describes the dataset and illustrates the derivation of models for every project in the dataset, and the application of such models to get effort estimates. Section 3 evaluates the accuracy of the obtained estimates. Section 4 discusses the threats to the validity of this study. Section 5 accounts for related work. Finally, Section 6 discusses the results found, draws some conclusions and outlines future work.

II. MODEL BUILDING

The analysed projects are a superset of those described in [5]. Also, the data is available from the authors on request.

 ISBN: 978-1-61208-230-1

They were selected because they have the following characteristics:

a) Requirements specifications were documented in a homogeneous way, namely via use cases.
b) Use cases were completely implemented, therefore the effort employed in every project concerns the same overall activity, consisting in complete implementation.
c) The hours worked in each project were homogeneously and accurately registered.

Some projects were developed at Universidad Austral, in a software engineering undergraduate context, as an assignment, consisting in the development of a business application. The hours worked on programming were verified and measured by the team leader and by a professor, as this was one of the academic requirements. The other projects were developed in two different contexts: the System and Technology (S&T) Department at Universidad Austral and a CMM level 4 Company. The S&T Department develops software for the university and other parties, with a contractual relationship with the students, similar to that one they would have in a company. The hours worked on programming were obtained from the company registration files. The information about such hours was used in each company to do quality control or for future project estimation. The involved human resources in both environments shared a similar profile: advanced undergraduate students –who had been similarly trained– worked in academy, at the S&T Department and at the CMM level 4 Company.

Table I reports the values of Productivity_{M} (the mean productivity), Productivity_{CA}, and $\text{Productivity}_{RTM}$. These productivity values were computed for each project by taking into account the rest of the project data. So, for instance, ProductivityM for project 1 is given by the mean of the productivity values of projects 2 to 20.

Productivity_{CA} was computed as the mean productivity of the two projects having minimum size distance with respect to the considered project. In case three or more projects had the same distance, they were all considered. Let us consider project 5, which has size of 110 UFP: the closest analogue is project 14 (distant just 1 UFP), while projects 4 and 13 are at a distance of 3 UFP. In this case, ProductivityCA is given by the average of the productivities of projects 4, 13 and 14, i.e., Productivity values (113/285 + 107/348 + 111/242.5)/3 = 0.386 UFP/PersonHour.

Having computed Productivity_{CA}, it was then possible to check for the conditions under which the regression to the mean phenomenon is bound to occur. In our case, the RTM is expected to occur in Productivity_{CA} with respect to the actual productivity, which is given by the ratio size/effort.

The conditions for RTM are: a) the distributions of the actual productivity and Productivity_{CA} are normal, b) they have similar variance, and c) they are not perfectly correlated. In our case, all these conditions are satisfied since:

TABLE I. PRODUCTIVITY VALUES

ProjID	Actual	Mean	CA	RTM corrected
1	0.451	0.397	0.543	0.515
2	0.568	0.391	0.450	0.438
3	0.447	0.397	0.240	0.270
4	0.396	0.400	0.515	0.493
5	0.335	0.403	0.386	0.389
6	0.434	0.398	0.269	0.294
7	0.170	0.412	0.266	0.294
8	0.296	0.405	0.072	0.136
9	0.867	0.375	0.786	0.707
10	0.658	0.386	0.525	0.498
11	0.878	0.375	0.553	0.519
12	0.161	0.412	0.236	0.270
13	0.307	0.405	0.317	0.333
14	0.458	0.397	0.389	0.391
15	0.133	0.414	0.179	0.224
16	0.401	0.400	0.373	0.378
17	0.361	0.402	0.386	0.389
18	0.595	0.389	0.256	0.281
19	0.037	0.419	0.072	0.139
20	0.039	0.419	0.068	0.135

a) the Shapiro-Wilk test applied to Productivity_{CA} and the actual productivity rejects the hypothesis of non-normality (p-value > 0.4 for both distributions);
b) the standard deviations of the two distributions are similar (being 0.24 and 0.19);
c) the Pearson correlation factor r is 0.808 (p-value $< 10^{-3}$). Accordingly, the percent of regression effects –computed via the formula $(1 - r) \times 100$– is 19.2%.

Even though the effect of regression toward the mean is not extremely relevant, we wanted to check whether RTM could still provide good results (as in [6]) or even better ones, when compared to other techniques. Moreover, some values of Productivity_{CA} are quite far from the actual productivity: it is thus worthwhile trying RTM correction to see if such deviations can be eliminated. To this end, we applied a correction formula suggested by Campbell and Kenny [7]:

$$\text{Productivity}_{RTM} = \text{Productivity}_{CA} + (1\text{-r}) \times (\text{Productivity}_{M} - \text{Productivity}_{CA}) \quad (4)$$

The resulting values of $\text{Productivity}_{RTM}$ are given in the rightmost column of Table I: it is easy to see that the RTM correction decreases high values of productivity and increases the small ones. This is exactly what RTM correction is expected to do. We shall evaluate if such correction actually improves the accuracy of estimates.

We also computed Effort vs. Size models using OLS regression, both with and without log-log transformation, and using LMS regression. As for productivity, each project's model was derived by excluding the project's data from the dataset.

When deriving the models via OLS regression, we excluded outliers according to Cook's distance [8]. Cook's distance is commonly used to identify projects that jointly exhibit a large influence and large residual. Projects with Cook's distance greater than 4/n, where n represents the total number of projects, are considered to have a high influence on the results [9].

This explains why some projects are associated with the same model. Consider for instance project 7: during the computation, Cook's distance of project 8 suggests that project 8 be excluded from the dataset. Similarly, project 7 is an outlier for project 8, according to Cook's distance. Therefore, the models for projects 7 and 8 are computed over the same dataset, which excludes the data from projects 7 and 8.

LMS linear regressions compute the model using only half the available data, thus it is quite expected that several projects share the same model.

We applied the models found to get effort estimates. The resulting estimates are given in Table II.

TABLE II. EFFORT ESTIMATES

ID	Actual Effort	Estimates				
		Lin. OLS	LogLog OLS	Linear LMS	CA	RTM corr.
1	–	322	351	341	341	360
2	–	433	494	405	598	614
3	–	321	332	331	712	633
4	285	279	285	287	220	229
5	328	237	232	216	285	283
6	198	208	196	270	320	293
7	442	190	175	235	282	255
8	723	408	405	451	2972	1574
9	392	605	586	647	433	481
10	–	369	369	397	341	359
11	131	263	271	297	208	222
12	1042	336	334	380	712	623
13	348	230	225	211	338	321
14	243	279	283	285	285	284
15	300	107	106	181	223	178
16	147	169	144	250	158	156
17	169	168	136	251	158	157
18	121	196	182	230	282	256
19	16809	1049	951	1087	8600	4484
20	5221	387	385	431	2972	1488

A 0.05 statistical significance threshold was used throughout the paper, as is customary in Empirical Software Engineering studies. All the results reported in the paper are characterized by p-value < 0.05. All validity requirements for the proposed models (e.g., the normal distribution of residuals of OLS regressions) were rigorously verified.

III. EVALUATION OF MODELS

After having obtained the estimates for all projects using the different models (Table II), we computed the estimation errors as the differences between the actual and estimated efforts.

With effort estimation, the size of an error is possibly not as relevant as its relative size. For instance, a 10 PersonMonth error is generally more easily accepted for a 200 PersonMonth project than a 4 PersonMonth error is considered acceptable for a 12 PersonMonth project. In fact, even though the former error is two and a half times the latter, it is just 5% of the entire effort, while in the second case it is 33%.

Accordingly, we computed the relative errors of the estimates, and reported them in Table III. Table III shows that the biggest errors occur with the estimation based on analogy. It is noticeable that the four biggest (in absolute terms) errors with Productivity_{CA} (concerning projects 8, 18, 3 and 6) are effectively reduced by the RTM correction. However, RTM correction has also the effect of increasing the estimation error for some projects (see, for instance, projects 11 and 15).

TABLE III. ESTIMATION RELATIVE ERRORS

ID	Linear OLS	Loglog OLS	Linear LMS	CA	RTM corr.
1	–	-14%	-17%	-17%	-12%
2	–	4%	-14%	26%	30%
3	–	-13%	-13%	86%	65%
4	-2%	0%	1%	-23%	-20%
5	-28%	-29%	-34%	-13%	-14%
6	5%	-1%	36%	62%	48%
7	-57%	-60%	-47%	-36%	-42%
8	-44%	-44%	-38%	311%	118%
9	54%	50%	65%	10%	23%
10	–	35%	46%	25%	32%
11	101%	107%	127%	59%	69%
12	-68%	-68%	-64%	-32%	-40%
13	-34%	-35%	-39%	-3%	-8%
14	15%	17%	18%	18%	17%
15	-64%	-65%	-40%	-25%	-40%
16	15%	-2%	70%	7%	6%
17	-1%	-20%	49%	-7%	-7%
18	62%	50%	90%	133%	111%
19	-94%	-94%	-94%	-49%	-73%
20	-93%	-93%	-92%	-43%	-71%

The distribution of errors is represented via boxplots in Figure 1, where the errors concerning projects 19 and 20 are omitted. As shown in Table II, all models estimate these projects with large errors. Including them in the boxplot would have resulted in squeezing the plots, thus making them hardly readable.

The distribution of relative errors is represented via boxplots in Figure 2. The mean value of errors is represented as a diamond on each boxplot. It is easy to see that the estimation based on the closest analogues provides good accuracy with respect to other models, except for a single project (project 8). It is also quite evident that the RTM correction eliminates such anomaly, though negative errors worsen slightly.

To fully appreciate the differences in accuracy, it is useful to look also at absolute relative errors, which are illustrated by the boxplots in Figure 3. It is also quite apparent in Figure 3 that the RTM correction eliminates the problem of huge relative estimation errors, at the expense of a higher median absolute relative error.

However, the effects of RTM correction should be evaluated by taking into consideration the effects on the projects (19 and 20) that required the biggest effort. By looking at Table II, it is easy to see that the effort of such projects is estimated much better by analogy without correction than with RTM correction. The fact that the corrections concerning these projects are relatively smaller than others (e.g. the one concerning project 8) does not imply that they are acceptable. Actually, all the models based on regression consider projects 19 and 20 outliers, thus excluding them from the models. Estimation based on closest analogues is the only way of taking into account these projects, but ,unfortunately, RTM corrections in these cases operate in the wrong direction, decreasing estimates that are already underestimated. In conclusion, we must be aware that projects (like 19 and 20) that feature quite unusual productivity values, can reduce the effectiveness of RTM correction.

Concerning RTM correction, our results are similar to those reported in the literature. In particular, the MMRE and MdMRE [10] for our set of projects (see Table IV), are close to those reported in [6] and [11].

Table V summarizes the results of some representative papers. It can be seen that our results (given in Table IV), are in line with these studies.

TABLE IV. MMRE AND MDMRE OF ANALOGY BASED ESTIMATES

	CA	RTM
MMRE	49.3%	42.3%
MdMRE	25.5%	36.0%

TABLE V. ACCURACY OF ESTIMATES REPORTED IN THE LITERATURE

Ref.	Results	
	MMRE	MdMRE
[12]	[23.2 -51.1]	[14.8 – 48.0]
[13]	[36.15 – 73.85]	[14.23 – 44.95]
[14]	[11.3 – 32.8]	[7. 2– 24.3]
[15]	[32.82 – 82.20]	[20.44 – 50.54]
[16]	N.A.	[26 – 85]

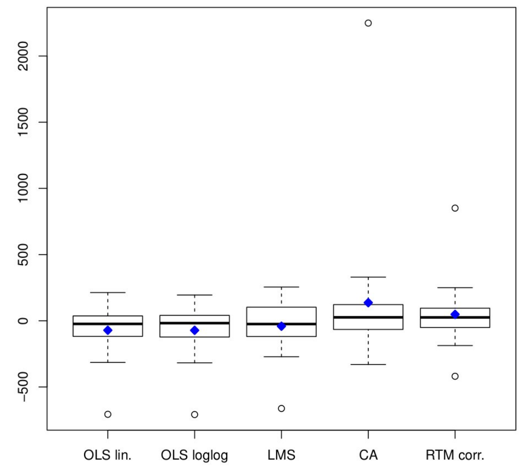

Figure 1. Boxplot of errors.

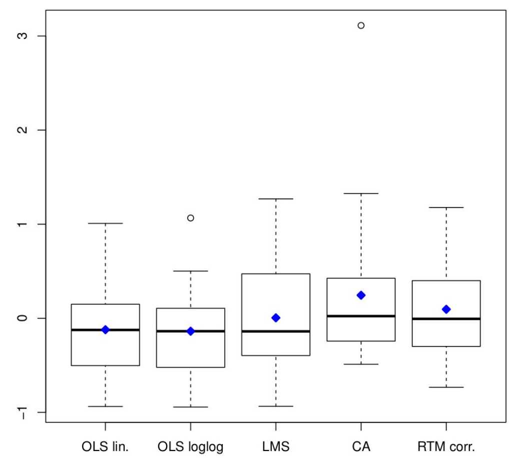

Figure 2. Boxplot of relative errors.

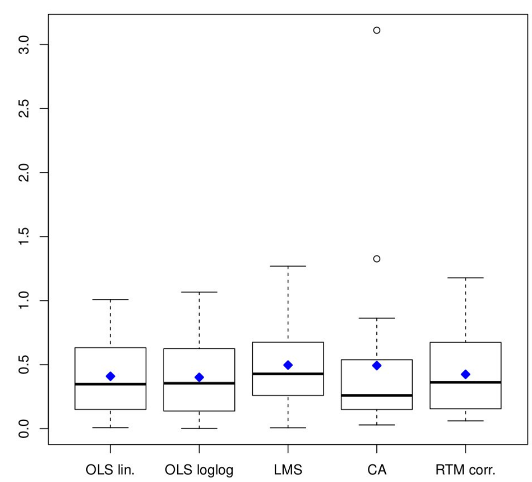

Figure 3. Boxplot of absolute relative errors.

IV. THREATS TO VALIDITY

Some of the projects that originated the dataset were carried out in industry, while some others were carried out in academia. So, treating all these projects as a single class of projects could be inaccurate, in principle. To make data as homogeneous as possible, academic developments were organized and conducted as industrial ones.

During the construction of models, we tested alternate ways of searching for analogue projects; one included the usage of projects carried out in the same environment. In such case, we also used different Productivity_M to correct RTM. So, for instance, the productivity of academic projects was estimated on the basis of the academic projects of similar size. Then RTM was corrected using in equation (4) the mean productivity of academic projects only. However, taking into account the development environment did not change much the results presented in Section III.

Another issue that deserves attention is the size of the considered projects: only three projects are substantially bigger than 200 FP. Accordingly, practitioners have to be cautious when applying the results reported in this paper to larger projects.

V. RELATED WORK

The phenomenon of “regression towards the mean” (RTM) is thoroughly described in [7]. RTM occurs where the estimation model is inaccurate and extreme observations appear, i.e., the values of the attribute of interest are much higher or lower than the expected value. The presence of these “extreme” values calls for the correction of regression models. Several adjustment approaches were proposed by Campebell and Kenny [7]. Jørgensen et al. were the first authors who described the occurrence of RTM in the context of software effort estimation and used one of the adjustment approaches by Campebell and Kenny [7] to evaluate five data sets [6]. They showed that analogy based effort estimates can be significantly improved through RTM-adjustments. Jørgensen et al. also hypothesized that, in cases with less extreme analogues and more accurate estimation models, there would be an expected improvement, if the underlying assumptions of the RTM-adjustment are met. However, they did not prove this hypothesis in [6].

Shepperd and Cartwright [11] performed an independent replication of the study by Jorgensen et al.: they used two further industrial data sets in which they compared accuracy levels with and without the RTM adjustment. Their results were consistent with those reported in [6], as using the RTM resulted in a small increase in predictive accuracy. However, for one data set it was necessary to first partition it into more homogeneous subsets. Their results added further support for the RTM approach, in that there is a small, but positive, effect upon prediction accuracy.

The RTM adjustment was improved by using the Model Tree adaptation strategy. Model Tree based attribute distance was proposed by Azzeh to adjust estimation by analogy and derive new estimates [17]. This is advantageous because it deals with categorical attributes, minimizes user interaction and improves the efficiency of model learning through classification. The experimental results showed that the proposed approach produced better results when compared to those obtained by using analogy based linear size adaptation, linear similarity adaptation, 'regression towards the mean' and null adaptation. However, this approach may only be applied to complex data sets with large number of categorical attributes.

The interest of finding analogies arises when historical data sets are available, thus making it possible to look for projects that are “similar” to the one for which an estimate is required. Similarity is defined as Euclidean distance in n-dimensional space where n is the number of considered project features. Each dimension is usually standardized, so that all dimensions have equal weight. The known effort values of the nearest neighbors to the new project are then used as the basis for the prediction. Shepperd and Schofield argued that estimation by analogy is a viable technique that can be used by project managers to complement current estimation techniques [18].

Several papers propose improvements of estimation based on closest analogues method. Chiu and Huan [19] investigated the effects on estimates obtained when a genetic algorithm method is adopted to adjust historical effort based on the similarity of distances between pairs of projects. The empirical results obtained using two data sets of 23 and 21 projects each showed that applying a suitable linear model to adjust the analogy-based estimations is a feasible approach to improve the accuracy of software effort estimates. A project selection technique for analogy-based estimations (PSABE), was then added to reduce the whole project base into a small subset that consists only of representative projects. The experimental results showed that applying the genetic algorithm to determine suitable weighted similarity measures of software effort drivers is a feasible approach to improve the accuracy of software effort estimates analogy-based. They also demonstrated that the nonlinearly weighted analogy method has better estimate accuracy than those obtained by using other methods [20].

Li and Ruhe [21] pointed out that a careful selection and weighting of attributes may improve the performance of the estimation methods. They considered the impact of weighting (and selecting) attributes as extensions of their estimation by analogy method AQUA+. With AQUA+ a qualitative analysis pre-step using rough set analysis –a machine learning technique for classification of objects– is performed to weight attribute. They reported that AQUA+ can improve the estimation accuracy, according to the empirical studies performed with six data sets.

Mittas, Athanasiades and Angelis [22] exploited the relationship between the estimation by analogy method and the nearest neighbor non-parametric regression technique in order to suggest a resampling procedure, known as iterated bagging, to reduce the prediction error. The positive effect of iterated bagging on estimation by analogy was validated using both artificial and real datasets from the literature.

Azzeh, Neagu, and Cowling [13] proposed a new formal estimation by analogy model based on the integration of the Fuzzy set theory with the Grey Relational Analysis (GRA) to overcome the inherent uncertainty in software attribute measurement. The Fuzzy set theory provides a representation scheme and mathematical operations to deal with uncertain, imprecise and vague concepts. GRA is a problem solving method that is used to assess the similarity between two tuples with M features, which is mainly used to reduce the uncertainty in the distance measurement between two software projects, for both continuous and categorical features. Both techniques are suitable when the relationship between effort and other effort drivers is complex. Experimental results showed that using integration of GRA with Fuzzy logic produced credible estimates, when compared to the results obtained using Case-Based Reasoning, Multiple Linear Regression and Artificial Neural Networks methods. In another paper [12], the same authors proposed a new approach to deal with each attribute, which has different influence on the project retrieval, based upon the idea of Kendall's coefficient of concordance between the similarity matrix of project attributes and the similarity matrix of known effort values of the dataset. The results showed improved prediction accuracy when multiple project attributes are used with certain weights. Moreover, they integrated analogy-based estimation with Fuzzy numbers in order to improve the performance of software project effort estimation during the early stages of a software development lifecycle, using all the available early data [14]. Particularly, this paper proposes a new software project similarity measurement and a new adaptation technique based on Fuzzy numbers. The results have also shown that the proposed method outperforms some well known estimation techniques, such as case-based reasoning and stepwise regression.

To overcome the inherent uncertainties of the estimation process, Li, Xie, and Goh focused on the generation of interval based estimates with a certain probability [15]. They proposed a novel method named Analogy Based Sampling (ABS) and compared it to the well established Bootstrapped Analogy Based Estimation method (BABE), which is the only existing variant of the analogy based method which has the capability to generate interval predictions. The results and comparisons showed that ABS could improve the performance of BABE with much higher efficiency and more accurate interval predictions. In another paper [23] they proposed a genetic algorithm to simultaneously optimize the K parameter and the feature weights for. Experiment results showed that their methods could significantly improve the prediction accuracy of conventional ABE.

Walkerden and Jeffery [24] stated that Analogy-based estimation is potentially easier to understand and apply than algorithmic methods. They compared several methods of analogy-based software effort estimation with each other and also with a simple linear regression model. The results showed that people are better than tools at selecting analogues. In particular, estimates based on selections made by people, with a linear size adjustment to the analogue's effort value, proved more accurate than estimates based on analogues selected by tools, and also more accurate than estimates based on the simple regression model.

However, controversial results were reported on this subject. Briand, Langley and Wieczorek [16] agree with Stensrud and Myrtveit [25] that estimation by analogy does not outperform regression models. Myrtveit and Stensrud [26] suggested that the results are sensitive to the experimental design, including dataset characteristics, criteria for removing outliers, test metrics, significance levels, the involvement of people, etc. Also, they pointed out that neither their results nor previous ones were robust enough to claim general validity.

Similarly, Mair and Shepperd found that there is approximately equal evidence in favour of and against analogy-based methods [27].

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have tested a few estimation models, using an experimental dataset. To the best of our knowledge, this is the first time that an effort prediction accuracy comparison is performed for a set of methods including OLS models, LMS models, and analogy-based methods both with and without RTM correction.

By looking at the results given in the tables and boxplot in Section III, it is possible to conclude that all the considered models yield similar performances, as far as estimation accuracy is concerned. Actually, the model based exclusively on analogy is slightly less precise then the others, but RTM correction makes the precision of analogy based estimation very close to that of regression models. It is also possible to consider RTM corrected models preferable over those based on regression because the median is closer to zero (Figure 2).

In conclusion, we can say that our results are of interest for practitioners, especially considering that a small dataset – i.e., a dataset very similar to the datasets that can be collected in most development environment– and popular techniques were used.

In the future, we aim at gaining a deeper theoretical understanding of, and generalizing the results presented here by studying larger projects, possibly involving additional effort-related factors, like product complexity and factors depending on the development environment.

ACKNOWLEDGEMENT

The research presented here was partially funded by the project "Metodi, tecniche e strumenti per l'analisi, l'implementazione e la valutazione di sistemi software" funded by Università degli Studi dell'Insubria, and by the Research Fund of Austral University School of Engineering.

REFERENCES

[1] A. Albrecht, Measuring application development productivity, IBM Application Development Symp. I.B.M. Press, 1979.

[2] ISO. 2003. ISO/IEC 20926: 2003, Software engineering – IFPUG 4.1 Unadjusted functional size measurement method – Counting Practices Manual.

[3] P. J. Rousseeuw and A. M Leroy, Robust regression and outlier detection, John Wiley & Sons, Inc., New York, 1987.

[4] L. Lavazza and S. Morasca, "Using a Generalized Robust Linear Regression Technique to Predict Effort in Software Engineering Projects", 16th Int. Conf. on Evaluation and Assessment in Software Engineering (May 2012).

[5] L. Lavazza and G.Robiolo, “The Role of the Measure of Functional Complexity in Effort Estimation”, 6th Int. Conf. on Predictive Models in Software Engineering, ACM, New York, NY, USA, Article 6, 10 pages, 2010.

[6] M. Jørgensen, D. I. K. Sjøberg and U. Indahl, “Software effort estimation by analogy and regression toward the mean”, Journal of Systems and Software 68, 3 (Dec. 2003), pp. 37-46

[7] D. Campbell and D. Kenny, A primer on regression artifacts. The Guilford Press, 2002.

[8] R. D. Cook and S. Weisberg, Residuals and Influence in Regression. Chapman and Hall, London, 1982.

[9] E. Mendes, Cost Estimation Techniques for Web Projects. IGI Publishing, Hershey, New York, 2008.

[10] S. Conte, H. Dunsmore, and V. Shen, Software Engineering, Metrics and Models. Benjamin/Cummings, 1986.

[11] M. Shepperd, and M. Cartwright, “A replication of the use of regression towards the mean (R2M) as an adjustment to effort estimation models”, 11th IEEE Int. Symp. on Software Metrics (Como, Italy, Sept. 19-22 2005), METRICS 2005, pp. 38-47.

[12] M. Azzeh, D. Neagu, and P. Cowling, “Software effort estimation based on weighted fuzzy grey relational analysis”, 5th Int. Conf. on Predictor Models in Software Engineering (PROMISE '09). ACM, New York, NY, USA, 2009.

[13] M. Azzeh, D. Neagu, and P. Cowling, “Fuzzy grey relational analysis for software effort estimation”, Empirical Software Engineering, 15, 1. Springer, 2010.

[14] M. Azzeh, D. Neagu and P. Cowling,“Analogy-based software effort estimation using Fuzzy numbers”, Journal of Systems and Software, 84, 2, Elsevier, 2011, pp. 270-284.

[15] Y.F. Li, M. Xie, and T.N. Goh, “A Study of Analogy Based Sampling for interval based cost estimation for software project management”, 4th IEEE Int.Con. on Management of Innovation and Technology (Bangkok, 2008). ICMIT 2008, pp. 281 – 286.

[16] L.C. Briand, T. Langley and I. Wieczorek., “A replicated assessment and comparison of common software cost modeling techniques”, in Proceedings of the 22nd international conference on Software engineering (ICSE '00), ACM, New York, NY, USA, pp. 377-386, 2000, doi: 10.1145/337180.337223.

[17] M. Azzeh, “Model Tree Based Adaption Strategy for Software Effort Estimation by Analogy”, 2011 IEEE 11th International Conference on Computer and Information Technology (CIT), Pafos, Sept. 2 2011, pp. 328 – 335.

[18] M. Shepperd, C. Schofield, "Estimating Software Project Effort Using Analogies," IEEE Trans. on Software Eng., vol. 23, no. 11, pp. 736-743, Nov. 1997.

[19] N.H Chiu and S.J. Huang, “The adjusted analogy-based software effort estimation based on similarity distances”, Journal of Systems and Software, 80, 4 (April 2007), pp. 628-640.

[20] Y.F. Li, and M. Xie, and TN. Goh, “A study of project selection and feature weighting for analogy based software cost estimation”, Journal of Systems and Software, 82, 2, Elsevier, pp. 241-252.

[21] J. Li, and G. Ruhe, “Analysis of attribute weighting heuristics for analogy-based software effort estimation method AQUA+”, Empirical Software Engineering, 13, 1, Springer, 2008.

[22] N.Mittas, M. Athanasiades, and L. Angelis. “Improving analogy-based software cost estimation by a resampling method”, Information and Software Technology, 50, 3, Elsevier, 2008.

[23] Y.F Li., M. Xie, and T.N. Goh, “Optimization of feature weights and number of neighbors for Analogy based cost Estimation in software project management”, IEEE Int. Con. on Industrial Engineering and Engineering Management (Singapore, 2008) IEEM 2008, pp. 1542 – 1546.

[24] F. Walkerden and R. Jeffery, “An Empirical Study of Analogy-based Software Effort Estimation”, Empirical Softw. Engg. 4, 2 (June 1999), pp. 135-158, doi:10.1023/A:1009872202035.

[25] E. Stensrud and I. Myrtveit, “Human Performance Estimating with Analogy and Regression Models: An Empirical Validation”, in Proceedings of the 5th International Symposium on Software Metrics (METRICS '98), IEEE Computer Society, Washington, DC, USA, 1998, pp. 205-.

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

[27] C. Mair, M. J. Shepperd, “ The consistency of empirical comparisons of regression and analogy-based software project cost prediction”, in 2005 International Symposium on Empirical Software Engineering (ISESE 2005), 17-18 November 2005, Noosa Heads, Australia, pp. 509-518, IEEE, 2005.