The Effects of Gradual Weighting on Duration-Based Moving Windows for Software Effort Estimation

Sousuke Amasaki[1] and Chris Lokan[2]

[1] Okayama Prefectural University,
Department of Systems Engineering, Soja, Japan
amasaki@cse.oka-pu.ac.jp
[2] UNSW Canberra,
School of Engineering and Information Technology, Canberra, Australia
c.lokan@adfa.edu.au

Abstract. Several studies in software effort estimation have found that it can be effective to use a window of recent projects as training data for building an effort estimation model. Windows can be defined as having a fixed size (containing a fixed number of projects), or as having a fixed duration. A recent study extended the idea of windows, by weighting projects differently according to their order within the window, and found that weighted moving windows could significantly improve estimation accuracy. That study used fixed-size windows. This study examines the effect on effort estimation accuracy of weighted moving windows that are based on fixed duration. We compare weighted and unweighted moving windows under the same experimental settings. Weighting methods are found to improve estimation accuracy significantly in larger windows, and the methods also significantly improved accuracy in smaller windows in terms of MRE. This result contributes further to understanding properties of moving windows.

1 Introduction

A software effort estimation model is developed from past project data. Most studies evaluate a model's accuracy with a hold-out or cross-validation approach. These approaches split project data into training data and testing data randomly.

In reality, software projects can be ordered chronologically. Using past projects as training data to predict future projects, instead of forming training and testing sets randomly, is more reasonable. Intuitively, it also seems appropriate to use only recent projects as a basis of effort estimation, because older projects might be less representative of an organization's current practices.

Lokan and Mendes [1, 2] examined whether using only recent projects improves estimation accuracy. They used a window to limit the size of training data so that an effort estimation model uses only recently finished projects. As new projects are completed, old projects drop out of the window. They used two types of window policies: fixed-size and fixed-duration. A fixed-size window policy determines the window size by the number of projects: the training

A. Jedlitschka et al. (Eds.): PROFES 2014, LNCS 8892, pp. 63–77, 2014.

set is the last N projects to finish before the target project starts. A fixed-duration policy determines the window size by calendar months: the training set is projects whose whole life cycle occurred during the last w months before the target project starts. Intuitively, we believe that a fixed-duration policy makes more sense: that estimators are more likely to think of "recent projects" in terms of calendar time rather than a given number of projects.

Lokan and Mendes found that estimation accuracy could improve by using either window policy, but the policies affected the accuracy differently.

Their studies assumed that projects within a window are all equally useful as training data. However, the chronological order of projects can be exploited further, by giving projects different importance according to their relative age to the target project, so that recent projects receive higher importance than older projects. Amasaki and Lokan examined this idea, and found that weighting the importance of training projects according to their order within the window of most recent projects affected estimation accuracy [3]. However, that study only used the fixed-size window policy.

In this paper, we turn to the fixed-duration policy, and explore the effects of weighted moving windows for software effort estimation with this approach. We address the following questions:

RQ1. Is there a difference in the accuracy of estimates between unweighted and weighted moving windows, when the definition of window size is based on duration?

RQ2. Can insights be gained from difference of trends in accuracy among weighted and unweighted moving windows as the window size varies?

RQ3. How do these results compare with results based on fixed-size windows (windows containing a fixed number of projects)?

2 Related Work

Research in software effort estimation models has a long history. However, few software effort estimation models were evaluated with consideration of the chronological order of projects.

Auer and Biffl [4] evaluated dimension weighting for analogy-based effort estimation, considering the effect of a growing data set. However, the authors used datasets having no date information. Thus, this evaluation method did not consider chronological order.

Mendes and Lokan [5] compared estimates based on a growing portfolio with estimates based on leave-one-out cross-validation, using two different data sets. In both cases, cross-validation estimates showed significantly superior accuracy. With cross-validation, all other projects in the data set — even some that were still in the future — are used as training data for a given project. Thus estimates using cross-validation are based on unrealistic information. If estimates based on unrealistic information are significantly more accurate than estimates considering chronology (based on realistic information), the implication is that

the apparent accuracy achieved when ignoring chronology does not reflect what an estimator would achieve in practice.

Some studies such as [6, 7] used a project year in software effort estimation model construction. However, these studies did not consider chronological order in evaluation. Maxwell [8] demonstrated the construction and evaluation of a software estimation model with the consideration of chronology. A candidate effort estimation model selected a year predictor. She also separated project data into training and test data according to a year.

Lokan and Mendes [1] studied the use of moving windows with linear regression models and a single-company dataset from the ISBSG repository. Training sets were defined to be the N most recently completed projects. They found that the use of a window could affect accuracy significantly; predictive accuracy was better with larger windows; some window sizes were 'sweet spots'. Later they also investigated the effect on accuracy when using moving windows of various durations to form training sets on which to base effort estimates [2]. They showed that the use of windows based on duration can affect the accuracy of estimates, but to a lesser extent than windows based on a fixed number of projects.

This study builds on both [2] and [3]. The same data set is investigated again. This study extends [2] by exploring the use of weighting functions. It differs from [3] in using duration as the basis for defining window size.

3 Research Method

3.1 Dataset Description

The data set used in this paper is the same one analyzed in [1–3]. This data set is sourced from Release 10 of the ISBSG Repository. Release 10 contains data for 4106 projects; however, 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 in which all projects provided the necessary data for size, effort and chronology, defined size and effort similarly, and had high quality data, we removed projects according to the following criteria:

- The projects are rated by ISBSG as having high data quality (A or B).
- Implementation date and overall project elapsed time are known.
- Size is measured in IFPUG 4.0 or later (because size measured with an older version is not directly comparable with size measured with IFPUG version 4.0 or later). We also removed projects that measured size with an unspecified version of function points, and whose completion pre-dated IFPUG version 4.0.
- The size in unadjusted function points is known.
- Development team effort (resource level 1) is known. Our analysis used only the development team's effort.
- Normalized effort and recorded effort are equivalent. This should mean that the reported effort is the actual effort across the whole life cycle.
- The projects are not web projects.

Table 1. Summary statistics for ratio-scaled variables

Variable	Mean	Median	StDev	Min	Max
Size	496	266	699	10	6294
Effort	4553	2408	6212	62	57749
PDR	16.47	8.75	31.42	0.53	387.10

In the remaining set of 909 projects, 231 were all from the same organization and 678 were from other organizations. We only selected the 231 projects from the single organization, as we considered that the use of single-company data was more suitable to answer our research questions than using cross-company data. Preliminary analysis showed that three projects were extremely influential and invariably removed from model building, so they were removed from the set. The final set contained 228 projects.

We do not know the identity of the organization that developed these projects.

Release 10 of the ISBSG database provides data on numerous variables; however, this number was reduced to a small set that we have found in past analyses with this dataset to have an impact on effort, and which did not suffer from a large number of missing data values. The remaining variables were size (measured in unadjusted function points), effort (hours), and four categorical variables: development type (new development, re-development, enhancement), primary language type (3GL, 4GL), platform (mainframe, midrange, PC, multi-platform), and industry sector (banking, insurance, manufacturing, other).

Table 1 shows summary statistics for size (measured in unadjusted function points), effort, and project delivery rate(PDR). PDR is calculated as effort divided by size; high project delivery rates indicate low productivity. In [1], the authors examined the project delivery rate and found it changes across time. This finding supports the use of a window.

The projects were developed for a variety of industry sectors, where insurance, banking and manufacturing were the most common. Start dates range from 1994 to 2002, although only 9 started before 1998. 3GLs are used by 86% of projects; mainframes account for 40%, and multi-platform for 55%; these percentages for language and platform vary little from year to year. There is a trend over time towards more enhancement projects and fewer new developments. Enhancement projects tend to be smaller than new development, so there is a corresponding trend towards lower size and effort.

In this study we adopt the same range of window sizes as [2]. In [2], the smallest window size was based on the statistical significance of linear regression with windowed project data: the smallest window size with which all regression models were statistically significant was 12 months. The largest window size was based on the necessary number of testing projects for evaluation. As a result, we used window sizes from 12 to 84 months.

Table 2. Formulae of weighting functions

Name	Formula
Triangular	$W(x) = 1 - \|x\|, \|x\| < 1$
Epanechnikov	$W(x) = 1 - x^2, \|x\| < 1$
Gaussian	$W(x) = \exp(-(2.5x)^2/2)$
Rectangular (Uniform)	$W(x) = 1, \|x\| < 1$

3.2 Weighted Moving Windows with Linear Regression

Linear regression is one of the popular methods for effort estimation. A typical effort estimation model is as follows:

$$\text{Effort} = b_0 + b_1 \text{Size} + \epsilon. \tag{1}$$

Here, b_0 and b_1 are regression coefficients, and ϵ represents an error term following a normal distribution. The regression coefficients are inferred from a training set so as to minimize the following function:

$$\sum_{i=1}^{n} \left(\text{Effort}_i - b_0 - b_1 \text{Size}_i\right)^2. \tag{2}$$

Here, n denotes the sample size of the training set.

Equation 2 assumes that the errors of the training set are to be minimized equivalently. Weighted linear regression controls the importance of training projects via weighting. It minimizes the following function:

$$\sum_{i=1}^{n} w_i \left(\text{Effort}_i - b_0 - b_1 \text{Size}_i\right)^2. \tag{3}$$

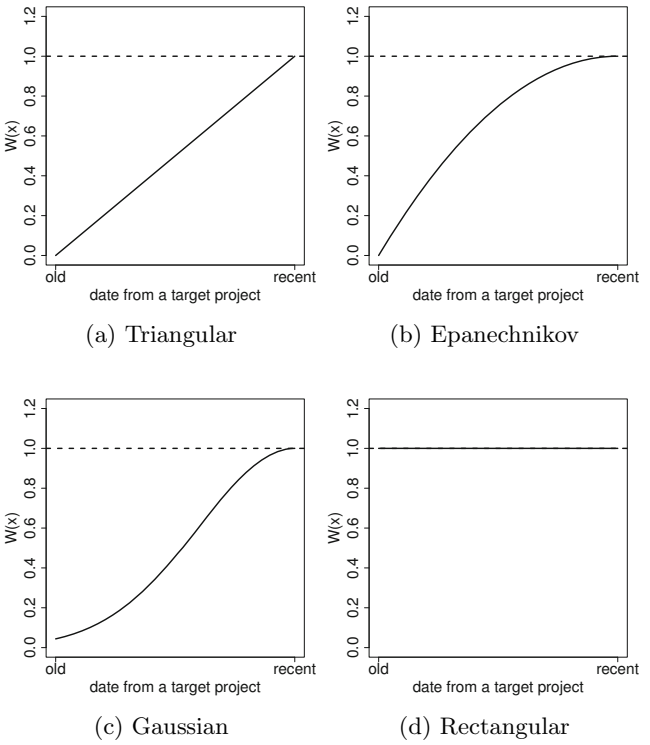
Here, w_i represents case weights for the training set.

From this perspective, an unweighted moving window assigns zero weight to projects that are too old to fall within the window, and equal weights to projects in the window. Weights can be introduced, to take into account the chronological order of projects in the window. This study weights projects in the training set so that a more recent project has a heavier weight. Table 2 shows four weight functions that we examined. We determined x as follows:

$$x = \frac{s - s_i}{w}. \tag{4}$$

Here, s represents the start date of the target project. s_i represents the start date of training project i. w represents the duration of the window. $s - s_i$ is larger for older projects, giving them less weight.

Figure 1 shows the forms of weight functions. A rectangular function is equivalent to unweighted moving windows. Different curve functions affect estimation accuracy differently. This study adopted three typical curves: linear, concave, S-shape. These functions are common in local regression [9].



(a) Triangular (b) Epanechnikov

(c) Gaussian (d) Rectangular

Fig. 1. weight function forms

3.3 Modeling Techniques

Weighted linear regression models were built using almost the same procedure as [1]:

1. The first step in building every regression model is to ensure numerical variables are normally distributed. We used the Shapiro-Wilk test on the training set to check if Effort and Size were normally distributed. Statistical significance was set at $\alpha = 0.05$. In every case, Size and Effort were not normally distributed. Therefore, we transformed them to a natural logarithmic scale.
2. Independent variables whose value is missing in a target project were not considered for inclusion in the estimation model.
3. Every model included $log(Size)$ as an independent variable. Beyond that, given a training set of N projects, no model was investigated if it involved more than $N/10$ independent variables (rounded to the nearest integer), assuming that at least 10 projects per independent variable is desirable [10].
4. Models were based on variables selected with Lasso[11] (the Lasso implementation we used is the "`glmnet`" function from `glmnet` package for R.)

5. To verify the stability of an effort model, we used the following approach: Calculate Cook's distance values 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, were removed from the analysis [8].

This procedure performs variable selection, and thus all variables introduced in Section 3.1 are just candidates for independent variables. Models constructed in our experiment can be different for every project.

3.4 Effort Estimation on Chronologically-Ordered Projects

This study evaluated the effects of moving windows of several sizes along with a timeline of projects' history. The effects were measured by performance comparisons between moving windows and a growing portfolio. A growing portfolio uses all past projects as the training set: no project has a weight of zero.

For a window of w months, this evaluation was performed as follows:

1. Sort all projects by start date
2. Find the earliest project p_0 for which using that window size could make a difference to the training set: that is, at least one project that had finished by the start of p_0 was "too old" to be included in the window (it had started more than w months previously);
3. For every project p_i in chronological sequence (ordered by start date), starting from p_0, form four estimates using weighted and unweighted moving windows, and another estimate using a growing portfolio. For moving windows, the training set is the finished projects whose whole life cycle had fallen within w months prior to the start of p_i. For the growing portfolio, the training set is all of the projects that had finished before the start of p_i.
4. Evaluate estimation results.

3.5 Performance Measures

Performance measures for effort estimation models are based on the difference between estimated effort and actual effort. As in previous studies, this study used MMRE, PRED(25), and MMAE [12] for performance evaluation.

To test for statistically significant differences between accuracy measures, we used the Wilcoxon ranked sign test and set statistical significance level at $\alpha = 0.05$. `wilcoxsign_test` function of `coin` package for R was used.

4 Results

4.1 Accuracy with Different Window Sizes

We begin by comparing estimation accuracy with each of the weighting functions against a common baseline: not using a window at all, but instead retaining all past projects as training data.

Table 3. Mean absolute residuals with different window durations

Duration	Testing	Growing	(a)		(b)		(c)		(d)	
(months)	Projects	MAE	MAE	p–val.	MAE	p–val.	MAE	p–val.	MAE	p–val.
12	165	2541	2730	0.127	2772	0.114	2667	0.306	2560	0.981
18	193	2630	2565	0.445	2601	0.514	2580	0.822	2549	0.287
24	201	2638	2501	0.275	2466	0.085	2541	0.183	2610	0.984
30	202	2647	2428	0.013	2491	0.093	2571	0.116	2581	0.365
36	206	2645	2518	0.139	2585	0.378	2492	0.191	2526	0.001
42	206	2645	2594	0.084	2613	0.050	2597	0.140	2559	0.004
48	206	2645	2572	0.049	2596	0.068	2618	0.157	2599	0.003
54	206	2645	2572	0.035	2593	0.007	2541	0.042	2597	0.086
60	198	2642	2550	0.005	2574	0.000	2564	0.019	2655	0.254
66	184	2622	2570	0.001	2576	0.000	2465	0.004	2702	0.226
72	153	2527	2447	0.000	2498	0.000	2490	0.000	2554	0.016
78	126	2300	2232	0.000	2281	0.000	2237	0.000	2327	0.031
84	80	2211	2165	0.000	2204	0.000	2139	0.000	2238	0.022

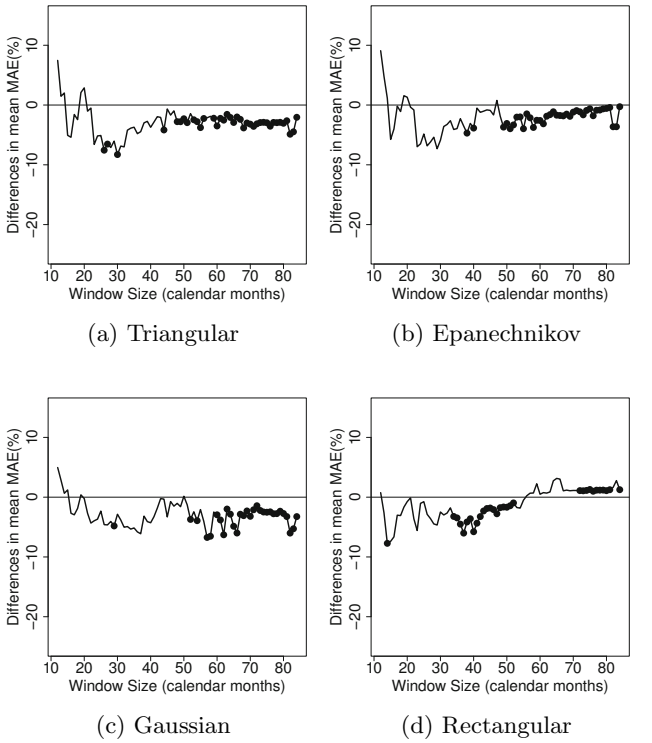
(a) Triangular, (b) Epanechnikov, (c) Gaussian, (d) Rectangular

Table 4. Mean MRE with different window durations

Duration	Testing	Growing	(a)		(b)		(c)		(d)	
(months)	Projects	MRE	MRE	p–val.	MRE	p–val.	MRE	p–val.	MRE	p–val.
12	165	1.35	1.30	0.905	1.31	0.743	1.25	0.752	1.12	0.029
18	193	1.29	1.13	0.004	1.20	0.002	1.12	0.022	1.15	0.001
24	201	1.28	1.13	0.001	1.11	0.000	1.16	0.003	1.14	0.038
30	202	1.28	1.14	0.000	1.20	0.002	1.19	0.002	1.23	0.008
36	206	1.26	1.15	0.001	1.20	0.020	1.17	0.008	1.18	0.000
42	206	1.26	1.22	0.001	1.19	0.000	1.23	0.009	1.18	0.000
48	206	1.26	1.23	0.001	1.20	0.002	1.22	0.001	1.19	0.000
54	206	1.26	1.23	0.000	1.21	0.000	1.24	0.000	1.20	0.000
60	198	1.29	1.19	0.000	1.26	0.000	1.27	0.000	1.25	0.000
66	184	1.32	1.24	0.000	1.24	0.000	1.27	0.000	1.28	0.001
72	153	1.39	1.31	0.000	1.34	0.000	1.38	0.000	1.31	0.001
78	126	1.48	1.40	0.000	1.43	0.000	1.38	0.000	1.40	0.002
84	80	1.44	1.32	0.000	1.35	0.000	1.31	0.000	1.40	0.000

(a) Triangular, (b) Epanechnikov, (c) Gaussian, (d) Rectangular

Tables 3 and 4 show the effect of window durations on mean absolute residuals and mean MRE. The first column shows window durations. The 2nd column shows the total number of projects used as a target project with the corresponding window duration. The 3rd column shows accuracy measures with a growing portfolio. The 4th column shows accuracy measures when the Triangular function was used to weight projects within the window. The 5th column shows the p–value from statistical tests on accuracy measures between a growing portfolio and the Triangular function. The remaining columns show accuracy measures and p–values for the other weighting functions. The results were computed for



(a) Triangular (b) Epanechnikov

(c) Gaussian (d) Rectangular

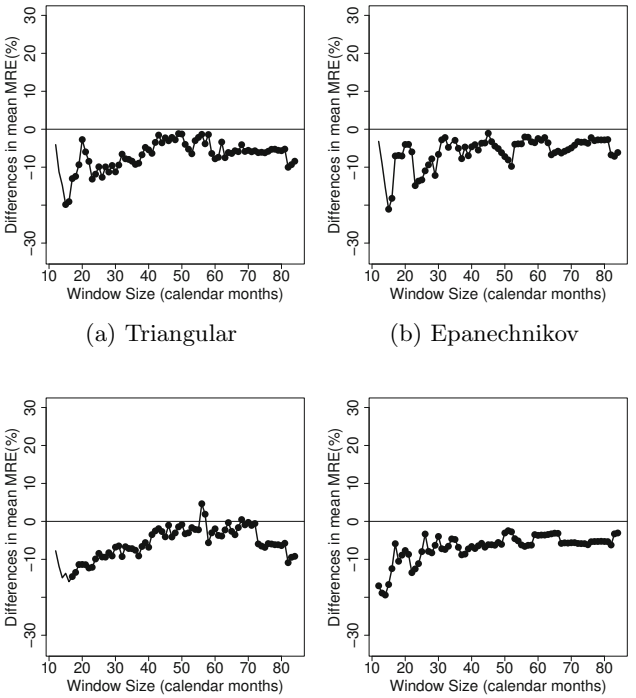
Fig. 2. The percent difference of accuracy measures between growing and windowing (mean MAE)

every month; the tables only show every 6 months, due to space limitations. This is sufficient to show the essential trends.

Figures 2 and 3 show the difference in mean MAE and mean MRE between a growing portfolio and moving windows. The x-axis is the duration of the window, and the y-axis is the subtraction of the accuracy measure value with a growing portfolio from that with moving windows at the given x-value (expressed in relative percentage terms). Smaller values of MAE and MRE are better, so the window is advantageous where the line is below 0. Circle points mean a statistically significant difference, in favor of moving windows.

Figures and tables revealed characteristics of unweighted and weighted moving windows compared to a growing portfolio:

– With windows of up to 30 months, MAE rarely shows significant preference with any approach. The line starts above zero and quickly goes below zero (favoring windows), but the difference is seldom significant (and not at all in Fig. 2(b)). In contrast, as shown in Fig. 3, with MRE the difference was significant regardless of weighting functions.



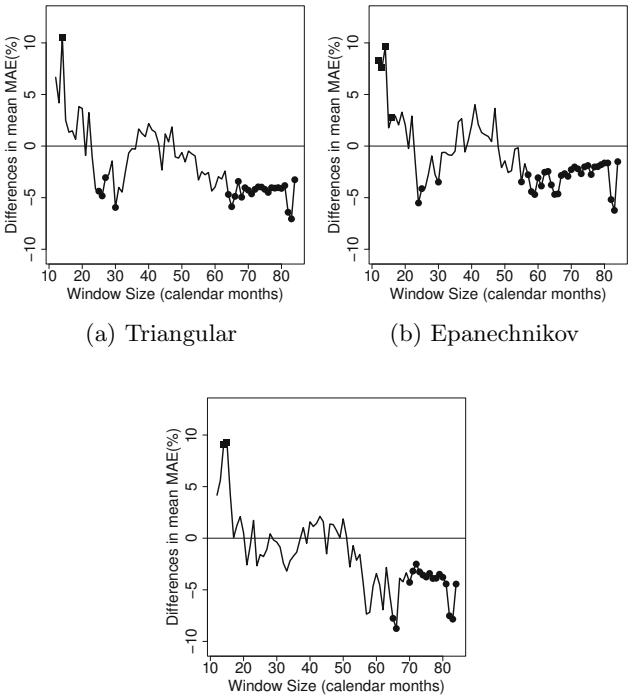
(a) Triangular (b) Epanechnikov

(c) Gaussian (d) Rectangular

Fig. 3. The percent difference of accuracy measures between growing and windowing (mean MRE)

– For windows of 30 to 48 months, moving windows become advantageous in terms of MAE, but the effect varies for different weighting functions: Figure 2(c) has no significant difference through this range of durations. The preference for moving windows is still seen in terms of MRE, regardless of weighting functions. However, the difference looks smaller than at smaller window sizes.
– With larger windows, all measures are better using moving windows in Figs. 2(a), 2(b), 2(c) and Fig. 3. However, the improvements in mean MRE and MAE decrease compared to smaller windows, especially for Epanechnikov and Rectangular. Sometimes circle points in Figure 2 are found above zero. This is due to the use of a non-parametric statistical test.

In summary, in this data set, weighted and unweighted windows improve estimation accuracy significantly, particularly with larger windows. Different weighting functions affect accuracy in different ways.



(a) Triangular (b) Epanechnikov

(c) Gaussian

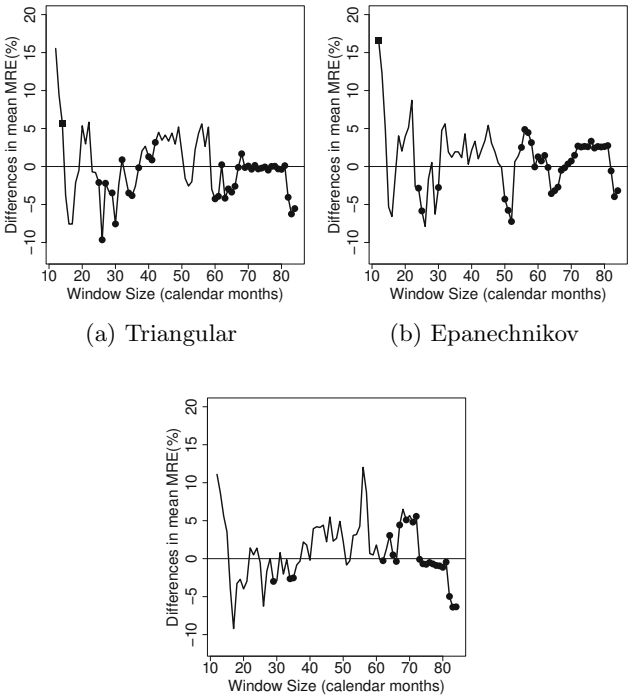
Fig. 4. The percent difference of accuracy measures between Rectangular and the other weight functions (Mean MAE)

4.2 Accuracy Comparisons among Different Weighting Functions

Figures 4 and 5 show the difference in mean MAE and mean MRE between Rectangular (unweighted) and the other functions (weighted). Weighted moving windows are advantageous where the line is below 0. Square points mean a statistically significant difference, with weighted moving windows being worse. The other notations are as same as Figs. 2 and 3.

Figures 4 and 5 reveal the following:

- With windows of up to 30 months, the advantage shifted from unweighted windows to weighted windows. Few differences are statistically significant.
- With windows of 30 to 54 months, weighted and unweighted moving windows are similar in terms of MAE. There is rarely clear preference between them. Statistical tests support weighted moving windows at some window sizes. The improvement by Gaussian function is small, and the circle points are rarely found, as shown in Fig. 4(c).
- With windows of more than 54 months, weighted moving windows are advantageous in terms of MAE and MRE. The Triangular and the Gaussian



(a) Triangular (b) Epanechnikov

(c) Gaussian

Fig. 5. The percent difference of accuracy measures between Rectangular and the other weight functions (Mean MRE)

functions make more difference than the Epanechnikov function. Most differences are small (plus or minus 2%).

5 Discussion

5.1 Answer to RQ1

First, the null hypothesis was rejected for the difference between weighted moving windows (with all weighting methods) and a growing portfolio. In this data set the use of both weighted and unweighted moving windows significantly improves estimation accuracy, compared to using a growing portfolio.

Next, statistical tests for differences in accuracy between unweighted and weighted moving windows also reject the null hypothesis at many window sizes. For the Epanechnikov function, for instance, the null hypothesis was rejected at durations around 30 months, and from 49 to 84 months, based on mean MAE. The difference based on mean MRE was significant at many window sizes.

We conclude that the use of weighted moving windows can improve estimation accuracy, compared to using unweighted moving windows, when fixed-duration windows are used.

5.2 Answer to RQ2

Even at small window durations, Figures 4 and 5 show some window sizes where weighted moving windows provide significantly better accuracy than unweighted windows. The difference in MAE becomes clear when using larger windows, of 54 months or more. Differences in MRE are significant at many window sizes, and the Gaussian and the Triangular functions showed better performance in larger windows as shown in Figures 4 and 5.

Results show that weighting is helpful, particularly at larger window sizes. However, it must be noted that the difference between accuracy with weighted and unweighted windows is small, mostly around 2%.

In [3], the effectiveness of weighting was reasoned to be due to an interaction between window sizes and the steepness of weight function curves. With small size windows, a weight function assigns steeply declining weights. With large window sizes, a weight function assigns gently declining weights. When the degree of steepness meshes with a window size, a weight function contributes to improvement of estimation accuracy.

Figure 1 depicts the difference of steepness among weight functions. Gaussian is the steepest function, and Epanechnikov is the most gentle function. The steepness of Triangular function is between them. Unweighted moving windows assigns equal weights and is more gentle than Epanechnikov function.

Figure 2 shows the gentlest Rectangular function meshed with window sizes earlier than steeper functions. The difference in larger windows is clear in steeper functions. For fixed-duration windows, steeper functions could appropriately reflect the importance of recent projects. Rectangular function eventually meshed with large window sizes again and improved estimation accuracy significantly. However, the range of significant durations was narrower than that of the other functions.

The results suggest that weighted moving windows can improve estimation accuracy when the steepness of its function is appropriately meshed. We conclude that all weight functions tend to mesh with large window sizes, as do unweighted windows, but their effectiveness differs depending on how well the steepness of the functions meshes with window sizes.

5.3 Answer to RQ3

In [2], the authors evaluated the difference between results with fixed-duration windows and fixed-size windows, and found:

- the preference of growing portfolio in smaller windows became smaller, and statistical significance almost diminished.
- the trend lines went upward as a window size increases.

- the significance range is narrower, around 40 months.
- the improvement in MAE and MRE was generally smaller with fixed-duration windows than with fixed-size windows.

Figure 2(d) supported these results, though there are additional significant window ranges because this study used another modeling approach. Fixed-duration windows allow a variable number of training projects, which may lead to improvement over unweighted moving windows, especially as short-duration windows might still contain numerous training projects. Figure 3(d) clearly reflected this effect. However, the trend lines still go upward, and window sizes around 40 months are still advantageous significantly. The range of significant durations varies with different weight functions, but the trends remain.

We conclude that the differences between fixed-duration and fixed-size windows found previously still apply when using weighted instead of unweighted moving windows.

6 Threats to Validity

This study shares the same threats to validity as the previous studies.

First, we used only one dataset. The dataset is a convenience sample and may not be representative of software projects in general. Thus, the results may not be generalized beyond this dataset; this is true of all studies based on convenience samples. We trust that numerous potential sources of variation can be removed from the dataset by the selection of a single-company dataset. Since the dataset is large and covers several years, we assume it is a fair representation of this organization's projects. The inclusion of the industry sector as an independent variable helps to allow for variations among sectors in the dataset.

Second, all the models employed in this study were built automatically. Automating the process necessarily involved making some assumptions, and the validity of our results depends on those assumptions being reasonable. For example, logarithmic transformation is assumed to be adequate to transform numeric data to an approximately normal distribution; residuals are assumed to be random and normally distributed without that being actually checked; multi-collinearity between independent variables is assumed to be handled automatically by the nature of Lasso. Based on our past experience building models manually, we believe that these assumptions are acceptable. One would not want to base important decisions on a single model built automatically, without at least doing some serious manual checking, but for calculations such as chronological estimation across a substantial data set we believe that the process here is reasonable.

Third, this study only used weighted linear regression. Many effort estimation models have been proposed, and each model can show better accuracy in particular situations. However, regression is a popular effort estimation approach. We thus think it is a reasonable choice.

7 Conclusion

This paper investigated the effect on effort estimation accuracy of using weighted moving windows, when fixed-duration windows are adopted. We have shown that it has a statistically significant effect; different weight functions affected estimation accuracy differently; with the steepness of the weight function being important; and weighted moving windows were particularly advantageous in larger windows. These findings reinforce previous results using fixed-size windows.

Compared to [2], the use of weight functions improves estimation accuracy significantly. Compared to [3], the percent improvements in MAE and MRE are smaller with fixed-duration windows than with fixed-size windows.

Our future work involves generalization with other settings: other companies' datasets and other effort estimation models. Furthermore, how to determine appropriate steepness is a crucial question for better estimation.

References

1. Lokan, C., Mendes, E.: Applying moving windows to software effort estimation. In: Proc. of ESEM 2009, pp. 111–122 (2009)
2. Lokan, C., Mendes, E.: Investigating the Use of Duration-Based Moving Windows to Improve Software Effort Prediction. In: Proc. of APSEC 2012, pp. 818–827 (2012)
3. Amasaki, S., Lokan, C.: The evaluation of weighted moving windows for software effort estimation. In: Heidrich, J., Oivo, M., Jedlitschka, A., Baldassarre, M.T. (eds.) PROFES 2013. LNCS, vol. 7983, pp. 214–228. Springer, Heidelberg (2013)
4. Auer, M., Biffl, S.: Increasing the accuracy and reliability of analogy-based cost estimation with extensive project feature dimension weighting. In: Proc. of ISESE 2004, pp. 147–155. IEEE (2004)
5. Mendes, E., Lokan, C.: Investigating the use of chronological splitting to compare software cross-company and single-company effort predictions: A replicated study. In: Proc. of EASE 2009 (2009)
6. Keung, J.W., Kitchenham, B.A., Jeffery, D.R.: Analogy-X: Providing Statistical Inference to Analogy-Based Software Cost Estimation. IEEE Trans. Softw. Eng. 34(4), 471–484 (2008)
7. Li, J., Ruhe, G.: Analysis of attribute weighting heuristics for analogy-based software effort estimation method AQUA+. Empir. Softw. Eng. 13(1), 63–96 (2007)
8. Maxwell, K.D.: Applied Statistics for Software Managers. Prentice Hall (2002)
9. Loader, C.: Local Regression and Likelihood. Statistics and Computing. Springer (1999)
10. Tabachnick, B.G., Fidell, L.S.: Using Multivariate Statistics. Harper-Collins (1996)
11. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. Roy. Statist. Soc. Ser. B, 267–288 (1996)
12. Port, D., Korte, M.: Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research. In: Proc. of ESEM 2008. ACM (2008)