JOURNAL OF SOFTWARE: EVOLUTION AND PROCESS
J. Softw. Evol. and Proc. 0000; **00**:1–20
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/smr

On the effectiveness of weighted moving windows: Experiment on linear regression based software effort estimation

S. Amasaki[1]*and C. Lokan[2]

[1]*Department of Systems Engineering, Okayama Prefectural University, 111 Kuboki, Soja, Okayama 719-1197, Japans*
[2]*School of Engineering and Information Technology, UNSW Canberra, Canberra, ACT 2600, Australia*

SUMMARY

In construction of an effort estimation model, it seems effective to use a window of training data so that the model is trained with only recent projects. Considering the chronological order of projects within the window, and weighting projects according to their order within the window, may also affect estimation accuracy. In this study, we examined the effects of weighted moving windows on effort estimation accuracy. We compared weighted and non-weighted moving windows under the same experimental settings. We confirmed that weighting methods significantly improved estimation accuracy in larger windows, though the methods also significantly worsened accuracy in smaller windows. This result contributes to understanding properties of moving windows. Copyright © 0000 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: effort estimation; moving window; gradual weighting

1. INTRODUCTION

Software effort estimation is an important activity in software development. Its accuracy has a significant effect on project success. Research on the topic has studied two types of effort estimation methods: non-model-based methods (e.g. "expert judgment"), and model-based approaches (e.g. COCOMO, CART, etc.) [1]. A systematic review revealed that model-based software effort estimation has been a popular research topic [2].

A software effort estimation model is developed from training data. Evaluation of the accuracy of the model is based on estimated efforts for testing data. Most studies split project data into training data and testing data randomly, or used a cross-validation approach.

In a practical sense, software projects can be ordered chronologically. Predicting the effort of future projects based on past projects, instead of forming training and testing sets without regard to chronology, is more reasonable. Furthermore, it also seems appropriate to use recent projects as a basis for effort estimation. This is because old projects might be less representative of an organization's current practices.

Lokan and Mendes [3] 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 found that estimation accuracy could increase by using the window.

Their window assumes that old projects that are no longer in the window have no value as training data, and projects within the window are all equally useful as training data. This assumption does

*Correspondence to: S. Amasaki, Okayama Prefectural University, Department of Systems Engineering, Soja Okayama Japan 719-1197. E-mail: amasaki@cse.oka-pu.ac.jp

not take into account the fact that when projects are ordered chronologically, even projects in a window may have different importance in effort estimation model construction.

This detailed consideration is achievable with an idea of *gradual weighting*. Gradual weighting assigns different weights of importance to projects according to their relative age to a target project. Gradual weighting can be applied to a growing data set (all past projects are retained, so the data set grows as more projects finish). It can also be combined with the idea of a moving window, by assigning different weights to projects within the window.

We can consider four strategies:

- *unweighted growing*: all past projects are retained, all with the same weight.
- *unweighted window*: old projects that no longer fit within the window have a weight of zero, and all projects in the window have the same weight.
- *weighted growing*: all past projects are retained, no project has a zero weight, and projects have different weights according to their age relative to the target project.
- *weighted window*: projects outside the window have zero weight, and projects are weighted differently within the window according to their age relative to the target project.

In this study we investigate whether these strategies affect estimation accuracy differently, in order to explore the effect of gradual weighting for software effort estimation. Linear regression is used to build estimation models for each target project, using each of the four strategies to select and weight training projects. Linear regression models can consider different importance with *case weights*. The case weights can assign gradual weights such that recent projects receive higher importance than older projects. We adopt linear regression in this paper because it is one of the most-used model building techniques in research in software effort estimation [2], and because of its case-weighting feature.

In this paper, we address the following questions:

RQ1. Does gradual weighting affect estimation accuracy?

RQ2. Is there a difference in the accuracy of estimates between gradual weighting and moving windows?

RQ3. Is there a difference in the accuracy of estimates when combining gradual weighting with moving windows?

RQ4. Are there any insights with regard to trends with the use of different weighting functions?

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 a growing portfolio, only projects that had been completed at the time a given target project starts are used as training data for that project. 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 estimation with cross-validation (based on unrealistic information) does significantly better than estimation considering chronology (based on realistic information), the implication is that the apparent accuracy achieved by ignoring chronology does not reflect what an estimator would achieve in practice.

DOI: 10.1002/smr

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.

To the best of our knowledge, Kitchenham et al. [9] first mentioned the use of moving windows. As a result of an experiment, they argued that old projects should be removed from the data set as new ones were added, so that the size of the dataset remained constant.

MacDonell and Shepperd [10] investigated moving windows as part of a study of how well data from prior phases in a project could be used to estimate later phases. They found that accuracy was better when a moving window of the 5 most recent projects was used as training data, rather than using all completed projects as training data.

Lokan and Mendes [3] 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 the following insights: the use of a window could affect accuracy significantly; predictive accuracy was better with larger windows; some window sizes were "sweet spots". In [11], we investigated the use of moving windows with Estimation by Analogy (EbA) on the same dataset as [3] except for an additional *sector* variable. We found the window could also improve estimation accuracy for EbA.

Later Lokan and Mendes also investigated the effect on accuracy when using moving windows of various durations to form training sets on which to base effort estimates [12]. They showed that the use of windows based on duration can improve the accuracy of estimates, but to a lesser extent than windows based on a fixed number of projects.

The idea of gradual weighting was first investigated in [13]. This study extends [13], by thoroughly investigating the relationships and combinations of moving windows and gradual weighting. We re-formulated the research questions into the following hypotheses:

H1: Unweighted window (formerly called moving windows) is more effective (i.e. using them can significantly improve estimation accuracy) than unweighted growing — originally examined in [3]).

H2: Weighted growing is more effective than unweighted growing.

H3: Weighted growing and unweighted window have different effects on accuracy.

H4: Weighted moving windows are more effective than unweighted growing.

H5: Weighted moving windows are more effective than weighted growing.

H6: Weighted moving windows are more effective than unweighted window.

Figure 1 clarifies the relationships between the hypotheses and the four strategies. Sections 4.1 to 4.6 examine H1 to H6, respectively.

Hypotheses H1 to H3 examine the effect of weighting and windowing separately, compared to the baseline of using neither. Hypotheses H4 to H6 examine the effect of weighting and windowing together, compared to using neither or only one of them. Between them the six hypotheses address research questions RQ1 to RQ3. Comparisons across sections 4.2 to 4.6 enable observations to be made that address RQ4.

3. RESEARCH METHOD

3.1. Dataset Description

The data set used in this paper is the same one analyzed in [3]. This data set is sourced from Release 10 of the ISBSG Repository.

DOI: 10.1002/smr

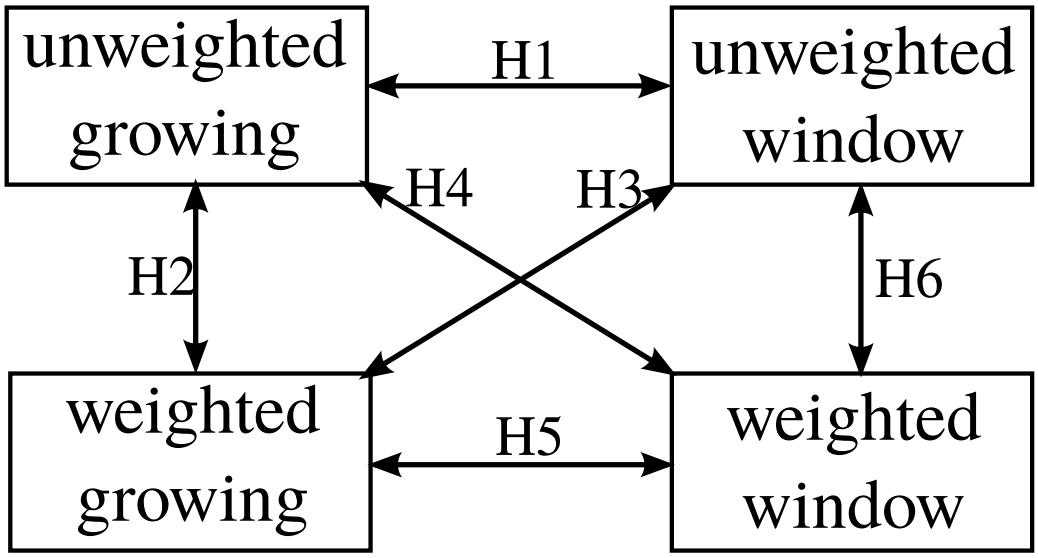


Figure 1. Research model

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 a 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.

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 the use of single-company data was more suitable to answer our research questions than using cross-company data. This is because single-company is governed under single management, and we assume that many factors that influence software development are likely to vary less within a single organization than across organizations. Preliminary analysis showed that three projects were extremely influential (according to Cook's distance, as described in Section 3.3) 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 [11, 14] 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

DOI: 10.1002/smr

Table I. 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

Table II. Formulae of weighted 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)	$W(x) = 1, \|x\| < 1$

language type (3GL, 4GL), platform (mainframe, midrange, PC, multi-platform), and industry sector (banking, insurance, manufacturing, other).

Table I 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 [3], 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, but only 9 started before 1998. 3GLs are used by 86% of the 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.

There are two ways in which a window size might be defined: by the number of projects [3], or duration [12]. A window based on duration can be scaled to include only those projects that reflect recent development projects and practices. In contrast, a window based on the number of projects can be scaled to provide enough data for sound analysis. This study defines a window as containing a fixed number of projects, which is more effective [12].

We adopted the same range of window sizes as [3]. 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 20 projects. The largest window size was based on retaining sufficient testing projects for evaluation. As a result, we used window sizes from 20 to 120.

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:

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

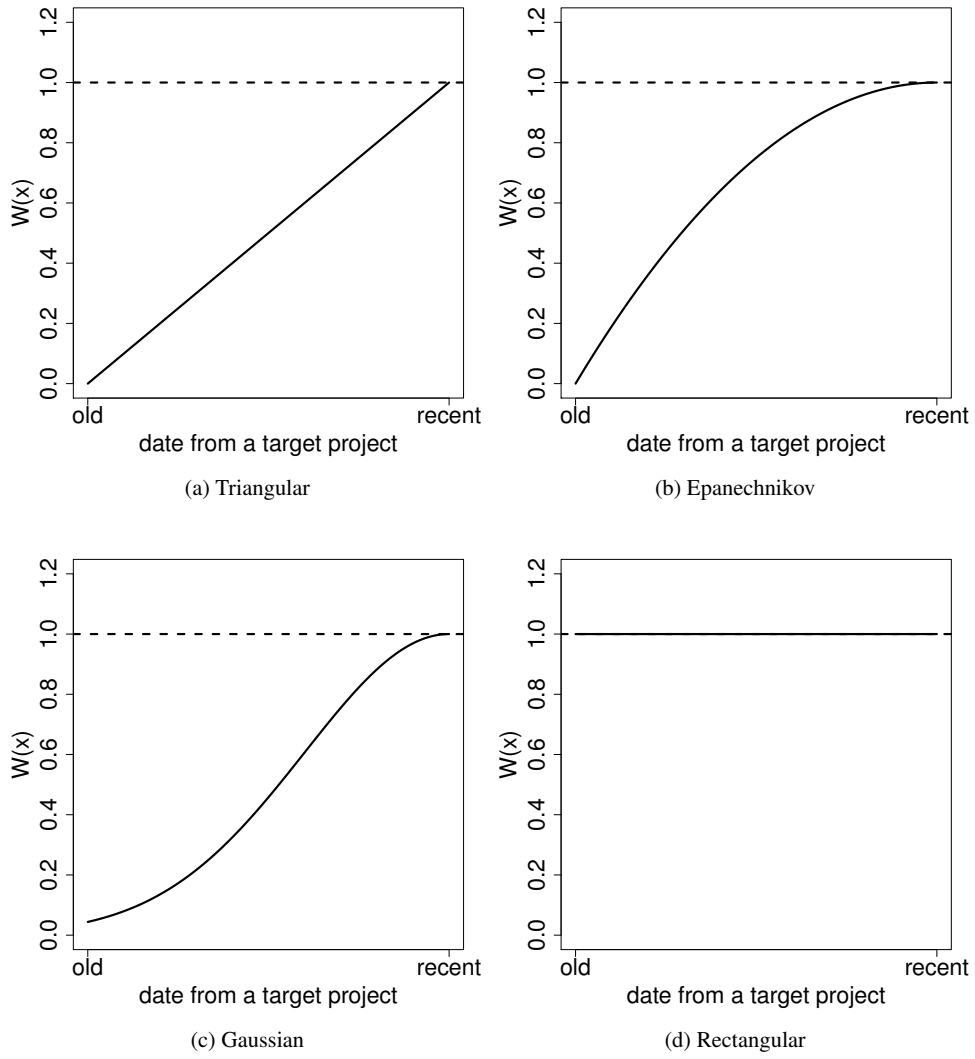
Here, b_0 and b_1 are regression coefficients, and ϵ represents an error term following a normal distribution. The regression coefficients are inferred from a training set so as to minimize the following function:

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

Here, n denotes the size of the training set.

Equation (2) assumes that the errors of the projects in the training set are to be minimized equivalently. Weighted linear regression controls the importance of training projects via weighting.

DOI: 10.1002/smr



(a) Triangular

(b) Epanechnikov

(c) Gaussian

(d) Rectangular

Figure 2. Weighted function forms

It minimizes the following function:

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

Here, w_i represents case weights for the training set.

From this perspective, an unweighted moving window assigns zero weight to old projects no longer in the window, and equal weights to projects in the window.

This study weights projects in the training set so that a more recent project has a heavier weight. Table II shows the four weight functions that we examined. We determined x as follows:

$$x = \frac{n_i}{n}. \tag{4}$$

Here, n_i represents a rank of project i in ascending order of date. That is, an older project has a lower rank and x is smaller.

Figure 2 shows the forms of the weighted functions. A rectangular function is equivalent to non-weighted moving windows. Different curve functions affect estimation accuracy differently.

DOI: 10.1002/smr

This study adopted three typical curves: linear, concave, S-shape. These functions are common in local regression [15].

3.3. Modeling Techniques

Weighted linear regression models were built using almost the same procedure as in [3]:

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 [16].
4. Models were based on variables selected with Lasso [17] instead of stepwise regression, because preliminary investigation (details not presented here) showed that Lasso gave more accurate estimates than stepwise (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.

This procedure performs variable selection, and all variables introduced in Section 3.1 are 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, gradual weighting, and their combinations. The effects were measured by accuracy comparisons among them. This evaluation method was performed with the following steps:

1. Sort all projects by starting time
2. For a given window size w, find the earliest project p_0 for which at least $w + 1$ projects were completed prior to the start of p_0 (note that projects from p_0 onwards are the ones whose estimate could be affected by using a window, so they form the set of evaluation projects for this window size).
3. For every project p_i in chronological sequence, starting from p_0, form estimates using unweighted and weighted moving windows, and using unweighted and weighted growing. For moving windows, the training set is the w most recent projects that finished before the start of p_i. For growing approach, the training set is all projects that finished before the start of p_i.
4. Evaluate estimation results.

3.5. Accuracy Measures

Accuracy measures for effort estimation models are based on the difference between estimated effort and actual effort. As in previous studies, this study used Mean Magnitude of Relative Error (MMRE) and Mean Absolute Error (MAE) [1] to evaluate accuracy.

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 the `coin` package for R was used, with default options.

DOI: 10.1002/smr

Table III. Accuracy with different window sizes (unweighted growing and unweighted window)

Window size(N)	Testing Projects	Growing MAE	Window MAE	p–val.	Growing MRE	Window MRE	p–val.
20	201	2638	2640	0.342	1.28	1.13	0.587
30	178	2578	2534	0.831	1.35	1.15	0.340
40	165	2541	2380	0.305	1.35	1.12	0.125
50	153	2527	2378	0.040	1.39	1.14	0.001
60	136	2458	2103	0.000	1.42	1.09	0.000
70	126	2300	2015	0.003	1.48	1.23	0.000
80	126	2300	2082	0.004	1.48	1.20	0.000
90	111	2236	2025	0.000	1.37	1.14	0.000
100	88	2314	2112	0.000	1.36	1.11	0.000
110	75	1981	1818	0.004	1.39	1.15	0.001
120	71	1982	1780	0.000	1.38	1.10	0.001

4. RESULTS

4.1. Accuracy Comparisons between Unweighted Growing and Unweighted Window

The main baseline against which other strategies are compared in this paper is *unweighted growing*. We begin by comparing the accuracy of estimates produced with unweighted growing and unweighted window. This was studied originally, with the same data set, in [3]; this study differs from [3] in the use of an additional independent variable (sector) and a different variable selection method (Lasso). These results effectively repeat findings from [11, 13]. They are presented here so that this paper is self-contained.

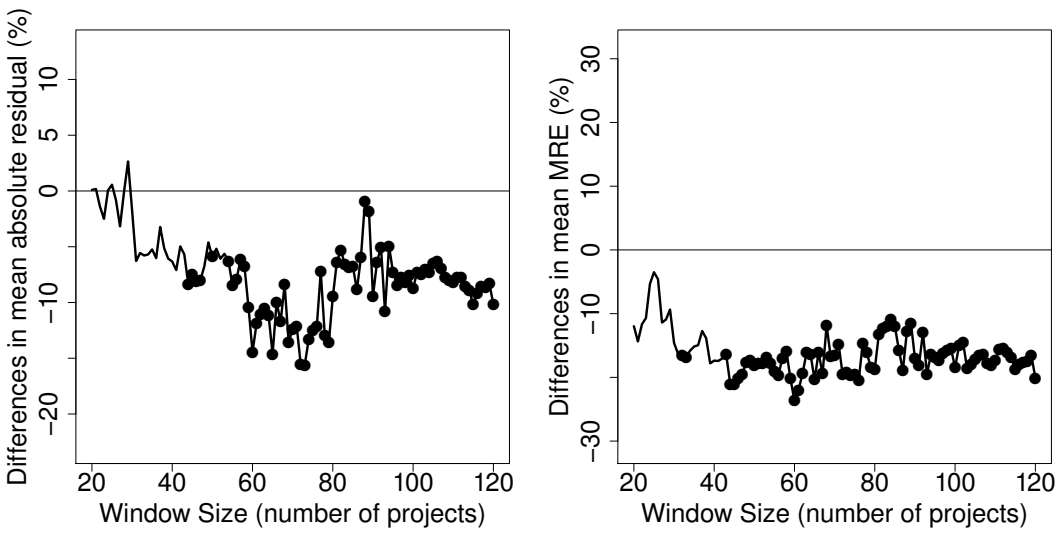
Table III shows the effect on MAE and MMRE from using unweighted window, compared to unweighted growing, retaining all training data without using gradual weighting. The first column shows window sizes, and the second column shows the total number of projects used as target projects with the corresponding window size. The larger the window, the smaller the number of testing projects. The 3rd column shows MAE with unweighted growing for the corresponding window sizes, the 4th column shows MAE with unweighted window, and the 5th column shows the p–value from statistical tests on the difference between the accuracy of the unweighted growing and unweighted window estimates. Columns 6 to 8 present the same information as columns 3 to 5, this time for MMRE. The results were computed for all window sizes; the tables only show every tenth window size, due to space limitations. This is still sufficient to show the essential trends.

Figure 3 shows the difference in MAE and MMRE between unweighted growing and unweighted window. The x-axis is the size of the window, and the y-axis is the subtraction of the accuracy measure value with unweighted growing from that with unweighted window at the given x-value. Smaller values of MAE and MMRE are better, so unweighted window is advantageous where the line is below 0. Circle points mean a statistically significant difference, with unweighted window being better than unweighted growing. Square points mean a statistically significant difference, with unweighted window being worse than unweighted growing. We consider a difference is significant if the corresponding p-value is below 0.05.

We observe that:

- With the smallest windows, using unweighted growing is better than applying unweighted window in MAE. However, the difference is insignificant in all of those window sizes.
- At medium window sizes, windows become more advantageous. Adopting an unweighted window strategy made a significant difference to MAE from $w = 40$, and from a smaller window size in MMRE.
- With larger windows, both measures are always better using unweighted window, and the difference was significant in $40 \leq w \leq 120$. Improvements in MMRE range from 10% to 24%, averaging 17%. Improvements in MAE range from 1% to 16%, averaging 8%.

DOI: 10.1002/smr



(a) mean absolute residuals

(b) MMRE

Figure 3. Percent difference in accuracy measures between unweighted growing and window

Table IV. Mean absolute residuals with unweighted growing and weighted growing

Window size(N)	Testing Projects	Growing MAE	Triangular Growing MAE	Triangular Growing p–val.	Epanechnikov Growing MAE	Epanechnikov Growing p–val.	Gaussian Growing MAE	Gaussian Growing p–val.
20	201	2638	2685	0.201	2661	0.040	2605	0.054
30	178	2578	2627	0.095	2593	0.010	2537	0.019
40	165	2541	2600	0.073	2562	0.010	2506	0.020
50	153	2527	2588	0.061	2549	0.011	2473	0.003
60	136	2458	2474	0.018	2433	0.003	2390	0.003
70	126	2300	2306	0.013	2284	0.001	2211	0.001
80	126	2300	2306	0.013	2284	0.001	2211	0.001
90	111	2236	2250	0.023	2256	0.004	2218	0.006
100	88	2314	2270	0.029	2309	0.007	2232	0.010
110	75	1981	1964	0.022	1981	0.003	1971	0.030
120	71	1982	1940	0.006	1971	0.001	1947	0.009

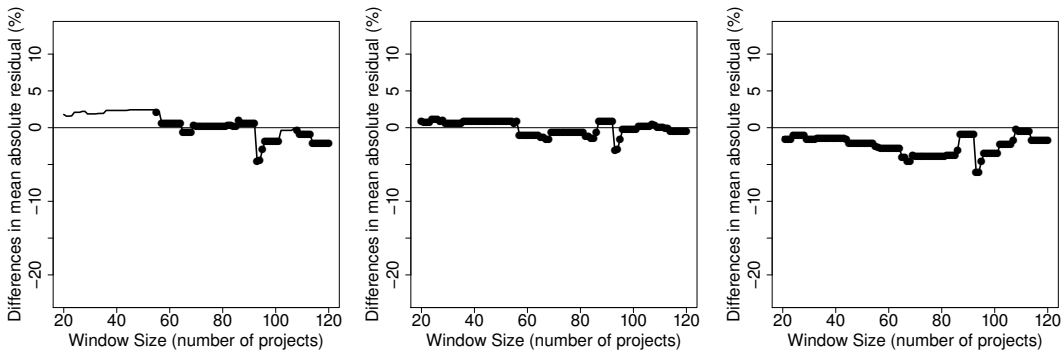
Table V. MMRE with unweighted growing and weighted growing

Window size(N)	Testing Projects	Growing MRE	Triangular Growing MRE	Triangular Growing p–val.	Epanechnikov Growing MRE	Epanechnikov Growing p–val.	Gaussian Growing MRE	Gaussian Growing p–val.
20	201	1.28	1.35	0.001	1.35	0.000	1.31	0.001
30	178	1.35	1.42	0.001	1.42	0.000	1.37	0.000
40	165	1.35	1.43	0.003	1.44	0.000	1.38	0.001
50	153	1.39	1.47	0.003	1.48	0.000	1.42	0.001
60	136	1.42	1.45	0.000	1.46	0.000	1.46	0.000
70	126	1.48	1.50	0.000	1.51	0.000	1.52	0.000
80	126	1.48	1.50	0.000	1.51	0.000	1.52	0.000
90	111	1.37	1.43	0.000	1.42	0.000	1.45	0.001
100	88	1.36	1.39	0.001	1.39	0.000	1.41	0.005
110	75	1.39	1.43	0.000	1.43	0.000	1.48	0.014
120	71	1.38	1.39	0.000	1.41	0.000	1.44	0.003

4.2. Accuracy Comparisons between Unweighted Growing and Weighted Growing

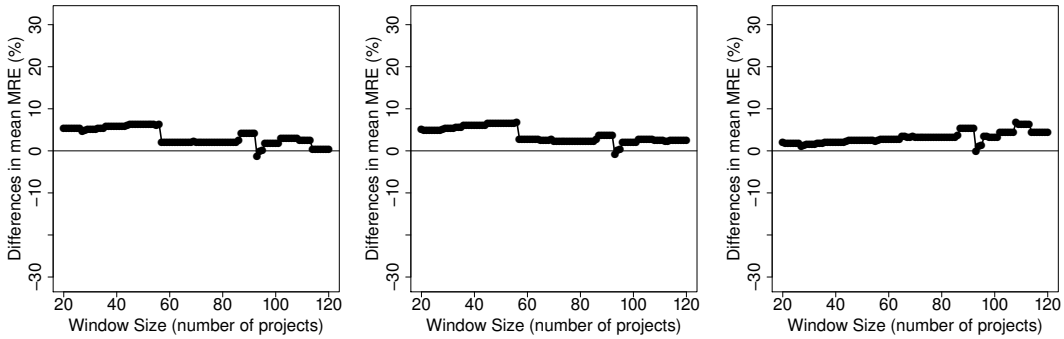
Tables IV and V show the effects of gradual weighting on MAE and MMRE. Note that windows are not used in the growing strategy. The different rows represent the sets of testing projects that are

DOI: 10.1002/smr



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 4. Percent difference in MAE between unweighted and weighted growing



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 5. Percent difference in MMRE between unweighted and weighted growing

used with different sizes when windowing strategies are used. They are tabulated here to support comparisons between results from this section and other sections.

Figures 4 and 5 show the difference in MAE and MMRE between unweighted and weighted growing. The y-axis is the subtraction of the accuracy measure value with unweighted growing from that weighted growing at the given x-value. Weighted growing (Gradual weighting) is advantageous where the line is below 0. Circle points mean a statistically significant difference, in favour of weighted growing. Square points would mean a statistically significant difference, to the detriment of weighted growing; however there is no point with weighted growing being worse, and the Figures only show circles.

The figures and tables reveal common results with the Triangular and Epanechnikov functions:

- With smaller windows, MAE and MMRE are better using unweighted growing (the points are above the x-axis).
- With medium windows, weighted growing becomes competitive for MAE. For MMRE, unweighted growing is still advantageous.
- With larger windows, weighted growing becomes advantageous for MAE. For MMRE, unweighted growing is still advantageous.

Although MMRE and MAE do not support weighted growing for small and medium ranges, statistical tests actually supported weighted growing. As noted, the Figures only have circle points, which means preference for weighted growing. This contradiction was due to the use of

DOI: 10.1002/smr

Table VI. Mean absolute residuals with weighted growing and unweighted window

Window size(N)	Testing Projects	Window MAE	Triangular Growing MAE	Triangular Growing p–val.	Epanechnikov Growing MAE	Epanechnikov Growing p–val.	Gaussian Growing MAE	Gaussian Growing p–val.
20	201	2640	2685	0.304	2661	0.281	2605	0.180
30	178	2534	2627	0.509	2593	0.487	2537	0.177
40	165	2380	2600	0.500	2562	0.457	2506	0.836
50	153	2378	2588	0.177	2549	0.192	2473	0.472
60	136	2103	2474	0.002	2433	0.002	2390	0.015
70	126	2015	2306	0.042	2284	0.051	2211	0.201
80	126	2082	2306	0.191	2284	0.336	2211	0.532
90	111	2025	2250	0.090	2256	0.015	2218	0.283
100	88	2112	2270	0.079	2309	0.025	2232	0.076
110	75	1818	1964	0.708	1981	0.533	1971	0.329
120	71	1780	1940	0.112	1971	0.027	1947	0.040

Table VII. MMRE with weighted growing and unweighted window

Window size(N)	Testing Projects	Window MRE	Triangular Growing MRE	Triangular Growing p–val.	Epanechnikov Growing MRE	Epanechnikov Growing p–val.	Gaussian Growing MRE	Gaussian Growing p–val.
20	201	1.13	1.35	0.651	1.35	0.730	1.31	0.396
30	178	1.15	1.42	0.716	1.42	0.962	1.37	0.737
40	165	1.12	1.43	0.154	1.44	0.207	1.38	0.406
50	153	1.14	1.47	0.004	1.48	0.009	1.42	0.013
60	136	1.09	1.45	0.000	1.46	0.000	1.46	0.000
70	126	1.23	1.50	0.011	1.51	0.009	1.52	0.029
80	126	1.20	1.50	0.009	1.51	0.013	1.52	0.018
90	111	1.14	1.43	0.086	1.42	0.016	1.45	0.082
100	88	1.11	1.39	0.024	1.39	0.000	1.41	0.001
110	75	1.15	1.43	0.466	1.43	0.205	1.48	0.151
120	71	1.10	1.39	0.018	1.41	0.003	1.44	0.001

a non-parametric statistical test: the *median* MRE and absolute residuals values supported weighted growing.

With the Gaussian function, Figure 4(c) shows that weighted growing is always better (all points are below the x-axis). Figure 5(c) has all points above the x-axis, but statistical tests actually supported weighted growing. Again, this contradiction was due to the use of a non-parametric statistical test.

We conclude that when retaining all training projects, weighted growing (gradual weighting) was effective compared to unweighted growing, weighting all projects equally.

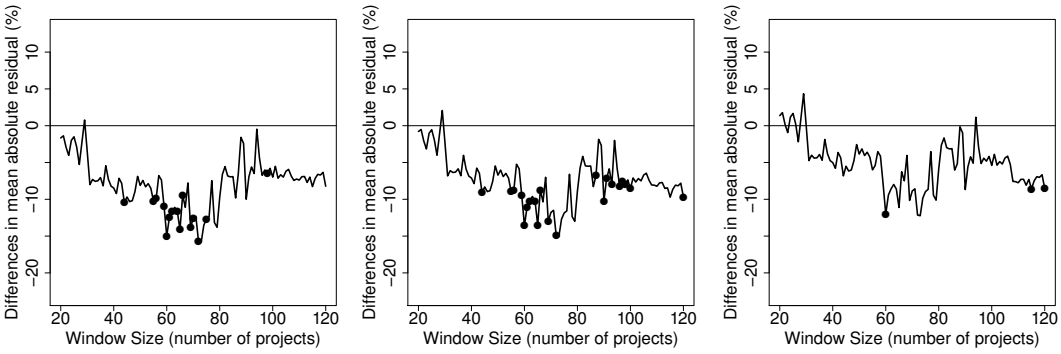
4.3. Accuracy Comparisons between Weighted Growing and Unweighted Window

The previous sections showed that weighted growing and unweighted window could each improve estimation accuracy, compared to unweighted growing. Weighted growing has significant effect, and its effect is stable across the range of window sizes. Unweighted Window has significant effect, and its effect seems related to window size. Here we compare them against each other.

Tables VI and VII show comparisons between unweighted windows and weighted growing in terms of MAE and MMRE. We can see window sizes where the difference is significant. Figures 6 and 7 show the difference in MAE and MMRE respectively. The y-axis is the subtraction of the accuracy measure value with weighted growing from that with unweighted window at the given x-value. Unweighted window is advantageous where the line is below 0. Circle points mean a statistically significant difference, in favor of unweighted window.

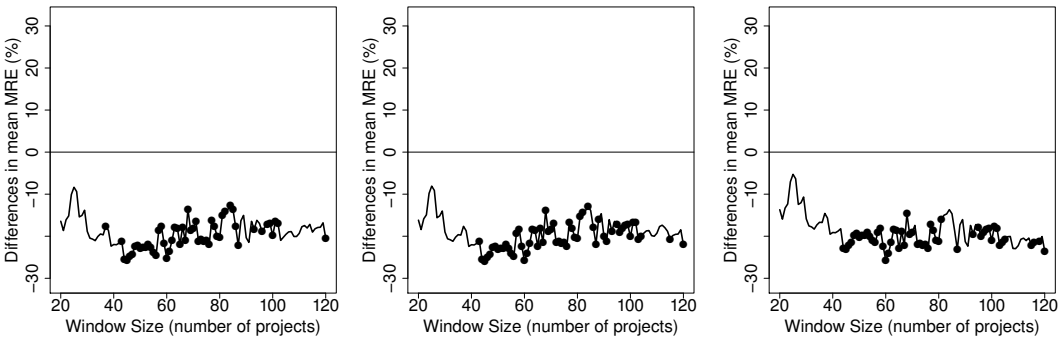
Fewer differences are statistically significant than in Figure 3. Especially, Figure 6(c) shows no statistically significant difference between weighted growing with Gaussian function and unweighted window in almost all window sizes, in terms of MAE. However, all figures only supported unweighted window. The lines are almost always below zero, and all statistically significant points support unweighted window. Figure 7 shows a clearer difference. We thus

DOI: 10.1002/smr



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 6. Percent difference in MAE between weighted growing and unweighted window



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 7. Percent difference in MMRE between weighted growing and unweighted window

Table VIII. Mean absolute residuals with unweighted growing and weighted window

Window size(N)	Testing Projects	Growing MAE	Triangular Window MAE	p–val.	Epanechnikov Window MAE	p–val.	Gaussian Window MAE	p–val.
20	201	2638	2728	0.223	2607	0.328	2791	0.112
30	178	2578	2610	0.416	2629	0.376	2689	0.181
40	165	2541	2504	0.773	2472	0.837	2538	0.600
50	153	2527	2479	0.351	2469	0.185	2480	0.558
60	136	2458	2283	0.050	2212	0.007	2286	0.016
70	126	2300	2055	0.001	2050	0.002	2057	0.012
80	126	2300	2073	0.006	2021	0.000	2111	0.004
90	111	2236	2060	0.012	2034	0.004	2069	0.011
100	88	2314	2011	0.027	2004	0.001	2100	0.083
110	75	1981	1702	0.004	1762	0.010	1673	0.012
120	71	1982	1703	0.002	1729	0.001	1642	0.001

conclude that unweighted window is more effective than weighted growing, if only one of them is to be applied.

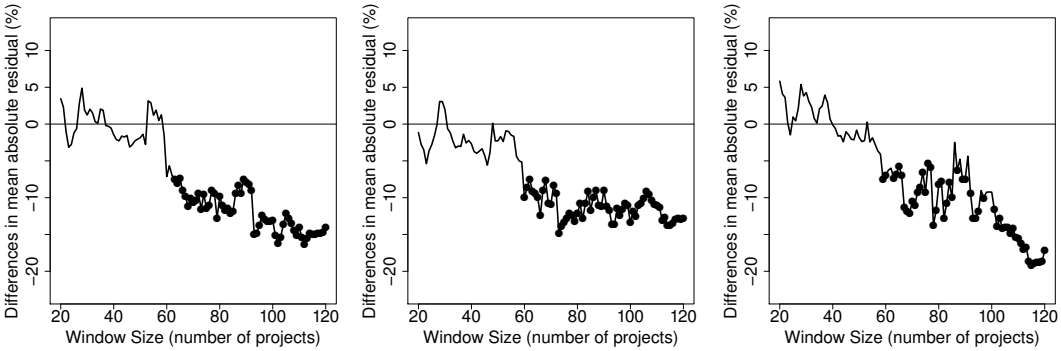
4.4. Accuracy Comparisons between Unweighted Growing and Weighted Window

Moving windows can be combined with gradual weighting. The combined method cuts off the old projects that do not fall within the window, and weights the projects within the window according to their age.

DOI: 10.1002/smr

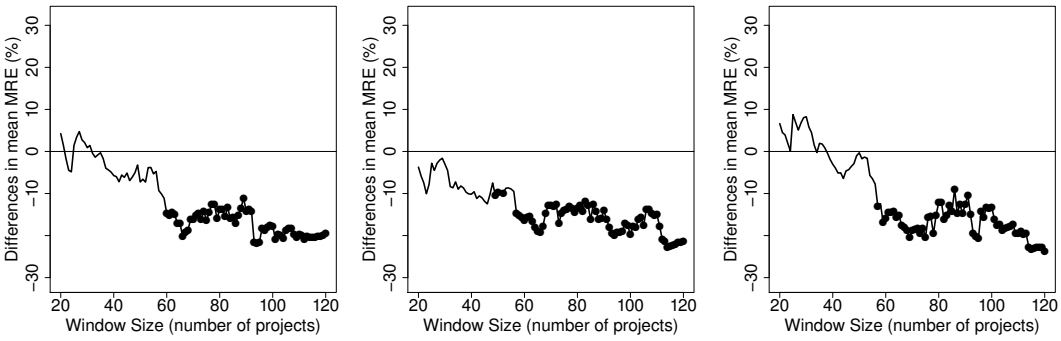
Table IX. MMRE with unweighted growing and weighted window

Window size(N)	Testing Projects	Growing MRE	Triangular Window MRE	Triangular Window p–val.	Epanechnikov Window MRE	Epanechnikov Window p–val.	Gaussian Window MRE	Gaussian Window p–val.
20	201	1.28	1.34	0.855	1.24	0.804	1.37	0.684
30	178	1.35	1.36	0.851	1.31	0.898	1.46	0.570
40	165	1.35	1.28	0.779	1.22	0.470	1.31	0.999
50	153	1.39	1.29	0.177	1.25	0.045	1.38	0.776
60	136	1.42	1.21	0.025	1.19	0.001	1.20	0.003
70	126	1.48	1.24	0.000	1.29	0.000	1.20	0.000
80	126	1.48	1.27	0.000	1.27	0.000	1.30	0.000
90	111	1.37	1.18	0.000	1.18	0.000	1.20	0.000
100	88	1.36	1.12	0.000	1.09	0.000	1.18	0.017
110	75	1.39	1.12	0.000	1.18	0.001	1.12	0.000
120	71	1.38	1.11	0.000	1.09	0.000	1.05	0.000



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 8. Percent difference in MAE between unweighted growing and weighted window



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 9. Percent difference in MMRE between unweighted growing and weighted window

Here, the question is whether the combination makes estimation accuracy better. To answer this question, we first examine the combined method (weighted window) against the baseline strategy of unweighted growing.

Tables VIII and IX show comparisons between unweighted growing and weighted window on MAE and MMRE. The tables show statistical significance for medium and large windows. Figures 8 and 9 depict the difference in MAE and MMRE. Here, the y-axis is the subtraction of the accuracy

DOI: 10.1002/smr

Table X. Significance of differences in MAE with weighted growing and weighted window

Window size(N)	Testing Projects	Triangular Weighting p–val. (MAE)	Epanechnikov Weighting p–val. (MAE)	Gaussian Weighting p–val. (MAE)
20	201	0.140	0.286	0.030
30	178	0.133	0.155	0.030
40	165	0.514	0.851	0.154
50	153	0.646	0.189	0.611
60	136	0.205	0.022	0.302
70	126	0.033	0.064	0.336
80	126	0.060	0.032	0.260
90	111	0.049	0.023	0.199
100	88	0.148	0.006	0.449
110	75	0.004	0.136	0.027
120	71	0.035	0.044	0.004

measure value with unweighted growing from that with weighted window at the given x-value. The weighted window is advantageous where the line is below 0.

The figures show common results:

- With the smallest windows, both MAE and MMRE tend to be better using unweighted growing (the points are above the x-axis). However, none of these differences are significant. The Epanechnikov function (shown in Figures 8(b) and 9(b)) appears to work slightly better than the other functions.
- In medium windows, weighted window becomes advantageous. The window size where it becomes advantageous is slightly different among types of weighted functions. In contrast to Figure 3, in which differences are significant for windows of 40 or more, here the first significant window size is around $w = 60$. On the other hand, the difference of MAE keeps below -5% around $80 \leq w \leq 100$ while unweighted window got worse around that range (see Figure 3(a)).
- With larger windows, MAE and MMRE are always better using weighted window, and the difference is significant in $100 \leq w \leq 120$. In contrast to Figure 3, the difference of MAE remains large for $100 \leq w \leq 120$. Gaussian function was better than the other functions.

These results showed that the combination reduced the window range of "sweet spots" but achieved better accuracy than unweighted window in larger windows.

4.5. Accuracy Comparisons between Weighted Growing and Weighted Window

Here we examine the combined method (weighted window) against weighted growing. This comparison also contributes to reveal how the combination method works better. Tables X and XI show comparisons between weighted growing and weighted window on MAE and MMRE. They were compared on the same weighted functions. For instance, the 3rd column shows p-values for them using the Triangular function. The tables show statistical significance for small and large windows. Figures 10 and 11 depict the difference in MAE and MMRE. Here, the y-axis is the subtraction of the accuracy measure value with weighted growing from that with weighted window at the given x-value. The weighted window is advantageous where the line is below 0.

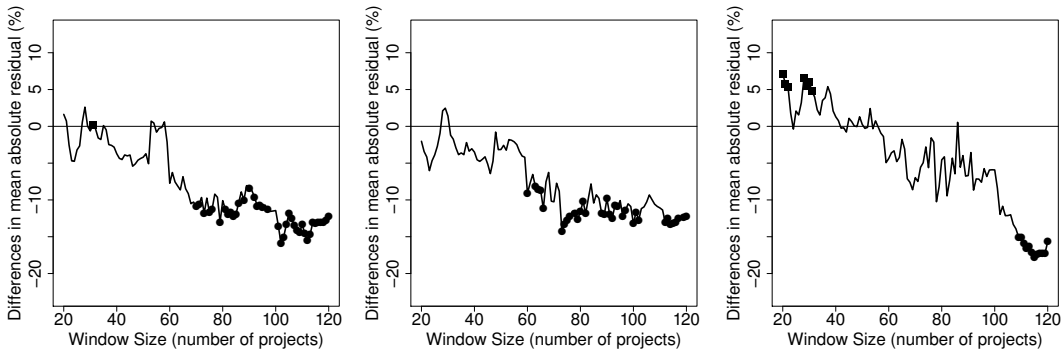
The figures show common results:

- With the smallest windows, MAE and MMRE both tend to be better using weighted growing. However, the difference is rarely significant for Triangular and Epanechnikov functions. Using the Gaussian function, weighted growing was better than weighted window, in terms of MAE.
- In medium windows, weighted windows become advantageous. The window size where it becomes advantageous differs for the weighted functions. These observations are the same as the comparison in Section 4.4.
- With larger windows, both MAE and MMRE are better with weighted window, and the difference is often significant. Gaussian function was better than the other functions in MAE.

DOI: 10.1002/smr

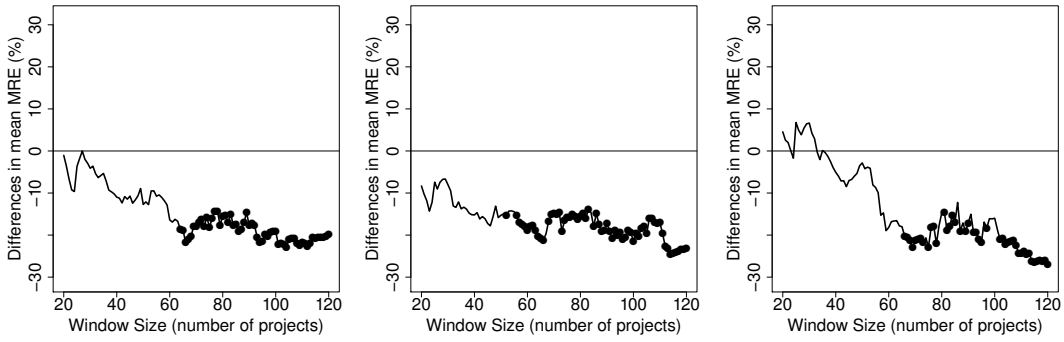
Table XI. Significance of differences in MMRE with weighted growing and weighted window

Window size(N)	Testing Projects	Triangular Weighting p–val. (MRE)	Epanechnikov Weighting p–val. (MRE)	Gaussian Weighting p–val. (MRE)
20	201	0.292	0.762	0.071
30	178	0.647	0.563	0.106
40	165	0.764	0.989	0.334
50	153	0.526	0.091	0.481
60	136	0.131	0.007	0.096
70	126	0.000	0.004	0.025
80	126	0.005	0.007	0.072
90	111	0.006	0.002	0.031
100	88	0.014	0.000	0.174
110	75	0.000	0.024	0.002
120	71	0.004	0.008	0.000



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 10. Percent difference in MAE between weighted growing and weighted window



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 11. Percent difference in MMRE between weighted growing and weighted window

These results show that the growing portfolio became better with gradual weighting, and the weighted growing is better than the combined approach (weighted window) in smaller windows. However, the weighted window was generally better than weighted growing because it was supported in wider ranges.

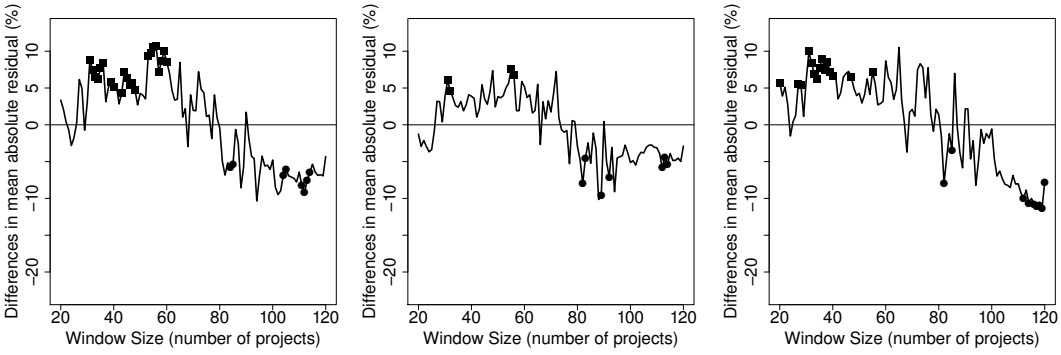
DOI: 10.1002/smr

Table XII. Mean absolute residuals with unweighted and weighted window

Window size(N)	Testing Projects	Window MAE	Triangular Window MAE	Triangular Window p–val.	Epanechnikov Window MAE	Epanechnikov Window p–val.	Gaussian Window MAE	Gaussian Window p–val.
20	201	2640	2728	0.176	2607	0.456	2791	0.044
30	178	2534	2610	0.360	2629	0.191	2689	0.054
40	165	2380	2504	0.033	2472	0.102	2538	0.045
50	153	2378	2479	0.167	2469	0.223	2480	0.096
60	136	2103	2283	0.015	2212	0.222	2286	0.069
70	126	2015	2055	0.976	2050	0.847	2057	0.698
80	126	2082	2073	0.650	2021	0.196	2111	0.727
90	111	2025	2060	0.753	2034	0.400	2069	0.568
100	88	2112	2011	0.705	2004	0.141	2100	0.967
110	75	1818	1702	0.056	1762	0.154	1673	0.111
120	71	1780	1703	0.525	1729	0.819	1642	0.047

Table XIII. MMRE with unweighted and weighted window

Window size(N)	Testing Projects	Window MRE	Triangular Window MRE	Triangular Window p–val.	Epanechnikov Window MRE	Epanechnikov Window p–val.	Gaussian Window MRE	Gaussian Window p–val.
20	201	1.13	1.34	0.038	1.24	0.097	1.37	0.014
30	178	1.15	1.36	0.026	1.31	0.019	1.46	0.003
40	165	1.12	1.28	0.000	1.22	0.001	1.31	0.000
50	153	1.14	1.29	0.145	1.25	0.306	1.38	0.070
60	136	1.09	1.21	0.004	1.19	0.080	1.20	0.047
70	126	1.23	1.24	0.805	1.29	0.837	1.20	0.676
80	126	1.20	1.27	0.790	1.27	0.438	1.30	0.987
90	111	1.14	1.18	0.284	1.18	0.180	1.20	0.193
100	88	1.11	1.12	0.967	1.09	0.200	1.18	0.594
110	75	1.15	1.12	0.031	1.18	0.070	1.12	0.068
120	71	1.10	1.11	0.470	1.09	0.973	1.05	0.015



(a) Triangular (b) Epanechnikov (c) Gaussian

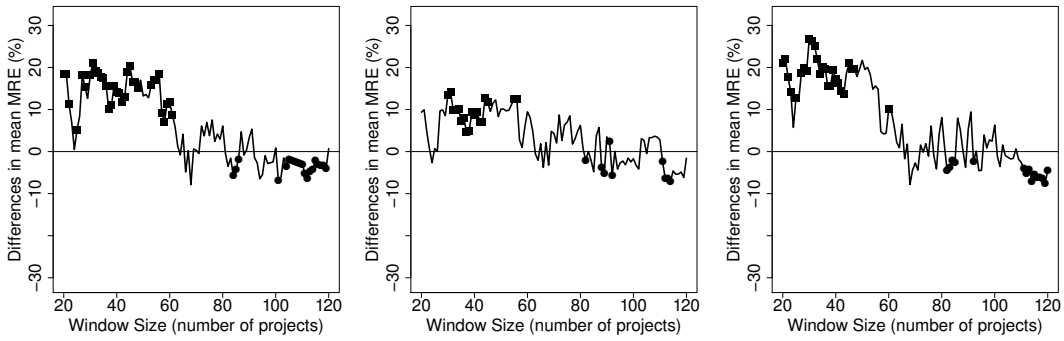
Figure 12. Percent difference in MAE between unweighted window and weighted window

4.6. Accuracy Comparisons between Unweighted Window and Weighted Window

The previous sections show that the combination of gradual weighting and moving windows has better accuracy than unweighted growing and weighted growing, and that the types of weighting functions affect estimation accuracy differently. This section examines in more detail how the weighting functions affect estimation accuracy.

Tables XII and XIII compare the effects of weighted window and unweighted window. There are window sizes showing statistical significance. Figures 12 and 13 show the difference in MAE and MMRE respectively. Weighted window is advantageous where the line is below 0. Figures 12 and 13 reveal the following:

DOI: 10.1002/smr



(a) Triangular (b) Epanechnikov (c) Gaussian

Figure 13. Percent difference in MMRE between unweighted window and weighted window

- With smaller windows, using unweighted window is advantageous. The difference is significant around $20 \leq w \leq 60$.
- With medium windows, both methods are competitive. There is no clear preference between them. There is almost no significant difference.
- With larger windows, weighted window is advantageous. The range of advantageous window sizes is different among types of weighted functions. The lines in Figure 12 are always below zero when $w > 80$ for the three functions. The lines in Figure 13 sometimes rise above zero, however statistical tests support only the weighted functions.

5. DISCUSSION

5.1. Answer to RQ1

In Section 4.2, H2 was supported for the difference between weighted growing with the three functions and unweighted growing.

For the Epanechnikov function, for instance, the null hypothesis was rejected in the whole range of window sizes. For the Triangular function, the range was narrower than that of Epanechnikov function for MMRE. However, the null hypothesis was rejected in the range of window sizes from 50 to 120. The improvement was small but the use of gradual weighting can affect estimation accuracy against unweighted growing.

In Section 4.6, H6 was supported for the difference between unweighted window and weighted window. The results show that unweighted window was advantageous in smaller windows. Weighted window becomes advantageous as the window size increases, and shows statistically significant improvement.

The above observations imply that gradual weighting affects estimation accuracy.

5.2. Answer to RQ2

In Section 4.3, H3 was supported for the difference between the effects of gradual weighting and moving windows. For smaller windows, gradual weighting was competitive to moving window. There was no statistically significant difference. For medium windows, significance tests only ever supported moving window. Large windows rarely supported either gradual weighting or moving window, especially for MAE. However, the small number of significant points all supported moving window. That is, there was a difference in the accuracy between them, and the effects of gradual weighting were less than those of moving window.

Prepared using smrauth.cls
DOI: 10.1002/smr

5.3. Answer to RQ3

In Section 4.4, H4 was supported for the difference between the effects of unweighted growing and weighted window. That is, the use of weighted window can also affect estimation accuracy against unweighted growing with large windows. The range of windows showing statistical significance is narrower than that of unweighted window. However, estimation accuracy becomes better in larger windows.

Section 4.5 shows the same trends and supports H5. Although weighted growing was sometimes advantageous and significantly better in small windows, the combined method was advantageous in larger windows.

We conclude that the combination worked well in larger windows.

5.4. Answer to RQ4

In Section 4.2, gradual weighting worked significantly. While the figures do not show clear differences in MAE and MMRE, the difference was statistically significant in a wide range of window sizes. The Triangular function only showed insignificant improvement.

In Section 4.6, using weighted window showed inferior estimation accuracy than using unweighted window, with small windows. The difference was statistically significant. In contrast, weighted window showed superior estimation accuracy when using larger windows. The difference was also statistically significant. Gaussian functions worked better than the other weighted functions.

The above characteristics can be explained by noting that there is an interaction between window sizes and the steepness of weighted function curves. The weighted functions all assign a small (near to zero) weight to the oldest projects in a window. The other projects receive progressively heavier weights in accordance with their chronological order within the window. With small windows, weighting functions assign steeply declining weights. With large window sizes, weighting functions assign more gently declining weights. When the degree of steepness meshes with a window size, a weighting function contributes to improved estimation accuracy. The same explanation applies to growing portfolio.

The difference in advantageous window sizes among weighted functions supports this explanation. Figure 2 depicts the difference of steepness among weighting functions. Gaussian is the steepest function, Epanechnikov is the most gentle function, and the steepness of the Triangular function is in between. Unweighted window assigns equal weights. In Figures 8 and 9, steeper functions became advantageous more slowly than gentle functions. The Epanechnikov function became advantageous with respect to MMRE at a smaller window size, for instance. For MAE, the Gaussian function was still unstable in medium windows while the Epanechnikov function became stable. This suggests that with small windows, the Gaussian function was too steep to reflect the importance of recent projects. With large windows, it meshed with window sizes and showed better results in estimation accuracy.

The results imply that gradual weighting and moving windows can work well together, and that assigning appropriate steepness to mesh them is crucial.

5.5. What are the practical implications of this study?

For companies using the whole past data, gradual weighting is suggested as another approach to effort estimation. Its effect is different from and could be combined with moving windows. Trying gradual weighting is valuable for companies where a simple cut-off by moving windows does not work, and keeps all projects for effort estimation.

Even for companies only considering recent project data by using a window, the results motivate managers to consider how each past project reflects their current situation. A moving window effectively accommodates changes in a company's practices by cutting off old projects. Gradual weighting adds another layer of detail to this, allowing a manager to reflect gradual change in the company by gradually adjusting the weight given to past projects.

DOI: 10.1002/smr

While it still remains difficult to define the best way to determine the weights, gradual weighting provides a chance to improve estimation accuracy.

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 the 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 a long time span, we assume it is a fair representation of this organization's projects. The inclusion of the 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 with this data set, 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 used weighted linear regression. Many effort estimation models have been proposed, and each model can show better accuracy in particular situations. However, linear regression is a popular and accurate effort estimation models. We think it is a good choice among major effort estimation methods. We see the use of other methods as an avenue for future work.

7. CONCLUSION

This paper investigated the use of weighted moving windows as a way to improve non-weighted moving windows. We have shown that it has a statistically significant effect on estimation accuracy in terms of MAE and MMRE. Although different weighting functions affect estimation accuracy differently, weighted moving windows are significantly advantageous in larger windows. Non-weighted moving windows are significantly advantageous with smaller windows.

What these results suggest is that it can be better to use a weighted window of projects with a weighted function having appropriate steepness. Weighted moving windows gradually decrease the importance of past projects. If a decrease curve is too steep or too gentle, weighted moving windows makes estimation accuracy worse. How to determine appropriate steepness is a crucial question.

Our future work is four-fold:

Generalization with other companies This study used one single-company dataset. We need to examine whether weighting and moving windows work in other companies. Public datasets rarely include time-stamps, but a few that do (such as Finnish, Maxwell and CSC) can contribute to knowing how and when the weighting approach is effective.

Generalization with other effort estimation models This study only used linear regression as an effort estimation method. Replications with other methods will help reveal to what extent the gradual weighting works. This requires development of methods to apply gradual weighting, analogous to case weighting for linear regression, for other effort estimation methods.

DOI: 10.1002/smr

Duration-based windows In this data set, with unweighted windows, windows containing a fixed number of projects and windows of fixed duration can both lead to more accurate estimates, but the effect is stronger with windows containing a fixed number of projects. That may not be the case with weighting methods. Thus, replications with windows based on fixed durations is worthwhile.

Optimization procedure for better estimation This study revealed that gradual weighting works, on its own and combined with moving windows, and that its steepness is crucial. How to find the best window size and steepness still remains in question. A good starting point is to analyze the relationship and continuity of window sizes between successive spans. Tracking organizational change records such as process improvement may be valuable as an approach other than metrics-driven optimization.

ACKNOWLEDGEMENTS

A part of this work is supported by Grant-in-Aid for Scientific Research (C) 25330083, Japan Society for the Promotion of Science, and Wesco Scientific Promotion Foundation.

REFERENCES

1. Port D, Korte M. Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research. *Proc. of the 2nd ACM-IEEE international symposium on Empirical software engineering and measurement*, ACM, 2008.
2. Jørgensen M, Shepperd M. A Systematic Review of Software Development Cost Estimation Studies. *IEEE Trans Softw Eng* 2007; **33**(1):33–53.
3. Lokan C, Mendes E. Applying moving windows to software effort estimation. *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, IEEE Computer Society, 2009; 111–122.
4. Auer M, Biffl S. Increasing the accuracy and reliability of analogy-based cost estimation with extensive project feature dimension weighting. *Proc. of International Symposium on Empirical Software Engineering*, IEEE, 2004; 147–155.
5. Mendes E, Lokan C. Investigating the use of chronological splitting to compare software cross-company and single-company effort predictions: a replicated study. *Proceedings of the 13th Conference on Evaluation & Assessment in Software Engineering (EASE 2009). BCS*, 2009.
6. Keung JW, Kitchenham BA, Jeffery DR. Analogy-X: Providing Statistical Inference to Analogy-Based Software Cost Estimation. *IEEE Trans Softw Eng* 2008; **34**(4):471–484.
7. Li J, Ruhe G. Analysis of attribute weighting heuristics for analogy-based software effort estimation method AQUA+. *Empir Softw Eng* 2007; **13**(1):63–96.
8. Maxwell KD. *Applied Statistics for Software Managers*. Prentice Hall, 2002.
9. Kitchenham B, Lawrence Pfleeger S, McColl B, Eagan S. An empirical study of maintenance and development estimation accuracy. *J. Syst. Softw.* 2002; **64**(1):57–77.
10. MacDonell SG, Shepperd M. Data accumulation and software effort prediction. *Proc. of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ACM, 2010.
11. Amasaki S, Lokan C. The Effects of Moving Windows to Software Estimation: Comparative Study on Linear Regression and Estimation by Analogy. *2012 Joint Conf of 22nd Int'l Workshop on Software Measurement and the 7th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA)*, IEEE, 2012; 23–32.
12. Lokan C, Mendes E. Investigating the Use of Duration-Based Moving Windows to Improve Software Effort Prediction. *Software Engineering Conference (APSEC), 2012 19th Asia-Pacific*, 2012; 818–827.
13. Amasaki S, Lokan C. The Evaluation of Weighted Moving Windows for Software Effort Estimation. *PROFES '13: Proceedings of the 14th International Conference on Product Focused Software*, 2013; 214–228.
14. Mendes E, Lokan C. Replicating studies on cross- vs single-company effort models using the ISBSG Database. *Empir Softw Eng* Feb 2008; **13**(1).
15. Loader C. *Local Regression and Likelihood*. Statistics and Computing, Springer, 1999.
16. Tabachnick BG, Fidell LS. *Using Multivariate Statistics*. Harper-Collins, 1996.
17. Tibshirani R. Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B* 1996; :267–288.

DOI: 10.1002/smr