Software Qual J (2011) 19:165–200
DOI 10.1007/s11219-010-9110-y

ORIGINAL PAPER

Comparison of weighted grey relational analysis for software effort estimation

Chao-Jung Hsu · Chin-Yu Huang

Published online: 14 September 2010

Abstract In recent years, grey relational analysis (GRA), a similarity-based method, has been proposed and used in many applications. However, we found that most traditional GRA methods only consider nonweighted similarity for predicting software development effort. In fact, nonweighted similarity may cause biased predictions, because each feature of a project may have a different degree of relevance to the development effort. Therefore, this paper proposes six weighted methods, including nonweighted, distance-based, correlative, linear, nonlinear, and maximal weights, to be integrated into GRA for software effort estimation. Numerical examples and sensitivity analyses based on four public datasets are used to show the performance of the proposed methods. The experimental results indicate that the weighted GRA can improve estimation accuracy and reliability from the nonweighted GRA. The results also demonstrate that the weighted GRA performs better than other estimation techniques and published results. In summary, we can conclude that weighted GRA can be a viable and alternative method for predicting software development effort.

Keywords Software effort estimation · Grey relational analysis (GRA) · Weighted assignment · Software development effort · Software cost

1 Introduction

One of the long-existing challenges faced by software project managers is to predict software development effort and cost[1] (Boehm 1981). Accurate and reliable software effort estimation is the foundation of successful project management. Generally, software project managers need to obtain sufficient information regarding the resource distribution to make correct decisions at early development stages (Conte et al. 1986). The allocation of appropriate resources and the planning of reasonable schedules based on the effort

[1] Software development effort can be further used to estimate software development cost. Therefore, the terms “software effort” and “software cost” are generally used in other studies in this field.

C.-J. Hsu · C.-Y. Huang (✉)
Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
e-mail: cyhuang@cs.nthu.edu.tw

estimation then become necessary. Furthermore, the software development process usually includes debugging and testing phases. Without the foresight of development effort, projects may conflict with the level of quality demanded and may possibly encounter failure. For example, underestimating the effort needed for software development may possibly cause the software product to have insufficient time to be tested and consequently force programmers to sacrifice software quality.

On the other hand, software quality cannot be viewed in isolation. In the past, Boehm's advanced cost model was usually tied to a quality model. A report based on a sample of 63 completed projects showed that a reduction in overall costs and improved productivity can come from applying formal methods or measurement activities (Boehm 1981; Boehm et al. 1995). Besides, a recent survey from Agrawal and Chari (2007) found that many CMM level 5 projects incorporate estimation methods to determine software effort, quality control, and cycle time in a software development process. On average, the usage of estimation methods can significantly predict effort and cycle time around 12% and defects to about 49% of the actual. From the above information, we can note that an accurate and reliable effort estimation technique may be important to both software development process and software quality assurance (Ejiogu 2005; Fenton and Pfleeger 1998).

Many studies have been conducted to investigate different kinds of effort estimation techniques (Jørgensen and Shepperd 2007). Expert judgment, algorithmic models, and similarity-based methods are the main categories of software effort predictions (Boehm 1981; Conte et al. 1986). Generally, the similarity-based method is based on a similarity comparison (usually Euclidean distance) between project features and software development effort (Marir and Watson 1994; Shepperd and Schofield 1997). The nearest neighbour algorithm is generally used to find the most similar project. However, the similarity-based methods still have some drawbacks for application. Many studies have aimed to improve the estimated performance of similarity-based methods (Chiu and Huang 2007; Jørgensen et al. 2003; Leung 2002; Li et al. 2007a).

In recent years, grey relational analysis (GRA), one of the similarity-based methods, has been used extensively in many scientific fields (Liu and Lin 2006). Nevertheless, GRA has rarely been applied to estimate software development effort. The similarity of GRA measures the relative distance between project features and maximal or minimal distance differences (Deng 1989; Song et al. 2005; Wen et al. 2006). We find that most of the existing GRA-based software effort estimation methods only adopt nonweighted (or equally weighted) similarity of project features (Hsu and Huang 2006; Li et al. 2007a; Song et al. 2005). In fact, the relevant features should be given more influential and significant weights in similarity computations. The problem is that equally weighted features will cause downgrades to similarity computations (Auer et al. 2006; Huang and Chiu 2006; Keung and Kitchenham 2007). By contrast, improper weights assigned to irrelative features may contrarily cause a biased determination and could thereby affect the estimated performance (Li and Ruhe 2006; Li et al. 2009b). As a result, how to appropriately determine the weight for each feature may become a research problem when using weighted GRAs.

Therefore, in this paper, we will propose six weighted methods to be integrated into the conventional GRA. Numerical examples based on four datasets and some comparative criteria are used to demonstrate estimated performance. Furthermore, a sensitivity analysis between the parametric settings and the analogous numbers is discussed, and other estimation techniques and published results are then used as comparisons. Finally, we also present some guidelines and management metrics for using weighted GRAs. The following propositions are addressed in this paper. (1) The extent of weighted alterations involved in the GRA methods may lead to improvement in the estimated accuracy and reliability.

(2) The different parametric settings and analogous numbers of the GRA methods may be a factor that affects the predicted result. (3) Weighted GRA may be an alternative and feasible method for predicting software development effort in the software development life cycle.

The remainder of this paper is structured as follows. In Sect. 2, we provide a survey of software effort estimation and basic concepts of the GRA method. After that, the proposed methods and experimental procedures are presented in Sect. 3. The explorative studies and numerical results will be demonstrated by the comparative criteria and sensitivity analysis in Sect. 4. Finally, a concluding discussion is described in Sect. 5.

2 Literature review

2.1 Software effort estimation survey

Over the past three decades, a variety of techniques have been proposed in the field of software development effort estimation (Jørgensen and Shepperd 2007). To begin with, Boehm presented two parametric software effort models–the constructive cost model (COCOMO I) (1981) and COCOMO II (1995), both of which are widely applied in practice (Benediktsson et al. 2003; Li et al. 2007b). The effort multipliers of COCOMO I and COCOMO II are used to capture characteristics of the software development that affect the effort to complete the project. If developers want to undertake effort estimation for a specific software project, they need to carefully examine their development process and grade a proper rating for effort multipliers.

Subsequently, the similarity-based method, such as case-based reasoning (CBR) (Marir and Watson 1994; Mendes et al. 2002a) or analogy (Shepperd and Schofield 1997), was discussed and developed along with similar historical projects. Several studies then tried to improve the performance of analogy by adjusting its similarity measure (Chiu and Huang 2007; Li et al. 2007a). Later, attention turned to other software effort estimation techniques unlike the parametric models, including neural network (NN) and classification and regression tree (CART) (Srinivasan and Fisher 1995), genetic algorithm (GA) (Huang and Chiu 2006), and fuzzy theory (Lima Júnior et al. 2003). These studies show that software effort estimation is an important issue in the software development process.

The GRA hardly appears to be used in software effort estimation, although it has been used in many other areas. Song et al. (2005) first introduced a GRA method called GRACE to predict software development effort that possessed superior merits. Later, Li et al. (2007a) adopted Song's method as a comparative method for evaluating the accuracy of different methods. In our previous work (Hsu and Huang 2006; Hsu and Huang 2007), we also proposed an improved grey method to enhance predicted results, but the parametric settings, analogous numbers, and sensitivity analyses were not completely considered. On the other hand, most of the existing GRA-based methods only adopted nonweighted (or equally weighted) similarity of project features. In fact, nonweighted similarity measures may cause biased predictions, because each project feature may have a different degree of relevance to the development effort. All these problems motivated us towards continuous research on GRA-based methods.

More recently, some studies have improved the traditional similarity-based method by attaching weights on project features for similarity computation. Mendes et al. (2002a, b) developed a weighted CBR for estimating web hypermedia development effort and comparing several regression methods. They claimed that using the weighted CBR in an implementation stage to predict hypermedia development effort was more accurate than using the nonweighted CBR. Li and Ruhe (2006) evaluated weighted heuristics in the analogy method. The results indicated that the estimated effort of the weighted heuristic performed better than that of the equally weighted heuristic. Huang and Chiu (2006) integrated GA into an analogy to determine the weighted similarity of software effort features. They suggested that the weighted analogy should prove a feasible approach to improve the accuracy of software effort estimates. Auer et al. (2006) proposed a brute-force approach applied to an analogy to determine the optimal weights for each project feature. Li et al. (2009b) combined project selection technique and feature weighting with analogy to improve the prediction performance. These studies provide a good basis for introducing weights into similarity-based approaches.

2.2 Conventional GRA

In 1982, Deng introduced grey theory. After that, the grey theory has been applied to a wide range of applications (Deng 1982; Liu and Lin 2006). The grey of a system is absolute, and the fuzziness of a system is relative. The “grey” refers to the information between “black” and “white” (Deng 1989). The “black” means that the required information is totally unknown or unclear, while the “white” indicates that the required information is fully explored. Incomplete information brings great difficulties in the limited availability of data. Grey theory provides a helpful mechanism for seeking the intrinsic information of the system and does not need a specific relationship as an assumption (Song et al. 2005). Since most software projects have incomplete information and uncertain relations between project features and required development effort, grey theory may be suitable to be introduced into software effort estimation.

GRA is a quantitative technique that can be used to analyse the similarity among objects (e.g., software projects). This similarity is the measure of the relative distance between the pairs of object features. According to the definition, if the basic relationship between the features of two respective objects is close, their similarity will be highly related (Deng 2000). Before introducing the detailed computations of the GRA method, a matrix of multi-index sequences should be defined:

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_i \end{bmatrix} = \begin{bmatrix} X_1(1) & X_1(2) & \dots & X_1(k) & X_1(Dep) \\ X_2(1) & X_2(2) & \dots & X_2(k) & X_2(Dep) \\ \vdots & \vdots & \dots & \vdots & \vdots \\ X_i(1) & X_i(2) & \dots & X_i(k) & X_i(Dep) \end{bmatrix}$$

$$= [X(1) \quad X(2) \quad \dots \quad X(k) \quad X(Dep)], \tag{1}$$

where $i = 1, 2,\ldots,N$, and N is the total number of projects; $k = 1, 2,\ldots,M$, and M is the total number of features of a project. Each sequence X_i represents a project consisting of M features, and each $X(k)$ represents the kth feature of dataset X. These features can be numeric or categorical values. $X_i(Dep)$ stands for a dependent variable that denotes the known effort of the ith project.

Next, one new sequence is regarded as an *observed project*, which wants to predict its development effort and is used to compare similarity with other projects. Other sequences in dataset X are taken as *comparative projects*; the known efforts of which can be used as a comparable base for deriving an estimated effort for the observed project. The degree of similarity can be calculated by comparing these two sequences:

$$X_0 = [X_0(1) \quad X_0(2) \quad \ldots \quad X_0(k)], \tag{2}$$

and

$$X_i = [X_i(1) \quad X_i(2) \quad \ldots \quad X_i(k)], \tag{3}$$

where X_0 is the observed project with k features, and each X_i is a comparative project. The similarity measure between the features of the observed project and that of the comparative project is defined as the grey relational coefficient (GRC) (Deng 2000; Wen et al. 2006):

$$\gamma(X_0(k), X_i(k)) = \frac{\min \Delta_{0i} + \zeta \max \Delta_{0i}}{\Delta_{0i}(k) + \zeta \max \Delta_{0i}}, \tag{4}$$

where

$$\Delta_{0i}(k) = \begin{cases} |X_0(k) - X_i(k)|, & \text{if } X_0(k) \text{ and } X_i(k) \text{ are numerical} \\ 1, & \text{if } X_0(k) \text{ and } X_i(k) \text{ are categorical and } X_0(k) \neq X_i(k) \\ 0, & \text{if } X_0(k) \text{ and } X_i(k) \text{ are categorical and } X_0(k) = X_i(k) \end{cases}, \tag{5}$$

and

$$\min \Delta_{0i} = \overset{\min}{\forall}\, i \overset{\min}{\forall}\, k |X_0(k) - X_i(k)|, \tag{6}$$

$$\max \Delta_{0i} = \overset{\max}{\forall}\, i \overset{\max}{\forall}\, k |X_0(k) - X_i(k)|. \tag{7}$$

Notice that ζ stands for a *distinguishing coefficient* that is limited from 0 to 1. In Eq. (4), the GRC scale takes both the global maximum difference and the global minimum difference into account. Thus, its similarity can be seen as a measurement that is distinct from traditional similarity-based methods.

Finally, the grey relational grade (GRG) between the observed project X_0 and the comparative project X_i can be quantified by giving an average value of the GRCs as follows (Liu and Lin 2006):

$$\Gamma_{0i} = \frac{1}{M} \sum_{k=1}^{M} \gamma(X_0(k), X_i(k)). \tag{8}$$

The GRG value can be treated as a comparable basis and applied to similarity judgment. For instance, if the similarity order is $\Gamma_{0,a} > \Gamma_{0,b}$, the comparative project X_a is much closer to the observed project X_0 than the project X_b is.

3 Weighted GRA

3.1 Weighted GRA

The relative importance between the project features and development effort should be considered within the similarity measure. When the weighted similarity of the GRA method is taken into account, Eq. (8) can be modified as follows:

$$\Gamma_{0i} = \sum_{k=1}^{M} \beta_k \gamma(X_0(k), X_i(k)), \tag{9}$$

where

$$\sum_{k=1}^{M} \beta_k = 1. \tag{10}$$

Notice that β_k is a stationary weight given to the kth feature. Because the relationship between the project features and development effort is still an open issue (Dolado 2001), applying a weighted GRA then poses an annoying problem in determining the appropriate weight for each feature. From previous studies (Jørgensen and Shepperd 2007), human judgment or expert opinion could be one of the solutions to assigning feature weights. However, experts may be reluctant to set the weights manually due to the additional effort required to analyse project features, and expert opinion is somehow subjective. Therefore, we propose six weighted methods based on statistical techniques as follows.

3.1.1 Nonweight (or equal weight)

In the general case, the nonweight or equal weight can be defined as:

$$\beta_k = \frac{1}{M}, \tag{11}$$

where M is the total number of features, meaning that each feature has an equal impact on similarity computations. Obviously, both Eqs. (8) and (11) are a special case of Eq. (9). This method is used as a baseline method for our experiments.

3.1.2 Distance-based weight

The distance measurement compares dissimilarity corresponding to the dependent variable (i.e., known effort) (Freedman et al. 1997). The distance-based weight can be defined as:

$$\beta_k = \frac{\frac{1}{\text{Distance}(k)}}{\sum_{k=1}^{M} \frac{1}{\text{Distance}(k)}}, \tag{12}$$

where

$$\text{Distance}(k) = \sqrt{\sum_{i=1}^{N} (X_i(k) - X_i(Dep))^2}. \tag{13}$$

Equation (13) is a kind of *Euclidean distance* (Marir and Watson 1994). Accordingly, features with a close distance to the dependent variable should be assigned a higher weight.

3.1.3 Correlative weight

Similarly, we can also use correlation analysis to determine feature weights. The correlation coefficient calculates any of a wide variety of similarities. If a feature's correlation coefficient corresponding to the dependent variable is significant, the feature and the dependent variable will exhibit a perfect relationship. The correlative weight is defined as:

$$\beta_k = \frac{|\mathrm{Correlation}(k)|}{\sum_{k=1}^{M} |\mathrm{Correlation}(k)|}, \tag{14}$$

where *Correlation*(*k*) denotes a *Pearson correlation coefficient* between the *k*th feature and the dependent variable (Hogg and Craig 1995). If a correlation coefficient is negative, the absolute value is taken.

3.1.4 Linear weight

The linear weight assumes that there is a linear relationship between the dependent variable and the independent variables (i.e., project features). The linear function can be defined as follows (Dolado 2001; Huang and Chiu 2006; Jørgensen et al. 2003):

$$X(Dep) = \sum_{k=1}^{M} a_k X(k) + c, \tag{15}$$

$$\beta_k = \frac{|a_k|}{\sum_{k=1}^{M} |a_k|}, \tag{16}$$

where a_k is a coefficient corresponding to the *k*th feature and *c* is a constant. The coefficients a_k are an aggregation of independent variables that affect the dependent variable in a linear manner. Thus, these coefficients can somehow show the different degree of relevance to effort and can be translated into a correspondent weight β_k for the *k*th feature.

3.1.5 Nonlinear weight

By contrast, the functional form between the software development effort and project features may also be assumed to be a nonlinear relationship (Dolado 2001; Huang and Chiu 2006; Jørgensen et al. 2003). The nonlinear function can be defined as follows:

$$X(Dep) = \sum_{k=1}^{M} a_k X(k)^{b_k} + c, \tag{17}$$

where b_k is an exponent of the *k*th feature. This nonlinear relationship adjusts the independent variables more dramatically than the linear relationship. The coefficient a_k can also be transferred into correspondent weights β_k, similar to Eq. (16). Note that the coefficients of linear and nonlinear equations can be solved computationally (Freedman et al. 1997; Hogg and Craig 1995).

3.1.6 Maximal weight

In an extreme case, if we only consider a maximum similarity to determine a maximal weight, the weights of other features in which their similarities are smaller than the maximum are set to zero. This weighted assignment follows an assumption that the minor similarity features may not greatly affect the computations for retrieving the most similar case. The maximal weight can be defined as:

$$\beta_k = \begin{cases} 1, & \text{if } \gamma(X_0(k), X_i(k)) = \max \gamma(X_0(k), X_i(k)), \text{and } X_i(k) \text{ is numeric} \\ 0, & \text{otherwise} \end{cases}, \tag{18}$$

where

$$\max \gamma(X_0(k), X_i(k)) = \overset{\max}{\forall}\, i \overset{\max}{\forall}\, k\, \gamma(X_0(k), X_i(k)). \tag{19}$$

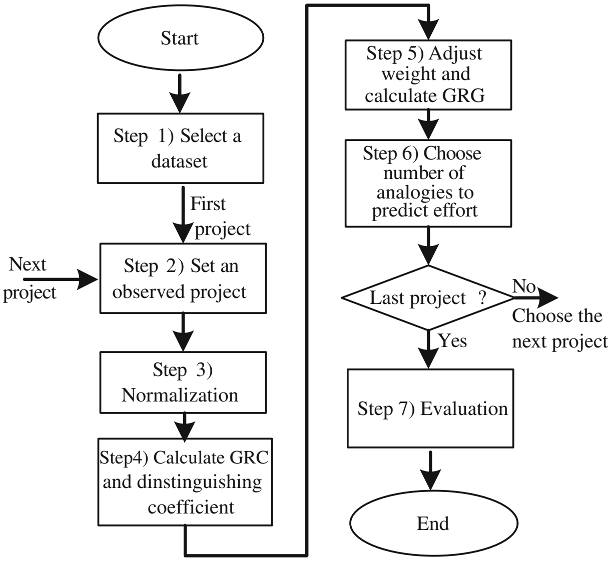
Consequently, this weighted method effectively decreases the use of similarity features down to one. This method can also be treated as a comparative method in the following experiments.

3.2 Implementation

In order to evaluate the weighted GRAs, the experimental procedure is illustrated in Fig. 1. The description of each step is presented as follows:

Step 1 Collect data from past projects in order to estimate the effort for development in the new project. In this paper, we use public datasets and choose the first project to start the procedure.
Step 2 Set the chosen project as an observed project.
Step 3 Normalize project sequences to range from 0 to 1 (Hsu and Huang 2006; Liu and Lin 2006; Song et al. 2005).
Step 4 Calculate GRC between the comparative projects and the observed project and set the distinguishing coefficient.
Step 5 Adjust an appropriate weight for each project feature and rank several of the most similar projects from the same dataset.
Step 6 Choose a suitable number of analogous projects to predict the development effort for the observed project. If all projects in the dataset have been estimated, this procedure goes to Step 7. Otherwise, the procedure will choose the next project and go back to Step 2.
Step 7 Evaluate the estimated performance by using comparison criteria.

Fig. 1 Experimental procedure


Start
Step 1) Select a dataset
First project
Next project
Step 2) Set an observed project
Step 3) Normalization
Step4) Calculate GRC and dinstinguishing coefficient
Step 5) Adjust weight and calculate GRG
Step 6) Choose number of analogies to predict effort
Last project ?
No
Choose the next project
Yes
Step 7) Evaluation
End

As shown in Fig. 1, the experimental procedure is an iterative loop. Consider N projects in the dataset. In each loop, one project is chosen from the dataset as the observed project, which is used as a testing example in order to estimate software development effort. The other N-1 projects are treated as a training base (i.e., comparative projects) to compare similarity. This procedure runs N times until the last project in the dataset is estimated. This kind of validation is a part of "leave-one-out cross-validation" (also known as jackknife validation) (Huang and Chiu 2006; Shepperd and Schofield 1997; Song et al. 2005).

Some settings for the experiments need to be discussed: (1) datasets and feature selection; (2) the similarity measure and distinguishing coefficient; (3) effort adaptation and number of analogous project; (4) evaluation criteria and statistical tests.

3.2.1 Datasets and feature selection

Four real datasets were used to conduct the experiments. The detailed information of these datasets is shown in Table 1. These datasets are widely used as a comparative standard in many other studies (Huang and Chiu 2006; Jeffery et al. 2000; Li and Ruhe 2006; Liu et al. 2008; Mendes et al. 2005; Samson et al. 1997; Shepperd and Schofield 1997; Srinivasan and Fisher 1995).

The raw data of the ISBSG repository originally contained 2027 projects (ISBSG 2006; Liu et al. 2008). For this reason, a suitable project subset is derived with the following selection criteria. The data quality rating is selected with the rating code of "A" and "B" denoted by the ISBSG reviewers, and the derived count approach with the "IFPUG" standard is chosen. In addition, the implementation date after the year 2000 is filtered out. Finally, projects that have missing values on the selected features are excluded, and two outlying observations are removed. This preprocessing process results in 127 projects in the ISBSG dataset.

Feature selection is of primary importance for software effort estimation (Cuadrado-Gallego et al. 2006), the purpose of which is to find some features that have significant impact on development effort. From Conte's study (1986), effort factors can be generally classified into four categories: people, process, product, and computer. Additionally, Boehm (1981) delineates several cost drives, which can be grouped into four categories including product, computer, personnel, and project attributes. According to these classifications, we

Table 1 Datasets information

Dataset	Cases	Feature	Missing value	Software size Mean and SD	Effort Mean and SD
Kemerer (1987)	15	6	No	186.57	219.24
				136.81	263.01
COCOMO (Boehm 1981)	63	8	No	66.57	683.52
				140.31	1821.51
Desharnais (1989)	77	9	No	284.74	4833.90
				174.76	4188.18
ISBSG[a] (2006)	127	8	Yes	154.89	3419.41
				160.05	4850.43

[a] The subset of raw data

Table 2 Selected features for each dataset

Dataset	Feature	Variable	p Value
Kemerer	Software size	Ind.	0.0006
	Schedule	Ind.	0.0206
	Development effort	Dep.	–
COCOMO	Software size	Ind.	0.0000
	Schedule	Ind.	0.0000
	Effort multiplier	Ind.	0.0010
	Development effort	Dep.	–
Desharnais	Software size	Ind.	0.0000
	Number of person	Ind.	0.0192
	Counting unit	Ind.	0.0000
	Development effort	Dep.	–
ISBSG	Software size	Ind.	0.0000
	Schedule	Ind.	0.0000
	Maximum team size	Ind.	0.0002
	Development effort	Dep.	–

Ind independent variable, *Dep* dependent variable

tentatively select six candidate features for the Kemerer dataset, nine features for the Desharnais dataset, and eight features for the COCOMO and ISBSG datasets. Then, we adopt statistical methods to analyse these candidates and select the most representative features in Table 2. It is noted that the categorical features use one-way ANOVA (Freedman et al. 1997), and the ratio features use Pearson's correlation test (Hogg and Craig 1995).

3.2.2 *Similarity measure and distinguishing coefficient*

The similarity measure is used to measure the degree of similarity between projects. After normalization, Eqs. (9, 10) and the weighted methods in Eqs. (11–19) are adopted to compute similarity. However, the distinguishing coefficient in Eq. (4) can decrease the effect of $\max\Delta_{0i}$ and thus change the magnitude of GRC significantly. Therefore, this coefficient should be carefully determined in advance (Liu and Lin 2006; Song et al. 2005; Wen et al. 2006). In the later experiments, we will use a sensitivity analysis to adjust the distinguishing coefficient and observe its influence on estimation accuracy.

3.2.3 *Effort adaptation and number of analogous projects*

From the similarity order, the development effort of an observed project can be estimated based on the known effort of the most similar projects. Here, we adopt the mean value of the closest projects to derive an estimated effort, which is given as (Huang and Chiu 2006; Mendes et al. 2002a, b):

$$E^*_{X_0} = \frac{1}{S} \sum_{\substack{X_i \in \text{similarity} \\ \text{order}}}^{S} E_{X_i}, \tag{20}$$

where all X_i belong to the results of the similarity order; S is the number of analogous projects we decide to choose; E^* is the predicted effort of the observed project X_0; and E is the known effort of each comparative project X_i.

In Eq. (20), a difficult problem arises concerning the decision regarding how many analogous projects to use when generating the predicted effort. Essentially, selecting suitable analogies in the similarity-based methods may be a factor that affects the predicted results (Hsu and Huang 2006; Li et al. 2007a; Song et al. 2005; Walkerden and Jeffery 1999). However, most studies only investigate a small range of analogous projects. In later experiments, we will also consider this factor and use sensitivity analysis to explore its influence on estimation accuracy.

3.2.4 Evaluation criteria and statistical tests

In an attempt to examine the experimental results, several predefined criteria are depicted as follows (Conte et al. 1986). A common criterion for evaluating the accuracy of effort estimations is defined in terms of the *Mean Magnitude of Relative Error* (MMRE or Mean MRE):

$$\mathrm{MMRE} = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|E_{X_i} - E^{*}_{X_i}\right|}{E_{X_i}}, \tag{21}$$

where N is the total number of observations and E^{*} is the prediction of known effort E. In general, MMRE is the relative error for a large group of estimates.

Another commonly used criterion is the *Prediction* (PRED) threshold, which can be defined as follows:

$$\mathrm{PRED}(l) = \frac{k}{N}, \tag{22}$$

where k is the number of observations whose MRE is less than or equal to level l. This criterion may not precisely show the improvement of estimates. If the level l is larger, the accuracy improvement will be more sensitive, but less confidence in the accuracy estimates will then be obtained (Korte and Port 2008; Port and Korte 2008). In our experiments, we use the level $l = 0.25$, because we can easily compare different published results among models (Song et al. 2005; Srinivasan and Fisher 1995).

However, MMRE and PRED may on occasions appear to give inconsistent results (Foss et al. 2003). For this reason, the variance of relative error and the boxplot of MRE can be complementarily used to evaluate the performance of prediction models (Auer et al. 2006; Chiu and Huang 2007; Jeffery et al. 2001; Li et al. 2009a). The variance can be used as a measure of the estimation method's reliability, whereas the boxplot is a type of graph that is used to display the shape of MRE distribution, central value, outlier values, extreme values, quartiles, and interquartile ranges. It is noted that the outlier denotes the value between 1.5 and 3 box lengths from the upper or lower edge of the box, and the extreme denotes a value more than 3 box lengths from the upper or lower edge of the box, where the box length is the interquartile range (Freedman et al. 1997). In general, a boxplot with a small box length or fewer outliers and extreme values usually has a reliable prediction capacity.

For determining the statistical significance among the methods, we further apply a sign test and confidence interval. Since the MRE values are positively skewed and nonnormal, a nonparametric method called *Wilcoxon Signed Rank Sum Test* (Briand et al. 2000; Jeffery et al. 2000, 2001) is used to conduct the sign test. On the other hand, the confidence interval of PRED can be determined by the standard error (Korte and Port 2008; Port and Korte 2008):

$$\mathrm{SE}_{\mathrm{PRED}(0.25)} \approx \frac{\mathrm{SD}_{\mathrm{MRE} \leq 0.25}}{\sqrt{N}}, \tag{23}$$

where SE denotes the standard error, and SD denotes the standard deviation whose MREs are less than or equal to 0.25. For both the sign test and the confidence interval, the significance level is set at $\alpha = 0.05$.

4 Experiments and discussions

In the following, two experiments are used to demonstrate the estimated performance. The first experiment investigates the sensitivity analysis between distinguishing coefficients and the number of analogous projects. The second experiment compares the weighted GRAs with other estimation methods and published results. Tabular and graphic results using the four datasets are presented.

4.1 Experiment 1: sensitivity analysis between distinguishing coefficients and analogous numbers

The first experiment can further be divided into two parts. First, a pilot experiment is used to separately observe the effect of distinguishing coefficients and analogous numbers on estimated accuracy. Second, a sensitivity analysis of changing both the distinguishing coefficients and the analogous numbers is investigated.

4.1.1 Comparison of accuracy with distinguishing coefficients and analogous numbers

In this experiment, the distinguishing coefficients are increased by an increment of 0.1. A summary of MMRE against four datasets is shown in Table 3.

From Table 3, there is a downward tendency in the MMRE criterion when the value of the distinguishing coefficient is increased from 0.1 to 0.5. Thereafter, the values of MMRE

Table 3 MMRE criterion of increasing distinguishing coefficients

Dataset	ζ									
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Kemerer										
MMRE	**0.618**	**0.618**	0.573	0.573	*0.571*	*0.571*	*0.571*	*0.571*	*0.571*	*0.571*
COCOMO										
MMRE	0.824	0.812	**0.832**	0.818	*0.755*	*0.755*	*0.755*	*0.755*	*0.755*	*0.755*
Desharnais										
MMRE	**0.380**	0.344	0.334	0.325	*0.323*	0.325	0.325	0.329	0.329	0.327
ISBSG										
MMRE	**1.772**	**1.772**	**1.772**	*1.577*	*1.577*	*1.577*	*1.577*	*1.577*	*1.577*	*1.577*

The italic number indicates the most accurate within one dataset

The bold number indicates the least accurate within one dataset

Analogous numbers = 3, and weighting approach = nonweight

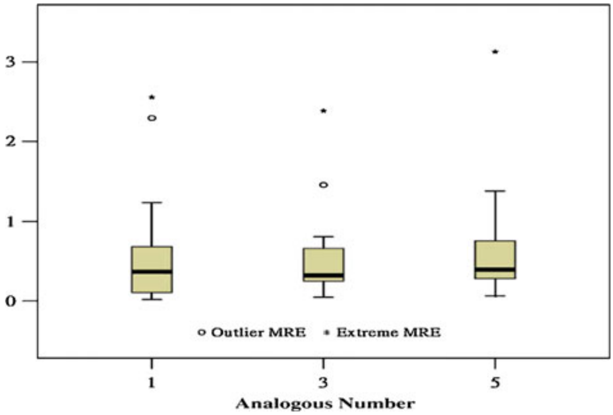
Table 4 Accuracy of analogous numbers

Datasets	Kemerer			COCOMO			Desharnais			ISBSG		
Analogous numbers	1	3	5	1	3	5	1	3	5	1	3	5
GRA												
MMRE	**0.653**	*0.571*	0.650	**0.865**	*0.755*	0.790	*0.278*	0.323	**0.360**	**2.693**	1.577	*1.557*
PRED	*0.333*	**0.200**	**0.200**	*0.158*	*0.158*	**0.142**	*0.623*	0.610	**0.571**	**0.192**	**0.192**	*0.240*

The italic number indicates the most accurate within one dataset; the bold number indicates the least accurate within one dataset

Distinguishing coefficient = 0.5, and weighting approach = nonweight

Fig. 2 Boxplot of MRE with Kemerer dataset



becomes stable in the four datasets. We find that the most accurate MMRE seems to lie in the area from 0.5 to 1 in three out of four datasets. Hence, the effect of choosing an appropriate distinguishing coefficient should be considered in GRA methods.

Next, a pilot experiment with different analogous numbers is performed to observe the effect on estimated accuracy. To begin with, we decide to choose analogous numbers “1”, “3”, and “5” as a small scale experiment, similar to some past studies (Jeffery et al. 2001; Mendes et al. 2002a, b). The estimated accuracy of the four datasets is shown in Table 4, and Figs. 2, 3, 4 and 5.

In Table 4, the result shows an accuracy improvement when there is an increase in the analogous number from “1” to “3” in terms of MMRE criterion for the Kemerer and COCOMO datasets, and from “1” to “5” in terms of MMRE and PRED criteria for the ISBSG dataset, but for the Desharnais dataset the analogous number “1” has the most accurate MMRE and PRED. Additionally, the boxplots from Figs. 2, 3, 4 and 5 indicate that the analogous number “3” presents the least variant of the MRE distribution in the Kemerer and COCOMO datasets, the analogous number “1” has the smallest value of outlier and extreme MREs in the Desharnais dataset, and the analogous number “5” shows the most accurate of MREs in the ISBSG dataset. Hence, we can also observe that different analogous numbers may cause a great impact on estimated accuracy.

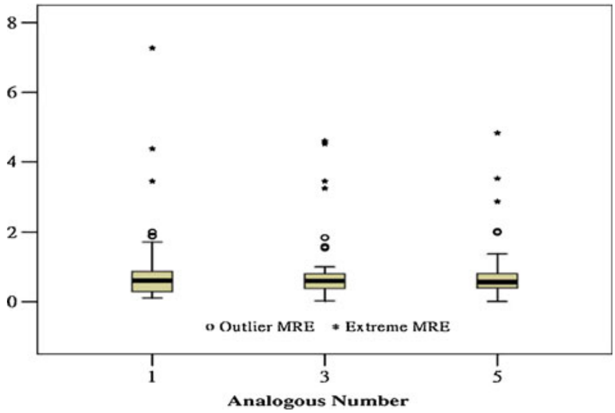


Fig. 3 Boxplot of MRE with COCOMO dataset

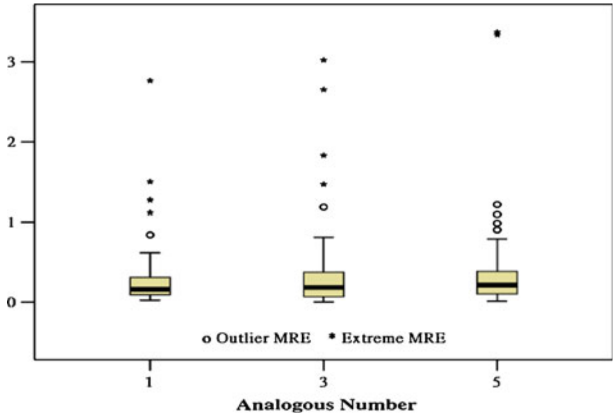


Fig. 4 Boxplot of MRE with Desharnais dataset

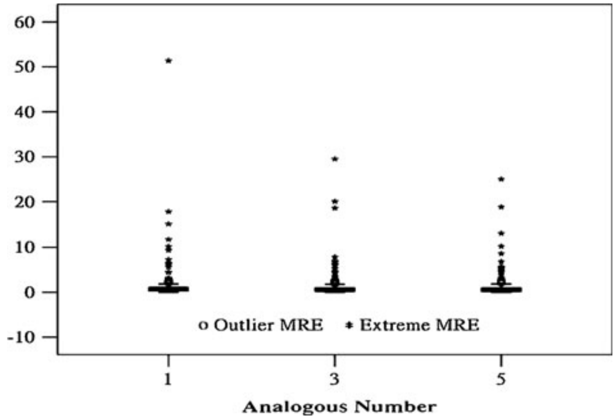


Fig. 5 Boxplot of MRE with ISBSG dataset

4.1.2 Sensitivity analysis between analogous numbers and distinguishing coefficients

In the following, a sensitivity analysis is performed by increasing both the distinguishing coefficient and the analogous number. The graphic results are shown in Figs. 6, 7, 8, 9, 10, 11, 12 and 13 with respect to four datasets. In the end, the distribution of accuracies for these figures is illustrated in Table 5.

From Figs. 6 and 7, the most precise MMRE is located at the analogous number “3”, and the most precise PRED is found at the analogous number “1”. If we only look at the different values of distinguishing coefficients, the MMRE and PRED values seem not to change too much with the same analogous number. This finding substantially agrees with the previous pilot experiments that the analogous number is a more influential factor than the distinguishing coefficient. In Figs. 8 and 9, the analogous number “4” is the most accurate, but the results with PRED fluctuate. However, we can still roughly see that PRED with the analogous numbers “18” to “36” seems to be the most accurate. Similarly, in Figs. 10 and 11, the most accurate area of MMRE and PRED lies in the analogous numbers from “1” to “3”, and from Figs. 12 and 13 the best value of MMRE and PRED is located around the analogous number “25”.

In short, we can see that both the distinguishing coefficient and analogous number may have different impacts on estimation accuracy. Figures 6, 7, 8, 9, 10, 11, and 12, 13 illustrate that the use of analogous numbers can refine the accuracy much more than that of distinguishing coefficients. On the other hand, we can also see that selecting a small analogous number seems to be more accurate with the MMRE and PRED criteria. Actually, many studies also suggested that a small number of analogies had acceptable accuracy for

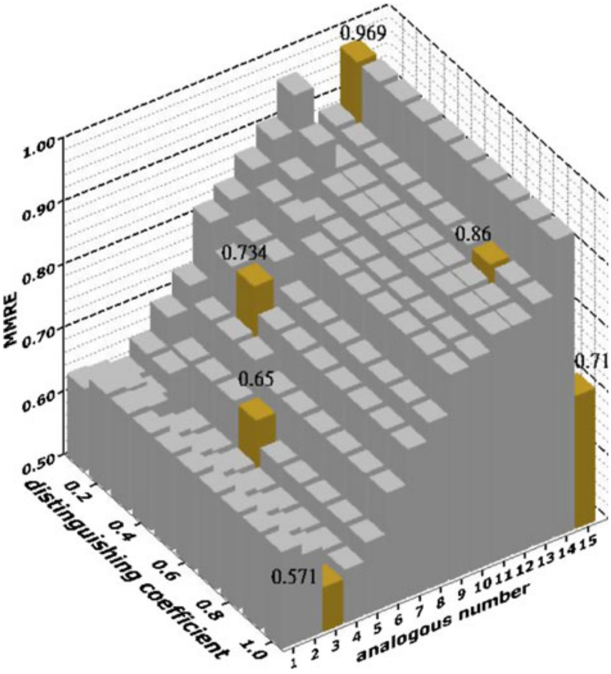


Fig. 6 MMRE of Kemerer dataset

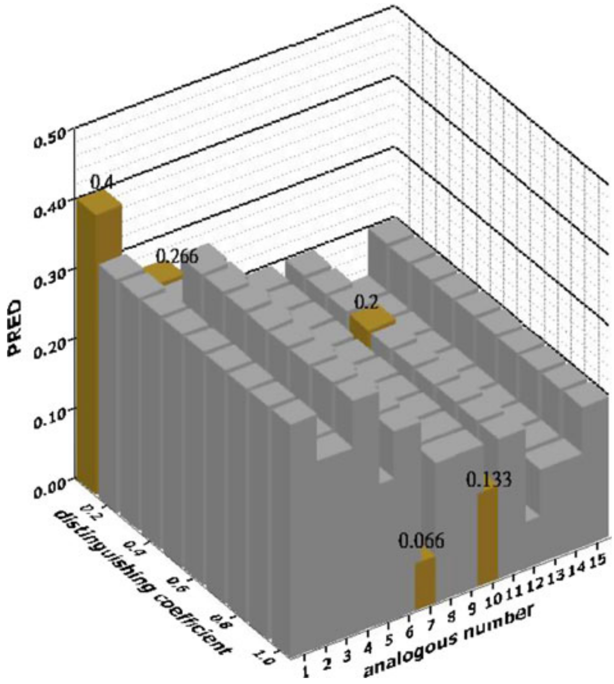


Fig. 7 PRED of Kemerer dataset

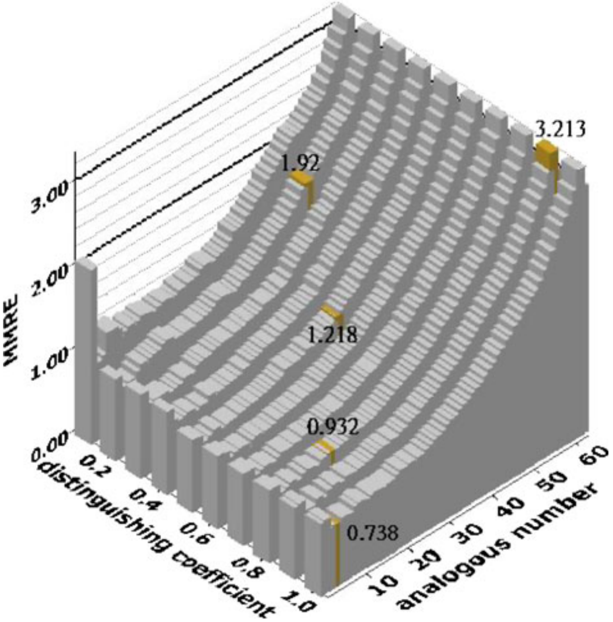


Fig. 8 MMRE of COCOMO dataset

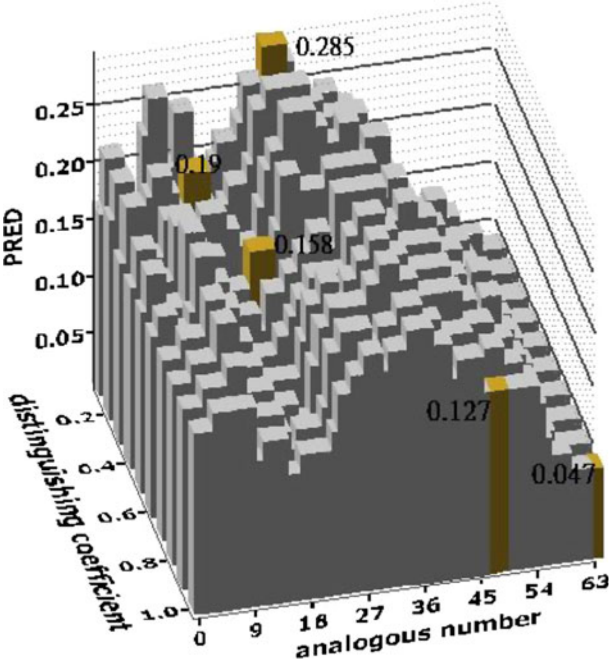


Fig. 9 PRED of COCOMO dataset

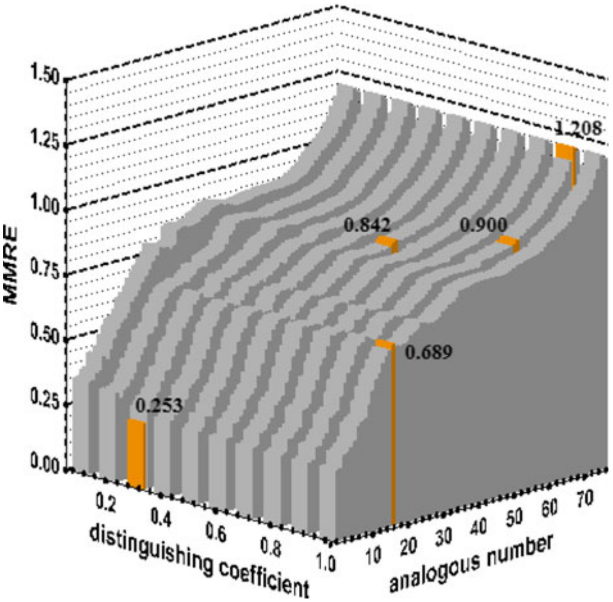


Fig. 10 MMRE of Desharnais dataset

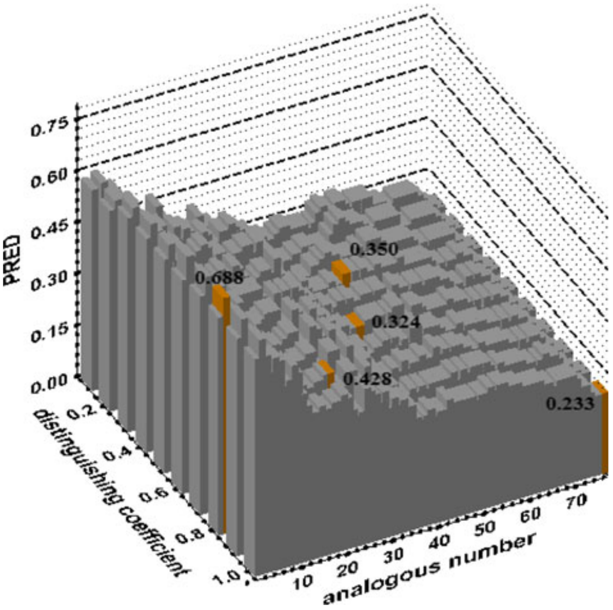


Fig. 11 PRED of Desharnais dataset

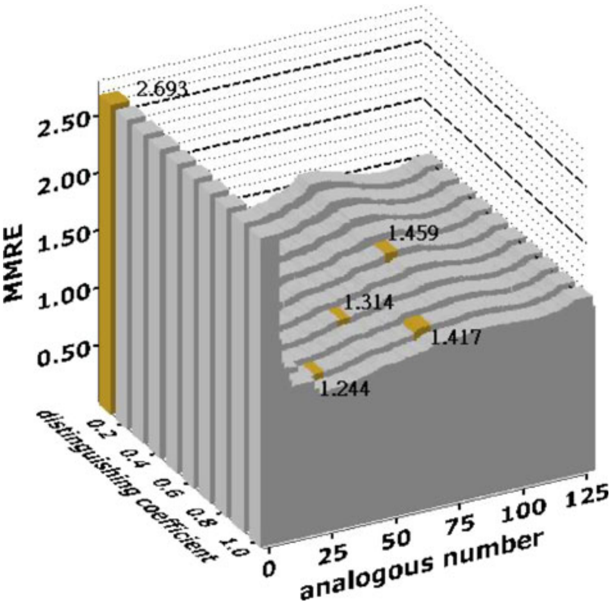


Fig. 12 MMRE of ISBSG dataset

software effort prediction (Huang and Chiu 2006; Jeffery et al. 2000; Li et al. 2009b). For example, Song et al. (2005) chose analogous numbers “1–5” to aggregate the estimated effort for small datasets, and Li et al. (2007) adopted analogous numbers “3”, “24–25”, and

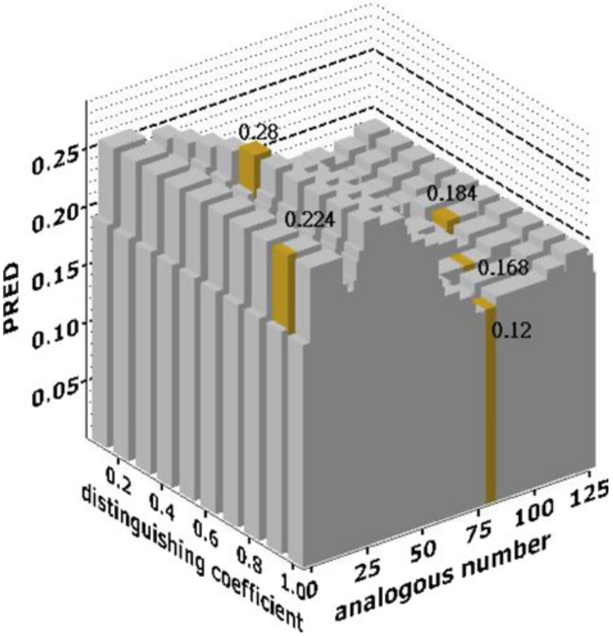


Fig. 13 PRED of ISBSG Dataset

Table 5 Distribution of accuracies over four datasets

Datasets	Distributions				
	Min	Max	Median	25%	75%
Kemerer					
MMRE	0.571	0.969	0.734	0.650	0.860
PRED	0.066	0.400	0.200	0.133	0.266
COCOMO					
MMRE	0.738	3.213	1.218	0.932	1.920
PRED	0.047	0.285	0.158	0.127	0.190
Desharnais					
MMRE	0.253	1.208	0.842	0.689	0.900
PRED	0.233	0.688	0.350	0.324	0.428
ISBSG					
MMRE	1.244	2.693	1.417	1.314	1.459
PRED	0.120	0.280	0.184	0.168	0.224

"74–75" to predict effort for the ISBSG dataset. Consequently, we suggest that "5" may be the best number for most small datasets. As for the ISBSG dataset, the analogous number should not be greater than half of the project cases.

4.2 Experiment 2: comparison of accuracy with weighted GRA

In the second experiment, the performance of weighted GRA and nonweighted GRA is compared first. Other estimation techniques and published results are then used as comparisons.

4.2.1 Comparison of accuracy between weighted and nonweighted GRA

Six weighted GRAs, including nonweight (NW), distance-based weight (DW), correlative weight (CW), linear weight (LW), nonlinear weight (NLW), and maximal weight (MW), are used to demonstrate the performance of the proposed methods. The comparative criteria are shown in Tables 6 and 7. It is noted that comparisons are made up of four analogous numbers (“3”, “5”, “10”, and “20”) based on previous experiments. The statistical test between the nonweighted and weighted GRA is presented in Table 8. Finally, boxplots of MRE with four datasets are shown in Figs. 14, 15, 16 and 17.

For the Kemerer dataset in Table 6, the CW gives the best results when compared to the NW with the MMRE and PRED criteria, and the other three weighted methods, LW, NLW, and MW, offer similar values in the MMRE criterion. Besides, we can find that the analogous number “3” is preferable to the number “5” in the Kemerer dataset. In Table 7, the CW also has the smallest variance of all weighted methods. Further, in Table 8 four weighted methods (CW, LW, NLW, and MW) show a significant difference from the NW GRA in both analogous numbers “3” and “5”. Finally, the boxplot of Fig. 14 shows that the five weighted methods (DW, CW, LW, NLW, and MW) have smaller box length in the distribution of the MRE when compared with the NW GRA. Therefore, for the Kemerer dataset, all of these experimental results indicate that the estimates obtained by the weighted GRA, especially by the CW GRA, appear to be more accurate and reliable than those using the NW GRA. For the most improved percentages, CW can improve the MMRE by 11.23% and NW improves PRED by 66.5%.

For the COCOMO dataset in Table 6, we can clearly see that the LW performs the most accurate prediction in MMRE and PRED criteria, and the DW, CW, and NLW obtain close results. In particular, in Table 7 the LW GRA can further improve variance when compared with other weighted methods. In Table 8, it is noted that the *p*-value of DW, CW, and LW shows enough evidence to perform better than the NW in analogous numbers “3” and “5”. On the other hand, Fig. 15 shows that four weighted methods (DW, CW, LW, and NLW) have less extreme MRE values against the NW and MW methods. However, based on our findings, the MW gives the least accurate results of all methods in the COCOMO dataset. The basic concept of MW only assigns the closest feature with a maximal weight while neglecting the effects of the other minor features. This may cause bias in the similarity order and may further influence the estimation accuracy. When the feature numbers increase, the information from the minor features becomes important in determining similarity order.

Similarly, for the Desharnais dataset in Table 6, we can see that the LW is the most accurate in both MMRE and PRED criteria, and the CW and NLW provide close values that are only next to LW in all analogous numbers. Besides, in Table 7 we can see that the LW GRA can greatly decrease the variance of estimates from the NW GRA in the analogous numbers “3”, “5”, and “10”. In Table 8, the statistical results show that the CW, LW, and NLW GRAs can provide more accurate estimates when compared with the NW GRA. On the other hand, from Fig. 16 the CW, LW, and NLW show a smaller range of MRE distribution and fewer extreme values when compared with the other methods. As for

Table 6 Comparison of accuracy between weighted and nonweighted GRA

Datasets	Analogous number	Weighted methods											
		NW		DW		CW		LW		NLW		MW	
		MMRE	PRED	MMRE	PRED	MMRE	PRED	MMRE	PRED	MMRE	PRED	MMRE	PRED
Kemerer	3	**0.571**	**0.200**	0.570	0.266	*0.551*	0.266	0.566	*0.333*	0.566	*0.333*	0.566	*0.333*
	5	**0.650**	**0.200**	0.594	0.266	*0.577*	*0.333*	0.583	*0.333*	0.583	*0.333*	0.583	*0.333*
COCOMO	3	0.755	**0.158**	0.715	*0.285*	0.696	0.222	*0.582*	*0.285*	0.751	0.206	**1.196**	0.206
	5	0.790	0.142	0.722	0.222	0.680	0.142	*0.610*	*0.253*	0.676	**0.095**	**1.229**	0.142
	10	0.821	0.174	0.692	0.142	0.701	**0.095**	*0.644*	0.174	0.702	0.142	**1.049**	*0.222*
Desharnais	3	0.323	0.610	0.261	0.675	0.225	0.740	*0.167*	*0.831*	0.234	0.818	**0.729**	**0.298**
	5	0.360	0.571	0.292	0.636	0.241	0.701	*0.209*	*0.740*	0.239	0.714	**0.707**	**0.272**
	10	0.490	0.532	0.405	0.623	0.325	0.623	*0.271*	*0.701*	0.288	0.649	**0.664**	**0.272**
ISBSG	3	**1.577**	*0.192*	1.564	*0.192*	1.564	*0.192*	*1.553*	**0.128**	*1.553*	**0.128**	*1.553*	**0.128**
	5	**1.557**	0.240	1.515	*0.248*	1.515	*0.248*	*1.480*	0.208	*1.480*	**0.168**	*1.480*	**0.168**
	10	**1.347**	0.216	1.332	0.216	1.330	0.224	*1.328*	*0.232*	1.329	*0.232*	1.333	**0.192**
	20	1.252	0.200	1.221	0.216	1.222	0.200	*1.214*	*0.224*	1.223	0.200	**1.265**	**0.176**

The italic number indicates the most accurate for an analogous number

The bold number indicates the least accurate for an analogous number

Table 7 Comparison of variance between weighted and nonweighted GRA

Datasets	Analogous number	Weighted Methods					
		NW Variance	DW Variance	CW Variance	LW Variance	NLW Variance	MW Variance
Kemerer	3	**1.349**	0.903	*0.698*	0.713	0.713	0.713
	5	**1.557**	1.287	*0.920*	0.934	0.934	0.934
COCOMO	3	2.346	1.490	0.925	*0.656*	2.027	**3.458**
	5	1.948	1.652	0.817	*0.663*	0.683	**3.270**
	10	1.550	0.670	0.702	*0.652*	0.872	**2.329**
Desharnais	3	0.266	0.133	0.117	*0.073*	0.114	**0.885**
	5	0.311	0.224	0.166	*0.093*	0.178	**0.627**
	10	0.480	0.460	0.202	*0.139*	0.159	**0.508**
ISBSG	3	**15.201**	15.158	15.158	*12.148*	*12.148*	*12.148*
	5	**12.439**	12.235	12.235	*8.615*	*8.615*	*8.615*
	10	**8.798**	7.974	7.969	7.960	*7.958*	8.647
	20	7.809	7.262	7.244	7.167	*7.140*	**8.057**

The italic number indicates the smallest variance for an analogous number

The bold number indicates the biggest variance for an analogous number

the improvement percentage of LW in the Desharnais dataset, the MMRE and PRED criteria can be separately improved by 48.29% and 36.22%.

For the ISBSG dataset in Table 6, four weighted methods (DW, CW, LW, and NLW) can improve accuracy against the NW GRA, and the LW seems to offer the most accurate prediction in the MMRE with the exception of PRED with the analogous number "3". Likewise, although MW provides an improvement in MMRE, the accuracy of PRED cannot be further increased for any analogous numbers. In Table 7, both LW and NLW GRAs have a reduction in variance when compared with the NW GRA in all analogous numbers. In Table 8, we find that DW, CW, LW, and NLW are significantly different from NW in most analogous numbers, but there is not enough difference with MW in the analogous numbers "5" and "20". Additionally, Fig. 17 shows that the MRE distributions of NW and MW are odd with a large interquartile in the analogous number "20".

As for improvement percentages in the ISBSG dataset, the weighted methods only show a small improvement when compared with the other datasets. The reason may be related to the dataset properties. In fact, the projects in the ISBSG dataset are voluntarily provided by a broad range of industries in the world, and therefore may come from different applications (ISBSG 2006; Liu et al. 2008; Mendes et al. 2005). Consequently, the subset of the ISBSG dataset is highly heterogeneous (refer to Table 1). Some studies have shown that underlying dataset characteristics may be influential in favoring the inhibiting of different estimation techniques (Chen et al. 2005; Jørgensen et al. 2003). Additionally, other studies show that adopting an ISBSG dataset while using different estimation techniques may lead to less accuracy (Huang and Chiu 2006; Jeffery et al. 2000; Jeffery et al. 2001).

4.2.2 Comparison of accuracy with other methods

In order to further evaluate the performance of weighted GRA, we compare four estimation methods in this experiment: analogy (Shepperd and Schofield 1997), linear regression (LR)

Table 8 Wilcoxon signed rank sum test between weighted and nonweighted GRA

(2-Tailed, $\alpha = 5\%$)		NW vs. DW		NW vs. CW		NW vs. LW		NW vs. NLW		NW vs. MW	
Datasets	Analogous number	*z* Statistic	*p* Value	*z* Statistic	*p* Value	*z* Statistic	*p* Value	*z* Statistic	*p* Value	*z* Statistic	*p* Value
Kemerer	3	1.988	0.046*	2.158	0.031*	2.045	0.040*	2.045	0.040*	2.045	0.040*
	5	2.108	0.035*	2.741	0.006*	2.556	0.010*	2.556	0.010*	2.556	0.010*
COCOMO	3	2.978	0.003*	2.827	0.004*	2.951	0.003*	1.588	0.112	−2.157	0.031*
	5	2.595	0.009*	2.328	0.020*	2.649	0.008*	1.534	0.125	−2.136	0.032*
	10	1.890	0.058	1.807	0.070	3.033	0.002*	2.109	0.035*	−0.726	0.468
Desharnais	3	1.304	0.192	2.904	0.003*	5.037	0.000*	3.153	0.001*	−4.928	0.000*
	5	3.040	0.002*	4.540	0.000*	3.410	0.006*	4.364	0.000*	−4.669	0.000*
	10	3.245	0.001*	4.719	0.000*	5.080	0.000*	5.024	0.000*	−2.689	0.007*
ISBSG	3	2.872	0.004*	2.872	0.004*	2.017	0.042*	2.023	0.043*	2.017	0.042*
	5	3.054	0.002*	3.056	0.002*	1.834	0.066	1.835	0.066	1.834	0.066
	10	2.771	0.005*	2.800	0.005*	3.022	0.002*	3.022	0.002*	1.995	0.046*
	20	2.889	0.003*	2.852	0.004*	2.877	0.004*	2.714	0.006*	1.103	0.270

* Denotes a significant difference existed between two models at significance level

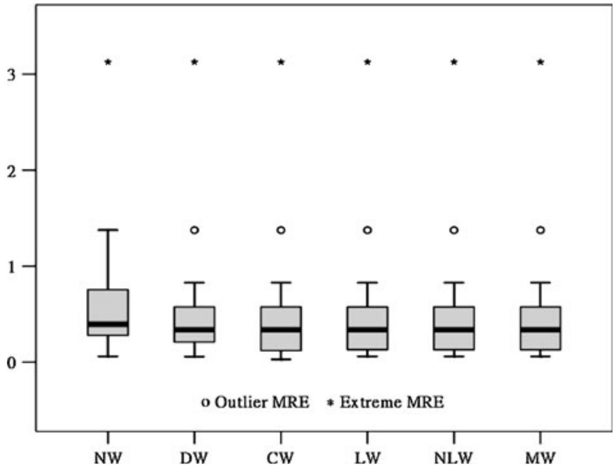


Fig. 14 Boxplot of MRE with Kemerer dataset (analogous number = 5)

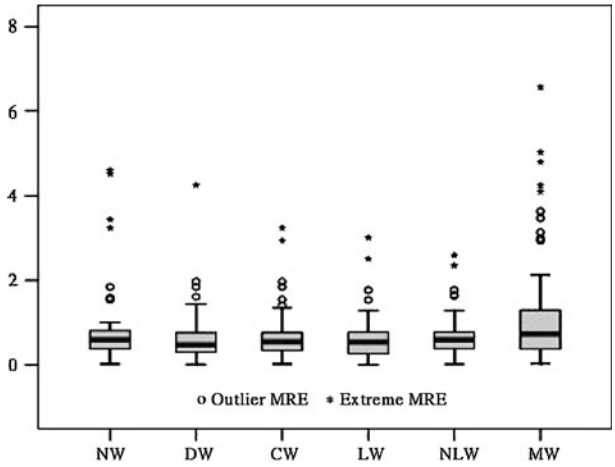


Fig. 15 Boxplot of MRE with COCOMO dataset (analogous number = 10)

(Freedman et al. 1997), nonlinear regression (NLR) (i.e., polynomial form) (Hogg and Craig 1995), and basic COCOMO model without effort multipliers (Boehm 1981; Boehm et al. 1995; Li et al. 2007b). From previous experiments, we choose NW GRA and LW GRA as comparisons, because NW is a baseline method and LW is stable in most datasets.

Before beginning the following experiments, each model setup is introduced. In the analogy method, all features are normalized into the interval [0, 1] for similarity computation. The effort adaptation and analogous number then adopt the same setting as GRA. For both linear and nonlinear regressions, all features are transformed to a natural logarithmic scale in order to approximate a normal distribution. Then, a linear and nonlinear equation with a constant variable are used to construct the estimation model. We adopt a stepwise variable selection method to solve the regression's coefficients (Freedman et al. 1997; Hogg and Craig 1995). For the COCOMO model, because the three datasets we used

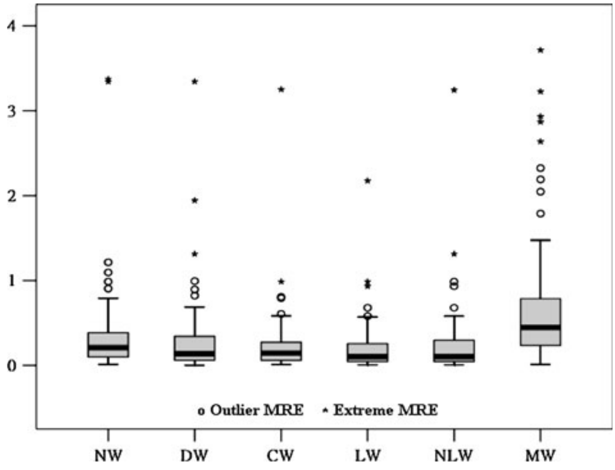


Fig. 16 Boxplot of MRE with Desharnais dataset (analogous number = 5)

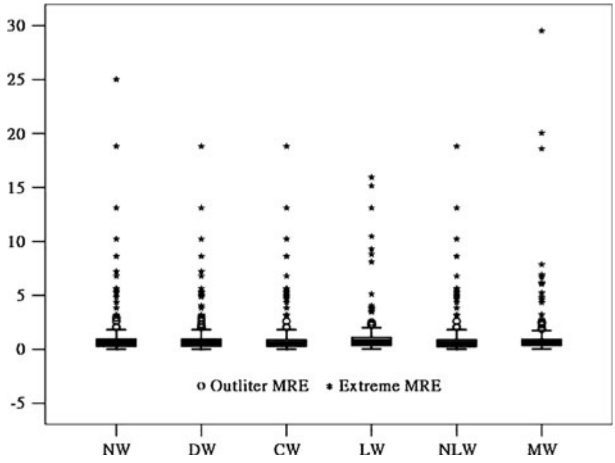


Fig. 17 Boxplot of MRE with ISBSG dataset (analogous number = 20)

in our experiments do not provide information for effort multipliers, it is hard to determine the value of effort multipliers manually. In order to compare the COCOMO model on the same basis, we decide to use basic COCOMO model without using effort multipliers (Korte and Port 2008; Port and Korte 2008). The estimated performance of these methods is presented in Tables 9 and 10, and the statistical test and confidence interval are summarized in Tables 11 and 12.

For the Kemerer dataset in Table 9, we can see that the MMRE criterion of LW GRA can be improved by 10.3% from NW GRA, 16.35% from analogy, 9.04% from the basic COCOMO model, 12.46% from LR, and 14.13% from NLR. In Table 10, the variance of LW GRA can be decreased by 40.01% from NW GRA, by 50.08% from analogy, by 7.06% from the basic COCOMO model, by 15.7% from LR, and by 28.42% from NLR, respectively. In Table 11, the LW GRA shows a significant difference from the analogy, LR, and NLR. Similarly, in Table 9 the PRED criterion of LW GRA can be increased by 66.5% from

Table 9 Accuracy with other methods

Method	Dataset criterion							
	Kemerer		COCOMO		Desharnais		ISBSG	
	MMRE	PRED	MMRE	PRED	MMRE	PRED	MMRE	PRED
NW GRA[a]	0.650	**0.200**	0.790	0.142	0.360	0.571	1.557	*0.240*
LW GRA[a]	*0.583*	*0.333*	*0.610*	*0.253*	*0.209*	*0.740*	*1.480*	0.208
Analogy[a]	**0.697**	**0.200**	**1.819**	0.174	0.369	0.558	1.834	0.216
COCOMO[b]	0.641	0.266	0.938	0.238	**0.630**	**0.402**	1.543	**0.118**
LR	0.666	*0.333*	1.583	**0.063**	0.411	0.675	**2.090**	0.168
NLR	0.679	*0.333*	0.932	0.206	0.420	0.701	1.853	0.176

The italic number indicates the most accurate within one dataset

The bold number indicates the least accurate within one dataset

[a] Analogous number = 5

[b] Basic COCOMO model without using effort multipliers

Table 10 Variance with other methods

Method	Dataset criterion			
	Kemerer Variance	COCOMO Variance	Desharnais Variance	ISBSG Variance
NW GRA[a]	1.557	1.948	0.311	12.439
LW GRA[a]	*0.934*	*0.663*	*0.093*	*8.615*
Analogy[a]	**1.871**	**2.587**	0.271	18.236
COCOMO[b]	1.005	1.751	**0.985**	14.507
LR	1.108	2.279	0.594	**20.106**
NLR	1.305	1.683	0.637	18.213

The italic number indicates the smallest variance for an analogous number

The bold number indicates the biggest variance for an analogous number

[a] Analogous number = 5

[b] Basic COCOMO model without using effort multipliers

NW GRA, 66.5% from analogy, and 25.18% from the basic COCOMO model, but is the same as those of LR and NLR. From Table 12, the confidence interval of LW GRA overlaps the upper limit of LR and NLR methods, indicating an insignificant improvement in the PRED criterion. This may be related to setting the PRED threshold at 25% and the sample size of the Kemerer dataset (Korte and Port 2008; Port and Korte 2008).

Similarly, for the COCOMO and Desharnais datasets in Tables 9 and 10, LW GRA obtains the most accurate criteria of all comparison methods. Moreover, we can observe the maximum improvement percentage of LW in these two datasets–the MMRE criterion can be separately improved by 66.46 and 66.82% based on analogy and the basic COCOMO model; the PRED criterion can be separately improved by 301.58 and 84.07% based on LR and the basic COCOMO model; and then the variance can be separately improved by 74.37 and 90.55% based on analogy and the basic COCOMO model. Also, from Table 11 the *p*-value of LW GRA is statistically different when compared with analogy, the basic COCOMO, LR, and NLR. From Table 12 the confidence interval of LW

Table 11 Wilcoxon signed rank sum test between other methods

(2-Tailed, $\alpha = 5\%$)	Kemerer		COCOMO		Desharnais		ISBSG	
	z Statistic	p Value	z Statistic	p Value	z Statistic	p Value	z Statistic	p Value
NW[a] vs. analogy[a]	−0.057	0.954	−2.281	0.022*	0.383	0.701	−3.054	0.002*
NW[a] vs. COCOMO[b]	1.396	0.162	−2.252	0.024*	−3.755	0.000*	−1.901	0.057
NW[a] vs. LR	−1.022	0.306	−6.353	0.000*	1.460	0.144	−2.507	0.012*
NW[a] vs. NLR	−0.170	0.864	−2.322	0.020*	1.363	0.172	−3.426	0.001*
LW[a] vs. analogy[a]	−2.101	0.035*	−2.290	0.022*	−3.008	0.002*	−2.364	0.018*
LW[a] vs. COCOMO[b]	−1.931	0.053	−2.421	0.015*	−6.294	0.000*	−2.391	0.016*
LW[a] vs. LR	−1.995	0.046*	−6.339	0.000*	−2.236	0.025*	−2.248	0.024*
LW[a] vs. NLR	−2.556	0.010*	−2.649	0.008*	−2.419	0.015*	−3.015	0.002*

* Denotes a significant difference existed between two models at significance level

[a] Analogous number = 5

[b] Basic COCOMO model without using effort multipliers

Table 12 Confidence interval of PRED with other methods

(95% Confidence level)	Kemerer		COCOMO		Desharnais		ISBSG	
	LCL	UCL	LCL	UCL	LCL	UCL	LCL	UCL
NW GRA[a]	0.179	0.220	0.127	0.157	0.555	0.587	0.227	0.252
LW GRA[a]	0.318	0.348	0.234	0.273	0.724	0.755	0.196	0.219
Analogy[a]	0.162	0.237	0.154	0.194	0.543	0.573	0.202	0.229
COCOMO[b]	0.220	0.313	0.218	0.257	0.385	0.420	0.110	0.126
LR	0.304	0.362	0.060	0.066	0.660	0.690	0.156	0.179
NLR	0.314	0.351	0.187	0.225	0.685	0.717	0.164	0.187

LCL lower confidence limit, *UCL* upper confidence limit

[a] Analogous number = 5

[b] Basic COCOMO model without using effort multipliers

GRA only covers that of the basic COCOMO model in COCOMO dataset. This reveals that LW GRA can generally improve the prediction accuracy and variance in both COCOMO and Desharnais datasets.

Finally, for the ISBSG dataset in Tables 9 and 10, the NW and LW GRAs appear to be the most accurate with the MMRE and PRED criteria, and the LW GRA can greatly reduce the variance of estimates from the NW GRA. That is, the MMRE criterion of LW GRA can be improved by 29.18% from LR, the PRED criterion of NW GRA can be increased by 103.38% from the basic COCOMO model, and then the variance of LW GRA can be decreased by 30.74% from NW GRA, respectively. Also, in Table 11 the *p*-value result shows that NW and LW GRAs are enough evidence to perform better than most of the other methods. Furthermore, in Table 12 we find that the confidence interval of NW and LW GRAs only overlaps analogy, indicating that NW and LW can significantly improve the performance of most of the other methods in the ISBSG dataset.

MRE boxplots for the four datasets are displayed in Figs. 18, 19, 20 and 21. It is noted that in Figs. 19 and 21 the scale over 30 is truncated in order to enlarge the axis interval. Consequently, seven extreme MREs of the LR are omitted from Fig. 19. One extreme

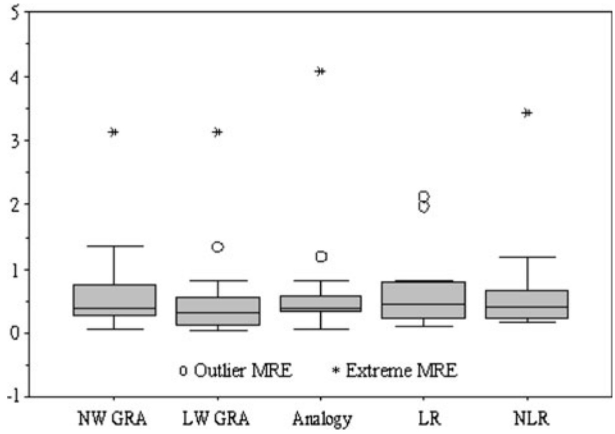
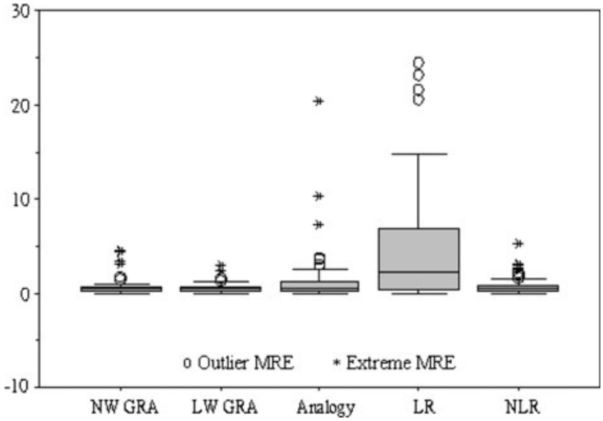


Fig. 18 Boxplot of MRE with Kemerer dataset

Fig. 19 Boxplot of MRE with COCOMO dataset (seven extreme MREs over 30 are omitted from LR)



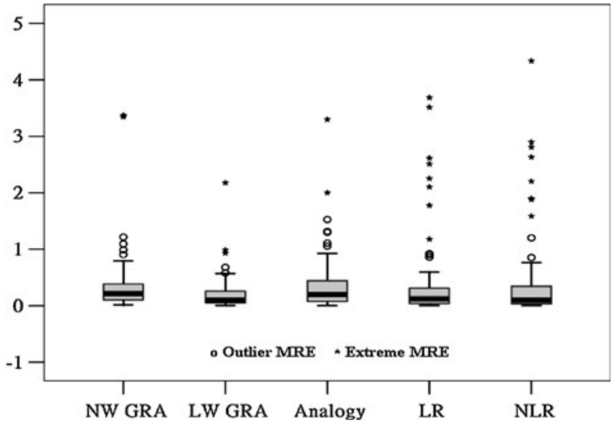


Fig. 20 Boxplot of MRE with Desharnais dataset

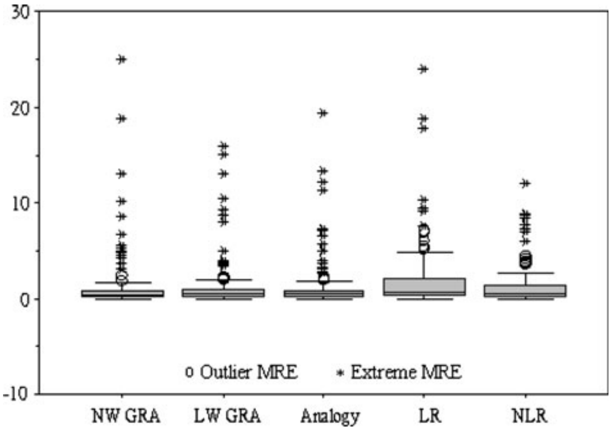


Fig. 21 Boxplot of MRE with ISBSG dataset (one and seven extreme MREs over 30 are separately excluded from analogy and NLR)

MRE of the analogy and three extreme MREs of the NLR are individually excluded from Fig. 21.

For the Kemerer dataset in Fig. 18, the NW, LW, and NLR obtain a similar accuracy in prediction. However, the LW GRA has the smallest quartiles and interquartile range, indicating most of the predictions of LW GRA are more accurate than those of NW, analogy, and NLR. For the COCOMO dataset in Fig. 19, we observe that the LW GRA has fewer extreme values and a smaller upper quartile than the other methods. Further, the LW GRA appears to greatly reduce the variability of MREs when compared with LR in the COCOMO dataset. As for the Desharnais dataset in Fig. 20, the LW GRA has the smallest MRE distribution, indicating the LW GRA can significantly improve the performance of effort prediction. Finally, for the ISBSG dataset in Fig. 21, the LW GRA provides much better accuracy than the NW GRA and LR. By contrast, the analogy and NLR methods are at variance in the MRE distribution since they contain some extreme MREs outside the interval scale. In summary, Figs. 18, 19, 20, and 21 are generally consistent with the results of Tables 9 and 10. When compared with NW GRA and the other methods, the MRE distribution of LW GRA is probably steady and contains few extreme estimates in four datasets.

4.2.3 Comparison of accuracy with published results

Other published results using the same dataset sources can also be compared with NW and LW GRAs. The methods collected here include GRACE, NN, CART, regression, analogy, and the COCOMO model and are quite diversified and commonly used in software effort estimation. Notice that GRACE is one of the GRA-based methods (Song et al. 2005); the NN includes albus perceptron (Samson et al. 1997) and back-propagation neural network (Srinivasan and Fisher 1995); the regression includes OLS regression (Huang and Chiu 2006) and stepwise regression (Mendes et al. 2005); the analogy includes CBR (Kadoda et al. 2000), traditional analogy (Shepperd and Schofield 1997) and weighted analogy (Auer et al. 2006; Huang and Chiu 2006); the COCOMO model includes OLS calibrated basic COCOMO models with or without effort multipliers (Korte and Port 2008; Port and Korte 2008). The comparison of accuracy among published results is shown in Table 13.

Table 13 Comparison of accuracy between published results

Method	Dataset								Remarks
	Kemerer		COCOMO		Desharnais		ISBSG		
	MMRE	PRED	MMRE	PRED	MMRE	PRED	MMRE	PRED	
NW GRA	0.571[1]	0.200[1]	0.755[1]	0.158[1]	0.323[1]	0.610[1]	1.347[3]	0.216[3]	[1] Analogous number = 3;
	0.650[2]	0.200[2]	0.790[2]	0.142[2]	0.360[2]	0.571[2]	1.252[4]	0.200[4]	[2] Analogous number = 5;
LW GRA	*0.566*[1]	0.333[1]	0.582[1]	0.285[1]	*0.167*[1]	*0.831*[1]	1.328[3]	*0.232*[3]	[3] analogous number = 10;
	0.583[2]	0.333[2]	0.610[2]	0.253[2]	0.209[2]	0.740[2]	1.214[4]	0.224[4]	[4] analogous number = 20.
GRACE	0.588	0.266	0.760	0.206	0.498	**0.300**	N.A.	N.A.	(Song et al. 2005)
NN	0.7	0.2	4.28	N.A.	N.A.	N.A.	1.7	0.12	(Huang and Chiu, 2006; Samson et al. 1997; Srinivasan and Fisher 1995)
CART	**3.64**	0.2	1.25	N.A.	N.A.	N.A.	1.89	0.21	(Huang and Chiu 2006; Jeffery et al. 2001; Shepperd and Schofield 1997; Song et al. 2005; Srinivasan and Fisher 1995)
							1.563	**0.08**	
Regression	1.07	**0.13**	5.20	N.A.	0.496	0.32	1.23	0.206	(Huang and Chiu 2006; Mendes et al. 2005; Samson et al. 1997; Shepperd and Schofield 1997; Song et al. 2005)
	1.02	0.33	**15.4**	**0.06**	0.66	0.42	1.9	0.12	
Analogy	0.62	*0.4*	N.A.	N.A.	0.64	0.36	*1.2*	0.09	(Auer et al. 2006; Huang and Chiu 2006; Jeffery et al. 2000; Kadoda et al. 2000; Shepperd and Schofield 1997; Song et al. 2005)
	0.676	0.267			0.476	N.A.	**2.39**	0.16	
COCOMO	N.A.	N.A.	*0.116*[5]	*0.984*[5,7]	0.576[6]	0.419[6,7]	N.A.	N.A.	[5] With effort multipliers;
			0.778[6]	0.269[6,7]	0.653[6]	0.506[6,7]			[6] Without effort multipliers;
			0.372[5]	0.428[5,7]	0.599[6,8]	0.456[6,7,8]			[7] PRED at level $l = 0.3$;
			0.997[6]	0.222[6,7]	**0.697**[6,8]	0.481[6,7,8]			[8] Software size is unadjusted FP. (Korte and Port 2008; Port and Korte 2008)
			0.325[5]	0.587[5,7]					

The italic number indicates the most accurate within one dataset

The bold number indicates the least accurate within one dataset

In the Kemerer dataset, NW and LW GRAs with analogous number "3" are more accurate than other published results in terms of MMRE criterion. Specifically, the MMRE criterion of LW GRA is the most accurate in the Kemerer dataset, and the PRED criterion of LW GRA is close to that of analogy. In the COCOMO dataset, NW GRA outperforms NN, CART, and regression in MMRE criterion. Further, LW GRA is significantly better than NW GRA and GRACE in terms of MMRE and PRED criteria. However, it is noted that the COCOMO model with effort multipliers is much more accurate than NW and LW GRAs in the COCOMO dataset. This may be due in part to the arbitrary effect of assigning effort multipliers. In fact, Boehm (1981) and Boehm et al. (1995) reported that the effort multipliers are to some extent plausible determinants for software development effort and suggested that the effort multipliers should be deliberately considered (i.e., the product of effort multipliers may range from 0.09 to 72.38). By contrast, if we only compare the COCOMO model without effort multipliers, LW GRAs still have better accuracy in the COCOMO dataset.

For the Desharnais dataset, NW and LW GRAs are generally better than other published results in MMRE and PRED criteria. In the ISBSG dataset, although the MMRE criterion of NW GRA and LW GRA are slightly worse than that of regression and analogy, our estimated results are still close to most of the other published results, and even outperform the NN and CART. In summary, by using the same data sources, the proposed methods can present acceptable accuracy compared to other studies. Particularly, LW GRA can further enhance the prediction performance of NW GRA. Therefore, we think that the weighted GRAs may be an alternative method in the field of software effort estimation.

4.3 Discussions

There are some factors that may affect the validity of our experiment, including adopted datasets, experimental process, and comparative criteria. First, the quality of datasets is an important factor for constructing prediction models. In this study, four publicly available datasets are adopted. The COCOMO and Desharnais datasets belong to a well-known PROMISE repository (Korte and Port 2008; Port and Korte 2008), and the ISBSG dataset is maintained by an international software benchmarking standards group (ISBSG 2006). These datasets consist of various application types, and the sample size varies from small to large. In addition, the data preprocessing and feature selection are fully explained in this paper. Hence, we believe these datasets are representative and reliable in quality.

This study focuses on weighted project features to determine a similarity measure. In fact, many studies have also noticed that each project feature has a different degree of influence on software development effort and considered weighted analogy methods (Huang and Chiu 2006; Li and Ruhe 2006; Mendes et al. 2002b). Thus, the originality of our proposed models is aligned with these studies. Six weighted approaches are integrated into GRA for software effort estimation, all of which are based on formal statistical methods to derive the corresponding weights for each feature. In order to assess the performance of weighted models, the leave-one-out cross-validation is then carefully implemented to evaluate estimated results. This validation is commonly used in the studies (Huang and Chiu 2006; Shepperd and Schofield 1997; Song et al. 2005). Therefore, with the above techniques, this experiment can be replicated for further improvement and comparison.

Because different criteria may reflect different attributes of model performance, it is better to compare more than one criterion in terms of reducing the risk of only trusting one criterion. In this paper, MMRE, PRED, variance, and boxplots of MRE are alternately used to demonstrate the estimated performance of the proposed models between other prediction techniques and published results. For these criteria, MMRE and PRED are used to measure the estimation model's accuracy, whereas variance and boxplot are used to show the reliability of estimates. Generally, the experiments obtain consistent results in these criteria. Furthermore, the statistical test and confidence interval are conducted to verify the difference among methods. We are then able to confirm that the assessment of experimental results is trustworthy and not due to any individual experiment or dataset.

According to the framework of CMMI maturity level 2 (CMMI Product Team 2002), projects of the organization have to be executed such that software development processes are planned, performed, measured, and controlled. Generally, software development effort can be viewed as a basic measure for software development cost and software quality assurance (Ejiogu 2005; Jeffery et al. 2001). After estimating software development effort, some management metrics related to software cost and quality can be derived. In Tables 14 and 15, an estimation example is presented. For demonstration purposes, here we select four software projects from the datasets and let $APC = \$1000$, $Schedule = 24$ months, and $Fault = 100$. As a result, for Project A we can obtain *software development cost* = \$74,280, *full-time software person* = 3.1 FSP, and *average cost per size* = \$1857/KLOC. This information can help to analyse cost expenditure, development schedule, and personnel distribution of software projects. Additionally, for Project B and C the *software productivities* are 0.22 KDSI/MM and 0.24 FP/MM, respectively. The productivity can provide a baseline for performance evaluation and control of software projects. If the productivity of a project team is far below a defined baseline, the project manager should take some improvement activities such as software reuse, outsourcing, adjusting the staff skill mix, or introduction of automatic development tools. Similarly, for Project D *debugging effort* = 13.58 MM/Fault, *debugging cost* = \$13580/Fault, and *fault density* = 0.38 Fault/FP. These three metrics are commonly used to evaluate software quality. A benchmarking figure provided by Fenton (1998) reported that a fault density of below 2 Fault/KLOC or 1.75 Fault/FP is considered to be good quality. Hence, we can see that Project D may have better testing efficiency in software development. In the testing and debugging or maintenance phases (Leung 2002), software managers can track these metrics in determining the amount of debugging effort expenditure, debugging cost, and releasing time policy. If the fault density of a development project is accepted at a specific level, the software product can be released; otherwise, testing processes or code inspections should be restarted (Myers 2004). In practice, all of the above-mentioned metrics are very useful for software managers. As these metrics can provide analytic information for managers, the software development process can be improved.

Table 14 Some useful metrics

Software cost metrics	Formula	Software quality metrics	Formula
Software development cost	$\mathrm{Cost_{total}} = \mathrm{MM} \times \mathrm{APC}$	Debugging effort per fault	$\frac{\mathrm{MM_{test}}}{\mathrm{Fault}}$
Average cost per size	$\mathrm{Cost_{size}} = \frac{\mathrm{Cost_{total}}}{\mathrm{Size}}$	Debugging cost per fault	$\frac{\mathrm{Cost_{test}}}{\mathrm{Fault}}$
Software productivity	$\mathrm{Prod} = \frac{\mathrm{Size}}{\mathrm{MM}}$	Fault density	$\frac{\mathrm{Fault}}{\mathrm{Size}}$
Average staffing	$\mathrm{FSP} = \frac{\mathrm{MM}}{\mathrm{Schedule}}$		

MM estimated effort in man-months, MM_{test} estimated effort in testing or debugging phase, $Cost_{total}$ total development cost, $Cost_{test}$ estimated cost in testing or debugging phase, *FSP* full-time software person, *APC* average personnel cost per month, *Schedule* development schedule, *Fault* observed number of faults, *Size* software size (e.g., thousand lines of code, function point, etc.)

Table 15 Estimation Example*

Metrics	Project (dataset)			
	Project A (Kemerer)	Project B (COCOMO)	Project C (Desharnais)	Project D (ISBSG)
Software size	40 KLOC	75 KDSI	188 FP	261 FP
Total estimated effort	74.28 MM	331.11 MM	758.20 MM	3233.49 MM
Phase—plans and Design (16%)	11.88 MM	52.97 MM	121.31 MM	517.35 MM
Phase—programming (26%)	19.31 MM	86.08 MM	197.13 MM	840.70 MM
Phase—test and debug (42%)	31.19 MM	139.06 MM	318.44 MM	1358.06 MM
Phase—maintenance (16%)	11.88 MM	52.97 MM	121.31 MM	517.35 MM
Software development cost	$74,280	$331,110	$758,200	$3,233,490
Average cost per size	$1857/KLOC	$4414/KDSI	$4032/FP	$12388/FP
Full-time software person	3.1 FSP	13.7 FSP	31.5 FSP	134.7 FSP
Software productivity	0.53 KLOC/MM	0.22 KDSI/MM	0.24 FP/MM	0.08 FP/MM
Debugging effort per fault	0.31 MM/Fault	1.39 MM/Fault	3.18 MM/Fault	13.58 MM/Fault
Debugging cost per fault	$311/Fault	$1390/Fault	$3184/Fault	$13580/Fault
Fault density	2.50 Fault/KLOC	1.33 Fault/KDSI	0.53 Fault/FP	0.38 Fault/FP

* Software development effort is estimated by LW GRA and analogous number = 5

APC = $1000; Schedule = 24 months; Fault = 100

5 Conclusions

In this paper, six weighted methods including nonweighted, distance-based, correlative, linear, nonlinear, and maximal weights are proposed for integrating into the conventional GRA. By using four public datasets, the performance of weighted GRAs is validated by comparing them with other techniques and published results. In addition, we also adopt sensitivity analyses and statistical tests to demonstrate the improvement of our proposed methods. The experimental results have several encouraging findings. First, the weighted GRAs perform better than the nonweighted GRA. Particularly, the linearly weighted GRA can mainly improve accuracy and reliability of estimates. Second, increasing distinguishing coefficients and choosing smaller analogous numbers can further enhance the accuracy of prediction results, but the analogous numbers are much more influential than the distinguishing coefficients. Third, the performance of weighted GRAs is generally better or close to other estimation techniques and published results. From the viewpoint of software practitioners, because there is no universally applicable method in all cases, they may still need more than one method or simultaneously adopt a series of methods to make correct decisions. In summary, we can recommend that GRA is an alternative or applicable method for software effort estimation.

Our proposed methods also have some advantages that traditional similarity-based methods have. Often, the usability of similarity-based methods is acceptable to software practitioners. That is, the proposed methods would be easy to calibrate and implement in the early stage of software development life cycle. Further, the proposed methods can support decision making for adaptation effort and can flexibly accommodate different similarity measures. Although our proposed methods require a few extra computations for weight assignment, this part can be solved automatically. In future work, we plan to develop a GRA-based CASE tool with features of proposed weight assignments and software project database for software project management. By using the developed tool, we will be able to collect more real data and further analyse the beneficial result of the proposed methods in industry. Finally, when more industrial data are available, the cost-effectiveness analysis and improvement productivity will be discussed and investigated in the near future.

Acknowledgments The work described in this paper was supported by the National Science Council, Taiwan, under Grant NSC 97-2221-E-007-052-MY3 and NSC 98-2221-E-007-067, and also substantially supported by a grant from the Ministry of Economic Affairs of Taiwan (Project No. 98-EC-17-A-02-S2-0097).The authors would like to thank several anonymous reviewers for their critical reviews and in-depth comments that helped to improve this paper. Thanks are also given to Amber Tsai, Prof. Nan-Hsing Chiu, and Prof. Swe-Kai Chen of National Tsing Hua University for their comments to enhance the quality of the paper.

References

Agrawal, M., & Chari, K. (2007). Software effort, quality, and cycle time: A study of CMM level 5 projects. *IEEE Transactions on Software Engineering, 33*(3), 145–156.

Auer, M., Trendowicz, A., Graser, B., Haunschmid, E., & Biffl, S. (2006). Optimal project feature weights in analogy-based cost estimation: Improvement and limitations. *IEEE Transactions on Software Engineering, 32*(2), 83–92.

Benediktsson, O., Dalcher, D., Reed, K., & Woodman, M. (2003). COCOMO-based effort estimation for iterative and incremental software development. *Software Quality Journal, 11*(4), 265–281.

Boehm, B. (1981). *Software engineering economics*. Englewood Cliffs: Prentice Hall.

Boehm, B., Clark, B., Horowitz, E., & Westland, C. (1995). Cost models for future software life cycle processes: COCOMO 2.0. *Annals of Software Engineering, 1*(1), 57–94.

Briand, L. C., Langley, T., & Wieczorek, I. (2000). A replicated assessment and comparison of common software cost modeling techniques. In *Proceedings of the 22nd international conference on software engineering (ICSE 2000), Limerick, Ireland* (pp. 377–386).

Chen, Z., Menzies, T., Port, D., & Boehm, B. (2005). Finding the right data for software cost modeling. *IEEE Software, 22*(6), 38–46.

Chiu, N. H., & Huang, S. J. (2007). The adjusted analogy-based software effort estimation based on similarity distances. *Journal of Systems and Software, 80*(4), 628–640.

CMMI Product Team. (2002). *Capability maturity model integration, Version1.1*. CMMI–SW/SE/IPPD/SS, staged representation, CMU/SEI-2002-TR-011.

Conte, S. D., Dunsmore, H. E., & Shen, V. Y. (1986). *Software engineering metrics and models*. Benjamin: Cummings Publishing Company.

Cuadrado-Gallego, J., Fernández-Sanz, L., & Sicilia, M. Á. (2006). Enhancing input value selection in parametric software cost estimation models through second level cost drivers. *Software Quality Journal, 14*(4), 339–357.

Deng, J. L. (1982). Control problems of grey systems. *Systems and Control Letters, 1*(5), 288–294.

Deng, J. L. (1989). Introduction to grey system theory. *The Journal of Grey System, 1*(1), 1–24.

Deng, J. L. (2000). *Theory and approach of grey system*. Taipei: Taipei Cau Lih Inc. (in Chinese).

Desharnais, J. M. (1989). *Analyse statistique de la productivitie des projets informatique a partie de la technique des point des fonction*. Masters Thesis, University of Montreal, QC.

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

Ejiogu, L. O. (2005). *Software metrics: The discipline of software quality* (1st ed.). North Charleston: BookSurge.

Fenton, N. E., & Pfleeger, S. L. (1998). *Software metrics: A rigorous and practical approach* (2nd ed.). Boston, MA: PWS Publishing.

Foss, T., Stensrud, E., Kitchenham, B., & Myrtveit, I. (2003). A simulation study of the model evaluation criterion MMRE. *IEEE Transactions on Software Engineering, 29*(11), 985–995.

Freedman, D., Pisani, R., & Purves, R. (1997). *Statistics* (3rd ed.). New York: W. W. Norton.

Hogg, R. V., & Craig, A. T. (1995). *Introduction to mathematical statistics* (5th ed.). Englewood Cliffs: Prentice Hall.

Hsu, C. J., & Huang, C. Y. (2006). Comparison and assessment of improved grey relation analysis for software development effort estimation. In *Proceedings of the 3rd international conference on management of innovation and technology (ICMIT 2006), Singapore* (pp. 663–667).

Hsu, C. J., & Huang, C. Y. (2007). Improving effort estimation accuracy by weighted grey relational analysis during software development. In *Proceedings of the 14th Asia-Pacific software engineering conference (APSEC 2007), Nagoya, Japan* (pp. 534–541).

Huang, S. J., & Chiu, N. H. (2006). Optimization of analogy weights by genetic algorithm for software effort estimation. *Information and Software Technology, 48*(11), 1034–1045.

ISBSG, International Software Benchmark and Standards Group. (2006). *Data repository* 8, 2006, www.isbsg.org.

Jeffery, R., Ruhe, M., & Wieczorek, I. (2000). A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data. *Information and Software Technology, 42*(14), 1009–1016.

Jeffery, R., Ruhe, M., & Wieczorek, I. (2001). Using public domain metrics to estimate software development effort. In *Proceedings of the 7th international symposium on software metrics (METRICS 2001)*, London, UK (pp. 16–27).

Jørgensen, M., Indahl, U., & Sjøberg, D. (2003). Software effort estimation by analogy and “regression toward the mean”. *Journal of Systems and Software, 68*(3), 253–262.

Jørgensen, M., & Shepperd, M. (2007). A systematic review of software development cost estimation studies. *IEEE Transactions on Software Engineering, 33*(1), 33–53.

Kadoda, G., Cartwright, M., Chen, L., & Shepperd, M. J. (2000). *Experiences using case-based reasoning to predict software project effort*. Empirical Software Engineering Research Group at Bournemouth University, Technical Reports, TR00-01.

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

Keung, J. W., & Kitchenham, B. (2007). Optimising project feature weights for analogy-based software cost estimation using the mantel correlation. In *Proceedings of the 14th Asia-Pacific software engineering conference (APSEC 2007)*, Nagoya, Japan (pp. 222–229).

Korte, M., & Port, D. (2008). Confidence in software cost estimation results based on MMRE and PRED. In *Proceedings of the 4th international workshop on predictor models in software engineering (ICSE 2008)*, Leipzig, Germany (pp. 63–70).

Leung, H. K. N. (2002). Estimating maintenance effort by analogy. *Empirical Software Engineering, 7*(2), 157–175.

Li, J., & Ruhe, G. (2006). A comparative study of attribute weighting heuristics for effort estimation by analogy. In *Proceedings of the 5th international symposium on empirical software engineering (ISESE 2006)*, Rio de Janeiro, Brazil (pp. 66–74).

Li, J., Ruhe, G., Al-Emran, A., & Richter, M. M. (2007a). A flexible method for software effort estimation by analogy. *Empirical Software Engineering, 12*(1), 65–106.

Li, Y. F., Xie, M., & Goh, T. N. (2009a). A study of mutual information based feature selection for case based reasoning in software cost estimation. *Expert Systems with Applications, 36*(3, Part 2), 5921–5931.

Li, Y. F., Xie, M., & Goh, T. N. (2009b). A study of project selection and feature weighting for analogy based software cost estimation. *Journal of Systems and Software, 82*(2), 241–252.

Li, M, Yang, Y, Wang, Q, & He, M. (2007b). COGOMO—an extension of COCOMO II for China Government contract pricing. In *Proceedings of the 22th international annual forum on COCOMO and systems/software cost modeling (COSYSMO 2007)*. Los Angeles, CA: USC Campus.

Lima Júnior, O.d. S., Farias, P. P. M., & Belchior, A. D. (2003). Fuzzy modeling for function points analysis. *Software Quality Journal, 11*(2), 149–166.

Liu, S., & Lin, Y. (2006). *Grey information: Theory and practical applications* (1st ed.). Berlin: Springer.

Liu, Q., Qin, W. Z., Mintram, R., & Ross, M. (2008). Evaluation of preliminary data analysis framework in software cost estimation based on ISBSG R9 data. *Software Quality Journal, 16*(3), 411–458.

Marir, F., & Watson, I. (1994). Case-based reasoning: A categorized bibliography. *The Knowledge Engineering Review, 9*(4), 355–381.

Mendes, E., Lokan, C., Harrison, R., & Triggs, C. (2005). A replicated comparison of cross-company and within-company effort estimation models using the ISBSG database. In *Proceedings of the 11th international symposium on software metrics (METRICS 2005)*, Como, Italy (pp. 36–45).

Mendes, E., Mosley, N., & Counsell, S. (2002a). The application of case-based reasoning to early web project cost estimation. In *Proceedings of the 26th international computer software and applications conference (COMPSAC 2002)*, Oxford, England (pp. 393–398).

Mendes, E., Watson, I., Triggs, C., Mosley, N., & Counsell, S. (2002b). A comparison of development effort estimation techniques for web hypermedia application. In *Proceedings of the 8th international symposium on software metrics (METRICS 2002)*, Ottawa, Canada (pp. 131–140).

Myers, G. J. (2004). *The art of software testing* (2nd ed.). New York: Wiley.

Port, D., & Korte, M. (2008). Comparative studies of the model evaluation criterions MMRE and PRED in software cost estimation research. In *Proceedings of the 2nd ACM/IEEE international symposium on empirical software engineering and measurement (ESEM 2008)*, Kaiserslautern, Germany (pp. 51–60).

Samson, B., Ellison, D., & Dugard, P. (1997). Software cost estimation using an albus perceptron (CMAC). *Information and Software Technology, 39*(1), 55–60.

Shepperd, M., & Schofield, C. (1997). Estimating software project effort using analogies. *IEEE Transactions on Software Engineering, 23*(11), 736–743.

Song, Q., Shepperd, M., & Mair, C. (2005). Using grey relational analysis to predict software effort with small data sets. In *Proceedings of the 11th international symposium on software metrics (METRICS 2005)*, Como, Italy.

Srinivasan, K., & Fisher, D. (1995). Machine learning approaches to estimating software development effort. *IEEE Transactions on Software Engineering, 21*(2), 126–137.

Walkerden, F., & Jeffery, R. (1999). An empirical study of analogy-based software effort estimation. *Empirical Software Engineering, 4*(2), 135–158.

Wen, K. L., Changchien, S. K., Yeh, C. K., Wang, C. W., & Lin, H. S. (2006). *Apply Matlab in grey system theory*. Taipei: Taipei Chwa Inc. (in Chinese).

Author Biographies

Chao-Jung Hsu He received the M.S. degree in information management from National Taiwan University of Science and Technology in 2005. He is currently pursuing the Ph.D. degree in computer science at National Tsing Hua University, Hsinchu, Taiwan. His current interests include software effort estimation, software measurement, and software project management.

Chin-Yu Huang He is currently an Associate Professor in the Department of Computer Science at National Tsing Hua University, Hsinchu, Taiwan. He received the M.S. (1994), and the Ph.D. (2000) in Electrical Engineering from National Taiwan University, Taipei. He was with the Bank of Taiwan from 1994 to 1999, and was a senior software engineer at Taiwan Semiconductor Manufacturing Company (TSMC) from 1999 to 2000. Before joining NTHU in 2003, he was a division chief of the Central Bank of China, Taipei. He received the Ta-You Wu Memorial Award from the National Science Council of Taiwan in 2008. In 2009, he was ranked as the top 15 scholars in Systems and Software Engineering worldwide between 2002 and 2006 by the Journal of Systems and Software based on his research on software reliability, software testing, and software metrics. His research interests are software reliability engineering, software testing, software metrics, software testability, and fault tree analysis.